Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte
FZKA 6965

Eine neue Methodik zur Erhöhung der Leistungsfähigkeit Evolutionärer Algorithmen durch die Integration lokaler Suchverfahren

W. Jakob

Institut für Angewandte Informatik

März 2004

Forschungszentrum Karlsruhe

in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte

FZKA 6965

Eine neue Methodik zur Erhöhung der Leistungsfähigkeit Evolutionärer Algorithmen durch die Integration lokaler Suchverfahren

Wilfried Jakob

Institut für angewandte Informatik

von der Fakultät für Maschinenbau
der Universität Karlsruhe (TH) genehmigte Dissertation

Forschungszentrum Karlsruhe GmbH, Karlsruhe
2004

Impressum der Print-Ausgabe:

Als Manuskript gedruckt

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

Mitglied der Hermann von Helmholtz-Gemeinschaft
Deutscher Forschungszentren (HGF)

ISSN 0947-8620

Eine neue Methodik zur Erhöhung der Leistungsfähigkeit Evolutionärer Algorithmen durch die Integration lokaler Suchverfahren

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Maschinenbau der
Universität Karlsruhe

genehmigte

Dissertation

von

Dipl.-Inform. Wilfried Jakob
aus Ettlingen

Tag der mündlichen Prüfung:	17.12.2003
Hauptreferent:	Prof. Dr.-Ing. habil. G. Bretthauer
Korreferent:	Prof. Dr.-Ing. habil. J. Wernstedt

Kurzfassung

Evolutionäre Algorithmen bilden die grundsätzlichen Wirkmechanismen der belebten Natur in algorithmischer Form nach, um durch *Vererbung*, *Selektion* und *Überleben der Besten* Lösungen iterativ zu verbessern. Ihr Haupteinsatzgebiet ist die Bearbeitung komplexer Optimierungsprobleme, für die es keine mathematischen Lösungen oder geeignete Heuristiken gibt oder es zu aufwendig wäre, solche zu entwickeln. Beispielhaft seien Anordnungsprobleme, Designoptimierungsaufgaben, Schedulingprobleme oder Reihenfolgeoptimierungen genannt. Da die global suchenden Evolutionären Algorithmen in der Nähe des Optimums schlecht konvergieren, sind nahezu alle erfolgreichen praktischen Anwendungen sogenannte Hybride, bei denen der Evolutionäre Algorithmus durch ein in der Regel problemspezifisches lokales Suchverfahren unterstützt wird. Dadurch gelingt es zwar, die Konvergenzgeschwindigkeit zum Teil erheblich (meist um Faktoren) zu steigern, andererseits wird aus dem allgemein anwendbaren evolutionären Verfahren eine problemspezifische Lösung.

Ziel der vorliegenden Arbeit ist die Schaffung eines allgemein anwendbaren hybriden Verfahrens, das die Vorteile der beteiligten Algorithmenklassen, nämlich *Robustheit* und *globales Suchverhalten* einerseits und *Schnelligkeit* andererseits unter Wahrung der *allgemeinen Anwendbarkeit* und der *Konvergenzsicherheit* in sich vereint. Die neu entwickelte Methodik besteht aus zwei Punkten: Erstens der Verwendung allgemein anwendbarer statt problemspezifischer lokaler Suchalgorithmen und zweitens der Entwicklung eines konvergenzabhängigen Steuerungsverfahrens zur Aufteilung der Rechenzeit zwischen den beteiligten Algorithmen. Dazu wurde eine Metrik zur Bestimmung der genotypischen Varianz zweier Individuen und darauf aufbauend ein Verfahren zu Bestimmung der Nischenbildung in einer Population entwickelt. Dabei wurde großer Wert darauf gelegt, daß die neue Methode bei allen populationsbasierten Evolutionären Algorithmen angewandt werden kann.

Zur Überprüfung der beiden Ziele, *Beibehaltung der Konvergenzsicherheit* und *Erhöhung der Konvergenzgeschwindigkeit,* wurde eine Testimplementierung durchgeführt. Dazu wurde der Evolutionäre Algorithmus GLEAM, der Aspekte der Evolutionsstrategie und der reellcodierten Genetischen Algorithmen in sich vereint, und zwei erprobte ableitungsfreie lokale Suchverfahren, nämlich der Rosenbrock-Algorithmus und das Complex-Verfahren ausgewählt. Als Testfälle wurden fünf mathematische Benchmarkfunktionen und drei praktische Probleme (Designoptimierung, Ressourcenoptimierung mit Scheduling und kollisionsfreie Roboterbahnplanung) benutzt. Überprüft wurden die Hybridisierungsarten Voroptimierung, Nachoptimierung und die (verzögerte) direkte Integration der lokalen Algorithmen in die Nachkommensbildung des Evolutionären Verfahrens, was durch die alternative Anwendung beider lokaler Suchalgorithmen, Verfahrenskombinationen und -modifikationen zu insgesamt dreizehn Hybriden führte. Die Experimente basierten auf je 100 Läufen pro Hybrid und Parametrierung, wobei eine Einstellung nur dann als erfolgreich galt, wenn alle 100 Läufe das vorgegebene Qualitätsziel erreichten. Als Ergebnis kann zusammenfassend festgestellt werden, daß das Ziel der Geschwindigkeitssteigerung unter Wahrung der Konvergenzsicherheit erreicht wurde: Am eindrucksvollsten bei der Ressourcenplanung und bei der Funktion nach Fletcher und Powell, die um den Faktor 90 bzw. 100 weniger Evaluationen im Durchschnitt benötigten als GLEAM. Dabei erwies sich die (verzögerte) direkte Integration als die beste Hybridisierungsart. Da leider keine allgemeingültige Parametrierung angegeben werden kann und auch nicht klar ist, welches der beiden lokalen Verfahren im Allgemeinen die besseren Resultate liefert, schließt die Arbeit mit einer Empfehlung zur Vorgehensweise bei praktischen Anwendungen und einem neuen Konzept zur adaptiven direkten Integration.

A New Method for the Increased Performance of Evolutionary Algorithms by the Integration of Local Search Procedures

Abstract

Evolutionary Algorithms form a procedure upon the pattern of the principals of biological evolution for improving solutions iteratively by means of *heredity*, *selection* and *survival of the fittest*. Their main area of application are complex optimization problems, for which no mathematical solutions or suitable heuristics exist or are too costly to develop. Examples for these tasks are design optimization problems, scheduling, resource optimization or sequencing problems. As the global searching Evolutionary Algorithm shows a poor convergence close to the optimum, nearly all successful real world applications use so called hybrids, where Evolutionary Algorithms are supported by, in most cases, problem-specific local searchers. This results in a considerable acceleration (frequently in the magnitude of factors) but turns the general applicable Evolutionary Algorithm into a domain specific tool.

The goal of the work on hand is the creation of a generally applicable hybrid procedure, which combines the advantages of both classes of algorithms, i.e. the *robustness* and *globality of the search* on the one hand and the *speediness* on the other, while maintaining the *generality* and the *convergence reliability.* The new method consists of two elements: The usage of generally applicable local searchers instead of problem-specific ones and second, the development of a convergence-based control procedure for distributing the computational power between the basic algorithms involved. This procedure is based on new methods for calculating the genotypic difference between individuals and for determining established niches within a population. Great importance was attached to the applicability of the new method to all population based Evolutionary Algorithms.

To verify the two goals of the *preservation of the convergence reliability* and the *raise of the convergence velocity* a test implementation was performed. The Evolutionary Algorithm GLEAM combining aspects of the Evolution Strategy and the real-coded Genetic Algorithms was chosen together with two approved derivation-free local search procedures, namely the Rosenbrock algorithm and the COMPLEX procedure. Five mathematical benchmark functions and three real world problems (design optimization, resource optimization in conjunction with scheduling and collision-free robot path planning) served as test cases. Four basic kinds of hybridization were investigated, pre-optimization of the initial population, post-optimization of the GLEAM results and (delayed) direct integration of the local search into the offspring production of the Evolutionary Algorithm. Based on combinations and modifications of these kinds and the two alternatively used local searchers this adds up to 13 hybrids. The experiments were based on 100 runs per hybrid and parameterization and for a success all runs had to accomplish the given target qualities. Summarizing the results it can be stated that the goal of improving the velocity was achieved: The most impressive speed-up was yielded by the resource optimization task and Fletcher's function which needed about 90 and 100 times less evaluations on average than the average of best GLEAM run. The (delayed) direct integration turned out to be the best kind of hybridization. Unfortunately no common parameterization could be extracted from the experiments. For direct integration the Rosenbrock procedure worked always but the COMPLEX algorithm delivered better results in those cases were it worked at all. The text concludes with a recommendation for practical applications of the results and with a new concept for an adaptive direct integration to overcome the parameterization problems.

Inhaltsverzeichnis

Symbole und Abkürzungen

AK	Aktionskette (erstmals in 3.2.2.2)
all	Nachoptimierung *aller* Nachkommen bei der direkten Integration (Jobparametrierung, erstmals in Kap. 5)
best	Nachoptimierung nur des *besten* Nachkommens bei der direkten Integration (Jobparametrierung, erstmals in Kap. 5)
C	Complex-Algorithmus (erstmals in Kap. 5)
Ci	GLEAM mit Complex-initialisierter Startpopulation (erstmals in Kap. 5)
Com	Erfolgsanteil des Complex-Algorithmus bei Ci-Läufen (Spaltenüberschrift, erstmals in Anhang B)
$\Delta_{akt}(AK_1, AK_2)$	Unterschied der Aktionspräsenz zweier Aktionsketten AK_1 und AK_2 (erstmals in 3.2.2.2)
$\Delta_{ges}(AK_1, AK_2)$	Gesamtabstand zweier Aktionsketten AK_1 und AK_2 (erstmals in 3.2.2.2)
$\Delta_{par}(AK_1, AK_2)$	Parameterabstand zweier Aktionsketten AK_1 und AK_2 (erstmals in 3.2.2.2)
$\Delta_{pos}(AK_1, AK_2)$	Positionsabstand zweier Aktionsketten AK_1 und AK_2 (erstmals in 3.2.2.2)
ε	Obere Grenze für Δ_{ges}, damit zwei AKs einander ähnlich sind (erstmals in 3.2.2.2)
ε_{Pop}	Obere Grenze für $\max(\Delta_{ges})$ (erstmals in 3.2.2.3)
E(<job>)	Erfolgsrate eines Jobs (erstmals in 5.2)
EA	Evolutionärer Algorithmus (erstmals in 1.2.2)
EP	Evolutionäre Programmierung (erstmals in 1.2.3)
ES	Evolutionsstrategie (erstmals in 1.2.2)
G	GLEAM, General Learning Evolutionary Algorithm and Method (erstmals in Kap. 5)
G1,G3	Überprüfung der Nischenbildung bei GDV- und GAk-Werten von 1 bzw. 3 (Jobparametrierung, erstmals in Kap. 5)
GA	Genetischer Algorithmus (erstmals in 1.2.2)
GAk	Generationen ohne Akzeptanz im Deme (erstmals in 3.2.2.1)
GAk_{min}	Untergrenze für den Stagnationsindikator GAk (erstmals in 3.2.2.3)
GC	GLEAM mit direkter Integration des Complex-Verfahrens (erstmals in Kap. 5)
GDV	Generationen ohne Deme-Verbesserung (erstmals in 3.2.2.1)
GDV_{min}	Untergrenze für den Stagnationsindikator GDV (erstmals in 3.2.2.3)
Gen	Generationen (Spaltenüberschrift, erstmals in Anhang B)
GLEAM	General Learning Evolutionary Algorithm and Method (erstmals in 2.2.3)

GR	GLEAM mit direkter Integration des Rosenbrock-Verfahrens (erstmals in Kap. 5)
GutSchn	Durchschnitt bezogen auf die erfolgreichen (guten) Läufe eines Jobs (Spaltenüberschrift, erstmals in Anhang B)
GutVari	Varianz *s*, basierend auf den erfolgreichen (guten) Läufen eines Jobs (Spaltenüberschrift, erstmals in Anhang B)
GvC	GLEAM mit verzögerter direkter Integration des Complex (erstmals in Kap. 5)
GvR	GLEAM mit verzögerter direkter Integration des Rosenbrock-Verfahrens (erstmals in Kap. 5)
h	hohe Präzision; Abbruchschranke für das Rosenbrock-Verfahren (Jobparametrierung, erstmals in Kap. 5)
Indivs	Bewertete Individuen; Evaluationen (Spaltenüberschrift, erstmals in Anhang B)
l100, l5, l0 L100, L5, L0	Lamarckanteil 100%, 5% oder 0% (Baldwin-Evolution); im Anhang mit L wegen der besseren Lesbarkeit (Jobparametrierung, erstmals in Kap. 5)
LSV	Lokales Suchverfahren (erstmals in 3)
m	mittlere Präzision; Abbruchschranke für das Rosenbrock-Verfahren (Jobparametrierung, erstmals in Kap. 5)
max(Δ_{ges})	Maximum der Δ_{ges} aller Nischenrepräsentanten (erstmals in 3.2.2.3)
MinNo(<job>)	Kleinste erreichte Note eines Jobs (erstmals in 5.2)
n	niedrige Präzision; Abbruchschranke für das Rosenbrock-Verfahren (Jobparametrierung, erstmals in Kap. 5)
No(<job>)	Durchschnittsnote eines Jobs (erstmals in 5.2)
N_{Anz}	Anzahl der Nischen (Subpopulationen einander ähnlicher Individuen) in einer Population (erstmals in 3.2.2.2)
N_{max}	Obergrenze für die Anzahl der Nischen (erstmals in 3.2.2.3)
N<anz>	maximale Nischenanzahl *N*, sofern von Tabelle B.4 abweichend (Jobparametrierung, erstmals in Anhang B)
NC1C	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei alle GLEAM-Ergebnisse zur Bildung eines Startcomplexes (1C) dienen (erstmals in Kap. 5)
NC1P	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei GLEAM meist mehrfach einen Startpunkt (1P) liefert (erstmals in Kap. 5)
Ni.Schn	Durchschnitt der Nischenanzahl beim Evolutionsabbruch der Nachoptimierung (Spaltenüberschrift, erstmals in Anhang B)
Nisch. GDV/GAk	Diese Spalte enthält zwei Angaben: 1.Zahl: Anzahl der Läufe, bei denen nachoptimiert bzw. zugeschaltet wurde. Wenn diese Zahl geringer als die Anzahl der Läufe des Jobs ist, terminierte beim Rest die Evolution entweder wegen Erfolg oder Stagnation. 2.Zahl: Anzahl der Läufe bei denen die Nachoptimierung oder Umschaltung wegen Stagnation (Erreichen des GAk/GDV-Limits) erfolgte (Spaltenüberschrift, erstmals in Anhang B)

Note	Fitness bei GLEAM im Bereich von 0.0 - 100000.0 (erstmals in 2.2.3)
Nopt Erf	Nachoptimierungserfolg, Angabe wie *Erfolg* in Prozent (Spaltenüberschrift, erstmals in Anhang B)
Noten-Verb.	Notenverbesserung durch die Nachoptimierung (Spaltenüberschrift, erstmals in Anhang B)
NR	GLEAM plus Nachoptimierung mit dem Rosenbrock-Verfahren (erstmals in Kap. 5)
NV(<aufgabe>)	Notenverbesserung der Nachoptimierung gegenüber den Ergebnissen bei Abbruch der Evolution (erstmals in 5.2.2.2)
O(f(n))	O-Notation; asymptotische obere Schranke für die Aufwandsabschätzung von Algorithmen. Auch Zeitkomplexität oder Komplexität (erstmals in 1.2)
p<anz>	Populationsgröße <anz> (Jobparametrierung, erstmals in Kap. 5)
<x>%	bei Ri oder Ci: Anteil der mit dem LSV voroptimierten Individuen an der Startpopulation in % (Jobparametrierung, erstmals in Kap. 5)
P1, P2, P3	Parametersätze 1 bis 3 zur Steuerung der Nachoptimierung und der verzögerten direkten Integration, Werte siehe Tabelle B.3 (Jobparametrierung, erstmals in Kap. 5)
Pa, ..., Pg	Parametersätze a bis g zur Steuerung der Nachoptimierung, Werte siehe Tabelle 5.2 (Jobparametrierung, erstmals in 5.2.1.4)
P0, P4, ..., P7	Parametersätze 0 und 4 bis 7 zur Steuerung der Nachoptimierung und der verzögerten direkten Integration, Werte siehe Tabelle 5.6 (Jobparametrierung, erstmals in 5.2.1.7)
R	Rosenbrock-Verfahren (erstmals in Kap. 5)
Ri	GLEAM mit Rosenbrock-initialisierter Startpopulation (erstmals in Kap. 5)
Ros	Erfolgsanteil des Rosenbrock-Verfahrens bei Ri-Läufen (Spaltenüberschrift, erstmals in Anhang B)
RS	Restart der Läufe eines Jobs wegen Nichtkonvergenz des LSVs (Spaltenüberschrift, erstmals in Anhang B)
s	Sonderparametrierung für das Rosenbrock-Verfahren; die entsprechenden Werte sind beim jeweiligen Experiment aufgeführt (Jobparametrierung, erstmals in Kap. 5)
Schn	Durchschnitt (Noten-, Generationen- oder Individuendurchschnitt) (Spaltenüberschrift, erstmals in Anhang B)
TCP	Tool Center Point (erstmals in 4.4)
TSP	Traveling Salesman Problem, kombinatorische Benchmarkaufgabe (erstmals in 2.2.2)
u	sehr hohe Präzision; Abbruchschranke für das Rosenbrock-Verfahren (Jobparametrierung, erstmals in Kap. 5)
v	extrem hohe Präzision; Abbruchschranke für das Rosenbrock-Verfahren (Jobparametrierung, erstmals in Kap. 5)

1. Einleitung

1.1 Einordnung der Arbeit

Bei einer Vielzahl technisch-wissenschaftlicher und wirtschaftlicher Aufgabenstellungen spielt die Optimierung der Organisation von Abläufen oder von Systemen, sei es in Form von Anlagen oder Produkten, eine wichtige Rolle. Daraus sind eigene Wissenschaftsdisziplinen wie z.B. das Operations Research entstanden. Manche Aufgaben können mit Hilfe mathematischer Modelle exakt gelöst werden, andere wiederum sind auf Grund ihrer Komplexität (z.B. Nichtlinearitäten) einer solchen Lösung unzugänglich. Zu letzteren gehören z.B. Scheduling-Probleme, Anordnungsprobleme, Reihenfolgeoptimierungen oder die Designoptimierung technischer Produkte. In diesen Fällen wird in der Praxis häufig mit heuristischen Methoden vorgegangen oder es werden einfach manuell Lösungen, gestützt auf Erfahrungswissen und Intuition, gefunden.

Demgegenüber kann meist schon eine Verbesserung erreicht werden, indem ein Computermodell der relevanten Problemeigenschaften erstellt und mit Hilfe von Simulationsläufen nach besseren Lösungen gesucht wird. Im Falle einer manuellen Suche hängen die Erfolgsaussichten vom Ausbildungsstand, der Erfahrung und der Intuition der mit der Durchführung und Auswertung der Simulationsläufe betrauten Personen ab. Da es ab einem gewissen Komplexitätsgrad unmöglich ist, auch nur annähernd alle wichtig erscheinenden Bereiche des Suchraums zu betrachten, sind Einschränkungen auf die „besonders interessant erscheinenden“ Teile notwendig, wodurch bisweilen jedoch nur die Ansichten und Vorurteile der Experimentatoren fortgeschrieben werden. Ein wesentlicher Fortschritt kann dagegen durch den Einsatz von Suchalgorithmen erreicht werden, die den Suchraum methodischer und unvoreingenommener als die beschriebene manuelle Stichprobennahme durchsuchen.

Der mit einer konkreten Optimierungs- oder Planungsaufgabe betraute Anwender steht vor dem Problem der Auswahl eines geeigneten Lösungsverfahrens. Dabei werden folgende Anforderungen eine wichtige Rolle spielen, siehe auch [Dav91, Gol99]:

1. Robustheit
   Da meist nicht die Zeit oder die Möglichkeit für eine umfassende mathematische Problemanalyse besteht, muss das Verfahren hinsichtlich schlechter Startwerte und des Konvergenzverhaltens robust sein. Außerdem muss es auch noch beim Auftreten unangenehmer Eigenschaften, wie z.B. Optima auf den Gültigkeitsgrenzen der Parameter oder Unstetigkeitsstellen funktionieren.

2. Arbeitsaufwand
   Da derartige Projekte meist unter Zeitdruck durchgeführt werden, muss der Arbeitsaufwand zur Vorbereitung und zu eventuellen Implementierungen gering sein. Das gilt vor allem dann, wenn immer wieder Variationen der gleichen Aufgabenklasse gelöst werden müssen, wie dies z.B. bei Planungs- oder Scheduling-Aufgaben, aber auch bei der Optimierung von Designvarianten der Fall ist. Um die Erstellung eines geeigneten Simulationsmodells oder eines anderen Verfahrens zur Berechnung der Qualität eines Lösungs-

vorschlags wird man zwar nicht herumkommen, aber eine fallweise Anpassung des Optimierungsverfahrens ist wegen des damit verbundenen Aufwands nicht tragbar.

3. Rechtzeitigkeit
   Je nach Aufgabenstellung wird der Zeitrahmen zur Lösungsfindung variieren. Die Arbeit an Designaufgaben kann sich über Tage und Wochen erstrecken, während bei Scheduling-Problemen oft nur Minuten zur Verfügung stehen. In jedem Fall aber muss die Lösung dann bereit stehen, wenn sie gebraucht wird: Der beste Plan nützt nichts, wenn er erst nach Arbeitsende fertig ist.

4. Verwertbarkeit
   Meist reicht eine Verbesserung der bisherigen Praxis oder eine Lösung in der Nähe des Optimums als befriedigendes Ergebnis aus. So wünschenswert die Ermittlung des exakten Optimums auch sein mag, häufig kann es angesichts unvermeidlicher Toleranzen nicht ebenso exakt umgesetzt werden und der möglicherweise erhöhte Aufwand war umsonst.

Der Anwender hat nun die Auswahl zwischen

- globalen Suchverfahren
  Globale Suchverfahren sind robust, allgemein anwendbar und finden in der Regel auch das globale Optimum oder gelangen zumindest in dessen Nähe. Eine große Parameteranzahl stellt für sie ebenso wenig ein Hindernis dar wie kombinatorische Aspekte der Aufgabenstellung. Damit sind die ersten beiden Forderungen erfüllt. Der Nachteil liegt in der vergleichsweise großen Anzahl an Berechnungen der Zielfunktion. Da sie meist relativ schnell gültige, wenn auch suboptimale Lösungen liefern, kann in der Regel auch die Forderung nach verwertbaren Ergebnissen befriedigt werden. Die Forderung nach rechtzeitigen Lösungen kann hingegen nur anwendungsabhängig beantwortet werden. Die zur Bewertung einer Lösung notwendige Rechenzeit ist ebenso aufgabenabhängig wie die zur Verfügung stehende Zeit zur Lösungsfindung. Da jedoch die Rechnerhardware immer leistungsfähiger und billiger wird, erschließen sich den globalen Verfahren immer mehr Aufgabenbereiche. Dieser Trend gilt besonders für solche Verfahren, die sich effizient, d.h. ohne wesentlichen zusätzlichen Kommunikations- und Koordinierungsaufwand, parallelisieren lassen.

- traditionellen mathematischen Verfahren (lokale Verfahren)
  Wenn es sich um reine Parameteroptimierungen handelt und kombinatorische Aspekte keine Rolle spielen, können bei hinreichend geringer Parameterzahl numerische Optimierungsverfahren, wie Gradienten- oder Simplexverfahren eingesetzt werden. Da sie recht schnell konvergieren, kann damit unter Berücksichtigung der Forderungen 3 und 4 nach verwertbaren Ergebnissen zum geforderten Zeitpunkt je nach Aufgabenstellung ein befriedigendes Ergebnis erzielt werden. Andererseits sind sie je nach Verfahren mehr oder weniger startpunktabhängig, was einer robusten Anwendbarkeit im Wege steht.

- spezialisierten Verfahren
  Für bestimmte eingeschränkte Aufgabenstellungen gibt es exakte oder Näherungsverfahren, die speziell auf diese Aufgabenklasse zugeschnitten wurden. Ihr Einsatz empfiehlt sich, wenn sie angesichts der konkreten Randbedingungen robust genug sind. Die Neuentwicklung solcher Verfahren scheitert jedoch meist an der zur Verfügung stehenden Zeit oder an der generellen Machbarkeit.

- heuristischen Verfahren
  Heuristiken spiegeln Erfahrungswissen wieder und können damit nur bei Problemen angewandt werden, wo solches Wissen auch vorliegt. Sie bringen in der Regel nur eine Verbesserung und keine optimale Lösung und haben als Spezialverfahren auch deren bereits erwähnte Vor- und Nachteile.

Wie die Aufstellung zeigt, sollte zuerst geprüft werden, ob eine auf das konkrete Problem zugeschnittene Speziallösung oder ein anwendbares heuristisches Verfahren existiert, da beide in der Regel vergleichsweise schnell zum Ziel führen. Wenn das nicht der Fall ist und die Aufgabenstellung, sei es auf Grund der Parameterzahl, der Struktur des Suchraums oder wegen kombinatorischer Aspekte, nicht für die traditionellen mathematischen Verfahren in Frage kommt, muß auf globale Verfahren zurückgegriffen werden. Die vorliegende Arbeit geht vom allgemeinen Anwendungsfall aus und konzentriert sich dementsprechend auch auf allgemein anwendbare Verfahren. Damit werden heuristische Ansätze oder Speziallösungen nicht weiter betrachtet. Sie hat zum Ziel, die Leistungsfähigkeit einer der mächtigsten Klasse globaler Suchverfahren, der Evolutionären Algorithmen, durch eine geeignete Kombination mit lokalen Optimierungsverfahren zu verbessern.

1.2 Darstellung des Entwicklungsstands von Such- und Optimierungsverfahren

Die Struktur und Komplexität des Suchraums ist von entscheidender Bedeutung für die Erfolgsaussichten und den notwendigen Aufwand einer Suche. Suchräume können unimodal mit nur einem Optimum oder multimodal mit mehreren oder einer Vielzahl von Suboptima sein, Unstetigkeitsstellen oder verbotene Bereiche auf Grund von Restriktionen enthalten. Erschwerend kommt meist hinzu, daß zu Beginn einer Optimierung bis auf die Anzahl der Variablen nichts oder nur wenig über den Suchraum bekannt ist. Im folgenden wird, sofern nicht anders vermerkt, der allgemeine Fall des unbekannten, mehrdimensionalen und multimodalen Suchraums unter Berücksichtigung von Nebenbedingungen angenommen. Manche Verfahren werden in der Literatur zum Auffinden eines Minimums, andere zur Bestimmung eines Maximums vorgestellt. Da es wegen $max\{F(x)\} = -min\{-F(x)\}$ immer möglich ist, die Suche nach einem Minimum als Suche nach dem Maximum umzuformulieren, wird im folgenden der sprachlichen Einfachheit halber von der Suche nach einem Maximum als der Suche nach dem Optimum ausgegangen.

Ein Parameteroptimierungsproblem kann nach [Bäc91, Hof92] formal folgendermaßen formuliert werden:
Ausgehend von der Funktion F

$$F{:}M \subseteq M_1 \times ... \times M_n \rightarrow R^1, \; M \neq \varnothing$$

wobei R^1 die Menge der reellen Zahlen ist, wird ein Vektor $x_{opt} \in M$ gesucht, für den gilt:

$$\forall x \in M \; : \; (F(x) \leq F(x_{opt})) \; = \; F_{opt} \; .$$

F_{opt} wird als *globales Optimum* bezeichnet. Dagegen handelt es sich um ein *lokales Optimum* $F_{lopt} = F(x_{lopt})$, wenn gilt:

$$\exists \varepsilon > 0 \; \forall x \in M \; : \; \|x - x_{lopt}\| < \varepsilon \Rightarrow F_{lopt} \geq F(x)$$

Bei den meisten Parameteroptimierungsaufgaben sind die M_i die Menge der reellen Zahlen. Wenn dagegen alle Vektorkomponenten ganzzahlig sind, wird von *ganzzahliger Optimierung* gesprochen, und von *gemischt ganzzahliger Optimierung*, wenn ein Teil reell und ein anderer ganzzahlig ist. *Beschränkungen* von M können formuliert werden als

$$G_j(x) \geq 0 \qquad j = 1, \ldots, m \tag{1.1}$$

Dabei wird zwischen *expliziten Beschränkungen*, die nur von einer Komponente des Vektors x abhängen und *impliziten*, bei denen mehrere Vektorkomponenten beteiligt sind, unterschieden.

Entsprechend dem Suchverhalten lassen sich die Algorithmen in *globale* und *lokale Suchverfahren* einteilen. Außerdem wird zwischen *stochastischen* und *deterministischen* Algorithmen unterschieden. Ein weiteres Differenzierungsmerkmal ist die Frage, ob anwendungsspezifisches Wissen in die Verfahren eingebracht wurde, wie bei den spezialisierten und den heuristischen Verfahren oder nicht.

1.2.1 Lokale Suchverfahren

Die lokalen Suchverfahren wurden dafür entwickelt, lokale Optima in der relativen Nähe eines Startpunktes effektiv und sicher zu finden. Sie sind bis auf die am Schluß dieses Abschnitts behandelten deterministischer Natur. Ihr Vorteil liegt in der vergleichsweise geringen Anzahl an Zielfunktionswertberechnungen, die zur Lösungsfindung benötigt werden. Sie konvergieren also relativ schnell. Um Aussagen über einen größeren Bereich oder gar den gesamten Suchraum selbst machen zu können, müssen lokale Verfahren mehrfach mit unterschiedlichen Startwerten ausgeführt werden. Unterscheiden sich die dabei erhaltenen Ergebnisse, kann auf ein multimodales Problem geschlossen werden. Andernfalls kann hingegen nicht von Unimodalität oder gar vom Optimum ausgegangen werden, da es durchaus sein kann, daß die Startpunkte nur ungünstig gewählt waren. Vor allem bei größeren Suchräumen, über die wenig bekannt ist, relativiert diese Startpunktabhängigkeit die schnelle Konvergenz unter Umständen erheblich. Ein weiterer Nachteil lokaler Verfahren liegt darin, daß sie als reine Funktionsoptimierer nicht mehr anwendbar sind, sobald z.B. kombinatorische Aspekte zur Aufgabenstellung dazugehören.

In Abb. 1.1 ist das generelle Ablaufschema lokaler deterministischer Suchverfahren dargestellt. Ausgehend von einem vorgegebenen Startvektor bzw. dem zuletzt erzeugten Vektor werden Suchrichtung und Schrittlänge ermittelt, um daraus einen neuen Punkt im Suchraum zu ermitteln. Wenn dieser Punkt das Abbruchkriterium erfüllt, ist die Suche zu Ende, ansonsten wird die nächste Iteration begonnen. Vor allem die Ermittlung der Suchrichtung unterscheidet die verschiedenen Verfahren, wie noch nachfolgend ausgeführt wird. Die Schrittlänge ist entweder fest vorgegeben oder wird im Laufe der Suche an die Gegebenheiten des Suchraums angepaßt. Häufig benutzte Abbruchkriterien beruhen auf:

- der Schrittweite
  Bei angepaßter Schrittweite kann deren Sinken unter einen Schwellwert als Abbruchkriterium dienen, da das Verfahren offensichtlich keine nennenswerten Änderungen mehr produziert.
- dem Unterschied der Zielfunktionswerte
  Sinkt die Differenz der Funktionswerte aufeinanderfolgender Iterationen n-mal hintereinander unter ein vorgegebenes Limit, wird abgebrochen. Dieses Kriterium ist unab-

hängig vom Verfahren zur Ermittlung von Schrittweite und -richtung anwendbar, da es sich nur auf die Funktionswerte bezieht.

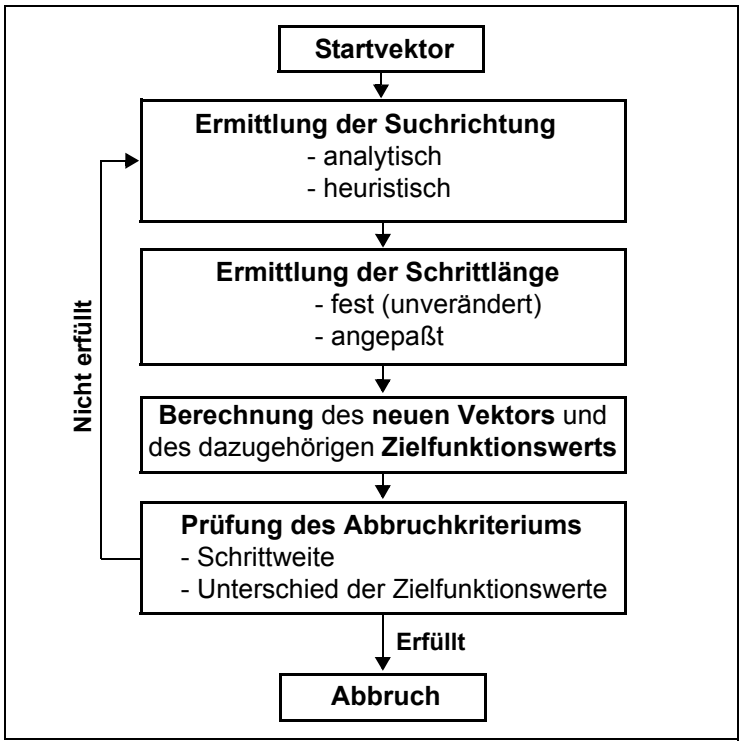


Abb. 1.1: Ablaufschema lokaler deterministischer Suchverfahren

Eine detaillierte Behandlung der verschiedenen lokalen Suchverfahren kann in [Schw95] gefunden werden. In der Literatur wird zwischen direkten und indirekten Strategien unterschieden. Die indirekten Verfahren bestimmen die Suchrichtung analytisch, wobei sie neben dem Funktionswert weitere Informationen über die Zielfunktion wie zum Beispiel partielle Ableitungen ausnutzen. So ermitteln das *Gradientenverfahren* [Fle87] oder das *konjugierte Gradientenverfahren* [Pow62, Fle64] die Richtung des stärksten Anstiegs, um die Suchrichtung festzulegen. Die Schrittweite wird so bestimmt, daß in Suchrichtung eine möglichst große Verbesserung erreicht wird. Beide Verfahren erfordern allerdings, daß die Zielfunktion nicht nur analytisch bekannt, sondern auch zumindest in hinreichender Umgebung des Optimums differenzierbar ist. Weniger strenge Voraussetzungen macht das *Quasi-Newton-Verfahren* [Fle63], das ein quadratisches Modell der Zielfunktion mittels Taylor-Reihenansatz bildet und dessen erste partielle Ableitung zur Bestimmung der Suchrichtung verwendet. Es gibt eine Fülle von Varianten, die es u.a. zu einem Verfahren mit konjugierten Richtungen macht (*DFP-Verfahren*) [Dav59, Fle63] oder Stabilitätsprobleme auf Grund von Rundungsfehlern [Tab69, Mur70, Bar68, Dix72] durch Rücksetzen der Hessematrix löst [Bar68, McC69]. Daneben gibt es noch die ableitungsfreie Version des DFP-Verfahrens von Stewart [Ste67], das den negativen Einfluß von Rundungsfehlern minimiert. Da die aufgezählten Verfahren Nebenbedingungen nicht berücksichtigen, müssen diese, soweit möglich, durch eine entsprechende Formulierung der Zielfunktion (z.B. durch die Hinzunahme von Straf- oder Barrierefunktionen) berücksichtigt werden.

Direkte Verfahren unterscheiden sich von den indirekten dadurch, daß Suchrichtung und Schrittweite heuristisch bestimmt werden. Damit entfällt die Voraussetzung der Differenzierbarkeit der Zielfunktion und es wird meist nur eine stückweise Stetigkeit verlangt. Eines der

einfachsten Verfahren ist der *Gauß-Seidel-Algorithmus*, bei dem iterativ die Achsen nacheinander mit fester Schrittweite durchsucht werden, siehe [Ort66, Van67]. Der so gewonnene beste Wert wird als Ausgangspunkt für eine erneute Suche in Richtung der Achsen genommen, bis keine Verbesserung mehr möglich ist. Es ist offensichtlich, daß dies in der Regel mehr Iterationen erfordert als eine ableitungsbasierte Suche. Das Verfahren konvergiert schlecht, wenn der Weg zum Optimum nicht parallel oder näherungsweise parallel zu den Hauptachsen liegt. Eine einfache Verbesserung besteht darin, einer Richtung nur solange zu folgen, wie keine Verschlechterung eintritt (Liniensuche). Damit wird allerdings auch der lokale Charakter des Verfahrens gestärkt, da nun bereits kleine Täler entlang einer Suchrichtung ausreichen können, um ein dahinter liegendes Optimum nicht zu finden. Auch die Schrittweite kann an den Suchraum angepaßt werden, um vor allem das Konvergenzverhalten in der Nähe des Optimums zu verbessern [Kow68]. Auch das kann bei einem entsprechenden Suchraum dazu führen, daß Täler nicht mehr übersprungen werden können.

Die *Pattern-Strategien* versuchen der Beschränkung auf die Suche in Richtung der Koordinatenachsen zu entgehen, indem sie die Suchrichtung den Gegebenheiten der Zielfunktion anpassen. Mit Hilfe von Erkundungsschritten wird die Richtung der nachfolgenden Suchschritte bestimmt. Das *Verfahren von Hooke und Jeeves* [Hoo61] bestimmt bei jeder Iteration mit Hilfe jeweils eines Schrittes entsprechend dem Gauß-Seidel-Verfahren die Suchrichtung durch Verbinden der Ausgangs- und Endpunkte der jeweils erfolgreichen Schritte. Entlang dieser Richtung erfolgt dann eine Liniensuche, bis eine Verschlechterung eintritt und die nächste Iteration beginnt. Vorteile des Verfahrens sind eine beschleunigte Suche und ein vergleichsweise einfacher Algorithmus. Nachteilig ist die Beschränkung der Suchschritte auf die Richtung der Koordinatenachsen, was zu vorzeitiger Terminierung des Algorithmus führen kann [Schw95]. Diesen Nachteil versucht das *Verfahren nach Powell* [Pow64] zu umgehen, indem es mit konjugierten Richtungen bei den Erkundungsschritten arbeitet. Dabei kann es aber im Falle von numerischen Ungenauigkeiten bei der Bestimmung der Suchrichtung oder der Durchführung der Liniensuche zu Konvergenzproblemen kommen [Zan67].

Schwefel [Schw95] gibt zur Komplexität der zuvor behandelten Verfahren Tabelle 1.1 an, die den unterschiedlichen Aufwand zur Minimierung einer quadratischen Funktion in Abhängigkeit von der Variablenanzahl *n* darstellt. Der besseren Übersicht halber werden nur die Größenordnungen angegeben. Der unterschiedliche Aufwand zur Berechnung eines Funktionswertes und dessen näherungsweisen Ableitungen wird im Vergleich zu den elementaren Operationen des Algorithmus' wie folgt gewichtet:

$$F : \nabla F : \nabla^2 F \text{ entspricht } n^0 : n^1 : n^2,$$

wobei mit ∇F der Gradientenvektor und mit $\nabla^2 F$ die Hesse-Matrix bezeichnet wird.

Damit können die Berechnungen des Funktionswerts und dessen angenäherten Ableitungen in einen ungefähren Bezug zu den Operationen des Verfahrens selbst gesetzt werden und es ergibt sich, daß der Vorteil einer geringeren Anzahl an Iterationen zum Teil durch den vermehrten Aufwand an elementaren Operationen wieder aufgehoben wird und umgekehrt. Tabelle 1.1 zeigt, daß kein Verfahren unter einen Aufwand von $O(n^3)$ kommt[1].

1. Die O-Notation gibt eine obere Schranke für eine Funktion *f(n)* an: *O(g(n))* heißt *asymptotische obere Schranke* von *f(n)*, wenn es Konstanten *c* und n_0 gibt, so daß $f(n) \leq c \cdot g(n)$ für alle $n \geq n_0$ gilt. Bei der Aufwandsabschätzung von Algorithmen wird *O(g(n))* auch als Zeitkomplexität oder einfach Komplexität bezeichnet, siehe auch [Saa02, Meh86].

Verfahren	Anzahl der Iterationen	Operationen pro Iteration			
		gewichtete Funktionsberechnungen			Elementare Operationen
		F	∇F	$\nabla^2 F$	
Newton (z.B. Newton-Raphson)	n^0	-	n^1	n^2	n^3
Variable Metrik (z.B. DFP)	n^1	n^0	n^1	-	n^2
Konjug. Gradienten (z.B. Fletcher-Reeves)	n^1	n^0	n^1	-	n^1
Konjug. Richtungen (z.B. Powell)	n^2	n^0	-	-	n^1

Tab. 1.1: Anzahl benötigter Operationen zur Minimierung einer quadratischen Funktion (entnommen aus [Schw95, S.171])

Der Grundgedanke des *Rosenbrock-Verfahrens* [Ros60] besteht darin, das Optimum entlang den Richtungen eines im n-dimensionalen Raum rotierenden Koordinatensystems zu suchen. Dabei wird die Hauptachse bei jedem Iterationsschritt in die den größten Fortschritt versprechende Richtung ausgehend vom aktuellen Punkt als Koordinatenursprung gelegt und die anderen Achsen werden orthogonal dazu aufgebaut. Eventuelle Beschränkungen werden mit einer partiellen internen Straffunktion berücksichtigt. Die Schrittweiten werden je nach Erfolg oder Mißerfolg verdreifacht oder halbiert [Ros60]. Es wird ein gültiger Startpunkt verlangt, der nicht zu nahe an den Grenzen liegen sollte. Numerische Untersuchungen zeigen, daß die Hauptachse bereits nach wenigen Iterationen in die Richtung des Gradienten zeigt, wodurch das Verfahren auch schmalen Graten in der Topologie des Suchraums folgen kann. Wie auch beim Powel-Verfahren und dem Algorithmus von Hooke und Jeeves werden keine Ableitungen benötigt. Da auch auf Liniensuche verzichtet wird, ist das Verfahren sehr robust. Nachteilig ist der nicht unerhebliche numerische Aufwand der Orthogonalisierung des rotierten Koordinatensystems, der bei *n* Parametern $O(n^3)$ beträgt [Schw95].

Einen anderen Weg gehen die *Polyederverfahren*, die sich schon dadurch von den bisher behandelten Algorithmen unterscheiden, daß sie mit mehreren statt mit nur einem Punkt arbeiten. $n+1$ Punkte spannen im n-dimensionalen Suchraum ein Polyeder auf, das durch die Operationen Kontraktion, Expansion und Reflexion (Spiegelung am Flächenschwerpunkt der restlichen Punkte) verändert wird. Das *Simplexverfahren* nach Nelder und Mead [Nel65], nicht zu verwechseln mit dem Simplex-Verfahren der linearen Programmierung nach Dantzig [Dan66], konstruiert den Simplex ausgehend von einem Startpunkt. Für jeden Eckpunkt werden die Funktionswerte bestimmt und der beste und schlechteste Punkt ermittelt. Die Iteration beginnt mit der Reflexion des schlechtesten Punkts. Im Falle einer Verbesserung gegenüber dem bisher besten Punkt wird zusätzlich eine Expansion versucht. Ist das Resultat degegen ein Funktionswert, der schlechter als der zweitschlechteste Punkt ist, wird eine Kontraktion durchgeführt. Wenn der so erhaltene Funktionswert schlechter als der schlechteste ist, erfolgt eine Kontraktion des Simplex' um den besten Punkt und eine neue Iteration beginnt. Das Verfahren endet, wenn die Standardabweichung der Funktionswerte des Simplex' vom Mittelpunkt geringer wird als ein vorgegebener Wert. Das Verfahren bewegt den Polyeder in Richtung eines möglicherweise lokalen Optimums und verkleinert ihn dabei letztlich. Da der Simplex bis auf die Schlußphase immer einen gewissen Ausschnitt des Suchraums erfaßt, kann das Verfahren auch Suboptima von geringerer Ausdehnung überspringen. Für wenige Parameter gilt das Verfahren als robust, wenn auch aufwendig [Schw95]. Die Komplexität liegt bei

$O(n^2)$ und die Anzahl der Funktionsaufrufe wird von Nelder und Mead mit $O(n^{2.11})$ bei maximal 10 Variablen angegeben.

Der Complex-Algorithmus wurde von Box speziell für beschränkte Problemstellungen in Anlehnung an das ursprünglich für unbeschränkte Aufgaben gedachte Simplex-Verfahren von Nelder und Mead entwickelt [Box65] (Complex = COnstrained siMPLEX). Die beiden wichtigsten Unterschiede sind die Expansion des Polyeders bei jeder Reflexion und die Betrachtung von mehr Eckpunkten, deren Zahl bei n Parametern nun zwischen $n+1$ und $2n$ liegt. Ausgehend von einem zulässigen Startpunkt (unzulässige Startwerte werden in der Initialphase in den zulässigen Bereich „geschoben“) werden die restlichen Punkte zufällig unter Berücksichtigung der Beschränkungen ermittelt. Ein weiterer Unterschied besteht im Abbruchkriterium, bei dem Box lediglich fordert, daß das Verfahren für eine vorgegebene Anzahl von Berechnungen der Zielfunktion hintereinander bis auf Rundungsfehler denselben Wert liefert. Box berichtet selbst, daß sein Verfahren in numerischen Tests ähnliche Ergebnisse liefert wie der Simplex-Algorithmus und daß beide hinsichtlich der Anzahl notwendiger Funktionsaufrufe schlechter sind als das Rosenbrock-Verfahren. Die Untersuchung wurde mit zwei-, drei-, fünf- und zehn-dimensionalen Testfunktionen durchgeführt und ab fünf Parametern schneiden beide Polyeder-Verfahren deutlich schlechter ab. Schwefel gibt an, daß hinsichtlich des Rechenaufwands für viele Variable keine Untersuchungen bekannt seien [Schw95]. Verglichen mit den anderen zuvor behandelten Verfahren haben die Polyeder-Strategien den Nachteil, daß sie dazu tendieren, in der Nähe des Optimums schlecht zu konvergieren, teilsweise sogar zu stagnieren. Ihr Vorteil besteht darin, Suboptima besser überspringen zu können als die zuvor genannten Algorithmen und ohne Ableitungen auszukommen.

Von den bisher betrachteten deterministischen Verfahren gibt es einige stochastische Varianten. Auf stochastische Ansätze wird häufig zurückgegriffen, wenn die deterministischen Verfahren entweder nicht anwendbar sind oder unbefriedigende Ergebnisse liefern. So wurde z.B. das Patternverfahren nach Hooke und Jeeves [Hoo61] derart modifiziert, daß die Erkundungsschritte nicht mehr starr parallel zu den Achsen sondern nach unterschiedlichen Strategien zufällig in alle Richtungen erfolgen. Dies erwies sich als vorteilhaft bei schmalen Graten oder bei Optima auf den Grenzen des Parameterraums [Eme66, Wil67, Bel72, Law72]. Brent hat den Powell-Algorithmus um eine hin und wieder angewandte zufällige Wahl der Suchrichtung bei der Liniensuche ergänzt, die bei Aufgabenstellungen mit vielen Parametern angewandt wird [Bre73]. Der lokale Charakter der Suche bleibt bei diesen Modifikationen allerdings erhalten.

indirekt	**direkt**
Gradientenverfahren	Gauß-Seidel-Verfahren
Verfahren konjugierter Gradienten	Verfahren nach Hooke und Jeeves
Quasi-Newton-Verfahren	Powel-Verfahren
DFP-Verfahren	Rosenbrock-Algorithmus
	Simplex- und Complex-Verfahren

Tab. 1.2: Überblick über lokale Suchverfahren

Tabelle 1.2 gibt einen Überblick über die besprochenen lokalen Suchverfahren. Zusammenfassend kann gesagt werden, daß die direkten lokalen Suchverfahren wegen der geringeren Voraussetzungen, die sie an die Zielfunktion stellen, allgemeiner angewandt werden können

als die indirekten Verfahren. Der Preis ist ein meist höherer Aufwand an Zielfunktionsberechnungen. In der Regel sind unter bestimmten Voraussetzungen Konvergenzbeweise und Aussagen über die Konvergenzgeschwindigkeit möglich. Allerdings kann es bei praktischer Anwendung aller Verfahren dazu kommen, daß die theoretisch vorhergesagten Werte über die Konvergenzgeschwindigkeit auf Grund numerischer Fehler nicht eingehalten werden können [Schw95]. Allen gemeinsam ist die bereits erwähnte Startpunktabhängigkeit. Als robuster Algorithmus kann das Rosenbrock-Verfahren angesehen werden, das bessere Ergebnisse liefert als der ebenfalls als robust geltende Complex-Algorithmus.

1.2.2 Globale Suchverfahren

Globale Suchverfahren zielen darauf ab, in einer umfassenderen Suche das globale Optimum zu ermitteln oder aber bei mehreren Suboptima von vergleichbarer Qualität eines oder mehrere davon zu identifizieren. Dieser Vorteil wird mit einer größeren Anzahl von Zielfunktionsberechnungen erkauft, sie konvergieren also langsamer. Zu den globalen Suchverfahren zählen die Evolutionären Algorithmen [Hol75, Rec73, Fog66], das Simulated Annealing [Met53, Kir83], das Branch-and-Bound-Verfahren [Dak65] und die Monte-Carlo-Methode [Bro58].

Der Vollständigkeit halber sei auch noch die Rastermethode als globales Suchverfahren erwähnt, bei der alle Achsen in meist äquidistante Abschnitte eingeteilt werden und so ein mehr oder weniger enges Raster über den gesamten Suchraum gelegt wird. Die Methode ist lediglich bei einer geringen Parameterzahl anwendbar: bei nur vier Parametern und einem groben Raster von zehn Werten je Parameter kommt man bereits auf 10000 Proben, wobei die Gefahr, z.B. eine etwas steiler aufsteigende Spitze mit schmaler Basis zu übersehen, recht groß ist.

Das Monte-Carlo-Verfahren ist als rein zufallsbasierte Suche bereits ab wenigen Parametern recht aufwendig. Es kann in Ausnahmefällen wie z.B. der Suche nach einer einsamen Spitze in einem ansonsten ebenen Suchraum effektiver als andere Verfahren sein, weil es keinerlei Berechnungen zur Durchführung einer wie auch immer gerichteten Suche ausführt. Da derartige Berechnungen der konkurrierenden Verfahren in der dargestellten Situation nutzlos sind, besteht nur im Falle der einsamen Spitze und ähnlich gelagerten Fällen ein Vorteil für die Monte-Carlo-Methode. Sie wird daher hier nicht weiter betrachtet.

Das Branch-and-Bound-Verfahren ist nur für beschränkte ganzzahlige und gemischt-ganzzahlige Probleme mit Nebenbedingungen geeignet. Bei gemischt-ganzzahligen Aufgaben muß ein Lösungsalgorithmus für den reellwertigen Teil existieren, der die Zielfunktion bei gegebenen Intervallgrenzen für die Integerwerte löst. Damit wird das Verfahren auf konkave Zielfunktionen eingeschränkt, bei denen die Nebenbedingungen zusammen mit der Bedingung, daß die Variablen positiv sind, ein konvexes Gebiet bilden. Unter der Voraussetzung, daß der reellwertige Teilalgorithmus endlich ist und die Integer-Variablen beschränkt sind, hat Dakin nachgewiesen, daß das Verfahren nach endlich vielen Schritten das Optimum findet [Dak65].

An Hand der ganzzahligen Variablen wird das Problem in Klassen eingeteilt, in dem ihnen nacheinander die zur Verfügung stehenden Werte zugewiesen werden (branch). Für jede Klasse wird so der maximal mögliche Wert der Zielfunktion berechnet und geprüft, ob Restriktionen verletzt wurden. Die Klasse mit zulässigen Parameterwerten und größtem möglichen Maximalwert wird zur weiteren Untersuchung ausgewählt, da in ihr das Optimum vermutet wird (bound). Der Algorithmus läuft so ab, daß zuerst für eine (beliebige) ganzzahlige Variable alle Klassen entsprechend ihrem Wertevorrat gebildet und geprüft werden. Mit der aussichtsreichsten wird weiterverfahren. Die anderen werden zurückgestellt und nur diejenigen mit Restrik-

tionsverletzungen werden ausgeschieden. Es entsteht so eine baumartige Suche, die zunächst auf die Teilbäume mit den besten möglichen Maximalwerten beschränkt wird. Wenn derart eine zulässige Lösung erreicht wurde, müssen noch diejenigen der vorher zurückgestellten Teilbäume untersucht werden, die einen größeren möglichen Maximalwert aufweisen, da in ihnen eine bessere Lösung liegen kann. Das Auffinden einer Lösung scheidet also nur die Teilbäume mit den schlechteren möglichen Maximalwerten aus. Je nach Aufgabenstellung kann das dazu führen, daß nach dem Auffinden des Optimums noch ein recht großer Teil des Baums untersucht werden muß, um die Optimalität nachzuweisen [Run73]. Die anfängliche Suche kann durch die Vorgabe einer unteren Schranke für das Optimum beschleunigt werden, sofern ein solches bekannt ist. Das Verfahren hat den Nachteil einer im ungünstigsten Fall exponentiellen Komplexität, die nur problembezogen eingeschränkt werden kann. Es ist daher bei größerer Parameterzahl bzw. größerem Wertevorrat nicht mehr allgemein anwendbar. In der Praxis wird es dann bisweilen als ausreichend angesehen, eine gültige Lösung mit einer vorgegebenen Mindestqualität zu finden, wobei auf das Optimum bzw. den Nachweis der Optimalität verzichtet wird. Dies kann zu erheblichen Rechenzeitersparnissen führen [Run73].

Das Simulated Annealing geht auf den Metropolis-Algorithmus [Met53] zurück, der ursprünglich dazu gedacht war, unter Ausnutzung der Boltzmann-Verteilung Stichproben in einem Parameterraum zu generieren. 30 Jahre später erkannten Kirkpatrick u.a. [Kir83] dessen Potential als Optimierungsverfahren. Das Simulated Annealing geht von einem willkürlich gewählten Startpunkt aus und berechnet dessen Zielfunktion, die als zu minimierende Energie E begriffen wird. Durch zufällige Veränderungen wird ein neuer Punkt erzeugt und mit dem alten verglichen: $\Delta E = E_{neu} - E_{alt}$. Ist die neue Energie kleiner als die alte, also ΔE kleiner als Null, wird er akzeptiert. Ist sie dagegen größer, wird er nur mit einer gewissen Wahrscheinlichkeit angenommen, die mit zunehmender Differenz kleiner wird und sich aus folgender Formel ergibt:

$$P(\Delta E) = \begin{cases} 1 & \text{für } \Delta E < 0 \\ \exp\left(-\frac{\Delta E}{T}\right) & \text{für } \Delta E \geq 0 \end{cases}$$

Der Steuerungsparameter T (Temperatur) ist am Anfang hoch und wird mit Fortschreiten des Algorithmus gesenkt, daher der Name “simuliertes Ausglühen”. Mit sinkender Temperatur wird es für Verschlechterungen immer unwahrscheinlicher angenommen zu werden, je größer sie sind. Damit sinkt auch die Möglichkeit, (lokale) Minima wieder zu verlassen. Die Formulierung einer geeigneten Temperaturverteilung ist der kritische Punkt bei der Anwendung des Verfahrens: die Temperatur muß erst erhöht und dann schrittweise verringert werden, wobei es auch auf die Dauer je Temperaturschritt ankommt. Werden viele kleine Schritte mit jeweils großer Verweildauer gewählt, wird das Optimum mit größerer Sicherheit gefunden oder ihm zumindest sehr nahe gekommen. Der Preis für dieses vorsichtige Vorgehen besteht in vielen, möglicherweise unnötig vielen Berechnungen der Zielfunktion. Wird dagegen die Temperatur zu rasch verändert, steigt die Wahrscheinlichkeit, an einem lokalen Optimum hängenzubleiben unter Umständen erheblich [Dav87a]. Leider sind die Temperaturverteilungen anwendungsabhängig. Aarts und Korst [Aar89] haben die Konvergenz des Verfahrens unter der Voraussetzung nachgewiesen, daß die Anzahl der Schritte zur Temperaturverringerung gegen unendlich strebt. Sie sprechen daher auch von einer asymptotischen Konvergenz. Häufig wird eine polynomiale Verteilung benutzt, die auf Aarts und Van Leuven [Aar85] zurückgeht. Sie bietet das Potential, Lösungen hoher Qualität zu finden und gilt als robust hinsichtlich des Startwertes. Andererseits reagiert sie empfindlich auf die Wahl der Schrittweite und damit der

Abkühlungsgeschwindigkeit [Aar89]. Letztlich basiert die Wahl einer geeigneten Temperaturverteilung auf „Daumenregeln“ und Erfahrung [Gro90].

Es gibt relativ wenig Literatur zum Simulated Annealing und wenn die seit 1990 alle zwei Jahre stattfindende Konferenz *Parallel Problem Solving from Nature* (PPSN), zu deren Themen das Simulated Annealing von Anfang an gehörte, zum Maßstab genommen wird, so kann ein stetig abnehmendes Interesse der Forscher an diesem Verfahren konstatiert werden [Schw91, Män92, Dav94, Ebe96, Eib98, Scho00, Mer02]. Und dies, obwohl durchaus über erfolgreiche Anwendungen des Verfahrens berichtet wurde [Aar89, Dav87b, Schw91]. Über die Motive dazu kann nur spekuliert werden.

Evolutionäre Algorithmen (EA) [Rec73, Hol75, Fog66] nutzen eine algorithmische Nachbildung des Evolutionsprozesses zur iterativen (generationsweisen) Verbesserung vorhandener Lösungsvorschläge. Da die Natur weder Vorwissen über die an die Umwelt anzupassenden Arten noch über die Art und die Herausforderungen dieser Umwelt hat, sind auch die daraus abgeleiteten Algorithmen hinsichtlich der Beschaffenheit des Suchraums nahezu voraussetzungsfrei. Es sind also keine speziellen Eigenschaften, wie z.B. die von den Gradientenverfahren geforderte Differenzierbarkeit, notwendig. Die einzige Voraussetzung besteht darin, daß die Zielfunktion so gestaltet ist, daß sie eine Suche in irgendeiner Weise zum Optimum hinführen kann. Das Gegenteil davon ist beispielsweise die bereits erwähnte einsame Spitze in einem ansonsten ebenen Suchraum, die alle Suchverfahren nur per Zufall finden können. Kombinatorische Komponenten der Aufgabenstellung stellen ebenso wenig ein Hindernis dar wie Variationen oder Störungen bei der Aufgabenstellung, die sich z.B. in verrauschten Ergebnissen der Zielfunktion, auch Fitnessfunktion genannt, niederschlagen. Evolutionäre Algorithmen unterscheiden sich in folgenden Punkten von den bisher vorgestellten Optimierungsverfahren:

1. Sie verwenden evolutionäre Operatoren wie Mutation oder Rekombination zur Erzeugung neuer Lösungsvorschläge ausgehend von einer geeigneten Darstellung möglicher Lösungen in Kettenform (Chromosome).
2. Sie arbeiten bis auf wenige Varianten mit einer Population von Lösungen und durchsuchen so den Lösungsraum parallel von verschiedenen Punkten aus.
3. Durch die Rekombination erfolgt ein Informationsaustausch zwischen den Mitgliedern einer Population.
4. Sie enthalten bewußt stochastische Elemente. Auf Grund der Selektion (survival of the fittest) entsteht daraus jedoch nicht einfach eine zufällige Suche sondern eine intelligente Durchmusterung des Suchraums, bei der sich der Suchprozeß schnell auf erfolgsversprechende Regionen konzentriert.

Die Evolutionären Algorithmen können in drei große Gruppen eingeteilt werden: die Evolutionsstrategie (ES) nach Rechenberg [Rec73, Schw81], die mit ihrer adaptiven Schrittweitensteuerung gute Eigenschaften bei der reinen Parameteroptimierung hat, die parallel dazu entwickelten Genetischen Algorithmen (GA) von Holland [Hol75], die eine gewisse Stärke bei mehr kombinatorischen Aufgabenstellungen haben und die von L.J. Fogel[2] [Fog66] ebenfalls unabhängig entwickelte Evolutionäre Programmierung (EP), die in vielen Aspekten der Evolutionsstrategie ähnlich ist.

2. Die Evolutionäre Programmierung wurde von L.J. Fogel in den frühen 60-iger Jahren entwickelt. Den aktuellen Stand gibt D.B. Fogel [Fog92] in seiner Dissertation wieder.

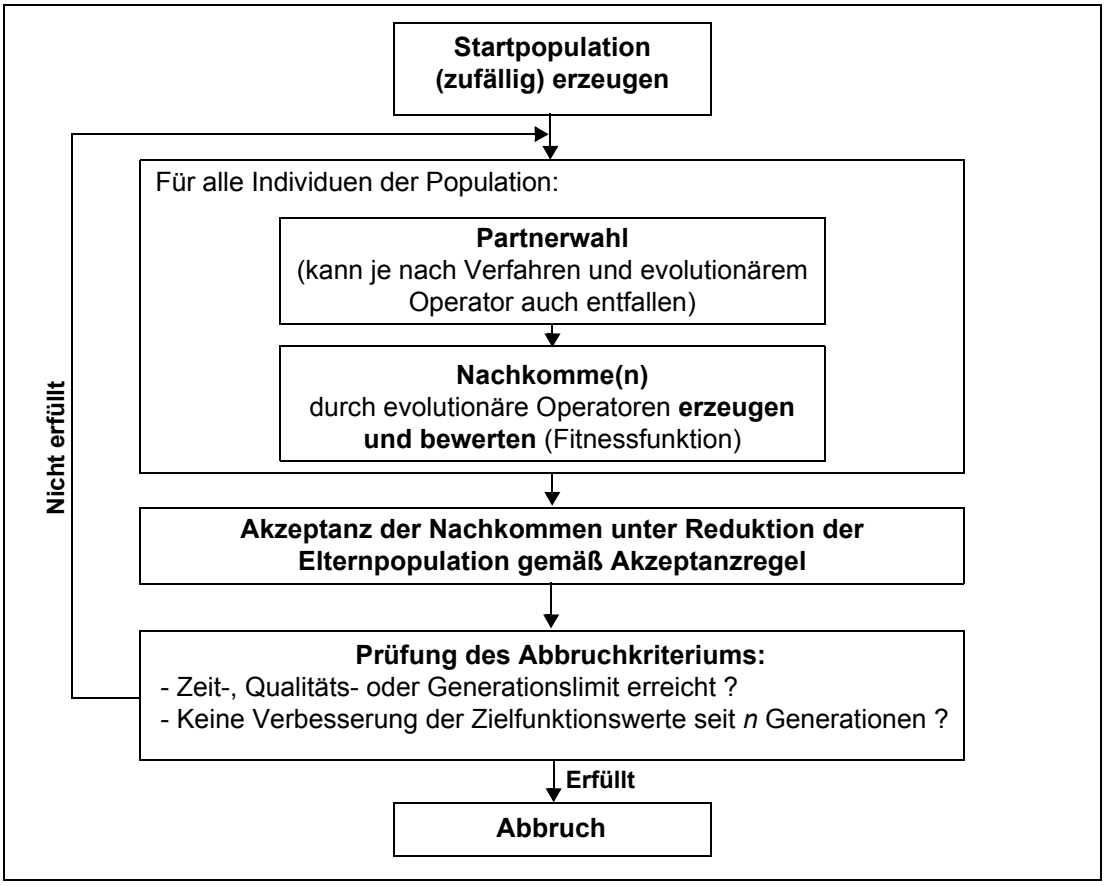


Abb. 1.2: Ablaufschema Evolutionärer Algorithmen

Abb. 1.2 gibt den generellen Ablauf eines Evolutionären Algorithmus wieder. Die Startpopulation kann vollständig per Zufall gebildet oder unter Verwendung von bereits bekannten Lösungen initialisiert werden. Bei Verfahren, die auch evolutionäre Operatoren verwenden, die zwei Individuen benötigen wie die Rekombination oder das Crossover, erfolgt eine Partnerwahl vor der Ausführung dieser Operatoren. Pro Individuum können auch mehrere Operatoren angewandt werden, so daß insgesamt auch mehrere Nachkommen erzeugt werden. Die Nachkommen werden bewertet und ersetzen gemäß einer Akzeptanzregel einen Teil der Elterngeneration, so daß die Größe der Population konstant bleibt. Es gibt verschiedene Formen der generationsweise oder pro Paarung erfolgenden Nachkommensakzeptanz. In jedem Fall aber findet dabei eine fitnessabhängige Selektion zu übernehmender Nachkommen und zu löschender Eltern statt. Schließlich wird entsprechend dem Ergebnis der Prüfung eines Abbruchkriteriums mit der nächsten Generation fortgefahren oder das beste Individuum als Ergebnis abgeliefert. Da sich auch mehrere unterschiedliche Individuen vergleichbarer Qualität in einer Population befinden können, kann das Ergebnis statt aus nur einer Lösung auch aus einer begrenzten Zahl von Alternativlösungen bestehen.

Durch eine entsprechende Ausgestaltung des Verfahrens kann die Balance zwischen Breiten- und Tiefensuche[3] beeinflusst und so eingestellt werden, daß dabei eine Adaption an die Gegebenheiten des Suchraums stattfindet [Gor90, Gor94]. Dadurch wird sowohl bei Genetischen Algorithmen als auch bei der Evolutionsstrategie [Gor98a, Gor99a, Gor99b] die Fähigkeit zum Auffinden des globalen Optimums gestärkt und die Konvergenzgeschwindigkeit erhöht.

3. Die Breitensuche versucht, den gesamten Suchraum grob zu durchmustern, während die Tiefensuche ein begrenztes Gebiet genauer exploriert.

Ein strenger mathematischer Beweis, wonach die Evolutionären Algorithmen das globale Optimum sicher mit einer bestimmten Anzahl an Generationen finden, steht noch aus. Aber die Erfahrung zeigt, daß die Wahrscheinlichkeit, es bei willkürlicher Startpunktwahl zu ermitteln, sehr hoch ist.

Seit Beginn der 90-iger Jahre hat die Zahl der Konferenzen und Literaturbeiträge zu diesem Thema sprunghaft zugenommen. Dabei spielen Berichte über erfolgreiche Anwendungen unterschiedlichster Art eine große Rolle. Stellvertretend für die Fülle der Veröffentlichungen sei auf die europäischen Konferenzserien *Parallel Problem Solving from Nature (PPSN)* [Schw91, Män92, Dav94, Ebe96, Eib98, Scho00, Mer02] und *Adaptive Computing in Engineering Design and Control* [Par94, Par96, Par98], auf die US-amerikanische Tagungsserie *International Conference on Genetic Algorithms (ICGA)* [Scha89a, Bel91, For93, Esh95, Bäc97], die daraus hervorgegangenen neuen Serien *Congress on Evolutionary Computing (CEC)* [CEC99, CEC00, CEC01] und *Genetic and Evolutionary Computation Conference (GECCO)* [Ban99, Whi00, Spe01] sowie auf eine Reihe von Buchveröffentlichungen [Dav91, Nis94, Schö94, Bäc98, Haf98] verwiesen. Nachstehende (sicher unvollständige) Liste gibt einen Eindruck von der Bandbreite der Anwendungen, wobei die meisten in die Klasse der NP-vollständigen Problemstellungen[4] fallen:

- Optimierung der Form von Leichtbaufachwerken (ES) [Höf76]
- Optimierung von Spiralfedern und Stoßdämpfern (ES) [Kob87]
- Kollisionsfreie Bahnplanung für Industrieroboter (Evolutionärer Algorithmus GLEAM, siehe Abschn. 1.3) [Blu90, Blu94b, Blu97, Blu00a]
- Optimierung der Struktur von Kohonen-Netzen (GA) [Pol92]
- Maschinelles Lernen (Genetische Algorithmen und die darauf aufbauenden Classifier-Systeme [Hol78], GLEAM [Blu90]) [Gol89, Jak92, Wei94, Kel00]
- Bestimmung von Fuzzy-Modellen ausgehend von Meßdaten (an das Problem angepaßter GA [Jäk97a, Jäk97b]
- Optimierung von Kaffeemischungen mit subjektiver Bewertung (ES) [Her96]
- Ressourcenoptimierung in der Verfahrenstechnik (GLEAM) [Blu94a]
- Werkstattbezogene Produktionsplanung (GLEAM) [Blu93a, Blu93b, Blu93c, Blu94c, Blu94d], (erweiterter hybrider GA[5]) [Bru93]
- Produktionsreihenfolgeplanung (erweiterter hybrider GA) [Mik98a, Mik98b]
- Optimierung von Stundentafeln (an das Problem angepaßter GA) [Col90]
- Optimierung von Zugfahrplänen (erweiterter GA GENOCOP [Mic92]) [Wez94]
- Designoptimierung (GLEAM) [Jak96a, Jak96b, Gor96, Süß97a, Süß97b, Jak98a, Jak98b, Gor98b, Pet99a, Sie99, Jak99b, Jak01a, Jak01c, Jak02b]
- Parameteranpassung von Makromodellen für die Designoptimierung (GLEAM) [Mei96, Mei98a, Mei98b]
- Optimierung der Lastverteilung bei Mehrprozessorrechnern (hybrider GA) [Dri92]
- Lösung eines Routing- und Scheduling-Problems aus dem Bereich der Telekommunikation (hybrider GA) [Cox91]
- Layoutoptimierung von Platinen mit mehreren Verbindungslagen (an das Problem angepaßter hybrider GA) [Lie94].

4. Für eine Erläuterung des Konzepts der NP-Vollständigkeit siehe auch [Sed92, Hor81].

5. Unter hybriden Genetischen Algorithmen wird die Kombination eines GA mit anderen Methoden wie heuristischen oder wissensbasierten Verfahren verstanden.

Bei Evolutionären Algorithmen mit elitärer Akzeptanzregel, bei der das beste Individuum nur durch ein besseres ersetzt werden kann, ist sichergestellt, daß die Qualität des besten Individuums einer Population von Generation zu Generation nur ansteigen kann:

Elitäre Akzeptanzregel:
Ein Individuum *x* wird gemäß folgender Vorschrift in der Nachfolgegeneration durch seinen aus genetischen Operationen wie Mutation oder Crossover hervorgegangenen Nachfolger x_{neu} ersetzt:

$$x^{(k+1)} = \begin{cases} x_{neu} & \text{für } F(x_{neu}) \geq F(x^{(k)}) \\ x^{(k)} & \text{für } F(x_{neu}) < F(x^{(k)}) \end{cases}$$

Dabei ist $F(x^{(k)})$ die zu optimierende Zielfunktion des Individuums *x* in der *k*-ten Generation. Unter Mutation wird die zufallsbedingte Änderung einer oder mehrerer Variablen des Individuums *x* verstanden und unter Crossover der ebenfalls zufallsbedingte Austausch von Variablenwerten zweier Individuen.

Wenn ein Optimum existiert, wird mit einer bestimmten Wahrscheinlichkeit $P > 0$ eine Verbesserung pro Generation stattfinden. Daher ergeben die Werte der Zielfunktion der jeweils besten Individuen x' einer Generation eine monoton nicht fallende Zahlenfolge, die bekanntermaßen beschränkt ist:

$$F(x'^{(1)}) \leq F(x'^{(2)}) \leq F(x'^{(3)}) \leq \ \ldots \ \leq F(x'^{(k)}) \leq \ \ldots$$

Daraus folgt die Konvergenz der Zahlenfolge gegen das Optimum. Die Aussage läßt allerdings keinen Rückschluß auf die Konvergenzgeschwindigkeit zu; im Extremfall sind beliebig viele Generationen zum Erreichen des Optimums notwendig. Für die Praxis gibt die Überlegung daher leider wenig her, außer der naheliegenden Empfehlung, elitäre Akzeptanzregeln zu verwenden.

Evolutionäre Algorithmen gehören zu den allgemeinsten bekannten Such- und Optimierungsverfahren. Ihr Nachteil besteht in einer vergleichsweisen großen Anzahl an Zielfunktionswertberechnungen und dem Umstand, in der unmittelbaren Nähe eines Optimums langsam zu konvergieren, was in der ungerichteten Suche der Mutation begründet ist[6]. Die langsame Konvergenz verwundert angesichts des natürlichen Vorbilds der Verfahren auch nicht: Die sich aus schneller Konvergenz ergebende Uniformität einer biologischen Population ist tödlich bei einer hinreichend großen Umweltänderung, da die Art dann mangels genetischer Variationen nicht mehr adaptionsfähig ist und aussterben muß. Der Vorteil Evolutionärer Algorithmen liegt vor allem in ihrer Fähigkeit, auch noch bei multimodalen und vieldimensionalen Suchräumen anwendbar zu sein und ohne problemspezifisches Vorwissen auskommen zu können. Ein weiterer Vorteil ist ihre leichte Parallelisierbarkeit, mit der die Kapazitätsreserven der heutzutage weit verbreiteten aber selten voll ausgelasteten Rechnernetze besser genutzt werden können. Diese Eigenschaft kompensiert in einem bestimmten Maße den erhöhten Rechenaufwand.

6. Dies gilt für die ES mit ihrer adaptiven Schrittweitensteuerung nur in begrenztem Maße.

1.2.3 Hybride Verfahren

Der Gedanke liegt nahe, Evolutionäre Algorithmen mit anderen meist problemspezifischen Verfahren zu kombinieren. Bereits Holland [Hol75] hat vorgeschlagen, GAs als eine Art *preprocessor* für die anfängliche Suche zu nutzen, bevor der Optimierungsprozeß mit einem lokalen Verfahren, das anwendungsspezifisches Wissen nutzen kann, fortgesetzt wird. Die dabei entstehenden hybriden Algorithmen (auch *memetic algorithms* genannt [Mos92]) können bezüglich ihres Aufbaus in vier Gruppen eingeteilt werden:

1. Aufgabenteilung
   Dabei wird das Problem in meist zwei Teilaufgaben zerlegt, von denen dann die eine vom EA bearbeitet wird. Der EA gibt jede generierte Lösung an das lokale Verfahren zur Optimierung der Parameter der zweiten Teilaufgabe, wobei die vom EA bestimmten Parameter fixiert bleiben. Der wesentliche Unterschied zu den drei nachfolgenden Gruppen besteht darin, daß beide Verfahren nur ihren jeweiligen Problem- und Parameteranteil sehen und bearbeiten.

2. Initialisierung der Startpopulation
   Mit Hilfe eines meist heuristischen Verfahrens wird die gesamte Startpopulation oder nur ein Teil davon initialisiert. Ein eventueller Rest wird zufällig bestimmt. Der Grundgedanke besteht darin, die evolutionäre Suche bereits mit meist zulässigen Lösungen einer bestimmten Qualität beginnen zu lassen und so die Zeit zum Auffinden befriedigender Bereiche des Suchraums zu sparen.

3. Nachoptimierung der EA-Ergebnisse
   Die Nachoptimierung der EA-Ergebnisse mit konventionellen Verfahren wird häufig auch als „Local Hill Finding + Local Hill Climbing“ bezeichnet. Der EA wird zum Auffinden einiger vielversprechender Regionen des Suchraums benutzt und anschließend wird ein konventionelles Verfahren zur Bestimmung der exakten Optima in der Hoffnung, dabei das globale Optimum zu finden, verwendet. Das beste dieser Optima ist dann jedenfalls die Lösung.

4. Direkte Integration
   Bei der direkten Integration konventioneller Verfahren in die evolutionäre Suche gibt es mehrere Möglichkeiten. Lokale Verfahren können als eine spezielle Mutation oder als Reparaturkomponente realisiert werden, die durch das Crossover und Mutationen entstandene unzulässige Lösungen wieder in zulässige verwandelt und gegebenenfalls verbessert. Eine weitere Variante besteht darin, daß alle oder nur ein Teil der vom EA erzeugten Nachkommen mit einem lokalen Verfahren optimiert werden. Ziel dabei ist, daß der EA nur noch über der Menge der lokalen Optima des Suchraums operieren muß.

Die ersten drei Gruppen lassen sich relativ leicht realisieren, da sie im Grunde auf eine Hintereinanderausführung unterschiedlicher Optimierungsverfahren hinauslaufen, die für sich jeweils unangetastet bleiben, während hingegen die direkte Integration einen größeren Eingriff in den Evolutionären Algorithmus selbst darstellt.

Süer et al. [Süe99] benutzen den Ansatz der Aufgabenteilung zur Produktionsplanung von Arbeitszellen. Der GA bestimmt die Auftragsreihenfolge, während die Auftragszuordnung zu den Zellen nach dem Prinzip der geringsten Auslastung erfolgt. Für die Arbeitsverteilung in den Zellen wird Moore’s Algorithmus [Moo68] verwendet. Mit diesem dreistufigen Ansatz erreichen sie gute Ergebnisse und zeigen, daß die Bedeutung des Genetischen Algorithmus

mit zunehmender Problemgröße ansteigt. Dahal et al. [Dah99] teilen die Aufgabe der Schiffsabfertigung an einer Tankanlange zur Aufbereitung von Ballastwasser in eine GA-basierte Tankzuordnung und eine regelbasierte Heuristik zur Bestimmung der Füllgeschwindigkeiten. Sie erreichen mit ihrem hybriden Ansatz bei realistischen Planungsaufgaben bessere Ergebnisse als im bisherigen Betrieb mit der Heuristik allein. Grimbleby [Gri99] verwendet einen GA zur Festlegung einer analogen Schaltung und parametriert sie mit dem DFP-Verfahren [Dav59, Fle63]. Er erreicht dabei bessere Ergebnisse als mit einem reinen GA- oder EP-Ansatz oder als mit Hand. Obwohl die Beispiele zeigen, daß es sich bei der Aufgabenteilung um einen interessanten und vielversprechenden Ansatz handelt, wird er im Rahmen dieser Arbeit nicht weiterverfolgt, da es sich immer um ein problemspezifisches Vorgehen handelt: Es kann eben nur an Hand einer konkreten Aufgabenstellung eine sinnvolle Teilung bestimmt werden.

Für praktische Anwendungen hat die Initialisierung der Startpopulation mit einer bekannten und erprobten Heuristik noch den Vorteil der besseren Akzeptanz beim Anwender, siehe auch Davis [Dav91]. Davis begründet das damit, daß das neue unbekannte Verfahren ja auf etwas Bekanntem aufsetzt, auf dessen Lösungen im Zweifelsfalle schnell zurückgegriffen werden kann. Mikut [Mik98a, Mik98b] benutzt eine Reihe von unscharfen Regeln zur Initialisierung der Startpopulation eines GAs und eine Heuristik als zusätzlichen Mutationsoperator bei einer Produktionsplanungsaufgabe für Ringwalzwerke. Beides führt zu einer deutlichen Verkürzung der Laufzeiten, die nur so auf ein für den Anwender akzeptables Maß reduziert werden konnten. Lienig und Brandt [Lie94] initialisieren ihren GA zur Layoutoptimierung von Platinen mit mehreren Verbindungslagen mit einer speziell auf die Aufgabenstellung zugeschnittenen Heuristik und untersuchen, ob so ein Vorteil gegenüber einer rein zufällig bestimmten Startpopulation erzielt werden kann. Dabei brechen sie einen heuristisch initialisierten Lauf bei Erreichen der Lösungsqualität, die sie mit einer zufälligen Startpopulation erreichen, ab und nehmen die benötigte Generationsanzahl als Vergleichsmaßstab. Interessanterweise wird eine Verbesserung bei allen Testaufgaben erreicht, wenn nur 2 von 50 Individuen heuristisch bestimmt sind, bei 4 von 50 kommt es nur teilweise zu einer Verbesserung und bei 10 von 50 kommt es dagegen zu zum Teil deutlichen Verschlechterungen. Dies liegt offenbar an der kleinen Populationsgröße von 50 und an dem verwendeten Populationsmodell, das eine schnelle Ausbreitung guter Individuen ermöglicht. Generell ist bei der Verwendung konventioneller Verfahren zur Initialisierung auf eine ausreichende Divergenz der Startpopulation zu achten, da sonst die Gefahr einer frühzeitigen Konvergenz auf ein Suboptimum besteht. Mikut umgeht dieses Problem, indem er nur ein aus den unscharfen Regeln abgeleitetes Individuum in die ansonsten zufällig erzeugte Startpopulation einfügt [Mik98b]. Auch bei der ES führt die heuristische Initialisierung der Startpopulation zu einer Verbesserung der Lösungsqualität. So berichtet Zimmermann [Zim85] von einem praktischen Produktionsplanungsproblem in der Tapetenindustrie, bei dem er ein Rangziffernverfahren sowohl zur Initialisierung als auch als Vergleichsmaßstab für die erzielte Qualität der ES benutzt. Er erreicht in seinen beiden Beispielen eine Verbesserung der Zielfunktion um 17,7% bzw. 30%.

Lienig und Brandt benutzen eine weitere Heuristik zur lokalen Nachoptimierung des besten Individuums, das ihr GA ermittelt hat, und vergleichen die Ergebnisse mit dem bereits erwähnten heuristischen Verfahren zur Erzeugung einiger Individuen der Startpopulation. Dabei kommt es in allen vier Testfällen zu einer Verbesserung der Lösungsqualität [Lie94]. Powell et al. [Pow89] kombinieren einen GA mit einem Expertensystem und lösen damit Designaufgaben iterativ in dem Sinne, daß GA und Expertensystem ihre Ergebnisse mehrmals, überwacht durch den Anwender, austauschen. Sie erreichen damit eine bis zu zehn mal schnellere Bearbeitung als von Hand. Cox et al. [Cox91] setzen einen GA für ein Routing- und Schedu-

ling-Problem aus dem Bereich der Telekommunikation ein und verbessern das Ergebnis mit dem 2-Opt-Verfahren[7]. Dieser hybride GA liefert bessere Ergebnisse als alle anderen getesteten Heuristiken und der GA ohne Heuristik. Nissen [Nis94] kombiniert eine EP- und eine ES-Variante ebenfalls mit dem 2-Opt-Verfahren zur Lösung von Quadratischen Zuordnungsproblemen [Bur90] und erreicht damit bei großen Problemstellungen bessere Ergebnisse, als jedes der beiden Verfahren für sich liefert. Er vergleicht unterschiedlich lange Läufe beider evolutionärer Verfahren mit anschließender lokaler Optimierung und ermittelt so problemgrößenabhängige Mindestwerte für die Generationsanzahl. Nissen weist auf das wichtige Problem des geeigneten Zeitpunkts zum Wechseln der Verfahren hin: Wann kann die evolutionäre Suche als beendet betrachtet werden, woran läßt sich erkennen, daß die Suche nahe genug am globalen Optimum ist, um auf ein lokales Verfahren umschalten zu können? Dieser Frage wird in Abschnitt 3.2.2 noch weiter nachgegangen werden.

Driessche und Piessens [Dri92] verwenden zwei einfache Heuristiken für die direkte Integration, um die Nachkommen bei ihrem GA zur Optimierung der statischen Lastverteilung bei Mehrprozessorrechnern zu verbessern. Sie berichten von einer deutlichen Leistungssteigerung. Mühlenbein [Mühl89] benutzt das 2-Opt-Verfahren zur lokalen Verbesserung der Nachkommen seines parallelen GAs zur Lösung des Quadratischen Zuordnungsproblems und findet eine bessere Lösung für das Beispielproblem von Steinberg, als seinerzeit bekannt war [She86]. Waagen et al. [Waa92] benutzen das Verfahren von Hooke und Jeeves [Hoo61], um die Nachkommen ihres Evolutionary Programming Algorithmus lokal zu optimieren und erproben es an drei Testfunktionen. Sie berichten von einer Verbesserung der Ergebnisgenauigkeit um mehrere Größenordnungen bei „gewöhnlich weniger Iterationen“. Nissen [Nis94, S.348] bezweifelt letzteres: „Hinsichtlich der Anzahl benötigter Iterationen ist der Vergleich von Waagen et al. nicht ganz fair, da er sich auf EP-Generationen und nicht auf die insgesamt benötigte Anzahl von Lösungsevaluierungen stützt. Der Hybrid-Ansatz dürfte, wegen der häufigen lokalen Optimierungen, deutlich mehr Lösungsevaluierungen benötigen als EP allein.“ Bruns [Bru93] verwendet anwendungsspezifisches Vorwissen für seine problembezogenen Mutations- und Crossover-Operatoren zur Bearbeitung großer und durch reale Aufgaben motivierter Produktionsplanungsprobleme. Er initialisiert die Population mit konventionellen Scheduling-Algorithmen und geht somit von gültigen, wenn auch suboptimalen Lösungen aus. Die Heuristik seines Crossover-Operators garantiert, daß immer gültige Lösungen erzeugt werden. Bei einem Vergleich zwischen anwendungsneutralen genetischen Operatoren und seinen anwendungsbezogenen konnte bereits nach einem Drittel der Evaluationen ein deutlich besseres Ergebnis erzielt werden, was bei einer Fortsetzung der Evolution auch noch weiter verbessert werden kann. Dorne und Hao [Dor98] benutzen ein problemspezifisches Crossover und Tabu Search[8] als lokalen Suchoperator, um das Problem der minimalen Färbung eines Graphen zu lösen. Dabei geht es darum, die Knoten eines Graphen so einzufärben, daß keine zwei benachbarten Knoten die gleiche Farbe aufweisen. Sie erreichen damit

7. Lin’s 2-Opt-Algorithmus beruht als lokales Verfahren auf dem Prinzp des Zweiertausches [Lin65].

8. Tabu Search [Glo97] geht von einer zufällig oder gezielt erzeugten Startlösung aus. Entsprechend einer zuvor zu definierenden Nachbarschaftsrelation werden alle benachbarten Mitglieder einer Lösung heuristisch bewertet und das beste ausgewählt. Die so erhaltene neue Lösung, die auch schlechter sein kann als die bisherige, wird akzeptiert, wenn der dazu notwenige Schritt (*move*) als Eintrag in der Tabu-Liste nicht gesperrt ist. In der zyklich organisierten Tabu-Liste werden alle zu den durchgeführten Schritten inversen Schritte für eine bestimmte Zeit notiert, um die Rückkehr zu bereits untersuchten Lösungen zu verhindern. Trotz der Sperre können Schritte vollzogen werden, wenn sie ein zuvor definiertes Kriterium erfüllen. Die Nachbarschaftsdefinition sind zusammen mit der Tabu-Listen-Länge und dem zovor erwähnten Kriterium die wesentlichen Elemente der Tabu-Search Methodik.

bei einer Reihe von Benchmarkaufgaben (2nd Dimacs Challenge) bessere Ergebnisse, als bisher in der Literatur bekannt waren, können aber leider keine vergleichbaren Angaben zu den Laufzeiten machen. Burke et al. [Bur95] benutzen ebenfalls Techniken zum Einfärben von Graphen für ihren hybriden GA zur Erstellung von Examensplänen, einem Spezialfall des Problems der Stundenplanerstellung. Sie arbeiten mit einer heuristischen Initialisierung, die konfliktfreie wenn auch suboptimale Lösungen produziert und mit heuristisch modifizierten genetischen Operatoren, die die Einhaltung der harten Restriktionen gewährleisten und damit ebenfalls konfliktfreie Nachkommen erzeugen. Yang et al. [Yan99] fügen das Simplex-Verfahren als zusätzlichen Operator in ihren GA zur Identifikation komplexer Systeme (Tomographiebilder und Modellierung des zentralen Stoffwechsels) ein. Der hybride GA wird von einer Überwachungsebene mit anwendungsspezifischen Algorithmen gesteuert. Sie erreichen damit hinsichtlich Geschwindigkeit und Ergebnisqualität bessere Resultate als mit einem traditionellen GA. Konak und Smith [Kon99] integrieren eine anwendungsspezifische Heuristik in den Mutationsoperator ihres binärcodierten GAs zur Optimierung von Kommunikationsnetzwerken. Sie erreichen damit bei ausgewählten Testproblemen in allen Fällen bis auf den Fall eines engmaschigen Netzwerks bessere Ergebnisse als mit Tabu Search. Sie führen dies darauf zurück, daß die verwendete Heuristik ab einer zu großen Kantenanzahl des Netzes versagen kann. Die gleiche Aufgabenstellung bearbeiten Ljubic, Raidl und Kratica [Lju00], wobei sie einen GA mit einer Heuristik zur Reparatur von Mutationsdefekten und zur Verbesserung der Nachkommen bei Lamarckscher Evolution (siehe Abschnitt 3.2.3) verwenden. Sie vergleichen ein gutes konventionelles Verfahren, den Original-GA und ihren hybriden GA an Hand von 25 Testfällen und kommen zu dem Ergebnis, daß ihr Hybrid durchweg die besten Ergebnisse liefert und dies meist auch deutlich schneller als der GA.

Vor allem die Arbeiten von Davis [Dav91], Bruns [Bru93], Nissen [Nis94] und Mikut [Mik98a, Mik98b] zeigen, daß durch den Einsatz problembezogener Heuristiken oder wissensbasierter Methoden deutliche Leistungsverbesserungen der benutzten Evolutionären Algorithmen erreicht werden konnten, die unter den gegebenen Randbedingungen der Laufzeit oder der Komplexität einen praktischen Einsatz zum Teil überhaupt erst ermöglichten. Der Preis dafür liegt in der Einschränkung der Anwendbarkeit des resultierenden hybriden EAs: Je mehr Vorwissen in das Verfahren einfließt, um so stärker wird die Anwendungsbandbreite eingeschränkt. Andererseits steigt so die Möglichkeit, auch noch bei hochkomplexen Anwendungsproblemen unter realistischen Bedingungen gute Lösungen in vorgegebener Zeit zu erzeugen.

Goldberg und Voessner fassen in ihrem Beitrag zur systemtheoretischen Analyse von Hybriden bestehend aus globalen und lokalen Suchverfahren [Gol99] die Geschichte der EA-Hybriden zusammen und ziehen daraus folgende Schlüsse:

1. Nahezu alle ernsthaften EA-Anwendungen für praktische Probleme integrieren irgendeine Form problemspezifischer Suche in den EA.

2. Bei der direkten Integration sollte eine genotypische Anpassung des verbesserten Nachkommens entsprechend dem Ergebnis der lokalen Suche gar nicht oder nur selten erfolgen, damit die genotypische Vielfalt erhalten bleibt. Orvos und Davis [Orv93] empfehlen pro 20 lokale Verbesserungen eine Anpassung.

3. Den vielen, zum Teil anwendungsspezifischen Untersuchungen zur geeigneten Verfahrenskombination auf der Ebene geeigneter Parametrierungen und anderer Details der konkreten Hybridisierung steht ein Mangel an Analysen auf globaler Ebene gegenüber:

Wie werden lokale und globale Suchverfahren geeignet kombiniert und wie die Ressourcen an Rechenzeit günstig aufgeteilt?

Goldberg und Voessner entwickeln einen systemtheoretischen Ansatz zur Beantwortung der Frage nach optimaler Rechnerzeitaufteilung zwischen (idealisierten) globalen und lokalen Suchverfahren ausgehend von zwei Zielen: zum einen minimaler Zeitverbrauch bei gegebener Erfolgswahrscheinlichkeit und zum anderen maximale Sicherheit bei gegebener Zeit. Sie teilen dabei den Suchraum in Ziel- und Attraktionsgebiete ein. Bei Erreichen des Zielgebiets gilt die Suche als erfolgreich abgeschlossen und ein Attraktionsgebiet ist dadurch definiert, daß von jedem seiner Punkte das Zielgebiet durch das lokale Verfahren innerhalb einer vorgegebenen Zeit garantiert erreicht wird. Wenn die Gebiete und die Wahrscheinlichkeit des globalen Verfahrens, sie zu erreichen, sowie die Zeitgrenze für das lokale Verfahren bekannt sind, lassen sich Aussagen über eine optimale Ressourcenaufteilung machen. Nur sind bei praktischen Problemstellungen die genannten Voraussetzungen meist nicht erfüllt, wie die Autoren selber einräumen. Der Beitrag zeigt allerdings auch, daß allgemeingültige oder zumindest verallgemeinerbare praktikable Regeln zur geeigneten Kombination lokaler und globaler Suchverfahren noch ausstehen.

Goldberg und seine Schüler haben in weiteren Arbeiten zwei Ansätze zur globalen Steuerung von EA-Hybriden geliefert. In einer früheren Arbeit wird versucht, das Problem dadurch zu lösen, daß die Verbesserung der kombinierten Verfahren jeweils getrennt gemessen und basierend auf dem jeweils festgestellten Erfolg entschieden wird, welches Verfahren in welchem Maße anzuwenden ist [Lob97]. Dabei können die „Erfolgserfahrungen“ zeitlich gewichtet werden, so daß ältere Verbesserungen gegenüber neueren weniger Bedeutung haben. In einem zweiten Ansatz wird ein Hybrid basierend auf der direkten Integration eines lokalen Suchverfahrens mit adaptiver Steuerung vorgeschlagen [Esp01]. Hierbei werden dem lokalen Verfahren eine Ausführungsfrequenz, -wahrscheinlichkeit und -genauigkeit zugeordnet, die wiederum basierend auf dem Erfolg gesteuert werden. Die Ausführungsfrequenz regelt die Häufigkeit der Generationen, in denen das lokale Verfahren angewandt wird. Die Ausführungswahrscheinlichkeit bestimmt den Anteil der Individuen, die in einer Generation mit lokaler Suche verbessert werden und die Genauigkeit regelt die Anzahl maximaler Iterationen des lokalen Verfahrens. Leider testen die Autoren ihren Ansatz nur an zwei Testfunktionen, die zwar multimodal aber nur zweidimensional sind und daher doch als eher einfach angesehen werden müssen, so daß der berichtete Erfolg mit der nötigen Vorsicht zu bewerten ist. Die Idee der direkten Integration besteht im Grunde darin, daß die globale Suche nur noch über die vom lokalen Verfahren gelieferten „Bergspitzen“ der lokalen Optima erfolgt. Dem widerspricht das Konzept von Generationen ohne lokale Optimierung genauso wie die Verbesserung nur eines Teils der Population. Bemerkenswert ist hingegen die Idee, den Aufwand für die lokale Suche anfänglich gering zu halten und dabei bestimmte Qualitätseinbußen bei den Zwischenergebnissen der gesamten Suche in Kauf zu nehmen.

Dieser Gedanke wird auch in der Arbeit von Zitzler, Teich und Bhattacharyya [Zit00] benutzt, die von lokalen Verfahren mit einem Strategieparameter ausgehen, über den sich Aufwand und Qualität steuern lassen. Sie beantworten die Frage nach der Veränderung dieses Parameters mit statischen und dynamischen Verteilungen basierend auf Stagnationsphasen, die beide von einem fixen Zeitbudget für die Optimierung ausgehen. Durch den so geschaffenen fixen Rahmen lassen sich Verteilungen auch leichter definieren, als wenn die Erreichung einer Mindestqualität oder das bestmögliche Ergebnis das Ziel ist. Generell entsteht das gleiche Pro-

blem wie bei der Definition einer geeigneten Kühlungsverteilung beim Simulated Annealing. Der Ansatz wurde zwar erfolgreich aber leider nur an einem Beispiel getestet.

1.2.4 Offene Probleme bei Evolutionären Algorithmen

Zusammenfassend bleiben für den Einsatz Evolutionärer Algorithmen nachstehende Probleme offen, die einen Einsatz erschweren:

1. Durch das Fehlen eines Konvergenzbeweises besteht keine Sicherheit in der Frage, ob die gefundene Lösung auch das globale Optimum darstellt.
2. Die Konvergenzgeschwindigkeit Evolutionärer Algorithmen nimmt in der Regel mit der Annäherung an das Optimum ab.
3. Es gibt keine verläßlichen Regeln für günstige Verfahrenseinstellungen sondern bestenfalls Erfahrungswerte für Parameter wie Populationsgröße oder Mutationsrate ausgehend von bekannten Eigenschaften eines Problems, wie z.B. seiner Parameteranzahl.

Als Abhilfe vor allem für den zweiten Punkt werden bei praktischen Anwendungen in der Regel EA-Hybride benutzt, bei denen folgende offene Probleme bestehen:

1. Sie benutzen fast alle anwendungsspezifisches Wissen oder Verfahren und sind damit nicht mehr allgemein anwendbar. Dadurch entsteht bei jeder neuen Aufgabenstellung ein zusätzlicher Anpassungs- und Implementierungsaufwand, bevor die eigentliche Aufgabe bearbeitet werden kann.
2. Die Frage nach einer geeigneten Kombination lokaler und globaler Suchverfahren und einer günstigen Aufteilung der Rechenzeit zwischen ihnen ist ungelöst.
3. Bei der direkten Integration ist es angesichts sich widersprechender Untersuchungsergebnisse nicht klar, ob und wenn ja mit welcher Häufigkeit eine genotypische Anpassung an die Lösung des lokalen Suchverfahrens erfolgen soll.

1.3 Ziele und Aufgaben

Die vorliegende Arbeit geht von einer der mächtigsten Klassen globaler Suchverfahren, den Evolutionären Algorithmen, aus und versucht, einen Beitrag zur Lösung ihrer Nachteile zu leisten. Dabei werden insbesondere die drei zuvor genannten offenen Punkte hybrider EA unter besonderer Berücksichtigung der Fragen der Konvergenzsicherheit und -geschwindigkeit behandelt.

Zum Erreichen des Ziels der Arbeit sind folgende Teilaufgaben zu lösen:

1. Erarbeitung einer neuen Methodik zur Kombination von lokalen Suchverfahren mit Evolutionären Algorithmen unter Wahrung der allgemeinen Anwendbarkeit des resultierenden Hybrids (Kapitel 3).
2. Auswahl geeigneter lokaler und evolutionärer Verfahren (Abschnitt 3.1).
3. Entwicklung eines neuen Steuerungsverfahrens zur Aufteilung der Rechenzeit zwischen Evolutionärem Algorithmus und lokalem Suchverfahren (Abschnitt 3.2.2).

4. Für empirische Untersuchungen der neuen Methode werden repräsentative Benchmarkaufgaben benötigt, die schnell genug sein müssen, um statistische Untersuchungen zu ermöglichen. Deren Auswahl wird in Kapitel 4 getroffen und begründet.
5. Konzeption und Implementierung der Erweiterung des ausgewählten EA um die vorgeschlagenen lokalen Suchverfahren entsprechend der neuen Methode (Abschnitt 5.1).
6. Zur Überprüfung der Ziele einer beschleunigten Konvergenz unter Beibehaltung der Konvergenzsicherheit werden experimentelle Untersuchungen an den zuvor ausgewählten Benchmarkaufgaben durchgeführt und ausgewertet (Abschnitt 5.2). Abschnitt 5.3.1 ist der Frage des Konvergenzverhaltens gewidmet und Abschnitt 5.3.2 faßt die Untersuchungsergebnisse zusammen.

Als Konsequenz aus den durchgeführten Untersuchungen wird in Kapitel 6 ein neues Konzept einer adaptiven Steuerung für die erfolgreichste Hybridisierungsart vorgestellt. Die Arbeit schließt in Kapitel 7 mit einer Zusammenfassung der gefundenen Ergebnisse und einem Ausblick auf die offen gebliebenen Probleme.

2. Evolutionäre Algorithmen und lokale Suchverfahren

In diesem Kapitel werden nach einer einleitenden Darstellung wesentlicher hier interessierender Elemente der Evolutionstheorie zwei der klassischen Verfahren, nämlich die Genetischen Algorithmen und die Evolutionsstrategie, näher vorgestellt. Danach erfolgt eine ausführlichere Beschreibung der in der vorliegenden Arbeit benutzten Algorithmen, deren Auswahl in Kapitel 3 begründet wird.

2.1 Evolutionstheorie - das Vorbild Evolutionärer Algorithmen

Da die Evolutionären Algorithmen von den grundlegenden Prinzipien der biologischen Evolution inspiriert sind, sollen in diesem Abschnitt die wesentlichen Elemente der Evolutionstheorie in der gebotenen Kürze dargestellt werden. Die Evolutionstheorie geht auf Darwins Werk „On the Origin of Species by Means of Natural Selection“ [Dar60] zurück. Darwin beschreibt darin die Evolution als einen stufenweisen Prozeß des Zusammenwirkens von zufälliger Variation einerseits und Selektion als natürlicher Zuchtwahl andererseits. Dabei spielt die Vorstellung von einem Nachkommenüberschuß eine zentrale Rolle: „As many more individuals of each species are born than can possibly survive; and as, consequently, there is a frequently recurring struggle for existence, it follows that any being, if it vary however slightly in any manner profitable to itself, under the complex and sometimes varying conditions of life, will have a better chance of surviving, and thus be *naturally selected*. From the strong principle of inheritance, any selected variety will tend to propagate its new and modified form.“ [Dar60, S.5]. Für Darwin, der noch nicht auf die Erkenntnisse der Genetik zurückgreifen konnte, ist die Evolution vor allem ein Selektionsprozeß basierend auf graduellen erblichen Änderungen.

Unsere heutigen Vorstellungen von den Mechanismen der Evolution basieren neben den Arbeiten Darwins auf Mendels Vererbungslehre, auf der theoretischen Populationsgenetik und auf molekularbiologischen Erkenntnissen. Heute dominiert die sogenannte synthetische Theorie der Evolution, die Wuketits folgendermaßen beschrieben hat: „Darwin plus klassische Genetik plus Populationsgenetik.“ [Wuk85, S.71]. Nissen [Nis94] hat daraus folgende Aspekte extrahiert, die in der Wissenschaftsgemeinde als gesicherte Erkenntnisse akzeptiert sind und die für die Evolutionären Algorithmen Bedeutung haben (Zusammenfassung):

- **Differenzierung von Genotyp und Phänotyp**
  Es wird zwischen der Erbinformation (Genotyp), die in den Chromosomen als genetischer Code hinterlegt ist, und der Erscheinungsform des dazugehörigen Lebewesens, dem Phänotyp, unterschieden. Der Phänotyp läßt nicht unbedingt einen Rückschluß auf den Genotyp zu. Bei den diploiden Lebewesen, zu denen die meisten höheren Arten zählen, liegen alle Chromosome doppelt vor und es kann zu mutationsbedingten Unterschieden in den Ausprägungen (Allelen) der beiden Gene kommen. Ein Gen ist ein zusam-

menhängender Teil des Chromosoms unterschiedlicher Länge, der für eine (oder mehrere) phänotypische Eigenschaften verantwortlich ist. Welches Gen sich dabei auf den Phänotyp auswirkt, hängt von seiner Dominanz bzw. Rezessivität ab. Das dominante Allel prägt den Phänotyp. Nur wenn beide Allelvarianten rezessiv sind, wirken sie sich auch phänotypisch aus. Neben diesen klaren Regeln gibt es noch die unvollständige Dominanz, bei der das dominante Allel lediglich einen stärkeren Einfluß auf den Phänotyp hat, und die intermediäre Vererbung, bei der beide Allele eine Art Kompromiß schließen. Erschwerend für Rückschlüsse vom Phäno- auf den Genotyp kommen noch die Polygenie und die Pleiotropie hinzu. Es wird von Polygenie gesprochen, wenn mehrere Gene ein phänotypisches Merkmal beeinflussen und von Pleiotropie, wenn umgekehrt ein Gen für mehrere phänotypische Aspekte verantwortlich ist.

- **Mutation und Selektion als Evolutionsfaktoren**
  Mutationen sind spontane Veränderungen am Erbmaterial, die phänotypische ungerichtete Variationen eines Grundtyps hervorrufen. Die Selektion bewirkt nun, daß die gemessen an den Umweltbedingungen bestangepaßten Individuen überleben, in dem Sinne, daß sie in der Lage sind, Nachkommen zu erzeugen und für deren anfängliches Überleben zu sorgen. Sie haben damit eine größere Chance, ihre Erbinformationen an die nächste Generation weiterzugeben. Während das Genmaterial aus der Sicht eines Individuums fixiert ist, kann sich der Genpool einer Population durchaus ändern und zu einem gegebenen Zeitpunkt auch unterschiedlich sein. Dies bestimmt die genetische Varianz innerhalb der Population, die ein wesentlicher Faktor für die Anpassungsfähigkeit einer Art auf sich ändernde Randbedingungen ist. Die Selektion betrifft das einzelne Individuum, den Phänotyp, während die biologische Evolution von der gesamten Population ausgeht.

  Es gibt verschiedene Mutationsformen. Genmutationen betreffen einzelne Gene und die spontane Mutationsrate, mit der ein Gen seinen Allelzustand wechselt, liegt innerhalb einer Generation bei 10^{-4} bis 10^{-7}. Diese Rate kann durch äußere Faktoren, wie ionisierende Strahlung oder die Temperatur variieren. Daneben gibt es Chromosomenmutationen, die die Chromosomenstruktur verändern und Genommutationen, die die Anzahl einzelner Chromosomen oder ganzer Chromosomensätze beeinflussen. Neu auftretende Mutationen sind oft rezessiv und damit phänotypisch nicht wirksam. Sie können allerdings durch Einwirkung sogenannter Modifikatorgene im Laufe der Evolution dominant werden. Mutationen mit größeren phänotypischen Effekten sind viel seltener als sogenannte Kleinmutationen, die nur zu geringfügigen Änderungen am Phänotyp führen. Daher wird heute angenommen, daß sich die Differenzierung der Arten über viele kleine Anpassungen vollzogen hat.

- **Rekombination als Evolutionsfaktor**
  Bei Lebewesen mit geschlechtlicher Fortpflanzung erfolgt bei der Befruchtung die Vereinigung der haploiden Keimzellen (einfacher Chromosomensatz) zur diploiden Zygote (befruchtete Eizelle) mit doppeltem Chromosomensatz. Bei der Bildung der haploiden Keimzellen entsteht der einfache Chromosomensatz durch weitgehend zufällige Vermischung der elterlichen Chromosomen. Neben dieser Form der Rekombination kommt es bei der Reifeteilung noch zum sogenannten Crossover, bei dem regelmäßig Brücken und Überkreuzungen der homologen Chromosomen (gleichartige Chromosomen, die sich lediglich in ihren Allelen unterscheiden können) entstehen. Dabei können Teilstücke aus-

getauscht werden, so daß sich am Schluß Allele des väterlichen Chromosoms auf dem mütterlichen befinden und umgekehrt. Wenn der Austausch wechselseitig stattfindet, was in der Regel der Fall ist, bleibt die Chromosomenstruktur unverändert. Andererseits entsteht ein sogenanntes illegitimes Crossover zwischen einander nicht entsprechenden Chromosomenabschnitten. Bei diesen Chromosomenmutationen können bei homologen Chromosomen Mittelstücke verlorengehen (Deletion) oder verdoppelt werden (Duplikation). Bei nicht-homologen Chromosomen kann es auch zum Austausch von Endstücken kommen (Translokation). Außerdem gibt es auch Mutationen innerhalb eines Chromosoms, etwa in der Form, daß Mittelstücke in ihrer Reihenfolge umgedreht werden (Inversion). Heute wird allerdings davon ausgegangen, daß größere Änderungen an den Organismen sich über eine Folge kleinerer Anpassungsschritte erklären lassen [Has82]. Andererseits ist bekannt, daß nur ein Bruchteil der Gene und der Aminosäuresequenzen auf einem Chromosom aktiv ist. Daher wird angenommen, daß vor allem die Duplikation quasi als Materiallieferant für die Evolution wirkt, da mutationsbedingte Änderungen am Duplikat keinen Ausfall eventuell notwendiger Gene bewirken und das Duplikat quasi als Experimentierfeld wirken kann [Schö94]. Auf die Chromosomenmutationen wird bei der Darstellung der Mutationsoperatoren von GLEAM in Abschnitt 2.2.3 noch einmal Bezug genommen werden.

Die Rekombination bewirkt eine regelmäßige Durchmischung der Erbinformationen und ist somit ein wesentlicher Faktor für die Anpassung einer Art an veränderte Umweltbedingungen [Has82]. Eine hinreichende genotypische Varianz einer Population vorausgesetzt, kann die Rekombination zu einer vorteilhaften Kombination von Erbanlagen in einem Teil der Nachkommen führen und so eine schnellere Anpassung bewirken, als das durch reine Mutation möglich ist. Die Mutationen liefern das Ausgangsmaterial für die Evolution und sorgen für genotypische Varianz. Die Rekombination formt aus diesem Material besser angepaßte Individuen und ergänzt so die Mutationen. Sie ist somit ein wichtiger Evolutionsfaktor.

- **Weitere Evolutionsfaktoren: Isolation, Gendrift und Migration**
  In großen Populationen kann es unter bestimmten restriktiven Bedingungen zu stagnierenden Allelhäufigkeiten kommen, so daß eine Evolution nicht mehr stattfindet (Hardy-Weinberg-Gesetz). Bisher wurde eine Population als eine abstrakte Menge von Individuen betrachtet, die gleichermaßen miteinander Nachkommen erzeugen können (Panmixie). In der Realität ist das aber nicht so, da die Gesamtpopulation einer Art allein schon auf Grund geographischer Gegebenheiten in mehr oder weniger getrennte Teilgruppen (Demes) zerfällt (Isolation). Diese Demes entwickeln sich dann zunächst unabhängig voneinander weiter und Mutanten haben eine höhere Chance, sich zu behaupten als in großen Populationen. Es kann dabei auch zu einer zufallsbedingten Verschiebung der Allelhäufigkeiten kommen (Gendrift).

  Der Austausch einzelner Individuen (Migration) kann bei etablierten unterschiedlichen Teilpopulationen von Bedeutung für das Evolutionsgeschehen sein, da er neues Genmaterial in das Deme einführt. Dabei können durch die zufallsbedingte Kombination positiver Eigenschaften besser angepaßte Individuen entstehen (evolutionärer Sprung). Tendenziell wirken solche Migrationen aber langfristig auf einen Ausgleich zwischen den Teilpopulationen hin.

2.2 Evolutionäre Algorithmen

Die Evolutionären Algorithmen bilden die Mechanismen der biologischen Evolution in mehr oder weniger abstrakter Weise mit dem Ziel nach, ein allgemein anwendbares Optimierungs- und Suchverfahren zu erhalten. Eine Lösung wird dabei häufig als Wertekette codiert, was dem Chromosom entspricht. Die meisten Algorithmen gehen im Gegensatz zum natürlichen Vorbild von nur einem Chromosom aus. Mutationen und Crossover setzen als genetische Operatoren an diesen Ketten an und manipulieren sie zufallsgesteuert ähnlich wie das natürliche Vorbild. Das Element des Überlebenskampfes findet durch die Bewertung der manipulierten Ketten, die ausschlaggebend für die Akzeptanz eines Nachkommens oder dessen Vernichtung ist, Eingang in die Verfahren. Die Bewertung spielt auch bei der Partnerwahl eine Rolle, was als algorithmisches Pendant zum Balzverhalten bei vielen Tierarten betrachtet werden kann. Die verschiedenen Algorithmenformen, die sich im Laufe der Zeit entwickelt haben, unterscheiden sich vor allem in der Ausgestaltung der Repräsentation des Problems in den Wertketten und in den genetischen Operatoren. Sie benutzen eine vom biologischen Vorbild geprägte Terminologie, deren wichtigste Begriffe in Tabelle 2.1 erläutert werden.

Begriff	Erklärung
Population	Menge von Individuen, die gleichzeitig betrachtet werden und in der Fortpflanzung miteinander interagieren.
Individuum	Eine Lösungsalternative, die alle Werte einer Lösung in geeigneter Struktur enthält. Meist in Form einer Wertekette (Chromosom).
Chromosom	Kette aus Elementen (Genen), die die Werte einer Lösung enthalten. Der Aufbau der Elemente reicht von einfachen Bits (klassischer GA) über reelle Zahlen bis hin zu komplexeren Strukturen. Meist besteht ein Individuum aus einem Chromosom.
Gen	Element eines Chromosoms (Begriff meist nur bei den GAs gebräuchlich).
Allel	Konkreter Wert in der Lösungsrepräsentation (Wertekette) eines Individuums.
Fitness	Lösungsqualität hinsichtlich vorgegebener Zielkriterien. Sie wird durch die Fitnessfunktion berechnet.
Generation	Ein Iterationsschritt des Verfahrens.
Eltern, Elter	Die an der Reproduktion beteiligten Individuen. Häufig werden alle Individuen einer Population im Laufe einer Generation durchgegangen und bilden nacheinander das Elter, das sich einen Partner sucht. Bei manchen EA ersetzen Nachkommen, die für die Nachfolgegeneration selektiert wurden, das Elter und nicht den Partner.
Kinder, Nachkommen, Offsprings	Aus den Eltern erzeugte Individuen. Bei manchen Algorithmen kann auch ein Elter Nachkommen durch reine Mutation erzeugen.
Klon	Identische Kopie eines Individuums
Genetische Operatoren	Mutationsoperatoren zur Veränderung der Allele. Crossover-Operatoren zum Austausch von Informationen zwischen den Eltern.

Tab. 2.1: Wichtige EA-Fachbegriffe, siehe auch die entsprechende VDI-Richtlinie [VDI3550]

Für die weiteren Betrachtungen ist es sinnvoll, den in der Einleitung bereits eingeführten Begriff der *Multimodalität* zu differenzieren, da die einheitliche Charakterisierung von Aufgaben mit einem oder tausenden Suboptima zu unhandlich ist. Mit *schwach multimodal* werden im folgenden Probleme bezeichnet, die weniger als *2n* Suboptima haben, wobei *n* die Anzahl der Dimensionen ist. Dagegen heißen Aufgaben mit mehr als *20n* Suboptima *stark multimodal*.

Wie bereits in der Einleitung erwähnt, können die Evolutionären Algorithmen in die drei großen Algorithmenklassen Evolutionsstrategien (ES), Genetische Algorithmen (GA) und Evolutionäre Programmierung (EP) eingeteilt werden, wobei letztere in vielen Aspekten der Evolutionsstrategie ähnlich ist. Im folgenden werden die beiden Hauptformen, GA und ES, näher vorgestellt, bevor der Evolutionäre Algorithmus GLEAM, der als Grundlage der Untersuchungen dieser Arbeit dient und der ES- und GA-Elemente miteinander verbindet, detailliert behandelt wird.

2.2.1 Klassische Genetische Algorithmen

Die von Holland [Hol75] entwickelten klassischen Genetischen Algorithmen benutzen Bit-Strings zur Darstellung der durch die Evolution zu verändernden Größen. Dabei erfolgt eine Abbildung der einzelnen phänotypischen Größen entsprechend ihrem Wertebereich auf binäre Substrings, die dann zusammen den Bitstring des Individuums darstellen. Zur Bewertung eines Individuums bedarf es einer entsprechenden inversen Abbildung, die den String aufteilt und die Teilstücke in die zugehörigen Allelwerte rückabbildet. Die Abbildungen werden *Codierung* und *Decodierung* genannt. Damit können logische, ganzzahlige und reellwertige Parameter des Phänotyps behandelt werden. Die genetischen Operatoren kennen den phänotypischen Zusammenhang nicht und operieren in diesem Sinne blind über den Bitketten. Das Crossover ist beim klassischen GA der Hauptoperator, der mit einer bestimmten vorgegebenen Wahrscheinlichkeit P_c (typischer Wert $P_c > 0.6$) ausgeführt wird und an einem gleichverteilt-zufällig bestimmten Punkt der beiden Eltern-Ketten ansetzt (Ein-Punkt-Crossover). Die beiden Elternketten werden zufällig mit einer Wahrscheinlichkeit ermittelt, die proportional zur Fitness steigt (fitness-proportionale Selektion). Nach dem Crossover findet eine Mutation beider Offsprings in der Form statt, daß jedes Bit mit einer kleinen Wahrscheinlichkeit P_m (z.B. $P_m = 0.001$) invertiert wird. Die Mutation ist dabei ein zweitrangiger Operator, der der Fixierung von Allelwerten entgegenwirken soll [Hol75]. Insgesamt werden soviel Nachkommen erzeugt, wie die Elternpopulation Mitglieder hatte. Der Algorithmus folgt dem in Abb. 1.2 angegebenen Ablauf, wobei die erzeugten Offsprings die Elternpopulation komplett ersetzen. Damit kann das bisher beste Individuum verloren gehen.

Die Codierung in Bitstrings hat mehrere Konsequenzen: Erstens lassen sich allgemeine genetische Operatoren formulieren, die völlig unabhängig von einer konkreten Anwendung implementiert werden können. Der Preis für den Vorteil sind die immer problemspezifisch zu realisierenden Codierungs- und Decodierungsalgorithmen. Bei der Mutation von ganzzahligen und reellwertigen Parametern führt diese Codierung zu relevanten Nachteilen, die sich aus der binären Repräsentation ergibt. So kann der Übergang von einer ganzen Zahl zur nächsten mit einer weitgehenden bis kompletten Änderung der Allele der Zahl einhergehen. Das gilt für alle Paare 2^n-1 und 2^n. Generell unterscheiden sich die meisten Zahlen in mehr als einer Bitposition und die Mutation von nur einem Bit führt zu einer Werteänderung, die in Abhängigkeit von der Position des Bits erheblich sein kann. Somit bedeuten kleine Unterschiede im Genotyp meist große Unterschiede im Phänotyp, was generell nicht wünschenswert ist und zu einer

künstlichen Erschwerung des Suchprozesses führt. Dieser Nachteil wurde bereits früh von Hollstien [Hol71] erkannt, der die Verwendung eines Gray-Codes[1] statt der binären Codierung erfolgreich erprobt hat. Ein weiterer Nachteil der binären oder Gray-Codierung besteht darin, daß sie es nicht ermöglicht, strukturelle Eigenschaften vieler Aufgabenstellungen angemessen wiederzugeben und es damit sehr schwer, wenn nicht unmöglich gemacht wird, problembezogene genetische Operatoren zu verwenden. Das ist aber für viele praktische Anwendungen sinnvoll und kann zu einer erheblichen Steigerung der Performance beitragen. Außerdem kann eine ungünstige Codierung sogar aus einem einfachen unimodalen Problem einen komplexen multimodalen Suchraum der codierten Darstellung erzeugen [Bäc93a].

Die Alternative zur binären oder Gray-Codierung sind Strings aus natürlichen oder reellen Zahlen, über deren erfolgreiche Anwendung Davis [Dav91] vor allem bei hybriden GA-Formen berichtet. Weitere empirische Vergleiche, die die Überlegenheit der reellwertigen Codierung bestätigen, können für exemplarische Beispielanwendungen bei Janikow [Jan91] und Wright [Wri91] gefunden werden.

Der Grund für die weite Verbreitung der binären Codierung liegt in Hollands Schema-Theorem [Hol75] und dem darauf errichteten Theoriegebäude begründet. Es basiert auf dem klassischen GA mit binärer Codierung und generationsweiser Ersetzung der Elternpopulation durch die generierten Nachkommen. Ein Schema kann als ein Ähnlichkeitsmuster verstanden werden, das Strings beschreibt, die an bestimmten Positionen übereinstimmen. Die Ordnung eines Schemas H wird mit $o(H)$ bezeichnet und ist bestimmt durch die Anzahl der fest definierten Bits. Unter seiner definierenden Länge $l(H)$ wird der Abstand zwischen der ersten (p_e) und der letzten fest definierten Position (p_l) im String verstanden. $l(H)$ ist dann die Differenz $l - e$, wobei für Schemata mit weniger als zwei fixierten Positionen $l(H) = 0$ festgelegt wird. Das Schema-Theorem besagt nun, daß beim klassischen GA Schemata mit überproportionaler Fitness, kurzer definierender Länge und niedriger Ordnung größere Chancen haben, sich in den Nachfolgegeneration zu reproduzieren. Als praktische Konsequenz daraus wird empfohlen, inhaltlich zusammengehörige Teile eines Chromosoms auch zusammenhängend und möglichst kompakt zu codieren. Außerdem wird gefolgert, daß, da Codierungen geringer Kardinalität[2] bei gleicher Darstellungsleistung mehr Schemata produzieren können als solche von höherer Kardinalität, ersteren der Vorzug bei der Codierung zu geben sei [Hol75]. Das führt unmittelbar zur Bitrepräsentation als Alphabet minimaler Kardinalität.

Viele Autoren messen dem Schema-Theorem für praktische Anwendungen mit endlichen Populationsgrößen in der herkömmlichen Interpretation eine geringe Bedeutung zu [Gre89, Mühl91, Mic92, Gre93, Har93], da erhebliche Stichprobenfehler hinsichtlich der Fitness von Schemata beobachtet worden sind. Zu dem Problem der Fitnessvarianz bei Vertretern desselben Schemas schreibt Mühlenbein [Mühl91, S.324]: „The estimate of the fitness of a schema is equal to the exact fitness in very simple applications only. (...) But if the estimated fitness is used in the interpretation, then the schema theorem is almost a tautology, only describing proportional selection.“ Grefenstette [Gre93, S.80] bemerkt dazu: „The effect of fitness variance within schemas is that, in populations of realistic size, the observed fitness of a schema may be arbitrarily far from the static average fitness, even in the initial population.“ Michalewicz plädiert für eine reellwertige Codierung und schreibt [Mic92, S95]: „The binary representation traditionally used in genetic algorithms has some drawbacks when applied to multi-

1. Bei Gray-Codes werden die Zahlen so codiert, daß sich benachbarte Werte in nur einem einzelnen Bit unterscheiden.

2. Unter der Kardinalität einer Codierung wird die Anzahl der verwendeten Zeichen verstanden.

dimensional, high-precision numerical problems. For example, for 100 variables with domains in the range [-500, 500] where a precision of six digits after the decimal point is required, the length of the binary solution vector is 3000. This, in turn, generates a search space of about 10^{1000}. For such problems genetic algorithms perform poorly.“ Er vergleicht in seinem Buch die binäre mit der reellwertigen Codierung an Hand einer Reihe von Beispielen und stellt dabei eine zum Teil erhebliche Überlegenheit der reellwertigen Darstellung fest.

Nissen [Nis94, S. 33] faßt die Einwände gegen und die Diskussion um die binäre oder Gray-Codierung folgendermaßen zusammen: „Nach Ansicht des Autors sollten sich die Lösungsrepräsentation und die Codierung möglichst direkt aus der gegebenen Problemstellung ableiten. Insbesondere sollten strukturelle Eigenheiten (Regelmäßigkeiten) des Lösungsraumes durch die Codierung erhalten bleiben, soweit sie den Suchprozeß erleichtern. Vermieden werden müssen dagegen solche Codierungen, die den GA implizit irreführen, so daß global optimale Lösungen nicht gefunden werden können.“

Ausgehend vom klassischen GA hat es neben den unterschiedlichen Codierungsansätzen auch noch hinsichtlich anderer Eigenschaften des Verfahrens eine Vielzahl von Weiterentwicklungen und Modifikationen gegeben, von denen hier nur die wichtigsten wiedergegeben werden können:

- **Selektion**
  Die Auswahl der Eltern kann statt auf der fitness-proportionalen Selektion des Holland’schen GAs auch auf anderen Selektionsmechanismen beruhen, siehe auch [Bäc94]. Ziel solcher Modifikationen ist eine effektive Balance zwischen *exploitation* und *exploration,* zwischen einem angemessenen Selektionsdruck und der Aufrechterhaltung einer ausreichenden Heterogenität der Population. Ein zu hoher Selektionsdruck birgt die Gefahr vorzeitiger Konvergenz auf suboptimale Lösungen in sich (zu große *exploitation*). Ein zu geringer Selektionsdruck erhält dagegen zwar die Lösungsvielfalt in der Population, bewirkt aber ein dem Random Search ähnliches Verhalten des GA, also zuviel *exploration*. Die fitness-proportionale Selektion hat die Tendenz eines zu starken Selektionsdrucks, was sich darin äußert, daß relativ früh auftretende gute Individuen die Population überschwemmen und so das Auffinden des Optimums verhindern. Auch wird berichtet, daß die Lösungen in einem fortgeschrittenen Stadium der Suche die Tendenz haben, sich immer ähnlicher zu werden, so daß der Crossover-Operator an Wirksamkeit verliert. Die wichtigsten Alternativen zur fitness-proportionalen Selektion sind die *rangbasierte* und die auf Brindle [Bri80] zurückgehende *Wettkampf-Selektion*.

  Aus Platzgründen wird hier nur die von Baker [Bak85] vorgeschlagene *rangbasierte Selektion* (*ranking*) vorgestellt. Bei ihr werden die zur Selektion anstehenden n Individuen entsprechend ihrer Fitness sortiert und erhalten dann gemäß ihrem Rang eine feste Selektionswahrscheinlichkeit zugeordnet:

$$P_s(a_i) = \frac{1}{n} \cdot \left(max - (max - min)\frac{i-1}{n-1} \right) \qquad 1 \le i \le n \qquad (2.1)$$

$$\text{wobei: } P_s(a_i) \ge 0 \;,\; \sum_{i=1}^{n} P_s(a_i) = 1$$

mit: $P_s(a_i)$: Selektionswahrscheinlichkeit des Individuums a_i mit Rangplatz i.

max: benutzerdefinierte Obergrenze des Erwartungswerts für die durchschnittliche Nachkommenzahl eines Elters ($1.0 \le max \le 2.0$).

min : Untergrenze des Erwartungswerts für die Nachkommenzahl eines Elters ($min = 2.0 - max$).

Da für die Selektion nun nur noch relative und keine absoluten Fitnesswerte maßgebend sind, werden besonders gute Individuen in ihrer Ausbreitung etwas gebremst und schlechtere erhalten im Gegenzug eine etwas größere Chance zur Reproduktion. Über die Funktion zur Zuordnung der Selektionswahrscheinlichkeiten kann der Selektionsdruck einfach und effektiv über die gesamte Zeit eines GA-Laufs gesteuert werden, ohne zusätzliche Maßnahmen zu erfordern, wie sie von verschiedenen Autoren [DeJ75, Gre89, Gol89] für die fitness-proportionale Selektion vorgeschlagen wurden. Baker [Bak85] erzielte ausgezeichnete Ergebnisse bei Testfunktionen, die bei fitness-proportionaler Selektion zu vorzeitiger Konvergenz führten. Dabei mußte allerdings eine niedrigere Konvergenzgeschwindigkeit in Kauf genommen werden. Auch Hoffmeister und Bäck [Hof92] berichten von sehr guten Ergebnissen mit *ranking* bei multimodalen Funktionen und von schlechten bei unimodalen.

- **Akzeptanzregel**:
  Beim klassischen GA werden alle Eltern durch ihre Nachkommen ersetzt (*generational replacement*). Auf Syswerda [Sys89], Whitley [Whi88] und Davis [Dav91] geht ein anderes Akzeptanzverfahren zurück, bei dem nur wenige Nachkommen in die Nachfolgegeneration übernommen werden (*Steady-State GA*). Die Anzahl ersetzter Individuen wird zum Strategieparameter des GAs (meistens ein oder zwei pro Generation). Es werden die jeweils schlechtesten Individuen durch Kinder ersetzt, die sich von allen Individuen der Elterngeneration unterscheiden müssen (Vermeidung von Duplikaten).

  Die Akzeptanzregel des Steady-State GAs ist damit nicht nur elitär, da das beste Individuum immer überlebt, sie fördert auch den Erhalt der Lösungsvielfalt innerhalb einer Population. Syswerda [Sys91, S.100] berichtet dazu: „ ... informal testing and comparison of the two approaches indicate that at least for some problems, steady-state GAs do find as good or better solutions in much less time than generational GAs.“ De Jong [DeJ93] weist auf zwei Nachteile hin: Da die Gefahr, Allele durch Zufallseinflüsse zu verlieren größer ist als beim *generational replacement*, sind größere Populationen notwendig. Außerdem kann sich die Performance verschiedener GA-Läufe zufallsbedingt beträchtlich unterscheiden. Insgesamt hat jedoch das Steady-State Konzept als fester Bestandteil des erfolgreichen GENITOR-Algorithmus' (GENetic ImplemenTOR) von Whitley und Kauth [Whi88, Whi89] zur breiteren Anwendung der GAs beigetragen. GENITOR benutzt auch die rangbasierte statt der fitness-proportionalen Selektion, womit sich durch eine geeignete Zuordnung der Selektionswahrscheinlichkeiten zu den Rangpositionen der Selektionsdruck leicht und gezielt variieren läßt. Whitley [Whi89] berichtet von einer rascheren und besseren Fokussierung der Optimierung auf erfolgversprechende Regionen des Suchraums im Vergleich zu den traditionellen GA-Formen.

- **Strategieparameter**
  Zu den Strategieparametern zählen unter anderem die Populationsgröße, Mutations- und Crossover-Raten und eventuelle Parameter für Selektions- und Akzeptanzverfahren. Für die Populationsgröße gilt, daß grundsätzlich abzuwägen ist zwischen der Gefahr vorzeitiger Konvergenz auf ein Suboptimum bei zu kleinen Populationen und unnötig hohem Rechenaufwand bei zu umfangreichen Populationen. Häufig wird mit Größen zwischen 30 und 200 gearbeitet, was empirisch gewonnenen Empfehlungen von Grefenstette [Gre86] in etwa entspricht. Bei komplexen Anwendungen und Parallelimplementierun-

gen [Gor90] werden auch erheblich größere Populationen benutzt. Hinsichtlich der Mutationsraten gibt es eine Vielzahl von Veröffentlichungen, die sich allerdings meist auf binäre Repräsentationen beziehen und die auch nur unter dieser Voraussetzung gültig sind [Scha89b, Mühl92, Bäc93a]. Typische Werte liegen zwischen 0.001 [DeJ75] und 0.01 [Gre86] pro Bit.

Vor allem bei nicht-binären Codierungen werden neben den bisher behandelten auch andere Mutations- und z.T. auch Rekombinationsoperatoren benutzt. Spielen z.B. kombinatorische Aspekte bei einer Aufgabenstellung eine Rolle, werden häufig ganzzahlige Codierungen benutzt und Permutations-Mutationen benötigt, die die Genreihenfolge auf dem Chromosom verändern. Bei reellwertiger Codierung gibt es z.B. Mutationsvarianten, die entweder einen komplett neuen Wert auswürfeln oder den vorhandenen Wert nur zufallsbedingt gering verändern [Dav91]. Neben dem Ein-Punkt-Crossover wurde auch mit Crossover-Arten experimentiert, die an mehreren Punkten im Chromosom ansetzen (n-Punkt-Crossover). Hinsichtlich der Wirkung gibt es unterschiedliche Aussagen, wobei darüber Einigkeit besteht, daß zu viele Crossover-Punkte keine Verbesserung bringen [DeJ75, Sys89, Esh89, Scha89b]. Da eine erschöpfende Darstellung aller GA-Varianten den Rahmen dieser Arbeit sprengen würde, wird auf die einschlägige Literatur verwiesen. Insbesondere geben Nissen [Nis94] und Bäck, Fogel und Michalewicz [Bäc98] einen guten Überblick.

2.2.2 Evolutionsstrategie

Seit Rechenbergs Veröffentlichung der Evolutionsstrategie 1973 [Rec73] wurde das Verfahren erheblich weiterentwickelt und verfeinert [Fal80, Bor83, Bor92, Rec94]. Daher wird hier auf eine Beschreibung von Schwefel und Bäck zurückgegriffen [Schw81, Schw95, Bäc91, Bäc93b]. Die ES geht von einem reellwertigen Vektor x_i aus n Parametern der betrachteten Optimierungsaufgabe (bei der ES als *Entscheidungsvariable* bezeichnet) und n' Mutationsschrittweiten σ_j $(1 \le n' \le n)$ als Individuum aus. Das Besondere ist nun, daß die Mutationsschrittweiten als Strategieparameter eines Individuums zusammen mit seinen Entscheidungsvariablen der Evolution unterworfen werden. Dadurch findet eine Optimierung auf zwei Ebenen statt, zum einen auf der Problemebene selbst und zusätzlich auf der Ebene der Schrittweitensteuerung. Das Verfahren läuft in fünf Schritten ab, siehe auch Abb. 1.2:

1. **Initialisierung der Startpopulation**
   Sofern kein Vorwissen verfügbar ist, werden die Entscheidungsvariablen der μ Individuen der Startpopulation zufällig erzeugt und die Schrittweiten einheitlich eher zu groß gewählt, um der Gefahr vorzeitiger Konvergenz vorzubeugen.

2. **Partnerwahl**
   Pro Generation werden λ Kinder von jeweils zwei zufällig bestimmten Eltern erzeugt, wobei alle Eltern mit gleicher Wahrscheinlichkeit ausgewählt werden. Bäck und Schwefel [Bäc93b] empfehlen für λ den siebenfachen Wert von μ, der bei stark multimodalen Problemen erhöht werden sollte, um den explorativen Charakter der Suche zu verstärken.

3. **Erzeugung eines Nachkommens**
   Das Kind wird durch Rekombination erzeugt und dann mutiert. Dabei kommen unterschiedliche Rekombinationsverfahren für die Entscheidungsvariablen und die Strategieparameter zum Einsatz. Die Entscheidungsvariablen des Kindes werden zufallsbestimmt von einem der Elternteile kopiert (diskrete Rekombination), während bei den Mutationsschrittweiten eine Durchschnittsbildung der korrespondierenden Elternwerte erfolgt (in-

termediäre Rekombination). Die nachfolgende Mutation des entstandenen Offsprings erfolgt in zwei Stufen: Zuerst werden die Mutationsschrittweiten verändert und dann damit die Entscheidungsvariablen mutiert. Die Mutationsschrittweiten σ_j des Kindes werden mit Hilfe der Log-Normalverteilung wie folgt neu berechnet [Bäc93b]:

$$\sigma'_j = \sigma_j \cdot e^{(\tau \cdot N(0,1) + \tau_j \cdot N_j(0,1))} \tag{2.2}$$

mit: $N(0, 1)$: normalverteilte Zufallsgröße mit Erwartungswert Null und Standardabweichung Eins

$N_j(0, 1)$: für jedes σ'_j neu bestimmte normalverteilte Zufallsgröße mit Erwartungswert Null und Standardabweichung Eins.

