DIPLOMARBEIT

Integration anwendungsneutraler lokaler Suchverfahren in den adaptiven memetischen Algorithmus HyGLEAM für die komplexe Reihenfolgeoptimierung

von

Björn Hahnenkamp

Referent: Prof. Dr. Hartmut Schmeck
Institut für Angewandte Informatik und
Formale Beschreibungsverfahren
Universität Karlsruhe

Universität Karlsruhe (TH)
Forschungsuniversität · gegründet 1825

Koreferent: Dr. Wilfried Jakob
Institut für Angewandte Informatik
Forschungszentrum Karlsruhe

Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Karlsruhe, 31.03.2007

Abstract

Inhalt dieser Diplomarbeit sind neue lokale Suchverfahren für den memetischen Algorithmus HyGLEAM (Hybrid GeneraL purpose Evolutionary Algorithm and Method). Der Ressource Broker GORBA (Global Optimizing Resource Broker and Allocator) dient der effizienten Ressourcenallokation und Planung im Grid. Das hierbei von GORBA zu lösende komplexe Schedulingproblem wird formuliert und analysiert. Das GORBA-Problem läßt sich mit Hilfe von Heuristiken auf ein Reihenfolgeproblem reduzieren, welches mit Hilfe von HyGLEAM optimiert werden kann. Durch die Integration lokaler Suchverfahren sollte die Leistung von HyGLEAM bei der Lösung dieses Reihenfolgeproblemes verbessert werden. Hierfür wurden fünf lokale Suchverfahren entwickelt und anhand von Benchmarks bewertet.

Das Verfahren Stoss basiert auf gezielten Verschiebungen einzelner Elemente in der Reihenfolge. Drei der untersuchten lokalen Suchverfahren basieren auf einem neuen Ansatz, Ameisenalgorithmen in HyGLEAM zu integrieren. Das Verfahren ANT 1 untersucht in einer ersten Version Elemente, die in der Reihenfolge nebeneinander liegen. In einer zweiten Version werden Abstände zwischen Elementen optimiert. Das lokale Suchverfahren ANT 2 basiert auf der relative pheromone evaluation rule und untersucht absolute Positionen der Elemente in der Permutation. Die Testergebnisse dieser Verfahren haben gezeigt, dass sie zur besseren Lösung des GORBA-Problems keinen Beitrag leisten. Das fünfte lokale Suchverfahren Triv basiert auf einer wiederholten, adaptierten Ausführung verschiedener Mutationsoperatoren. Dieses Verfahren erreichte fast die Leistung der bisher genutzten Algorithmen und kann als Ausgangspunkt zukünftiger Entwicklungen dienen.

Inhaltsverzeichnis

1 Einführung **1**

1.1 Grid-Computing 1

1.2 Der Ressource Broker GORBA 2

1.3 Scheduling mit lokalen Suchverfahren 3

1.4 Der evolutionäre Algorithmus HyGLEAM 3

1.5 Lokale Suchverfahren in HyGLEAM 4

1.6 Aufbau dieser Arbeit 4

2 Das Optimierungsproblem **7**

2.1 Formulierung des Schedulingproblems 8

2.1.1 Kernkomponenten 8

2.1.2 Gültigkeitskriterien 9

2.1.3 Optimierungsziel 10

2.2 Fitnessfunktion 11

2.2.1 Qualitätskriterien 11

2.2.2 Relaxierungen und Straffunktionen 15

2.3 Vereinfachungen, Suchräume und Heuristiken 17

2.3.1 Vereinfachende Annahmen über das Schedulingproblem 18

2.3.2 Konventionelle Planungsheuristiken von GORBA 19

2.3.3 Planungsheuristiken für das Modell K2 19

2.3.4 Planungsheuristik für das Modell K1 20

2.4 Einordnung des Problems 21

2.4.1 Klassifizierung . . . 21
2.4.2 Komplexität . . . 22
2.5 Benchmarkaufgaben . . . 23
2.5.1 Testszenarien . . . 23
2.5.2 Erfolgskriterien . . . 24

3 Der evolutionäre Algorithmus HyGLEAM 27
3.1 Evolutionäre Algorithmen . . . 28
3.2 Das Konzept der Aktionskette . . . 29
3.2.1 Handlungsmodell K2 . . . 30
3.2.2 Handlungsmodell K1 . . . 30
3.3 Die genetischen Operatoren . . . 30
3.4 Das Nachbarschaftsmodell . . . 31
3.5 Lokaler Suchverfahren und Adaptionsmechanismus in HyGLEAM . . . 32
3.6 Durchführung der Benchmarks . . . 33

4 Lokale Suchverfahren für Reihenfolgeprobleme 37
4.1 No Free Lunch . . . 38
4.2 Ein neues Konzept: Optimierungsmuster . . . 39
4.2.1 Beispiele für Optimierungsmuster . . . 40
4.2.2 Modellierung Lokaler Suchverfahren . . . 42

5 Untersuchte lokale Suchverfahren 43
5.1 Lokales Suchverfahren: Stoss . . . 44
5.1.1 Algorithmus und Implementierung . . . 44
5.1.2 Ergebnisse der Benchmarks . . . 46
5.2 HyGLEAM und Ameisenalgorithmen . . . 47
5.2.1 Einführung in Ameisenalgorithmen . . . 47
5.2.2 Hybridisierungsansatz . . . 49
5.3 Die erste Version von ANT 1 . . . 50
5.3.1 Grundgerüst des Algorithmus . . . 50

5.3.2 Pheromonwerte 50
5.3.3 Heuristikwerte 50
5.3.4 Auswahlmechanismus 52
5.3.5 Pheromonupdate 53
5.3.6 Adaptionsmechanismus 53
5.3.7 Parameterstudien 53
5.3.8 Bewertung des Verfahrens 56
5.4 Lokales Suchverfahren: ANT 1, Version 2 57
5.4.1 Grundgerüst des Algorithmus 58
5.4.2 Pheromonwerte 58
5.4.3 Heuristikwerte 59
5.4.4 Auswahlmechanismus 59
5.4.5 Pheromonupdate 60
5.4.6 Adaptionsmechanismus 60
5.4.7 Parameterstudien 61
5.4.8 Ergebnisse der Benchmarks 61
5.5 Lokales Suchverfahren: ANT 2 63
5.5.1 Grundgerüst des Algorithmus 65
5.5.2 Pheromonwerte 65
5.5.3 Heuristikwerte 65
5.5.4 Auswahlmechanismus 65
5.5.5 Pheromonupdate 66
5.5.6 Parametrisierung des Verfahrens 66
5.5.7 Ergebnisse der Benchmarks 67
5.6 Lokales Suchverfahren: TRIV 68
5.6.1 Algorithmus und Implementierung 70
5.6.2 Ergebnisse der Benchmarks 71

6 Fazit und Ausblick **75**
6.1 Zusammenfassung der Arbeit 75

6.2 Ausblick 77
6.3 Dank 78

A Testergebnisse **79**
A.1 Benchmarks und verwendete Abkürzungen 79
A.2 Benchmarkergebnisse für GLEAM / K1 / gRep 80
A.2.1 Szenarien „gR gA“ 80
A.2.2 Szenarien „gR kA“ 81
A.2.3 Szenarien „kR gA“ 82
A.2.4 Szenarien „kR kA“ 83
A.3 Benchmarkergebnisse für GLEAM / K2 / pRep 84
A.3.1 Szenarien „gR gA“ 84
A.3.2 Szenarien „gR kA“ 85
A.3.3 Szenarien „kR gA“ 86
A.3.4 Szenarien „kR kA“ 87
A.4 Benchmarkergebnisse für Stoss / K2 / pRep 88
A.4.1 Szenarien „gR gA“ 88
A.4.2 Szenarien „gR kA“ 89
A.4.3 Szenarien „kR gA“ 90
A.4.4 Szenarien „kR kA“ 91
A.5 Benchmarkergebnisse für ANT 1v2 / K2 / pRep 92
A.5.1 Szenarien „gR gA“ 92
A.5.2 Szenarien „gR kA“ 94
A.5.3 Szenarien „kR gA“ 96
A.5.4 Szenarien „kR kA“ 97
A.6 Benchmarkergebnisse für ANT 2 / K2 / pRep 98
A.6.1 Szenarien „gR gA“ 98
A.6.2 Szenarien „gR kA“ 100
A.6.3 Szenarien „kR gA“ 101
A.6.4 Szenarien „kR kA“ 102

A.7 Benchmarkergebnisse für TRIV / K2 / pRep 103
A.7.1 Szenarien „gR gA“ 103
A.7.2 Szenarien „gR kA“ 104
A.7.3 Szenarien „kR gA“ 105
A.7.4 Szenarien „kR kA“ 106

Abbildungsverzeichnis

3.1 Erfolgsraten von GLEAM / K1 / gRep 35
3.2 Erfolgsraten GLEAM / K2 / pRep 35

5.1 Erfolgsraten von Stoss / K2 / pRep 46
5.2 Pseudocode der Startroutine von ANT 1, Version 1 51
5.3 Parameterstudie zu Ant 1v1: Parameter Anzahl Ameisen vs. β 55
5.4 Parameterstudie zu Ant 1v1: Parameter p_{wahl} vs. β 55
5.5 Pseudocode der Startroutine von ANT 1, Version 2 58
5.6 Erfolgsraten von Ant 1 / K2 / pRep 62
5.7 Parameterstudie zu Ant 2: Parameter α vs. p_{wahl} 66
5.8 Erfolgsraten von Ant 2 / K2 / pRep 67
5.9 Anzahl der Evaluationen von GLEAM, Ant 1v2 und ANT 2 68
5.10 Erfolgsraten von Triv / K2 / pRep 72

Kapitel 1

Einführung

1.1 Grid-Computing

Grid-Computing ist ein aktuelles Thema der Informatik und basiert auf zwei wesentlichen Phänomenen. Sinkende Preise und wachsende Leistungen von Ressourcen wie Rechen-, Leitungs-, und Speicherkapazität sowie ein meist nur zeitlich beschränkter Spitzenbedarf führen zu Überkapazitäten in Forschung, Unternehmen und im Privatsektor. Gleichzeitig werden neue Anwendungen mit teils immensen Ressourcenanforderungen entwickelt. Realistischere Simulationen oder neuartige Analysemethoden mit sehr großer Datenbasis oder komplexen Rechnungen versprechen wirtschaftliche Vorteile oder wissenschaftlichen Fortschritt.

Mit dem Begriff „Grid Computing“ werden Techniken zur Vermittlung zwischen Anwendern mit Ressourcenbedarf und Betreibern von Ressourcen mit freien Kapazitäten umfasst. Eine prägnante Zusammenfassung liefert [Fos02]. Demnach vermittelt das Grid dem Anwender über offene Schnittstellen einen Zugriff auf Ressourcen, welche den Anforderungen seiner Anwendung genügen. Diese Ressourcen können auch in anderen Organisationen angesiedelt sein.

Während an vielen Konzepten und Anwendungen schon lange geforscht wird und durchaus schon Lösungen existieren, so kann von einer flächendeckenden Verbreitung von Grid Computing noch nicht gesprochen werden. Die Ziele der Grid-Entwicklungen sind

noch wesentlich ambitionierter, als die obige Darstellung vermuten lässt: Grids haben nicht nur das Potenzial, die Effizienz von Rechenressourcen steigern. Vielmehr darf erwartet werden, dass mit dem Aufkommen einer funktionierenden Grid-Infrastruktur neue Geschäftsmodelle auf standardisierten Ressourcenmärkten und wissenschaftliche Forschungsgebiete mit besonders hohem Rechenbedarf überhaupt erst eröffnet werden.

Gute Einführungen in das Thema Grid Computing bieten [FKT01] sowie [Fos99]. Einen Standard im Bereich Grid Computing stellt die Open Grid Services Architecture (OGSA, [FKNT02]) dar, eine Beispielimplementierung ist das Globus Toolkit. Wesentlicher Motor der Entwicklung ist das Global Grid Forum.

1.2 Der Ressource Broker GORBA

Der Begriff „Ressourcenmanagement“ bezeichnet eine Komponente einer Grid Middleware, welche mehrere Aufgaben erfüllen muss. In [FKT01] werden Jobmanager und Ressourcebroker als wesentliche Teile eines Ressourcenmanagements identifiziert (siehe auch [NSW04] für eine umfassende Darstellung). Der Ressourcebroker GORBA (Global Optimizing Resource Broker and Allocator) wird am Institut für Angewandte Informatik im Forschungszentrum Karlsruhe entwickelt, siehe hierzu auch [HSS04, SJQS05, JQSS05, SJQS06a, SJQS06b].

Aus Jobbeschreibungen der Benutzer ergibt sich ein abzuarbeitender Workflow von Operationen [Fis05]. Dem gegenüber stehen detaillierte Beschreibungen von verschiedensten Ressourcen, wie zum Beispiel CPUs, Lizenzen oder Speicherplatz (vgl. [Man99]). GORBA führt eine Zuordnung von Operationen zu Ressourcen sowie die zeitliche Planung der Operationen durch. Dabei werden die Leistungsfähigkeit von Ressourcen, deren Kosten sowie Vorgaben von Anwendern berücksichtigt, um optimale Pläne zu erstellen. Bei GORBA handelt es sich um eine global optimierende Planungssoftware und nicht um ein Queueing-System [HKKS03], beim Eintreffen neuer Jobs wird eine Neuplanung aller noch nicht begonnenen Operationen vorgenommen.

GORBA hat damit ein sehr komplexes Schedulingproblem mit mehreren Nebenbedingungen zu lösen. Im Nachfolgenden wird dieses als das GORBA-Problem bezeichnet.

1.3 Scheduling mit lokalen Suchverfahren

Während die Anwendung, das Ressourcenmanagement im Grid, recht neu ist, so ist die Erforschung von Schedulingproblemen eine klassische Disziplin des Operations Research. Beispiele für Schedulingprobleme treten bei Produktionsprozessen [Pin95], im Projektmanagement [Sch05] oder bei der Erstellung von Dienst- und Stundenplänen auf. Scheduling ist ein Teilgebiet der kombinatorischen Optimierung. Einen Überblick über Schedulingprobleme und entsprechende Lösungsansätze liefert zum Beispiel [Bru04].

Zahlreiche Probleme des Scheduling gehören der Komplexitätsklasse NP an. Da der Umfang der Aufgabe und die zur Lösung verfügbare Zeit oft eine deterministische Lösung verbieten, werden im Scheduling oftmals heuristische Verfahren eingesetzt. Randomisierte, teils an natürliche Prozesse angelehnte lokale Suchverfahren haben sich als besonders erfolgreich erwiesen. Neben Verfahren auf Basis der Evolution zählen auch Ameisenalgorithmen [DS04], Schwarmoptimierung [BDT99], Simulated Annealing, Tabu Search oder Directed Local Search zu den erfolgreichen Verfahren. Überblicke bieten zum Beispiel [HS05, AL97, Par02].

1.4 Der evolutionäre Algorithmus HyGLEAM

Evolutionäre Algorithmen haben die natürliche Evolution nach Darwin [Dar59] zum Vorbild, für eine gute Einführung zu evolutionären Algorithmen siehe [ES03]. Verschiedene Lösungen eines Problems werden als Individuen einer Population gesehen. Analog zur Mutation und Rekombination der natürlichen Evolution werden Eigenschaften dieser Individuen durch genetische Operatoren manipuliert und vererbt. Durch einen Selektionsmechanismus werden vorteilhafte Eigenschaften von Individuen identifiziert und Individuen sind über mehrere Generationen zunehmends mit diesen Eigenschaften ausgestattet. Der Anpassung von Individuen an ihre Umwelt durch die natürliche Evolution entspricht bei evolutionären Algorithmen der Konvergenz der repräsentierten Lösungen hin zu optimalen Lösungen. Beispiele für entsprechende Verfahren sind Evolutionary Programming [FOW66, Mic96], genetische Algorithmen [Hol75, Dav92, Gol04] oder Evolutionsstrategien [Rec94, Sch95].

GORBA verwendet für die Optimierung den Memetischen Algorithmus HyGLEAM (Hybrid GeneraL purpose Evolutionary Algorithm and Method). HyGLEAM ist eine hybride Variante des Algorithmus GLEAM [Blu90, JGSB92, BJ02]. Die Fortentwicklung von GLEAM zu HyGLEAM wurde ebenfalls am Institut für Angewandte Informatik am Forschungszentrum Karlsruhe betrieben, siehe hierzu auch [Jak02, JBB04, Jak04a, Jak04b, Jak05, Jak06]. HyGLEAM kann nicht nur kombinatorische Optimierungsaufgaben bewältigen, sondern ist zum Beispiel auch zur multidimensionalen Parameteroptimierung in der Lage.

1.5 Lokale Suchverfahren in HyGLEAM

Evolutionäre Algorithmen lassen sich durch die Integration von lokalen Suchverfahren zu Memetischen Algorithmen erweitern, wodurch die Geschwindigkeit sowie die Lösungsgüte erhöht werden können. Zwei lokale Suchverfahren zur mehrdimensionalen Parameteroptimierung wurden bereits in HyGLEAM integriert. Ziel dieser Arbeit ist es, entsprechende Verfahren für kombinatorische, insbesondere permutationsbasierte Optimierungsprobleme zu identifizieren. Als ein solches Problem wird das GORBA-Problem als Beispiel herangezogen. Allerdings ist es ein erklärtes Ziel, HyGLEAM zu einem universell einsetzbaren Werkzeug fortzuentwickeln. Allzu spezifische beziehungsweise auf das GORBA-Problem zugeschnittene Verfahren sollen dann von der Betrachtung ausgenommen werden, wenn ihr Einsatz in anderen Problemen unmöglich ist.

1.6 Aufbau dieser Arbeit

Nach dieser Einführung in den Kontext der Arbeit wird im zweiten Kapitel zunächst das GORBA-Problem aufgestellt.

Im dritten Kapitel wird der Algorithmus HyGLEAM vorgestellt. Schwerpunkt der Betrachtung ist die Fähigkeit von HyGLEAM, kombinatorische Probleme zu lösen. HyGLEAM besitzt Mechanismen zur Einbindung zusätzlicher lokaler Suchverfahren sowie zur Adaption von Steuerungsparametern für diese Verfahren.

Das vierte Kapitel führt in die Optimierung von Reihenfolgen mit Hilfe lokaler Suchverfahren ein. Zur Charakterisierung und Analyse von Reihenfolgeproblemen wird das neue Konzept der Optimierungsmuster eingeführt. Der Erläuterung dienen Beispiele von Optimierungsmustern.

Im fünften Kapitel werden die betrachteten lokalen Suchverfahren zur Reihenfolgeoptimierung vorgestellt. Alle Verfahren werden motiviert und beschrieben. Die Ergebnisse der durchgeführten Tests werden zusammengefasst.

Das sechste Kapitel schließ mit einem Fazit und einem Ausblick für zukünftige Entwicklungen.

Kapitel 2

Das Optimierungsproblem

In diesem Kapitel wird das Schedulingproblem untersucht, welches von GORBA bzw. HyGLEAM zu lösen ist.

Im ersten Abschnitt wird eine allgemeine Version des Schedulingproblems eingeführt, welche als Basis einer Klassifikation des Schedulingproblems und damit der Vergleichbarkeit mit anderen Problemen und Lösungen dienen kann. Hierzu werden wesentliche Elemente des Problems, Gültigkeitskritierien und Beispiele für Optimierungsziele vorgestellt.

Die Implementierung eines evolutionären Lösungsverfahrens erfordert die Aufstellung einer Fitnessfunktion. Im zweiten Abschnitt werden zunächst verschiedene Qualitätskriterien für Schedules erklärt. Es werden Relaxationen von zwei Nebenbedingungen motiviert sowie jeweilige Straffunktionen abgeleitet. Über diese Qualitätskriterien und Straffunktionen wird die Fitnessfunktion definiert.

Der dritte Abschnitt befasst sich mit Vereinfachungen des relaxierten Schedulingproblems. Zum einen werden Annahmen über das Problem getroffen, welche die Anwendung bestimmter Heuristiken ermöglichen und die Implementierung vereinfachen. GORBA beziehungsweise HyGLEAM durchsuchen nicht den gesamten (unbeschränkten) Lösungsraum des Problems. Vielmehr werden verschiedene (abgeschlossene) Suchräume genutzt, von denen mit Hilfe passender Heuristiken auf den Lösungsraum abgebildet wird. Diese Suchräume sowie die passenden Heuristiken werden erläutert.

Im vierten Abschnitt wird eine grobe Klassifikation des Schedulingproblems vorgenommen. Dabei wird auch auf die Komplexität des Problems eingegangen, indem ein Beweis für dessen NP-Vollständigkeit skizziert wird.

Im letzten Abschnitt dieses Kapitels werden die zur Evaluation herangezogenen Benchmarks vorgestellt, welche bereits bei der bisherigen Untersuchung von GLEAM benutzt wurden.

2.1 Formulierung des Schedulingproblems

Zunächst soll ein allgemeines Schedulingproblem formuliert werden, auf dem das von GORBA-Problem aufbaut. Diese allgemeine Form dient dem Verständnis des Problems. Seine Formulierung ermöglicht es, Vergleiche zu anderen Schedulingproblemen zu ziehen. Deswegen wurde Wert auf eine mit der Scheduling-Literatur konsistente Notation gelegt, insbesondere wird an jene in [Bru04] angelehnt (siehe auch dort die Literaturverweise zur Notation und Klassifikation von Schedulingproblemen).

Es werden erst einige Mengen, Funktionen, Parameter und Begriffe eingeführt. Im nächsten Schritt wird der Lösungsraum über Kriterien für die Gültigkeit eines Schedules definiert. Im dritten Unterabschnitt werden zunächst typische Zielfunktionen beschrieben, der tatsächlich genutzten Funktion ist später ein eigener Abschnitt gewidmet.

2.1.1 Kernkomponenten

Gegeben seien:

- Eine Menge M von **Ressourcen** im Grid. In der Scheduling-Literatur werden hierunter m Maschinen $M_j \in \{M_1, ..., M_m\}$ verstanden, im vorliegenden Kontext können damit beispielsweise Rechner oder Computercluster gemeint sein.

- Eine Menge J von **Jobs**. Im HyGLEAM-Kontext werden diese zur Verdeutlichung mit „Application Jobs“ bezeichnet, in der Scheduling-Literatur werden diese n Jobs mit $J_i \in \{J_1, ..., J_n\}$ oder auch nur mit i angegeben.

- Eine Menge O von **Operationen**, die jeweils einer Anwendung zugeordnet werden. Im HyGLEAM-Kontext werden diese „Grid Jobs“ genannt, womit die n_i elemtaren-Operationen O_{ij} des Jobs J_i mit $O_{i1}, ..., O_{in_i}$ gemeint sind. Im Nachfolgenden wird mit O_{in_i} auch die letzte *abgearbeitete* Operation des Jobs J_i bezeichnet, auf die die Beendigung des Jobs folgt.

- Eine **Vorgängerfunktion** $p : O \times O \rightarrow \{TRUE, FALSE\}$, die eine vorgegebene Vorgänger-Nachfolger-Beziehung zwischen zwei Operationen eines Jobs anzeigt. p induziert über O einen Digraph, welcher in mindestens n disjunkte azyklische nicht-reflexive Digraphen zerfällt. Das Schedulingproblem ist offensichtlich über die Bildung der transitiven (nicht-reflexiven) Hülle dieses Digraphen abgeschlossen.

- Eine **Zuordnungsfunktion** $\mu : O \rightarrow \mathcal{P}(\mathcal{P}(M))$ von Operationen zu Ressourcenmengen. Mit μ_{ij} wird die Menge aller möglichen Kombinationen von Ressourcen aus M bezeichnet, welche gemeinsam zur Bearbeitung von O_{ij} fähig sind.

- Eine **Zeitbedarfsfunktion** $t : O \times \mathcal{P}(M) \rightarrow \mathbb{R}$ welche für jede Operation O_{ij} den Zeitbedarf für die Bearbeitung durch eine Ressourcenmenge aus μ_{ij} angibt.

- Eine **Kostenfunktion** $c : \mathbb{R} \times \mathcal{P}(M) \rightarrow \mathbb{R}$, die zu jedem Zeitpunkt z die Kosten pro Zeiteinheit einer Menge von Ressourcen angibt.

- Die Menge der **Deadlines**. Für jeden Job J_i ist ein Zeitpunkt $d_i \in \mathbb{R}$ gegeben, zu dem alle Operationen des Jobs beendet sein sollen.

- Die Menge der **Kostenlimits**. Für jeden Job J_i ist ein Betrag $c_i \in \mathbb{R}$ gegeben, den die Kosten für dessen Bearbeitung nicht überschreiten dürfen.

2.1.2 Gültigkeitskriterien

Optimiert wird über die geeignete Wahl von Startzeitpunkten $s(O_{ij}) \in \mathbb{R}$ und Ressourcenallokationen $R_{ij} \in \mu_{ij}$. Eine gültige Lösung besteht, wenn gilt:

1. Alle Operationen werden geplant und die Ressourcen werden reserviert:

$$\forall\, O_{ij} : \exists\, s(O_{ij}) \in \mathbb{R}, R_{ij} \in \mu_{ij} : \forall M_j \in R_{ij} :$$

$$M_j \textit{ ist in } [s(O_{ij}); s(O_{ij}) + t(O_{ij}, R_{ij})] \textit{ von } O_{ij} \textit{ exklusiv belegt.}$$

2. Vorgängerbeziehungen werden eingehalten:

$$\forall\ i,\ j \neq k : p(O_{ij}, O_{ik}) \Rightarrow s(O_{ik}) \geq s(O_{ij})\ +\ t(O_{ij}, R_{ij})$$

3. Fristen werden eingehalten:

$$\forall\ i : d_i \geq s(O_{in_i})\ +\ t(O_{in_i}, R_{in_i})$$

4. Kostenlimits werden eingehalten:

$$\forall\ i : c_i \geq \sum_{j=1}^{n_i} \int_{s(O_{ij})}^{s(O_{ij})\ +\ t(O_{ij},R_{ij})} c(R_{ij}, s)\ ds$$

2.1.3 Optimierungsziel

Ziel der Optimierung ist, gültige Schedules zu erzeugen, welche allen Anwendern jeweils eine möglichst rasche und kostengünstige Bearbeitung ihrer Jobs ermöglicht sowie den Betreibern von Ressourcen eine hohe Auslastung und damit hohe Einnahmen ermöglicht. Offensichtlich stehen diese verschiedenen Ziele in Konflikt.