Für die beiden Steuerparameter τ und τ_j werden Werte von $(\sqrt{2 \cdot n})^{-1}$ und $(\sqrt{2\sqrt{n}})^{-1}$ empfohlen. Häufig werden jedoch beide auf Eins gesetzt. Die Schrittweiten werden durch den globalen Term $\tau \cdot N(0, 1)$ einheitlich verändert, während der Term $\tau_j \cdot N_j(0, 1)$ individuelle Korrekturen an den einzelnen Schrittweiten gestattet. Die Entscheidungsvariablen x_i werden mit Hilfe normalverteilter Zufallsgrößen mit den Mutationsschrittweiten σ'_j des Kindes als Standardabweichung mutiert:

$$x'_i = x_i + N_j(0, \sigma'_j)$$

mit: $N_j(0, \sigma)$: Für jede Entscheidungsvariable neu bestimmte normalverteilte Zufallsgröße mit Erwartungswert Null und Standardabweichung σ.

Falls $n' < n$ ist, gelten einzelne Mutationsschrittweiten für die Mutation von mehr als einer Entscheidungsvariablen.

4. **Akzeptanzregel**
   Nach Bildung aller λ Nachkommen wird die Elterngeneration nach zwei unterschiedlichen Strategien gebildet. Bei der (μ, λ)-ES ersetzen die μ besten Offsprings die Elterngeneration vollständig. Bei der $(\mu + \lambda)$-ES bilden Eltern und Kinder eine gemeinsame Gruppe, aus der die μ besten Individuen für die Nachfolgegeneration ausgewählt werden (elitäre Strategie).

5. **Abbruchkriterium**
   Nach Schwefel kann das Verfahren beendet werden, wenn sich die Fitnesswerte des besten und schlechtesten Individuums einer Population nur noch um ein vorgegebenes $\varepsilon > 0$ unterscheiden [Schw81, Schw95].

Eine der wesentlichen Besonderheiten der Evolutionsstrategie ist die *adaptive Mutationsschrittweitensteuerung*, zu deren Ausgestaltung Schwefel folgende Anforderungen stellt [Schw81, Schw95]: Kleine Änderungen der Schrittweite sollen häufiger erfolgen als große, wobei der Erwartungswert für den Faktor zur Multiplikation mit der Schrittlänge gemäß Gl. (2.2) bei Eins, also bei keiner Änderung liegen soll. Außerdem sollen Verkleinerungen genauso wahrscheinlich sein wie Vergrößerungen. Die verwendete Log-Normalverteilung zur Mutation der Strategieparameter erfüllt die Anforderungen. Schwefel betont die Notwendigkeit der Rekombination der Mutationsschrittweiten, damit im Lauf der Evolution ein internes Modell günstiger an die Topologie der Fitnessfunktion angepaßter Schrittweiten gebildet werden kann. Er begründet das folgendermaßen: Da die Konvergenzgeschwindigkeit einer ES mit geringerer Anzahl an Entscheidungsvariablen zunimmt, haben Individuen mit einzelnen Muta-

tionsraten nahe Null eine höhere Reproduktionschance. Das führt aber in der Regel zu vorzeitiger Stagnation bei einem Suboptimum, da nur noch ein Unterraum des ursprünglichen Suchraums betrachtet wird. Rekombination wirkt dem Phänomen entgegen, indem sie tendenziell zu geringe Mutationsraten durch die Durchschnittsbildung wieder erhöht. Schwefel hat diese Überlegung durch praktische Experimente bestätigt. Hoffmeister und Bäck [Hof92] betonen, daß zu geringe Populationsgrößen μ wegen des starken Selektionsdrucks die Variationsbreite der Mutationsraten zu sehr verringern und so eine effektive Selbstadaption verhindern, während zu große Populationen schlechten Strategieparametersätzen eine zu große Überlebenschance einräumen. Die richtige Wahl von μ erfordert somit Erfahrung und ist von der Komplexität der Anwendung, also der Anzahl der Entscheidungsvariablen und der Fitnesstopologie abhängig. Leider ist letztere häufig vorher nicht bekannt. Bäck [Bäc91] begründet die normalverteilte Mutation der Entscheidungsvariablen mit der Beobachtung, daß in der Natur kleine Veränderungen an den Nachkommen häufiger auftreten als große.

Die elitäre Variante der ES weist laut Bäck, Hoffmeister und Schwefel [Bäc91] einige Nachteile auf. Dazu gehört ihre Tendenz zur vorzeitigen Stagnation bei einem Suboptimum und eine verschlechterte Selbstanpassungsfähigkeit der Strategieparameter vor allem bei zu geringen Populationsgrößen. Daher wird von Bäck und Schwefel die Verwendung der (μ, λ)-ES empfohlen [Bäc93b].

Ein weiteres wesentliches Merkmal, das die ES von den Genetischen Algorithmen unterscheidet, ist neben der Schrittweitensteuerung die *auslöschende Selektion*, die sich darin äußert, daß Individuen mit niedrigen Fitnesswerten keine Chance zur Reproduktion haben. Die Fitness eines Offsprings entscheidet lediglich darüber, ob es in die Elterngeneration aufgenommen wird, aber nicht wie bei den GAs darüber, mit welcher Wahrscheinlichkeit es wieviel Nachkommen erzeugt.

Explizite Beschränkungen stellen für die ES kein Problem dar, denn ihre Verletzung wird einfach als Letalmutation behandelt und der Offspring nicht akzeptiert [Schw81, Schw95]. Hingegen verringern Nebenbedingungen, die sich aus mehreren Variablen zusammensetzen und zu verbotenen Bereichen führen, generell die Konvergenzsicherheit der ES hinsichtlich des globalen Optimums, da die Schrittweitenanpassung dazu führen kann, daß verbotene Zonen nicht mehr übersprungen werden können. Eine hilfreiche Maßnahme besteht darin, die Populationsgröße μ deutlich größer zu wählen als die Anzahl derartiger Nebenbedingungen und die Startpopulation möglichst gleichmäßig im Suchraum zu verteilen. Schwefel [Schw81, Schw95] hat bei hochbeschränkten Problemstellungen unter Verwendung eines Verfahrens von Box [Box65] vorgeschlagen, eine Ersatzzielfunktion zu verwenden, die die Anzahl verletzter Nebenbedingungen bewertet, um überhaupt erst einmal gültige Startlösungen zu erhalten.

Bisher wurde die ES nur unter dem Gesichtspunkt der reellwertigen Optimierung betrachtet und ganzzahlige oder kombinatorische Problemstellungen blieben unberücksichtigt. Rudolph [Rud90] hat eine parallelisierte Variante der Standard-ES mit zeitweiser Migration auf das kombinatorische Travelling Salesman Problem (TSP) angewandt. Die Aufgabe[3] besteht darin, die kürzeste Tour zum Besuch einer vorgegebenen Liste von Städten zu bestimmen, wobei jede Stadt nur einmal besucht werden darf. Er benutzt die Indizes der nach der Größe sortierten Liste der Entscheidungsvariablen zur Bestimmung der TSP-Tour, ein durch die notwendigen Sortierungsvorgänge aufwendiges Verfahren. Zu einer dem kombinatorischen Charakter

3. Eine formale Definition dieses klassischen mathematischen Problems kann unter anderem bei Gorges-Schleuter [Gor90] gefunden werden.

der Aufgabenstellung angemesseneren Vorgehensweise kommt Herdy [Her90] durch die Einführung ganzzahliger Entscheidungsvariablen und problemangepaßter Mutationsoperatoren unter Beibehaltung des Mechanismus zur Schrittweitensteuerung. Herdy löst mit seiner ES-Modifikation auch erfolgreich andere ganzzahlige und kombinatorische Aufgabenstellungen wie das Mimikry-Problem, ein magisches Quadrat und Rubik's Cube [Her90].

Born [Bor78] geht bei seinem Beweis für die globale Konvergenz einer (1+1)-ES von einem regulären Optimierungsproblem [Bäc91] aus und zeigt globale Konvergenz bei unendlich vielen Generationen. Borns wie auch andere Konvergenzbeweise [Bäc91, Bäc93b] lassen sich auf $(\mu + \lambda)$-ES übertragen, nicht dagegen auf (μ, λ)-ES, da letztere Verschlechterungen zulassen und das globale Optimum nicht monoton angestrebt wird (vgl. auch den Konvergenzbeweis in Abb. 1.2.2). Da die genannten Beweise nicht nur von einem unendlichen Zeithorizont ausgehen, sondern auch für die ES so wichtige Aspekte wie die Selbstanpassung der Strategieparameter und die Rekombination vernachlässigen, ist ihr praktischer Nutzen eher gering.

2.2.3 GLEAM-Verfahren

Das von Blume entwickelte GLEAM-Verfahren (General Learning Evolutionary Algorithm and Method) [Blu90] geht von einer allgemeinen Repräsentation des Anwendungsproblems in Chromosomen aus. Ein Gen entspricht dabei einer sogenannten *Aktion,* die anwendungsabhängig keinen, einen oder mehrere Parameter haben kann. Die mit Wertebereichsgrenzen versehenen Parameter können ganzzahlig (und damit auch boolesch) oder reellwertig sein. Die Aktionen bilden ähnlich wie die biologischen Gene lineare Listen, die *Aktionsketten* genannt werden. Der Informationsgehalt eines Gens, d.h. seine Länge, ist in der Biologie völlig unterschiedlich. Entsprechend können auch die Aktionen soviel Parameter haben, wie es die Anwendung erfordert. Dazu werden im sogenannten *Aktionsmodell* Aktionstypen definiert, die den Aufbau einer Aktion beschreiben. Es enthält außerdem anwendungsabhängige Vorschriften zur Konstruktion der Ketten aus den Aktionen: Dabei wird zunächst festgelegt, ob jede Aktion genau einmal in der Kette vorkommen muß und die Aktionsketten damit eine feste Länge haben, oder ob die Aktionen nur mit einer bestimmten Wahrscheinlichkeit auftreten und somit Ketten unterschiedlicher Länge entstehen. Die Regeln spielen bei der Ketten-Erzeugung per Zufall eine Rolle und werden bei der Reparatur von durch Mutation oder Rekombination hervorgerufenen Defekten benötigt.

Das Aktionsmodell hat Blume am Beispiel der Bahnplanung für einen sieben-achsigen Industrieroboter durch Ansteuerung der einzelnen Achsen erläutert, dessen Motoren für eine bestimmte Zeit mit einer vorgebbaren Rampe an- und ausgeschaltet werden können [Blu94b]. Es wird hier in leicht veränderter Form für einen Mitsubishi R500 Tischroboter mit fünf rotatorischen Achsen wiedergegeben. Zweckmäßigerweise werden zunächst drei Aktionsgrundtypen definiert: einen zum Anschalten, einen zum Abschalten und einen zum Beibehalten des aktuellen Motorzustands für eine bestimmte Zeit. Die Ketten werden als Anweisungsliste für eine einfache Robotersteuerung interpretiert. Die Aktionen werden sequentiell hintereinander, so wie sie in der Genkette stehen, ausgeführt. Dabei wird jeder Aktion eine bestimmte Zeit zugeordnet, so daß eine getaktete Abarbeitung entsteht. Da es wünschenswert sein kann, mehrere Motoren gleichzeitig zu starten oder zu bremsen, wird eine weitere Aktionsform benötigt, die mehrere Motoraktionen klammert, damit sie in einem Zeittakt bearbeitet werden. Dazu dienen die beiden Aktionen BLOCK-BEGINN und BLOCK-ENDE. Zusammen mit der Beibehaltungsaktion wurden damit bereits drei Aktionstypen definiert:

1. BLOCK-BEGINN
   Beginn eines Blocks gleichzeitig in einem Takt ausgeführter Anweisungen.
   Keine Parameter.

2. BLOCK-ENDE
   Ende eines Blocks gleichzeitig in einem Takt ausgeführter Anweisungen.
   Keine Parameter.

3. UNVERÄNDERT
   Alle Motorvorgaben bleiben die nächsten *n* Takte unverändert.
   Parameter: Taktanzahl (ganzzahlig).

Die Motoraktionen können nun so codiert werden, daß jedem Motor ein eigener Aktionstyp jeweils zum An- und Ausschalten zugeordnet wird. Alternativ dazu kann die Motornummer als ein weiterer Parameter einer allgemeinen Ein- bzw. Ausschaltaktion dienen. Aus rein genotypischer Sicht ist der einfacheren Codierung in zwei allgemeinen Aktionen der Vorzug zu geben. Wenn jedoch bedacht wird, welche Auswirkung die Änderung einer Motornummer auf die resultierende Bewegung hat, so wird klar, das bei einer solchen Codierung Parameter mit höchst unterschiedlicher Empfindlichkeit entstehen: So hat beispielsweise die Änderung der Motorspannung um 20% des Wertebereichs eine wesentlich geringere Auswirkung, als die Wahl eines anderen Motors, was ebenfalls einer Werteänderung um 20% des Wertebereichs entsprechen kann. Da es wünschenswert ist, daß kleine Änderungen durch Parametermutationen auch nur kleine phänotypische Wirkungen haben, ist der Codierung in getrennten Aktionen der Vorzug zu geben. An dem Beispiel wird auch deutlich, daß es bei der Wahl der Repräsentation Freiheitsgrade gibt, deren Ausgestaltung große Auswirkung auf die Sensibilität den Genmaterials hinsichtlich kleiner Änderungen haben kann. Eine „geeignete“ Codierung für ein konkretes Problem zu finden, basiert damit auch auf Erfahrung und dem Verständnis für evolutionäre Abläufe. Die Motoraktionen sehen also folgendermaßen aus:

4. MOTOR-x-AN
   Jeder Motor hat seinen eigenen Aktionstyp. Die Parameter bestimmen die Geschwindigkeit und die Rampe, mit der die neue Geschwindigkeit erreicht wird.
   Parameter: Geschwindigkeit in grad/sec und Beschleunigung in grad/sec^2 (beide reellwertig).

5. MOTOR-x-AUS
   Jeder Motor hat seinen eigenen Aktionstyp. Der Parameter bestimmt die Rampe, mit der die Geschwindigkeit auf Null reduziert wird.
   Parameter: Verzögerung (Rampe) in grad/sec^2 (reellwertig).

Mit den fünf Aktionstypen lassen sich Bewegungsprogramme für einen Industrieroboter auf Achsebene formulieren. Da nicht vorhergesehen werden kann, wieviel Motoraktionen zur Erreichung eines vorgegebenen Bewegungsziels notwendig sind, dürfen die einzelnen Aktionen in der Kette beliebig oft vorkommen. Es entstehen damit *Ketten dynamischer Länge*. Abb. 2.1 zeigt eine kurze Kette als Beispiel [Blu94b]. Die dazugehörigen Geschwindigkeitsdiagramme sind in Abb. 2.2 dargestellt, wobei eine Taktdauer von 500 msec zugrundegelegt wird. Zuerst wird Motor 1 mit einer Geschwindigkeit von zehn grad/sec bei einer Rampe von zwei grad/sec^2 angeschaltet. Nach fünf Sekunden oder zehn Takten hat er seine Endgeschwindigkeit erreicht. Wegen der ersten UNVERÄNDERT-Aktion werden erst bei Takt 32 die vier nachfolgenden Aktionen, die einen Block bilden, in einem Takt ausgewertet. Sie bewirken ein gleich-

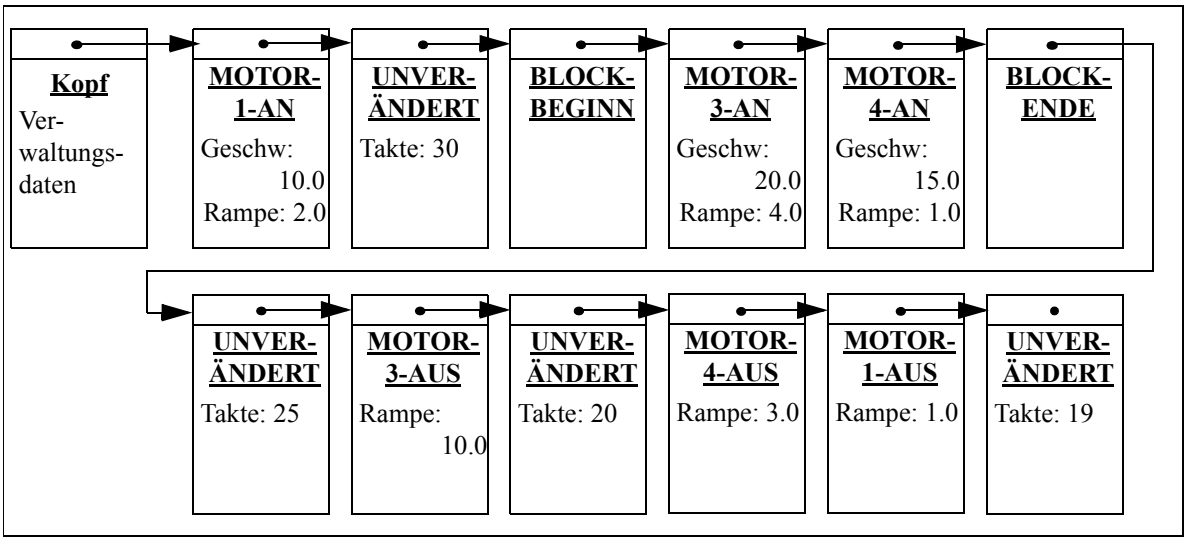


Abb. 2.1: Kleines Aktionskettenbeispiel

zeitiges Einschalten der Motoren 3 und 4. Nach fünf Sekunden (zehn Takten) hat Motor 3 seine vorgegebene Geschwindigkeit bei Takt 41 erreicht, die er wegen der zweiten UNVERÄNDERT-Aktion für weitere 16 Takte beibehalten wird. Motor 4 erreicht seine Endgeschwindigkeit wegen der geringeren Beschleunigung erst bei Takt 61. Bei Takt 58 bewirkt die achte Aktion das Abbremsen von Motor 3, der wegen der Verzögerung von zehn grad/sec^2 nach zwei Sekunden oder vier Takten zur Ruhe kommt. Die dritte UNVERÄNDERT-Aktion verschiebt das Abschalten der beiden verbleibenden Motoren auf die Takte 79 (Motor 4) und 80 (Motor 1). Nach 19 weiteren Takten kommt bei Takt 99 auch Motor 1 zur Ruhe. Wenn, was bei evolutionierten Ketten häufig vorkommt, bei Erreichen des Aktionskettenendes noch Motoren angeschaltet sind, werden sie mit einer fest voreingestellten Rampe heruntergefahren, so daß die Bewegung immer zu einem definierten Ende kommt.

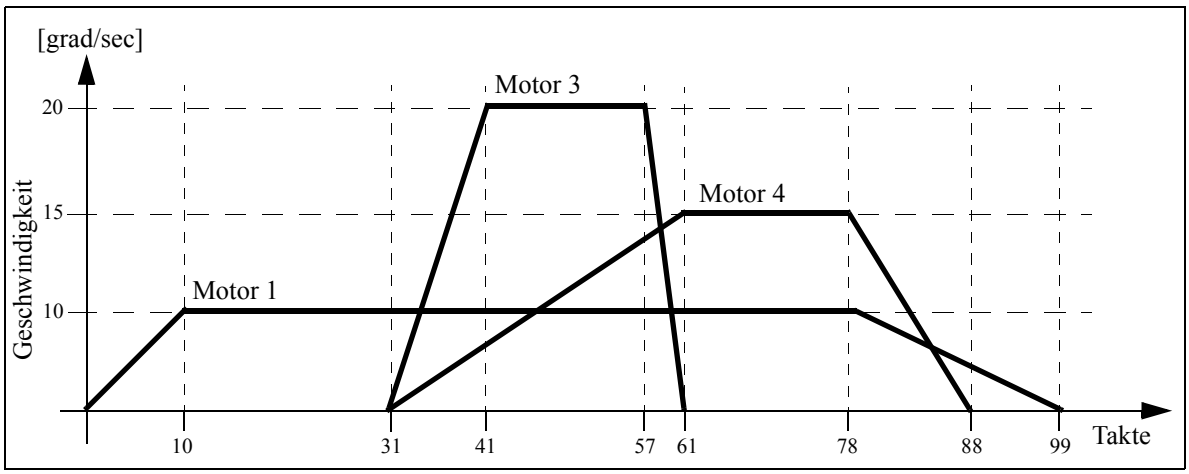


Abb. 2.2: Gechwindigkeitsdiagramme zur Aktionskette von Abb. 2.1 (nach [Blu94b])

Die resultierende Bahn für den 5-achsigen Mitsubishi R500 Tischroboter ist als Simulation in Abb. 2.3 dargestellt. Der Roboter ist zunächst in seiner Grundstellung, bei der alle Achsen ihre Nullposition einnehmen (linkes Teilbild). Bei der Bewegung wurden die Achsen 1, 3 und 4 verändert, siehe rechtes Teilbild. Die resultierende Bahn des Greifpunkts zwischen den Greifbacken, des sogenannten *tool center points*, ist ebenfalls dargestellt.

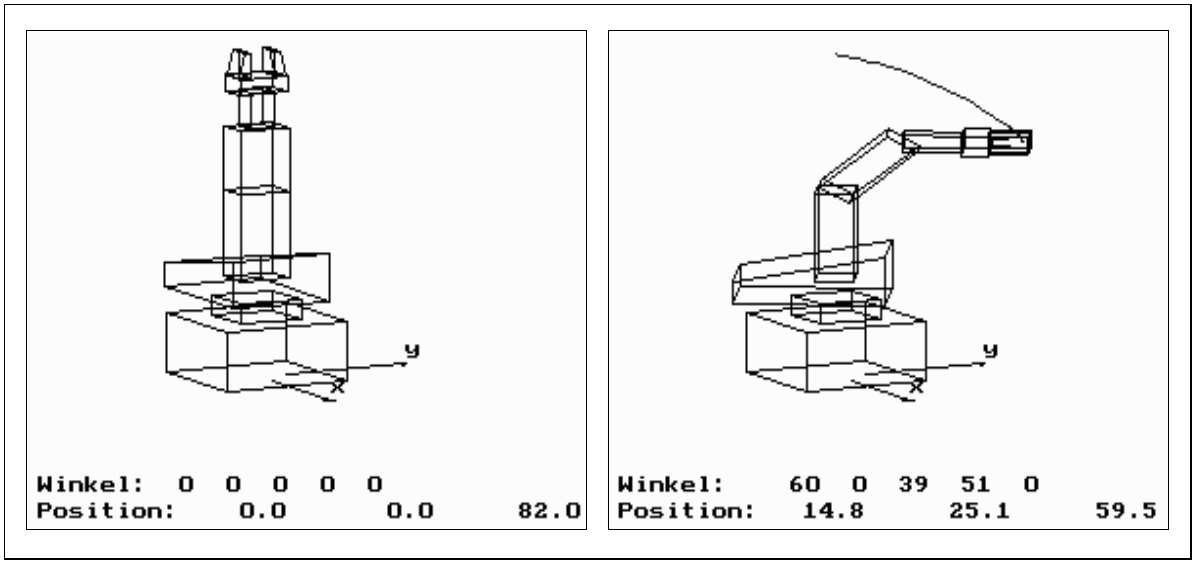


Abb. 2.3: Mitsubishi R500 Tischroboter vor (links) und nach (rechts) der simulierten Ausführung der Aktionskette von Abb. 2.1

Das vorgestellte Handlungsmodell für Industrieroboter wurde von Blume mit leichten Abwandlungen und erweitert um die Orientierung als weiteres Bewertungskriterium sowie die Ansteuerung von Zwischenpunkten erfolgreich für industrielle Bahnplanungsaufgaben mit Kollisionsvermeidung an Hindernissen bei ABB- und Kuka-Robotern angewandt [Blu97, Blu00a]. Als Vorteil des Einsatzes von GLEAM gegenüber konventionellen Bahnplanungsalgorithmen gibt Blume die resultierenden harmonischen Bewegungsabläufe an, die den Verschleiß an den Robotergelenken verringern und damit auch weniger Energie benötigen [Blu98]. Bei einer verwandten Anwendung ging es um einen Portalroboter, der auf einer Palette Schalungsteile für die Herstellung von Betonplatten auslegt und Markierungen für manuell nachzurüstende Schalungen aufsprüht. Das Ziel bestand in der Minimierung der durch Leerfahrten zwischen den Arbeitsschritten zurückgelegten Strecke durch eine geeignete Reihenfolge der Arbeitsschritte. GLEAM wurde hier in Verbindung mit einer einfachen Heuristik eingesetzt. Dabei konnten einzelne Aufträge mit maximal 229 Leerfahrten auf bis zu 2/3 des von der konventionellen Arbeitsplanung ermittelten Leerwegs unter Einhaltung vorgegebener Planungszeiten reduziert werden [Blu00b].

Bei Aufgabenstellungen aus anderen Gebieten, wie zum Beispiel der reinen Parameteroptimierung, kann bei fester Parameteranzahl dagegen ein Aktionsmodell mit Ketten fester Länge benutzt werden. Ob dabei jedem Parameter genau eine Aktion zugeordnet wird oder aber einige Parameter zu einer Aktion zusammengefaßt werden, ist ein Freiheitsgrad, der je nach Aufgabenstellung und Zielsetzung unterschiedlich gestaltet werden kann, wie auch nachstehende Überlegungen zeigen.

Die Zweckmäßigkeit einer Codierung hängt auch mit den verwendeten genetischen Operatoren zusammen. Wie bereits in Abschnitt 2.2.1 erwähnt, kommen die klassischen GAs auf Grund der universellen Bitrepräsentation mit allgemeinen anwendungsneutralen genetischen Operatoren aus, was insofern ein Vorteil ist, als der Kern der genetischen Maschine anwendungsneutral implementiert werden kann. Der offensichtliche Nachteil besteht darin, daß problembezogenes Wissen vom Kernprozeß der Evolution ausgeschlossen bleibt. Das hier vorgestellte Modell der Repräsentation in Aktionsketten erlaubt dagegen die Formulierung eines Satzes neutraler genetischer Operatoren, die unter Einhaltung der im Aktionsmodell hinter-

legten Informationen wie z.B. Wertebereichsgrenzen an die Aufgabenstellung angepaßte Mutationen durchführen können. Außerdem kann durch geeignete Zusammenfassung von Parametern in einer Aktion erreicht werden, daß sie nicht durch Rekombinationsoperatoren aufgespaltet werden können. Letzteres ist ein häufiges Problem bei den klassischen GAs. Zusätzlich können anwendungsspezifische Operatoren implementiert werden, da die Aktionsketten in Verbindung mit dem Aktionsmodell genügend problembezogene phänotypische Informationen beinhalten. Das in GLEAM benutzte Aktionsmodell erlaubt zusammen mit den vordefinierten genetischen Operatoren eine flexible Abbildung unterschiedlichster Aufgabenstellungen auf eine festimplementierte genetische Maschine ohne die Option zusätzlicher aufgabenbezogener Operatoren aufzugeben. Blume beschreibt das folgendermaßen, wobei er die Aktionsketten mit *plan* bezeichnet [Blu00a, S.328]: „In particular GLEAM generates a sequence of basic actions or statements, which are the elements of the plan, which is a member of the evolution population. A plan represents directly the genetic information. The purpose of the plan is not of interest for the evolution itself, therefore the kernel of GLEAM including the evolution algorithm was applied to different problems with minor changes. For example, these actions can be the basic commands of a simple robot controller or allocation steps to reserve a machine in a production plan.“

GLEAM enthält einen Satz fest implementierter genetischer Operatoren, die in vier große Gruppen eingeteilt werden können: Segmentgrenzen-, Parameter- und Aktionsmutationen sowie Crossover-Operatoren [Blu90]. Applikationsspezifische Operatoren können über eine Standardschnittstelle leicht integriert werden. Alle Individuen einer Population werden der Reihe nach zum Elter und die Auswahl des Partners für das Crossover erfolgt durch rangbasierte Selektion gemäß Gl. (2.1) innerhalb einer vorgegebenen Nachbarschaft des Elter (Deme, vgl. auch Abschn. 2.1). Das den Demes zugrundeliegende Populationsmodell wird weiter unten beschrieben.

- Segmentgrenzenmutationen
  Die Aktionsketten enthalten als logische Metastruktur eine Segmentierung des Chromosoms, die phänotypisch nicht relevant ist, aber Einfluß auf die meisten genetischen Operatoren hat. Sie ist durch die biologische Rekombination der Chromosome und durch die Chromosomenmutationen motiviert und soll die Existenz einer über der Genstruktur liegenden Ordnung nachbilden [Blu02]. Die Segmentstruktur ist ihrerseits der Evolution unterworfen; dem dienen die Mutationen der Segmente und Segmentgrenzen: Segmente können geteilt oder zusammengefaßt und Segmentgrenzen verschoben werden.

- Parametermutationen
  Parametermutationen verändern den Wert eines oder mehrerer Parameter einer Aktion innerhalb der durch das Handlungsmodell vorgegebenen expliziten Beschränkungen. Entweder wird der Wert neu ausgewürfelt oder es findet eine Veränderung ausgehend vom aktuellen Wert statt, wobei zuerst gleichverteilt entschieden wird, ob verkleinert oder vergrößert wird. Danach wird das Änderungspotential als Betrag der Differenz zwischen dem aktuellen Wert und der betroffenen Wertebereichsgrenze bestimmt und in zehn Klassen eingeteilt. Die erste Klasse umfaßt Änderungen bis zu 10% des Änderungspotentials, die zweite bis zu 20% usw. Nach zufällig gleichverteilter Bestimmung der Klasse wird der Änderungsbetrag ausgewürfelt. Die sich daraus ergebende Verteilung der Änderungswahrscheinlichkeiten, zeigt Abb. 2.4. Kleinere Beträge sind dabei wesentlich wahrscheinlicher als größere. Die Änderungsmutation führt zu einem ähnlichen Resultat wie die Mutation der Entscheidungsvariablen bei der ES, hat aber dem ge-

genüber den Vorteil einer wesentlich schnelleren Berechnung, da auf die Bestimmung der Normalverteilung verzichtet wird. Ein weiterer Mutationsoperator ist für kleine relative Änderungen zuständig. Er arbeitet ebenfalls mit der in Abb. 2.4 dargestellten Verteilung, nutzt aber nur einen Bereich von 1% des Änderungspotentials. Neben den Mutationen zur Veränderung der Parameter *einer* Aktion gibt es noch segmentbezogene Mutationen, die *alle* Aktionen eines Segments in der zuvor beschriebenen Weise mutieren.

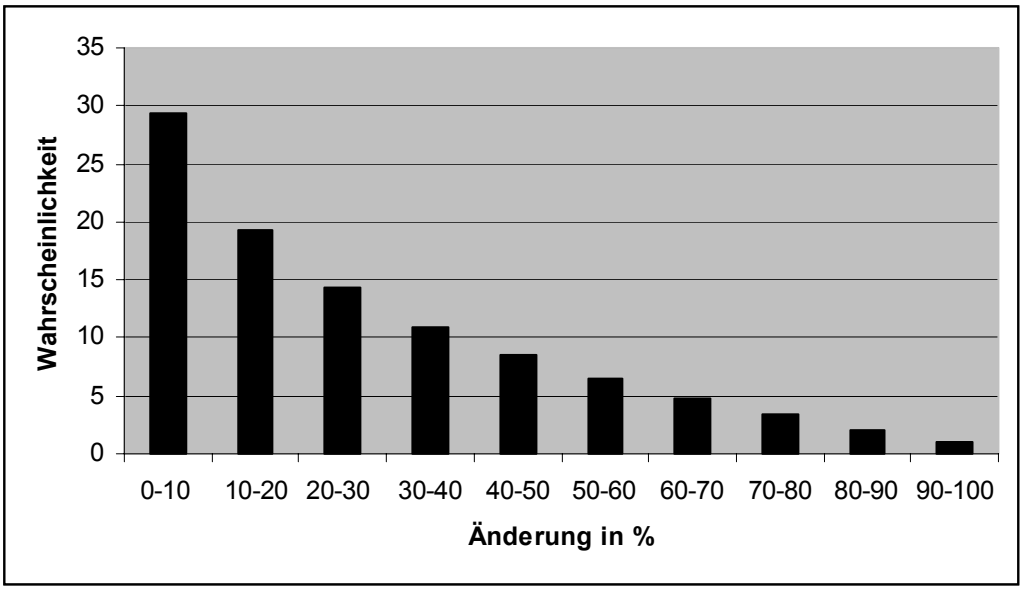


Abb. 2.4: Wahrscheinlichkeitsklassen für Veränderung eines Parameterwerts in Prozent des Änderungspotentials

- Aktionsmutationen
  Aktionsmutationen verändern die Anzahl und/oder die Reihenfolge der Aktionen innerhalb einer Kette. Sie sind nur für Anwendungen relevant, bei denen kombinatorische Aspekte eine Rolle spielen oder für deren Lösung Aktionsketten dynamischer Länge benötigt werden. Auf die Änderung der Aktionsreihenfolge beschränkt sind die Mutationen zur Verschiebung und zum Austausch von Aktionen oder Segmenten sowie die Inversion, die die Reihenfolge der Aktionen innerhalb eines Segments umkehrt. Reihenfolge und Länge verändern dagegen das Einfügen, Löschen und Verdoppeln von Aktionen oder Segmenten. Durch Löschen und Verschieben können Segmente ihre letzte Aktion verlieren, wodurch sie ebenfalls gelöscht werden und indirekt die Anzahl der Segmente verringert wird. Ein Löschen wird nur bei Ketten mit mindestens zwei Aktionen oder Segmenten ausgeführt, um das Entstehen leerer Ketten zu verhindern. Insbesondere die Segmentmutationen sind durch die Chromosomenmutationen des biologischen Vorbilds motiviert.

- Crossover-Operatoren
  Im GLEAM-Standard sind drei Crossover-Operatoren implementiert: das konventionelle Ein-Punkt- und ein n-Punkt-Crossover sowie ein Operator zum Austausch eines ganzen Segments. Bei den beiden ersteren befinden sich die Crossover-Punkte immer auf zufällig ausgewählten Segmentgrenzen.

Die Wahrscheinlichkeit, ob und wie oft ein Operator auf eine Kette angewandt wird, kann in Abhängigkeit von der Kettenlänge und der Fitness des Elter durch eine Steuerdatei für die

Evolution parametriert werden. Darüberhinaus können beliebige Mutationen zu einer hintereinander auf den gleichen Offspring angewandten Operatorsequenz zusammengefaßt werden, wobei jedem Operator eine eigene Ausführungswahrscheinlichkeit zugeordnet wird. Jede Sequenz erzeugt einen (Mutationen) oder zwei Offsprings (Crossover). Damit entstehen pro Paarung in der Regel mehr als ein Offspring, was GLEAM von nahezu allen anderen EA unterscheidet. Nur der beste Nachkomme einer Paarung konkurriert mit seinem Elter gemäß nachstehenden alternativen *Akzeptanzregeln* um den Fortbestand in der nächsten Generation.

- Akzeptiere alle
  Der beste Nachkomme ersetzt immer sein Elter. Bei der elitären Variante geschieht das nur, wenn entweder das Elter nicht das beste Individuum des Demes ist oder aber der Nachkomme eine bessere Bewertung als das beste Individuum des Demes hat.

- Lokal Schlechtestes
  Der beste Nachkomme ersetzt sein Elter nur, wenn er besser als das schlechteste Individuum des Demes ist. Auch hier gibt es eine elitäre Variante, die der Akzeptiere-alle-Regel entspricht.

- Elter-Verbesserung
  Der beste Nachkomme ersetzt sein Elter nur, wenn er besser als das Elter ist. Die Regel ist immer elitär.

Die besten Ergebnisse wurden bei der Roboteranwendung mit der Elter-Verbesserung- und der elitären Variante der Lokal-Schlechtestes-Strategie erzielt [Jak92].

Entsprechend dem natürlichen Vorbild kann auch in GLEAM ein Individuum seinen Partner nicht frei aus der gesamten Population wählen, sondern nur aus seiner Nachbarschaft, die z.B. geographisch definiert werden kann, siehe auch Abschn. 2.1. Das implementierte Nachbarschaftsmodell geht auf Gorges-Schleuter [Gor90, Gor94] zurück und verwendet eine ringförmige Topologie, auf der die Individuen linear angeordnet sind. Jedes Individuum hat eine gleichgroße Nachbarschaft zur rechten und zur linken, seinen Deme, siehe Abb. 2.5. Da die Nachbarschaften benachbarter Individuen sich überlappen, findet trotz der Isolation der Reproduktion auf das Deme ein mehr oder weniger langsamer Informationsaustausch über die Demegrenzen hinweg statt. Abb. 2.5 zeigt je zwei Demes der Größe 5 mit minimaler Überlappung (oben) und mit maximaler Überlappung (unten). Die Geschwindigkeit des Informationsflusses innerhalb der Gesamtpopulation und damit des Verhältnisses von *exploration* zu *exploitation* kann neben dem den Selektionsdruck bestimmenden *ranking*-Parameter damit auch über die Demegröße gesteuert werden. Das Konzept wurde von Gorges-Schleuter für multimodale Aufgabenstellungen auch bei der ES mit Erfolg angewandt [Gor98a, Gor99a].

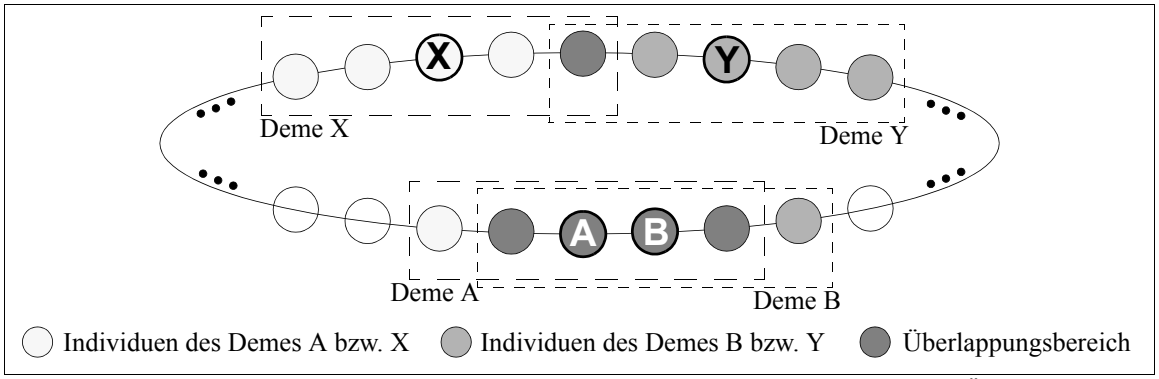


Abb. 2.5: Zwei Beispiele für sich überlappende Demes: oben minimale, unten maximale Überlappung

Partnerwahl und Akzeptanz findet wie zuvor beschrieben nur unter Verwendung von Informationen aus dem Deme statt, weswegen keine zentrale Kontrolle wie bei panmiktischen Populationen[4] notwendig ist. Da bei einer Parallelimplementierung dieses Populationsmodells mit Ausnahme der Initialisierungsphase lediglich ein lokaler Informationsaustausch stattfindet, skaliert die Rechenleistung linear mit der Anzahl der Prozessoren. Gorges-Schleuter [Gor90] gibt dazu folgende Überlegung beruhend auf Amdahls Gesetz [Amd67] an: Die mit *speedup* bezeichnete Beschleunigung durch n Prozessoren bei einer Aufgabe mit einem Anteil s an sequentieller und p an paralleler Bearbeitungszeit beträgt:

$$speedup = \frac{s+p}{s+p/n} = \frac{1}{s+p/n} \qquad \text{da } s+p \text{ normierend auf 1 gesetzt wird.}$$

Der sequentielle Teil s besteht bei einer Parallelisierung basierend auf dem Nachbarschaftsmodell aus der anfänglichen Initialisierung und der Übertragung des Ergebnisses am Schluß. Er ist somit gegenüber p sehr klein und es ergibt sich eine näherungsweise Beschleunigung $speedup_{Nm}$ von

$$speedup_{Nm} \approx \frac{n}{p}$$

In experimentellen Untersuchungen mit bis zu 64 Rechnern in einem Transputercluster konnte die Vorhersage praktisch bestätigt werden [Gor90, Blu93c].

Als Abbruchbedingungen gibt es neben den beiden üblichen Kriterien *Generationsanzahl* und *erreichte Qualität* noch die *verbrauchte Zeit,* die eine unter Umständen vorzeitige Beendigung nach Ablauf einer vorgegebenen Bearbeitungszeit bewirkt. Diese Bedingung ist typisch für Systeme mit realem Einsatzhintergrund, da in der Praxis die Zeit für die Lösung einer Optimierungsaufgabe nahezu immer limitiert ist.

Die Bewertung erfolgt bei Mehrzieloptimierung durch Bildung einer gewichteten Summe aus den zuvor normierten Bewertungen der Einzelkriterien. Die Normierung kann dabei linear, exponentiell oder gemischt linear-exponentiell erfolgen. Sie erfolgt im Bereich von 0 bis 100000, wobei 0 die schlechteste Fitness, die auch als *Note* bezeichnet wird, darstellt. Eine Besonderheit besteht noch in der Priorisierung der Kriterien, die bewirkt, daß Kriterien der nächst niedrigeren Priorität erst zum Zuge kommen, wenn die Kriterien der aktuellen jeweils eine vorgegebene Mindestqualität erreicht haben. Damit kann die Suche anfangs auf die wichtigsten Kriterien beschränkt werden und Nebenziele finden erst bei einer bestimmten Erfüllung der wichtigeren Optimierungsziele Berücksichtigung.

Beschränkungen werden im GLEAM-Standard auf zwei Arten behandelt: Die expliziten Beschränkungen sind bereits durch das eingangs behandelte Handlungsmodell abgedeckt und implizite Beschränkungen können über Straffunktionen berücksichtigt werden. Darüber hinaus stellt GLEAM eine Schnittstelle zur Behandlung von Verletzungen impliziter Beschränkungen zur Verfügung. Entweder wird die Verletzung beseitigt oder das betroffene Individuum wird als Letalmutation verworfen. Beide Maßnahmen sollen unnötige Bewertungen verhindern, die meist mit mehr oder weniger aufwendigen Simulationsläufen verbunden sind.

Zwischen GLEAM einerseits und den klassischen GAs und der ES andererseits gibt es eine Reihe von Gemeinsamkeiten und Unterschieden: GLEAM und die ES setzen auf eine dem

4. Bei einer panmiktischen Population kann jedes Individuum den Partner aus der gesamten Population wählen.

Phänotyp möglichst nahe kommende Codierung und auf die Beachtung des Prinzips der starken Kausalität, wonach kleine Änderungen auch nur kleine Wirkungen verursachen sollen. Die aufwendige Berechnung normalverteilter Mutationen wurde durch die einfacheren klassengesteuerten Änderungswahrscheinlichkeiten (siehe Abb. 2.4) ersetzt, die aber einen ähnlichen Effekt haben. Auf die adaptive Mutationschrittweitensteuerung durch zusätzliche Strategieparameter, die vor allem bei unimodalen Problemen und bei der Annäherung an das Optimum von Vorteil ist, wurde wegen ihrer Tendenz zu suboptimalen Lösungen bei beschränkten oder stark multimodalen Problemen verzichtet. Ein Vergleich der Ähnlichkeit der Codierung der biologischen Chromosomen mit denen der drei Algorithmen ergibt, daß die klassischen GAs auf der Ebene der Basenpaare ansetzen, GLEAM und in bestimmter Weise auch die ES hingegen auf der Ebene der Gene, siehe Abb. 2.6.

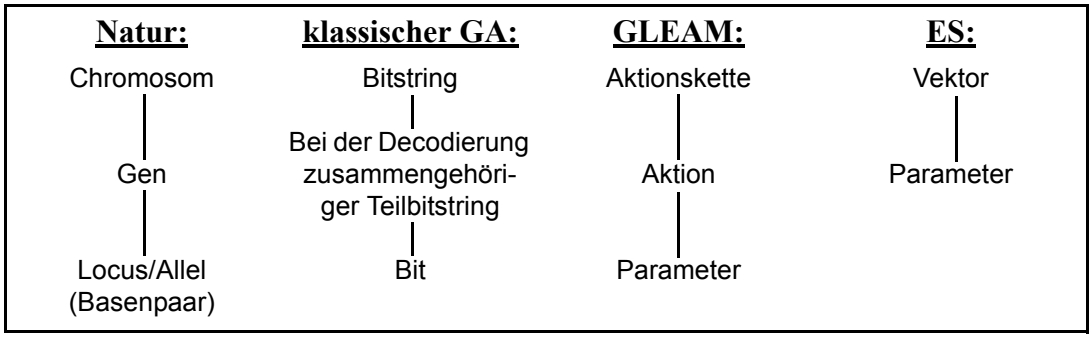


Abb. 2.6: Vergleich der Codierungsebenen

Die Selektions- und Akzeptanzmechanismen von GLEAM entsprechen eher den bei den GAs üblichen Verfahren. Das Nachbarschaftsmodell ist weder in einem der beiden klassischen Ansätze noch im ersten GLEAM-Entwurf [Blu90] enthalten. Es ist vielmehr eine eigenständige Erweiterung des sonst üblichen panmiktischen Populationsmodells. Bei einem Vergleich zwischen GAs und der ES stellen Hoffmeister und Bäck [Hof92, S.23] fest: „ESs are predestined to parameter optimization, while GAs cover a much broader application domain.“ Letzteres trifft ebenfalls auf GLEAM zu, da es ohne Modifikationen auch für kombinatorische oder (gemischt) ganzzahlige Probleme geeignet ist. Weitere kennzeichnende Merkmale von GLEAM [Blu02] sind das allgemeine Aktionsmodell mit seiner Ausrichtung auf die Planung und Optimierung dynamischer Abläufe, die Aktionsketten (Chromosome) dynamischer Länge und die der Evolution unterworfene Segmentierung der Ketten mit den darauf aufbauenden genetischen Operatoren.

2.3 Lokale Suchverfahren zur Kombination mit GLEAM

2.3.1 Rosenbrock-Verfahren

Rosenbrock hat mit seinem Algorithmus die Begrenzung der Pattern-Verfahren, Suchschritte entlang der Koordinatenachsen zu vollziehen, überwunden, indem er entlang der Achsen eines im Raum rotierenden Koordinatenssystems sucht [Ros60]. Restriktionen werden mit einer internen Straffunktion berücksichtigt, die in der Nähe einer Beschränkung wirksam wird. Dazu wird die Zielfunktion *F(x)* ausgehend von Beschränkungen der in Gl. (1.1) angegebenen Form wie folgt modifiziert [Schw95 nach Ros60]:

$$\tilde{F}(x) = F(x) + \sum_{j=1}^{m} \varphi_j(x)(f_j - F(x))$$

Darin sind:

$$\varphi_j(x) = \begin{cases} 0 & \text{wenn } G_j(x) \geq \delta \\ 3\eta - 4\eta^2 + 2\eta^3 & \text{wenn } 0 < G_j(x) < \delta \\ 1 & \text{wenn } G_j(x) \leq 0 \end{cases} \qquad \text{mit } \eta = 1 - \frac{1}{\delta} G_j(x)$$

Dabei ist f_j der Wert der Zielfunktion des letzten erfolgreich ermittelten Punktes, bevor die „verbotene" Region an der j-ten Grenze erreicht wurde. Als sinnvollen Wert für das die Grenzregion bestimmende δ wird 10^{-4} angegeben.

Ausgehend von einem zulässigen Startpunkt, der nicht zu nahe an den Grenzen liegen sollte, werden Suchschritte entlang der Koordinatenachsen unternommen. Ein neuer Punkt gilt als erfolgreich, wenn seine Zielfunktion nicht schlechter ist als die des aktuellen Punktes. In diesem Fall wird er gespeichert und die Schrittweite mit einem Faktor $\alpha > 1$ multipliziert. Andernfalls wird der neue Punkt verworfen, die Schrittlänge verkleinert und die Suchrichtung umgekehrt, indem sie mit einem Faktor $-1 < \beta < 0$ multipliziert wird. Es werden solange Suchschritte versucht, bis für jede Achse mindestens ein Erfolg gefolgt von einem Mißerfolg erreicht wurde. Das Vorgehen garantiert, daß keine Achse des n-dimensionalen Raums verloren geht und stets der Suchraum in allen Dimensionen durchsucht wird. Als sinnvolle Werte gibt Rosenbrock für $\alpha = 3$ und $\beta = -0.5$ an.

Als Abbruchkriterium überwacht Rosenbrock die Länge und die Richtungsänderung des zurückgelegten Weges einer jeden Iteration. Dazu werden die Vektoren a_1 und a_2 in der k-ten Iteration wie folgt berechnet:

$$a_i = \sum_{j=i}^{n} d_j^{(k)} \upsilon_j^{(k)} \qquad \text{für } i = 1, 2$$

Der Skalar $d_j^{(k)}$ repräsentiert dabei die in der Richtung $\upsilon_j^{(k)}$ zurückgelegte Distanz. Wenn die beiden Bedingungen $\left\|a_1^{(k)}\right\| < \varepsilon$ und $\left\|a_2^{(k)}\right\| > 0.3\left\|a_1^{(k)}\right\|$ in sechs hintereinander folgenden Iterationen erfüllt sind, terminiert das Verfahren. Die zweite Bedingung soll einen vorzeitigen Abbruch bei kleinen Schrittweiten verhindern, indem als weiteres Kriterium die Richtungsänderungen herangezogen werden.

Abb. 2.7 gibt einen Überblick über den Ablauf des Rosenbrock-Verfahrens. Eine Iteration besteht aus der großen Schleife zur Durchführung der Suchschritte. Nach der Konstruktion eines neuen Punkts und der Berechnung seiner Zielfunktion wird im Erfolgsfall je nach dem Vorhandensein von Beschränkungen geprüft, ob sie verletzt sind bzw. ob sich der neue Punkt einer Grenze zu weit genähert hat oder nicht. Ersteres führt zur modifizierten Zielfunktion $\tilde{F}$, wie zuvor angegeben. Entsprechend dem Wert der (modifizierten) Zielfunktion und der Einhaltung der Restriktionen wird über Annahme oder Ablehnung des Punktes entschieden und die internen Zähler und Merker entsprechend aktualisiert. Dazu gehört insbesondere die Schrittweite, die, wie zuvor beschrieben, bei Erfolg um den Faktor α vergrößert und bei Mißerfolg um den Faktor β unter Umkehrung der Suchrichtung verkleinert wird. Danach wird in der Kontrollschleife der Suchschritte geprüft, ob für alle Achsen ein Erfolg gefolgt von einem Mißerfolg aufgetreten ist und damit die Iteration beendet werden kann. Andernfalls wird ein

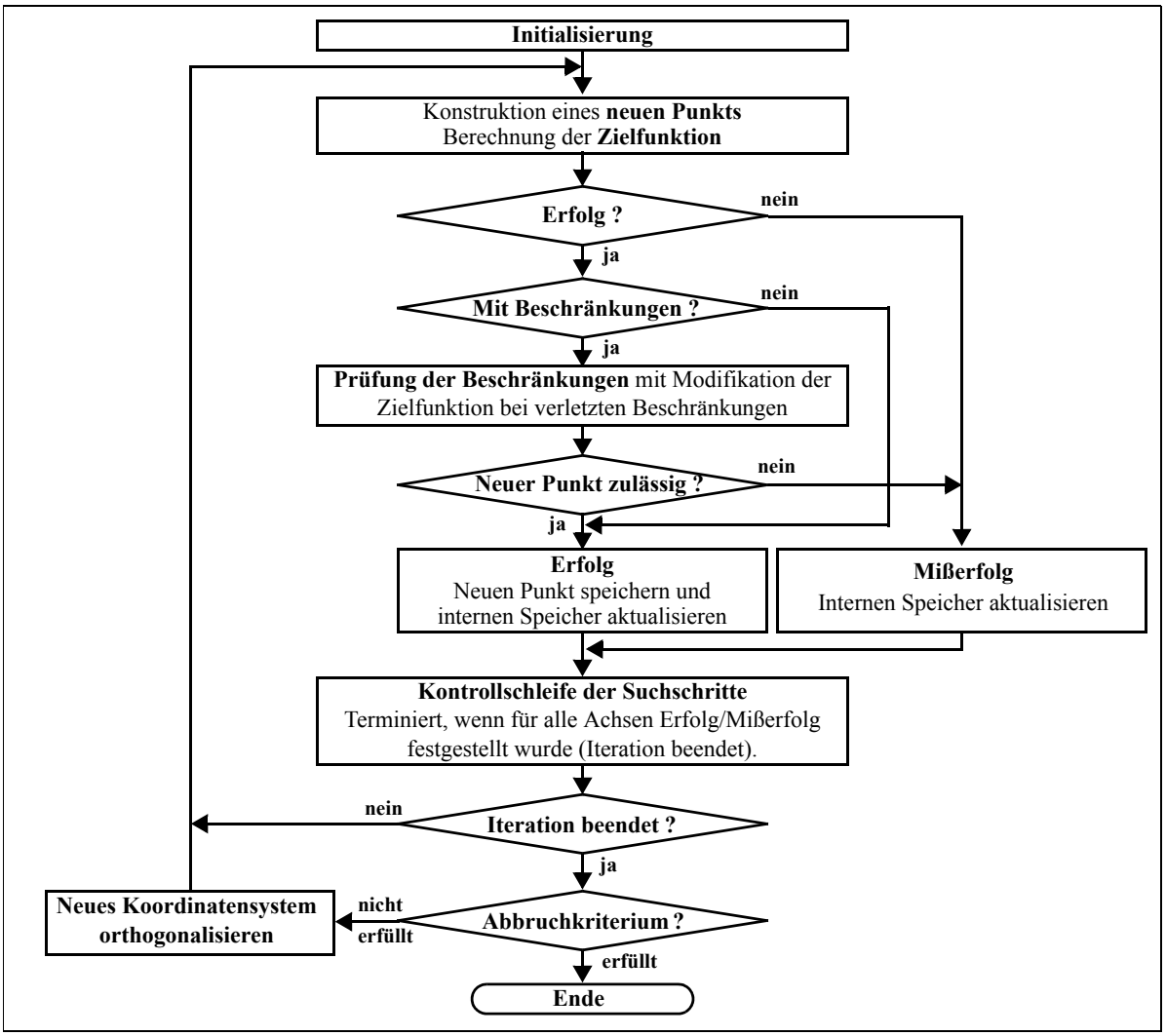


Abb. 2.7: Ablaufschema des Rosenbrock-Verfahrens

neuer Punkt berechnet und der Zyklus innerhalb einer Iteration beginnt von neuem. Bei Beendigung einer Iteration erfolgt die Überprüfung des Abbruchkriteriums, wie zuvor beschrieben. Wenn es nicht erfüllt ist, wird ausgehend von der Richtung zum besten neuen Punkt mit ihm als Ursprung ein neues Koordinatensystem konstruiert und orthogonalisiert.

Das Verfahren gilt unter anderem wegen des Verzichts auf eine Liniensuche als sehr robust. Ein weiterer Vorteil besteht darin, daß es ohne Ableitungen auskommt. Nachteilig ist der hohe Aufwand zur Orthogonalisierung [Schw95], der bei n Dimensionen mit $O(n^3)$ ansteigt, was allerdings nur bei Zielfunktionen, die sich mit entsprechend geringem Aufwand berechnen lassen, zum Tragen kommt. Der Speicheraufwand steigt wegen der Matrizen mit $O(n^2)$ an, was aber bei der Speicherausstattung heutiger Rechner kein ernsthaftes Problem darstellen sollte.

2.3.2 Complex-Algorithmus

Der Complex-Algorithmus von Box [Box 65] modifiziert das Simplex-Verfahren von Nelder und Mead [Nel65] mit seinen Reflexions-, Kontraktions und Expansions-Operationen für beschränkte Aufgabenstellungen. Im Gegensatz zu seinem Vorbild arbeitet er mit mehr Eck-

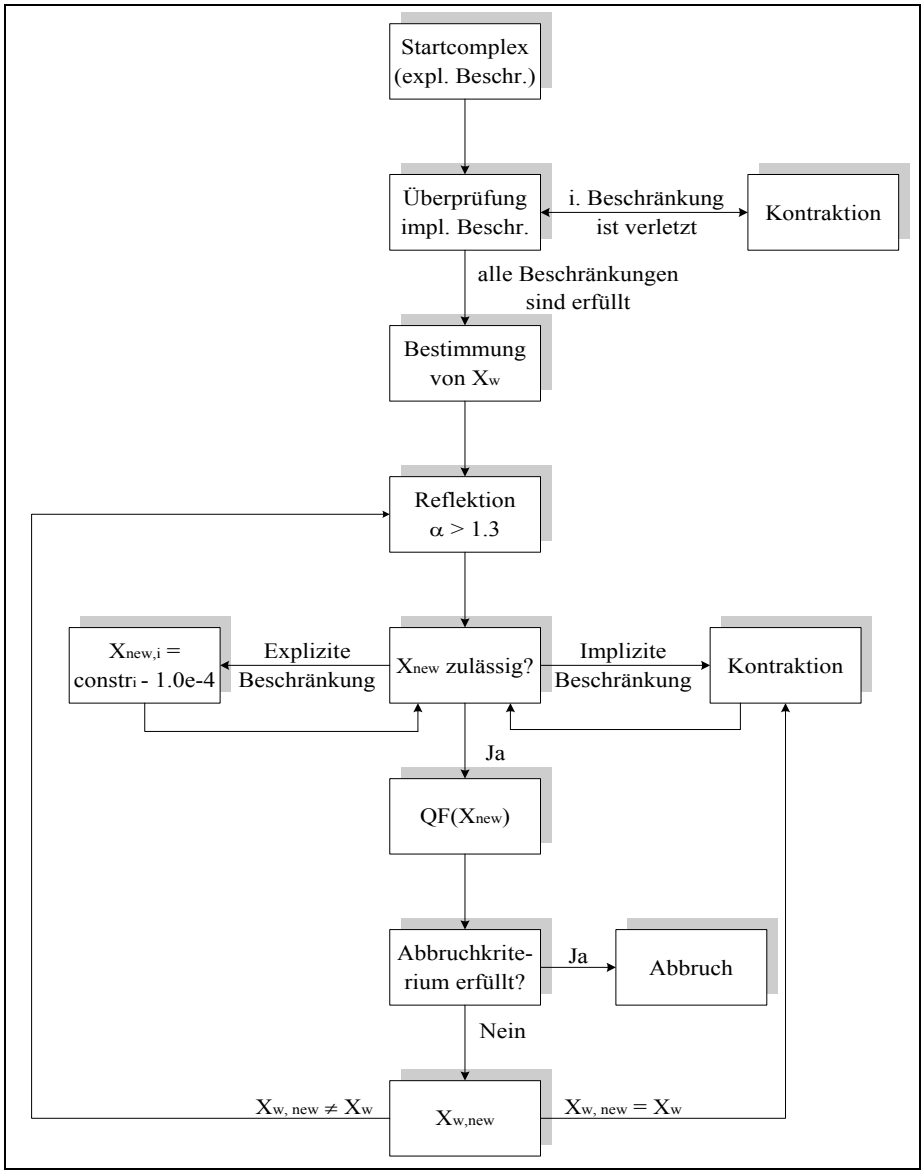


Abb. 2.8: Complex-Algorithmus von Box, aus [Pet99b]

punkten (bei *n* Dimensionen zwischen *n+1* und *2n*) und faßt die Reflexion und Expansion zu einer Operation zusammen. Die Änderungen sollen eine vorzeitige Stagnation des Simplex-Verfahrens vor allem bei aktiven Beschränkungen verhindern. Abb. 2.8 gibt einen Überblick über den Algorithmus. Wenn der Ausgangspunkt der Suche nicht innerhalb des zulässigen Bereichs liegt, wird er solange verschoben, bis alle expliziten Beschränkungen erfüllt sind. Er bildet damit den ersten Punkt des Polyeders, dessen restliche Punkte zufällig innerhalb des Gültigkeitsbereichs gewählt werden. Danach werden die impliziten Beschränkungen geprüft und der Polyeder solange schrittweise verkleinert, bis alle Restriktionen erfüllt sind. Damit ist die Konstruktion des Startpolyeders beendet, alle Eckpunkte werden bewertet und der schlechteste X_w ermittelt. Die Iteration beginnt nun mit einer Reflexion des schlechtesten Punktes (Spiegelung am Flächenschwerpunkt der restlichen Punkte) mit anschließender Expansion um den Faktor $\alpha = 1.3$. Die Prüfung der Beschränkungen beginnt mit den expliziten, deren Verletzung eine schrittweise Veränderung der jeweils betroffenen Vektorkomponente des gespiegelten Punktes bewirkt, bis er wieder innerhalb des zulässigen Bereichs liegt. Die dabei verwendete Schrittweite ist im Bild mit 10^{-4} gewählt. Die Verletzung einer implizi-

ten Beschränkung führt zu einer Kontraktion des Polyeders, indem der neue Punkt in Richtung Flächenschwerpunkt solange verschoben wird, bis alle impliziten Beschränkungen ebenfalls erfüllt sind. Erst wenn keine Beschränkungen mehr verletzt sind, wird der Wert der Zielfunktion des neuen Punktes X_{new} bestimmt.

Wenn X_{new} besser als einer der Eckpunkte mit Ausnahme des schlechtesten X_w ist, wird X_w durch X_{new} ersetzt und die nächste Iteration beginnt wieder mit einer Reflexion (linke Schleife). Andernfalls wird mit der Kontraktion und Prüfung der Beschränkungen fortgefahren, wie in Abb. 2.8 in der rechten Schleife dargestellt. Es gibt zwei Abbruchkriterien: Zum einen wird abgebrochen, wenn fünf mal hintereinander keine Verbesserung eintrat und der Wert der Zielfunktion unverändert blieb und zum anderen, wenn die gleiche implizite Beschränkung fünf mal hintereinander eine unzureichende Kontraktion veranlaßt hat.

Box fand in numerischen Untersuchungen, daß die Anzahl der Eckpunkte des Polyeders und der Expansionsfaktor α keinen signifikanten Einfluß auf die Leistungsfähigkeit des Verfahrens haben [Box 65]. Er gibt auch an, daß ab sechs Parametern eine Anzahl von *2n* Eckpunkten insbesondere bei unbeschränkten Problemen unnötig hoch ist und daß sein Algorithmus keine Verbesserung gegenüber dem Simplex-Verfahren - außer eben Beschränkungen berücksichtigen zu können - bietet. Die Komplexität liegt bei $O(n^2)$ und die Anzahl der Funktionsaufrufe steigt wie beim Simplex-Verfahren mit $n^{2.11}$ an [Nel65, Box65]. Nachteilig ist die Tendenz aller Polyeder-Verfahren, in der Nähe des Optimums schlecht zu konvergieren. Dem steht der Vorteil gegenüber, kleinere Suboptima gut überspringen zu können.

3. Neue Methode zur allgemein anwendbaren Integration lokaler Suchverfahren in Evolutionäre Algorithmen

Die neue Methode zur Integration lokaler Suchverfahren (LSV) in Evolutionäre Algorithmen unter Beibehaltung der allgemeinen Anwendbarkeit des resultierenden Hybrids besteht aus zwei zentralen Punkten:

1. Der Verwendung von lokalen Suchverfahren, die wie der EA allgemein anwendbar sind, also möglichst geringe Voraussetzungen an den Suchraum stellen.
2. Einem neuen Steuerungsverfahren zur geeigneten Aufteilung der Rechenzeitressourcen zwischen dem EA und dem LSV, das ebenfalls allgemein anwendbar ist[1].

Die Methode umfaßt die in Abschnitt 1.2.3 vorgestellten Integrationsarten *Initialisierung der Startpopulation (Voroptimierung)*, *Nachoptimierung der EA-Ergebnisse* und die *direkte Integration*. Die bisweilen auch gebräuchliche und in Abschnitt 1.2.3 vorgestellte *Aufgabenteilung* wird hier nicht weiter berücksichtigt, da die Aufteilung einer Aufgabe immer nur problemspezifisch gelöst werden kann. Die Verwendung anwendungsneutraler lokaler Suchverfahren stellt eine grundsätzliche Abkehr von der bisher weitverbreiteten Praxis dar, bei hybriden EA spezialisierte lokale Suchalgorithmen zu verwenden.

Für die Nachoptimierung wird in Abschnitt 3.2.2 ein neues Steuerungsverfahren vorgestellt, das den Abbruchzeitpunkt für den EA in Abhängigkeit von der Konvergenz der Population bestimmt. Damit soll ein Beitrag zur Frage der geeigneten Rechenzeitaufteilung zwischen globaler und lokaler Suche geleistet werden. Das gleiche Verfahren wird benutzt, um eine neue Variante der direkten Integration, die *verzögerte direkte Integration*, zu erhalten.

Ein wichtiges Ziel der neuen Methode besteht neben der allgemeinen Anwendbarkeit darin, die Anzahl der notwendigen Berechnungen der Fitnessfunktion, meist in Form von Simulationsläufen, zu senken, ohne die hohe Konvergenzsicherheit und Robustheit der beteiligten Verfahren zu gefährden.

3.1 Verfahrensauswahl

Die vorliegende Arbeit benutzt das in Abschnitt 2.2.3 beschriebene GLEAM-Verfahren [Blu90] als Vertreter der Evolutionären Algorithmen, da es sich hierbei um eine EA-Variante handelt, die sowohl für die reine Parameteroptimierung als auch für kombinatorischer Aufgabenstellungen und für Optimierungsprobleme, die eine dynamische Parameteranzahl erfordern [Blu94b, Jak01a], geeignet ist. Da GLEAM auch noch wichtige Elemente der beiden klassischen Ansätze, nämlich der Evolutionsstrategie und der Genetischen Algorithmen, ent-

1. Das Gegenteil einer allgemein anwendbaren Steuerung stellt z.B. der in Abschnitt 1.2.3 beschriebene Ansatz von Zitzler et al. [Zit00] dar, der von einem fixen Zeitbudget für die Optimierung ausgeht.

hält, handelt es sich hierbei um einen EA mit breitem Anwendungsfeld, der verschiedene EA-Aspekte in sich vereint [Blu02]. Damit zeichnet sich GLEAM durch eine hinreichende Allgemeinheit aus, wie sie für die vorliegenden Untersuchungsziele notwendig ist. Die neue Methodik ist aber keinesfalls auf GLEAM beschränkt, sondern kann vielmehr bei einer Vielzahl Evolutionärer Algorithmen angewandt werden. Darauf wird in Abschn. 3.3 näher eingegangen werden.

Ausgehend von der Forderung an geeignete lokale Suchverfahren, möglichst geringe Voraussetzungen an den Suchraum zu stellen, kann die Auswahl bereits auf die direkten lokalen Suchverfahren begrenzt werden. Da praktische Anwendungen in der Regel Beschränkungen aufweisen, ist deren Berücksichtigung ebenfalls von einem geeigneten LSV zu fordern. Ausgehend von den Ausführungen des Abschnitts 1.2.1 besteht nun die Wahl zwischen den Pattern-Strategien, dem Rosenbrock-Verfahren und den Polyeder-Strategien. Die Pattern-Strategien scheiden wegen der bereits erwähnten Nachteile vor allem im Vergleich zum robusteren Rosenbrock-Verfahren aus. Die Polyeder-Strategien unterscheiden sich von den anderen dadurch, daß sie im Grunde von mehreren Startpunkten ausgehen. Das macht sie besonders interessant für Evolutionäre Algorithmen, die ja - einen entsprechenden Suchraum vorausgesetzt - meist mehrere Lösungen liefern, welche dann als Startpunkte genutzt werden können. Die Wahl fällt auf den Complex-Algorithmus, da er Beschränkungen berücksichtigt und in der Leistung dem Simplex-Verfahren ähnlich ist. Das Rosenbrock-Verfahren soll als lokales Verfahren dienen, das von nur einem Startpunkt ausgeht. Bei allgemeinen Leistungsvergleichen hat, wie in Abschn. 1.2.1 ausgeführt, der Rosenbrock-Algorithmus besser abgeschnitten als die beiden Polyeder-Strategien. Ob das bei durch einen Evolutionslauf vorbestimmten Startpunkten auch noch so ist, müssen die nachfolgenden Untersuchungen zeigen. Somit wurden zwei allgemein anwendbare lokale Suchverfahren zur Kombination mit GLEAM ausgewählt: das Rosenbrock-Verfahren und der Complex-Algorithmus.

3.2 Integrationsarten

3.2.1 Initialisierung der Startpopulation durch ein lokales Suchverfahren (Voroptimierung)

Bei der Initialisierung der Startpopulation ist generell auf eine möglichst große Varianz der Individuen zu achten. Die meist verwendete zufällige Bestimmung der Startpopulation gewährleistet das, hat aber zum Nachteil, daß die Startlösungen ziemlich schlecht sein können[2]. Wird dagegen die Startpopulation mit Individuen initialisiert, die zufällig bestimmt und danach mit einem LSV lokal optimiert werden, hängt die resultierende Population auch von der Struktur des Lösungsraums ab. Wenn sie im wesentlichen aus einem oder wenigen Individuen und den dazugehörigen Klonen besteht, so kann das als ein Hinweis auf ein unimodales Problem gedeutet werden. Besteht sie hingegen aus unterschiedlichen Individuen, so deutet das auf eine multimodales Problem hin. Somit bringt eine Analyse der Varianz der Startindividuen mit den in Abschn. 3.2.2.2 vorgestellten Meßverfahren bereits eine Aussage über den wahrscheinlichen Modalitätscharakter des Problems.

2. Bei GLEAM werden zur Initialisierung nur Startwerte zugelassen, die alle expliziten Beschränkungen erfüllen.

Beide in Abschn. 2.3 ausgewählten LSV sind grundsätzlich zur Initialisierung geeignet und die praktischen Experimente werden ergeben, ob eines von beiden bessere Startpopulationen liefert. Außerdem wird ermittelt werden, ob mit kleineren Populationen als bei der klassischen Zufallsinitialisierung gearbeitet werden kann.

3.2.2 Steuerung der lokalen Verbesserung der Evolutionsergebnisse (Nachoptimierung)

Bei der Nachoptimierung soll der EA Lösungen liefern, die sich soweit in der Nähe des globalen Optimums befinden, daß sie vom LSV erreicht werden können (Attraktionsgebiet). Bei multimodalen Problemen ist damit zu rechnen, daß einige Lösungen auch im Attraktionsgebiet von Suboptima liegen. Das Problem besteht dabei darin, zu entscheiden, wann das oder die Attraktionsgebiete erreicht sind und damit sinnvoll vom EA auf das LSV umgeschaltet werden kann. Da das Attraktionsgebiet einer unbekannten Zielfunktion nicht vorher bestimmt werden kann, müssen stattdessen Indikatoren aus dem Fortgang der Optimierung herangezogen werden.

Abb. 3.1 zeigt den typischen Verlauf der Entwicklung einer EA-Optimierung. Dargestellt ist die Fitness des jeweils besten Individuums einer Generation. Es wird deutlich, daß nach anfänglichen großen Steigerungen der Fitnesszuwachs mit Annäherung an das Optimum nachläßt. Oft wird auch das exakte Optimum nicht erreicht und nur eine Näherungslösung geliefert. Im Verlauf der Optimierung gibt es öfter Stagnationsphasen unterschiedlicher Dauer. Im Bild sind einige mit A, B, C und D gekennzeichnet. Wenn die Stagnation als Kriterium herangezogen wird, müssen zu frühe Phasen wie A oder B übergangen werden und der EA darf erst in einer Phase wie C oder besser D abgebrochen werden. Die zeitliche Dauer der Stagnation ist jedoch ebensowenig ein geeignetes Kriterium wie die bisher erreichte Fitness, da längere Stagnationsphasen auch am Anfang einer Evolution auftreten können und der Abstand zum Optimum in praktischen Anwendungen nicht bekannt ist.

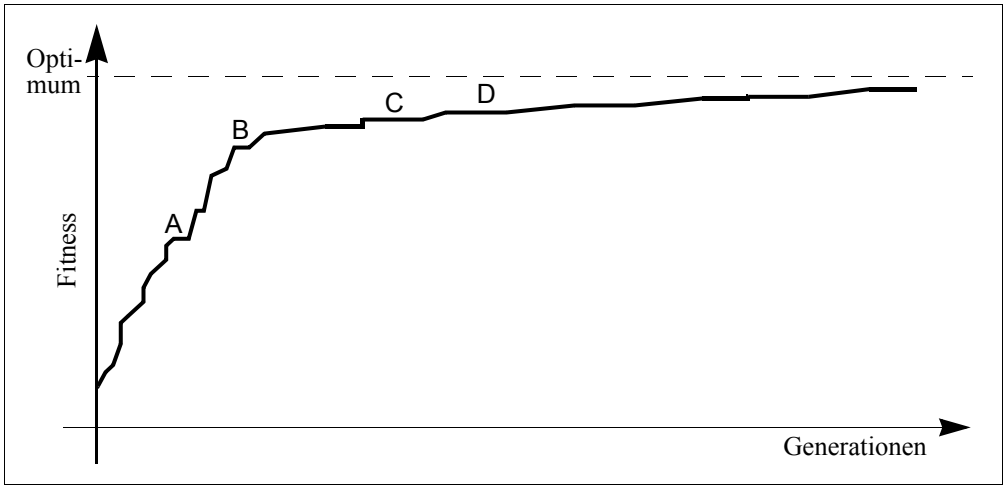


Abb. 3.1: Typischer Verlauf einer EA-Optimierung

Die Dauer des Ausbleibens von Verbesserungen ist allerdings ein Indikator dafür, daß ein größerer Fortschritt vom EA nicht mehr zu erwarten ist. Ein weiteres Kriterium stellt die genotypische Varianz der Population dar, da die Crossover-Operatoren mit abnehmender Varianz an Wirksamkeit verlieren und von einer Konvergenz der Population ausgegangen werden kann.

3.2.2.1 Fortschrittsorientierte Stagnationsindikatoren

Die Beobachtung der Fitness des Generationsbesten sagt nichts darüber aus, was sich in der Population unterhalb dessen Fitness abspielt: Kommt es noch zur Akzeptanz von Nachkommen, findet noch ein lokaler Fortschritt statt oder stagniert tatsächlich die gesamte Population? Das in GLEAM verwendete Populationskonzept erlaubt die Formulierung zweier an den Demes orientierter Stagnationsindikatoren, die eine stärkere Differenzierung erlauben:

Generationen ohne Deme-Verbesserung (GDV)
Es werden die hintereinander auftretenden Generationen gezählt, in denen in keinem Deme eine Verbesserung auftrat. Das ist härter als die Generationen zu zählen, in denen die Fitness des besten Individuums gleich blieb, da jetzt bereits lokale Verbesserungen ausreichen, um die Stagnationsannahme zurückzuweisen.

Generationen ohne Akzeptanz im Deme (GAk)
Es werden die hintereinander auftretenden Generationen gezählt, in denen es in keinem Deme zu einer Akzeptanz eines Individuums gekommen ist. Dieses Kriterium ist nochmals härter als die GDV, da nunmehr bereits lokale Akzeptanz genügt, um noch Bewegung in der Population festzustellen.

Insbesondere das Auftreten von bereits geringen GAk-Werten (ab ca. zehn) ist ein deutlicher Hinweis auf die Konvergenz einer Population.

3.2.2.2 Stagnationsindikator Genetische Varianz

Die Ähnlichkeit von Individuen kann auf der phänotypischen Ebene als erste Näherung durch ihren Fitnessunterschied und auf genotypischer exakt durch den Vergleich der beiden Aktionsketten bestimmt werden. Da bekanntlich eine phänotypische Übereinstimmung wenig über die genotypische aussagt, kann sie nur als notwendige Bedingung für die Ähnlichkeit der Ketten dienen und im Falle ihrer Erfüllung ist eine Überprüfung der beiden Genotypen unerläßlich. Es gibt wenige Autoren, die sich mit dem Problem der Messung des genotypischen Unterschieds zweier Individuen beschäftigt haben. Tagawa et al. [Tag98] experimentieren mit einem neuen Crossover-Operator für das TSP-Problem, der den Hammingabstand zweier pfadcodierter Chromosomen nutzt, um gleiche Gensequenzen in den beiden Eltern möglichst unverändert auf das Kind zu übertragen. Daß der hierfür benutzte Hammingabstand unterschiedliche Positionen im Chromosom lediglich zählt und ihre Distanz ignoriert, mag für die TSP-Anwendung ausreichen, für eine allgemeine Bestimmung des genotypischen Unterschieds genügt das jedoch nicht. Andere Autoren beschränken sich entweder ebenfalls auf metrische Funktionen, basierend auf der Anzahl gemeinsamer Städtepositionen [Frei96] oder gemeinsamer benachbarter Städte [Ron97]. Die genannten Metriken sind zu TSP-spezifisch, als daß sie allgemein anwendbar sind. Daher wird nachfolgend eine eigene Metrik zur Bestimmung der genotypischen Distanz zweier Chromosomen (hier Aktionsketten) entwickelt.

Ein genotypischer Vergleich hängt vom verwendeten Aktionsmodell ab: Bei Aktionsketten fester Länge, bei denen die Reihenfolge der Aktionen nicht bedeutungstragend ist, genügt der Vergleich der korrespondierenden Parameter, andernfalls müssen auch noch die Positionsunterschiede berücksichtigt werden. Noch aufwendiger wird der Vergleich bei Ketten dynamischer Länge. Generell gilt, daß nur nichtleere Ketten verglichen werden. Der Beweis, daß die nachfolgend definierten Abstandsmaße den vier Anforderungen an eine Metrik genügen, ist im Anhang A zu finden.

Für Ketten fester Länge ohne Bedeutungsrelevanz der Aktionsreihenfolge wird der Unterschied durch ein Abstandsmaß der Werte korrespondierender Parameter beider Ketten bestimmt. Damit das Abstandsmaß unabhängig vom verwendeten Aktionsmodell bleibt, wird es auf den Wertebereich zwischen 0 bis 1 normiert, wobei 0 für identische Ketten steht und 1 für maximalen Unterschied. Der Parameterabstand $\Delta_{par}(AK_1, AK_2)$ wird für alle Parameter des Handlungsmodells, für die die Obergrenze des Wertebereichs größer als die Untergrenze ist ($og_i > ug_i$), wie folgt definiert:

Parameterabstand:

$$\Delta_{par}(AK_1, AK_2) = \frac{1}{anz} + \sum_{i=1}^{anz} \frac{\left|param_{i,1} - param_{i,2}\right|}{og_i - ug_i} \tag{3.1}$$

mit: $param_{i,j}$: Wert des i-ten Parameters in der Kette AK_j. Die Parameter werden dabei in der Reihenfolge ihrer Definition im Aktionsmodell durchnumeriert.

ug_i, og_i: Unter- und Obergrenze des Wertebereichs des i-ten Parameters.

anz: Anzahl aller Parameter aller Aktionen.

Bei Ketten mit fester Länge und bedeutungstragender Aktionsreihenfolge wird der Positionsabstand $PA_{1,2}(A_i)$ einer einzelnen Aktion in zwei Ketten AK_1 und AK_2 dadurch definiert, daß die Aktionen jeder Kette sequentiell durchnumeriert werden und dann der Betrag der Differenz ihrer Indizes bestimmt wird:

$$PA_{1,2}(A_i) = \left|I_1(A_i) - I_2(A_i)\right| \tag{3.2}$$

mit: $I_j(A_i)$: Index der Aktion i in der Aktionskette j.

Der gesamte Positionsabstand zweier Ketten $\Delta_{pos}(AK_1,AK_2)$, deren Länge größer als 1 ist, wird dann wie folgt definiert:

Positionsabstand:

$$\Delta_{pos}(AK_1, AK_2) = \frac{1}{abst_{max}} \sum_{i=1}^{len} PA_{1,2}(A_i) \tag{3.3}$$

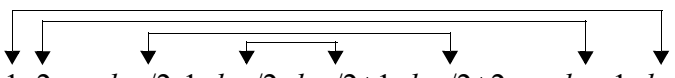
mit: len: Länge der Aktionsketten, entspricht der Anzahl der Aktionen ($len > 1$).

$abst_{max}$: Maximaler Abstand aller Aktionen der Kette mit der Länge len, siehe auch Gl. (3.7) und (3.10).

Der Gesamtabstand Δ_{ges} zweier Ketten errechnet sich aus dem arithmetischen Mittelvon Δ_{par} und Δ_{pos}.

Zur Berechnung des maximalen Positionsabstands $abst_{max}$ kann davon ausgegangen werden, daß der maximale Abstand entweder durch Verschieben aller Aktionen um einen gleichen maximalen Weg erreicht werden kann oder durch Verschieben einer Aktion um *len-1*, der nächsten um *len-3* usw. (symmetrisches Vertauschen).

Das symmetrische Vertauschen sieht für Ketten gerader Länge folgendermaßen aus:



1, 2, ..., *len*/2-1, *len*/2, *len*/2+1, *len*/2+2, ..., *len*-1, *len* *len* gerade

Die *len*/2 Vertauschungen haben die Wege *len*-1, *len*-3, ..., 1. Die Summe der Wege ergibt den maximalen Positionsabstand für symmetrisches Vertauschen $abst_{max,sv}$:

$$abst_{max,sv} = 2\sum_{i=1}^{len/2}(2i-1) = 4\sum_{i=1}^{len/2} i - 2\sum_{i=1}^{len/2} 1 \qquad len \text{ gerade} \qquad (3.4)$$

$$= 4\frac{\frac{len}{2}\left(\frac{len}{2}+1\right)}{2} - len = len\left(\frac{len}{2}+1\right) - len = \frac{len^2}{2}$$

Bei Ketten gerader Länge ergibt sich für alle Aktionen ein gleicher Positionsabstand $PA_{1,2}(A_i)$, wenn um *len/2* Positionen verschoben wird. Dabei erfolgt beim Verschieben über das Kettenende hinaus ein Umbruch, so daß sich der Index $I_{neu}(A_i)$ der verschobenen Aktion A_i berechnet durch:

$$I_{neu}(A_i) = (I(A_i) + weg) \bmod len \qquad (3.5)$$

mit: *weg*: Anzahl der Positionen, um die die Aktion insgesamt verschoben wird.
mod: Modulo-Operator (Divisionsrest).

Die Summe der Positionsabstände bei Verschiebungen aller Aktionen um *len*/2 Positionen beträgt für gerades *len* ebenfalls $len^2/2$. Von *len*/2 abweichende Verschiebungen führen zu einem kleineren Abstand, wie nachfolgende Überlegung zeigt. Bei Abweichungen von *len*/2 kann eine Zunahme um ein δ wegen des Umbruchs am Kettenende auf eine Abnahme um den gleichen Betrag zurückgeführt werden. Für kleinere Verschiebungen ergibt sich:

1. Wegen des verringerten Verschiebungswegs können δ mehr Verschiebungen innerhalb der Kette mit einem Abstand *len*/2 - δ erfolgen.
2. Die Anzahl der restlichen Verschiebungen mit Umbruch beträgt *len*/2 - δ und es wird um *len* -(*len*/2 - δ) = *len*/2 + δ verschoben.

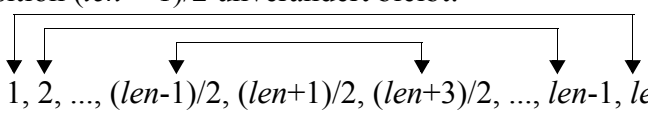
Daraus ergibt sich für Verschiebungen um *len*/2 - δ Positionen für den Abstand $abst_{max,v}$:

$$abst_{max,v} = 2\left(\frac{len}{2}+\delta\right)\left(\frac{len}{2}-\delta\right) = \frac{len^2}{2} - 2\delta^2 \qquad \delta = 0,..., \frac{len}{2}-1 \qquad (3.6)$$

Aus den Gleichungen (3.4) und (3.6) ergibt sich für Aktionsketten gerader Länge $abst_{max}$ als:

$$abst_{max} = \frac{len^2}{2} \qquad len \text{ gerade} \qquad (3.7)$$

Für den ungeraden Fall gelten die gleichen Überlegungen, wobei sich jedoch leicht abweichende Werte ergeben. Das Bild für symmetrisches Vertauschen ändert sich insofern, als die Aktion mit der Position (*len* + 1)/2 unverändert bleibt:



1, 2, ..., (*len*-1)/2, (*len*+1)/2, (*len*+3)/2, ..., *len*-1, *len* *len* ungerade

Für die (*len*-1)/2 Vertauschungen um die Wege *len*-1, *len*-3, ..., 2 ergibt sich für die Wegsumme als den maximalen Positionsabstand für symmetrisches Vertauschen $abst_{max,sv}$:

$$abst_{max,sv} = 2\sum_{i=1}^{\frac{len-1}{2}} 2i = 4\sum_{i=1}^{\frac{len-1}{2}} i = 4\frac{\frac{len-1}{2}\cdot\frac{len+1}{2}}{2} \qquad len \text{ ungerade} \qquad (3.8)$$
$$= 2\frac{len^2-1}{4} = \frac{len^2-1}{2}$$

Auf Grund der ungeraden Länge gibt es keine Verschiebungen aller Aktionen mit gleichem Positionsabstand. Stattdessen wird zunächst der Fall von (*len* - 1)/2 Verschiebungen um (*len* + 1)/2 Positionen betrachtet. Die restlichen (*len* + 1)/2 Verschiebungen mit Umbruch gemäß Gl. (3.5) haben eine Distanz von (*len* - 1)/2. Wird wie im geraden Fall die Verschiebung um ein δ verringert, so ergibt sich folgendes:

1. Es können (*len* - 1)/2 + δ Verschiebungen innerhalb der Kette mit einem Abstand (*len* + 1)/2 - δ erfolgen.
2. Die Anzahl der restlichen Verschiebungen mit Umbruch beträgt *len* - ((*len* - 1)/2 + δ) = (*len* + 1)/2 - δ und es wird um *len* -((*len* + 1)*/2* - δ) = (*len* - 1)*/2* + δ verschoben.

Daraus ergibt sich für Verschiebungen um (*len* + 1)*/2* - δ Positionen und ungeradem *len* für den Abstand $abst_{max,v}$:

$$abst_{max,v} = 2\left(\frac{len-1}{2}+\delta\right)\left(\frac{len+1}{2}-\delta\right) \qquad \delta = 0,\ldots,\frac{len}{2}-1 \qquad (3.9)$$
$$= \frac{(len-1)(len+1)}{2} - len\cdot\delta + \delta + len\cdot\delta - \delta - 2\delta^2$$
$$= \frac{len^2-1}{2} + 2\delta - 2\delta^2 = \frac{len^2-1}{2} - 2(\delta^2-\delta)$$

Daher ist für alle in Frage kommenden δ (len^2 - 1)/2 der größte Wert, den $abst_{max,v}$ annehmen kann. Somit ergibt sich aus den Gleichungen (3.8) und (3.9) für Aktionsketten ungerader Länge $abst_{max}$ als:

$$abst_{max} = \frac{len^2-1}{2} \qquad len \text{ ungerade} \qquad (3.10)$$

Für Ketten mit dynamischer Länge und bedeutungstragender Aktionsreihenfolge müssen die Festlegungen der zuvor definierten Abstandsmaße erweitert werden. Die vorliegende Aufgabenstellung erfordert eine genaue Bestimmung des Unterschieds ähnlicher Ketten, während eine exakte Bestimmung der Differenz wenig ähnlicher Ketten von geringerem Interesse ist und mit zunehmender Unterschiedlichkeit immer geringer wird. Daher genügt eine genaue Bestimmung des Abstands ähnlicher Ketten, was auch den positiven Nebeneffekt eines nicht zu großen Rechenaufwands für die recht häufig vorzunehmenden Abstandsbestimmungen mit sich bringt.

A_{gem} sei die Menge der gemeinsamen Aktionen der beiden Ketten AK_1 und AK_2. Mit *card*(*A*) wird die Anzahl der Elemente einer Menge bezeichnet und mit *len*(AK_i) die Länge der Aktionskette AK_i. Da Aktionen in Ketten dynamischer Länge mehrfach vorkommen können, wird festgelegt, daß sie in der Reihenfolge ihrer Indizierung betrachtet werden. Wenn $A_{gem} = \varnothing$ gilt, wird der Gesamtabstand Δ_{ges} auf 1 gesetzt. Daher gelten die nachstehenden Aussagen nur für nichtleere Ketten mit $A_{gem} \neq \varnothing$.

Der Parameterabstand Δ_{par} ist über A_{gem} definiert. Eine weitergehende Definition, z.B. auch über die restlichen Aktionen der kürzeren Kette, wobei in beiden Ketten nicht vorhandene Aktionen mit maximalem Parameterabstand bewertet werden, ist insofern überflüssig, als der genannte Fall als Nichtpräsenz von Aktionen bereits durch das noch zu definierende Δ_{akt} abgedeckt wird.

Der Positionsabstand Δ_{pos} wird ebenfalls über A_{gem} definiert, wobei das $abst_{max}$ der kürzeren Kette genommen wird. Der dabei gemachte Fehler kann vor allem bei Ketten mit erheblicher Längendifferenz zu einem zu groß bewertetem Δ_{pos} führen. Es kann insbesondere größer als 1 werden, da die Summe der einzelnen Positionsabstände größer als das $abst_{max}$ der kürzeren Kette werden kann. Da derartige Ketten sich aber erheblich unterscheiden, wird der Fehler in Kauf genommen und Δ_{pos} auf den Wert 1 begrenzt. Δ_{pos} wird auf 1 gesetzt, wenn die Länge einer der beiden Ketten kleiner als 2 ist.

Den Unterschied der Aktionspräsenz in beiden Ketten bewertet $\Delta_{akt}(AK_1, AK_2)$. Dazu wird die Anzahl gemeinsamer Aktionen durch das Maximum der Längen beider Aktionsketten dividiert. Dieser Quotient nimmt den Wert 1 bei identischen und den Wert 0 bei vollständig verschiedenen Ketten an. Daher lautet die Formel zur Berechnung von $\Delta_{akt}(AK_1, AK_2)$:

Unterschied der Aktionspräsenz:

$$\Delta_{akt}(AK_1, AK_2) = 1 - \frac{card(A_{gem})}{max(len(AK_1), len(AK_2))} \tag{3.11}$$

mit: $len(AK_i)$: Länge der Aktionskette AK_i, entspricht der Anzahl ihrer Aktionen.
A_{gem}: Menge der gemeinsamen Aktionen der beiden Ketten AK_1 und AK_2.
$card(A)$: Anzahl der Elemente der Menge A.

Zur Bildung des Gesamtabstands Δ_{ges} zweier Ketten als arithmetisches Mittel der drei Abstände wird der Unterschied in der Aktionspräsenz Δ_{akt} dreifach bewertet, da Δ_{par} und Δ_{pos} nur über der Menge der gemeinsamen Aktionen beider Ketten definiert sind.

Tabelle 3.1 faßt die unterschiedlichen Definitionen des Gesamtabstands $\Delta_{ges}(AK_1, AK_2)$ in Abhängigkeit vom Aktionsmodell zusammen.

Aktionsmodell	Gesamtabstand $\Delta_{ges}(AK_1, AK_2)$	Voraussetzungen
feste Länge, Reihenfolge nicht bedeutungstragend	$\Delta_{par}(AK_1, AK_2)$	$len\,(AK) > 0$
feste Länge, Reihenfolge bedeutungstragend	$\frac{\Delta_{par}(AK_1, AK_2) + \Delta_{pos}(AK_1, AK_2)}{2}$	$len\,(AK) > 1$, sonst $\Delta_{ges} = 1$
dynamische Länge, Reihenfolge bedeutungstragend	$\frac{\Delta_{par}(AK_1, AK_2) + \Delta_{pos}(AK_1, AK_2) + 3 \cdot \Delta_{akt}(AK_1, AK_2)}{5}$	$len\,(AK) > 1$ und $A_{gem} \neq \varnothing$, sonst $\Delta_{ges} = 1$

Tab. 3.1: Definitionen des Gesamtabstands $\Delta_{ges}(AK_1, AK_2)$ in Abhängigkeit vom Aktionsmodell.

Um eine Aussage über die genotypische Varianz einer Population machen zu können, werden die Individuen der Reihe nach paarweise mit Hilfe von Δ_{ges} verglichen. Zwei Individuen gelten dabei als ähnlich, wenn $\Delta_{ges} < \varepsilon$, wobei ε ein extern einzustellender Strategieparameter darstellt. Die auftretenden Abweichungen werden aufsummiert und wenn die Summe 2ε überschreitet, erfolgt ein Vergleich des anfänglichen Individuums der Sequenz ähnlicher Individuen mit dem aktuellen. Es wird so eine Sequenz ähnlicher Individuen ermittelt, die mit Auftreten eines zu großen Δ_{ges} beendet wird. Das Individuum in der Mitte einer Sequenz wird als ihr Repräsentant festgelegt. Schließlich ist die Population in mehrere Sequenzen ähnlicher Individuen aufgeteilt. Einzelindividuen, deren Fitness klein gegenüber der Fitness des besten Individuums ist, werden bei dieser Betrachtung vernachlässigt, da sie zur Ermittlung etablierter Subpopulationen nichts beitragen. Als *klein* werden in diesem Zusammenhang Fitnesswerte von weniger als 50% des besten Individuums angesehen. Es könnten noch nichtbenachbarte Sequenzen einander ähnlich sein, was an Hand ihrer Repräsentanten überprüft wird. Das Ergebnis besteht aus der Anzahl sich voneinander unterscheidender Individuengruppen (Nischen) N_{Anz} und deren qualitativen Unterschiede. Von letzterem ist vor allem das Maximum interessant, da es Rückschlüsse auf den Umfang der noch vorhandenen Unterschiede gestattet.

3.2.2.3 Kriterien zum Abbruch des Evolutionären Algorithmus

Das neue Verfahren zur Steuerung der Rechenzeitaufteilung zwischen globaler und lokaler Suche basiert auf der Messung der Konvergenz der Population durch die Bestimmung ihrer genetischen Varianz. Da deren Ermittlung aber recht aufwendig ist, sollte sie nicht nach jeder Generation sondern nur bei begründetem Verdacht auf Konvergenz durchgeführt werden. Dazu dienen die Indikatoren GDV und GAk, für die die Strategieparameter GDV_{min} und GAk_{min} Grenzwerte vorgeben, bei deren Überschreitung die genotypische Varianz der Population bestimmt wird. Wenn es weniger als N_{max} Nischen mit Subpopulationen einander ähnlicher Individuen gibt und deren Unterschiede alle ein ε_{Pop} unterschreiten, wird die Evolution abgebrochen. Die Repräsentanten der Nischen bilden das Ergebnis, wobei noch das beste Individuum der Population hinzugefügt wird, wenn es nicht bereits ein Nischenrepräsentant ist. Die Ergebnis-Individuen dienen anschließend als Startpunkte für entsprechend viele Läufe des Rosenbrock- bzw. des Complex-Verfahrens oder werden zusammen zur Bildung eines Startcomplexes zur Initialisierung eines Complex-Laufs genommen. Auch hier soll ermittelt werden, ob mit kleineren Populationen als bei unverändertem GLEAM gearbeitet werden kann. Tabelle 3.2 faßt die Indikatoren und ihre Strategieparameter für den Wechsel vom EA zum LSV zusammen.

3.2.3 Direkte Integration eines lokalen Suchverfahrens

Bei der direkten Integration werden die Nachkommen einer Paarung lokal optimiert, so daß aus Sicht der Evolution nur die Spitzen der Suboptima betrachtet werden. Dabei kann auch von einer bereits lokal optimierten Startpopulation ausgegangen werden (Kombination mit der Voroptimierung). Pro Paarung werden in GLEAM mehrere Nachkommen erzeugt, die um die Ersetzung des Elter konkurrieren. Somit besteht die Wahl, entweder nur den besten dieser Offsprings lokal zu optimieren oder alle und dann zu vergleichen. Letzteres ist aufwendiger aber dafür auch genauer. Beide Vorgehensweisen sollen erprobt werden.

Die lokale Optimierung eines Nachkommen kann als Lernen innerhalb der Lebensspanne eines Individuums interpretiert werden. Dabei wird von *Lamarckscher Evolution* (*Lamarckian evolution*) gesprochen, wenn das Individuum genotypisch an das Ergebnis der lokalen Suche

Indikator	Strategie-parameter	Erfüllung	Beschreibung
GDV	GDV_{min}	Initiierung der Überprüfung der genotypischen Varianz, wenn $GDV \geq GDV_{min}$ oder $GAk \geq GAk_{min}$	Generationen ohne Deme-Verbesserung
GAk	GAk_{min}		Generationen ohne Akzeptanz im Deme
Δ_{ges}	ε	Zwei Individuen sind einander ähnlich, wenn $\Delta_{ges} \leq \varepsilon$	Unterschied zweier Individuen
N_{Anz}	N_{max}	Wechsel vom EA zum LSV, wenn $N_{Anz} \leq N_{max}$ und $max(\Delta_{ges}) \leq \varepsilon_{Pop}$	Anzahl der Nischen (Subpopulationen ähnlicher Individuen)
$max(\Delta_{ges})$	ε_{Pop}		Maximum der Δ_{ges} der Nischenrepräsentanten

Tab. 3.2: Indikatoren und Strategieparameter für den Wechsel vom EA zum LSV

angepaßt wird. Bleibt das Individuum dagegen unverändert und erhält nur den verbesserten Fitnesswert, so handelt es sich um *Baldwin-Evolution* (*Baldwinian evolution)* [Whi94]. Die wesentliche Wirkung des Verzichts auf eine genotypische Anpassung ist die Bewahrung einer größeren genetischen Varianz in der Population. Whitley et al. [Whi94] haben die Fragestellung an Hand eines einfachen binär codierten elitären GAs und dreier relativ leichter Testfunktionen untersucht. Sie kommen im Gegensatz zu Gruau und Whitley [Gru93] nicht zu dem Ergebnis, daß die Baldwin- sich als genauso effektiv erweist wie die Lamarcksche Evolution. Bei einigen Läufen wurde das globale Optimum zwar nur gefunden, wenn auf den Update des Genotyps verzichtet wurde, andererseits erwies sich die Baldwin-Evolution als deutlich langsamer. Insgesamt kommen sie je nach verwendeter Testfunktion zu uneinheitlichen Ergebnissen. Orvosh und Davis [Orv93] gelangen in ihrer detaillierteren Untersuchung mit unterschiedlichen Anpassungsraten bei komplexeren Anwendungen (Netzwerkdesign und minimale Färbung eines Graphen) zu dem Schluß, daß weder die eine noch die andere Strategie generell die bessere sei, sondern eine Mischung mit geringem Anteil an Lamarckscher Evolution. Sie erhalten in beiden Testfällen die besten Ergebnisse, wenn 5% der akzeptierten lokal optimierten Nachkommen genotypisch angepaßt werden. Die Überprüfung des Ergebnisses wird ebenfalls Gegenstand der Untersuchung sein. Ein weiteres Untersuchungsziel ist die Klärung der Frage, ob mit geringeren Populationsgrößen als beim unveränderten GLEAM gearbeitet werden kann, was wegen der permanenten lokalen Optimierung stark zu vermuten ist. Außerdem werden die Ergebnisse beider LSV miteinander verglichen.

3.2.4 Verzögerte direkte Integration eines lokalen Suchverfahrens

Diese neue Integrationsart basiert auf dem Gedanken, daß es bei manchen Aufgaben sinnvoll sein kann, das LSV erst dann bei der Nachkommensbildung hinzuzunehmen, wenn die Evolution bereits zumindest einen Teil der Population zu einer bestimmten Qualität entwickelt hat. Der Überlegung liegt die Vermutung zu Grunde, daß die Evolution schneller aus verbotenen oder schlechten Teilen des Suchraums herausfindet als ein lokales Verfahren und dessen Einsatz damit verzögert stattfinden sollte. Das im vorigen Abschnitt vorgestellte Instrumentarium zur Kontrolle der Nachoptimierung kann auch zur Steuerung der verzögerten Zuschaltung des LSVs genutzt werden. Die Parametrierung sollte allerdings eine Zuschaltung des LSVs zu einem vergleichsweise früheren Zeitpunkt als bei der Nachoptimierung vorsehen.

3.3 Allgemeinheit der neuen Methode

Die Integrationsarten basieren auf der Anwendung lokaler Suchverfahren auf die Parameter einer Aktionskette zu bestimmten Zeitpunkten.

Bei reiner Parameteroptimierung erfolgt die Zuordnung der Aktionsparameter in der Reihenfolge ihrer Definition im Genmodell. Das entspricht dem Vorgehen bei Parameter-Strings, wie sie bei den meisten EA-Implementierungen üblich sind. Bei Optimierungsaufgaben, bei denen die Reihenfolge der Aktionen bedeutungstragend ist, erfolgt die Parameterzuordnung in der Reihenfolge der Aktionen in der Kette. Auch das entspricht dem üblichen Vorgehen bei Benutzung von Strings, da bei derartigen Aufgaben die Genreihenfolge in den Strings enthalten ist. Bei Ketten dynamischer Länge kann kein Vergleich angestellt werden, da die meisten EA-Implementierungen nur mit Chromosomen fester Länge arbeiten.

Das Complex-Verfahren kann bei Ketten fester Länge sowohl mit einer als auch mit mehreren Ergebnisketten zur Initialisierung des Startcomplexes begonnen werden. Bei Aufgabenstellungen, die hingegen Ketten dynamischer Länge verlangen, kann nur jeweils eine AK zur Initialisierung benutzt werden, da die Dimension aller Eckpunkte des Polyeders gleich sein muß.

Wichtig für die Übertragbarkeit der Methode sind außerdem die Verfahren zur Bestimmung des Zeitpunkts zum Abbruch der Evolution bei nachgeschalteter lokaler Optimierung bzw. zum Zuschalten eines lokalen Verfahrens bei der verzögerten direkten Integration. Sie basieren auf der Bestimmung der genotypischen Varianz (siehe Abschn. 3.2.2.2), die sich problemlos auf Parameter-Strings übertragen läßt, und auf den in Abschn. 3.2.2.1 beschriebenen Stagnationsindikatoren. Letztere dienen zur Bestimmung eines geeigneten Zeitpunktes zur Überprüfung der genotypischen Varianz der Gesamtpopulation. Sie basieren auf dem Deme-Konzept und sind somit nicht unmittelbar auf andere EA übertragbar. Der Indikator der Generationen ohne Deme-Verbesserung (GDV) ist ohne Populationsstruktur nicht anwendbar, während die Anzahl von Generationen ohne Offspring-Akzeptanz (GAk) auch ohne Demes gemessen werden kann. Der GDV-Indikator kann durch die Bestimmung der Generationen ohne Verbesserung des Generationsbesten ersetzt werden. Da das ein weicheres Kriterium als GDV ist, wird es in der Regel zu einer häufigeren Überprüfung der genotypischen Varianz kommen, wodurch ein späterer Abbruch als bei der vorgestellten Methode ausgeschlossen ist. Somit ist auch kein schlechteres Konvergenzverhalten zu erwarten, und das Verfahren zur Bestimmung eines geeigneten Abbruchzeitpunkts der Evolution bzw. zum Zuschalten eines lokalen Verfahrens kann mit den angegebenen Modifikationen als allgemein anwendbar gelten.

3.4 Zusammenfassung der zu untersuchenden Integrationsarten

Eingangs wurden vier Integrationsarten aufgelistet: die Voroptimierung, die Nachoptimierung, die direkte Integration und als neue Art die verzögerte direkte Integration. Abb. 3.2 gibt einen Überblick über die vier Arten zur Kopplung lokaler Suchverfahren mit global suchenden Evolutionären Algorithmen und dem Einsatz der neuen Steuerung zur Verteilung der Rechenzeit zwischen den Verfahren.

Zu den vier Integrationsarten kommen noch die Kombinationen aus Voroptimierung und direkter oder verzögerter direkter Integration hinzu, so daß nunmehr sechs Integrationsarten zu untersuchen sind. Da für die Hybridisierung zwei lokale Suchverfahren ausgewählt wurden,

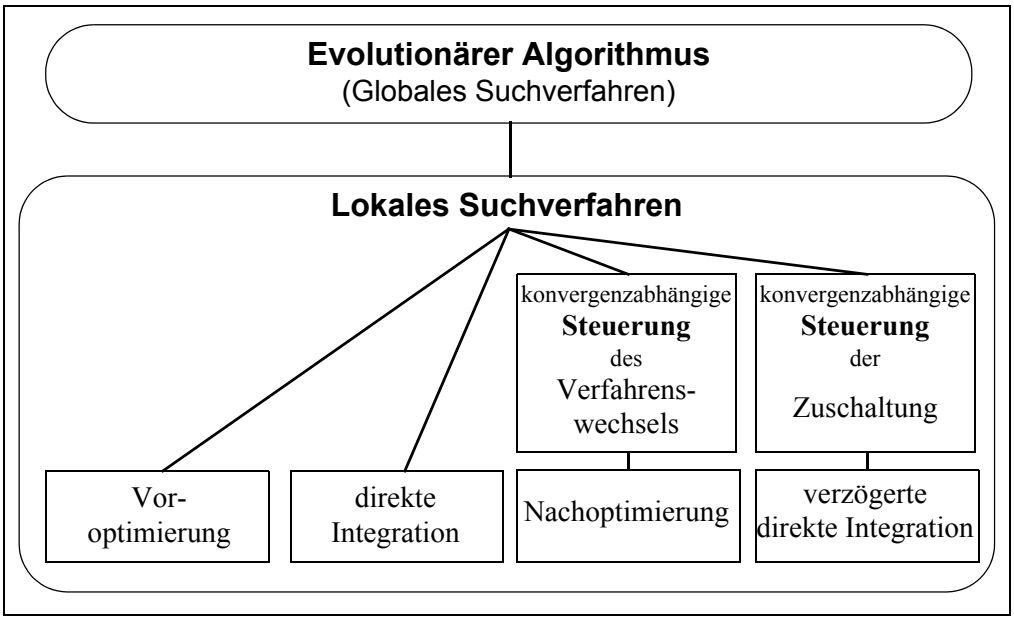


Abb. 3.2: Überblick über die vier Integrationsarten und das Einsatzgebiet der Steuerung der neuen Methode.

verdoppelt sich der Umfang auf insgesamt zwölf EA-Hybride. Es kommt allerdings noch ein weiterer Hybrid hinzu, da bei der Nachoptimierung mit dem Complexverfahren die Alternative besteht, alle EA-Ergebnisse separat zu verbessern oder sie zur Bildung eines einzigen Startpolyeders zu verwenden. Somit sind neben den drei Einzelverfahren folgende 13 Hybride zu untersuchen:

1. Voroptimierung mit dem Rosenbrock-Verfahren.
2. Voroptimierung mit dem Complex-Algorithmus.
3. Nachoptimierung mit dem Rosenbrock-Verfahren.
4. Nachoptimierung mit dem Complex-Algorithmus, wobei alle EA-Ergebnisse getrennt optimiert werden.
5. Nachoptimierung mit dem Complex-Algorithmus, wobei alle EA-Ergebnisse einen Startcomplex bilden und in einem Lauf optimiert werden.
6. Direkte Integration mit dem Rosenbrock-Verfahren.
7. Direkte Integration mit dem Rosenbrock-Verfahren und mit Voroptimierung.
8. Direkte Integration mit dem Complex-Algorithmus.
9. Direkte Integration mit dem Complex-Algorithmus und mit Voroptimierung.
10. Verzögerte direkte Integration mit dem Rosenbrock-Verfahren.
11. Verzögerte direkte Integration mit dem Rosenbrock-Verfahren und mit Voroptimierung.
12. Verzögerte direkte Integration mit dem Complex-Algorithmus.
13. Verzögerte direkte Integration mit dem Complex-Algorithmus und mit Voroptimierung.

Generell geht es bei den nachfolgenden Untersuchungen immer um die Frage, ob der zusätzliche Aufwand, den eine wie auch immer geartete Hinzunahme eines LSVs verursacht, durch eine verbesserte Leistung des EAs wieder wettgemacht oder besser sogar übertroffen wird.

4. Benchmarkaufgaben

Die in Kap. 3 beschriebene Methode zur Verbesserung der Leistungsfähigkeit Evolutionärer Algorithmen soll an Hand von Benchmarkaufgaben überprüft werden. Für eine möglichst repräsentative Auswahl sind folgende Kriterien maßgebend:

1. Art des Problems
   Neben der reinen Parameteroptimierung kontinuierlicher oder ganzzahliger Größen können auch kombinatorische Aspekte eine Rolle spielen. Außerdem kann die Anzahl der Parameter in Abhängigkeit von der gewählten Lösung schwanken, wie z.B. bei der Roboterbahnplanung.

2. Anzahl der Parameter
   Die Schwere einer Aufgabenstellung hängt auch von der Anzahl der zu betrachtenden Parameter ab. Die Parameteranzahl wird in vier Klassen eingeteilt:
   klein: bis zu fünf Parameter
   mittel zwischen sechs und 20 Parameter
   groß zwischen 21 und 50 Parameter
   sehr groß mehr als 50 Parameter

3. Modalität
   Die Unterscheidung zwischen uni- und multimodalen Problemen sagt insofern wenig aus, als die Klasse der multimodalen Probleme sehr heterogen ist. Es wird daher noch zwischen multimodalen und stark multimodalen Problemen mit sehr vielen Suboptima unterschieden. Reale Aufgaben sind typischerweise multimodal bis stark multimodal. Trotzdem soll auch eine unimodale Aufgabe betrachtet werden, um Aussagen über das Verhalten des Verfahrens bei scheinbar einfachen Problemen treffen zu können. Als Extremform multimodaler Aufgaben kann die Klasse der fraktalen Funktionen angesehen werden.

4. Beschränkungen
   Reale Aufgaben haben typischerweise explizite und häufig auch implizite Beschränkungen.

Tabelle 4.1 enthält eine Bewertung der ausgewählten Benchmarkaufgaben hinsichtlich der vier genannten Kriterien. Sie zeigt, daß die gewählten Aufgaben den gesamten Bereich der Auswahlkriterien abdecken und daher als repräsentativ betrachtet werden können. Bis auf die mathematischen Benchmarkfunktionen handelt es sich bei allen Aufgaben um praktische Probleme mit realem Hintergrund, wobei auch ein Teil der Benchmarkfunktionen von realen Problemen abgeleitet ist. Ein weiteres mehr praktisches Kriterium ist die erforderliche Zeit zur Berechnung der Bewertung einer Lösung. Sie darf nicht zu groß sein (maximal wenige Sekunden), da die stochastische Natur des Verfahrens statistische Untersuchungen notwendig macht, so daß eine Vielzahl von Läufen für jeweils eine Verfahrenseinstellug durchgeführt werden muß.

Aufgabe	Parameter-Optimierung	Kombinatorische Optimierung	Parameteranzahl fix / variabel	Parameteranzahl	Modalität	implizite Beschränkungen
Auswahl mathematischer Benchmarkfunktionen	ja	nein	fix	klein bis groß (wählbar)	unimodal bis fraktal	nein
Designoptimierung	ja	nein	fix	klein	stark multimodal	nein
Ressourcenplanung	ja (ganzzahlig)	ja	fix	sehr groß	multimodal	ja
Kollisionsfreie Roboterbahnplanung	gemischt ganzzahlig	ja	variabel	groß bis sehr groß	stark multimodal	ja

Tab. 4.1: Benchmarkaufgaben und ihre Zuordnung zu den Auswahlkriterien

4.1 Mathematische Benchmarkfunktionen

Die betrachteten Benchmarkfunktionen sind der Softwaresammlung von Bäck [Bäc92] entnommen. Die nachfolgend angegebenen Wertebereiche beziehen sich auf alle Dimensionen. Aus Aufwandsgründen wird mit Ausnahme von Shekel's Foxholes in den Experimenten nicht verlangt, daß das exakte Minimum gefunden wird. Stattdessen wird bei den Benchmarkfunktion eine Annäherung angegeben, die als ausreichend angesehen wird.

4.1.1 Schwefel's Sphere

Beim 30-dimensionalen Kugelproblem von Schwefel [Schw95, Problem 1.1, S.325] [Bäc92, Funktion 1] handelt es sich um eine unimodale Funktion, die erst durch den großen Wertebereich von $[-10^{10},10^{10}]$ und durch die hohe Parameteranzahl schwierig wird. Abb. 4.1 zeigt den zweidimensionalen Fall im Bereich von [-1000, 1000]. Das Minimum liegt im Koordinatenursprung und für die Versuche wurde ein Zielwert von 0.01 als hinreichend genau angesehen, um die Laufzeiten nicht zu groß werden zu lassen. Der extrem geringe Gradient der Funktion in der Nähe des Minimums stellt für viele Suchverfahren ein Pro-

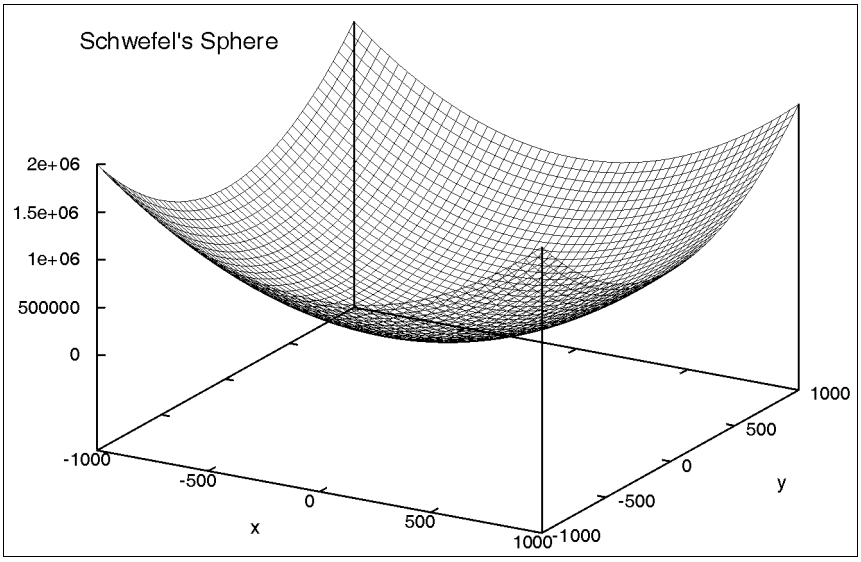


Abb. 4.1: Schwefels Kugelproblem (Ausschnitt)

blem dar. Daher schneiden bei der Aufgabe Suchverfahren mit adaptiver Schrittweitensteuerung wie z.B. die ES besonders gut ab.

4.1.2 Shekel's Foxholes

Shekel's Foxholes [She71] [Bäc92, Funktion 5] ist eine vergleichseise einfache multimodale Funktion, bei der sich in einer Ebene 25 Vertiefungen, die sogenannten Fuchslöcher, befinden, siehe Abb. 4.2. Das Minimum hat einen Wert von 0.99804 und liegt bei (-31.92, -31.9481). Da die Fläche außerhalb der Lochzone nahezu eben ist, haben lokale Suchverfahren häufig bereits Probleme, in den interessanten Bereich zu kommen. Aber auch für die Standard-ES gilt diese Aufgabe als schwierig [Gor99a].

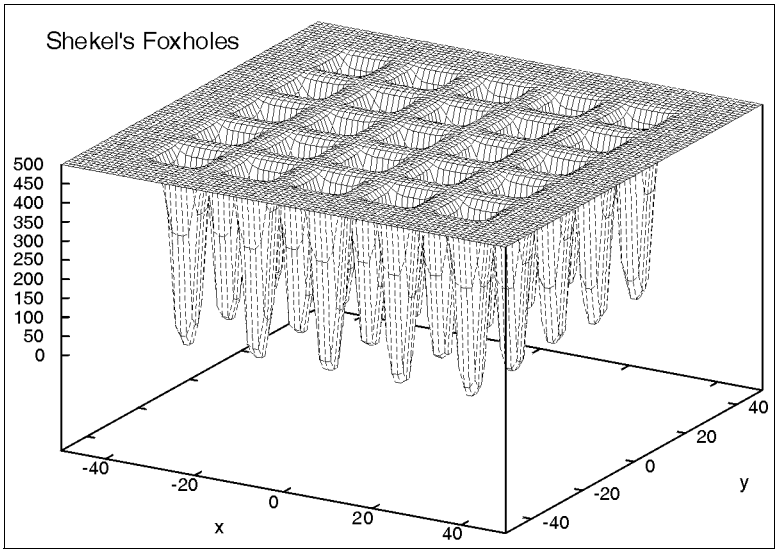


Abb. 4.2: Shekel's Foxholes (Ausschnitt)

In den Experimenten wurde das exakte Minimum im Wertebereich von [-500,500] gesucht.

4.1.3 Verallgemeinerte Rastrigin Funktion

Die verallgemeinerte Rastrigin Funktion [Tör89] [Bäc92, Funktion 7] geht auf ein verallgemeinertes regelungstechnisches Problem zurück und gilt ebenfalls als schwierig für die Standard-ES [Gor99a]. Die 20-dimensionale multimodale Funktion ist im Bereich [-5.12, 5.12] definiert und hat ihr Minimum im Ursprung. Bei den Experimenten wurde ein Zielwert von 0.0001 als ausreichend betrachtet. Abb. 4.3 zeigt den 2-dimensionalen Fall im gesamten Wertebereich.

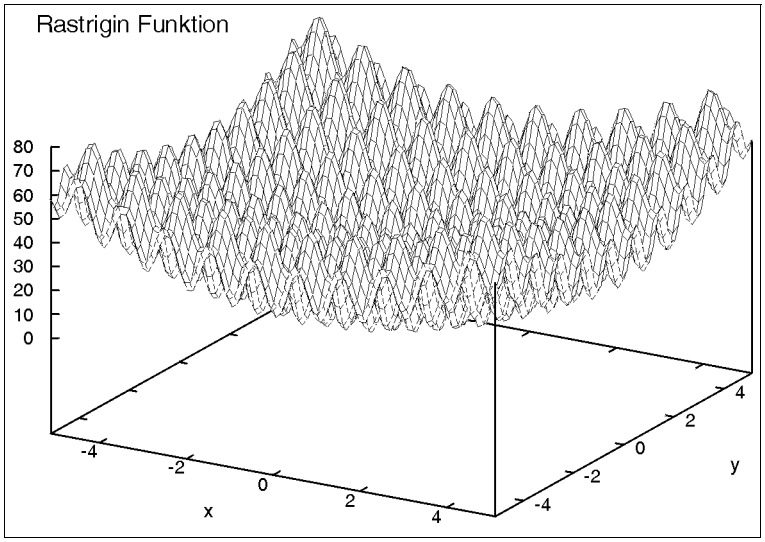


Abb. 4.3: Verallgemeinerte Rastrigin Funktion

4.1.4 Fletcher's Function

Die fünfdimensionale Funktion nach Fletcher und Powell [Schw95, Problem 2.13, S.335] [Bäc92, Funktion 16] ist im Wertebereich [-3.14, 3.14] definiert und hat den Wert Null als Minimum. Bei den Experimenten wurde ein Zielwert von 0.00001 als ausreichend angesehen. Abb. 4.4 zeigt die multimodale Funktion.

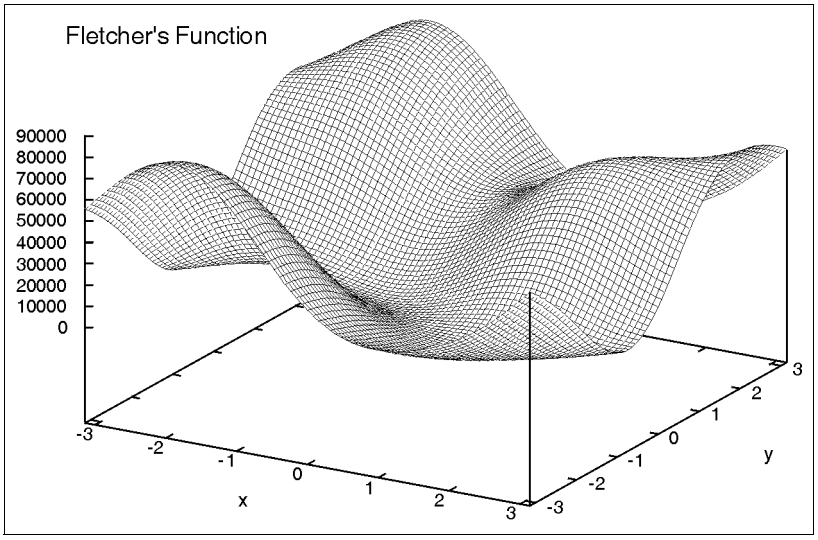


Abb. 4.4: Funktion nach Fletcher und Powell

4.1.5 Fraktale Funktion

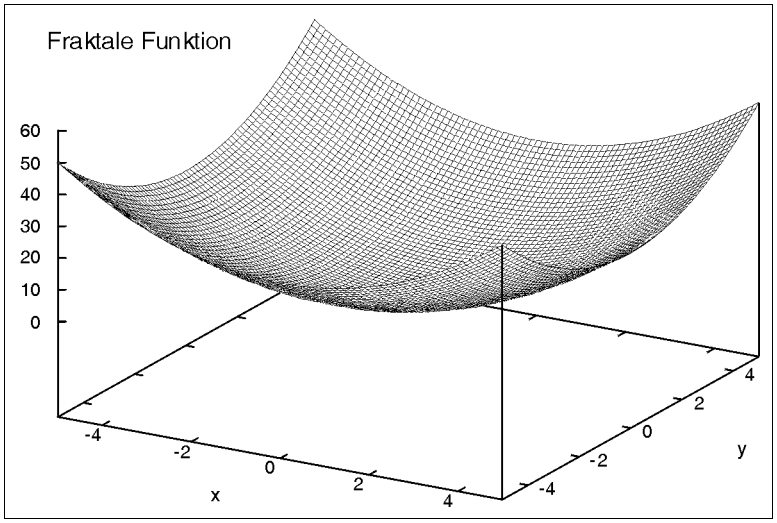


Abb. 4.5: Fraktale Funktion im gesamten Definitionsbereich

Die fraktale Funktion nach Weierstrass und Mandelbrot [Ber80] [Bäc92, Funktion 13] wird hier in ihrer 20-dimensionalen Ausprägung als ein Beispiel für eine extrem multimodale Funktion genutzt. Sie ist im Wertebereich [-5, 5] definiert und hat ein unbekanntes Minimum kleiner Null. Als Zielwert für die Experimente wird daher ein Wert von -0.05 genommen. Abb. 4.5 zeigt ihren Verlauf bei grober Auflösung im gesamten Definitionsbereich. Der Eindruck einer einfachen unimodalen Funktion täuscht jedoch, wie Abb. 4.6 zeigt.

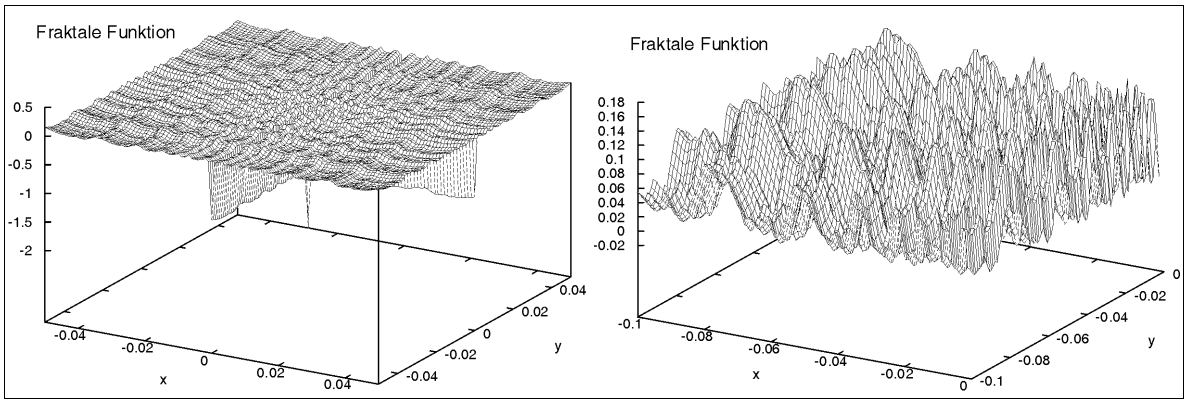
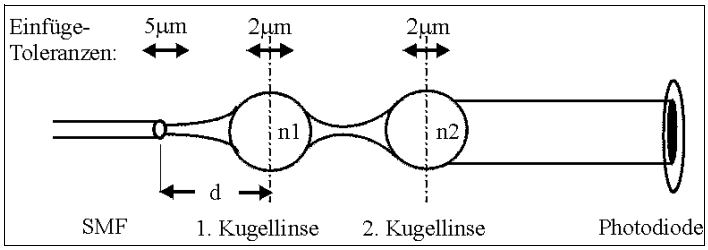


Abb. 4.6: Zwei Ausschnitte aus der fraktalen Funktion, die ihren extrem multimodalen Charakter verdeutlichen.

4.2 Designoptimierungsaufgabe Heterodynempfänger

Der Heterodynempfänger ist ein mikrooptisches Empfangsmodul für die optische Kommunikation basierend auf Lichtwellenleitern. Er gehört zu den kohärenten optischen Lichtwellensystemen. Sieber beschreibt den Empfänger folgendermaßen [Sie99, S.27 u. 28]: „Diesen Systemen liegt die Idee zugrunde, das empfangene Signal kohärent mit einer weiteren optischen Welle zu überlagern, bevor diese auf einen Photodetektor trifft. Die optische Welle wird am Empfänger durch eine Halbleiterlaserdiode (lokaler Oszillator), die eine schmalbandige Abstrahlcharakteristik besitzt, erzeugt. Der durch die Überlagerung erzielte Effekt ist eine Verbesserung des Verhältnisses von Signal zu Rauschen. ... Die Aufgabe des Linsensystems in einem Heterodynempfänger besteht darin, die verschiedenen Abstrahlungen der das Signal führenden Faser und des lokalen Oszillators zu refokussieren, so daß sie an der Position der Überlagerung dieselbe Strahlcharakteristik besitzen. Dies kann erreicht werden, wenn sowohl das empfangene Signal als auch das lokale Oszillatorsignal am Ort der Photodiode kollimiert sind. ... Der Aufbau des Heterodynempfängers ist extrem positionssensitiv bezüglich der aktiven und passiven optischen Komponenten, so daß Positionsfehler der einzelnen Komponenten die Leistung des Empfangsmoduls beeinträchtigen.“ Zum Heterodynempfänger wurden eine Vielzahl von Untersuchungen durchgeführt, siehe [Egg97, Sie98a, Sie98b, Egg98, Gor98c, Jak99b, Sie00]. Für die vorliegende Arbeit soll eines der Optimierungsprobleme, das sich wegen seiner kurzen Rechenzeiten und seinem hohen Grad an Multimodalität gut für statistische Untersuchungen eignet, weiter behandelt werden.



4.7: Das Kollimationssystem des Heterodynempfängers, aus [Jak99a]

Die Optimierungsaufgabe [Jak99a] besteht darin, den Entwurf so zu parametrieren, daß die Toleranzeffekte, die durch das Einfügen der optischen Elemente in bereits vorgefertigte LIGA-Strukturen [Ehr87] entstehen, minimiert werden. Die in Abb. 4.7 dargestellten Einfügetoleranzen der beiden Kugellinsen und des Lichtleiters (SMF, single mode fiber) haben Auswirkungen auf die Strahlweite am Ort der Photodiode und auf die Lage der Strahltaille. Es soll ein Kollimationssystem bestimmt werden, das möglichst unempfindlich hinsichtlich der zu erwartenden Einfügeungenauigkeiten ist, damit bei der Fertigung eine möglichst geringe Ausschußquote auftritt. Die variierbaren Parameter sind die Brechungsindizes der beiden Kugellinsen (*n1* und *n2*) und der Abstand des Lichtleiters zu der ersten Kugellinse (*d*). Als Optimierungskriterien werden *Ausleuchtung*, *Stabilität*, *Strahltaillenposition* und *Linsenabstand* definiert:

- Ausleuchtung
  Die Ausleuchtung wird als Quotient aus der maximalen Strahlweite an der Position der Photodiode und dem Durchmesser des photosensitiven Bereichs der Photodiode definiert. Als Optimierungsziel wird eine Ausleuchtung von 90% bis maximal 95% gewählt, um bei lateralem Strahlversatz eine Überstrahlung der Photodiode zu verhindern.

- Stabilität
  Die Stabilität wird als Quotient aus minimaler und maximaler Strahlweite in Abhängigkeit von den Toleranzeffekten definiert. Das Optimierungsziel besteht aus einem mög-

lichst großen Wert nahe bei 100%, um so toleranzbedingte Schwankungen gering zu halten.

- Strahltaillenposition
  Der optimale Wert der Strahltaillenposition liegt bei 4300 µm, was dem Abstand der zweiten Kugellinse zu der Photodiode entspricht. Das Optimierungskriterium sorgt dafür, daß die kollimierte Strahltaille auf die Photodiode abgebildet wird.

- Linsenabstand
  Das Kriterium *Linsenabstand* wurde eingeführt, um eine Grenze für die Ausdehnung des Gesamtsystems festzulegen. Die angestrebten Werte sollen im Bereich zwischen 80 µm und 1400 µm liegen

Die Werte der Kriterien werden auf den Wertebereich [0.0, 100000.0] der Zielfunktion von GLEAM zu einem Notenwert (Fitness) normiert, siehe auch Abschn. 2.2.3. Die einzelnen Normierungsfunktionen sind in Abb. 4.8 dargestellt. Die Gewichtung und Priorisierung der Kriterien ergibt sich aus Tabelle 4.2. Dabei entspricht die Priorität Eins der höchsten Priorität.

Kriterium	Gewicht	Priorität
Linsenabstand	10 %	1
Ausleuchtung	25 %	2
Strahltaillenposition	25 %	2
Stabilität	40 %	3

Tab. 4.2: Gewichtung und Priorisierung der Bewertung für den Heterodynempfänger

Die Funktion für den Linsenabstand (Abb. 4.8 a) besteht nur aus Geradenstücken und gibt vor, daß ein Abstand zwischen 100 und 1000 µm optimal ist und Werte im Bereich 80 µm - 1400 µm noch akzeptabel sind (Erfüllungswerte). Erst wenn die Erfüllungswerte erreicht worden sind, werden die Kriterien der nächst niedrigeren Priorität Zwei betrachtet. Das sind gemäß Tabelle 4.2 die Ausleuchtung und die Strahltaillenposition. Wenn bei beiden die jeweiligen Erfüllungswerte (siehe Abb. 4.8 b und c) erreicht sind, wird auch das letzte Kriterium, die Stabilität (siehe Abb. 4.8 d) mitbewertet. So wird erreicht, daß nur Entwürfe, die die durch den Linsenabstand vorgegebene Größe nicht überschreiten, hinsichtlich Ausleuchtung und Strahltaillienposition soweit verbessert werden, bis die Erfüllungswerte beider Kriterien mindestens erreicht sind. Erst dann erfolgt die eigentliche Optimierung in Bezug auf die Stabilität. Dabei bewirkt ein Abfallen eines der anderen Kriterien unter seinen Erfüllungswert eine deutliche Absenkung der Gesamtnote. Durch die hohe Gewichtung der Kriterien niedrigerer Priorität wird dieser Effekt noch einmal verstärkt. Dadurch wird die eigentliche Fitnessfunktion entsprechend den Optimierungszielen verzerrt und erhöht. Die linke Hälfte von Abb. 4.9 zeigt den Verlauf der Fitnessfunktion bei fixiertem Abstand *d*. Es wird deutlich, daß die recht große Ebene hoher Qualität in sich zerklüftet ist. Das wird bei der im rechten Teil von Abb. 4.9 dargestellten Ausschnittsvergrößerung noch deutlicher. Der Grund für die starke Multimodalität liegt in der schwankenden Stabilität hinsichtlich der Einfügetoleranzen. Damit ist diese Designoptimierungsaufgabe trotz ihrer nur drei Parameter für eine globale Optimierung interessant.

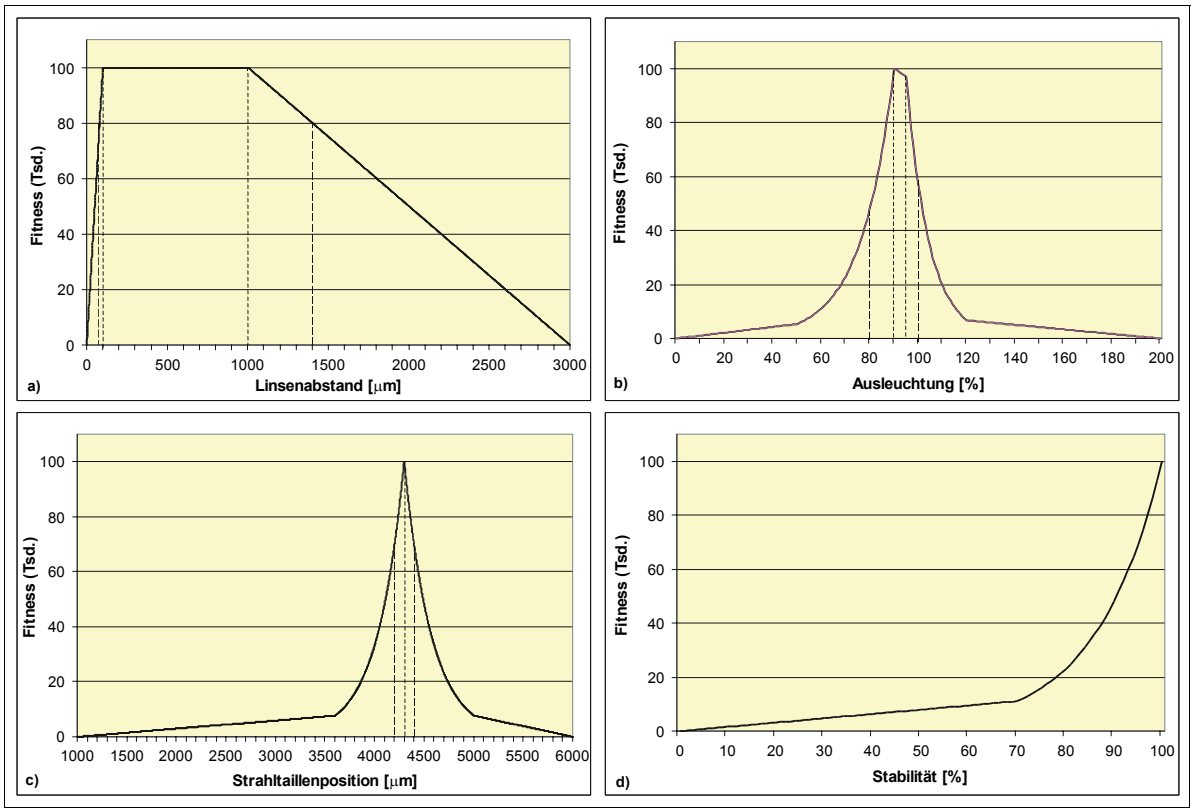


Abb. 4.8: Normierungsfunktionen für die 4 Bewertungskriterien des Heterodynempfängers mit Erfüllungswerten (grob gestrichelte Linien) und Stützpunkten (fein gestrichelte Linien)

Als Optimierungsziel wird eine Stabilität von mehr als 90% unter Einhaltung der anderen Kriterien gefordert, wobei für die Experimente einschränkend zu dem eingangs erwähnten Bereich von 90 - 95% eine Ausleuchtung zwischen 90 und 92% als Einhaltung gefordert wird. Die ersten drei Kriterien ergeben gemäß Tabelle 4.2 damit eine Fitness von knapp 60000. Zusammen mit dem gewichteten Wert für eine Stabilität von mindestens 90% beträgt die geforderte Gesamtfitness mindestens 80500.

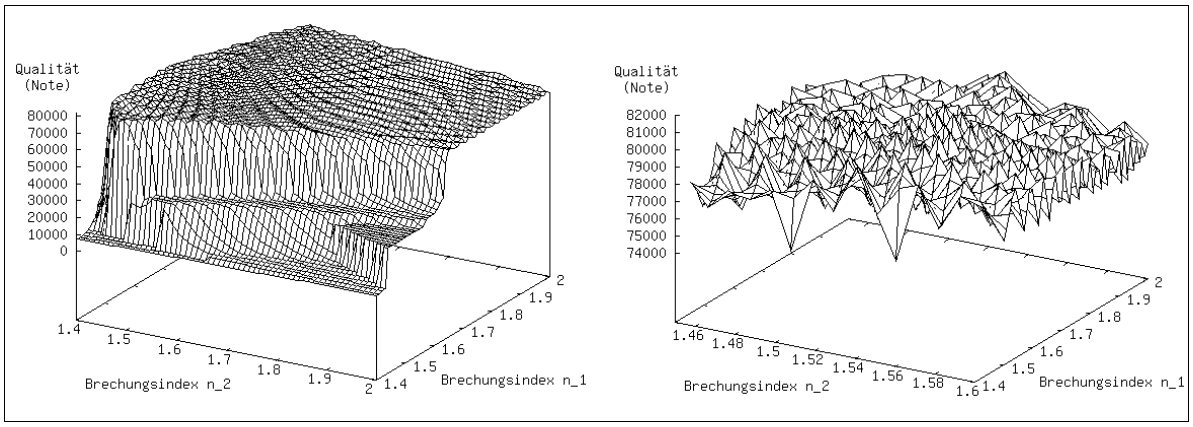


Abb. 4.9: Fitnessfunktion bei fixiertem Abstand d. Der rechts dargestellte Ausschnitt verdeutlicht die starke Multimodalität.

4.3 Ressourcenoptimierung in der Verfahrenstechnik

Blume beschreibt die Aufgabenstellung folgendermaßen [Blu94a, S.25]: „In der Verfahrenstechnik werden zur Herstellung einer Charge häufig bei Produktionsbeginn und -ende mehr Mitarbeiter benötigt als in der Zwischenzeit, in der kontrollierende Tätigkeiten dominieren. Abb. 4.10 zeigt einige typische Beispiele. Startet man einen Durchlauf eines Verfahrens (Charge), so ist die Anzahl der benötigten Mitarbeiter für die nächsten Stunden oder Tage festgelegt, das heißt, zu Beginn und am Ende der Charge werden mehrere Mitarbeiter benötigt, dazwischen meist nur wenige, da z.B. ein Mitarbeiter mehrere Verfahren beaufsichtigt. Daher kann man nach der Startphase einer Charge, wenn nur noch wenige Mitarbeiter gebraucht werden, parallel dazu eine zweite Charge starten, um die freie Mitarbeiterkapazität sinnvoll einzusetzen, siehe Abb. 4.11. Dies gilt nun sinngemäß für weitere Chargen von Verfahren, solange man nicht die vorgegebene maximale Mitarbeiteranzahl überschreitet. Das Problem besteht nun darin, die Chargen so beginnen zu lassen und ihren Kapazitätsbedarf so zu "verzahnen", daß der kumulierte Mitarbeiterbedarf möglichst homogen ist und im Rahmen einer vorgegebenen oder zu optimierenden Obergrenze verbleibt. Eine weitere Anforderung, die die Planung erschwert, besteht darin, daß einige Verfahren zum Start ein Vorprodukt benötigen, welches von einem anderen Verfahren erzeugt wurde. Daher können Chargen dieser Verfahren erst nach Abschluß anderer Chargen, die eine ausreichende Menge der Vorprodukte hergestellt haben, beginnen (Verfahrenskette).“

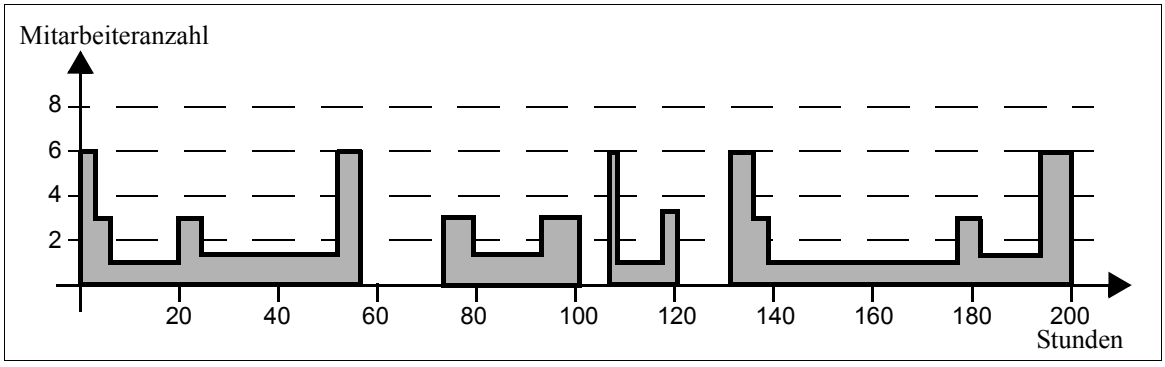


Abb. 4.10: Typische Mitarbeiterbedarfsprofile zur Herstellung unterschiedlicher Chargen, aus [Blu94a]

Die konkrete Planungsaufgabe bestand darin, unter Einhaltung vorgegebener Endtermine 87 Chargen mit maximal zwölf Mitarbeitern in neun Anlagen mit stundengenauer Planung innerhalb von maximal zehn Wochen (210 Schichten) herzustellen. Dabei wurde in Schichten zu je acht Stunden rund um die Uhr und sieben Tage die Woche produziert.

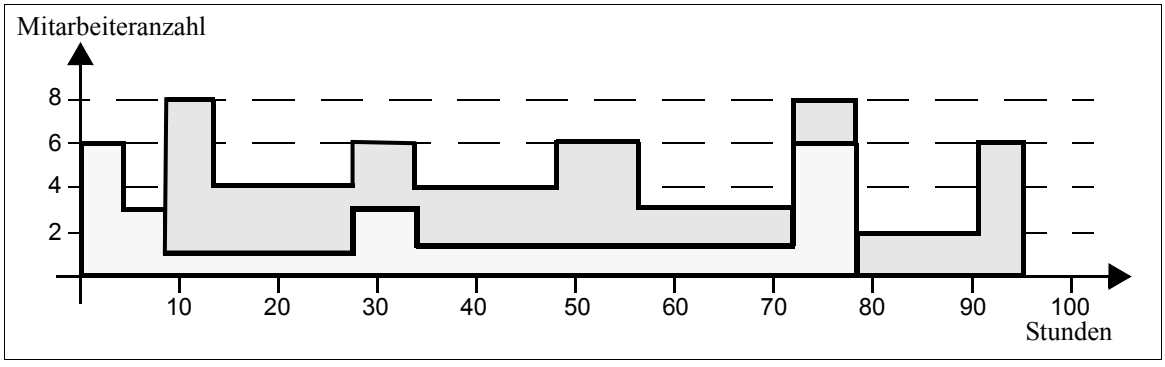


Abb. 4.11: Parallele Bearbeitung zweier Chargen, aus [Blu94a]

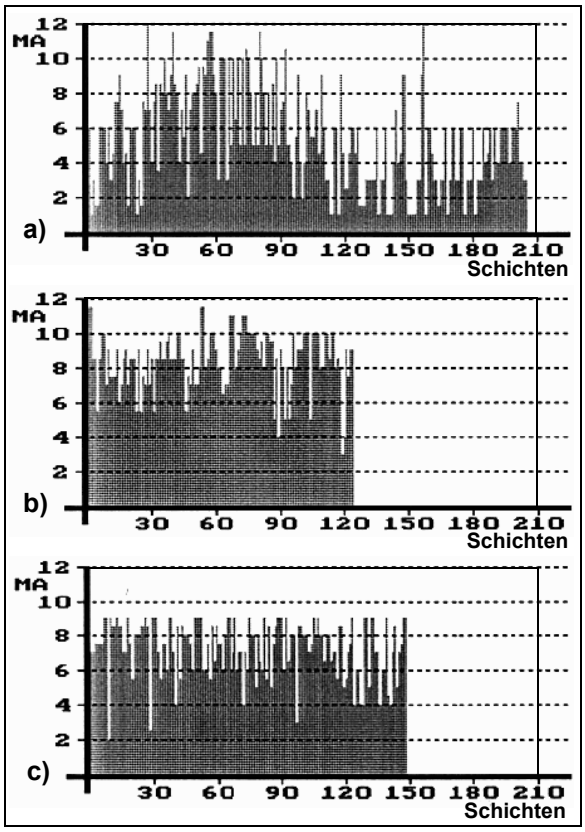


Abb. 4.12: 3 Planungen im Vergleich, aus [Blu94a]:
a) Standardlösung (Vorgabe)
b) zeitoptimierte Planung
c) zeit- und mitarbeiteroptimierte Planung

Abb. 4.12 a) zeigt eine typische manuell erstellte Lösung, bestehend aus knapp 210 Schichten mit maximal zwölf Mitarbeitern pro Schicht, wobei solch hohe Schichtspitzenwerte selten vorkamen. Der typische Mitarbeiterbedarf schwankte zwischen sechs und acht; es gab aber auch Schichten, die nur maximal ein oder zwei Mitarbeiter benötigten. Ein derartig „zerklüftetes" Schichtspitzenprofil legt natürlich die Vermutung nahe, daß hier planerisch einiges verbessert werden kann. Blume [Blu94a] gelang es, bei einer rein zeitoptimierten Planung unter Beibehaltung des Schichtspitzenwertes von zwölf Mitarbeitern den Zeitbedarf von den vorgegebenen 70 Tagen auf 41 Tage und 2 Stunden zu reduzieren (siehe Abb. 4.12 b), was einer Einsparung von 41% entspricht. Bei einer zeit- und mitarbeiteroptimierten Zielvorgabe konnten die Schichtspitzen wie in Abb. 4.12 c) dargestellt, auf neun Mitarbeiter reduziert werden, wobei die zeitliche Verkürzung mit einer Dauer von 49 Tagen und sechs Stunden etwas geringer ausfiel (30% Einsparung). Bei dem Produkt aus maximal benötigten Mitarbeitern und Zeit ergeben sich Einsparungen von 41% (Fall b) bzw. 47% (Fall c).

Als Benchmark soll die mitarbeiter- und zeitoptimierte Planung mit einem Planungsziel von 50 Tagen (150 Schichten) bei maximal neun Mitarbeitern pro Schicht dienen. Bei der Planung müssen die Reihenfolge der Chargen, die belegte Anlage und der Startzeitpunkt (in Stunden) bestimmt werden. Neben dem ganzzahligen Parameter Zeit spielen also auch kombinatorische Aspekte eine Rolle. Um sich ein Bild von der Komplexität der Aufgabenstellung zu machen, genügt es bereits, nur die Anzahl der möglichen Startszenarien der 87 Chargen bei einem Zeithorizont 1680 Stunden zu bestimmen: $1680^{87} \approx 4 \cdot 10^{280}$. Dabei sind die Alternativen bei den Anlagenzuordnungen noch gar nicht berücksichtigt.

Die Bewertung erfolgt in Anlehnung an Blume [Blu94a] durch vier Kriterien, deren Erreichung durch zwei Hilfskriterien unterstützt wird:

Kriterium	Gewicht	Priorität	Straffunktion
Gesamtzeit	70 %	1	ja
Lückenstunden (Hilfskriterium)	5 %	1	nein
Schichtspitzenmaximum	6 %	2	ja
Schichtspitzenüberhang (Hilfskriterium)	19 %	1	nein
Terminverzug	0 %	1	ja
Minderproduktion	0 %	1	ja

Tab. 4.3: Kriterien zu Bewertung der Ressourcenoptimierung

- Gesamtzeit
  Die Gesamtzeit umfaßt die Zeit zwischen Arbeitsbeginn und Ende der letzten Schicht in Stunden. Abb. 4.13, links zeigt, daß mit Erreichen des Zielwerts von 1200 Stunden der volle Notenwert erreicht ist. Bei weniger als 1360 Stunden gilt das Kriterium im Sinne der Prioritätensteuerung als erfüllt. Unterhalb von 1400 Stunden geht die Gerade in eine Exponentialfunktion über, um in der schwierigen Endphase kleine Zeitverringerungen stärker zu honorieren. Ab 1680 Stunden (70 Schichten) wird die Gesamtnote durch die Straffunktion abgewertet, die als Exponentialfunktion ausgeführt ist, um auch noch sehr große Zeiten zu erfassen. Sie ergibt einen Faktor zwischen Null und Eins, mit dem die Gesamtnote multipliziert wird, siehe Abb. 4.13, rechts. Mehrere Straffunktionen wirken kumulativ.

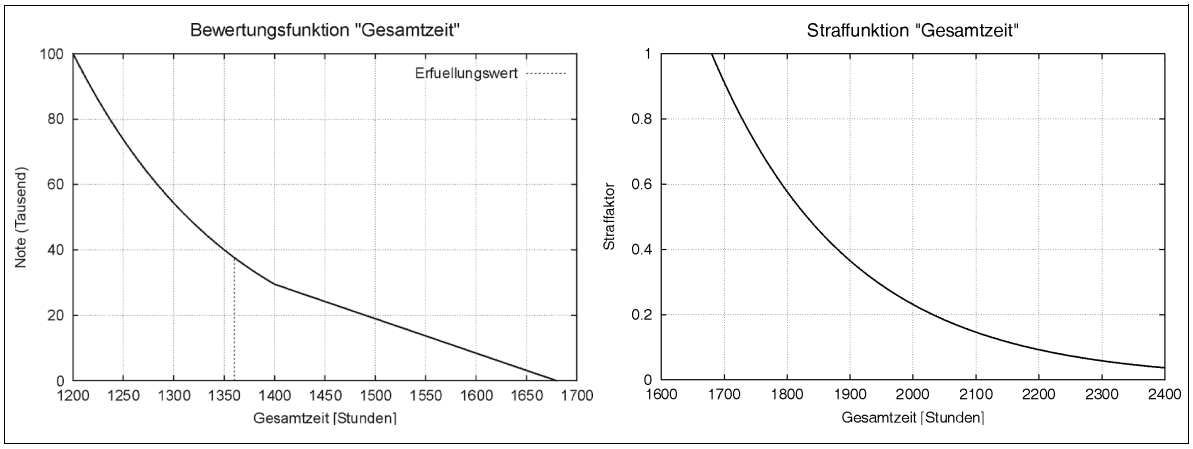


Abb. 4.13: Bewertungskriterium „Gesamtzeit“ (links) mit Straffunktion (rechts)

- Lückenstunden
  Das Hilfskriterium *Lückenstunden* erfaßt Zeiten ohne Arbeit zwischen dem Beginn der ersten Schicht und dem Ende der letzten. Es stellt für den EA einen gezielten Anreiz dar, Zeitlücken zu schließen, auch wenn das zunächst noch nicht zu einer Verkürzung der Gesamtzeit führt. Abb. 4.14 zeigt die zugehörige Normierungsfunktion mit dem Erfüllungswert. Es wurde eine Exponentialfunktion gewählt, um auch noch sehr große Lücken zu erfassen.

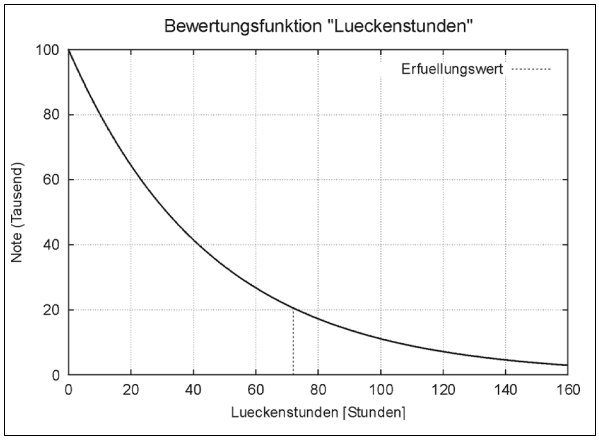


Abb. 4.14: Bewertungskriterium „Lückenstunden“

- Schichtspitzenmaximum
  Der Maximalwert der Schichtspitzen soll neben der Zeit minimiert werden. Abb. 4.15 zeigt die zum Kriterium gehörige Bewertungsfunktion (links) und die Straffunktion (rechts), die bei einem Überschreiten von maximal zwölf Mitarbeitern pro Schicht aktiviert wird. Die Normierungsfunktion wurde hier aus Geradensegmenten zusammengesetzt, um die Werte für ganze Mitarbeiter besser vorgeben zu können. Das Ziel liegt bei einem Maximum von neun Mitarbeitern bei allen Schichten. Der Erfüllungswert spielt insofern keine Rolle, da es keine Kriterien geringerer Priorität gibt.

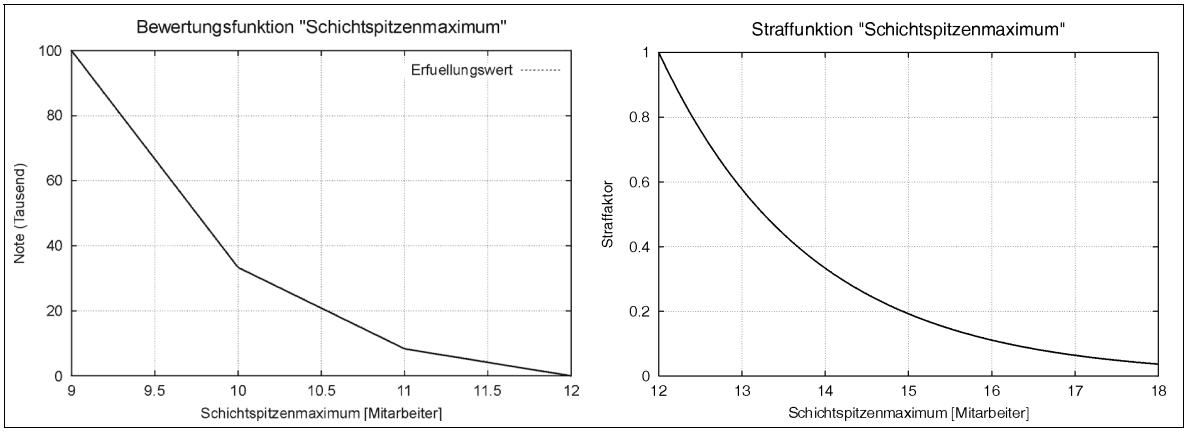


Abb. 4.15: Bewertungskriterium „Schichtspitzenmaximum“ (links) mit Straffunktion (rechts)

- Schichtspitzenüberhang
  Das Hilfskriterium *Schichtspitzenüberhang* erfaßt alle Werte, die den Zielwert des Schichtspitzenmaximums überschreiten. Es stellt die Fläche dar, die durch die Schichten mit Überschreitung des Schichtspitzenmaximums und dem Wert ihrer jeweiligen Überschreitung gebildet wird. Die Verkleinerung der Fläche ist für den EA der Weg zu einer Reduzierung des Schichtspitzenmaximums. Das Hilfskriterium erleichtert damit dem EA die Reduzierung von Schichtspitzenüberschreitungen, bis schließlich der Zielwert erreicht wird. Abb. 4.16 zeigt die Bewertungsfunktion und den Erfüllungswert.

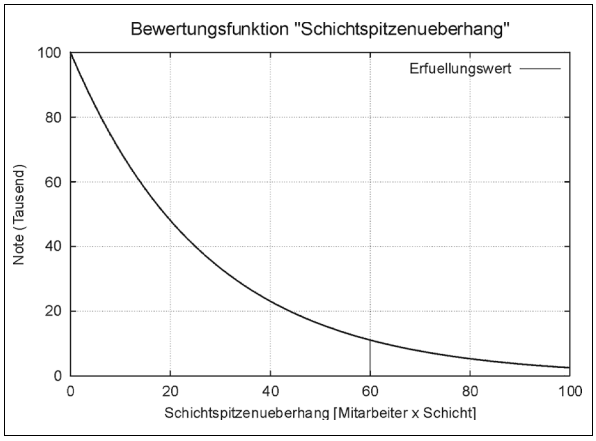


Abb. 4.16: Bewertungskriterium „Schichtspitzenüberhang“

- Terminverzug
  Der Terminverzug einzelner Aufträge gegenüber den vorgegebenen Endterminen wird nur als Straffunktion bewertet, siehe Abb. 4.17. Daher hat das Kriterium auch die Gewichtung Null in Tabelle 4.3. Die Einhaltung aller Endtermine bringt also keine Verbesserung der Gesamtfitness sondern verhindert lediglich ihre Abwertung. Die Funktion ist wie alle Straffunktionen als Exponentialfunktion ausgeführt, um auch noch große Zeiten erfassen zu können.

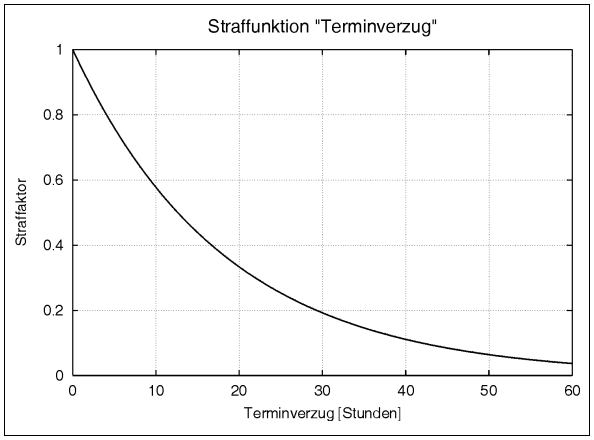


Abb. 4.17: Straffunktion „Terminverzug“

- Minderproduktion
  Durch eine ungeschickte Planung der Chargenstarts kann es bei Verfahrensketten passieren, daß nur eine ungenügende Menge des jeweiligen Vorprodukts vorhanden ist. Die dadurch entstehende Verringerung der Gesamtproduktionsmenge wird durch das Kriterium *Minderproduktion* erfaßt, das wie der Terminverzug lediglich als Straffunktion berücksichtigt wird, siehe Abb. 4.18.

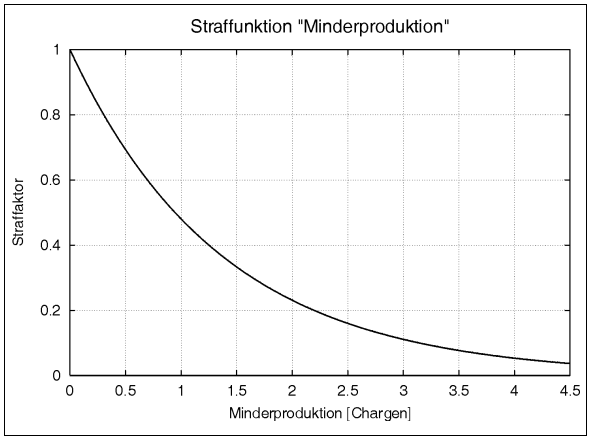


Abb. 4.18: Straffunktion „Minderproduktion“

4.4 Kollisionsfreie Roboterbahnplanung

Die Benchmarkaufgabe zur kollisionsfreien Roboterbahnplanung besteht darin, daß sich der Mitsubishi R500 auf einer möglichst geraden Linie von einer Start- zu einer Zielposition bewegen und dabei eine Kollision mit drei Hindernissen und mit sich selbst vermieden werden soll. Abb. 4.19 zeigt die drei Hindernisse sowie die Start- und die Zielposition. Der Roboter muß sich zuerst drehen, um dann zwischen den beiden großen Quadern die Abwärtsbewegung hinter die kleine Säule durchführen zu können. Die Bahnplanung wird mit dem in Abb. 2.2.3 erläuterten Aktionsmodell durchgeführt.

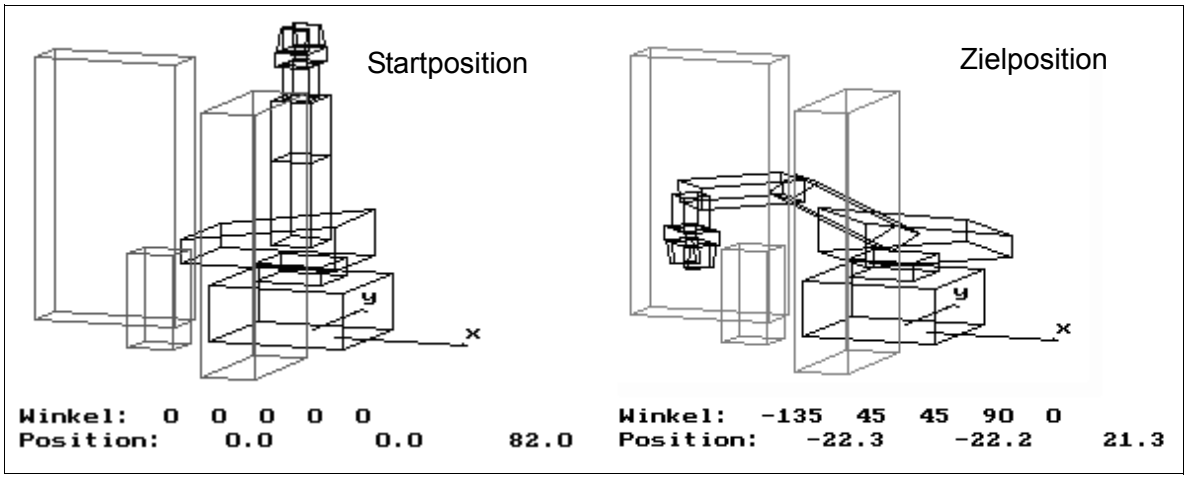


Abb. 4.19: Start- und Zielposition der Benchmarkaufgabe zur kollisionsfreien Roboterbahnplanung mit drei Hindernissen

Bei der Bewertung spielen neben den beiden Hauptkriterien der Zielabweichung und der Bahnabweichung noch zwei Hilfskriterien eine Rolle, siehe Tabelle 4.4:

Kriterium	Gewicht	Priorität	Straffunktion
Zielabweichung	40 %	1	nein
Bahnabweichung	55 %	1	nein
Restweg bei Kollision (Hilfskriterium)	0 %	2	ja
Aktionskettenlänge (Hilfskriterium)	5 %	2	ja

Tab. 4.4: Bewertungskriterien für die kollisionsfreie Roboterbahnplanung. Der Kollisionsrestweg wird nur als Straffunktion bewertet (Gewicht 0).

- Zielabweichung
  Bei der Bewertung wird nur das Erreichen der Zielposition des tool center points (TCP, Mittelpunkt zwischen den Greifbacken) und nicht seine Orientierung bewertet. Der Roboter kann also das Ziel auch aus anderen Richtungen anfahren als senkrecht von oben, wie in Abb. 2.2.3 dargestellt. Der verbleibende Abstand zum Ziel wird in Prozent des Gesamtwegs ausgedrückt, um die Bewertung unabhängig von der konkreten Aufgabenstellung zu machen. Abb. 4.20 zeigt die zugehörige Normierungsfunktion. Bei weniger als 30% Abweichung gilt das Kriterium in dem Sinne als erfüllt, daß die beiden Hilfskriterien mitbewertet werden.

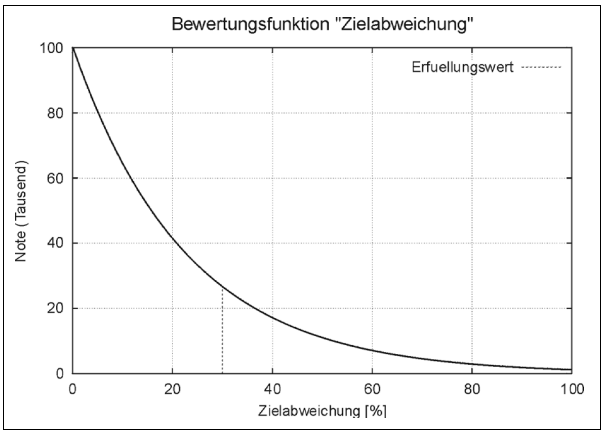


Abb. 4.20: Bewertungskriterium „Zielabweichung“

- Bahnabweichung
  Die Bahnabweichung mißt die Abweichung von der Geraden zwischen Start- und Zielpunkt und setzt sie in Relation zu einer angenommenen maximalen Abweichung. Der sich daraus ergebende Prozentwert bewertet demnach zu große Bahnabweichungen einheitlich mit 100% Abweichung. Das ist insofern zulässig, als die Bahn erst ab einer bestimmten Annäherung an das Ziel interessant wird. Da im konkreten Fall die ideale Gerade aus kinematischen Gründen nicht möglich ist, wird eine Bahnabweichung von 10% toleriert, wie Abb. 4.21 zeigt. Das Kriterium gilt im Sinne der Prioritätensteuerung bei Abweichungen von weniger als 60% als erfüllt.

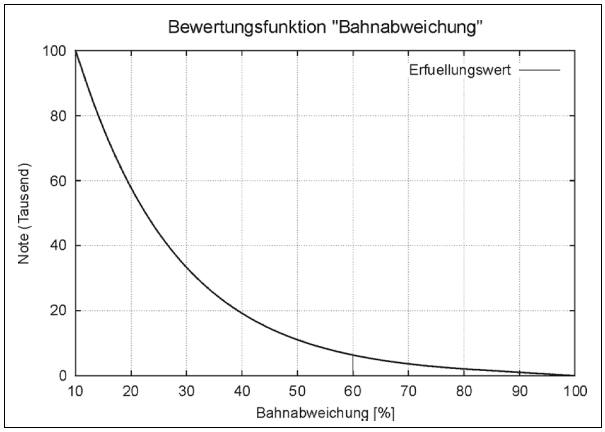


Abb. 4.21: Bewertungskriterium „Bahnabweichung“

- Restweg bei Kollision
  Bei einer Kollision wird der Restweg zum Ziel gemessen und als Straffunktion in die Gesamtbewertung integriert, siehe Abb. 4.22. Da das Kriterium erst ab einer Zielabweichung von weniger als 30 % Berücksichtigung findet, genügt eine Bewertung geringer Restwege. Für die konkrete Aufgabenstellung bedeutet das, daß Kollisionen mit der kleinen Säule differenziert bewertet werden, während Zusammenstöße mit den beiden Quadern nur durch die große Zielabweichung in die Fitness eingehen.

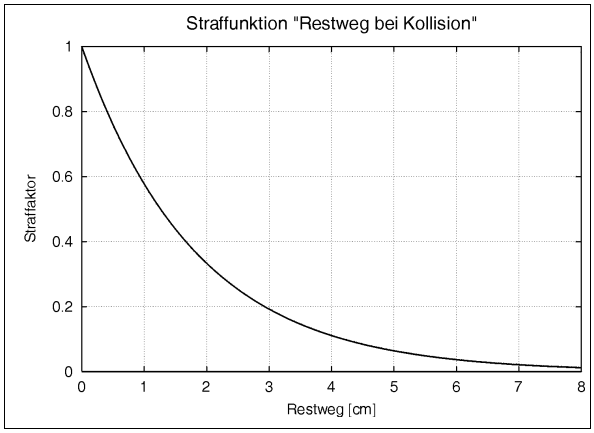


Abb. 4.22: Straffunktion „Restweg bei Kollision“

- Aktionskettenlänge
  Bei dynamischen Ketten sollte ihre Länge immer zu einem geringen Prozentsatz mitbewertet werden, da sonst die Gefahr der Entstehung extrem langer Ketten durch gar nicht oder nur wenig wirksame Aktionssequenzen besteht. Im konkreten Fall können das z.B viele „Motor aus“-Aktionen sein, die ausgeschaltete Motoren betreffen. Die Bewertung erfolgt mit Hilfe der in Abb. 4.23 dargestellten Bewertungs- und Straffunktion. Ketten bis zu einer Länge von 50 Aktionen erhalten den vollen Notenwert, der ab 70 bis zu einer Länge von 200 stark abnimmt. Ab einer Länge von 500 wird die positive Bewertung durch die Straffunktion abgelöst. Da dieser Wert willkürlich ist und längere Ketten nicht in dem Maße unerwünscht sind, wie z.B. eine Minderproduktion bei der Ressourcenplanung, erfolgt die Abwertung nicht mit Hilfe einer Exponentialfunktion, sondern beginnt zunächst mit geringer Abwertung, wie die rechte Seite von Abb. 4.23 zeigt.

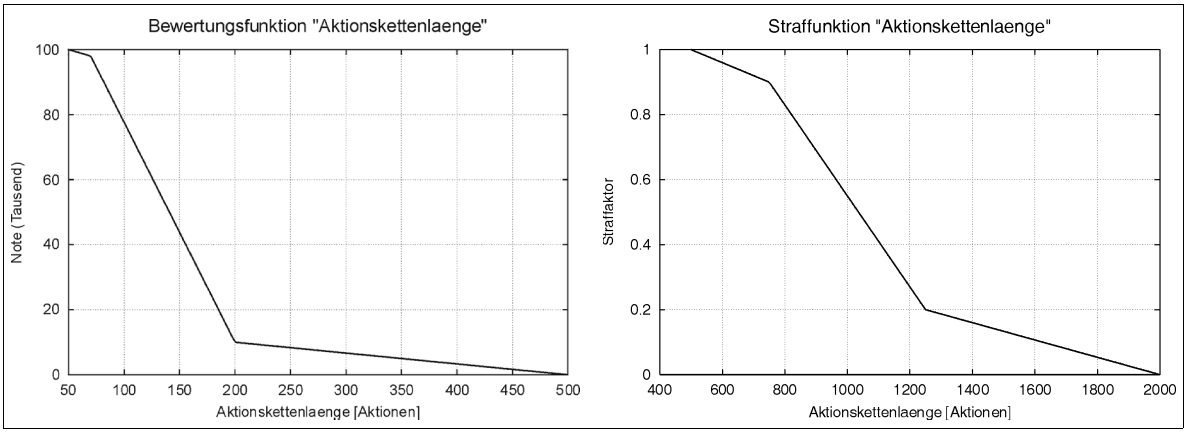


Abb. 4.23: Bewertungskriterium „Aktionskettenlänge“ (links) mit Straffunktion (rechts)

Bild Abb. 4.24 zeigt zwei Beispiele akzeptierter Lösungen der Bahnplanungsaufgabe. Das Ziel wurde mit geringen Abweichungen gut erreicht, während die Bahn des TCP in beiden Fällen noch verbesserungsfähig ist. Für die Verwendung der Aufgabe als Benchmark wurde jedoch auf eine strengere Bewertung und damit bessere Bahnqualität verzichtet, um den Rechenzeitaufwand in Grenzen zu halten.

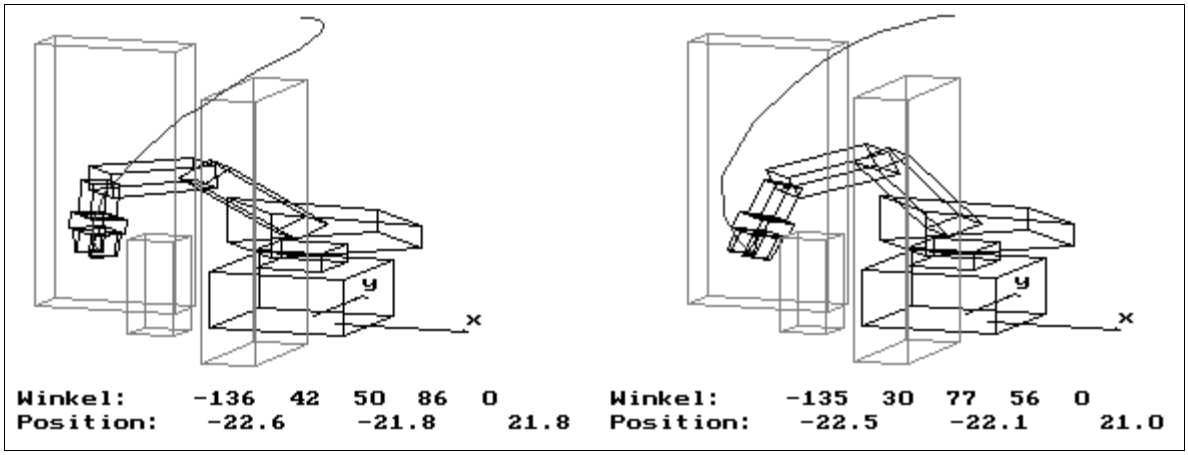


Abb. 4.24: Zwei Beispiele für ausreichende Lösungen der Testaufgabe „kollisionsfreie Roboterbahnplanung“

In diesem Kapitel wurden acht Benchmarkaufgaben, darunter fünf mathematische Benchmarkfunktionen und drei praktische Probleme, vorgestellt. Sie dienen als Grundlage für die in Kapitel 5 beschriebenen experimentellen Untersuchungen.

5. Experimentelle Untersuchungen

Auf Grund der umfangreichen Untersuchungsergebnisse wurde deren Präsentation in zwei Teile aufgeteilt:

1. Abschnitt 5.2 enthält eine detaillierte Auswertung der Ergebnisse zu den Einzelverfahren und Hybridisierungsarten samt deren Parametrierungen für alle Benchmarkaufgaben von Kapitel 4. In Abschnitt 5.2.1 werden zunächst die Resultate pro Benchmarkaufgabe dargelegt. Es folgt in Abschnitt 5.2.2 eine zusammenfassende Betrachtung gegliedert nach den Hybridisierungsarten. Auf die Problematik von Regelmäßigkeiten parallel zu den Koordinatenachsen einiger mathematischer Benchmarkfunktionen wird in Abschnitt 5.2.3 eingegangen, der auch die Ergebnisse zweier als Konsequenz aus den Regelmäßigkeiten gedrehter Testfunktionen enthält.
2. Die Ergebniszusammenfassung in Abschnitt 5.3 gibt zunächst noch einmal einen Überblick über die wesentlichen Ergebnisse aller Benchmarkfunktionen, bevor in den Unterabschnitten das Konvergenzverhalten (5.3.1) behandelt, eine zusammenfassende Auswertung (5.3.2) präsentiert und als Konsequenz Anwendungsempfehlungen (5.3.3) gegeben werden.

In den Schaubildern werden folgende Abkürzungen für die Verfahren und Hybridisierungen verwendet:

G	GLEAM
R	Rosenbrock-Verfahren
C	Complex-Algorithmus
Ri	GLEAM mit Rosenbrock-initialisierter Startpopulation
Ci	GLEAM mit Complex-initialisierter Startpopulation
NR	GLEAM plus Nachoptimierung mit dem Rosenbrock-Verfahren
NC1P	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei die EA-Ergebnisse jeweils einen Startpunkt (1P) für separate Complex-Läufe liefern.
NC1C	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei alle EA-Ergebnisse zur Bildung eines Startcomplexes (1C) für einen Complex-Lauf dienen.
GR	GLEAM mit direkter Integration des Rosenbrock-Verfahrens
GvR	GLEAM mit verzögerter direkter Integration des Rosenbrock-Verfahrens
GC	GLEAM mit direkter Integration des Complex-Algorithmus
GvC	GLEAM mit verzögerter direkter Integration des Complex-Algorithmus

Dazu kommen noch vier Kombinationen der Voroptimierung mit der verzögerten und der unverzögerten direkten Integration für beide LSV, die durch ein Hintereinanderschreiben der beteiligten Hybridisierungsarten, getrennt durch ein Komma, dargestellt werden. So bezeichnet beispielsweise GR, Ri die Voroptimierung und direkte Integration mit dem Rosenbrock-Verfahren.

Außerdem werden folgende Abkürzungen für die Parametrierungen benutzt:

p<anz>	Populationsgröße <anz>
n, m, h, u, v, s	Präzisionen (Abbruchschranken) für das Rosenbrock-Verfahren: n=niedrig, m=mittel, h=hoch, u=sehr hoch, v=extrem hoch, s=Sonderparametrierung. Die entsprechenden Werte sind im Anhang B, Tabelle B.2 enthalten.
<x>%	bei Ri oder Ci: Anteil der mit dem LSV voroptimierten Individuen an der Startpopulation in %
P1, P2, P3	Parametersätze 1 bis 3 zur Steuerung der Nachoptimierung, Werte siehe Anhang, Tabelle B.3
G1, G3	Überprüfung der Nischenbildung bei GDV- und GAk-Werten von 1 bzw. 3
all, best	Nachoptimierung aller oder nur des besten Nachkommens bei der direkten Integration
l100, l5, l0	Lamarckanteil 100%, 5% oder 0% (Baldwin-Evolution)

In den Schaubildern und Tabellen werden folgende Abkürzungen für die Benchmarkaufgaben verwendet:

Sphere	Schwefel's Sphere
Foxholes	Shekel's Foxholes
Rastrigin	verallgemeinerte Rastrigin Funktion
Fletcher	Fletcher's Function
Fraktal	Fraktale Funktion
Design	Designoptimierungsaufgabe Heterodynempfänger
Ressourcen	Ressourcenoptimierung in der Verfahrenstechnik
Roboter	Kollisionsfreie Roboterbahnplanung

5.1 Benutzte Software

GLEAM, das weitgehend in ANSI-C implementiert ist und unter Solaris (Unix-Variante) auf Sun-Workstations läuft, wurde um die in Kapitel 3 beschriebenen Hybridisierungsarten erweitert und mit Schnittstellen für die beiden lokalen Suchverfahren ausgestattet. Letztere wurden am Institut für Theoretische Elektrotechnik und Mikroelektronik der Universität Bremen implementiert und in GLEAM integriert. Als Simulatoren wurde für den Heterodynempfänger *Mathematica 3.0*[1] mit einem Modell von Sieber [Sie00], für die Ressourcenoptimierung eine Eigenimplementierung und für die Roboterbahnplanung ein Simulationsmodul von Blume [Blu98] verwendet.

Abb. 5.1 zeigt den Ablauf von GLEAM als Pseudocode. Wie jeder EA (vergleiche Abb. 1.2) besteht GLEAM aus zwei großen Schleifen. Die äußere umfaßt eine Generation und wird ausgeführt, bis das Abbruchkriterien erfüllt ist. Es setzt sich aus dem Zielkriterium der erreichten Fitness, zwei begrenzenden Kriterien, nämlich verbrauchter Zeit und erreichter Generationsanzahl, und den beiden Stagnationskriterien Generationen ohne Deme-Verbesserung (GDV) oder -Akzeptanz (GAk) zusammen. Die innere Schleife wird mit allen Individuen einer Population ausgeführt und enthält die Nachkommensbildung, Bewertung und Akzeptanz der Kinder. Alle nicht akzeptierten Nachkommen werden gelöscht. Das Bild dient auch als Referenz

1. Mathematica ist ein eingetragenes Warenzeichen der Firma Wolfram Research, Inc.

zur Darstellung der implementierten Ergänzungen und Änderungen für vier Hybridisierungsarten. Dabei werden die jeweils durchgeführten Neuerungen kursiv dargestellt.

```
Generiere Startpopulation aus eventuellen vorgegebenen Individuen und
ergänze durch zufällig erzeugte, die zu bewerten sind
REPEAT
  FOR (jedes Individuum der Population)
    Wähle Partner aus der Nachbarschaft durch Ranking
    Erzeuge eine vorgegebene Anzahl Nachkommen durch Mutation und Cross-
    over mit vorgegebenen Wahrscheinlichkeiten
    Bewerte die Nachkommen
    Wähle den besten Nachkommen aus und lösche den Rest
    Ausgewählter Nachkomme ersetzt das Elter gemäß Akzeptanzregel oder
    wird gelöscht
  END FOR
  Berechne Abbruchbedingung indem geprüft wird, ob eine der Bedingungen
  Zielfitness, Zeit, Generationen, Generationen ohne Deme-Verbesserung
  (GDV) oder -Akzeptanz (GAk) erfüllt ist
UNTIL Abbruchbedingung erfüllt
Bestimme das beste Ergebnis und weitere Ergebnisse wenn gefordert
```

Abb. 5.1: Pseudocode für GLEAM

```
Generiere Startpopulation aus eventuellen vorgegebenen Individuen und
ergänze durch zufällig erzeugte, die zu bewerten sind
Wende das LSV auf zufällig bestimmte Individuen an bis ein vorgegebener
Anteil an der Startpopulation erreicht ist
REPEAT
  FOR (jedes Individuum der Population)
    Wähle Partner aus der Nachbarschaft durch Ranking
    Erzeuge eine vorgegebene Anzahl Nachkommen durch Mutation und Cross-
    over mit vorgegebenen Wahrscheinlichkeiten
    Bewerte die Nachkommen
    Wähle den besten Nachkommen aus und lösche den Rest
    Ausgewählter Nachkomme ersetzt das Elter gemäß Akzeptanzregel oder
    wird gelöscht.
  END FOR
  Berechne Abbruchbedingung indem geprüft wird, ob eine der Bedingungen
  Zielfitness, Zeit, Generationen, Generationen ohne Deme-Verbesserung
  (GDV) oder -Akzeptanz (GAk) erfüllt ist
UNTIL Abbruchbedingung erfüllt
Bestimme das beste Ergebnis und weitere Ergebnisse wenn gefordert
```

Abb. 5.2: Pseudocode für GLEAM mit Voroptimierung

Abb. 5.2 zeigt die Erweiterung von GLEAM durch die Voroptimierung mit Hilfe eines der beiden lokalen Suchverfahren. Das Bild zeigt, daß die Veränderung nur die Vorbereitungsphase betrifft.

Der Eingriff für die Nachoptimierung ist dagegen schon aufwendiger, wie Abb. 5.3 zeigt. Die Anpassungen betreffen die Berechnung der Abbruchbedingung und die Bestimmung der Ergebnisindividuen gefolgt von der eigentlichen Nachoptimierung. Letztere ist im Bild nur summarisch dargestellt, da im Falle des Complex-Verfahrens noch zwischen der getrennten Verbesserung aller Ergebnisindividuen durch separate Complex-Läufe und der Bildung eines Startpolyeders unter Verwendung aller Ergebnisse zur Initialisierung eines Complex-Laufs unterschieden werden muß.

```
Generiere Startpopulation aus eventuellen vorgegebenen Individuen und
ergänze durch zufällig erzeugte, die zu bewerten sind
REPEAT
  FOR (jedes Individuum der Population)
    Wähle Partner aus der Nachbarschaft durch Ranking
    Erzeuge eine vorgegebene Anzahl Nachkommen durch Mutation und Cross-
    over mit vorgegebenen Wahrscheinlichkeiten
    Bewerte die Nachkommen
    Wähle den besten Nachkommen aus und lösche den Rest
    Ausgewählter Nachkomme ersetzt das Elter gemäß Akzeptanzregel oder
    wird gelöscht
  END FOR
  IF (GDV- oder GAk-Limit erreicht) THEN
    Ermittle Nischenbildung mit den Kenngrößen Nmax und EpsilonPop
  END IF
  Berechne Abbruchbedingung indem geprüft wird, ob eine der Bedingungen
  Zielfitness, Zeit, Generationen, Nischenbildung oder Generationen ohne
  Deme-Verbesserung (GDV) oder -Akzeptanz (GAk) erfüllt ist
UNTIL Abbruchbedingung erfüllt
IF (Abbruch wegen Nischenbildung) THEN
  Ergebnis sind die Nischenrepräsentanten und das beste Individuum
ELSE
  Ergebnis ist das beste Individuum
END IF
IF (NOT Abbruch wegen Erreichung der Zielfitness) THEN
  Nachoptimierung aller Ergebnisindividuen mit dem LSV
END IF
Bestimme das beste Ergebnis
```

Abb. 5.3: Pseudocode für GLEAM mit Nachoptimierung

Bei der direkten Integration wird in den Prozeß der Nachkommensbildung eingegriffen, wie Abb. 5.4 zeigt. Nach der Bewertung der Nachkommen wird je nach Strategie der beste oder alle lokal verbessert, bevor der resultierende beste Offspring mit seinem Elter gemäß der Akzeptanzregel um das Überleben kämpfen muß.

```
Generiere Startpopulation aus eventuellen vorgegebenen Individuen und
ergänze durch zufällig erzeugte, die zu bewerten sind
REPEAT
  FOR (jedes Individuum der Population)
    Wähle Partner aus der Nachbarschaft durch Ranking
    Erzeuge eine vorgegebene Anzahl Nachkommen durch Mutation und Cross-
    over mit vorgegebenen Wahrscheinlichkeiten
    Bewerte die Nachkommen
    IF (lokale Optimierung aller Nachkommen) THEN
      Wende LSV auf alle Nachkommen an
      Wähle den besten verbesserten Nachkommen aus und lösche den Rest
    ELSE
      Wähle den besten Nachkommen aus und lösche den Rest
      Wende LSV auf den ausgewählten Nachkommen an
    END IF
    Ausgewählter Nachkomme ersetzt das Elter gemäß Akzeptanzregel oder
    wird gelöscht
  END FOR
  Berechne Abbruchbedingung indem geprüft wird, ob eine der Bedingungen
  Zielfitness, Zeit, Generationen, Generationen ohne Deme-Verbesserung
  (GDV) oder -Akzeptanz (GAk) erfüllt ist
UNTIL Abbruchbedingung erfüllt
Bestimme das beste Ergebnis und weitere Ergebnisse wenn gefordert
```

Abb. 5.4: Pseudocode für die direkte Integration eines LSVs in GLEAM

Im Falle der verzögerten direkten Integration kommt noch die Steuerung für die Zuschaltung der lokalen Offspring-Verbesserung hinzu. Abb. 5.5 zeigt die dazu notwendigen Erweiterungen. Anfänglich, das heißt solange *lsv_zugeschaltet* den Wert *FALSE* hat, läuft alles so wie beim unveränderten GLEAM ab. Am Ende einer Generation wird der Nischenbildung ermittelt und bei erfüllter Nischenbedingung *lsv_zugeschaltet* auf *TRUE* gesetzt. Letzteres bewirkt ab der nächsten Generation den Wechsel zur direkten Integration und verhindert die weitere Prüfung der Nischenbildung.

```
Generiere Startpopulation aus eventuellen vorgegebenen Individuen und
ergänze durch zufällig erzeugte, die zu bewerten sind
lsv_zugeschaltet = FALSE
REPEAT
  FOR (jedes Individuum der Population)
    Wähle Partner aus der Nachbarschaft durch Ranking
    Erzeuge eine vorgegebene Anzahl Nachkommen durch Mutation und Cross-
    over mit vorgegebenen Wahrscheinlichkeiten
    Bewerte die Nachkommen
    IF (lsv_zugeschaltet) THEN
      IF (lokale Optimierung aller Nachkommen) THEN
         Wende LSV auf alle Nachkommen an
         Wähle den besten verbesserten Nachkommen aus und lösche den Rest
      ELSE
         Wähle den besten Nachkommen aus und lösche den Rest
         Wende LSV auf den ausgewählten Nachkommen an
      END IF
    ELSE
      Wähle den besten Nachkommen aus und lösche den Rest
    END IF
    Ausgewählter Nachkomme ersetzt das Elter gemäß Akzeptanzregel oder
    wird gelöscht
  END FOR
  IF ((GDV- oder GAk-Limit erreicht) AND NOT lsv_zugeschaltet) THEN
    Ermittle Nischenbildung mit den Kenngrößen Nmax und EpsilonPop
    IF (Nischenbedingung erfüllt) THEN
       lsv_zugeschaltet = TRUE
    END IF
  END IF
  Berechne Abbruchbedingung indem geprüft wird, ob eine der Bedingungen
  Zielfitness, Zeit, Generationen, Generationen ohne Deme-Verbesserung
  (GDV) oder -Akzeptanz (GAk) erfüllt ist
UNTIL Abbruchbedingung erfüllt
Bestimme das beste Ergebnis und weitere Ergebnisse wenn gefordert
```

Abb. 5.5: Pseudocode für die verzögerte direkte Integration eines LSVs in GLEAM

5.2 Experimente und ihre Ergebnisse

Die für die experimentellen Untersuchungen notwendigen Läufe wurden auf bis zu 19 Sun-Workstations der Typen Ultra-Sparc 1, 2, 5 und 10 mit insgesamt 22 Prozessoren und Taktraten zwischen 140 und 440 MHz verteilt. Der gesamte Rechenzeitaufwand aller Läufe für die in Kapitel 4 aufgeführten Benchmarkaufgaben betrug 9.2 CPU-Jahre, Details siehe Anhang B.

Die Bewertung erfolgt anhand der beiden Kriterien *Erfolgsrate* und *Aufwand*. Die Erfolgsrate ist der Quotient aus den Läufen, die das Optimierungsziel erreichen, und der Gesamtzahl der Läufe. Der Aufwand wird an der Anzahl der Evaluationen gemessen, da das angesichts der unterschiedlichen Leistung der verwendeten Workstations und deren Benutzung durch ihre Besitzer während der Tagesstunden objektiver ist als die Bearbeitungszeit. Dadurch entsteht insofern eine Verfälschung, als die Komplexität der beiden lokalen Verfahren im Gegensatz zur linearen von GLEAM beim Complex-Algorithmus $O(n^2)$ und beim Rosenbrock-Verfahren $O(n^3)$ beträgt. Je nach der Parameteranzahl der Aufgabe und Verfahren ist also der Zusatzaufwand durch das LSV und damit der Zeitbedarf pro Evaluation unterschiedlich. Die Bedeutung dieses Zusatzaufwands sinkt allerdings bei Problemen mit vergleichsweise langen Bewertungszeiten.

Im Verlauf der Experimente stellte sich heraus, daß es auch sinnvoll ist, die Zeit zu erfassen und normiert auf die Taktrate einer Ultra-Sparc 1 mit 170 MHz anzugeben. Die Werte erlauben einen ungefähren Vergleich und dienen zum Beleg, wann bei bestimmten Aufgaben und Verfahrenseinstellungen der Zeitbedarf auf Grund schlechter Konvergenz des lokalen Verfahrens so erheblich steigt, daß detailliertere Untersuchungen sich verbieten.

Der sprachlichen Einfachheit halber wird unter dem Begriff *Job* ein Verfahren oder eine Integrationsart mit konkreter Parametrierung verstanden. Auf Grund der stochastischen Natur des EAs oder der Startpunktwahl bei den lokalen Verfahren wurden pro Aufgabe und Job in der Regel 100 Läufe durchgeführt und davon der Mittelwert der Evaluationen genommen. Wegen längerer Laufzeiten wurde die Grenze auf 50 in nachstehenden Fällen reduziert:

- Schwefel's Sphere: direkte Integration mit dem Rosenbrock-Verfahren bei allen anderen Parametrierungen als LSV-Optimierung des besten Nachkommen und Lamarckanteil 100%, alle Complex-Jobs
- Designoptimierung des Heterodynempfängers, außer bei den unkombinierten Verfahren
- Ressourcenplanung mit Ausnahme der reinen GLEAM-Jobs
- Roboterbahnplanung: Nachoptimierung mit dem Rosenbrock-Verfahren, voroptimierte Startpopulationen mit dem Rosenbrock-Verfahren.

Wenn die Grenzen wegen extrem langer Laufzeiten weiter reduziert wurden, wird darauf bei der Behandlung der einzelnen Aufgaben gesondert hingewiesen.

In den Fällen, in denen eine von 100% abweichende Erfolgsrate angegeben werden muß, wird die Rate durch ein *E(<Verfahrensangaben>)* bezeichnet. *E(Ci,p50,20%)* ist also die Erfolgsrate eines GLEAM-Jobs mit einer Populationsgröße von 50 und einer zu 20% mit dem Complex-Algorithmus voroptimierten Startpopulation. Entsprechend wird bei der Darstellung von Noten verfahren: *No(...)* bezeichnet die Durchschnittsnote eines Jobs und *MinNo(...)* die kleinste erreichte Note.

Bis auf Schwefel's Sphere wird der beste GLEAM-Job als Vergleich herangezogen. In den Schaubildern wird der Vergleichswert entweder numerisch angegeben oder durch eine gestrichelte Linie dargestellt. Die Ergebnisse werden in diesem Kapitel in der Regel in Form von Diagrammen dargestellt. Das ausführliche Zahlenmaterial ist in Anhang B enthalten.

5.2.1 Ergebnisse der einzelnen Benchmarkaufgaben

Bei Untersuchungen zum Einfluß variierender Populationsgrößen auf Erfolgsrate und Aufwand einer EA-Anwendung ergeben sich die in Abb. 5.6 dargestellten typischen Verläufe. Die entsprechenden Schaubilder bei der Designoptimierungsaufgabe oder der Roboterbahnplanung sind gute Beispiele dafür, siehe Abb. 5.78 und Abb. 5.111. Wird die Populationsgröße zu klein gewählt, steigt die Gefahr, daß der EA vorzeitig bei einem Suboptimum konvergiert. Ab einer bestimmten, aufgabenabhängigen Größe erreicht der EA das Optimum zuverlässig bezogen auf den Stichprobenumfang, also der Anzahl der Läufe pro Job. Bei einem weiteren Ansteigen der Populationsgröße steigt die Wahrscheinlichkeit, daß Läufe mit überdurchschnittlich großem Aufwand ausbleiben und es kommt häufig zu einer Verminderung des durchschnittlichen Aufwands. Mit weiter steigendem Umfang der Population werden vergleichsweise unnötig viele Individuen betrachtet und der Aufwand beginnt wieder langsam zu steigen. Es gibt also einen aufgabenspezifischen Bereich günstiger Populationsgrößen, eine Art *Arbeitsbereich* des EAs.

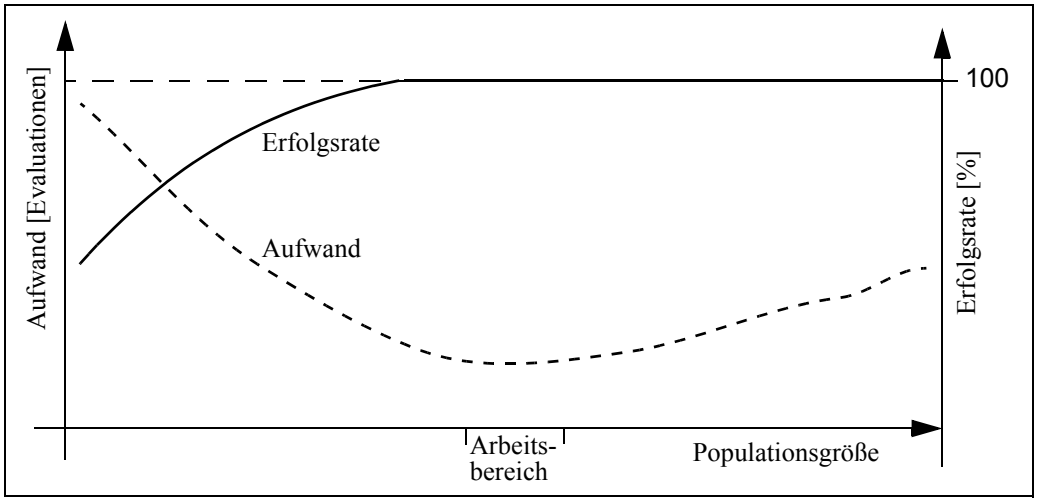


Abb. 5.6: Typischer Verlauf von Aufwand und Erfolgsrate bei steigender Populationsgröße

Wie nachfolgend dargelegt, zeigen nicht alle Benchmarkaufgaben dieses Verhalten. Das hat zwei unterschiedliche Ursachen. Zum einen wirken die Stagnationsindikatoren GDV und GAk bisweilen und bewirken einen Abbruch der Läufe bevor das meist recht hoch gesetzte Generationslimit erreicht wird. Das ist aus Gründen der Aufwandsbegrenzung auch sinnvoll, liefert aber sinkenden Aufwand zusammen mit sinkenden Erfolg bei abnehmenden Populationsgrößen. Die zweite Ursache kann daran liegen, daß die Aufgabe „zu einfach" für den EA ist. Das äußert sich darin, daß sie auch noch bei extrem kleinen Populationsgrößen von fünf oder zehn zuverlässig gelöst wird und der Aufwand bei abnehmender Populationsgröße kontinuierlich sinkt.

Eine Variation der Abbruchschranken des Rosenbrock-Verfahrens hat einen ähnlichen Effekt: Bei zu kleiner Präzision besteht die Gefahr eines Abbruchs vor Erreichung des Optimums,

womit die Erfolgsrate sinkt. Andererseits bedeutet eine Erhöhung der Präzision auch mehr Iterationen und damit steigenden Aufwand. Bei zu großen Präzisionen kann der Effekt beobachtet werden, daß das Verfahren nicht mehr konvergiert und abgebrochen werden muß.

Die nachfolgenden Untersuchungen müssen zeigen, ob der durch die Hybridisierung verursachte Mehraufwand durch das LSV zu einer verbesserten Leistung des Gesamtalgorithmus führt oder nicht.

5.2.1.1 Schwefel's Sphere

Die Aufgabe wird auch in ihrer etwas entschärften Form (Zielwert=0.01 statt 0.0) von GLEAM nicht gelöst: Zwei Läufe wurden nach 330000 bzw. 390000 Generationen mit einem Ergebnis im vierstelligen Bereich abgebrochen. Das Rosenbrock-Verfahren versagt ebenfalls bei üblichen Präzisionen. Bei extrem hoher Präzision (*v*), die bei allen anderen Aufgaben eine Nichtkonvergenz zur Folge hat, konvergiert das Verfahren allerdings mit durchschnittlich 6440 Evaluationen, siehe Abb. 5.7. Der Complex-Algorithmus kann das Problem nicht lösen.

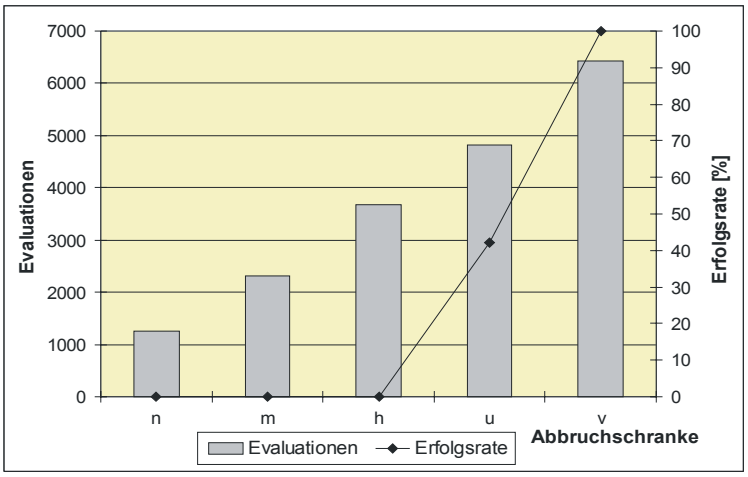


Abb. 5.7: Rosenbrock-Verfahren: Vergleich unterschiedlicher Präzisionen (Abbruchschranken). (Sphere)

Auch die Voroptimierung der Startpopulationen von GLEAM bringt wenig Erfolg: Beim Rosenbrock-Verfahren und hoher Präzision werden Erfolgsraten von zehn Prozent bei 22.2 und 33.6 Millionen Evaluationen erreicht (Populationsgröße 30 und 50, jeweils 10 Läufe). Beim Complex-Algorithmus ergab sich für Populationsgrößen von 10, 20 und 30 kein Erfolg bei 4.8 - 26.1 Evaluationen (jeweils 10 Läufe).

Günstiger sieht es bei der Nachoptimierung mit dem Rosenbrock-Verfahren aus: Abb. 5.8 zeigt die Ergebnisse für die drei Standardparametrierungen und beide GDV/GAk-Werte. Die Jobs wurden mit Präzision *u* durchgeführt, da bei *v* das Rosenbrock-Verfahren bereits allein sicher konvergiert. Alle Parametrierungen liefern im wesentlichen Erfolgsraten zwischen 70 und 85% mit Ausnahme von Job *NR,P3,G1*, der ab einer Populationsgröße von 30 regelmäßig über 85% liegt.

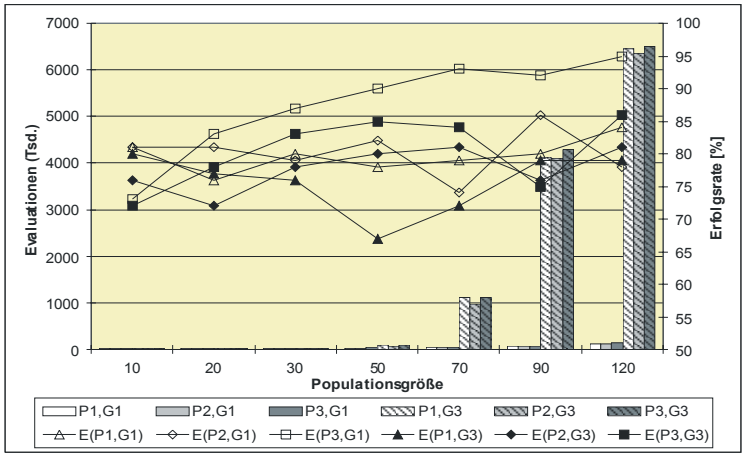


Abb. 5.8: Nachoptimierung mit dem Rosenbrock-Verfahren bei sehr hoher Präzision (u). (Sphere)

Abb. 5.9 beantwortet die Frage, ob sich dieser Wert durch größere Populationen eventuell noch steigern läßt. Das ist nicht der Fall, die Erfolgsrate sinkt auf die bei den anderen Parametrierungen üblichen Werte zurück. Abb. 5.10 zeigt, daß auch eine Verkleinerung der anfänglichen Schrittweite von 1/10 auf 1/100 und 1/1000 des Wertebereichs keine wesentliche Verbesserung bringt.

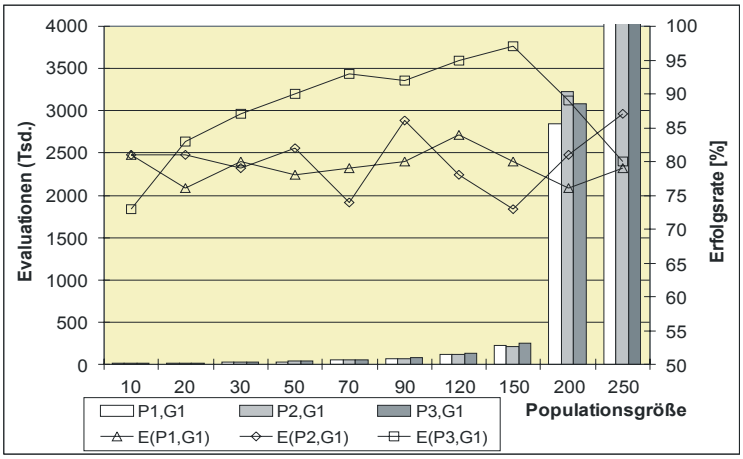


Abb. 5.9: Nachoptimierung mit dem Rosenbrock-Verfahren und Präzision u bei größeren Populationen. (Sphere)

Die erreichte Qualität ist allerdings auch bei Nichterfolg sehr gut. In Abb. 5.11 sind der beste und der schlechteste Job hinsichtlich der kleinsten erreichten Note dargestellt. Selbst die schlechtesten Läufe erreichen, von einer Ausnahme abgesehen, Notenwerte von 99998.2 oder besser, was einem Zielfunktionswert von 450 oder kleiner entspricht. Der in dieser Hinsicht günstigste Job ist *NR,p10,x,P2,G3*, der bei 14118 Evaluationen einen Notenschnitt von 100000 bei einem Minimum von 99999.3 (entspricht einem Zielfunktionswert von 175) liefert. Damit zeigt die Nachoptimierung, daß sie bei der vorliegenden Aufgabe bei vergleichsweise geringem Aufwand dem Ziel in der Regel sehr nahe kommt.

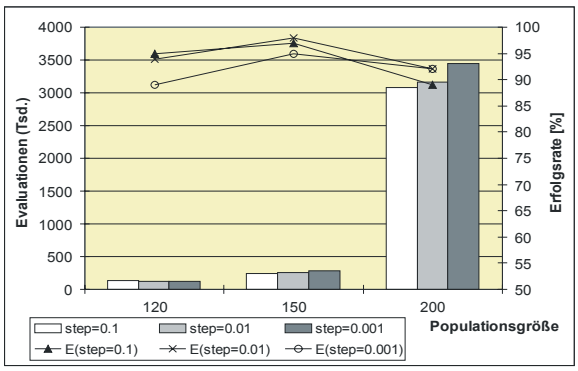


Abb. 5.10: Rosenbrock-Nachoptimierung: Vergleich des Aufwands und der Erfolgsrate bei Präzision x und verschiedenen Schrittweiten. (Sphere)

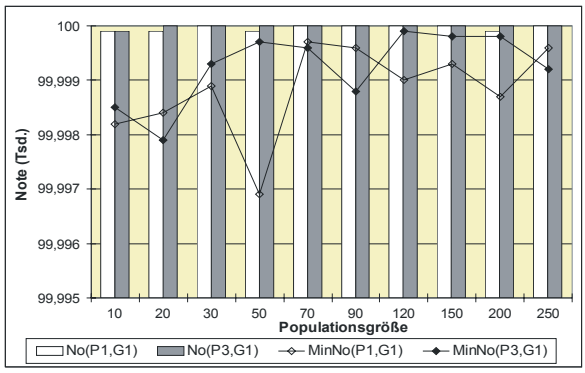


Abb. 5.11: Rosenbrock-Nachoptimierung: Vergleich des besten und des schlechtesten Jobs hinsichtlich der erreichten Mindestnoten (MinNo). No(...) gibt die Durchschnittsnote an. (Sphere)

Die Nachoptimierung mit dem Complex-Algorithmus war bei keiner Parametrierung erfolgreich. Getestet wurden Populationsgrößen von 10, 20, 30 und 50 mit allen drei Standardparametrierungen und beiden GDV/GAk-Werten, siehe auch Anhang B.1.3.

Anders sieht es bei der direkten Integration mit dem Rosenbrock-Verfahren aus. Bis auf die Jobs mit einer Lamarckrate von 0% (Baldwin-Evolution) haben alle eine Erfolgsrate von 100%. Abb. 5.12 zeigt den Effekt einer Verringerung der Lamarckrate und der lokalen Optimierung aller Nachkommen beim Rosenbrock-Verfahren bei niedriger Präzision. Alle Kombinationen niedriger Lamarckraten mit *all* oder *best* bei der Nachkommensverbesserung bringen schlechtere Ergebnisse oder mußten wegen Erfolglosigkeit abgebrochen werden. Daher

wurden alle weiteren Jobs mit 100% Lamarckrate und lokaler Optimierung des besten Nachkommen durchgeführt. In Abb. 5.13 sind die Ergebnisse der Variation der Präzision wiedergegeben. Die Überlegenheit der hohen Präzision gegenüber allen anderen ist deutlich zu erkennen.

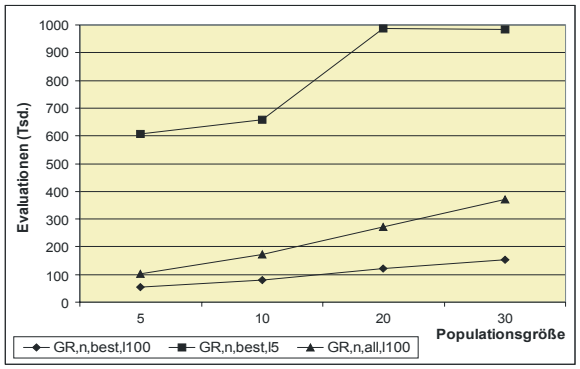


Abb. 5.12: Direkte Integration des Rosenbrock-Verfahrens bei niedriger Präzision. (Sphere)

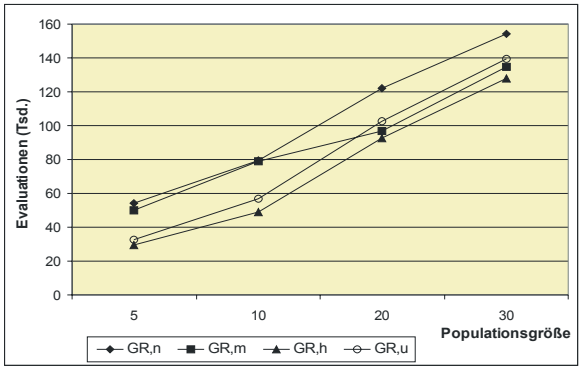


Abb. 5.13: Direkte Rosenbrock-Integration: Vergleich der Präzisionen n, m, h und u(bester Nachkomme und 100% Lamarckanteil).

Die Kombination von voroptimierter Startpopulation und direkter Integration bringt keine Vorteile, wie Abb. 5.14 für mittlere und hohe Präzision zeigt. Die Kurven der Jobs ohne Voroptimierung liegen zum Teil erheblich unter denen mit Voroptimierung.

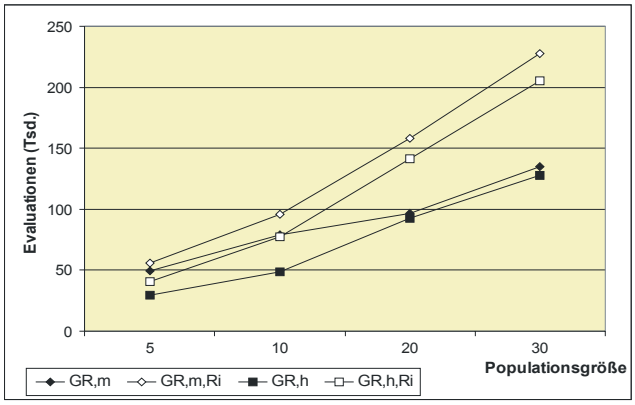


Abb. 5.14: Direkte Rosenbrock-Integration: Vergleich von zufälligen und voroptimierten Startpopulationen bei mittlerer und hoher Präzision (bester Nachkomme, 100% Lamarckanteil). (Sphere)

Bei der verzögerten direkten Integration spielt die Parametrierung für die Verzögerung so gut wie keine Rolle, wichtig ist vielmehr die Präzision wie Abb. 5.15 zeigt. Auch schneidet die hohe Präzision (*h*) etwas besser ab als die sehr hohe (*u*). Abb. 5.16 faßt die besten Ergebnisse nochmal für niedrige Populationsgrößen zusammen, um die Unterschiede deutlicher werden zu lassen. Die direkte Integration liefert

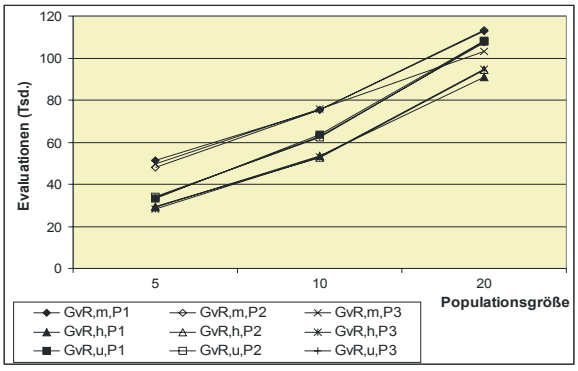


Abb. 5.15: Verzögerte direkte Rosenbrock-Integration: Vergleich verschiedener Präzisionen und Verzögerungen (bester Nachkomme, 100% Lamarckanteil). (Sphere)

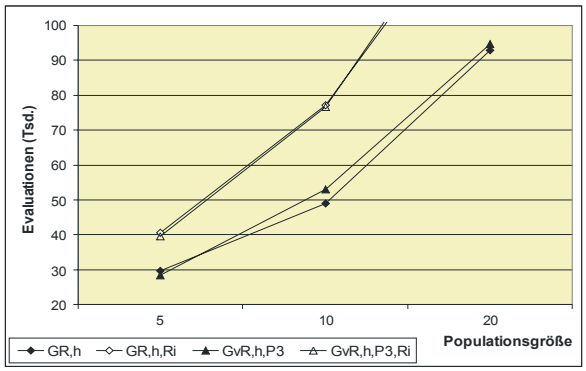


Abb. 5.16: Beste Jobs der verzögerten und unverzögerten direkten Rosenbrock-Integration (bester Nachkomme, 100% Lamarckanteil). (Sphere)

bei hoher Präzision sowohl in der verzögerten als auch in der unverzögerten Variante ohne voroptimierte Startpopulationen die besten Ergebnisse.

Da ein Complex-Lauf ca. 2.2 Stunden benötigt, werden Jobs mit direkt integriertem Complex-Algorithmus nicht als sinnvoll erachtet.

Bild Abb. 5.17 faßt die besten Ergebnisse aller Integrationsarten mit 100% Erfolgsrate zusammen und vergleicht sie mit dem erfolgreichen Job des Rosenbrock-Verfahrens und der fast erfolgreichen Rosenbrock-Nachoptimierung, die die geforderte Zielqualität nahezu erreicht hat. Im Vergleich zu dem enormen Aufwand der erfolglosen GLEAM-Jobs und Complex-Hybridisierungen liegen die Aufwände der beiden besten Jobs der verzögerten und unverzögerten direkten Rosenbrock-Integration in einer vergleichbar niedrigen Größenordnung wie der erfolgreiche Rosenbrock-Job. Bei der Beurteilung der Unterschiede ist die bereits erwähnte extrem hohe Präzision zu berücksichtigen, die beim Rosenbrock-Verfahren erforderlich ist und bei der gewöhnlich eine Nichtkonvergenz des Verfahrens eintritt. So gesehen können die beiden Hybridisierungen durchaus mit dem Rosenbrock-Verfahren konkurrieren. Auch die Rosenbrock-Nachoptimierung schneidet vergleichsweise gut ab: Eine schnelle Ermittlung von Resultaten, die dem Ziel sehr nahe kommen, ist oft besser als eine vergleichsweise langwierige Bestimmung des Optimums.

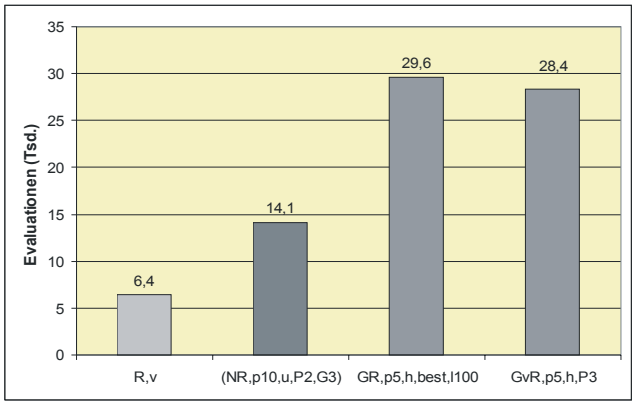


Abb. 5.17: Gesamtvergleich der besten Jobs je Integrationsart mit (fast) 100% Erfolgsrate. (Sphere)

5.2.1.2 Shekel's Foxholes

Diese noch recht einfache multimodale Aufgabe wird von GLEAM problemlos gelöst, wie Abb. 5.18 zeigt. Auffällig ist, daß eine Verkleinerung der Populationsgröße auch regelmäßig zu einer Verringerung des Aufwands führt und selbst extrem kleine Populationen (bis zu 10 Individuen) noch erfolgreich sind. In Abschnitt 5.2.1 wurde bereits darauf hingewiesen, daß ein solches Verhalten darauf hindeutet, daß die Aufgabe für GLEAM einfach ist. Auf die Problematik wird in Abschnitt 5.2.3 noch einmal eingegangen werden. Da GLEAM mit 1437 Evaluationen bei einer Populationsgröße von fünf bereits eine sehr niedrige Marge vorgibt, ist mit größeren Einsparungen durch die Hybridisierung nicht zu rechnen. Beides mindert die Aussagekraft der Benchmarkaufgabe.

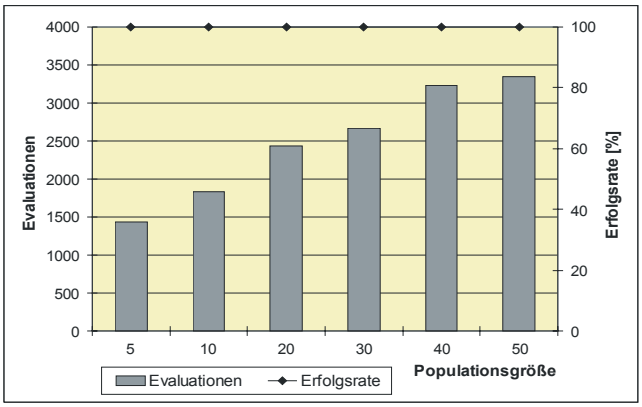


Abb. 5.18: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Foxholes)

Die beiden lokalen Verfahren können mit einer Erfolgsrate von 3% beim Rosenbrock-Verfahren und hoher Präzision bzw. 1% beim Complex-Algorithmus die Aufgabe im wesentlichen

nicht lösen. Bei Verwendung der sehr hohen Präzision *u* konvergiert das Rosenbrock-Verfahren nicht mehr.

Voroptimierte Startpopulationen reduzieren beim Rosenbrock-Verfahren den Aufwand auf 63% des GLEAM-Aufwands, bringen jedoch beim Complex-Algorithmus keinen Erfolg. Abb. 5.19 gibt einen Überblick über die drei möglichen Präzisionen bei variierendem Anteil an voroptimierten Individuen an der Startpopulation. Da sich der interessantere Teil im unteren Aufwandsbereich abspielt, zeigt Abb. 5.20 davon einen Ausschnitt.

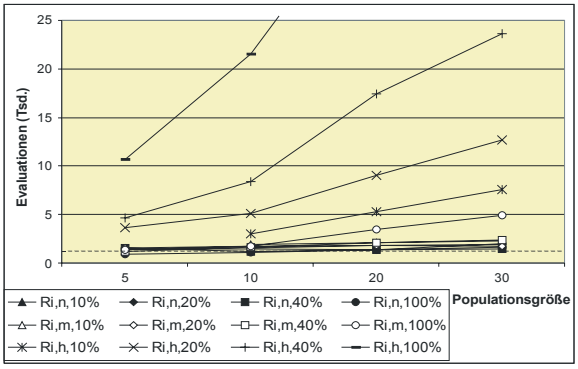


Abb. 5.19: GLEAM mit Rosenbrock-Voroptimierung bei den drei möglichen Präzisionen und verschiedenen Anteilen an voroptimierten Individuen an der Startpopulation. (Foxholes)

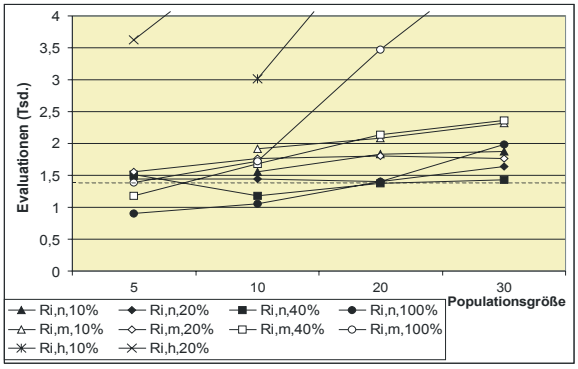


Abb. 5.20: Detailansicht von Abb. 5.19

Die Voroptimierung mit dem Complex-Algorithmus bringt dagegen keinen Verbesserung, wie Abb. 5.21 zeigt. Abb. 5.22 faßt die besten Jobs der Voroptimierung zusammen.

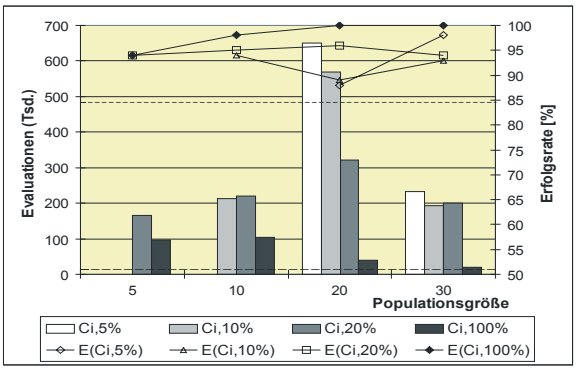


Abb. 5.21: GLEAM mit Complex-Voroptimierung bei verschiedenen Anteilen an voroptimierten Individuen an der Startpopulation. (Foxholes)

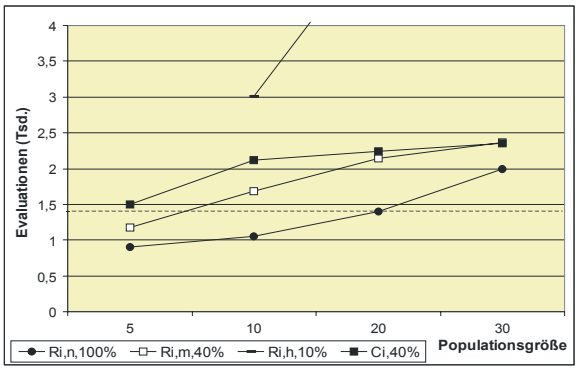


Abb. 5.22: GLEAM mit Voroptimierung: Vergleich der besten Jobs mit Rosenbrock- und Complex-Verfahren. (Foxholes)

Bei der Nachoptimierung zeigt sich, daß bei einer derart vergleichsweise einfachen Aufgabe das Erreichen des Attraktionsgebiets eines (lokalen) Optimums und seine genaue Bestimmung für GLEAM kaum einen Aufwandsunterschied ausmacht. Daher hat die Nachoptimierung bei Erreichen des Attraktionsgebiets des globalen Optimums kaum Spielraum für Verbesserungen. Abb. 5.23 zeigt das beim Rosenbrock-Verfahren: Je später abgebrochen wird,

desto höher steigt die Erfolgsrate. Der Grund für den Erfolg liegt jedoch so gut wie nie in der Nachoptimierung, wie die in Abb. 5.24 und Abb. 5.25 dargestellten Erfolgsursachen für die jeweils besten Jobs für G1 und G3 deutlich machen. Bei der Nachoptimierung mit dem Complex-Algorithmus ergibt sich das gleiche Bild, siehe Abb. 5.26. Auch hier trägt die Nachoptimierung nur in Einzelfällen zum Erfolg bei, siehe Anhang B.2.3.

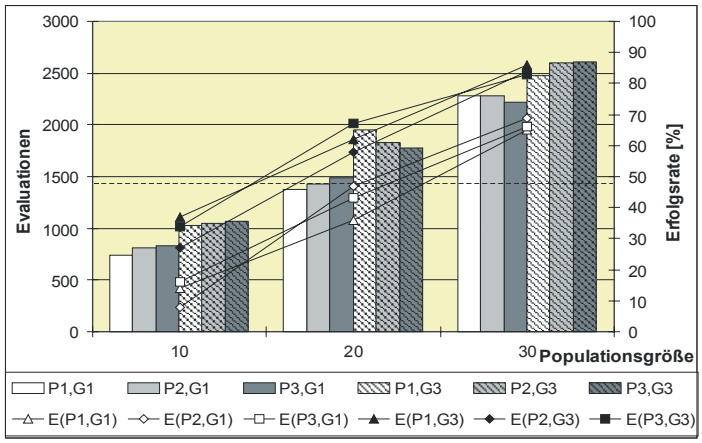


Abb. 5.23: Nachoptimierung mit Rosenbrock und Präzision h bei drei Populationsgrößen. (Foxholes)

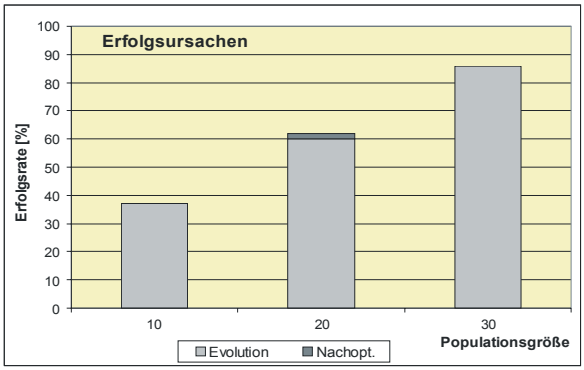


Abb. 5.24: Verteilung der Erfolgsursachen auf reine Evolution und Nachoptimierung bei Job NR,h,P1,G3. (Foxholes)

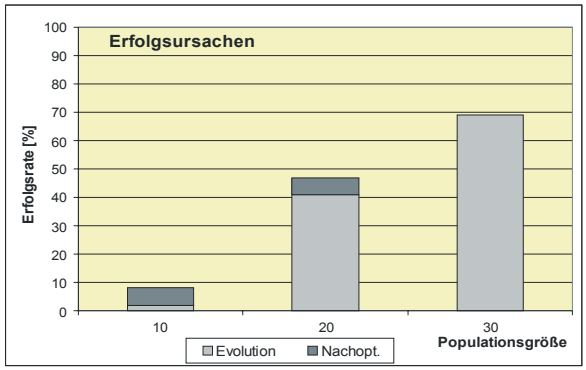


Abb. 5.25: Verteilung der Erfolgsursachen auf reine Evolution und Nachoptimierung bei Job NR,h,P2,G1. (Foxholes)

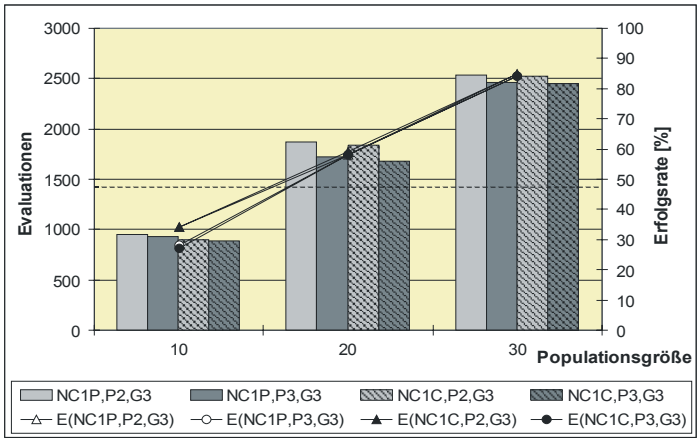


Abb. 5.26: Nachoptimierung mit dem Complex-Algorithmus. Vergleich der jeweils beiden besten Jobs mit einem Startpunkt und einem Startcomplex. (Foxholes)

Bei der direkten Integration liefern die beiden lokalen Verfahren bei allen Parametrierungen eine Erfolgsrate von 100%. Eine Verbesserung des GLEAM-Ergebnisses kann jedoch in keinem Fall erreicht werden. Bild Abb. 5.27 zeigt, daß sich mit einem Lamarckanteil von 100% beim Rosenbrock-Verfahren bei allen drei möglichen Präzisionen und auch beim Complex-

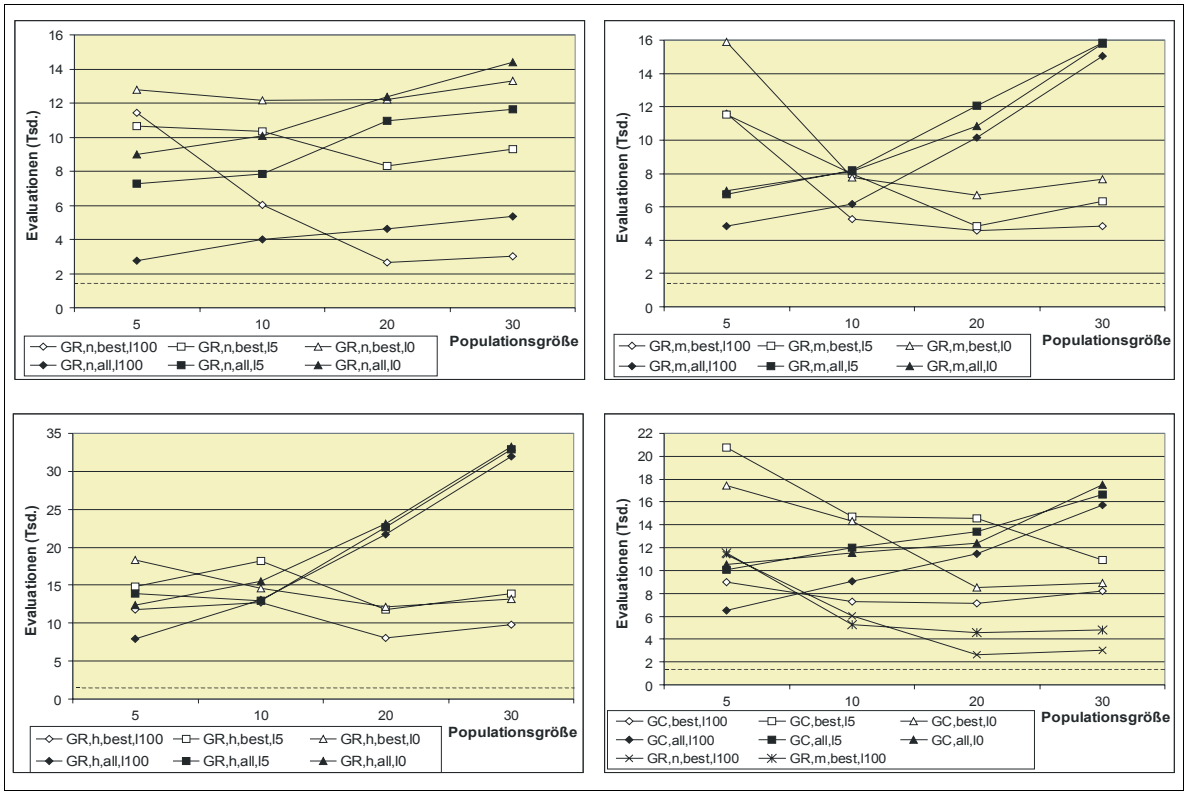


Abb. 5.27: Direkte Integration beider lokaler Verfahren: Vergleich der Präzisionen n, m und h des Rosenbrocks, der lokalen Optimierung des besten und aller Nachkommen sowie der Lamarckanteile 100%, 5% und 0% (Baldwin Evolution). (Foxholes)

Algorithmus die besten Ergebnisse erzielen lassen. Die Unterschiede zwischen der lokalen Optimierung des besten oder aller Nachkommen sind bei Shekel's Foxholes marginal.

Die zusätzliche Voroptimierung der Startpopulationen verbessert zwar die Ergebnisse der direkten Integration beim Rosenbrock-Verfahren bei allen Präzisionen, erreicht aber nicht die durch GLEAM vorgegebene Marge, wie Abb. 5.28 zeigt. Beim Complex-Algorithmus bringt die Voroptimierung dagegen kein besseres Ergebnis.

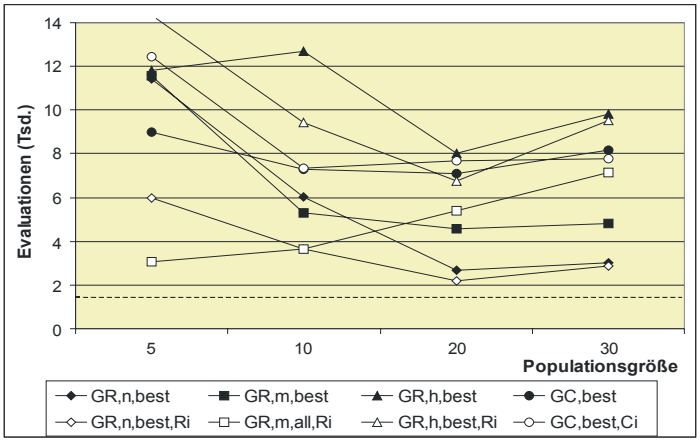


Abb. 5.28: Direkte Integration: Vergleich zwischen zufälligen und voroptimierten Startpopulationen bei den besten Parametrierungen und 100% Lamarckanteil. (Foxholes)

Abb. 5.29 zeigt die besten Jobs der verzögerten direkten Integration mit beiden lokalen Verfahren, allen drei möglichen Präzisionen des Rosenbrock-Verfahrens, der lokalen Verbesserung aller oder des besten Nachkommens und mit zufälliger oder voroptimierter Startpopulation. Alle verglichenen Jobs benutzen reine Lamarcksche Evolution. Wie Abb. 5.29 zeigt, bringt die lokale Verbesserung aller Nachkommen regelmäßig die besten Ergebnisse. Nur bei einer Parametrierung des Rosenbrock-Verfahrens mit voroptimierter Startpopulation, nämlich *GvR,p10,n,all,l100,P3,Ri*, wird der beste GLEAM-Wert erreicht. Abb. 5.30 vergleicht noch-

mal die jeweils besten Jobs aller Verfahrenskombinationen und Parametrierungen: Die verzögerte oder unverzögerte direkte Integration mit dem Rosenbrock-Verfahren schneidet bei niedriger Präzision und voroptimierten Startpopulationen am besten ab. Auffallend ist auch, daß nur Jobs mit niedriger Präzision unter den besten sind. Sie bringen bei der vorliegenden Aufgabenstellung offenbar das günstigste Aufwands-/Leistungsverhältnis.

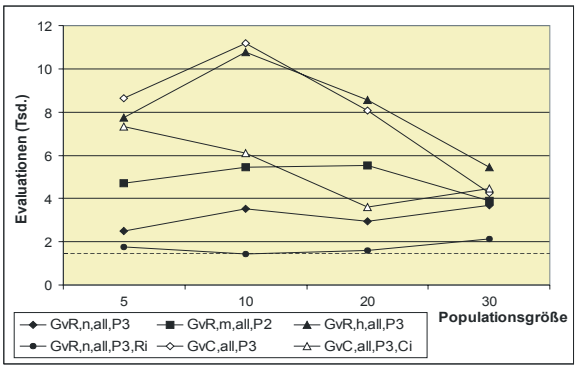


Abb. 5.29: Verzögerte direkte Integration beider lokaler Verfahren: Vergleich der besten Jobs aller Parametrierungen. (Foxholes)

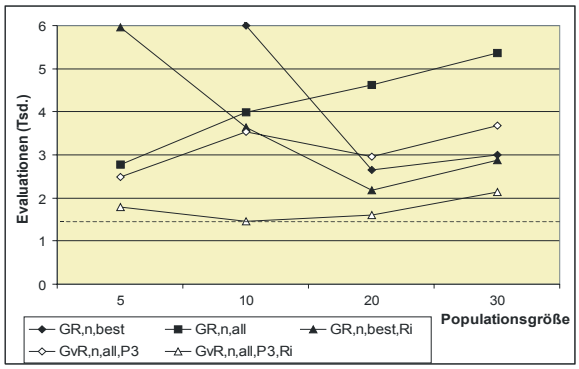


Abb. 5.30: Vergleich der besten Jobs der (verzögerten) direkten Integration. Alle Jobs mit 100% Lamarckanteil. (Foxholes)

Bei Shekel's Foxholes haben alle Hybridisierungsarten bis auf die Nachoptimierung eine Erfolgsrate von 100% geliefert. Abb. 5.31 vergleicht die besten der erfolgreichen Jobs aller Verfahrenskombinationen und Parametrierungen. Es wird deutlich, daß bei der vergleichsweise einfachen multimodalen Aufgabenstellung GLEAM allein zuverlässig das Optimum liefert und eine Hybridisierung kaum zu einer Leistungssteigerung führt. Die einzige Ausnahme ist die Kombination in Form einer mit dem Rosenbrock-Verfahren bei niedriger Präzision voroptimierten Startpopulation (Job *Ri,p5,n,100%*), die mit 63% des GLEAM-Aufwands das Optimum liefert.

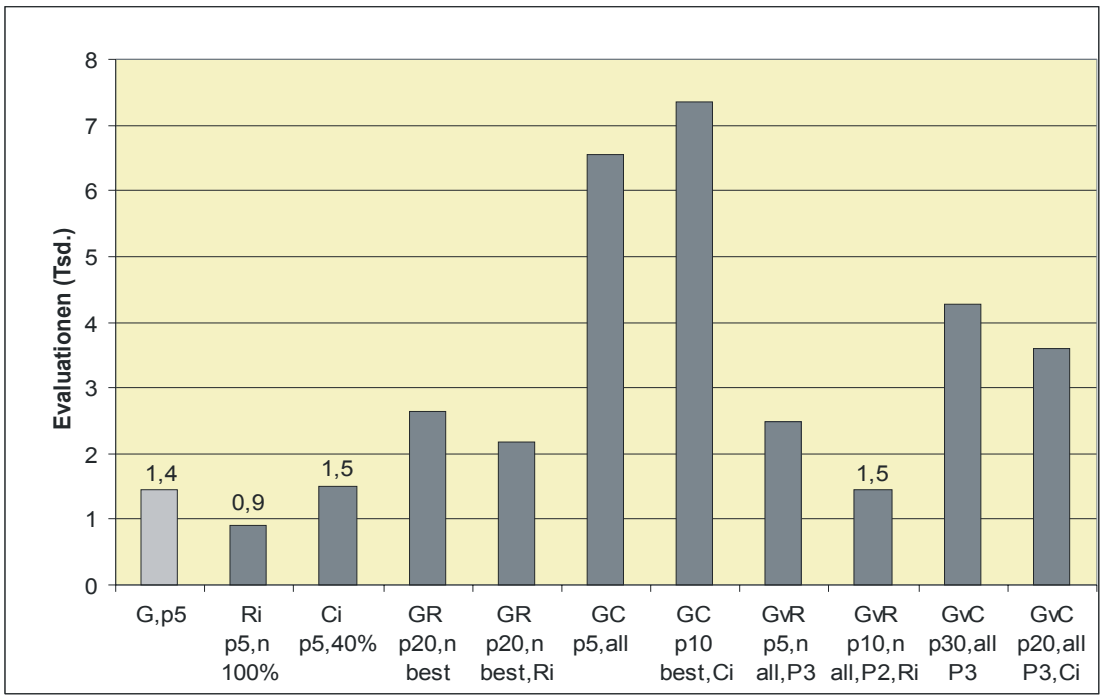


Abb. 5.31: Gesamtvergleich der besten Jobs je Integrationsart und Parametrierung bei 100% Erfolgsrate. Alle Jobs der (verzögerten) direkten Integration mit reiner Lamarckscher Evolution. (Foxholes)

5.2.1.3 Verallgemeinerte Rastrigin Funktion

Auch bei der verallgemeinerten Rastrigin-Funktion ist GLEAM noch bei extrem kleinen Populationsgrößen erfolgreich und der Aufwand sinkt ebenfalls mit dem Kleinerwerden der Population wie bei den Foxholes, siehe Abb. 5.32. Obwohl der Aufwand wesentlich höher ist, was angesichts der größeren Komplexität des 20-dimensionalen Problems auch nicht überrascht, muß die Aufgabe als vergleichsweise „einfach für GLEAM“ angesehen werden. Darauf wird in Abschnitt 5.2.3 noch einmal eingegangen werden.

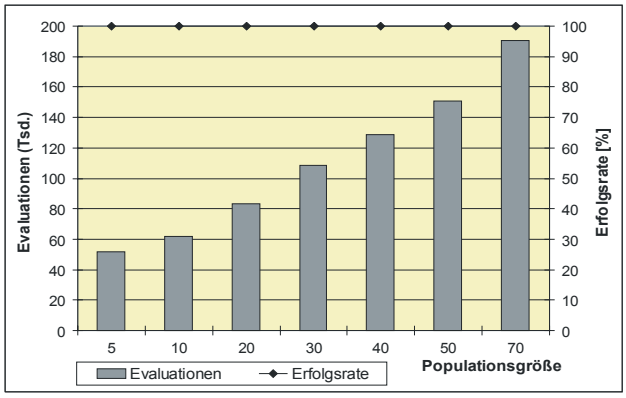


Abb. 5.32: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Rastrigin)

Die beiden lokalen Verfahren sind bei keiner Parametrierung in der Lage, das Optimum zu finden. Da es ab einer sehr hohen Präzision (*x*) zur Nichtkonvergenz des Rosenbrock-Verfahrens kommt, wurden nur Jobs mit maximal hoher Präzision (*h*) durchgeführt.

Die Vorinitialisierung der Startpopulation bringt nur eine geringe Verbesserung gegenüber GLEAM. Abb. 5.33 zeigt die Verhältnisse für das Rosenbrock-Verfahren bei niedriger und hoher Präzision. Auch hier zeigt sich, daß die Jobs mit niedriger Präzision meist besser abschneiden als die mit hoher. Die Vorinitialisierung mit dem Complex-Algorithmus erreicht ein geringfügig besseres Ergebnis, wie Bild Abb. 5.34, das die besten Jobs mit dem Complex- und dem Rosenbrock-Verfahren bei allen Präzisionen vergleicht, zeigt. Der Job *Ci,p5,20%* benötigt 97% des Aufwands des besten GLEAM-Laufs.

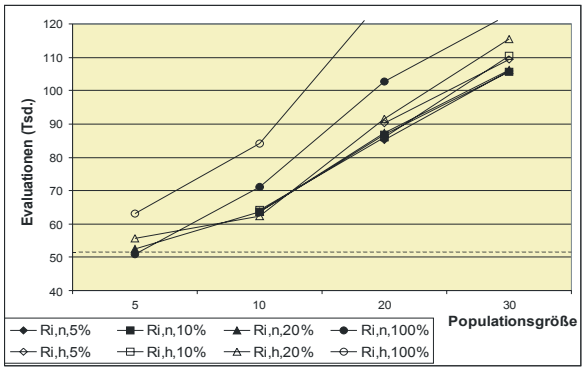


Abb. 5.33: GLEAM mit Rosenbrock-Voroptimierung: Vergleich der Präzisionen n und h bei unterschiedlichen stark voroptimierter Startpopulation. (Rastrigin)

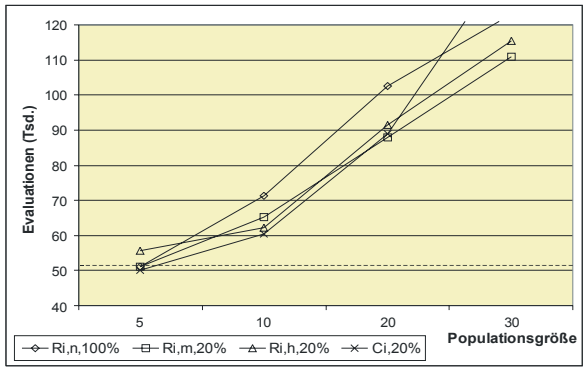


Abb. 5.34: GLEAM mit Voroptimierung: Vergleich der besten Jobs der beiden lokalen Verfahren bei allen Parametrierungen. (Rastrigin)

Die Nachoptimierung mit dem Rosenbrock-Verfahren ist insofern erfolgreich, als es ab einer Populationsgröße von 50 bei den Präzisionen *m* und *h* zu Jobs mit Erfolgsraten von 99% und mehr kommt, siehe Abb. 5.35. Der Unterschied zwischen den beiden Präzisionen ist gering. Der Aufwand liegt jedoch immer über dem des besten GLEAM-Jobs und ist bei dem erfolgreichen Job *NR,p90,m,P2,G1* mit 160% des GLEAM-Wertes am geringsten. Die erreichten Notenwerte liegen bei Nischenchecks erst ab drei Generationen ohne Deme-Verbesserung oder -Akzeptanz (*G3*) fast immer mindestens über 98000, siehe Abb. 5.36. Die Aufwände liegen jeweils deutlich unter dem GLEAM-Job gleicher Größe, jedoch eben nicht unter dem

günstigsten GLEAM-Job, wie Abb. 5.37 zeigt. Die Erfolgsquote geht ohne Ausnahme auf das Konto der Nachoptimierung. Die Nachoptimierung mit dem Complex-Algorithmus führt bei keiner Parametrierung zum Erfolg.

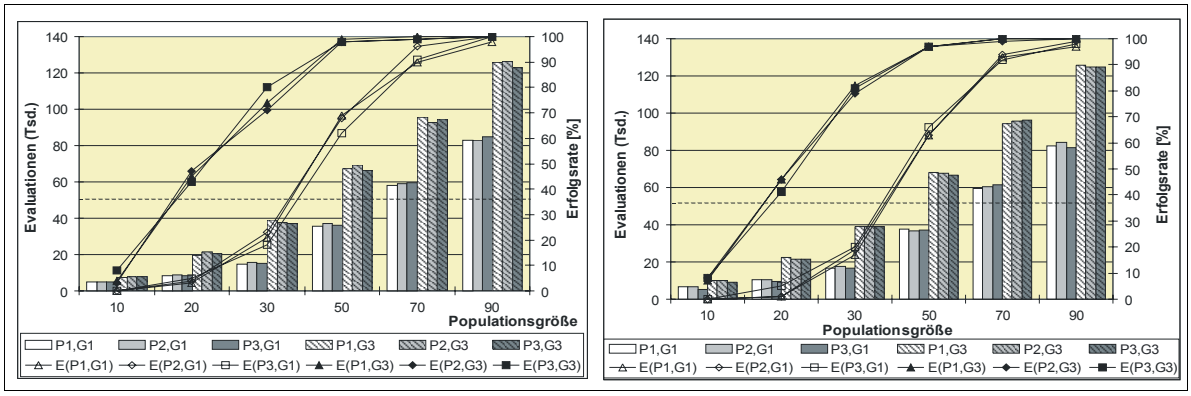


Abb. 5.35: Rosenbrock-Nachoptimierung: Vergleich von Präzision m (links) mit h (rechts) bei allen Parametrierungen. (Rastrigin)

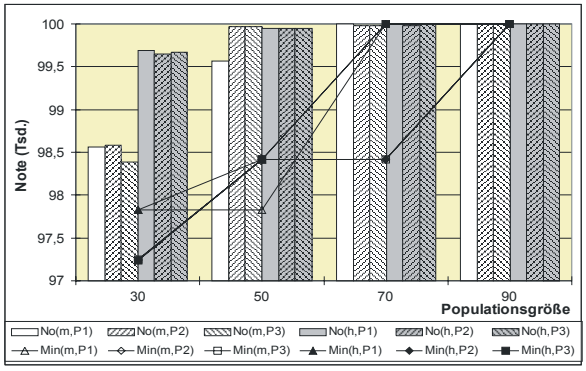


Abb. 5.36: Rosenbrock-Nachoptimierung: Vergleich der Mindest- (Min) und Durchschnittsnoten (No) bei G3. (Rastrigin)

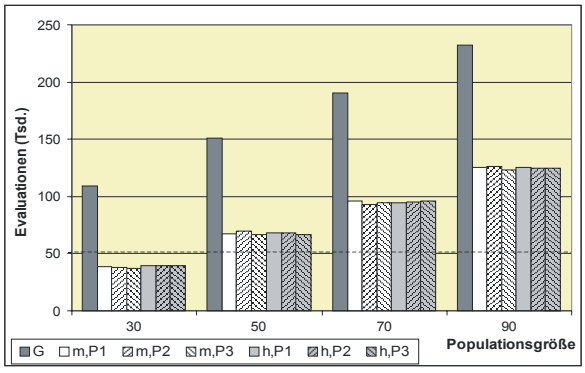


Abb. 5.37: Rosenbrock-Nachoptimierung: Vergleich der Aufwände der Jobs von Abb. 5.36 mit GLEAM-Jobs gleicher Größe. (Rastrigin)

Bei der <u>direkten Rosenbrock-Integration</u> schneiden die Jobs mit Baldwin-Evolution so schlecht ab, daß nur 10 Läufe pro Job durchgeführt werden konnten. Abb. 5.38 faßt die Ergebnisse für Präzision *n* zusammen. Die Baldwin-Jobs (*l0*) verursachen einen extremen Aufwand, ohne zuverlässig zum gewünschten Erfolg zu führen. Wesentlich günstiger sieht es bei einer Lamarckrate von 5% aus. Noch weiter wird der Aufwand bei einer Rate von 100% gesenkt, wobei die Läufe aller Jobs erfolgreich sind. Auch die Frage, ob nur der beste oder alle Nachkommen lokal optimiert werden sollen, wird in der Regel zu Gunsten des Besten entschieden. Daher werden die weiteren Jobs mittlerer Präzision (*m*) nur mit *best*

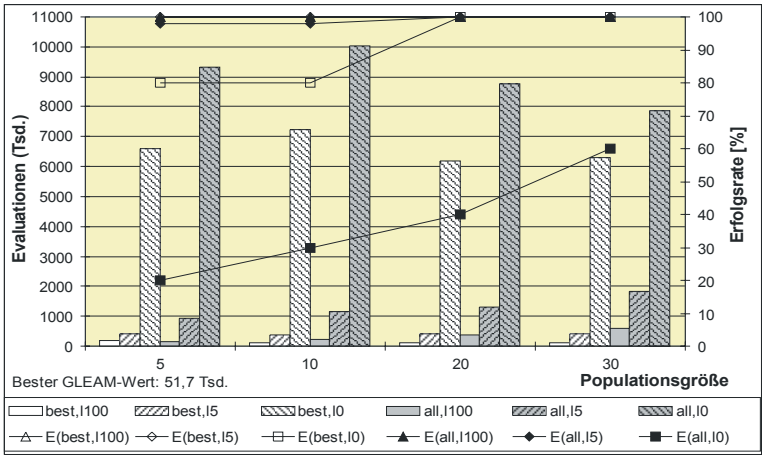


Abb. 5.38: Direkte Integration mit dem Rosenbrock: Vergleich aller Parametrierungen bei Präzision n. (Rastrigin)

und *l100* durchgeführt. Abb. 5.39 zeigt die Ergebnisse. *GR,p5,m, best,l100* kommt mit 43864 Evaluationen aus, das sind 85% des besten GLEAM-Jobs. Die direkte Integration des Rosenbrock-Verfahrens mit hoher Präzision (*h*) verursachte regelmäßig Nichtkonvergenzen, siehe Anhang B.3.4. Die Jobs mußten daher eingestellt werden.

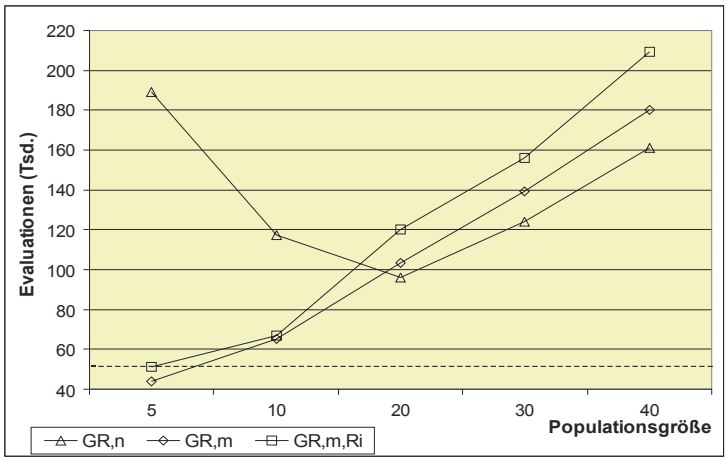


Abb. 5.39: Direkte Rosenbrock-Integration: Vergleich bei Optimierung bester Nachkommen (best) und 100%-iger Lamarckrate. (Rastrigin)

Die verzögerte direkte Rosenbrock-Integration erreicht zwar ebenfalls niedrigere Werte als der beste GLEAM-Job, bringt aber gegenüber der unverzögerten keine Vorteile, wie Abb. 5.40 zeigt.

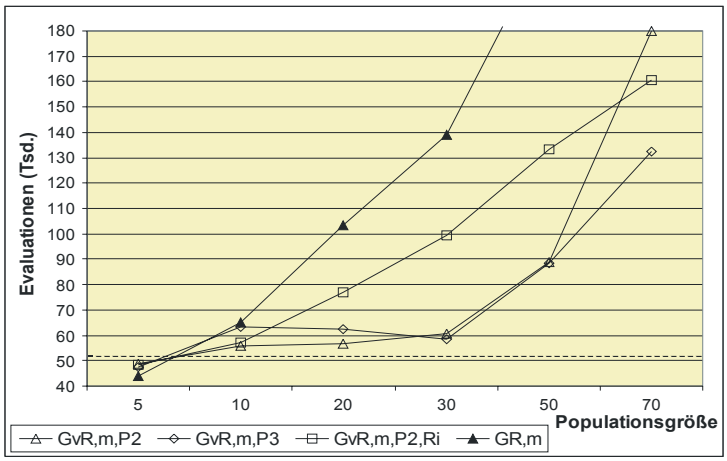


Abb. 5.40: Verzögerte direkte Rosenbrock-Integration: Vergleich der besten Jobs mit dem besten unverzögerten Job (best und 100%-ige Lamarckrate). (Rastrigin)

Die direkte Integration mit dem Complex-Algorithmus konnte wegen extrem langer Laufzeiten (zwischen 44 und 77 Stunden pro Lauf) nicht weiter untersucht werden, siehe auch Anhang B.3.4 und B.4.5. Da Werte von 4.8 - 6.5 Millionen Evaluationen bei der unverzögerten und 4.3 - 6.9 bei der verzögerten direkten Integration auftraten, kann davon ausgegangen werden, daß die Verwendung des Complex-Algorithmus keine Konkurrenz zur Integration mit dem Rosenbrock-Verfahren darstellt.

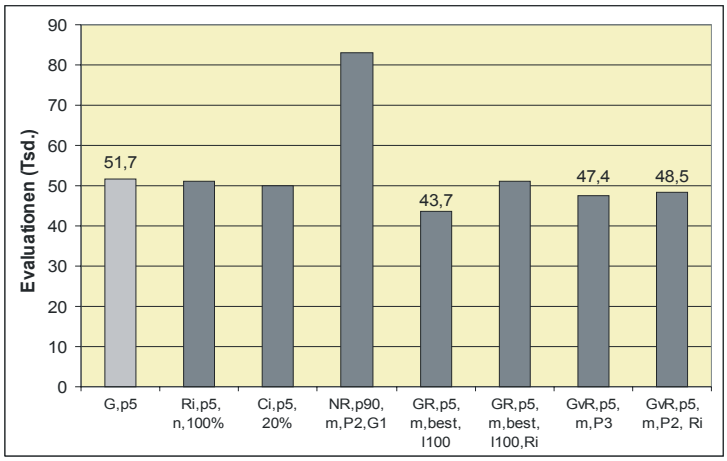


Abb. 5.41: Gesamtvergleich der besten Jobs je Integrationsart und Parametrierung mit 100% Erfolgsrate. (Rastrigin)

Die besten Jobs sind in Abb. 5.41 zusammengefaßt. Keine der untersuchten Hybridisierungsarten konnte gegenüber dem besten GLEAM-Job eine wesentliche Aufwandsreduktion liefern. Da allerdings auch diese Benchmarkaufgabe von GLEAM selbst bei extrem kleinen Populationsgrößen gelöst werden kann und der Aufwand mit sinkenden Individuenanzahl ebenfalls sinkt, ist - wie bei den Foxholes - die Aufgabenstellung offenbar für GLEAM zu leicht, als daß sich durch eine Hybridisierung noch große Verbesserungen erreichen ließen.

5.2.1.4 Fletcher's Function

Bei Fletcher's Function zeigt GLEAM das von realen Anwendungen bekannte Verhalten: Bei zu kleinen Populationsgrößen sinkt die Erfolgsquote bei steigendem Aufwand, siehe Abb. 5.42. Die Verringerung der Fitness-Berechnungen bei einer Populationsrate von 50 widerspricht dem nicht, denn sie beruht auf zwei Effekten, siehe auch Abschnitt 5.2.1: Viele Jobs wurden wegen Erreichen des Generationslimits oder wegen einsetzender Stagnation (1000 Generationen ohne Deme-Verbesserung) abgebrochen. Die günstigste Populationsgröße liegt im Bereich von 600, danach steigt der Aufwand wieder an.

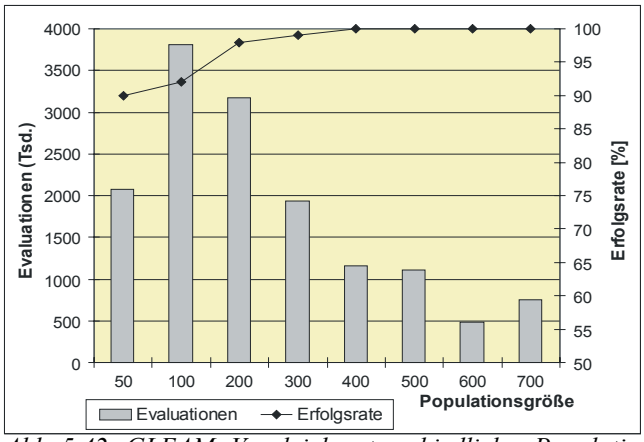


Abb. 5.42: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Fletcher)

Tabelle 5.1 zeigt die Ergebnisse für die beiden lokalen Verfahren, die mit einer Erfolgsrate von immerhin 10% das Ziel erreichen. Beim Rosenbrock-Verfahren ist nur der Job mit mittlerer Präzision interessant, da bei den beiden anderen Präzisionen kein Erfolg eintritt bzw. die Konvergenz nicht gewährleistet ist (siehe Spalte *Restarts*). Um einen Vergleich mit GLEAM herstellen zu können, ist die Frage zu beantworten, wie häufig die lokalen Verfahren gestartet werden müssen, um eine vergleichbare Erfolgsquote zu erreichen. Als vergleichbar mag angesichts des üblichen Stichprobenumfangs von 100 eine durchschnittliche Erfolgsrate von mindestens 99.5% genügen. Davon ausgehend beträgt die Anzahl der notwendige Läufe für die beiden Jobs 51 und die rechte Spalte von Tabelle 5.1 zeigt die sich daraus ergebenden Evaluationen. Da die Werte deutlich unter dem GLEAM-Wert liegen, sollen auch sie zum Vergleich mit den Resultaten der Hybridisierung herangezogen werden.

Verfahren	Erfolgsrate [%]	Restarts	Evaluationen	Durchschnittsnote	Bei 99.5% Erfolgsrate: Anzahl Läufe	Evaluationen
Rosenbrock, n	0	0	464	44572		
Rosenbrock, m	10	0	1090	51213	51	55590
Rosenbrock, h	0	60	1318	25159		
Complex	10	0	243	23003	51	12393

Tab. 5.1: Ergebnisse des Rosenbrock- und des Complex-Algorithmus. (Fletcher)

Die Voroptimierung bringt bei beiden Verfahren eine erhebliche Verbesserung. Beim Rosenbrock-Verfahren werden nur die Jobs mit mittlerer Präzision betrachtet, da sie deutlich besser ausfallen als die mit niedriger. Bei den in Abb. 5.43 dargestellten Ergebnissen ist der vergleichbare Wert für eine reine Rosenbrock-Optimierung aus Tabelle 5.1 grob gestrichelt dargestellt. Er wird von GLEAM mit zu 100% voroptimierten Startpopulationen ab einer Populationsgröße von 20 unterboten. Auch der Complex-Algorithmus erreicht ab einer Populationsgröße von 20 und 100%-iger Voroptimierung sicher das Ziel, wie Abb. 5.44 zeigt.

Er benötigt aber mehr Evaluationen als die reine Complex-Optimierung, die in Abb. 5.44 ebenfalls durch eine grob gestrichelte Linie dargestellt ist.

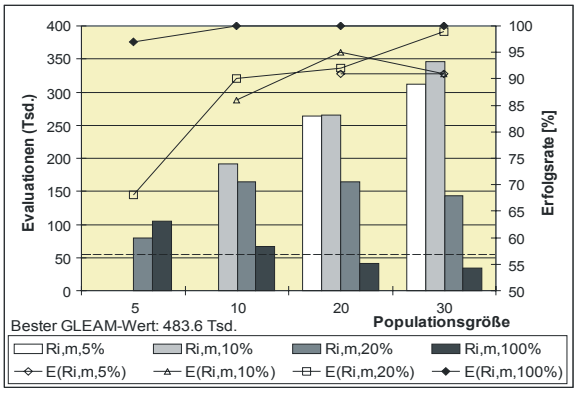


Abb. 5.43: GLEAM mit Rosenbrock-Voroptimierung: Vergleich verschiedener Anteile an voroptimierten Individuen bei mittlerer Präzision. Grob gestrichelte Linie: reine Rosenbrock-Optimierung. (Fletcher)

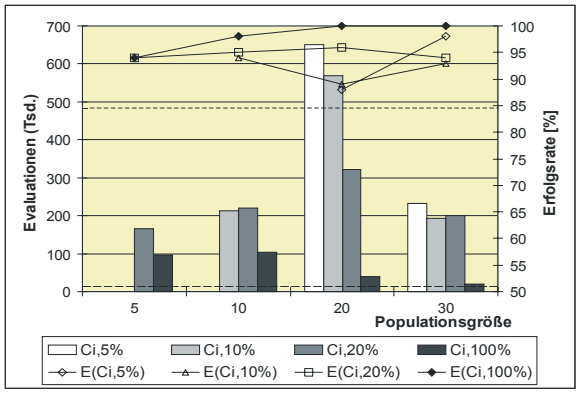


Abb. 5.44: GLEAM mit Complex-Voroptimierung: Vergleich verschiedener Anteile an voroptimierten Individuen. Grob gestrichelte Linie: reine Complex-Optimierung. (Fletcher)

Abb. 5.45 vergleicht die Jobs mit 100% Erfolgsrate und 100%-iger Voroptimierung durch die beiden lokalen Verfahren. Der Complex-Algorithmus erzielt bessere Ergebnisse, wobei die reine Complex-Optimierung (untere grob gestrichelte Linie) jedoch nicht unterboten wird. Interessant ist allerdings ein Blick auf die Erfolgsursachen, der deutlich macht, daß bei beiden lokalen Verfahren ein erheblicher Anteil auf bereits erfolgreiche Voroptimierungen zurückzuführen ist und die Evolution nur selten benötigt wird, siehe Abb. 5.46.

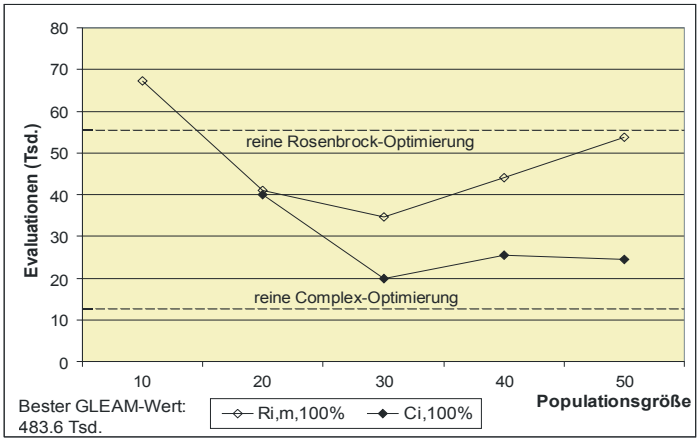


Abb. 5.45: GLEAM mit Voroptimierung: Vergleich der erfolgreichen Jobs bei 100%-iger Voroptimierung. (Fletcher)

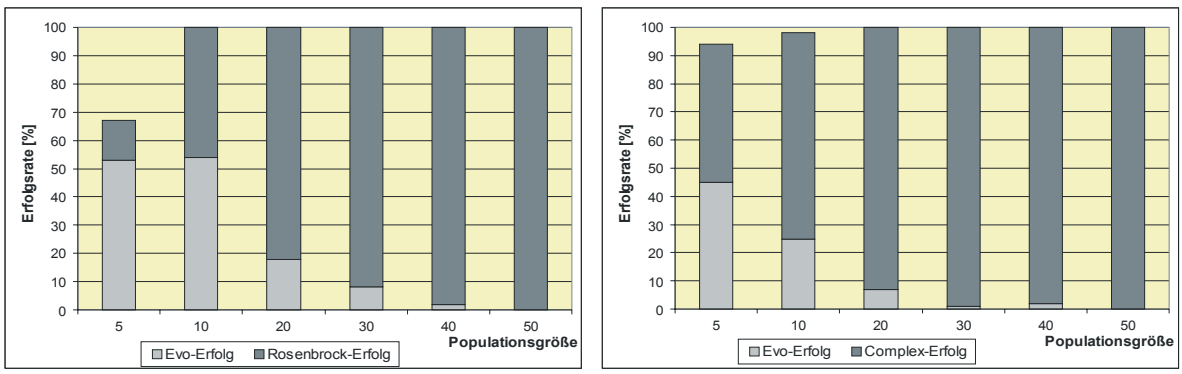


Abb. 5.46: GLEAM mit Voroptimierung: Vergleich der Erfolgsursachen bei Rosenbrock- und Complex-voroptimierten Startpopulationen. (Fletcher)

Die Nachoptimierung mit dem Rosenbrock-Verfahren kommt über eine Erfolgsrate von 50% kaum hinaus, wie Abb. 5.47 zeigt. Es werden allerdings recht hohe Fitness-Werte erreicht. Entsprechend günstiger wird die Erfolgsquote, wenn die Genauigkeitsanforderung an die Zielfunktion um eine Größenordnung auf 0.0001 gesenkt wird, siehe Abb. 5.48. Die durchschnittlichen Noten zeigt Abb. 5.49: Ab einer Populationsgröße von 20 kann zuverlässig mit einem durchschnittlichen Notenwert von mindestens 98000 bei einem Spitzenwert von 99788 gerechnet werden. Schlechter sieht es dagegen bei den Jobs mit den besten und schlechtesten Mindestnoten aus: Die Parametrierungen beider Jobs fallen auf den Job *NR,m,P3,G1* zusammen und seine in Abb. 5.50 dargestellten Daten zeigen, daß im Gegensatz zu Schwefel's Sphere und der Rastrigin Funktion hier durchaus schlechte Läufe auftreten können. Die Erfolgsursachen liegen auch bei Flecher's Function bis auf wenige Ausnahmen in der Nachoptimierung selbst.