Verfahren zur multikriteriellen Optimierung versuchen, diesen Zielkonflikten zu begegnen, indem mehrere Ziele bei der Optimierung berücksichtigt werden. Dem Nutzer wird meist eine Menge möglichst Pareto-optimaler Lösungen präsentiert, unter denen er wählen kann. Gegen den Einsatz eines solchen Verfahrens spricht die Komplexität der Darstellung entsprechender Lösungen sowie die Notwendigkeit von Nutzereingriffen. [1]

Klassische Schedulingalgorithmen konzentrieren sich dagegen zumeist nur auf ein einziges Optimierungskriterium und vernachlässigen somit untergeordnete Ziele. Meist werden Minimierungsprobleme mit folgenden Zielfunktionen aufgestellt (vgl. z.B. [Bru04]):

$$makespan = \max_{O}\{s(O_{ij}) + t(O_{ij}, R_{ij})\}$$

[1] Ich bedanke mich bei Dr. Sanaz Mostaghim für ihre Anregungen zur multikriteriellen Optimierung, welche aus den beschriebenen Gründen keinen Eingang in diese Arbeit gefunden haben.

$$total\ flow\ time = \sum_{J} s(O_{in_i}) + t(O_{in_i}, R_{in_i})$$

$$maximum\ lateness = \max_{J}\{d_i - s(O_{in_i}) + t(O_{in_i}, R_{in_i})\}$$

$$total\ costs = \sum_{O} \int_{s(O_{ij})}^{s(O_{ij})+t(O_{ij},R_{ij})} c(R_{ij}, s)ds$$

Ziel von GORBA und HyGLEAM ist aber, den Interessen von sowohl Nutzern als auch von Betreibern von Gridressourcen gerecht zu werden. Obige Kriterien sind hierfür nicht geeignet, da sie jeweils nur einen Aspekt (Zeit oder Kosten) und diesen nur aus Nutzersicht betrachten. Daher wird als Optimierungsziel eine Funktion gewählt, welche verschiedene Kriterien mit entsprechenden Gewichtungen zusammenfasst. Diese Funktion ist Inhalt des nächsten Abschnittes.

2.2 Fitnessfunktion

Als Optimierungsziel für GORBA und HyGLEAM wurde die Maximierung einer Funktion festgelegt, welche auch mit „Note“ oder „Fitness“ bezeichnet wird. Diese reellwertige Funktion hat einen Wertebereich von 0 bis 100.000, höhere Werte bedeuten bessere Schedules.

Im ersten Unterabschnitt werden vier Kriterien für die Qualität eines Schedules eingeführt, welche per Addition in die Note eingehen. Im nächsten Unterabschnitt wird eine Relaxierung von Nebenbedingungen des Schedulingproblems motiviert. Ebenso werden die resultierenden „Straffunktionen“ aufgestellt, aus welchen sich Faktoren zwischen 0 und 1 ableiten, mit denen Bewertungen ungültiger Schedules verringert werden.

2.2.1 Qualitätskriterien

Im Folgenden werden fünf primäre Kritierien für die Qualität eines Schedules angegeben. Sämtliche Kriterien haben Wertebereiche zwischen 0 und einem jeweiligem Maximum, höhere Werte bezeugen eine höhere Qualität des Schedules bezüglich des Kriteriums. Die Einhaltung des Wertebereiches wird jeweils durch eine geeignete Transformationsfunktion

gesichert. In der Regel ist diese Funktion eine umgekehrt proportionale und abschnittsweise lineare bzw. exponentielle Transformation der im Folgenden beschriebenen Ausdrücke. Durch diese Transformation wird in der Umgebung der theoretisch erreichbaren Optima genauer differenziert. Die Werte der Kriterien werden addiert, somit werden mit den jeweiligen Maximalwerten Gewichte der Kriterien festgelegt[2]. Die Kriterien lauten:

1. **Relative Kosten**: Es werden mögliche Einsparungen gegenüber den jobbezogenen Budgets bewertet. Ermittelt werden für alle J_i die Mindestkosten $JK_{min}(i)$, die sich aus dem Schedule ergebenen Kosten $JK_{ist}(i)$ sowie die Maximalkosten $JK_{max}(i)$:

$$JK_{min}(i) = \sum_{j=1}^{n_i} \min_{R_{ij} \in \mu_{ij}} \int_0^{t(O_{ij},R_{ij})} c(R_{ij}, s)ds$$

$$JK_{ist}(i) = \sum_{j=1}^{n_i} \int_{s(O_{ij})}^{s(O_{ij})+t(O_{ij},R_{ij})} c(R_{ij}, s)ds$$

$$JK_{max}(i) = c_i$$

   Der Wert des Kriteriums errechnet sich, indem zuerst der Ausdruck

$$\min \left\{ 1, \frac{1}{n} \sum_J \frac{JK_{ist}(i) - JK_{min}(i)}{JK_{max}(i) - JK_{min}(i)} \right\}$$

   berechnet wird und danach auf einen Wertebereich zwischen 0 (alle Budgets ausgereizt) und 35.000 (alle Jobs billigstmöglich bearbeitet) transformiert wird.

2. **Relative Fertigstellungszeiten**: Es werden mögliche Beschleunigungen gegenüber den jobbezogenen Zeitlimits bewertet. Ermittelt werden für alle J_i die Mindestbearbeitungszeiten $JBZ_{min}(i)$, die vom Schedule gegebene Fertigstellungszeiten $JBZ_{ist}(i)$

[2]Die Festlegung dieser Gewichte basiert bislang auf ersten Erfahrungswerten, mit Hilfe der AHP-Methode könnten diese in einem strukturierten Prozess ermittelt werden. Hierfür würden Aspekte wie Nutzersicht versus Betreibersicht, Kosten versus Zeit oder globale versus jobbezogene Orientierung zu einer Gewichtung geeigneter Kriterien führen. Dies übersteigt jedoch den Rahmen dieser Diplomarbeit. Als weiterführende Lektüre wird zum Beispiel [FS01] empfohlen.

sowie die Maximalbearbeitungszeiten $JBZ_{max}(i)$:

$$JBZ_{min}(i) = \sum_{j=1}^{n_i} \min_{R_{ij} \in \mu_{ij}} s(O_{ij}, R_{ij})$$

$$JBZ_{ist}(i) = \sum_{J} s(O_{in_i}) + t(O_{in_i}, R_{in_i}$$

$$JBZ_{max}(i) = d_i$$

Der Wert des Kriteriums errechnet sich, indem der Ausdruck

$$\min \left\{ 1; \frac{1}{n} \sum_{J} \frac{JBZ_{ist}(i) - JBZ_{min}(i)}{JBZ_{max}(i) - JBZ_{min}(i)} \right\}$$

auf einen Wertebereich zwischen 0 (alle Deadlines ausgereizt) und 24.000 (alle Jobs schnellstmöglich bearbeitet) transformiert wird.

3. **Relative Gesamtfertigstellungszeit**: Ermittelt wird eine untere Abschätzung für die Bearbeitungszeit BZ_{min}, die Bearbeitungszeit des gesamten Schedules BZ_{ist} sowie eine obere Abschätzung BZ_{max}:

$$BZ_{min} = \frac{\sum_{J_i} JBZ_{min}(i)}{\min\{n; m\}}$$

$$BZ_{ist} = \max_{J} JBZ_{ist}(i)$$

$$BZ_{max} = \max_{i} d_i$$

Für dieses Kriterium wird zunächst

$$\min \left\{ 1; \frac{BZ_{ist} - BZ_{min}}{BZ_{max} - BZ_{min}} \right\}$$

berechnet und dann auf einen Wertebereich zwischen 0 (Jobs sind erst zur letzten Deadline fertig) und 14.000 transformiert. Der Maximalwert wird nur dann erreicht, wenn stets die schnellste Ressource verwendet wird und dadurch entweder stets alle Ressourcen ausgelastet sind (mehr Jobs als Ressourcen) oder alle Jobs gleichzeitig ablaufen (mehr Ressourcen als Jobs).

4. **Auslastung**: Es ist anzunehmen, dass schon vor Abarbeitung des aktuellen Schedules neue Jobs hinzukommen und eine Neuplanung angestoßen werden muß. Um ein Maß für die Auslastung der Ressourcen zu definieren werden nur die ersten 75% der Bearbeitungszeit des Schedules und die durchschnittliche Auslastung der Ressourcen betrachtet. Berechnet wird zunächst

$$\int_0^{0.75*BZ_{ist}} \frac{\sum_M 1_{\{M_j \ ist \ zum \ Zeitpunkt \ s \ reserviert\}}}{m} \, ds$$

   Dieser Wert wird dann auf einen Bereich zwischen 0 (es wird stets nur eine Ressource beansprucht) und 7.000 (alle Ressourcen sind durchgängig ausgelastet) transformiert.

5. **Relative Jobzeiten**: Während der Erstellung dieser Arbeit wurde von Dr. Jakob ein zusätzliches Kriterium eingeführt. Dabei wird der Beitrag der Bearbeitungszeit einzelner Grid Jobs zur Fertigstellungszeit des entsprechenden Application Jobs gemessen. Zunächst wird für alle Grid Jobs deren geplante Fertigstellungszeit $GJFZ(O_{ij})$, der Minimale Zeitbedarf des Grid Jobs $GJBZ_{min}(O_{ij})$ ermittelt. Für jeden Application Job wird die früheste Startzeit $AJSZ_{min}(i)$ ermittelt (welche in den meisten Fällen 0 beträgt, da Application Jobs normalerweise sofort begonnen werden können. Außerdem wird $JBZ_{ist}(i)$ wie zuvor berechnet.

$$GJFZ(O_{ij}) = s(O_{ij}) + t(O_{ij}, R_{ij})$$

$$GJBZ_{min}(O_{ij}) = \min_{R_{ij} \in \mu_{ij}} s(O_{ij}, R_{ij})$$

   Danach wird der folgende Ausdruck berechnet:

$$\sum_J \frac{1}{m} * \frac{\sum_{O_{ij} \neq O_{in_i}} GJFZ(O_{ij}) - GJBZ_{min}(O_{ij}) - AJSZ_{min}(i)}{\left(\sum_{O_{ij}} 1_{\{O_{ij} \neq O_{in_i}\}} \right) (JBZ_{ist}(i) - AJSZ_{min}(i))}$$

   Dieser Wert wird auf einen Skala von 0 bis 20.000 transformiert. Mit dieser Funktion können Auswirkungen durch die Manipulation einzelner Grid Jobs erfasst werden.

2.2.2 Relaxierungen und Straffunktionen

Die restriktiven Nebenbedingungen des Anfangs vorgestellten Schedulingproblems führen dazu, dass GORBA und HyGLEAM durchaus mit unlösbaren Problemen konfrontiert werden. Der Nutzer, der einen neuen Job einstellt, kann kein Wissen über alle Ressourcen oder über die zukünftige Last der Systeme haben. GORBA führt zunächst nur eine triviale Prüfung der Kosten- und Zeitlimits durch, welche nicht die Last des Grids durch andere Jobs berücksichtigt. Hierdurch ist es möglich, dass nicht alle Budgets und/oder nicht alle Fälligkeiten eingehalten werden können, auch wenn die Eingaben der Nutzer zunächst akzeptiert wurden.

Ein Mittel, um mit Nebenbedingungen umzugehen, ist deren Relaxierung. Relaxierungen bilden beispielsweise die Grundlage der Lagrange-Multiplikation als auch die vieler Verfahren zur gemischt-ganzzahligen Optimierung. Zur Lösung kombinatorischer Probleme mit Hilfe evolutionärer und anderer Black-Box-Verfahren können mit Hilfe von Relaxierungen Nebenbedingungen auf eine einzige Fitnessfunktion abgebildet werden. Somit wird ein Feedback von verletzten Nebenbedingungen an den Optimierungsalgorithmus möglich und Nebenbedingungen müssen außerhalb der „Black Box“ nicht berücksichtigt werden.

Im vorliegenden Fall wurden die Gültigkeitskriterien 3 und 4 (siehe Abschnitt 2.1.2) relaxiert. Die ersten beiden Nebenbedingungen sichern die Lauffähigkeit der Jobs und sind unbedingt nötig. Eine Verletzung der Nebenbedingungen 3 oder 4 hat jedoch keinen Einfluss auf die Lauffähigkeit der Jobs, und eine Überschreitung von Kosten- und Zeitlimits ist gegebenenfalls tolerierbar.

Bei der unplanmäßigen Verletzung der Zeitbedingung im Verlauf der Abarbeitung wäre es möglich, den Nutzer nach Bearbeitung vor die Wahl zu stellen, ob er die Ergebnisse seines Jobs nach wie vor kaufen möchte. Hiermit geht der Betreiber des Grids das Risiko ein, bearbeitete Jobs nicht bezahlt zu bekommen. Bei Überschreitung des Budgets kann der Betreiber einen Verlust machen und dem Nutzer nur das ursprüngliche Budget in Rechnung stellen. Ebenso wäre eine Umlegung der Gesamtkosten den jeweiligen Budgets (und nicht den tatsächlichen Kosten) entsprechend möglich. Außerdem kann er dem Nutzer die Wahl lassen, die Ergebnisse zum höheren Preis zu kaufen. Wird dem Nutzer eine

Wahl gelassen, so hat der Betreiber hierfür die Ergebnisse der Jobs zurückzuhalten und gegebenenfalls zu vernichten.

Für den kommerziellen Einsatz empfiehlt sich auch eine spieltheoretische Betrachtung des folgenden Phänomens: Für den Nutzer ist es bislang stets eine dominante Strategie, das Zeitlimit möglichst bald sowie das Budget möglichst billig anzugeben, um seine Jobs bevorzugen zu lassen. Mögliche Reaktionen des Betreibers hierauf sind zu untersuchen, so sollte der Betreiber geeignete Heuristiken verwenden, um zu niedrige Limits nicht zu akzeptieren. [3]

Im folgenden werden vier der sechs Straffunktionen angegeben, welche das Ausmaß der Verletzung von Nebenbedingungen anzeigen. Die Straffunktionen haben in der Regel Wertebereiche zwischen 0 und 1, wobei ein Funktionswert von 1 die vollständige Erfüllung der entsprechenden Nebenbedingung anzeigt. Diese Eigenschaft wird wieder über eine geeignete Transformation sichergestellt, welche abschnittsweise exponentiell oder linear ist. Sämtliche Straffunktionen liefern Faktoren, mit denen die Summe der Werte der Qualitätskriterien (siehe Abschnitt 2.2.1) multipliziert wird. Somit bleibt der Wertebereich der Fitnessfunktion von 0 bis 100.000 erhalten, während ungültige Schedules durch besonders schlechte Noten abgestraft werden. Die Straffunktionen lauten im Einzelnen:

1. Die Funktion **Verzugsjobs** ist eine Transformation der Funktion

$$\frac{\sum_{J} 1_{\{JBZ_{ist}(i) > JBZ_{max}(i)\}}}{n}$$

   Damit geht die Anzahl der zu langsam bearbeiteten Application Jobs in die Bewertung ein.

[3] Bei genauerer Betrachtung des ökonomischen Modells von GORBA entfaltet sich eine Vielfalt von Phänomenen aus der Informationswirtschaft. Unter anderem werden vom Nutzer Entscheidungen unter Unsicherheit verlangt (zukünftige Auslastung) und es herrschen Informationsasymmetrien (Jobs anderer Nutzer, tatsächliche Limits). Für den Nutzer ist Ehrlichkeit momentan keine dominante Strategie bei der Angabe der Limits. Als eine interessante weiterführende Lektüre zu einigen dieser Fragen wird zum Beispiel [MR92] empfohlen. GORBA kann auch als Agent von Nutzer und Betreiber gesehen werden, welcher vertragsrechtlich bindende Verträge eingeht.

2. Die Die Funktion **Verzugszeit** ist eine Transformation der Funktion

$$\frac{\sum_{J} 1_{\{JBZ_{ist}(i)>JBZ_{max}(i)\}} * \frac{JBZ_{ist}(i)}{JBZ_{max}(i)}}{\sum_{i|JBZ_{ist}(i)>JBZ_{max}(i)} 1}$$

   Mit dieser Funktion wird das Ausmaß der Verspätungen jener Application Jobs bewertet, bei denen die Deadline nicht eingehalten wurde.

3. Die Funktion **Überschreitungsjobs** ist eine Transformation der Funktion

$$\frac{\sum_{J} 1_{\{JK_{ist}(i)>JK_{max}(i)\}}}{n}$$

   Hiermit wird die Anzahl der Application Jobs einbezogen, bei denen das Budget überschritten wurde.

4. Die Funktion **Überschreitungskosten** ist eine Transformation der Funktion

$$\frac{\sum_{J} 1_{\{JK_{ist}(i)>JK_{max}(i)\}} * \frac{JK_{ist}(i)}{JK_{max}(i)}}{\sum_{J} 1_{\{JK_{ist}(i)>JK_{max}(i)\}}}$$

   Dadurch werden die zu viel angefallenen Kosten berücksichtigt.

Die Funktionen **Relative Anzahl zu früh geplanter Jobs** sowie **Relative Anzahl ungeplanter Jobs** sind im Rahmen dieser Arbeit irrelevant.

2.3 Vereinfachungen, Suchräume und Heuristiken

Der Einsatz der von GORBA und HyGLEAM genutzten Heuristiken ist erst nach bestimmten Annahmen möglich. Im ersten Unterabschnitt werden diese Annahmen in Kürze vorgestellt. GORBA und HyGLEAM nutzt verschiedene Suchräume und eine jeweilige Menge von Heuristiken, um Schedules zu erzeugen. HyGLEAM kann mit zwei alternativen Suchräumen arbeiten, welche im HyGLEAM-Kontext mit „Handlungsmodellen“ oder Genmodellen bezeichnet werden. Bislang werden beide Konzepte weiter entwickelt und untersucht. In den letzten beiden Unterabschnitten werden diese beiden Suchräume definiert und analysiert. Dazu werden auch die jeweiligen Heuristiken vorgestellt, mit denen

anhand der von HyGLEAM gefundenen Individuen bewertbare Schedules generiert werden.

2.3.1 Vereinfachende Annahmen über das Schedulingproblem

Ressourcen können mit Hilfe von Abhängigkeitsbeziehungen miteinander verknüpft werden. So kann unter anderem abgebildet werden, dass ein System in verschiedenen Konfigurationen betrieben werden kann. Eine Ressourcenbeschreibung könnte zum Beispiel lauten „Linux-Ressource 1 benötigt Rechner A“ und „Windows-Ressource 2 benötigt Rechner A“. Außerdem können „virtuelle“ Ressourcen definiert werden mit dem Zweck, Ressourcengruppen geeignet abzubilden. Eine Ressourcenbeschreibung könnte in diesem Fall „Cluster A benötigt Rechner 1 bis 8“ lauten. Eine Vereinfachung besteht darin, dass die Zuordnungsfunktion μ nunmehr nur noch alternative Ressourcen, nicht Ressourcenmengen betrachtet. Dadurch vereinfachen sich auch Kosten- und Zeitbedarfsfunktion.

Die Kosten einer Ressourcenmenge entsprechen der Summe der Kosten ihrer Elemente, die Kosten können über die Zeit als weitgehend konstant angenommen werden:

$$\int_{s(O_{ij})}^{s(O_{ij})+t(O_{ij},R_{ij})} c(R_{ij}, z)\ dz = \int_{0}^{t(O_{ij},R_{ij})} \sum_{M_j \in R_{ij}} c(M_j, const)$$

Dies ermöglicht einfache Heuristiken zur Ressourcenallokation: Die Kosten der Alternativen aus der Zuordnungsfunktion werden einmalig ermittelt und dienen dann den Allokationsheuristiken für die gesamte Planungsphase als Grundlage. Die Note eines Individuums wird jedoch nach wie vor auf Basis der tatsächlichen, über die Zeit gegebenenfalls variablen Kostenfunktion berechnet, wodurch solche Preisschwankungen dennoch berücksichtigt werden.

Der Zeitbedarf für einen Job wird über eine Einheit „Leistungseinheiten“ ermittelt. Jeder Job hat einen definierten Bedarf $t_1(O_{ij})$ an Ressourcenleistung in Leistungseinheiten, Leistungen von Ressourcen werden in „Leistungseinheiten pro Zeit“ $t_2(M_j)$ angegeben. Dadurch vereinfacht sich die Zeitbedarfsfunktion:

$$t(O_{ij}, R_{ij}) = \frac{t_1(O_{ij})}{\sum_{M_j \in R_{ij}} t_2(M_j)}$$

Bisher wurde die Zeit als ein stetiger Parameter dargestellt. Tatsächlich wird in Zeitschritten (zum Beispiel Sekunden) gerechnet. Eine aufwändige numerische Berechnung der Integrale wird somit über deren Umformulierung zu Summen über diskrete Zeitschritte überflüssig.

2.3.2 Konventionelle Planungsheuristiken von GORBA

GORBA verfügt über zwei verschiedene grundsätzliche Heuristiken, um Initiallösungen für die Optimierung zu generieren. Dabei werden jeweils durch einfache Prioritätsregeln Elemente in die Lösung aufgenommen und jeweils einer der drei im Folgenden beschriebenen Allokationsheuristiken zugeführt. Dadurch entstehen sechs initiale Lösungen, welche zum Beispiel als Individuen der Startpopulation für die evolutionäre Optimierung mit Hilfe von HyGLEAM dienen. Diese Initiallösungen sind besser als zufällig generierte Lösungen, jedoch erfüllen sie praktisch nie die Nebenbedingungen des GORBA-Problemes.

2.3.3 Planungsheuristiken für das Modell K2

Die Elemente dieses Suchraumes bestehen aus einer Permutation aller Operationen sowie der Angabe einer Allokationsheuristik. Sowohl die Permutation als auch die Auswahl der Heuristik wird von GLEAM oder HyGLEAM im Rahmen der Optimierung evolutionär optimiert. Die Allokationsheuristiken erstellen einen Belegungsplan, indem den Operationen in der gegebenen Reihenfolge durch eine der folgenden Heuristiken jeweils Ressourcen und eine Startzeit zugeordnet werden:

Global Fast: Die billigste aller schnellstmöglichen Ressourcen wird zum nächstmöglichen Zeitpunkt reserviert.

Global Cheap: Die schnellste aller billigstmöglichen Ressourcen wird zum nächstmöglichen Zeitpunkt reserviert.

Applikationsspezifisch: Für jede Operation wird ein Parameter des zugehörigen Jobs ausgewertet, der angibt, welche der beiden obigen Ressourcen verwendet werden soll.

Jeder Job J_i besitzt n_i Operationen und es existieren 3 Allokationsheuristiken. Es werde vereinfachend angenommen, dass auch Elemente dieses Suchraumes die Vorgängerbeziehung berücksichtigen müssen (für alle anderen Permutationen wird diese Eigenschaft mit Hilfe von Reparaturmechanismen gesichert). Als Abschätzung von p werde angenommen, dass jede Operation im Durchschnitt einen Vorgänger und einen Nachfolger habe.

Die Größe des Suchraumes lässt sich (wenn auch nur sehr grob) mit

$$\frac{\left(\sum_{i=0}^{n} n_i\right)! * 3}{2^{\left(\sum_{i=0}^{n} n_i\right)}} = 3 * \prod_{g=1}^{|O|} \frac{g}{2}$$

disjunkten Elementen abschätzen. Für ein Szenario mit 50 Operationen beträgt dieser Ausdruck etwa 10^{50}. Für die momentane Implementierung gilt, dass eine Optimierung durch vollständige Enumeration bereits bei 16 Operationen länger als einen Tag dauern würde.

2.3.4 Planungsheuristik für das Modell K1

Elemente des Suchraumes bestehen aus einer Permutation aller Operationen sowie einer Zuordnung jeder Operation zu einer für sie möglichen Ressource. Die (triviale) Planungsheuristik besteht darin, die Operationen zum jeweils frühstmöglichem Zeitpunkt (d.h. wenn die die Vorgänger abgearbeitet sind und die Ressource für die Dauer der Operation frei ist) zu belegen. Der Suchraum ist offensichtlich deutlich größer als der Suchraumes bei heuristischer Ressourcenauswahl. Eine Lösung durch Enumeration verbietet sich auch hier.

Das Modell K1 bietet den Vorteil einer differenzierten Zuordnung von Operationen zu Ressourcen. Die Allokationsheuristiken für K2 können teils nicht die guten Lösungen erstellen, die mit K1 erreicht werden. Allerdings sind die Allokationsheuristiken sehr schnell, während der größere Suchraum von K1 mehr Suchaufwand erfordert. Bei zeitgesteuerten Experimenten werden die besseren Ergebnisse bei Verwendung von K2 erreicht. Im Rahmen dieser Arbeit wurde daher ein Schwerpunkt auf dieses Modell gelegt. Bei Experimenten ohne Zeitlimit werden durch die Verwendung des Modells K1 die besseren Ergebnisse erreicht.

2.4 Einordnung des Problems

2.4.1 Klassifizierung

Das Gorba-Problem erweist sich als sehr komplex. Es werden viele Komponenten berücksichtigt, es existieren mehrere Nebenbedingungen und die Zielfunktion ist eine Komposition unterschiedlichster Funktionen.

Als Ansatz einer Klassifizierung werden an dieser Stelle einige Charakteristiken gemäß [Bru04] angegeben. Mit **o** sei dabei gekennzeichnet, dass die entsprechende Eigenschaft für das GORBA-Problem nicht angegeben werden kann.

$\alpha_1 =$ **GMPM:** Es handelt sich um so genannte multi-purpose Maschinen, die Zuordnungsfunktion μ gibt an, welcher Job auf welcher Maschine bearbeitet werden kann. Auch wenn GMPM-Probleme normalerweise nicht über ein Operationsmodell definiert werden, wie dies hier getan wird, so kann jedoch nicht von einem Shop-Scheduling Problem gesprochen werden, da die bekannten Shop-Scheduling Probleme stets weitere Einschränkungen an die Vorgängerbeziehung und die Zuordnungsfunktion umfassen.

$\alpha_2 =$ **o:** Die Anzahl der Ressourcen m ist beliebig, insbesondere kann $m >> 2$ angenommen werden.

$\beta_1 =$ **o :** Operationsunterbrechungen (preemption) sind nicht erlaubt.

$\beta_2 =$ **prec:** Es existieren Vorgängerbeziehungen, angegeben durch die Funktion p. Bis auf Transitivität und Nichtreflexivität werden über diese Relation keine Aussagen gemacht.

$\beta_3 =$ **o:** Jobs haben keine früheste Startzeit.

$\beta_4 =$ **individuelle** $t(O_{ij}, R_{ij})$**:** Die Laufzeiten werden durch t angegeben und sind job- und ressourcenspezifisch.

$\beta_5 =$ **individuelle** d_i**:** Es existieren Zeitlimits d_i.

β_6 = **o:** Es handelt sich nicht um ein Batch-Problem, Reihenfolgen müssen lediglich die Vorgängerbeziehungen respektieren. Rüstzeiten spielen keine Rolle.

γ = **Fitness:** Die komponierte Optimierungsfunktion ist sehr komplex.

Das GORBA-Problem ist also mit

$$GMPM|prec, t(i,\mu), c(i,\mu), d(j), c_j|Fitness$$

zu notieren. Ein vergleichbares Problem konnte in der Literatur nicht identifiziert werden (siehe [Bru04] sowie [Bru06] für umfangreiche Darstellungen bekannter Schedulingprobleme).

2.4.2 Komplexität

Offensichtlich ist das GORBA-Problem NP-vollständig. An dieser Stelle sei nur eine Beweisskizze geliefert. Gegeben sei:

1. Es gilt $2 < m < n$, es gibt weniger Maschinen als Jobs.
2. d_i entspricht der Summe aller Jobbearbeitungszeiten, Deadlines werden also vernachlässigt.
3. Es wird nur noch das Kriterium der relativen Gesamtfertigstellungszeit betrachtet, alle anderen Kriteriengewichte seien 0.
4. c_i ist sehr groß und c liefere immer 1. Kosten werden also vernachlässigt.
5. Alle μ seien einelementig, jede Operation ist also genau einer Maschine zugeordnet.