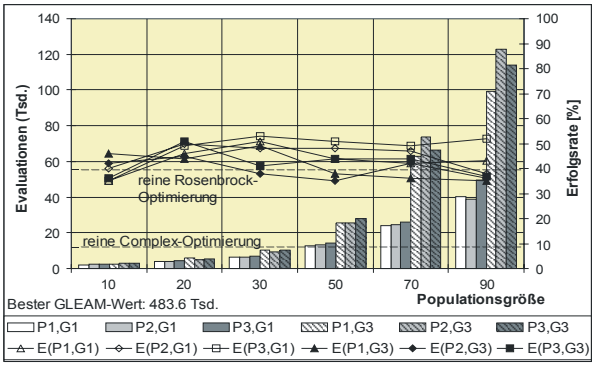


Abb. 5.47: Rosenbrock-Nachoptimierung: Vergleich der Parametrierungen bei mittlerer Präzision und verschiedenen Populationsgrößen. (Fletcher)

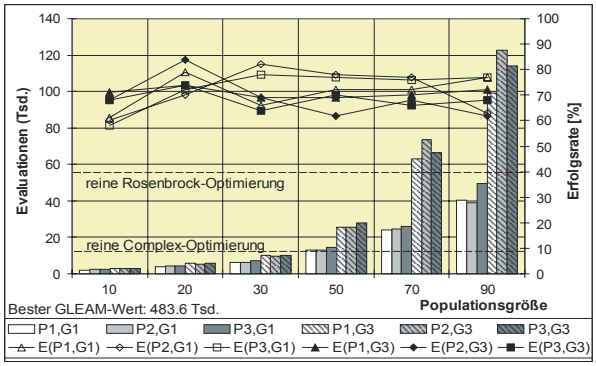


Abb. 5.48: Rosenbrock-Nachoptimierung: Wie Abb. 5.47 jedoch mit geringerem Zielfunktionswert (0.0001 statt 0.00001). (Fletcher)

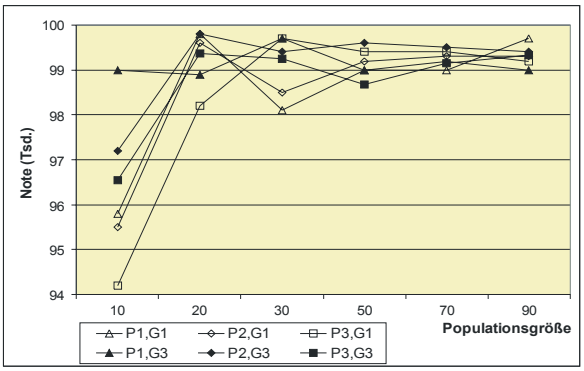


Abb. 5.49: Rosenbrock-Nachoptimierung: Vergleich der Durchschnittsnoten bei allen Parametrierungen und mittlerer Präzision. (Fletcher)

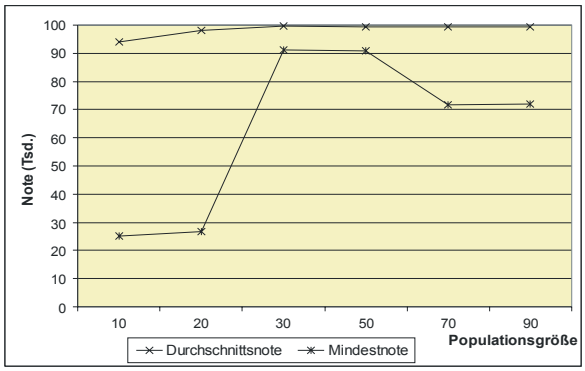


Abb. 5.50: Rosenbrock-Nachoptimierung: Job mit der schlechtesten und besten Mindestnote, NR,m,P3,G1. (Fletcher)

Bei Fletcher's Function greift die Nachoptimierung mit dem Complex-Verfahren zum ersten Mal. Abb. 5.51 vergleicht die beiden Möglichkeiten zur Weiterverwendung der Evolutionsergebnisse: Einerseits separate Complex-Läufe mit allen Ergebnissen, wobei jedes Ergebnis jeweils einzeln in einem Startcomplex enthalten ist (links) und andererseits die Verwendung aller Ergebnisse zur Bildung eines gemeinsamen Startcomplexes für einen Lauf (rechts). Letzteres bringt deutlich bessere Ergebnisse bei vergleichbarem Aufwand. Auch die Erfolgsrate übertrifft mit bis zu 75% das Rosenbrock-Verfahren (max. 53%). Allerdings sind die er-

reichten Durchschnittsnoten wesentlich geringer und streuen angesichts der höheren Erfolgsrate stärker als beim Rosenbrock-Verfahren, siehe Abb. 5.52. Die Erfolge sind auch beim Complex-Algorithmus im wesentlichen auf die Nachoptimierung zurückzuführen.

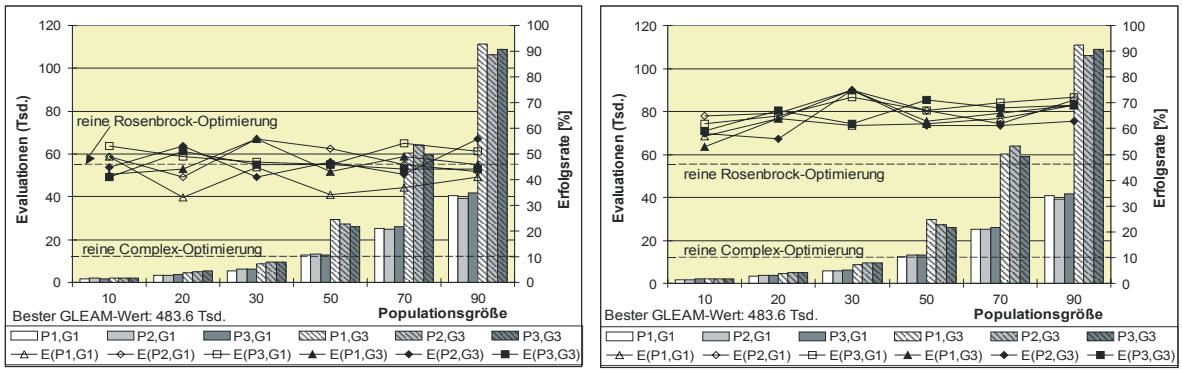


Abb. 5.51: Complex-Nachoptimierung: Vergleich der Nachoptimierung der einzelnen Evolutionsergebnisse jeweils für sich (links) mit ihrer Zusammenfassung zu einem Startcomplex (rechts). (Fletcher)

Angesichts der dargelegten Ergebnisse erschienen Versuche zur Variation der Nischenkriterien sinnvoll. Sie wurden mit einer Populationsrate von 30 durchgeführt, da hiermit zuvor die besten Werte bei NC1C erzielt wurden. Die neuen Werte für ε und ε_{Pop} bei maximaler Nischenanzahl $N = 5$ bzw. 6 bei *Pf* und *Pg* sind in Tabelle 5.2 zu finden.

Kennung	ε	ε_{Pop}
Pa	0.0005	0.005
Pb	0.001	0.01
Pc	0.003	0.03
Pd	0.005	0.05
Pe	0.01	0.1
Pf	0.01	0.35
Pg	0.01	0.5

Tab. 5.2: Zusätzliche Variationen von ε und ε_{Pop}

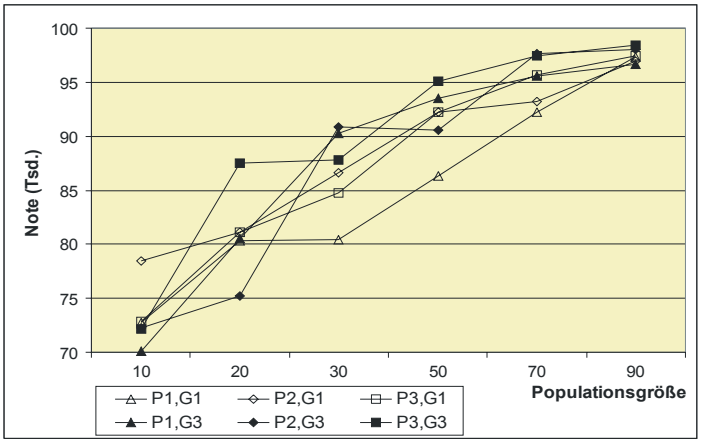


Abb. 5.52: Complex-Nachoptimierung: Durchschnittsnoten bei der Zusammenfassung der Evolutions-Ergebnisse zu einem Startcomplex. (Fletcher)

Wie Abb. 5.53 jedoch zeigt, konnte nur eine geringe Verbesserung gegenüber dem links dargestellten besten Job *NC1C,p30, P2,G1* erreicht werden. Die Verschärfung der Kriterien zur Nischenbildung (*Pa* und *Pb*) bringt hier kaum einen Vorteil. Dage-

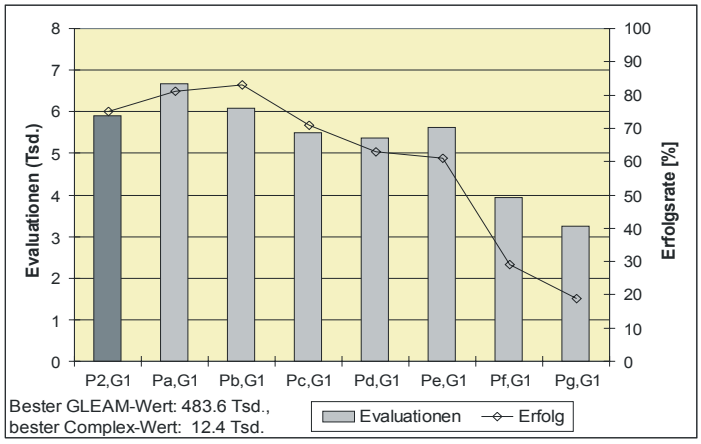


Abb. 5.53: Complex-Nachoptimierung: Vergleich des besten Jobs NC1C,p30,P2,G1(links) mit Variationen der Nischenbedingungen bei gleicher Populationsgröße. (Fletcher)

gen führt eine Abschwächung (*Pc* bis *Pg*) erwartungsgemäß zu schlechteren Ergebnissen, da nun häufiger zu früh mit der Nachoptimierung begonnen wird.

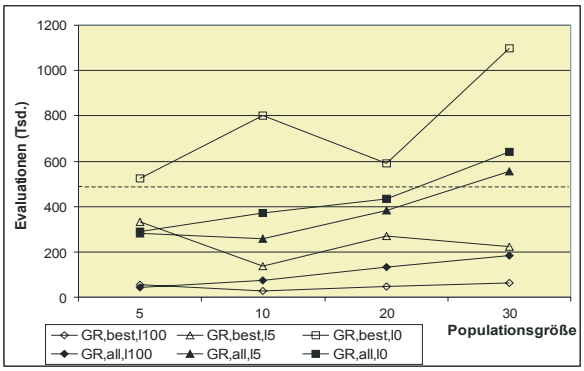


Abb. 5.54: Direkte Rosenbrock-Integration: Vergleich der Lamarckraten sowie der Verbesserung aller oder nur des besten Nachkommens bei niedriger Präzision. (Fletcher)

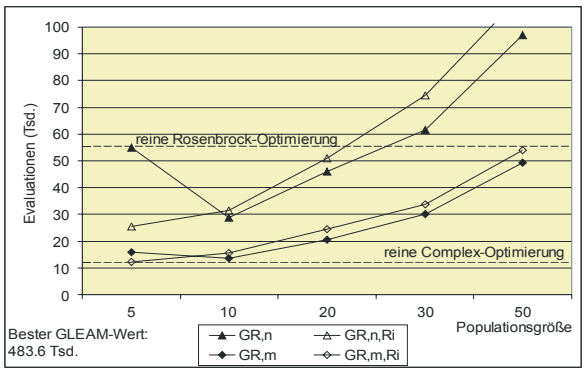


Abb. 5.55: Direkte Rosenbrock-Integration: Vergleich von Jobs mit Verbesserung des besten Nachkommens und Lamarck-Evolution. (Fletcher)

Bei der direkten Integration setzt sich auch bei Fletcher's Function der bisher beobachtete Trend fort, daß Jobs mit reiner Baldwin-Evolution oder einer Lamarckrate von 5% schlechter abschneiden als bei einer Rate von 100%. Abb. 5.54 zeigt die Details für niedrige Präzision. Dabei ist zu beachten, daß bei der kleinsten Populationsgröße und einer geringeren Lamarckrate als 100% die Erfolgsrate auf bis zu 96% absinkt. Die Lamarcksche Evolution (*l100*) und die Verbesserung nur des besten Nachkommens (*best*) erweist sich hier als am günstigsten. Abb. 5.55 vergleicht Jobs mit den genannten Einstellungen bei niedriger und mittlerer Präzision sowie mit und ohne Voroptimierung. Die Jobs mit mittlerer Präzision liefern die besten Ergebnisse und unterschreiten sogar den niedrigen Wert der reinen Complex-Optimierung. Die direkte Integration mit dem Complex-Verfahren unterbietet nochmal die besten Werte der direkten Rosenbrock-Integration, wie Abb. 5.56 zeigt.

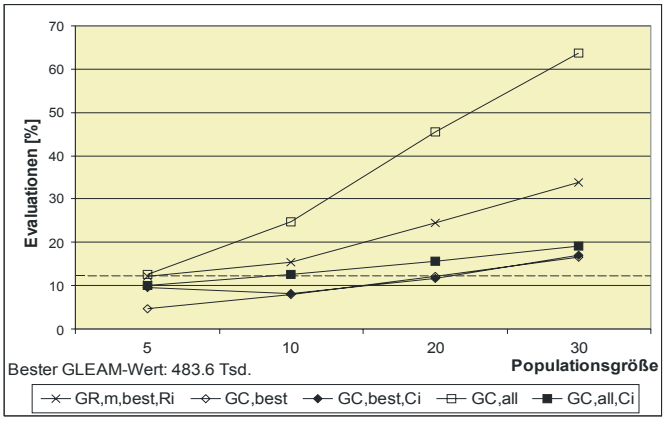


Abb. 5.56: Direkte Complex-Integration: Jobs mit Lamarckscher Evolution und dem besten Rosenbrock-Job zum Vergleich. Grob gestrichelte Linie: reine Complex-Optimierung. (Fletcher)

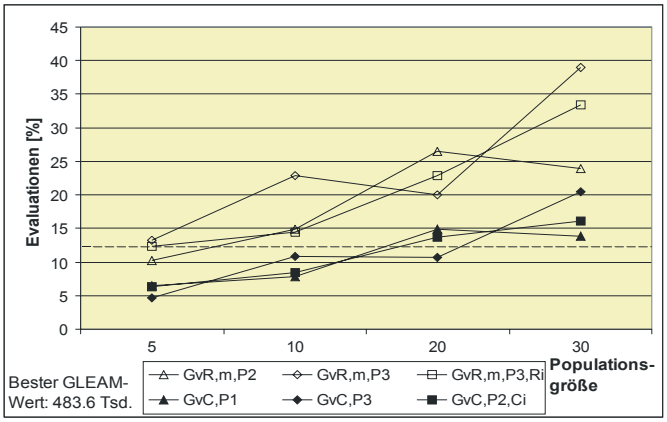


Abb. 5.57: Verzögerte direkte Integration: Vergleich der jeweils drei besten Jobs von Rosenbrock- und Complex-Optimierung des besten Nachkommen bei reiner Lamarck-Evolution. Grob gestrichelte Linie: reine Complex-Optimierung. (Fletcher)

Abb. 5.57 zeigt die jeweils drei besten Jobs der verzögerten direkten Integration mit dem Rosenbrock- und dem Complex-Verfahren. Es ist deutlich zu erkennen, daß die Complex-Variante dem Rosenbrock-Verfahren überlegen ist. In Abb. 5.58 werden die jeweils besten Jobs je Integrationsart und Verfahren gegenüberstellt und es zeigt sich, daß die verzögerte direkte Integration gegenüber der unverzögerten insgesamt nur noch geringe Verbesserungen bringt. Alle Jobs der verzögerten direkten Integration wurden mit reiner Lamarck-Evolution und lokaler Verbesserungen des besten Nachkommens durchgeführt.

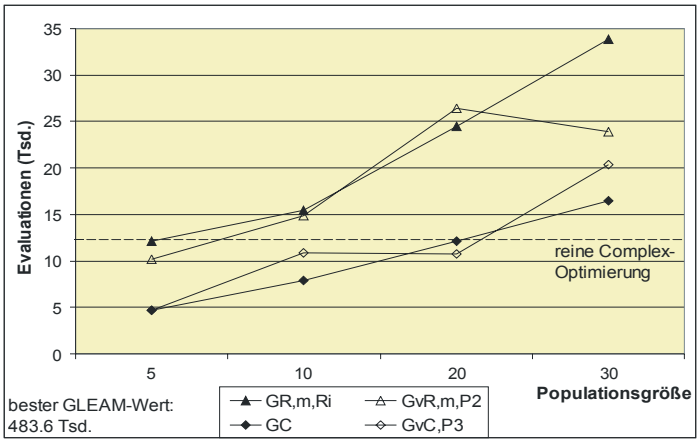


Abb. 5.58: Vergleich der besten Jobs der direkten verzögerten und unverzögerten Integration mit dem Rosenbrock- und dem Complex-Verfahren (alle best und l100). (Fletcher)

Abb. 5.59 faßt die Ergebnisse der besten Jobs je Hybridisierungsart zusammen und vergleicht sie mit dem besten GLEAM-Job. Dabei handelt es sich ausschließlich um Jobs mit reiner Lamarckscher Evolution und Verbesserung des besten Nachkommens einer Paarung. Auf Grund des enormen Unterschieds zwischen der reinen Evolution und den hybriden Jobs zeigt Abb. 5.60 nochmals letztere, um die Unterschiede zwischen ihnen deutlicher hervortreten zu lassen. Am besten schneidet die direkte Integration mit dem Complex-Algorithmus (*GC, p5,best,l100*) ab. Sie benötigt nur 0.97% vom Aufwand des GLEAM-Jobs und 30% gegenüber der reinen Complex-Optimierung. Beim Rosenbrock-Verfahren schneidet die verzögerte direkte Integration (*GvR,p5,m,best, l100,P2*) am besten ab. Sie benötigt 2.1% vom GLEAM-Aufwand und 14% gegenüber der reinen Rosenbrock-Optimierung. Damit unterbieten die besten hybriden Varianten bei Fletcher's Function die Optimierung mit den einzeln angewandten Verfahren in allen Fällen und in erheblichem Ausmaß.

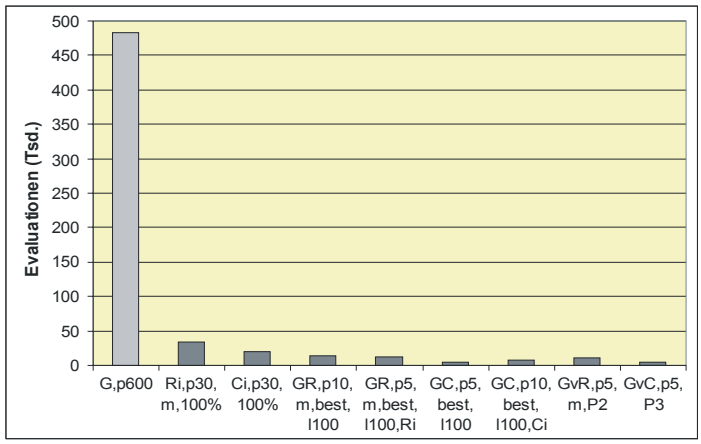


Abb. 5.59: Gesamtvergleich der besten Jobs je Hybridisierungsart und Parametrierung mit 100% Erfolgrate im Vergleich mit dem besten GLEAM-Job. Alle Jobs mit reiner Lamarck-Evolution und Verbesserung des besten Nachkommens. (Fletcher)

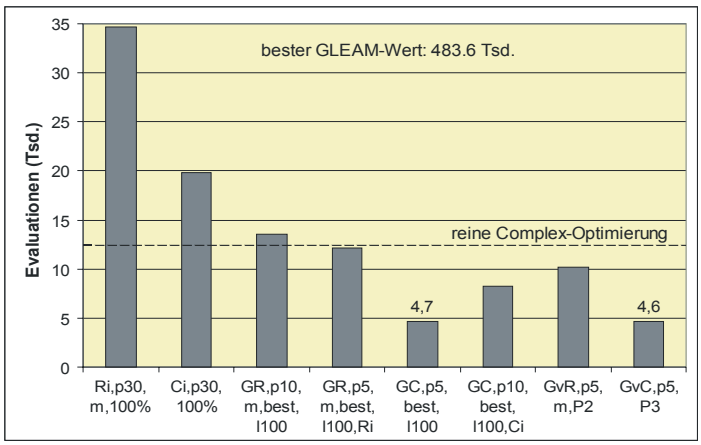


Abb. 5.60: Detailansicht des Gesamtvergleichs von Abb. 5.59 ohne GLEAM-Job. (Fletcher)

5.2.1.5 Fraktale Funktion

Wie bei Fletcher's Function kommt es auch bei der fraktalen Funktion bei zu kleinen Populationsgrößen zu Läufen, die das Ziel nicht erreichen, siehe Abb. 5.61. Der Aufwand steigt jedoch nicht, da die meisten betroffenen Läufe wegen Stagnation bei 1000 Generationen ohne Deme-Verbesserung abgebrochen wurden. Den günstigsten Wert für den Aufwand in Höhe von 195100 liefert der Job mit einer Populationsgröße von 20.

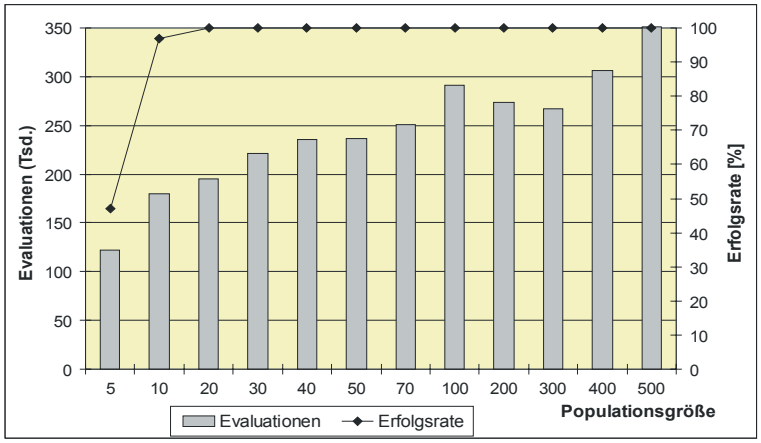


Abb. 5.61: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Fraktal)

Die beiden lokalen Verfahren erreichen bei keiner Einstellung das Ziel, siehe Tabelle 5.3. Interessant dabei ist, daß das Rosenbrock- im Gegensatz zum Complex-Verfahren bei allen Einstellungen recht gute Fitnesswerte erreicht.

Verfahren	Erfolgsrate [%]	Restarts	Evaluationen	Durchschnittsnote
Rosenbrock, n	0	0	1103	89836
Rosenbrock, m	0	0	1614	89808
Rosenbrock, h	0	0	2665	88957
Rosenbrock, u	0	0	3416	90416
Rosenbrock, v	0	0	4320	89641
Complex	0	0	781	57102

Tab. 5.3: Ergebnisse des Rosenbrock- und des Complex-Algorithmus (Fraktal)

Bei der Voroptimierung bringt vor allem das Rosenbrock-Verfahren eine Verbesserung gegenüber GLEAM unter Wahrung der 100-prozentigen Erfolgsquote. Die Jobs wurden mit den Präzisionen n, m und h durchgeführt, wobei die niedrige Präzision die besten Ergebnisse und die hohe die schlechtesten lieferte. Daher wurde auch auf Untersuchungen mit der sehr hohen Präzision verzichtet. Abb. 5.62 zeigt die Ergebnisse für die niedrige Präzision bei verschiedenen Anteilen vorinitialisierter Individuen an der Startpopulation. Abb. 5.63

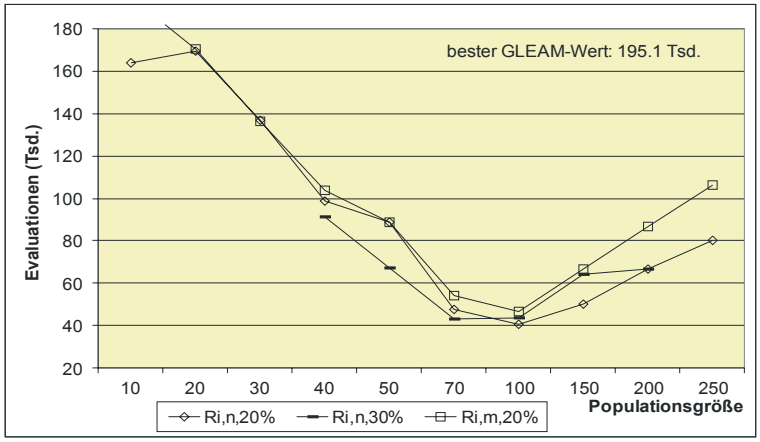


Abb. 5.64: GLEAM mit Voroptimierung: Vergleich der drei besten Jobs. (Fraktal)

vergleicht dies mit den Ergebnissen der Complex-Voroptimierung. In Abb. 5.64 sind die drei besten Jobs der Vorinitialisierung dargestellt.

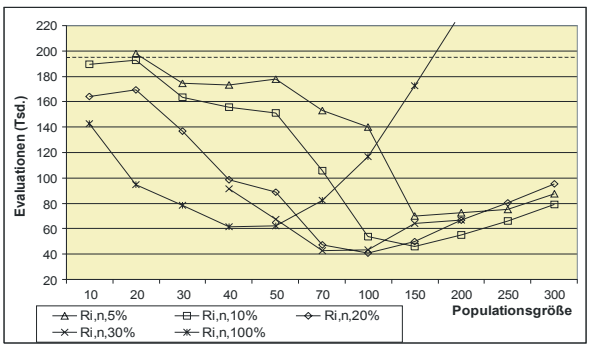


Abb. 5.62: GLEAM mit Rosenbrock-Voroptimierung: Vergleich mehrer Anteile voroptimierter Individuen an der Startpopulation bei niedriger Präzision. (Fraktal)

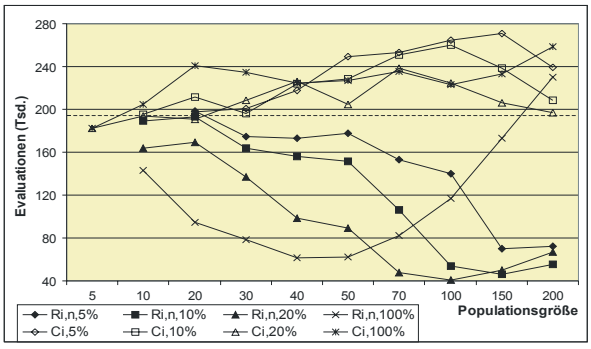


Abb. 5.63: GLEAM mit Voroptimierung: Vergleich zwischen Rosenbrock-Verfahren mit niedriger Präzision und dem Complex-Algorithmus. (Fraktal)

Die Nachoptimierung wurde beim Rosenbrock-Verfahren mit mittlerer und sehr hoher Präzision (*m* und *u*) durchgeführt, da beide Parametrierungen bei reiner Rosenbrock-Optimierung gute Ergebnisse bei wenig Aufwand bzw. die besten Ergebnisse brachten. Abb. 5.65 zeigt, daß erst bei recht großen Populationen Erfolgsraten von über 90% erreicht werden und das auch nur bei einem Nischencheck bei drei Generationen ohne Deme-Verbesserung oder -Akzeptanz (GDV/GAk = 3). Der benötigte Aufwand nähert sich dann auch dem des besten GLEAM-Jobs. Die Quelle des Erfolgs liegt allerdings im Gegensatz zu den beiden vorherigen Aufgaben mit steigender Populationsgröße in zunehmendem Maße bei der Evolution, wie die in Abb. 5.66 dargestellten beiden besten Jobs zeigen. Die erreichten Noten sind ab einer Populationsgröße von 70 von recht hoher Qualität, siehe Abb. 5.67.

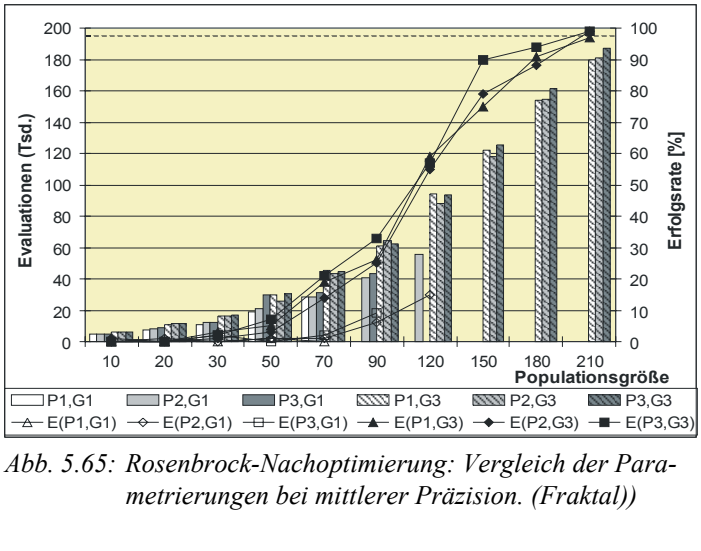


Abb. 5.65: Rosenbrock-Nachoptimierung: Vergleich der Parametrierungen bei mittlerer Präzision. (Fraktal))

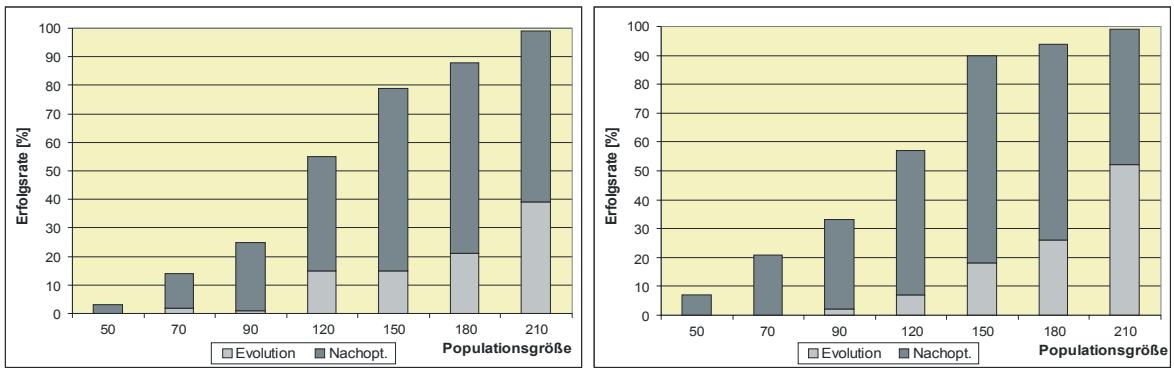


Abb. 5.66: Rosenbrock-Nachoptimierung: Vergleich der Erfolgsursachen bei den beiden erfolgreichsten Jobs NR,m,P2,G3 und NR,m,P3,G3. (Fraktal)

Wird der Nischencheck durch steigende GDV/GAk-Werte verzögert, so steigt auch erwartungsgemäß die Erfolgsrate, wie Abb. 5.68 zeigt. Allerdings steigt der Aufwand entsprechend mit und der Erfolg geht schließlich zu 90% auf das Konto der Evolution, siehe Abb. 5.69. Mit anderen Worten, wenn die Nachoptimierung bei der fraktalen Funktion sicher zum Erfolg gebracht werden soll, findet immer weniger Nachoptimierung statt, da die Evolution bereits das Ziel erreicht.

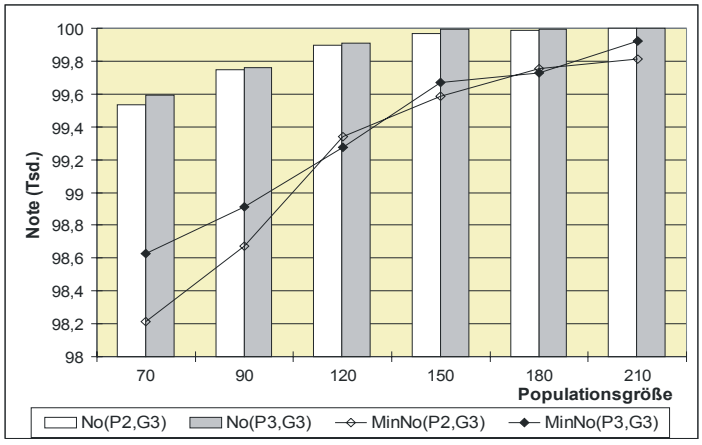


Abb. 5.67: Rosenbrock-Nachoptimierung: Jobs mit der schlechtesten und besten Mindestnote bei mittlerer Präzision. (Fraktal)

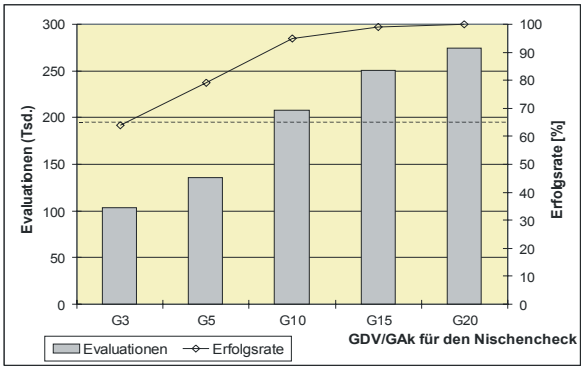


Abb. 5.68: Rosenbrock-Nachoptimierung: Einfluß eines verzögerten Nischenchecks bei NR,p120,m,P3 und fester maximaler Nischenzahl N=2. (Fraktal)

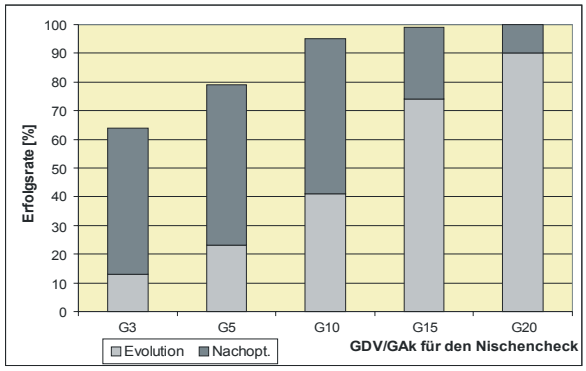


Abb. 5.69: Rosenbrock-Nachoptimierung: Erfolgsrate und -ursachen bei den Jobs von Abb. 5.68. (Fraktal)

Eine weitere Möglichkeit, die Nachoptimierung mit dem Rosenbrock-Verfahren zu verbessern, könnte in der Erhöhung der Präzision bestehen. Abb. 5.70 zeigt das Ergebnis für GDV/GAk = 3 bei sehr hoher Präzision (*u*), das sich allerdings kaum von dem mit mittlerer unterscheidet. Bei leicht gestiegenem Aufwand wird immerhin mit Job *NR,p210,u,P3,G3* eine Erfolgsrate von 100% erreicht. Aber auch hier steigt der Anteil der Evolution am Erfolg auf über 50% an, siehe Abb. 5.71. Die erreichten Noten weichen ebenfalls kaum von denen bei mittlerer Präzision ab, wie Abb. 5.72 zeigt. Bei der fraktalen Funktion bringt also eine Erhöhung der Präzision kaum einen Vorteil.

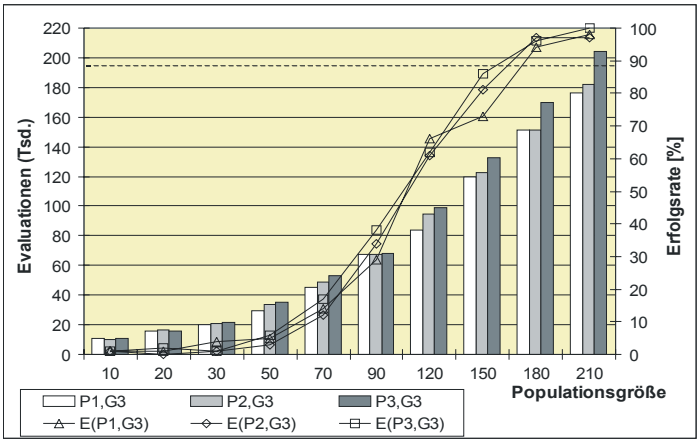


Abb. 5.70: Rosenbrock-Nachoptimierung: Vergleich unterschiedlicher Abbruchparametrierungen bei GDV/GAk=3 und sehr hoher Präzision (u). (Fraktal)

Die Nachoptimierung mit dem Complex-Algorithmus führt bei keiner Parametrierung zu nennenswertem Erfolg.

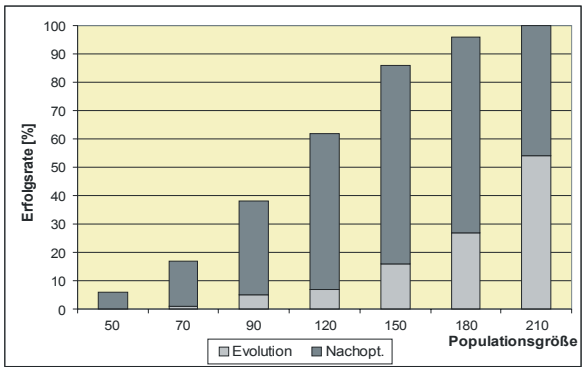


Abb. 5.71: Rosenbrock-Nachoptimierung: Erfolgsursachen beim erfolgreichsten Job NR,x,P3,G3. (Fraktal)

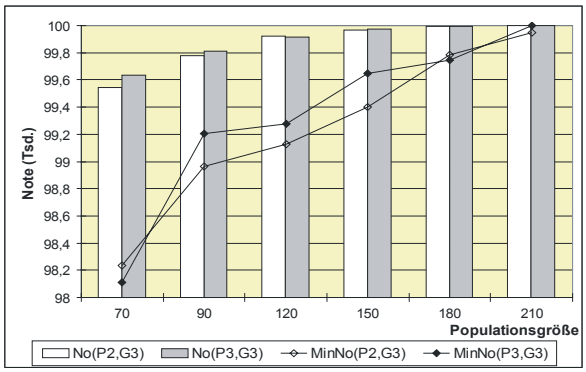


Abb. 5.72: Rosenbrock-Nachoptimierung: Jobs mit der schlechtesten und besten Mindestnote bei sehr hoher Präzision (ohne Populationsgrößen 10 bis 50). (Fraktal)

Bei der direkten Integration konnte nur die Kombination mit dem Rosenbrock-Verfahren näher untersucht werden, da beim Complex-Algorithmus extrem lange Laufzeiten zwischen 10,5 und 24 Stunden auftraten. Der Aufwand liegt mit Werten zwischen einer und 1.4 Millionen bei der direkten Integration und 0.9 bis zwei Millionen bei der verzögerten deutlich über dem bisher erreichten, siehe Anhang B.5.4 und B.5.5. Daher wurde auf weitergehende Untersuchungen zur (verzögerten) direkten Integration mit dem Complex-Algorithmus verzichtet.

Wie schon zuvor schneiden auch bei der fraktalen Funktion die Jobs mit einer Lamarckrate von fünf oder Null Prozent (Baldwin-Evolution) genauso wie die lokale Verbesserung aller Nachkommen deutlich schlechter ab als reine Lamarck-Evolution bei lokaler Optimierung des jeweils besten Nachkommens. Bei den in Abb. 5.73 dargestellten Ergebnissen sind die hohen Werte der Baldwin-Evolution und des Jobs *GR,n,all,l5* insofern etwas unsicher, als daß sie wegen des hohen Aufwands auf nur 10 Läufen pro Job beruhen.

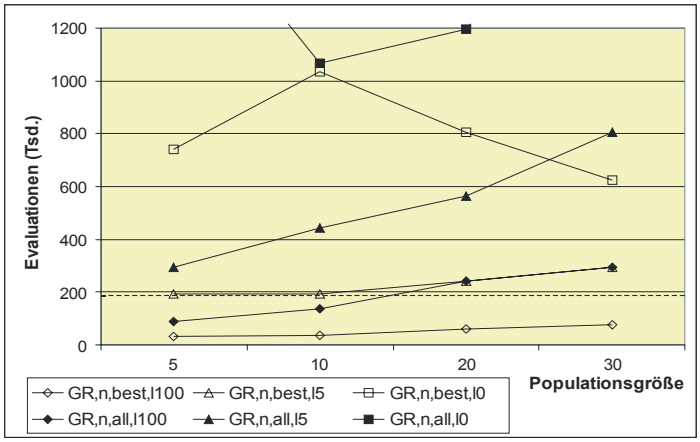
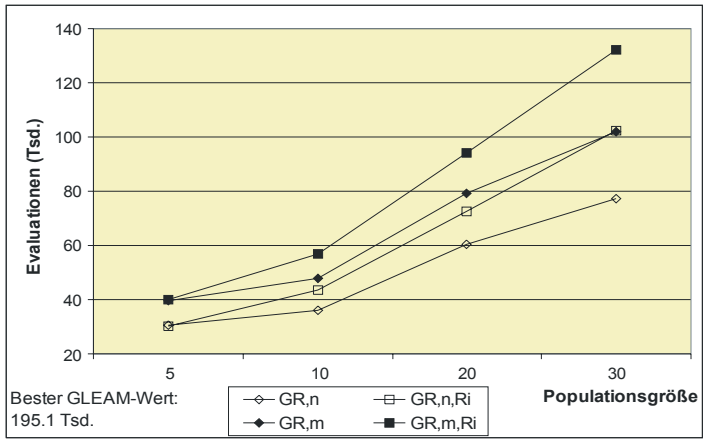


Abb. 5.73: Direkte Rosenbrock-Integration: Vergleich der Lamarckrate und der Verbesserung aller/des besten Nachkommens bei niedriger Präzision. (Fraktal)

Abb. 5.74 vergleicht die Ergebnisse bei reiner Lamarckscher Evolution und Optimierung des besten Nachkommens für die niedrige und die mittlere Präzisionen mit und ohne Voroptimierung. Auf Untersuchungen mit höheren Präzisionen mußte verzichtet werden, da bereits bei hoher Präzision erhebliche Konvergenzprobleme auftraten. Offenbar ist die erfolgreiche Durchführung von 100 Rosenbrockläufen bei zufälliger Initialisierung kein Garant dafür, zuverlässig bei mehreren Tausend Läufen in Verbindung mit evolutionär verbesserten Startpunkten zu konvergieren.



5.74: Direkte Rosenbrock-Integration: Vergleich zwischen Jobs mit niedriger und mittlerer Präzision mit und ohne Voroptimierung bei Lamarck-Evolution und Verbesserung des besten Nachkommens. (Fraktal)

In Abb. 5.75 sind die besten Jobs mit verzögerter direkter Integration im Vergleich mit dem besten unverzögerten dargestellt. Gegenüber der unverzögerten konnte keine Verbesserung erreicht werden.

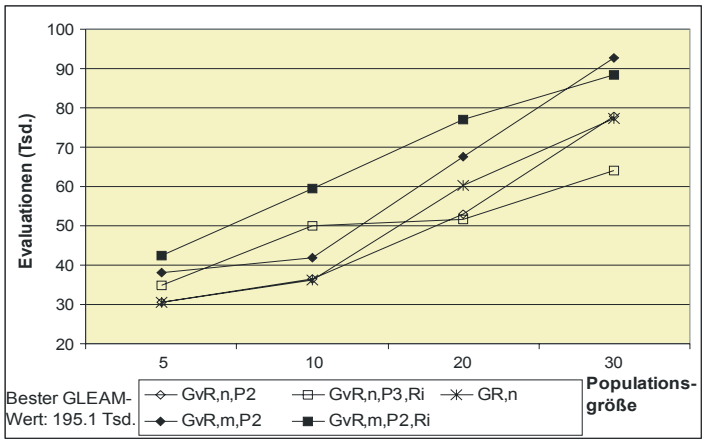


Abb. 5.75: Verzögerte direkte Rosenbrock-Integration: Vergleich der besten Jobs mit dem besten unverzögerten. Alles bei Lamarckscher Evolution und Verbesserung des besten Nachkommens. (Fraktal)

Abb. 5.76 faßt die Ergebnisse der besten erfolgreichen Jobs je Hybridisierungsart zusammen und vergleicht sie mit dem besten GLEAM-Job. Dabei handelt es sich ausschließlich um Jobs mit reiner Lamarck-Evolution und Verbesserung des besten Nachkommens. Abb. 5.77 fokussiert auf den Vergleich des besten GLEAM-Jobs mit solchen, die besser als GLEAM abschneiden. Die direkte Integration des Rosenbrock-Verfahrens bringt in ihrer verzögerten oder unverzögerten Variante ähnlich gute Ergebnisse wie der beste Job mit direkter Integration und voroptimierter Startpopulation (*GR,p5,n,Ri*), nämlich eine Reduzierung auf 15% vom Aufwand des günstigsten GLEAM-Jobs.

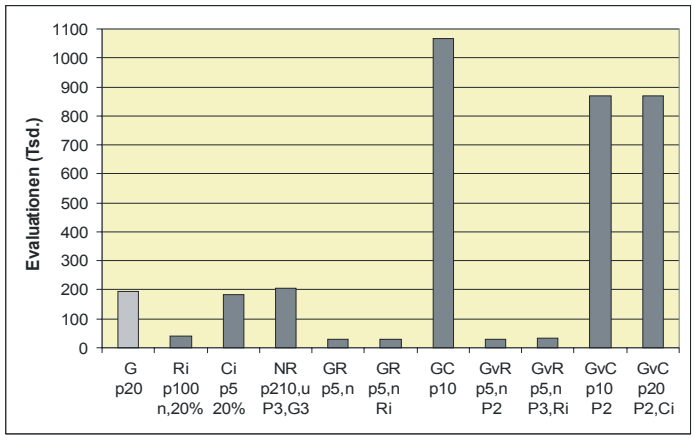


Abb. 5.76: Gesamtvergleich der besten Jobs je Hybridisierungsart mit 100% Erfolgsrate im Vergleich zum besten GLEAM-Job. Details siehe Abb. 5.77 (Fraktal)

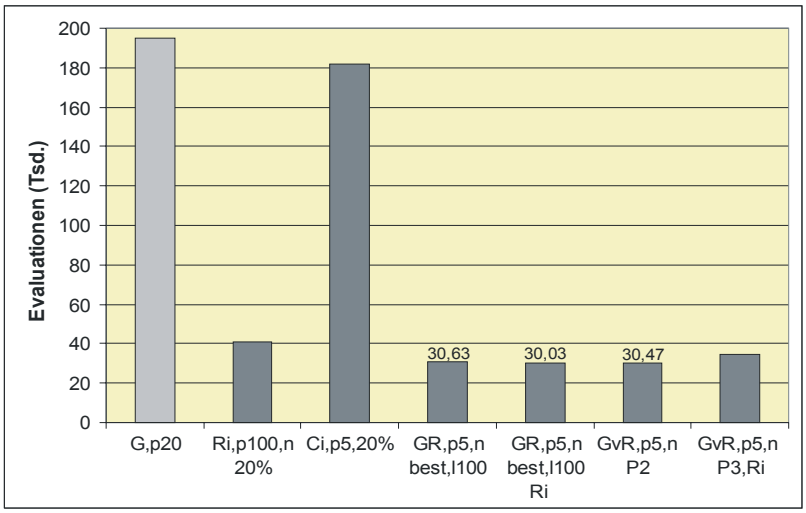


Abb. 5.77: Gesamtvergleich der besten Jobs je Hybridisierungsart mit 100% Erfolgsrate und geringerem Aufwand als der beste GLEAM-Job. Alle Jobs mit reiner Lamarck-Evolution und Optimierung des besten Nachkommens. (Fraktal)

5.2.1.6 Designoptimierung

Bei der Designoptimierungsaufgabe ist in Bild Abb. 5.78 deutlich zu erkennen, daß GLEAM bei einer Populationsgröße von etwa 210 am effektivsten arbeitet. Ab 180 und weniger Individuen sinkt die Erfolgsrate zunächst nur gering und dann immer stärker unter 100%. Gleichzeitig beginnt der Aufwand zunächst langsam und unterhalb einer Populationsgröße von 120 immer stärker zu steigen. Der Job *G,p210* liefert mit 5773 Evaluationen das günstigste Ergebnis.

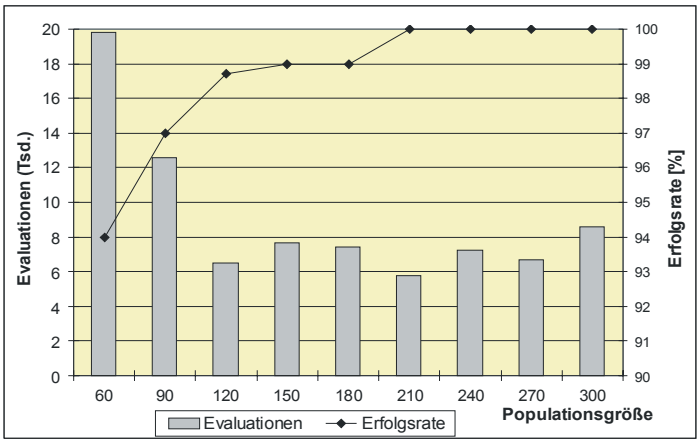


Abb. 5.78: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Design)

Auch die beiden lokalen Verfahren erreichen das Ziel, wenn auch mit niedriger Erfolgsquote, wie in Tabelle 5.4 dargestellt. Auf Grund von Konvergenzproblemen des Rosenbrock-Verfahrens bei sehr hoher Präzision sind hier nur Jobs zwischen niedriger und hoher Präzision aufgeführt. Da alle GLEAM und LSV-

Verfahren	Erfolgsrate [%]	Restarts	Evaluationen	Durchschnittsnote	Bei 99.5% Erfolgsrate: Anzahl Läufe	Evaluationen
Rosenbrock, n	4	0	94	62252	130	12220
Rosenbrock, m	3	0	232	60942	174	40368
Rosenbrock, h	15	0	764	60169	33	25212
Complex	12	0	102	59373	42	4284

Tab. 5.4: Ergebnisse des Rosenbrock- und des Complex-Algorithmus. Die Zielnote ist 80500. (Design)

Jobs mit jeweils 100 Läufen durchgeführt wurden, wird auch bei der Designoptimierungsaufgabe zum Vergleich mit GLEAM von einer Erfolgsquote in Höhe von 99.5% ausgegangen. Die beiden rechten Spalten von Tabelle 5.4 zeigen die sich daraus ergebenden Läufe und Aufwände. Lediglich der Complex-Algorithmus unterbietet den besten GLEAM-Job und zwar um 26%.

Die Voroptimierung mit dem Rosenbrock-Verfahren bringt bei keiner der drei anwendbaren Präzisionen eine Verbesserung gegenüber GLEAM. Bei den bis zu einer Populationsgröße von 120 durchgeführten Jobs kommt es nur bei hoher Präzision punktuell zu Erfolgsquoten von 100%, wie Abb. 5.79 zeigt. Bei steigender Populationsgröße und Anteil voroptimierter Individuen an der Startpopulation steigt allerdings auch der Anteil der Jobs, bei denen die Lösung bereits in der Startpopulation steckt, stark an. Das ist auch beim Complex-Algorithmus nicht viel anders. Ab einer Populationsgröße von 70 kommt es bei 100%-iger Erfolgsquote allerdings zu deutlichen Unterschreitungen des GLEAM- und zum Teil auch des Complex-Aufwands. Diese Jobs sind in Abb. 5.80 schraffiert dargestellt.

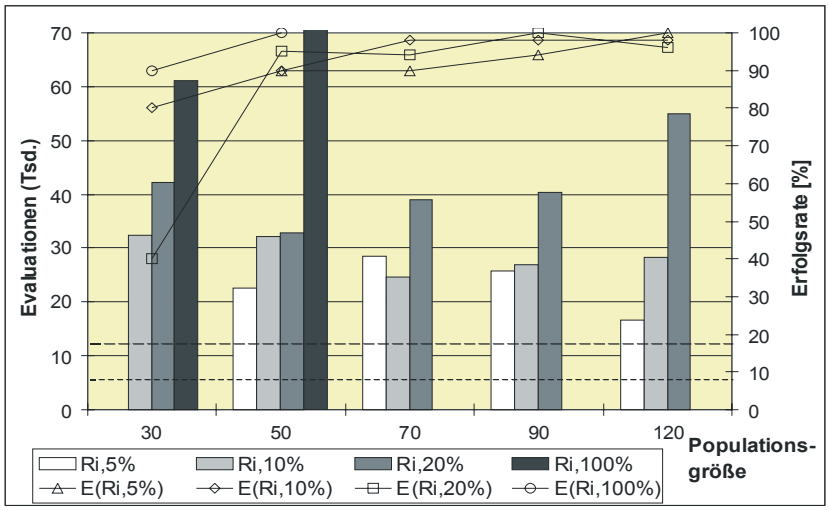


Abb. 5.79: GLEAM mit Rosenbrock-Voroptimierung: Vergleich verschiedener Anteile voroptimierter Individuen an der Startpopulation bei hoher Präzision. Grob gestrichelte Linie: reine Rosenbrock-Optimierung. (Design)

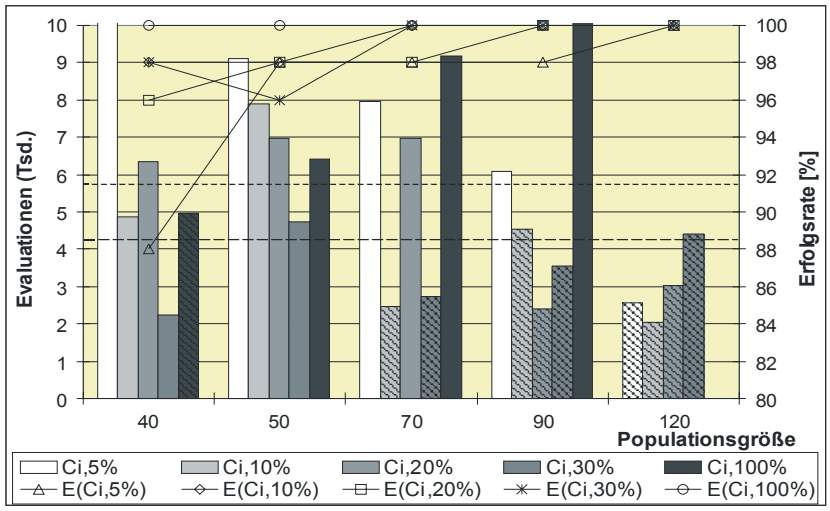


Abb. 5.80: GLEAM mit Complex-Voroptimierung: Vergleich verschiedener Anteile voroptimierter Individuen an der Startpopulation. Schraffiert: Erfolgreiche Jobs mit weniger Aufwand als GLEAM und zum Teil als die reine Complex-Optimierung (grob gestrichelte Linie). (Design)

Die Nachoptimierung mit dem Rosenbrock-Verfahren erreicht bei keinem Job eine Erfolgsquote von 100%, kommt jedoch bei steigender Populationsgröße immerhin auf 96%, siehe Abb. 5.81. Allerdings geht der Erfolg in steigendem Maße auf das Konto der Evolution, wie Abb. 5.82 zeigt. In Abb. 5.83 sind die Durchschnittsnoten und die kleinsten Noten der beiden

Jobs mit der kleinsten und der größten Minimalnote dargestellt. Die durchschnittlichen Notenwerte nähern sich ab einer Populationsgröße von 30 dem Zielwert von 80500 recht gut an. Allerdings erreicht der Aufwand dann auch den Bereich der reinen Complex-Optimierung. Daher ist von einer Vergrößerung der Population keine aus der eigentlichen Nachoptimierung resultierende Verbesserung zu erwarten.

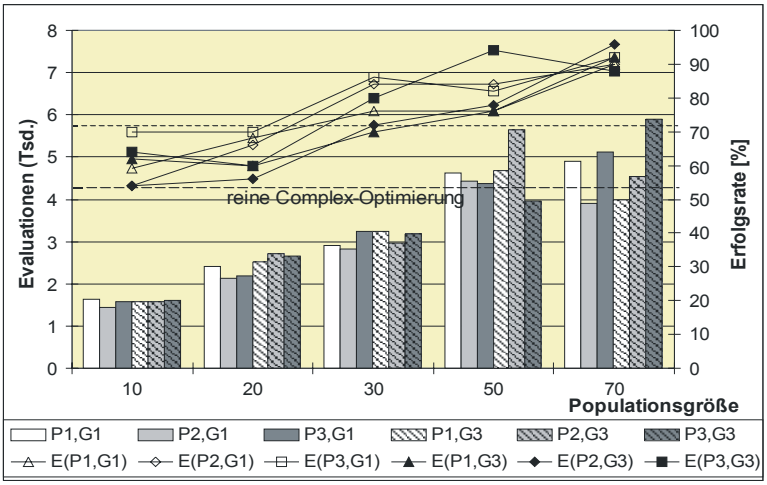


Abb. 5.81: Rosenbrock-Nachoptimierung: Vergleich der Parametrierungen bei hoher Präzision und verschiedenen Populationsgrößen. (Design)

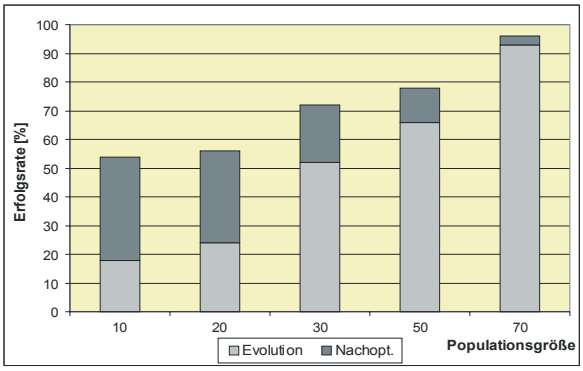


Abb. 5.82: Rosenbrock-Nachoptimierung: Erfolgsursachen beim erfolgreichsten Job NR,h,P2,G3. (Design)

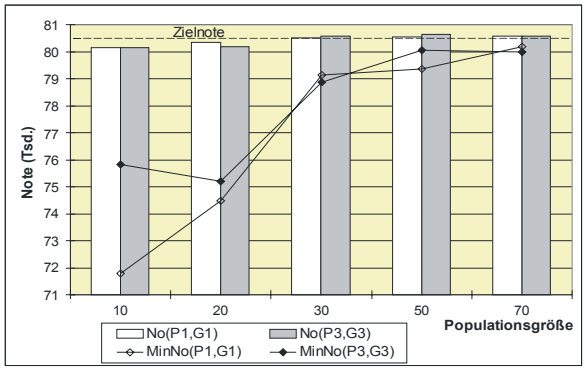


Abb. 5.83: Rosenbrock-Nachoptimierung: Durchschnitts- und Mindestnote bei den beiden Jobs mit der kleinsten und größten Mindestnote. (Design)

Günstiger sieht es bei der Nachoptimierung mit dem Complex-Algorithmus aus. Bereits bei der Verwendung der GLEAM-Resultate zur Bildung jeweils eines Startcomplexes werden ähnliche Werte und Trends wie bei der Nachoptimierung mit dem Rosenbrock-Verfahren erreicht, siehe Abb. 5.84 bis Abb. 5.86. Die Parametrierung *P2,G3* liefert hier die beste Erfolgsquote in Höhe von ebenfalls 96%.

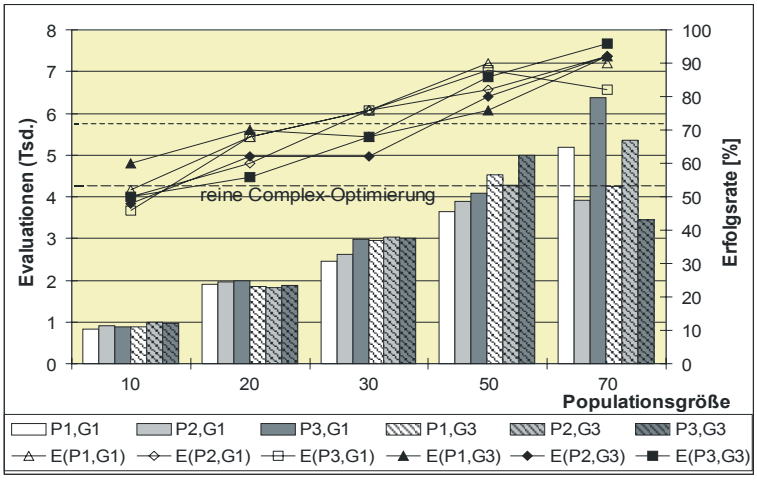


Abb. 5.84: Complex-Nachoptimierung der einzelnen GLEAM-Resultate (NC1S): Vergleich der Parametrierungen bei hoher Präzision und verschiedenen Populationsgrößen. (Design)

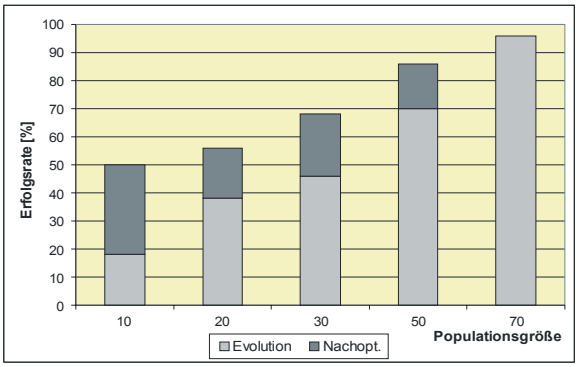


Abb. 5.85: Complex-Nachoptimierung (NC1S): Erfolgsursachen beim erfolgreichsten Job NC1S,P3,G3. (Design)

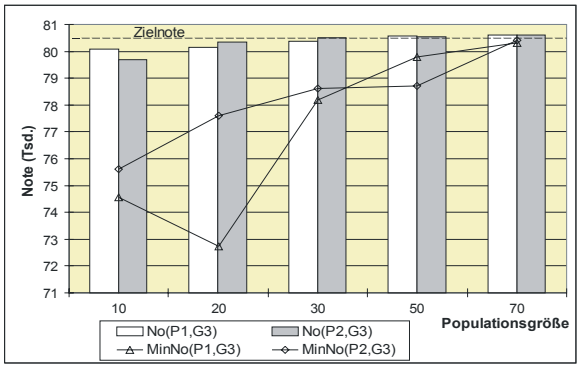


Abb. 5.86: Complex-Nachoptimierung (NC1S): Durchschnitts- und Mindestnote bei den beiden Jobs mit der kleinsten und größten Mindestnote. (Design)

Auch die bei Fletcher's Function getesteten Variationen der Nischenkriterien ändern an der Situation nichts Grundsätzliches, siehe Abb. 5.87. Es werden zwar bessere Erfolgsraten erzielt, aber der Anteil der Nachoptimierung daran nimmt mit steigender Populationsgröße stetig ab. Da die Details ähnlich wie bei der besser abschneidenden zweiten Variante der Complex-Nachoptimierung sind, wird auf deren nachfolgende Auswertung verwiesen.

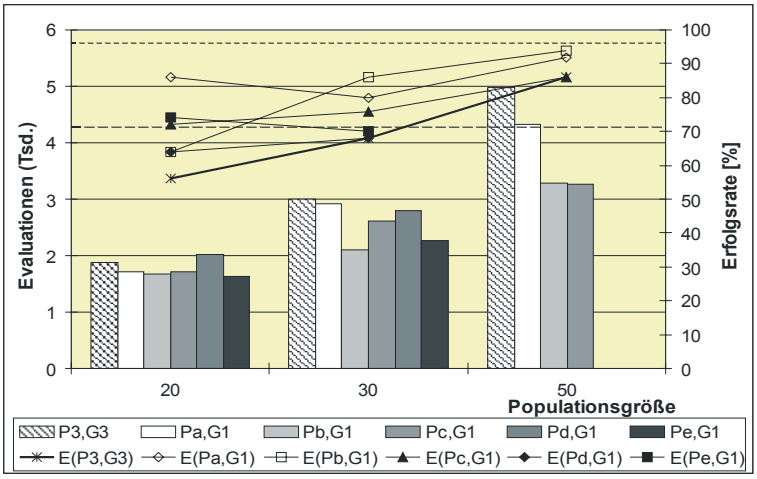


Abb. 5.87: Complex-Nachoptimierung (NC1S): Vergleich der besten Parametrierung NC1S, P3,G3 mit den variierten Nischenbedingungen von Tabelle 5.2. Grob gestrichelte Linie: reine Complex-Optimierung. (Design)

Wenn alle GLEAM-Resultate zusammen einen Startcomplex bilden, fallen die Ergebnisse günstiger als bei der zuvor behandelten getrennten Nachoptimierung der Resultate aus. Eine Parametrierung schafft sogar die 100%-Quote, siehe Abb. 5.88. Auch hier gehen die Erfolge mit steigender Populationsgröße in wachsendem Maße auf das Konto der Evolution und die Noten sind ab einer Populationsgröße von 50 auch im schlechtesten Fall noch recht nahe am Zielwert, siehe Abb. 5.89 und Abb. 5.90.

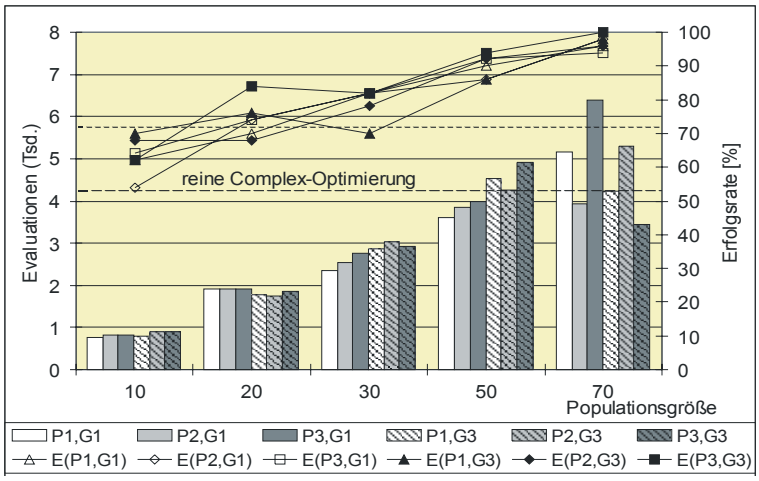


Abb. 5.88: Complex-Nachoptimierung ausgehend von allen GLEAM-Resultaten in einem Startcomplex (NC1C): Vergleich der Parametrierungen bei hoher Präzision und verschiedenen Populationsgrößen. (Design)

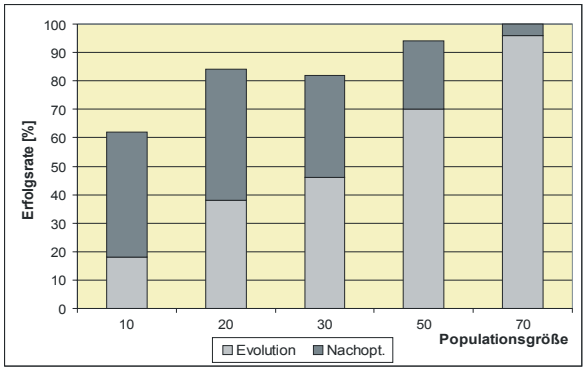


Abb. 5.89: Complex-Nachoptimierung (NC1C): Erfolgsursachen beim erfolgreichsten Job NC1C,P3,G3. (Design)

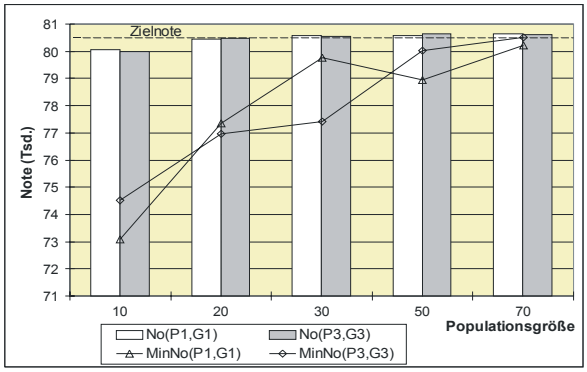


Abb. 5.90: Complex-Nachoptimierung (NC1C): Durchschnitts- und Mindestnote bei den beiden Jobs mit der kleinsten und größten Mindestnote. (Design)

Abb. 5.91 vergleicht die bereits erwähnten variierten Nischenkriterien mit der besten Parametrierung *NC1C,P3,G3*. Es ist eine kleine Verbesserung bei vermindertem Aufwand ab einer Populationsgröße von 30 feststellbar. Die Dominanz der Evolution wird durch Abb. 5.92 dokumentiert. Die Jobs mit der besten und schlechtesten Mindestnote fallen auf die Parametrierung *NC1C,Pb,G1* zusammen und sind in Abb. 5.93 dargestellt. Es zeigt auch, daß die Noten im gleichen Bereich wie die Standardparametrierungen liegen.

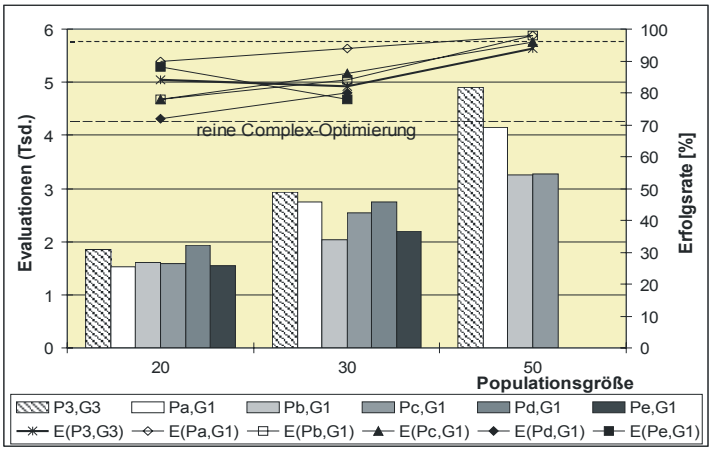


Abb. 5.91: Complex-Nachoptimierung (NC1C): Vergleich der besten Parametrierung NC1C,P3,G3 (schraffiert) mit den variierten Nischenbedingungen von Tabelle 5.2. (Design)

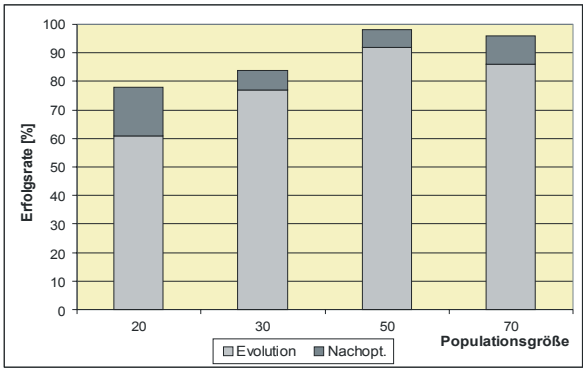


Abb. 5.92: Complex-Nachoptimierung (NC1C mit variierten Nischenbedingungen): Erfolgsursachen beim erfolgreichsten Job NC1C,Pb,G1. (Design)

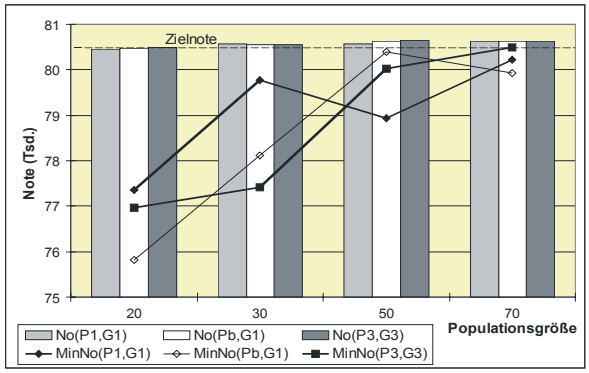


Abb. 5.93: Complex-Nachoptimierung (NC1C mit variierten Nischenbedingungen): Vergleich der Durchschnitts- und Mindestnote mit NC1C,P1,G1 und NC1C,P3,G3 (fette Linien). (Design)

Die besten Nachoptimierungsjobs fasst Abb. 5.94 zusammen. Eine Unterschreitung der Aufwände von GLEAM und der reinen Complex-Optimierung bei einer Erfolgsquote von 100%

konnte von Job *NC1C, p70,P3,G3* erreicht werden. Allerdings ist in den meisten Fällen die Evolution bereits allein erfolgreich, so daß mit steigender Populationsgröße die Nachoptimierung eine immer geringere Rolle spielt. Bemerkenswert ist aber, daß bei der Complex-Nachoptimierung in Einzelfällen weniger Evaluationen als bei den Einzelverfahren benötigt werden.

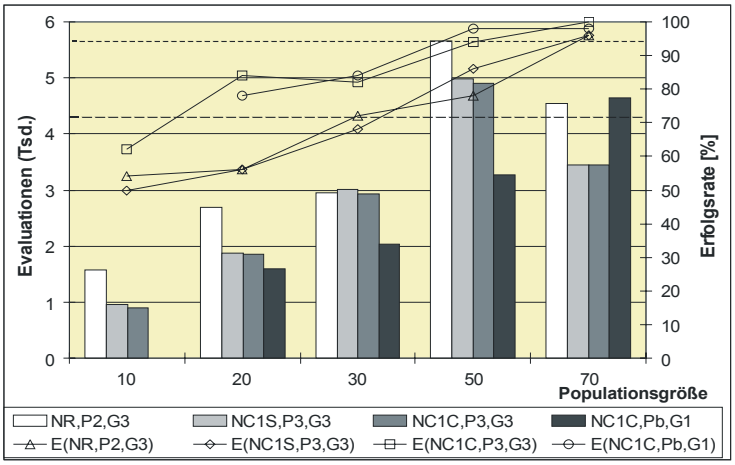


Abb. 5.94: Nachoptimierung: Vergleich der besten Parametrierungen bei beiden Verfahren. Grob gestreichelte Linie: reine Complex-Optimierung. (Design)

Bei der direkten Integration kann bei Verwendung des Rosenbrock-Verfahrens bei der Designoptimierungsaufgabe zum ersten Mal ein Vorteil für geringere Lamarckraten als 100% festgestellt werden. Abb. 5.95 zeigt die Ergebnisse bei niedriger Präzision und Abb. 5.96 bei mittlerer. Eine Lamarckrate von 5% bei Verbesserung nur des besten Nachkommens einer Paarung schneidet in beiden Fällen am besten ab. Der Versuch, mit voroptimierten Startpopulationen eine Verbesserung zu erreichen, scheiterte, wie Job *GR,m,best,l5,Ri* zeigt. Etwas anders sieht es dagegen beim direkt integriertem Complex-Algorithmus aus, siehe Abb. 5.97. Es werden nicht nur die Rosenbrock-Resultate erheblich unterboten, sondern die reine Baldwin-Evolution schneidet genauso gut ab wie die Lamarcksche. Die lokale Optimierung aller Nachkommen einer Paarung führt in allen untersuchten Fällen zu schlechteren Resultaten, wobei die Verschlechterungen beim Rosenbrock-Verfahren geringer ausfallen als beim Complex-Algorithmus.

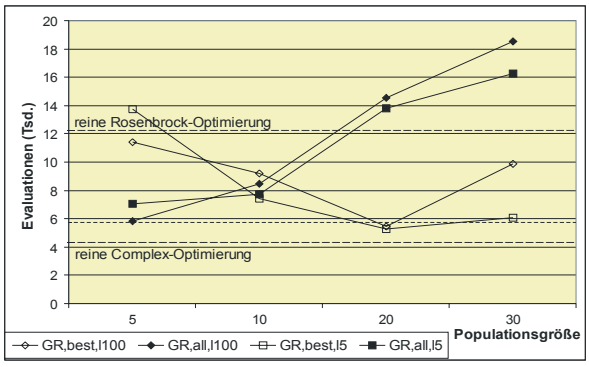


Abb. 5.95: Direkte Rosenbrock-Integration: Vergleich bei niedriger Präzision zwischen der Verbesserung des besten und aller Nachkommen bei unterschiedlichen Lamarckraten. (Design)

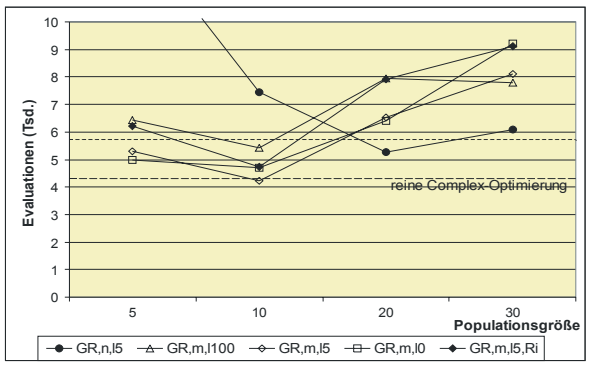


Abb. 5.96: Direkte Rosenbrock-Integration: Vergleich zwischen unterschiedlichen Lamarckraten bei mittlerer Präzision und der Verbesserung des besten Nachkommens. Job GR,n,best,l5 zum Vergleich. (Design)

Die verzögerte direkte Integration wird daher nur mit lokaler Optimierung des besten Nachkommen durchgeführt. Sie bringt beim Rosenbrock-Verfahren nochmal eine deutliche Verbesserung. Beim Complex-Algorithmus sind dagegen die Verbesserungen so gering, daß von gleichen Ergebnissen gesprochen werden kann. Abb. 5.98 zeigt die Ergebnisse beim Rosenbrock-Verfahren mit mittlerer und hoher Präzision: Der beste Job mit hoher Präzision unter-

unterbietet die unverzögerte Integration um mehr als 1000 Evaluationen (25%). Durch Job *GvR,h,best,l100,P1,Ri* wird deutlich, daß eine Voroptimierung der Startpopulation einen erheblichen Mehraufwand mit sich bringt. Das steht im Gegensatz zum Complex-Algorithmus, bei dem die Voroptimierung zu eher etwas günstigeren Ergebnissen führt, wie Abb. 5.99 zeigt. Die jeweils besten Jobs werden in Abb. 5.100 verglichen. Am besten schneidet die unverzögerte direkte Integration mit dem Complex-Algorithmus ab, da sie auch bei steigender Populationsgröße die besseren Resultate zeigt.

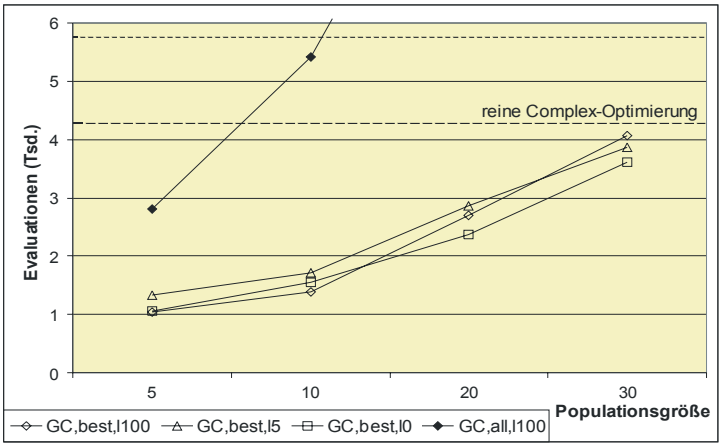


Abb. 5.97: Direkte Complex-Integration: Vergleich zwischen unterschiedlichen Lamarckraten bei Verbesserung des besten und aller Nachkommen. (Design)

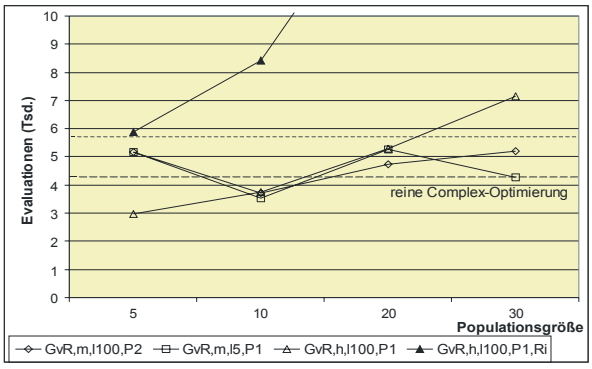


Abb. 5.98: Verzögerte direkte Rosenbrock-Integration: Vergleich der jeweils besten Jobs der verschiedenen Parametrierungen und der Voroptimierung. (Design)

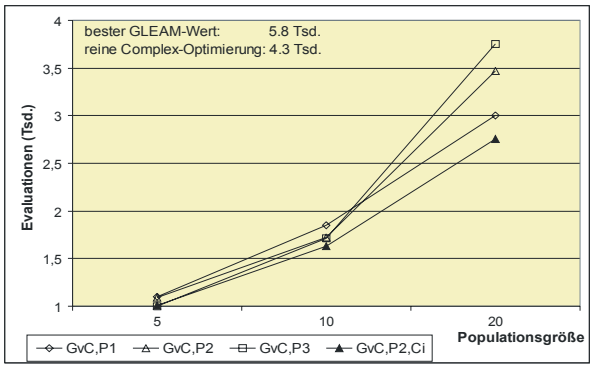


Abb. 5.99: Verzögerte direkte Complex-Integration: Vergleich der Zuschaltparametrierungen P1 bis P3 und der Voroptimierung. (Design)

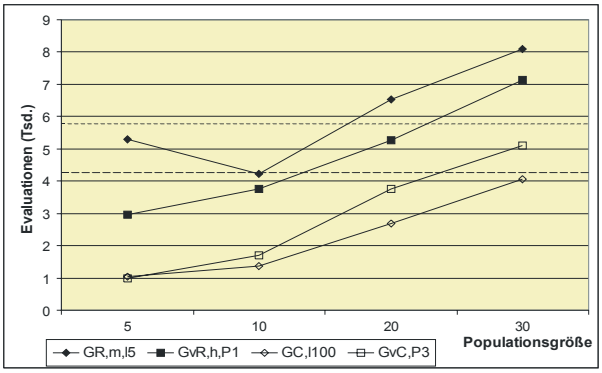


Abb. 5.100: Vergleich der besten Jobs der direkten verzögerten und unverzögerten Integration mit dem Rosenbrock- und dem Complex-Verfahren. Grob gestrichelte Linie: reine Complex-Optimierung. (Design)

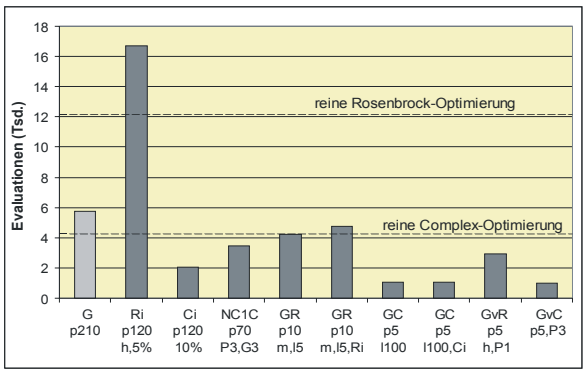


Abb. 5.101: Gesamtvergleich der besten Jobs je Hybridisierungsart und Parametrierung mit 100% Erfolgsrate im Vergleich mit dem besten GLEAM-Job. Alle Jobs mit Verbesserung des besten Nachkommens und reiner Lamarck-Evolution soweit nicht anders vermerkt. (Design)

Die besten Jobs je Hybridisierungsart und Parametrierung mit 100%-iger Erfolgsquote werden in Abb. 5.101 verglichen. Der Überblick zeigt, daß die Hybridisierung von GLEAM mit dem Rosenbrock-Verfahren bei dieser Aufgabe schlechter als die mit dem Complex-Algorithmus abschneidet. Abb. 5.102 zeigt einen Ausschnitt aus Abb. 5.101 und vergleicht nur die Jobs, die weniger Aufwand als GLEAM benötigen. Am besten schneidet der Job *GvC,p5,best, l100,P3* mit 17% Aufwand des besten GLEAM-Jobs und 23% der reinen Complex-Optimierung ab.

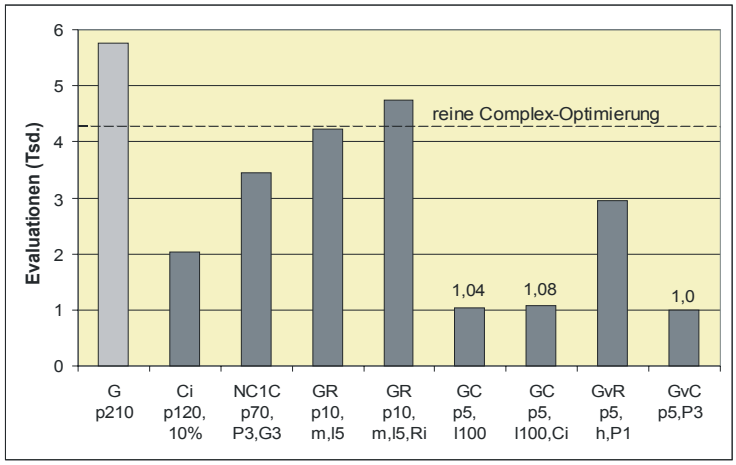


Abb. 5.102: Detailansicht des Gesamtvergleichs von Abb. 5.101: Es werden nur die Jobs dargestellt, die weniger Aufwand als der beste GLEAM-Job verursachen. Alle Jobs mit Verbesserung des besten Nachkommens und reiner Lamarck-Evolution soweit nicht anders vermerkt.(Design)

5.2.1.7 Ressourcenplanung

Die Zielstellung der Ressourcenplanungsaufgabe ist offenbar so nahe am globalen Optimum definiert, daß GLEAM vergleichsweise sehr großen Populationen benötigt, um sie zuverlässig zu lösen. Abb. 5.103 zeigt, daß erst ab einer Populationsgröße von 1800 die Erfolgsquote auf 100% steigt und bei durchschnittlich 5.4 Millionen Evaluationen das Ziel erreicht wird. Dieser Job wird daher zum Vergleich mit den Hybridisierungen herangezogen. Die Schwierigkeit des Problems wird auch daran deutlich, daß es bei größeren Populationen vereinzelt noch zu einem erfolglosen Lauf kommen kann, wie im Falle des Jobs *G,p2200*. Der fallende Aufwand bei geringeren Populationsgrößen erklärt sich durch Abbruch wegen Stagnation, meist festgestellt durch Erreichen von 500 Generationen ohne Akzeptanz, ansonsten durch 1000 Generationen ohne Deme-Verbesserung.

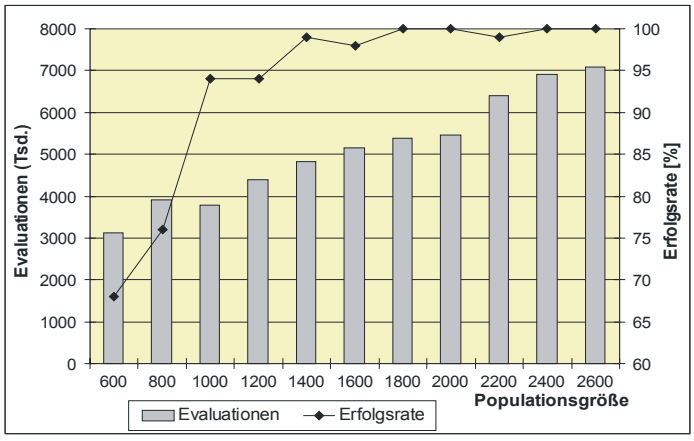


Abb. 5.103: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Ressourcen)

Bei der Ressourcenplanung muß von der bisherigen Art der Codierung der Parameter in den Aktionsketten, nach der jedem Parameter der Aufgabe eine Aktion mit ebenfalls einem Parameter entspricht, abgegangen werden. Jeder Charge eines Verfahrens ist ein Aktionstyp zugeordnet, dessen ganzzahliger Parameter die Startzeit in Stunden angibt. Der jeweilige Wertebereich ergibt sich aus der maximal zur Verfügung stehenden Zeit abzüglich der Produktionszeit des Verfahrens und der Mindestherstellungszeit eventueller Vorprodukte. Damit ergeben sich unterschiedliche Wertebereiche, die bei der konkreten Aufgabenstellung zwischen 472 und 1656 liegen. Die Reihenfolge der Aktionen und damit der Chargenstarts bestimmt, welcher

Charge im Falle eines Belegungskonflikts der Vorrang eingeräumt wird. Damit spielt der kombinatorische Anteil der Codierung gegenüber den Parametern insofern eine untergeordnete Rolle, als die Reihenfolge nur bei Konflikten wichtig ist und durch eine geeignete Veränderung der Parameter überspielt werden kann.

Zur Bearbeitung der Planungsaufgabe durch die lokalen Suchverfahren ist zu klären, wie erstens die Anpassung zwischen den ganzzahligen Parametern der Aufgabe und den reelwertigen der Suchverfahren erfolgen und zweitens wie mit dem kombinatorischen Anteil an den von GLEAM generierten Lösungen umgegangen werden soll. Es wird festgelegt, daß der kombinatorische Anteil nicht Gegenstand der lokalen Optimierung ist und vollständig unter der Kontrolle der Evolution verbleibt. Die ganzzahligen Parameter werden auf reelwertige abgebildet und die Ergebnisse gerundet, was einen ausreichend großen Wertebereich (größer als 100) voraussetzt. Da im vorliegenden Fall die Wertebereiche 472 und größer sind, sind die Voraussetzungen für eine vollständige Behandlung der Parameter durch die lokalen Verfahren gegeben. Beim Complex-Algorithmus entfällt die Variante der Verwendung aller GLEAM-Ergebnisse zur Bildung eines einzigen Startcomplexes, da die Parameter aller Eckpunkte die gleiche semantische Bedeutung haben müssen, wenn der Complex-Algorithmus sinnvoll agieren können soll.

Erste Versuche legen beim Rosenbrock-Verfahren die Verwendung einer modifizierten Parametrierung nahe, da die bisher üblichen Präzisionseinstellungen keine Konvergenz ergeben. In Voruntersuchungen haben sich folgende Parameter für die lokale Verbesserung frei ausgewürfelter Lösungen mit schlechten Notenwerten (bis zu 2), wie sie bei der Initialisierung entstehen, und vorverbesserter mittlerer Qualität im Notenbereich von 25000 bis 32000 als geeignet gezeigt: Die initiale Schrittweite wird von 10% auf 40% des Wertebereichs eines Parameters erhöht und die Abbruchschranke wird auf 0.6 festgelegt, was deutlich außerhalb der bisher üblichen Werte zwischen 10^{-2} und 10^{-9} liegt (Sonderpräzision *s*).

Die beiden lokalen Verfahren erreichen das Ziel nicht und liefern auch recht schlechte Fitnesswerte ab, siehe Tabelle 5.5. Sie benötigen dazu beide durchschnittlich 7 Minuten, ein hoher Wert, der intensiven Untersuchungen hybrider Strategien im Wege steht.

Verfahren	Erfolgsrate [%]	Restarts	Evaluationen	Durchschnittsnote
GLEAM, p1800	100	-	5376334	100000
GLEAM, p2000	100	-	5460446	100000
Rosenbrock	0	0	3091	27451
Complex	0	0	473	2404

Tab. 5.5: Ergebnisse von GLEAM, Rosenbrock- und Complex-Algorithmus im Vergleich. (Ressourcen)

Die Voroptimierung der Startpopulation bringt weder beim Rosenbrock- noch beim Complex-Algorithmus eine Verbesserung der Situation, wie Abb. 5.104 zeigt.

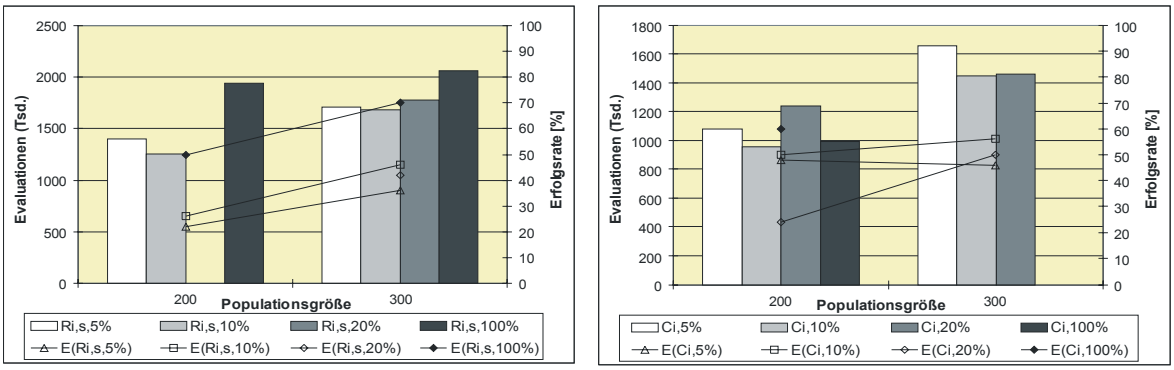


Abb. 5.104:GLEAM mit Voroptimierung: Vergleich verschiedener Anteile voroptimierter Individuen an der Startpopulation für das Rosenbrock- (links) und das Complex-Verfahren (rechts). (Ressourcen)

Für die Nachoptimierung reichen auf Grund der erweiterten Chromosomeigenschaften die bisherigen Parametrierungen für die Nischenbildungskontrolle nicht aus. Tabelle 5.6 faßt die neuen zusätzlichen Parametrierungen für die Ressourcenplanung und die Roboterbahnplanung zusammen. Mit ihnen soll geprüft werden, ob sich mit schärferen Kriterien (P0) oder mit gelockerten Schranken (P4 - P7) für die Nischenbildung bessere Ergebnisse erzielen lassen als mit den bisher bei der reinen Parameteroptimierung benutzten Werten (P1 - P3).

Kennung	ε	ε_{Pop}
P0	0.001	0.005
P4	0.01	0.03
P5	0.02	0.05
P6	0.02	0.1
P7	0.04	0.08

Tab. 5.6: Zusätzliche Variationen von ε und ε_{Pop} für die Ressourcenoptimierung und die Roboterbahnplanung.

Abb. 5.105 zeigt die Ergebnisse für das Rosenbrock-Verfahren. Bei einem recht hohen Aufwand wird bestenfalls lediglich eine Erfolgsquote von 92% erreicht (*NR,p400,s,P1, G3*). Auch hier geht bei steigender Erfolgsquote und größeren Populationen der Erfolg in steigendem Maße auf das Konto der Evolution, wie in Abb. 5.106 für den besten Job dargestellt ist. Die erreichten Noten der Jobs mit den schlechtesten und den besten Minimalnoten zeigen, daß die Ergebnisse vergleichsweise stark streuen und die Nachoptimierung keinen verläßlichen Lösungsansatz bei der Ressourcenoptimierung darstellt, siehe Abb. 5.107.

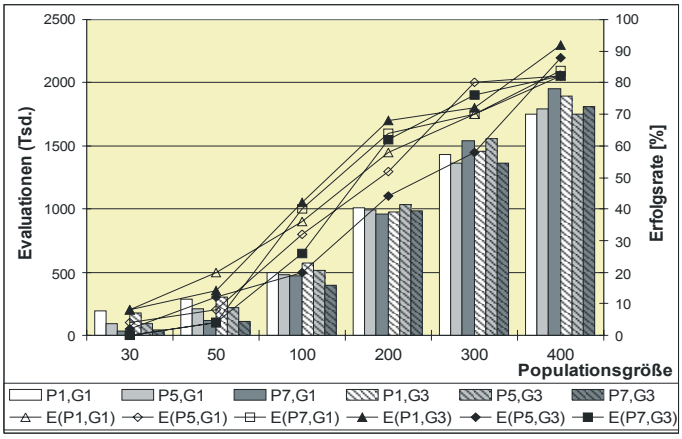


Abb. 5.105:Rosenbrock-Nachoptimierung: Vergleich dreier Parametrierungen bei GDV/GAk=1 und 3 (Sonderpräzision s). (Ressourcen)

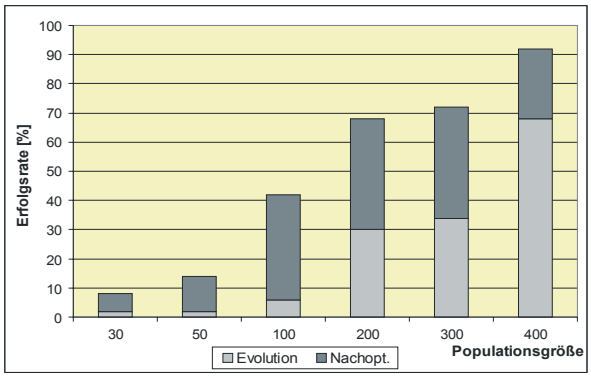


Abb. 5.106: Rosenbrock-Nachoptimierung: Erfolgsursachen beim erfolgreichsten Job NR,s P1,G3. (Ressourcen)

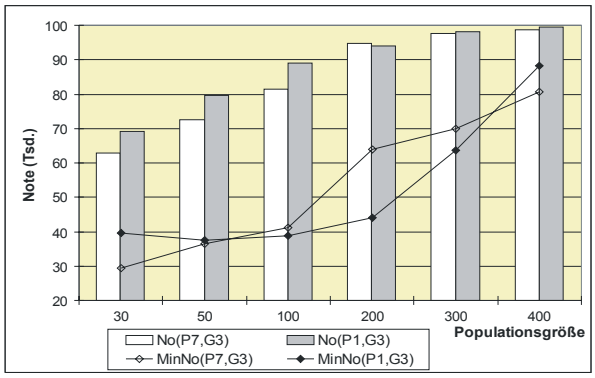


Abb. 5.107:Rosenbrock-Nachoptimierung: Durchschnitts- und Mindestnote bei den beiden Jobs mit der kleinsten und größten Mindestnoten. (Ressourcen)

Die Nachoptimierung mit dem Complex bringt überhaupt keinen Erfolg, was angesichts der Ergebnisse der reinen Complex-Optimierung von Tabelle 5.5 auch nicht erstaunlich ist. Die in Abb. 5.108 dargestellte Erfolgsquote geht in vollem Umfang auf erfolgreiche Evolutionsläufe, die das Ziel vor Herausbildung der Nischen erreicht haben, zurück, siehe auch Anhang B.7.3. Daher und wegen des recht hohen Zeitaufwands wurde auf die Untersuchung weiterer Parametrierungen verzichtet.

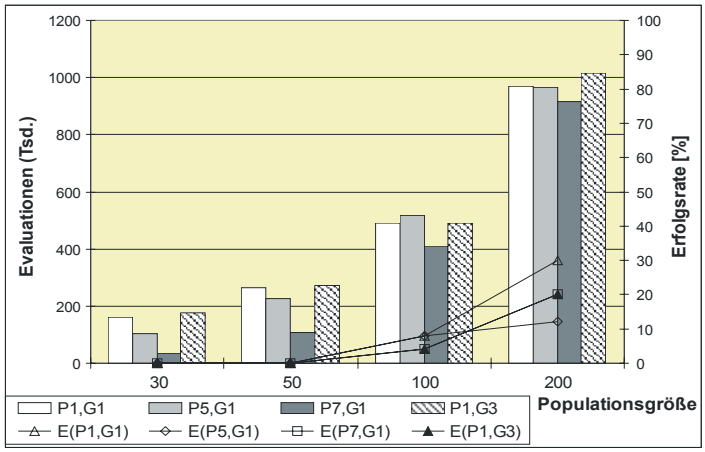


Abb. 5.108: Complex-Nachoptimierung (NC1S): Vergleich der Parametrierungen von Abb. 5.105. (Ressourcen)

Günstiger sieht es dagegen bei der direkten Integration mit dem Rosenbrock-Verfahren aus. Bei einer Erfolgsquote von 100% konnte bei reiner Lamarckscher Evolution und Verbesserung nur des besten Nachkommens einer Paarung die Anzahl der Evaluationen auf 69448 gedrückt werden, das sind 1.3% des Aufwands des besten GLEAM-Jobs. Versuche mit der Verbesserung aller Nachkommen und/oder geringeren Lamarckraten führten bei einer Populationsgröße von 10 zu Laufzeiten von 20 bis 30 Stunden bei einem Aufwand von 3.8 bis 5.3 Millionen Evaluationen, so daß auf weitere Experimente in dieser Richtung verzichtet werden mußte, siehe Anhang

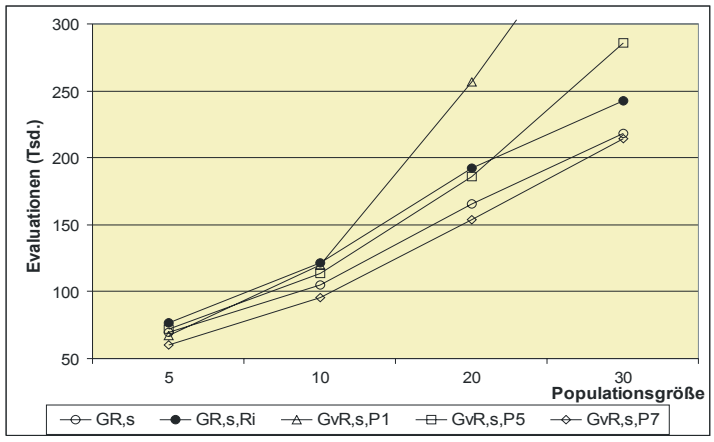


Abb. 5.109: Verzögerte und unverzögerte direkte Rosenbrock-Integration: Vergleich der besten Parametrierungen. Alle Jobs mit Lamarckscher Evolution bei Verbesserung des besten Nachkommens einer Paarung. (Ressourcen)

B.7.4. In Abb. 5.109 werden die Ergebnisse der Parametrierung *GR,s,best,L100* mit und ohne Voroptimierung miteinander verglichen. Es läßt sich eine leichte Überlegenheit zufallsinitialisierter Startpopulationen feststellen.

Die verzögerte direkte Integration führt bei Nischen-Parametrierung P7 noch einmal zu einer geringen Verbesserung, wie Job *GvR,s,best,L100,P7* in Abb. 5.109 zeigt. P7 unterbietet als einziger der untersuchten Parametrierungen zuverlässig die unverzögerte direkte Integration. Versuche mit der Voroptimierung der besten Jobs führten zu keiner Verbesserung gegenüber der zufallsinitialisierten Variante, siehe Anhang B.7.5.

Die verzögerte und unverzögerte direkte Integration des Complex-Verfahrens ergibt bei sehr hohen Laufzeiten zwischen 70 und 157 Stunden pro Lauf keinen Erfolg, siehe Anhang B.7.4 und B.7.5. Daher wurden keine weiteren Untersuchungen mit dem Complex-Algorithmus durchgeführt.

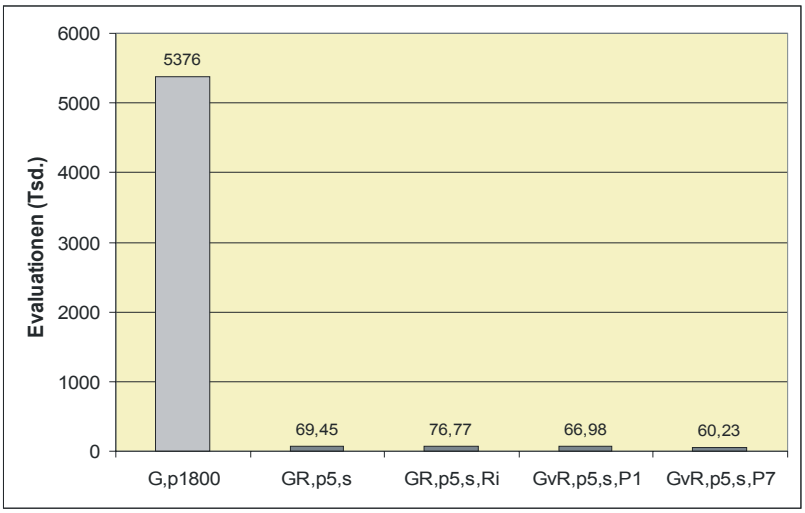


Abb. 5.110: Gesamtvergleich der besten Jobs mit 100% Erfolgsquote mit GLEAM. Alle Jobs mit Lamarckscher Evolution bei Verbesserung des besten Nachkommens einer Paarung. (Ressourcen)

Abb. 5.110 vergleicht die drei besten Läufe der (verzögerten) direkten Rosenbrock-Integration mit dem besten GLEAM-Lauf. Der Aufwand konnte auf 1.1% gesenkt werden. Außerdem zeigen die Hybriden von Abb. 5.110 bei keiner Parametrierung auch nur einen Fall von Nichterreichung der Zielvorgaben, siehe Abb. 5.109. Damit kann eine sicherere Konvergenz durch die (verzögerte) direkte Rosenbrock-Integration als bei den reinen GLEAM-Jobs konstatiert werden.

5.2.1.8 Kollisionsfreie Roboterbahnplanung

Bei der Roboterbahnplanung kann der Complex-Algorithmus wie auch bei der vorherigen Aufgabe nur so eingesetzt werden, daß jedes GLEAM-Ergebnis jeweils einen Startcomplex bildet. Die Aktionsparameter des in Abschnitt 2.2.3 beschriebenen Aktionsmodells sind bis auf die Anzahl der Takte, bei denen die aktuellen Einstellungen unverändert weiter wirken sollen, alle reelwertig. Da die Taktanzahl mit einem Wertebereich zischen 1 und 20 zu klein für die im vorigen Abschnitt beschriebene Abbildung auf die reelwertigen Größen der lokalen Verfahren ist, gehört sie zusammen mit den Reihenfolgeinformationen der Bewegungsbefehle zu dem für die lokalen Verfahren invarianten Teil eines Lösungsvorschlags. Der Rosenbrock- und der Complex-Algorithmus operieren also nur über die Beschleunigungs- und Geschwindigkeitswerte der Motoren.

Da bei der Roboterbahnplanung nicht nur die Reihenfolge der Aktionen relevant ist, sondern zusätzlich noch die Kettenlänge verändert wird, funktionieren ähnlich wie bei der Ressourcenplanung die bisher üblichen Präzisionseinstellungen nicht mehr. Folgende Rosenbrock-Parametrierung (Sonderpräzision *s*) wurde in Vorversuchen für Aktionsketten minderer (Noten-

werte zwischen 11 und 3000) und vorverbesserter mäßiger Qualität (Notenwerte zwischen 10000 und 50000) ermittelt: Die initiale Schrittweite wird von 10% auf 50% des Wertebereichs eines Parameters erhöht und die Abbruchschranke wird auf 0.7 festgelegt. Sie liegt damit deutlich außerhalb der bisher üblichen Werte zwischen 10^{-2} und 10^{-9}.

Bei den Jobs, bei denen die Nischenbildung relevant ist, werden auch die zusätzlichen Parametrierungen von Tabelle 5.6 benutzt.

Abb. 5.111 zeigt die GLEAM-Ergebnisse. Ab einer Populationsgröße von 120 wird die Aufgabe zuverlässig gelöst und bei einem Wert von 150 wird mit 312318 Individuen der geringste Aufwand benötigt. Kleinere Populationsgrößen führen auch wegen der vergleichsweise spät abgebrochenen Fehlläufe zu einer stark ansteigenden Evaluationsrate. Das ist in geringerem Maße auch bei steigender Populationsgröße ab 180 der Fall.

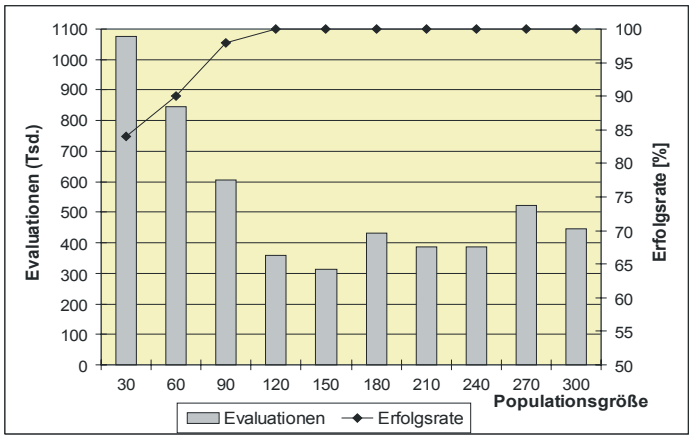


Abb. 5.111: GLEAM: Vergleich unterschiedlicher Populationsgrößen. (Roboter)

Die beiden lokalen Verfahren lösen die Aufgabe nicht und die erreichten Noten sind nur von mäßiger (Rosenbrock-Verfahren) oder ausgesprochen schlechter Qualität (Complex-Algorithmus).

Verfahren	Erfolgsrate [%]	Restarts	Evaluationen	Durchschnittsnote	Maximalnote
Rosenbrock	0	0	10064	3468	58634
Complex	0	0	66	812	6595

Tab. 5.7: Ergebnisse von Rosenbrock- und Complex-Algorithmus. (Roboter)

Bei der Voroptimierung tritt zum ersten Mal das interessante Phänomen auf, daß die Voroptimierung der gesamten Startpopulation bei beiden lokalen Verfahren die Erfolgsquote unter 100% bei allen untersuchten Populationsgrößen drückt. 100%-Quoten kommen beim Rosenbrock-Verfahren nur bei einer Voroptimierungsrate von 5% und 10% vor, beim Complex-Al-

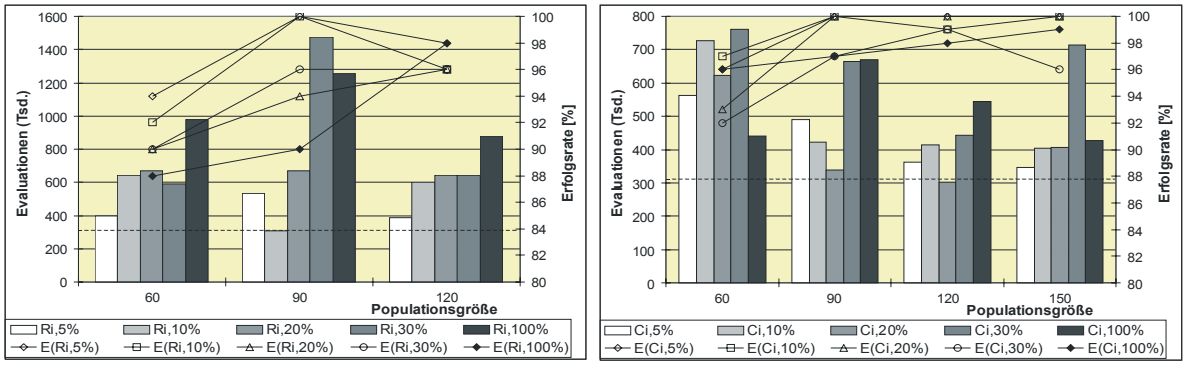


Abb. 5.112: Voroptimierung mit dem Rosenbrock- (links) und dem Complex-Verfahren (rechts). (Roboter)

gorithmus auch noch bei 20%, siehe Abb. 5.112. Der kleinste GLEAM-Aufwand konnte nur in zwei Fällen geringfügig unterboten werden, nämlich bei den Jobs *G,p90,s,10%,Ri* und *G,p120,20%,Ci*.

Die Nachoptimierung führt weder bei Verwendung des Rosenbrock-Verfahrens noch des Complex-Algorithmus' zu Erfolgsquoten über 90%, wie Abb. 5.113 und Abb. 5.114 zeigen. Die Erfolge gehen fast ausschließlich auf das Konto der Evolution, siehe Abb. 5.115, und die erreichten Notenverbesserungen fallen eher gering aus. Bei kleinen Populationen werden die Notenwerte um maximal 7600 beim Rosenbrock-Verfahren bzw. 4200 beim Complex-Algorithmus verbessert und bei großen schrumpfen sie auf bis zu 22 beim Rosenbrock-Verfahren bzw. bis auf 0 beim Complex-Algorithmus. Da auch noch der Aufwand mit steigender Erfolgsquote die von GLEAM gesetzte Marge deutlich überschreitet, konnte auf weitere Untersuchungen verzichtet werden. Es zeigt sich, daß die nachträgliche Verbesserung nur von Parametern ohne Einfluß auf die Struktur der Befehlssequenz bei der Roboterbahnplanung nicht ausreicht, um eine funktionierende Nachoptimierung zu erhalten. Im wesentlichen findet hier eine Evolution mit der Nischenbildung als zusätzlichem Abbruchkriterium statt.

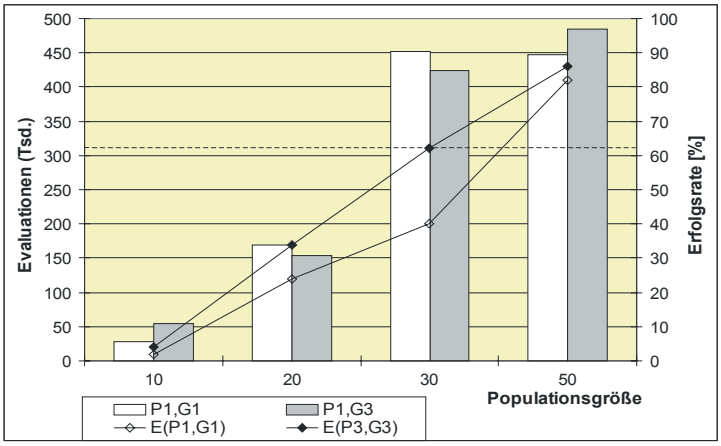


Abb. 5.113: Rosenbrock-Nachoptimierung: Vergleich zweier Parametrierungen bei variierenden Populationsgrößen. (Roboter)

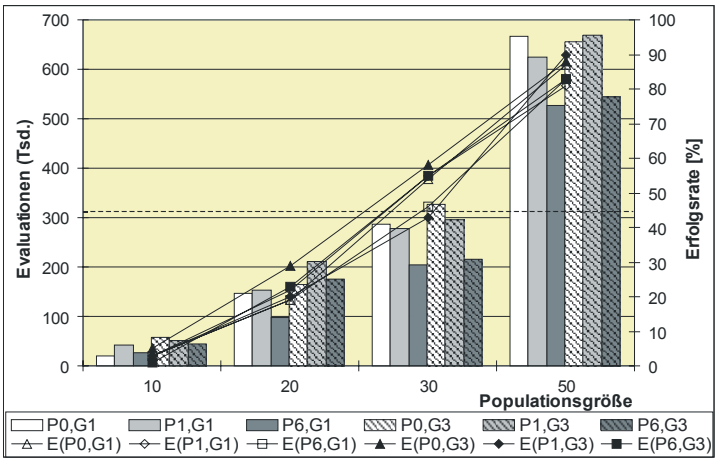


Abb. 5.114: Complex-Nachoptimierung: Vergleich mehrerer Parametrierungen bei variierenden Populationsgrößen. (Roboter)

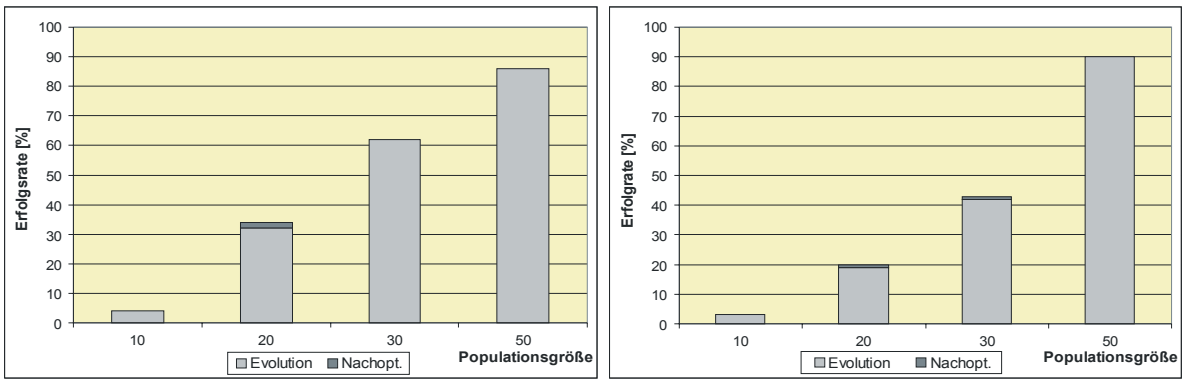


Abb. 5.115: Nachoptimierung: Erfolgsursachen des jeweils erfolgreichsten Jobs beim Rosenbrock- (links) und beim Complex-Verfahren (rechts). (Roboter)

Da die direkte Integration bei beiden Verfahren zu sehr langen Laufzeiten von 30 Stunden und mehr führt, konnten nur wenige Jobs und das auch nur bei eingeschränkter Laufzahl durchgeführt werden, siehe Anhang B.8.4. Versuche mit geringeren Lamarckraten als 100% oder der Verbesserung aller Nachkommen einer Paarung brachten keinen Erfolg und mußten wegen nochmals gesteigerter Laufzeiten von bis zu 100 Stunden abgebrochen werden. Abb. 5.116 vergleicht die Ergebnisse der direkten Integration mit beiden Verfahren bei Verbesserung nur des besten Nachkommens einer Paarung und reiner Lamarck-Evolution. Da die Werte auf 10 Läufen bzw. ab einer Populationsgröße von 20 auf nur noch 5 Läufen pro Job beruhen, sind günstige Erfolgsquoten, wie bei *GC,p30,best,l100* mit Vorsicht zu betrachten. Auch die Voroptimierung bringt keine Verbesserung, wie das Beispiel des Jobs *GC,best,l100,Ci* zeigt. Der Vergleich mit dem besten GLEAM-Job (gestrichelte Linie) macht den enormen Rechenaufwand deutlich.

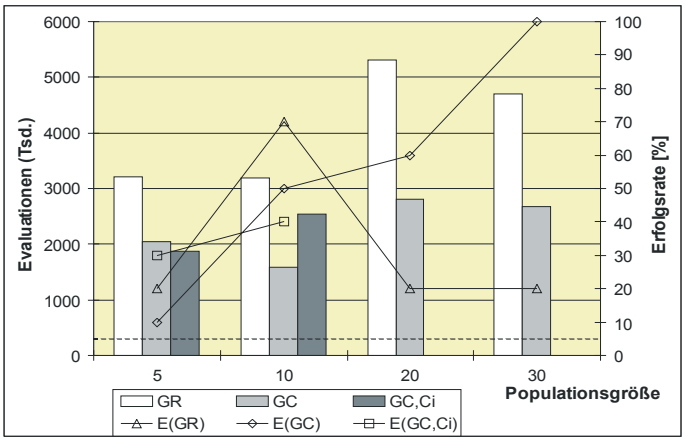


Abb. 5.116: Vergleich der direkten Integration mit beiden lokalen Verfahren bei reiner Lamarcksche Evolution und Verbesserung des besten Nachkommens. (Roboter)

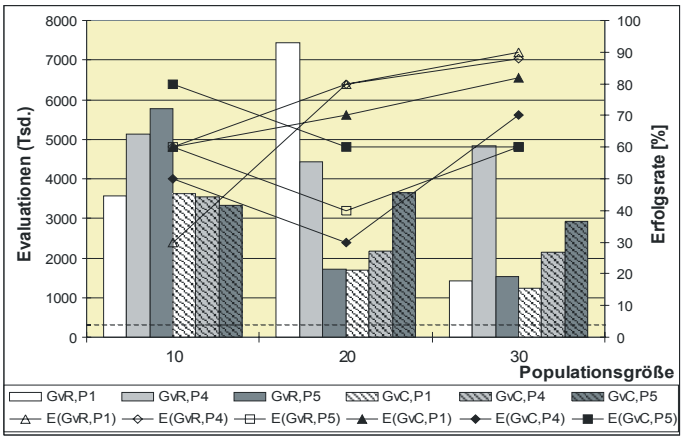


Abb. 5.117: Vergleich der verzögerten direkten Integration mit beiden lokalen Verfahren bei mehreren Parametrierungen (reine Lamarcksche Evolution und Verbesserung des besten Nachkommens). (Roboter)

Die verzögerte direkte Integration bringt keine wesentlich anderen Ergebnisse, was angesichts des Mißerfolgs der unverzögerten auch nicht überrascht. Abb. 5.117 zeigt die Ergebnisse im Detail. Auch hier muß ein erheblicher Mehraufwand im Vergleich zum besten GLEAM-Job konstatiert werden.

Zusammenfassend kann festgestellt werden, daß die Hybridisierung bei der Roboterbahnplanung zu keiner nennenswerten Verbesserung geführt hat. Lediglich die Voroptimierung mit dem Rosenbrock-Verfahren bringt magere 3% weniger Aufwand, siehe Abb. 5.118. Der Grund liegt im hohen kombinatorischen Anteil bei dieser Aufgabenstellung. Die lokalen Verfahren haben weder Einfluß auf Anzahl und Reihenfolge der Roboterbefehle, noch können sie die nicht unwichtige *Unverändert*-Aktion parametrieren. Offenbar reicht der verbliebene Spielraum nicht für eine sinnvolle Anwendung der bei allen anderen Aufgaben erfolgreichen Hybridisierung aus.

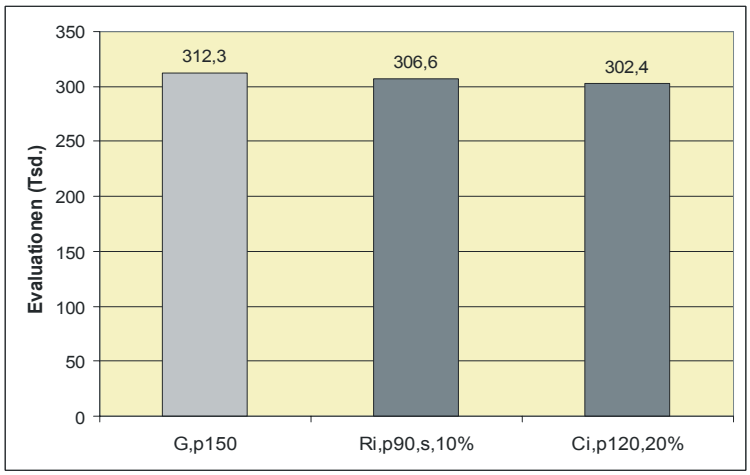


Abb. 5.118: Gesamtvergleich aller Jobs mit 100% Erfolgsquote, die besser als GLEAM abschneiden. (Roboter)

5.2.2 Ergebnisse der Integrationsarten

Zum Vergleich der Integrationsarten werden nur Jobs mit einer Erfolgsrate von 100% herangezogen. Verglichen wird an Hand der gegenüber GLEAM erreichten Verbesserung (Quotient der vom besten GLEAM-Job benötigten Evaluationen durch die Evaluationen des zu vergleichenden Jobs). In den Schaubildern ist der Verbesserungsfaktor Eins entweder durch eine gestrichelte Linie verdeutlicht oder bei geeigneter Skalierung durch eine dicker gezeichnete Hilfslinie. Damit sollen Quotienten kleiner als Eins, die ja eine Verschlechterung darstellen, deutlich von geringen Verbesserungen zu unterscheiden sein. Die Auswahl der Jobs für die Gegenüberstellungen berücksichtigt neben der erreichten Noten- und Aufwandsverbesserung auch die (in den Schaubildern nicht dargestellte) erzielte Gesamtqualität der Lösung. Durch den Vergleich soll geklärt werden, ob sich ein Trend erkennen läßt, von dem eine allgemeingültige Parametrierung abgeleitet werden kann.

5.2.2.1 Voroptimierung

Die Voroptimierung mit dem Rosenbrock-Verfahren ist nur bei den mathematischen Benchmarkfunktionen erfolgreich. Bei der Ressourcenplanung nützt sie gar nichts und bei den beiden anderen Aufgaben mit realem Hintergrund bringt sie keine nennenswerte Verbesserung. Da sie auch bei Schwefel's Sphere nicht zum Erfolg führt, ist diese Testfunktion in Abb. 5.119, die einen Überblick gibt, weggelassen. Der große Erfolg der Voroptimierung bei Fletcher's Function macht eine Darstellung ohne sie notwendig, um die Details deutlicher werden zu lassen, siehe Abb. 5.120. Die Trends sind hinsichtlich der Populationsgröße recht uneinheitlich: Als günstigste Größen stellen sich je nach Aufgabe Werte von 5, 30, und 100 heraus.

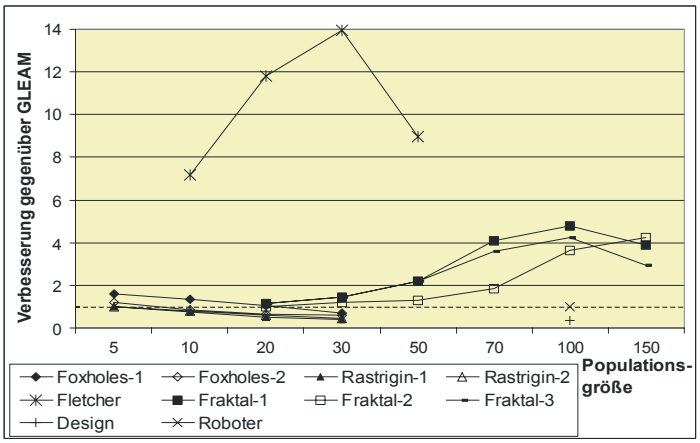


Abb. 5.119: Rosenbrock-Voroptimierung: Vergleich der besten Jobs je Testaufgabe mit den besten GLEAM-Jobs.

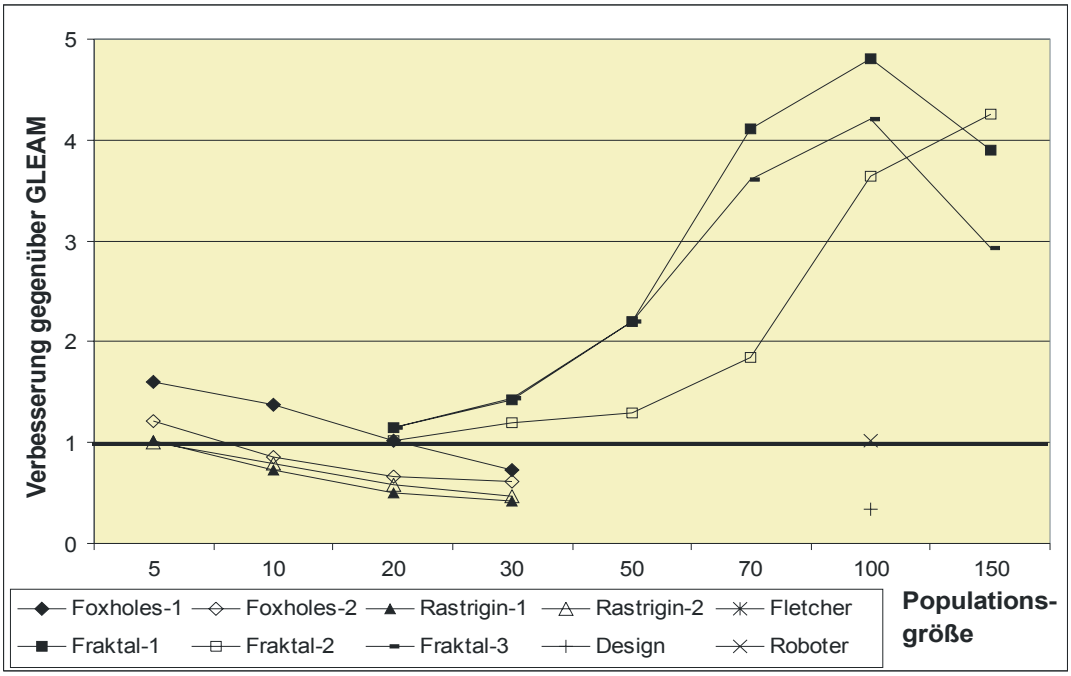


Abb. 5.120: Rosenbrock-Voroptimierung: Vergleich der besten Jobs je Testaufgabe mit dem jeweils besten GLEAM-Job (Ausschnitt aus Abb. 5.119).

Tabelle 5.8 vergleicht alle Jobs von Abb. 5.119 und deren Parametrierung. Da in der Tabelle keine einheitlichen Trends erkennbar sind, kann festgestellt werden, daß eine günstige Parametrierung der Voroptimierung mit dem Rosenbrock-Verfahren anwendungsabhängig ist.

Testaufgaben	Verbesserung	Populationsgröße Voropt.	GLEAM	Anteil voroptimierter Individuen	Präzision
Sphere	kein Erfolg				
Foxholes-1	1.60	5	5	100%	n
Foxholes-2	1.22	5	5	40%	m
Rastrigin-1	1.01	5	5	100%	n
Rastrigin-2	1.01	5	5	20%	m
Fletcher	13.93	30	600	100%	m
Fraktal-1	4.81	100	20	20%	n
Fraktal-2	4.25	150	20	10%	n
Fraktal-3	4.21	100	20	20%	m
Design	0.35	100	210	5%	h
Ressourcen	kein Erfolg				
Roboter	1.02	100	150	10%	s

Tab. 5.8: Rosenbrock-Voroptimierung: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job.

Auch die Frage nach der günstigsten Populationsgröße kann nicht einheitlich beantwortet werden. Bei Fletcher's Function konnte sie drastisch verkleinert werden, während sie bei der fraktalen Funktion erheblich erhöht werden mußte.

Wird der Complex-Algorithmus statt des Rosenbrock-Verfahrens zur Verbesserung der Startpopulation benutzt, ergibt sich bei Unterschieden im Detail kein grundsätzlich anderes Bild, siehe Abb. 5.121. Auch hier sind Schwefel's Sphere und die Ressourcenplanung wegen Erfolglosigkeit nicht in den Diagrammen enthalten. Die herausragende Anwendung ist wieder Fletcher's Function mit einem Verbesserungsfaktor von 24.4. Aber auch die Designoptimierung konnte von der Voroptimierung mit dem Complex-Algorithmus profitieren. Demgegenüber wurde bei der fraktalen Funktion keine Verbesserung mehr erreicht. Abb. 5.122 zeigt die Details ohne Fletcher's Function und Tabelle 5.9 fasst die Ergebnisse aller Testaufgaben zusammen. Es ergibt sich ein mit dem Rosenbrock-Verfahren vergleichbar uneinheitliches Bild.

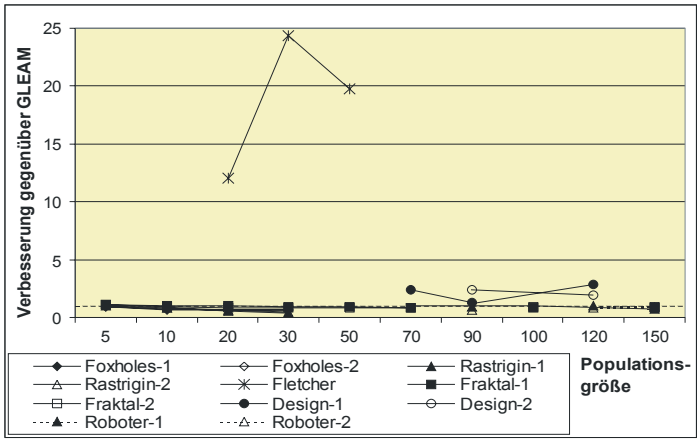


Abb. 5.121: Complex-Voroptimierung: Vergleich der besten Jobs je Testaufgabe mit den besten GLEAM-Jobs.

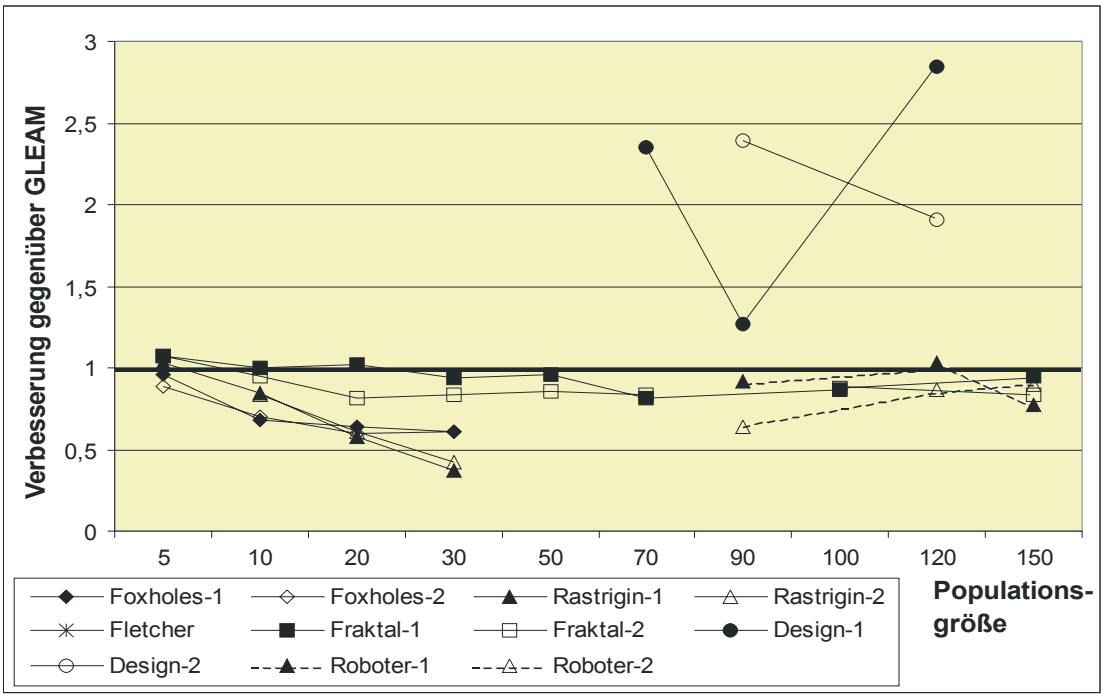


Abb. 5.122: Complex-Voroptimierung: Vergleich der besten Jobs je Testaufgabe mit dem jeweils besten GLEAM-Job (Ausschnitt aus Abb. 5.121).

Da relevante bis erhebliche Verbesserungen sich auf drei Testaufgaben, nämlich Fletcher's Function, die fraktale Funktion und die Designoptimierung, beschränken und es bei den anderen Aufgaben zum Teil zu Verschlechterungen kommt, kann die Voroptimierung nicht allgemein empfohlen werden. Vielmehr handelt es sich hierbei um eine Hybridisierungsform, deren Erfolg stark von der jeweiligen Anwendung abhängt.

Testaufgabe	Verbesserung	Populationsgröße Voropt.	GLEAM	Anteil voroptimierter Individuen
Sphere	kein Erfolg			
Foxholes-1	0.96	5	5	40%
Foxholes-2	0.89	5	5	20%
Rastrigin-1	1.03	5	5	20%
Rastrigin-2	0.83	10	5	10%
Fletcher	24.39	30	600	100%
Fraktal-1	1.07	5	20	20%
Fraktal-2	1.07	5	20	100%
Design-1	2.84	120	210	10%
Design-2	2.39	90	210	20%
Ressourcen	kein Erfolg			
Roboter-1	1.03	120	150	20%
Roboter-2	0.90	150	150	5%

Tab. 5.9: Complex-Voroptimierung: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job.