Durch die Restriktionen 1. und 2. entspricht das Kriterium der relativen Gesamtfertigstellungszeit dem Makespan. Wegen 3. ist dies das einzige Optimierungskriterium. Durch 2. und 4. sind alle Lösungen straffrei. Durch die Restriktion 5. entspricht das so beschriebene Problem dem Job Shop Schedulingproblem.

Das Job Shop Scheduling Problem ist demnach ein Spezialfall des GORBA-Problemes ist. Anders ausgedrückt kann mit Hilfe dieser Parameterwahlen das Job Shop Scheduling Problem einfach auf das GORBA-Problem reduziert werden. Das Job Shop Scheduling Problem ist bis auf wenige Spezialfälle NP-vollständig, für eine Übersicht über NP-vollständige Probleme sowie entsprechende Beweise sei erneut auf [Bru04] verwiesen. Daraus folgt, dass das GORBA-Problem ebenfalls NP-vollständig ist (siehe zum Beispiel [GD79]).

2.5 Benchmarkaufgaben

Eine Beschreibung der benutzten Benchmarks findet sich in [SJQS06b]. Zur Bewertung eines Lösungsverfahrens wurden synthetische Benchmarkszenarien entwickelt, im Rahmen dieser Arbeit wurden 16 Szenarien untersucht. Jedes Szenario umfasst eine Beschreibung der Ressourcen, den Listen von Application Jobs und den zugehörigen Grid Jobs sowie allen weiteren in Abschnitt 2.1.1 eingeführten Elementen.

2.5.1 Testszenarien

Die Szenarien unterscheiden sich durch folgende Parameter:

- Die Grad der Ressourcenauswahlmöglichkeiten oder Parallelität misst die Anzahl der Elemente in den Mengen μ_{ij}. Dieser Parameter misst also nicht die Anzahl der Ressourcen, sondern die Anzahl der Alternativen, die für einen Grid Job gewählt werden kann. Ist dieser Parameter hoch, so hat ein Lösungsverfahren mehr Freiheiten bei der Ressourcenallokation. Dieser Parameter kann in den Benchmarks einen hohen oder einen niedrigen Wert annehmen.

- Der Abhängigkeitsgrad bezieht sich auf die von p geforderten Vorgängerbeziehungen. Ein hoher Abhängigkeitsgrad bedeutet weniger Freiheiten bei der zeitlichen Planung der Grid Jobs. Auch hier sind die Szenarien so konstruiert, dass dieser Abhängigkeitsgrad zwei Werte, einen großen oder geringen, annehmen kann.

- Die Anzahl der Grid Jobs beträgt 50, 100 oder 200. Zusätzlich existiert ein Szenario mit dem Kürzel „200d“. Hierbei wurde von jeder Ressource des Szenarios mit 200 Grid Jobs ein zweites Exemplar konfiguriert, es steht also die doppelte Anzahl an Ressourcen zur Verfügung.

Zur Berechnung des Abhängigkeitsgrades und der Ressourcenauswahlmöglichkeiten sei auf [SJQS06b] verwiesen. Im Folgenden werden die Benchmarkszenarien durch Abkürzungen wie „kRgA 200d“ bezeichnet, welches in diesem Fall das Szenario mit kleinem Ressourcenauswahlgrad, großem Abhängigkeitsgrad, 200 Grid Jobs sowie vielen Ressourcen darstellt. Die Abkürzungen der anderen 15 Szenarien ergeben sich entsprechend.

2.5.2 Erfolgskriterien

Da die untersuchten Optimierungsverfahren auf Zufallsprozessen beruhen, werden zur Bewertung eines Verfahrens stets mehrere Läufe durchgeführt. In den durchgeführten Experimenten wurden, sofern nicht anders angegeben, jeweils 50 Läufe durchgeführt. Dem Optimierungsverfahren stehen pro Lauf 3 Minuten Rechenzeit zur Verfügung. Jeder Test eines der 16 Benchmarkszenarios benötigt demnach 2:30 Stunden. Die Ergebnisse werden anschließend mit statistischen Methoden ausgewertet.

Als Referenzwerte wird zunächst für jedes Szenario die höchste Note einer Initiallösung der konventionellen Planungsheuristik berechnet. Dabei werden die Straffunktionen bei der Berechnung ignoriert. Das Überschreiten dieser „Rohnote“ durch einen Optimierungsalgorithmus wird als Erfolg gewertet, wobei für den Optimierungsalgorithmus die Straffunktionen aktiviert werden. Als Erfolgsraten werden dabei die Wahrscheinlichkeiten bezeichnet, die vorgegebene Rohnote zu überschreiten beziehungsweise eine straffreie Lösung zu finden.

Ein zweites Kriterium ist die Note des besten gefundenen Individuums. Nach den Läufen wurde jeweils die minimale, mittlere und maximale Note berechnet. Ebenso wurde die Breite des 95% Konfidenzintervalls berechnet.

Im Falle von HyGLEAM sind Konvergenzkriterien definiert. Der Algorithmus bricht ab, wenn über eine definierbare Anzahl Generationen weder global noch einer begrenzten

Nachbarschaft ein neues Optimum gefunden wird. Die mittleren Zeiten bis zum Erreichen der Konvergenz werden zwar ermittelt, jedoch nicht weiter beachtet. Weitere Konvergenzkriterien wie das Erreichen eines Notenniveaus können ebenfalls angegeben werden, auch hiervon wurde abgesehen.

Bei der Verwendung des Modells K2 wird auch die Verteilung der Allokationsheuristiken, mit denen das jeweils beste Ergebnis erzielt wurde, angegeben.

Weiterhin werden verfahrensspezifische Daten erhoben, siehe hierzu den Abschnit 3.6.

Kapitel 3

Der evolutionäre Algorithmus HyGLEAM

In diesem Kapitel wird der evolutionäre Algorithmus GLEAM sowie die aktuelle und hybridisierte Variante HyGLEAM beschrieben. Das Ziel bei der Entwicklung von GLEAM war die Schaffung eines universell einsetzbaren praxistauglichen Optimierungswerkzeuges. Mit der Erweiterung zu HyGLEAM wurden neue Möglichkeiten geschaffen, den Algorithmus an besondere Probleme anzupassen. Seine Leistungsfähigkeit haben beide Algorithmen schon oft unter Beweis gestellt, in [JBB04] werden unter anderem Anwendungen zur Robotersteuerung, Designoptimierung oder Dienstplanerstellung aufgeführt.

Im ersten Abschnitt dieses Kapitels wird zunächst in knappen Zügen die Entwicklung evolutionärer Algorithmen dargestellt, für eine detailliertere Einführung wird auf entsprechende Literatur verwiesen.

Im zweiten Abschnitt werden einige besondere Konzepte und Mechanismen von GLEAM und HyGLEAM beschrieben, welche diese von anderen evolutionären Algorithmen abheben. Aktionsketten als Repräsentationskonzept bieten vielfältige Möglichkeiten zur Modellierung unterschiedlichster Problemstellungen. Die genetischen Mutations- und Rekombinationsoperatoren werden vorgestellt. Die Selektion beruht auf einem ausgefeilten Populationsmodell, welches zum Beispiel Nischenbildung ermöglicht. Zur Einbindung zusätzlicher lokaler Suchverfahren wurde ein Adaptionsmechanismus geschaffen, mit dem

lokale Suchverfahren gezielt angesteuert werden können.

Zur Nutzung von HyGLEAM bei der Lösung des GORBA-Problemes sind einige Konfigurationen und Anpassungen nötig. Im dritten Abschnitt werden die zwei Genmodelle für die Lösung des GORBA-Problemes beschrieben. Außerdem wird beschrieben, wie die Benchmarks aus Abschnitt 2.5 durchzuführen sind.

3.1 Evolutionäre Algorithmen

Evolutionäre Algorithmen beruhen auf der Evolutionstheorie nach Darwin [Dar59]. Der zugrunde gelegte natürliche Evolutionsprozess basiert auf zufälligen Mutationen des Erbgutes (Genotyp) von Individuen, der Rekombination durch sexuelle Reproduktion und Selektionsmechanismen. Die zufälligen Mutationen des Genotyps bewirken eine Veränderung des Erscheinungsbildes (Phänotyp) der Lebewesen, welche sich anhand dieses Genotyps entwickeln. Lebewesen, die aufgrund ihres Phänotyps besser an ihre Umgebung angepasst sind, erhalten durch die natürliche Selektion eine höhere Chance zur Fortpflanzung. Nachteilige Mutationen des Genotyps werden dadurch mit geringerer Wahrscheinlichkeit an Nachkommen vererbt als vorteilbehaftete Änderungen des Erbgutes.

Die Evolution nach Darwin ist also ein Anpassungsprozess, der die Überlebensfähigkeit (Fitness) der Individuen einer Population in einer jeweiligen Umgebung optimiert. Evolutionäre Algorithmen haben dieses Optimierungsverfahren zum Vorbild. Unter dem Oberbegriff des evolutionären Algorithmus können mehrere teils unabhängige Entwickungen vereinigt werden, so zum Beispiel:

- Genetische Algorithmen [Hol75]
- Evolutionäres Programmieren [FOW66]
- Evolutionsstrategien [Rec94]

Evolutionäre Algorithmen sind Inhalt zahlreicher Lehrbücher. Einen guten und aktuellen Überblick über das Feld geben zum Beispiel [ES03, WH04, Poh00], einen theoretischeren Ansatz verfolgt [DeJ06]. Alle aufgeführten Strömungen werden intensiv weiter

erforscht, und neben den oben aufgeführten und grundlegenden Werken gibt es viele weiterführende Veröffentlichungen.

Genetische Algorithmen erfreuen sich einer hohen Popularität, da sie vergleichsweise einfach zu erklären sind und bereits vielfach theoretisch untersucht wurden. Das Genmodell beruht auf Bitfolgen, die Mutations- und Rekombinationsoperatoren sind teilwese sehr einfach gehalten. Ein beispielorientiertes Lehrbuch ist [Dav92], einen Überblick über Anwendungen gibt [Gol04]. Evolutionary Programming wird in [Mic96] gut erklärt. Das wesentliche Merkmal dieses Verfahrens sind komplexere Datenstrukturen zur Repräsentation von Individuen. In das Thema der Evolutionsstrategien wird auch in [Kos03, Bey01, Sch95] eingeführt.

3.2 Das Konzept der Aktionskette

Die Repräsentation von Individuen geschieht in HyGLEAM mit Hilfe von so genannten Aktionsketten (siehe hierzu beispielsweise [BJ02]). Der Aufbau dieser segmentierbaren Abfolge von Aktionen wird im Genmodell beschrieben. Im GLEAM-Kontext wird dies auch als Handlungsmodell bezeichnet, es wird in Dateien mit der Endung *.mod* konfiguriert. Aktionsketten können variable oder feste Längen haben. Es können Typen von Aktionen unterschieden werden, an einzelne Aktionen können reellwertige oder ganzzahlige Parameter geknüpft werden. Über die Auswahl zulässiger genetischer Operatoren können weitere Eigenschaften zugesichert werden, beispielsweise kann eine Löschung oder Kopie eines Elementes verhindert werden.

Die Interpretation von Chromosomen als parametrisierbare Aktionen rührt von einem in [Blu90] eingeführten Verständnis von GLEAM. Demnach wird mit GLEAM ein Lebewesen modelliert, welches Sensoren und Aktoren besitzt und Verhaltensmuster kennt und lernen kann. Verschiedene Verhaltensmuster können zu komplexen Aktionsketten verkettet werden, wodurch eine segmentierte Aktionskette entsteht. GLEAM wird unter anderem zur Steuerung von Industrierobotern benutzt, die Auffassung von Lösungen als Aktionsketten sind hierbei besonders anschaulich: Eine Aktion kann zum Beispiel der Befehl an einen Aktor sein, mit einer angegebenen Beschleunigung eine Bewegung vorge-

gebener Weite durchzuführen. Ein Beispiel für ein Verhaltensmuster wäre eine Folge von Aktionen zum Wechseln eines Werkzeuges oder ein Nothalt aller Motoren.

3.2.1 Handlungsmodell K2

In Abschnitt 2.3.3 wurde der Suchraum mit heuristischer Ressourcenallokation vorgestellt. Mit GKomb2 oder K2 wird das dazugehörige kombinatorische Handlungsmodell für das GORBA-Problem bezeichnet. Für jeden GridJob gibt es genau eine Aktion, die jedoch keine weiteren Parameter mehr besitzt. Die Reihenfolge der Aktionen gibt die Reihenfolge an, mit denen die GridJobs der Allokations- und Planungsheuristik übergeben werden. Des Weiteren gibt es in jeder Aktionskette genau eine so genannte Ressourcen-Auswahlstrategie-Aktion. Sie besitzt einen ganzzahligen Parameter, der die zu verwendende Allokationsheuristik angibt und der der Evolution unterworfen werden kann. Die Position dieser Aktion ist irrelevant.

3.2.2 Handlungsmodell K1

In Abschnitt 2.3.4 wurde der Suchraum mit evolutionärer Ressourcenallokation eingeführt. Das entsprechende Handlungsmodell wird auch mit GKomb1 oder kurz K1 abgekürzt. Wiederum existiert für jeden GridJob existiert genau eine Aktion. Die Aktionen besitzen hierbei einen Integer Parameter, der evolutionär variiert werden kann. Der Parameter gibt an, welche Ressource aus der Liste der möglichen Ressourcen für diesen GridJob eingeplant werden soll. Die Reihenfolge der Aktionen gibt die Reihenfolge an, mit denen die GridJobs der Planungsheuristik übergeben werden. Eine Ressourcen-Auswahlstrategie-Aktion wird in diesem Fall ignoriert.

3.3 Die genetischen Operatoren

Aktionsketten werden im Rahmen des Evolutionsprozesses durch Mutationsoperatoren manipuliert und Nachkommen durch Anwendung von Rekombinationsoperatoren generiert. So können ein oder alle Parameter einer oder aller Aktionen innerhalb von gegebe-

nen Grenzen zufällig neu gesetzt oder um definierbare Schritt variiert werden. Beispiele für implementierte Operatoren mit Einfluss auf die Aktionsreihenfolge sind:

- Verschiebung eines Segmentes, also einer Folge von Aktionen
- Verschiebung einer einzelnen Aktion, auch über Segmentgrenzen hinaus
- Invertierung eines Segmentes, also eine Umkehrung der Anordnung der Aktionen
- Verschiebung von Segmentgrenzen
- Aufteilung eines Segmentes in zwei neue Segmente
- Verschmelzung zweier Segmente zu einem Segment

Über Rekombinationsoperatoren können Nachkommen erzeugt werden, welche Eigenschaften zweier Eltern vereinen. Das Verhalten der Rekombinationsoperatoren kann vielfältig konfiguriert werden. So kann vor Anwendung eines Rekombinationsoperators eine Prüfung des Hammingabstandes zwischen zwei Eltern stattfinden. Sind sich Eltern zu ähnlich, so wird auf eine Paarung verzichtet.

Es existieren zahlreiche Rekombinationsoperatoren zur Reihenfolgeoptimierung. Entsprechende Operatoren finden sich zum Beispiel in [ES03]. Auf Anraten des Autors dieser Arbeit wurden zwei solcher Operatoren von Herrn Dr. Jakob implementiert, ein Order-Crossover aus [Dav92] und der Precedence Preserving Crossover, der sowohl bei in [BW93] als auch in [BMK96] Erwähnung findet. Bei ersten Tests konnte eine leichte Steigerung der Performanz von GLEAM erreicht werden. Detaillierte Ergebnisse stehen jedoch noch aus.

3.4 Das Nachbarschaftsmodell

Das in HyGLEAM implementierte Populationsmodell wird in [GS90] dargestellt. Demnach wird eine Population als eine angeordnete Menge von Individuen aufgefasst, zwischen denen durch die Anordnung Nachbarschaftsbeziehungen bestehen können. Die Größe dieser durch Nachbarschaft (Demes) verknüpften Mengen wird als Deme-Size bezeichnet. Der

Selektionsmechanismus kann diese Nachbarschaft berücksichtigen, wodurch eine Nischenbildung ermöglicht wird. Dadurch wird eine genetische Vielfalt zwischen allen Individuen gesichert und eine zu schnelle Konvergenz des Verfahrens vermieden.

Das Verfahren verhält sich sowohl auf die Nachbarschaft, als auch auf diese Nachbarschaft von Knoten bezogen, elitär. Das bedeutet, dass weder die global beste Lösung, noch die beste Lösung der jeweiligen Demes im Rahmen der Evolution verworfen werden kann. Ein positives Konvergenzverhalten kann dadurch gesichert werden. Die Anzahl an Generationen, die ohne Verbesserung des globalen beziehungsweise lokalen besten Individuums verstreicht, wird neben einem Zeitlimit als ein Abbruchkriterium verwendet.

3.5 Lokaler Suchverfahren und Adaptionsmechanismus in HyGLEAM

Lokale Suchverfahren können auf verschiedene Arten in evolutionäre Algorithmen eingebunden werden. In [Jak04b] werden verschiedene untersuchte Möglichkeiten zur Integration solcher Verfahren vorgestellt. Für die vorliegende Arbeit wurde eine adaptive direkte Integration jeweils eines lokales Suchverfahrens gewählt. HyGLEAM ist in dieser Konfiguration als adaptiver, einfach-memetischer evolutionärer Algorithmus zu klassifizieren.

Der Adaptionsmechanismus entscheidet sehr differenziert, welche Individuen durch welche lokalen Suchverfahren bearbeitet werden sollen. Der Adaption sind folgende Parameter unterworfen:

- Die Wahrscheinlichkeit, alle Nachkommen einer Generation durch ein lokales Suchverfahren zu optimieren
- Bei Konfigurationen von HyGLEAM als adaptivem multi-memetischen Algorithmus jeweilige Wahrscheinlichkeiten zur Auswahl der verschiedenen lokalen Suchverfahren
- Für jedes Suchverfahren eine beliebige Anzahl definierbarer Steuerparameter

Für jeden Parameter ist eine sortierte Liste gültiger Werte anzugeben, unter denen der Adaptionsmechanismus wählt. Beim Aufruf eines lokalen Suchverfahrens wird diesem

eine Aktionskette und der Steuerungsparameter übergeben. Anhand der Anzahl der vom lokalen Suchverfahren aufgerufenen Evaluationen und der erzielten Note wird der Erfolg des Aufrufes als Notengewinn pro Rechenzeit abgeschätzt. Die Adaption kann zwischen verschiedenen Notenniveaus von Individuen differenzieren und die Parameter für diese Notenklassen unterschiedlich anpassen.

3.6 Durchführung der Benchmarks

Zur Durchführung eines Benchmarks für das GORBA-Problem sind zahlreiche Schritte durchzuführen. Zunächst muss das zu untersuchende Verfahren in HyGLEAM implementiert und integriert werden. Die Steuerungsparameter des lokalen Suchverfahrens sind zu definieren. Danach muss eine Version von HyGLEAM kompiliert werden. Hierzu gibt es zwei Möglichkeiten: Zum einen kann eine interaktive Version erstellt werden, die zahlreiche Kommandos zur Analyse bietet. Ebenso kann eine durch Skripte aufrufbare Version generiert werden, wie sie im Produktiveinsatz genutzt wird und die ohne weitere Benutzereingriffe Optimierungsläufe durchführen kann.

Das Handlungsmodell ist in einer entsprechenden Konfigurationsdatei zu definieren, unter anderem sind alle zu adaptierenden Parameter inklusive einer Werteliste vorzugeben. Weiterhin muss eine Experimentdatei erstellt werden, welche weitere Parameter definiert und in der sich Verweise auf zusätzliche Dateien befinden. Hierzu gehören Konfigurationsdateien wie jene zur Beschreibung des Handlungsmodells, Joblisten zur Beschreibung der Probleminstanz (siehe Abschnitt 2.5), die Ausgaben der konventionellen Planung als Initiallösungen und ein Ort zur Speicherung der Optimierungsergebnisse.

Über Shell-Skripte wird HyGLEAM aufgerufen, wobei der Ort der Experimentdatei, ein Speicherort für eine Logdatei sowie diverse Steuerungsparameter übergeben werden. Diese Steuerungsparameter beinhalten unter anderem das Zeitlimit und die zu verwendende Populationsgröße. Über geeignete Skripte kann HyGLEAM mehrfach aufgerufen werden, wodurch statistisch aussagekräftige Analysen möglich werden.

Für jedes in Abschnitt 2.5 beschriebene Testszenario ist dieser Prozess zu wiederholen. Jedes Szenario wird mit verschiedenen Handlungsmodellen untersucht und jeweils ist

durch Versuche die optimale Populationsgröße zu bestimmen.

Die Ergebnisse der Aufrufe werden in Logfiles geschrieben, welche mit Hilfe eines speziell generierten Auswertungsprogrammes analysiert werden können. Neben den zuvor beschriebenen Erfolgsraten und Notenverteilungen wird ebenso das Verhalten des evolutionären Algorithmus untersucht. Dies sind:

- Die Populationsgröße, welche vorgegeben wurde
- Die Anzahl der Adaptionen durch den Adaptionsmechanismus in HyGLEAM
- Die mittleren Level der adaptierten Parameter (nicht die mittleren Werte!)
- Die per Adaption festgelegte Wahrscheinlichkeit, das LSV auf alle Individuen anzuwenden
- Die mittlere Anzahl der untersuchten Generationen
- Die mittlere Anzahl der Evaluationen (LSVs zählen die Evaluationsaufrufe mit)

Für vergleichbare Ergebnisse wurden alle Tests auf identischer Hardware ausgeführt. Hierfür wurden 5 identisch ausgestattete Arbeitsplatzrechner genutzt, wobei ein Testprozess an einen von zwei CPU-Kernen gebunden war. Die Rechner sind mit AMD Opteron Dual Core Prozessoren und mehreren Gigabyte Arbeitsspeicher ausgestattet, wobei HyGLEAM einen vernachlässigbaren Speicherbedarf hat.

Nur wenn sichergestellt werden konnte, dass einem Testprozess stets die gesamte CPU-Leistung zur Verfügung stand, wurde ein entsprechender Prozess gestartet. Dies ist insbesondere nachts und am Wochenende oder bei geringer Belastung während der Arbeitszeit der Fall. Zur Testdurchführung wurden die vorbereiteten Shellskripte per SSH auf den Maschinen ausgeführt, während die Testumgebung über einen Fileserver per NFS zur Verfügung gestellt wurde.

Entsprechende Testläufe der Auswertung wurden mit Hilfe der Optimierung mittels GLEAM durchgeführt und dienen als Referenz für alle untersuchten lokalen Suchverfahren in HyGLEAM. In Abbildung 3.1 wurden die Erfolgsraten von GLEAM mit dem Genmodell K1 und einem genotypischem Reparaturmechanismus abgebildet. Abbildung 3.2 zeigt

die Erfolgsraten mit dem Genmodell K2 sowie einem phänotypischen Reparaturmechanismus.

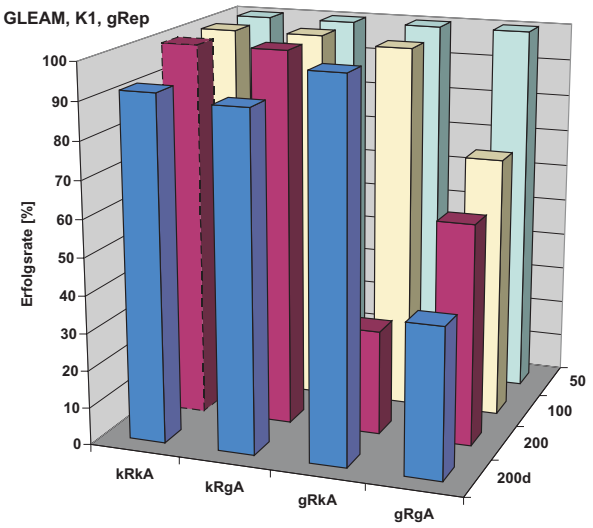

Abbildung 3.1: Erfolgsraten von GLEAM / K1 / gRep

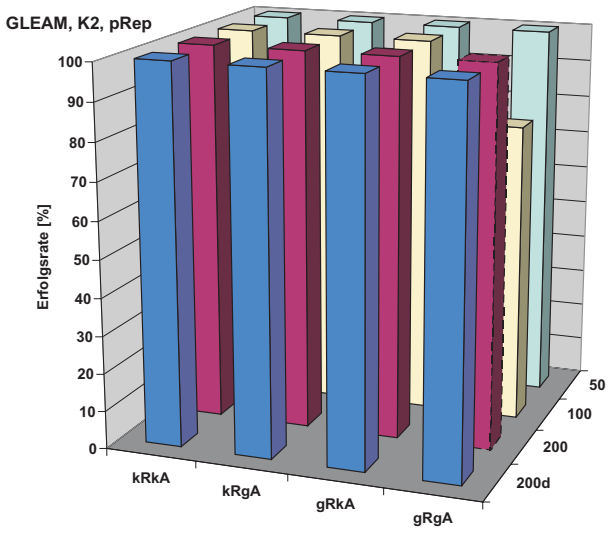

Abbildung 3.2: Erfolgsraten GLEAM / K2 / pRep

Kapitel 4

Lokale Suchverfahren für Reihenfolgeprobleme

Lokale Suchverfahren lassen sich durch folgendes Vorgehen beschreiben: Von einem Element des Suchraumes ausgehend werden schrittweise neue Elemente des Suchraumes ausgewählt und untersucht. Die Auswahlmöglichkeiten in jedem Schritt werden durch eine Nachbarschaftsfunktion angegeben. Die Gestalt des Suchraumes, die Nachbarschaftsfunktion sowie der Auswahlmechanismus sind wesentliche Charakteristiken lokaler Suchverfahren.

Eine gute Einführung in das Design lokaler Suchverfahren bietet das Kapitel 12 in [KT05]. Anwendungen in der kombinatorischen Optimierung werden in Kapitel 19 in [PS98], in Kapitel 3.5 in [Par02] sowie in [AL97] beschrieben. Einen Überblick über stochastische lokale Suchverfahren bietet [HS05].

Metaheuristiken sind besondere lokale Suchverfahren, evolutionäre Algorithmen wie HyGLEAM sind Vertreter dieser Algorithmenklasse. Suchraum, Nachbarschaftsfunktion und Auswahlmechanismus sind hierbei für verschiedene Optimierungsprobleme einsetzbar. Metaheuristiken arbeiten nach dem Black-Box Prinzip, die Lösung eines Problemes erfordert lediglich die Einbindung einer Funktion in die Metaheuristik, welche für ein gegebenes Element des Suchraumes dessen Qualität bezüglich einer Zielfunktion liefert. Die Anwendung einer Metaheuristik ist daher sehr einfach und kann in vielen Fällen bereits

zufrieden stellende Ergebnisse liefern.