5.2.2.2 Nachoptimierung

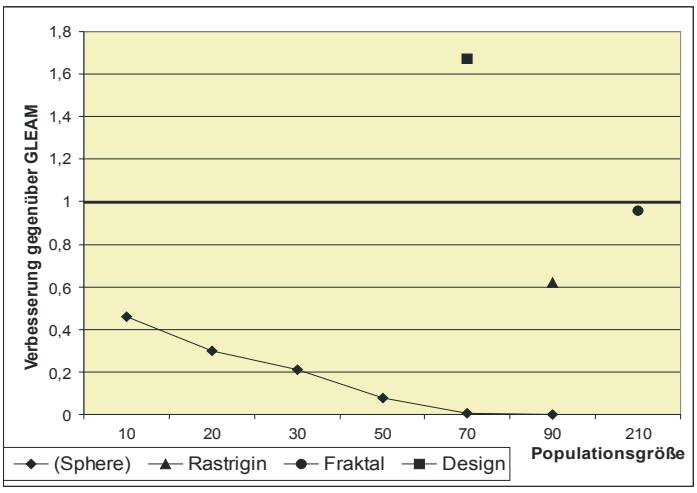


Abb. 5.123: Nachoptimierung: Jobs mit 100% Erfolgsrate. Die Nachoptimierung erfolgte bis auf die Designoptimierung mit dem Rosenbrock-Verfahren. Bei Schwefel's Sphere erfolgt der Vergleich mit den Ergebnissen der reinen Rosenbrock-Optimierung.

Bereits in Abschnitt 5.2.1 wurde bei der Auswertung der Ergebnisse der einzelnen Testaufgaben deutlich, daß die Nachoptimierung insgesamt nicht so erfolgreich abgeschnitten hat wie erhofft. Abb. 5.123 faßt alle Jobs mit einer Erfolgsrate von 100% zusammen. Bis auf Schwefel's Sphere kommt es nur zu Einzelerfolgen, die auch bis auf das Beispiel der Designoptimierung zu keiner Verbesserung hinsichtlich des Aufwands gegenüber GLEAM führen. Damit ist die Nachoptimierung basierend auf der Steuerung durch Nischenbildung gescheitert. Sie konvergiert nicht so zuverlässig wie GLEAM selbst.

Nachfolgend soll auf die interessante Frage eingegangen werden, ob mit der Nachoptimierung schneller als mit GLEAM gute bis sehr gute Ergebnisse unterhalb der ursprünglichen Zielqualität erreicht werden können. Wenn das gelänge, wäre es immerhin ein Beitrag zur ingenieurtechnisch-praktischen Nutzung, der rechtzeitig gute Ergebnisse wichtiger sind, als das verspätet eintreffende Optimum.

Bei der Nachoptimierung mit dem Rosenbrock-Verfahren kann bei den mathematischen Benchmarkfunktionen und der Designoptimierung zum Teil eine erhebliche Notenverbesserung festgestellt werden, siehe Abb. 5.124. Da auch hier Fletcher's Function dominiert, zeigt Abb. 5.126 einen Ausschnitt aus Abb. 5.124, bei dem die sehr guten Ergebnisse von Fletcher's Function weggelassen wurden. Es wird deutlich, daß bei der Rastrigin Funktion und der fraktalen Funktion ebenfalls erhebliche Verbesserungen bei geringerem Aufwand erreicht werden. Die sehr gute Ergebnisqualität bei Schwefel's Sphere geht ebenfalls zu einem erheblichen Teil auf die Nachoptimierung zurück. Bei der Designoptimierung sind hingegen die Verbesserungen eher gering. Abb. 5.125 zeigt die Verhältnisse bei der Ressourcenplanung. Auch hier konnten relevante Verbesserungen bei verringertem Aufwand erreicht werden. Bei der Roboterbahnplanung traten hingegen lediglich minimal verbesserte Notenwerte auf.

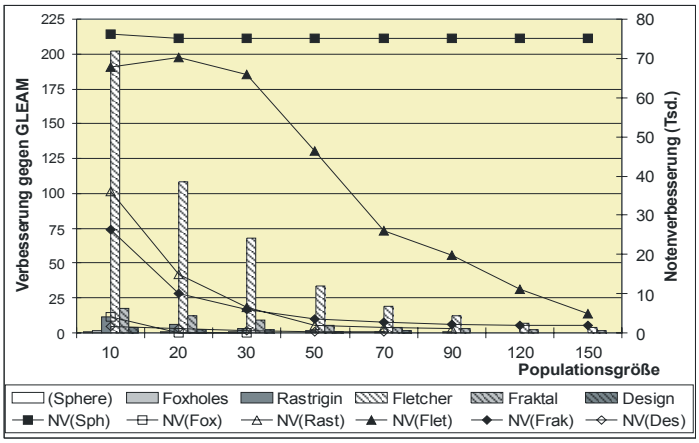


Abb. 5.124: Rosenbrock-Nachoptimierung: Verbesserung beim Aufwand gegenüber dem jeweils besten GLEAM-Job (Säulen) und die Notenverbesserung (NV) gegenüber dem Ergebnis bei Terminierung der Evolution.

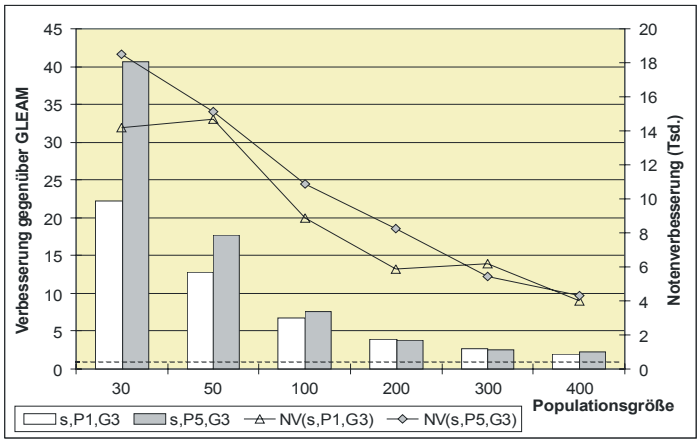


Abb. 5.125: Rosenbrock-Nachoptimierung bei der Ressourcenplanung: Darstellung wie in Abb. 5.124.

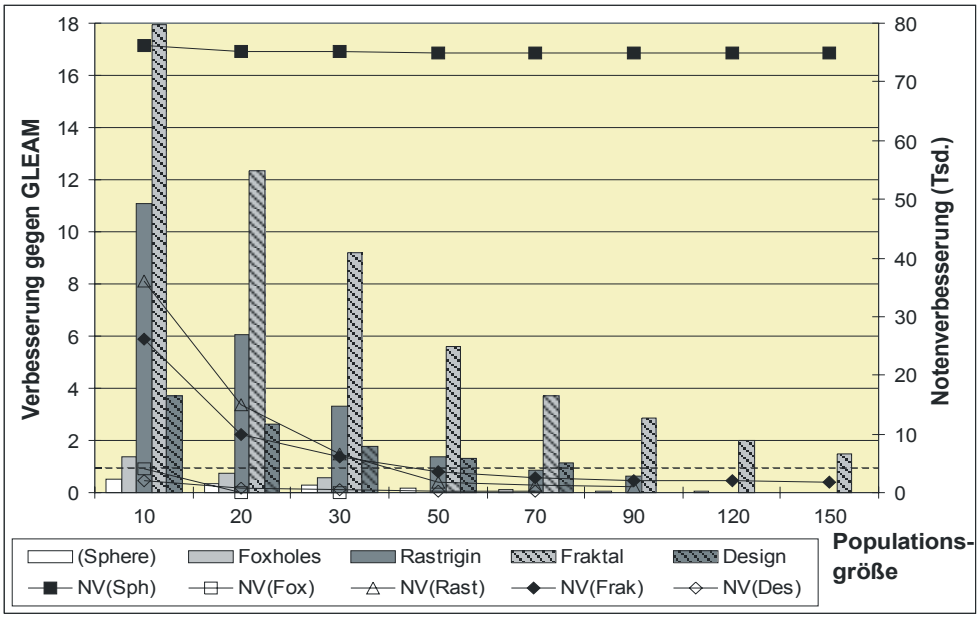


Abb. 5.126: Rosenbrock-Nachoptimierung: Ausschnitt aus Abb. 5.124 (ohne Fletcher).

Tabelle 5.10 listet die jeweils auch hinsichtlich des Gesamtergebnisses günstigsten Jobs mit ihren Parametrierungen auf. Die Nachoptimierung bringt bei fast allen Anwendungen relevante bis erhebliche Notenverbesserungen bei geringerem Aufwand. Ausnahmen davon sind Schwefel's Sphere, Shekel's Foxholes und die in der Tabelle erst gar nicht aufgeführte Roboterbahnplanung, bei der bei guten Endnoten nur eine geringe Verbesserung bei gestiegenem Aufwand festzustellen ist. Schwefel's Sphere ist insofern ein Erfolg, da bei nur etwa doppelt so großem Aufwand wie die reine Rosenbrock-Optimierung bei unüblicher Präzision das Optimierungsziel annähernd erreicht werden konnte, siehe auch Abschn. 5.2.1.1. Bei den Foxholes gilt die bereits in Abschn. 5.2.1.2 gemachte Aussage, daß die Aufgabe von GLEAM bereits so effizient gelöst wird, daß für Verbesserungen kaum Raum bleibt. Als Tendenz kann aus der Tabelle abgeleitet werden, daß kleine Populationsgrößen meist ausreichend sind und die höchstmögliche Präzision zu verwenden ist. Die Bedeutung von *höchstmöglich* ist allerdings ebenso wie die Nischenparametrierung anwendungsabhängig. Darüberhinaus kann keine allgemeine Empfehlung für die Nachoptimierung mit dem Rosenbrock-Verfahren abgeleitet werden. Insgesamt kann aber eine positive Wirkung der Nachoptimierung mit dem Rosenbrock-Verfahren konstatiert werden, wenn auch unterhalb der Marge der Zielerreichung.

Aufgabe	Endnote	Noten-verbesserung	Aufwands-verbesserung	Populations-größe	Präzision	Parametrierung P	G
Sphere-1	99999.9	76115	(0.51)	10	u	3	1
Sphere-2	100000	75283	(0.45)	10	u	2	3
Foxholes	97199	19747	1.01	20	h	2	1
Rastrigin-1	96495	13424	5.8	10	h	3	3
Rastrigin-2	96324	14861	6.0	20	m	2	1
Fletcher-1	99761	72262	126.8	20	m	1	1
Fletcher-2	99788	65133	94.5	20	m	2	3
Fraktal-1	98512	9948	12.3	20	u	3	3
Fraktal-2	97978	25988	24.2	20	m	2	1
Design-1	80539	1923	3.7	10	h	3	1
Design-2	80593	657	2.0	30	h	2	1
Ressourcen-1	88993	8892	6.7	100	s	1	3
Ressourcen-2	99533	4008	2.0	400	s	1	3
Roboter	keine nennenswerten Verbesserungen						

Tab. 5.10: Rosenbrock-Nachoptimierung: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job. Bei Schwefel's Sphere bezieht sich die Aufwandsverbesserung auf die reine Rosenbrock-Optimierung. Die Zielnote bei der Designoptimierung ist 80500, sonst immer 100000.

Auch bei der Nachoptimierung mit dem Complex-Algorithmus (NC1S) ausgehend von einem Startpunkt je GLEAM-Ergebnis dominieren die erreichten Verbesserungen bei Fletcher's Function, wie Abb. 5.127 zeigt. In dem Bild sind einige Testfunktionen aus folgenden Gründen nicht enthalten: Bei Schwefel's Sphere werden nur geringe Notenverbesserungen (zwischen 80 und 2500) bei einem schlechten Endergebnis von rund 25000 erreicht. In Ausnahmen kommt es zwar zu Verbesserungen von rund 15000, allerdings beträgt das Endergebnis dann auch nur ca. 40000. Die Verbesserungen finden also auf einem so schlechten Qualitätsniveau statt, daß sie nicht in Betracht gezogen werden können. Bei der Rastrigin Funktion und der Ressourcenplanung kommt es, wenn überhaupt, nur zu marginalen Verbesserungen. Bei der Roboterbahnplanung werden nennenswerte Verbesserungen (maximal 4200) nur bei einer mäßigen Qualität von rund 42000 erreicht. Bei besseren Endresultaten halbieren sich die Verbesserungen bei einem Aufwand, der mit GLEAM vergleichbar oder sogar deutlich höher ist.

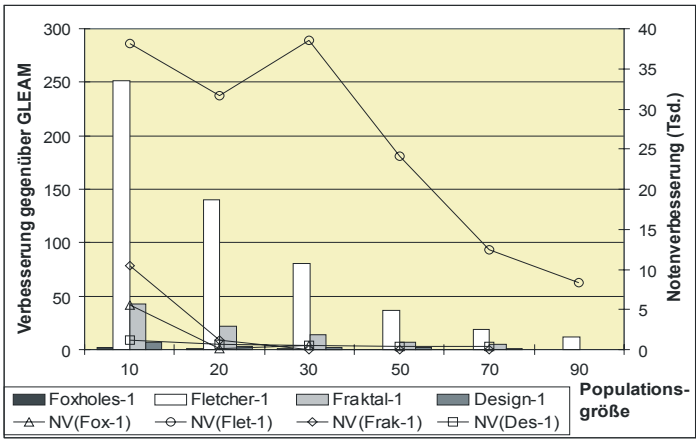


Abb. 5.127: Complex-Nachoptimierung (NC1S): Verbesserung beim Aufwand gegenüber den besten GLEAM-Jobs (Säulen) und die Notenverbesserung (NV) gegenüber dem Ergebnis bei Terminierung der Evolution.

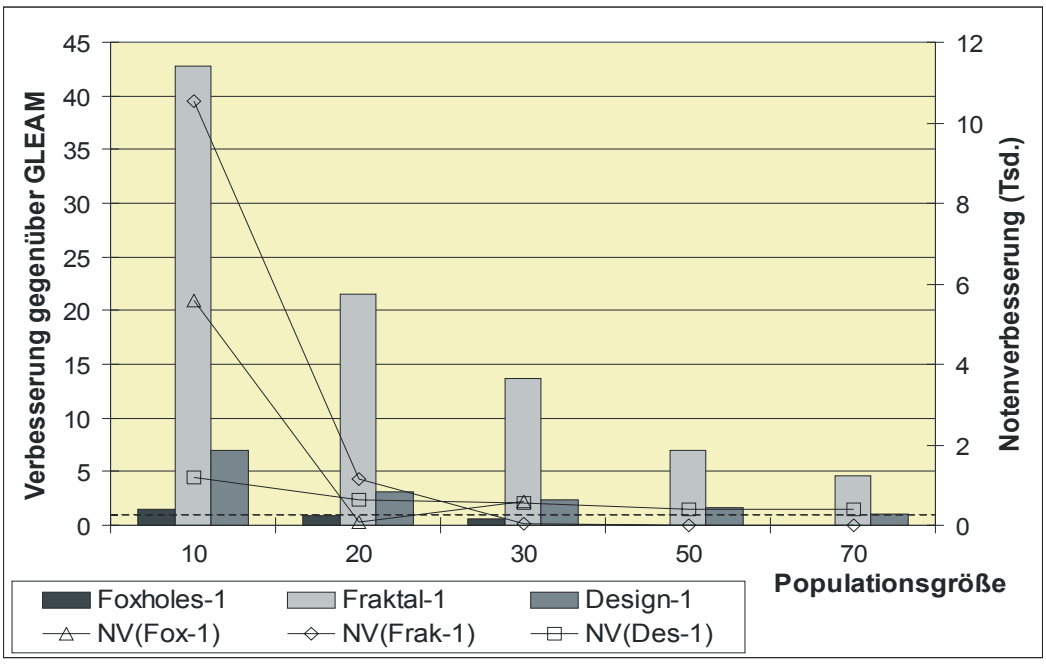


Abb. 5.128: Complex-Nachoptimierung (NC1S): Ausschnitt aus Abb. 5.127 (ohne Fletcher).

Abb. 5.127 und die Ausschnittsvergrößerung von Abb. 5.128 zeigen zusammen mit der Tabelle 5.11, daß im Vergleich zur Nachoptimierung mit dem Rosenbrock-Verfahren weder vergleichbare Endnoten noch konkurrenzfähige Verbesserungen hinsichtlich der Noten und des Aufwands erreicht werden konnten. Außerdem erweisen sich geeignete Populationsgrößen

und Nischenparameter als anwendungsabhängig. Aber immerhin kommt die Parametrierung *P1, G3* bei drei von vier Anwendungsfällen in der Tabelle vor.

Aufgabe	Endnote	Noten-verbesserung	Aufwands-verbesserung	Populations-größe	Parametrierung P	G
Sphere	keine nennenswerten Verbesserungen					
Foxholes-1	92848	5573	1.54	10	3	3
Foxholes-2	93721	3122	1.51	10	2	3
Foxholes-3	82076	12727	2.41	10	3	1
Rastrigin	keine nennenswerten Verbesserungen					
Fletcher-1	84647	24162	36.8	50	2	1
Fletcher-2	87325	11824	16.4	50	1	3
Fletcher-3	83439	26979	37.7	50	a	1
Fraktal-1	90137	1125	22,1	20	1	3
Fraktal-2	83378	8884	30,4	20	3	1
Fraktal-3	89649	1146	21,6	20	2	3
Design-1	80451	642	3.06	20	1	1
Design-2	80381	540	3,10	20	1	3
Design-3	80593	1011	3,35	20	a	1
Ressourcen	keine nennenswerten Verbesserungen					
Roboter	keine nennenswerten Verbesserungen					

Tab. 5.11: Complex-Nachoptimierung ausgehend von einem Startpunkt (NC1S): Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job. Die Zielnote bei der Designoptimierung ist 80500, sonst immer 100000.

Die Ergebnisse der Nachoptimierung ausgehend von einem Startcomplex (NC1C) bringen gegenüber NC1S mit Ausnahme der Designoptimierung nur eine geringe Verbesserung der Situation, vor allem hinsichtlich der erreichten Endnoten und der Verbesserung des Aufwands verglichen mit dem GLEAM-Aufwand, siehe Abb. 5.129 und Tabelle 5.12. Abb. 5.130 enthält die auch hier wegen Fletcher's Function notwendige Ausschnittsvergrößerung. Die Populationsgrößen und die Nischenparameter sind ebenso anwendungsabhängig wie beim separaten Complex-Start je GLEAM-Ergebnis (NC1S).

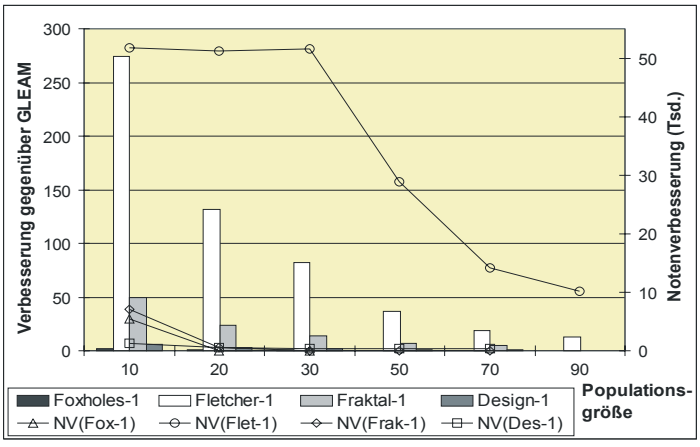


Abb. 5.129: Complex-Nachoptimierung (NC1C): Verbesserung beim Aufwand gegenüber dem jeweils besten GLEAM-Job (Säulen) und die Notenverbesserung (NV) gegenüber dem Ergebnis bei Terminierung der Evolution.

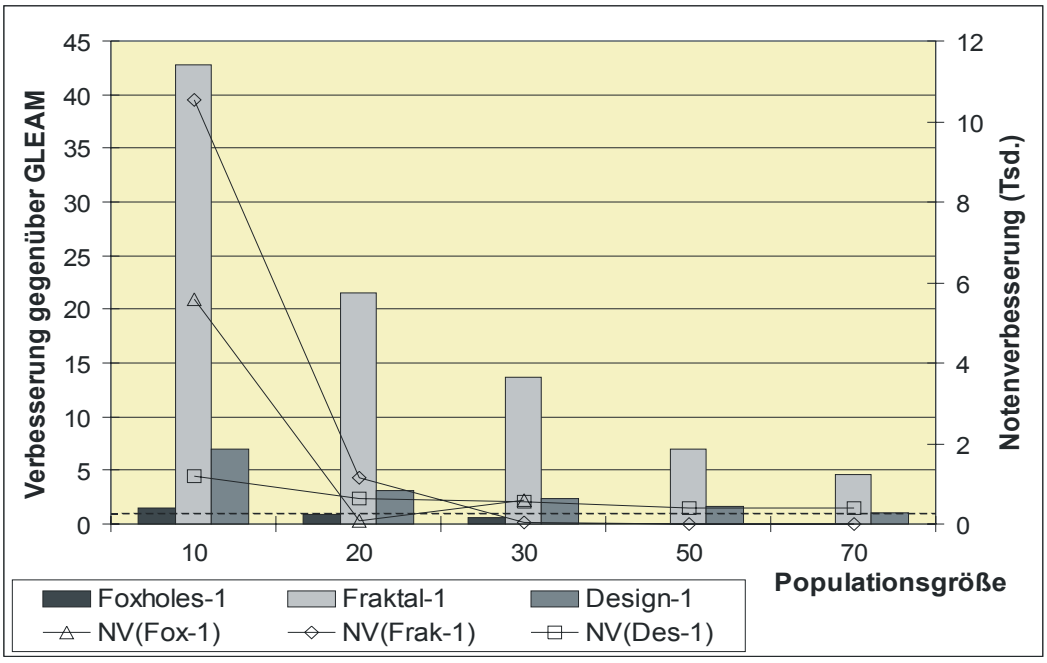


Abb. 5.130: Complex-Nachoptimierung (NC1C): Ausschnitt aus Abb. 5.127 (ohne Fletcher).

Aufgabe	Endnote	Noten-verbesserung	Aufwands-verbesserung	Populations-größe	Parametrierung P	G
Sphere	keine nennenswerten Verbesserungen					
Foxholes-1	92993	5410	1.62	10	3	3
Foxholes-2	93388	2625	1.60	10	2	3
Foxholes-3	83686	11991	2.55	10	2	1
Rastrigin	keine nennenswerten Verbesserungen					
Fletcher-1	92265	28843	36.4	50	2	1
Fletcher-2	95134	18095	18.6	50	3	3
Fletcher-3	89522	29358	17,7	50	a	1
Fraktal-1	89352	553	23.4	20	2	3
Fraktal-2	81514	6270	35,4	20	3	1
Fraktal-3	89211	404	23,8	20	1	3
Design-1	80428	904	3.29	20	1	3
Design-2	80455	606	3,18	20	1	1
Design-3	80560	496	2,75	30	a	1
Ressourcen	keine nennenswerten Verbesserungen					
Roboter	keine nennenswerten Verbesserungen					

Tab. 5.12: Complex-Nachoptimierung ausgehend von einem Startcomplex (NC1C): Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job. Die Zielnote bei der Designoptimierung ist 80500, sonst immer 100000.

Zusammenfassend kann festgestellt werden, daß

- die Nachoptimierung basierend auf der Steuerung durch die Nischenbildung gescheitert ist, da ihr Konvergenzverhalten schlechter als das von GLEAM ausfällt, und
- zur Verbesserung der GLEAM-Resultate nach Evolutionsabbruch die Verwendung aller GLEAM-Ergebnisse zur Bildung eines Startcomplexes (NC1C) zwar bessere Ergebnisse liefert als die separate Nachoptimierung (NC1S), aber nicht mit den Resultaten der Rosenbrock-Nachoptimierung konkurrieren kann. Der Nachoptimierung mit dem Rosenbrock-Verfahren ist also der Vorzug zu geben. Insgesamt kann jedoch keine günstige allgemeingültige Parametrierung hinsichtlich Populationsgröße, Präzision und Nischenabbruchkriterien angegeben werden. Als Empfehlung kann lediglich gesagt werden, daß kleine Populationsgrößen meist ausreichend sind und die anwendungsbedingt höchstmögliche Präzision zu verwenden ist.

5.2.2.3 Direkte Integration

Die direkte Integration mit dem Rosenbrock-Verfahren funktioniert bei allen Anwendungsaufgaben mit Ausnahme der Roboterbahnplanung. Abb. 5.131 zeigt die besten Jobs und Abb. 5.132 gibt eine Ausschnittsvergrößerung wieder, bei der die beiden erfolgreichsten Anwendungen, nämlich Fletcher's Function und die Ressourcenplanung, weggelassen wurden. In der dadurch entstandenen Vergrößerung wird deutlich, daß auch bei der Designoptimierung und der Rastrigin Funktion eine kleine Verbesserung erreicht werden konnte. Schwefel's Sphere ist ein besonderer Fall, da GLEAM keinen Erfolg hat und damit als Vergleichsmaßstab ausfällt. Außerdem ist die reine Rosenbrock-Optimierung nur bei extrem hoher Präzision zu 100% bei 6428 Evaluationen erfolgreich. Wird das Ergebnis der nächst niedrigeren Präzision herangezogen, so werden durchschnittlich 4821 Evaluationen bei einer Erfolgsquote von 42% erreicht. Werden nun die gleichen Maßstäbe wie bei den anderen reinen LSV-Optimierungen (siehe Abschnitte 5.2.1.4 und 5.2.1.6) angelegt, so müssen 10 Läufe durchgeführt werden, was einem Aufwand von 48210 Evaluationen entspricht. So gesehen sind 29600 Evaluationen des besten Jobs kein schlechtes Ergebnis. Es kann also festgestellt werden, daß selbst bei einer so extremen unimodalen Aufgabe wie Schwefel's Sphere die direkte Rosenbrock-Integration immerhin so gut funktioniert, daß bei unbekannter Natur der Fitnesslandschaft kein Fehler gemacht wird, wenn die direkte Integration benutzt wird und die Aufgabe sich hinterher als unimodal erweist. Bei Shekel's Foxholes setzt sich der Trend, daß die Aufgabe zu einfach ist, um Raum für Verbesserungen zu lassen, im wesentlichen fort.

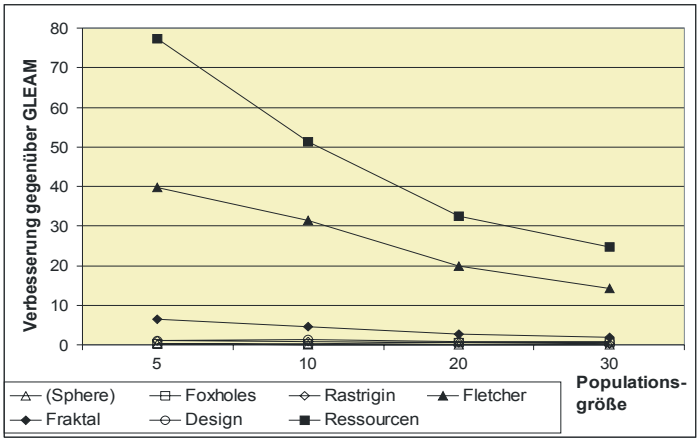


Abb. 5.131: Direkte Rosenbrock-Integration: Vergleich der besten Jobs je Testaufgabe mit dem jeweils besten GLEAM-Job. Bei Schwefel's Sphere erfolgt der Vergleich mit den Ergebnissen der reinen Rosenbrock-Optimierung.

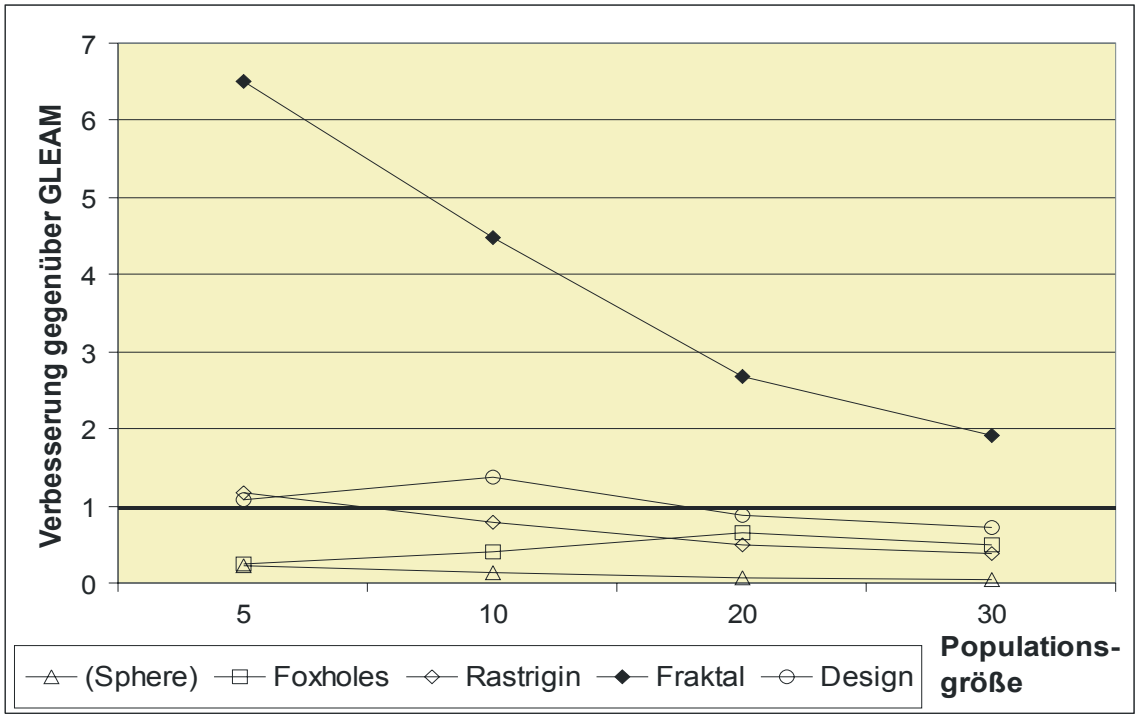


Abb. 5.132: Direkte Rosenbrock-Integration: Ausschnitt aus Abb. 5.131 (ohne Fletcher und Ressourcen).

Aufgabe	Ver-besserung	Populations-größe	Präzision	maximale Präzision	best / all	Lamarck-rate	Voropti-mierung
Sphere-1	(0.22)	5	h	v	best	100	-
Sphere-2	(0.20)	5	x	v	best	100	-
Foxholes-1	0.66	20	n	h	best	100	ja
Foxholes-2	0.54	20	n	h	best	100	-
Foxholes-3	0.52	5	n	h	all	100	-
Rastrigin-1	1.18	5	m	m	best	100	-
Rastrigin-2	1.01	5	m	m	best	100	ja
Fletcher-1	39.87	5	m	m	best	100	ja
Fletcher-2	35.71	10	m	m	best	100	-
Fraktal-1	6.50	5	n	m	best	100	ja
Fraktal-2	6.37	5	n	m	best	100	-
Fraktal-3	4.94	5	m	m	best	100	-
Design-1	1.37	10	m	h	best	5	-
Design-2	1.07	10	m	h	best	100	-
Design-2	1.22	10	m	h	best	5	ja
Ressourcen-1	77.41	5	s	s	best	100	-
Ressourcen-2	70.03	5	s	s	best	100	ja

Tab. 5.13: Direkte Rosenbrock-Integration: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job. Bei Schwefel's Sphere erfolgt der Vergleich mit den Ergebnissen der reinen Rosenbrock-Optimierung.

Tabelle 5.13 zeigt, daß es an gemeinsamer Parametrierung erstens kleine Populationsgrößen von fünf bis zehn, zweitens die lokale Verbesserung nur des besten Nachkommens (best) und drittens eine Lamarckrate von 100% gibt. Bei der Präzision ist die Lage insofern etwas uneinheitlich, als abgesehen von den Foxholes neben der anwendungsbedingt höchstmöglichen auch die zweithöchste Präzision vorkommt. Voroptimierung hilft häufig und wenn nicht, fallen die Ergebnisse nicht deutlich schlechter aus als ohne. Bemerkenswert sind noch die kleinen Populationsgrößen, die, wie im Falle von fünf, zu vollständig panmiktischen oder wie im Falle von zehn zu nahezu panmiktischen Populationen führen.

Die direkte Integration mit dem Complex-Algorithmus funktioniert nur bei Shekel's Foxholes, Fletcher's Function und der Designoptimierung, dafür aber bei den beiden letzten Aufgaben besser als mit dem Rosenbrock-Verfahren. Abb. 5.133 zeigt die Ergebnisse und vergleicht sie mit den jeweiligen Rosenbrock-Varianten. Wie zuvor beim Rosenbrock-Verfahren können auch hier die Ergebnisse von Shekel's Foxholes aus der Betrachtung weitgehend herausgelassen werden. Abb. 5.134 zeigt einen Ausschnitt ohne die Ergebnisse von Fletcher's Function, um die klare Überlegenheit der direkten Complex-Integration gegenüber der mit dem Rosenbrock-Verfahren bei der Designoptimierung deutlich zu machen.

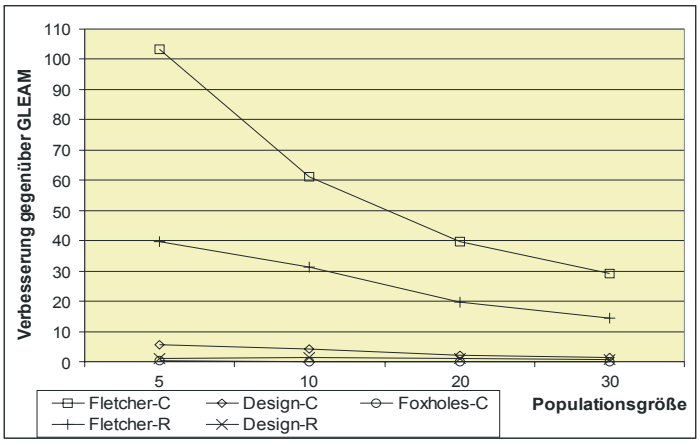


Abb. 5.133: Direkte Complex-Integration: Vergleich der besten Jobs je Testaufgabe mit dem jeweils besten GLEAM-Job und mit den jeweiligen besten Rosenbrock-Varianten.

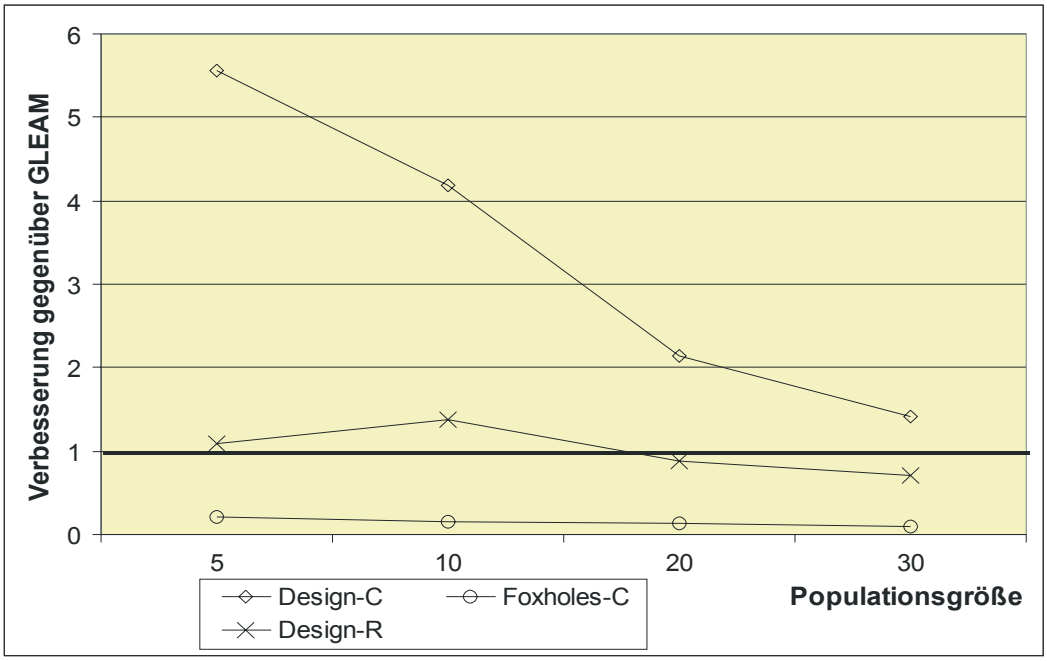


Abb. 5.134: Direkte Complex-Integration: Shekel's Foxholes und Vergleich der beiden integrierten Verfahren bei der Designoptimierung (Ausschnitt aus Abb. 5.133, ohne Fletcher).

Tabelle 5.14 legt den Schluß nahe, daß die Voroptimierung der Startpopulation mit dem Complex-Algorithmus die Aufwandsverbesserung eher vermindert. Ansonsten läßt sich die Parametrierung einer Populationsgröße von fünf bei lokaler Verbesserung des besten Nachkommens (best) und reiner Lamarckscher Evolution ableiten. Da aber nur zwei funktionierende Beispiele zur Verfügung stehen, sind die Aussagen mit einer entsprechenden Unsicherheit behaftet. Es kann außerdem festgestellt werden, daß die direkte Rosenbrock-Integration wesentlich sicherer arbeitet als die Integration des Complex-Algorithmus, der aber, wenn er funktioniert, die besseren Ergebnisse liefert. Daher lohnt sich eine Überprüfung der Leistungsfähigkeit der direkten Complex-Integration vor allem in solchen Fällen, in denen von einer Problemklasse viele Aufgabenvarianten zu optimieren sind.

Aufgabe	Verbesserung	Populationsgröße	best / all	Lamarckrate	Voroptimierung
Foxholes-1	0.22	5	all	100	-
Foxholes-2	0.20	20	best	100	-
Foxholes-3	0.19	10	best	100	ja
Fletcher-1	103.33	5	best	100	-
Fletcher-2	50.85	5	best	100	ja
Design-1	5.55	5	best	100	-
Design-2	5.35	5	best	100	ja

Tab. 5.14: Direkte Complex-Integration: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job.

5.2.2.4 Verzögerte direkte Integration

Da die verzögerte direkte Integration eine Variante der unverzögerten ist, entsprechen die grundsätzlichen Ergebnissituationen bei den einzelnen Testaufgaben denen der unverzögerten. Die Unterschiede liegen im Detail der erreichten Verbesserungen gegenüber dem besten GLEAM-Job und zum Teil auch in den Parametrierungen. Pro integriertem lokalen Suchverfahren erfolgt nach der Ergebnisdarstellung ein Vergleich zwischen der verzögerten und der unverzögerten direkten Integration.

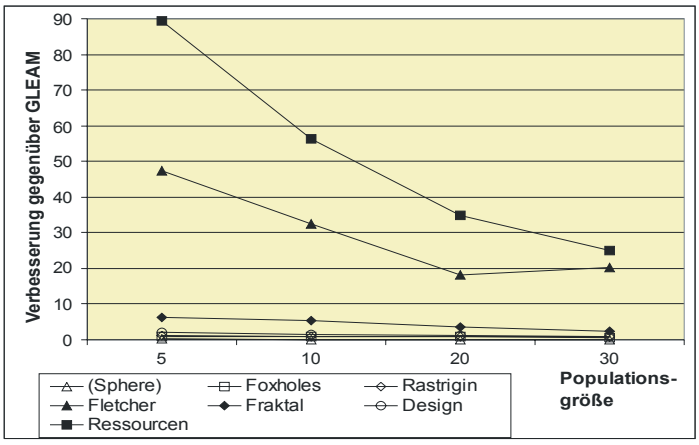


Abb. 5.135: Verzögerte direkte Rosenbrock-Integration: Vergleich der besten Jobs je Testaufgabe mit dem jeweils besten GLEAM-Job. Bei Schwefel's Sphere erfolgt der Vergleich mit den Ergebnissen der reinen Rosenbrock-Optimierung.

Die verzögerte direkte Integration mit dem Rosenbrock-Verfahren funktioniert wie die unverzögerte für alle Testaufgaben mit Ausnahme der Roboterbahnplanung und es gelten die gleichen Aussagen zu Schwefel's Sphere wie in Abschnitt 5.2.2.3 mit dem positiven Unterschied, daß im günstigsten Fall anstelle von 29600 nur 28400 Evaluationen benötigt werden. Abb. 5.135 zeigt die Ergebnisse und Abb. 5.136 gibt einen Ausschnitt

davon wieder, bei dem wieder die beiden erfolgreichsten Anwendungen, nämlich Fletcher's Function und die Ressourcenplanung weggelassen wurden. Bei der Designoptimierung zeigt sich eine hinsichtlich der Populationsgröße stabilere Verbesserung als bei der unverzögerten direkten Integration und auch die Rastrigin Funktion kommt zuverlässiger in die Nähe des GLEAM-Niveaus. Insgesamt ist ein Trend der Verbesserung gegenüber der unverzögerten direkten Integration feststellbar.

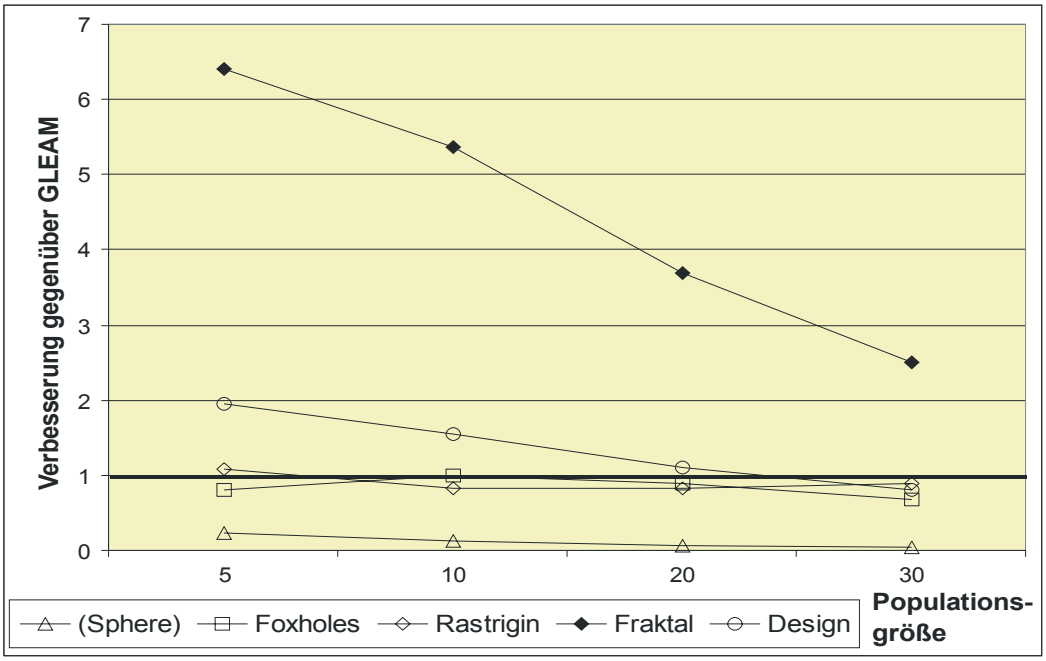


Abb. 5.136: Verzögerte direkte Rosenbrock-Integration: Ausschnitt aus Abb. 5.135 (ohne Fletcher und Ressourcen).

Tabelle 5.15 vergleicht die Parametrierungen der besten Jobs je Testaufgabe. Abgesehen von Schwefel's Foxholes, auf deren Sonderrolle ja schon in Abschnitt 5.2.1.2 hingewiesen wurde, kann als gemeinsame günstige Parametrierung eine Populationsgröße von fünf, Verbesserung nur des besten Nachkommens, reine Lamarckscher Evolution und keine Voroptimierung herausgelesen werden. Abweichungen davon kommen, wenn überhaupt, nur bei den schlechter plazierten Jobs vor. Bei der Präzision kommt die höchstmögliche mehr zum Einsatz als bei der unverzögerten Variante. Als Nischenparametrierung taucht P3 am häufigsten auf, aber auch P2 und P1 kommen an erster Stelle vor. P1 und die noch frühere Zuschaltung P7 sind interessanterweise bei den beiden noch betrachteten Aufgaben mit realem Hintergrund am erfolgreichsten. Allerdings sind die Unterschiede zu den etwas später zuschaltenden Parametrierungen nicht gravierend.

Ein Vergleich der jeweils besten Jobs der verzögerten mit denen der unverzögerten direkten Integration ergibt das in Abb. 5.137 dargestellte Bild. Bei Fletcher's Function, der Designoptimierung und der Ressourcenplanung konnte eine relevante Verbesserung bis zu einem Faktor von 1.4 erreicht werden, während die verzögerte direkte Integration bei der fraktalen und der Rastrigin-Funktion keinen Vorteil bringt. Bemerkenswert sind auch die Aufwandsverbesserungen bei Schwefel's Sphere und Shekel's Foxholes. Tabelle 5.16 zeigt die Parametrierungen der Jobs zusammen mit dem jeweiligen Verbesserungsfaktor. Dabei fällt auf, daß sich die Parametrierungen zum Teil unterscheiden und bis auf Shekel's Foxholes die Voroptimierung keine Rolle mehr spielt.

Aufgabe	Ver-besserung	Populations-größe	Präzision	maximale Präzision	best / all	Lamarck-rate	Nischen-param.	Voropti-mierung
Sphere-1	(0.23)	5	h	v	best	100	P3	-
Sphere-2	(0.22)	5	h	v	best	100	P1	-
Sphere-3	(0.16)	5	h	v	best	100	P3	ja
Foxholes-1	0.81	10	n	h	all	100	P3	ja
Foxholes-2	0.58	5	n	h	all	100	P3	-
Rastrigin-1	1.09	5	m	m	best	100	P3	-
Rastrigin-2	1.07	5	m	m	best	100	P2	ja
Rastrigin-3	1.06	5	m	m	best	100	P2	-
Fletcher-1	57.16	5	m	m	best	100	P2	-
Fletcher-2	47.10	5	m	m	best	100	P3	ja
Fletcher-3	44.01	5	m	m	best	100	P3	-
Fraktal-1	6.40	5	n	m	best	100	P2	-
Fraktal-2	6.29	5	n	m	best	100	P3	-
Fraktal-3	5.60	5	n	m	best	100	P3	ja
Design-1	1.95	5	h	h	best	100	P1	-
Design-2	1.73	5	h	h	best	100	P2	-
Design-3	1,64	10	m	h	best	5	P1	-
Ressourcen-1	89,27	5	s	s	best	100	P7	-
Ressourcen-2	80,27	5	s	s	best	100	P1	-

Tab. 5.15: Verzögerte direkte Rosenbrock-Integration: Parametrierungen der Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job. Bei Schwefel's Sphere erfolgt der Vergleich mit den Ergebnissen der reinen Rosenbrock-Optimierung.

Aufgabe	GvR	GR	GvR-Verbesserung
Sphere	p5, h, best, l100, P3	p5, h, best, l100	1.04
Foxholes	p10, n, all, l100, Ri, P3	**p20**, n, **best**, l100, Ri	1.50
Rastrigin	p5, m, best, l100, P3	p5, m, best, l100	0.92
Fletcher	p5, m, best, l100, P2	p5, m, best, l100, **Ri**	1.19
Fraktal	p5, n, best, l100, P2	p5, n, best, l100, **Ri**	0.99
Design	p5, h, best, l100, P1	**p10**, **m**, best, **l5**	1.43
Ressourcen	p5, s, best, l100, P7	p5, s, best, l100	1.15

Tab. 5.16: Parametrierungen der Jobs von Abb. 5.137. Abweichungen bei GR sind fett hervorgehoben.

Das wirft die Frage auf, ob die verzögerte direkte Integration bei sonst gleicher Parametrierung eher ein Vor- oder ein Nachteil ist. Dazu werden die besten verzögerten Jobs je Aufgabe mit den unverzögerten bei sonst gleicher Parametrierung (Präzision, best/neu, Lamarckrate, mit/ohne Voroptimierung) verglichen. Wenn dabei der unverzögerte Job ebenfalls der beste seiner Klasse ist, bleibt es bei dem einen Vergleich. Sonst wird auch der zum besten unverzögerten Job gehörige verzögerte verglichen (in den Bildern mit *-2* gekennzeichnet). Der besseren Darstellung halber wird der Vergleich auf zwei Bilder aufgeteilt: Abb. 5.138 enthält die Aufgaben mit den kleineren Unterschieden und Abb. 5.139 diejenigen mit den größeren. Die Jobs mit dem geringsten Aufwand haben bis auf drei Ausnahmen eine Populationsgröße von fünf. Die Ausnahmen sind in Abb. 5.139 durch kleine Kreise gekennzeichnet. Tabelle 5.17 gibt die Parametrierungen der Jobs an. Zunächst fällt auf, daß die Jobs mit der größten Verbesserung gegenüber der unverzögerten Integration meist nicht auch die mit dem geringsten Aufwand sind.

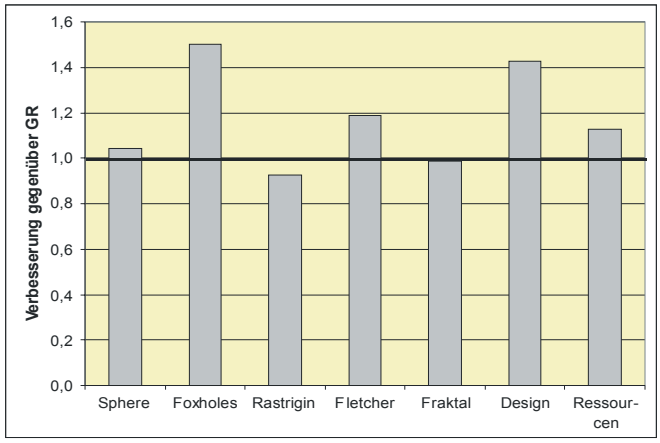


Abb. 5.137: Rosenbrock-Integration: Vergleich der Verbesserung beim Aufwand des jeweils besten Jobs der verzögerten direkten Integration gegenüber der unverzögerten. Die zu den Jobs gehörigen Parametrierungen sind in Tabelle 5.16 angegeben. Sie sind nur bei Schwefel's Sphere, der Rastrigin Funktion und der Ressourcenplanung gleich.

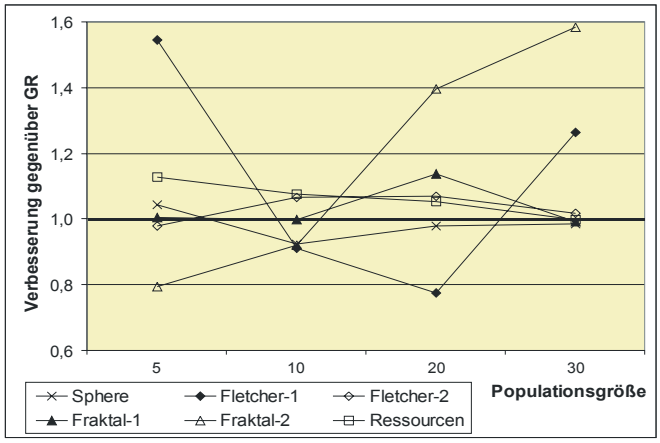


Abb. 5.138: Rosenbrock-Integration: Vergleich der Verbesserung beim Aufwand zwischen verzögerter und unverzögerter direkter Integration bei sonst gleich parametrierten Jobs. Bei Aufgaben ohne Nummernangabe haben die besten Jobs der verzögerten und der unverzögerten Integration die gleichen Parameter. Ansonsten wird die Parametrierung erst durch den besten GvR- (-1) und dann durch den besten GR-Job (-2) bestimmt.

Außerdem ist der Aufwandsunterschied zischen verzögerter und unverzögerter direkter Integration sehr unterschiedlich und zum Teil stark von der Populationsgröße abhängig. Auch die Nischenparametrierung der Jobs ist uneinheitlich, während bei der Populationsgröße der Wert fünf dominiert. Es kann also keinen Trend festgestellt werden und daher ist der Schluß zu ziehen, daß die Frage der vorteilhaften Anwendbarkeit der verzögerten direkten Integration nur aufgabenspezifisch beantwortet werden kann. Da im Einzelfall ein Verbesserungsfaktor von 1.4 erreicht wurde, kann sich je nach Anwendungsfall eine genauere Untersuchung der Anwendbarkeit der verzögerten direkten Integration durchaus lohnen.

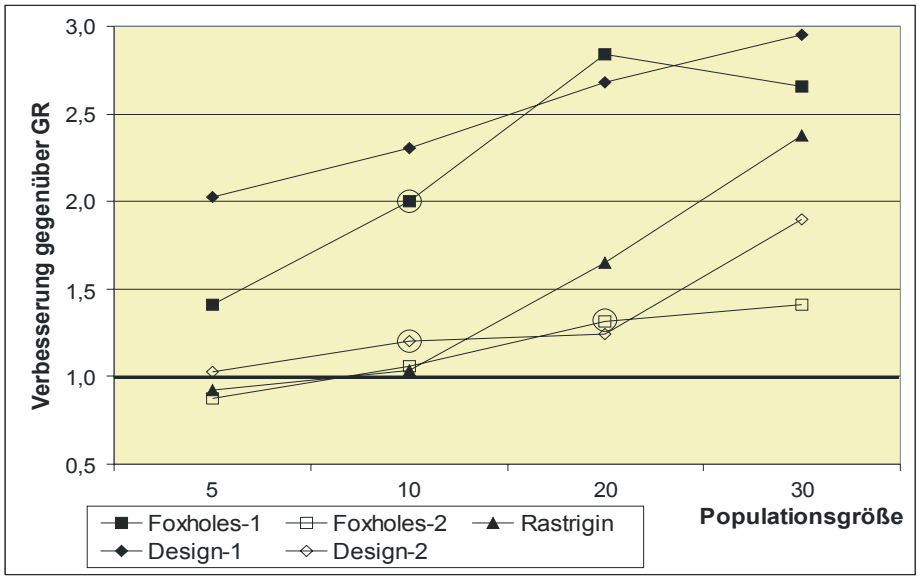


Abb. 5.139: Rosenbrock-Integration: Zweiter Teil von Abb. 5.138. Die Kreise kennzeichnen die Jobs mit dem geringsten Aufwand, wenn die Populationsgröße von fünf abweicht.

Aufgabe	Präzision	all / best	Lamarck-rate	Voropti-mierung	Nischen-parame-trierung	bester Job	
						Populations-größe	Verbesserung gegenüber GR
Sphere	h	best	100	-	3	5	1.04
Foxholes-1	n	all	100	ja	3	10	2.0
Foxholes-2	n	best	100	ja	2	20	1.32
Rastrigin	m	best	100	-	3	5	0.92
Fletcher-1	m	best	100	-	2	5	1.55
Fletcher-2	m	best	100	ja	3	5	0.98
Fraktal-1	n	best	100	-	2	5	1.01
Fraktal-2	n	best	100	ja	2	5	0.79
Design-1	h	best	100	-	1	5	2.03
Design-2	m	best	5	-	1	10	1.2
Ressourcen	s	best	100	-	7	5	1.15

Tab. 5.17: Parametrierungen der in Abb. 5.138 und Abb. 5.139 dargestellten Jobs

Die verzögerte direkte Integration mit dem Complex-Algorithmus funktioniert wie die unverzögerte nur mit den drei Testaufgaben Shekel's Foxholes, Fletcher's Function und der Designoptimierung, wobei auf die geringe Aussagekraft von Shekel's Foxholes ja schon in Abschnitt 5.2.1.2 hingewiesen wurde. In den beiden verbleibenden Fällen bringt die Verwendung des Complex-Algorithmus statt des Rosenbrock-Verfahrens noch einmal eine deutliche Leistungssteigerung, wie in Abb. 5.140 dargestellt. In den beiden Teilbildern werden die besten Jobs je Verfahrenskombination und Aufgabe verglichen. Tabelle 5.18 gibt die dazugehörigen

Parametrierungen an. Dabei fällt auf, daß die Einstellungen für die Optimierung des besten oder aller Nachkommen (best/all) und die Lamarckrate je Aufgabenstellung gleich sind. Außerdem schneidet beim Complex-Algorithmus die Nischenparametrierung P3 regelmäßig am besten ab. Allerdings ist die Aussagekraft dieser Beobachtung bei nur zwei wirklich funktionierenden Beispielen begrenzt.

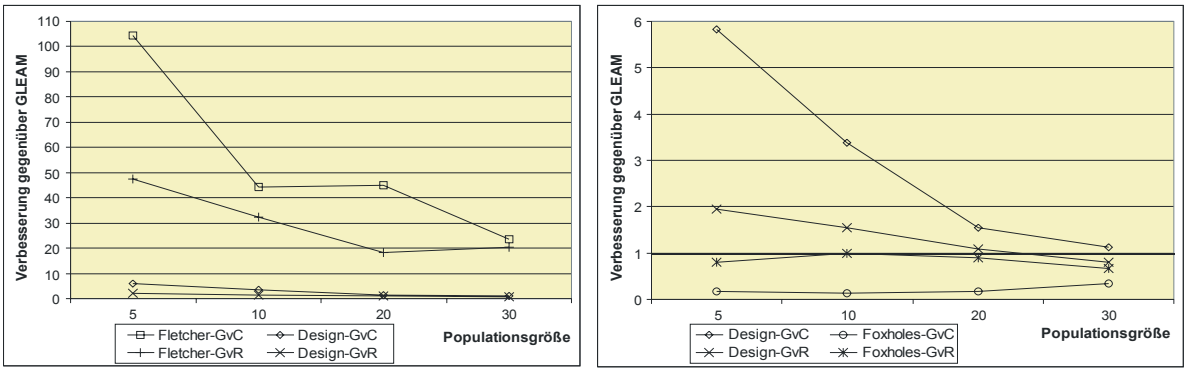


Abb. 5.140: Verzögerte direkte Integration: Vergleich der Integration des Complex-Algorithmus mit der des Rosenbrock-Verfahrens. Aus Gründen der besseren Darstellung erfolgt eine Aufteilung der Ergebnisse auf zwei Schaubilder.

Aufgabe	GvC	GvR	GvC-Verbesserung
Foxholes	all, l100, P3	n, all, l100, P3, **Ri**	-
Fletcher	best, l100, P3	m, best, l100, **P2**	2.20
Design	best, l100, P3	h, best, l100, **P1**	2.97

Tab. 5.18: Parametrierung der Jobs von Abb. 5.140. Abweichungen bei GvR sind fett gedruckt. Die GvC-Verbesserung bezieht sich auf den entsprechenden GvR-Job bei Populationsgröße fünf (beste Ergebnisse).

Abb. 5.141 beantwortet die Frage, ob die Verzögerung bei der direkten Complex-Integration von Vorteil ist. Im Gegensatz zur direkten Rosenbrock-Integration muß das bei den hier untersuchten Testaufgaben verneint werden. Die beobachteten Verbesserungen sind eher marginal und ihnen stehen bei anderen Populationsgrößen deutliche Verschlechterungen gegenüber. Bemerkenswert ist allerdings, daß die Parametrierung hinsichtlich Lamarckrate und der Verbesserung des besten oder aller Nachkommen (best/all) bei den jeweils besten Jobs bei allen drei Testaufgaben identisch ist.

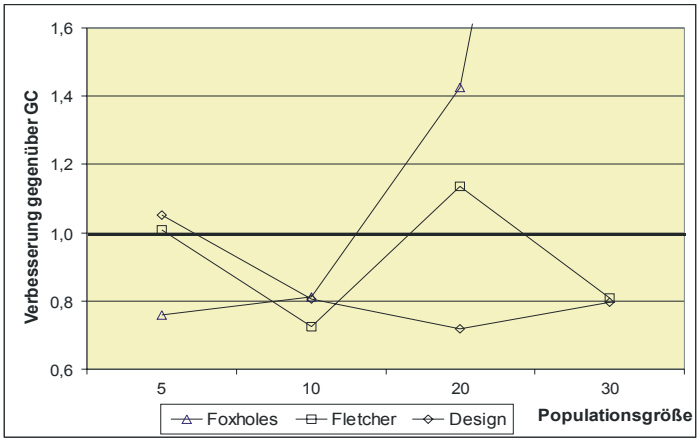


Abb. 5.141: Complex-Integration: Vergleich der Verbesserung beim Aufwand zischen verzögerter und unverzögerter direkter Integration. Aus Gründen der Darstellung ist bei der eher uninteressanten Testaufgabe Foxholes der Wert für die Populationsgröße 30 weggelassen worden.

5.2.3 Gedrehte Benchmarkfunktionen

Obwohl die beiden Benchmarkfunktionen Shekel's Foxholes und die Rastrigin Funktion als schwierig für die Standard-ES gelten, wurden sie von GLEAM so gut gelöst, daß für Verbesserungen durch eine Hybridisierung kaum Raum bleibt. Entsprechend gering fielen, wie zuvor dargelegt, die Verbesserungen aus. Die meisten Mutationen von GLEAM suchen, wie bei den EA üblich, bevorzugt parallel zu den Koordinatenachsen. Das gilt vor allem bei Problemen mit wenigen Parametern, da die Wahrscheinlichkeit, daß mehr als ein Parameter durch eine einzige Mutation verändert wird, mit abnehmender Parameteranzahl sinkt. Im Extremfall von Shekel's Foxholes mit seinen zwei Parametern kommt bei GLEAM eine gleichzeitige Änderung beider Parameter überhaupt nicht vor.

Bei den Benchmarkfunktionen wurde jeder Parameter durch eine einzelne Aktion abgebildet und die Segmentierung wurde so gewählt, daß erst bei mehr als fünf Parametern ein Segment aus mehr als einer Aktion bestehen kann. Tabelle 5.19 gibt einen Überblick und zeigt die Anzahl maximal betroffener Aktionen der Aktionsmutationen in Abhängigkeit von der Parameteranzahl (Kettenlänge). Zusammen mit den Segmentmutationen, die alle Aktionen eines Segments verändern, ergeben sich bis zu sechs gleichzeitig veränderbare Parameter.

Benchmarkfunktion	Dimensionen	Segmentmutationen: Aktionen pro Segment	Aktionen pro Aktionsmutation
Schwefel's Sphere	30	2 - 6	bis zu 4
Shekel's Foxholes	2	1	1
Rastrigin Funktion	20	3 - 6	bis zu 3
Fletcher's Function	5	1	1
Fraktale Funktion	20	2 - 5	bis zu 3

Tab. 5.19: Maximale Anzahl betroffener Aktionen und damit Parameter einer Mutation.

Neben den Mutationen sind noch die Crossover-Operatoren in der Lage, mehrere Aktionen und damit Parameter in einem Schritt zu verändern. Bei *n* Parametern bewirken sie die Änderung von bis zu *n-1* Parametern.

Bei einer Betrachtung der Bilder von Abschnitt 4.1 wird schnell klar, daß bei den beiden eingangs angesprochenen Testfunktionen eine Strukturierung der Funktion parallel zu den Koordinatenachsen gegeben ist. Daher wurde die Frage untersucht, ob eine Drehung des Koordinatensystems im Raum einen Einfluß auf das Suchverhalten der beteiligten Verfahren hat. Abb. 5.142 vergleicht die unveränderte Foxhole-Funktion mit einer um 30^{o} gedrehten. Das Beispiel zeigt anschaulich, daß es bei Schritten parallel zu den Koordinatenachsen bei der gedrehten Variante ungleich schwerer ist, in einem Schritt ein „Fuchsloch" zu verlassen und tief genug in ein anderes zu gelangen, um einen besseren Fitnesswert zu erzielen. Auch bei der Rastrigin Funktion zeigt der in Abb. 5.143 dargestellte Vergleich ihrer zweidimensionalen Variante, daß das Bild der gedrehten Funktion unregelmäßiger wird und vor allem nicht mehr parallel zu den Koordinatenachsen ausgerichtet ist.

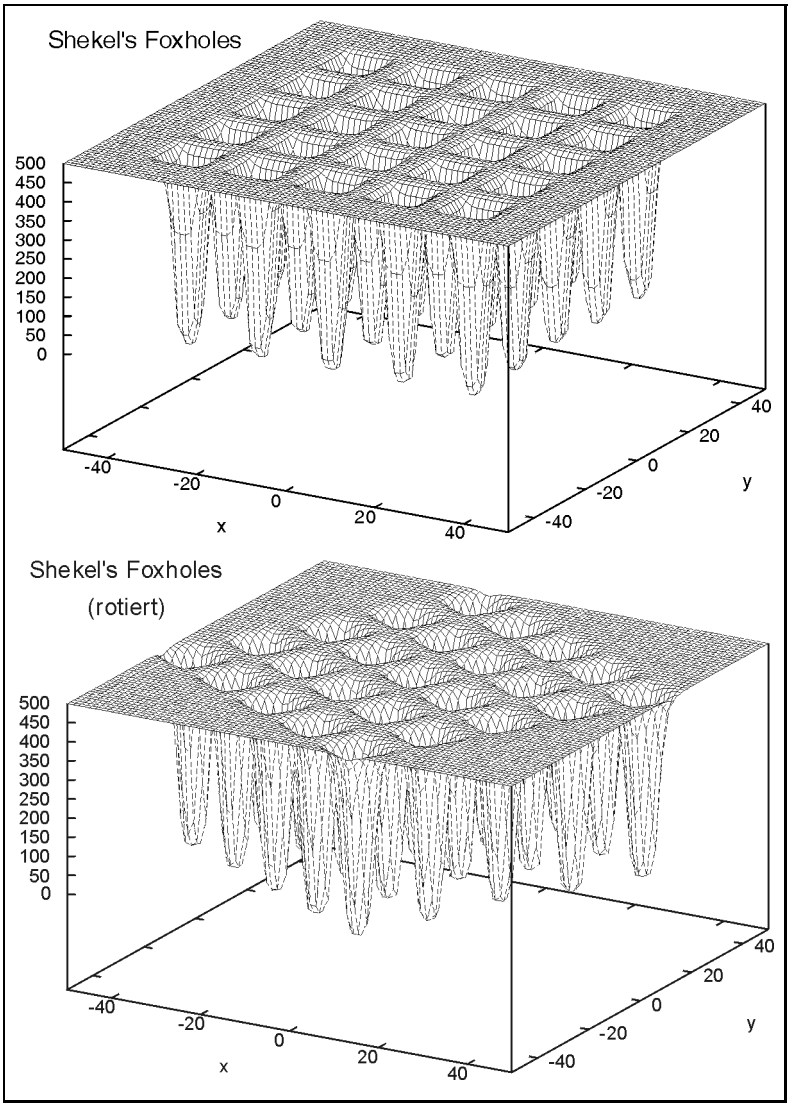


Abb. 5.142: Shekel's Foxholes: Vergleich der Originalfunktion mit der um 30^o gedrehten.

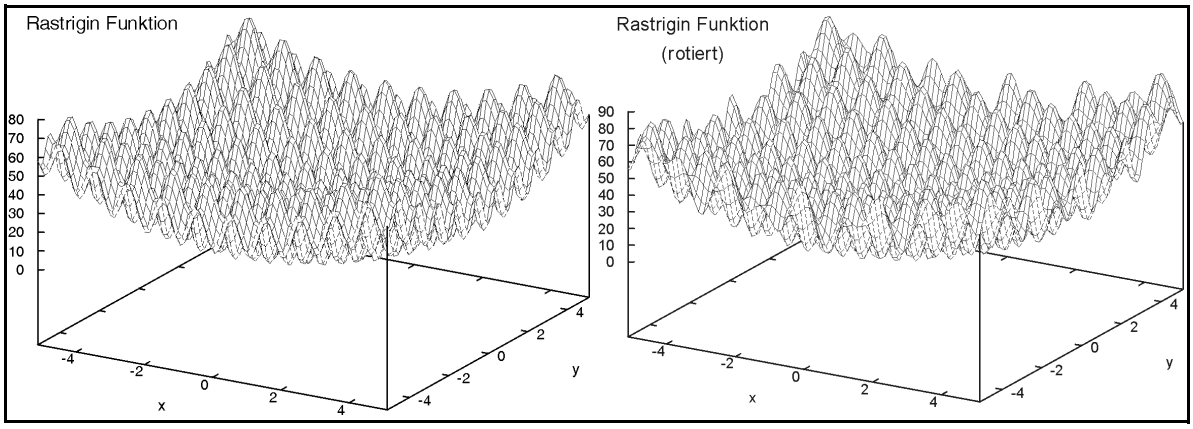


Abb. 5.143: Rastrigin Funktion: V ergleich der Originalfunktion mit der um 30^o gedrehten

Tabelle 5.20 vergleicht die durch die Drehung bewirkten Unterschiede im Lösungsverhalten der einzelnen Verfahren, wobei aus Gründen der Rotationssymmetrie auf eine Untersuchung von Schwefel's Sphere verzichtet werden konnte. Erwartungsgemäß verhält sich GLEAM bei

Shekel's Foxholes und der Rastrigin Funktion eindeutig anders, wenn die Funktion gedreht wird. Bei der Rastrigin Funktion ist in ihrer ursprünglichen 20-dimensionalen Form sogar eine Lösung mit GLEAM innerhalb vorgegebener 50000 Generationen und Populationsgrößen zwischen 120 und 1000 nicht möglich, siehe auch Anhang B.3.6. Eine Reduktion auf fünf Dimensionen bringt GLEAM immerhin wieder in den Bereich der Lösungsfähigkeit, wenn auch erst bei vergleichsweise extrem großen Populationen, siehe Abschnitt 5.2.3.2. Die Unterschiede bei Fletcher's Function sind vernachlässigbar und bei der fraktalen Funktion vergleichsweise gering. Die beiden lokalen Verfahren erweisen sich als relativ stabil hinsichtlich der durchgeführten Drehung. Kurioserweise verbessert sich sogar das Konvergenzverhalten des Rosenbrock-Verfahrens bei der gedrehten Fletcher's Function etwas.

Benchmark-funktion	GLEAM		Rosenbrock-Verfahren		Complex-Algorithmus	
	Erfolgsrate [%]	Aufwands-zunahme	Erfolgsrate [%]	Aufwands-zunahme	Erfolgsrate [%]	Aufwands-zunahme
Foxholes	100 100	71.8	3 0	~	1 0	~
Rastrigin	100 0		0 0	~	0 0	3.0
Rastrigin, 5 Parameter	100 100	1171.4	0 0	~	0 0	0.9
Fletcher	100 100	1.3	18 16	0.9	10 5	0.6
Fraktal	100 100	3.7	0 0	~	0 0	~

Tab. 5.20: Vergleich der durch die Drehung bewirkten Unterschiede im Lösungsverhalten. Angegeben ist links die Erfolgsrate der unrotierten Funktionen und rechts die der rotierten. Ähnliche Aufwände bei den lokalen Verfahren sind durch eine ~ gekennzeichnet. Bei eindeutig großen Unterschieden sind die Funktionen grau hinterlegt.

Abb. 5.144 und Abb. 5.145 verdeutlichen noch einmal das unterschiedliche Lösungsverhalten von GLEAM bei den Foxholes und bei der fünf-dimensionalen Variante der Rastrigin-Funktion. Neben den Unterschieden beim Evaluationsbedarf wird auch deutlich, daß die Bereiche günstiger Populationsgrößen weit auseinanderliegen. Die abnehmende Anzahl an Evaluationen bei der rotierten Variante der Rastrigin-Funktion in Abb. 5.145 erklärt sich durch das häufige Erreichen des Generationslimits, das hier auf 100000 festgesetzt wurde, oder der Stagnationsgrenzen (GDV oder GAk). Ein Erfolg stellt sich erst bei wesentlich größeren Populationen ein, worauf nachfolgend noch eingegangen wird.

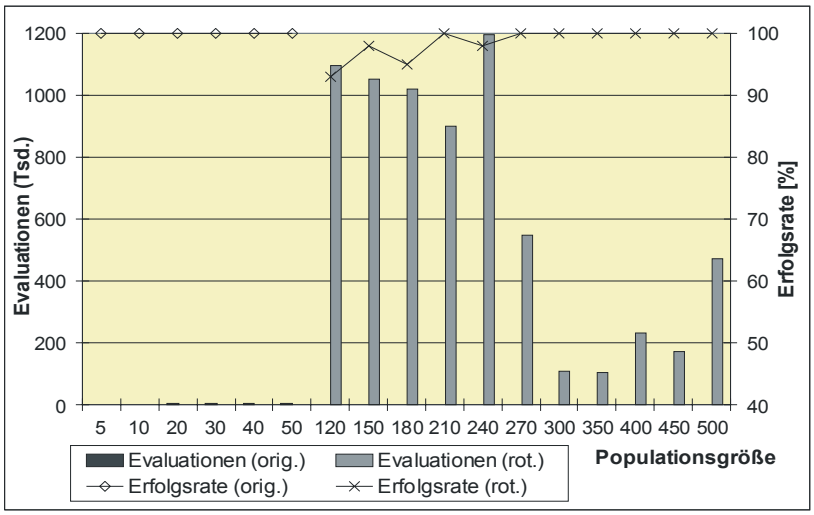


Abb. 5.144: Das Lösungsverhalten von GLEAM bei unrotierten und rotierten Shekel's Foxholes im Vergleich.

Am ähnlichsten ist das Lösungsverhalten von GLEAM hingegen bei der in Abb. 5.146 dargestellten Fletcher's Function. Der Bereich sicheren Auffindens der Lösung liegt in beiden Fällen ab einer Populationsgröße von 350 an aufwärts und der benötigte Aufwand differiert deutlich geringer. Aber auch bei der fraktalen Funktion (Abb. 5.147) gibt es ein ähnliches Lösungsverhalten, wenn auch die Unterschiede hinsichtlich des Aufwands etwas größer sind als bei Fletcher's Function. Der erforderliche Mindestaufwand differiert um einen vergleichweise moderaten Faktor von 3.7.

Zusammenfassend kann festgestellt werden, daß bei Testfunktionen mit einer Strukturierung der Funktion parallel zu den Koordinatenachsen Evolutionäre Algorithmen im Vorteil sind und daher solche Funktionen gedreht werden sollten.

Bei den im Rahmen der vorliegenden Arbeit untersuchten Hybridisierungsarten kann davon ausgegangen werden, daß lediglich die verzögerte und die unverzögerte direkte Integration wesentlich andere Ergebnisse bringen werden. Da sich die beiden lokalen Verfahren als relativ robust gegenüber der Drehung gezeigt haben und GLEAM weniger leistet, kann bei der Vor- und Nachoptimierung mit bestenfalls ähnlichen Ergebnissen wie

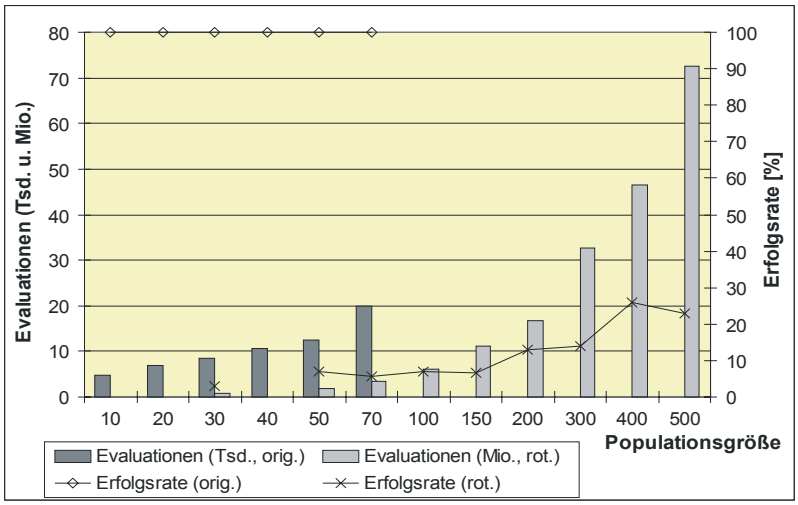


Abb. 5.145: Das Lösungsverhalten von GLEAM bei unrotierter und rotierter 5-dimensionaler Rastrigin Funktion im Vergleich.

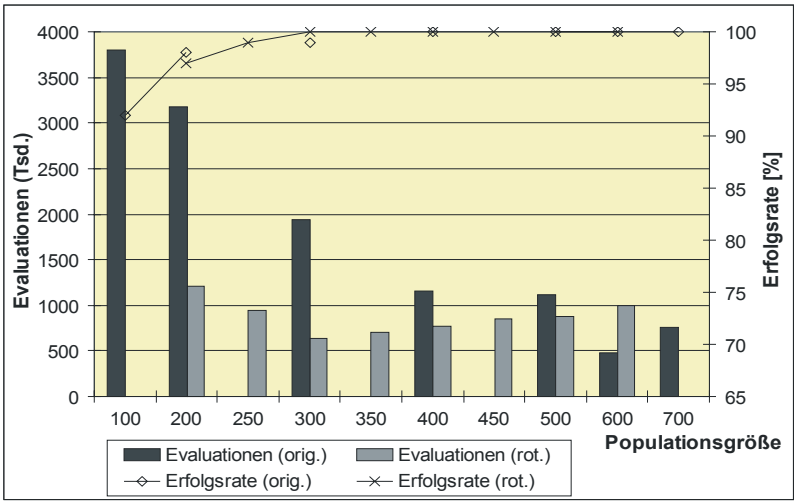


Abb. 5.146:Das Lösungsverhalten von GLEAM bei den unrotierten und rotierten Versionen von Fletcher's Function im Vergleich.

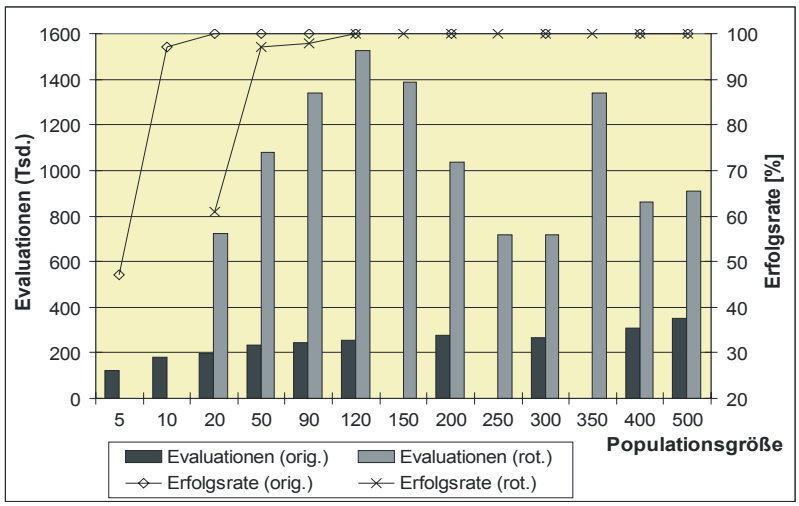


Abb. 5.147:Das Lösungsverhalten von GLEAM bei den unrotierten und rotierten Versionen von der fraktalen Funktion im Vergleich.

bei den ungedrehten Funktionen gerechnet werden. An Hand der beiden Funktionen, bei denen die Drehung den größten Effekt bewirkte, soll abschließend geprüft werden, ob die im vorigen Abschnitt gefundenen Ergebnisse mit der rotierten Funktion bestätigt werden können. Dazu wurde die verzögerte und unverzögerte direkte Integration mit allen Parametrierungen wie bisher mit einer Ausnahme untersucht: Bei der verzögerten direkten Integration wurde auf die Nischenparametrierung P2 verzichtet, da P1 und P3 ausreichen, um beurteilen zu können, ob strengere oder weniger strenge Nischenkriterien eine Rolle spielen.

5.2.3.1 Gedrehte Version von Shekel's Foxholes

Abb. 5.148 vergleicht die Ergebnisse der besten Parametrierungen beider direkt integrierter lokaler Verfahren. Der Vergleich beruht auf allen Jobs mit lokaler Verbesserung des besten (*best)* oder aller Nachkommen (*all)* sowie der niedrigen und der mittleren Präzision. Daraus wurden die Jobs ausgewählt, deren Lamarckraten die besten Ergebnisse gebracht haben. Auf die Darstellung der Ergebnisse mit hoher Präzision konnte verzichtet werden, da die besten Jobs mit 119030 Evaluationen (*GR, p20,h,best,l5*) für *best* und 385354 Evaluationen (*GR, p10,h,all,l5*) für *all* deutlich über den guten Werten von Abb. 5.148 liegen, siehe Anhang B.2.6. Da alle Jobs unter 20 Tausend Evaluationen kommen, wird in Abb. 5.149 ein Ausschnitt von Abb. 5.148 gezeigt, der die Unterschiede der guten Jobs klarer hervortreten läßt. Nun ist deutlich zu erkennen, daß die direkte Rosenbrock-Integration mit mittlerer Präzision, Lamarckscher Evolution und lokaler Verbesserung nur des besten Nachkommen (*GR,m,best,l100*) am besten abschneidet. Das Ergebnis bestätigt bis auf die etwas größerer Population damit die Resultate der bisherigen Untersuchungen. Die

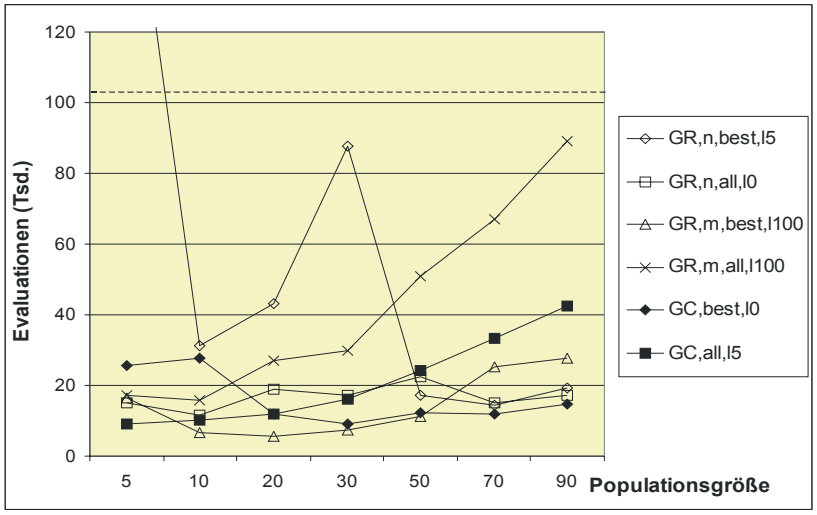


Abb. 5.148: Direkte Rosenbrock- und Complex-Integration: Vergleich der besten Jobs für die lokale Verbesserung aller oder nur des besten Nachkommen sowie niedrige und mittlere Präzision. Dargestellt sind die Jobs, deren Lamarckrate die besten Resultate lieferte. (Foxholes, rotiert)

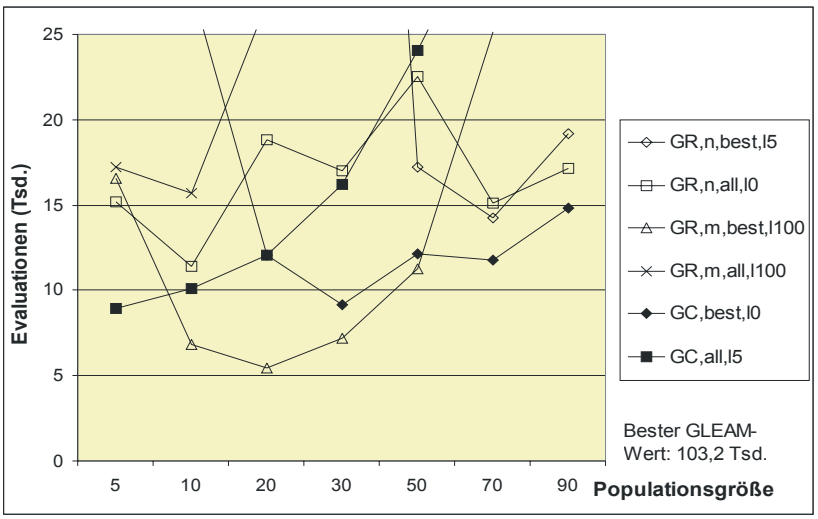


Abb. 5.149: Detailansicht aus Abb. 5.148. Dargestellt sind die Jobs aller Parametrierungen mit bis zu 25000 Evaluationen. (Foxholes, rotiert)

nächstbesten Ergebnisse ergeben allerdings die Complex-Integrationen mit reiner Baldwin-Evolution bzw. einer Lamarckrate von 5%. Ähnlich sieht es bei den beiden Rosenbrock-Jobs mit niedriger Präzision aus. Damit setzt sich der bei der Originalversion von Shekel's Foxholes beobachtete Trend fort, daß andere Lamarckraten als 100% oder die lokale Verbesserung aller Nachkommen ebenfalls gute Ergebnisse liefern.

Die verzögerte direkte Integration bringt hier keine Vorteile, wie aus Abb. 5.150 entnommen werden kann. Es zeigt die beiden Complex-Jobs und die zwei Rosenbrock-Jobs, deren Präzision das beste Resultat geliefert hat.

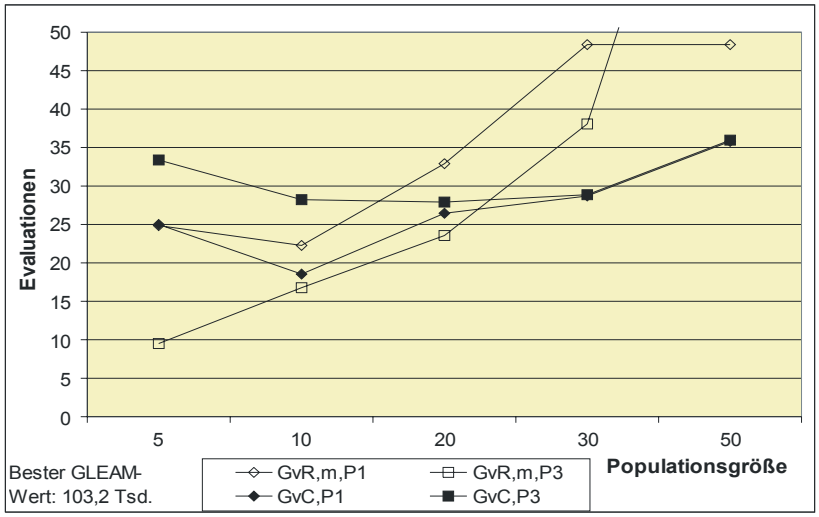


Abb. 5.150: Verzögerte direkte Integration: Vergleich der besten Rosenbrock-Präzision mit den Complex-Jobs. (Foxholes, rotiert)

Zusammenfassend vergleicht Abb. 5.151 die besten Jobs je untersuchter Integrationsart und Parametrierung. Im Falle der rotierten Foxhole-Funktion schneidet die direkte Rosenbrock-Integration bei mittlerer Präzision und reiner Lamarckscher Evolution am besten ab. Sie übertrifft GLEAM hinsichtlich der benötigten Evaluationen um den Faktor 18.8 und liegt damit im Bereich der bisher durch die Hybridisierungen erreichten Verbesserungen.

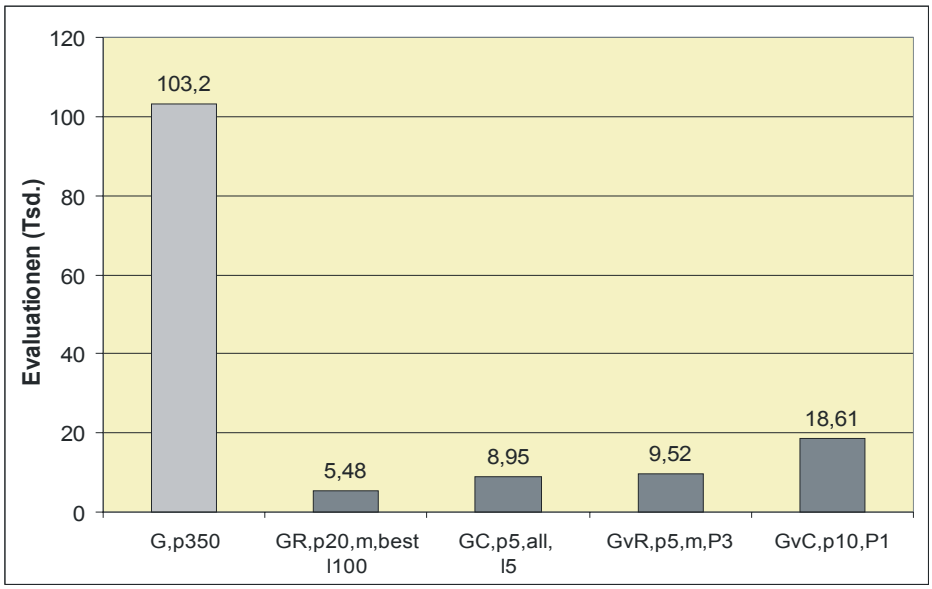


Abb. 5.151: Gesamtvergleich der besten Jobs je Hybridisierungsart und Parametrierung mit 100% Erfolgsrate im Vergleich mit dem besten GLEAM-Job. (Foxholes, rotiert)

5.2.3.2 Gedrehte Version der verallgemeinerten Rastrigin Funktion

Abb. 5.145 hat gezeigt, daß GLEAM bei der gedrehten Version der verallgemeinerten Rastrigin Funktion im Bereich bis zu einer Populationsgröße von 500 keinen Erfolg hat. Da die ersten erfolgreichen Jobs erst ab einer Populationsgröße von 6000 auftreten, werden die Untersuchungsergebnisse zur Ermittlung geeigneter Populationsgrößen auf die Bilder 5.152 und 5.153 aufgeteilt. Abb. 5.152 zeigt die Resultate von Populationen zwischen ein- und achttausend Individuen. Der Aufwand steigt mit sinkender Populationsgröße nicht kontinuierlich an, da bei den erfolglosen Läufen meist das Generationslimit, das hier mit 100000 vorgegeben wurde, erreicht wird. In Abb. 5.153 sind die Ergebnisse ab der ersten erfolgreichen Populationsgröße dargestellt. Der günstigste Wert ergibt sich bei 11200 Individuen mit einem durchschnittlichen Aufwand von 3518702 Evaluationen. Der Erfolg der in Abb. 5.153 dargestellten Jobs ist mit einer geringen Unsicherheit behaftet, da es immer wieder zu erfolglosen (z.B. *G,p12800*) oder außergewöhnlich aufwendigen Läufen (z.B. *G,p10400*) kommt. Das unterstreicht zusammen mit den vergleichsweise extrem großen Populationen die Schwierigkeit der gedrehten Variante der Rastrigin Funktion.

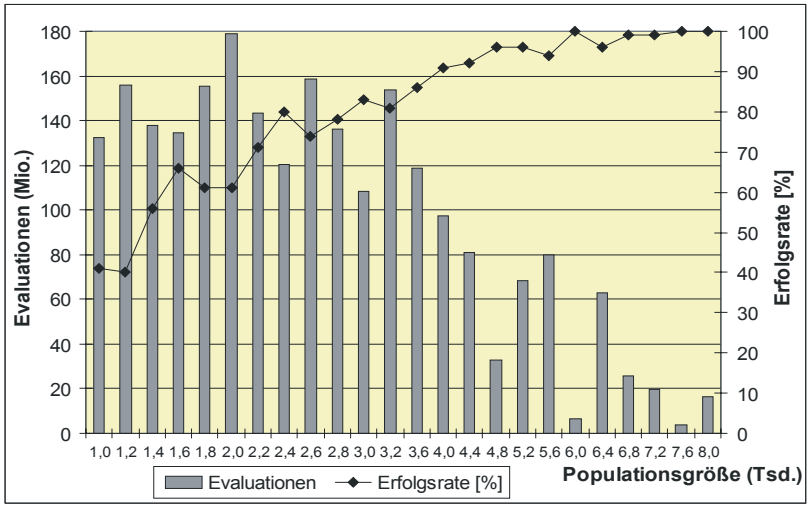


Abb. 5.152: GLEAM: Vergleich der Populationsgrößen zwischen ein- und achttausend. (Rastrigin, 5 Parameter, rotiert)

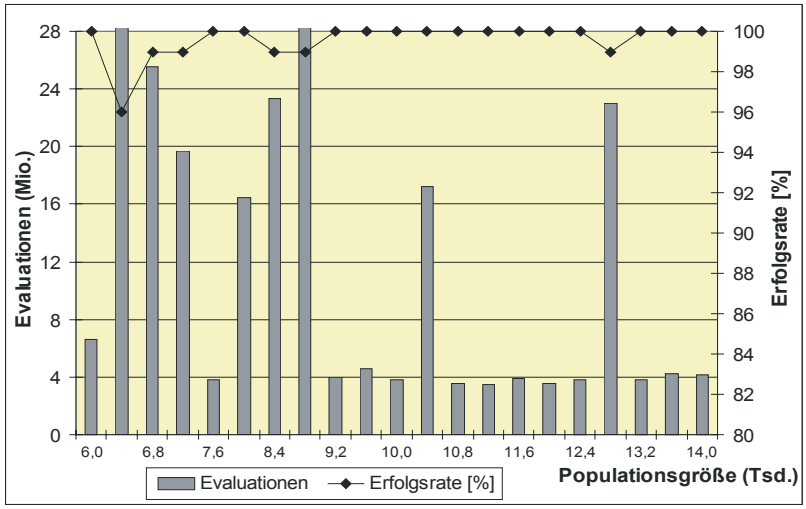


Abb. 5.153: GLEAM: Vergleich der Populationsgrößen zwischen 6000 und 14000. Aus Gründen der besseren Lesbarkeit wurde jede zweite Populationsgröße weggelassen. Die Schrittweite zwischen den Größen beträgt 400. (Rastrigin, 5 Param., rotiert)

Die direkte Integration hat auch im Falle der rotierten Rastrigin Funktion mit beiden lokalen Suchverfahren Erfolg, wie Abb. 5.154 zeigt. Es basiert auf allen erfolgreichen Jobs mit lokaler Verbesserung des besten oder aller Nachkommen sowie der niedrigen und der mittleren Präzision. Daraus wurden die Jobs ausgewählt, deren Lamarckraten die besten Ergebnisse gebracht haben. Die Jobs mit hoher Präzision werden nicht dargestellt, da sie alle mehr als 600000 Evaluationen benötigen, was deutlich über den anderen Ergebnissen liegt, siehe An-

hang B.3.6. Als erstes Ergebnis kann festgehalten werden, daß die Jobs mit Lamarckscher Evolution durchgängig am besten abschneiden. Die Populationsgröße des erfolgreichsten Jobs (*GR,p50,n,best,l100*) beträgt 50, ein Wert, der deutlich über den bisher festgestellten Größen von 5 - 10 liegt. Allerdings kommt der nächst erfolgreiche Job, nämlich *GR,p10,n,all,l100*, mit einer Populationsgröße von 10 wieder in den gewohnten Bereich. Außerdem fällt auf, daß der Unterschied zwischen den *best, l100*- und den *all,l100*-Jobs vergleichsweise gering ist und daß bei der lokalen Optimierung nur des besten Nachkommens einer Paarung kein Job mit geringeren Populationsgrößen als 20 erfolgreich ist. Alle genannten Beobachtungen zur Populationsgröße können mit dem gesteigerten Schwierigkeitsgrad der gedrehten Benchmarkfunktion erklärt werden, der sich auch bereits in den extrem großen Populationen der erfolgreichen GLEAM-Jobs gezeigt hat. Der Verbesserungsfaktor des besten Jobs der direkten Integration gegenüber dem besten GLEAM-Job liegt mit 37.4 im Bereich des bisher Beobachteten.

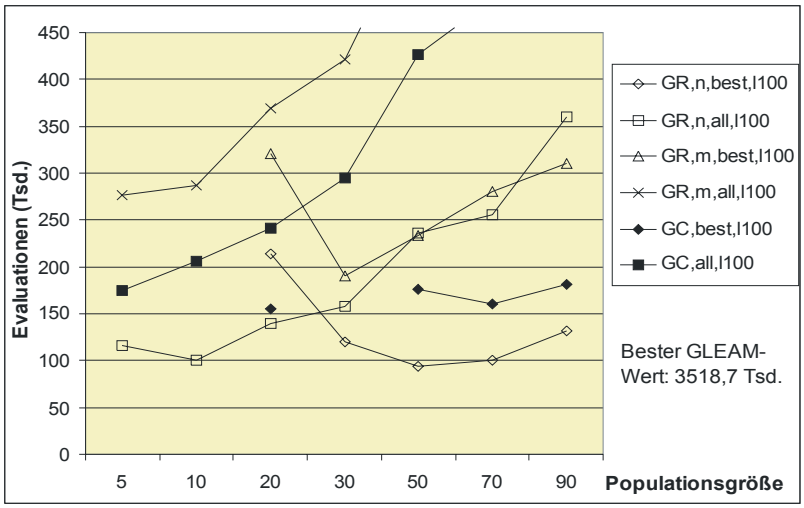


Abb. 5.154: Direkte Rosenbrock- und Complex-Integration: Vergleich der besten Jobs für die lokale Verbesserung aller oder nur des besten Nachkommen sowie niedrige und mittlere Präzision. Dargestellt sind alle erfolgreichen Jobs, deren Lamarckrate die besten Resultate lieferte. Aus Gründen der besseren Darstellung wurden die Jobs mit größeren Populationen bei GR,m,all,l100 und bei GC,all,l100 weggeglassen.(Rastrigin, 5 Parameter, rotiert)

Bei der verzögerten direkten Integration schneidet die Verwendung des Complex-Verfahrens deutlich schlechter ab als bei der unverzögerten, wie Abb. 5.155 zeigt. Die verzögerte Rosenbrock-Integration liegt beim günstigsten Job um etwa den Faktor zwei schlechter als der beste unverzögerte. Wie bei der unverzögerten Variante liefert die Verwendung der niedrigen Präzision die besten Resultate, diesmal bei einer Populationsgröße von 30. Die Rosenbrock-Jobs mit

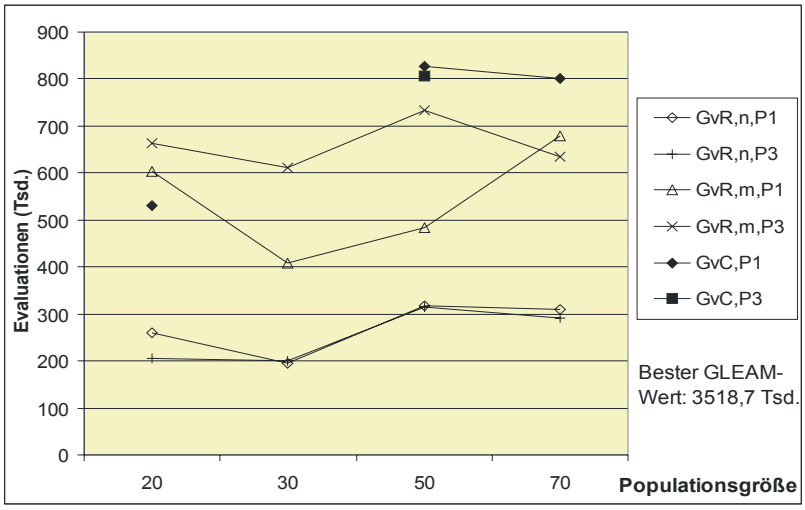


Abb. 5.155: Verzögerte direkte Rosenbrock- und Complex-Integration: Vergleich der besten Jobs der niedrigen und mittleren Rosenbrock Präzision mit den Complex-Jobs.(Rastrigin, 5 Parameter, rotiert)

hoher Präzision liegen mit einem Aufwand von 1.3 Millionen und höher deutlich über den Werten von Abb. 5.155. Im Gegensatz zu den im vorigen Abschnitt untersuchten Benchmarkfunktionen schneidet die verzögerte direkte Integration bei beiden gedrehten Funktionen deutlich schlechter ab als die unverzögerte. Dies unterstreicht die Aussage von Abschnitt 5.2.2.4, wonach die Frage der vorteilhaften Anwendung der verzögerten direkten Integration nur aufgabenspezifisch beantwortet werden kann.

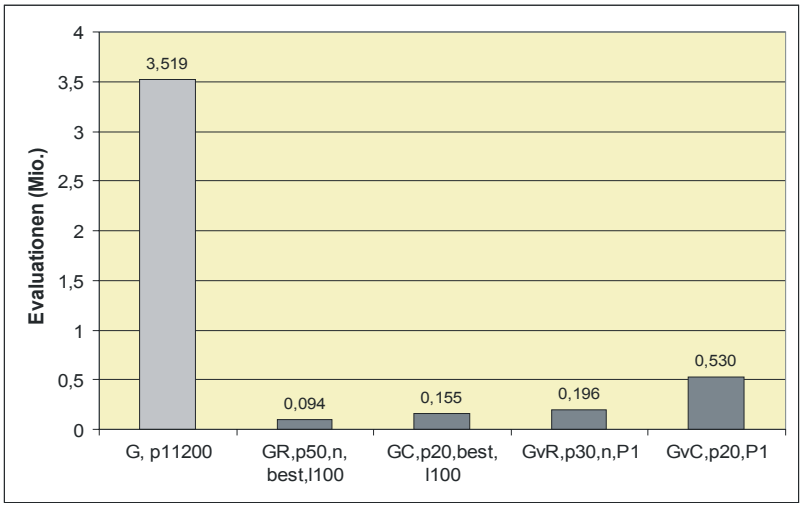


Abb. 5.156: Gesamtvergleich der besten Jobs je Hybridisierungsart und Parametrierung mit 100% Erfolgsrate im Vergleich mit dem besten GLEAM-Job. (Rastrigin, 5 Parameter, rotiert)

Abb. 5.156 faßt die Ergebnisse der (verzögerten) direkten Integration bei der rotierten verallgemeinerten Rastrigin Funktion zusammen. Die Überlegenheit der unverzögerten direkten Rosenbrock-Integration ist deutlich zu erkennen.

5.2.3.3 Auswertung der Ergebnisse der gedrehten Benchmarkfunktionen

Durch die Drehung der Benchmarkfunktionen Shekel's Foxholes und der 5-dimensionalen verallgemeinerten Rastrigin Funktion konnte gezeigt werden, daß die beiden Funktionen damit für GLEAM deutlich schwerer werden und der so entstandene Raum für Verbesserungen durch die (verzögerte) direkte Integration auch im bisher üblichen Rahmen ausgefüllt wird. Dabei schneidet die verzögerte Variante der direkten Integration bei beiden LSV und beiden Testfällen regelmäßig schlechter ab als die unverzögerte.

Tabelle 5.21 vergleicht die Parametrierungen der beiden erfolgreichsten Jobs für die direkte Integration mit beiden LSV für die gedrehten und ungedrehten Funktionen. Da die direkte Complex-Integration bei der ursprünglichen Rastrigin Funktion wegen extrem langer Laufzeiten nicht weiter untersucht werden konnte, fehlen die entsprechenden Einträge in der Tabelle. Wie bei den bisher behandelten Benchmarkaufgaben zeigt sich auch hier ein uneinheitliches Bild der Parametrierungen. Eines fällt jedoch auf und unterscheidet die Ergebnisse: Bei beiden gedrehten Benchmarkfunktionen spielt die lokale Optimierung aller Nachkommen einer Paarung (*all*) eine größere Rolle als bisher. Da dies bei der gedrehten Rastrigin Funktion durchgängig zu beobachten ist, kann es als ein Hinweis darauf gedeutet werden, daß es bei schwierigen Problemen mit stark multimodaler Fitnessfunktion günstig sein kann, mehr als nur den besten Nachkommen lokal zu verbessern. Um diese Vermutung zu erhärten sind weitere, über den Umfang der vorliegenden Arbeit hinausgehende Untersuchungen notwendig.

Aufgabe	LSV	Ver-besserung	Populations-größe	Präzision	maximale Präzision	best / all	Lamarck-rate
Foxholes-1	R	0.54	20	n	h	best	100
Foxholes-2	R	0.52	5	n	h	all	100
Foxholes-rot.-1	R	18.85	20	m	h	best	100
Foxholes-rot.-2	R	13.84	20	m	h	best	5
Rastrigin-1	R	1.18	5	m	m	best	100
Rastrigin-2	R	0.54	20	n	m	best	100
5dim-Rastrigin-rot-1	R	37.38	50	n	h	best	100
5dim-Rastrigin-rot-2	R	34.97	10	n	h	all	100
Foxholes-1	C	0.22	5			all	100
Foxholes-2	C	0.20	20			best	100
Foxholes-rot.-1	C	11.53	5			all	5
Foxholes-rot.-2	C	11.30	30			best	0
5dim-Rastrigin-rot-1	C	22.68	20			best	100
5dim-Rastrigin-rot-2	C	20.06	5			all	100

Tab. 5.21: Direkte Rosenbrock- und Complex-Integration: Vergleich der Parametrierungen der beiden Jobs mit den größten Verbesserungen hinsichtlich des Aufwands im Vergleich zum jeweils besten GLEAM-Job für die beiden rotierten und unrotierten Benchmarkfunktionen.

5.3 Ergebniszusammenfassung

Dieser Abschnitt faßt die im Abschnitt 5.2 detailliert ausgewerteten Ergebnisse der einzelnen Benchmarkaufgaben und der gedrehten Varianten zweier mathematischer Testfunktionen zusammen, indem zunächst die Bilder zum Gesamtvergleich jeder Benchmarkaufgabe angegeben werden. Die Bilder enthalten den besten GLEAM-Job als Vergleich und die jeweils besten Ergebnisse aller Hybridisierungsarten, die mindestens einen *erfolgreichen Job*, worunter ein Job mit einer Erfolgsrate von 100% verstanden wird, aufzuweisen haben. Der besseren Übersicht halber sind die Parametrierungen der Jobs in den Bildern unterhalb der Bezeichnungen für die Hybridisierungsarten angegeben. Danach erfolgt eine Betrachtung des Konvergenzverhaltens gefolgt von einem Vergleich der Untersuchungsergebnisse. Eine Empfehlung zur praktischen Umsetzung beendet den Abschnitt.

Die Details der Untersuchungen können für die einzelnen Benchmarkaufgaben den Abschnitten 5.2.1.1 bis 5.2.1.8 entnommen werden. Hinsichtlich der Analyse der Ergebnisse je Hybridisierungsart wird auf die Abschnitte 5.2.2.1 bis 5.2.2.4 verwiesen.

Abb. 5.157 zeigt die Ergebnisse von Schwefel's Sphere. Hier gibt es gleich zwei Ausnahmen vom zuvor angekündigten Bildinhalt: Erstens ist GLEAM bei dieser Aufgabe nicht erfolgreich und somit steht kein GLEAM-Job als Vergleichswert zur Verfügung. Stattdessen wird hier der erfolgreichste Rosenbrock-Job angegeben, der allerdings nur bei der höchsten (und unüblichen) Präzision *v* Erfolg hat. Zweitens enthält das Bild einen Nachoptimierungsjob, obwohl er streng genommen die Erfolgsmarge nicht erreicht hat. Er ist hier trotzdem enthalten,

da er sehr gute Ergebnisse geliefert hat: Die schlechteste Fitness von 100 Läufen war nur 0.0007% vom Zielwert entfernt. Angesichts der verwendeten Präzision, bei der gewöhnlich eine Nichtkonvergenz des Rosenbrock-Verfahrens eintritt, können die Hybridisierungen durchaus mit dem Rosenbrock-Verfahren konkurrieren, siehe auch Abschnitt 5.2.1.1.

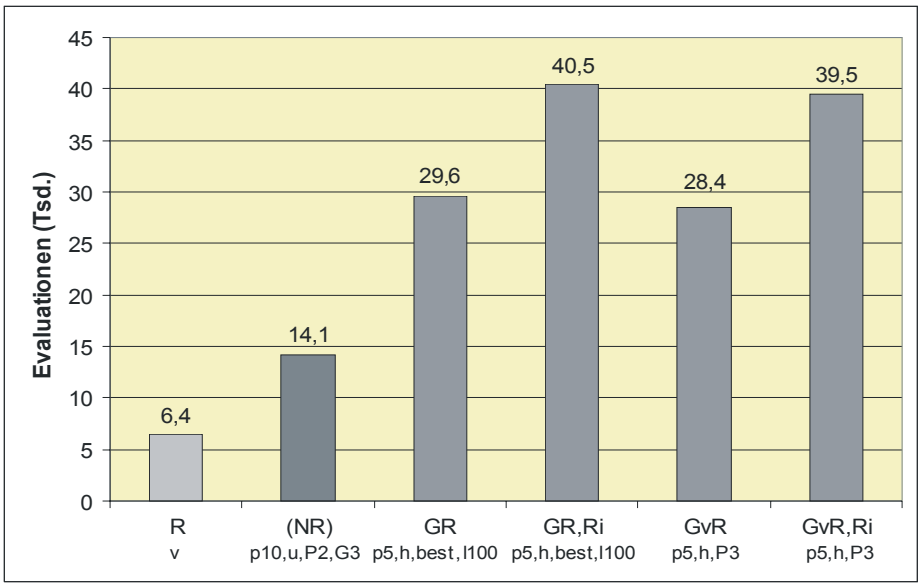


Abb. 5.157: Schwefel's Sphere: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart. Die Rosenbrock-Nachoptimierung (NR) ist nur fast erfolgreich, siehe Text.

Bei Shekel's Foxholes sind wesentlich mehr Hybridisierungsarten erfolgreich, wie Abb. 5.158 zeigt. Am besten schneidet hier die Voroptimierung mit dem Rosenbrock ab. Da GLEAM noch bei der extrem kleinen Populationsgröße von fünf bei allen Läufen das Optimum findet und der Aufwand mit fallenden Populationsgrößen stetig sinkt, muß diese Aufgabe als so „einfach" für den EA angesehen werden, daß kaum Raum für Verbesserungen durch eine Hybridisierung bleibt, siehe auch Abschnitt 5.2.1.

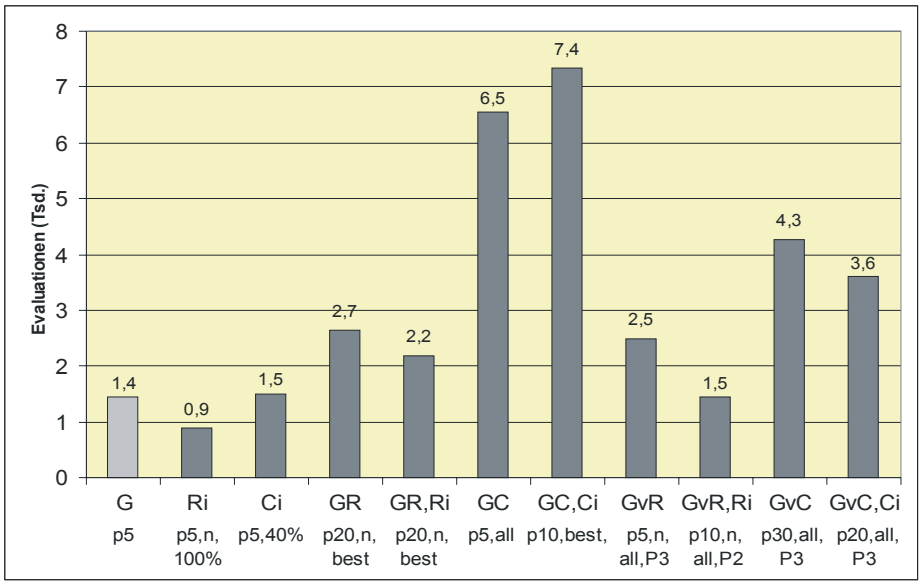


Abb. 5.158: Shekel's Foxholes: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart. Alle Jobs der (verzögerten) direkten Integration mit reiner Lamarckscher Evolution.

Wie in Abschnitt 5.2.3 dargelegt, wurde daher die Aufgabe durch Drehung um einen geeigneten Winkel (hier 30°) erschwert. Die rotierte Version wurde mit der verzögerten und der unverzögerten direkten Integration beider lokaler Verfahren getestet, da sich diese beiden als die erfolgreichsten Hybridisierungsarten erwiesen haben. Abb. 5.159 zeigt die Ergebnisse. Mit der gedrehten Funktion konnte eine Verringerung des Aufwands gegenüber GLEAM um den Faktor 18.8 bei der direkten Rosenbrock-Integration erreicht werden. Da auch die direkte Complex-Integration mit einem Faktor von 11.5 eine beachtliche Verbesserung erzielt, reiht sich der Fall der gedrehten Foxhole Funktion in die Reihe der Erfolge der direkten Integration ein.

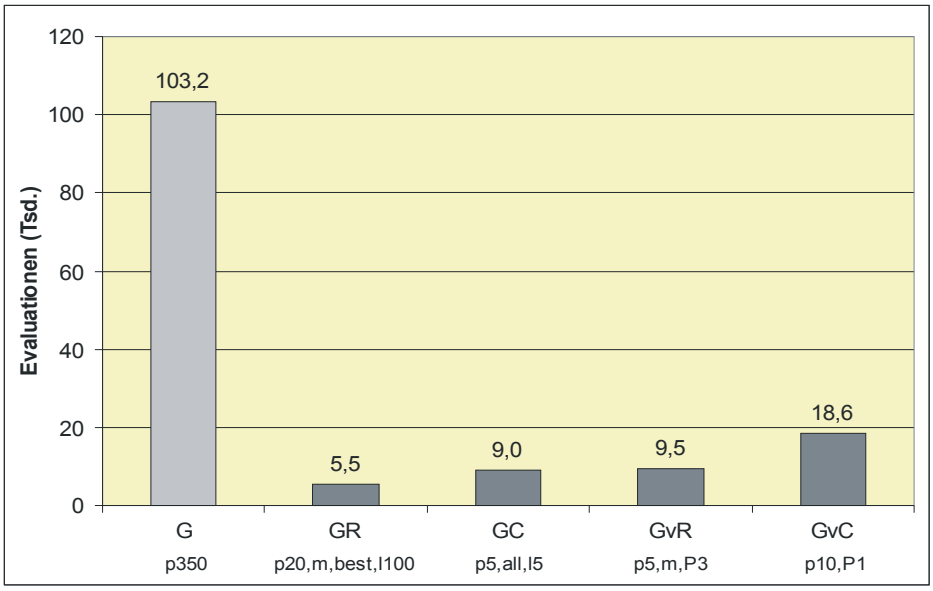


Abb. 5.159: Shekel's Foxholes, um 30° gedreht: Gesamtvergleich der besten Jobs je erfolgreicher (verzögerter) direkter Integration.

Für die verallgemeinerte Rastrigin Funktion gilt hinsichtlich der Lösungsfähigkeit durch GLEAM und dem damit verbundenen Schwierigkeitsgrad für den EA ähnliches wie für Shekel's Foxholes. Auch hier wird die Aufgabe mit einer Population von nur fünf Individuen ge-

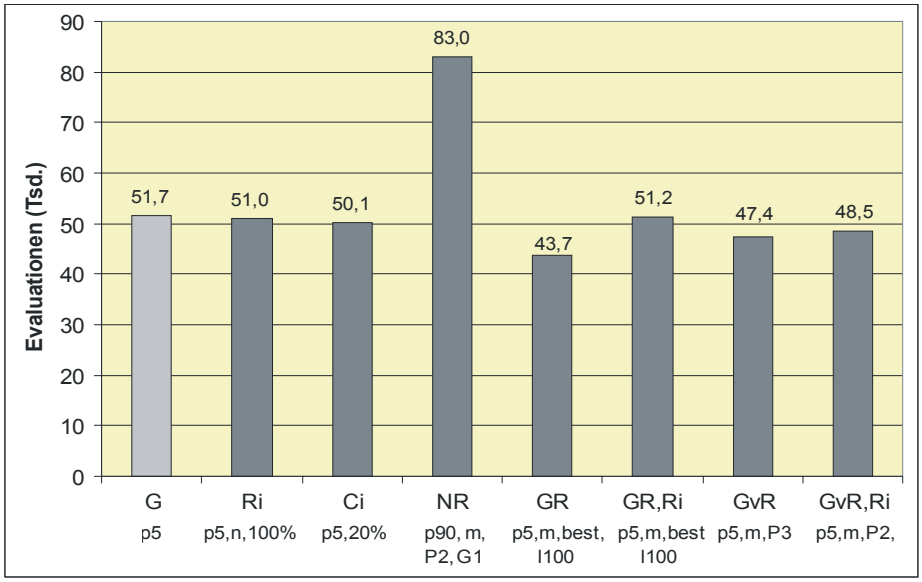


Abb. 5.160: 20-dimensionale verallgemeinerte Rastrigin Funktion: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

löst und lediglich die (verzögerte) direkte Rosenbrock-Integration kann gegenüber GLEAM mit einem kleinen Vorteil aufwarten, wie Abb. 5.160 zeigt. Bemerkenswert ist auch, daß die sonst erfolglose Nachoptimierung hier mit dem Rosenbrock-Verfahren einen Erfolg aufzuweisen hat.

Wie bei Shekel's Foxholes wurde eine um 30^o gedrehte Version der Rastrigin Funktion getestet. Dabei stellte es sich heraus, daß GLEAM die ursprünglich 20-dimensionale Variante der Funktion nicht lösen konnte, weswegen die weiteren Untersuchungen auf den 5-dimensionalen Fall beschränkt wurden, siehe auch Abschnitt 5.2.3. Die rotierte 5-dimensionale Version wurde ebenfalls mit der verzögerten und der unverzögerten direkten Integration beider lokaler Verfahren getestet. Dabei konnte die direkte Rosenbrock-Integration mit einem Verbesserungsfaktor von 37.4 am besten abschneiden, gefolgt von der direkten Complex-Integration mit einem Faktor von 30.6, siehe Abb. 5.161. Auch hier schneiden die verzögerten Varianten schlechter ab.

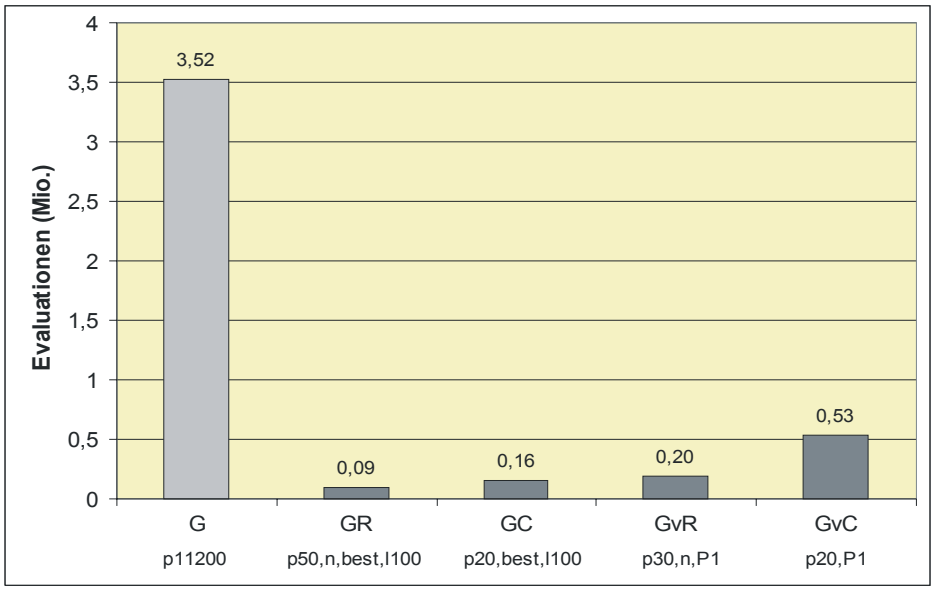


Abb. 5.161: 5-dimensionale verallgemeinerte Rastrigin Funktion, um 30^o gedreht: Gesamtvergleich der besten Jobs je erfolgreicher (verzögerter) direkter Integration.

Bei Fletcher's Function benötigt GLEAM eine Population mit 600 Individuen, um das Problem zuverlässig mit über 480 Tausend Evaluationen zu lösen. Die Voroptimierung, aber vor allem die (verzögerte) direkte Integration beider LSV nutzen diesen Raum für erhebliche Verbesserungen, wie Abb. 5.162 im Überblick zeigt. Um die Unterschiede zwischen den verschiedenen Hybridisierungsarten deutlicher hervortreten zu lassen, werden sie ohne GLEAM in

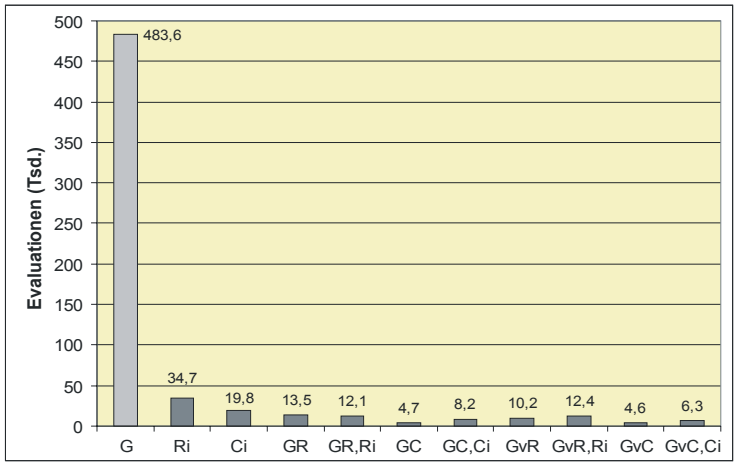


Abb. 5.162: Fletcher's Function: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

Abb. 5.163 dargestellt. Darin ist auch der hochgerechnete Wert der reinen Complex-Optimierung zum Vergleich enthalten. Wie in Abschnitt 5.2.1.4 ausgeführt, ergibt er sich, wenn eine 99,5% Erfolgswahrscheinlichkeit vom Complex-Verfahren mit seiner Erfolgsrate von 10% gefordert wird. Das Bild zeigt deutlich, daß hier die direkte Complex-Integration erstmals der direkten Rosenbrock-Integration überlegen ist und die Verzögerung Vorteile bringt. Eine detaillierte Analyse dazu kann in den Abschnitten 5.2.2.3 und 5.2.2.4 nachgelesen werden. Die verzögerte direkte Complex-Integration ergibt einen Verbesserungsfaktor von 104.2.

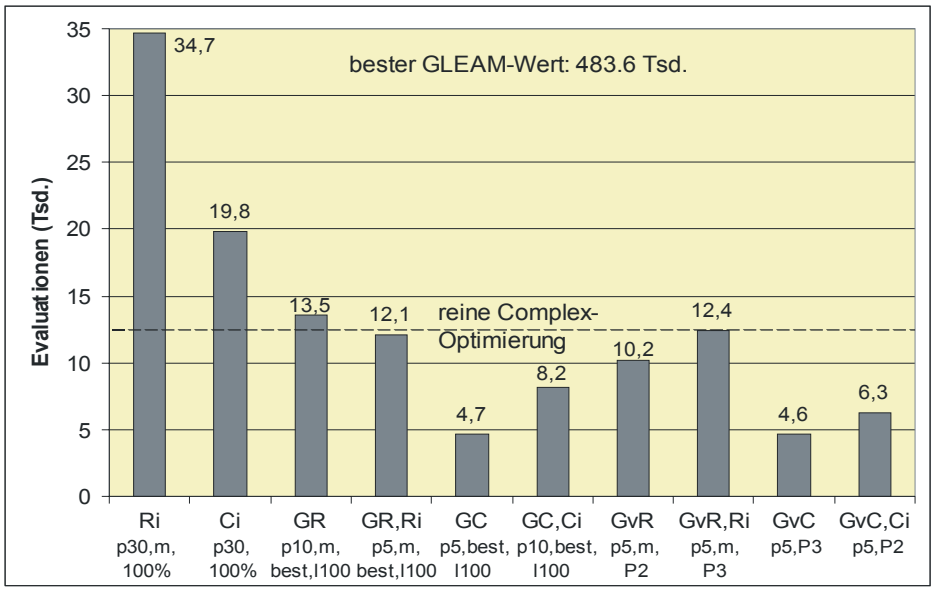


Abb. 5.163: Fletcher's Function: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart ohne den besten GLEAM-Job (Detailansicht von Abb. 5.162).

Die fraktale Funktion bringt wieder ein Beispiel erfolgreicher Rosenbrock-Nachoptimierung wie Abb. 5.164 zeigt. Die Jobs der (verzögerten) direkten Complex-Integration sind hier eingeklammert, da sie auf nur jeweils 10 erfolgreichen Läufen beruhen. Die Laufzeiten waren so groß, daß weitergehende Untersuchungen auch angesichts der erheblichen Anzahl an Evaluationen nicht sinnvoll erschienen. Die auf dem Prinzip der direkten Integration beruhenden Rosenbrock-Hybridisierungsarten unterbieten GLEAM, wobei die direkte Integration mit Voroptimierung mit einem Verbesserungsfaktor von 6.5 am besten abschneidet. Allerdings sind die Jobs ohne Voroptimierung und mit Verzögerung kaum schlechter, wie Abb. 5.165 zeigt. In dem Bild sind die Hybridisierungsarten mit schlechteren Ergebnissen als GLEAM weggelassen, damit die Unterschiede der besseren Jobs deutlicher werden. Wie fast alle bisher behandelten mathematischen Benchmarkfunktionen liefert auch die fraktale Funktion bei reiner Lamarckscher Evo-

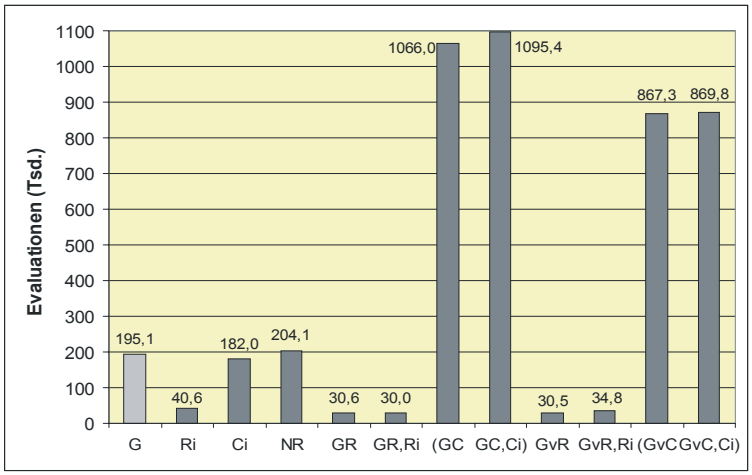


Abb. 5.164: Fraktale Funktion: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

lution die besten Ergebnisse. Die Frage, ob die Verbesserung aller oder nur des besten Nachkommens einer Paarung günstiger sei, wird ebenfalls wie bei den meisten anderen Funktionen zu Gunsten des besten Nachkommens entschieden.

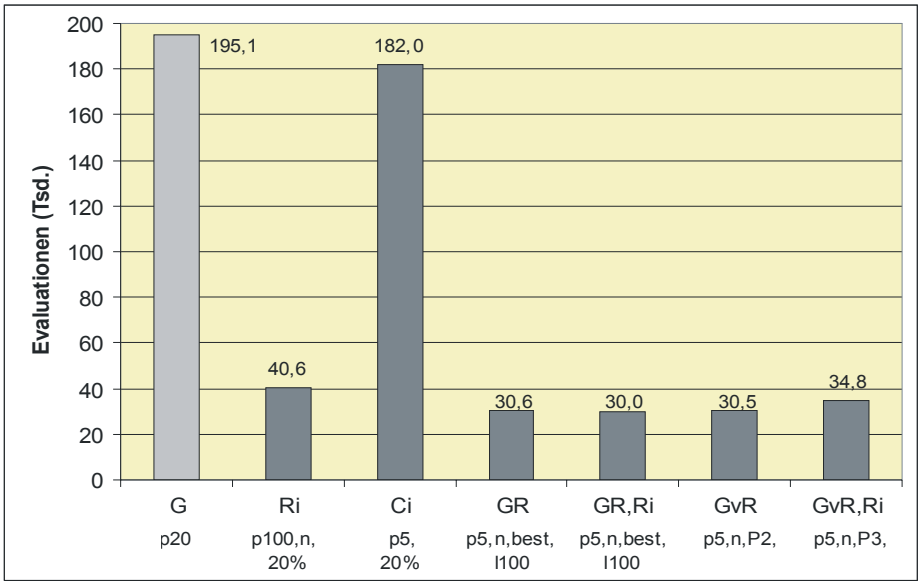


Abb. 5.165: Fraktale Funktion: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart, deren Ergebnisse GLEAM übertreffen (Detailansicht von Abb. 5.164).

Die Designoptimierungsaufgabe setzt den Trend der erfolgreichen (verzögerten) direkten Integration mit beiden LSV fort, wie Abb. 5.166 zeigt. Dazu kommen noch die Voroptimierung und ein erfolgreicher Nachoptimierungsjob. Zum Vergleich sind auch die hochgerechneten Aufwände der reinen LSV-Optimierungen (vgl. Abschnitt 5.2.1.6) eingezeichnet. Abb. 5.167 zeigt nur die Jobs, die weniger Aufwand als GLEAM verursachen, um die Unterschiede zu verdeutlichen. Am günstigsten schneidet die direkte Complex-Integration ab und zwar nahezu unabhängig davon, ob verzögert oder nicht und mit oder ohne Voroptimierung. Die größte Verbesserung gegenüber GLEAM schafft die verzögerte Complex-Integration mit einem Faktor von 5.8. Bei der schlechter abschneidenden Rosenbrock-Integration dominiert eine Lamarckrate von 5%, während die Complex-Integration mit reiner Lamarckscher Evolution am besten arbeitet. Die Frage nach der Verbesserung des besten oder aller Nachkommen einer Paarung wird bei dieser Aufgabe eindeutig zu Gunsten des besten entschieden.

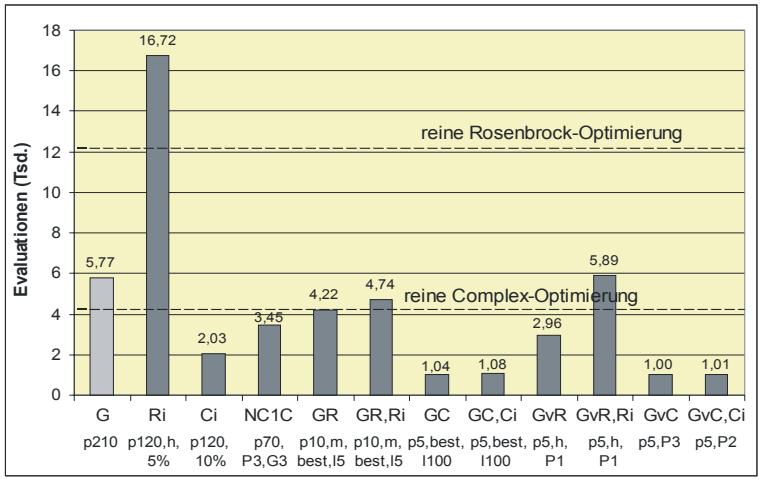


Abb. 5.166: Designoptimierung: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

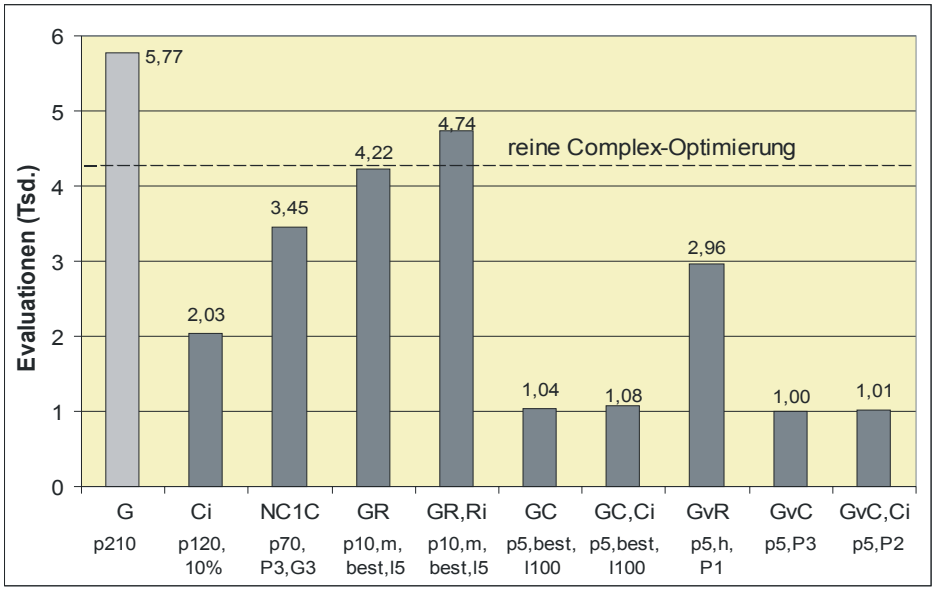


Abb. 5.167: Designotimierung: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart, deren Ergebnisse GLEAM übertreffen (Detailansicht von Abb. 5.166).

Bei der Ressourcenoptimierung treten wieder wie bei der gedrehten Rastrigin Funktion ungewöhnlich hohe Populationsgrößen auf: GLEAM benötigt 1800 Individuen, um die Aufgabe mit einer Erfolgsrate von 100% zu lösen. Wie in Abschnitt 5.2.1.7 dargelegt, entfällt wegen des kombinatorischen Charakters dieser und der nächsten Aufgabe die Complex-Nachoptimierung unter Verwendung aller GLEAM-Ergebnisse zur Bildung eines Startcomplexes. Außerdem wurde in Voruntersuchungen eine geeignete Sonderparametrierung *s* für das Rosenbrock-Verfahren ermittelt. Abb. 5.168 vergleicht die bei dieser Aufgabe einzigen erfolgreichen Hybridisierungsarten, nämlich die (verzögerte) direkte Rosenbrock-Integration mit und ohne Voroptimierung mit GLEAM. Abweichend von den bisher benutzten Parametrierungen für die verzögerte direkte Integration sind hier Nischenkriterien (*P7*) erfolgreich, die ein frü-

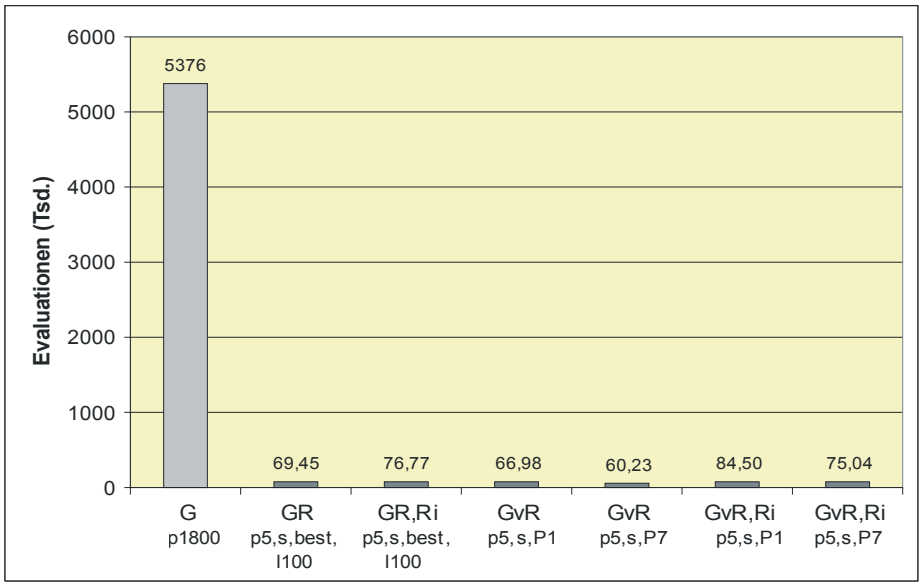


Abb. 5.168: Ressourcenoptimierung: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

heres Zuschalten des LSVs bewirken. Allerdings sind die Unterschiede zwischen *GvR,p5,s,P1* und *GvR,p5,s,P7* nicht groß. Die zusätzliche Voroptimierung bringt hier keine Verbesserung. Das beste Ergebnis liefert die verzögerte direkte Rosenbrock-Integration mit einem Verbesserungsfaktor von 89.3.

Bei der Roboterbahnplanung spielt der kombinatorische Aspekt eine noch viel größere Rolle als bei der Ressourcenoptimierung, da bei letzterer die Reihenfolge der Chargen durch geeignete Starttermine quasi überschrieben werden konnte. Das ist bei den Befehlssequenzen für die Robotersteuerung nicht der Fall. Für die lokale Verbesserung stehen nur die Motorgeschwindigkeiten und -beschleunigungen zur Verfügung, während der Zeit-Parameter der wichtigen Aktion zur unveränderten Beibehaltung der Motoreinstellungen nicht mitoptimiert werden kann, wie in Abschnitt 5.2.1.8 begründet wird. Auf Grund der sehr großen Laufzeiten von 30 Stunden und mehr konnten nur wenige Untersuchungen durchgeführt werden. Bis auf die Voroptimierung hatte keine Hybridisierungsart Erfolg. Abb. 5.169 zeigt das Ergebnis. Bei diesem Typ von Aufgabenstellung stoßen die hier behandelten Hybridisierungsarten offensichtlich an ihre Grenzen.

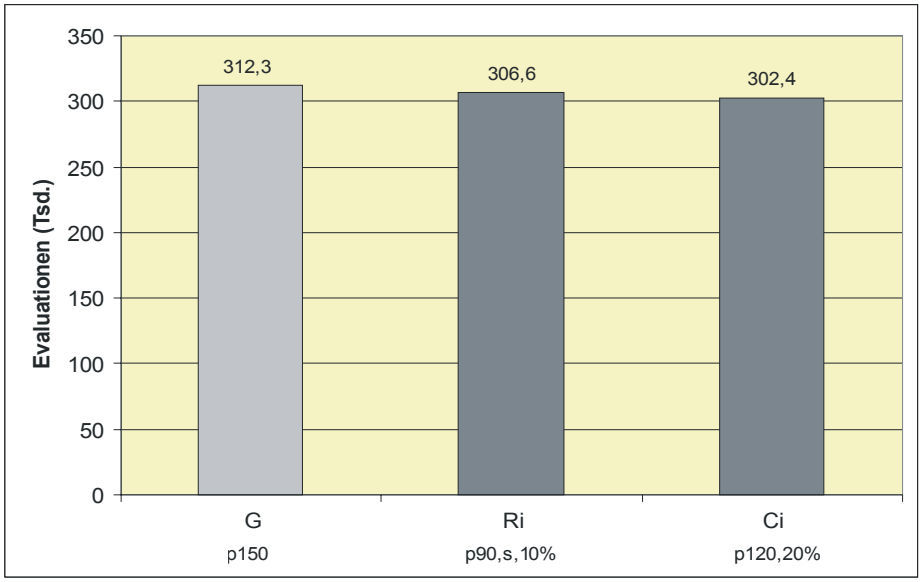


Abb. 5.169: Roboterbahnplanung: Gesamtvergleich der besten Jobs je erfolgreicher Integrationsart.

Zur Beantwortung der Frage, welche Lamarckrate am günstigsten ist und ob alle (*all*) oder nur der beste Nachkomme (*best*) einer Paarung lokal verbessert werden soll, werden die Jobs der unverzögerten direkten Integration verglichen, da bei dieser Integrationsart alle Parametrierungen getestet wurden, soweit die Laufzeiten dies zugelassen haben. Aus den Tabellen 5.13, 5.14 und 5.21 kann eine Dominanz der *best*-Parametrierung abgelesen werden. Bei schwierigen Aufgaben, wie den rotierten Funktionen, kann es aber vorkommen, daß *all* Ergebnisse liefert, die an *best* heranreichen. Bei der Lamarckrate dominiert der 100%-Wert (Lamarcksche Evolution), aber im Einzelfall wie bei der (rotierten) Foxhole-Funktion und der Designoptimierung liefert auch die 5%-Rate günstige oder sogar das beste Ergebnis. Wesentlich uneinheitlicher sieht es dagegen bei der Populationsgröße oder der Rosenbrock-Präzision aus. Als einziger Trend kann festgehalten werden, daß die Populationsgrößen mit Werten zwischen 5 und 50 im Vergleich zu reiner GLEAM-Optimierung wesentlich kleiner ausfallen.

5.3.1 Konvergenzverhalten

Da es bekanntlich bisher nicht gelang, streng mathematische Beweise für die Konvergenz Evolutionärer Algorithmen bei nichtlinearen Aufgabenstellungen zu finden, bleibt die Frage der Konvergenzsicherheit ein offenes Problem. Trotzdem können auf Grund der vorliegenden Untersuchungen empirisch begründete Aussagen über das Konvergenzverhalten gemacht werden.

Wie in Abschnitt 5.2.1 dargelegt, zeigt GLEAM bei neun der zehn Testaufgaben[2] ab einer hinreichenden Populationsgröße ein sicheres Konvergenzverhalten. Die Ausnahme ist die mathematische Benchmarkaufgabe Schwefel's Sphere, bei der die Läufe wegen extrem langer Laufzeiten abgebrochen wurden. Die gelieferten Resultate waren sehr nahe am geforderten Zielwert, so daß von einer extrem langsamen Annäherung an das Optimum ausgegangen werden kann. Das Beispiel bestätigt die oft beobachtete langsame Konvergenz Evolutionärer Algorithmen in der Nähe des Optimums.

Interessant ist nun die Frage, ob und wie sich das Konvergenzverhalten bei den jeweils erfolgreichsten Hybridisierungsarten verbessert hat. Tabelle 5.22 faßt die Ergebnisse zusammen, wobei alle Angaben auf 100 Läufen basieren und die Verbesserungen gegenüber GLEAM als Quotient der vom besten GLEAM-Job benötigten Evaluationen durch die Evaluationen des zu vergleichenden Jobs berechnet sind.

Testaufgabe	Hybridisierungsart	Erfolgsrate [%]	Verbesserung gegenüber GLEAM
Schwefel's Sphere	verzögerte direkte Rosenbrock-Integration	100	28400 Evaluationen statt 26 Millionen und mehr
Shekel's Foxholes	Rosenbrock-Voroptimierung	100	1.60
Shekel's Foxholes (rotiert)	direkte Rosenbrock-Integration	100	18.85
Rastrigin Funktion	direkte Rosenbrock-Integration	100	1.18
Rastrigin Funktion (rotiert)	direkte Rosenbrock-Integration	100	37.38
Fletcher's Function	verzögerte direkte Complex-Integration	100	104.22
Fraktale Funktion	direkte Rosenbrock-Integration mit Rosenbrock-Voroptimierung	100	6.50
Designoptimierung	verzögerte direkte Complex-Integration	100	5.80
Ressourcenplanung	verzögerte direkte Rosenbrock-Integration	100	89.27
Roboterbahnplanung	Complex-Voroptimierung	100	1.03

Tab. 5.22: Verbesserung der Konvergenzgeschwindigkeit durch Hybridisierung.

Das wichtigste Ergebnis ist, daß bei allen Testaufgaben das Optimierungsziel mit einer Erfolgsquote von 100% gefunden wurde. Bei Schwefel's Sphere konnte die Aufgabe nur durch die hybride Form von GLEAM gelöst werden, wenn von dem Rosenbrock-Job mit extremer

2. Die beiden rotierten Benchmarkfunktionen können in diesem Zusammenhang als eigenständige Testfälle betrachtet werden.

Parametrierung einmal abgesehen wird. Damit konnte die Konvergenzsicherheit nicht nur gewahrt sondern sogar verbessert werden. Dies ging nicht zu Lasten der Konvergenzgeschwindigkeit, die zum Teil drastisch gesteigert werden konnte und sich in keinem Fall verschlechtert hat, siehe Tabelle 5.22.

5.3.2 Untersuchungsergebnisse

Untersucht wurden folgende dreizehn Hybridisierungsarten und Kombinationen:

1. **Voroptimierung**
   Voroptimierung eines Teils oder der gesamten der Startpopulation mit dem Rosenbrock-Verfahren (Ri) oder dem Complex-Algorithmus (Ci).

2. **Nachoptimierung**
   Nachoptimierung der GLEAM-Ergebnisse mit dem Rosenbrock- (NR) oder mit dem Complex-Algorithmus, wobei bei letzterem entweder jedes GLEAM-Ergebnis als ein Startpunkt in einem separaten Complex-Lauf bearbeitet wird (NC1S) oder alle Ergebnisse zur Bildung eines Startcomplexes für einen Nachoptimierungslauf herangezogen werden (NC1C).

3. **Direkte Integration ohne Voroptimierung**
   Lokale Verbesserung aller oder nur des besten Nachkommens einer Paarung mit unterschiedlichen Lamarckraten unter Verwendung des Rosenbrock- (GR) oder des Complex-Verfahrens (GC).

4. **Direkte Integration mit Voroptimierung**
   Kombination der Hybridisierungsarten Voroptimierung und direkte Integration, wobei die gesamte Startpopulation voroptimiert wird (GR,Ri und GC,Ci).

5. **Verzögerte direkte Integration ohne Voroptimierung**
   Verzögerte lokale Verbesserung aller oder nur des besten Nachkommens einer Paarung bei meist reiner Lamarckscher Evolution unter Verwendung des Rosenbrock- (GvR) oder des Complex-Verfahrens (GvC). Die Verzögerung wird durch den gleichen Steuerungsmechanismus wie bei der Nachoptimierung kontrolliert.

6. **Verzögerte direkte Integration mit Voroptimierung**
   Kombination der Hybridisierungsarten Voroptimierung und verzögerte direkte Integration, wobei die gesamte Startpopulation voroptimiert wird (GvR,Ri und GvC,Ci).

Die wichtigsten Ergebnisse, die auch auf zwei internationalen Konferenzen [Jak01b], darunter der *Parallel Problem Solving from Nature VII* [Jak02a], vorgestellt und in einer Sonderausgabe der *Evolutionary Computation* über Memetische Algorithmen [Jak03] veröffentlicht wurden, werden nachfolgend dargestellt[3]. Tabelle 5.23 und Abb. 5.170 fassen die Ergebnisse der Untersuchungen für alle Hybridisierungsarten zusammen. Dargestellt sind die jeweils erfolgreichsten Parametrierungen der untersuchten Hybridisierungsarten, ihrer Varianten und Kombinationen für sechs der acht ursprünglichen Benchmarkaufgaben, mit denen alle Hybridisierungsarten getestet wurden. Schwefel's Sphere wurde weggelassen, da GLEAM hier keinen Vergleichsmaßstab liefert. Auf die kollisionsfreie Roboterbahnplanung wurde verzichtet, da die geringen Verbesserungen der lokalen Verfahren in keinem Verhältnis zum verursachten

3. Darüberhinaus flossen Resultate in das BMBF-Verbundprojekt OMID ein [Jak01c, Jak02b].

Mehraufwand stehen. Das liegt in der Natur der Aufgabe begründet, bei der sowohl die Parametrierung als auch die Reihenfolge der Befehle für den Erfolg ausschlaggebend sind und die lokale Optimierung nur eines Teils der Parameter offenbar nicht für relevante Verbesserungen ausreicht. Die Testaufgaben wurden im Diagramm nach der erreichten maximalen Verbesserung (Faktor gegenüber dem besten GLEAM-Job) sortiert. Es ist deutlich zu erkennen, daß die Voroptimierung nur punktuelle Erfolge bei vier Testaufgaben aufweisen kann. Da sie bei den anderen Aufgaben mit Ausnahme der Ressourcenplanung zu keiner Verschlechterung geführt hat, erscheint es sinnvoll, über ihre Verwendung im Einzelfall der konkreten Anwendung zu entscheiden. Die Nachoptimierung hat nicht den erhofften Erfolg gebracht, obwohl sie bei fünf der acht Testaufgaben durchaus in der Lage war, die GLEAM-Ergebnisse relevant zu verbessern, ohne allerdings dabei eine Erfolgsrate von 100% erreichen zu können. Die direkte Rosenbrock-Integration liefert hingegen in allen untersuchten Fällen gute bis sehr gute Resultate, die bis zu einem Verbesserungsfaktor von 77 (Ressourcenplanung) reichen. Bei Verwendung des Complex- statt des Rosenbrock-Verfahrens können in den Fällen, in denen diese Kombination funktioniert, Verbesserungen bis zum Faktor 103 (Fletcher's Function) beobachtet werden. Die Verzögerung der direkten Rosenbrock-Integration bringt gegenüber der unverzögerten nochmal eine Verbesserung bis zum Faktor 1.4 (Designoptimierung mit Rosenbrock-Integration).

Hybride	Rastrigin	Foxholes	Design	Fraktal	Ressourcen	Fletcher
Ri	1.01	1.6	0	4.8		13.9
Ci	1.02	0	2.8	1.1		24.4
NR	0			0		
NC1S						
NC1C			1.7			
GR	1.18	0	1.4	6.4	77.4	35.7
GR,Ri	1.01	0	1.2	6.5	70.0	39.9
GC		0	5.6	0		103.3
GC,Ci		0	5.3	0		59.0
GvR	1.09	0	1.9	6.4	89.3	47.4
GvR,Ri	1.07	1.0	0	5.6	71.6	39.0
GvC		0	5.8	0		104.2
GvC,Ci		0	5.7	0		76.9

Tab. 5.23: Gesamtvergleich der Aufwandsverbesserungen gegenüber GLEAM für sechs der acht Testaufgaben. Leere Felder kennzeichnen Hybridisierungsarten ohne ausreichenden Erfolg (Erfolgsrate < 100%) und Felder mit Null stehen für eine zuverlässige Lösung der Aufgabe aber bei höherem Aufwand.

Wenig gebracht hat die Hybridisierung dagegen generell bei den beiden mathematischen Benchmarkaufgaben Shekel's Foxholes und der verallgemeinerten Rastrigin Funktion, die in ihrer ursprünglichen Form „einfach genug" sind, um bereits durch GLEAM so effizient gelöst zu werden, daß eine weitere Verbesserung kaum möglich ist (Verbesserungsfaktoren von 1.6 und 1.18). Das wird unter anderem daran deutlich, daß GLEAM noch bei der extrem kleinen Populationsgröße von fünf sicher konvergiert. Dieses Teilergebnis ist in gewisser Hinsicht überraschend, da beide Aufgaben als schwierig für die Standard-ES gelten. Bei einer Drehung

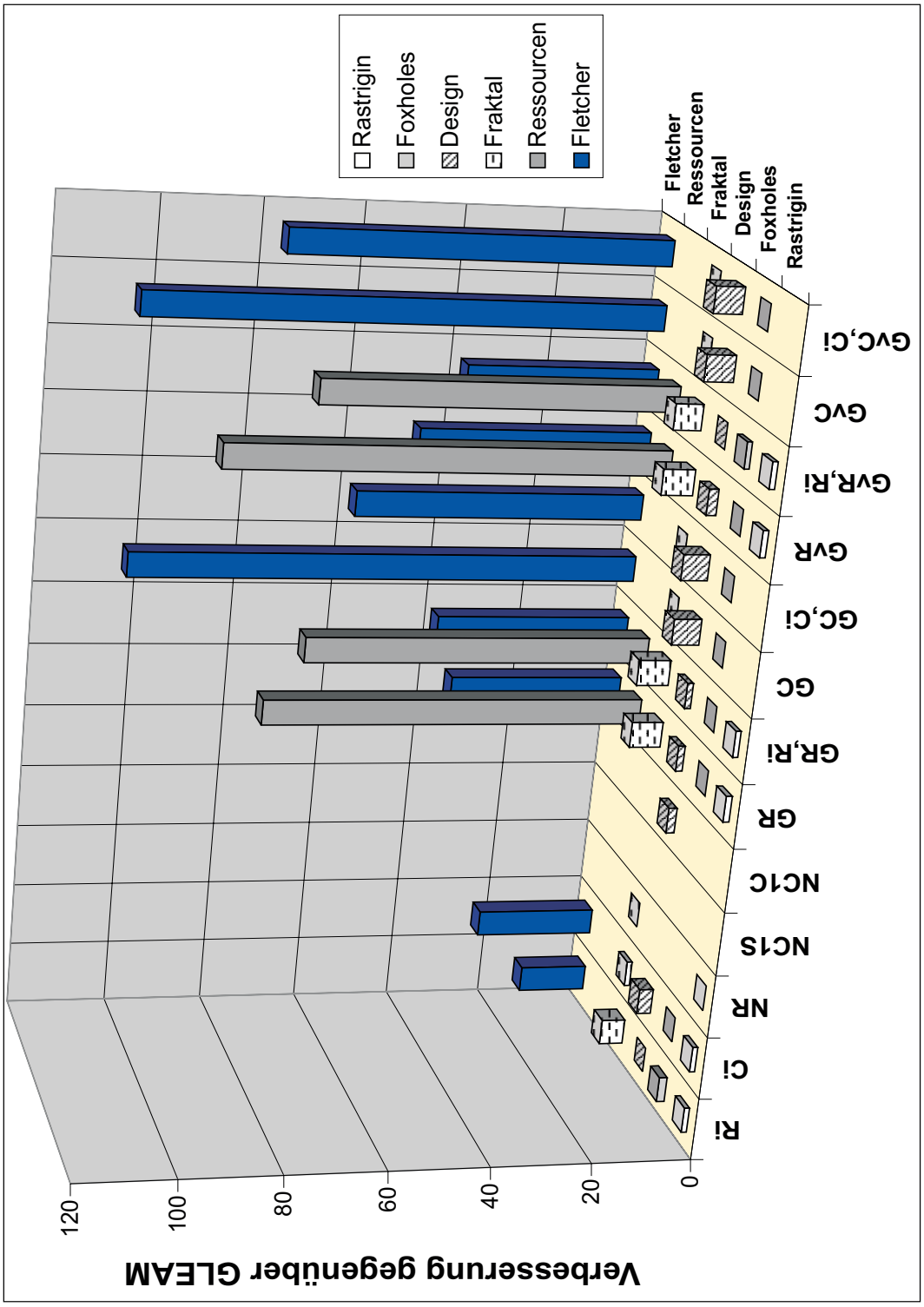


Abb. 5.170: Gesamtvergleich der Aufwandsverbesserung gegenüber GLEAM für sechs der acht Testaufgaben, siehe auch Tabelle 5.23. Leere Felder kennzeichnen Hybridisierungsarten ohne ausreichenden Erfolg (Erfolgsrate < 100%) und Felder ohne Höhe stehen für eine zuverlässige Lösung der Aufgabe aber bei höherem Aufwand.

der beiden Funktionen im Raum werden sie jedoch, wie in Abschnitt 5.2.3 beschrieben, auch für GLEAM „schwierig“. Abb. 5.171 vergleicht die Ergebnisse der rotierten Funktionen und der anderen Benchmarkaufgaben von Abb. 5.170 für die verzögerte und unverzögerte direkte Integration beider LSV. Die erreichten Verbesserungen der beiden gedrehten Funktionen fügen sich in das Bild der vier anderen Benchmarkaufgaben ein. Der wesentliche Unterschied besteht darin, daß bei den rotierten Funktionen der Rosenbrock-Algorithmus besser abschneidet als die Hybridisierungen mit dem Complex-Verfahren und daß die verzögerten Varianten schlechtere Resultate liefern. Dies bestätigt die Aussage von Abschnitt 5.2.2.4, wonach die Frage, ob und welche Verzögerung der direkten Integration vorteilhaft ist, nur aufgabenspezifisch beantwortet werden kann.

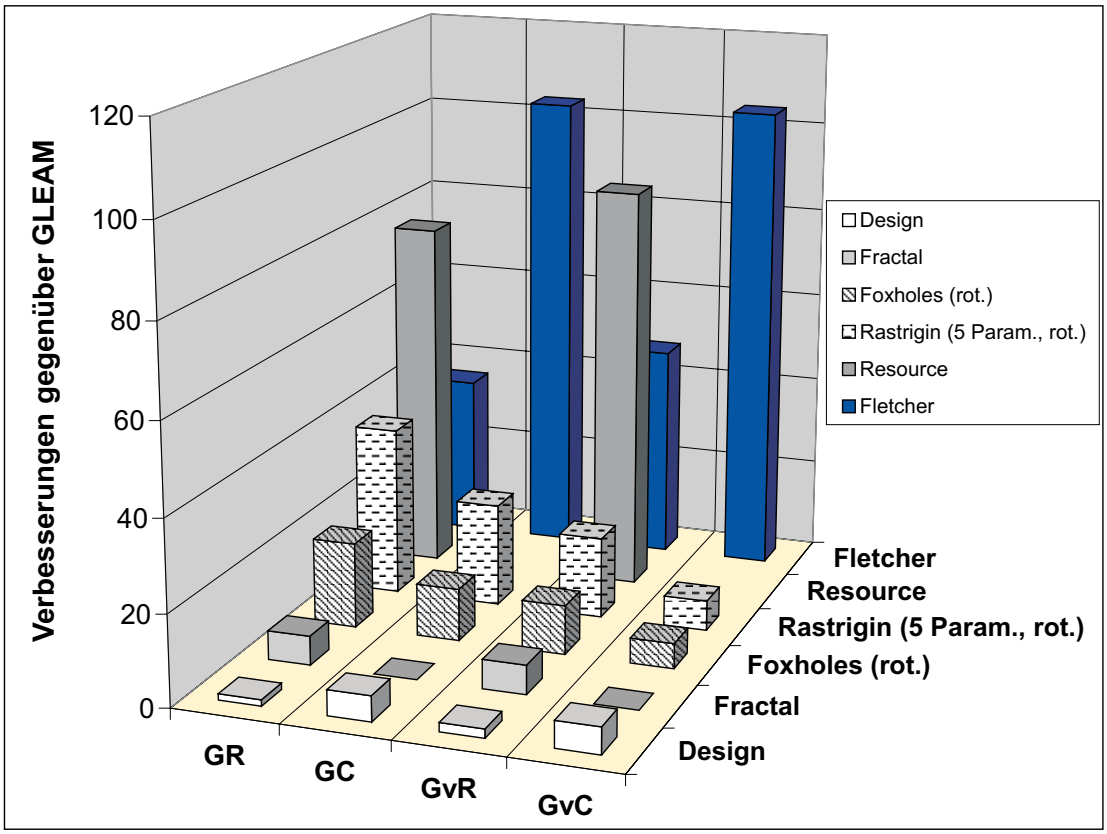


Abb. 5.171: Vergleich der (verzögerten) direkten Integration beider LSV für die rotierten Benchmarkfunktionen und die restlichen Benchmarkaufgaben von Abb. 5.170.

Zusammenfassend kann festgestellt werden, daß die (verzögerte) direkte Integration sowohl die Konvergenzsicherheit als auch die Konvergenzgeschwindigkeit erhöht. Das Ausmaß der Leistungssteigerung ist anwendungsabhängig und kann im Einzelfall erheblich sein.

5.3.3 Anwendungsempfehlung

Zur Bestimmung des praktischen Nutzwertes der vorliegenden Untersuchung sind zwei Fälle zu unterscheiden: Die Einzeloptimierung unterschiedlicher Aufgabenstellungen und die immer wiederkehrende Optimierung von Varianten einer Aufgabe. Nur im letzteren Fall lohnt eine genauere Bestimmung der günstigsten Hybridisierungsart und ihrer Parametrierung, sofern der zeitliche Aufwand dazu vertretbar ist.

Im Falle der Einzeloptimierung kann folgendes Vorgehen empfohlen werden (vgl. Abschnitte 5.2.2.3 und 5.2.2.4):

1. In einer Voruntersuchung mit dem Rosenbrock-Algorithmus wird die beste Rosenbrock-Präzision ausgehend von einem vertretbar großen Satz von Startwerten ermittelt. Sollte dabei immer das gleiche Optimum gefunden werden, so wird die sich daraus ergebende Hypothese der Unimodalität durch weitere Läufe geprüft. Bestätigt sich die Annahme, ist das Optimierungsproblem gelöst.
2. Mit der so ermittelten Präzision wird das Rosenbrock-Verfahren parametriert. Die Hybridisierung wird auf verzögerte direkte Rosenbrock-Integration bei Verbesserung des besten Nachkommens und reiner Lamarckscher Evolution eingestellt. Die Populationsgröße wird je nach geschätzter Komplexität des Suchraums auf fünf, zehn oder vorsichtshalber auf zwanzig festgesetzt. Da sich die Voroptimierung bei der unverzögerten Integration meist als erfolgreich erwiesen hat, sollten die Ergebnisse vom ersten Schritt unter Ausschluß gleicher Genotypen zur Bildung eines Anteils von maximal 20% der Startpopulation herangezogen und der Rest zufällig bestimmt werden. Der 20%-Anteil ist durch die Ergebnisse der reinen Voroptimierung motiviert. Als Nischenparametrierung wird $\varepsilon = 0.001$, $\varepsilon_{Pop} = 0.003$ (P3) und die maximale Nischenanzahl $N_{max} = 2$ gewählt.

Bei der Varianten-Optimierung wird unter Einbeziehung des Complex-Algorithmus zunächst genauso vorgegangen und dann mit der zuerst nicht genutzten Populationsgröße verglichen, wobei soviel Läufe pro Parametrierung wie vertretbar anzusetzen sind. Je nachdem, ob dabei die Complex- oder die Rosenbrock-Integration besser abschneidet, erfolgt die Auswahl des geeigneteren lokalen Verfahrens. Sollte dabei ein Trend zur höheren Populationsgröße feststellbar sein, wird durch schrittweise Vergrößerung der Population geprüft, wie lange sich dieser fortsetzt. Je nach Aufwand und verfügbaren Ressourcen können abschließend noch Variationen der Nischenparametrierung getestet werden.

6. Neues Konzept einer adaptiven Steuerung für die direkte Integration

Wie in Abschnitt 5.2.2 beschrieben, ist für die (verzögerte) direkte Integration als erfolgreichste Hybridisierung aus den Ergebnissen leider keine allgemeingültige Parametrierung ableitbar. Auch die Frage, welches der lokalen Verfahren zu verwenden sei, ist nur anwendungsspezifisch beantwortbar: Das Rosenbrock-Verfahren funktionierte zwar in allen Anwendungsfällen, wurde aber vom Complex-Algorithmus in den Fällen, in denen er anwendbar ist, übertroffen. Die positiven Ergebnisse der verzögerten direkten Integration legen den Gedanken nahe, daß bei multimodalen Problemstellungen am Anfang der Suche auf die genaue Bestimmung der lokalen Suboptima verzichtet werden kann. Es ist also gerechtfertigt und aus Aufwandsgründen wünschenswert, die lokale Suche anfänglich nur recht grob zu betreiben und erst im Verlaufe der Evolution zu präzisieren. Schließlich genügt die ungefähre Bestimmung eines lokalen Optimums, solange sie genau genug ist, um zwischen den lokalen Optima korrekt differenzieren zu können. Da mit der Obergrenze für die Evaluationen und dem Präzisionsparameter des Rosenbrock-Verfahrens entsprechende Einstellmöglichkeiten zur Verfügung stehen, bietet es sich an, sie zur dynamischen Steuerung der Aufteilung der Rechnerressourcen zwischen lokaler und globaler Suche zu nutzen. Anstelle der von Zitzler, Teich und Bhattacharyya [Zit00] bei einem vergleichbaren Ansatz benutzten vordefinierten Verteilungen sollte eine neue adaptive Steuerung verwendet werden, da weder das Optimum noch der Weg dahin bekannt sind. Ihre ermutigenden Ergebnisse zeigen allerdings, daß der Gedanke es wert ist, weiterverfolgt zu werden.

Die vorgeschlagene adaptive Steuerung beruht auf dem beobachteten *Erfolg* und den dazu notwenigen *Kosten* in Form von Fitnessberechnungen. Sie wird zunächst am Beispiel der Aufteilung zwischen den beiden lokalen Verfahren beschrieben. Zunächst ist die Wahrscheinlichkeit für die Anwendung beider Verfahren gleich. Die Anzahl ihrer Anwendungen wird gezählt und der jeweils erreichte Fitnesszuwachs *fz* wird zusammen mit den dazu benötigten Evaluationen *eval* aufsummiert. Aus den zuvor beschriebenen Experimenten kann entnommen werden, daß etwa 45 Paarungen bei der direkten und 100 bis 200 bei der verzögerten direkten Integration benötigt wurden. Um eine häufige Überprüfung bei hinreichender Datenmenge zu erreichen, werden die Ausführungswahrscheinlichkeiten der lokalen Verfahren neu justiert, wenn entweder jedes Verfahren mindestens dreimal benutzt wurde oder nach spätestens 15 Paarungen. Die neue Relation zwischen den beiden lokalen Verfahren berechnet sich wie folgt:

$$\frac{\sum fz_{i,\,compl}}{\sum eval_{i,\,compl}} : \frac{\sum fz_{j,\,rosen}}{\sum eval_{j,\,rosen}}$$

Wenn das Verhältnis für ein lokales Verfahren bei drei aufeinanderfolgenden Neujustierungen unter 1:10 sinkt, wird es nicht weiter benutzt. Die Summen werden für die nächste Berechnung auf Null gesetzt, um eine schnellere Adaption zu erreichen.

Der Ansatz kann leicht, wie nachstehend beschrieben, auf weitere Suchverfahren oder Parameter erweitert werden. Die fünf Rosenbrock-Präzisionen der Experimente werden auf neun Werte zwischen 10^{-1} und 10^{-9} erweitert und für das Evaluationslimit können z.B. zehn Werte im Bereich zwischen 100 und 2000 benutzt werden: 100, 200, 350, 500, 750, 1000, 1250, 1500, 1750, 2000. Nur drei der Limits können gleichzeitig aktiv sein und anfänglich haben die jeweils drei niedrigsten die gleiche Wahrscheinlichkeit. Wenn das niedrigste oder höchste Limit mehr als 50% Wahrscheinlichkeit erhält und es daneben noch ein unbenutztes Limit gibt, wird es aktiviert und das Limit am anderen Ende gestrichen. Seine Wahrscheinlichkeit wird dem jeweiligen Nachbarn zugeschlagen. Das neue Limit erhält initial jeweils 20% von den beiden anderen. Damit bewegen sich die drei zusammenhängenden Limits innerhalb des Bereichs möglicher Limits gesteuert durch ihre Leistung, gemessen in Fitnesszuwachs und verursachtem Aufwand. Abb. 6.1 veranschaulicht das an einem Beispiel: Oben ist die Ausgangssituation dargestellt, bei der eine Erhöhung der Limits ansteht, da das rechte Limit die 50%-Marke überschritten hat. Das niedrigste Limit, hier Limit 3, wird gestrichen und sein Wahrscheinlichkeitswert dem Nachbar zugeschlagen (mittlere Situation). Anschließend geben die beiden alten Limits jeweils 20% ihrer Wahrscheinlichkeitswerte an das neue Limit ab, und die untere Situation entsteht als Ergebnis der Parameterneujustierung. Im Ergebnis wurde der Bereich aktiver Limits nach rechts verschoben und die damit verbundene Parametrierung erhöht.

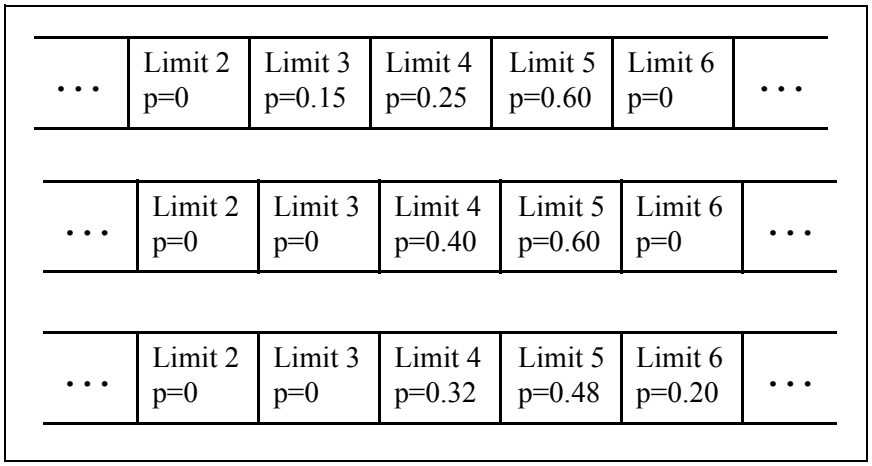


Abb. 6.1: Wirkungsweise der adaptiven Parameteranpassung am Beispiel der Streichung eines niedrigen Limits und Einführung eines höheren.

Der aktuelle Parameter wird dann entsprechend den Wahrscheinlichkeiten der drei aktiven Limits ausgewählt. Wenn die aktiven Limits die untere oder obere Grenze erreichen, dürfen die Wahrscheinlichkeiten nie unter einen Mindestwert von z.B. 5% sinken, damit immer die Möglichkeit besteht, eine einmal gefundene Parameter-Adaption wieder in die entgegengesetzte Richtung zu verändern.

Um am Ende eines Laufs die Qualität der lokalen Suche gezielt zu erhöhen, können die Mechanismen der Nischenermittlung genutzt werden. Die Parametrierungen der Nachoptimierung könnten z.B. dazu dienen, den geeigneten Zeitpunkt für eine Änderung der Grenzwerte für die Bewegungen der Limits zu bestimmen: Eine Verkleinerung von Präzision und Evaluationslimit kann z.B. erst bei 70% Wahrscheinlichkeit des kleinsten Limits erfolgen und eine Vergrößerung schon bei 30%.

Das Konzept gestattet eine adaptive Anpassung der Auswahl des lokalen Suchverfahrens und der Intensität seiner Suche. Es ermöglicht damit auch eine dynamische Verteilung der Rechenkapazitäten zwischen lokaler und globaler Suche entsprechend dem erreichten Fitnessgewinn in Relation zu den Kosten. Es stellt einen vielversprechenden Ansatz zur Lösung der zuvor behandelten Problematik einer problemangepaßten Parametrierung der Hybridisierung dar. Abb. 6.2 zeigt seine wesentlichen Komponenten im Vergleich zum Überblicksbild 3.2 der im Rahmen dieser Arbeit behandelten Hybridisierungsarten. Der in sich geschlossene Charakter der vorgeschlagenen adaptiven direkten Integration ist deutlich zu erkennen.

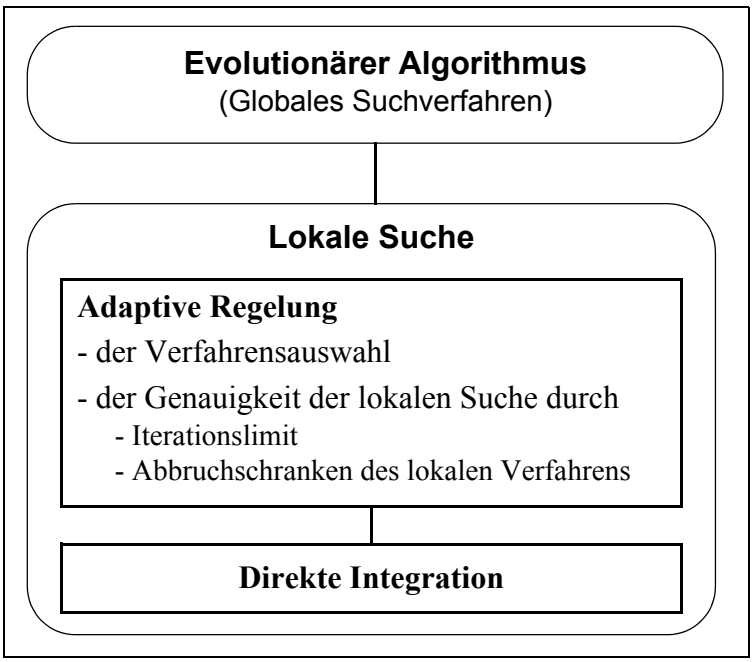


Abb. 6.2: Struktur der neuen adaptiven direkten Integration.

Grundsätzlich kann der Ansatz auch auf weitere Parameter übertragen werden. Allerdings lassen die gemachten Erfahrungen den Schluß zu, daß es sich bei den zuvor genannten Parametern um diejenigen mit der größten Relevanz handelt. Außerdem lassen sie sich problemlos während eines Evolutionslaufs verändern und sind damit für eine adaptive Anpassung gut geeignet. Die Ergebnisse legen den Schluß nahe, daß bei Aufgabenstellungen, bei denen der Anteil der Parameteroptimierung zumindest dominiert, die Frage nach Lamarck- oder Baldwin-Evolution meist zu Gunsten der Lamarckschen Evolution beantwortet werden kann und die Begrenzung auf eine lokale Optimierung nur des besten statt aller Nachkommen sinnvoll ist. Generell erlaubt aber das vorgestellte Konzept auch diese Parameter adaptiv zu kontrollieren. Dazu kann die Lamarckrate in geeignete Wertegruppen eingeteilt werden, z.B. 100, 75, 50, 25, 10, 5 und 0 Prozent. Eine ähnliche Einteilung kann zur Bestimmung der Wahrscheinlichkeit dienen, mit der alle oder nur der beste Nachkomme lokal zu verbessern ist. Schwieriger ist dagegen die Wahl einer geeigneten Populationsgröße, die sich aber auch kaum adaptiv regeln läßt. Hier können nur weitere Untersuchungen helfen, aus denen sich hoffentlich grobe Regeln ableiten lassen.

7. Zusammenfassung und Ausblick

Bei der praktischen Anwendung Evolutionärer Algorithmen wird das Problem, daß sie in der Nähe des Optimums meist langsam konvergieren, häufig durch eine Hybridisierung mit lokalen und schnelleren Verfahren gelöst. Da die dabei verwendeten lokalen Suchverfahren in der Regel aufgabenspezifischer Natur sind, wird aus dem allgemein anwendbaren EA ein anwendungsspezifisches Werkzeug. Den vielen, meist problemspezifischen Untersuchungen zur zweckmäßigen Verfahrenskombination auf der Ebene geeigneter Parametrierungen und anderer Details der konkreten Hybridisierung steht ein Mangel an Analysen auf globaler Ebene gegenüber: Wie werden lokale und globale Suchverfahren passend kombiniert und die zur Verfügung stehende Rechenzeit so aufgeteilt, daß das hybride Verfahren zuverlässiger und schneller eine Lösung findet als die beteiligten Algorithmen allein? [Gol99]. Auch ist die Frage angesichts sich widersprechender Untersuchungsergebnisse ungeklärt, ob und wenn ja mit welcher Häufigkeit eine genotypische Anpassung an die Lösung des lokalen Suchverfahrens erfolgen soll.

Zur Lösung dieser offenen Probleme bei der Anwendung Evolutionärer Algorithmen liefert die vorliegende Arbeit einen Beitrag durch die Bearbeitung folgender Teilaufgaben:

1. Eine neue Methodik zur Kombination von lokalen Suchverfahren mit Evolutionären Algorithmen unter Wahrung der allgemeinen Anwendbarkeit des resultierenden hybriden Verfahrens wurde erarbeitet.
2. Für die beispielhafte praktische Erprobung der neuen Methodik wurden als allgemein anwendbare lokale Verfahren der Rosenbrock-Algorithmus und das Complex-Verfahren ausgewählt. GLEAM wurde als repräsentativer Vertreter der Evolutionären Algorithmen für die Untersuchungen benutzt, da das Verfahren Elemente der ES und der GA in sich vereint und somit die bestmöglichen Voraussetzungen für die Übertragbarkeit der gefundenen Resultate auf andere Evolutionäre Algorithmen bietet.
3. Ein neues Steuerungsverfahren zur Aufteilung der Rechenzeit zwischen Evolutionärem Algorithmus und lokalem Suchverfahren wurde entwickelt. Dabei wurde Wert darauf gelegt, daß die neue Steuerung basierend auf der Bildung von Nischen einander ähnlicher Individuen innerhalb einer Population auch bei anderen EA als dem ausgewählten anwendbar ist.
4. Für empirische Untersuchungen der neuen Methode wurden repräsentative Benchmarkaufgaben ausgewählt, die schnell genug sind, um statistische Untersuchungen zu ermöglichen. Sie decken die Bereiche der reinen Parameteroptimierung, der kombinatorischen und der gemischt-ganzzahligen Optimierung sowie der Behandlung dynamischer Parametersätze ab.
5. Es wurde ein Konzept zur Integration der vorgeschlagenen lokalen Suchverfahren entsprechend der neuen Methode in das vorhandene Softwaresystem von GLEAM entwikkelt und implementiert.
6. Zur Überprüfung der Ziele einer beschleunigten Konvergenz unter Beibehaltung der Konvergenzsicherheit wurden umfangreiche experimentelle Untersuchungen an den zuvor ausgewählten Benchmarkaufgaben durchgeführt und ausgewertet.

Untersucht wurden folgende Hybridisierungsarten und Kombinationen:

1. Voroptimierung
   Voroptimierung eines Teils oder der gesamten Startpopulation mit dem Rosenbrock-Verfahren oder dem Complex-Algorithmus.

2. Nachoptimierung
   Nachoptimierung der GLEAM-Ergebnisse mit dem Rosenbrock-Verfahren oder dem Complex-Algorithmus, wobei bei letzterem entweder jedes GLEAM-Ergebnis als ein Startpunkt in einem separaten Complex-Lauf bearbeitet wird oder alle Ergebnisse zur Bildung eines Startcomplexes für einen Nachoptimierungs-Lauf herangezogen werden.

3. Direkte Integration mit und ohne Voroptimierung
   Lokale Verbesserung aller oder nur des besten Nachkommens einer Paarung mit unterschiedlichen Lamarckraten unter Verwendung des Rosenbrock- oder des Complex-Verfahrens. Eine weitere Variation ist die Voroptimierung der Startpopulation durch das jeweilige lokale Verfahren.

4. Verzögerte direkte Integration mit und ohne Voroptimierung
   Verzögerte lokale Verbesserung aller oder nur des besten Nachkommens einer Paarung bei meist reiner Lamarckscher Evolution unter Verwendung des Rosenbrock- oder des Complex-Verfahrens. Die Verzögerung wird durch den gleichen Steuerungsmechanismus wie bei der Nachoptimierung kontrolliert. Eine weitere Variation ist die Voroptimierung der Startpopulation durch das jeweilige lokale Verfahren.

Eine relevante Steigerung der Konvergenzgeschwindigkeit konnte bei fünf der acht Testaufgaben erreicht werden, da sich drei aus unterschiedlichen Gründen als schlecht geeignet für eine Verbesserung durch die Hybridisierung erwiesen haben. Zum einen sind das die mathematischen Benchmarkaufgaben Shekel's Foxholes und die verallgemeinerte Rastrigin Funktion, die in ihrer ursprünglichen Form „einfach genug" sind, um bereits durch GLEAM so effizient gelöst zu werden, daß eine weitere Verbesserung kaum möglich ist (Verbesserungsfaktoren von 1.6 und 1.18). Das wird unter anderem daran deutlich, daß GLEAM noch bei der extrem kleinen Populationsgröße von fünf sicher konvergiert. Dieses Teilergebnis ist in gewisser Hinsicht überraschend, da beide Aufgaben als schwierig für die Standard-ES gelten. Bei einer Drehung der beiden Funktionen im Raum werden sie jedoch auch für GLEAM „schwierig". Es konnte gezeigt werden, daß die direkte Integration auch bei den gedrehten Funktionen in der Lage ist, mit Verbesserungsfaktoren gegenüber GLEAM von 18.8 bei der Foxhole- und 37.4 bei der Rastrigin-Funktion ein hervorragendes Ergebnis zu erzielen. Zum anderen handelt es sich dabei um die kollisionsfreie Roboterbahnplanung, bei der sowohl die Parametrierung als auch die Reihenfolge der Befehle für den Erfolg relevant ist. Die geringen Verbesserungen, die die benutzten lokalen Verfahren zur Parameteroptimierung hier erreichen konnten, stehen in keinem Verhältnis zum verursachten Mehraufwand.

Die wichtigsten Ergebnisse der Arbeit sind:

1. Die Konvergenzsicherheit konnte durch die Hybridisierung nicht nur beibehalten, sondern sogar verbessert werden.
   Dies wird bei der Testaufgabe Schwefel's Sphere deutlich, die GLEAM alleine nicht lösen konnte.

2. Die Konvergenzgeschwindigkeit konnte zum Teil erheblich verbessert werden.
   Wie in Abb. 5.170 dargestellt, konnte bei fünf der acht Testaufgaben eine Steigerung um den Faktor sechs bis 104 erreicht werden. Werden die beiden „zu einfachen“ Testfunktionen gedreht, so zeigt Abb. 5.171, daß auch hier erhebliche Steigerungsraten möglich sind.

3. Als beste Hybridisierungsart hat sich die direkte Integration in ihrer verzögerten und unverzögerten Variante erwiesen.
   Dabei arbeitet die Integration des Rosenbrock-Verfahrens zuverlässiger als die des Complex-Algorithmus, da sie bei allen sechs hier noch betrachteten Testaufgaben und den beiden gedrehten Funktionen erfolgreich anwendbar ist. Wenn dagegen die Complex-Integration funktioniert, liefert sie in zwei von vier Fällen bessere Ergebnisse als das Rosenbrock-Verfahren (siehe Abb. 5.170 und 5.171).

4. Die Lamarcksche Evolution schneidet fast immer am besten ab.
   Bis auf die eine Ausnahme der direkten Rosenbrock-Integration bei der Testaufgabe Designoptimierung lieferte die reine Lamarcksche Evolution die besseren Ergebnisse im Vergleich zur gemischten oder reinen Baldwin-Evolution. Dieses Ergebnis steht im Widerspruch zu den Empfehlungen von Goldberg und Voessner [Gol99] sowie Orvosh und Davis [Orv93], bestätigt aber die Ergebnisse von Whitley et al. [Whi94].

5. Es gibt keine allgemeingültige „beste Parametrierung“.
   Bedauerlicherweise erwies sich die Parametrierung vor allem der verzögerten direkten Integration als aufgabenspezifisch. Immerhin kann die Lamarcksche Evolution bei Verbesserung nur des besten Nachkommens einer Paarung empfohlen werden.

6. Grenzen für eine erfolgreiche Hybridisierung
   Von einer Hybridisierung mit lokalen Verfahren zur Parameteroptimierung sollte abgesehen werden, wenn die Aufgabe bereits durch ein lokales oder ein evolutionäres Verfahren effizient gelöst werden kann oder wenn der kombinatorische Anteil an der Lösungsfindung überwiegt.

7. Anwendungsempfehlung
   Eine generelle Vorgehensweise zur praktischen Umsetzung der Resultate wurde vorgeschlagen.

8. Adaptive direkte Integration
   Die Ergebnisse der empirischen Untersuchungen führten zur Formulierung eines neuen Konzepts für eine adaptive direkte Integration basierend auf Erfolg (Fitnesszugewinn) und Kosten (Evaluationen).

Mit diesen Ergebnissen konnte gezeigt werden, daß die Kombination eines repräsentativen Evolutionären Algorithmus mit allgemein anwendbaren lokalen Suchverfahren wie dem Rosenbrock- und dem Complex-Algorithmus ein hybrides Verfahren ergibt, das die Vorteile beider Algorithmenklassen unter weitgehender Vermeidung der jeweiligen Nachteile in sich vereint. Sowohl die Konvergenzsicherheit als auch die Konvergenzgeschwindigkeit konnten in erheblichem Maße gesteigert werden.

Als ein Nebenergebnis konnte dargelegt werden, daß vermeintlich zu einfache Testfunktionen durch Drehung „schwieriger“ gemacht werden können. Dabei geht es im wesentlichen um die Beseitigung von parallel zu den Koordinatenachsen ausgerichteten Regelmäßigkeiten der Funktionen, die die Suche für EA zu leicht machen.

Die vorliegende Arbeit mußte sich auf die Untersuchung von acht Testaufgaben beschränken, was dem dafür benötigten Gesamtaufwand von über neun CPU-Jahren geschuldet ist. Die hier gefundenen Ergebnisse sollten durch andere Testaufgaben weiter verifiziert werden, wobei solche Aufgaben zu bevorzugen sind, die einen praktischen Hintergrund haben und komplex genug sind, um Raum für Verbesserung durch die Hybridisierung zu lassen. Andererseits müssen sie schnell genug evaluierbar sein, damit statistische Untersuchungen möglich sind. Damit kann auch die Frage der geeigneten Parametrierung der unverzögerten und vor allem der verzögerten direkten Integration weiter verfolgt werden. Auch muß das in Kapitel 6 vorgestellte Konzept zur adaptiven Steuerung der Aufteilung der Rechenkapazitäten zwischen lokaler und globaler Suche und zwischen den lokalen Verfahren bei der direkten Integration erprobt werden.

Bei rein kombinatorischen Aufgaben wurden mit entsprechenden lokalen Algorithmen, wie zum Beispiel dem 2-Opt-Verfahren, beachtliche Erfolge erzielt. Nachdem bei der Roboterbahnplanung die Hybridisierung mit lokaler Parameteroptimierung nicht erfolgreich war, ist die Untersuchung der Frage von Interesse, ob eine kombinatorische lokale Suche eventuell in Verbindung mit Verfahren zur Parameterverbesserung zum Ziel führt.

8. Literatur

[Aar85] E. H. L. Aarts, P. J. M. van Laarhoven:
A New Polynomial Time Cooling Schedule.
Proc. IEEE Int. Conf. on Computer-Aided Design, Santa Clara, S.206-208, 1985.

[Aar89] E. H. L. Aarts, J. H. M. Korst:
Simulated Annealing and Boltzmann Machines.
John Wiley & Sons, Chichester, 1989.

[Amd67] G. Amdahl:
Validity of the single-processor approach to achieving large scale computing capabilities.
In: AFIPS Conference Proceedings, Vol. 30, AFIPS Press, Reston, S.483-485, 1967.

[Bäc91] T. Bäck, F. Hoffmeister, H.-P. Schwefel:
A Survey of Evolution Strategies.
In: R. K. Belew, L. B. Booker: Proc. of the 4th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.2-9, 1991.

[Bäc92] T. Bäck:
GENEsYs 1.0
ftp://lumpi.informatik.uni-dortmund.de/pub/GA, 1992.

[Bäc93a] T. Bäck:
Optimal Mutation Rates in Genetic Search.
In: S.Forrest (ed): Proc. of the 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.2-8, 1993.

[Bäc93b] T. Bäck, H.-P. Schwefel:
An Overview of Evolutionary Algorithms for Parameter Optimization
Evolutionary Computation 1, Heft 1, S.1-23, 1993.

[Bäc94] T. Bäck:
Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms.
In: Conf. Proc. of the 1st IEEE Int. Conf. on Evolutionary Computation (ICEC'94), IEEE Press, S.57-62, 1994.

[Bäc97] T. Bäck (ed.):
Genetic Algorithms.
Proc. of the 7th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, 1997.

[Bäc98] T. Bäck, D. B. Fogel, Z. Michalewicz (eds.):
Handbook of Evolutionary Computation.
IOP Publishing, Bristol and Oxford University Press, New York, 1998.

[Bak85] J. E. Baker:
Adaptive Selection Methods for Genetic Algorithms.
In: J. J. Grefenstette (ed): Proc. of an Int. Conf. on Genetic Algorithms and Their Applications, Hillsdale/NJ, Lawrence Erlbaum, S.101-111, 1985.

[Ban99]: W. Banzhaf, J. Daida, A. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, R. E. Smith (eds.):
GECCO'99 - Proc. of the Genetic And Evolutionary Computation Conference.
Morgan Kaufmann, San Francisco, CA, 1999.

[Bar68] Y. Bard:
On a Numerical Instability of Davidon-like Methods.
Math. Comp. 22, S.665-666, 1968.

[Bel72] E. J. Beltrami, J. P. Indusi:
An Adaptive Random Search Algorithm for Constrained Minimization.
IEEE Trans. C-21, S.1004-1008, 1972.

[Bel91] R. K. Belew, L. B. Booker (eds.):
Genetic Algorithms.
Proc. of the 4th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991.

[Ber80] M. V. Berry, Z. V. Lewis:
On the Weierstrass-Mandelbrot Fractal Function.
Proc. of Royal Society London, A(370), S.459-484, 1980.

[Blu90] C. Blume:
GLEAM - A System for Simulated "Intuitive Learning".
In: H.-P. Schwefel, R. Männer (eds.): Proc. of PPSN I, LNCS 496, Springer-Verlag, S.48-54, 1991.

[Blu93a] C. Blume, W. Jakob:
Produktionsplanung und -optimierung durch Simulation, Genetische Algorithmen und Parallelisierung.
Tagungsband des 8. Symposiums „Simulationstechnik" der Arbeitsgemeinschaft Simulation in der GI (ASIM), Gesellschaft für Informatik, Fachausschuß 4.5, 1993.

[Blu93b] C. Blume, W. Jakob:
Verbesserte Planung und Optimierung mit Hilfe eines erweiterten genetischen Algorithmus.
Tagungsband des Transputer-Anwender-Treffen (TAT'93), RWTH Aachen, 1993.

[Blu93c] C. Blume, W. Jakob:
Closing the Optimization Gap in Production by Genetic Algorithms.
In: H.-J. Zimmermann (Hrsg.): Proc. of 1st European Congress on Fuzzy and Intelligent Techniques (Eufit), Verlag Mainz, Wissenschaftsverlag, 1993.

[Blu94a] C. Blume, M. Gerbe:
Deutliche Senkung der Produktionskosten durch Optimierung des Ressourceneinsatzes.
Automatisierungstechnische Praxis (atp) 36, Heft 5/94, Oldenbourg, München, S.25-29, 1994.

[Blu94b] C. Blume, W. Jakob, S. Krisch:
Robot Trajectory Planning with Collision Avoidance Using Genetic Algorithms and Simulation.
Proc. 25th Int. Symposium on Industrial Robots (ISIR), S.169-175, 1994.

[Blu94c] C. Blume, W. Jakob, J. Kaltwasser:
Kosten senken durch verbesserte Nutzung der Ressourcen.
Tagungsband des 9. Symposiums „Simulationstechnik“ der Arbeitsgemeinschaft Simulation in der GI (ASIM), Gesellschaft für Informatik, Fachausschuß 4.5, S.643-648, 1994.

[Blu94d] C. Blume, W. Jakob:
Cutting Down Production Costs by a New Optimization Method.
Proc. of the Japan - U.S.A. Symposium on Flexible Automation, ASME, 1994.

[Blu97] C. Blume:
Automatic Generation of Collision Free Moves for the ABB Industrial Robot Control.
In: L. C. Jain (ed.): Proc. 1st Int. Conf. on Knowledge-Based Intelligent Electronic Systems, Adelaide, S.672-683, 1997.

[Blu98] C. Blume:
Planung kollisionsfreier Bewegungen für Industrieroboter.
In: S. Hafner (Hrsg.): Industrielle Anwendungen evolutionärer Algorithmen. Oldenburg Verlag, München, S.45-56, 1998.

[Blu00a] C. Blume:
Optimized Collision Free Robot Move Statement Generation by the Evolutionary Software GLEAM.
In: S. Cagnoni et al. (eds.): Real-World Applications of Evolutionary Computing, Proc. of EvoWorkshops 2000: EvoIASP, EvoSCONDI, EvoTel, EvoSTIM, EvoRob and EvoFlight, Springer-Verlag, S.327-338, 2000.

[Blu00b] C. Blume:
Optimization in Concrete Precasting Plants by Evolutionary Computation.
In: D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, H.-G. Beyer (eds.): Genetic and Evolutionary Computation Conference (GECCO 2000), Vol. Late Breaking Papers, Morgan Kaufmann, San Francisco, CA, S.43-50, 2000.

[Blu02] C. Blume, W. Jakob:
GLEAM - an Evolutionary Algorithm for Planning and Control Based on Evolution Strategy.
In: E. Cantú-Paz (ed.): GECCO 2002, Vol. Late-Breaking Papers, L. Livermore National Laboratory. S.31-38, 2002.

[Bor78] J. Born:
Evolutionsstrategien zur numerischen Lösung von Adaptionsaufgaben.
Dissertation, Humboldt-Universität, Berlin, 1978.

[Bor83] J. Born, K. Bellmann:
Numerical Adaptation of Parameters in Simulation Models Using Evolution Strategies.
In: K. Bellmann (Hrsg.): Molecular Genetic Information Systems. Modelling and Simulation, Akademie-Verlag, Berlin, S.291-320, 1983.

[Bor92] J. Born, H.-M. Voigt, I. Santibánez-Koref:
Alternative Evolution Strategies to Global Optimization.
In: R. Männer, B. Manderick (eds.): Conf. Proc. PPSN II, North-Holland, Amsterdam, S.187-195, 1992.

[Box65] M. J. Box:
A New Method of Constrained Optimization and a Comparison with Other Methods.
Comp. Journal 8, S.42-52, 1965.

[Bre73] R. P. Brent:
Algorithms for Minimization without Derivatives.
Prentice-Hall, Englewood Cliffs, N.J., 1973.

[Bri80] A. Brindle:
Genetic Algorithms for Function Optimization.
Dissertation, University of Alberta, Edmonton, 1980.

[Bro58] S. H. Brooks:
A Discussion of Random Methods for Seeking Maxima.
Oper. Res. 6, S.244-251, 1958.

[Bru93] R. Bruns:
Direct Chromosome Representation and Advanced Genetic Operators for Production Scheduling.
In: S. Forrest (ed.): Genetic Algorithms, Proc. of the 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.352-359, 1993.

[Bur90] R. E. Burkhard:
Locations with Spatial Interactions: The Quadratic Assignment Problem.
In: B. P. Mirchandani, R. L. Francis: (eds.): Discrete Location Theory, John Wiley & Sons, New York, S.387-437, 1990.

[Bur95] E. K. Burke, D. G. Elliman, R. F. Weare:
A Hybrid Genetic Algorithm for Highly Constrained Timetabling Problems.
In: L. J. Eshelman (ed): Proc. of the 6th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, S.605-610, 1995.

[CEC99] *Congress on Evolutionary Computing.*
Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., 1999.

[CEC00] *Congress on Evolutionary Computing.*
Conf. Proc. CEC 2000, IEEE press, Piscataway, N.J., 2000.

[CEC01] *Congress on Evolutionary Computing.*
Conf. Proc. CEC 2001, IEEE press, Piscataway, N.J., 2001.

[Col90] A. Colorni, M. Dorigo, V. Maniezzo:
Applying Evolutionary Algorithms to Solve the Time-Table Problem.
In: H.-P. Schwefel, R. Männer (eds.): Proc. of PPSN I, LNCS 496, Springer-Verlag, S.55-59, 1991.

[Cox91] L. A. Cox jr., L. Davis, Y. Qiu:
Dynamic Anticipatory Routing in Circuit-Switched Telecommunications Networks.
In: L. Davis (ed.): Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, S.124-143, 1991.

[Dah99] K. P. Dahal, C. J. Aldridge, J. R. McDonald, G. M. Burt:
A GA-based Technique for the Scheduling of Storage Tanks.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.2199-2206, 1999.

[Dak65] R. J. Dakin:
A Tree-Search Algorithm for Mixed Integer Programming Problems.
Comp. Journal 8, S.250-255, 1965.

[Dan66] G. B. Danzig:
Lineare Programmierung und Erweiterungen.
Springer, Berlin, 1966.

[Dar60] C. Darwin:
On the Origin of Species by Means of Natural Selection or the Preservation of Favoured Races in the Struggle for Life.
London, John Murray, 1860.

[Dav59] W. C. Davidon:
Variable Metric Method for Minimization.
Argonne Nat. Lab., Report ANL-5990 rev., Lemont, 1959.

[Dav87a] L. Davis, M. Steenstrup:
Genetic Algorithms and Simulated Annealing: An Overview.
In: L. Davis (ed.): Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence, Pitman, London, Morgan Kaufmann, Los Altos, 1987.

[Dav87b] L. Davis (ed.):
Genetic Algorithms and Simulated Annealing.
Research Notes in Artificial Intelligence, Pitman, London, Morgan Kaufmann, Los Altos, 1987.

[Dav91] L. Davis (ed.):
Handbook of Genetic Algorithms.
Van Nostrand Reinhold, New York, 1991.

[Dav94] Y. Davidor, H.-P. Schwefel, R. Männer (eds.):
Parallel Problem Solving from Nature III.
Conf. Proc. PPSN III, LNCS 866, Springer-Verlag, Berlin, 1994.

[DeJ75] K. De Jong:
An Analysis of the Behavior of a Class of Genetic Adaptive Systems.
Dissertation, University of Michigan, Ann Arbor, 1975.

[DeJ93] K. De Jong, J. Sarma:
Generation Gaps Revisited.
In: L. D. Whitley (ed): Foundations of Genetic Algorithms 2, Morgan Kaufmann, S.19-28, 1993.

[Dix72] L. C. W. Dixon:
The Choice of Step Length, a Cruical Factor in the Performance of Variable Metric Algorithms.
In: F. A. Lootsma (ed.): Numerical Methods for Non-linear Optimization. Academic Press, London, 1972.

[Dor98] R. Dorne, J.-K. Hao:
A New Genetic Search Algorithm for Graph Coloring.
In: A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (eds): Conf. Proc. PPSN V, LNCS 1498, Springer-Verlag, Berlin, S.745-754, 1998.

[Dri92] R. van Driessche, R. Piessens:
Load Balancing with Genetic Algorithms.
In: R. Männer, B. Manderick (eds.): Conf. Proc. PPSN II, North-Holland, Amsterdam, S.341-350, 1992.

[Ebe96] W. Ebeling, I. Rechenberg, H.-P. Schwefel, H.-M. Voigt (eds.):
Parallel Problem Solving from Nature IV.
Conf. Proc. PPSN IV, LNCS 1141, Springer-Verlag, Berlin, 1996.

[Egg97] H. Eggert, H. Guth, W. Jakob, S. Meinzer, I. Sieber, W. Süß:
Designoptimierung für Mikrosysteme.
In: R. Laur, W. John, W. Groß (Hrsg.): Methoden und Werkzeuge zum Entwurf von Mikrosystemen, 6.GMM-Workshop, S.3-10, 1997.

[Egg98] H. Eggert, H. Guth, W. Jakob, S. Meinzer, I. Sieber, W. Süß:
Design Optimization of Microsystems.
In: Proc. of International Conference on Modelling and Simulation of Microsystems, Semiconductors, Sensors and Actuators (MSM 98), S.344-349, 1998.

[Ehr87] W. Ehrfeld, E. W. Becker:
Das LIGA-Verfahren zur Herstellung von Mikrostrukturkörpern mit großem Aspektverhältnis und großer Strukturhöhe.
KfK-Nachrichten, Heft 4/87, S.167-179, 1987.

[Eib98] A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (eds):
Parallel Problem Solving from Nature V.
Conf. Proc. PPSN V, LNCS 1498, Springer-Verlag, Berlin, 1998.

[Eme66] F.E. Emery, M. O'Hagan:
Optimal Design of Matching Networks for Microwave Transistor Amplifiers.
IEEE Trans. MTT-14, S.696-698, 1966.

[Esh89] L. J. Eshelman, R. A. Curuana, J. D. Schaffer:
Biases in the Crossover Landscape.
In: J. D. Schaffer (ed.): Genetic Algorithms, Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.10-19, 1989.

[Esh95] L. J. Eshelman (ed.):
Genetic Algorithms.
Proc. of the 6th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, 1995.

[Esp01] F. P. Espinoza, B. S. Minsker, D. E. Goldberg:
A Self Adaptive Hybrid Genetic Algorithm.
In: L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, E. Burke (eds.): Genetic and Evolutionary Computation Conference (GECCO 2001), Vol. Late Breaking Papers, Morgan Kaufmann, San Francisco, CA, S.75-80, 2001.

[Fal80] K. von Falkenhausen:
Optimierung regionaler Entsorgungssysteme mit der Evolutionsstrategie.
In: L. von Dobschütz, B. Fleischmann, C. Schneeweiß, H. Steckhan (Hrsg.): Proc. in Operations Research 9, Phyisca 1980, Würzburg, S.40-51, 1980.

[Fog66] D. B. Fogel, A. J. Owens, M. J. Walsh:
Artifical Intelligence through Simulated Evolution.
John Wiley & Sons, Chichester, 1966.

[Fog92] D. B. Fogel:
Evolving Artifical Intelligence.
Dissertation, University of California, San Diego, 1992.

[For93] S. Forrest (ed):
Genetic Algorithms.
Proc. of the 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1993.

[Fle63] R. Fletcher, M. J. D. Powell:
A Rapidly Convergent Descent Method for Minimization.
Comp. Journal 6, S.163-168, 1963.

[Fle64] R. Fletcher, C. M. Reeves:
Function Minimization by Conjugate Gradients.
Comp. Journal 7, S.194-154, 1964.

[Fle87] R. Fletcher:
Practical Methods of Optimization.
John Wiley & Sons, Chichester, 1987.

[Frei96] B. Freisleben, P. Merz:
A Genetic Local Search Algorithm for Solving Symmetric and Asymmetric Traveling Salesman Problems.
In: Conf. Proc. of the 1996 IEEE Int. Conf. on Evolutionary Computation (ICEC'96), IEEE, S.616-621, 1996.

[Glo97] F. Glover, M. Laguna:
Tabu Search.
Kluwer Academic Publishers, 1997.

[Gol89] D. E. Goldberg:
Genetic Algorithms in Search, Optimization, and Machine Learning.
Reading/MA, Addison-Wesley, 1989.

[Gol99] D. E. Goldberg, S. Voessner:
Optimizing Global-Local Search Hybrids.
In: W. Banzhaf, J. Daida, A. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, R. E. Smith: Proc. of the Genetic And Evolutionary Computation Conference (GECCO'99), Morgan Kaufmann, San Francisco, CA, S.220-228, 1999.

[Gor90] M. Gorges-Schleuter:
Genetic Algorithms and Population Structures - A Massively Parallel Algorithm.
Doktorarbeit, Universität Dortmund, 1990.

[Gor94] M. Gorges-Schleuter:
Parallel Evolutionary Algorithms and the Concept of Population Structures.
In: V. Plantamura, B. Soucek, G. Vissaggio (eds.): Frontier Decision Support Concepts. Chapt. 15 and 16, Wiley, New York, S.261-319, 1994.

[Gor96] M. Gorges-Schleuter, W. Jakob, S. Meinzer, A. Quinte, W. Süß, H. Eggert:
An Evolutionary Algorithm for Design Optimization of Microsystems.
In: W. Ebeling, I. Rechenberg, H.-P. Schwefel, H.-M. Voigt (eds.): Conf. Proc. PPSN IV, LNCS 1141, Springer-Verlag, Berlin, S.1022-1032, 1996.

[Gor98a] M. Gorges-Schleuter:
A Comparative Study of Global and Local Selection in Evolution Strategies.
In: A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (eds): Conf. Proc. PPSN V, LNCS 1498, Springer-Verlag, Berlin, S.367-377, 1998.

[Gor98b] M. Gorges-Schleuter, W. Jakob, W. Süß:
Designoptimierung am Beispiel einer Mikropumpe.
In: S. Hafner (Hrsg.): Industrielle Anwendungen evolutionärer Algorithmen. Oldenburg Verlag, München, S.71-82, 1998.

[Gor98c] M. Gorges-Schleuter, W. Jakob, Sieber, I.:
Evolutionary Design Optimization of a Microoptical Collimation System.
In: H.-J. Zimmermann (Hrsg.): Proc. of 6th European Congress on Intelligent Techniques and Soft Computing (Eufit 98), Verlag Mainz, Wissenschaftsverlag, S.392-396, 1998.

[Gor99a] M. Gorges-Schleuter, I. Sieber, W. Jakob:
Local Interaction Evolution Strategies for Design Optimization.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.2167-2174, 1999.

[Gor99b] M. Gorges-Schleuter:
An Analysis of Local Selection in Evolution Strategies.
In: W. Banzhaf, J. Daida, A. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, R. E. Smith: Proc. of the Genetic And Evolutionary Computation Conference (GECCO'99), Morgan Kaufmann, San Francisco, CA, S.847-854, 1999.

[Gre86] J. J. Grefenstette:
Optimization of Control Parameters for Genetic Algorithms.
IEEE Transactions on Systems, Man, and Cybernetics SMC-16, Vol.1, S.122-128, 1986.

[Gre89] J. J. Grefenstette, J. E. Baker:
How Genetic Algorithms Work: A Critical Look at Implicit Parallelism.
In: J. D. Schaffer (ed.): Genetic Algorithms, Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.20-27, 1989.

[Gre93] J. J. Grefenstette:
Deception Considered Harmful.
In: L. D. Whitley (ed): Foundations of Genetic Algorithms 2, Morgan Kaufmann, S.75-91, 1993.

[Gri99] J. B. Grimbleby:
Hybrid Genetic Algorithms for Analogue Network Synthesis.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.1781-1787, 1999.

[Gro90] C. de Groot, D. Würtz, K. H. Hoffmann:
Optimizing Complex Problems by Nature's Algorithms: Simulated Annealing and Evolution Strategy - a Comparative Study.
In: H.-P. Schwefel, R. Männer (eds.): Proc. of PPSN I, LNCS 496, Springer-Verlag, S.445-454, 1991.

[Gru93] F. Gruau, D. Whitley:
Adding Learning to the Cellular Development of Neural Networks: Evolution and the Baldwin Effect.
Evolutionary Computation 1, Heft 3, S.213-233, 1993.

[Haf98] S. Hafner (Hrsg.):
Industrielle Anwendungen evolutionärer Algorithmen.
Oldenbourg Verlag, München, 1998.

[Har93] I. Harvey:
The Puzzle of the Persistent Question Marks: A Case Study of Genetic Drift.
In: S.Forrest (ed): Proc. of the 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.15-22, 1993.

[Has82] I. Hasenfuss:
Die Selektionstheorie.
In R. Siewing (Hrsg.): Evolution, UTB-Fischer, Stuttgart, S.307-318, 1982.

[Her90] M. Herdy:
Application of the Evolutionsstrategie to Discrete Optimization Problems.
In: H.-P. Schwefel, R. Männer (eds.): Proc. of PPSN I, LNCS 496, Springer-Verlag, Berlin, S.188-192, 1991.

[Her96] M. Herdy:
Evolution Strategies with Subjective Selection.
In: W. Ebeling, I. Rechenberg, H.-P. Schwefel, H.-M. Voigt (eds.): Conf. Proc. PPSN IV, LNCS 1141, Springer-Verlag, Berlin, S.22-31, 1996.

[Hof92] F. Hoffmeister, T. Bäck:
Genetic Algorithms and Evolution Strategies: Similarities and Differences.
Technical Report SYS-1/92, Universität Dortmund, FB Informatik, 1992.

[Höf76] A. Höfler:
Formoptimierung von Leichtbaufachwerken durch Einsatz einer Evolutionsstrategie.
Dissertation, FB Verkehrswesen, Inst. für Luft- und Raumfahrt, TU Berlin, ILR-Bericht 17, 1976.

[Hol71] R. B. Hollstien:
Artificial Genetic Adaptation in Computer Control Systems.
Dissertation, University of Michigan, Ann Arbor. 1971.

[Hol75] H. J. Holland:
Adaptation in Natural and Artificial Systems.
The University of Michigan Press, Ann Arbor. 1975.

[Hol78] H. J. Holland, J. S. Reitman:
Cognitive Systems Based on Adaptive Algorithms.
In: D. A. Waterman, F. Hayes-Roth (eds.): Pattern Directed Inference Systems, Academic Press, New York, S.313-329, 1978.

[Hoo61] R. Hooke, T. A. Jeeves:
Direct Search Solution of Numerical and Statistical Problems.
JACM 8, S.212-229, 1961.

[Hor81] E. Horowitz, S. Sahni:
Algorithmen - Entwurf und Analyse.
Springer-Verlag, Berlin, 1981.

[Jäk97a] J. Jäkel:
Fuzzy Model Identification Based on a Genetic Algorithm and Optimization Techniques.
Conf. Proc. 5th European Congress on Intelligent Techniques and Soft Computing, EUFIT'97, S.774-778, 1997.

[Jäk97b] J. Jäkel:
Ein Genetischer Algorithmus zur Strukturidentifikation von Fuzzy-Modellen.
Proceedings 42, Internationales Wissenschaftliches Kolloquium, TU Ilmenau, S.234-239, 1997.

[Jak92] W. Jakob, M. Gorges-Schleuter, C. Blume:
Application of Genetic Algorithms to Task Planning and Learning.
In: R. Männer, B. Manderick (eds.): Conf. Proc. PPSN II, North-Holland, Amsterdam, S.291-300, 1992.

[Jak96a] W. Jakob, S. Meinzer, A. Quinte, W. Süß, M. Gorges-Schleuter, H. Eggert:
Partial Automated Design Optimization Based on Adaptive Search Techniques,
In: I. C. Parmee (ed.): Adaptive Computing in Engineering Design and Control ’96, PEDC, University of Plymouth, S.236-241, 1996.

[Jak96b] W. Jakob, S. Meinzer, A. Quinte, G. Clemens:
Simulator und Genetischer Algorithmus verkürzen die Entwicklungsphase.
elektronik industrie 5, S.56-69, 1996.

[Jak98a] W. Jakob, M. Gorges-Schleuter, I. Sieber:
Comparison of Evolutionary Algorithms for Design Optimization.
In: A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (eds): Conf. Proc. PPSN V, LNCS 1498, Springer-Verlag, Berlin, S.917-926, 1998.

[Jak98b] W. Jakob, M. Gorges-Schleuter, I. Sieber:
Evolutionäre Verfahren zur Designoptimierung von Mikrosystemen.
3.PMT-Statuskolloquium, FZKA-Bericht 6080, Forschungszentrum Karlsruhe, S.189-190, 1998.

[Jak99a] W. Jakob, B. Knorr, S. Parodat, D. Peters, A. Uhlig:
Optimierung von Mikrosystemen.
In: W. John, H. Luft, W. Groß (Hrsg.): Methoden und Werkzeuge zum Entwurf von Mikrosystemen, 8.GMM-Workshop, S.241-254, 1999.

[Jak99b] W. Jakob, M. Gorges-Schleuter, I. Sieber, W. Süß, H. Eggert:
Solving a Highly Multimodal Design Optimization Problem Using the Extended Genetic Algorithm GLEAM.
In: S. Hernandez, A.J. Kassab, C. A. Brebbia: Computer Aided Design of Structures VI, WIT Press, Southampton, Conf. Proc. OPTI 99, S.205-214, 1999.

[Jak01a] W. Jakob, A. Quinte, K.-P. Scherer, H. Eggert:
Optimisation of a Micro Fluidic Component Using a Parallel Evolutionary Algorithm and Simulation Based on Discrete Element Methods.
In: S. Hernandez, A.J. Kassab, C. A. Brebbia: Computer Aided Design of Structures VII, WIT Press, Southampton, Conf. Proc. OPTI 2001, S.337-346, 2001.

[Jak01b] W. Jakob:
HyGLEAM: Hybrid General Purpose Evolutionary Algorithm and Method.
In: N. Callaos, S. Esquivel, J. Burge (eds.): World Multiconference on Systematics, Cybernetics and Informatics (SCI 2001), IIIS and IEEE, Venezuela, Vol.3, S.187-192, 2001.

[Jak01c] W. Jakob, S.Peters:
Designoptimierung mit HyGADO.
In: R. Laur, D. Peters (Hrsg.): OMID – Optimierung von Mikrosystemen für Diagnose- und Überwachungsanwendungen (Statusseminar), VDI/VDE, S.15-24, Oktober 2001.

[Jak02a] W. Jakob:
HyGLEAM - An Approach to Generally Applicable Hybridization of Evolutionary Algorithms.
In: J.J. Merelo, et.al (eds): Conf. Proc. PPSN VII, LNCS 2439, Springer-Verlag, Berlin, S.527-536, 2002.

[Jak02b] W. Jakob, S.Peters, A. Reiffer:
Internetbasierte Designoptimierung mit HyGLEAM.
In: H. Eggert. A. Reiffer (Hrsg.): Optimierung von Mikrosystemen für Diagnose- und Überwachungsanwendungen (Abschlussseminar des BMBF-Verbundprojektes OMID), Wissenschaftl. Berichte FZKA 6774, S.14-27, November 2002.

[Jak03] W. Jakob:
HyGLEAM - Towards a Generally Applicable Self-adapting Hybridiztion of Evolutionary Algorithms.
Submitted to Evolutionary Computation, Special Issue on Memetic Algorithms.

[Jan91] C. Z. Janikow, Z. Michalewicz:
An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms.
In: R. K. Belew, L. B. Booker: Proc. of the 4th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.31-36, 1991.

[Kel00] H. B. Keller:
Maschinelle Intelligenz.
Vieweg Verlag, Wiesbaden, 2000.

[Kir83] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi:
Optimization by Simulated Annealing.
Science 220, S.671-680, 1983.

[Kob87] E. Kobes:
Entwicklung von Computeralgorithmen zur Optimierung von Strukturkomponenten nach der Evolutionstheorie.
Diplomarbeit, Inst. für Computer-Anwendungen, Universität Stuttgart, 1987.

[Kon99] A. Konak, A. E. Smith:
A Hybrid Genetic Algorithm Approach for Backbone Design of Communication Networks.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.1817-1823, 1999.

[Kow68] J. Kowalik, M. R. Osborne:
Methods for Unconstrained Optimization Problems.
Amer. Elsevier, New York, 1968.

[Law72] J. P. Lawrence, K. Steiglitz:
Randomized Pattern Search.
IEEE Trans. C-21, S.382-385, 1972.

[Lie94] J. Lienig, H. Brandt:
An Evolutionary Algorithm for the Routing of Multi Chip Modules.
In: Y. Davidor, H.-P. Schwefel, R. Männer (eds.): Conf. Proc. PPSN III, LNCS 866, Springer-Verlag, Berlin, S.588-597, 1994.

[Lin65] S. Lin:
Computer Solutions of the Traveling Salesman Problem.
Bell. Sys., Tech. Journ., 44, S.2245-2269, 1965.

[Lju00] I. Ljubic, G. R. Raidl, J. Kratica:
A Hybrid GA For the Edge-Biconnectivity Augmentation Problem.
In: M. Schoenauer (ed.): Conf. Proc. PPSN VI, LNCS 1917, Springer-Verlag, Berlin, S.641-650, 2000.

[Lob97] F. G. Lobo, D. E. Goldberg:
Decision Making in a Hybrid Genetic Algorithm.
In: Conf. Proc. of the 1997 IEEE Int. Conf. on Evolutionary Computation (ICEC'97), IEEE, S.121-125, 1997.

[Män92] R. Männer, B. Manderick (eds.):
Parallel Problem Solving from Nature II.
Conf. Proc. PPSN II, North-Holland, Amsterdam, 1992.

[McC69] G. P. McCormic, G. P. Pearson:
Variable Metric Methods and Unconstrained Optimization.
In: R. Fletcher (ed.): Optimization. Academic Press, London, S.307-326, 1969.

[Meh86] K. Mehlhorn:
Datenstrukturen und effiziente Algorithmen.
Band 1, B. G. Teubner, Stuttgart, 1986.

[Mei96] S. Meinzer, A. Quinte, M. Gorges-Schleuter, W.Jakob, W. Süß, H. Eggert:
Simulation and Design Optimization of Microsystems Based on Standard Analog Simulators and Adaptive Search Techniques
Proc. of the SIG-VHDL Spring '96 Working Conference, Shaker Verlag, S.169-180, 1996.

[Mei98a] S. Meinzer:
Entwicklung von Verfahren zur Erstellung adaptierter Makromodelle für den Einsatz bei der Designoptimierung von Mikrosystemen,
Dissertation an der Universität Bremen und
FZKA Bericht 5952, Forschungszentrum Karlsruhe, 1998.

[Mei98b] S. Meinzer, H. Eggert, M. Gorges-Schleuter, W. Jakob, A. Quinte, W. Süß:
Development of Methods for Building Adapted Macromodels for the Application in the Design Optimization of Microsystems.
Proc. of 6th Int. Conf. on Micro Electro, Opto, Mechanical Systems and Components (MICRO SYSTEM Technologies 98), VDE-Verlag, S.485-490, 1998.

[Mer02] J.J. Merelo, A. Panagiotis, H.-G. Beyer, J.-L. Fernández-Villacañas, H.-P. Schwefel (eds):
Parallel Problem Solving from Nature VII.
Conf. Proc. PPSN VII, LNCS 2439, Springer-Verlag, Berlin, 2002.

[Met53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller:
Equation of State Calculations by Fast Computing Machines.
Journ. Chem. Phys. 21, S.1987-1092, 1953.

[Mic92] Z. Michalewicz:
Genetic Algorithms + Data Structures = Evolution Programs.
Springer Verlag, Berlin, 1992.

[Mik98a] R. Mikut, F. Hendrich:
Produnktionsreihenfolgeplanung in Ringwalzwerken mit wissensbasierten und evolutionären Methoden.
Automatisierungstechnische Praxis (atp) 46, Heft 1/98, Oldenbourg, München, S.15-21, 1998.

[Mik98b] R. Mikut, F. Hendrich, G. Bretthauer:
Einsatz von Methoden der Computational Intelligence zur Produnktionsplanung in Ringwalzwerken.
In: Computational Intelligence - Neuronale Netze, Evolutionäre Algorithmen, Fuzzy Control im industriellen Einsatz, VDI-Berichte 1381, VDI Verlag Düsseldorf, S.53-67, 1998.

[Moo68] J. Moore:
An n Job, One Machine Sequencing Algorithm for Minimizing the Number of Late Jobs.
Management Science, Vol. 15, Heft 1, S.102-109, 1968.

[Mos92] P. Moscato, M. G. Norman:
A „Memetic“ Approach for the Travelling Salesman Problem - Implementation of a Computational Ecology for Combinatorical Optimisation on a Messag-Passing System.
In: M.Valero et al. (eds.): Proc. of the Int. Conf. on Parallel Computing and Transputer Applications. IOS Press, Amsterdam, S.177-186, 1992.

[Mühl89] H. Mühlenbein:
Parallel Genetic Algorithms, Population Genetics and Combinatorical Optimization.
In: J. D. Schaffer (ed.): Genetic Algorithms, Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.416-421, 1989.

[Mühl91] H. Mühlenbein:
Evolution in Time and Space - The Parallel Genetic Algorithm.
In: G. J. E. Rawlins (ed): Foundations of Genetic Algorithms, Morgan Kaufmann, S.316-337, 1991.

[Mühl92] H. Mühlenbein:
How Genetic Algorithms Really Work I. Mutation and Hillclimbing.
In: R. Männer, B. Manderick (eds.): Conf. Proc. PPSN II, North-Holland, Amsterdam, S.15-22, 1992.

[Mur70] B. A. Murtagh, R. W. H. Sargant:
Computational Experience with Quadratically Convergent Minimization Methods.
Comp. Journal 13, S.185-194, 1972.

[Nel65] J. A. Nelder, R. Mead:
A Simplex Method for Function Minimization.
Comp. Journal 7, S.308-313, 1965.

[Nis94] V. Nissen:
Evolutionäre Algorithmen - Darstellung, Beispiele, Betriebswirtschaftliche Anwendungsmöglichkeiten.
Deutscher Universitätsverlag, Wiesbaden, 1994.

[Orv93] D. Orvosh, L. Davis:
Shall we Repair? Genetic Algorithms, Combinatorical Optimization, and Feasibility Constraints.
In: S.Forrest (ed): Proc. of the 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.650, 1993.

[Ort66] J. M. Ortega, M. L. Rockoff:
Nonlinear Difference Equations and Gauss-Seidel Type Iterative Methods.
SIAM J. Numer. Anal. 3, S.497-513, 1966.

[Par94] I. C. Parmee (ed.):
Adaptive Computing in Engineering Design and Control '94.
Conf. Proc., PEDC, University of Plymouth, 21-22 September, 1994.

[Par96] I. C. Parmee (ed.):
Adaptive Computing in Engineering Design and Control '96.
Conf. Proc., PEDC, University of Plymouth, 26-28 March, 1996.

[Par98] I. C. Parmee (ed.):
Adaptive Computing Design and Manufacture '98.
Conf. Proc., PEDC, University of Plymouth, 21-23 April, 1998.

[Pet99a] D. Peters, W. Jakob, M. Gorges-Schleuter, S. Parodat:
Modellbasierte Optimierung mit mathematischen Methoden - ein Vergleich von klassischen und evolutionären Verfahren.
In: W. John, W. Groß, R. Laur (Hrsg.): Methoden und Werkzeuge zum Entwurf von Mikrosystemen, 7.GMM-Workshop, S.127-136, 1999.

[Pet99b] D. Peters:
Einführung in die deterministischen ableitungsfreien Verfahren.
Beitrag zum Seminar „Werkzeuge und Verfahren zur Optimierung von Mikrosystemen" im Rahmen des 8.GMM-Workshops Methoden und Werkzeuge zum Entwurf von Mikrosystemen, 1999.

[Pol92] D. Polani, T. Uthmann:
Adaptation of Kohonen Feature Map Topologies by Genetic Algorithms.
In: R. Männer, B. Manderick (eds.): Conf. Proc. PPSN II, North-Holland, Amsterdam, S.421-430, 1992.

[Pow62] M. J. D. Powell:
An Iterative Method for Finding Stationary Values of a Function of Several Variables.
Comp. Journal 5, S.147-151, 1962.

[Pow64] M. J. D. Powell:
An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivativs.
Comp. J. 7, S.155-162, 1964.

[Pow89] D. J. Powell, S. S. Tong, M. M. Skolnick:
EnGENEous - Domain Independent, Machine Learning for Design Optimization.
In: J. D. Schaffer (ed.): Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.151-159, 1989.

[Rec73] I. Rechenberg:
Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.
Frommann-Holzboog, Stuttgart-Bad Cannstatt. 1973.

[Rec94] I. Rechenberg:
Evolutionsstrategie ’94.
Werkstatt Bionik und Evolutionstechnik, Frommann-Holzboog, Stuttgart-Bad Cannstatt. 1994.

[Ron97] S. Ronald:
Distance Functions for Order Based Encodings.
In: Conf. Proc. of the 1997 IEEE Int. Conf. on Evolutionary Computation (ICEC’97), IEEE, S.46-54, 1997.

[Ros60] H. H. Rosenbrock:
An Automatic Method for Finding the Greatest or Least Value of a Function.
Comp. Journal 3, S.175-184, 1960.

[Rud90] G. Rudolph:
Global Optimization by Means of Distributed Evolution Strategies.
In: H.-P. Schwefel, R. Männer (eds.): Proc. of PPSN I, LNCS 496, Springer-Verlag, S.209-213, 1991.

[Run73] W. Runggaldier:
Ganzzahlige, Null-Eins- und Gemischt-Ganzzahlige Programmierung im Zusammenhang mit der Branch-and-Bound-Technik.
In: F. Weinberg (Hrsg.): Branch and Bound: Eine Einführung. Lecture Notes in Economics and Mathematical Systems, Springer, Berlin, 1973.

[Saa02] G. Saake, K.-U. Sattler:
Algorithmen und Datenstrukturen - Eine Einführung mit Java.
dpunkt-Verlag, Heidelberg, 2002

[Scha89a] J. D. Schaffer (ed.):
Genetic Algorithms.
Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989.

[Scha89b] J. D. Schaffer, R. A. Caruana, L. J. Eshelman:
A Study of Control Parameters Affecting Online Performance of Genetic Algorithms for Function Optimization.
In: J. D. Schaffer (ed.): Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.51-50, 1989.

[Scho00] M. Schoenauer (ed.):
Parallel Problem Solving from Nature VI.
Conf. Proc. PPSN VI, LNCS 1917, Springer-Verlag, Berlin, 2000.

[Schö94] E. Schöneburg, F. Heinzmann, S. Feddersen:
Genetische Algorithmen und Evolutionsstrategien - Eine Einführung in Theorie und Praxis der simulierten Evolution.
Addison-Wesley, Bonn, 1994.

[Schw81] H.-P. Schwefel:
Numerical Optimization of Computer Models.
John Wiley & Sons, Chichester, 1981.

[Schw91] H.-P. Schwefel, R. Männer (eds.):
Parallel Problem Solving from Nature I.
Conf. Proc. PPSN I (October 1-3, 1990), LNCS 496, Springer-Verlag, Berlin, 1991[1].

[Schw95] H.-P. Schwefel:
Evolution and Optimum Seeking.
John Wiley & Sons, Chichester, 1995.

[Sed92] R. Sedgewick:
Algorithmen.
Addison-Wesley, Bonn, 1992.

[She71] J. Shekel:
Test Functions for Multimodal Search Techniques.
Fifth Annual Princeton Conference on Information Science and Systems, 1971.

[She86] H. D. Sheraldi, P. Rajgopal:
A Flexible, Polinomial-time Construction and Improvement Heuritic for the Quadratic Assignment Problem.
Operations Research, 13, S.587-600, 1986.

1. Der zur Konferenz erschienene und im wesentlichen nur an die Teilnehmer verteilte Tagungsband wurde 1991 als Buch veröffentlicht. Daher wird das Buch zitiert. Alle Bezüge auf Konferenzbeiträge werden mit 1990 angegeben, da die Erstveröffentlichung in diesem Jahr erfolgte.

[Sie98a] I. Sieber, H. Eggert, H. Guth, W. Jakob:
Design Simulation and Optimization of Microoptical Components.
In: Novel Optical Systems and Large-Aperture Imaging, SPIE´s 43rd Annual Meeting, SPIE Vol.3430, S.138-149, 1998.

[Sie98b] I. Sieber, H. Eggert, H. Guth, W. Jakob, K.-P. Scherer, P. Ziegler:
Design Optimization Considering Tolerance Effects of Microoptical Benches.
In: MicroSystem Technologies 98, VDE-Verlag GmbH, S.65-70, 1998.

[Sie99] I. Sieber, H. Eggert, H. Guth, W. Jakob, K.-P. Scherer:
Designoptimierung in der Mikrosystemtechnik mit Evolutionären Algorithmen.
Informationstechnik und Technische Informatik (it+ti) 41, Heft 4/99, Oldenbourg, München, S.27-32, 1999.

[Sie00] I. Sieber, H. Eggert, H. Guth:
Entwicklung einer rechnergestützten Entwurfsmethode für optische Mikrosysteme und deren Anwendung auf einen Heterodynempfänger.
Dissertation, Universität Bremen, FZKA-Bericht 6403, Forschungszentrum Karlsruhe, 2000.

[Spe01] L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, E. Burke (eds.):
GECCO 2001 - Proc. of the Genetic and Evolutionary Computation Conference.
Morgan Kaufmann, San Francisco, CA, 2001.

[Ste67] G.W. Stewart:
A Modification of Davidon's Minimization Method to Accept Difference Approximations of Derivatives.
JACM 14, S.72-83, 1967.

[Süe99] G. A. Süer, R. Vázquez, M. Cortés:
A Hybrid Approach of Genetic Algorithms and Local Optimizers in Cell Loading.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.2287-2293, 1999.

[Süß97a] W. Süß, W. Jakob, M. Gorges-Schleuter, S. Meinzer, A. Quinte, H. Eggert, H. Guth, I. Sieber:
Design Optimization of Microsystems Based on Adaptive Search Techniques.
In: S. Hernandez, C. A. Brebbia: Computer Aided Design of Structures V, Computational Mechanics Publications, Southampton, S.121-130, 1997.

[Süß97b] W. Süß, S. Meinzer, A. Quinte, W. Jakob, H. Eggert, M. Gorges-Schleuter:
Simulation and Design Optimization of a Micropump.
In: R. A. Adey, P. H. Renaud: Proc. of 2nd Int. Conf. on the Simulation and Design of Microsystems and Microstructures, Computational Mechanics Publications, Southampton, S.127-135, 1997.

[Sys89] G. Syswerda:
Uniform Crossover in Genetic Algoriths.
In: J. D. Schaffer (ed.): Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.2-9, 1989.

[Sys91] G. Syswerda:
A Study of Reproduction in Generational and Steady-State Genetic Algorithms.
In: G. J. E. Rawlins (ed): Foundations of Genetic Algorithms, Morgan Kaufmann, S.94-101, 1991.

[Tab69] D. Tabak:
Comparative Study of Various Minimization Techniques Used in Mathematical Programming.
IEEE Trans. AC-14, S.572, 1969.

[Tag98] K. Tagawa, Y. Kanzaki, D. Okada, K. Inoue, H. Haneda:
A New Metric Function and Harmonic Crossover for Symmetric and Asymmetric Traveling Salesman Problems.
In: Conf. Proc. of the 1998 IEEE Int. Conf. on Evolutionary Computation (ICEC'98), IEEE, S.822-827, 1998.

[Tör89] A. Törn, A. Zilinskas:
Global Optimization.
LNCS 350, Springer Verlag, Berlin, 1989.

[Van67] R. VanNorton:
Lösung linearer Gleichungssysteme nach dem Verfahren von Gauß-Seidel.
In: A. Ralston, H. S. Wilf (eds.): Mathematische Methoden für Digitalrechner, Oldenbourg, München, S.92-105, 1967.

[VDI3550] H.-G. Beyer, E. Brucherseifer, W. Jakob, W. Pohlheim, B. Sendhoff, T.B. To:
VDI/VDE-Richtlinie 3550, Blatt 3: Evolutionäre Algorithmen - Begriffe und Definitionen (Gründruck).
VDI/VDE-Handbuch Regelungstechnik, Verein Deutscher Ingenieure, Düsseldorf, 2001.

[Waa92] D. Waagen, P. Diercks, J. McDonnell:
The Stochastic Direction Set Algorithm: A Hybrid Technique for Finding Function Extrema.
In: D. B. Fogel, W. Atmar (eds.): Proc. 1st Conf. on Evolutionary Programming, Evolutionary Programming Society, San Diego, CA, S.35-42, 1992.

[Wei94] T. Weinberger, H. B. Keller, W. Jakob, B. große Osterhues:
Modelle maschinellen Lernens - Symbolische und subsymbolische Ansätze.
KfK-Bericht 5184, Kernforschungszentrum Karlsruhe, Karlsruhe, 1994.

[Wez94] M. C. van Wezel, J. N. Kok, J. van den Berg, W. van Kampen:
Genetic Improvement of Railway Timetables.
In: Y. Davidor, H.-P. Schwefel, R. Männer (eds.): Conf. Proc. PPSN III, LNCS 866, Springer-Verlag, Berlin, S.566-575, 1994.

[Whi88] D. Whitley, J. Kauth:
GENITOR: A Different Genetic Algorithm.
Technical Report CS-88-101, Colorado State University, Dep. of Computer Science, Fort Collins, 1988.

[Whi89] D. Whitley:
The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials Is Best.
In: J. D. Schaffer (ed.): Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, S.116-121, 1989.

[Whi94] D. Whitley, V. S. Gordon, K. Mathias:
Lamarckian Evolution, The Baldwin Effect and Function Optimization.
In: Y. Davidor, H.-P. Schwefel, R. Männer (eds.): Conf. Proc. PPSN III, LNCS 866, Springer-Verlag, Berlin, S.6-14, 1994.

[Whi00] D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, H.-G. Beyer (eds.):
GECCO 2000 - Proc. of the Genetic and Evolutionary Computation Conference.
Morgan Kaufmann, San Francisco, CA, 2000.

[Wil67] D. J. Wilde, C. S. Beightler:
Foundations of Optimization.
Prentice-Hall, Englewood Cliffs, N.J., 1967.

[Wri91] A. H. Wright:
Genetic Algorithms for Real Parameter Optimization.
In: G. J. E. Rawlins (ed): Foundations of Genetic Algorithms, Morgan Kaufmann, S.205-218, 1991.

[Wuk85] F. M. Wuketits:
Die systemtheoretische Innovation der Evolutionstheorie.
In: J. A. Ott, G. P. Wagner, F. M. Wuketits (Hrsg.): Evolution, Ordnung und Erkenntnis. Berlin, Parey, S.69-81, 1985.

[Yan99] L. Yang, J. Yen, A. Rajesh, K. D. Kihm:
A Supervisory Architecture and Hybrid GA for the Identifications of Complex Systems.
In: Conf. Proc. CEC 99, IEEE press, Piscataway, N.J., S.862-869, 1999.

[Zan67] W. I. Zangwill:
Minimizing a Function Without Calculating Derivatives.
Comp. Journal 10, S.293-296, 1967.

[Zim85] A. Zimmermann:
Evolutionsstrategische Modelle bei einstufiger, losweiser Produktion.
Peter Lang, Frankfurt a.M., 1985.

[Zit00] E. Zitzler, J. Teich, S. S. Bhattacharyya:
Optimizing the Efficiency of Parameterized Local Search within Global Search: A Preliminary Study.
In: Conf. Proc. CEC 2000, IEEE press, Piscataway, N.J., S.365-372, 2000.

Anhang A

Abstandsmaße

In Abschnitt 3.2.2.2 wurden drei Abstandsmaße eingeführt, für die hier der Nachweis ihrer Konformität mit den 4 Anforderungen an eine Metrik Δ erbracht wird. Diese Anforderungen lauten:

$$\Delta(AK_1, AK_2) = 0 \Leftrightarrow AK_1 = AK_2 \tag{A.1}$$

$$\Delta(AK_1, AK_2) \geq 0 \tag{A.2}$$

$$\Delta(AK_1, AK_2) = \Delta(AK_2, AK_1) \tag{A.3}$$

$$\Delta(AK_1, AK_3) \leq \Delta(AK_1, AK_2) + \Delta(AK_2, AK_3) \tag{A.4}$$

A.1 Parameterabstand

Der Parameterabstand zweier Aktionsketten AK_1 und AK_2, die beide nicht leer sind, $\Delta_{par}(AK_1, AK_2)$ wurde für alle Parameter des Handlungsmodells, für die die Obergrenze des Wertebereichs größer als die Untergrenze ist ($og_i > ug_i$), in Gl. (3.1) wie folgt definiert:

Parameterabstand:

$$\Delta_{par}(AK_1, AK_2) = \frac{1}{anz} \sum_{i=1}^{anz} \frac{|param_{i,1} - param_{i,2}|}{og_i - ug_i}$$

mit: $param_{i,j}$: Wert des i-ten Parameters in der Kette AK_j. Die Parameter werden dabei in der Reihenfolge ihrer Definition im Aktionsmodell durchnumeriert.

ug_i, og_i: Unter- und Obergrenze des Wertebereichs des i-ten Parameters.

anz: Anzahl aller Parameter aller Aktionen.

Forderung A1:
Unter der Gleichheit zweier Aktionsketten AK_1 und AK_2 wird hier die Gleichheit ihrer korrespondierenden Parameter verstanden. Da $anz > 0$ und $og_i > ug_i$ folgt aus der Tatsache, daß eine Summe positiver Summenaden genau dann 0 ist, wenn ihre Summanden 0 sind, für alle $i = 1,...,anz$:

$$\Delta_{par}(AK_1, AK_2) = 0 \Rightarrow |param_{i,1} - param_{i,2}| = 0 \Rightarrow param_{i,1} = param_{i,2} \Rightarrow AK_1 = AK_2 \tag{A1.1}$$

$$AK_1 = AK_2 \Rightarrow param_{i,1} = param_{i,2} \Rightarrow |param_{i,1} - param_{i,2}| = 0 \Rightarrow \Delta_{par}(AK_1, AK_2) = 0 \tag{A1.2}$$

Aus (A1.1) und (A1.2) folgt die Erfüllung von (A.1).

Forderung A2:
Aus $anz > 0$, $og_i - ug_i > 0$ und $|param_{i,1} - param_{i,2}| \geq 0$ folgt, daß $\Delta_{par}(AK_1, AK_2) \geq 0$. Damit ist (A.2) erfüllt.

Forderung A3:
Ein Vertauschen der Aktionsketten bei der Abstandsbestimmung wirkt sich lediglich auf die Bildung des Betrags $|param_{i,1} - param_{i,2}|$ aus:

$$|param_{i,1} - param_{i,2}| = |param_{i,2} - param_{i,1}| \Rightarrow \Delta_{par}(AK_1, AK_2) = \Delta_{par}(AK_2, AK_1)$$

Damit ist (A.3) erfüllt.

Forderung A4:
Es muß gezeigt werden, daß

$$\frac{1}{anz}\sum_{i=1}^{n} \frac{|param_{i,1} - param_{i,3}|}{og_i - ug_i} \leq \frac{1}{anz}\sum_{i=1}^{n} \frac{|param_{i,1} - param_{i,2}|}{og_i - ug_i} + \frac{1}{anz}\sum_{i=1}^{n} \frac{|param_{i,2} - param_{i,3}|}{og_i - ug_i}$$

Wenn diese Relation für jeden Summanden der Summe gilt, dann gilt sie auch für die Summe. Also genügt es zu zeigen, daß für jeden Parameter $param_i$ gilt

$$\frac{|param_1 - param_3|}{og - ug} \leq \frac{|param_1 - param_2|}{og - ug} + \frac{|param_2 - param_3|}{og - ug}$$

oder

$$|param_1 - param_3| \leq |param_1 - param_2| + |param_2 - param_3| \qquad \text{(A1.3)}$$

Um dies zu zeigen, sind 6 Fälle zu unterscheiden:

Fall 1: $param_1 \leq param_2 \leq param_3$

$$\begin{aligned} & param_3 - param_1 \leq param_2 - param_1 + param_3 - param_2 \\ \Leftrightarrow \quad & param_3 - param_1 \leq param_3 - param_1 \end{aligned}$$ q.e.d.

Fall 2: $param_1 \geq param_2 \geq param_3$

$$\begin{aligned} & param_1 - param_3 \leq param_1 - param_2 + param_2 - param_3 \\ \Leftrightarrow \quad & param_1 - param_3 \leq param_1 - param_3 \end{aligned}$$ q.e.d.

Fall 3: $param_1 \leq param_2, \quad param_3 \leq param_2, \quad param_1 \leq param_3$

$$\begin{aligned} & param_3 - param_1 \leq param_2 - param_1 + param_2 - param_3 \\ \Leftrightarrow \quad & param_3 \leq 2 \cdot param_2 - param_3 \\ \Leftrightarrow \quad & param_3 \leq param_2 \end{aligned}$$ q.e.d.

Fall 4: $param_1 \leq param_2, \quad param_3 \leq param_2, \quad param_3 \leq param_1$

$$\begin{aligned} & param_1 - param_3 \leq param_2 - param_1 + param_2 - param_3 \\ \Leftrightarrow \quad & param_1 \leq 2 \cdot param_2 - param_1 \\ \Leftrightarrow \quad & param_1 \leq param_2 \end{aligned}$$ q.e.d.

Fall 5: $param_2 \leq param_1, \quad param_2 \leq param_3, \quad param_3 \leq param_1$

$$\begin{aligned} & param_1 - param_3 \leq param_1 - param_2 + param_3 - param_2 \\ \Leftrightarrow \quad & -param_3 \leq -2 \cdot param_2 + param_3 \\ \Leftrightarrow \quad & param_3 \geq param_2 \end{aligned}$$ q.e.d.

Fall 6: $param_2 \le param_1, \quad param_2 \le param_3, \quad param_1 \le param_3$

$$\begin{aligned} & param_3 - param_1 \le param_1 - param_2 + param_3 - param_2 \\ \Leftrightarrow \quad & -param_1 \le param_1 - 2 \cdot param_2 \\ \Leftrightarrow \quad & param_1 \ge param_2 \end{aligned}$$ q.e.d.

Daraus folgt die Gültigkeit von (A1.3) und damit ist (A4) erfüllt.

A.2 Positionsabstand

Der Positionsabstand zweier Ketten $\Delta_{pos}(AK_1,AK_2)$, deren Länge größer als 1 ist, wurde in (3.2) und (3.3) wie folgt definiert:

Positionsabstand:

$$\Delta_{pos}(AK_1, AK_2) = \frac{1}{abst_{max}} \sum_{i=1}^{len} PA_{1,2}(A_i)$$

$$PA_{1,2}(A_i) = \left| I_1(A_i) - I_2(A_i) \right|$$

mit: *len*: Länge der Aktionsketten, entspricht der Anzahl der Aktionen (*len* > 1).

$abst_{max}$: Maximaler Abstand aller Aktionen einer Kette. Er ist gemäß den Gl. (3.10) und (3.13) abhängig von der Länge *len* der Kette und für *len* > 1 größer als 0.

$I_j(A_i)$: Index der Aktion *i* in der Aktionskette *j*.

Forderung A1:
Unter der Gleichheit zweier Aktionsketten AK_1 und AK_2 wird hier gleiche Position ihrer korrespondierenden Aktionen unabhängig von deren Parameterwerten verstanden. Da $abst_{max} > 0$ und eine Summe positiver Summenaden genau dann 0 ist, wenn ihre Summanden 0 sind, folgt:

$$\Delta_{pos}(AK_1, AK_2) = 0 \Rightarrow \left| I_1(A_i) - I_2(A_i) \right| = 0 \quad \forall i = 1, \ldots, len \Rightarrow AK_1 = AK_2 \qquad \text{(A1.4)}$$

$$AK_1 = AK_2 \Rightarrow \left| I_1(A_i) - I_2(A_i) \right| = 0 \quad \forall i = 1, \ldots, len \Rightarrow \Delta_{pos}(AK_1, AK_2) = 0 \qquad \text{(A1.5)}$$

Aus den Gl. (A1.4) und (A1.5) folgt die Erfüllung von (A.1).

Forderung A2:
Da die Summanden $PA_{1,2}(A_i)$ defintionsgemäß alle größer oder gleich 0 sind und $abst_{max} > 0$ ist, gilt auch $\Delta_{pos}(AK_1, AK_2) \ge 0$. Damit ist (A.2) erfüllt.

Forderung A3:
Ein Vertauschen der Aktionsketten bei der Abstandsbestimmung wirkt sich lediglich auf die Bildung des Betrags $\left| I_1(A_i) - I_2(A_i) \right|$ aus:

$$\left| I_1(A_i) - I_2(A_i) \right| = \left| I_2(A_i) - I_1(A_i) \right| \Rightarrow \Delta_{pos}(AK_1, AK_2) = \Delta_{pos}(AK_2, AK_1)$$

Damit ist (A.3) erfüllt.

Forderung A4:
Es muß gezeigt werden, daß

$$\frac{1}{abst_{max}}\sum_{i=1}^{len}\left|I_1(A_i)-I_3(A_i)\right| \le \frac{1}{abst_{max}}\sum_{i=1}^{len}\left|I_1(A_i)-I_2(A_i)\right| + \frac{1}{abst_{max}}\sum_{i=1}^{len}\left|I_2(A_i)-I_3(A_i)\right|$$

Wenn diese Relation für jeden Summanden der Summe gilt, dann gilt sie auch für die Summe. Also genügt es zu zeigen, daß für jede Aktion A_i gilt:

$$\left|I_1(A_i)-I_3(A_i)\right| \le \left|I_1(A_i)-I_2(A_i)\right| + \left|I_2(A_i)-I_3(A_i)\right|$$

Da diese Relation bereits in Abschn. A.1, (A1.3) für allgemeine Parameter bewiesen wurde, gilt sie auch für Variable > 0, wie sie die Positionsindizes der Aktionen $I_j(A_i)$ darstellen. Damit ist (A.4) erfüllt.

A.3 Unterschied der Aktionspräsenz

Nach (3.11) bewertet $\Delta_{akt}(AK_1, AK_2)$ den Unterschied der Aktionspräsenz zweier nichtleerer Aktionsketten AK_1 und AK_2:

Unterschied der Aktionspräsenz:

$$\Delta_{akt}(AK_1, AK_2) = 1 - \frac{card(A_{gem}(AK_1, AK_2))}{max(len(AK_1), len(AK_2))} \qquad \text{Ketten nicht leer und } A_{gem} \neq \varnothing$$

mit: $len(AK_i)$: Länge der Aktionskette AK_i, entspricht der Anzahl ihrer Aktionen.
$A_{gem}(AK_i,AK_j)$: Menge der gemeinsamen Aktionen der beiden Ketten AK_i und AK_j.
$card(A)$: Anzahl der Elemente der Menge A.

Forderung A1:
Unter der Gleichheit zweier Aktionsketten AK_1 und AK_2 wird hier die Identität der Mengen verstanden, die sich jeweils aus ihren Aktionen bilden lassen. Es sei A_{AKi} die Menge der Aktionen der Aktionskette AK_i.

$$\Delta_{akt}(AK_1, AK_2) = 0 \Rightarrow \frac{card(A_{gem})}{max(len(AK_1), len(AK_2))} = 1 \Rightarrow card(A_{gem}) = max(len(AK_1), len(AK_2)) \quad \text{(A1.6)}$$

Daß aus der rechten Gleichung von (A1.6) die Gleichheit der beiden Aktionsketten AK_1 und AK_2 folgt, wird durch indirekten Beweis nachgewiesen, indem angenommen wird, daß sich beide Ketten unterscheiden:

Annahme: $card(A_{gem}) = max(len(AK_1), len(AK_2)) \Rightarrow AK_1 \neq AK_2$
d.h.: $\exists a \in A_{AK1}$: a $\notin A_{AK2}$
$\Rightarrow len(AK_1) > card(A_{gem})$
$\Rightarrow max((len(AK_1), len(AK_2)) \ge len(AK_1) > card(A_{gem}))$ W!

Der umgekehrte Fall, nämlich daß in AK_2 eine Aktion enthalten ist, die nicht in AK_1 vorkommt, wird genauso behandelt. Damit ist nachgewiesen, daß gilt:

$$\Delta_{akt}(AK_1, AK_2) = 0 \Rightarrow AK_1 = AK_2 \tag{A1.7}$$

Bleibt noch zu beweisen, daß aus der Identität von AK_1 und AK_2 folgt, daß $\Delta_{akt}(AK_1, AK_2) = 0$ ist:

$$AK_1 = AK_2 \Rightarrow A_{AK1} = A_{AK2} = A_{gem} \Rightarrow max(len(AK_1),len(AK_2)) = card(A_{gem})$$

$$\Rightarrow \frac{card(A_{gem})}{max(len(AK_1), len(AK_2))} = 1 \Rightarrow \Delta_{akt}(AK_1, AK_2) = 0 \tag{A1.8}$$

Aus (A1.7) und (A1.8) folgt die Erfüllung von (A.1).

Forderung A2:
$\Delta_{akt}(AK_1, AK_2) \geq 0$ wenn $1 \geq \frac{card(A_{gem})}{max(len(AK_1), len(AK_2))}$ oder $max(len(AK_1), len(AK_2)) \geq card(A_{gem})$.

Für identische Ketten gilt die Gleichheit der beiden Terme und für nicht identische gilt die Relation. Damit ist die Erfüllung von (A.2) nachgewiesen.

Forderung A3:
Da sich ein Vertauschen der Aktionsketten bei der Bestimmung des Unterschieds der Aktionspräsenz gemäß (3.11) nicht auf $\Delta_{akt}(AK_1, AK_2)$ auswirkt, ist (A.3) erfüllt.

Forderung A4:
Es muß gezeigt werden, daß

$$1 - \frac{card(A_{gem}(AK_1, AK_3))}{max(len(AK_1), len(AK_3))} \leq 1 - \frac{card(A_{gem}(AK_1, AK_2))}{max(len(AK_1), len(AK_2))} + 1 - \frac{card(A_{gem}(AK_2, AK_3))}{max(len(AK_2), len(AK_3))}$$

$$1 - \frac{card(A_{AK1} \cap A_{AK3})}{max(card(A_{AK1}), card(A_{AK3}))} \leq 2 - \frac{card(A_{AK1} \cap A_{AK2})}{max(card(A_{AK1}), card(A_{AK2}))} - \frac{card(A_{AK2} \cap A_{AK3})}{max(card(A_{AK2}), card(A_{AK3}))} \tag{A1.9}$$

Aus Symmetriegründen kann angenommen werden, daß $card(AK_1) \geq card(AK_3)$. Damit sind folgende 3 Fälle zu unterscheiden.

<u>Fall 1:</u> $card(AK_2) \geq card(AK_1) \geq card(AK_3)$

Es ist zu zeigen:

$$1 - \frac{card(A_{AK1} \cap A_{AK3})}{card(A_{AK1})} \leq 2 - \frac{card(A_{AK1} \cap A_{AK2})}{card(A_{AK2})} - \frac{card(A_{AK2} \cap A_{AK3})}{card(A_{AK2})}$$

$$-1 - \frac{card(A_{AK1} \cap A_{AK3})}{card(A_{AK1})} \leq - \frac{card(A_{AK1} \cap A_{AK2}) + card(A_{AK2} \cap A_{AK3})}{card(A_{AK2})}$$

$$\frac{card(A_{AK1} \cap A_{AK2}) + card(A_{AK2} \cap A_{AK3})}{card(A_{AK2})} \leq 1 + \frac{card(A_{AK1} \cap A_{AK3})}{card(A_{AK1})}$$

$$card(A_{AK1} \cap A_{AK2}) + card(A_{AK2} \cap A_{AK3}) \leq card(A_{AK2}) + card(A_{AK1} \cap A_{AK3}) \cdot \frac{card(A_{AK2})}{card(A_{AK1})} \tag{A1.10}$$

Die linke Seite von (A1.10) ist äquivalent mit

$$card((A_{AK1} \cup A_{AK3}) \cap A_{AK2}) + card((A_{AK1} \cap A_{AK3}) \cap A_{AK2})$$

Aus $(A_{AK1} \cup A_{AK3}) \cap A_{AK2} \subseteq A_{AK2}$ und $(A_{AK1} \cap A_{AK3}) \cap A_{AK2} \subseteq A_{AK1} \cap A_{AK3}$ folgt:

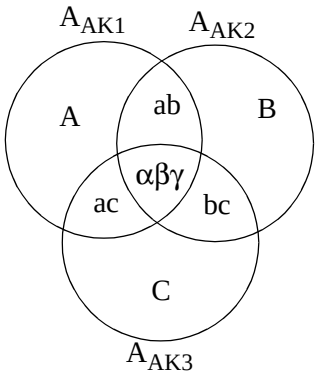
$$card(A_{AK1} \cap A_{AK2}) + card(A_{AK2} \cap A_{AK3}) \leq card(A_{AK2}) + card(A_{AK1} \cap A_{Ak3}) \leq$$

$$card(A_{AK2}) + card(A_{AK1} \cap A_{AK3}) \cdot \frac{card(A_{AK2})}{card(A_{AK1})} \quad \text{da} \quad \frac{card(A_{AK2})}{card(A_{AK1})} \geq 1$$

Damit ist (A1.10) bewiesen.

Fall 2: $card(AK_1) \geq card(AK_2) \geq card(AK_3)$

Zur Vereinfachung der Schreibweise werden für die unterschiedlichen Teilmengen der Mengen der 3 Aktionsketten folgende Benennungen eingeführt:



A, B, C, ab, ac, bc und αβγ bezeichnen die Anzahl der Elemente der entsprechenden Teilmengen, z.B.:

$card\ (A_{AK1})$ $= A + ab + ac + \alpha\beta\gamma$ oder

$card(A_{AK1} \cap A_{AK2})$ $= ab + \alpha\beta\gamma$ oder

$card(A_{AK1} \cap A_{AK2} \cap A_{AK3}) = \alpha\beta\gamma$

(A1.9) kann geschrieben werden als:

$$1 + \frac{card(A_{AK1} \cap A_{AK3})}{max(card(A_{AK1}),card(A_{AK3}))} \geq \frac{card(A_{AK1} \cap A_{AK2})}{max(card(A_{AK1}),card(A_{AK2}))} + \frac{card(A_{AK2} \cap A_{AK3})}{max(card(A_{AK2}),card(A_{AK3}))} \qquad \text{(A1.11)}$$

Mit den Voraussetzungen für diesen Fall und obiger Notation kann (A1.11) formuliert werden als:

$$1 + \frac{ac + \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} \geq \frac{ab + \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} + \frac{bc + \alpha\beta\gamma}{B + ab + bc + \alpha\beta\gamma}$$

$$\frac{A + ab + ac + \alpha\beta\gamma + ac + \alpha\beta\gamma - ab - \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} \geq \frac{bc + \alpha\beta\gamma}{B + ab + bc + \alpha\beta\gamma}$$

$$(A + 2ac + \alpha\beta\gamma) \cdot (B + ab + bc + \alpha\beta\gamma) \geq (bc + \alpha\beta\gamma) \cdot (A + ab + ac + \alpha\beta\gamma)$$

$$AB + Aab + 2Bac + 2ac \cdot ab + ac \cdot bc + ac \cdot \alpha\beta\gamma + B \cdot \alpha\beta\gamma \geq bc \cdot ab \qquad \text{(A1.12)}$$

Der Term $AB + Aab + Bac + ac \cdot ab$ ist sicher kleiner als die linke Seite der Ungleichung von (A1.12), da er kleiner als die vier ersten Summanden ist. Für diesen Term gilt:

$$AB + Aab + Bac + ac \cdot ab = (A + ac) \cdot (B + ab) \geq (B + bc) \cdot (B + ab) = B^2 + Bab + Bbc + bc \cdot ab \geq bc \cdot ab$$

Für die rechte Ungleichung wurde die Voraussetzung $card(AK_1) \geq card(AK_2)$ benutzt, aus der sich $A + ac \geq B + bc$ ableiten läßt. Damit ist (A.1.11) bewiesen, woraus die Gültigkeit von (A1.9) für diesen Fall folgt.

Fall 3: $card(AK_1) \geq card(AK_3) \geq card(AK_2)$

Für den Beweis dieses Falls wird auf die Notation von Fall 2 zurückgegriffen. Basierend auf (A1.11) ist damit zu beweisen, daß:

$$1 + \frac{ac + \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} \geq \frac{ab + \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} + \frac{bc + \alpha\beta\gamma}{C + ac + bc + \alpha\beta\gamma}$$

$$\frac{A + ab + ac + \alpha\beta\gamma + ac + \alpha\beta\gamma - ab - \alpha\beta\gamma}{A + ab + ac + \alpha\beta\gamma} \geq \frac{bc + \alpha\beta\gamma}{C + ac + bc + \alpha\beta\gamma}$$

$$(A + 2ac + \alpha\beta\gamma) \cdot (C + ac + bc + \alpha\beta\gamma) \geq (bc + \alpha\beta\gamma) \cdot (A + ab + ac + \alpha\beta\gamma)$$

$$\underline{AC + Aac + 2Cac + 2ac^2} + ac \cdot bc + \underline{\cdots C \cdot \alpha\beta\gamma + 2ac \cdot \alpha\beta\gamma} \geq bc \cdot ab + ab \cdot \alpha\beta\gamma \qquad \text{(A1.12)}$$

Die Summe der beiden unterstrichenen Teile der linken Seite der Ungleichung von (A1.12) ist kleiner oder gleich der kompletten linken Seite. Für die 4 linken Summanden (durchgezogene Linie) gilt:

$$AC + Aac + 2Cac + 2ac^2 \geq AC + Aac + Cac + ac^2 = (A + ac) \cdot (C + ac) \geq (B + bc) \cdot (B + ab) \geq bc \cdot ab \qquad \text{(A1.13)}$$

Dabei wurden die folgenden beiden Voraussetzungen dieses Falls $card(AK_1) \geq card(AK_2)$ und $card(AK_3) \geq card(AK_2)$ benutzt, aus denen sich $A + ac \geq B + bc$ und $C + ac \geq B + ab$ ableiten lassen.

Für den gepunktet unterstrichenen Teil gilt

$$C \cdot \alpha\beta\gamma + 2ac \cdot \alpha\beta\gamma \geq C \cdot \alpha\beta\gamma + ac \cdot \alpha\beta\gamma = \alpha\beta\gamma(C + ac) \geq \alpha\beta\gamma(B + ab) \geq \alpha\beta\gamma \cdot ab \qquad \text{(A1.14)}$$

Auch hier wurde die Voraussetzung $card(AK_3) \geq card(AK_2)$ benutzt. Mit (A1.14) und (A1.13) ist (A.12) bewiesen, woraus die Gültigkeit von (A1.9) für diesen Fall folgt.

Mit den Beweisen für alle 3 Fälle ist (A.4) erfüllt.

Anhang B

Experimente

Die Experimente wurden je nach Verfügbarkeit auf bis zu 19 Sun-Workstations der Typen Ultra-Sparc 1, 2, 5 und 10 mit CPU-Taktraten zwischen 140 und 440 MHz durchgeführt. Darunter befanden sich drei Doppelprozessormaschinen, sodaß bis zu 22 Prozessoren im Einsatz waren. Auf die Bedeutung und Problematik der Erfassung der unterschiedlichen Laufzeiten wurde bereits in Abschnitt 5.2 hingewiesen. Die Umrechnung erfolgte lediglich auf der Basis der Taktzeiten, was z.B. unterschiedliche LAN-Anbindungen nicht berücksichtigt. Letzteres war auch nicht möglich, da die Netzhardware im Laufe der Versuche mehrmals umgebaut wurde und auch sonst temporäre Schwankungen durch die normale Benutzung der Maschinen durch ihre Besitzer zu verzeichnen waren. Alle in den Ergebnis-Tabellen enthaltenen Zeitangaben werden in CPU-Stunden bezogen auf eine Ultra-Sparc 1 mit 170 MHz angegeben (Spalte *Zeit[h] Ultra1*).

Der Aufwand für die einzelnen Testaufgaben war sehr unterschiedlich, wie Tabelle B.1 zeigt. Ein Job faßt die Läufe zu einer Testaufgabe mit einem konkreten Verfahren oder Verfahrenskombination und Parametrierung zusammen.

Testaufgabe	Jobs	Läufe	CPU-Tage	CPU-Std. pro Job
Sphere	212	16304	422.3	47.8
Foxholes	374	37800	2.4	0.2
Rastrigin	274	24584	305.8	26.8
Fletcher	377	36900	24.3	1.6
Fractal	398	35552	284.8	17.2
Designoptimierung	281	13633	254.1	21.7
Ressourcenoptimierung	136	6179	870.1	153.6
Roboterbahnplanung	123	8135	419.8	81.9
Kalibrierungs- u. Fehlläufe	47	3445	61.8	
Rotierte Benchmarkfunktionen	513	49453	727.4	34.0
Summen	2735	231985	3372.8	

Tab. B.1: Verteilung des Rechenzeitaufwands von 9.24 CPU-Jahren auf die einzelnen Testaufgaben. Aufgaben, deren Jobs ganz oder teilweise aus nur 50 Läufen bestehen, sind kursiv geschrieben.

Bei kleinen Populationen wurde bei GLEAM die Deme-Größe von ihrem Standardwert acht reduziert und zwar auf sechs bei einer Populationsgröße von zehn und auf vier bei einer Größe von fünf. Das bedeutet, daß es sich im letzteren Fall um eine panmiktische Population handelt.

Beim Rosenbrock-Verfahren kann das Konvergenzverhalten durch die Vorgabe einer Abbruchschranke beeinflußt werden. Tabelle B.2 gibt die verwendeten Einstellungen an, wobei

zu beachten ist, daß die Parameterwertebereiche bei der verwendeten Implementierung auf einen festen Bereich zwischen Null und Eins normiert werden. Neben diesen fünf Größen gibt es noch die Sonderparametrierung *s*, die im Falle ihrer Verwendung bei den entsprechenden Testaufgaben näher angegeben wird.

Bezeichnung	Abkürzung	Wert
niedrig	n	10^{-2}
mittel	m	10^{-4}
hoch	h	10^{-6}
sehr hoch	u	10^{-8}
extrem hoch	v	10^{-9}

Tab. B.2: Abbruchschranken (Präzisionen) für das Rosenbrock-Verfahren

Für die Bezeichnungen der Jobs finden folgende Abkürzungen Verwendung:

Abkürzungen für die drei Verfahren und die neun Hybridisierungsarten:

G	GLEAM
R	Rosenbrock-Verfahren
C	Complex-Algorithmus
Ri	GLEAM mit Rosenbrock-initialisierter Startpopulation
Ci	GLEAM mit Complex-initialisierter Startpopulation
NR	GLEAM plus Nachoptimierung mit dem Rosenbrock-Verfahren
NC1P	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei GLEAM meist mehrfach einen Startpunkt liefert.
NC1C	GLEAM plus Nachoptimierung mit dem Complex-Algorithmus, wobei GLEAM einen Startcomplex liefert.
GR	GLEAM mit direkter Integration des Rosenbrock-Verfahrens
GvR	GLEAM mit verzögerter direkter Integration des Rosenbrock-Verfahrens
GC	GLEAM mit direkter Integration des Complex-Verfahrens
GvC	GLEAM mit verzögerter direkter Integration des Complex-Verfahrens

Abkürzungen für die Parametrierungen:

p<anz>	Populationsgröße <anz>
n,m,h, u,v,s	Abbruchschranken für das Rosenbrock-Verfahren: n=niedrig, m=mittel, h=hoch, u=sehr hoch, v=extrem hoch, s=Sonderparametrierung, die entsprechenden Werte sind beim jeweiligen Experiment aufgeführt
<x>%	bei Ri oder Ci: Anteil der mit dem LSV voroptimierten Individuen an der Startpopulation in %
P1,P2,P3	Parametersätze 1 bis 3 zur Steuerung der Nachoptimierung und der verzögerten direkten Integartion, Werte siehe Tabelle B.3

N<anz>	maximale Nischenanzahl N_{max}, sofern von Tabelle B.4 abweichend
G1,G3	Überprüfung der Nischenbildung bei GDV- und GAk-Werten von 1 bzw. 3
all,best	Nachoptimierung aller oder nur des besten Nachkommens bei der direkten Integration
L100,L5,L0	Lamarckanteil 100%, 5% oder 0% (Baldwin-Evolution). Wegen der besseren Unterscheidbarkeit von der 1 wird ein L statt dem bisherigen l benutzt.

Parametrierung	Limit für den genotypischer Unterschied zweier Individuen ε	Limit für den genotypischen Unterschied zweier Nischenrepräsentanten ε_{Pop}
P1	0.005	0.01
P2	0.002	0.005
P3	0.001	0.003

Tab. B.3: ε und ε_{Pop} der drei Parametrierungen P1, P2 und P3 zur Steuerung der Nachoptimierung und der verzögerten direkten Integration

Bei der Steuerung der Nachoptimierung und der verzögerten direkten Integration spielt noch die maximale Nischenanzahl N_{max} eine Rolle, die in Abhängigkeit von der Populationsgröße gemäß Tabelle B.4 eingestellt wird. Abweichungen werden in der Job-Kennung vermerkt.

Populationsgröße	maximale Nischenanzahl N_{max}
< 20	2
20 - 40	3
50 - 90	4
> 90	5

Tab. B.4: Maximale Nischenanzahl N

Die Spaltenüberschriften haben, soweit nicht selbsterklärend, folgende Bedeutung:

Erfolg	Erfolgsrate in Prozent. Wenn weniger Läufe durchgeführt wurden als der Standardwert für die jeweilige Testaufgabe, erfolgt die Angabe in absoluten Zahlen in der Form <erfolg> / <gesamt>.
Ros/Com	LSV-Erfolgsanteil bei Ri- oder Ci-Läufen.
RS	ReStart der Läufe eines Jobs wegen Nichtkonvergenz des LSV.
Gen	Generationen
Schn	Durchschnitt (Noten-, Generationen- oder Individuendurchschnitt)
Indivs	Bewertete Individuen; Evaluationen
GutSchn	Durchschnitt bezogen auf die erfolgreichen Läufe eines Jobs
GutVari	Varianz *s,* basierend auf den erfolgreichen Läufen eines Jobs

Nisch. GDV/GAk	Diese Spalte enthält zwei Angaben: 1.Zahl: Anzahl der Läufe, bei denen nachoptimiert bzw. zugeschaltet wurde. Wenn diese Zahl geringer als die Anzahl der Läufe des Jobs ist, terminierte beim Rest die Evolution entweder wegen Erfolg oder Stagnation. 2.Zahl: Anzahl der Läufe bei denen die Nachoptimierung bzw. Umschaltung wegen Stagnation (Erreichen des GAk/GDV-Limits, siehe unten) erfolgte.
Noten-Verb.	Notenverbesserung durch die Nachoptimierung
Nopt Erf	Nachoptimierungserfolg, Angabe wie *Erfolg* in Prozent.
Ni.Schn	Durchschnitt der Nischenanzahl beim Evolutionsabbruch (Nachoptimierung)

Alle Jobs der verzögerten direkten Integration wurden, soweit nicht anders angegeben, mit reiner Lamarckscher Evolution (*L100*) und der lokalen Verbesserung des besten Nachkommens (*best*) durchgeführt.

Alle Läufe wurden wegen Stagnation abgebrochen, wenn seit 1000 Generationen keine Demeverbesserung eintrat (GDV), oder seit 500 Generationen kein Nachkomme akzeptiert wurde (GAk). Das Ereichen der Hälfte dieser Werte begründet zusätzlich die Erfüllung des Nischenkriteriums für Nachoptimierung oder LSV-Zuschaltung.

B.1 Schwefel's Sphere

Es wurden, soweit nicht anders vermerkt, 100 Läufe pro Job durchgeführt bzw. 50 bei allen Complex-Jobs und bei direkter Integration mit dem Rosenbrock-Verfahren bei allen anderen Parametrierungen als *best* und *L100*.

B.1.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Es wurden zwei GLEAM-Läufe durchgeführt, die beide vor Erreichen der Zielfitness abgebrochen werden mußten. Der erste lief bei einer Populationsgröße von 30 390000 Generationen, führte dabei 26.13 Mio Evaluationen durch und erreichte einen Notenwert von 99979. Der zweite arbeitete mit der 10-fachen Populationsgröße, lief 330000 Generationen mit 220.1 Mio Evaluationen und lieferte einen Notenwert von 99968. Die erreichten Noten entsprechen einem Funktionswert von ca. 7500 und liegen damit noch deutlich vom Zielwert 0.01 entfernt.