Der Vorteil von Metaheuristiken, ihre universelle Einsetzbarkeit, ist mit einem Nachteil verbunden: Auf ein bestimmtes Problem zugeschnittene Algorithmen können in der Regel wesentlich bessere Ergebnisse erzielen als allgemeingültige Verfahren. Im Folgenden Abschnitt wird zunächst begründet, warum Metaheuristiken nur begrenzt leistungsfähig sein können und wie dem begegnet werden kann.

Im zweiten Abschnitt dieses Kapitels werden Optimierungsmuster als ein neues Konzept eingeführt. Optimierungsmuster dienen der Charakterisierung von Problemen und helfen beim Design neuer lokaler Suchverfahren. Dabei werden mehrere Optimierungsmuster eingeführt. Diese Optimierungsmuster wurden bei der Integration von lokalen Suchverfahren in HyGLEAM berücksichtigt.

4.1 No Free Lunch

Einem universellen und leistungsfähigen Optimierungswerkzeug sind gemäß dem „no free lunch“- Theorem Grenzen gesetzt [WM97]: Gegeben sei ein Verfahren, welches eine maximale erwartete Performanz über alle Probleme einer allgemeinen Problemklasse besitzt. Vereinfachend kann davon ausgegangen werden, dass es spezielle Verfahren gibt, welche bezüglich eines beschränkten Anwendungsbereiches eine höhere erwartete Performanz bieten als das allgemeingültige Verfahren. Beim Design eines Optimierungsverfahrens muß demnach ein Tradeoff zwischen seiner Leistungsfähigkeit im Spezialfall und der Bandbreite der bearbeitbaren Probleme gefunden werden.

Diesem Problem begegnet man beim tatsächlichen Einsatz von Metaheuristiken oft durch die Verwendung hybrider Algorithmen. Zusätzlich zur Evaluierungsfunktion, der „Black Box“, werden problemspezifische Algorithmen integriert. Diese nutzen Wissen über die spezielle Gestalt des Problems, um beispielsweise den Suchraum einzuschränken oder die Auswahlfunktion zu verbessern. Solche Algorithmen definieren teils andere Nachbarschaftsbeziehungen, welche eine effektivere Optimierung ermöglichen.

Das Spektrum der von HyGLEAM lösbaren Probleme ist durchaus groß. Durch die Integration lokaler Suchverfahren (LSVs) kann der Algorithmus auf bestimmte Problem-

klassen zugeschnitten werden. Möglichkeiten zur Integration von LSVs in HyGLEAM werden in [Jak04b] vorgestellt.

Eine allgemeine Problemklasse ist die multidimensionale Parameteroptimierung. Für diese Klasse wurden bereits zwei lokale Suchverfahren in HyGLEAM integriert, namentlich das Rosenbrock-Verfahren [Ros60] und der Complex-Algorithmus nach Box [Box65]. Im Rahmen zahlreicher Benchmarks konnte gezeigt werden, dass die Integration dieser Suchverfahren einen deutlichen Performanzgewinn bedeutet. Anwendbar sind die Verfahren jedoch nur dort, wo Parameter auf reellwertigen, großen ganzzahligen oder entsprechenden gemischt-ganzzahligen Skalen zu optimieren sind.

Eine andere Problemklasse ist die Optimierung von Permutationen von Elementen. Insbesondere das Genmodell mit heuristischer Ressourcenallokation von GORBA stellt ein solches Permutationsproblem dar. Ziel dieser Arbeit ist die Integration lokaler Suchverfahren für ebendiese Problemklasse - wobei die Universalität von HyGLEAM nicht aufgegeben werden soll und eine alleinige Fokussierung auf das GORBA-Problem vermieden werden soll. Für ein genaueres Verständnis dieser Problemklasse dient das im folgenden Abschnitt eingeführte Konzept der Optimierungsmuster.

4.2 Ein neues Konzept: Optimierungsmuster

Als ein neues Konzept soll an dieser Stelle das der „Optimierungsmuster“ eingeführt werden. Die Funktion $f_A(x, a) \in \{TRUE, FALSE\}$ gebe für jedes Element x eines Suchraums (zum Beispiel eine Permutation oder ein Bitstring) und für jeden in einer geeigneten Sprache A formulierten Ausdruck a an, ob das Element dem Muster von a entspricht. Mit „Optimierungsmustern“ seien im folgenden die Ausdrücke entsprechender Sprachen gemeint.

Zur Entwicklung eines lokalen Suchverfahrens sind nun jene Sprachen A zu identifizieren, für die ein Zusammenhang zwischen $f_A(x, a)$ und der Güte der Lösung x besteht. Ein geeignetes lokales Suchverfahren identifiziert jene Muster a, für die ein positiver (negativer) Effekt auf die Lösungsgüte zu erwarten ist und bevorzuget (meidet) bei der Auswahl eines Nachbarn jene Elemente, die diesen Mustern entsprechen.

Der Begriff des Optimierungsmusters soll zunächst anhand einiger Beispiele deutlicher gemacht werden.

4.2.1 Beispiele für Optimierungsmuster

Im folgenden werden acht Sprachen, A1 bis A8 definiert.

Schemata und Building Blocks

Schemata werden insbesondere bei der Analyse von genetischen Algorithmen herangezogen, siehe zum Beispiel [ES03]. Schemata sind Ausdrücke in einer Sprache, die bereits 1975 von Holland in seiner grundlegenden Arbeit zu genetischen Algorithmen formuliert wird [Hol75]. Ausdrücke a der Schema-Sprache **A1** für $n-$ dimensionale binäre Suchräume haben die Form $\{0, 1, -\}^n$. Dabei ist $f_{A1}(x, a)$ genau dann wahr, wenn x und a sich in allen Stellen, an denen in a kein $-$ steht, entsprechen.

Zahlreiche theoretische Untersuchen zum Laufzeitverhalten evolutionärer Algorithmen beruhen auf der Analyse von Schemata oder dem verwandten Building Blocks Konzept. Die Auswirkungen genetischer Operatoren auf die Verteilung von Schemata sind sehr gut untersucht. Nichtsdestotrotz spielen Schemata bei der Analyse von Permutationsproblemen eine eher untergeordnete Rolle.

Auf zwei Elemente bezogene Muster

Ebenso einfache Optimierungsmuster sind direkte Nachfolger von Elementen einer Permutation. Die Muster der Sprache **A2** bestehe aus Tupeln (i, j) von jeweils einem Vorgänger i und einem Nachfolger j. $f_{A2}(x, (i, j))$ ist genau dann wahr, wenn in der Permutation x das Element j direkt hinter i steht.

Das Traveling Salesman Problem (TSP) in der euklidschen Ebene ist ein Beispiel für ein Optimierungsproblem, bei dem die Optimierung dieser Muster eine große Rolle spielt. Die Wegstrecke des Handlungsreisenden entspricht der Summe der Wegstrecken zwischen allen besuchten Punkten. Jede einzelne Strecke hängt nur von den beiden durch sie verbundenen Punkten ab. Im Gegensatz zu diesen direkten Nachfolgerbeziehungen lassen

absolute Positionen von Punkten in der Reihenfolge wenig Aussagen über die Qualität einer Lösung zu. Ein sehr bekanntes und erfolgreiches lokales Suchverfahren hierzu ist das variable r-Opt Verfahren [LK73].

Eine einfache Erweiterung der Optimierungsmuster „direkte Nachfolger“ sind zweiwertige Ordnungen von Elementen. Die zugehörige Funktion der Sprache **A3**, $f_{A3}(x,(i,j))$, ist genau dann wahr, wenn in der Permutation x das Element j an einer beliebigen Stelle hinter i steht. Genauer als Optimierungsmuster der Sprache A3 und damit ebenfalls eine Erweiterung der Optimierungsmuster von $A2$ sind jene der Sprache **A4**. Diese Muster umfassen Elementabstände: Die Funktion $f_{A4}(x,(i,j,d))$ ist in diesem Fall genau dann wahr, wenn die Differenz der Positionen der Elemente j und i genau d beträgt.

Auf Mengen von Elementen bezogene Muster

Segmente sind Kombinationen aus mehreren direkten Nachfolgermustern, welche an einer beliebigen Stelle innerhalb einer Lösung vorkommen können. In der Sprache **A5** sei a ein Segment der Länge n_a von Elementen $a_0, a_1, ..., a_{n_a-1}, a_{n_a}$. Ein Element des Suchraumes sei durch eine Reihenfolge von n_x Elementen $x_1, x_2, ..., x_{n_x-1}, x_{n_x}$. Dann gilt: $f_{A5}(x,a)$ ist genau dann wahr, wenn es eine Position i gibt mit $x_i = a_0, x_{i+1} = a_1, ..., x_{i+n_a-1} = a_{n_a-1}, x_{i+n_a} = a_{n_a}$.

Während zweiwertige Ordnungen nur die relative Position zweier Elemente angeben, so beschreibt die Funktion $f_{A6}(x,a)$ der Sprache **A6** mehrwertige Ordnungen a, bei der die Elemente $a_0, a_1, ..., a_{n_a-1}, a_{n_a}$ in dieser Reihenfolge, aber mit beliebigen Zwischenabständen in x enthalten sind.

Im vorliegenden Kontext seien mit Clustern Muster a bezeichnet, welche mehrwertigen Ordnungen entsprechen, erweitert um die Angabe von Abständen zwischen den Elementen. Eine Sprache **A7** könnte als Sequenz von Elementen und dazwischenliegenden Abständen definiert sein. Der Ausdruck $a = a_0, d_1, a_1, d_2, ...$ bedeutet also, dass $f_{A7}(x,a)$ genau dann wahr ist, wenn es eine Position i gibt mit $x_i = a_0, x_{i+1+d_1} = a_1, x_{i+1+d_1+1+d_2} = a_2, ...$

Absolute Positionen

Optimierungsmuster dieses Typs sind Zuordnungen $a = (i, p)$ von Elementen i zu Positionen p: $f_{A8}(x, (i, p))$ ist wahr genau dann, wenn $x_p = i$ gilt. Damit ist die Sprache **A8** definiert.

4.2.2 Modellierung Lokaler Suchverfahren

Eine mögliche Vorgehensweise beim Design und der Analyse lokaler Suchverfahren basiert auf der Suche nach Optimierungsmustern. Durch Ausprobieren von Mustern kann ein Zusammenhang zwischen $f_A(x, a)$ und der Lösungsqualität ermittelt werden. Wenn durch mehrere Versuche ein Zusammenhang der Form $E(note(x)|f_A(x, a)) \neq E(note(x)|\overline{f_A(x, a)})$ festgestellt wird, so sind bei weiteren Versuchen solche Lösungen, die dem Muster a entsprechen, zu bevorzugen oder zu meiden.

Je mehr über ein Problem bekannt ist, desto besser läßt sich ein Lösungsverfahren entwickeln. Das Konzept des Optimierungsmusters ist bei der Analyse eines Problemes hilfreich, da es mit Hilfe von Sprachen charakterisiert werden kann. Kann für ein Problem eine Sprache A angegeben werden, deren Muster ein sicherer Indikator für die Lösungsqualität sind, so können lokale Suchverfahren angewendet werden, welche gezielt nach Ausdrücken der Sprache A suchen.

GLEAM besitzt Operatoren, welche vor allem Ausdrücke der Sprachen A2 und A3 (Elementmutationen, elementweise Crossover), A5 (Segmentmutationen, Segmentcrossover) und A8 (alle Reihenfolgemutationen und -Crossover) entsprechen. Desweiteren besitzt GLEAM Operatoren zur Parameteroptimierung, diese wurden aber im Rahmen dieser Arbeit nicht betrachtet. Im folgenden Kapitel werden verschiedene lokale Suchverfahren für das GORBA-Problem untersucht. Dabei wird jeweils auch die entsprechende Sprache angegeben.

Kapitel 5

Untersuchte lokale Suchverfahren

In diesem Kapitel werden die untersuchten lokalen Suchverfahren beschrieben. Zu jedem Verfahren wird jeweils die wesentliche Idee vorgestellt. Die Algorithmen werden beschrieben, wobei auch auf Besonderheiten bei der Implementierung eingegangen wird. Alle Verfahren wurden mit den Benchmarks zum GORBA-Problem getestet.

Im ersten Abschnitt wird das Verfahren „Stoss“ beschrieben. Es basiert darauf, dass einzelne Elemente innerhalb der Reihenfolge nach vorne verschoben werden - maximal soweit, bis sie an ihren Vorgänger „anstoßen“ und durch eine weitere Verschiebung die Vorgängerbeziehung verletzt werden würde.

Der zweite Abschnitt befasst sich mit dem Versuch, ein lokales Suchverfahren in Anlehnung an Ameisenalgorithmen zu entwerfen. Ein erstes Verfahren, „Ant 1“ wurde zunächst in einer einfachen Version getestet, die sich als unbrauchbar herausstellte. Eine erweiterte Version konnte bessere Ergebnisse erzielen, diese wird ebenfalls vorgestellt.

Ebenfalls an die Optimierung mit Hilfe von Ameisenalgorithmen angelehnt ist das lokale Suchverfahren „Ant 2“, welches im vierten Abschnitt beschrieben wird.

Im fünften und letzten Abschnitt dieses Kapitels wird der Ansatz „Triv“ beschrieben. Verschiedene, vergleichsweise triviale Operatoren definieren die Nachbarschaft, die hierbei untersucht wird. Durch einen einfachen Lernmechanismus werden günstige Operatoren identifiziert und wiederholt aufgerufen.

5.1 Lokales Suchverfahren: Stoss

Der Konstruktion lagen folgende Annahmen über die Interpretation von Reihenfolgen beim GORBA-Problem zugrunde:

1. Eine Verschiebung eines Elementes nach vorne erhöht dessen Priorität im Rahmen der Planung, wodurch der zugehörige Application Job schneller bearbeitet werden könnte.

2. Die Vorgängerbeziehungen des GORBA-Problemes sollten auch von der Jobreihenfolge berücksichtigt werden, auf der die Planung beruht.

3. Geringfügige Verschiebungen von Elementen haben meist nur geringfügige Effekte.

Das Suchverfahren manipuliert eine Reihenfolge von Elementen unter Berücksichtigung gegebener Vorgängerbeziehungen. Das Verfahren versucht, einzelne Elemente so weit nach vorne zu verschieben, bis diese an ihren Vorgänger „anstoßen“. Die durchsuchte Nachbarschaft ist darüberhinaus dadurch eingeschränkt, dass die Verschiebung nur in eine Richtung erfolgen kann, gültigkeitserhaltend sein muß und eine (adaptierbare) Mindestverschiebungsweite zu überschreiten ist.

Dem in Abschnitt 4.2 eingeführtem Konzept des Optimierungsmusters entsprechen ist dieses Verfahren der Sprache A1 zuzuordnen, welche in Abschnitt 4.2.1 erläutert wird. Hierbei wird gezielt untersucht, ob zwei Elemente eines Application Jobs besser in direkter Nachbarschaft stehen sollten oder nicht. Gleichzeitig wird für das ausgewählte Element dessen optimale Position gesucht, dies entspricht der Suche nach einem Muster der Sprache A8.

5.1.1 Algorithmus und Implementierung

Bei Verwendung des LSVs wird zu Beginn eines Optimierungslaufes von HyGLEAM eine Initialisierungsroutine aufgerufen, welche eine Matrix der Vorgängerbeziehungen ausliest und diese mit Hilfe des Warshall-Algorithmus zu ihrer transitiven Hülle erweitert [War62]. Die Erweiterung zur transitiven Hülle erfordert einmalig etwas Rechenzeit und ist ansonsten performanzneutral. Es kann so jedoch garantiert werden, dass nach Anwendung des

LSVs nicht mehr Vorgängerbeziehungen verletzt werden als dies in der übergebenen Permutation der Fall war.

In zufälliger Reihenfolge wird probiert, die einzelnen Elemente der Permutation zu verschieben. Für das jeweils ausgewählte Element wird dessen Vorgänger bestimmt. Ist der Abstand zwischen beiden größer als die Mindestverschiebungsweite, so wird das Element direkt hinter den Vorgänger verschoben und die neue Lösung wird bewertet. Andernfalls werden in zufälliger Reihenfolge alle anderen Elemente untersucht.

Dem LSV wird für einen Suchschritt eine Aktionskette übergeben, über 3 Parameter wird das Verhalten des LSVs gesteuert: die Pivotwahrscheinlichkeit, die Updatestrategie sowie die minimale Verschiebungsweite.

Die Pivotwahrscheinlichkeit gibt an, mit welcher Wahrscheinlichkeit der Parameter Pivot auf ERSTER gesetzt wird, andernfalls wird Pivot mit STEILSTER festgelegt. Die Pivotwahrscheinlichkeit wird über den Adaptionsmechanismus in HyGLEAM adaptiert und kann sieben Werte von 0 bis 1 annehmen. Ist Pivot auf ERSTER gesetzt, so bricht das Verfahren bei der ersten gefundenen Verbesserung ab und liefert das Ergebnis zurück. Ist der Parameter auf STEILSTER gesetzt, so werden alle Elemente der Permutation auf Verschiebbarkeit untersucht und das Ergebnis der erfolgreichsten Verschiebung wird zurückgeliefert.

Wenn der Steuerungsparameter Udate auf LAMACK gesetzt wurde, so wird die neue Note zurückgeliefert sowie die Verschiebung durchgeführt. Ist Update auf BALDWIN gesetzt, so wird lediglich die erzielbare Note zurückgeliefert. Die Wahrscheinlichkeit, mit der Update auf LAMACK gesetzt wird, kann vorgegeben werden.

Die minimale Verschiebungsweite wird ebenfalls über den Adaptionsmechanismus von HyGLEAM gesteuert. Die zehn möglichen Werte des Parameters liegen zwischen 0,05 und 0,7. Multipliziert man diesen Faktor mit der Anzahl zu planender Jobs, so ergibt sich die Distanz zwischen zwei Positionen, ab der eine Verschiebung eines Elementes als sinnvoll erachtet wird.

5.1.2 Ergebnisse der Benchmarks

Für jedes in Abschnitt 2.5 beschriebene Testszenario wurden Benchmarks mit verschiedenen Populationsgrößen durchgeführt. Abbildung 5.1 zeigt die Erfolgsraten, also die Wahrscheinlichkeit, die Rohnote der konventionellen Planung zu überschreiten. Ergebnisse dieser Tests sind im Anhang A.4 aufgeführt. Als Beobachtungen und Interpretationen lassen sich vor allem festhalten:

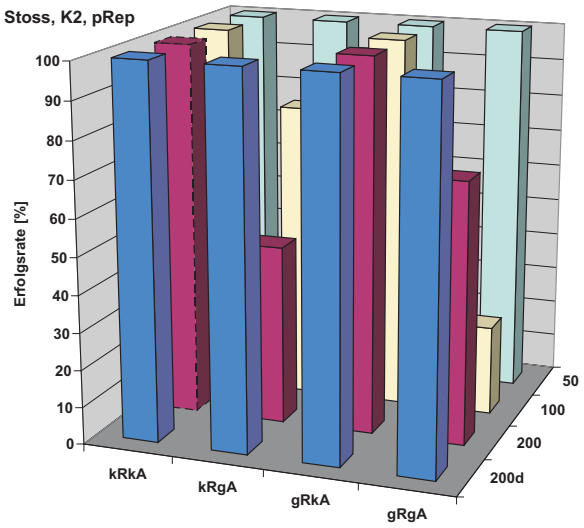

Abbildung 5.1: Erfolgsraten von Stoss / K2 / pRep

- Die erreichten Noten sowie die Erfolgsraten verfehlen in vielen Fällen nur sehr knapp das Niveau der Optimierung mit GLEAM, in manchen Situationen ist die Differenz jedoch deutlich.
- Die Anzahl der in 3 Minuten untersuchten Individuen liegt deutlich unter jenen bei der Optimierung mit GLEAM. Dadurch erklärt sich auch, warum die optimale Populationsgröße für alle Benchmarks unterhalb jenen von GLEAM liegt: Durch geringere Populationsgrößen können in der gleichen Zeit mehr Generationen untersucht werden.
- Die Anzahl durchgeführter Evaluationen liegt geringfügig über jenen der Optimierung mit GLEAM. Dies kann darauf zurückgeführt werden, dass das LSV pro Aufruf

teilweise mehrere Evaluationen durchführt und gewisse Aufgaben zur Pflege der Aktionskettenstruktur nur einmal pro LSV-Aufruf durchführt.

- Der Adaptionsmechanismus tendiert weder zu besonders niedrigen noch zu besonders hohen Werten für die Pivotwahrscheinlichkeit.
- Das Verhältnis von Mindestverschiebeweite zur Permutationslänge wird vom Adaptionsmechanismus meist auf etwa 1:10 reguliert. Dies bestätigt die Vermutung, dass zu kleine Verschiebungen zu wenig Effekt zeigen.

Als Fazit kann festgehalten werden, dass das LSV Stoss weder einen Vorteil, noch einen bedeutenden Nachteil brachte. Gleichwohl waren einige während der Implementierung gemachte Erfahrungen hilfreich bei der Programmierung neuer lokaler Suchverfahren.

Zukünftige Untersuchungen könnten sich damit beschäftigen, ob die Verschiebung von Elementen zu ihrem Vorgänger auch als intelligenter Mutationsoperator in GLEAM eingebunden werden kann. Bislang berücksichtigt kein Mutationsoperator in GLEAM die Vorgängerbeziehung.

5.2 HyGLEAM und Ameisenalgorithmen

Ameisenalgorithmen haben sich als besonders Leistungsfähig bei der Lösung von kombinatorischen Optimierungsproblemen erwiesen. Genau wie dies für evolutionäre Algorithmen gilt, so gibt es auch für Ameisenalgorithmen ein Vorbild in der Natur: Ameisenalgorithmen imitieren das Verhalten von Ameisenkolonien zur verteilten Suche optimaler Wege. An dieser Stelle sollen Ansätze beschrieben werden, bei der Konstruktion neuer lokaler Suchverfahren für HyGLEAM auf Konzepte von Ameisenalgorithmen zurückzugreifen.

5.2.1 Einführung in Ameisenalgorithmen

Zunächst soll jedoch kurz auf das Verhalten von Ameisen bei der Wegsuche und die Umsetzung im Rahmen von Ameisenalgorithmen beschrieben werden. Detailliertere Darstellungen finden sich in [BDT99, DS04]. Beispiele für die Anwendung von Ameisenalgorithmen im Scheduling liefern [Blu04, Blu02, BBHS00, MM05, SMM00].

Verhalten von Ameisen

Ameisen kommunizieren und speichern Wissen über den Verlauf von Wegen durch Konzentrationen von Duftstoffen (den Pheromonen). Ameisen orientieren sich bei der Suche eines Weges an Pheromonspuren, die von vorausgegangenen Ameisen gelegt wurden. Während Ameisen einen Weg verfolgen, deponieren sie dabei eigene Pheromone entlang der Strecke und bewahren somit die Information, welche ansonsten durch Verwitterung allmählich untergehen würde. Dieser Vorgang wird auch als Ausbleichen oder Evaporation bezeichnet.

Findet eine Ameise einen neuen, besseren Weg, so wird dieser neue Weg über Pheromonspuren markiert und den anderen Ameisen kommuniziert. Wird ein Weg hingegen unpassierbar, so werden Ameisen entlang dieses Weges keine Pheromone mehr deponieren. Durch die Evaporation werden keine Ameisen mehr auf diese Fährte gelockt.

Umsetzung als Ameisenalgorithmus

Ameisenalgorithmen imitieren dieses Verhalten. Dem Lauf einer Ameise zu einem Ziel entspricht ein Aufruf einer Konstruktionsfunktion, welche einen neuen Weg oder allgemeiner eine neue Lösung konstruiert. Bei der Konstruktionsroutine werden Entscheidungen auf Basis von Pheromonwerten und einer Heuristik getroffen. Nach der Konstruktion und Evaluation der neuen Lösung werden die Pheromonwerte angepasst.

Die Pheromonkonzentrationen werden üblicherweise in einer Pheromonmatrix T mit Werten τ_{ij} gespeichert. Die Werte η_{ij} beschreiben die Ergebnisse einer Heuristik H. Während des Konstruktionsprozesses wählt die Ameise im Zustand i die Alternative j gemäß des Ausdrucks $\tau_{ij}^{\alpha}\ \eta_{ij}^{\beta}$. Der Parameter α steuert dabei den Einfluss der Pheromonverteilung auf die Entscheidung, β tut dies mit den Ergebnissen der Heuristik.

Es existieren zwei Verfahren, nach denen eine Auswahl getroffen wird. Über den Parameter p_{wahl} wird die Wahrscheinlichkeit angegeben, die Alternative j gemäß der Verteilung von

$$P_i(j) = \frac{\tau_{ij}^{\alpha}\ \eta_{ij}^{\beta}}{\sum_j \tau_{ij}^{\alpha}\ \eta_{ij}^{\beta}}$$

auszuwählen. Ansonsten wird mit Wahrscheinlichkeit $1 - p_{wahl}$ der Ausdruck

$$\tau_{ij}^{\alpha} \; \eta_{ij}^{\beta}$$

maximiert.

Nach der Konstruktion und Evaluation einer oder mehrerer Lösung(en) wird die Pheromonmatrix anhand des gewonnenen Wissens aktualisiert. Mittels einer Updatefunktion werden hierbei in der Regel bestimmte Pheromonwerte τ_{ij} um einen Betrag c_{depot} erhöht oder erniedrigt. Außerdem werden die Werte der Pheromonmatrix durch eine Evaporationsfunktion regelmäßig abgesenkt, wodurch eine zu schnelle Konvergenz des Verfahrens vermieden wird.

5.2.2 Hybridisierungsansatz

Im Gegensatz zu klassischen Implementierungen des Ameisenalgorithmus wird dieser im vorliegenden Fall als Heuristik zur weiteren Verbesserung eines vorgegebenen Individuums verwendet. Heuristiken bei Ameisenalgorithmen dienen üblicherweise dazu, dem Ameisenalgorithmus problemspezifisches Wissen über gute Lösungen zuzuführen. Im vorliegenden Fall soll jedoch anwendungsneutral, also ohne die Einbeziehung einer solchen Information optimiert werden.

Der Ameisenalgorithmus als lokales Suchverfahren in HyGLEAM wird jedoch stets mit einer Information initialisiert: der Gestalt des zu optimierenden Individuums. Vor allem im späteren Verlauf der Optimierung mit HyGLEAM kann die Lösung, die das Individuum repräsentiert, sehr gut sein. So wurde sie im Laufe der Zeit von den evolutionären Operationen in HyGLEAM optimiert und das Individuum hat sich durch die Selektion gegenüber schlechteren Lösungen durchgesetzt. Gegebenenfalls hat das Individuum bereits Verbesserungen mit Hilfe anderer Optimierungsalgorithmen erfahren.

Die Heuristik H bewertet jene Alternativen j als besser, durch deren Auswahl im Zustand i die neu konstruierte Lösung der übergebenen Lösung möglichst ähnlich ist. Diese Ähnlichkeit kann auf verschiedene Distanzmaße bezogen sein. Die folgenden Suchverfahren, die auf diesem Hybridisierungsansatz beruhen sind genau wegen dieser Ähnlichkeit als lokale Suchverfahren einzuordnen.