Job	Er-	Zeit[h]	Noten	---------- Indivs ----------			
	folg	Ultra1	Schn	GutSchn	Schn	Min	Max	GutVari
R_n	0		59868	0	1267	949	1778	0
R_m	0		60435	0	2286	1802	2748	0
R_h	0		97969	0	3686	2938	4378	0
R_u	54		99999	5607	5404	4408	6393	426
R_v	100		100000	6440	6440	5257	7655	474
C	0	2.185	24806	0	52610	30706	104292	0

B.1.2 Vorinitialisierte Startpopulationen

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_h_p30_100%	1/10		99998	300000	20182312	20178793	20186995	0
Ri_h_p50_100%	1/10		99999	300000	33634986	33630412	33641280	0
Ci_p10_10%	0/10	2.56	99830	212324	4779400	3466228	5764399	0
Ci_p10_20%	0/10	5.37	99874	237138	5386516	3933947	7322644	0
Ci_p10_30%	0/10	6.03	99885	247983	5673409	3579166	6779541	0
Ci_p10_100%	0/10	19.9	99755	207492	5127592	3550385	7308571	0
Ci_p20_5%	0/10	3.48	99939	308906	13804741	9383927	20426965	0
Ci_p20_10%	0/10	4.78	99914	280093	12560638	7778317	16508543	0
Ci_p20_20%	0/10	8.85	99916	267922	12122559	7858483	15791370	0
Ci_p30_5%	0/10	6.36	99961	369858	24804572	15900598	33529176	0
Ci_p30_10%	0/10	7.99	99966	389152	26140602	16206219	33598558	0

B.1.3 Nachoptimierung

Job	Er-		Noten-	Nopt	Gen	Indivs			
	folg	Schn	Verb.	Erf	Schn	Schn	Min	Max	GutVari
NR_p10_u_P1_G1	81	99999.9	76780	81	37	12194	6397	19272	2811
NR_p20_u_P1_G1	76	99999.9	75341	76	61	16131	9212	24466	2152
NR_p30_u_P1_G1	80	100000	75109	80	82	20820	15370	28006	2461
NR_p30_u_P1_G1	81	100000	75115	81	81	20698	13060	25504	2555
NR_p50_u_P1_G1	78	99999.9	75026	78	114	32408	21077	42029	3857
NR_p70_u_P1_G1	79	100000	75010	79	153	48347	30315	69630	7384
NR_p90_u_P1_G1	80	100000	75006	80	199	69207	48124	101248	12867
NR_p120_u_P1_G1	84	100000	75003	84	284	111521	59232	295127	37704
NR_p150_u_P1_G1	80	100000	75002	80	523	217351	93394	682474	104727
NR_p200_u_P1_N6_G1	76	99999.9	71065	76	6357	2847066	138821	10651989	3526168
NR_p250_u_P1_N7_G1	79	100000	58287	79	19886	10973939	162544	13971106	4256449
NR_p10_u_P2_G1	81	99999.9	76247	81	45	12934	5499	19682	2427
NR_p20_u_P2_G1	81	99999.9	75287	81	65	17136	10166	28324	3065
NR_p30_u_P2_G1	79	100000	75087	79	86	21881	14366	32521	3017
NR_p50_u_P2_G1	82	99999.9	75023	82	117	33006	22486	44323	4376
NR_p70_u_P2_G1	74	100000	75011	74	147	47725	33121	68061	6458
NR_p90_u_P2_G1	86	100000	75006	86	195	68380	43946	114973	12562
NR_p120_u_P2_G1	78	99999.9	75003	78	284	112303	66703	181476	26496
NR_p150_u_P2_G1	73	99999.9	75001	73	489	205738	94287	486466	85169
NR_p200_u_P2_N6_G1	81	100000	70116	81	7213	3222558	144416	10601573	4046260
NR_p250_u_P2_N7_G1	87	100000	58623	87	19505	10765726	196114	13897468	4461705
NR_p10_u_P3_G1	73	99999.9	76115	73	48	12688	5435	19478	2543
NR_p20_u_P3_G1	83	100000	75198	83	72	18494	10876	27748	3945
NR_p30_u_P3_G1	87	100000	75091	87	88	24079	13503	35327	4403
NR_p50_u_P3_G1	90	100000	75024	90	121	39538	20719	56636	7217
NR_p70_u_P3_G1	93	100000	75011	93	156	55475	35976	97386	9504
NR_p90_u_P3_G1	92	100000	75005	92	205	77503	56271	115352	13618
NR_p120_u_P3_G1	95	100000	75003	95	327	135137	67338	236604	34468
NR_p150_u_P3_G1	97	100000	75002	97	573	247813	85101	661524	115741
NR_p200_u_P3_N6_G1	89	100000	69672	89	6854	3076023	178268	10528882	3572755
NR_p250_u_P3_N7_G1	80	100000	57637	80	20209	11262921	312897	24057920	4555128
NR_p10_u_P1_G3	80	99999.9	75283	80	112	14127	6688	17412	1966
NR_p20_u_P1_G3	77	100000	75047	77	156	21156	10903	31723	3022
NR_p30_u_P1_G3	76	100000	75015	76	225	31973	23879	51416	5745
NR_p50_u_P1_G3	67	99999.9	75003	67	602	88512	34467	429468	50142
NR_p70_u_P1_G3	72	99999.9	70558	72	7100	1117898	49189	3966276	1404661
NR_p90_u_P1_G3	79	100000	57790	79	20702	4118008	86845	5378759	1536690
NR_p120_u_P1_G3	79	100000	52502	79	24401	6460614	5723075	7199118	290191
NR_p10_u_P2_G3	76	100000	75283	76	106	14118	7091	20474	2246
NR_p20_u_P2_G3	72	99999.9	75052	72	159	21229	14493	30107	2990
NR_p30_u_P2_G3	78	99999.9	75017	78	219	31321	20329	51349	6055
NR_p50_u_P2_G3	80	99999.9	75003	80	530	80431	33913	247947	34760
NR_p70_u_P2_G3	81	100000	71674	81	6097	969498	72855	3806693	1220800
NR_p90_u_P2_G3	76	99999.9	57452	76	20676	4112697	70723	5357906	1462805
NR_p120_u_P2_G3	81	100000	54535	81	23989	6352737	157684	7393641	324240
NR_p10_u_P3_G3	72	99999.9	75289	72	108	14190	7538	19778	1848
NR_p20_u_P3_G3	78	99999.9	75053	78	148	21097	12730	30876	3324
NR_p30_u_P3_G3	83	100000	75013	83	245	34406	15849	57867	7518
NR_p50_u_P3_G3	85	100000	75003	85	563	87193	32539	241506	40623
NR_p70_u_P3_G3	84	100000	69693	84	7013	1108845	59764	3874139	1338418
NR_p90_u_P3_G3	75	99999.9	56414	75	21596	4294993	83136	5417713	1395772
NR_p120_u_P3_G3	86	100000	54686	86	24539	6497393	5828373	7345255	309994

Variationen der anfänglichen *step size* des Rosenbrock-Verfahrens von 0.1 bei x, P3 und G1:

Job	step size	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NR_p120	0.01	94	100/0	0.079	100000	75003	94	310	126538	66827	263273
NR_p150_N6	0.01	98	100/0	0.110	100000	75002	98	599	252507	93632	847190
NR_p200_N7	0.01	92	100/0	0.575	100000	68941	92	7051	3159508	153559	10432836
NR_p120	0.001	89	100/0	0.073	100000	75003	89	316	125661	68664	234430
NR_p150_N6	0.001	95	100/0	0.101	100000	75002	95	685	278096	94615	859004
NR_p200_N7	0.001	92	100/0	0.578	100000	69507	92	7699	3440854	159597	10906501

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1P_p10_P1_G1	0	1.2	50/0	3.493	24959	1655	0	38	95885	35486	185429
NC1P_p20_P1_G1	0	1.1	50/0	3.61	24986	305	0	61	103238	44237	173625
NC1P_p30_P1_G1	0	1.0	50/0	3.462	25121	201	0	83	103480	44778	277398
NC1P_p50_P1_G1	0	1.0	50/0	3.143	25324	348	0	117	109074	69598	267819
NC1P_p10_P2_G1	0	1.3	50/0	3.664	24954	1512	0	43	104190	44076	177025
NC1P_p20_P2_G1	0	1.4	50/0	3.652	24988	227	0	66	111004	49424	192071
NC1P_p30_P2_G1	0	1.4	50/0	4.103	25008	105	0	83	123815	76227	200499
NC1P_p50_P2_G1	0	1.2	50/0	3.845	25899	927	0	116	123680	50492	271305
NC1P_p10_P3_G1	0	1.4	50/0	3.626	24955	1132	0	48	100311	43026	246969
NC1P_p20_P3_G1	0	1.3	50/0	3.606	24989	185	0	75	106758	39831	192656
NC1P_p30_P3_G1	0	1.6	50/0	4.142	24996	80	0	89	126653	66486	287222
NC1P_p50_P3_G1	0	2.1	50/0	4.952	25983	1004	0	121	157093	86325	356005
NC1P_p10_P1_G3	0	1.0	50/0	2.95	24982	321	0	106	89087	36474	144029
NC1P_p20_P1_G3	0	1.0	50/0	3.2	24996	54	0	159	101543	43822	143696
NC1P_p30_P1_G3	0	1.0	50/0	4.54	27547	2560	0	245	125825	69613	273674
NC1P_p50_P1_G3	0	1.0	50/0	12.982	40104	15107	0	516	280104	103301	525922
NC1P_p10_P2_G3	0	1.0	50/0	3.24	24984	253	0	101	99324	32005	165240
NC1P_p20_P2_G3	0	1.0	50/0	3.11	25057	112	0	152	100127	43319	251042
NC1P_p30_P2_G3	0	1.0	50/0	4.312	26558	1572	0	224	124721	59145	426063
NC1P_p50_P2_G3	0	1.0	50/0	11.895	37854	12857	0	572	283945	85134	523824
NC1P_p10_P3_G3	0	1.1	50/0	3.217	24986	286	0	112	95622	35265	196362
NC1P_p20_P3_G3	0	1.2	50/0	3.969	25102	161	0	164	115658	51115	247723
NC1P_p30_P3_G3	0	1.2	50/0	4.196	27009	2026	0	220	125970	47259	281181
NC1P_p50_P3_G3	0	1.8	50/0	15.839	40096	15099	0	524	351212	100348	706731

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1C_p10_P1_G1	0	1.2	50/0	1.81	24907	1572	0	38	50514	30232	118778
NC1C_p20_P1_G1	0	1.1	50/0	1.712	24976	283	0	61	53101	37327	86084
NC1C_p30_P1_G1	0	1.0	50/0	1.588	24994	72	0	83	55529	36412	105020
NC1C_p50_P1_G1	0	1.0	50/0	1.376	24998	20	0	117	63190	44898	97811
NC1C_p10_P2_G1	0	1.3	50/0	1.94	24992	1360	0	43	55015	32029	116850
NC1C_p20_P2_G1	0	1.4	50/0	1.603	24979	211	0	66	51642	32506	104182
NC1C_p30_P2_G1	0	1.4	50/0	1.612	24993	85	0	83	56421	36494	96128
NC1C_p50_P2_G1	0	1.2	50/0	2.54	25882	907	0	116	79042	47661	229638

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1C_p10_P3_G1	0	1.4	50/0	1.746	24919	1088	0	48	50508	32963	89525
NC1C_p20_P3_G1	0	1.3	50/0	1.575	24986	179	0	75	51502	33150	77408
NC1C_p30_P3_G1	0	1.6	50/0	1.586	24993	73	0	89	56005	31780	101171
NC1C_p50_P3_G1	0	2.1	50/0	2.195	26329	1348	0	121	75512	44818	235898
NC1C_p10_P1_G3	0	1.0	50/0	1.538	24977	313	0	106	48749	29745	82305
NC1C_p20_P1_G3	0	1.0	50/0	1.584	25322	378	0	159	56277	39136	100297
NC1C_p30_P1_G3	0	1.0	50/0	2.398	26455	1457	0	245	75621	41918	231403
NC1C_p50_P1_G3	0	1.0	50/0	6.224	38272	13274	0	516	174090	63850	380314
NC1C_p10_P2_G3	0	1.0	50/0	1.435	24974	242	0	104	48253	33448	77038
NC1C_p20_P2_G3	0	1.0	50/0	1.485	24995	48	0	152	54725	36100	109187
NC1C_p30_P2_G3	0	1.0	50/0	3.164	26781	1795	0	224	86189	38503	225753
NC1C_p50_P2_G3	0	1.0	50/0	4.925	37712	12715	0	572	166146	65311	394673
NC1C_p10_P3_G3	0	1.1	50/0	1.503	24980	277	0	112	48083	30738	89338
NC1C_p20_P3_G3	0	1.2	50/0	1.63	24996	54	0	164	58450	38059	97117
NC1C_p30_P3_G3	0	1.2	50/0	2.263	25517	533	0	220	74361	43695	233723
NC1C_p50_P3_G3	0	1.8	50/0	5.856	40682	15685	0	524	177379	51314	312904

Läufe mit Pa (ε=0.0005, ε_{Pop}=0.005), Pb (ε=0.001, ε_{Pop}=0.01), Pc (ε=0.0001, ε_{Pop}=0.001) und G3:

Job	Er- folg	Nischen Schn	GDV/	Zeit Ultr1	Noten- Schn	Verb.	NOpt Erf	Gen Schn	Indivs Schn	Min	Max
NC1P_p30_Pa_N5	0	3.5	50/0	8.415	28900	3917	0	222	235231	67853	709647
NC1P_p30_Pb_N5	0	1.2	50/0	3.78	26047	1063	0	217	122030	52471	313386
NC1P_p70_Pc_N3	0	3.2	49/1	8.93	98965	4645	0	45213	7181933	6674400	7791245
NC1P_p70_Pb_N3	0/10	2.8	10/0	18.27	78458	21119	0	24823	4154942	3754301	4412182
NC1C_p30_Pa_N5	0	3.5	50/0	2.443	27606	2622	0	222	77985	41965	233372
NC1C_p30_Pb_N5	0	1.2	50/0	2.009	26134	1148	0	217	71550	40785	225013
NC1C_p70_Pc_N3	0	3.2	49/1	3.434	99270	4485	0	45213	7042936	6535384	7757050
NC1C_p70_Pa_N3	0/10	2.8	10/0	8.213	87534	27534	0	24823	3932887	3657240	4165873

B.1.4 Direkte Integration

Job	Er- folg	Noten Schn	Gen Schn	GutSchn	Indivs Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	100000	9	54114	54114	17401	115543	21735
GR_p10_n_best_L100	50/50	100000	7	79499	79499	24812	150647	30576
GR_p20_n_best_L100	50/50	100000	5	122239	122239	29809	225717	51923
GR_p30_n_best_L100	50/50	100000	5	154374	154374	31992	316224	67118
GR_p5_n_best_L5	50/50	100000	102	606280	606280	144416	1788774	416762
GR_p10_n_best_L5	50/50	100000	59	659272	659272	115286	1432708	371311
GR_p20_n_best_L5	50/50	100000	47	988380	988380	198073	2349511	494269
GR_p30_n_best_L5	50/50	100000	31	984192	984192	277815	3184501	533443
GR_p20_n_best_L0	0/2	99971	1000	0	37478514	36732722	38224305	0

Job	Er-	Noten	Gen	Indivs				
	folg	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GR_p5_n_all_L100	50/50	100000	5	101212	101212	56084	226281	31670
GR_p10_n_all_L100	50/50	100000	4	171454	171454	89814	275329	46790
GR_p20_n_all_L100	50/50	100000	3	269953	269953	150144	454803	73388
GR_p30_n_all_L100	50/50	100000	3	371248	371248	241799	525849	70472
GR_p5_n_all_L5	20/20	100000	21	677213	677213	206479	1786289	382830
GR_p20_n_all_L0	0/2	99736	63	0	9956630	9846681	10066579	0
GR_p5_m_best_L100	100	100000	7	49871	49871	12335	107426	18119
GR_p10_m_best_L100	100	100000	6	78979	78979	21024	159177	29724
GR_p20_m_best_L100	100	100000	3	97980	97980	39802	214311	38871
GR_p30_m_best_L100	100	100000	3	134710	134710	63439	299902	57329
GR_p5_h_best_L100	100	100000	2	29595	29595	15502	42508	4757
GR_p10_h_best_L100	100	100000	2	48985	48985	28928	80370	11932
GR_p20_h_best_L100	100	100000	2	92810	92810	59617	122363	23635
GR_p30_h_best_L100	100	100000	1	127661	127661	91606	184823	34038
GR_p5_u_best_L100	100	100000	2	32348	32348	16785	46046	7387
GR_p10_u_best_L100	100	100000	1	56871	56871	38387	87777	15584
GR_p20_u_best_L100	100	100000	1	102725	102725	77134	163360	25780
GR_p30_u_best_L100	100	100000	1	139518	139518	116212	242279	22377
GR_p5_m_best_L10_Ri	100	100000	7	56101	56101	21262	127938	16807
GR_p10_m_best_L100_Ri	100	100000	6	95546	95546	53656	168231	22555
GR_p20_m_best_L100_Ri	100	100000	5	158222	158222	65545	261594	39614
GR_p30_m_best_L100_Ri	100	100000	5	227335	227335	121146	352622	45615
GR_p5_h_best_L100_Ri	100	100000	2	40455	40455	28415	56382	4470
GR_p10_h_best_L100_Ri	100	100000	2	77069	77069	56343	90093	9663
GR_p20_h_best_L100_Ri	100	100000	2	141131	141131	109028	171972	23731
GR_p30_h_best_L100_Ri	100	100000	1	205430	205430	165637	253925	36178

B.1.5 Verzögerte direkte Integration

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_m_P1	100	100/0	0.128	100000	23	51420	11470	110397	18873
GvR_p10_m_P1	100	100/0	0.185	100000	44	75287	22380	167353	29760
GvR_p20_m_P1	100	100/0	0.299	100000	62	112948	44211	228681	43811
GvR_p5_m_P2	100	100/0	0.118	100000	25	48008	12554	128136	19022
GvR_p10_m_P2	100	100/0	0.182	100000	50	75453	22077	137581	26502
GvR_p20_m_P2	100	100/0	0.355	100000	68	113428	42057	247180	45623
GvR_p5_m_P3	100	100/0	0.121	100000	28	49762	14104	121081	19213
GvR_p10_m_P3	100	100/0	0.183	100000	53	76012	22209	168105	26998
GvR_p20_m_P3	100	100/0	0.258	100000	73	103283	39130	214958	46748
GvR_p5_h_P1	100	100/0	0.89	100000	19	29136	12949	44908	6365
GvR_p10_h_P1	100	100/0	0.161	100000	41	53362	30543	82364	11002
GvR_p20_h_P1	100	100/0	0.27	100000	60	91021	62627	125165	22728

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_h_P2	100	100/0	0.09	100000	20	29454	15428	46533	5352
GvR_p10_h_P2	100	100/0	0.158	100000	49	52756	29983	78478	10680
GvR_p20_h_P2	100	100/0	0.274	100000	67	94226	63680	126167	22641
GvR_p5_h_P3	100	100/0	0.095	100000	20	28433	14829	46502	48890
GvR_p10_h_P3	100	100/0	0.172	100000	50	53004	27813	81705	11123
GvR_p20_h_P3	100	100/0	0.288	100000	71	94733	61820	124846	23564
GvR_p30_h_P3	100	100/0	0.368	100000	91	129543	95052	186272	33742
GvR_p5_u_P1	100	100/0	0.125	100000	19	33153	16413	45288	7753
GvR_p10_u_P1	100	100/0	0.241	100000	39	63450	33080	87367	15523
GvR_p20_u_P1	100	100/0	0.374	100000	64	108098	84188	171145	22167
GvR_p5_u_P2	100	100/0	0.121	100000	18	34253	15841	52883	8080
GvR_p10_u_P2	100	100/0	0.219	100000	48	62504	37756	89401	15420
GvR_p20_u_P2	100	100/0	0.374	100000	67	107854	82850	170250	21361
GvR_p5_u_P3	100	100/0	0.12	100000	19	33841	18488	47892	8096
GvR_p10_u_P3	100	100/0	0.219	100000	50	62766	40911	87819	15684
GvR_p20_u_P3	100	100/0	0.373	100000	72	107406	85374	171061	21854
GvR_p5_h_P3_Ri	100	100/0	0.135	100000	3	39475	28060	50917	4379
GvR_p10_h_P3_Ri	100	100/0	0.25	100000	3	76693	53311	87491	9659
GvR_p20_h_P3_Ri	100	100/0	0.469	100000	3	144922	105984	170159	23528
GvR_p30_h_P3_Ri	100	100/0	0.713	100000	2	204968	163202	255540	36262

B.2 Shekel's Foxholes

Es wurden100 Läufe pro Job durchgeführt.

B.2.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	Note	Indivs				
	folg	Schn	GutSchn	Schn	Min	Max	GutVari
G_p5	100	100000	1437	1437	283	9149	1406
G_p10	100	100000	1827	1827	473	6105	1044
G_p10	100	100000	1941	1941	325	6417	1292
G_p20	100	100000	2434	2434	739	7970	1526
G_p30	100	100000	2667	2667	554	6831	1426
G_p40	100	100000	3225	3225	1174	8538	1492
G_p50	100	100000	3346	3346	615	12256	1289
R_n	0	86924	0	60	38	90	0
R_m	0	86795	0	128	86	177	0
R_h	3	89340	338	257	141	516	129
R_u	aufgegeben (5 Laeufe ohne Konvergenz)						
C	1	51513	193	105	79	339	0

B.2.2 Vorinitialisierte Startpopulationen

Job	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_p5_n_20%	100	4	0.0002	100000	120	1454	52	5501	1220
Ri_p5_n_40%	100	8	0.0001	100000	121	1513	106	10359	1405
Ri_p5_n_100%	100	19	0.0001	100000	54	900	262	12665	1421
Ri_p10_n_10%	100	0	0.0001	100000	60	1562	250	5280	1052
Ri_p10_n_20%	100	1	0.0001	100000	54	1450	151	10915	1339
Ri_p10_n_40%	100	0	0.0001	100000	39	1178	308	5888	1170
Ri_p10_n_100%	100	2	0.0001	100000	19	1048	594	3704	683
Ri_p20_n_10%	100	0	0.0001	100000	31	1836	451	7453	1226
Ri_p20_n_20%	100	0	0.0001	100000	22	1396	552	7919	1412
Ri_p20_n_40%	100	10	0.0001	100000	17	1372	475	4812	889
Ri_p20_n_100%	100	7	0.0001	100000	5	1395	1051	2988	243
Ri_p30_n_10%	100	0	0.0002	100000	19	1879	464	6426	1124
Ri_p30_n_20%	100	0	0.0001	100000	15	1644	564	6352	1082
Ri_p30_n_40%	100	8	0.0002	100000	8	1430	642	4114	669
Ri_p30_n_100%	100	17	0.0002	100000	3	1988	1572	6924	558
Ri_p5_m_20%	100	5	0.0001	100000	118	1553	140	6757	1133
Ri_p5_m_40%	100	14	0.0001	100000	71	1182	295	4958	977
Ri_p5_m_60%	100	9	0.0002	100000	94	1583	436	7013	1435
Ri_p5_m_80%	100	38	0.0001	100000	59	1359	585	8353	1228
Ri_p5_m_100%	100	27	0.0001	100000	56	1382	573	5885	1233
Ri_p10_m_10%	100	9	0.0002	100000	72	1923	121	6389	14167
Ri_p10_m_20%	100	9	0.0002	100000	58	1758	344	8365	1534
Ri_p10_m_30%	100	29	0.0002	100000	44	1586	412	7317	1344
Ri_p10_m_40%	100	23	0.0002	100000	41	1686	559	9929	1444
Ri_p10_m_100%	100	40	0.0002	100000	11	1718	1147	7312	1129

Job	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_p20_m_10%	100	6	0.0002	100000	32	2087	270	11420	1775
Ri_p20_m_20%	100	30	0.0002	100000	24	1808	452	7038	1476
Ri_p20_m_40%	100	31	0.0002	100000	16	2143	934	9794	1922
Ri_p20_m_100%	100	61	0.0003	100000	2	3477	2366	12844	2051
Ri_p30_m_10%	100	16	0.0002	100000	21	2316	326	8578	1876
Ri_p30_m_20%	100	29	0.0002	100000	11	1767	694	7919	1324
Ri_p30_m_40%	100	41	0.0002	100000	5	2368	1393	11332	1682
Ri_p30_m_100%	100	66	0.0005	100000	1	4969	3592	13679	2116
Ri_p5_h_20%	100	2	0.0003	100000	151	3621	372	10460	2769
Ri_p5_h_40%	100	11	0.0003	100000	99	4650	419	13328	3480
Ri_p5_h_60%	100	23	0.0004	100000	105	6689	602	18938	4662
Ri_p5_h_80%	100	19	0.0005	100000	85	9477	1038	20004	4256
Ri_p5_h_100%	100	14	0.0006	100000	81	10674	1153	28497	5537
Ri_p10_h_10%	100	7	0.0002	100000	63	3008	214	10940	2666
Ri_p10_h_20%	100	11	0.0003	100000	53	5105	434	15536	3776
Ri_p10_h_30%	100	13	0.0005	100000	65	7637	959	20400	4341
Ri_p10_h_40%	100	23	0.0005	100000	47	8410	961	21385	5014
Ri_p10_h_10%0	100	45	0.0011	100000	18	21491	7515	40593	6924
Ri_p20_h_10%	100	13	0.0004	100000	37	5301	382	17766	3603
Ri_p20_h_20%	100	18	0.0005	100000	21	9014	1205	21017	4974
Ri_p20_h_40%	100	28	0.0007	100000	14	17401	6909	32355	5548
Ri_p20_h_100%	100	54	0.0021	100000	2	39112	14720	71524	10335
Ri_p30_h_10%	100	18	0.0004	100000	18	7540	743	18386	4445
Ri_p30_h_20%	100	22	0.0004	100000	16	12662	1493	27149	5667
Ri_p30_h_40%	100	47	0.0012	100000	9	23660	2978	42684	8097
Ri_p30_h_100%	100	76	0.0031	100000	1	58597	26849	88297	10610

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	Ges	Com	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ci_p5_20%	100	0	0.0002	100000	130	1631	172	11765	1565
Ci_p5_40%	100	5	0.0002	100000	106	1497	316	5327	1128
Ci_p5_60%	100	2	0.0002	100000	141	1988	340	7047	1482
Ci_p5_80%	100	3	0.0002	100000	107	1717	485	7582	1248
Ci_p5_100%	100	15	0.0003	100000	123	1998	539	6959	1480
Ci_p10_10%	100	0	0.0002	100000	76	1996	440	8321	1420
Ci_p10_20%	100	4	0.0002	100000	74	2051	296	8445	1439
Ci_p10_30%	100	5	0.0002	100000	68	2008	435	13207	1746
Ci_p10_40%	100	5	0.0002	100000	68	2113	468	6085	1431
Ci_p10_100%	100	15	0.0004	100000	42	2215	1045	13950	1571
Ci_p20_10%	100	4	0.0002	100000	24	2114	316	7157	1226
Ci_p20_20%	100	5	0.0003	100000	37	2409	509	6749	1449
Ci_p20_40%	100	12	0.0003	100000	24	2245	890	5753	1208
Ci_p20_100%	100	33	0.0006	100000	8	2858	2023	7240	740
Ci_p30_10%	100	9	0.0003	100000	26	2576	353	8841	1578
Ci_p30_20%	100	12	0.0003	100000	19	2354	685	6033	1195
Ci_p30_40%	100	25	0.0004	100000	10	2357	1277	5575	892
Ci_p30_100%	100	47	0.0009	100000	4	3892	3234	6090	424

B.2.3 Nachoptimierung

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	----- Indivs		--------
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NR_p10_h_P1_G1	14	96/0	0.0002	93000	31247	10	11	738	258	1598
NR_p20_h_P1_G1	36	71/0	0.0002	96018	17795	7	15	1379	405	2816
NR_p30_h_P1_G1	65	36/0	0.0003	99038	334	1	24	2283	903	4649
NR_p10_h_P2_G1	8	98/0	0.0003	92118	24147	6	14	814	240	1488
NR_p20_h_P2_G1	47	61/0	0.0003	97199	19747	6	18	1429	467	2587
NR_p30_h_P2_G1	69	31/0	0.0002	98808	5426	0	24	2278	756	3738
NR_p10_h_P3_G1	16	98/0	0.0003	93940	33636	12	14	828	294	1676
NR_p20_h_P3_G1	43	69/0	0.0002	96714	11794	9	19	1491	428	2452
NR_p30_h_P3_G1	66	35/0	0.0003	98891	1020	1	23	2223	498	3965
NR_p10_h_P1_G3	37	63/0	0.0002	95832	4072	0	28	1029	210	1667
NR_p20_h_P1_G3	62	40/0	0.0002	98264	1956	2	32	1956	917	3541
NR_p30_h_P1_G3	86	14/0	0.0003	99494	45	0	36	2476	1607	4514
NR_p10_h_P2_G3	27	76/0	0.0002	95035	9714	2	26	1049	354	1949
NR_p20_h_P2_G3	58	42/0	0.0002	98498	2111	0	29	1826	876	3053
NR_p30_h_P2_G3	84	16/0	0.0002	99530	215	0	35	2598	1202	4800
NR_p10_h_P3_G3	34	67/0	0.0001	96173	12392	1	25	1071	503	2128
NR_p20_h_P3_G3	67	33/0	0.0002	98925	188	0	31	1779	871	3883
NR_p30_h_P3_G3	83	17/0	0.0003	99509	4	0	36	2611	1374	5519

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	----- Indivs		--------
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NOC1S_10_h_P1_G1	3	99/0	0.0001	74424	19095	2	11	515	154	1261
NOC1S_20_h_P1_G1	31	79/0	0.0001	92638	3154	5	18	1391	320	2684
NOC1S_30_h_P1_G1	59	45/0	0.0002	97981	612	0	23	2222	227	4593
NOC1S_10_h_P2_G1	6	98/0	0.0001	82076	12727	2	13	596	160	1348
NOC1S_20_h_P2_G1	30	74/0	0.0002	93314	3585	0	20	1445	266	2508
NOC1S_30_h_P2_G1	58	45/0	0.0002	96311	6336	1	22	2168	292	3836
NOC1S_10_h_P3_G1	11	93/0	0.0001	78200	15809	3	13	594	160	1175
NOC1S_20_h_P3_G1	31	76/0	0.0002	94019	3319	2	20	1430	237	2245
NOC1S_30_h_P3_G1	62	40/0	0.0002	97812	1626	2	24	2227	537	3887
NOC1S_10_h_P1_G3	23	78/0	0.0001	89264	5832	0	26	896	264	2218
NOC1S_20_h_P1_G3	62	39/0	0.0001	98262	480	0	30	1740	768	4461
NOC1S_30_h_P1_G3	83	17/0	0.0001	99525	39	0	35	2508	1078	4794
NOC1S_10_h_P2_G3	34	68/0	0.0001	93721	3122	1	28	954	308	1952
NOC1S_20_h_P2_G3	58	42/0	0.0001	98090	281	0	34	1874	917	3349
NOC1S_30_h_P2_G3	85	15/0	0.0001	99606	0	0	36	2537	1254	5499
NOC1S_10_h_P3_G3	28	74/0	0.0001	92848	5573	1	26	934	269	1662
NOC1S_20_h_P3_G3	58	42/0	0.0001	98086	80	0	30	1718	740	2846
NOC1S_30_h_P3_G3	84	16/0	0.0001	99771	595	0	37	2461	904	4810
NOC1C_10_h_P1_G1	5	99/0	0.0002	73280	16889	4	11	485	140	1197
NOC1C_20_h_P1_G1	31	79/0	0.0002	95857	7564	3	18	1321	383	2656
NOC1C_30_h_P1_G1	59	45/0	0.0002	98354	1551	0	23	2173	227	4593

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NOC1C_10_h_P2_G1	12	98/0	0.0001	83686	11991	8	13	564	148	1204
NOC1C_20_h_P2_G1	30	74/0	0.0001	92751	2583	0	20	1393	227	2412
NOC1C_30_h_P2_G1	57	42/0	0.0001	97044	2142	0	22	2132	281	3723
NOC1C_10_h_P3_G1	12	93/0	0.0001	78895	15147	3	13	559	152	1083
NOC1C_20_h_P3_G1	31	76/0	0.0001	92960	1818	0	20	1371	223	2142
NOC1C_30_h_P3_G1	62	40/0	0.0001	97300	354	1	24	2189	520	3657
NOC1C_10_h_P1_G3	24	78/0	0.0001	88910	4780	1	26	847	247	2156
NOC1C_20_h_P1_G3	62	39/0	0.0001	98105	79	0	30	1710	654	4461
NOC1C_30_h_P1_G3	83	17/0	0.0001	99518	0	0	35	2493	1078	4794
NOC1C_10_h_P2_G3	34	68/0	0.0001	93388	2625	1	28	900	301	1643
NOC1C_20_h_P2_G3	59	42/0	0.0001	98100	294	1	34	1834	917	3263
NOC1C_30_h_P2_G3	85	15/0	0.0001	99606	0	0	36	2524	1254	5403
NOC1C_10_h_P3_G3	27	74/0	0.0001	92993	5410	0	26	887	238	1586
NOC1C_20_h_P3_G3	58	42/0	0.0001	98052	0	0	30	1684	740	2846
NOC1C_30_h_P3_G3	84	16/0	0.0001	99686	61	0	37	2446	904	4810

B.2.4 Direkte Integration

Job	Er- folg	Note Schn	Indivs GutSchn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	100000	11417	11417	215	63010	13941
GR_p10_n_best_L100	100	100000	6014	6014	450	52661	11003
GR_p20_n_best_L100	100	100000	2647	2647	1561	30535	3078
GR_p20_n_best_L100	100	100000	2998	2998	2371	5482	692
GR_p5_n_best_L5	100	100000	10655	10655	242	59662	12928
GR_p10_n_best_L5	100	100000	10352	10352	510	121775	14412
GR_p20_n_best_L5	100	100000	8305	8305	999	42007	6909
GR_p20_n_best_L5	100	100000	9319	9319	1542	50065	8028
GR_p5_n_best_L0	100	100000	12781	12781	480	114716	18461
GR_p10_n_best_L0	100	100000	12131	12131	487	100527	12511
GR_p20_n_best_L0	100	100000	12226	12226	988	48944	10486
GR_p20_n_best_L0	100	100000	13304	13304	1448	66824	11528
GR_p5_n_all_L100	100	100000	2771	2771	804	12328	2326
GR_p10_n_all_L100	100	100000	3976	3976	1795	22970	2696
GR_p20_n_all_L100	100	100000	4633	4633	3190	8052	1360
GR_p20_n_all_L100	100	100000	5356	5356	1652	14924	2154
GR_p5_n_all_L5	100	100000	7261	7261	918	40774	6389
GR_p10_n_all_L5	100	100000	7868	7868	1680	39243	5587
GR_p20_n_all_L5	100	100000	10980	10980	3691	31821	5918
GR_p20_n_all_L5	100	100000	11621	11621	4929	37236	6421
GR_p5_n_all_L0	100	100000	8967	8967	715	32849	6821
GR_p10_n_all_L0	100	100000	10074	10074	1725	45047	7394
GR_p20_n_all_L0	100	100000	12373	12373	3562	41999	8276
GR_p20_n_all_L0	100	100000	14396	14396	5587	50691	9148

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_m_best_L100	100		100000	25	11571	428	69922	14836
GR_p10_m_best_L100	100		100000	6	5278	923	37517	7779
GR_p20_m_best_L100	100		100000	3	4590	1955	29718	5307
GR_p30_m_best_L100	100		100000	2	4828	2984	54720	5235
GR_p5_m_best_L5	100	0.002	100000	23	11539	480	74697	15850
GR_p10_m_best_L5	100	0.002	100000	8	8003	1091	93099	11795
GR_p20_m_best_L5	100	0.001	100000	2	4862	2227	49213	5668
GR_p30_m_best_L5	100	0.001	100000	2	6298	3456	44840	5600
GR_p5_m_best_L0	100	0.003	100000	26	15899	577	378663	29031
GR_p10_m_best_L0	100	0.002	100000	6	7759	1029	86104	12192
GR_p20_m_best_L0	100	0.001	100000	3	6687	2337	40424	7331
GR_p30_m_best_L0	100	0.002	100000	2	7626	3540	102549	12863
GR_p5_m_all_L100	100	0.001	100000	3	4830	1603	28817	4490
GR_p10_m_all_L100	100	0.001	100000	1	6193	3715	26291	3127
GR_p20_m_all_L100	100	0.002	100000	1	10162	7959	16430	1253
GR_p30_m_all_L100	100	0.003	100000	1	15043	11923	17919	1172
GR_p5_m_all_L5	100	0.001	100000	3	6766	1175	57244	8602
GR_p10_m_all_L5	100	0.002	100000	2	8156	3638	44200	6877
GR_p20_m_all_L5	100	0.002	100000	1	12058	7552	46339	6141
GR_p30_m_all_L5	100	0.003	100000	1	15830	13000	43018	3019
GR_p5_m_all_L0	100	0.001	100000	3	6938	1747	55604	7626
GR_p10_m_all_L0	100	0.002	100000	2	8153	2732	45708	7447
GR_p20_m_all_L0	100	0.002	100000	1	10841	8359	30364	2863
GR_p30_m_all_L0	100	0.003	100000	1	15790	12463	29919	1893
GR_p5_h_best_L100	100		100000	16	11802	840	116551	19327
GR_p10_h_best_L100	100		100000	9	12689	1594	126858	20761
GR_p20_h_best_L100	100		100000	2	8023	3453	49765	6815
GR_p30_h_best_L100	100		100000	2	9793	5579	51034	6088
GR_p5_h_best_L5	100	0.004	100000	13	14852	1010	122445	18037
GR_p10_h_best_L5	100	0.004	100000	8	18246	2216	184076	28537
GR_p20_h_best_L5	100	0.003	100000	2	11744	4670	104448	13506
GR_p30_h_best_L5	100	0.003	100000	2	13874	7285	185517	18944
GR_p5_h_best_L0	100	0.004	100000	14	18287	1194	164504	26432
GR_p10_h_best_L0	100	0.004	100000	5	14640	2207	105368	17281
GR_p20_h_best_L0	100	0.003	100000	2	12163	4520	75727	12160
GR_p30_h_best_L0	100	0.003	100000	2	13163	7335	74981	10491
GR_p5_h_all_L100	100	0.002	100000	2	7935	3845	24341	3953
GR_p10_h_all_L100	100	0.003	100000	1	13092	7589	58394	6090
GR_p20_h_all_L100	100	0.005	100000	1	21664	17842	32578	2287
GR_p30_h_all_L100	100	0.008	100000	1	31970	25120	45500	3113
GR_p5_h_all_L5	100	0.003	100000	3	13832	2382	145296	17594
GR_p10_h_all_L5	100	0.003	100000	1	12927	8221	28301	4418
GR_p20_h_all_L5	100	0.006	100000	1	22657	17903	50110	4038
GR_p30_h_all_L5	100	0.009	100000	1	32901	26902	39761	2561
GR_p5_h_all_L0	100	0.003	100000	2	12397	3601	58016	10981
GR_p10_h_all_L0	100	0.004	100000	1	15455	7678	80644	13070
GR_p20_h_all_L0	100	0.005	100000	1	23135	17026	66010	5940
GR_p30_h_all_L0	100	0.008	100000	1	33264	27609	40091	2426

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_n_best_L100_Ri	100	0.0013	100000	16	5964	263	77389	12925
GR_p10_n_best_L100_Ri	100	0.0008	100000	5	3632	590	53316	8247
GR_p20_n_best_L100_Ri	100	0.0005	100000	1	2182	1010	4808	741
GR_p20_n_best_L100_Ri	100	0.0007	100000	1	2884	1494	6484	861
GR_p5_n_all_L100_Ri	100	0.0006	100000	3	2513	704	14394	2583
GR_p10_n_all_L100_Ri	100	0.0007	100000	2	2901	490	10403	1825
GR_p20_n_all_L100_Ri	100	0.0010	100000	1	4573	1075	17396	2264
GR_p20_n_all_L100_Ri	100	0.0013	100000	1	5679	1481	14561	2898
GR_p5_m_best_L100_Ri	100	0.0017	100000	16	7583	596	49475	11516
GR_p10_m_best_L100_Ri	100	0.0012	100000	5	5230	1172	72334	9911
GR_p20_m_best_L100_Ri	100	0.0010	100000	1	4482	2372	80395	8144
GR_p30_m_best_L100_Ri	100	0.0011	100000	0	4662	3583	12535	1439
GR_p5_m_all_L100_Ri	100	0.0007	100000	2	3058	550	18907	3449
GR_p10_m_all_L100_Ri	100	0.0009	100000	1	3649	1056	19819	3393
GR_p20_m_all_L100_Ri	100	0.0012	100000	1	5405	2446	19772	3918
GR_p30_m_all_L100_Ri	100	0.0017	100000	0	7139	3617	28482	5303
GR_p5_h_best_L100_Ri	100	0.0030	100000	18	14330	1102	160465	26135
GR_p10_h_best_L100_Ri	100	0.0020	100000	5	9430	2258	74087	14155
GR_p20_h_best_L100_Ri	100	0.0016	100000	0	6750	4656	14898	2069
GR_p30_h_best_L100_Ri	100	0.0023	100000	0	9542	7139	20162	2474
GR_p5_h_all_L100_Ri	100	0.0019	100000	3	8609	893	37444	8186
GR_p10_h_all_L100_Ri	100	0.0019	100000	1	8686	2183	58517	7699
GR_p20_h_all_L100_Ri	100	0.0024	100000	0	10621	4761	27618	6702
GR_p30_h_all_L100_Ri	100	0.0030	100000	0	12914	7491	42171	8111

Direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	100		100000	19	8965	581	35429	8205
GC_p10_best_L100	100		100000	7	7310	954	106780	11207
GC_p20_best_L100	100		100000	3	7086	1949	57701	6500
GC_p30_best_L100	100		100000	3	8194	3160	23888	4676
GC_p5_best_L5	100	0.005	100000	41	20737	583	115471	25256
GC_p10_best_L5	100	0.003	100000	14	14733	1098	111182	20937
GC_p20_best_L5	100	0.003	100000	7	14554	1971	119823	21018
GC_p30_best_L5	100	0.003	100000	3	10939	3312	76761	12018
GC_p5_best_L0	100	0.004	100000	33	17413	517	164834	30148
GC_p10_best_L0	100	0.003	100000	13	14341	1064	98009	19880
GC_p20_best_L0	100	0.002	100000	4	8514	2126	46921	8210
GC_p30_best_L0	100	0.002	100000	3	8907	3073	36770	7000
GC_p5_all_L100	100		100000	4	6537	1646	47362	6180
GC_p10_all_L100	100		100000	2	9073	3181	34606	6307
GC_p20_all_L100	100		100000	1	11488	7740	25985	3828
GC_p30_all_L100	100		100000	1	15691	11024	34703	4491

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_all_L5	100	0.003	100000	5	10090	1384	51492	9768
GC_p10_all_L5	100	0.005	100000	3	12027	2709	59528	10503
GC_p20_all_L5	100	0.004	100000	1	13439	6987	50179	7818
GC_p30_all_L5	100	0.004	100000	1	16673	11666	55869	7145
GC_p5_all_L0	100	0.002	100000	5	10550	1171	91286	13283
GC_p10_all_L0	100	0.003	100000	3	11568	3500	61575	10227
GC_p20_all_L0	100	0.003	100000	1	12414	6717	35293	6092
GC_p30_all_L0	100	0.004	100000	1	17486	11798	57492	7645
GC_p5_best_L100_Ci	100	0.003	100000	25	12436	597	57925	14973
GC_p10_best_L100_Ci	100	0.002	100000	6	7350	1048	71342	9071
GC_p20_best_L100_Ci	100	0.002	100000	3	7667	2082	85722	9063
GC_p30_best_L100_Ci	100	0.002	100000	1	7757	3081	20818	4789
GC_p5_all_L100_Ci	100	0.002	100000	5	8498	589	43067	9105
GC_p10_all_L100_Ci	100	0.002	100000	2	9281	1048	46367	7800
GC_p20_all_L100_Ci	100	0.002	100000	1	11216	2163	37657	7103
GC_p30_all_L100_Ci	100	0.003	100000	1	12891	3093	35158	8140

B.2.5 Verzögerte direkte Integration

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	-------- Indivs --------			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_n_P1_best	100	1	99/0	0.0009	100000	28	6745	298	66409	11451
GvR_p10_n_P1_best	100	6	94/0	0.0023	100000	25	9256	466	55967	13279
GvR_p20_n_P1_best	100	26	74/0	0.0018	100000	22	7406	740	62573	12263
GvR_p30_n_P1_best	100	63	37/0	0.0010	100000	23	4930	998	33569	6063
GvR_p50_n_P1_best	100	93	7/0	0.0005	100000	20	3714	1698	12528	1951
GvR_p5_n_P2_best	100	0	100/0	0.0022	100000	33	8554	334	85313	12591
GvR_p10_n_P2_best	100	6	94/0	0.0018	100000	23	7177	480	89145	13931
GvR_p20_n_P2_best	100	43	57/0	0.0015	100000	24	6662	955	60974	9883
GvR_p30_n_P2_best	100	66	34/0	0.0012	100000	24	5846	1059	78610	10513
GvR_p5_n_P3_best	100	0	100/0	0.0029	100000	32	8123	253	74616	12517
GvR_p10_n_P3_best	100	1	99/0	0.0027	100000	30	11230	504	74253	14757
GvR_p20_n_P3_best	100	48	52/0	0.0012	100000	24	5546	810	62074	9776
GvR_p30_n_P3_best	100	67	33/0	0.0012	100000	25	6317	1407	91750	11585
GvR_p5_n_P1_all	100	0	100/0	0.0014	100000	12	3190	524	16685	2993
GvR_p10_n_P1_all	100	9	91/0	0.0010	100000	15	4147	297	21682	3707
GvR_p20_n_P1_all	100	38	62/0	0.0008	100000	19	3625	973	12505	2463
GvR_p30_n_P1_all	100	72	28/0	0.0005	100000	21	3335	976	19955	2975
GvR_p5_n_P2_all	100	0	100/0	0.0007	100000	13	2954	772	14761	2878
GvR_p10_n_P2_all	100	7	93/0	0.0007	100000	15	3064	494	16900	2170
GvR_p20_n_P2_all	100	42	58/0	0.0009	100000	21	4119	955	17307	3399
GvR_p30_n_P2_all	100	71	29/0	0.0006	100000	23	3569	1092	18672	3081
GvR_p5_n_P3_all	100	0	100/0	0.0006	100000	11	2491	573	18186	2469
GvR_p10_n_P3_all	100	5	95/0	0.0008	100000	15	3525	547	16522	2619
GvR_p20_n_P3_all	100	54	46/0	0.0006	100000	22	2961	810	19886	2493
GvR_p30_n_P3_all	100	67	33/0	0.0006	100000	23	3677	852	17908	2943

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	-------- Indivs --------			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_m_P1_best	100	0	100/0	0.0031	100000	37	13504	498	113261	16691
GvR_p10_m_P1_best	100	3	97/0	0.0029	100000	24	11093	517	102416	16333
GvR_p20_m_P1_best	100	33	67/0	0.0016	100000	21	6500	1002	164532	17013
GvR_p30_m_P1_best	100	68	32/0	0.0013	100000	22	6876	1193	60834	11352
GvR_p5_m_P2_best	100	0	100/0	0.0021	100000	27	9036	545	69445	13340
GvR_p10_m_P2_best	100	14	86/0	0.0026	100000	26	11329	510	83021	15325
GvR_p20_m_P2_best	100	51	59/0	0.0015	100000	23	7268	750	80632	12031
GvR_p30_m_P2_best	100	75	25/0	0.0010	100000	22	5691	963	85826	11949
GvR_p5_m_P3_best	100	0	100/0	0.0028	100000	35	11961	513	77451	13633
GvR_p10_m_P3_best	100	4	96/0	0.0029	100000	25	11372	397	94915	16856
GvR_p20_m_P3_best	100	40	60/0	0.0017	100000	23	6957	791	45802	9999
GvR_p30_m_P3_best	100	70	30/0	0.0013	100000	22	6932	1120	73124	12418
GvR_p5_m_P1_all	100	0	100/0	0.0009	100000	11	4145	1159	15167	3355
GvR_p10_m_P1_all	100	4	96/0	0.0015	100000	14	6469	488	29843	5394
GvR_p20_m_P1_all	100	27	73/0	0.0016	100000	19	7373	549	51519	6633
GvR_p30_m_P1_all	100	67	33/0	0.0009	100000	22	5476	1275	31662	6063
GvR_p5_m_P2_all	100	1	99/0	0.0011	100000	12	4689	283	29647	3945
GvR_p10_m_P2_all	100	12	88/0	0.0012	100000	16	5449	510	22130	4378
GvR_p20_m_P2_all	100	46	54/0	0.0010	100000	20	5536	853	30164	5108
GvR_p30_m_P2_all	100	81	19/0	0.0006	100000	20	3888	901	34148	4769
GvR_p5_m_P3_all	100	1	99/0	0.0010	100000	12	4482	355	21859	3960
GvR_p10_m_P3_all	100	6	94/0	0.0016	100000	16	5662	412	22239	3575
GvR_p20_m_P3_all	100	34	66/0	0.0015	100000	21	6909	864	24485	5580
GvR_p30_m_P3_all	100	72	28/0	0.0009	100000	20	5335	1016	27899	6504

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	-------- Indivs --------			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_h_P1_best	100	0	100/0	0.0057	100000	32	17550	880	129560	23753
GvR_p10_h_P1_best	100	3	97/0	0.0047	100000	21	14552	408	92682	17989
GvR_p20_h_P1_best	100	29	71/0	0.0050	100000	24	16417	1137	170133	26298
GvR_p30_h_P1_best	100	77	23/0	0.0016	100000	21	6339	895	50969	10199
GvR_p5_h_P2_best	100	0	100/0	0.0061	100000	35	18881	751	115615	21819
GvR_p10_h_P2_best	100	2	98/0	0.0066	100000	28	21152	775	139013	28276
GvR_p20_h_P2_best	100	40	60/0	0.0051	100000	24	16417	743	93820	23203
GvR_p30_h_P2_best	100	80	20/0	0.0019	100000	23	7649	1236	85191	15295
GvR_p5_h_P3_best	100	0	100/0	0.0046	100000	36	19299	873	93023	23057
GvR_p10_h_P3_best	100	9	91/0	0.0043	100000	25	17964	477	112287	23529
GvR_p20_h_P3_best	100	44	56/0	0.0019	100000	23	8598	846	55468	11418
GvR_p30_h_P3_best	100	79	21/0	0.0014	100000	22	6883	928	61694	11876
GvR_p5_h_P1_all	100	0	100/0	0.0028	100000	11	8239	2876	29682	5893
GvR_p10_h_P1_all	100	9	91/0	0.0033	100000	14	10355	513	45072	6076
GvR_p20_h_P1_all	100	37	63/0	0.0033	100000	19	11245	890	52405	9912
GvR_p30_h_P1_all	100	75	25/0	0.0021	100000	21	8128	1317	66020	12390
GvR_p5_h_P2_all	100	0	100/0	0.0028	100000	12	8535	2450	33962	6265
GvR_p10_h_P2_all	100	12	88/0	0.0032	100000	14	9593	678	32567	5795
GvR_p20_h_P2_all	100	60	40/0	0.0023	100000	20	8306	920	54755	10138
GvR_p30_h_P2_all	100	73	27/0	0.0022	100000	22	8091	1188	54396	11146

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	Indivs			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_h_P3_all	100	0	100/0	0.0026	100000	11	7752	2664	47347	6258
GvR_p10_h_P3_all	100	11	89/0	0.0034	100000	16	10769	438	31064	6514
GvR_p20_h_P3_all	100	55	45/0	0.0024	100000	17	8549	446	36377	9153
GvR_p30_h_P3_all	100	84	16/0	0.0012	100000	20	5456	1136	56289	9008

Job	Erfolg			Nisch.	Zeit[h]	Gen	Indivs			
	Ges	Ros	Evo	GDV/GAk	Ultra1	Schn	Schn	Min	Max	GutVari
GvR_p5_n_P1_2bestRi	100	2	21	77/0	0.0036	24	6814	273	72589	12933
GvR_p10_n_P1_2bestRi	100	5	49	46/0	0.0016	10	3285	533	71829	8222
GvR_p20_n_P1_2bestRi	100	9	83	8/0	0.0011	6	2494	1044	50054	6652
GvR_p30_n_P1_3bestRi	100	12	87	1/0	0.0010	4	2041	1543	10562	896
GvR_p5_n_P2_2bestRi	100	1	19	80/0	0.0036	25	6802	230	72589	11751
GvR_p10_n_P2_2bestRi	100	9	58	33/0	0.0016	10	3433	482	51529	8473
GvR_p20_n_P2_2bestRi	100	10	84	6/0	0.0008	6	1654	1030	13164	1428
GvR_p30_n_P2_3bestRi	100	10	89	1/0	0.0010	4	2046	1549	10562	897
GvR_p5_n_P3_2bestRi	100	10	25	65/0	0.0018	23	6478	271	73157	11609
GvR_p10_n_P3_2bestRi	100	0	69	31/0	0.0006	8	2462	526	45491	6025
GvR_p20_n_P3_2bestRi	100	9	83	8/0	0.0004	6	1718	1005	17705	1838
GvR_p30_n_P3_3bestRi	100	9	91	0/0	0.0004	4	2016	1521	2636	231
GvR_p5_n_P1_2all_Ri	100	2	20	78/0	0.0011	8	2302	239	16785	2908
GvR_p10_n_P1_2all_Ri	100	6	63	31/0	0.0007	6	1515	499	8476	1629
GvR_p20_n_P1_2all_Ri	100	10	71	19/0	0.0008	5	1874	973	8879	1233
GvR_p30_n_P1_3all_Ri	100	17	81	2/0	0.0009	4	2110	1447	8274	774
GvR_p5_n_P2_all_Ri	100	6	25	69/0	0.0006	7	1787	255	12848	2397
GvR_p10_n_P2_all_Ri	100	11	65	24/0	0.0004	6	1454	525	13421	2055
GvR_p20_n_P2_all_Ri	100	8	82	10/0	0.0004	6	1674	1069	6593	909
GvR_p30_n_P2_all_Ri	100	13	86	1/0	0.0004	3	2025	1469	8818	722
GvR_p5_n_P3_2all_Ri	100	6	11	83/0	0.0005	7	1775	288	17382	2277
GvR_p10_n_P3_2all_Ri	100	13	58	29/0	0.0003	6	1449	493	9322	1672
GvR_p20_n_P3_2all_Ri	100	0	93	7/0	0.0003	5	1612	1093	6593	787
GvR_p30_n_P3_3all_Ri	100	16	82	2/0	0.0004	4	2136	1529	11478	1193

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	Indivs			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P1_best	100	1	99/0	0.0032	100000	38	14328	404	77429	16641
GvC_p10_P1_best	100	7	93/0	0.0025	100000	24	11397	388	45449	10739
GvC_p20_P1_best	100	33	67/0	0.0027	100000	24	12822	978	122327	19570
GvC_p30_P1_best	100	77	23/0	0.0013	100000	21	5324	1189	37974	7876
GvC_p5_P2_best	100	0	200/0	0.0031	100000	38	13793	614	86525	15505
GvC_p10_P2_best	100	6	188/0	0.0028	100000	25	11500	485	56954	11272
GvC_p20_P2_best	100	42	117/0	0.0019	100000	23	8764	708	52429	10656
GvC_p30_P2_best	100	69	63/0	0.0014	100000	23	7187	1115	55118	9752
GvC_p5_P3_best	100	0	100/0	0.0030	100000	37	13887	610	116078	17404
GvC_p10_P3_best	100	12	88/0	0.0026	100000	25	11841	364	68018	13982
GvC_p20_P3_best	100	34	66/0	0.0025	100000	24	12214	763	65331	13243
GvC_p30_P3_best	100	85	15/0	0.0008	100000	23	4964	1500	41775	7959

Job	Erfolg		Nisch.	Zeit[h]	Noten	Gen	Indivs			
	Ges	Evo	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P1_all	100	3	97/0	0.0021	100000	13	8185	261	39480	6887
GvC_p10_P1_all	100	7	93/0	0.0025	100000	15	9949	422	33396	7282
GvC_p20_P1_all	100	44	56/0	0.0020	100000	19	8935	728	39239	9484
GvC_p30_P1_all	100	58	42/0	0.0018	100000	21	8574	1401	37312	9388
GvC_p5_P2_all	100	0	100/0	0.0018	100000	13	7554	1514	28253	5883
GvC_p10_P2_all	100	5	95/0	0.0026	100000	17	11623	438	50736	10008
GvC_p20_P2_all	100	50	50/0	0.0014	100000	20	6731	446	36574	7508
GvC_p30_P2_all	100	69	31/0	0.0013	100000	22	7241	821	44434	9644
GvC_p5_P3_all	100	0	100/0	0.0020	100000	14	8626	1615	38132	7138
GvC_p10_P3_all	100	0	100/0	0.0026	100000	16	11169	2581	50783	8636
GvC_p20_P3_all	100	45	55/0	0.0017	100000	20	8053	755	30464	7995
GvC_p30_P3_all	100	85	15/0	0.0007	100000	20	4273	683	40880	6840

Lauf	Erfolg			Nisch.	Zeit[h]	Gen	Indivs			
	Ges	Com	Evo	GDV/GAk	Ultra1	Schn	Schn	Min	Max	GutVari
GvC_p5_P3_best_Ci	100	14	7	79/0	0.0017	27	10959	546	69731	13391
GvC_p10_P3_best_Ci	100	14	35	51/0	0.0016	15	9002	975	132373	17290
GvC_p20_P3_best_Ci	100	31	60	9/0	0.0008	5	3880	1907	57048	6068
GvC_p30_P3_best_Ci	100	42	58	0/0	0.0010	4	3925	2926	5136	399
GvC_p5_P3_all_Ci	100	9	12	79/0	0.0026	11	7333	596	33660	6394
GvC_p10_P3_all_Ci	100	23	24	53/0	0.0019	9	6162	1122	29190	6748
GvC_p20_P3_all_Ci	100	33	56	11/0	0.0010	5	3610	1992	26010	3336
GvC_p30_P3_all_Ci	100	48	50	2/0	0.0008	3	4456	3009	42934	4243

B.2.6 Rotierte Variante von Shekel's Foxholes

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p10	38	0.028	97256	12602	281268	200	447074	116471
G_p20	42	0.047	98829	13681	610786	1233	893738	257045
G_p30	57	0.069	99228	11039	739384	1325	1340801	425313
G_p50	64	0.102	99433	10956	1223056	2162	2233040	719985
G_p90	81	0.103	99781	6364	1279942	2473	4019978	979684
G_p120	93	0.101	99902	4088	1097294	3782	5358777	1290466
G_p150	98	0.098	99978	3136	1053152	5855	6694902	1523244
G_p180	95	0.089	99945	2529	1019608	4832	8034962	1274710
G_p210	100	0.101	100000	1909	899344	4894	9120469	1979874
G_p240	98	0.096	99978	2225	1196909	6392	10714322	2313211
G_p270	100	0.046	100000	900	548332	6279	7206266	1445268
G_p300	100	0.008	100000	98	108431	19591	223877	40603
G_p350	100	0.009	100000	123	103192	9461	2482497	391581
G_p400	100	0.021	100000	249	230703	7904	240689	854193
G_p450	100	0.015	100000	165	174320	5372	7284698	888645
G_p500	100	0.044	100000	414	471791	11594	13330260	1927568
R_n	0	0.000	88187		47	36	61	0
R_m	0	0.000	86849		154	140	158	0
R_h	0	0.000	92929		201	142	263	0
R_u	3	0.001	89722		20001	20001	20001	0
C	0	0.000	62409		99	88	117	0

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	92	0.028	99999	1284	139996	255	621976	156604
GR_p10_n_best_L100	100	0.020	100000	474	110720	507	892145	184782
GR_p20_n_best_L100	100	0.008	100000	93	49095	893	866038	129368
GR_p30_n_best_L100	100	0.007	100000	48	41348	1254	303465	59862
GR_p50_n_best_L100	100	0.006	100000	26	40118	2335	616060	79473
GR_p70_n_best_L100	100	0.004	100000	11	24507	3271	207753	33399
GR_p90_n_best_L100	100	0.004	100000	8	25641	4117	266080	36241
GR_p120_n_best_L100	100	0.003	100000	5	21096	5602	134977	26074
GR_p150_n_best_L100	100	0.004	100000	5	29091	6961	218577	40977
GR_p180_n_best_L100	100	0.003	100000	3	22795	8530	249293	32968
GR_p210_n_best_L100	100	0.004	100000	3	27983	9823	191310	30824
GR_p5_n_best_L5	100	0.007	100000	216	179127	290	8429761	982310
GR_p10_n_best_L5	100	0.002	100000	49	31318	519	649473	84773
GR_p20_n_best_L5	100	0.003	100000	32	43072	1009	1334992	145536
GR_p30_n_best_L5	100	0.004	100000	25	87851	1426	5601571	570297
GR_p50_n_best_L5	100	0.002	100000	8	17259	2339	152998	22013
GR_p70_n_best_L5	100	0.002	100000	4	14249	3365	103698	15870
GR_p90_n_best_L5	100	0.002	100000	4	19177	4399	93949	16863
GR_p120_n_best_L5	100	0.002	100000	3	16904	5979	83349	14256
GR_p150_n_best_L5	100	0.002	100000	2	18431	7347	73808	13626
GR_p180_n_best_L5	100	0.002	100000	2	19189	8807	90761	14111
GR_p210_n_best_L5	100	0.002	100000	2	20146	10380	79372	16097
GR_p5_n_best_L0	100	0.003	100000	94	59539	292	2339398	248433
GR_p10_n_best_L0	100	0.002	100000	44	38961	501	1568500	161109
GR_p20_n_best_L0	100	0.002	100000	19	19988	964	578991	61083
GR_p30_n_best_L0	100	0.002	100000	14	20240	1533	319211	38630
GR_p50_n_best_L0	100	0.002	100000	7	15736	2461	109390	17587
GR_p70_n_best_L0	100	0.002	100000	6	19955	3484	219456	28509
GR_p90_n_best_L0	100	0.002	100000	3	16305	4340	105380	20655
GR_p120_n_best_L0	100	0.002	100000	3	17881	5914	94867	15647
GR_p150_n_best_L0	100	0.002	100000	3	20275	7538	90455	14751
GR_p180_n_best_L0	100	0.002	100000	2	22127	9101	88408	17723
GR_p210_n_best_L0	100	0.002	100000	2	22537	10525	97936	17080
GR_p5_n_all_L100	100	0.0063	100000	54	30411	835	120883	28676
GR_p10_n_all_L100	100	0.0067	100000	26	34704	2050	164598	34791
GR_p20_n_all_L100	100	0.0069	100000	13	35870	3444	191822	39166
GR_p30_n_all_L100	100	0.0067	100000	8	33954	5096	135603	30234
GR_p50_n_all_L100	100	0.0049	100000	3	26192	8600	194596	30470
GR_p70_n_all_L100	100	0.0065	100000	3	34590	12746	256437	38640
GR_p90_n_all_L100	100	0.0056	100000	2	34489	16793	197912	33586
GR_p5_n_all_L5	100	0.0009	100000	11	12573	906	175382	21190
GR_p10_n_all_L5	100	0.0012	100000	6	13494	1871	70548	12762
GR_p20_n_all_L5	100	0.0012	100000	3	14135	3213	72673	11850
GR_p30_n_all_L5	100	0.0014	100000	2	15922	5262	75928	11319
GR_p50_n_all_L5	100	0.0016	100000	2	20022	9155	80335	13313
GR_p70_n_all_L5	100	0.0021	100000	2	24119	12888	69983	13428
GR_p90_n_all_L5	100	0.0023	100000	1	26381	16800	57222	11784
GR_p5_n_all_L0	100	0.0011	100000	14	15162	906	183846	26498
GR_p10_n_all_L0	100	0.0008	100000	5	11443	1406	77737	12726
GR_p20_n_all_L0	100	0.0014	100000	4	18822	3294	221763	30274
GR_p30_n_all_L0	100	0.0014	100000	3	17040	5117	88007	12497
GR_p50_n_all_L0	100	0.0019	100000	2	22559	9215	81291	13622
GR_p70_n_all_L0	100	0.0022	100000	2	25114	13307	89268	14056
GR_p90_n_all_L0	100	0.0025	100000	1	27947	17145	95441	17374

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_m_best_L100	100	0.003	100000	17	16596	541	235057	36324
GR_p10_m_best_L100	100	0.001	100000	4	6807	1150	115254	14591
GR_p20_m_best_L100	100	0.001	100000	2	5475	2349	29924	5643
GR_p30_m_best_L100	100	0.001	100000	1	7199	3573	39102	6575
GR_p50_m_best_L100	100	0.001	100000	1	11238	5920	72265	10373
GR_p5_m_best_L5	100	0.002	100000	13	45320	630	1667213	214441
GR_p10_m_best_L5	100	0.001	100000	5	17267	1135	684876	74171
GR_p20_m_best_L5	100	0.001	100000	2	7456	2333	88659	12647
GR_p30_m_best_L5	100	0.001	100000	2	11062	3694	122150	16085
GR_p50_m_best_L5	100	0.001	100000	1	14655	6245	65737	12995
GR_p5_m_best_L0	100	0.003	100000	39	72780	481	1199909	206702
GR_p10_m_best_L0	100	0.001	100000	7	16728	1128	779357	82044
GR_p20_m_best_L0	100	0.001	100000	2	7761	2307	85058	12864
GR_p30_m_best_L0	100	0.001	100000	1	9055	3488	67888	10553
GR_p50_m_best_L0	100	0.001	100000	1	13087	6263	57923	11582
GR_p5_m_all_L100	100	0.0012	100000	3	17194	1620	162229	28371
GR_p10_m_all_L100	100	0.0009	100000	2	15695	3725	113924	18835
GR_p20_m_all_L100	100	0.0016	100000	1	26968	8333	213941	32236
GR_p30_m_all_L100	100	0.0026	100000	1	29646	12641	125939	18350
GR_p50_m_all_L100	100	0.0044	100000	1	50877	24337	124370	22150
GR_p70_m_all_L100	100	0.0042	100000	1	67100	33367	156926	24706
GR_p90_m_all_L100	100	0.0056	100000	1	88957	40055	202351	31260
GR_p5_m_all_L5	100	0.0012	100000	5	24265	1444	158086	33560
GR_p10_m_all_L5	100	0.0009	100000	1	18358	3285	91467	18880
GR_p20_m_all_L5	100	0.0012	100000	1	19850	8387	100942	17028
GR_p30_m_all_L5	100	0.0016	100000	1	29226	13244	96847	19050
GR_p50_m_all_L5	100	0.0030	100000	1	50038	22779	206747	28429
GR_p5_m_all_L0	100	0.0008	100000	3	18840	1527	223744	35327
GR_p10_m_all_L0	100	0.0008	100000	2	18432	3688	174102	27386
GR_p20_m_all_L0	100	0.0014	100000	1	24103	8548	180802	29026
GR_p30_m_all_L0	100	0.0017	100000	1	31833	13424	137588	21963
GR_p50_m_all_L0	100	0.0028	100000	1	47937	22719	107420	21165
GR_p5_h_best_L100	100	0.016	100000	6	342776	1091	4043717	680154
GR_p10_h_best_L100	100	0.015	100000	3	301346	41772	3819192	479377
GR_p20_h_best_L100	100	0.013	100000	1	250848	64201	1864792	224806
GR_p30_h_best_L100	100	0.015	100000	1	298746	126377	1030329	123585
GR_p50_h_best_L100	100	0.022	100000	1	432854	229718	953707	102727
GR_p5_h_best_L5	100	0.030	100000	10	250392	1127	4017589	553003
GR_p10_h_best_L5	100	0.018	100000	3	149768	12901	2832102	337478
GR_p20_h_best_L5	100	0.013	100000	1	110030	43722	587096	72385
GR_p30_h_best_L5	100	0.018	100000	1	151320	76886	509152	68767
GR_p50_h_best_L5	100	0.026	100000	1	219211	120808	811549	83202
GR_p5_h_best_L0	100	0.034	100000	13	694536	1416	15005749	1828157
GR_p10_h_best_L0	100	0.017	100000	4	330089	2406	4800284	693888
GR_p20_h_best_L0	100	0.014	100000	2	265959	84356	1589924	212209
GR_p30_h_best_L0	100	0.015	100000	1	296191	145538	1701746	181903
GR_p50_h_best_L0	100	0.025	100000	1	469320	270094	1928985	220950

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_h_all_L100	100	0.0783	100000	4	507433	83234	2790367	566106
GR_p10_h_all_L100	100	0.0236	100000	2	501751	147055	1796237	396625
GR_p20_h_all_L100	100	0.0306	100000	1	617450	335518	2144488	220160
GR_p30_h_all_L100	100	0.0401	100000	1	875212	457618	1221793	135255
GR_p5_h_all_L5	100	0.0204	100000	3	442129	24499	4276017	702897
GR_p10_h_all_L5	100	0.0176	100000	1	385354	146462	1882728	243198
GR_p20_h_all_L5	100	0.0289	100000	1	621295	393946	1141450	111167
GR_p30_h_all_L5	100	0.0405	100000	1	877839	350262	1222371	141987
GR_p5_h_all_L0	100	0.0269	100000	4	570426	44190	3829710	775258
GR_p10_h_all_L0	100	0.0186	100000	1	397759	147552	2576870	282790
GR_p20_h_all_L0	100	0.0308	100000	1	627286	377313	1795579	187817
GR_p30_h_all_L0	100	0.0359	100000	1	897976	439865	1259216	142969

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	100	0.005	100000	58	27993	576	269287	39134
GC_p10_best_L100	100	0.003	100000	17	16846	1007	138151	21800
GC_p20_best_L100	100	0.003	100000	8	16205	2046	95814	18947
GC_p30_best_L100	100	0.003	100000	6	18078	2978	166765	21604
GC_p50_best_L100	100	0.004	100000	3	16795	4909	98974	17434
GC_p5_best_L5	100	0.006	100000	60	29562	610	178271	40022
GC_p10_best_L5	100	0.003	100000	18	18423	1039	181582	32417
GC_p20_best_L5	100	0.003	100000	8	16450	1949	350287	41944
GC_p30_best_L5	100	0.002	100000	3	10213	3137	54476	9375
GC_p50_best_L5	100	0.002	100000	2	10371	5210	53339	6916
GC_p5_best_L0	100	0.005	100000	52	25644	556	247842	40092
GC_p10_best_L0	100	0.006	100000	28	27877	1042	329546	53006
GC_p20_best_L0	100	0.002	100000	5	12026	2076	124960	18065
GC_p30_best_L0	100	0.002	100000	3	9132	2938	45084	8524
GC_p50_best_L0	100	0.002	100000	2	12155	4852	41498	8373
GC_p5_all_L100	100	0.0019	100000	5	9461	1679	36879	8881
GC_p10_all_L100	100	0.0023	100000	3	12108	3655	65352	11669
GC_p20_all_L100	100	0.0026	100000	1	13117	7755	46585	7620
GC_p30_all_L100	100	0.0030	100000	1	15478	11575	37877	3969
GC_p5_all_L5	100	0.0026	100000	4	8953	1481	30741	7434
GC_p10_all_L5	100	0.0030	100000	2	10105	3069	41647	7291
GC_p20_all_L5	100	0.0037	100000	1	12087	7241	36195	6231
GC_p30_all_L5	100	0.0051	100000	1	16185	11304	40991	5085
GC_p50_all_L5	100	0.0053	100000	1	24072	18389	45281	3358
GC_p70_all_L5	100	0.0072	100000	1	33460	28875	37854	1824
GC_p90_all_L5	100	0.0092	100000	1	42628	36789	47610	2266
GC_p5_all_L0	100	0.0032	100000	5	10277	1612	48615	9782
GC_p10_all_L0	100	0.0028	100000	2	9120	3695	37064	6533
GC_p20_all_L0	100	0.0044	100000	2	14652	7167	49055	8375
GC_p30_all_L0	100	0.0049	100000	1	16432	11285	41126	5946

Verzögerte direkte Integration:

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvR_p5_n_P1	90	81/0	0.138	99978	1467	156333	213	773953	167036
GvR_p10_n_P1	100	100/0	0.064	100000	752	168696	566	1221577	257792
GvR_p20_n_P1	100	78/0	0.046	100000	283	123267	921	1253315	224639
GvR_p30_n_P1	100	64/0	0.049	100000	216	136008	1535	1152946	222148
GvR_p50_n_P1	100	59/0	0.123	100000	140	124300	1721	802618	190591
GvR_p70_n_P1	100	48/0	0.126	100000	117	132464	1858	1747242	264782
GvR_p5_n_P3	90	100/0	0.034	99999	1556	166040	526	813177	185207
GvR_p10_n_P3	100	100/0	0.031	100000	696	158185	963	1020767	232050
GvR_p20_n_P3	100	88/0	0.023	100000	270	119944	983	1186004	202030
GvR_p30_n_P3	100	66/0	0.021	100000	190	113378	1303	781017	174423
GvR_p50_n_P3	100	64/0	0.023	100000	173	163024	1362	1184154	253907
GvR_p70_n_P3	100	68/0	0.023	100000	147	168926	1905	1259183	248556
GvR_p5_m_P1	100	100.0	0.002	100000	27	24798	633	278548	49179
GvR_p10_m_P1	100	98.0	0.002	100000	19	22234	672	350991	49281
GvR_p20_m_P1	100	85.0	0.003	100000	26	32919	707	594699	85232
GvR_p30_m_P1	100	73.0	0.003	100000	32	48364	733	510044	93873
GvR_p50_m_P1	100	61.0	0.002	100000	37	48454	2196	535458	79465
GvR_p5_m_P3	100	100/0	0.001	100000	17	9521	633	260477	29813
GvR_p10_m_P3	100	100/0	0.001	100000	22	16718	648	419963	54506
GvR_p20_m_P3	100	71/0	0.001	100000	24	23548	664	249413	41875
GvR_p30_m_P3	100	66/0	0.002	100000	30	38067	925	785774	98663
GvR_p50_m_P3	100	68/0	0.005	100000	42	94596	2001	1458561	183158
GvR_p5_h_P1	100	100/0	0.026	100000	17	592029	1445	14328478	1906988
GvR_p10_h_P1	100	100/0	0.022	100000	16	475331	41996	7989317	1104475
GvR_p20_h_P1	100	83/0	0.023	100000	22	499838	773	5314443	727224
GvR_p30_h_P1	100	70/0	0.045	100000	33	1028765	1011	14096474	2189621
GvR_p50_h_P1	100	68/0	0.062	100000	38	1401852	1887	18231376	2584564
GvR_p5_h_P3	100	100/0	0.012	100000	11	248927	1197	4059582	534563
GvR_p10_h_P3	100	100/0	0.030	100000	17	645540	22376	19839630	2309762
GvR_p20_h_P3	100	71/0	0.026	100000	21	576635	632	11226008	1294221
GvR_p30_h_P3	100	67/0	0.044	100000	30	949816	1729	9419924	1780740
GvR_p50_h_P3	100	59/0	0.075	100000	36	1663240	2321	17464784	2928731

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvC_p5_P1	100	100/0	0.005	100000	59	24934	604	113258	24791
GvC_p10_P1	100	100/0	0.004	100000	30	18610	1222	147983	22775
GvC_p20_P1	100	92/0	0.005	100000	32	26509	769	152017	26674
GvC_p30_P1	100	76/0	0.006	100000	36	28736	1122	189891	32882
GvC_p50_P1	100	71/0	0.007	100000	43	35732	1509	165181	38798
GvC_p5_P3	100	100/0	0.007	100000	76	33366	645	396082	55187
GvC_p10_P3	100	99/0	0.006	100000	40	28235	603	206442	39417
GvC_p20_P3	100	87/0	0.006	100000	34	27835	982	139138	29326
GvC_p30_P3	100	69/0	0.006	100000	39	28873	745	192229	32957
GvC_p50_P3	100	50/0	0.007	100000	39	35897	2168	262664	52608

Die Zeiten folgender Jobs sind wegen anderweiter Rechnerauslastung nicht korrekt: GR,m,best,l0, GR,h,best,l0, GvR,n,P1, GvR,h,P3, GR,n,all,l100, GR,m,all,l100, GC,all,l0 und GC,all,l5.

B.3 Verallgemeinerte Rastrigin Funktion

Soweit nicht anders vermerkt, wurden bei allen Jobs 100 Läufe durchgeführt. Bei den später eingesetzten verbesserten LSV-Versionen konnte die Iterationsanzahl begrenzt werden, sodaß Restarts entfallen. Ein * bei Restart bedeutet eine Iterationsbegrenzung auf 5000 beim Rosenbrock-Verfahren und 50000 beim Complex-Algorithmus, ** eine Begrenzung auf 200000.

B.3.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	Zeit[h]	Noten	Re-	Indivs				
	folg	Ultra1	Schn	start	GutSchn	Schn	Min	Max	GutVari
G_p5	100		100000		51721	51721	29039	102591	14555
G_p10	100		100000		61708	61708	33076	135749	15517
G_p20	100		100000		82972	82972	56785	121290	14818
G_p30	100		100000		108795	108795	71399	166428	18379
G_p40	100		100000		128522	128522	88922	166795	18246
G_p50	100		100000		151077	151077	114236	190748	16962
G_p70	100		100000		190431	190431	133127	266639	24874
G_p90	100		100000		232029	232029	170020	304105	24517
R_n	0		66428	0	0	1394	478	3497	0
R_m	0		69546	0	0	2000	1305	4217	0
R_h	0		71423	0	0	2711	2039	5141	0
R_u	0		71568	1	0	3852	3149	5047	0
R_v	0		69705	2	0	17863	4863	149930	0
C	0	0.0022	11477	5	0	202	159	281	0

B.3.2 Vorinitialisierte Startpopulationen

Job	R	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	S	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_n_p5_20%	*	100	0	0.005	100000	4542	52532	29875	113176	16776
Ri_n_p5_40%	*	100	0	0.006	100000	4191	50014	24361	81694	11905
Ri_n_p5_100%	*	99	0	0.004	99984	4492	51011	27240	121583	16317
Ri_n_p10_10%	*	100	0	0.006	100000	2739	63768	31519	133848	15380
Ri_n_p10_20%	*	100	0	0.007	100000	2690	63673	34732	103588	13943
Ri_n_p10_100%	*	100	0	0.019	100000	2532	71216	45184	117849	15511
Ri_n_p20_5%	*	100	0	0.007	100000	1817	85173	52783	144836	16647
Ri_n_p20_10%	*	100	0	0.009	100000	1833	86717	56574	130619	17658
Ri_n_p20_20%	*	100	0	0.012	100000	1791	87257	54917	123539	14411
Ri_n_p20_100%	*	100	0	0.034	100000	1654	102654	69945	151567	16245
Ri_n_p30_4%	*	100	0	0.015	100000	1489	105920	68427	165940	15809
Ri_n_p30_10%	*	100	0	0.015	100000	1466	105551	68128	161312	17048
Ri_n_p30_20%	*	100	0	0.018	100000	1422	106068	67215	169492	17943
Ri_n_p30_100%	*	100	0	0.056	100000	1173	122945	43721	184228	28224
Ri_m_p5_20%	*	99	0	0.005	99984	4372	51219	21489	137934	15744
Ri_m_p5_40%	*	99	0	0.007	99984	4115	50364	27041	89496	12058
Ri_m_p5_100%	*	100	0	0.014	100000	4408	59484	34315	123525	16299

Job	RS	Erfolg[%] Ges	Ros	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
Ri_m_p10_10%	*	100	0	0.006	100000	2676	62759	36812	97481	12860
Ri_m_p10_20%	*	100	0	0.009	100000	2688	65118	42229	124380	15367
Ri_m_p10_100%	*	100	0	0.025	100000	2570	77875	48651	129809	14645
Ri_m_p20_5%	*	100	0	0.008	100000	1818	85756	53501	136077	15021
Ri_m_p20_10%	*	100	0	0.01	100000	1785	85893	51922	121387	14597
Ri_m_p20_20%	*	100	0	0.014	100000	1753	87944	59899	117208	13193
Ri_m_p20_100%	*	100	0	0.047	100000	1624	113935	85367	167032	16077
Ri_m_p30_5%	*	100	0	0.013	100000	1500	107826	72823	152044	16336
Ri_m_p30_10%	*	100	0	0.018	100000	1483	110322	69728	167743	18557
Ri_m_p30_20%	*	100	0	0.022	100000	1433	110934	70024	151993	17475
Ri_m_p30_100%	*	100	0	0.072	100000	1212	142979	52055	195114	29909
Ri_h_p5_20%	*	100	0	0.007	100000	4541	54085	31636	108729	14029
Ri_h_p5_40%	*	100	0	0.01	100000	4426	55711	12798	99628	15289
Ri_h_p5_100%	0	100	0		100000	4412	63231	40209	122646	15969
Ri_h_p10_10%	*	100	0	0.007	100000	2698	64171	32752	112518	14369
Ri_h_p10_20%	*	100	0	0.011	100000	2484	62291	41831	91435	10651
Ri_h_p10_100%	0	100	0		100000	2505	84024	51290	138440	14606
Ri_h_p20_5%	*	100	0	0.01	100000	1898	90171	54920	136298	17410
Ri_h_p20_10%	*	100	0	0.012	100000	1748	86010	51918	140477	15278
Ri_h_p20_20%	*	100	0	0.019	100000	1747	91588	50349	160792	17212
Ri_h_p20_100%	0	100	0		100000	1611	128331	91463	177288	16707
Ri_h_p30_5%	*	100	0	0.014	100000	1495	109303	65645	195143	18981
Ri_h_p30_10%	*	100	0	0.018	100000	1470	110426	79669	151330	16006
Ri_h_p30_20%	*	100	0	0.026	100000	1420	115405	79947	158498	15982
Ri_h_p30_100%	0	100	0		100000	1209	165581	84598	228384	28590
Ri_h_p40_100%	0	100	0		100000	890	192761	110033	268778	48319

Vorinitialisierung mit dem Complex-Algorithmus:

Job	RS	Er-folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
Ci_p5_20%	2	100	0.007	100000	4400	50089	23680	85212	13158
Ci_p5_40%	*	100	0.083	100000	4413	54015	26268	139378	17463
Ci_p5_60%	*	100	0.125	100000	4535	57407	30276	168823	22145
Ci_p5_100%	*	100	0.23	100000	4659	62625	24460	205513	28787
Ci_p10_10%	2	100	0.031	100000	2652	62101	38647	132959	15891
Ci_p10_20%	3	100	0.041	100000	2572	60524	31388	135813	14326
Ci_p10_100%	13	99	0.336	99854	2666	75856	26705	361517	33372

Job	RS	Er-folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
Ci_p20_5%	3	100	0.05	100000	1787	84843	58700	160742	16741
Ci_p20_10%	4	100	0.054	100000	1786	84919	54319	178586	20315
Ci_p20_20%	*	100	0.11	100000	1823	89038	52444	163612	17744
Ci_p20_100%	*	100	0.647	100000	1805	109986	59039	241153	37039
Ci_p30_5%	*	100	0.123	100000	1502	109196	61267	298895	28952
Ci_p30_10%	*	100	0.78	100000	1500	121973	65577	325608	52816
Ci_p30_20%	*	100	1.919	100000	1500	140539	71880	515488	81814
Ci_p30_100%	**	100	7.525	100000	1465	253957	86119	743839	164827

B.3.3 Nachoptimierung

Job	Er-	Noten-		Nopt	Gen	Indivs			
	folg	Schn	Verb.	Erf	Schn	Schn	Min	Max	GutVari
NR_p10_m_P1_G1	0	90599	39115	0	29	4810	2282	9018	0
NR_p20_m_P1_G1	3	96243	16870	3	65	8440	4727	13169	1893
NR_p30_m_P1_G1	21	97880	7452	21	113	14606	7110	27134	3548
NR_p50_m_P1_G1	69	99444	2093	69	220	35619	17950	51123	6920
NR_p70_m_P1_G1	90	99836	1254	90	278	58033	29928	79169	8611
NR_p90_m_P1_G1	98	99968	1114	98	320	82706	54958	110912	11708
NR_p10_m_P2_G1	0	91833	36017	0	35	4660	2007	8216	0
NR_p20_m_P2_G1	4	96324	14861	4	72	8563	4912	13438	886
NR_p30_m_P2_G1	23	98201	6634	23	125	15516	8606	23174	3151
NR_p50_m_P2_G1	68	99469	1871	68	229	36988	16138	53588	7048
NR_p70_m_P2_G1	96	99931	1292	96	284	59260	38503	87088	9990
NR_p90_m_P2_G1	100	100000	1113	100	320	82984	44896	109050	12176
NR_p10_m_P3_G1	0	91620	36269	0	35	4929	2569	7874	0
NR_p20_m_P3_G1	5	96104	16101	5	70	8468	4397	14791	1919
NR_p30_m_P3_G1	18	97998	6934	18	122	15353	6124	24387	3933
NR_p50_m_P3_G1	62	99339	1804	62	224	36289	15268	60722	7323
NR_p70_m_P3_G1	91	99840	1191	91	286	59334	28873	83151	10821
NR_p90_m_P3_G1	100	100000	1109	100	329	84846	44947	116844	11312
NR_p10_m_P1_G3	4	79693	16499	4	129	7086	2381	11767	1658
NR_p20_m_P1_G3	45	95138	3710	45	308	19696	3428	32255	4520
NR_p30_m_P1_G3	74	98561	1194	74	460	38815	21661	54731	6421
NR_p50_m_P1_G3	99	99574	996	99	504	67511	43623	91116	10872
NR_p70_m_P1_G3	100	100000	941	100	519	95684	69611	137328	11879
NR_p90_m_P1_G3	100	100000	906	100	534	125613	92889	165140	15628
NR_p10_m_P2_G3	3	96243	14105	3	162	7839	3506	15218	1907
NR_p20_m_P2_G3	47	98912	3071	47	315	21520	9393	35248	4993
NR_p30_m_P2_G3	71	98583	1118	71	444	37741	22423	59570	7545
NR_p50_m_P2_G3	98	99968	1023	98	520	69432	42805	110753	12043
NR_p70_m_P2_G3	99	99984	945	99	498	92497	65828	133994	12666
NR_p70_m_P2_G3	100	100000	956	100	539	126477	94606	166644	14292
NR_p10_m_P3_G3	8	96323	14253	8	160	7992	3041	15610	2940
NR_p20_m_P3_G3	43	98796	3167	43	319	20349	8089	31039	4802
NR_p30_m_P3_G3	80	98386	1446	80	436	37114	15746	58951	7707
NR_p50_m_P3_G3	98	99968	1052	98	495	66540	45539	95159	9615
NR_p70_m_P3_G3	99	99984	927	99	511	94652	65695	128861	12049
NR_p70_m_P3_G3	100	100000	930	100	522	123166	88030	169895	17036

NR_p10_h_P1_G1	0	100/0	0.010	91756	38876	0	29	6673	3472	10632
NR_p20_h_P1_G1	1	100/0	0.010	95480	18348	1	61	10244	6094	15889
NR_p30_h_P1_G1	17	100/0	0.009	97858	7428	17	113	16551	9886	25610
NR_p50_h_P1_G1	63	100/0	0.008	99349	1912	63	225	37405	17857	52862
NR_p70_h_P1_G1	93	100/0	0.009	99889	1256	93	283	59619	32203	80220
NR_p90_h_P1_G1	97	100/0	0.010	99947	1100	97	318	82433	59459	105204
NR_p10_h_P2_G1	0	100/0	0.010	91668	35707	0	35	6776	3387	11235
NR_p20_h_P2_G1	1	100/0	0.009	95911	15663	1	70	10382	6343	15048
NR_p30_h_P2_G1	19	100/0	0.008	97802	6385	19	128	17388	9177	29034
NR_p50_h_P2_G1	63	100/0	0.008	99355	1824	63	217	36500	15488	51906
NR_p70_h_P2_G1	94	100/0	0.010	99910	1358	94	289	60255	26299	83501
NR_p90_h_P2_G1	99	100/0	0.012	99984	1097	99	327	84401	46369	122266

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NR_p10_h_P3_G1	0	100/0	0.003	91447	35188	0	35	5256	2584	7725
NR_p20_h_P3_G1	5	100/0	0.003	95927	15215	5	73	9435	6104	14541
NR_p30_h_P3_G1	20	100/0	0.004	97953	6583	20	123	16581	8128	30429
NR_p50_h_P3_G1	66	100/0	0.005	99337	1904	66	223	37282	15296	54515
NR_p70_h_P3_G1	92	100/0	0.006	99867	1174	92	295	61414	35368	82555
NR_p90_h_P3_G1	98	100/0	0.007	99968	1095	98	312	81458	45980	114625
NR_p10_h_P1_G3	7	100/0	0.010	96277	12886	7	167	9963	3843	15014
NR_p20_h_P1_G3	46	100/0	0.008	98867	2821	47	334	22504	11094	35127
NR_p30_h_P1_G3	82	100/0	0.008	99691	1287	82	453	39263	19206	68256
NR_p50_h_P1_G3	97	100/0	0.010	99953	1021	97	504	68185	46981	93127
NR_p70_h_P1_G3	100	100/0	0.012	100000	1005	100	506	94445	71218	144359
NR_p90_h_P1_G3	100	100/0	0.015	100000	914	100	535	125616	88680	159418
NR_p10_h_P2_G3	8	100/0	0.009	96258	13426	8	169	10033	4333	16581
NR_p20_h_P2_G3	46	100/0	0.008	98914	2922	46	309	21435	9892	35325
NR_p30_h_P2_G3	79	100/0	0.007	99650	1365	79	451	39090	19461	57163
NR_p50_h_P2_G3	97	100/0	0.008	99953	1017	97	497	67776	45629	102080
NR_p70_h_P2_G3	99	100/0	0.010	99984	948	99	515	95580	70096	126146
NR_p70_h_P2_G3	100	100/0	0.012	100000	963	100	530	124626	89143	160632
NR_p10_h_P3_G3	8	100/0	0.003	96495	13424	8	169	8954	4497	14908
NR_p20_h_P3_G3	41	100/0	0.003	98698	3031	41	319	21406	8452	31724
NR_p30_h_P3_G3	81	100/0	0.004	99669	1211	81	454	39226	20116	56216
NR_p50_h_P3_G3	97	100/0	0.006	99953	1032	97	489	66707	49802	91683
NR_p70_h_P3_G3	100	100/0	0.009	100000	955	100	519	96105	70764	130620
NR_p70_h_P3_G3	100	100/0	0.011	100000	911	100	531	124892	86228	170057

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P1_G1	0	100/0	0.004	54320	85	0	31	1739	660	3329
NC1S_p20_P1_G1	0	100/0	0.005	79203	0	0	60	5090	2058	8976
NC1S_p30_P1_G1	0	100/0	0.006	90927	0	0	111	11945	5594	28703
NC1S_p50_P1_G1	0	100/0	0.007	97528	0	0	219	33503	13194	63917
NC1S_p70_P1_G1	0	100/0	0.008	98782	0	0	288	57906	23233	83963
NC1S_p10_P2_G1	0	100/0	0.003	54967	62	0	33	1788	526	3151
NC1S_p20_P2_G1	0	100/0	0.005	80526	0	0	67	5453	2501	10674
NC1S_p30_P2_G1	0	100/0	0.005	91977	0	0	125	13058	5277	24072
NC1S_p50_P2_G1	0	100/0	0.009	97471	8	0	210	32684	11548	71975
NC1S_p70_P2_G1	0	100/0	0.008	98810	0	0	283	57447	28978	84683
NC1S_p10_P3_G1	0	100/0	0.004	58066	0	0	38	1956	712	3528
NC1S_p20_P3_G1	0	100/0	0.005	81826	0	0	71	5671	2103	10454
NC1S_p30_P3_G1	0	100/0	0.006	91510	0	0	117	12443	6537	24351
NC1S_p50_P3_G1	0	100/0	0.007	97681	0	0	220	33871	15524	58641
NC1S_p70_P3_G1	0	100/0	0.009	98899	0	0	291	58719	37155	79789
NC1S_p10_P1_G3	0	100/0	0.004	84130	0	0	173	5567	1595	10364
NC1S_p20_P1_G3	0	100/0	0.005	96328	0	0	313	18068	7669	28452
NC1S_p30_P1_G3	0	100/0	0.007	98567	0	0	465	37363	20297	60146
NC1S_p50_P1_G3	0	100/0	0.008	99574	0	0	500	65509	43401	95723
NC1S_p70_P1_G3	0	100/0	0.010	99788	0	0	504	91879	59566	125969

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P2_G3	0	100/0	0.003	83367	0	0	167	5366	1839	11603
NC1S_p20_P2_G3	0	100/0	0.006	95976	0	0	316	18215	5331	29361
NC1S_p30_P2_G3	0	100/0	0.007	98477	0	0	443	36135	20190	58709
NC1S_p50_P2_G3	0	100/0	0.010	99535	0	0	502	65791	45188	92291
NC1S_p70_P2_G3	0	100/0	0.011	99819	0	0	522	94769	65995	142731
NC1S_p10_P3_G3	0	100/0	0.004	83567	0	0	162	5293	1092	12229
NC1S_p20_P3_G3	0	100/0	0.006	96337	0	0	332	18912	5827	30941
NC1S_p30_P3_G3	0	100/0	0.007	98557	0	0	431	35179	22274	51014
NC1S_p50_P3_G3	0	100/0	0.009	99614	0	0	498	65412	40601	96231
NC1S_p70_P3_G3	0	100/0	0.011	99813	0	0	502	91431	67684	134512

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1C_p10_P1_G1	0	100/0	0.002	54238	2	0	31	1541	634	3218
NC1C_p20_P1_G1	0	100/0	0.009	79268	65	0	60	5207	1756	43062
NC1C_p30_P1_G1	0	100/0	0.003	90927	0	0	111	11701	5152	28502
NC1C_p50_P1_G1	0	100/0	0.004	97528	0	0	219	33272	12757	63742
NC1C_p70_P1_G1	0	100/0	0.009	98791	9	0	288	57890	23023	85104
NC1C_p10_P2_G1	0	100/0	0.002	54907	0	0	33	1581	483	2962
NC1C_p20_P2_G1	0	100/0	0.002	80526	0	0	67	5228	2241	10502
NC1C_p30_P2_G1	0	100/0	0.005	91993	16	0	125	13000	5102	27277
NC1C_p50_P2_G1	0	100/0	0.004	97462	0	0	210	32288	11345	53723
NC1C_p70_P2_G1	0	100/0	0.006	98810	0	0	283	57243	28736	84563
NC1C_p10_P3_G1	0	100/0	0.002	58066	0	0	38	1751	706	3099
NC1C_p20_P3_G1	0	100/0	0.003	81826	0	0	71	5444	1957	10218
NC1C_p30_P3_G1	0	100/0	0.015	91547	37	0	117	12935	6198	47621
NC1C_p50_P3_G1	0	100/0	0.016	97686	6	0	220	34245	15305	85525
NC1C_p70_P3_G1	0	100/0	0.010	98907	8	0	291	58777	36979	80672
NC1C_p10_P1_G3	0	100/0	0.002	84130	0	0	173	5365	1438	10137
NC1C_p20_P1_G3	0	100/0	0.012	96331	3	0	313	18391	7436	57271
NC1C_p30_P1_G3	0	100/0	0.007	98567	0	0	465	37314	20163	59918
NC1C_p50_P1_G3	0	100/0	0.011	99585	12	0	500	65732	43203	95389
NC1C_p70_P1_G3	0	100/0	0.009	99789	1	0	504	91795	59398	125789
NC1C_p10_P2_G3	0	100/0	0.002	83367	0	0	167	5164	1561	11401
NC1C_p20_P2_G3	0	100/0	0.006	95991	15	0	316	18181	5147	29164
NC1C_p30_P2_G3	0	100/0	0.006	98477	0	0	443	36001	19994	60085
NC1C_p50_P2_G3	0	100/0	0.011	99535	10	0	502	65877	44957	92075
NC1C_p70_P2_G3	1	100/0	0.013	99823	4	1	522	94905	65812	142582
NC1C_p10_P3_G3	0	100/0	0.003	83567	0	0	162	5093	926	12027
NC1C_p20_P3_G3	0	100/0	0.006	96344	7	0	332	18875	5740	39161
NC1C_p30_P3_G3	0	100/0	0.004	98557	0	0	431	34947	21748	50778
NC1C_p50_P3_G3	0	100/0	0.008	99615	2	0	498	65329	40255	96021
NC1C_p70_P3_G3	0	100/0	0.010	99813	0	0	502	91328	67514	134266

B.3.4 Direkte Integration

Job	Er-	R	Noten	Gen	Indivs				
	folg	S	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	0	100000	53	188875	188875	11882	1111817	229870
GR_p10_n_best_L100	100	0	100000	14	117397	117397	23984	724222	133312
GR_p20_n_best_L100	100	0	100000	4	96003	96003	26192	418517	56440
GR_p30_n_best_L100	100	0	100000	3	124030	124030	55931	210774	27621
GR_p40_n_best_L100	100	0	100000	3	161228	161228	73247	246808	27032
GR_p5_n_best_L5	100	0	100000	77	426869	426869	23151	3755419	518515
GR_p10_n_best_L5	100	0	100000	30	363779	363779	54255	3814273	427087
GR_p20_n_best_L5	100	0	100000	15	399368	399368	89117	1612746	251098
GR_p30_n_best_L5	100	0	100000	8	426594	426594	152106	813671	149692
GR_p5_n_best_L0	8/10	0	99683	617	4944101	6604921	1394571	16484633	5061307
GR_p10_n_best_L0	8/10	0	99683	382	4694183	7227161	859537	21911222	3705670
GR_p20_n_best_L0	10/10	0	100000	197	6205004	6205004	1875668	17983587	4954405
GR_p30_n_best_L0	10/10	0	100000	139	6294066	6294066	2051349	16721213	4701985
GR_p5_n_all_L100	100	0	100000	7	166390	166390	66217	667286	105740
GR_p10_n_all_L100	100	0	100000	3	220255	220255	94888	356167	47000
GR_p20_n_all_L100	50/50	0	100000	2	385969	385969	187876	498744	61761
GR_p30_n_all_L100	50/50	0	100000	2	582561	582561	239260	834147	109928
GR_p5_n_all_L5	49/50	0	99968	27	791929	936506	146276	8020799	883202
GR_p10_n_all_L5	49/50	0	99968	13	999634	1151028	345186	8569306	745590
GR_p20_n_all_L5	50/50	0	100000	7	1315298	1315298	462867	2457837	431699
GR_p30_n_all_L5	50/50	0	100000	6	1832009	1832009	707600	3453347	616029
GR_p5_n_all_L0	2/10	0	98024	376	3573676	9308062	3371621	11454250	285748
GR_p10_n_all_L0	3/10	0	98773	192	7525855	10039828	4476599	11957938	2656600
GR_p20_n_all_L0	4/10	0	98931	63	5086486	8745246	4466271	12025668	537601
GR_p30_n_all_L0	6/10	0	99189	33	5665276	7863074	3839805	11521945	2209580
GR_p5_m_best_L100	100	0	100000	6	43864	43864	16687	170260	25410
GR_p10_m_best_L100	100	0	100000	5	65298	65298	35973	252583	29517
GR_p20_m_best_L100	100	0	100000	3	103213	103213	69864	167832	19530
GR_p30_m_best_L100	100	0	100000	3	139245	139245	100620	232728	25510
GR_p40_m_best_L100	100	0	100000	3	179933	179933	85377	240428	31380
GR_p50_m_best_L100	100	0	100000	2	216933	216933	102729	287823	35640
GR_p5_m_best_L100_Ri	100	0	100000	6	51229	51229	21344	223442	32692
GR_p10_m_best_L100_Ri	100	0	100000	3	66832	66832	37600	209340	24920
GR_p20_m_best_L100_Ri	100	0	100000	3	120157	120157	76326	250413	25877
GR_p30_m_best_L100_Ri	100	0	100000	2	156225	156225	71971	222601	23482
GR_p40_m_best_L100_Ri	100	0	100000	2	209129	209129	154386	282372	28883
GR_p5_h_best_L100	Nach 3 Läufen abgebrochen, } da das								
GR_p10_h_best_L100	Nach 4 Läufen abgebrochen, } Rosenbrock-								
GR_p20_h_best_L100	Nach 3 Läufen abgebrochen, } Verfahren								
GR_p30_h_best_L100	Nach 4 Läufen abgebrochen, } nicht konvergiert.								

Direkte Integration mit dem Complex-Algorithmus:

Job	Er-	R	Zeit[h]	Noten	Gen	Indivs			
	folg	S	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	2/5	3	53.17	99679	4261	4782542	3536914	5361324	576715
GC_p10_best_L100	3/5	3	55.02	99986	2134	4872391	4357904	5421542	316688
GC_p20_best_L100	1/5	*	70.77	99992	1391	6485659	6094810	6688590	0
GC_p30_best_L100	1/5	4	67.53	99979	871	6147010	4580715	6560789	0
GC_p10_best_L100_Ci	5/5	*	57.33	100000	2290	5201136	3454557	6327407	1070324

B.3.5 Verzögerte direkte Integration

Job	Er-	Noten	Gen	Indivs				
	folg	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GvR_p5_m_P1	100	100000	23	48866	48866	14118	289632	38969
GvR_p10_m_P1	100	100000	35	58321	58321	15371	245129	38512
GvR_p20_m_P1	100	100000	64	61936	61936	30110	161825	28132
GvR_p5_m_P2	100	100000	22	48727	48727	8894	262803	41072
GvR_p10_m_P2	100	100000	38	55952	55952	17945	183715	30570
GvR_p20_m_P2	100	100000	75	56627	56627	27392	168422	27887
GvR_p30_m_P2	100	100000	123	60655	60655	34621	252613	26932
GvR_p50_m_P2	100	100000	226	89015	89015	70563	142628	10102
GvR_p70_m_P2	100	100000	290	180009	180009	155339	206055	11833
GvR_p5_m_P3	100	100000	22	47378	47378	16052	298427	37797
GvR_p10_m_P3	100	100000	41	63128	63128	16433	359835	53844
GvR_p20_m_P3	100	100000	72	62654	62654	31588	230352	36174
GvR_p30_m_P3	100	100000	128	58581	58581	39449	117674	14818
GvR_p40_m_P3	100	100000	174	72747	72747	34500	137165	14838
GvR_p50_m_P3	100	100000	217	88418	88418	73139	135460	9936
GvR_p70_m_P3	100	100000	288	132522	132522	103799	165367	12084
GvR_p5_m_P1_Ri	100	100000	11	51340	51340	13561	257555	40393
GvR_p10_m_P1_Ri	100	100000	12	55877	55877	26911	219656	29904
GvR_p20_m_P1_Ri	100	100000	15	80510	80510	54710	223971	27789
GvR_p5_m_P2_Ri	100	100000	12	48464	48464	12957	228683	32499
GvR_p10_m_P2_Ri	100	100000	13	56988	56988	28366	168866	26842
GvR_p20_m_P2_Ri	100	100000	19	76770	76770	40599	210497	23968
GvR_p30_m_P2_Ri	100	100000	25	99353	99353	56145	209002	22955
GvR_p50_m_P2_Ri	100	100000	27	133282	133282	94589	223222	29967
GvR_p70_m_P2_Ri	100	100000	20	160488	160488	134397	280584	32297
GvR_p5_m_P3_Ri	100	100000	12	49705	49705	15380	225178	32808
GvR_p10_m_P3_Ri	100	100000	14	57064	57064	25178	128311	21914
GvR_p20_m_P3_Ri	100	100000	21	74712	74712	50985	201073	22780

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P1	10/10	10/0	43.92	100000	3828	4267176	2833188	6562423	1309165
GvC_p10_P1	10/10	10/0	59.13	100000	2667	5846254	3965329	7558432	1212800
GvC_p5_P2	10/10	10/0	44.8	100000	3897	4479722	2522452	6104104	1200423
GvC_p10_P2	10/10	10/0	47.57	100000	2186	4790770	2946802	7077246	1287074

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P3	10/10	10/0	54.65	100000	4624	5149351	3767292	7673566	1485679
GvC_p10_P3	10/10	10/0	56.42	100000	2441	5484812	4037067	8627772	1507307
GvC_p20_P3	10/10	10/0	66.89	100000	1480	6490189	3716411	8593018	1525757
GvC_p30_P3	10/10	10/0	69.78	100000	1106	6878699	2918618	8712196	1783660
GvC_p5_P1_Ci	10/10	10/0	37.66	100000	3408	3768414	2585033	5217418	818231
GvC_p5_P2_Ci	10/10	10/0	46.16	100000	4206	4674442	3763667	6274597	758164
GvC_p5_P3_Ci	9/10	10/0	47.76	100000	4163	4635591	1926935	7951188	1455279

B.3.6 Rotierte Variante der Rastrigin Funktion

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p120	0/32	1.257	93838	50000	13412667	13404841	13420602	0
G_p150	0/21	1.622	94311	50000	16764971	16755149	16789675	0
G_p210	0/8	1.864	94800	50000	23469571	23461141	23481972	0
G_p270	0/7	2.409	94873	50000	30175569	30170827	30181719	0
G_p350	0/4	3.656	95021	50000	39135588	39113407	39192431	0
G_p500	0/3	4.586	94873	50000	55900261	55897655	55903344	0
G_p750	0/2	7.748	94873	50000	83829207	83820767	83837647	0
G_p1000	0/2	19.855	95464	50000	111800346	111797249	111803442	0

Wegen des großen Aufwands und der Erfolglosigkeit wurden die GLEAM-Läufe abgebrochen.

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
R_n	0	0.0008	16245		758	438	1636	0
R_m	0	0.0017	17504		1430	1007	2370	0
R_h	0	0.0067	16114		2529	1335	3383	0
R_u	0	0.0092	16631		2773	1313	4162	0
R_v	0	0.0090	17779		2830	1386	4121	0
C	0	0.0078	12588		599	148	20000	0

Variante mit 5 Parametern:

Jobs mit der unrotierten Version für Vergleichszwecke:

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p5	100	0.0002	10000	247	3004	1342	5743	868
G_p10	100	0.0002	10000	187	4716	2837	9646	1306
G_p20	100	0.0003	10000	128	6789	2984	11884	1584
G_p30	100	0.0005	10000	105	8519	4645	12613	1588
G_p40	100	0.0004	10000	96	10500	5675	15765	1909
G_p50	100	0.0006	10000	89	12410	7405	16084	1766
G_p70	100	0.0006	10000	82	16276	10676	21060	2058
G_p90	100	0.0010	10000	78	20010	14121	28364	2627
R_n	0	0.0	89503		139	85	282	0
R_m	0	0.0	86679		535	222	20001	0
R_h	0	0.0001	89950		994	462	20001	0
R_u	0	0.0021	88350		14658	775	20001	0
R_v	0	0.0029	89235		20001	20001	20001	0
C	0	0.0001	76607		102	83	879	0

Jobs mit der rotierten Version:

GLEAM-Jobs mit einem Generationslimit von 100000. Die rechte mit „99,5% Erfolg“ überschrieben Doppelspalte gibt die Anzahl notwendiger Läufe (AL) an, um mit 99,5% Wahrscheinlichkeit Erfolg zu haben und die sich daraus ergebende Anzahl an Evaluationen (Indivis).

Job	Er-	Zeit[h]	Noten	Gen	Indivs				99,5% Erfolg:	
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari	AL	Indivs
p30	3	0.051	97433	13191	884724	8708	2303876	112027	174	153941976
p50	7	0.103	97907	17742	1982264	9005	5202424	492668	74	146687536
p70	6	0.170	98001	22146	3462971	15597	7745523	2153964	94	325519274
p100	7	0.291	98122	26992	6028430	28320	13597256	852833	74	446103820
p150	7	0.553	98182	33417	11193493	38099	21184222	223863	79	884285947
p200	13	0.842	98365	37785	16873598	41163	33426415	4148715	39	658070322
p300	14	2.169	98490	48658	32588606	67965	52836529	1430224	36	1173189816
p400	26	3.098	98704	51994	46424438	102123	89256166	742692	18	835639884
p500	23	4.845	98715	65020	72557536	124292	111579331	2060534	21	1523708256
p600	22	5.862	98705	70624	94569605	132036	133941569	19006877	22	2080531310
p700	32	6.774	98881	64579	100889682	162085	156250871	20097576	14	1412455548
p800	40	6.849	99032	58687	104792781	147610	178571479	10237306	11	1152720591
p900	34	8.085	98948	67573	135735355	210233	200916756	42045077	13	1764559615
p1000	41	8.686	99053	59244	132243333	243608	223228182	26580169	11	1454676663
p1200	40	7.932	99038	58183	155867636	274879	267892427	20822547	11	1714543996
p1400	56	8.991	99291	44133	137972264	304384	312562412	45434055	7	965805848
p1600	66	8.974	99416	37727	134823039	331877	357410043	64498128	5	674115195
p1800	61	9.465	99382	38587	155120748	431524	401895458	2176894	6	930724448
p2000	61	11.770	99382	40006	178699543	481750	446583349	69724447	6	1072197258
p2200	71	7.199	99541	29156	143321216	537341	491275474	61956158	5	716606080
p2400	80	6.140	99683	22414	120249666	521201	535969136	79636622	4	480998664
p2600	74	9.086	99588	27230	158722152	546963	580548193	67285832	4	634888608
p2800	78	6.825	99651	21800	136455800	587254	625205227	48475743	4	545823200
p3000	83	5.651	99731	16104	108081996	631878	669871496	38086417	3	324245988
p3200	81	10.655	99699	21508	153868539	647784	714720981	124706344	4	615474156
p3600	86	7.378	99778	14713	118540617	682780	804028693	35284846	3	355621851
p4000	91	6.019	99857	10862	97308921	734308	893351415	94048874	3	291926763
p4400	92	4.937	99873	8223	81144753	934683	982545036	47615075	3	243434256
p4800	96	1.607	99937	3032	32893233	935404	1071351535	3430848	2	65786466
p5200	96	3.583	99937	5834	68146680	1037107	1161239799	128548900	2	136293360
p5600	94	5.171	99905	6371	80089441	1275444	1250349311	18785015	2	160178882
p6000	100	0.425	100000	458	6632761	1102872	290177441	29732438	1	6632761
p6400	96	3.817	99937	4365	62876550	1158968	1429330853	23822144	2	125753100
p6800	99	1.304	99984	656	25571660	1199568	1518370253	68989093	2	51143320
p7200	99	1.005	99984	1188	19665028	1566491	1607766614	3680259	2	39330056
p7600	100	0.247	100000	191	3838174	1706777	32872834	5098739	1	3838174
p8000	100	0.904	100000	882	16428005	1429755	1076427753	109059409	1	16428005
p8400	99	1.461	99984	1207	23301089	1692479	1875761586	8108106	2	46602178
p8800	99	2.533	99984	1909	38188588	1810269	1965099021	147049199	2	76377176
p9200	100	0.225	100000	159	3972869	1279523	34447562	4973813	1	3972869
p9600	100	0.296	100000	179	4573759	1984918	47571131	7642080	1	4573759
p10000	100	0.213	100000	138	3839166	2059760	32079711	3822152	1	3839166
p10400	100	1.116	100000	708	17247709	1969687	1395794664	139260402	1	17247709
p10800	100	0.229	100000	114	3555174	2052564	16113109	2175963	1	3555174
p11200	100	0.189	100000	107	3518702	2275457	15709635	1679996	1	3518702
p11600	100	0.178	100000	116	3880518	1970175	31169469	3469198	1	3880518
p12000	100	0.201	100000	101	3589797	2530161	10467742	1152960	1	3589797
p12400	100	0.228	100000	104	3815254	2689134	11951906	1266276	1	3815254
p12800	99	1.568	99984	769	22955604	2635994	1877726709	2939821	2	45911208
p13200	100	0.215	100000	97	3836059	2537104	16688458	1667080	1	3836059
p13600	100	0.242	100000	106	4214909	2935224	20469713	1999140	1	4214909
p14000	100	0.237	100000	100	4142513	2796694	22924962	2168245	1	4142513

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
R_n	0	0.0	74278		114	79	179	0
R_m	0	0.0	79047		494	238	20001	0
R_h	0	0.0001	78368		957	401	20001	0
R_u	0	0.0023	76211		16186	774	20001	0
R_v	0	0.0027	76416		20001	20001	20001	0
C	0	0.0001	73721		89	71	108	0

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GR_p5_n_best_L100	91	0.0236	99806	359	225015	3116	1024015	252581
GR_p10_n_best_L100	97	0.0450	99942	216	249763	5513	2288501	420484
GR_p20_n_best_L100	100	0.0219	100000	83	213623	7219	2842313	510529
GR_p30_n_best_L100	100	0.0089	100000	17	119840	7426	2098542	269128
GR_p50_n_best_L100	100	0.0056	100000	5	94136	26352	358272	53060
GR_p70_n_best_L100	100	0.0064	100000	4	100188	37522	255203	46728
GR_p90_n_best_L100	100	0.0078	100000	4	132230	31507	307597	52904
GR_p5_n_best_L5	93	0.1792	99877	381	2674459	18159	69911368	2280579
GR_p10_n_best_L5	98	0.1208	99967	263	1611986	10125	34272876	4778936
GR_p20_n_best_L5	100	0.0892	100000	164	1076380	29239	42146174	5325743
GR_p30_n_best_L5	100	0.0794	100000	74	1065864	35694	69430527	7123369
GR_p50_n_best_L5	100	0.1428	100000	54	1989380	24806	181655380	18148409
GR_p5_n_best_L0	34	0.3717	98889	2578	8682501	64846	63037102	8830761
GR_p10_n_best_L0	33/50	1.7370	99450	3083	23151736	6572	114221597	26108956
GR_p20_n_best_L0	10/10	2.0297	100000	1966	29005209	329916	71989990	29358868
GR_p30_n_best_L0	9/10	0.9887	99842	1115	13330991	91425	81333201	26296050
GR_p5_n_all_L100	100	0.0081	100000	25	115802	15035	687134	115365
GR_p10_n_all_L100	100	0.0056	100000	5	100623	15490	456427	71815
GR_p20_n_all_L100	100	0.0081	100000	4	139215	24070	1021945	118148
GR_p30_n_all_L100	100	0.0092	100000	3	158353	36475	409031	73940
GR_p50_n_all_L100	100	0.0206	100000	3	235756	50344	599805	100233
GR_p70_n_all_L100	100	0.0241	100000	2	255638	59781	540438	99403
GR_p90_n_all_L100	100	0.0715	100000	3	359729	92708	776719	137158
GR_p5_n_all_L5	99	0.0397	99984	53	533973	17862	31446627	466384
GR_p10_n_all_L5	100	0.0147	100000	19	156790	11883	1206649	143839
GR_p20_n_all_L5	100	0.0142	100000	8	158264	32697	426543	79785
GR_p30_n_all_L5	100	0.0189	100000	8	206102	67909	523135	76160
GR_p50_n_all_L5	100	0.0275	100000	7	289986	114108	661858	95849
GR_p5_n_all_L0	57	0.4850	99230	1054	4861194	24417	70455390	2479714
GR_p10_n_all_L0	32/50	0.9708	99394	685	11955545	51459	155652999	1730953
GR_p20_n_all_L0	19/25	2.6883	99620	335	38061791	117976	325263686	8242934
GR_p30_n_all_L0	47/50	0.4383	99905	112	5227106	120934	141259137	4876944

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_m_best_L100	89	0.0669	99761	359	637083	24126	3033094	729256
GR_p10_m_best_L100	99	0.0375	99978	99	389916	25754	4050246	637378
GR_p20_m_best_L100	100	0.0319	100000	40	320555	32689	6468088	815506
GR_p30_m_best_L100	100	0.0139	100000	6	190826	38719	782246	130089
GR_p40_m_best_L100	100	0.0162	100000	5	217345	45353	1412634	157636
GR_p50_m_best_L100	100	0.0171	100000	5	233437	35349	644608	109916
GR_p70_m_best_L100	100	0.0189	100000	4	280278	64929	802039	137871
GR_p90_m_best_L100	100	0.0219	100000	4	310058	106942	808257	118771
GR_p5_m_best_L5	87	0.4414	99770	580	6025333	41882	92407473	7752031
GR_p10_m_best_L5	94	0.5433	99887	400	7524076	26100	147262917	16955853
GR_p20_m_best_L5	99	0.2883	99978	148	3630129	51648	134405397	6534596
GR_p30_m_best_L5	99	0.2947	99978	129	3552472	66253	128113501	13753460
GR_p50_m_best_L5	100	0.0483	100000	20	525893	36013	3555054	615087
GR_p70_m_best_L5	100	0.0550	100000	17	552104	61062	4508548	640131
GR_p90_m_best_L5	100	0.0592	100000	14	587218	97151	2372406	412485
GR_p5_m_best_L0	35	0.8050	98799	1733	9211570	49270	57852108	7408615
GR_p10_m_best_L0	9/25	3.2300	98892	2472	41460793	1826341	191306202	58288179
GR_p20_m_best_L0	8/10	3.8078	99683	1271	52375956	738359	216578190	73158032
GR_p30_m_best_L0	9/10	4.1928	99842	989	61813856	1161264	374288240	48978994
GR_p5_m_all_L100	100	0.0208	100000	21	275896	32729	2185141	310144
GR_p10_m_all_L100	100	0.0205	100000	9	287555	48875	1815533	291486
GR_p20_m_all_L100	100	0.0239	100000	4	369007	90434	3774917	373247
GR_p30_m_all_L100	100	0.0278	100000	3	420692	133907	1317319	206666
GR_p50_m_all_L100	100	0.0391	100000	3	584951	235856	1181858	189728
GR_p70_m_all_L100	100	0.0505	100000	3	764872	184804	1637644	301154
GR_p90_m_all_L100	100	0.0588	100000	2	892167	309637	1777695	261390
GR_p5_m_all_L5	95	0.1811	99909	130	2068675	66360	76839530	1829123
GR_p10_m_all_L5	100	0.0450	100000	21	499181	46866	6595468	948478
GR_p20_m_all_L5	100	0.0383	100000	9	453181	126063	1609241	266598
GR_p30_m_all_L5	100	0.0422	100000	7	491092	64527	1169085	185788
GR_p50_m_all_L5	100	0.0631	100000	6	711196	250490	1723562	230225
GR_p5_m_all_L0	51	0.6601	99176	947	6421490	77527	72613934	3788440
GR_p10_m_all_L0	37/50	0.6884	99576	559	6535491	81627	29067071	4792339
GR_p20_m_all_L0	46/50	0.7896	99861	147	8405880	139422	126159074	5814983
GR_p30_m_all_L0	93	0.9294	99889	130	9971723	213695	196345614	13645758
GR_p5_h_best_L100	96	0.4533	99913	342	2227223	25299	14801081	3081270
GR_p10_h_best_L100	100	0.3839	100000	144	2001853	44445	19220508	3932186
GR_p20_h_best_L100	100	0.3564	100000	63	1826094	31204	22617124	4346068
GR_p30_h_best_L100	100	0.1286	100000	13	782210	37064	23746686	2388594
GR_p50_h_best_L100	100	0.1025	100000	5	692013	231057	4195267	476941
GR_p70_h_best_L100	100	0.1114	100000	4	753915	59252	2467489	345399
GR_p5_h_best_L5	85	0.7583	99703	425	8525948	28174	62805256	10731029
GR_p10_h_best_L5	43/50	1.4469	99719	345	16893256	157970	162708345	13291690
GR_p20_h_best_L5	49/50	0.7683	99968	127	7818620	83954	60126576	11577321
GR_p30_h_best_L5	49/50	0.8100	99957	81	9229016	222704	75481208	14866415

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GR_p5_h_best_L0	37	1.6336	98760	1245	14485123	165597	39108608	7952517
GR_p10_h_best_L0	7/10	3.1958	99525	1190	28842563	6323547	54798333	18461086
GR_p20_h_best_L0	6/10	7.3382	99366	1310	67618754	3392018	159594965	31970969
GR_p30_h_best_L0	8/10	8.0614	99683	1086	77404685	19628213	211369577	46046406
GR_p5_h_all_L100	100	0.0767	100000	22	796510	89899	4817115	889588
GR_p10_h_all_L100	100	0.0578	100000	7	635873	162748	2958433	443260
GR_p20_h_all_L100	100	0.0722	100000	3	825545	314865	2048691	355667
GR_p30_h_all_L100	100	0.1017	100000	3	1157713	265697	3818613	555494
GR_p5_h_all_L5	86	0.6700	99755	176	7294371	61556	83765056	10430281
GR_p10_h_all_L5	100	0.2875	100000	31	3122189	362608	51537937	5899490
GR_p20_h_all_L5	100	0.2019	100000	9	2089628	420285	15492208	2190680
GR_p30_h_all_L5	100	0.2042	100000	7	1933146	381555	7006753	1030517
GR_p50_h_all_L5	100	0.3206	100000	6	2591342	723695	5613200	973594
GR_p5_h_all_L0	27/50	1.6455	99153	638	13448377	587187	42951788	5399304
GR_p10_h_all_L0	19/25	2.3479	99596	336	20801151	238215	125599008	8699183
GR_p20_h_all_L0	23/25	2.0953	99873	155	17423575	144292	145304272	11887167
GR_p30_h_all_L0	24/25	1.240	99937	66	10317740	1139087	107809901	7834791

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GC_p5_best_L100	93	0.1072	99795	310	154266	5745	849633	144627
GC_p10_best_L100	96	0.1215	99900	187	183316	12552	2014705	257869
GC_p20_best_L100	100	0.1062	100000	79	155181	16099	2569433	293191
GC_p30_best_L100	99	0.1628	99978	80	235276	12068	4537852	558792
GC_p50_best_L100	100	0.1350	100000	35	176186	13755	3475587	385427
GC_p70_all_L100	100	0.1228	100000	23	160943	44469	299462	54426
GC_p90_all_L100	100	0.1386	100000	20	181367	34627	461298	76387
GC_p5_best_L5	12	0.5619	97651	1520	736293	35825	1870604	376347
GC_p10_best_L5	10/50	1.4111	98069	1926	1845204	31156	3534961	850571
GC_p20_best_L5	20/50	2.6722	98729	2034	3904001	162426	10099984	2276292
GC_p30_best_L5	12/25	3.5706	98989	1506	4342761	51248	12486864	2057530
GC_p5_best_L0	1	0.5817	96542	1477	700883	323622	1292851	0
GC_p10_best_L0	1/50	1.5081	97142	1998	1894930	1011786	4063120	0
GC_p20_best_L0	0/25	3.2721	97483	2349	4410513	2146715	8037548	0
GC_p30_best_L0	1/10	7.3806	98102	3095	8760016	3506327	15701247	0
GC_p5_all_L100	100	0.1303	100000	87	175378	8038	1176761	159648
GC_p10_all_L100	100	0.1519	100000	49	206319	36109	1305954	169721
GC_p20_all_L100	100	0.1783	100000	27	241340	48426	696807	131636
GC_p30_all_L100	100	0.2172	100000	22	294755	62314	755397	133401
GC_p50_all_L100	100	0.3147	100000	19	426442	112401	1093978	187149
GC_p70_all_L100	100	0.3658	100000	16	485808	90516	1223953	204985
GC_p90_all_L100	100	0.4517	100000	15	582982	162999	1329646	206482
GC_p5_all_L5	19/50	2.4656	98899	1470	3022397	381760	5999811	1349832
GC_p10_all_L5	22/25	2.8347	99810	895	3448334	190934	9506672	2203846
GC_p20_all_L5	25/25	2.4139	100000	478	3226043	255421	9015324	2541715
GC_p30_all_L5	10/10	3.8303	100000	444	4770593	302142	20362228	6773463

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GC_p5_all_L0	3/50	2.6461	98135	1638	3394619	1375578	7518275	1120290
GC_p10_all_L0	3/10	5.6175	98773	1760	7220777	3038291	10823488	3474422
GC_p20_all_L0	7/10	7.0864	99525	1330	9587429	1724661	16819180	6027914
GC_p30_all_L0	10/10	8.0631	100000	1243	10826596	4660937	20195212	5159362

Verzögerte direkte Integration:

Lauf	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvR_p5_n_05_1	95	100/0	0.0222	99891	345	200013	21925	1162827	251137
GvR_p10_n_05_1	99	100/0	0.0239	99978	198	227381	13943	1883256	385822
GvR_p20_n_05_1	100	100/0	0.0278	100000	146	259114	5820	3492161	613563
GvR_p30_n_05_1	100	100/0	0.0194	100000	106	196257	18083	4566465	524849
GvR_p50_n_05_1	100	100/0	0.0272	100000	125	317757	27675	3648971	630332
GvR_p70_n_05_1	100	100/0	0.0250	100000	136	308679	46814	3418308	555444
GvR_p5_n_01_03	94	100/0	0.0203	99870	318	196562	2482	1234295	247773
GvR_p10_n_01_03	95	100/0	0.0325	99891	282	291177	4727	2356134	446680
GvR_p20_n_01_03	100	100/0	0.0211	100000	126	205882	12144	2491460	447196
GvR_p30_n_01_03	100	100/0	0.0183	100000	108	201101	17750	4668912	535697
GvR_p50_n_01_03	100	100/0	0.0247	100000	118	315915	21043	2418514	497071
GvR_p70_n_01_03	100	97/0	0.0225	100000	127	291216	19572	5702407	653813
GvR_p5_m_05_1	95	100/0	0.0453	99891	260	455413	22881	3419909	499735
GvR_p10_m_05_1	99	100/0	0.0606	99978	193	605445	25973	6488076	874718
GvR_p20_m_05_1	100	100/0	0.0625	100000	131	603664	33948	9251334	1515868
GvR_p30_m_05_1	100	100/0	0.0392	100000	101	409186	40835	7676134	998783
GvR_p50_m_05_1	100	100/0	0.0389	100000	107	483071	68187	6088117	893263
GvR_p70_m_05_1	100	100/0	0.0611	100000	137	678111	72010	9269005	1247599
GvR_p5_m_01_03	89	100/0	0.0481	99761	293	450944	27342	3627560	527452
GvR_p10_m_01_03	99	100/0	0.0697	99978	220	686262	26470	5471906	1133701
GvR_p20_m_01_03	100	100/0	0.0667	100000	135	662281	38758	13775019	1738282
GvR_p30_m_01_03	100	100/0	0.0567	100000	109	612369	42517	10049835	1463525
GvR_p50_m_01_03	100	100/0	0.0691	100000	117	733989	41212	10589687	1710947
GvR_p70_m_01_03	100	100/0	0.0539	100000	132	634340	18546	7705823	1189394
GvR_p5_h_05_1	97	100/0	0.1543	99935	206	1387792	25171	11822046	2254354
GvR_p10_h_05_1	99	100/0	0.2183	99978	155	1937921	89270	17608091	3791110
GvR_p20_h_05_1	100	100/0	0.2132	100000	99	1915697	113062	21372046	4143636
GvR_p30_h_05_1	100	100/0	0.1361	100000	82	1391297	97292	26473283	3055434
GvR_p50_h_05_1	100	100/0	0.2656	100000	117	2457053	145493	46339400	5396178
GvR_p5_h_01_03	97	100/0	0.1725	99935	234	1621534	24006	14142768	2524405
GvR_p10_h_01_03	100	100/0	0.1613	100000	130	1511526	31412	19329680	3104820
GvR_p20_h_01_03	100	100/0	0.1831	100000	90	1735833	75370	22887140	4046481
GvR_p30_h_01_03	100	100/0	0.1607	100000	87	1494466	77037	29991770	384262
GvR_p50_h_01_03	100	100/0	0.1726	100000	103	1734744	183528	32399389	4018519

Lauf	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_05_1	89	100/0	0.1266	99708	376	178849	5781	911148	136983
GvC_p10_05_1	97	100/0	0.2212	99917	341	311815	19800	2467881	357499
GvC_p20_05_1	100	100/0	0.3763	100000	311	529842	7331	2866213	593737
GvC_p30_05_1	98	100/0	0.5196	99951	305	730335	57495	4887488	500137
GvC_p50_05_1	100	99/0	0.5880	100000	253	827818	11543	3218264	634195
GvC_p70_05_1	100	100/0	0.5691	100000	218	801823	82248	3583347	665282
GvC_p90_05_1	100	97/0	1.0452	100000	285	1296392	26646	9091106	1492053
GvC_p5_01_03	96	100/0	0.1274	99901	361	172875	14079	761574	136350
GvC_p10_01_03	95	100/0	0.2351	99872	352	318646	19070	1966721	289660
GvC_p20_01_03	99	100/0	0.3399	99972	273	461024	29714	2365277	387376
GvC_p30_01_03	98	100/0	0.5509	99951	315	748316	41344	5776904	554462
GvC_p50_01_03	100	100/0	0.5764	100000	248	807597	13478	6020317	878854
GvC_p70_01_03	100	99/0	0.6514	100000	242	914053	19834	5859385	848654

B.4 Fletcher's Function

Soweit nicht anders vermerkt, wurden bei allen Jobs 100 Läufe durchgeführt. Die einzigen Ausnahmen betreffen einige Jobs mit direkter Rosenbrock-Integration.

B.4.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	Zeit[h]	RS	Noten	Gen	---------- Indivs ----------				
	folg	Ultra1		Schn	Schn	GutSchn	Schn	Min	Max	GutVari
G_p50	90		0	99514	18179	1134405	2070306	13418	10869010	1236653
G_p100	92		0	99694	16400	2304015	3804587	39086	21595288	2594155
G_p200	98		0	99819	6645	2377378	3170199	72313	42297174	3493979
G_p300	99		0	99993	2664	1326850	1941272	91737	62769066	2806149
G_p400	100		0	100000	920	1153283	1153283	145190	13415562	2432996
G_p500	100		0	100000	716	1110300	1110300	147187	12164804	2447239
G_p600	100		0	100000	222	483566	483566	191412	8609572	967312
G_p700	100		0	100000	324	752339	752339	219838	17157108	2429937
R_n	0		0	44572		0	464	123	1398	0
R_m	10		0	51213		737	1090	369	2080	283
R_h	0		60	25159		0	1318	1189	1972	0
C	10	0.00023	0	23003		886	243	11	4324	147

B.4.2 Vorinitialisierte Startpopulationen

Job	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_n_p5_20%	57	0	0.010	99882	9163	107394	15286	467354	55344
Ri_n_p5_40%	52	0	0.012	99128	9944	119253	9874	395181	77220
Ri_n_p5_100%	63	0	0.009	99687	7616	102417	13536	428033	75965
Ri_n_p10_10%	84	0	0.020	99694	9618	221316	16661	705023	163162
Ri_n_p10_20%	84	0	0.018	99597	8491	197719	15353	738658	147511
Ri_n_p10_100%	89	0	0.012	99965	5590	157236	37313	559047	114695
Ri_n_p20_5%	94	0	0.035	99792	7772	356896	28957	1497505	310474
Ri_n_p20_10%	93	0	0.025	99875	5478	253932	36476	1069234	200707
Ri_n_p20_20%	96	0	0.026	99896	5699	270933	24705	1535406	240248
Ri_n_p20_100%	97	0	0.015	99979	2956	197085	66990	986441	168327
Ri_n_p30_5%	96	0	0.032	99889	5123	354773	18986	1770747	331298
Ri_n_p30_10%	96	0	0.025	99972	4011	282040	35635	1691582	216626
Ri_n_p30_20%	97	0	0.026	99812	4040	293326	47546	1359652	267676
Ri_n_p30_100%	100	0	0.015	100000	2050	240654	124541	916275	160202
Ri_n_p40_100%	100	0	0.019	100000	1538	269767	110918	831683	134694
Ri_n_p50_100%	100	0	0.017	100000	1275	309121	160736	1593442	161304
Ri_m_p5_20%	68	11	0.007	99149	6611	79188	438	544234	59145
Ri_m_p5_40%	75	13	0.006	99708	5615	72182	5462	430931	62546
Ri_m_p5_100%	97	44		99900	9208	104689	3627	2116004	72839

Job	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_m_p10_10%	86	3	0.016	99785	8257	191196	3903	751614	180125
Ri_m_p10_20%	90	9	0.013	99799	6869	164535	5397	654473	151658
Ri_m_p10_100%	100	46		100000	2532	67186	7689	1153051	155768
Ri_m_p20_5%	91	9	0.02	99778	5723	263913	689	1177293	238608
Ri_m_p20_10%	95	9	0.02	99806	5652	265107	2527	1778791	295849
Ri_m_p20_20%	92	29	0.013	99883	3329	165159	7321	1127664	175160
Ri_m_p20_100%	100	82		100000	434	40978	18096	1018321	112397
Ri_m_p30_5%	91	14	0.026	99854	4560	312476	5422	2560542	339337
Ri_m_p30_10%	91	10	0.028	99605	4922	345656	6245	3528630	310664
Ri_m_p30_20%	99	29	0.011	99993	1729	142980	21104	1315445	191533
Ri_m_p30_100%	100	92		100000	31	34705	27100	213857	18312
Ri_m_p40_100%	100	98	0.023	100000	10	44197	37274	127074	8801
Ri_m_p50_100%	100	100		100000	0	53861	47410	60180	2630

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Com	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ci_p5_20%	94	25	0.018	99875	14586	164528	611	1003832	201309
Ci_p5_40%	94	38	0.018	99875	13121	148642	659	1015068	221351
Ci_p5_100%	94	49	0.013	99958	8368	97126	851	822201	160985
Ci_p10_10%	94	23	0.02	99792	9681	213298	566	1872951	199762
Ci_p10_20%	95	34	0.027	99882	9846	220562	617	1927405	253742
Ci_p10_100%	98	73	0.02	99903	4374	103787	896	1801203	239598
Ci_p20_5%	88	15	0.074	99583	15163	650097	708	3815951	252588
Ci_p20_10%	89	33	0.054	99673	13273	568789	658	3805736	265058
Ci_p20_20%	96	38	0.032	99889	7340	322315	783	3752832	287907
Ci_p20_30%	95	60	0.037	99789	6643	289429	797	3793117	195538
Ci_p20_40%	94	73	0.037	99792	6531	281695	639	3835386	134894
Ci_p20_50%	99	73	0.018	99910	2561	117784	819	3748119	206979
Ci_p20_60%	95	77	0.034	99882	5508	239601	825	3770025	127622
Ci_p20_70%	97	81	0.028	99979	3489	156173	1059	3746956	104419
Ci_p20_80%	96	89	0.027	99972	3193	142114	1134	3764895	54181
Ci_p20_90%	98	82	0.027	99986	2912	134892	981	3750944	192291
Ci_p20_100%	100	93	0.018	100000	665	40078	897	1613466	173683
Ci_p25_100%	99	93	0.024	99910	814	57115	1224	2998052	64788
Ci_p30_5%	98	34	0.019	100000	3395	231960	624	1811598	293872
Ci_p30_10%	93	49	0.017	100000	2825	192468	654	1080451	233645
Ci_p30_20%	94	52	0.016	100000	2922	200747	673	1167693	261213
Ci_p30_100%	100	99	0.023	100000	60	19826	1932	410022	40377
Ci_p35_100%	99	99	0.033	99910	900	83204	1501	6556112	9310
Ci_p40_100%	100	98	0.031	100000	46	25564	5543	336101	34149
Ci_p45_100%	100	99	0.032	100000	64	28593	1438	660416	64514
Ci_p50_100%	100	100	0.035	100000	0	24414	1827	45402	9238

B.4.3 Nachoptimierung

Bei der Nachoptimierung mit dem Rosenbrock-Verfahren werden zwei Werte für die Erfolgsrate angegeben, der erste basiert auf dem sonst auch üblichen Erreichen des Optimierungsziels wie in Abschn. 4.1.4 angegeben. Beim zweiten wurde die Genauigkeit, mit der das Ziel zu erreichen ist, um eine Größenordnung verringert, es genügt also ein Funktionswert von 0.0001, was einem Notenwert von 99999 entspricht.