5.3 Die erste Version von ANT 1

Eine mögliche Bedeutung eines hohen Pheromonwertes τ_{ij} ist, dass es günstig ist, Elemente i und j in der Reihenfolge direkt nebeneinander zu positionieren. Diese Interpretation der Pheromonwerte wird beispielsweise oft bei Handlungsreisendenproblemen angewandt. Das Verfahren ist der Sprache A1 zuzuordnen, welche in Abschnitt 4.2.1 eingeführt wird.

5.3.1 Grundgerüst des Algorithmus

Beim Aufruf des Verfahrens wird jeweils der als Pseudocode angegebene Algorithmus durchgeführt, siehe hierzu Abbildung 5.3.1.

5.3.2 Pheromonwerte

Der Pheromonwert $\hat{\tau}_{ij}$ bedeutet bei diesem Verfahren einen Zusammenhang zwischen der Lösungsqualität und der Tatsache, dass die Elemente i und j in der Reihenfolge direkt nebeneinander stehen.

5.3.3 Heuristikwerte

Eigenschaften der übergebenen und eventuell guten Lösung wird in Form der Heuristik H als Vorlage integriert. Der Heuristikwert wurde zunächst als

$$\hat{\eta}_{ij} = \begin{cases} 1,0 & j \text{ folgt in der Vorlage auf } i \\ 0,1 & \text{sonst} \end{cases}$$

definiert. Nach ersten Tests konnte jedoch eine Verbesserung der Ergebnisse erzielt werden, indem auf die Definition

$$\hat{\eta}_{ij} = \frac{1}{|ij|}$$

umgestellt wurde, der Wert entspricht also dem Kehrwert der Distanz zwischen den Elementen i und j in der vorgegeben Lösung.

```
WIEDERHOLE für jede Ameise:
  Konstruieren:
    Beginne bei Position 0 -> Wählen
  Wählen:
    wähle gemäß Auswahlverfahren einen Nachfolger
    WENN Vorgänger im Original anderen Nachfolger hatte
      DANN übernehme gewähltes Element -> Vorziehen
    WENN Ende erreicht
      DANN übernehme gewähltes Element -> Update
    Übernehme gewähltes Element -> Wählen
  Vorziehen:
    WENN Ende nicht erreicht UND Originalnachfolger ist erlaubt
      DANN übernehme Originalnachfolger -> Vorziehen
    SONST: Springe zum ersten noch nicht übernommenem...
                 ...Element aus Referenz -> Nachholen
  Nachholen:
    WENN Originalnachfolger noch nicht aufgenommen wurde
      DANN übernehme Originalnachfolger -> Nachholen
    SONST
      WENN alle Elemente geplant
         DANN -> Update
      SONST: Springe zum ersten noch nicht übernommenem...
                   ...Element aus Referenz -> Wählen
  Update:
    WENN Notenverbesserung erreicht
      DANN Deponiere Pheromone, TERMINIERE
      SONST Evaporiere Pheromone, TERMINIERE
```

Abbildung 5.2: Pseudocode der Startroutine von ANT 1, Version 1

5.3.4 Auswahlmechanismus

Der Auswahlmechanismus unterscheidet sich nicht von herkömmlichen Auswahlverfahren von Ameisenalgorithmen. Die Menge G aller gültigen Nachfolger g wird ermittelt, also all jener Elemente, deren Vorgänger bereits aufgenommen wurden. Zu beachten ist in diesem Fall, dass der Auswahlmechanismus nicht in jedem Schritt aufgerufen wird. Befindet sich der Algorithmus nicht im Modus „wählen“, so wird der Nachfolger des aktuellen Jobs in der Originalmatrix übernommen.

Für die Auswahl des ersten Elements wurden ebenso Pheromon- und Heuristikwerte definiert. Vereinfachend können diese als entsprechende Werte bezüglich eines nicht an eine andere Stelle positionierbaren „0-ten“ Elements gesehen werden.

In jedem Auswahlschritt werden aus den Pheromonwerten $\hat{\tau}_{ij}$ die Werte

$$\tau_{ij} = \frac{\hat{\tau}_{ij} * 1_{\{j \in G\}}}{\sum\limits_{G} \hat{\tau}_{ig}}$$

berechnet. Der Vektor τ_i ist demnach ein normierter Vektor, mit denen gültige Nachfolger des Jobs i bewertet werden und dessen Werte τ_{ij} für ungültige j stets 0 sind. Nach dem gleichen Verfahren wird auch der Heuristikvektor η_i gemäß

$$\eta_{ij} = \frac{\hat{\eta}_{ij} * 1_{\{j \in G\}}}{\sum\limits_{G} \hat{\eta}_{ig}}$$

gebildet.

Es wurden zwei Verfahren zur Auswahl implementiert, von denen eines mit Hilfe eines Steuerungsparameters p_{wahl} zum Zuge kommt. Mit einer Wahrscheinlichkeit von p_{wahl} wird für Position p ein Job j gemäß der Verteilung

$$P(p, j) = \frac{\tau_{pj}^{\alpha} * \eta_{pj}^{\beta}}{\sum\limits_{\tilde{j}} \tau_{p\tilde{j}}^{\alpha} * \eta_{p\tilde{j}}^{\beta}}$$

ausgewählt. Dagegen wird mit einer Wahrscheinlichkeit von $1 - p_{wahl}$ lediglich der Ausdruck $\tau_{pj}^{\alpha} * \eta_{pj}^{\beta}$ maximiert.

5.3.5 Pheromonupdate

Nach einer Bewertung der konstruierten Lösung wird verglichen, ob gegenüber der Ausgangslösung eine Verbesserung oder eine Verschlechterung erreicht wurde. Erreichte die konstruierte Lösung nicht die Qualität der bisherigen Lösung, so werden alle Pheromone gemäß $\hat{\tau}_{ij} = 0,9 * \hat{\tau}_{ij}$ evaporiert.

Wurde eine Verbesserung erzielt, so werden Pheromone in der Pheromonmatrix $\hat{T}$ deponiert. Zunächst wurde dafür folgendes Verfahren implementiert: Die Pheromonwerte $\hat{\tau}_{ij}$ werden bei allen Jobs i und j, welche in der neu konstruierten, nicht aber in der vorgegebenen Reihenfolge direkt aufeinander folgten um einen konstanten Wert (c_{depot}) erhöht. In ersten Tests erwies sich auch dieser Mechanismus als nicht sehr gut und es konnte eine leichte Verbesserung erreicht werden, wenn die Menge des deponierten Pheromons auch von der Qualität der Lösung abhängig gemacht wurde: Durch die Zuweisung

$$\hat{\tau}_{ij} = \hat{\tau}_{ij} + \sqrt{\frac{note}{c_{depot}}}$$

wird das Wissen über besonders gute Lösungen deutlicher in der Pheromonmatrix gespeichert als schlechte.

5.3.6 Adaptionsmechanismus

Über den Adaptionsmechanismus von HyGLEAM wurde gesteuert, wie viel Zeit in die Verbesserung eines Individuums investiert wird. Der entsprechende Steuerungsparameter wurde mit „Anzahl der Ameisen“ bezeichnet. Dem Adaptionsmechanisus wurden hierfür Werte zwischen 20 und 640 vorgegeben.

5.3.7 Parameterstudien

Die Steuerungsparameter α, β sowie p_{wahl} wurden im Rahmen von ersten Experimenten bestimmt. Als Grundlage diente das Benchmarkset mit 100 GridJobs, großem Abhängigkeitsgrad und vielen Ressourcenalternativen (siehe 2.5). Für jeden Parameter wurden jeweils 6 plausible Werte bestimmt. Jede Wertekombination wurde jeweils 100 mal ge-

testet. Ein Test bestand aus 20 bis 640 Iterationen des Suchverfahrens (der Anzahl der Ameisen), vor jedem Test wurde die Pheromonmatrix in einen Initialzustand versetzt.

Dieses Vorgehen zur Bestimmung der Parameter wurde gewählt, da nur so in vertretbarer Zeit sinnvoll erscheinende Werte für die Parameter bestimmt werden konnten. Eine Bestimmung der optimalen Populationsgröße für jeden GORBA-Benchmark ist selbst dann schwer an einem Tag zu bewältigen, wenn alle verwendbaren CPUs zur Verfügung stehen. Da die drei untersuchten Parameter jeweils voneinander abhängen, wäre hierfür ein Vielfaches an Rechenzeit nötig gewesen.

Die Ergebnisse der konventionellen Planungsheuristiken von GORBA wurden als Ausgangslösung bereitgestellt. Die Güte einer Konfiguration wurde über die durchschnittliche Differenz von erzielter Note und der besten Note der Ausgangslösung (Notenwert 7003) gemessen.

Mit Hilfe von AWK- sowie Linux-Shellskripten wurden die Ergebnisse aus den Logdateien extrahiert und aggregiert. Für jede Parameterkombination wurde für jede Wertekombination der durchschnittliche Notengewinn berechnet und in eine jeweilige Datei geschrieben. Zur Visualisierung der Ergebnisse wurden diese Dateien über ein gnuplot-Skript zu Grafiken aufgearbeitet. Als Beispiel wird in der Abbildung 5.3 der Effekt der Anzahl der Iteration (Anzahl Ameisen) und des Parameters β dargestellt. In Abbildung 5.4 werden p_{wahl} und β gegenübergestellt.

Die Auswertung ergab folgendes Bild:

- Für α kann festgestellt werden, dass sehr kleine Werte (deutlich unter 1) bessere Ergebnisse lieferten. Dies ist sehr schlecht, da dies eine starke Relativierung der Pheromonwerte bedeutet, diese also kaum einen Hinweis zur Findung besserer Lösungen lieferten.

- Die besten Ergebnisse wurden erzielt, wenn β auf 1,0 gesetzt war, wie in Abbildungen 5.3 und 5.4 leicht erkannt werden kann. Lediglich in der Anfangsphase konnten etwas bessere Ergebnisse mit geringeren Werten erzielt werden. Da eine Potenzierung mit 1,0 unnötig ist kann durch diese Parameterwahl Rechenzeit gespart werden.

- Für p_{wahl} konnte ein Bereich zwischen 0,05 und 0,15 als gut identifiziert werden,

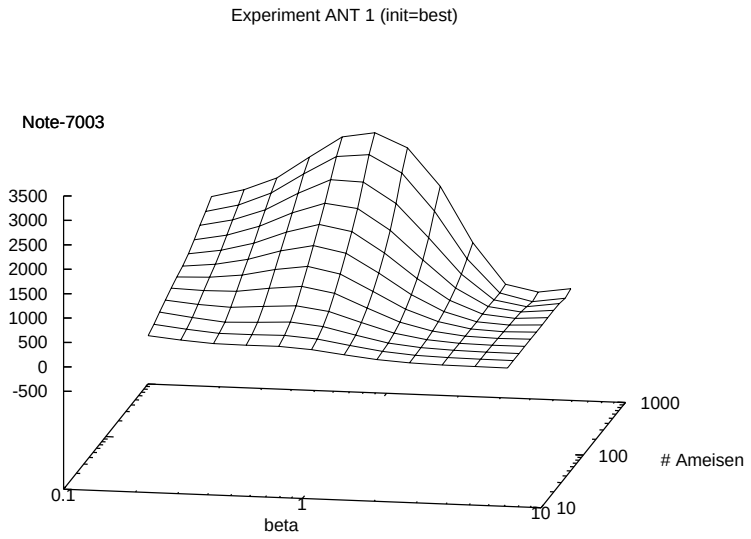

Abbildung 5.3: Ant 1v1: Parameter Anzahl Ameisen vs. β

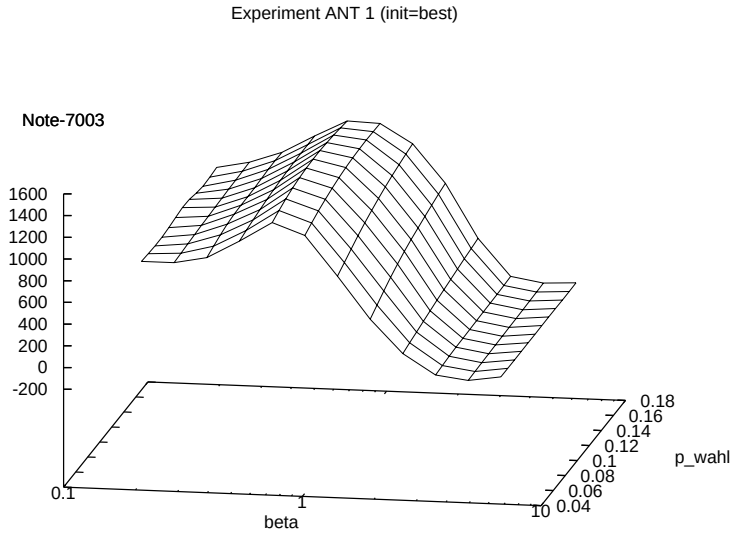

Abbildung 5.4: Ant 1v1: Parameter p_{wahl} vs. β

weswegen dieser Parameter im Folgenden mit 0,1 festgelegt wurde.

- Der offensichtliche Zusammenhang zwischen der Anzahl der Iterationen des Verfahrens und des Erfolges war abzusehen. Die Laufzeit nach einem Aufruf des Suchverfahrens hängt fast linear mit der Anzahl der Ameisen zusammen.

5.3.8 Bewertung des Verfahrens

Die ersten durchgeführten GORBA-Benchmarks für dieses Verfahren lieferten sehr schlechte Ergebnisse, was auf folgende Effekte zurückgeführt wurde:

- Die Anzahl der Ameisen wird vom Adaptionsmechanismus in die Nähe des mit 640 offensichtlich zu hoch angesetzten Maximalwertes geregelt. Der Adaptionsmechanismus zählt zur Aufwandsbewertung des LSV die Aufrufe der Planungs- und Bewertungsfunktionen. Der beträchtliche Aufwand für die Konstruktion einer Lösung wird ignoriert.

- Aus dem gleichen Grund wurde die Wahrscheinlichkeit, dass alle Individuen einer Generation durch das Verfahren optimiert werden auf sehr hohe Werte adaptiert. Die Anzahl der untersuchbaren Generationen sinkt dadurch.

- Die Anzahl der Evaluationen sank erheblich, dieses erste Ameisenverfahren war vermutlich obendrein sehr ineffizient implementiert.

- Der Aufwand für Konstruktion, Planung und Bewertung nahm bei großen Probleminstanzen überproportional zu. Die Anzahl der Evaluationen lassen einen mindestens quadratischen Aufwand der Konstruktionsroutine bezüglich der Anzahl von Elementen vermuten.

Die Vermutung lag nahe, dass das Verfahren ungeeignet ist. Um dies zu bestätigen wurde noch ein abschließendes Experiment unternommen. Der Auswahlmechanismus wurde dahingehend verändert, dass zwar eine gültige Alternative gemäß Pheromon- und Heuristikwerten berechnet wurde, dem Konstruktionsprozess jedoch eine zufällige gültige Alternative zurückgeliefert wurde. Kurze Tests ergaben, dass kein Nachteil gegenüber der

originalen Variante bestand. Ausführliche Benchmarks wurden mit dem Verfahren daher nicht unternommen. Im Ergebnis muss diese Version einer lokalen Suche also als gescheitert betrachtet werden.

5.4 Lokales Suchverfahren: ANT 1, Version 2

Die Experimente mit der ersten Version von ANT 1 haben bestätigt, dass die Qualität von Lösungen wenig von der direkten Nachbarschaft von Elementen in der Reihenfolge abhängt. Weiterhin sollte untersucht werden, ob der Abstand zwischen Elementen einer Permutation einen Rückschluss auf die Qualität zulässt.

Hierzu wurde im Gegensatz zum vorherigen Experiment der Versuch unternommen, Distanzen zwischen Elementen zu messen und zu optimieren. Dies entspricht einer Suche nach Optimierungsmustern der Sprache A2, wie sie in Abschnitt 4.2.1 vorgestellt wurde. Motiviert wurde der Ansatz aufgrund folgender Vermutungen:

- Nur die Position des letzten Grid Jobs eines Application Jobs hat einen wesentlichen Einfluss auf die Bewertung, diese Position sollte möglichst weit vorne sein. Die vorherigen Grid Jobs sollten jedoch möglichst spät bearbeitet werden, um die wertvollen Positionen zu Beginn der Reihenfolge nicht zu belegen. Daraus folgt, dass die Grid Jobs eines Application Jobs so dicht wie möglich beieinander stehen sollten.

- Ist diese Distanz jedoch zu gering, so könnte das im Falle von zwei Ressourcenkonkurrenten eines Application Jobs dazu führen, dass die Ressource noch belegt ist und der zweite Job eine schlechtere Alternative zugewiesen bekommt.

- Bei großen Distanzen steigt wiederum die Wahrscheinlichkeit, dass vor dem zweiten Job ein dritter Konkurrent um die Ressource zuvorkommt.

Es ist anzunehmen, dass der Effekt bei sehr großen Distanzen vernachlässigbar ist. Deswegen geht das im folgenden beschriebene Verfahren vom nahen Umfeld eines Elements aus. Es wird für alle Paare von Elementen untersucht, ob diese in guten Ketten „nahe beieinander“ stehen sollten oder eher nicht.

```
WIEDERHOLE für jede Ameise:
  WIEDERHOLE für jede Position:
    Bestimme nächses Element:
      Bestimme gültige Elemente
      Bestimme Pheromonwerte
      Bestimme Heuristikwerte\
      Wähle ein gültiges Element anhand beider Werte
  Update:
    WENN Notenverbesserung erreicht
      DANN Deponiere Pheromone
      SONST Evaporiere Pheromone
```

Abbildung 5.5: Pseudocode der Startroutine von ANT 1, Version 2

Erneut wurden Ameisenalgorithmen als Vorbild des Verfahrens herangezogen. Neue Lösungen werden konstruiert, in dem eine Auswahlroutine diesmal in jedem Schritt anhand von Pheromon- und Heuristikwerten das nächste Element bestimmt. Elemente, deren Vorgänger noch nicht in die Reihenfolge aufgenommen wurde, werden durch die Auswahlroutine nicht berücksichtigt. Zunächst werden die wesentlichen Komponenten des Verfahrens dargestellt.

5.4.1 Grundgerüst des Algorithmus

Im Gegensatz zum vorher dargestellten Verfahren wird die Auswahlroutine in diesem Fall bei jeder Belegung einer Position aufgerufen. Der Pseudocode ist in Abbildun 5.4.1 dargestellt.

5.4.2 Pheromonwerte

Ein hoher Wert $\hat{\tau}_{ij}$ drückt aus, dass das Element j möglichst kurz nach Element i zu positionieren ist. Ein geringer Wert bedeutet hingegen, dass ein Zusammenhang zwischen i und j nicht zu erkennen ist. Die Konstruktion berücksichtigt bei der Besetzung einer

Position für jeden Kandidaten j die Werte $\hat{\tau}_{ij}$ der zuletzt besetzten Elemente.

Für die Besetzung einer Position p wird für jedes mögliche Element j der Pheromonwert gemäß

$$\tau_{pj} = \sum_{a=1}^{9}(1 - a * 0,1)\hat{\tau}_{i_{p-a}j}$$

berechnet (wobei dieser Ausdruck bei den ersten Elementen etwas angepasst werden muß). Durch eine Gewichtung werden also die zuletzt belegten Positionen bzw. die dort positionierten Elemente stärker berücksichtigt.

5.4.3 Heuristikwerte

Die Heuristik H hat auch in diesem Fall eine andere Bedeutung als in herkömmlichen Ameisenalgorithmen. Der Wert $\hat{\eta}_{ij} = \frac{1}{|ij|}$ entspricht dem Kehrwert der Distanz zwischen den Elementen i und j in der vorgegeben Lösung. Die Heuristik berücksichtigt bei der Besetzung einer Position p für jedes mögliche Element j den (hier ebenfalls vereinfacht dargestellten) Ausdruck

$$\eta_{pj} = \sum_{b=1}^{9}(1 - b * 0,1)\hat{\eta}_{i_{p-b}j} = \sum_{b=1}^{9}\frac{1 - b * 0,1}{|i_{p-b}j|}$$

als Heuristikwert. Wie zuvor wird der Einfluss der Elemente auch durch die Heuristik gewichtet.

Wieder wird über die Heuristik eine Nähe zur übergebenen Lösung bewirkt. Statt über die Heuristik Wissen über gute Lösungen als Vorbild einzubinden dient die übergebene Lösung als Vorbild. Auch diese Version des Verfahrens ist eindeutig als lokales Suchverfahren einzuordnen.

5.4.4 Auswahlmechanismus

Der Auswahlmechanismus ist nicht weiter außergewöhnlich: Bei der Belegung von Position p werden die auf eine Summe von 1 normierten Vektoren τ_{pj} und η_{pj} mit Steuerungsparametern α und β potenziert. Es wurden zwei Verfahren zur Auswahl implementiert, von denen eines mit Hilfe eines Steuerungsparameters p_{wahl} zum Zuge kommt.

Mit einer Wahrscheinlichkeit von p_{wahl} wird für Position p ein Job j gemäß der Verteilung

$$P(p,j) = \frac{\tau_{pj}^{\alpha} * \eta_{pj}^{\beta}}{\sum_{\tilde{j}} \tau_{p\tilde{j}}^{\alpha} * \eta_{p\tilde{j}}^{\beta}}$$

ausgewählt. Dagegen wird mit einer Wahrscheinlichkeit von $1 - p_{wahl}$ der Ausdruck

$$\tau_{pj}^{\alpha} * \eta_{pj}^{\beta}$$

maximiert.

5.4.5 Pheromonupdate

Nach einer Bewertung der konstruierten Lösung wird verglichen, ob gegenüber der Ausgangslösung eine Verbesserung oder eine Verschlechterung erreicht wurde. Erreichte die konstruierte Lösung nicht die Qualität der bisherigen Lösung, so werden die Pheromone gemäß $\hat{\tau}_{ij} = 0,9 * \hat{\tau}_{ij}$ evaporiert.

Wurde eine Verbesserung erzielt, so werden Pheromone in der Pheromonmatrix $\hat{T}$ deponiert. Die Menge der Pheromone entspricht dem relativen Notengewinn gegenüber der vorgegebenen Lösung. Die Menge des tatsächlich an $\hat{\tau}_{ij}$ ausgeschütteten Pheromons ist durch $\min\{0; 1-(0,1*|ij|)\}$ gegeben, wobei $|ij|$ die Differenz der Positionen der Elemente i und j kennzeichne.

5.4.6 Adaptionsmechanismus

Über den Adaptionsmechanismus von HyGLEAM wird auch hier gesteuert, wie viel Zeit in die Verbesserung eines Individuums investiert wird. Der Steuerungsparameter hiervon ist wieder die Anzahl der Ameisen. Nach den vorangegangenen schlechten Erfahrungen mit besonders hohen Werten wurden dem Adaptionsmechanismus hierbei Werte zwischen 5 und 50 vorgegeben.

5.4.7 Parameterstudien

Erneut wurden zunächst die Steuerungsparameter α, β sowie p_{wahl} im Rahmen von einfachen Tests bestimmt. Das Verfahren entspricht weitgehend dem zuvor beschriebenen. Wieder wurde versucht, das beste Ergebnis der konventionellen Planung zu verbessern, wobei auch hier wiederholt Kombinationen plausibel erscheinender Parameterwerte verwendet wurden. Wie schon bei der ersten Version wurden diese Tests mit Hilfe von Skripten durchgeführt, ausgewertet und zu Grafiken verarbeitet.

Für p_{wahl} haben sich erneut niedrige Werte um $0,1$ als besonders günstig erwiesen. Noch wesentlich niedrigere Werte von p_{wahl} führen dazu, dass die Lösungen dem übergebenen Individuum zu ähnlich werden und dadurch seltener andere Bewertungen zustande kommen. Durch höhere Werte von p_{wahl} wird offenbar Wissen über gute Lösungen nicht ausreichend genutzt und es werden zufällige Lösungen konstruiert - was sehr selten erfolgreich ist.

Ähnliche Ergebnisse wie bei niedrigen Werten von p_{wahl} können auch mit hohen α und β erreicht werden. Hier machen sich jedoch zwei Effekte bemerkbar. Zum einen bedeuten höhere Werte von p_{wahl} häufigere Aufrufe einer Routine, die eine Auswahl gemäß einer Verteilung durchführt. Dies kostet etwas mehr Rechenzeit. Außerdem haben die Potenzierungen mit hohen α und β zu etwas weniger Evaluationen pro Zeit geführt - ein Indiz dafür, dass große Exponenten ebenso mehr Rechenzeit kosten.

Mit einem Wert von $0,1$ für p_{wahl} wurden die besten Ergebnisse mit β um $1,0$ erreicht werden. Dadurch können Potenzierungen eingespart werden, die etwas Rechenzeit kosten würden. Unter der Maßgabe $p_{wahl} = 0,1$ und $\beta = 1,0$ hat sich $\alpha = \frac{3}{4}$ als günstig erwiesen.

5.4.8 Ergebnisse der Benchmarks

Zwar sind die Ergebnisse der zweiten Version von ANT 1 wesentlich besser zu bewerten als jene der ersten Version. Wiederum wurden allerdings nicht überall die selben Erfolgsraten oder Notenwerte wie durch die Optimierung mit GLEAM erreicht. Abbildung 5.6 zeigt die Erfolgsraten von ANT 1 in der aktuellen Verision, ausführliche Ergebnisse finden sich in Anhang A.5.

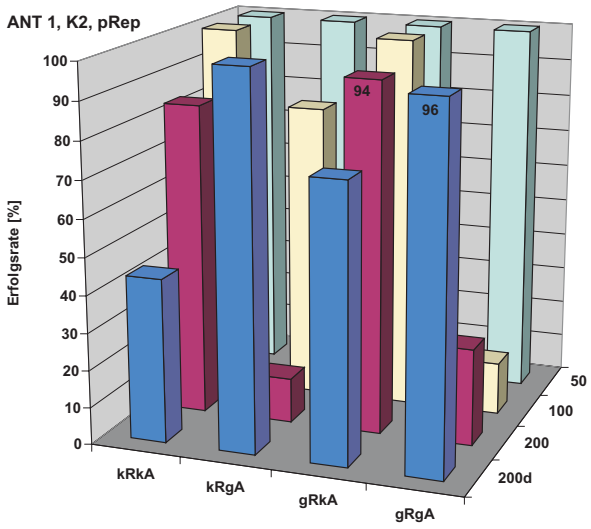

Abbildung 5.6: Erfolgsraten von Ant 1 / K2 / pRep

Die wichtigsten Erkenntnisse sind:

- Für alle Benchmarks mit 50 Grid Jobs wurden bereits nach kurzer Zeit Schedules gefunden, bei denen keine Straffunktion mehr zum Zuge kommt. Bei 100 Grid Jobs ist dies nur in der Hälfte der Benchmarks der Fall, nämlich bei kleinem Abhängigkeitsgrad. Bei 200 Grid Jobs wurden nur bei einem von acht Benchmarks zuverlässig straffreie Lösungen gefunden.