Job	Erfolg		Noten-Schn	Noten-Verb.	Nopt Erf	Gen Schn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
NR_p10_m_P1_G1	35	61	95824	69370	35	20	2117	814	4939	592
NR_p20_m_P1_G1	46	79	99761	72262	46	28	3815	1943	6667	670
NR_p30_m_P1_G1	51	66	98125	65679	51	41	6305	4052	10321	1081
NR_p50_m_P1_G1	44	72	98961	48263	44	66	12852	6676	34752	4701
NR_p70_m_P1_G1	42	72	99035	25711	42	102	23993	15673	41155	5564
NR_p90_m_P1_G1	43	77	99712	16046	43	147	40316	17415	97559	14365
NR_p10_m_P2_G1	40	60	95504	68927	40	23	2406	1006	5050	486
NR_p20_m_P2_G1	50	70	99575	72198	50	32	4134	2579	8270	772
NR_p30_m_P2_G1	48	82	98451	62959	48	41	6200	3747	11201	1458
NR_p50_m_P2_G1	48	78	99192	48091	48	66	13085	7039	24058	3000
NR_p70_m_P2_G1	47	77	99326	25711	47	106	24480	13490	51546	6849
NR_p90_m_P2_G1	38	63	99301	16800	35	141	39020	21626	133069	23482
NR_p10_m_P3_G1	35	58	94176	67753	35	24	2389	1041	5651	592
NR_p20_m_P3_G1	49	72	98232	70187	49	36	4464	2134	8178	778
NR_p30_m_P3_G1	53	78	99672	65799	53	45	7090	3585	11268	1602
NR_p50_m_P3_G1	51	77	99387	46365	51	72	14412	7675	23799	3431
NR_p70_m_P3_G1	49	76	99402	26160	49	112	25799	13516	67293	8631
NR_p90_m_P3_G1	52	77	99242	19764	52	146	40564	21285	121981	16463
NR_p120_m_P3_G1	43	75	99434	11067	42	199	68552	38780	212688	28316
NR_p150_m_P3_G1	41	72	99265	4766	38	288	116904	59255	305991	49292
NR_p10_m_P1_G3	46	71	99032	72091	46	35	2652	1234	5879	651
NR_p20_m_P1_G3	44	74	98847	57428	44	62	5671	2523	11388	1525
NR_p30_m_P1_G3	49	69	99648	46468	49	95	10241	4286	33846	3488
NR_p50_m_P1_G3	38	69	99031	23040	38	172	25515	10102	75339	9775
NR_p70_m_P1_G3	36	70	99243	13578	36	343	62968	19230	243377	35806
NR_p90_m_P1_G3	35	72	99054	5767	35	431	99032	29503	350189	75940
NR_p120_m_P1_G3	35	63	99304	2920	32	603	179856	58403	682331	126417
NR_p10_m_P2_G3	42	68	97173	68468	42	39	2781	1047	6037	637
NR_p20_m_P2_G3	45	84	99788	65133	45	53	5116	2726	11233	1603
NR_p30_m_P2_G3	38	69	99416	52267	38	86	9520	4792	23459	3519
NR_p50_m_P2_G3	35	62	99591	24311	35	174	25372	10423	86616	11873
NR_p70_m_P2_G3	42	68	99496	11209	42	407	73479	18722	340618	45546
NR_p90_m_P2_G3	36	62	99435	8791	36	544	122634	27437	545528	110390
NR_p120_m_P2_G3	36	73	99693	1688	34	545	163046	59987	732001	104739
NR_p10_m_P3_G3	36	68	96545	67801	36	36	2714	1050	5919	631
NR_p20_m_P3_G3	51	74	99363	64442	51	57	5607	3144	10939	1238
NR_p30_m_P3_G3	41	64	99246	45406	41	94	10225	4776	25285	3003
NR_p50_m_P3_G3	44	70	98670	29594	44	192	27933	10205	87822	17834
NR_p70_m_P3_G3	44	66	99148	11871	44	362	66279	24585	364861	66935
NR_p90_m_P3_G3	37	68	99327	6077	37	502	114119	35130	535480	82598
NR_p120_m_P3_G3	44	74	99364	1368	40	552	165132	62778	767172	106726
NR_p150_m_P3_G3	44	78	99694	217	35	553	203023	88878	663030	63937

Job	Er-		Noten-		Nopt	Gen	Indivs			
	folg		Schn	Verb.	Erf	Schn	Schn	Min	Max	GutVari
NR_p50_m_P2_N5_G10	38	69	99514	11935	37	841	102727	20199	344940	67397
NR_p50_m_P2_N2_G5	35	60	99663	16310	35	437	55745	16084	251707	33470
NR_p200_m_P3_N3_G10	97	98	99904	81	10	1171	462850	100394	1482090	281008
NR_p50_m_P2_N2_G3	45	76	99594	23844	45	202	28638	9829	86319	14693

Initiale Rosenbrock-Schrittweite von 0.1 des Wertebereichs auf 0.01 verkleinert.

Nachoptimierung mit dem Complex-Algorithmus:

Bei der Complex-Nachoptimierung wurden bei dieser Benchmarkaufgabe die in Tabelle B.5 dargestellten zusätzlichen Variationen von ε und ε_{Pop} getestet. Dabei wurde zusätzlich die durchschnittliche Nischenanzahl bei Beendigung der Evolution gemessen (Ni.Schn).

Kennung	ε	ε_{Pop}
Pa	0.0005	0.005
Pb	0.001	0.01
Pc	0.003	0.03
Pd	0.005	0.05
Pe	0.01	0.1
Pf	0.01	0.35
Pg	0.01	0.5

Tab. B.5: Zusätzliche Variationen von ε und ε_{Pop}

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	Indivs		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P1_G1	49	100/0	0.001	64733	38188	49	20	1742	439	16689
NC1S_p20_P1_N2_G1	33	100/0	0.001	55197	26128	33	30	3266	1552	10088
NC1S_p30_P1_G1	45	100/0	0.001	65480	30233	45	40	5462	2363	12983
NC1S_p50_P1_G1	34	100/0	0.003	75127	21108	34	65	12632	5755	33598
NC1S_p70_P1_N5_G1	37	100/0	0.004	88587	12256	37	110	25219	13861	47968
NC1S_p90_P1_N5_G1	41	100/0	0.004	93326	4690	41	151	40709	22170	88506
NC1S_p10_P2_G1	49	100/0	0.001	64682	38095	49	21	1922	582	16749
NC1S_p20_P2_N2_G1	41	100/0	0.001	60873	31617	41	33	3456	1665	13082
NC1S_p30_P2_G1	56	100/0	0.002	72289	38517	56	45	6023	2979	15012
NC1S_p50_P2_G1	52	100/0	0.003	84647	24162	52	72	13138	6652	26298
NC1S_p70_P2_N5_G1	46	100/0	0.004	89182	12498	46	111	25000	10445	65383
NC1S_p90_P2_N5_G1	43	100/0	0.005	93805	8325	43	143	39144	22821	103308
NC1S_p10_P3_G1	53	100/0	0.001	66256	39589	53	24	1836	471	8367
NC1S_p20_P3_N2_G1	49	100/0	0.002	67823	38356	49	36	3906	1552	12254
NC1S_p30_P3_G1	47	100/0	0.002	71676	34335	47	46	6265	2894	16661
NC1S_p50_P3_G1	46	100/0	0.003	81380	24556	46	66	12640	6542	20223
NC1S_p70_P3_N5_G1	54	100/0	0.004	93212	18884	54	116	26250	14054	119940
NC1S_p90_P3_N5_G1	51	100/0	0.006	94161	4519	51	154	41598	21291	110451

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Noten- Verb.	Nopt Erf	Gen Schn	Indivs Schn	Indivs Min	Indivs Max
NC1S_p10_P1_G3	42	100/0	0.001	61263	33188	42	33	2104	686	13268
NC1S_p20_P1_N2_G3	44	100/0	0.002	69955	31063	44	50	4605	1523	12541
NC1S_p30_P1_G3	56	100/0	0.002	79186	27321	56	82	8652	3509	25842
NC1S_p50_P1_G3	43	100/0	0.005	87325	11824	43	211	29563	9529	102909
NC1S_p70_P1_N5_G3	49	100/0	0.008	90961	9476	49	327	60436	18623	276976
NC1S_p90_P1_N5_G3	46	100/0	0.015	93959	3157	45	488	111121	28479	507650
NC1S_p10_P2_G3	45	100/0	0.001	62654	34688	44	36	2071	576	8114
NC1S_p20_P2_N2_G3	53	100/0	0.002	73661	38003	53	55	4869	2136	15704
NC1S_p30_P2_G3	41	100/0	0.003	76756	24874	41	89	9546	4025	23323
NC1S_p50_P2_G3	47	100/0	0.003	83123	9883	47	193	27167	6285	91128
NC1S_p70_P2_N5_G3	42	100/0	0.007	92553	5324	42	351	64073	15281	293696
NC1S_p90_P2_N5_G3	56	99/0	0.013	96988	3941	52	466	106177	28167	362193
NC1S_p10_P3_G3	41	100/0	0.001	59027	31647	41	37	2026	568	7257
NC1S_p20_P3_N2_G3	51	100/0	0.002	78252	35394	51	60	5348	2019	15098
NC1S_p30_P3_G3	46	100/0	0.003	78505	20962	46	85	9466	3883	24782
NC1S_p50_P3_G3	46	100/0	0.004	89385	12880	46	182	26068	8167	79290
NC1S_p70_P3_N5_G3	44	100/0	0.009	94523	7550	44	318	59395	16687	247629
NC1S_p90_P3_N5_G3	44	98/0	0.014	96184	2504	42	484	108725	24630	398770

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Noten- Verb.	Nopt Erf	Gen Schn	Indivs Schn	Indivs Min	Indivs Max
NC1C_p10_P1_G1	57	100/0	0.001	72714	46154	57	20	1804	482	11559
NC1C_p20_P1_N2_G1	64	100/0	0.001	80317	51031	64	30	3466	1598	10967
NC1C_p30_P1_G1	61	100/0	0.001	80394	43757	61	40	5577	2513	12814
NC1C_p50_P1_G1	62	100/0	0.002	86341	29986	62	65	12413	5949	24692
NC1C_p70_P1_N5_G1	64	100/0	0.004	92266	13902	64	111	25001	13246	47811
NC1C_p90_P1_N5_G1	69	100/0	0.004	97379	7614	69	151	40740	22596	88442
NC1C_p10_P2_G1	65	100/0	0.001	78406	51814	65	21	1762	543	11676
NC1C_p20_P2_N2_G1	66	100/0	0.001	81146	51271	66	33	3665	1558	9703
NC1C_p30_P2_G1	75	100/0	0.002	86601	51562	75	45	5915	2683	12403
NC1C_p50_P2_G1	67	100/0	0.003	92265	28843	67	72	13286	5741	26275
NC1C_p70_P2_N5_G1	62	100/0	0.004	93204	14189	62	111	25190	10949	65705
NC1C_p90_P2_N5_G1	71	100/0	0.005	96996	10175	71	143	39113	22956	103156
NC1C_p10_P3_G1	62	100/0	0.001	72875	46047	62	24	1900	534	8345
NC1C_p20_P3_N3_G1	65	100/0	0.002	81111	51217	65	36	3775	1639	11290
NC1C_p30_P3_G1	72	100/0	0.002	84757	46123	72	46	6082	3451	13020
NC1C_p50_P3_G1	67	100/0	0.003	92265	28843	67	72	13286	5741	26275
NC1C_p70_P3_N5_G1	70	100/0	0.004	95682	17747	70	116	26162	13351	120055
NC1C_p90_P3_N5_G1	72	100/0	0.006	97421	6385	72	154	41572	21621	110368
NC1C_p10_P1_G3	53	100/0	0.001	70086	42004	53	33	2123	737	9015
NC1C_p20_P1_N2_G3	64	100/0	0.002	80549	40781	64	50	4489	1848	10676
NC1C_p30_P1_G3	75	100/0	0.002	90280	37442	75	82	8644	3405	25286
NC1C_p50_P1_G3	63	100/0	0.005	93509	17109	63	211	29515	9534	101344
NC1C_p70_P1_N5_G3	66	100/0	0.008	95559	13425	66	372	60371	19835	277577
NC1C_p90_P1_N5_G3	69	100/0	0.015	96621	5546	67	488	111052	28383	508161
NC1C_p10_P2_G3	58	100/0	0.001	72257	44316	57	36	2138	497	8092
NC1C_p20_P2_N2_G3	56	100/0	0.002	75169	38524	56	55	4767	1849	11375
NC1C_p30_P2_G3	75	100/0	0.002	90883	38171	75	89	9448	4558	19723
NC1C_p50_P2_G3	61	100/0	0.003	90522	16708	61	193	27278	7077	91165
NC1C_p70_P2_N5_G3	61	100/0	0.007	97664	9832	61	351	64008	13585	293594
NC1C_p90_P2_N5_G3	63	99/0	0.013	98050	4700	61	466	106078	32353	362049

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1C_p10_P3_G3	59	100/0	0.001	72198	44529	59	37	2062	739	9281
NC1C_p20_P3_N2_G3	67	100/0	0.002	87527	44094	67	60	5135	2368	12834
NC1C_p30_P3_G3	62	100/0	0.003	87833	29447	62	85	9355	3469	18662
NC1C_p50_P3_G3	71	100/0	0.004	95134	18095	71	182	26047	8719	78656
NC1C_p70_P3_N5_G3	68	100/0	0.008	97445	9904	68	318	58955	16732	252672
NC1C_p90_P3_N5_G3	69	98/0	0.014	98404	3875	67	848	108845	24971	399094

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p30_Pa_N5_G1	64	4.0	100/0	0.003	80762	43859	64	49	7115	2838	19706
NC1S_p30_Pb_N5_G1	51	2.4	100/0	0.002	75121	39082	51	43	6168	3167	13760
NC1S_p30_Pc_N5_G1	40	1.6	100/0	0.002	65908	30095	40	39	5454	2252	17972
NC1S_p30_Pd_N5_G1	53	1.4	100/0	0.002	72710	31437	53	37	5585	2617	15608
NC1S_p30_Pe_N5_G1	50	1.3	100/0	0.002	70902	37024	50	38	5615	2528	17186
NC1S_p30_Pf_N6_G1	55	2.5	100/0	0.004	72973	42892	55	28	5182	2409	21188
NC1S_p30_Pg_N6_G1	67	3.5	100/0	0.003	78004	51365	67	23	4903	2124	16794
NC1S_p50_Pc_N5_G1	47	1.2	100/0	0.003	83439	26979	47	68	12840	6786	27985
NC1S_p50_Pf_N6_G1	66	2.2	100/0	0.004	87493	44305	66	46	10404	5107	24583
NC1S_p50_Pg_N6_G1	64	2.9	100/0	0.002	81020	42767	64	37	8609	4835	18048
NC1C_p30_Pa_N5_G1	81	4.0	100/0	0.003	94573	54503	81	49	6681	3292	13143
NC1C_p30_Pb_N5_G1	83	2.4	100/0	0.002	94817	56521	83	43	6076	3389	14020
NC1C_p30_Pc_N5_G1	71	1.6	100/0	0.002	87891	51549	71	39	5504	2865	11949
NC1C_p30_Pd_N5_G1	63	1.4	100/0	0.002	80865	49428	63	37	5361	2556	13356
NC1C_p30_Pe_N5_G1	61	1.3	100/0	0.002	79405	44441	61	38	5626	1961	12429
NC1C_p30_Pf_N6_G1	29	2.5	100/0	0.001	52945	21020	29	28	3929	2090	11216
NC1C_p30_Pg_N6_G1	18	3.5	100/0	0.0005	42443	14646	18	23	3237	1855	9070
NC1C_p50_Pc_N5_G1	67	1.2	100/0	0.003	89522	29358	67	68	12855	7118	23593
NC1C_p50_Pf_N6_G1	27	2.2	100/0	0.002	59454	13468	27	46	9109	3927	22013
NC1C_p50_Pg_N6_G1	14	2.9	100/0	0.0007	46979	6183	14	37	7417	3739	17206

B.4.4 Direkte Integration

Job	Er- folg	Noten Schn	Gen Schn	Indivs GutSchn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	100000	42	55131	55131	2582	2782505	276972
GR_p10_n_best_L100	100	100000	17	28779	28779	6113	128293	19323
GR_p20_n_best_L100	100	100000	9	46174	46174	14704	102503	15712
GR_p30_n_best_L100	100	100000	8	61533	61533	24737	111001	18586
GR_p50_n_best_L100	100	100000	7	97025	97025	49131	187502	25412
GR_p5_n_best_L5	48/50	99999	261	225020	331047	6352	3515448	504714
GR_p10_n_best_L5	50/50	100000	122	135907	135907	26642	721372	116358
GR_p20_n_best_L5	50/50	100000	84	270951	270951	47151	4000744	558107
GR_p30_n_best_L5	50/50	100000	57	222589	222589	19160	692928	122977
GR_p5_n_best_L0	48/50	100000	475	430483	525372	59958	3685708	412494
GR_p10_n_best_L0	50/50	100000	409	800518	800518	54769	4146988	938289
GR_p20_n_best_L0	10/10	100000	168	589608	589608	159557	1387956	463727
GR_p30_n_best_L0	10/10	100000	201	1099755	1099755	179365	2950600	991919

Job	Er-	Noten	Gen	Indivs				
	folg	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GR_p5_n_all_L100	100	100000	10	42294	42294	16055	103660	18063
GR_p10_n_all_L100	100	100000	7	74830	74830	31977	165194	23889
GR_p20_n_all_L100	100	100000	5	133125	133125	42205	282041	33299
GR_p30_n_all_L100	100	100000	5	184530	184530	74275	338563	42272
GR_p50_n_all_L100	100	100000	4	277466	277466	127373	414346	51466
GR_p5_n_all_L5	49/50	100000	71	229367	283251	16913	2923553	168305
GR_p10_n_all_L5	50/50	100000	20	257932	257932	43231	1679884	233095
GR_p20_n_all_L5	50/50	100000	12	383947	383947	56209	816392	187693
GR_p30_n_all_L5	50/50	100000	12	554756	554756	245426	1222568	202306
GR_p5_n_all_L0	10/10	100000	65	288991	288991	66466	778211	213121
GR_p10_n_all_L0	100	100000	27	370511	370511	73523	1349340	242378
GR_p20_n_all_L0	50/50	100000	13	434949	434949	55432	1056264	202474
GR_p30_n_all_L0	10/10	100000	12	640854	640854	348736	1404310	333015
GR_p5_n_best_L100_Ri	100	100000	32	25364	25364	1675	100278	19369
GR_p10_n_best_L100_Ri	100	100000	15	31401	31401	12199	134029	20921
GR_p20_n_best_L100_Ri	100	100000	8	50858	50858	21950	109875	17047
GR_p30_n_best_L100_Ri	100	100000	7	74499	74499	28104	121262	19317
GR_p50_n_best_L100_Ri	100	100000	5	112319	112319	50201	186282	23907
GR_p5_m_best_L100	100	100000	3	15765	15765	2464	524999	52314
GR_p10_m_best_L100	100	100000	2	13535	13535	6264	44670	7995
GR_p20_m_best_L100	100	100000	1	20449	20449	13831	39832	4749
GR_p30_m_best_L100	100	100000	1	30147	30147	23204	60422	5973
GR_p50_m_best_L100	100	100000	1	49364	49364	38714	58387	3586
GR_p5_m_best_L100_Ri	100	100000	2	12130	12130	3067	70158	10550
GR_p10_m_best_L100_Ri	100	100000	1	15414	15414	8448	40446	7732
GR_p20_m_best_L100_Ri	100	100000	0	24503	24503	16919	45243	6937
GR_p30_m_best_L100_Ri	100	100000	0	33905	33905	28680	63743	6385
GR_p50_m_best_L100_Ri	100	100000	0	53954	53954	47677	99400	5220

Direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	100	0.006	100000	2	4684	898	20386	4090
GC_p10_best_L100	100	0.011	100000	1	7897	1074	39104	6321
GC_p20_best_L100	100	0.018	100000	1	12173	1275	32523	7517
GC_p30_best_L100	100	0.042	100000	1	16489	1622	36439	7788
GC_p50_best_L100	100	0.034	100000	1	26530	3240	53117	9837
GC_p5_all_L100	100	0.016	100000	1	12490	1109	32301	8264
GC_p10_all_L100	100	0.032	100000	1	24650	3177	49747	9839
GC_p20_all_L100	100	0.059	100000	1	45562	7356	80043	13969
GC_p30_all_L100	100	0.081	100000	1	63664	33738	112724	16536
GC_p5_best_L100_Ci	100	0.014	100000	2	9509	717	195413	21109
GC_p10_best_L100_Ci	100	0.012	100000	0	8204	1014	48751	7197
GC_p20_best_L100_Ci	100	0.018	100000	0	11624	1443	33178	6742
GC_p30_best_L100_Ci	100	0.024	100000	0	17052	1514	43242	8661
GC_p50_best_L100_Ci	100	0.033	100000	0	23691	3629	43930	9897

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GC_p5_all_L100_Ci	100	0.013	100000	1	10000	543	34940	8548
GC_p10_all_L100_Ci	100	0.016	100000	0	12655	907	55762	12197
GC_p20_all_L100_Ci	100	0.021	100000	0	15635	1125	116911	16061
GC_p30_all_L100_Ci	100	0.024	100000	0	19085	1693	132098	19851

B.4.5 Verzögerte direkte Integration

Job	Er- folg	Noten Schn	Gen Schn	Indivs GutSchn	Schn	Min	Max	GutVari
GvR_p5_m_P1	100	100000	15	17052	17052	1342	400860	46607
GvR_p10_m_P1	100	100000	22	17177	17177	3658	380584	47112
GvR_p20_m_P1	100	100000	30	19568	19568	8008	243299	28600
GvR_p5_m_P2	100	100000	13	10206	10206	1514	187150	23744
GvR_p10_m_P2	100	100000	22	14894	14894	3726	391979	42240
GvR_p20_m_P2	100	100000	32	26429	26429	8105	337721	52475
GvR_p30_m_P2	100	100000	44	23881	23881	13616	105170	11814
GvR_p50_m_P2_N3	100	100000	67	48068	48068	24318	375651	45871
GvR_p5_m_P3	100	100000	12	13261	13261	1556	220292	32479
GvR_p10_m_P3	100	100000	23	22901	22901	3618	974309	101740
GvR_p20_m_P3	100	100000	37	19971	19971	8281	295315	35023
GvR_p30_m_P3_N4	100	100000	48	38937	38937	12618	321135	49588
GvR_p50_m_P3	100	100000	72	44794	44794	25388	128881	17398
GvR_p50_m_P3_N2	100	100000	79	44788	44788	24070	250388	26783
GvR_p5_m_P1_RI	100	100000	6	12443	12443	2609	103248	14344
GvR_p10_m_P1_RI	100	100000	14	13801	13801	7779	65282	7131
GvR_p20_m_P1_RI	100	100000	14	23298	23298	18456	38817	4153
GvR_p5_m_P2_RI	100	100000	7	13121	13121	3364	76156	13218
GvR_p10_m_P2_RI	100	100000	14	12911	12911	7779	27602	3591
GvR_p20_m_P2_RI	100	100000	15	22528	22528	16569	38288	4301
GvR_p5_m_P3_RI	100	100000	8	12389	12389	3796	179377	19841
GvR_p10_m_P3_RI	100	100000	12	14438	14438	7874	61204	8237
GvR_p20_m_P3_RI	100	100000	12	22931	22931	17337	57261	5530
GvR_p30_m_P3_N4_RI	100	100000	13	33347	33347	25835	56108	5839

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvC_p5_P1	100	100/0	0.008	100000	11	6479	798	149112	16709
GvC_p10_P1	100	100/0	0.009	100000	21	7800	1309	286339	28621
GvC_p20_P1	100	100/0	0.017	100000	29	14866	3036	335114	42676
GvC_p30_P1	100	100/0	0.012	100000	39	13791	4586	160625	16607
GvC_p5_P2	100	100/0	0.009	100000	11	6698	928	214245	24231
GvC_p10_P2	100	100/0	0.008	100000	23	6975	1246	72310	9871
GvC_p20_P2	100	100/0	0.010	100000	33	10538	2832	117469	16134
GvC_p30_P2	100	100/0	0.013	100000	40	14695	5222	177886	17994
GvC_p50_P2	100	100/0	0.015	100000	73	24366	9966	217415	23020

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvC_p5_P3	100	100/0	0.005	100000	11	4635	816	42138	6671
GvC_p10_P3	100	100/0	0.013	100000	25	10892	1651	415637	41827
GvC_p20_P3	100	100/0	0.010	100000	34	10715	3194	91676	11636
GvC_p30_P3	100	100/0	0.022	100000	45	20400	3357	541815	56219
GvC_p50_P3	100	100/0	0.016	100000	69	24228	10494	205118	22440
GvC_p5_P1_Ci	100	45/0	0.02	100000	9	10694	766	360210	36938
GvC_p10_P1_Ci	100	19/0	0.018	100000	15	9883	958	130439	13743
GvC_p20_P1_Ci	100	1/0	0.025	100000	31	13491	1407	38123	7100
GvC_p5_P2_Ci	100	35/0	0.011	100000	10	6292	859	66531	9780
GvC_p10_P2_Ci	100	15/0	0.012	100000	20	8387	959	26397	5145
GvC_p20_P2_Ci	100	2/0	0.022	100000	29	13687	2216	32622	6480
GvC_p30_P2_Ci	100	3/0	0.02	100000	38	16110	1328	44118	8186
GvC_p5_P3_Ci	100	41/0	0.02	100000	9	11446	738	299392	36587
GvC_p10_P3_Ci	100	18/0	0.015	100000	16	8182	801	40768	6203
GvC_p20_P3_Ci	100	2/0	0.021	100000	9	11538	1502	35065	6253

B.4.6 Rotierte Variante von Fletcher's Function

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
G_p200	97	0.100	99277	2691	1215320	144518	22330159	342692
G_p250	99	0.085	99913	1618	944698	261596	27916616	302041
G_p300	100	0.050	100000	880	637653	289804	2747379	327316
G_p350	100	0.055	100000	821	700345	247154	1740282	307202
G_p400	100	0.061	100000	790	770853	257365	1980285	296577
G_p450	100	0.075	100000	763	843944	410032	1798068	270486
G_p500	100	0.081	100000	714	879405	405472	1852360	289942
G_p600	100	0.079	100000	672	998640	497731	1892281	285056

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
R_n	0	0.0001	40266		5549	95	20001	0
R_m	3	0.0003	35182		8290	184	20001	0
R_h	13	0.0007	24284		15113	340	20001	0
R_u	15	0.0011	25042		17671	426	20001	0
R_v	16	0.0012	26750		17895	604	20001	0
C	5	0.0002	19302		152	13	3960	144

B.5 Fraktale Funktion

Soweit nicht anders vermerkt, wurden bei allen Jobs 100 Läufe durchgeführt. Die Ausnahmen betreffen einige Jobs mit direkter Rosenbrock-Integration und alle Jobs mit direkter Complex-Integration. Ein * bei Restart (RS) bedeutet beim Rosenbrock-Verfahren eine Iterationsbegrenzung auf 5000 und beim Complex-Algorithmus auf 200000.

B.5.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p5	47	0.161	99853	10893	122566	24350	244926	50946
G_p10	97	0.243	99989	7946	179940	14580	590540	116277
G_p20	100	0.256	100000	4229	195129	18503	478935	103281
G_p30	100	0.289	100000	3162	221579	10617	501839	112337
G_p40	100	0.315	100000	2488	235001	19552	671170	135283
G_p50	100	0.323	100000	1974	236390	19030	630914	131751
G_p70	100	0.347	100000	1467	250885	14963	705822	133464
G_p90	100	0.321	100000	1086	244634	44060	696122	134901
G_p100	100	0.395	100000	1168	290951	51999	778675	151795
G_p120	100	0.340	100000	832	257143	65878	608655	115106
G_p200	100	0.368	100000	501	274202	74823	828860	134340
G_p300	100	0.348	100000	294	266658	109662	568557	74910
G_p400	100	0.454	100000	242	306082	118212	555975	82761
G_p500	100	0.638	100000	217	350749	179221	543178	87070
R_n	0		89836		1103	550	2080	0
R_m	0		89808		1614	1053	2608	0
R_h	0		88957		2665	1646	3940	0
R_u	0		90416		3416	2036	6604	0
R_v	0		89641		4320	2172	7456	0
C	0	0.0088	57102		781	532	1209	0

B.5.2 Vorinitialisierte Startpopulationen

Job	R	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	S	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_n_p10_10%	*	92	0	0.266	99982	8387	189424	26462	436298	97065
Ri_n_p10_20%	*	97	0	0.235	99996	7201	163878	13536	422757	84792
Ri_n_p10_100%	*	98	1	0.197	99999	5892	142805	10741	533041	85348
Ri_n_p20_5%	*	100	0	0.267	100000	4326	198157	46049	470526	93881
Ri_n_p20_10%	*	100	0	0.261	100000	4192	192752	13373	573943	108283
Ri_n_p20_20%	*	100	1	0.23	100000	3642	169447	5387	534082	115856
Ri_n_p20_100%	*	100	2	0.146	100000	1603	94842	20686	376639	65176
Ri_n_p30_5%	*	100	0	0.218	100000	2494	174353	10050	545619	104566
Ri_n_p30_10%	*	100	1	0.205	100000	2331	163816	3753	466861	105083
Ri_n_p30_20%	*	100	0	0.175	100000	1895	136986	8503	440081	91459
Ri_n_p30_100%	*	100	1	0.126	100000	656	78632	30329	306931	53716
Ri_n_p40_5%	*	100	0	0.225	100000	1837	173231	24724	477216	109802
Ri_n_p40_10%	*	100	0	0.204	100000	1633	156029	18832	484515	110043
Ri_n_p40_20%	*	100	0	0.133	100000	966	98649	9692	293050	61787
Ri_n_p40_30%	*	100	1	0.131	100000	841	91414	12462	446918	76836
Ri_n_p40_100%	*	100	0	0.121	100000	177	61268	41422	280139	33116

Job	R	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	S	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_n_p50_5%	*	100	0	0.243	100000	1492	177983	10559	689195	116572
Ri_n_p50_10%	*	100	0	0.206	100000	1246	151191	11334	435278	92067
Ri_n_p50_20%	*	100	0	0.121	100000	661	88899	15341	444624	77542
Ri_n_p50_30%	*	100	0	0.099	100000	422	67381	19889	206654	45884
Ri_n_p50_100%	*	100	3	0.128	100000	49	62338	52003	122761	10972
Ri_n_p70_5%	*	100	0	0.191	100000	887	153097	21314	471301	97381
Ri_n_p70_10%	*	100	1	0.146	100000	579	105799	8918	381896	77876
Ri_n_p70_20%	*	100	1	0.075	100000	173	47542	14608	189717	32497
Ri_n_p70_30%	*	100	2	0.073	100000	101	42984	21950	142934	25045
Ri_n_p70_100%	*	100	5	0.177	100000	17	82052	72388	95895	3933
Ri_n_p100_5%	*	100	0	0.204	100000	545	139953	24773	420775	84121
Ri_n_p100_10%	*	100	1	0.085	100000	153	53501	11493	352851	46990
Ri_n_p100_20%	*	100	0	0.071	100000	56	40599	23127	114420	16460
Ri_n_p100_30%	*	100	2	0.095	100000	24	43450	33249	52625	3200
Ri_n_p100_100%	*	100	6	0.248	100000	14	116642	106628	127222	4329
Ri_n_p150_5%	*	100	1	0.102	100000	140	70066	9579	214462	41589
Ri_n_p150_10%	*	100	0	0.078	100000	56	45872	24526	105080	11969
Ri_n_p150_20%	*	100	8	0.092	100000	28	50007	30566	68602	7336
Ri_n_p150_30%	*	100	2	0.122	100000	24	64372	49061	73143	4301
Ri_n_p150_100%	*	100	8	0.368	100000	12	172901	156990	187092	5801
Ri_n_p200_5%	*	100	1	0.105	100000	97	72408	11628	264051	31212
Ri_n_p200_10%	*	100	2	0.097	100000	45	55361	20331	97778	10713
Ri_n_p200_20%	*	100	1	0.123	100000	27	66858	43482	82068	6633
Ri_n_p200_100%	*	100	16	0.209	100000	10	230235	210794	250583	8313
Ri_n_p250_5%	*	100	1	0.118	100000	72	75447	15701	135935	19075
Ri_n_p250_10%	*	100	1	0.106	100000	40	65831	28377	97832	10748
Ri_n_p250_20%	*	100	7	0.164	100000	25	80402	52809	106499	9992
Ri_n_p300_5%	*	100	1	0.135	100000	69	87217	16960	177950	23108
Ri_n_p300_10%	*	100	2	0.127	100000	40	78969	33612	109100	11991
Ri_n_p300_20%	*	100	5	0.174	100000	24	95456	63297	116856	10341
Ri_m_p10_10%	*	98	0	0.254	99994	8278	187577	12268	431176	86464
Ri_m_p10_20%	*	92	0	0.264	99980	8542	194859	25741	498630	95207
Ri_m_p10_100%	*	98	0	0.216	99998	6045	151959	19073	428289	88074
Ri_m_p20_5%	*	100	0	0.243	100000	4107	188781	18861	462763	104097
Ri_m_p20_10%	*	100	0	0.22	100000	3695	171415	18202	547251	116914
Ri_m_p20_20%	*	100	0	0.221	100000	3613	170488	20384	479581	111487
Ri_m_p20_100%	*	100	3	0.178	100000	1905	119419	30289	374629	71491
Ri_m_p30_5%	*	100	0	0.227	100000	2383	168233	8212	459540	100966
Ri_m_p30_10%	*	100	1	0.24	100000	2507	177504	4174	587829	111809
Ri_m_p30_20%	*	100	0	0.189	100000	1832	136198	12704	463364	87477
Ri_m_p30_100%	*	100	4	0.145	100000	427	79886	46993	378444	45506
Ri_m_p40_5%	*	100	1	0.256	100000	2111	199166	2780	594611	118065
Ri_m_p40_10%	*	100	1	0.218	100000	1737	167477	5362	534338	100210
Ri_m_p40_20%	*	100	1	0.141	100000	968	103587	12976	351905	64420
Ri_m_p40_30%	*	100	1	0.135	100000	805	95405	19357	389137	72285
Ri_m_p40_100%	*	100	3	0.152	100000	101	76116	61013	148985	16990

Job	R	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	S	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_m_p50_5%	*	100	1	0.233	100000	1529	183494	6994	573200	123252
Ri_m_p50_10%	*	100	0	0.206	100000	1196	147883	14635	649455	94099
Ri_m_p50_20%	*	100	1	0.131	100000	609	88563	16762	270770	59826
Ri_m_p50_30%	*	100	2	0.104	100000	378	69958	24211	499439	67123
Ri_m_p50_100%	*	100	5	0.182	100000	22	86807	77513	116082	5397
Ri_m_p70_5%	*	100	1	0.19	100000	835	147146	6558	470759	91749
Ri_m_p70_10%	*	100	0	0.156	100000	600	112143	16996	490475	89006
Ri_m_p70_20%	*	100	4	0.087	100000	168	54082	21809	321224	43554
Ri_m_p70_30%	*	100	2	0.095	100000	75	50691	35592	152054	20691
Ri_m_p70_100%	*	100	7	0.253	100000	14	120285	108264	129850	4502
Ri_m_p100_5%	*	100	0	0.29	100000	464	123817	21086	604920	86542
Ri_m_p100_10%	*	100	1	0.091	100000	162	60767	16726	238767	42893
Ri_m_p100_20%	*	100	4	0.087	100000	37	46341	30734	115411	10038
Ri_m_p100_30%	*	100	2	0.12	100000	27	60126	44245	123316	7504
Ri_m_p100_100%	*	100	18	0.363	100000	11	170045	160283	182121	4752
Ri_m_p150_5%	*	100	0	0.097	100000	128	69853	26275	230619	43706
Ri_m_p150_10%	*	100	2	0.085	100000	48	50968	23779	173993	15091
Ri_m_p150_20%	*	100	1	0.127	100000	28	66802	47349	77381	4518
Ri_m_p150_30%	*	100	7	0.177	100000	19	87367	69187	100410	5731
Ri_m_p150_100%	*	100	17	0.546	100000	9	255442	239867	271868	6946
Ri_m_p200_5%	*	100	1	0.097	100000	75	67075	15938	178905	19124
Ri_m_p200_10%	*	100	3	0.105	100000	36	61285	31115	78838	8173
Ri_m_p200_20%	*	100	4	0.162	100000	25	86902	65367	100012	6331
Ri_m_p250_5%	*	100	2	0.119	100000	63	76161	20470	135667	17263
Ri_m_p250_10%	*	100	2	0.133	100000	36	76657	39933	95537	9898
Ri_m_p250_20%	*	100	4	0.204	100000	23	106518	76998	117997	7491
Ri_m_p300_5%	*	100	2	0.131	100000	60	87910	24190	149841	21229
Ri_m_p300_10%	*	100	1	0.167	100000	35	91099	49953	113202	10459
Ri_m_p300_20%	*	100	5	0.246	100000	22	127928	94950	140198	8874
Ri_h_p10_10%	*	100	0	0.205	99999	7219	164778	3151	322824	82599
Ri_h_p10_20%	*	100	0	0.24	99997	8267	190518	24918	449863	93069
Ri_h_p10_100%	1	100	0		100000	6174	163189	23212	529657	98119
Ri_h_p20_5%	*	100	0	0.225	100000	3911	180794	12660	597189	116521
Ri_h_p20_10%	*	100	0	0.2	100000	3401	159842	11320	726275	110900
Ri_h_p20_20%	*	100	0	0.23	100000	3769	181278	43614	497689	87476
Ri_h_p20_100%	3	100	0		100000	1926	137625	47768	457055	79934
Ri_h_p30_5%	*	100	0	0.251	100000	2625	186153	25388	533285	108979
Ri_h_p30_10%	*	100	0	0.243	100000	2482	178321	13215	538361	113934
Ri_h_p30_20%	*	100	1	0.189	100000	1742	135156	14937	402297	80229
Ri_h_p30_100%	4	100	0		100000	464	107640	69971	237918	34721
Ri_h_p40_5%	*	100	0	0.238	100000	1875	180021	29289	604003	99356
Ri_h_p40_10%	*	100	0	0.205	100000	1515	150921	18884	507548	95880
Ri_h_p40_20%	*	100	0	0.176	100000	1069	119440	22338	351458	73529
Ri_h_p40_100%	2	100	3		100000	79	109384	95511	267030	18654

Job	R	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	S	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_h_p50_5%	*	100	0	0.258	100000	1505	184018	16907	557103	112862
Ri_h_p50_10%	*	100	1	0.211	100000	1177	150259	12582	601146	104794
Ri_h_p50_20%	*	100	2	0.151	100000	636	100601	22834	517605	73582
Ri_h_p50_30%	*	100	1	0.146	100000	426	88840	36318	419033	66429
Ri_h_p50_100%	8	100	2		100000	29	130486	121569	189657	9006
Ri_h_p70_5%	*	100	0	0.226	100000	999	176710	17460	444933	98843
Ri_h_p70_10%	*	100	0	0.146	100000	451	95226	20604	377444	68189
Ri_h_p70_20%	*	100	2	0.123	100000	205	72708	33217	303451	51407
Ri_h_p70_30%	*	100	5	0.132	100000	58	65937	50245	208088	18457
Ri_h_p70_100%	*	100	7	0.389	100000	12	181844	171030	195821	5439
Ri_h_p100_5%	*	100	0	0.166	100000	461	127870	18989	444007	89066
Ri_h_p100_10%	*	100	0	0.095	100000	114	58608	32681	244805	31176
Ri_h_p100_20%	*	100	4	0.127	100000	38	64646	49365	162263	11972
Ri_h_p100_30%	*	100	4	0.188	100000	25	86212	72327	98391	5105
Ri_h_p100_100%	*	100	14	0.566	100000	11	258646	244593	273505	6449
Ri_h_p150_5%	*	100	0	0.118	100000	133	78551	33235	274873	43845
Ri_h_p150_10%	*	100	1	0.112	100000	45	62879	36457	96323	9026
Ri_h_p150_20%	*	100	3	0.201	100000	27	93082	71068	104179	5401
Ri_h_p150_30%	*	100	4	0.265	100000	20	127708	110515	141259	5977
Ri_h_p150_100%	*	100	26	0.873	100000	8	387335	368289	405862	8248
Ri_h_p200_5%	*	100	1	0.128	100000	83	79203	27621	287469	31680
Ri_h_p200_10%	*	100	5	0.144	100000	37	79109	47698	104205	10169
Ri_h_p200_20%	*	100	4	0.245	100000	23	121474	94998	140983	7444
Ri_h_p200_30%	*	100	6	0.371	100000	18	168119	147920	185120	7295
Ri_h_p250_5%	*	100	1	0.138	100000	65	89269	31414	141645	16568
Ri_h_p250_10%	*	100	3	0.203	100000	34	97505	55161	126851	12023
Ri_h_p250_20%	*	100	6	0.318	100000	21	150014	122441	165908	9366
Ri_h_p300_5%	*	100	1	0.162	100000	62	103259	34850	141639	16725
Ri_h_p300_10%	*	100	1	0.209	100000	34	116248	74891	137965	11033
Ri_h_p300_20%	*	100	11	0.39	100000	20	176871	142966	201210	12486

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Erfolg[%]		R	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Com	S	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ci_p5_20%	100	0	0	0.381	100000	16188	181952	6816	510425	112324
Ci_p5_40%	100	0	0	0.368	100000	15222	172001	25572	522472	102169
Ci_p5_100%	100	0	0	0.32	100000	15956	182369	21336	430460	102292
Ci_p10_10%	100	0	0	0.367	100000	8634	195307	22898	594628	116169
Ci_p10_20%	100	0	0	0.309	100000	8548	194194	25891	499699	979254
Ci_p10_30%	100	0	0	0.372	100000	8329	189974	28779	540748	105508
Ci_p10_100%	100	0	0	0.394	100000	8761	204926	31433	616963	121218
Ci_p20_5%	100	0	1	0.315	100000	4288	197583	35248	588632	107888
Ci_p20_10%	100	0	0	0.346	100000	4589	211602	28539	527796	97955
Ci_p20_20%	100	0	0	0.33	100000	4094	190626	31592	469601	102131
Ci_p20_100%	100	0	0	0.535	100000	4939	240439	16222	576198	112120
Ci_p30_5%	100	0	*	0.29	100000	2853	200691	28962	697465	123155
Ci_p30_10%	100	0	*	0.292	100000	2768	195861	37206	493189	97503
Ci_p30_20%	100	0	*	0.305	100000	2932	208550	27259	509181	98768
Ci_p30_100%	100	0	*	0.556	100000	3055	234733	37644	749894	119794

Job	Erfolg[%]		R	Zeit[h]	Noten	Gen	Indivs			
	Ges	Com	S	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ci_p40_5%	100	0	*	0.296	100000	2296	217568	28032	579580	118356
Ci_p40_10%	100	0	*	0.322	100000	2352	224079	24058	823574	129436
Ci_p40_20%	100	0	*	0.351	100000	2345	225916	34969	713600	124193
Ci_p40_100%	100	0	*	0.695	100000	2069	224839	62447	500982	102823
Ci_p50_5%	100	0	*	0.387	100000	2087	249247	38481	683892	123911
Ci_p50_10%	100	0	*	0.34	100000	1894	228393	38501	605736	126559
Ci_p50_20%	100	0	*	0.35	100000	1653	204398	29136	513718	108992
Ci_p50_100%	100	0	*	0.704	100000	1592	226952	40246	622344	108830
Ci_p70_5%	100	0	*	0.358	100000	1481	253424	38701	897323	138731
Ci_p70_10%	100	0	*	0.381	100000	1458	250636	37242	802676	147669
Ci_p70_20%	100	0	*	0.421	100000	1355	238649	27353	618690	131676
Ci_p70_100%	100	0	*	0.931	100000	1065	235018	56149	710972	114871
Ci_p100_5%	100	0	*	0.425	100000	1058	264915	23578	1040619	162255
Ci_p100_10%	100	0	*	0.427	100000	1026	259896	34665	599822	126601
Ci_p100_20%	100	0	*	0.454	100000	845	224982	21418	549302	128053
Ci_p100_100%	100	0	*	1.102	100000	573	222831	83276	626426	97144
Ci_p150_5%	100	0	*	0.429	100000	690	270831	39735	741714	139975
Ci_p150_10%	100	0	*	0.437	100000	585	238411	55582	652833	123928
Ci_p150_20%	100	0	*	0.512	100000	464	206116	63843	485826	97833
Ci_p150_100%	100	0	*	1.513	100000	279	233086	119532	392864	68420
Ci_p200_5%	100	0	*	0.398	100000	429	238998	21836	728439	125015
Ci_p200_10%	100	0	*	0.424	100000	350	208114	65634	534236	94639
Ci_p200_20%	100	0	*	0.573	100000	295	196591	75962	505762	80131
Ci_p200_100%	100	0	*	2.057	100000	166	258394	154199	397718	34580

B.5.3 Nachoptimierung

Job	Er-	RS	Noten-	Nopt	Gen	Indivs			
	folg		Schn	Verb. Erf	Schn	Schn	Min	Max	GutVari
NR_p10_m_P1_G1	0	0	93587	38234 0	36	4550	1903	7777	0
NR_p20_m_P1_G1	0	0	95715	27086 0	51	7356	4678	10463	0
NR_p30_m_P1_G1	0	0	96677	13148 0	67	10996	7455	15768	0
NR_p50_m_P1_G1	1	0	97954	5462 1	84	19213	14147	24814	0
NR_p70_m_P1_G1	0	0	98749	3818 0	97	28667	19754	40015	0
NR_p10_m_P2_G1	0	0	97046	40575 0	39	4736	2221	8634	0
NR_p20_m_P2_G1	1	1	97978	25988 0	56	8049	5095	12483	0
NR_p30_m_P2_G1	0	2	98241	13330 0	70	12160	8135	16835	0
NR_p50_m_P2_G1	1	0	98704	6022 1	85	21043	14878	100849	0
NR_p70_m_P2_G1	1	0	95748	4253 1	94	28687	19805	39830	0
NR_p90_m_P2_G1	6	0	97087	3196 6	109	40680	29799	72476	13260
NR_p120_m_P2_G1	15	0	99629	2546 13	119	55815	37927	92252	11691
NR_p10_m_P3_G1	0	0	94269	36384 0	43	4774	1841	8150	0
NR_p20_m_P3_G1	0	0	96117	20689 0	63	8670	5694	12628	0
NR_p30_m_P3_G1	2	0	96832	11107 1	74	12416	6600	17663	5645
NR_p50_m_P3_G1	0	1	98082	5276 0	87	20892	13343	27469	0
NR_p70_m_P3_G1	2	1	98885	3675 2	102	30893	22533	47422	8187
NR_p90_m_P3_G1	9	0	99334	2909 6	117	43146	29809	59010	8915

Job	Er-	R	Noten-		Nopt	Gen	Indivs			
	folg	S	Schn	Verb.	Erf	Schn	Schn	Min	Max	GutVari
NR_p10_m_P1_G3	0	0	98016	26838	0	93	6194	3128	8791	0
NR_p20_m_P1_G3	0	0	98368	10447	0	112	10818	5859	14654	0
NR_p30_m_P1_G3	3	0	98719	5713	2	128	16579	10462	22425	3017
NR_p50_m_P1_G3	5	2	99263	3245	5	158	29748	19622	62983	14799
NR_p70_m_P1_G3	19	2	99575	2690	18	182	45135	24172	68124	8592
NR_p90_m_P1_G3	26	0	99664	2187	21	204	60957	31321	99961	17181
NR_p120_m_P1_G3	59	0	99919	1816	49	254	94443	34252	142152	21652
NR_p150_m_P1_G3	75	0	99955	1762	62	267	122085	52327	176552	27123
NR_p180_m_P1_G3	91	0	99984	1560	60	292	154227	67792	257187	37657
NR_p210_m_P1_G3	97	0	99999	1395	46	287	179398	38388	293130	49648
NR_p10_m_P2_G3	1	0	97864	24392	1	97	6398	4035	9433	0
NR_p20_m_P2_G3	0	0	98289	9848	0	113	11195	7439	14666	0
NR_p30_m_P2_G3	1	0	98618	6273	1	126	16427	10714	26362	0
NR_p50_m_P2_G3	3	2	99010	4526	3	127	25506	15511	40527	5607
NR_p70_m_P2_G3	14	0	99537	2734	12	177	43673	27550	67987	9632
NR_p90_m_P2_G3	25	1	99747	2162	24	216	64584	42921	101926	15729
NR_p120_m_P2_G3	55	0	99899	1941	40	235	88056	30002	169580	24087
NR_p150_m_P2_G3	79	0	99968	1688	64	250	117663	64156	183978	22405
NR_p180_m_P2_G3	88	0	99988	1481	67	290	154623	54642	254557	33872
NR_p210_m_P2_G3	99	0	99998	1329	60	305	181206	67941	302604	46971
NR_p10_m_P3_G3	0	0	97982	26406	0	92	6391	2943	8868	0
NR_p20_m_P3_G3	0	0	98280	10137	0	110	11302	6033	17979	0
NR_p30_m_P3_G3	2	0	98665	6308	2	126	16709	10347	24024	3909
NR_p50_m_P3_G3	7	0	99247	3112	7	160	30828	18937	50956	7477
NR_p70_m_P3_G3	21	0	99590	2766	21	178	45026	29409	66329	8540
NR_p90_m_P3_G3	33	0	99762	2192	31	202	62291	36542	92257	11929
NR_p120_m_P3_G3	57	0	99907	1893	50	239	93735	52082	162852	22144
NR_p150_m_P3_G3	90	1	99991	1926	72	277	125482	49420	215471	33474
NR_p180_m_P3_G3	94	0	99992	1698	68	298	161400	68365	304278	42326
NR_p210_m_P3_G3	99	0	99999	1595	47	329	187341	74008	362395	56037
NR_p120_m_P3_N2_G3	64	0	99928	1578	51	289	103676	19099	206083	32574
NR_p120_m_P3_N2_G5	79	0	99964	1333	56	416	136016	27232	289061	52752
NR_p120_m_P3_N2_G10	95	0	99995	1311	54	706	208265	18699	446401	81208
NR_p120_m_P3_N2_G15	99	0	100000	927	25	858	250418	73398	615768	111399
NR_p120_m_P3_N2_G20	100	0	100000	833	10	713	274035	25031	613657	125217

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	Indivs		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NR_p10_u_P1_G3	1	100/0	0.024	97969	28005	1	90	10463	4750	19332
NR_p20_u_P1_G3	1	100/0	0.033	98403	10346	1	112	15793	10251	25670
NR_p30_u_P1_G3	4	100/0	0.035	98791	6155	4	126	20297	13617	31343
NR_p50_u_P1_G3	5	100/0	0.039	99316	3477	5	158	29548	19333	39800
NR_p70_u_P1_G3	14	98/0	0.059	99582	2771	12	184	45089	30495	77644
NR_p90_u_P1_G3	29	98/0	0.098	99726	2211	27	212	67275	44946	101398
NR_p120_u_P1_G3	66	94/0	0.129	99908	1941	60	240	94083	18686	134792
NR_p150_u_P1_G3	73	86/0	0.166	99957	1701	59	264	119743	24755	199436
NR_p180_u_P1_G3	94	67/0	0.208	99997	1549	61	291	151004	32298	264867
NR_p210_u_P1_G3	98	57/0	0.236	99999	1531	55	296	176134	74040	339564

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NR_p10_u_P2_G3	1	100/0	0.021	97894	26131	1	93	10119	5080	18543
NR_p20_u_P2_G3	0	100/0	0.036	98339	10256	0	115	16423	9804	30099
NR_p30_u_P2_G3	1	100/0	0.036	98698	5857	1	130	20778	14975	33191
NR_p50_u_P2_G3	3	100/0	0.052	99261	3594	3	159	33956	21326	48638
NR_p70_u_P2_G3	12	97/0	0.071	99541	2764	9	183	48662	15158	73782
NR_p90_u_P2_G3	34	98/0	0.117	99777	2215	32	214	67635	38630	108493
NR_p120_u_P2_G3	61	94/0	0.136	99925	1846	55	241	94465	49869	144121
NR_p150_u_P2_G3	81	88/0	0.165	99966	1623	69	268	122656	47584	184388
NR_p180_u_P2_G3	97	78/0	0.199	99995	1306	75	287	151370	78175	240158
NR_p210_u_P2_G3	97	63/0	0.240	99999	1426	60	302	181686	83510	285756
NR_p10_u_P3_G3	1	100/0	0.025	98036	26278	1	94	10892	4849	31683
NR_p20_u_P3_G3	2	100/0	0.032	98512	9948	2	114	15844	9809	28867
NR_p30_u_P3_G3	1	100/0	0.038	98736	6098	1	125	21245	12313	37983
NR_p50_u_P3_G3	6	100/0	0.078	99206	3574	6	151	34834	22193	57633
NR_p70_u_P3_G3	17	99/0	0.108	99634	2586	16	191	52743	31288	78262
NR_p90_u_P3_G3	38	95/0	0.114	99808	2101	33	202	67957	22015	95813
NR_p120_u_P3_G3	62	93/0	0.142	99918	1905	55	239	98544	47525	144623
NR_p150_u_P3_G3	86	84/0	0.185	99977	1792	70	284	132454	57780	203633
NR_p180_u_P3_G3	96	73/0	0.226	99995	1585	69	311	169559	37454	271839
NR_p210_u_P3_G3	100	46/0	0.269	100000	1516	46	329	204080	60712	409571

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p10_P1_G1	0	100/0	0.016	70741	15469	0	36	2823	827	4155
NC1S_p20_P1_G1	0	100/0	0.019	81537	11168	0	52	5663	2885	8409
NC1S_p30_P1_G1	0	100/0	0.024	87328	3056	0	66	9393	6117	13172
NC1S_p50_P1_G1	0	100/0	0.034	93507	342	0	85	17797	10067	24906
NC1S_p70_P1_G1	1	100/0	0.046	95882	57	1	97	27290	18735	37095
NC1S_p10_P2_G1	0	100/0	0.016	71918	15474	0	39	2989	1639	4327
NC1S_p20_P2_G1	0	100/0	0.020	81367	11214	0	55	5933	3079	8440
NC1S_p30_P2_G1	0	100/0	0.025	87888	2539	0	69	9772	6417	13919
NC1S_p50_P2_G1	0	100/0	0.034	93392	53	0	82	17494	12171	28106
NC1S_p70_P2_G1	0	100/0	0.042	95960	6	0	98	27554	17192	37439
NC1S_p10_P3_G1	0	100/0	0.016	73009	15162	0	43	3113	1711	4894
NC1S_p20_P3_G1	0	100/0	0.024	83378	8884	0	60	6412	3872	9873
NC1S_p30_P3_G1	1	99/0	0.027	88813	1485	0	72	10254	7253	14029
NC1S_p50_P3_G1	0	100/0	0.040	94159	90	0	89	18983	13680	29583
NC1S_p70_P3_G1	0	100/0	0.054	96371	20	0	101	28963	20630	40139
NC1S_p10_P1_G3	0	100/0	0.017	82865	9123	0	97	4632	2451	6607
NC1S_p20_P1_G3	0	100/0	0.023	89610	1037	0	106	8840	6002	12982
NC1S_p30_P1_G3	1	100/0	0.03	94232	63	1	128	14737	9818	21707
NC1S_p50_P1_G3	0	100/0	0.047	96906	54	0	161	28291	16822	43376
NC1S_p70_P1_G3	3	97/0	0.067	98211	0	0	190	44282	27280	72599
NC1S_p10_P2_G3	0	100/0	0.017	83095	10518	0	93	4566	2513	6309
NC1S_p20_P2_G3	0	100/0	0.023	89649	1146	0	108	9035	5483	12876
NC1S_p30_P2_G3	0	100/0	0.029	93573	42	0	124	14309	9014	20569
NC1S_p50_P2_G3	1	99/0	0.045	96950	0	0	158	27917	16231	39059
NC1S_p70_P2_G3	0	100/0	0.064	97970	0	0	179	42692	26527	67336

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P3_G3	0	100/0	0.017	81793	9080	0	94	4544	2808	7478
NC1S_p20_P3_G3	0	100/0	0.023	90137	1125	0	109	9115	5022	12361
NC1S_p30_P3_G3	0	100/0	0.032	94256	13	0	132	15218	10070	20354
NC1S_p50_P3_G3	2	98/0	0.048	96867	0	0	157	27743	9034	44475
NC1S_p70_P3_G3	2	99/0	0.069	97963	36	1	181	43442	25968	66991

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	------ Indivs --------		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1C_p10_P1_G1	0	100/0	0.009	67550	12102	0	36	2238	857	3529
NC1C_p20_P1_G1	0	100/0	0.011	79932	8667	0	52	4983	2435	7688
NC1C_p30_P1_G1	0	100/0	0.016	86635	1791	0	66	8724	5280	12518
NC1C_p50_P1_G1	0	100/0	0.031	93458	119	0	85	17141	9230	24093
NC1C_p70_P1_G1	0	100/0	0.038	95839	0	0	97	26625	17978	36311
NC1C_p10_P2_G1	0	100/0	0.009	67599	11082	0	39	2318	1258	3385
NC1C_p20_P2_G1	0	100/0	0.011	79891	8963	0	55	5188	2502	7794
NC1C_p30_P2_G1	0	100/0	0.016	87104	1188	0	69	8964	5905	12983
NC1C_p50_P2_G1	0	100/0	0.025	93354	0	0	82	16664	11077	27370
NC1C_p70_P2_G1	0	100/0	0.037	95960	0	0	98	26653	16473	36867
NC1C_p10_P3_G1	0	100/0	0.009	69891	11929	0	43	2460	1230	3781
NC1C_p20_P3_G1	0	100/0	0.012	81514	6270	0	60	5507	3171	8101
NC1C_p30_P3_G1	1	99/0	0.016	88215	541	0	72	9206	6576	12560
NC1C_p50_P3_G1	0	100/0	0.026	94155	0	0	89	17759	12907	28931
NC1C_p70_P3_G1	0	100/0	0.038	96355	0	0	101	27433	17991	38633
NC1C_p10_P1_G3	0	100/0	0.01	80127	5843	0	97	4017	2135	5891
NC1C_p20_P1_G3	0	100/0	0.015	89211	404	0	106	8195	5359	13023
NC1C_p30_P1_G3	0	100/0	0.023	94186	0	0	128	14062	9269	20917
NC1C_p50_P1_G3	0	100/0	0.039	96880	0	0	161	27631	16189	42715
NC1C_p70_P1_G3	3	97/0	0.059	98211	0	0	190	43624	26613	71992
NC1C_p10_P2_G3	0	100/0	0.01	79969	6998	0	93	3934	2504	5582
NC1C_p20_P2_G3	0	100/0	0.015	89352	553	0	108	8343	4835	12177
NC1C_p30_P2_G3	0	100/0	0.022	93577	22	0	124	13620	8272	19836
NC1C_p50_P2_G3	1	99/0	0.038	96950	0	0	158	27275	15564	38409
NC1C_p70_P2_G3	0	100/0	0.056	97970	0	0	179	41997	25891	66624
NC1C_p10_P3_G3	0	100/0	0.01	79863	6594	0	94	3936	2333	6077
NC1C_p20_P3_G3	0	100/0	0.015	89632	365	0	109	8386	4368	11745
NC1C_p30_P3_G3	0	100/0	0.022	94256	0	0	132	14348	9378	19139
NC1C_p50_P3_G3	2	98/0	0.037	96867	0	0	157	26711	9034	43131
NC1C_p70_P3_G3	1	99/0	0.056	97932	0	0	181	42270	24646	64251

B.5.4 Direkte Integration

Job	Er-	RS	Noten	Gen	----------	Indivs	----------		
	folg		Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	0	100000	6	30626	30626	8684	85575	16898
GR_p10_n_best_L100	100	0	100000	3	36271	36271	11816	97804	12843
GR_p20_n_best_L100	100	0	100000	3	60237	60237	25109	115842	15653
GR_p30_n_best_L100	100	1	100000	2	77360	77360	35736	136647	21423
GR_p40_n_best_L100	100	0	100000	2	89460	89460	42523	167429	27644
GR_p50_n_best_L100	100	1	100000	2	110108	110108	59189	176379	33233

Job	Er-folg	RS	Noten Schn	Gen Schn	Indivs GutSchn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
GR_p5_n_best_L5	100	0	100000	29	192128	192128	16951	750794	130900
GR_p10_n_best_L5	100	0	100000	15	194078	194078	11294	667415	118857
GR_p20_n_best_L5	100	0	100000	9	241964	241964	23796	596600	110762
GR_p30_n_best_L5	100	1	100000	7	293637	293637	32585	888005	141329
GR_p5_n_best_L0	10/10	0	100000	127	740724	740724	40005	2835055	840605
GR_p10_n_best_L0	10/10	0	100000	97	1036134	1036134	111819	1795447	573556
GR_p20_n_best_L0	10/10	0	100000	37	804388	804388	40669	1687954	673466
GR_p30_n_best_L0	10/10	0	100000	19	622636	622636	65344	2009491	648180
GR_p5_n_all_L100	100	0	100000	3	86861	86861	27617	213557	31579
GR_p10_n_all_L100	100	0	100000	2	138636	138636	53438	233435	33190
GR_p20_n_all_L100	50/50	0	100000	2	243305	243305	119715	364101	61560
GR_p30_n_all_L100	50/50	0	100000	2	292167	292167	162710	553361	104892
GR_p5_n_all_L5	10/10	0	100000	10	292645	292645	41375	701970	183399
GR_p10_n_all_L5	10/10	0	100000	8	444776	444776	81442	724078	190561
GR_p20_n_all_L5	10/10	0	100000	5	562185	562185	232646	1192759	271993
GR_p30_n_all_L5	10/10	0	100000	4	803546	803546	187819	1274579	387297
GR_p5_n_all_L0	10/10	0	100000	115	1744272	1744272	225049	5094342	1526253
GR_p10_n_all_L0	10/10	0	100000	30	1066833	1066833	168328	3896668	1190398
GR_p20_n_all_L0	10/10	0	100000	11	1194069	1194069	210290	2771211	713446
GR_p30_n_all_L0	10/10	0	100000	11	1793611	1793611	335775	3105285	1022687
GR_p5_m_best_L100	100	0	100000	5	39541	39541	9895	132193	22978
GR_p10_m_best_L100	100	2	100000	3	47879	47879	18346	115549	14847
GR_p20_m_best_L100	50/50	3	100000	2	79168	79168	32893	135810	22910
GR_p30_m_best_L100	50/50	0	100000	2	101866	101866	47584	155024	24826
GR_p5_h_best_L100	9/9	4	100000	4	49063	49063	25293	77291	19237
GR_p10_h_best_L100	0	4							
GR_p20_h_best_L100	11/11	4	100000	2	149779	149779	50753	308719	89544
GR_p30_h_best_L100	2/2	5	100000	2	137280	137280	102826	171735	48726

Alle Jobs wurden wegen Konvergenzprobleme des Rosenbrock-Verfahrens bei mindestens 5 Läufen abgebrochen.

Job	Er-folg	RS	Noten Schn	Gen Schn	Indivs GutSchn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
GR_p5_n_best_L100_Ri	100	0	100000	5	30025	30025	8545	87071	15712
GR_p10_n_best_L100_Ri	100	0	100000	3	43479	43479	24529	217253	21221
GR_p20_n_best_L100_Ri	100	0	100000	2	72717	72717	46484	128905	17057
GR_p30_n_best_L100_Ri	100	*	100000	2	102313	102313	34875	160182	20940
GR_p5_m_best_L100_Ri	100	2	100000	4	40038	40038	15502	78454	16843
GR_p10_m_best_L100_Ri	100	0	100000	3	56767	56767	14806	123728	15346
GR_p20_m_best_L100_Ri	100	6	100000	2	94265	94265	30703	132301	18979
GR_p30_m_best_L100_Ri	100	*	100000	2	132051	132051	49168	187093	25496

Direkte Integration mit dem Complex-Algorithmus:

Job	Er- folg	R S	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GC_p5_best_L100	10/10	0	16.15	100000	351	1067685	167772	2063547	583869
GC_p10_best_L100	50/50	0	12.94	100000	175	1065986	56287	4700997	1058718
GC_p20_best_L100	25/25	1	14.75	100000	104	1266790	87471	4446825	1178825
GC_p30_best_L100	10/10	0	16.81	100000	76	1405532	386687	2791734	884715
GC_p10_best_L100_Ci	25/25	0	13.14	100000	179	1095361	122971	4270863	1136563

B.5.5 Verzögerte direkte Integration

Job	Er- folg	RS	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvR_p5_n_P1	100	*	0.056	100000	22	31197	5785	104045	17454
GvR_p10_n_P1	100	*	0.062	100000	38	34666	10775	90709	14576
GvR_p20_n_P1	100	*	0.099	100000	57	54810	24460	208620	22646
GvR_p30_n_P1	100	*	0.130	100000	67	72684	37001	166891	27590
GvR_p5_n_P2	100	*	0.052	100000	24	30473	4972	96279	17028
GvR_p10_n_P2	100	*	0.064	100000	44	36355	12567	150518	19295
GvR_p20_n_P2	100	*	0.094	100000	59	53002	24622	114455	18230
GvR_p30_n_P2	100	*	0.139	100000	71	77922	36222	170914	24689
GvR_p5_n_P3	100	*	0.054	100000	24	31018	6707	121713	20358
GvR_p10_n_P3	100	*	0.064	100000	46	36755	11428	135492	20010
GvR_p20_n_P3	100	*	0.095	100000	62	54311	24720	131318	18438
GvR_p30_n_P3	100	*	0.129	100000	74	73605	35317	186414	25591
GvR_p5_n_P1_Ri	100	*	0.074	100000	11	41369	6381	184179	25682
GvR_p10_n_P1_Ri	100	*	0.082	100000	10	45316	10167	125570	24344
GvR_p20_n_P1_Ri	100	*	0.107	100000	13	56219	23710	129267	19773
GvR_p30_n_P1_Ri	100	*	0.122	100000	15	61776	31045	117611	17714
GvR_p5_n_P2_Ri	100	*	0.067	100000	11	37799	10114	109140	21276
GvR_p10_n_P2_Ri	100	*	0.096	100000	11	47255	19314	129371	22423
GvR_p20_n_P2_Ri	100	*	0.108	100000	13	52031	20304	118339	19720
GvR_p30_n_P2_Ri	100	*	0.129	100000	17	64608	33165	114928	13935
GvR_p5_n_P3_Ri	100	*	0.062	100000	10	34818	6570	89403	19809
GvR_p10_n_P3_Ri	100	*	0.090	100000	13	49927	21742	117000	23612
GvR_p20_n_P3_Ri	100	*	0.100	100000	15	51650	20594	99267	15982
GvR_p30_n_P3_Ri	100	*	0.127	100000	18	64039	29805	136706	15529
GvR_p5_m_P1	100	1	100000	25	64173	64173	15445	171203	33392
GvR_p10_m_P1	100	0	100000	41	75527	75527	14069	188068	34663
GvR_p20_m_P1	100	0	100000	55	101198	101198	40347	195282	32483
GvR_p5_m_P2	100	2	100000	22	37978	37978	10266	119596	20188
GvR_p10_m_P2	100	0	100000	43	41839	41839	16347	116030	17510
GvR_p20_m_P2	100	3	100000	57	67573	67573	32902	127895	20974
GvR_p30_m_P2	100	2	100000	68	92654	92654	18903	213673	32925
GvR_p40_m_P2	100	0	100000	78	117414	117414	63301	208232	37078
GvR_p50_m_P2	100	2	100000	85	137873	137873	86361	245352	45008

Job	Er- folg	RS	Noten Schn	Gen Schn	Indivs GutSchn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
GvR_p5_m_P3	100	0	100000	28	62714	62714	6988	163767	32721
GvR_p10_m_P3	100	0	100000	50	75874	75874	30401	192344	40532
GvR_p10_m_P3	100	0	100000	48	73691	73691	27676	184781	38487
GvR_p20_m_P3	100	0	100000	63	106820	106820	33471	315293	41554
GvR_p5_m_P1_Ri	100	2	100000	9	47182	47182	6294	121007	23768
GvR_p10_m_P1_Ri	100	2	100000	9	59340	59340	27908	153298	25138
GvR_p20_m_P1_Ri	100	1	100000	11	70832	70832	16854	186039	24416
GvR_p5_m_P2_Ri	100	1	100000	9	42512	42512	13308	123385	23470
GvR_p10_m_P2_Ri	100	3	100000	10	59545	59545	14907	130490	27493
GvR_p20_m_P2_Ri	100	3	100000	13	76918	76918	32004	167106	21507
GvR_p30_m_P2_Ri	100	*	100000	16	88327	88327	47104	148187	21185
GvR_p5_m_P3_Ri	100	1	100000	10	46051	46051	14140	134246	24551
GvR_p10_m_P3_Ri	100	1	100000	11	57691	57691	15844	142296	22732
GvR_p20_m_P3_Ri	100	2	100000	15	74059	74059	29757	189116	24850

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
GvC_p10_P1	10/10	10/0	11.34	100000	192	968617	191298	2384280	758079
GvC_p20_P1	10/10	10/0	11.53	100000	132	1001738	77304	2209543	614772
GvC_p30_P1	10/10	10/0	14.86	100000	126	1256165	335208	2017120	586752
GvC_p5_P2	10/10	10/0	11.81	100000	330	952620	104726	2634664	959250
GvC_p10_P2	10/10	10/0	10.93	100000	180	867312	13958	2420505	884711
GvC_p20_P2	10/10	10/0	24.12	100000	217	1998247	102300	5057010	1512359
GvC_p30_P2	10/10	10/0	14.61	100000	124	1152329	226835	2477346	823541
GvC_p50_P2	10/10	10/0	16.77	100000	134	1491626	388093	3864075	1189436
GvC_p10_P3	10/10	10/0	16.2	100000	280	1446091	252827	4989783	1436667
GvC_p20_P3	9/10	10/0	18.22	99923	191	1589390	345623	6271406	557504
GvC_p30_P3	10/10	10/0	14.2	100000	148	1259013	100764	3459229	1014619
GvC_p50_P3	10/10	10/0	19.36	100000	141	1701078	195059	4989900	1631510
GvC_p5_P1_Ci	10/10	10/0	11.64	100000	336	1011316	121559	2881853	1006897
GvC_p10_P1_Ci	10/10	10/0	12.76	100000	195	1116898	57179	3577139	1049264
GvC_p20_P1_Ci	10/10	10/0	10.31	100000	92	900626	30747	3254404	1089471
GvC_p30_P1_Ci	10/10	10/0	12.27	100000	84	1072378	81424	2771885	850848
GvC_p5_P2_Ci	10/10	10/0	15.9	100000	458	1371454	49377	7144974	2099726
GvC_p10_P2_Ci	10/10	10/0	18.79	100000	273	1574596	292448	4583058	1409333
GvC_p20_P2_Ci	10/10	10/0	10.25	100000	91	869815	89913	2459307	756734
GvC_p30_P2_Ci	10/10	10/0	12.64	100000	86	1041621	80407	3231737	992007
GvC_p5_P3_Ci	10/10	10/0	14.38	100000	473	1417191	225869	3113459	968213
GvC_p10_P3_Ci	10/10	10/0	12.15	100000	189	1069432	62992	2570884	868913
GvC_p20_P3_Ci	10/10	10/0	19.21	100000	184	1665472	52711	4166676	1421180
GvC_p30_P3_Ci	10/10	10/0	17.31	100000	105	1341551	119416	3030680	1041779

B.5.6 Rotierte Variante der Fraktalen Funktion

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p20	61	1.169	98031	15997	720360	10422	2238440	389782
G_p50	97	1.310	99900	9506	1078700	15478	5596315	973451
G_p90	98	1.772	99938	6514	1339921	21173	10084628	1323249
G_p120	100	2.058	100000	5526	1526171	31634	13410080	2653520
G_p150	100	1.897	100000	3972	1388141	52974	9681943	1758017
G_p180	100	1.499	100000	2562	1098717	41434	8060669	1354110
G_p200	100	1.485	100000	2142	1035390	59443	11780629	1701184
G_p250	100	1.039	100000	1120	717399	51756	4054997	679380
G_p300	100	0.924	100000	909	717197	69726	4678196	711152
G_p350	100	1.893	100000	1547	1337154	62321	19245022	2721476
G_p400	100	1.240	100000	810	858761	75033	7218010	974787
G_p500	100	1.424	100000	666	907913	89827	5365842	826047

Job	Er-	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
R_n	0	0.0022	86549		1090	609	2328	0
R_m	0	0.0036	87212		1666	910	5167	0
R_h	0	0.0058	87164		2380	1454	4959	0
R_u	0	0.0089	86676		3108	2023	6006	0
R_v	0	0.0131	86646		4316	2504	8192	0
C	0	0.0086	57042		769	552	1491	0

B.6 Designoptimierungsaufgabe Heterodynempfänger

Soweit nicht anders vermerkt, wurden bei allen Jobs 50 Läufe durchgeführt. Eine Ausnahme davon stellen alle Jobs des Abschnitt B.6.1 und die Jobs *GvC,p5,best,L100,P1* bis *P3* dar, die mit jeweils 100 Läufen durchgeführt wurden. Im Gegensatz zu den anderen Testaufgaben beträgt die Zielnote hier 80500.

B.6.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	RS	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg		Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p60	94		4.47	80600	141	19795	494	232195	20438
G_p90	97		2.94	80619	56	12558	90	198869	16396
G_p120	98.7		1.48	80640	18	6513	120	205014	3949
G_p150	99		1.75	80633	17	7658	150	125884	10464
G_p180	99		1.63	80629	13	7414	180	167929	10871
G_p210	100		1.28	80622	7	5773	1730	11234	2566
G_p240	100		1.75	80623	7	7220	240	20323	3927
G_p270	100		1.49	80639	6	6705	1174	15321	3613
G_p300	100		2.11	80651	6	8567	1309	23726	3959
R_n	4	1		62252		94	35	244	49
R_m	3	1		60942		232	86	1583	270
R_h	15	1		60169		764	159	2143	357
R_u	0/8	4	Wegen Konvergenzproblemen abgebrochen.						
C	12	1	0.02	59373		102	27	583	90

B.6.2 Vorinitialisierte Startpopulationen

Job	Erfolg[%]		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_n_p30_10%	8/10	10	7.528	80531	501	34599	1846	94476	30013
Ri_n_p30_20%	30/40	30	6.778	80526	445	33492	2345	99680	19040
Ri_n_p30_100%	9/11	64	6.583	80605	349	41809	8323	116676	23774
Ri_n_p50_5%	90	6	4.783	80600	166	20827	220	114621	23828
Ri_n_p50_10%	9/10	20	2.873	80631	96	14758	373	83591	26433
Ri_n_p50_20%	8/10	20	7.258	80544	263	36000	5137	119510	23209
Ri_n_p50_100%	9/10	80	5.771	80667	156	51287	29668	142282	21169
Ri_m_p30_10%	6/10	0	7.937	80484	526	41094	5866	100105	14735
Ri_m_p30_20%	8/10	30	8.813	80689	574	46985	10132	99389	31498
Ri_m_p30_100%	9/10	80	3.167	80755	97	56888	43149	87851	13023
Ri_m_p50_5%	10/10	30	4.533	80642	166	24490	3900	94261	33480
Ri_m_p50_10%	9/10	20	4.89	80701	178	28640	8120	121056	21215
Ri_m_p50_20%	9/10	50	5.252	80644	136	36832	12536	118338	33982
Ri_m_p50_100%	9/10	90	3.127	80867	18	87566	79502	110046	3755

Job	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	Ges	Ros	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_h_p10_100%	92	88		80793	21	9421	3267	29803	2833
Ri_h_p20_100%	20/20	100		80987	0	16173	9856	23650	3262
Ri_h_p30_10%	8/10	20	6.156	80596	397	32314	6003	99579	25261
Ri_h_p30_20%	4/10	20	7.423	80333	472	42225	12006	100468	4611
Ri_h_p30_100%	9/10	90	2.096	80814	15	61114	60030	70163	237
Ri_h_p50_5%	18/20	20	4.712	80603	150	22553	6003	115499	20461
Ri_h_p50_10%	18/20	20	5.736	80649	192	32101	10005	119112	32370
Ri_h_p50_20%	19/20	45	3.788	80730	109	32861	20010	136193	24151
Ri_h_p50_100%	20/20	95	3.752	80850	23	102515	100050	149346	11023
Ri_h_p70_5%	90	20	5.101	80608	127	28562	8004	170168	15322
Ri_h_p70_10%	98	34	2.939	80648	64	24619	14007	168427	23521
Ri_h_p70_20%	94	52	3.159	80695	67	38926	28014	189348	3281
Ri_h_p70_30%	20/20	70	2.988	80809	41	48595	42021	162667	26870
Ri_h_p90_5%	94	38	3.813	80674	70	34499	10005	175864	18011
Ri_h_p90_10%	98	46	2.643	80692	41	43103	18009	192755	24866
Ri_h_p90_20%	100	60	2.17	80788	19	72690	36018	173735	30914
Ri_h_p120_5%	100	34	1.558	80656	12	16724	12006	75133	10048
Ri_h_p120_10%	98	62	1.71	80756	11	28258	24012	125431	6980
Ri_h_p120_20%	96	80	3.205	80775	24	54990	48024	193814	18020

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Erfolg[%]		Zeit[h]	Noten	Gen	Indivs			
	Ges	Compl	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ci_p5_20%	36	16	2.058	79528	1145	12435	51	22074	5636
Ci_p5_100%	33	52	1.156	80475	614	7279	382	22271	952
Ci_p40_5%	88	26	4.141	80632	205	18879	231	109543	15184
Ci_p40_10%	98	56	1.046	80769	43	4879	387	110214	6600
Ci_p40_20%	96	68	1.114	80779	59	6342	506	106039	4987
Ci_p40_30%	98	74	0.56	80808	7	2231	967	15358	1059
Ci_p40_100%	100	100	0.877	81060	0	4974	2569	8006	1247
Ci_p50_5%	98	44	2.184	80734	73	9096	380	109587	20448
Ci_p50_10%	98	50	1.763	80736	62	7892	197	110388	18310
Ci_p50_20%	98	68	1.607	80827	49	6962	591	109846	10425
Ci_p50_30%	96	88	1.16	80920	23	4744	794	116341	774
Ci_p50_100%	100	100	1.473	81079	0	6416	3887	9055	1398
Ci_p70_5%	98	46	1.392	80743	42	7957	365	156774	9260
Ci_p70_10%	100	62	0.541	80844	7	2455	482	21351	3363
Ci_p70_20%	98	78	1.562	80885	32	6982	1054	155415	11544
Ci_p70_30%	100	98	0.488	80955	0	2731	1388	7178	900
Ci_p70_100%	100	100	2.356	81093	0	9176	6362	13342	1430
Ci_p90_5%	98	54	1.083	80738	23	6103	402	160770	6217
Ci_p90_10%	100	78	0.824	80856	14	4548	598	94918	14208
Ci_p90_20%	100	94	0.437	80996	0	2413	1263	4484	735
Ci_p90_30%	100	96	0.689	81000	0	3539	1609	7134	997
Ci_p90_100%	20/20	100	2.91	81106	0	11088	8842	15338	1524
Ci_p120_5%	100	60	0.694	80775	4	2561	476	9734	2519
Ci_p120_10%	100	80	0.539	80868	1	2034	731	6050	1273
Ci_p120_20%	100	100	0.812	81006	0	3024	1352	4736	845
Ci_p120_30%	100	100	1.194	81006	0	4417	2190	6871	1035

B.6.3 Nachoptimierung

Job	Er-	Noten-		Nopt	Gen	Indivs			
	folg	Schn	Verb.	Erf	Schn	Schn	Min	Max	GutVari
NR_p10_h_P1_G1	59	80141	3089	38	16	1618	10	4778	1193
NR_p20_h_P1_G1	68	80331	667	42	24	2408	231	5090	1302
NR_p30_h_P1_G1	76	80507	492	16	32	2910	729	9782	1525
NR_p50_h_P1_G1	76	80550	154	8	41	4634	655	15332	2687
NR_p70_h_P1_N5_G1	90	80589	100	2	73	4910	301	22904	3976
NR_p10_h_P2_G1	54	80035	1396	34	17	1443	228	3865	943
NR_p20_h_P2_G1	66	80424	948	20	26	2123	255	5892	1384
NR_p30_h_P2_G1	84	80593	657	26	30	2833	257	7805	2115
NR_p50_h_P2_G1	84	80620	447	12	49	4427	597	14131	3034
NR_p70_h_P2_N5_G1	90	80618	59	1	69	3885	70	22760	1956
NR_p10_h_P3_G1	70	80539	1923	42	19	1565	80	4072	1008
NR_p20_h_P3_G1	70	80435	701	28	25	2193	166	5701	1513
NR_p30_h_P3_G1	86	80648	627	30	35	3243	491	8011	2279
NR_p50_h_P3_G1	82	80591	223	14	47	4381	194	13510	3336
NR_p70_h_P3_N5_G3	92	80634	291	6	66	5124	307	16381	4331
NR_p10_h_P1_G3	62	80135	1362	35	26	1573	172	5018	1113
NR_p20_h_P1_G3	60	80331	1031	34	32	2514	87	5932	1646
NR_p30_h_P1_G3	70	80455	422	18	39	3244	605	8546	2071
NR_p50_h_P1_G3	76	80548	164	8	63	4671	10	14352	2845
NR_p70_h_P1_N5_G3	92	80602	281	2	75	3999	561	18312	3415
NR_p10_h_P2_G3	54	79801	1281	36	26	1579	190	3834	925
NR_p20_h_P2_G3	56	80226	849	32	37	2702	491	5399	1387
NR_p30_h_P2_G3	72	80533	498	20	44	2947	356	9605	2096
NR_p50_h_P2_G3	78	80577	245	12	76	5658	439	17110	4174
NR_p70_h_P2_N5_G3	96	80628	310	3	103	4551	292	29563	5231
NR_p10_h_P3_G3	64	80138	1331	40	28	1601	289	3515	941
NR_p20_h_P3_G3	60	80186	660	28	36	2674	266	6227	1718
NR_p30_h_P3_G3	80	80577	708	28	44	3175	261	8900	2350
NR_p50_h_P3_G3	94	80652	512	18	41	3952	633	14359	3507
NR_p70_h_P3_N5_G3	88	80581	132	2	98	5899	600	41471	4148

Nachoptimierung mit dem Complex-Algorithmus:

Bei der Nachoptimierung mit dem Complex-Algorithmus kommen die in Tabelle B.5 auf Seite 236 beschriebenen Sonderparametrierungen *Pa* bis *Pe* für die Nischenbildung ebenfalls zum Einsatz. Bei einigen Jobs wurde zusätzlich die durchschnittliche Nischenanzahl bei Beendigung der Evolution gemessen (Ni.Schn).

Job	Er-	Nisch.	Zeit[h]	Noten-		Nopt	Gen	Indivs		
	folg	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P1_G1	52	41/0	0.151	79976	1174	34	20	834	217	1644
NC1S_p20_P1_G1	68	29/0	0.396	80451	642	26	36	1885	171	6561
NC1S_p30_P1_G1	76	22/0	0.377	80564	535	20	40	2450	139	6482
NC1S_p50_P1_G1	90	11/0	0.742	80586	414	12	58	3648	412	12683
NC1S_p70_P1_N5_G1	90	10/0	0.96	80624	402	10	64	5198	874	17564

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p10_P2_G1	50	40/0	0.163	79527	918	30	22	897	232	1781
NC1S_p20_P2_G1	60	33/0	0.381	80492	423	26	35	1970	164	5154
NC1S_p30_P2_G1	76	26/0	0.49	80565	667	28	39	2617	148	6608
NC1S_p50_P2_G1	82	15/0	0.794	80570	250	12	55	3899	391	12857
NC1S_p70_P2_N5_G1	92	5/0	0.783	80583	113	2	75	3924	885	18742
NC1S_p10_P3_G1	46	42/0	0.095	79520	1030	30	22	880	124	2091
NC1S_p20_P3_G1	68	32/0	0.467	80381	540	32	37	1994	293	4893
NC1S_p30_P3_G1	76	27/0	0.793	80567	521	30	40	2873	360	6199
NC1S_p50_P3_G1	88	13/0	0.928	80631	492	14	51	4065	609	12171
NC1S_p70_P3_N5_G1	82	9/0	1.476	80581	0	0	83	6374	896	21300

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p10_P1_G3	60	1.0	36/0	0.173	80098	976	32	24	876	64	1582
NC1S_p20_P1_G3	70	1.1	30/0	0.369	80152	449	30	34	1862	90	3759
NC1S_p30_P1_G3	68	1.0	22/0	0.582	80366	374	12	53	2941	30	7224
NC1S_p50_P1_G3	76	1.0	14/0	0.896	80564	69	4	71	4520	225	15671
NC1S_p70_P1_N5_G3	92	1.0	5/0	0.846	80618	105	2	88	4237	307	21958
NC1S_p10_P2_G3	48	1.1	39/0	0.197	79700	623	26	27	992	252	2612
NC1S_p20_P2_G3	62	1.0	30/0	0.379	80340	581	22	33	1815	184	3942
NC1S_p30_P2_G3	62	1.1	24/0	0.617	80502	131	10	50	3039	509	7486
NC1S_p50_P2_G3	80	1.0	15/0	0.83	80543	244	10	63	4273	214	19039
NC1S_p70_P2_N5_G3	92	1.1	8/0	1.027	80618	386	8	87	5343	597	22310
NC1S_p10_P3_G3	50	1.1	41/0	0.224	79530	842	32	25	961	381	2348
NC1S_p20_P3_G3	56	1.4	31/0	0.422	80255	337	18	36	1869	313	4517
NC1S_p30_P3_G3	68	1.5	27/0	0.678	80410	402	22	46	3014	389	6513
NC1S_p50_P3_G3	86	1.6	15/0	1.150	80598	287	16	76	4984	445	15043
NC1S_p70_P3_N5_G3	96	1.5	2/0	0.811	80597	0	0	112	3447	564	25372

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1C_p10_P1_G1	62	41/0	0.139	80054	1172	44	20	764	217	1552
NC1C_p20_P1_G1	70	29/0	0.385	80455	606	28	36	1814	171	6599
NC1C_p30_P1_G1	82	22/0	0.433	80573	521	26	40	2360	139	6582
NC1C_p50_P1_G1	90	11/0	0.732	80579	377	12	58	3599	412	12612
NC1C_p70_P1_N5_G1	96	10/0	0.952	80629	421	16	64	5152	874	17263
NC1C_p10_P2_G1	54	40/0	0.150	79901	1334	34	22	823	232	1619
NC1C_p20_P2_G1	74	33/0	0.368	80561	507	40	35	1900	164	4463
NC1C_p30_P2_G1	82	26/0	0.476	80497	521	34	39	2538	148	6737
NC1C_p50_P2_G1	86	15/0	0.786	80598	336	16	55	3853	391	12850
NC1C_p70_P2_N5_G1	98	5/0	0.784	80596	243	8	75	3927	885	18871
NC1C_p10_P3_G1	64	42/0	0.157	79926	1473	48	22	827	124	2357
NC1C_p20_P3_G1	74	37/0	0.448	80472	628	38	37	1909	293	4114
NC1C_p30_P3_G1	82	27/0	0.614	80643	673	36	40	2771	360	6348
NC1C_p50_P3_G1	92	13/0	0.911	80598	350	18	51	3991	609	12163
NC1C_p70_P3_N5_G1	94	9/0	1.479	80630	274	12	83	6383	896	21267

Job	Er- folg	Ni. Schn	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1C_p10_P1_G3	70	1.0	36/0	0.159	80062	889	42	24	799	64	1650
NC1C_p20_P1_G3	76	1.1	30/0	0.353	80428	904	36	34	1781	90	3581
NC1C_p30_P1_G3	70	1.0	22/0	0.571	80293	206	14	53	2882	30	7180
NC1C_p50_P1_G 3	86	1.0	14/0	0.897	80634	319	14	71	4521	225	15624
NC1C_p70_P1_N5_G3	98	1.0	5/0	0.847	80639	313	8	88	4237	307	22013

Job	Er-	Ni.	Nisch.	Zeit[h]		Noten-	Nopt	Gen	------ Indivs ------		
	folg	Schn	GDV/GAk	Ultra1		Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1C_p10_P2_G3	68	1.1	39/0	0.180		80199	1210	46	27	902	252	1662
NC1C_p20_P2_G3	68	1.0	30/0	0.369		80371	599	28	33	1757	184	3834
NC1C_p30_P2_G3	78	1.1	24/0	0.615		80572	275	26	50	3021	509	7332
NC1C_p50_P2_G3	92	1.0	15/0	0.826		80594	393	22	63	4252	214	18647
NC1C_p70_P2_N5_G3	96	1.1	8/0	1.022		80618	435	12	87	5312	597	21868
NC1C_p10_P3_G3	62	1.1	41/0	0.209		79979	1336	44	25	894	374	1978
NC1C_p20_P3_G3	84	1.4	31/0	0.421		80486	706	46	36	1859	313	3965
NC1C_p30_P3_G3	82	1.5	27/0	0.658		80555	667	36	46	2925	389	6211
NC1C_p50_P3_G3	94	1.6	15/0	1.134		80638	415	24	76	4912	445	15243
NC1C_p70_P3_N5_G3	100	1.5	2/0	0.813		80624	678	4	112	3452	564	25423
NC1S_p20_Pa_N5_G1	86	3.8	22/0	0.334		80593	1011	30	32	1721	162	4479
NC1S_p20_Pb_N5_G1	64	1.5	28/0	0.364		80371	590	20	32	1672	107	4289
NC1S_p20_Pc_N5_G1	72	1.2	30/0	0.339		80500	643	32	27	1705	340	3097
NC1S_p20_Pd_N5_G1	64	1.2	35/0	0.444		80449	524	34	33	2028	322	4014
NC1S_p20_Pe_N5_G1	74	1.1	27/0	0.382		80533	672	28	29	1628	80	4214
NC1S_p30_Pa_N5_G1	80	4.4	25/0	0.548		79750	766	30	43	2916	245	8862
NC1S_p30_Pb_N5_G1	86	1.7	15/0	0.413		80560	496	16	34	2096	247	5257
NC1S_p30_Pc_N5_G1	76	1.2	24/0	0.51		80573	429	24	41	2608	124	7018
NC1S_p30_Pd_N5_G1	68	1.2	27/0	0.542		80530	460	22	38	2796	375	5966
NC1S_p30_Pe_N5_G1	70	1.3	25/0	0.449		80462	489	20	34	2269	389	5664
NC1S_p50_Pa_N8_G1	92	5.1	14/0	0.857		80632	585	20	58	4322	218	13060
NC1S_p50_Pb_N8_G1	94	2.1	7/0	0.652		80593	471	8	54	3280	422	10995
NC1S_p50_Pc_N8_G1	86	1.1	12/0	0.746		80574	275	10	50	3273	405	10990
NC1S_p70_Pb_N8_G1	94	2.9	8/0	1.136		80626	417	10	62	4694	276	16165
NC1C_p20_Pa_N5_G1	90	3.8	22/0	0.298		80650	1013	34	32	1534	162	3885
NC1C_p20_Pb_N5_G1	78	1.5	28/0	0.350		80474	760	34	32	1602	107	4467
NC1C_p20_Pc_N5_G1	78	1.2	30/0	0.322		80538	674	38	29	1583	340	3000
NC1C_p20_Pd_N5_G1	72	1.2	35/0	0.424		80473	503	42	33	1935	322	3482
NC1C_p20_Pe_N5_G1	88	1.1	27/0	0.361		80565	721	42	29	1537	80	3244
NC1C_p30_Pa_N5_G1	94	4.0	24/0	0.527		80691	1041	42	44	2746	245	8939
NC1C_p30_Pb_N5_G1	84	1.7	15/0	0.399		80556	461	14	34	2027	247	4838
NC1C_p30_Pc_N5_G1	86	1.2	24/0	0.499		80611	481	34	41	2546	124	6554
NC1C_p30_Pd_N5_G1	80	1.2	27/0	0.532		80582	466	34	38	2740	375	6102
NC1C_p30_Pe_N5_G1	78	1.3	25/0	0.433		80510	521	26	34	2185	389	5489
NC1C_p50_Pa_N8_G1	98	5.1	14/0	0.824		80634	552	26	58	4151	218	11383
NC1C_p50_Pb_N8_G1	98	2.1	7/0	0.649		80639	796	12	54	3258	422	10864
NC1C_p50_Pc_N8_G1	96	1.1	12/0	0.747		80655	580	20	50	3276	405	10887
NC1C_p70_Pb_N8_G1	98	2.9	8/0	1.126		80626	412	14	62	4651	276	16183

B.6.4 Direkte Integration

Job	Er-	Noten	Gen	Indivs				
	folg	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GR_p5_n_best_L100	100	80731	27	11533	11533	455	71299	13865
GR_p10_n_best_L100	100	80743	11	9203	9203	774	42692	10015
GR_p20_n_best_L100	100	80744	3	5476	5476	1510	33862	7233
GR_p30_n_best_L100	100	80755	3	9894	9894	2410	56701	10246
GR_p5_n_all_L100	100	80788	3	5810	5810	1222	32836	5465
GR_p10_n_all_L100	100	80849	1	8464	8464	3471	21636	3405
GR_p20_n_all_L100	100	80839	1	14551	14551	7853	31431	6057
GR_p30_n_all_L100	100	80900	1	18526	18526	10662	36335	4365
GR_p5_n_best_L5	100	80738	29	13723	13723	358	155886	23947
GR_p10_n_best_L5	100	80732	8	7447	7447	10	39118	8603
GR_p20_n_best_L5	100	80769	2	5278	5278	1617	21317	4241
GR_p30_n_best_L5	100	80755	2	6096	6096	2492	32375	5281
GR_p5_n_all_L5	100	80742	3	7028	7028	1552	18813	4530
GR_p10_n_all_L5	100	80800	1	7738	7738	3606	26763	4285
GR_p20_n_all_L5	100	80818	1	13777	13777	8290	37050	4867
GR_p30_n_all_L5	100	80885	1	16271	16271	2795	33436	5781
GR_p5_m_best_L100	100	80834	6	6418	6418	639	69168	11131
GR_p10_m_best_L100	100	80848	2	5421	5421	1376	36378	5884
GR_p20_m_best_L100	100	80876	2	7963	7963	2845	22012	4976
GR_p30_m_best_L100	100	80929	1	7790	7790	4111	17513	2788
GR_p5_m_best_L5	100	80827	4	5303	5303	767	73235	10437
GR_p10_m_best_L5	100	80876	2	4222	4222	1425	12721	2547
GR_p20_m_best_L5	100	80886	1	6524	6524	2778	15641	3001
GR_p30_m_best_L5	100	80869	1	8102	8102	4378	22090	3170
GR_p5_m_best_L0	100	80842	4	4992	4992	706	23083	4601
GR_p10_m_best_L0	100	80862	2	4694	4694	10	37755	5559
GR_p20_m_best_L0	100	80870	1	6406	6406	3007	16529	3215
GR_p30_m_best_L0	100	80895	1	9210	9210	4506	21120	3724
GR_p5_m_best_L100_Ri	100	80819	5	8047	8047	789	59740	12755
GR_p10_m_best_L100_Ri	100	80829	1	6095	6095	1657	22043	4330
GR_p20_m_best_L100_Ri	100	80836	0	7100	7100	3318	20958	3834
GR_p5_m_best_L5_Ri	100	80784	4	6206	6206	716	70756	10254
GR_p10_m_best_L5_Ri	100	80868	1	4741	4741	1390	19574	3506
GR_p20_m_best_L5_Ri	100	80841	1	7908	7908	3207	21338	4311
GR_p30_m_best_L5_Ri	100	80833	0	9116	9116	4610	21938	3177
GR_p5_h_best_L100	100	80902	2	5996	5996	1542	22032	4259
GR_p10_h_best_L100	100	80916	1	8647	8647	4086	18224	3213
GR_p20_h_best_L100	100	80988	1	14140	14140	8814	22893	2519
GR_p30_h_best_L100	100	81019	1	21102	21102	7853	34173	4589
GR_p5_h_best_L5	100	80861	2	5248	5248	1148	25902	4345
GR_p10_h_best_L5	100	80898	1	8005	8005	2859	18609	2966
GR_p10_h_best_L0	100	80893	1	8131	8131	10	16861	2880

Direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Zeit[h]	Noten	Gen	Indivs			
	folg	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	100	0.243	80923	2	1041	400	2453	495
GC_p10_best_L100	100	0.244	80938	1	1384	490	2881	583
GC_p20_best_L100	100	0.597	81014	1	2690	1349	5456	814
GC_p30_best_L100	100	0.897	81035	1	4059	2305	7168	1193
GC_p5_best_L5	100	0.303	80899	2	1325	460	3926	723
GC_p10_best_L5	100	0.4	80925	1	1720	693	4387	772
GC_p20_best_L5	100	0.584	80982	1	2859	1324	5875	1055
GC_p30_best_L5	100	0.882	81049	1	3858	1824	6051	970
GC_p5_best_L0	100	0.239	80884	2	1054	453	2206	430
GC_p10_best_L0	100	0.35	80943	1	1546	649	2997	517
GC_p20_best_L0	100	0.541	80972	1	2368	1343	4518	667
GC_p30_best_L0	100	0.826	81059	1	3618	1701	6416	1079
GC_p5_all_L100	100	0.64	81005	1	2805	1195	4814	879
GC_p10_all_L100	100	1.23	81034	1	5413	1588	10044	1764
GC_p20_all_L100	100	2.34	81102	1	10168	2318	15723	2583
GC_p5_best_L100_Ci	100	0.249	80885	1	1083	431	2312	453
GC_p10_best_L100_Ci	100	0.359	80954	0	1559	661	4416	661
GC_p20_best_L100_Ci	100	0.662	80982	0	2859	1210	6934	1012
GC_p5_best_L5_Ci	100	0.293	80924	1	1274	422	3119	645
GC_p10_best_L5_Ci	100	0.39	80941	0	1702	567	3613	659
GC_p20_best_L5_Ci	100	0.606	80982	0	2646	1284	4979	906
GC_p5_best_L0_Ci	100	0.321	80974	1	1410	466	3645	670
GC_p10_best_L0_Ci	100	0.385	80945	0	1695	438	3961	842
GC_p20_best_L0_Ci	100	0.619	80998	0	2699	1035	5184	969
GC_p5_all_L100_Ci	100	0.461	80970	1	2037	429	5645	1237
GC_p10_all_L100_Ci	100	0.616	80961	0	2668	627	8236	2268
GC_p20_all_L100_Ci	100	0.793	81002	0	3423	1278	14862	2970

B.6.5 Verzögerte direkte Integration

Job	Erfolg		Noten	Gen	Indivs				
	Ges	Evo	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GvR_p10_n_L100_P1	100	22	80680	28	12438	12438	48	171488	26670
GvR_p10_n_L100_P2	100	16	80713	34	11333	11333	94	75369	13515
GvR_p5_m_L100_P1	100	4	80823	15	4571	4571	275	36065	7752
GvR_p10_m_L100_P1	100	12	80768	19	7445	7445	36	99996	16138
GvR_p20_m_L100_P1_N2	100	34	80753	26	4739	4739	264	25865	4555
GvR_p30_m_L100_P1_N2	100	44	80726	28	5689	5689	127	19807	4704
GvR_p5_m_L100_P2	100	0	80777	17	5180	5180	720	63883	9864
GvR_p10_m_L100_P2	100	30	80764	17	3725	3725	55	27945	4644
GvR_p20_m_L100_P2_N2	100	30	80774	24	4730	4730	264	24965	4793
GvR_p30_m_L100_P2_N2	100	52	80686	27	5197	5197	127	17570	4707

Job	Erfolg		Noten	Gen	Indivs				
	Ges	Evo	Schn	Schn	GutSchn	Schn	Min	Max	GutVari
GvR_p5_m_L5_P1	100	2	80837	18	5174	5174	114	68809	10363
GvR_p10_m_L5_P1	100	24	80786	22	3518	3518	171	18875	3478
GvR_p20_m_L5_P1_N2	100	36	80740	26	5252	5252	250	29812	5855
GvR_p30_m_L5_P1_N2	100	58	80686	24	4261	4261	130	12330	3800
GvR_p5_m_L5_P2	100	8	80803	16	4534	4534	135	62139	9381
GvR_p10_m_L5_P2	100	20	80759	21	4779	4779	229	25114	4863
GvR_p20_m_L5_P2_N2	100	40	80712	22	3634	3634	260	11927	2723
GvR_p30_m_L5_P2_N2	100	44	80727	28	5987	5987	266	19594	5026
GvR_p5_m_L5_P3	100	2	80760	18	7482	7482	241	68105	13463
GvR_p10_m_L5_P3	100	16	80808	21	4513	4513	314	21166	4671
GvR_p20_m_L5_P3_N2	100	44	80724	26	7014	7014	403	86776	13739
GvR_p30_m_L5_P3_N2	100	60	80683	27	4697	4697	363	17717	4535
GvR_p5_h_L100_P1	100	8	80789	13	2962	2962	131	7904	1615
GvR_p10_h_L100_P1	100	30	80802	17	3753	3753	192	9590	2630
GvR_p20_h_L100_P1_N2	100	51	80733	21	5281	5281	82	15608	4875
GvR_p30_h_L100_P1_N2	100	56	80716	25	7136	7136	260	17643	6718
GvR_p5_h_L100_P2	100	0	80874	12	3346	3346	1283	12179	1861
GvR_p10_h_L100_P2	100	20	80856	22	4613	4613	192	15604	3019
GvR_p20_h_L100_P2_N2	100	44	80797	22	5941	5941	324	17245	5066
GvR_p30_h_L100_P2_N2	100	42	80799	29	9881	9881	146	23767	7697
Alle Ri:									
GvR_p5_h_L100_P1	100	62	80833	4	5887	5887	1659	21069	3916
GvR_p10_h_L100_P1	100	84	80919	3	8414	8414	3624	16839	2600
GvR_p20_h_L100_P1_N2	100	100	80972	0	14990	14990	11101	19287	2093
GvR_p5_h_L100_P2	100	64	80879	5	6490	6490	2339	35535	6551
GvR_p10_h_L100_P2	100	84	80914	3	9053	9053	4839	26345	3949
GvR_p20_h_L100_P2_N2	100	100	80972	0	14990	14990	11101	19287	2093

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Erfolg		Zeit[h]	Noten	Gen	Indivs			
	Ges	Evo	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P1	100	5	0.225	80921	12	1103	61	2903	541
GvC_p10_P1	100	14	0.382	80920	19	1853	42	4034	890
GvC_p20_P1	100	44	0.55	80823	24	2995	82	7955	2364
GvC_p5_P2	100	8	0.194	80894	13	1088	174	2259	486
GvC_p10_P2	100	20	0.334	80870	20	1716	154	3793	883
GvC_p20_P2	100	42	0.63	80860	25	3468	252	8954	2435
GvC_p5_P3	100	24	0.192	80886	14	996	118	3627	553
GvC_p10_P3	100	20	0.338	80894	18	1708	202	3461	803
GvC_p20_P3	100	24	0.692	80922	27	3754	335	7577	2058
GvC_p30_P3	100	48	1.213	80831	31	5089	30	14038	3747
GvC_p5_P1_Ci	100	68	0.244	80880	5	1230	406	2828	582
GvC_p10_P1_Ci	100	78	0.362	80997	5	1835	744	4336	886
GvC_p20_P1_Ci	100	90	0.588	80969	3	2975	1184	6524	1102

Job	Erfolg		Zeit[h]	Noten	Gen	---------- Indivs ----------			
	Ges	Evo	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p5_P2_Ci	100	64	0.200	80840	6	1012	232	2354	475
GvC_p10_P2_Ci	100	86	0.324	80911	4	1632	615	5164	869
GvC_p20_P2_Ci	100	92	0.545	80979	2	2759	1357	7504	1186
GvC_p5_P3_Ci	100	52	0.238	80911	7	1284	266	2757	632
GvC_p10_P3_Ci	100	66	0.376	80925	8	1898	615	4867	1084
GvC_p20_P3_Ci	100	92	0.537	80944	3	2743	1107	7775	1349

B.7 Ressourcenoptimierung

Soweit nicht anders vermerkt, wurden bei allen Jobs 50 Läufe durchgeführt. Davon abweichend basieren die GLEAM-Jobs ab *p1000* und *GvR,p5,s,best,L100,P7* auf 100 Läufen. Bei der Nachoptimierung und der verzögerten direkten Integration kommen die in Tabelle 5.6 auf Seite 112 beschriebenen Sonderparametrierungen *P5* und *P7* für die Nischenbildung ebenfalls zum Einsatz.

B.7.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-folg	RS	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
G_p600	68	0	3.29	98293	880	3111704	1133752	7816072	720305
G_p800	76	0	4.19	98967	835	3912385	1291120	8630096	1324606
G_p1000	94	0	4.70	99783	630	3773443	1566558	9485104	1115809
G_p1200	94	0	4.72	99802	614	4403147	2076361	13464404	1413470
G_p1400	99	0	5.11	99943	570	4818266	2382778	15517145	1893232
G_p1600	98	0	5.97	99953	531	5159073	2730769	18347099	1685168
G_p1800	100	0	5.79	100000	488	5376334	2796157	12642938	1663849
G_p2000	100	0	6.44	100000	442	5460446	3309116	10056106	1144977
G_p2200	99	0	6.90	99977	472	6392530	3538653	28031963	1757382
G_p2400	100	0	8.24	100000	468	6896360	4048325	16031910	1950925
G_p2600	100	0	8.79	100000	440	7075263	4127746	13267218	1733185
R_s	0	0	0.124	27451		3091	1946	4548	0
C	0	0	0.12	2404		473	180	1511	0

B.7.2 Vorinitialisierte Startpopulationen

Job	Er-folg	RS	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
Ri_p100_s_100%	0/10	0	17.44	91782	1024	892414	776799	1059613	0
Ri_p200_s_5%	22	0	3.05	84315	1145	1398448	255496	3881105	513239
Ri_p200_s_10%	26	0	5.1	89787	1011	1252099	287427	2552103	664748
Ri_p200_s_100%	5/10	0	36.19	97275	1154	1942939	753247	3224463	969476
Ri_p300_s_3%	36	0	3.16	91203	943	1706444	414029	3724123	777392
Ri_p300_s_5%	46	0	4.19	91122	916	1681629	106368	3401486	628726
Ri_p300_s_10%	42	0	6.74	92926	962	1777182	219001	3521235	459678
Ri_p300_s_100%	7/10	0	56.06	98996	683	2062892	1306972	3770708	355783

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Er-folg	RS	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Indivs Min	Indivs Max	Indivs GutVari
Ci_p200_5%	48	0	2.24	97285	921	1076352	247794	2021360	401837
Ci_p200_10%	50	0	2.84	97812	818	957245	219356	1946713	284070
Ci_p200_20%	24	0	4.88	96253	1056	1239212	339085	2971170	711791
Ci_p200	6/10	0	22.04	98260	779	994788	394061	1780014	193664
Ci_p300_5%	46	0	2.84	97321	950	1656995	528437	3859514	391477
Ci_p300_10%	56	0	7.72	97890	826	1448192	424536	2760924	467670
Ci_p300_20%	50	0	7.23	97783	827	1460929	330284	3022857	568220

B.7.3 Nachoptimierung

Job (Präzision=s)	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	------ Indivs Schn	Min	Max
NR_p30_P1_G1	8	50/50	0.691	71889	12529	8	964	192677	86368	357496
NR_p50_P1_G1	20	49/49	1.0	75256	10334	18	856	283066	150017	579881
NR_p100_P1_N4_G1	36	44/44	1.184	83094	8777	24	718	492577	180902	993792
NR_p100_P1_N6_G1	28	47/47	1.287	84989	11590	22	788	504290	258316	1066037
NR_p200_P1_N6_G1	58	40/40	2.713	95479	8730	38	856	1007627	375977	1686795
NR_p300_P1_N8_G1	70	34/34	4.244	95715	5248	38	873	1434021	517917	2446465
NR_p400_P1_N10_G1	82	26/26	5.508	97835	8017	34	869	1751037	702756	3037982
NR_p30_P5_G1	4	50/20	0.691	67206	17634	4	425	95584	25582	223852
NR_p50_P5_G1	8	50/31	0.78	73960	15526	8	611	207273	48299	629240
NR_p100_P5_N4_G1	32	48/46	1.056	85679	10812	28	754	475957	117238	1248544
NR_p200_P5_N6_G1	52	44/44	2.121	93020	6403	40	833	989215	437894	1779226
NR_p300_P5_N8_G1	80	34/34	2.885	98517	5922	48	825	1367591	592141	3133957
NR_p400_P5_N10_G1	82	25/25	3.481	98598	5987	32	902	1794979	798222	3243653
NR_p30_P7_G1	0	50/0	0.485	65680	27066	0	118	33383	16089	68052
NR_p50_P7_G1	4	49/7	0.663	74633	24189	2	397	115795	30641	411464
NR_p100_P7_N4_G1	40	49/40	1.177	87436	12073	38	747	471688	157624	1009148
NR_p200_P7_N6_G1	64	44/39	2.013	95025	9617	52	801	961404	303523	1577587
NR_p300_P7_N8_G1	70	34/32	2.679	97099	5468	38	957	1538531	544315	2654574
NR_p400_P7_N10_G1	84	27/26	3.232	99245	3748	38	966	1950385	957105	3967675
NR_p30_P1_G3	8	49/49	0.7	69041	14177	6	840	173603	87771	461819
NR_p50_P1_G3	14	49/49	1.01	79714	14692	12	907	301104	163848	530858
NR_p100_P1_N4_G3	42	47/47	1.43	88993	8892	36	923	574878	289147	1083323
NR_p200_P1_N6_G3	68	35/35	2.24	94042	5905	38	881	972731	339478	1811777
NR_p300_P1_N8_G3	72	33/33	4.548	98276	6196	38	887	1457781	424206	2800557
NR_p400_P1_N10_G3	92	24/24	6.036	99533	4008	40	1003	1892608	815859	3473545
NR_p30_P5_G3	2	50/11	0.528	67814	18495	2	438	94875	19552	342876
NR_p50_P5_G3	12	49/35	0.826	75996	15144	10	645	217078	36876	535681
NR_p100_P5_N4_G3	20	50/50	1.171	81392	10871	20	812	515837	251646	933007
NR_p200_P5_N6_G3	44	47/46	2.097	89767	8275	38	843	1032147	364649	2350893
NR_p300_P5_N8_G3	58	39/39	3.194	96172	5455	36	923	1557991	692568	2933745
NR_p400_P5_N10_G3	88	42/42	4.827	99057	4282	60	954	1752365	751520	3537756
NR_p30_P7_G3	0	50/1	0.46	62820	23931	0	155	39918	16109	139717
NR_p50_P7_G3	4	50/12	0.652	72510	19285	8	298	109229	28289	415460
NR_p100_P7_N4_G3	26	49/32	0.969	81525	13090	24	593	398024	93235	1010117
NR_p200_P7_N6_G3	62	39/37	1.81	94656	7815	40	858	982841	362884	1595573
NR_p300_P7_N8_G3	76	30/26	2.214	97515	6292	36	865	1362785	573728	2650887
NR_p400_P7_N10_G3	82	25/25	2.866	98576	4426	32	896	1813608	797909	3011905

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	------ Indivs Schn	Min	Max
NC1S_p30_P1_G1	0	50/50	0.524	53601	375	0	821	160689	83158	356654
NC1S_p50_P1_G1	0	50/50	0.74	61282	0	0	830	263065	119454	621323
NC1S_p100_P1_N4_G1	8	46/46	1.032	69640	0	0	779	492342	215586	840654
NC1S_p200_P1_N6_G1	30	35/35	2.109	89201	0	0	876	971822	345934	2032786
NC1S_p30_P5_G1	0	50/23	0.414	46186	68	0	523	104498	18581	262735
NC1S_p50_P5_G1	0	50/36	0.669	58736	0	0	705	227147	35737	495177
NC1S_p100_P5_N4_G1	8	46/44	0.893	79586	0	0	731	516161	272148	842234
NC1S_p200_P5_N6_G1	12	44/44	1.851	88397	0	0	823	967063	526677	1886599

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p30_P7_G1	0	50/3	0.316	39127	764	0	155	33939	8859	185181
NC1S_p50_P7_G1	0	50/10	0.493	51183	137	0	318	107777	28152	394666
NC1S_p100_P7_N4_G1	4	48/35	0.958	71608	0	0	643	411250	82084	702565
NC1S_p200_P7_N6_G1	20	40/33	1.468	85919	0	0	770	917876	329021	2121669
NC1S_p30_P1_G3	0	50/50	0.531	52667	0	0	894	175005	92791	379944
NC1S_p50_P1_G3	0	50/50	0.742	62087	0	0	854	271745	135625	622256
NC1S_p100_P1_N4_G3	4	48/48	1.024	75352	0	0	794	491488	279450	1045553
NC1S_p200_P1_N6_G3	20	40/40	2.271	88367	52	0	900	1016472	363027	1891271

B.7.4 Direkte Integration

Job	Er- folg	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GR_p5_s_best_L100	100	3.83	100000	5	69448	15032	195984	3640
GR_p10_s_best_L100	100	5.92	100000	4	105026	31577	209134	44396
GR_p20_s_best_L100	100	9.33	100000	3	165733	57472	306092	56131
GR_p30_s_best_L100	100	12.19	100000	2	218282	79309	458152	89359
GR_p10_s_all_L100	10/10	21.03	100000	2	385526	207230	540107	101661
GR_p10_s_best_L5	10/10	22.3	100000	14	380353	132108	744029	200904
GR_p10_s_all_L5	8/10	31.07	99231	12	534618	163151	1038660	295181
GR_p5_s_best_L100_Ri	100	4.48	100000	4	76773	28649	167385	31083
GR_p10_s_best_L100_Ri	100	6.97	100000	3	121773	58287	223518	36656
GR_p20_s_best_L100_Ri	100	11.08	100000	2	192496	107585	312634	51223
GR_p30_s_best_L100_Ri	100	14.03	100000	2	242898	163311	367967	51599
GR_p10_s_best_L5_Ri	10/10	26.28	100000	12	475285	117780	795495	240839

Direkte Integration mit dem Complex-Algorithmus:

Job	Er- folg	R S	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GC_p5_best_L100	0/5	0	69.9	79816	155	371028	297317	627465	0
GC_p10_best_L100	0/5	0	72.5	77751	66	329833	328404	332423	0
GC_p20_best_L100	0/5	0	73.9	75429	32	323504	314536	326796	0
GC_p10_best_L100_Ci	0/5	0	157.3	37234	768	735953	731087	739284	0

Wegen extrem langer Complex-Laufzeiten sind weitere Läufe nicht sinnvoll.

B.7.5 Verzögerte direkte Integration

Job	Er- folg	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
GvR_p5_s_P1	100	50/0	3.739	100000	35	66981	16007	148735	27168
GvR_p10_s_P1	100	50/6	5.69	100000	249	120262	33355	219452	47685
GvR_p20_s_P1	100	50/45	8.738	100000	877	257127	95914	449621	87969
GvR_p30_s_P1	10/10	10/10	13.68	100000	967	414408	256450	581239	101287

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p5_s_P5	100	50/0	4.116	100000	20	72064	24792	154677	33121
GvR_p10_s_P5	100	50/0	6.298	100000	61	113978	30666	265556	5657
GvR_p20_s_P5	100	49/4	10.945	100000	185	186279	57049	449911	85955
GvR_p30_s_P5	100	50/19	10.925	100000	468	285753	103661	620676	122873
GvR_p100_s_P5_N4	100	48/47	26.861	100000	794	1461358	203012	12546248	2312586
GvR_p5_s_P7	100	100/0	3.404	100000	13	60225	11561	161310	27289
GvR_p10_s_P7	100	50/0	5.36	100000	34	95452	26984	204686	49091
GvR_p20_s_P7	100	50/0	7.904	100000	74	154148	42183	385106	70556
GvR_p30_s_P7	100	50/0	10.584	100000	140	213934	91259	514542	104730
GvR_p5_s_6_12	100	50/0	3.705	100000	16	66235	13357	200816	36128
GvR_p5_s_P1_Ri	100	50/9	2.3	100000	66	84500	22853	173497	34924
GvR_p5_s_P5_Ri	100	50/2	2.28	100000	25	76869	25717	224077	40673
GvR_p5_s_P7_Ri	100	50/0	2.26	100000	12	75043	27723	164643	30535

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	Indivs			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GLVZC_5_P1	0/5	4/0	143.1	84515	283	628760	626004	631241	0
GLVZC_10_P1	0/5	5/0	143.2	69445	444	649453	632074	673516	0
GLVZC_20_P1	0/5	5/0	118.3	43085	1040	653649	572045	761824	0
GLVZC_10_1_3	0/5	5/0	142.9	42108	764	676936	673598	678829	0
GLVZC_10_P5	0/5	5/0	142.9	48795	821	681807	671807	696442	0
GLVZC_5_P7	0/10	10/0	72.2	76923	146	329846	327820	332365	0

Wegen extrem langer Complex-Laufzeiten sind weitere Läufe nicht sinnvoll.

B.8 Kollisionsfreie Roboterbahnplanung

Bei den Jobs von Abschnitt B.8.1 und den Hybridisierungen mit dem Complex-Algorithmus wurden 100 Läufe je Job durchgeführt, bei den Hybridisierungen mit dem Rosenbrock-Verfahren je 50. Abweichungen von diesen Werten wegen extrem langer Laufzeiten sind bei den betroffenen Jobs gesondert vermerkt.

Bei der Nachoptimierung und der verzögerten direkten Integration kommen die in Tabelle 5.2 auf Seite 95 und Tabelle 5.6 auf Seite 112 beschriebenen Sonderparametrierungen *Pa* bis *Pe* bzw. *P0* bis *P7* für die Nischenbildung ebenfalls zum Einsatz.

B.8.1 GLEAM, Rosenbrock-Verfahren und Complex-Algorithmus

Job	Er-	RS	Zeit[h]	Noten	Gen	Indivs			
	folg		Ultra1	Schn	Schn	Schn	Min	Max	GutVari
G_p30	84	0	0.601	97728	5665	1075223	161097	5213200	821158
G_p60	90	0	0.696	99150	2201	845010	63217	5189145	691425
G_p90	98	0	0.384	100000	996	607148	61202	9469735	708456
G_p120	100	0	0.16	100000	441	357877	100060	7441557	771866
G_p150	100	0	0.187	100000	291	312318	98715	1002026	157828
G_p180	100	0	0.203	100000	334	433364	89275	4500889	601360
G_p210	100	0	0.164	100000	254	385666	109478	4995245	482185
G_p240	100	0	0.205	100000	233	387511	199543	899980	131649
G_p270	100	0	0.236	100000	267	522290	146313	5754044	613103
G_p300	100	0	0.279	100000	206	446662	162629	1444783	201742
R_s	0	0	0.008	3468		10064	7	50001	0
C	0	0	0.0004	812		66	9	2438	0

B.8.2 Vorinitialisierte Startpopulationen

Job	Er-	RS	Zeit[h]	Noten	Gen	Indivs			
	folg		Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_p60_s_5%	94	0	0.238	98029	951	398057	61321	2381421	454712
Ri_p60_s_10%	92	0	0.466	98334	1576	641717	40011	4486477	686421
Ri_p60_s_20%	90	0	0.512	97431	1549	668255	71424	5156159	499255
Ri_p60_s_30%	90	0	0.561	97278	1338	589381	75185	6442708	338740
Ri_p60_s_40%	90	1	0.693	97126	1846	792247	109025	4828949	757490
Ri_p60_s_100%	88	5	1.197	97749	2153	978929	130472	5291902	928870

Job	Er-	RS	Zeit[h]	Noten	Gen	Indivs			
	folg		Ultra1	Schn	Schn	Schn	Min	Max	GutVari
Ri_p90_s_5%	100	0	0.447	100000	854	535177	80309	6450859	975603
Ri_p90_s_10%	100	1	0.237	100000	466	306620	106599	1702037	281171
Ri_p90_s_20%	94	2	0.811	98922	1076	670816	82449	5766300	882727
Ri_p90_s_30%	96	0	1.177	99229	2383	1474054	479244	7732116	1032991
Ri_p90_s_40%	90	3	1.103	97734	1795	1177301	116562	7689255	1174463
Ri_p90_s_100%	90	7	1.70	97807	1785	1255098	177896	7127746	1254632
Ri_p120_s_5%	98	0	0.299	99719	440	387303	77280	3852653	274651
Ri_p120_s_10%	96	5	0.37	99418	713	602467	109609	6571851	556673
Ri_p120_s_20%	96	5	0.581	98835	727	642716	119599	5467641	614141
Ri_p120_s_30%	96	2	0.696	99385	700	643621	125730	7854446	358593
Ri_p120_s_40%	96	2	1.109	98884	773	711623	143059	7789469	227200
Ri_p120_s_100%	98	9	2.063	99459	846	875942	265633	5549548	990653

Vorinitialisierung mit dem Complex-Algorithmus:

Job	Er-folg	RS	Zeit[h] Ultra1	Noten Schn	Gen Schn	Indivs Schn	Min	Max	GutVari
Ci_p60_5%	96	0	0.207	99721	1426	561757	48839	4419813	758526
Ci_p60_10%	97	0	0.372	99668	1829	727121	46525	5114601	1048606
Ci_p60_20%	93	0	0.366	99535	1583	623356	52279	5176836	591231
Ci_p60_30%	92	0	0.483	98766	1883	760910	53849	5178001	1012729
Ci_p60_100%	96	0	0.492	99751	1039	441023	56267	3915818	436892
Ci_p90_5%	100	0	0.302	100000	804	491201	44975	5837238	896640
Ci_p90_10%	100	0	0.247	100000	674	420983	53983	3879562	624889
Ci_p90_20%	100	0	0.269	100000	537	339674	58563	5691936	651566
Ci_p90_30%	97	0	0.370	99724	1129	663410	77797	8291135	915540
Ci_p90_100%	97	0	0.630	100000	1061	670734	10269	8249100	1193134
Ci_p120_55	100	0	0.21	100000	420	361664	107728	5784013	623852
Ci_p120_10%	99	0	0.256	99980	485	413850	66512	5869836	625061
Ci_p120_20%	100	0	0.161	100000	359	302393	114578	1488218	234027
Ci_p120_30%	99	0	0.343	99988	517	442390	78001	5549838	657423
Ci_p120_100%	98	0	0.406	99947	612	543478	103677	12856773	285626
Ci_p150_5%	100	0	0.198	100000	323	347641	129045	2477015	333264
Ci_p150_10%	100	0	0.262	100000	374	404039	123160	8470842	845994
Ci_p150_20%	100	0	0.304	100000	373	405968	115035	3900828	461546
Ci_p150_30%	96	0	0.252	99705	655	713403	109552	16072696	226077
Ci_p150_100%	99	0	0.486	99966	374	428531	145669	9067641	252536

B.8.3 Nachoptimierung

Job	Er-folg	RS	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NR_p10_s_P1_G1	2	1	56/3	0.051	49270	7590	0	317	28375	173	333263
NR_p20_s_P1_G1	24	2	40/3	0.159	80923	5527	2	934	169561	1193	1284522
NR_p30_s_P1_G1	40	1	28/2	0.306	89026	967	0	2304	451094	1729	2567039
NR_p50_s_P1_G1	82	1	7/1	0.289	99065	22	0	2186	446584	47969	2876531
NR_p10_s_P1_G3	4	1	48/3	0.086	54136	963	0	536	55039	3152	642510
NR_p20_s_P1_G3	34	0	34/5	0.14	83657	1211	4	1056	154473	8993	1265769
NR_p30_s_P1_G3	62	0	16/2	0.274	94798	84	0	2023	423437	30954	1974395
NR_p50_s_P1_G3	86	0	6/3	0.343	98149	108	0	3573	483865	38641	3230897

Die Verbesserungen durch das Rosenbrock-Verfahren sind bei guten Ergebnis-AKs marginal.

Nachoptimierung mit dem Complex-Algorithmus:

Job	Er-folg	RS	Nisch. GDV/GAk	Zeit[h] Ultra1	Noten- Schn	Verb.	Nopt Erf	Gen Schn	Indivs Schn	Min	Max
NC1S_p10_P0_G1	3	0	97/2	0.02	39873	3567	0	242	19987	95	287989
NC1S_p20_P0_G1	19	0	81/4	0.116	68803	1419	2	959	147050	376	1268025
NC1S_p30_P0_G1	54	0	44/2	0.22	89370	1763	1	1193	287063	1198	1848129
NC1S_p50_P0_G1	87	*	99/91	0.374	97386	2242	0	1947	665710	2286	3135958
NC1S_p10_P1_G1	3	0	96/6	0.113	42946	1649	0	409	42635	227	637823
NC1S_p20_P1_G1	22	0	77/4	0.101	71978	1035	0	971	153751	165	1223713
NC1S_p30_P1_G1	55	0	43/4	0.226	90025	1627	0	1331	278729	1637	1707880
NC1S_p50_P1_G1	81	*	99/84	0.33	97931	160	1	1854	625367	3097	3004313

Job	Er-	R	Nisch.	Zeit[h]	Noten-	Nopt	Gen	----- Indivs ----		
	folg	S	GDV/GAk	Ultra1	Schn	Verb.	Erf	Schn	Schn	Min	Max
NC1S_p10_P6_G1	3	0	97/3	0.046	43841	4207	0	281	25972	246	318049
NC1S_p20_P6_G1	19	0	82/6	0.094	69855	2084	1	681	98693	486	735478
NC1S_p30_P6_G1	46	0	56/3	0.133	87219	2076	2	924	204360	1035	1566381
NC1S_p50_P6_G1	83	*	99/79	0.392	98582	488	2	1515	525918	36665	3622545
NC1S_p10_Pa_N3_G1	1	0	99/9	0.071	31125	2838	0	429	31125	89	385402
NC1S_p10_Pb_N3_G1	2	0	98/7	0.052	37171	3646	0	445	32126	120	504378
NC1S_p10_Pc_N3_G1	2	0	96/4	0.073	44248	3247	0	400	40050	173	646484
NC1S_p10_Pd_N3_G1	1	0	99/4	0.14	40245	4589	0	375	32799	221	517344
NC1S_p10_Pe_N3_G1	1	0	99/7	0.045	44880	4269	0	228	21432	238	432336
NC1S_p20_Pa_N5_G1	25	0	74/9	0.2	71201	1000	0	1215	165728	313	1156671
NC1S_p20_Pb_N5_G1	25	0	76/5	0.118	70220	1476	1	696	122273	368	1075546
NC1S_p20_Pc_N5_G1	23	0	78/6	0.133	73659	1547	1	972	142526	294	1258402
NC1S_p20_Pd_N5_G1	12	0	87/5	0.169	71448	1910	0	956	142344	325	1290424
NC1S_p20_Pe_N5_G1	13	0	87/8	0.312	70234	1164	1	739	124774	337	1295490
NC1S_p10_P0_2_G3	5	2	92/4	0.046	57798	734	0	437	58667	152	641737
NC1S_p20_P0_3_G3	29	1	73/7	0.133	81397	702	3	1209	163419	1962	1292401
NC1S_p30_P0_3_G3	58	0	41/8	0.558	94420	171	0	2221	327275	18390	1988607
NC1S_p50_P0_4_G3	88	*	100/90	0.359	98649	0	0	2042	656656	142486	4229304
NC1S_p10_P1_2_G3	3	1	97/6	0.042	55692	987	0	690	52041	2017	626544
NC1S_p20_P1_3_G3	20	0	78/9	0.144	80045	403	1	1382	211117	1743	1297278
NC1S_p30_P1_3_G3	43	0	54/3	0.186	91724	244	1	1240	295247	3625	1918114
NC1S_p50_P1_4_G3	90	*	100/90	0.363	99248	65	0	2061	668935	182127	4562489
NC1S_p10_P6_2_G3	1	0	100/9	0.08	53227	1188	1	535	45439	403	645513
NC1S_p20_P6_3_G3	23	0	76/8	0.134	83987	194	1	1103	174462	1377	1293681
NC1S_p30_P6_3_G3	55	0	45/2	0.151	90669	758	1	864	216395	983	1946512
NC1S_p50_P6_4_G3	83	*	99/85	0.377	98127	419	2	1530	543853	80779	4857512

Die mit einem * bei Restarts (RS) gekennzeichneten Jobs wurden mit einer Complex-Version durchgeführt, die eine Iterationsbegrenzung erlaubt. Das Limit betrug 200000.

B.8.4 Direkte Integration

Job	Er-	RS	Zeit[h]	Noten	Gen	---------- Indivs			
	folg		Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GR_p5_s_best_L100	2/10	3	29.9	84389	436	3213431	230054	8570740	497425
GR_p10_s_best_L100	7/10	1	62.6	92055	184	3199146	193170	7571560	2409046
GR_p20_s_best_L100	1/5	3	120.9	78555	92	5314778	2565048	7222090	0
GR_p30_s_best_L100	1/5	4	140.6	81141	63	4699308	2858231	7108632	0
GR_p5_s_all_L100	3/5	2	87.8	94802	26	2380211	689200	4520523	683486
GR_p5_s_best_L5	1/3	0	102.0	95897	694	5628204	2342349	11886359	0
GR_p5_s_best_L100_Ri	1/5	0	55.9	83668	394	4053282	858933	6320942	0

Auf die Jobs *GR,p5,s,all,L5* und *GR,p5,s,best,L5,Ri* konnte wegen der Ergebnisse von *GR,p5,s,all, L100* und *GR,p5,s,best,L5* bzw. *GR,p5,s,best,L100,Ri* und *GR,p5,s,best,L5* verzichtet werden.

Direkte Integration mit dem Complex-Algorithmus:

Job	Er-	S	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	A	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GC_p5_best_L100	1/10	2	57.5	79082	556	2038495	712566	3707484	0
GC_p10_best_L100	5/10	3	39.4	90482	220	1579087	61336	2832405	548693
GC_p20_best_L100	3/5	0	97.2	91775	122	2817780	882177	5381033	2329040
GC_p30_best_L100	5/5	0	54.3	100000	286	2673560	1598713	4521199	1348719
GC_p10_best_L100_Ci	3/10	0	53.6	85869	266	1876955	950116	4951794	463883
GC_p20_best_L100_Ci	4/10	0	85.7	88698	156	2538662	331583	5167993	1165635

B.8.5 Verzögerte direkte Integration

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvR_p10_s_P1	3/10	10/0	64.22	82571	327	3560495	34421	9098756	1584946
GvR_p20_s_P1	8/10	10/1	75.37	96995	523	7436710	47203	21823412	9088983
GvR_p30_s_P1	9/10	3/1	17.92	97900	1180	1422138	43840	7263818	1491050
GvR_p10_s_P4	6/10	9/0	54.01	91173	427	5119609	1713030	11760838	1304224
GvR_p20_s_P4	8/10	8/1	45.12	98146	1426	4419718	22116	13518321	2203375
GvR_p30_s_P4	88	25/4	37.13	97591	1328	4840694	42590	22710461	6739084
GvR_p10_s_P5	6/10	10/1	56.05	94781	1241	5758570	1663916	14390631	1668181
GvR_p20_s_P5	2/5	5/1	47.68	83388	749	1704765	786244	3084909	817080
GvR_p30_s_P5	3/5	2/0	30.45	96478	1851	1542298	90658	3766262	84520
GvR_p10_s_P7	2/5	5/0	105.7	93639	263	4339542	3324222	6333734	205642

Verzögerte direkte Integration mit dem Complex-Algorithmus:

Job	Er-	Nisch.	Zeit[h]	Noten	Gen	---------- Indivs ----------			
	folg	GDV/GAk	Ultra1	Schn	Schn	Schn	Min	Max	GutVari
GvC_p10_P1	6/10	10/0	89.61	94772	842	3630682	283435	8832276	3033852
GvC_p20_P1	7/10	8/0	54.19	97317	1537	1680094	139798	5420343	929010
GvC_p30_P1	82	23/1	31.67	98299	1687	1246364	25057	10799157	1146145
GvC_p10_P4	5/10	9/1	81.2	92106	951	3538218	28285	7122362	2589848
GvC_p20_P4	3/10	10/1	55.52	91463	790	2177244	492455	4521355	294007
GvC_p30_P4	7/10	6/0	48.43	96375	1389	2146766	36953	5827498	1587373
GvC_p10_P5	4/5	5/0	40.57	99870	1048	3341984	284256	8572141	3821695
GvC_p20_P5	3/5	3/1	65.15	99571	1347	3647370	38213	10487141	371782
GvC_p30_P5	3/5	2/1	64.23	93582	1014	2037326	90929	8270280	94064