- Die Anzahl der Evaluationen nimmt mit wachsender Anzahl von Grid Jobs stärker als bei GLEAM ab. Dies erklärt, warum ANT 1 auch in dieser Version gegenüber GLEAM sehr schlecht abschnitt, während die Ergebnisse bei 50 Grid Jobs fast vergleichbar waren.

- In fast allen Fällen wurde die Anzahl der Ameisen vom Adaptionsmechanismus recht niedrig festgesetzt, oft wurde das Minimum von fünf Iterationen gewählt.

- Experimente, bei denen HyGLEAM und ANT 1 mehr Zeit zugestanden wurde (6, 9, 12 und 15 Minuten), brachten keine überraschende Erkenntnis: Die Erfolgsraten und die mittleren Noten steigen an, bei steigender Anzahl an Evaluationen profi-

tiert HyGLEAM zunehmends von größeren Populationsgrößen, dies entspricht einer ausführlicheren Breitensuche.

- Ebenso wurden ausnahmsweise auch einzelne Experimente mit dem Genmodell K1 vorgenommen. Wie erwartet schnitt das Verfahren in diesen Fällen deutlich schlechter ab.

Das Verfahren besitzt also ebenfalls Schwachstellen, ist aber nicht als vollständiger Misserfolg zu werten. So stellt das Benchmarkszenario GG 200 für GLEAM ein großes Problem dar, nur bei einer Populationsgröße wurde in drei Minuten stets Straffreiheit erreicht. Daher wurde dieses Szenario auch mit längeren Laufzeiten von HyGLEAM getestet.

Bei Zeiten von 12 oder 15 Minuten konnte ANT 1 in der Hälfte der Tests Straffreiheit erreichen. Dies ist zwar nach wie vor schlechter als das Ergebnis von GLEAM. Bemerkenswert ist aber, dass HyGLEAM und ANT 1 dieses Ergebnis mit einem Bruchteil an durchgeführten Evaluationen erreichte. Dies könnte so interpretiert werden, dass ANT 1 die Information aus einer Evaluation besser nutzt als GLEAM.

Der Programmcode von GLEAM ist bereits hochgradig optimiert. Bei der vorliegenden Implementierung von ANT 1 wurde jedoch zunächst vor allem Wert darauf gelegt, wiederverwendbare Schnittstellen und verständlichen Code zu produzieren. Gelänge es, diesen Code hinsichtlich kurzer Laufzeiten zu optimieren, so könnte in diesem Fall mit geringfügigen Verbesserungen gerechnet werden.

Ebenso wäre zu erwarten, dass bei Problemen mit rechenintensiveren Evaluationsroutinen HyGLEAM mit dieser Version von ANT 1 besser abschneidet als ohne. Entsprechende Untersuchungen sind jedoch nicht das Ziel dieser Arbeit gewesen.

5.5 Lokales Suchverfahren: ANT 2

Ein weiteres Verfahren auf der Basis des in Abschnitt 5.2.2 vorgestellten Hybridisierungsansatzes ist das lokale Suchverfahren ANT 2. Beide Versionen von ANT 1 haben gemeinsam, dass sie die relative Position von Elementen zueinander untersuchen. Ein alternativer

Ansatz ist, absolute Positionen von Elementen in der Reihenfolge zu bestimmen.

Elemente werden durch die Allokationsheuristiken von GORBA der ermittelten Reihenfolge nach in einen Belegungsplan übernommen. Das Verhalten der Planungs- und Allokationsheuristiken ist für jedes Element vom Zustand des bis dato erstellten Belegungsplanes abhängig. Das hier dargestellte Verfahren beruht auf zwei Annahmen:

1. Es existiert eine Korrelation zwischen der Position eines Elementes und der Qualität der repräsentierten Lösung.

2. Für ein gegebenes Element sind die Korrelationen zwischen der Lösungsqualität und zwei benachbarten Positionen ähnlich.

Die erste Annahme impliziert, dass zur Optimierung von Reihenfolgen Muster der Sprache A8 (vgl. Abschnitt 4.2.1) untersucht werden sollten. Für die identifizierten Muster a sollte dann gelten:

$$E(note(x)|f_{A8}(x,i,p)) > E(note(x)|\overline{f_{A8}(x,i,p)})$$

Indem die Werte einer Pheromonmatrix τ_{ip} als $E(note(x)|f_{A8}(x,i,p))$ interpretiert werden kann dies erreicht werden.

Die zweite Annahme lässt sich auch wie folgt übersetzen: Die Verteilung der erwarteten Lösungsqualität über alle möglichen Positionen eines Elementes besitzt wenige große Sprünge. Bei der Positionierung eines Elementes innerhalb einer gewissen Nachbarschaft ist die erwartete Lösungsqualität für die einzelnen Positionen dieser Nachbarschaft also ähnlich. Der Ausdruck $E(note(x)|f_{A8}(x,i,p))$ lässt sich dann aus den Werten der Nachbarpositionen $E(note(x)|f_{A8}(x,i,p-1))$ und $E(note(x)|f_{A8}(x,i,p+1))$ und somit aus der Dichtefunktion des Erwartungswertes über alle Positionen abschätzen.

Ein Verfahren hierzu ist die Relative Pheromone Evaluation Rule, siehe hierzu [DM02]. Dabei handelt es sich um eine erweiterte Variante der Summation Rule, welche in [MM00, MM03] vorgestellt wurde. Bei diesem LSV entsprechen die Werte τ_{ip} einer Abschätzung für $E(note(x)|f_{A8}(x,i,p))$. Die Relative Pheromone Evaluation Rule bestimmt die Interpretation der Pheromonmatrix: Dabei wird $E(note(x)|f_{A8}(x,i,p))$ anhand der Verteilung der τ_{ip} geschätzt.

5.5.1 Grundgerüst des Algorithmus

Das Verfahren entspricht im wesentlichen jenem der zweiten Version von ANT 1, siehe Abschnitt 5.4.1.

5.5.2 Pheromonwerte

Die Pheromonwerte τ_{ip} repräsentieren einen Zusammenhang zwischen der Lösungsqualität und der Tatsache, dass das Element i in der Reihenfolge an Position p steht. Bei der Auswertung der Pheromone wird jedoch die Verteilung F_i über die Positionen p verwendet:

$$F_i(p) = \frac{\sum_{a=1}^{p} \tau_{ia}^{\alpha}}{\sum_{b=1}^{p_{max}} \tau_{ib}^{\alpha}}$$

5.5.3 Heuristikwerte

Auch in diesem Fall wird anstelle einer anwendungsspezifischen Heuristik über η_{ip} eine Ähnlichkeit der konstruierten Lösung zum übergebenen Individuum bewirkt. Dabei wird bei der Belegung einer Position p für jedes Element i der Kehrwert der Distanz von p und der Position p_i^* von i im übergebenen Individuum genutzt:

$$\eta_{ip} = \frac{1}{|p - p_i^*|}$$

5.5.4 Auswahlmechanismus

Der Auswahlmechanismus besteht wieder in einer Auswahl eines Elementes i, wobei mit Wahrscheinlichkeit p_{wahl} entsprechend einer Verteilung $P(i|p)$ gewählt wird. In allen anderen Fällen, also mit einer Wahrscheinlichkeit von $1 - p_{wahl}$, wird dieser Ausdruck hingegen maximiert.

$$P(i|p) = \frac{\left(\frac{\sum_{a=1}^{p} \tau_{ia}^{\alpha}}{\sum_{b=1}^{p_{max}} \tau_{ib}^{\alpha}} \right) * \eta_{ip}^{\beta}}{\sum_{j \in O} \left(\frac{\sum_{a=1}^{p} \tau_{ja}^{\alpha}}{\sum_{b=1}^{p_{max}} \tau_{jb}^{\alpha}} \right) * \eta_{jp}^{\beta}}$$

5.5.5 Pheromonupdate

Bei Erfolg des Verfahrens werden jene Werte τ_{ip} der Pheromonmatrix T erhöht, bei denen in der gefundenen Lösung Element i an Position p steht. Ansonsten entspricht der Mechanismus jenem von ANT 1 in der ersten Version (siehe Abschnitt 5.3.5).

5.5.6 Parametrisierung des Verfahrens

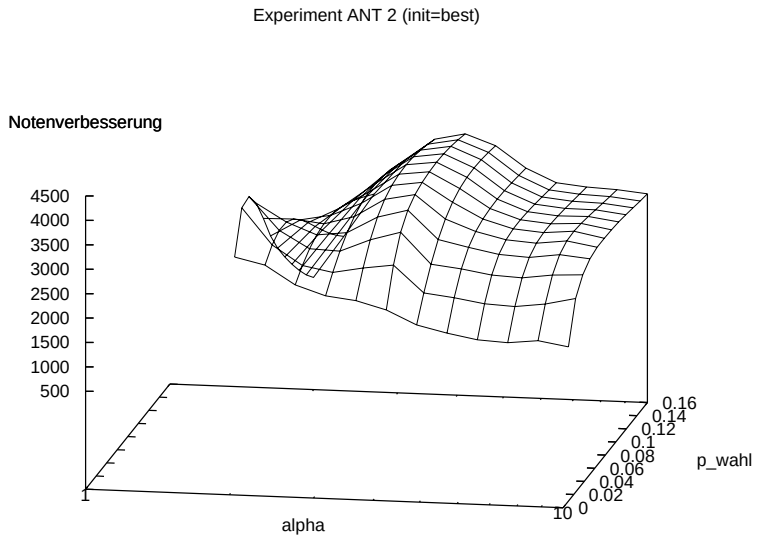

Abbildung 5.7: Ant 2: Parameter α vs. p_{wahl}

Auch die Steuerungsparameter des Verfahrens ANT 2 wurden mit Hilfe einer Voruntersuchung bestimmt. Das vorher angewandte Verfahren wurde beibehalten. Entsprechend dieser Untersuchung wurden folgende Werte ermittelt:

- $\alpha = 4$, Abbildung 5.7 zeigt, dass diese Wahl nahe liegt.
- $\beta = 2$
- $p_{wahl} = 0.1$, auch diese Wahl kann anhand Abbildung 5.7 nachvollzogen werden: Für kleinere Werte als $0,1$ bricht die Leistung meist ein, bei größeren Werten steigt der Bedarf an Rechenzeit pro Iteration.

5.5.7 Ergebnisse der Benchmarks

Ergebnisse der Tests sind in Abbildung 5.8 dargestellt, umfangreiche Messwerte befinden sich in Anhang A.6.

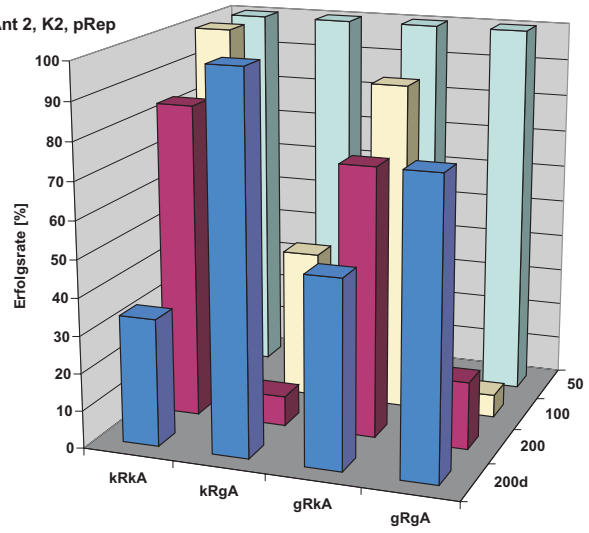

Abbildung 5.8: Erfolgsraten von Ant 2 / K2 / pRep

Die wichtigsten Erkenntnisse sind:

- Die Ergebnisse sind erneut als unbefriedigend zu bewerten, Erfolgsraten und erreichte Noten sind stets unterhalb jenen von GLEAM.
- Die Konstruktionsroutine hat einen extrem hohen Rechenaufwand. In Tabelle 5.9 werden die in 3 Minuten durchgeführen Evaluationen von GLEAM, Ant 1 in der

Anzahl der Grid Jobs	GLEAM	Ant 1v2	ANT 2
50	1.500.000	290.000	235.000
100	570.000	70.000	56.000
200	190.000	12.000	10.000

Abbildung 5.9: Anzahl der Evaluationen von GLEAM, Ant 1v2 und ANT 2 für gRgA (gerundet)

zweiten Version sowie ANT 2 gegenübergestellt, wobei gerundete Werte für die Benchmarkszenarien gRgA benutzt wurden.

Auch das Verfahren ANT 2 ist somit als nicht geeignet zu bewerten. Dies ist aufgrund der Anzahl durchgeführter Evaluationen nicht weiter verwunderlich. Zu untersuchen bleibt, ob bei der gleichen Anzahl an Evaluationen gegenüber GLEAM die Ergebnisse ebenso schlecht ausfallen. Hierzu wurden einige Versuche mit wesentlich längeren Optimierungszeiten unternommen, die jedoch aufgrund des enormen Zeitbedarfs abgebrochen werden mußten.

5.6 Lokales Suchverfahren: TRIV

Die von Ameisenalgorithmen inspirierten Verfahren zuvor sind nicht zuletzt am gewaltigen Rechenaufwand gescheitert, den die Auswahl einzelner Grid Jobs erfordert hat. Dieser Overhead lässt sich daran festmachen, dass für große Anzahlen von gridJobs der Anteil der verwendeten Rechenzeit für die Evaluationsroutine abnimmt, während der Anteil für die Konstruktionsroutine zunimmt.

Als Antwort hierauf wurde ein neuer Ansatz verfolgt. Dieses Suchverfahren besitzt verschiedene mehr oder weniger triviale Operatoren, mit denen Reihenfolgen manipuliert werden können. Jeder dieser Operatoren berücksichtigt dabei Optimierungsmuster einer jeweilgen Sprache und definiert eine eigene Nachbarschaftsfunktion im Suchraum. Das Verfahren wird von einem Lernmechanismus gesteuert, welcher die anzuwendende Operation bestimmt.

Ziel des Verfahrens ist auch, Hinweise für weitere zu testende Operatoren oder lokale

Suchverfahren zu finden. Hierzu kann analysiert werden, welche Operatoren wesentlich zum Erfolg eines Verfahrens beitragen und welche eher unbedeutend sind.

Cluster beschreiben Teilmengen von Elementen einer Lösung, welche in einer beliebigen Weise über Vorgängerbeziehungen miteinander verknüpft sind. Zur Beschreibung eines Clusters gehören auch die zwischen den Elementen liegenden Abstände. Das Konzept des Clusters wurde an dieser Stelle eingeführt, um die Anwendungsneutralität zu unterstreichen: Zwar sind Cluster in diesem Beispiel stets gleichbedeutend mit Application Jobs. Das Verfahren nutzt zur Ermittlung der Zusammenhänge jedoch nur die Vorgängerbeziehung, wodurch auch andere Anwendungen angepasst werden können.

Die einzelnen untersuchten Operatoren sind:

Translation von Elementen: Elemente werden verschoben. Der Verschiebungsweite ist durch die Vorgängerbeziehung eine Grenze gesetzt, da kein Element vor einem seiner Vorgänger oder hinter einem seiner Nachfolger positioniert werden kann. Bei dieser Operation werden vor allem Muster der Sprachen A2 und A8 untersucht.

Translation von Clustern: Bei dieser Operation werden alle Elemente eines Clusters um eine identische Weite nach vorne oder nach hinten verschoben. Aufgrund der Abgeschlossenheit des Clusters über die Vorgängerbeziehungen können diese ignoriert werden. Diese Operation bewahrt bei jeder Operation ein Muster der Sprache A7 bei, während es dessen Elemente verschiebt und somit mehrere Muster der Sprache A8 untersucht.

Translation von Segmenten: Hierbei werden alle Elemente eines Segments um eine identische Weite nach vorne oder nach hinten verschoben. Erneut muß eine Prüfung stattfinden, ob durch die Verschiebung eine Vorgängerbeziehung verletzt wird. Hierbei werden also Muster der Sprache A5 während einer Manipulation gezielt beibehalten, während viele neue Muster der Sprache A8 untersucht werden.

Permutation von Elementen: Zwei beliebige Elemente können ihre Position tauschen. Dabei ist zu prüfen, ob zwischen diesen beiden Positionen ein Element steht, welches Nachfolger des ersten oder Vorgänger des zweiten Elements ist. Erst wenn dies nicht der Fall ist darf ein Austausch durchgeführt werden. Die zugehörige Sprache ist A3.

Permutation von Clustern: Diese Operation ordnet die Elemente von zwei Clustern neu an. Dabei behalten die Elemente der beiden Cluster jeweils ihre Reihenfolge. Alle Elemente, die nicht Teil eines der beiden Cluster sind, behalten ihre absoluten Positionen. Bei dieser Operation werden also Muster der Sprache A6 beibehalten, während sowohl Muster der Sprachen A7 und A8 untersucht werden.

Kontraktion von Clustern: Die Operation besitzt zwei Parameter (Fixpunkt und Skalierungsfaktor) und wird auf Cluster angewendet. Der Fixpunkt bezeichnet eine beliebige Position der Permutation. Der positive, reellwertige Skalierungsfaktor ist stets kleiner als 1. Vom Fixpunkt ausgehend werden durch Verschiebungen alle Abstände zwischen dem Fixpunkt und den Elementen des Clusters anhand des Skalierungsfaktors verringert. Hier und beim folgenden Operator werden Muster der Sprache A7 untersucht, während die über die Sprache A6 beschriebenen relativen Ordnungen erhalten bleiben.

Expansion von Clustern: Als Umkehrfunktion der Kontraktion wird bei der Expansion der Abstand zwischen Fixpunkt und den Elementen des Clusters erhöht.

Nachdem für die entsprechenden Elemente mit Hilfe der Operatoren neue Positionen festgelegt wurden, werden alle verbleibenden Elemente in ihrer bisherigen Reihenfolge auf die noch nicht vergebenen Positionen verteilt.

Welche Operation jeweils durchgeführt wird, wird über einen eigenen Mechanismus entschieden. Für jeden Operationsaufruf wird protokolliert, welche Operation durchgeführt wurde und ob sie Erfolg hatte. Die Wahrscheinlichkeit, mit der ein Operator durchgeführt wird, richtet sich an seiner Treffsicherheit in der Vergangenheit, wobei Mindestwahrscheinlichkeiten beachtet werden. Dieses Verfahren ist auch als wiederholte, adaptiv gesteuerte Mutation ausgewählter Individuen zu sehen.

5.6.1 Algorithmus und Implementierung

Zur Identifikation der über Vorgängerbeziehungen zusammenhängenden Elemente wird zunächst eine Initialisierungsroutine aufgerufen. Zunächst wird das Produkt aus der ge-

gebenen Vorgängermatrix mit der transponierten Vorgängermatrix gebildet. Die Diagonalelemente der so ermittelten Matrix werden auf TRUE gesetzt. Anschließend wird mit Hilfe des Warshall-Algorithmus [War62] die reflexiv-transitive Hülle gebildet. Anschließend werden die Elemente nacheinander analysiert: Wenn sie nicht mit einem bereits analysiertem Element verbunden sind, so gründen sie einen neuen Cluster. Andernfalls sind sie mit einem Element verbunden, dem bereits ein Cluster zugeordnet ist, und erben diese Zuordnung.

Die in Abschnitt 5.6 vorgestellten Operatoren wurden als Funktionen implementiert, die mit Angaben über die zu manipulierenden Elemente sowie gegebenenfalls zufälligen Steuerungsparametern (der Verschiebungsweite oder dem Fixpunkt und dem Skalierungsfaktor) aufgerufen werden. Die Operatoren werden nur ausgeführt, wenn durch sie keine Vorgängerbeziehung verletzt wird.

Der triviale Lernmechanismus speichert für jedes Verfahren die Anzahl der erfolgreichen und der erfolglosen Versuche, per Mutation eine Verbesserung erbeizuführen. Findet ein Operator eine bessere Lösung, so steigt seine Aufrufwahrscheinlichkeit. Findet er keine Lösung, so wird die Wahrscheinlichkeit allmählich wieder heruntergeregelt. Ebenso wäre eine Ansteuerung der einzelnen Operatoren mit Hilfe des von HyGLEAM bereitgestellten Adaptionsmechanismus möglich. Hiergegen sprach die Vielzahl der Operatoren, da Erfahrungen gezeigt hatten, dass HyGLEAM nicht zu viele Parameter gleichzeitig sinnvoll adaptieren kann.

Bei jedem Aufruf des Verfahrens werden 8 Versuche unternommen, mit Hilfe eines Operators eine Verbesserung zu erzielen.

5.6.2 Ergebnisse der Benchmarks

Abbildung 5.10 zeigt die Erfolgsraten. Im Anhang A.7 sind die ausführlichen Benchmarkergebnisse dargestellt.

Die Messwerte der Anzahl an Evaluationen ist nicht plausibel, da nicht weniger Evaluationen als das Produkt aus Populationsgröße und Generationsanzahl berechnet werden können. Offenbar wurden durch einen Fehler zahlreiche Evaluationen nicht gezählt. Damit kann erklärt werden, warum der Adaptionsmechanismus von HyGLEAM die Wahrschein-

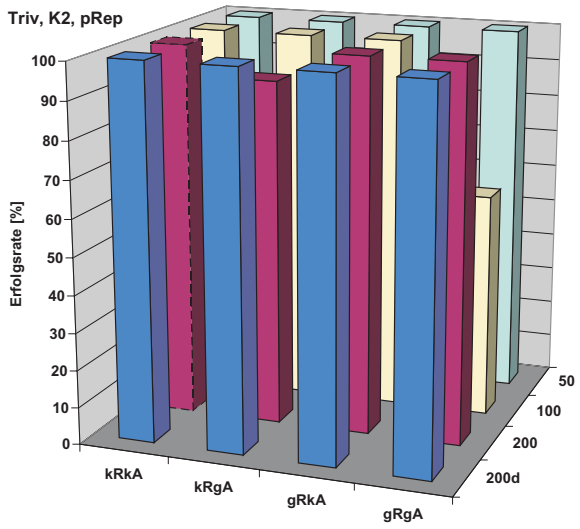

Abbildung 5.10: Erfolgsraten von Triv / K2 / pRep

lichkeit, alle Individuen zu optimieren, stets auf den Maximalwert geregelt hat. (Das in den Benchmarkergebnissen angegebene Level 5 entspricht der höchsten Wahrscheinlichkeit.)

Werden dem lokalen Suchverfahren zu wenige Evaluationen zugerechnet, so steigt die Erfolgsrate dieser Evaluationen. Folgerichtig entscheidet der Adaptionsmechanismus, mehr Rechenzeit in das Verfahren zu investieren. Dies macht er, indem er die Wahrscheinlichkeit erhöht, das lokale Suchverfahren Triv auf alle Individuen jeder Generation einmal anzuwenden.

Pro Aufruf von Triv werden 8 Evaluationen durchgeführt. Schätzt man die Anzahl der Evaluationen mit 8 mal dem Produkt aus Populationsgröße und Generationsanzahl ab, so erhält man eine vorsichtige Abschätzung der Evaluationen. Diese scheint jedoch realistischer zu sein als die gemessene Anzahl. Eine Wiederholung aller Tests war leider nicht möglich, da diese erneut etwa 200 CPU-Stunden gekostet hätte, zusätzlich zu einem Vor- und Nachbereitungsaufwand.

Die Erfolgsraten von Triv sind jedoch wesentlich besser als die der Ameisen-inspirierten Verfahren. In zwei von 16 Szenarien wurden in 8% beziehungsweise 40% der Tests keine straffreien Lösungen gefunden. Im zweiten Fall, beim Benchmark gRgA 100, gelang dies

aber auch mit GLEAM in 31% der Fälle nicht.

Unter allen untersuchten Verfahren ist Triv nach GLEAM als das beste Verfahren zu bewerten. Ziel weiterführender Untersuchungen sollte sein, diejenigen Operatoren zu identifizieren, die zum Erfolg des Verfahrens beigetragen haben. Dies entspricht - wie in Abschnitt 4.2.2 beschrieben - der Suche nach Sprachen, mit denen Indikatoren guter Lösungen beschrieben werden können. Mit diesem Wissen ist zum einen die Konstruktion neuer lokaler Suchverfahren möglich. Zum anderen können entsprechende Mutationsoperatoren in den evolutionären Algorithmus GLEAM aufgenommen werden.

Kapitel 6

Fazit und Ausblick

6.1 Zusammenfassung der Arbeit

GORBA ist eine Software für das Ressourcenmanagement im Grid. Zur effizienten Planung und Allokation von Jobs ist ein komplexes Schedulingproblem zu lösen. Das GORBA-Problem wurde mit Hilfe einer etablierten Notation formuliert. Zur Lösung des Problemes werden Vereinfachungen genutzt, hierzu zählen auch mehrere Heuristiken zur Einschränkung des Suchraumes, welche beschrieben wurden. Ähnliche Aufgabenstellungen konnten in der Literatur nicht ausgemacht werden. Das GORBA-Problem gehört der Klasse NP-vollständig an, eine entsprechende Beweisskizze wurde geliefert. Für die Bewertung von Lösungsverfahren wurden 16 Benchmarkszenarien eingeführt.

Zur Lösung des GORBA-Problems ist von einem Optimierungsverfahren eine geeignete Reihenfolge von Elementen zu finden. Hierzu wird der evolutionäre Algorithmus HyGLEAM genutzt. Die wesentlichen von HyGLEAM genutzten Mechanismen und Konzepte wurden erklärt. Hierzu zählen vor allem das Genmodell, die genetischen Operatoren und das Nachbarschaftsmodell. HyGLEAM die memetische Variante des Algorithmus GLEAM. Die Einbindung lokaler Suchverfahren sowie ein Adaptionsmechanismus zur Ansteuerung lokaler Suchverfahren wurden beschrieben. Ebenso wurde auf die Vorgehensweise zur Durchführung der Benchmarks des GORBA-Problems eingegangen. Als Referenz wurden Benchmarks mit GLEAM durchgeführt.

Der Einsatz hybrider Algorithmen, also Kombinationen von Metaheuristiken und spezifischeren lokalen Suchverfahren, wurde mit Hilfe des No Free Lunch Theorems motiviert. Als ein Werkzeug zur Charakterisierung lokaler Suchverfahren wurde das Konzept des Optimierungsmusters eingeführt. Durch die Angabe relevanter Optimierungsmuster können sowohl lokale Suchverfahren als auch Problemstellungen beschrieben werden. Durch eine entsprechende Analyse einer Optimierungsaufgabe können so gezielt lokale Suchverfahren identifiziert werden.

Zur Lösung des GORBA-Problems wurden fünf lokale Suchverfahren untersucht. Alle Verfahren wurden ausführlich motiviert und beschrieben. Die Güte der Verfahren wurde mit Hilfe der Benchmarks des GORBA-Problems bewertet. Mit dem Verfahren „Stoss“ wurde zunächst der Versuch unternommen, durch wiederholte Verschiebungen einzelner Elemente neue und bessere Lösungen zu finden. Die Ergebnisse waren jedoch durchgehend schlechter als die zuvor mit GLEAM erzielten.

Als ein neuer Ansatz wurden lokale Suchverfahren in Anlehnung an Ameisenalgorithmen entwickelt. Wissen, welches durch die Evaluation konstruierter Individuen gewonnen wurde, wird hierbei in einer Pheromonmatrix gespeichert. Mit diesem Wissen sollten zunehmends bessere Varianten der Individuen aus der Population von HyGLEAM erstellt werden. Bei der ersten Version von ANT 1 wurde hierbei auf die Nachbarschaft von Elementen in der Permutation abgezielt. Die ersten Benchmarkergebnisse des Verfahrens waren schlecht, und so wurde der Ansatz verworfen.

Zwei weitere Varianten dieses Ansatzes wurden entwickelt. Eine erweiterte Version von ANT 1 untersucht Elemente dahingehend, ob sie in guten Lösungen nahe beieinander stehen sollten oder nicht. Das lokale Suchverfahren ANT 2 untersucht die Verteilungen des Zusammenhangs zwischen der Lösungsqualität und der absoluten Position von Elementen. Diese beiden Verfahren erzielten zwar bessere Ergebnisse als die erste Version von ANT 1. Die Leistungen beider Verfahren liegen jedoch ebenfalls deutlich unter jenen von GLEAM. Der Aufwand für die Konstruktion neuer Individuen ist dabei sehr zeitaufwändig, worin ein wesentlicher Grund für den Misserfolg der Algorithmen zu sehen ist.

Mit „Triv“ wurde ein letzter Versuch unternommen, Individuen durch geeignete Manipulationen zu verbessern. Hierbei wurde ein Katalog von mehreren, vergleichsweise tri-

vialen Operatoren definiert, welche wiederholt auf Lösungen angewendet werden. Ein simpler Lernmechanismus adaptiert dabei die Wahrscheinlichkeiten, mit denen einer der Operatoren gewählt wird. Mit „Triv“ wurden die besten Benchmarkergebnisse aller lokaler Suchverfahren erzielt, die Erfolgsraten liegen nur knapp unter jenen von GLEAM.

6.2 Ausblick

Im Rahmen dieser Diplomarbeit konnte keine Verbesserung der Leistung von HyGLEAM zur Lösung des GORBA-Problemes erreicht werden. Dennoch können viele, im Laufe der Analyse des GORBA-Problems und der Implementierung gewonnenen Erkenntnisse zur Weiterentwicklung von HyGLEAM beitragen.

GORBA ist bislang noch nicht im Produktiveinsatz. Die wohl wichtigsten Hinweise, inwiefern das GORBA-Problem richtig gestellt ist und wie HyGLEAM darauf angepasst werden kann, wird der erste Einsatz von GORBA bringen. Das GORBA-Problem beruht bislang auf vielen Hypothesen über zukünftige Anwendungen und Anwender. Eine Konkretisierung der Anforderungen an GORBA sollte daher stets mit der Weiterentwicklung von GORBA einhergehen.

Die untersuchten lokalen Suchverfahren operieren in einem vereinfachten Suchraum des GORBA-Problemes. Eine anwendungsspezifische Suche im Lösungsraum und damit eine Umplanung kann viele Vorteile bringen. Beim Eintreffen neuer Jobs könnte eine Umplanung schneller sein als eine komplette Neuplanung aller Jobs. Viel wesentlicher ist der Vorteil, dass nicht bei jeder Evaluation eines Individuums alle Jobs einzuplanen sind. Die Implementierung eines derartigen Mechanismus in HyGLEAM ist allerdings als sehr aufwändig einzuschätzen.

Die genetischen Operatoren sind ein wesentlicher Erfolgsfaktor evolutionärer Algorithmen. Die begonnene Integration neuer Rekombinationsoperatoren in GLEAM zeigt bereits erste positive Ergebnisse. Die Mutationsoperatoren des Verfahrens Triv könnten ebenso in GLEAM integriert werden. Weitere Untersuchungen des Verfahrens Triv könnten die Frage beantworten, ob der Adaptionsmechanismus von HyGLEAM auch für die Auswahl der genetischen Operatoren sinnvoll einzusetzen ist. Von einer Integration dieser genetischen

Operatoren würden auch andere Anwendungen von HyGLEAM profitieren.

6.3 Dank

Mein Dank gilt Herrn Prof. Dr. Schmeck und Herrn Dr. Jakob für die Betreuung meiner Diplomarbeit. Ebenso möchte ich mich für die Unterstützung durch Frau Dr. Mostaghim, Herrn Dr. Stucky, den Teilnehmern des Diplomandenseminars am AIFB sowie der Forschungsgruppe Grid am IAI bedanken. Ebenso danke ich meiner Familie für ihre Unterstützung.

Anhang A

Testergebnisse

A.1 Benchmarks und verwendete Abkürzungen

Im Folgenden befinden sich die aufgearbeiteten Testergebnisse von GLEAM und den untersuchten lokalen Suchverfahren Stoss, Ant 1 in der zweiten Version, Ant 2 sowie Triv. Die Bezeichnungen bedeuten dabei:

Bezeichnung	Bedeutung
kR / gR	kleiner / großer Ressourcenauswahlgrad
kA / gA	kleiner / großer Abhängigkeitsgrad
GJobAnz	Anzahl der Grid Jobs des Benchmarks
Pop-Size	Populationsgröße des evolutionären Algorithmus
Zeit Schn	Durchschnittliche Zeit bis zur Konvergenz
StdA, rStdA	(relative) Standardabweichung
Gen Schn	Durchschnittliche Anzahl an Generationen
RAuswStrat	Verteilung der gewählten Allokationsheuristik
fst, che, AJ	Global Fast, Global Cheap oder Application Job spezifisch
StrFk	Anteil der Lösungen mit Straffunktionen (X: irrelevant)
95%-Ki	Breite des 95%-Konfidenzintervalls der Note

A.2 Benchmarkergebnisse für GLEAM / K1 / gRep

A.2.1 Szenarien „gR gA“

GJob Anz	Pop-Size	Erfolg Note	NSF	Zeit Schn	Note Schn	Min	Max	StdA	rStdA	Gen Schn	Indiv Schn	RAuswStrat fst che	AJ	StrFk Ue	X
50	300	98	98	2:34	41473	21819	42425	2849	6.9	1073	1662836	-AktParam--	1	0	
50	400	100	100	2:57	41923	40811	42650	371	0.9	922	1911743	-AktParam--	0	0	
50	500	100	100	3:00	41972	41201	42973	280	0.7	746	1936323	-AktParam--	0	0	
100	300	68	68	3:00	37645	22244	42722	5470	14.5	545	792833	-AktParam--	16	0	
100	400	66	66	3:00	37052	23444	42178	5563	15.0	393	790734	-AktParam--	17	0	
100	500	70	70	3:00	37227	24375	42015	5450	14.6	303	787879	-AktParam--	15	0	
100	600	58	58	3:00	35949	23536	42146	5673	15.8	251	801695	-AktParam--	21	0	
200	90	38	38	3:00	35571	28245	39581	3086	8.7	773	288507	-AktParam--	31	0	
200	120	44	44	3:00	35773	29792	39688	3205	9.0	558	285586	-AktParam--	28	0	
200	150	59	59	3:00	36789	29694	39411	2761	7.5	426	283826	-AktParam--	21	0	
200	200	54	54	3:00	36529	29908	39195	2683	7.3	295	277279	-AktParam--	23	0	
200	250	46	46	3:00	35983	29808	39116	2621	7.3	226	276601	-AktParam--	27	0	
200	300	48	48	3:00	35967	29783	38814	2622	7.3	183	276788	-AktParam--	25	0	
d200	120	30	30	3:00	35753	31580	40450	2648	7.4	711	411788	-AktParam--	35	0	
d200	150	40	40	3:00	36223	26921	40396	3104	8.6	548	402716	-AktParam--	30	0	
d200	200	32	32	3:00	35862	33766	40155	2581	7.2	396	400465	-AktParam--	34	0	
d200	250	34	34	3:00	35560	26889	39678	2842	8.0	307	396705	-AktParam--	33	0	
d200	300	32	32	3:00	35291	26953	39336	2862	8.1	250	393375	-AktParam--	34	0	

A.2.2 Szenarien „gR kA“

GJob Anz	Pop-Size	Erfolg Note	Erfolg NSF	Zeit Schn	Note Schn	Note Min	Note Max	Note StdA	Note rStdA	Gen Schn	Indiv Schn	RAuswStrat fst che	RAuswStrat AJ	StrFk Ue	StrFk X
50	400	100	100	2:49	46385	44602	48199	969	2.1	1392	2618321	-AktParam--		0	0
50	500	100	100	2:58	46797	44930	49768	988	2.1	1156	2747340	-AktParam--		0	0
50	600	100	100	2:59	46748	44716	48704	983	2.1	972	2766016	-AktParam--		0	0
100	400	90	90	3:00	41345	30329	43108	3251	7.9	591	1103152	-AktParam--		5	0
100	500	96	96	3:00	41964	32756	43244	1942	4.6	455	1099677	-AktParam--		2	0
100	600	98	98	3:00	41782	29767	42893	1784	4.3	369	1099148	-AktParam--		1	0
100	700	86	86	3:00	40533	29563	43250	3801	9.4	312	1117341	-AktParam--		7	0
100	800	90	90	3:00	40630	31181	42404	3099	7.6	268	1118423	-AktParam--		5	0
200	150	18	18	3:00	32485	30374	39647	3176	9.8	683	413213	-AktParam--		41	0
200	200	28	28	3:00	33326	30500	39522	3603	10.8	472	410212	-AktParam--		36	0
200	250	24	24	3:00	32617	26494	39042	3639	11.2	352	401464	-AktParam--		38	0
200	300	22	22	3:00	32423	29074	38839	3321	10.2	283	400970	-AktParam--		39	0
200	400	22	22	3:00	32023	26278	38417	3541	11.1	203	403223	-AktParam--		38	0
d200	120	92	92	3:00	40878	35320	41890	1615	4.0	1191	577223	-AktParam--		4	0
d200	150	96	96	3:00	40976	27308	42085	2174	5.3	901	562884	-AktParam--		2	0
d200	200	100	100	3:00	41290	40504	41902	286	0.7	630	554882	-AktParam--		0	0
d200	250	100	100	3:00	41121	40489	41715	305	0.7	484	551361	-AktParam--		0	0
d200	300	98	98	3:00	40883	35100	41448	866	2.1	391	550088	-AktParam--		1	0

A.2.3 Szenarien „kR gA“

GJob Anz	Pop-Size	Erfolg Note	Erfolg NSF	Zeit Schn	Note Schn	Note Min	Note Max	Note StdA	Note rStdA	Gen Schn	Indiv Schn	RAuswStrat fst che	RAuswStrat AJ	StrFk Ue	StrFk X
50	300	100	100	1:26	45714	44220	47012	880	1.9	781	1178367	-AktParam--	0	0	
50	400	100	100	2:05	46133	44694	47259	878	1.9	834	1696186	-AktParam--	0	0	
50	500	100	100	2:31	46102	44309	47259	903	2.0	815	2050983	-AktParam--	0	0	
100	200	100	100	2:49	47063	44016	48198	716	1.5	1059	952185	-AktParam--	0	0	
100	250	100	100	2:56	47333	45439	48306	536	1.1	878	973597	-AktParam--	0	0	
100	300	100	100	3:00	47280	45788	48086	442	0.9	720	992496	-AktParam--	0	0	
100	400	100	100	3:00	47418	46061	48387	431	0.9	521	988988	-AktParam--	0	0	
100	500	100	100	3:00	47303	46220	48524	374	0.8	407	988619	-AktParam--	0	0	
100	600	100	100	3:00	47293	46141	47840	382	0.8	334	1004404	-AktParam--	0	0	
100	700	100	100	3:00	47366	46675	49068	441	0.9	275	1006049	-AktParam--	0	0	
200	150	100	92	3:00	44852	37735	46361	1993	4.4	575	374956	-AktParam--	4	0	
200	200	100	100	3:00	45421	44422	46318	417	0.9	388	366097	-AktParam--	0	0	
200	250	100	100	3:00	45217	44431	46181	393	0.9	296	364933	-AktParam--	0	0	
200	300	100	100	3:00	44907	43582	45629	376	0.8	236	365364	-AktParam--	0	0	
200	400	96	96	3:00	44165	37681	45222	1377	3.1	168	367104	-AktParam--	2	0	
200	500	98	98	3:00	43725	37245	44621	1001	2.3	132	373585	-AktParam--	1	0	
d200	150	68	68	3:00	43202	34341	47806	4857	11.2	739	520266	-AktParam--	16	0	
d200	200	84	84	3:00	44653	35238	47575	3808	8.5	524	511899	-AktParam--	8	0	
d200	250	90	90	3:00	45037	35942	47506	2980	6.6	398	505500	-AktParam--	5	0	
d200	300	82	82	3:00	44009	33954	46982	3911	8.9	324	504338	-AktParam--	9	0	
d200	400	86	86	3:00	43713	34987	46154	3035	6.9	232	503355	-AktParam--	7	0	

A.2.4 Szenarien „kR kA“

GJob Anz	Pop-Size	Erfolg Note	Erfolg NSF	Zeit Schn	Note Schn	Note Min	Note Max	Note StdA	Note rStdA	Gen Schn	Indiv Schn	RAuswStrat fst che	RAuswStrat AJ	StrFk Ue	StrFk X
50	300	100	100	1:05	49258	48578	49967	290	0.6	838	1201296	-AktParam--	0	0	0
50	400	100	100	1:32	49263	47520	50003	386	0.8	860	1644212	-AktParam--	0	0	0
50	500	100	100	1:56	49374	48280	50003	315	0.6	896	2140169	-AktParam--	0	0	0
100	250	100	100	2:57	44110	43032	45332	522	1.2	1240	1316422	-AktParam--	0	0	0
100	300	100	100	3:00	44049	43354	45431	447	1.0	997	1321576	-AktParam--	0	0	0
100	400	100	100	3:00	44104	43108	45418	495	1.1	733	1321025	-AktParam--	0	0	0
100	500	100	100	3:00	44021	42964	45094	440	1.0	555	1308506	-AktParam--	0	0	0
200	200	96	96	3:00	40614	30502	41605	2090	5.1	527	496178	-AktParam--	2	0	0
200	250	90	90	3:00	39916	30260	41652	2968	7.4	403	497215	-AktParam--	5	0	0
200	300	98	98	3:00	40312	29662	41095	1565	3.9	331	506032	-AktParam--	1	0	0
200	400	96	96	3:00	39743	30163	40895	1688	4.2	238	510709	-AktParam--	2	0	0
200	500	100	100	3:00	39566	38916	40485	339	0.9	184	509944	-AktParam--	0	0	0
200	600	90	90	3:00	38461	29268	39874	2123	5.5	146	503909	-AktParam--	5	0	0
d200	150	86	86	3:00	41133	32346	42839	3029	7.4	1030	652229	-AktParam--	7	0	0
d200	200	80	80	2:56	40577	32046	43297	3621	8.9	719	635500	-AktParam--	10	0	0
d200	250	92	92	3:00	41411	32089	42752	2698	6.5	537	631370	-AktParam--	4	0	0
d200	300	90	90	3:00	41253	32046	42719	2674	6.5	432	632907	-AktParam--	5	0	0
d200	400	80	80	3:00	40189	31963	42583	3492	8.7	310	639305	-AktParam--	10	0	0
d200	500	92	92	3:00	40968	31618	42313	2215	5.4	238	637481	-AktParam--	4	0	0
d200	600	76	76	3:00	39151	31446	42754	3610	9.2	185	608581	-AktParam--	12	0	0
d200	700	70	70	3:00	38232	31319	42858	4188	11.0	165	636676	-AktParam--	15	0	0

A.3 Benchmarkergebnisse für GLEAM / K2 / pRep

A.3.1 Szenarien „gR gA“

GJob Anz	Pop-Size	Erfolg Note	Erfolg NSF	Zeit Schn	Note Schn	Note Min	Note Max	Note StdA	Note rStdA	Gen Schn	Indiv Schn	RAuswStrat fst	RAuswStrat che	RAuswStrat AJ	StrFk Ue	StrFk X
50	250	100	100	2:15	42156	41267	43639	364	0.9	761	874367	0	49	50	0	0
50	300	100	100	2:34	42254	41238	43730	391	0.9	723	984600	0	56	44	0	0
50	400	100	100	2:55	42317	41291	43327	334	0.8	647	1174889	0	52	48	0	0
50	500	100	100	3:00	42359	41694	43213	356	0.8	401	936536	0	48	52	0	0
100	250	59	59	2:59	35645	20348	41554	5883	16.5	478	523749	0	11	89	41	0
100	300	75	75	3:00	37599	23175	42786	4756	12.6	394	526138	0	10	90	25	0
100	400	79	79	3:00	38133	27359	41854	4295	11.3	282	514938	0	14	86	21	0
100	500	76	76	3:00	37627	24062	42136	4703	12.5	219	513961	0	15	85	24	0
100	600	70	70	3:00	36917	26526	41186	4774	12.9	178	514281	0	5	95	30	0
200	90	92	92	3:00	39098	34215	40585	1472	3.8	458	174443	0	2	98	8	0
200	120	98	98	3:00	39376	33975	40252	815	2.1	331	174781	0	3	97	2	0
200	150	97	97	3:00	39420	34466	40458	834	2.1	255	173825	0	8	92	2	0
200	200	100	100	3:00	39575	38342	40275	348	0.9	185	173677	0	5	95	0	0
200	250	99	99	3:00	39415	34860	40458	565	1.4	142	171055	0	6	94	1	0
200	300	99	99	3:00	39306	34594	40362	609	1.6	115	170953	0	6	94	1	0
200	400	99	99	3:00	39141	34018	39901	615	1.6	83	169573	0	8	92	1	0
d200	250	100	100	3:00	41798	40832	42755	393	0.9	161	198657	99	1	0	0	0
d200	300	100	100	3:00	41828	40792	42689	354	0.8	129	198392	100	0	0	0	0
d200	400	100	100	3:00	41618	40752	42780	362	0.9	94	197484	100	0	0	0	0
d200	500	100	100	3:00	41469	40787	42070	306	0.7	72	197024	100	0	0	0	0

A.3.2 Szenarien „gR kA“

GJob Anz	Pop-Size	Erfolg Note	NSF	Zeit Schn	Note Schn	Min	Max	StdA	rStdA	Gen Schn	Indiv Schn	RAuswStrat fst	che	AJ	StrFk Ue	X
50	250	100	100	1:50	48618	46709	50270	760	1.6	929	973330	50	7	43	0	0
50	300	100	100	2:10	48877	47159	50376	662	1.4	944	1160973	48	10	42	0	0
50	400	100	100	2:38	48961	47074	51095	628	1.3	849	1403754	38	4	58	0	0
50	500	100	100	2:58	49205	47401	50698	555	1.1	753	1573192	36	8	56	0	0
100	250	100	100	2:59	43777	42328	45136	638	1.5	580	574114	25	6	69	0	0
100	300	100	100	3:00	43968	41939	45411	707	1.6	470	573151	31	6	63	0	0
100	400	100	100	3:00	43880	42327	45149	585	1.3	334	570729	33	5	62	0	0
100	500	100	100	3:00	43831	42328	45265	617	1.4	257	569253	37	3	60	0	0
200	120	100	100	3:00	40187	38060	41505	515	1.3	404	194853	18	60	22	0	0
200	150	100	100	3:00	40281	38960	41701	447	1.1	307	194219	23	57	20	0	0
200	200	100	100	3:00	40269	39465	40958	348	0.9	217	192789	21	44	35	0	0
200	250	100	100	3:00	40266	39206	40997	374	0.9	164	190599	17	51	32	0	0
200	300	100	100	3:00	40208	38960	41193	404	1.0	132	189923	17	51	32	0	0
200	400	100	100	3:00	40028	38832	41182	366	0.9	94	190022	26	45	29	0	0
200	500	100	100	3:00	39865	38907	40825	336	0.8	71	188300	26	49	25	0	0
d200	120	100	100	3:00	42520	41251	43418	487	1.1	403	200093	95	0	5	0	0
d200	150	100	100	3:00	42541	41358	43865	484	1.1	310	198928	92	0	8	0	0
d200	200	100	100	3:00	42494	40834	43705	530	1.2	216	195076	94	0	6	0	0
d200	250	100	100	3:00	42437	41054	43978	522	1.2	166	193079	96	0	4	0	0
d200	300	100	100	3:00	42326	40772	43364	415	1.0	133	191393	93	0	7	0	0
d200	400	100	100	3:00	42108	40486	43140	438	1.0	96	191183	95	0	5	0	0
d200	500	100	100	3:00	42006	41055	42844	374	0.9	74	190755	91	1	8	0	0

A.3.3 Szenarien „kR gA“

GJob Anz	Pop-Size	Erfolg Note	NSF	Zeit Schn	Note Schn	Min	Max	StdA	rStdA	Gen Schn	Indiv Schn	RAuswStrat fst	che	AJ	StrFk Ue	X
50	300	100	100	1:37	45102	44669	45504	330	0.7	781	1190460	38	6	56	0	0
50	400	100	100	2:00	45124	44516	45504	306	0.7	720	1467926	38	8	54	0	0
50	500	100	100	2:39	45243	44669	45504	303	0.7	770	1945242	44	2	54	0	0
100	300	100	100	3:00	46886	45416	47811	541	1.2	632	855114	54	10	36	0	0
100	400	98	98	3:00	46901	34584	48515	1844	3.9	469	852301	52	12	36	1	0
100	500	100	100	3:00	47137	46379	48555	482	1.0	365	859175	70	6	24	0	0
100	600	100	100	3:00	47117	46357	48906	417	0.9	296	852514	52	6	42	0	0
200	200	100	98	3:00	45816	39910	47126	1018	2.2	322	300835	36	40	24	1	0
200	250	100	100	3:00	45988	44977	47320	516	1.1	244	298308	48	28	24	0	0
200	300	100	100	3:00	45821	43973	46634	492	1.1	197	296570	16	46	38	0	0
200	400	98	98	3:00	45549	38722	46492	1085	2.4	143	297267	40	24	36	1	0
d200	150	100	100	3:00	47630	46256	49466	609	1.3	583	378056	52	2	46	0	0
d200	200	100	100	3:00	47853	46310	49870	698	1.5	422	374225	62	0	38	0	0
d200	250	100	100	3:00	47926	46668	48854	517	1.1	317	367560	66	2	32	0	0
d200	300	100	100	3:00	47852	46525	49330	554	1.2	254	366226	68	2	30	0	0
d200	400	100	100	3:00	47751	46963	48488	364	0.8	183	364917	64	2	34	0	0

A.3.4 Szenarien „kR kA“

GJob Anz	Pop-Size	Erfolg Note	NSF	Zeit Schn	Note Schn	Min	Max	StdA	rStdA	Gen Schn	Indiv Schn	RAuswStrat fst	che	AJ	StrFk Ue	X
50	300	100	100	1:16	45313	44495	46151	373	0.8	849	1124339	50	16	34	0	0
50	400	100	100	1:44	45344	44659	46151	343	0.8	856	1523925	58	16	26	0	0
50	500	100	100	2:15	45399	44642	46204	384	0.8	896	1971821	58	14	28	0	0
100	300	100	100	2:59	42486	41609	43471	387	0.9	822	1009482	26	28	46	0	0
100	400	100	100	3:00	42432	41746	43806	413	1.0	588	1004796	30	20	50	0	0
100	500	100	100	3:00	42630	41815	43491	366	0.9	455	1003692	26	24	50	0	0
100	600	100	100	3:00	42567	41960	43406	349	0.8	367	1000287	14	42	44	0	0
200	90	96	96	3:00	40480	33782	41271	1349	3.3	1038	371544	62	12	26	2	0
200	120	100	100	3:00	40807	39889	41532	357	0.9	733	376501	58	18	24	0	0
200	150	100	100	3:00	40782	40058	41345	305	0.7	554	369688	56	12	32	0	0
200	200	100	100	3:00	40649	39987	41305	320	0.8	395	370131	66	10	24	0	0
200	250	100	100	3:00	40545	39880	41042	284	0.7	306	369872	60	10	30	0	0
200	300	100	100	3:00	40406	39690	40941	270	0.7	247	370730	68	18	14	0	0
d200	150	100	100	3:00	41801	41058	42613	373	0.9	599	386278	60	0	40	0	0
d200	200	100	100	3:00	41791	40856	42552	368	0.9	424	383385	70	0	30	0	0
d200	250	100	100	3:00	41780	41180	42506	307	0.7	327	382361	62	0	38	0	0
d200	300	100	100	3:00	41675	41045	42414	303	0.7	264	381119	62	0	38	0	0

A.4 Benchmarkergebnisse für Stoss / K2 / pRep

A.4.1 Szenarien „gR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Pivotregel"  0=0, 1=0.2, ...
|   |   |   |    |   d. Parameters "Verschiebungsweite"  0=0, 1=0.2, ...
|   |   |   |    |   |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |   |    erreichte Note
|   |   |   |    |   |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |   |    |     |       |    |   Evaluationen
|   |   |   |    |   |   |   |    |     |       |    |   |        Anteil "Fast"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   "Cheap"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   "Appl."
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   ÜbLä
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  X
gR_gA 50 K2 stoss 39850 ============================================================
10   98  98 3121 3.0 5.0 2.7 0:36 41176 24842   681  841  282536   0  74  26  1  0
20  100 100 5976 3.3 5.7 2.5 1:15 41737 40149   157  762  574237   0  66  34  0  0
50  100 100 127012.8 5.5 2.3 2:43 42054 41194   128  655 1262831   0  60  40  0  0
90  100 100 141242.7 5.2 3.1 3:00 42153 41243   137  397 1463506   2  40  58  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_gA 100 K2 stoss 36917 ===========================================================
10   14  14 3506 2.4 5.1 3.4 2:51 26380 16852     -  904  516429   8   2  90 43  0
20    4   4 3806 3.1 5.1 3.5 2:59 23941 14900     -  484  539282   8   6  86 48  0
50   16  16 3876 2.7 4.8 3.4 3:00 28957 18434     -  178  543155   2   6  92 42  0
90   24  24 4075 2.8 4.8 3.2 3:00 29395 17354     -   98  541808   0   8  92 38  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_gA 200 K2 stoss 37745 ===========================================================
10   54  54 1105 2.6 4.5 2.7 3:00 36071 27346     -  307  181254   0   4  96 22  0
20   64  64  976 2.6 4.8 3.1 3:00 36817 30811     -  131  182895   0   2  98 18  0
50   70  70 1039 2.8 5.1 2.9 3:01 37077 31832     -   47  182101   0   6  94 15  0
90   56  56  998 3.0 4.6 3.3 3:02 36671 31791     -   23  180354   0   0 100 18  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_gA 200d K2 stoss 37931 ==========================================================
10  100 100 1088 2.9 4.9 3.0 3:00 41052 40236   130  298  206316 100   0   0  0  0
20  100 100 1070 2.8 5.2 2.8 3:00 40991 39990   121  133  204894 100   0   0  0  0
50  100 100 1100 3.1 5.6 3.5 3:01 41075 40431    99   48  205716 100   0   0  0  0
90  100 100 1151 2.7 4.8 4.0 3:01 40780 39992   109   26  204255 100   0   0  0  0
```

A.4.2 Szenarien „gR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Pivotregel"  0=0, 1=0.2, ...
|   |   |   |    |   d. Parameters "Verschiebungsweite"  0=0, 1=0.2, ...
|   |   |   |    |   |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |   |    erreichte Note
|   |   |   |    |   |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |   |    |     |       |    |   Evaluationen
|   |   |   |    |   |   |   |    |     |       |    |   |        Anteil "Fast"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   "Cheap"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   "Appl."
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   ÜbLä
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  X
gR_kA 50 K2 stoss 40793 ============================================================
10  100 100 3690 3.8 5.6 2.1 0:43 47057 43897   270 1005  380683  10  40  50  0  0
20  100 100 7105 3.2 5.3 2.7 1:27 47266 45378   259  934  768849  28  38  34  0  0
50  100 100 128763.7 5.6 2.8 2:36 47829 46031   250  686 1403188  38  30  32  0  0
90  100 100 147583.4 5.5 2.8 2:59 48317 46656   270  431 1623075  38  18  44  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_kA 100 K2 stoss 38333 ===========================================================
10  100 100 3522 3.2 6.2 3.1 2:52 42667 39994   310 1025  568451  20  16  64  0  0
20  100 100 3649 3.2 5.3 2.3 3:00 42888 41260   221  510  597237  30  10  60  0  0
50  100 100 3483 3.6 5.1 2.6 3:00 42781 39143   254  176  593572  24  14  62  0  0
90  100 100 3445 3.3 5.6 2.9 3:00 42870 41308   215   89  593473  16  10  74  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_kA 200 K2 stoss 37289 ===========================================================
10  100 100  898 3.1 5.3 2.5 3:00 39624 37944   164  255  202973   2  68  30  0  0
20  100 100  789 3.3 5.4 2.6 3:00 39709 38074   173  105  203324   4  68  28  0  0
50  100 100  769 3.0 5.8 3.3 3:02 39824 38413   144   33  204686  10  70  20  0  0
90  100 100  759 2.9 5.0 3.0 3:01 39481 38311   133   17  202308   6  84  10  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
gR_kA 200d K2 stoss 39889 ==========================================================
10   92  92 1034 3.0 5.9 2.8 3:00 41427 35908   303  305  205020  88   2  10  1  0
20  100 100  955 3.6 5.2 2.6 3:00 41744 40030   180  128  205549  94   0   6  0  0
50  100 100  835 3.5 5.2 2.8 3:01 41588 40054   175   36  204993  94   2   4  0  0
90   96  96  807 3.0 4.8 2.8 3:01 41206 39442   181   20  204231  96   0   4  0  0
```

A.4.3 Szenarien „kR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Pivotregel"  0=0, 1=0.2, ...
|   |   |   |    |   d. Parameters "Verschiebungsweite"  0=0, 1=0.2, ...
|   |   |   |    |   |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |   |    erreichte Note
|   |   |   |    |   |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |   |    |     |       |    |   Evaluationen
|   |   |   |    |   |   |   |    |     |       |    |   |        Anteil "Fast"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   "Cheap"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   "Appl."
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   ÜbLä
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  X
kR_gA 50 K2 stoss 40386 ============================================================
 5  100 100 1101 3.1 6.3 2.8 0:09 44414 43684    87  637  119771  38  22  40  0  0
10  100 100 2221 2.9 5.9 2.6 0:19 44500 43743    96  606  245562  36  22  42  0  0
20  100 100 4288 3.3 6.5 2.5 0:39 44524 43638    75  584  489109  44  14  42  0  0
30  100 100 7021 3.5 5.4 2.6 1:05 44776 43946    91  651  810302  58   8  34  0  0
50  100 100 112893.4 5.9 2.4 1:50 44963 44283    88  610 1361971  48   6  46  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_gA 100 K2 stoss 38497 ===========================================================
 5   68  68 2342 3.2 6.5 3.0 1:25 42247 32435     - 1274  417752  48  28  24 16  0
10   68  68 3559 3.0 5.6 2.8 2:17 42361 32214     -  903  675633  46  24  30 16  0
20   80  80 4727 2.7 5.1 3.1 2:56 43849 33021  1435  576  868679  48  24  28 10  0
30   74  74 4725 2.7 5.8 3.3 3:00 43192 32043     -  384  887730  62  20  18 13  0
50   78  78 4832 3.0 5.1 3.3 3:00 43677 32756  1468  219  889213  56  14  30 11  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_gA 200 K2 stoss 39639 ===========================================================
 5   34  34 1429 2.9 5.3 3.4 2:57 38559 28029     -  825  333875  22  52  26 33  0
10   42  42 1458 2.7 5.9 3.2 3:00 39199 31245     -  406  339144  26  50  24 29  0
20   32  32 1471 2.7 5.7 3.5 3:00 38834 33156     -  182  337592  22  46  32 34  0
30   44  44 1463 2.6 5.2 2.8 3:00 39671 28281     -  118  337454  24  46  30 28  0
50   48  48 1514 3.0 5.8 3.4 3:00 40053 34006     -   66  337499  28  46  26 26  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_gA 200d K2 stoss 41238 ==========================================================
 5   98  98 1957 2.9 7.1 2.7 2:53 46322 32953   588 1120  389144  72   4  24  1  0
10  100 100 2022 2.8 6.6 3.1 3:00 46628 45232   158  554  407498  70   2  28  0  0
20  100 100 1966 3.0 5.8 3.0 3:00 46700 45508   170  257  405867  52   4  44  0  0
30  100 100 1980 3.0 5.2 2.9 3:00 46773 45615   122  165  404997  56   6  38  0  0
50  100 100 2026 3.0 5.7 3.1 3:00 46725 45499   131   93  406656  60   4  36  0  0
```

A.4.4 Szenarien „kR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Pivotregel"  0=0, 1=0.2, ...
|   |   |   |    |   d. Parameters "Verschiebungsweite"  0=0, 1=0.2, ...
|   |   |   |    |   |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |   |    erreichte Note
|   |   |   |    |   |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |   |    |     |       |    |   Evaluationen
|   |   |   |    |   |   |   |    |     |       |    |   |        Anteil "Fast"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   "Cheap"
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   "Appl."
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   ÜbLä
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  X
kR_kA 50 K2 stoss 38705 ============================================================
 5  100 100 1702 3.2 5.5 2.9 0:13 44313 42645   184  978  215803  46  24  30  0  0
10  100 100 2961 3.2 5.4 2.3 0:24 44389 42878   177  814  381840  46  22  32  0  0
20  100 100 6359 3.3 5.7 2.9 0:54 44478 43528   135  828  851122  48  24  28  0  0
30  100 100 8867 3.2 6.4 2.3 1:15 44487 43612   164  774 1177422  54  16  30  0  0
50  100 100 139872.9 5.2 2.4 1:59 44703 43189   173  750 1864832  48  26  26  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_kA 100 K2 stoss 36481 ===========================================================
 5   96  96 3508 3.2 5.5 2.8 1:59 40541 23967   824 2010  742813  36  32  32  2  0
10  100 100 4712 3.0 5.7 3.0 2:46 41176 40091   170 1350 1039487  34  30  36  0  0
20  100 100 5165 3.2 5.5 2.6 2:59 41388 40046   167  715 1118658  28  30  42  0  0
30  100 100 5079 3.1 4.7 3.0 3:00 41329 40033   180  453 1126397  32  34  34  0  0
50  100 100 5093 3.0 5.1 2.9 3:00 41567 40595   151  263 1145132  38  28  34  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_kA 200 K2 stoss 36243 ===========================================================
 5  100 100 1439 3.4 6.2 2.8 3:00 39796 39191    87  874  431665  64  14  22  0  0
10  100 100 1302 2.3 5.8 3.0 3:00 39718 39323    86  370  434649  70  14  16  0  0
20  100 100 1253 3.3 5.8 3.1 3:00 39713 39123    87  152  434358  52  10  38  0  0
30  100 100 1259 2.9 4.6 3.3 3:00 39650 38912    76   99  435494  54  18  28  0  0
50  100 100 1335 3.0 5.0 3.3 3:00 39485 38990    84   56  439918  62  18  20  0  0
|   |   |   |    |   |   |   |    |     |       |    |   |        |   |   |   |  |
kR_kA 200d K2 stoss 39156 ==========================================================
 5   90  90 1593 3.1 5.2 2.8 3:00 40098 31436   722  952  442487  58   6  36  5  0
10   90  90 1559 3.7 5.5 2.6 3:00 40020 31145   773  443  440680  66   2  32  5  0
20   98  98 1421 3.2 4.8 3.1 3:00 40712 31271   402  181  445166  68   0  32  1  0
30   96  96 1438 3.0 5.5 3.0 3:00 40543 31399   456  111  445738  78   0  22  2  0
50  100 100 1461 3.2 5.4 3.2 3:00 40652 39948    95   64  440434  52   8  40  0  0
```

A.5 Benchmarkergebnisse für ANT 1v2 / K2 / pRep

A.5.1 Szenarien „gR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_gA 50 K2 Ant1 39850 ==========================================================
20  100 100 2365 1.0 1.1 1:49 41487 40680   111  713  226707   0  48  52  0  0
30  100 100 3209 1.0 1.2 2:36 41624 40671   113  670  321106   0  38  62  0  0
50  100 100 3437 1.0 1.3 3:00 41877 40800   143  455  371408   0  38  62  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_gA 100 K2 Ant1 36917 =========================================================
20    4   4 1033 1.0 1.0 3:00 23880 12968     -  277   83487   2  10  88 48  0
30   14  14  985 1.0 1.1 3:00 25960 16642     -  179   84631   0  10  90 43  0
50   10  10  946 1.0 1.1 3:00 26141 15842     -  105   85368   2  10  88 45  0
70   12  12  923 1.1 1.1 3:00 27261 17762     -   74   86018   0  10  90 44  0
90   10  10  930 1.0 1.1 3:00 27027 17921     -   56   85658   0  12  88 45  0
```

-- Fortsetzung auf der nächsten Seite --

-- Fortsetzung der vorherigen Seite --

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_gA 200 K2 Ant1 37745 ==================================================
10   18  18  194 1.0 1.1 3:00 33502 26103     -  105   14636   0   0 100 41  0
20   26  26  121 1.1 1.0 3:03 34330 25047     -   30    9887   0   2  98 36  0
30   22  22  115 1.1 1.1 3:03 34063 28885     -   18    9802   0   0 100 38  0
50    6   6  114 1.2 1.0 3:06 32038 27937     -   11    9930   0   2  98 46  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
 5   32  32  379 1.0 1.2 6:00 32353  4326     -  436   26891   0  26  74 33  0
10   40  40  383 1.1 1.2 6:00 35334 28864     -  207   28466   0   0 100 29  0
20   26  26  369 1.1 1.1 6:00 34433 28169     -   95   28860   0   0 100 37  0
30   28  28  287 1.1 1.1 6:04 34754 26544     -   48   23464   0   2  98 36  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   44  44  564 1.1 1.2 9:00 35913 29519     -  314   41971   0   0 100 26  0
20   42  42  556 1.1 1.1 9:01 35942 29713     -  144   42953   0   2  98 27  0
30   34  34  464 1.0 1.2 9:02 35459 30167     -   78   37799   0   0 100 32  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   30  30  777 1.1 1.2   12 34973 29560     -  422   56184   0   2  98 35  0
20   46  46  746 1.1 1.1   12 36022 28939     -  196   57900   0   4  96 27  0
30   56  56  742 1.0 1.1   12 36720 33530     -  126   58684   0   0 100 22  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   52  52  937 1.0 1.2   15 35993 27266     -  520   69891   0   0 100 24  0
20   40  40  942 1.0 1.1   15 35654 30279     -  246   71932   0   2  98 30  0
30   46  46  918 1.0 1.0   15 36008 30428     -  159   72847   0   0 100 27  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_gA 200d K2 Ant1 37931 =================================================
10   90  90  191 1.0 1.1 3:00 39556 29811   649   98   14607 100   0   0  5  0
20   88  88  186 1.0 1.0 3:01 39570 33934   548   45   15198 100   0   0  6  0
30   96  96  182 1.0 1.1 3:01 39949 33798   363   29   15424 100   0   0  2  0
50   82  82  174 1.2 1.1 3:02 38762 26132   853   17   15623 100   0   0  9  0
90   72  72  173 1.0 1.1 3:01 37410 29268     -   10   15672 100   0   0 14  0
150  12  12  166 1.1 1.3 3:03 31276 26132     -    5   15464  98   2   0 44  0
```

A.5.2 Szenarien „gR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_kA 50 K2 Ant1 40793 ==================================================
30  100 100 3368 1.0 1.1 2:54 46618 44125   306  625  279534  30  18  52  0  0
50  100 100 3241 1.0 1.1 3:00 47106 45316   271  378  288172  36  14  50  0  0
90  100 100 3115 1.0 1.0 3:00 47269 45480   253  206  290588  54  10  36  0  0
150 100 100 2939 1.0 1.0 3:00 47893 45637   255  121  296157  52   8  40  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 100 K2 Ant1 38333 =================================================
50   98  98  817 1.0 1.0 3:00 41166 30472   513   83   67646  32  18  50  1  0
70  100 100  785 1.0 1.0 3:00 41528 39497   271   58   68084  24  24  52  0  0
90  100 100  791 1.1 1.1 3:01 41177 38775   251   45   69865  30  18  52  0  0
120 100 100  766 1.0 1.1 3:01 41131 38441   240   33   69961  28  30  42  0  0
150 100 100  808 1.1 1.1 3:00 41011 39382   201   27   70737  26  10  64  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 200 K2 Ant1 37289 =================================================
20   92  92  142 1.5 1.0 3:01 38311 32010   490   34   12409  12  76  12  4  0
30   94  94  143 1.3 1.1 3:02 38282 32228   346   23   12585  20  68  12  2  0
50   82  82  143 1.4 1.1 3:02 37026 27860   744   14   12698  20  72   8  8  0
90   28  28  146 1.4 1.1 3:03 33395 25859     -    8   12832  16  58  26 33  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   92  92  294 1.3 1.0 6:01 38554 32831   364  158   23322   6  92   2  2  0
20   98  98  284 1.3 1.0 6:02 38667 32499   316   73   24212  18  70  12  1  0
30   92  92  286 1.4 1.1 6:02 38347 31771   469   47   24546  20  64  16  4  0
50   96  96  280 1.3 1.1 6:04 38553 30745   361   27   25077  20  72   8  1  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   96  96  437 1.3 1.1 9:00 38893 37024   204  241   34267   6  84  10  0  0
20   92  92  431 1.2 1.1 9:01 38209 31011   562  112   35276  10  74  16  4  0
30   96  96  416 1.3 1.1 9:02 38930 32852   388   71   35935  16  70  14  2  0
50   98  98  424 1.3 1.0 9:04 39040 37286   139   41   36793  26  54  20  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 200d K2 Ant1 39889 ================================================
20   74  74  153 1.0 1.2 3:01 39220 28164     -   37   12368  66   0  34 11  0
30   68  68  149 1.0 1.0 3:02 39070 30701     -   24   12663  72   0  28 13  0
```

```
50   36  36  150 1.1 1.1 3:03 36199 27713     -   14   12864  76   0  24 28  0
```

A.5.3 Szenarien „kR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
kR_gA 50 K2 Ant1 40386 ==================================================
30  100 100 2710 1.0 1.1 2:08 44588 44130    97  576  300827  44  20  36  0  0
50  100 100 3560 1.0 1.3 2:58 44719 44283   106  462  408716  38  10  52  0  0
70  100 100 3514 1.0 1.3 3:00 44731 44283    99  334  412033  40   6  54  0  0
90  100 100 3539 1.0 1.3 3:00 44793 43946    92  261  414257  46   2  52  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 100 K2 Ant1 38497 =================================================
30   56  56  973 1.0 1.1 3:00 40574 31624     -  184   88898  50  20  30 22  0
50   70  70  847 1.0 1.2 3:00 42296 32164     -  105   90073  50  12  38 15  0
70   72  72  863 1.0 1.1 3:00 42709 33083     -   74   91098  50  20  30 14  0
90   80  80  821 1.0 1.2 3:00 43312 32908  1420   57   92047  80   4  16 10  0
120  62  48  367 5.1 3.4 3:01 39481 28946     -   10   84811  66  12  22 26  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 200 K2 Ant1 39639 =================================================
 5   10  10  195 1.0 1.0 3:00 33772 15673     -  215   14575  24  32  44 45  0
10   12  12  193 1.0 1.0 3:00 36067 28226     -   99   15304  52  20  28 44  0
20   12   6   80 5.1 2.5 3:11 34804 27266     -   16   16100  42  26  32 47  0
30   12  12  185 1.1 1.1 3:01 36107 31224     -   30   16144  42  40  18 44  0
50    4   4  187 1.0 1.2 3:02 34853 27554     -   18   16431  42  36  22 48  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 200d K2 Ant1 41238 ================================================
20  100  80   78 5.0 2.6 3:09 44975 42840   187   16   16270  64   2  34 10  0
30  100 100  189 1.1 1.0 3:01 45334 44119   172   31   16428  66   2  32  0  0
50  100 100  188 1.0 1.1 3:02 44896 43890   145   18   16773  60   2  38  0  0
70  100 100  183 1.0 1.2 3:02 44680 43522   146   13   16758  68   0  32  0  0
90   98  98  181 1.0 1.0 3:02 44119 34986   414   10   16938  72   4  24  1  0
```

A.5.4 Szenarien „kR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
kR_kA 50 K2 Ant1 38705 ==================================================
30  100 100 3063 1.0 1.2 2:44 44602 43019   170  622  296827  52  32  16  0  0
50  100 100 2984 1.0 1.2 3:00 44684 43462   140  413  324316  40  32  28  0  0
70  100 100 2920 1.0 1.1 3:00 44926 43773   134  293  324001  46  16  38  0  0
90  100 100 3045 1.0 1.2 3:00 44843 43463   143  226  327000  60  24  16  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 100 K2 Ant1 36481 =================================================
30  100 100  799 1.0 1.1 3:00 41023 39436   156  140   68531  38  22  40  0  0
50  100 100  730 1.1 1.1 3:00 40945 39821   122   80   69592  36  16  48  0  0
70  100 100  715 1.1 1.0 3:00 40893 39400   134   57   70687  32  24  44  0  0
90  100 100  733 1.0 1.1 3:00 40743 39568   133   44   71113  32  28  40  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 200 K2 Ant1 36243 =================================================
 5   58  58  149 1.1 1.0 3:00 34310 21559     -  164   11563  58   4  38 21  0
10   72  72  149 1.2 1.1 3:00 36002 23670     -   75   12190  58  10  32 14  0
30   84  84  146 1.2 1.1 3:02 36546 28208   700   23   12862  60  10  30  8  0
50   68  68  146 1.2 1.0 3:02 35011 27575     -   14   12932  50  10  40 16  0
70   50  50  147 1.3 1.1 3:03 33998 26732     -   10   13076  58   4  38 22  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 200d K2 Ant1 39156 ================================================
 5   44  44  160 1.0 1.2 3:00 36185 30513     -  174   11539  48   0  52 28  0
10   24  24  155 1.0 1.0 3:00 34360 29108     -   78   12091  34   2  64 38  0
30   28  28  154 1.0 1.0 3:02 34474 29154     -   24   12888  42   4  54 34  0
50   24  24  151 1.0 1.0 3:03 33661 28984     -   14   13065  48   6  46 38  0
70   10  10  150 1.0 1.1 3:02 32143 28870     -   10   13041  38   4  58 45  0
```

A.6 Benchmarkergebnisse für ANT 2 / K2 / pRep

A.6.1 Szenarien „gR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_gA 50 K2 Ant2 39850 ==================================================
20  100 100  140 6.1 1.7 2:57 41194 40481   120  313  320481   0  64  36  0  0
30  100 100  148 6.0 2.1 3:00 41311 40460   128  184  322575   0  62  38  0  0
50  100 100  186 6.1 2.5 3:02 41295 40447   126   84  326559   0  56  44  0  0
70  100 100  203 6.1 2.9 3:02 41369 40581    95   73  329205   0  48  52  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_gA 100 K2 Ant2 36917 =================================================
 5    6   6   21 3.2 1.2 2:54 20821 11736     -  779   71748   2  14  84 47  0
10    2   2   24 3.6 1.5 3:00 20542 10268     -  322   75452   2  12  86 49  0
20    4   4   27 4.1 1.8 3:01 21168 11432     -  142   76636   0  10  90 48  0
30    2   2   36 4.3 2.2 3:02 20468  9418     -   73   75825   2  14  84 49  0
50    2   2   45 4.1 2.5 3:03 19663  9125     -   46   77441   2  16  82 49  0
70    2   2   56 4.9 2.7 3:06 17391  9536     -   23   77308   0  22  78 49  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_gA 200 K2 Ant2 37745 =================================================
 5    8   8   13 3.1 1.6 3:00 28632  5984     -  148   13315   0  12  88 46  0
10   18  18   16 3.3 1.9 3:02 32337 25963     -   61   13602   0   2  98 41  0
20   12  12   18 3.0 2.3 3:09 31618 23630     -   17    8886   0   0 100 44  0
30   10  10   20 3.9 2.4 3:20 29653 19307     -    9    9452   0   8  92 45  0
50    0   0   28 4.0 2.7 3:34 26529 20800     -    5    9964   0   2  98 50  0
90    0   0   37 4.5 2.9 3:22 23329 19307     -    3    9145   2   8  90 50  0
150   0   0   49 4.3 3.1 3:15 22173 19307     -    2    9221   6  20  74 50  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
 5   22  22   20 3.3 1.5 6:00 31730  5447     -  278   26328   2   6  92 38  0
10   18  18   21 3.3 1.5 6:01 33626 26820     -  135   26463   0   2  98 40  0
20   22  22   27 3.7 2.0 6:05 33492 22373     -   51   27250   0   0 100 39  0
30   16  16   28 4.1 2.0 6:14 33048 22923     -   31   27251   0   0 100 42  0
```

-- Fortsetzung auf der nächsten Seite --

```
               -- Fortsetzung der vorherigen Seite  --
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_gA 200 K2 Ant2 37745 ==================================================
 5   34  34   23 3.8 1.4 8:59 32692  2896     -  395   39110   2  10  88 32  0
10   24  24   24 3.6 1.6 9:03 34303 26194     -  194   39708   0   0 100 38  0
20   22  22   32 3.7 2.0 9:09 34273 27500     -   75   40444   0   2  98 39  0
30   12  12   30 3.8 2.3 9:18 32940 25204     -   27   25254   0   2  98 44  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
 5   32  32   24 3.8 1.4 12   33581 12926     -  534   52932   2  18  80 32  0
10   30  30   29 3.6 1.3 12   34876 26074     -  266   54262   0   0 100 34  0
20   26  26   36 4.5 2.0 12   34215 26160     -   94   54824   0   0 100 37  0
30   32  32   38 4.3 2.1 12   34987 23814     -   57   54825   0   2  98 33  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
 5   16  16   27 3.9 1.2 15   32414  6864     -  649   65143   0  10  90 42  0
10   32  32   35 4.0 1.4 15   35310 29335     -  304   67293   0   4  96 30  0
20   50  50   43 4.1 1.8 15   35813 28146     -  117   67804   0   0 100 25  0
30   28  28   43 4.3 2.2 15   34252 26288     -   69   68332   0   2  98 36  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_gA 200d K2 Ant2 37931 ==================================================
 5   64  64   19 3.6 1.7 3:00 37346 21366     -  132   13584 100   0   0 18  0
10   78  78   22 3.4 1.7 3:01 38696 33507   709   60   14027 100   0   0 11  0
20   78  78   22 3.7 2.2 3:04 38203 30871   892   21   13575 100   0   0 11  0
30   58  58   25 4.0 2.3 3:08 36471 26223     -   13   13921 100   0   0 21  0
50   38  38   35 4.0 2.8 3:11 33951 26132     -    7   13848  98   2   0 30  0
90   18  18   50 4.0 3.4 3:09 30554 26132     -    4   13709 100   0   0 41  0
```

A.6.2 Szenarien „gR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
gR_kA 50 K2 Ant2 40793 ==================================================
30  100 100  162 5.9 1.9 3:01 46303 44179   284  176  238687  40  16  44  0  0
50  100 100  193 5.5 2.5 3:02 46194 43062   364   75  236810  34  28  38  0  0
90  100 100  246 6.0 2.6 3:03 46345 43249   362   38  237651  62  14  24  0  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 100 K2 Ant2 38333 =================================================
 5   88  88   27 4.0 1.4 2:56 39420 11951  1674  557   54524  26  24  50  6  0
10   80  80   34 4.3 1.7 3:00 38997 28246  1350  206   56740  24  38  38 10  0
20   86  86   42 4.2 2.1 3:01 39593 28032  1124   86   55382  32  30  38  7  0
30   84  84   48 4.0 2.1 3:02 39528 29352  1131   62   56303  30  30  40  8  0
50   74  74   57 4.5 2.6 3:04 38017 20800     -   27   55368  30  18  52 13  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 200 K2 Ant2 37289 =================================================
 5   68  68   16 3.2 1.4 3:01 36106  3513     -  122   10559  16  76   8 12  0
10   72  72   16 3.3 1.9 3:03 36797 29594     -   41   10688  16  72  12 12  0
20   70  70   18 3.3 2.0 3:04 36233 27094     -   19   11040  16  68  16 15  0
30   58  58   21 3.8 2.3 3:09 34805 23917     -   12   11209  26  58  16 21  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
 5   80  80   22 3.5 1.5 6:00 37627 30195   674  219   21192  16  68  16  8  0
10   92  92   23 3.4 1.5 6:02 38100 32157   423   96   21707   4  90   6  3  0
20   84  84   24 4.1 1.9 6:10 37634 27074   827   36   22236  20  68  12  6  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
10   84  84   30 3.7 1.6 9:02 37734 31266   673  141   32077  10  82   8  8  0
20   80  80   31 4.1 2.1 9:14 37359 27524   952   51   32777  10  78  12  8  0
30   86  86   34 4.2 2.5 9:18 37480 25733   838   32   33117  12  74  14  7  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
gR_kA 200d K2 Ant2 39889 ================================================
 5   50  50   13 3.0 1.5 3:00 36392 24612     -  116   10405  50   2  48 21  0
10   50  50   14 3.1 1.7 3:02 36895 26352     -   51   10744  46   0  54 23  0
20   36  36   18 3.4 2.4 3:08 34798 24243     -   17   10873  76   0  24 29  0
```

A.6.3 Szenarien „kR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
kR_gA 50 K2 Ant2 40386 ==================================================
30  100 100  356 5.7 2.4 3:00 44471 43494    89  157  352838  40  24  36  0  0
50  100 100  383 5.9 2.3 3:01 44484 44089    76   75  354401  44  12  44  0  0
70  100 100  396 5.8 3.6 3:04 44531 43472    98   43  357033  48   8  44  0  0
90  100  98  468 5.7 3.9 3:05 44462 44068    62   32  356663  42  12  46  1  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 100 K2 Ant2 38497 =================================================
 5   32  32   51 4.5 1.5 2:58 36199 22542     -  602   77125  50  26  24 34  0
10   40  40   56 4.9 1.9 3:01 37562 30987     -  241   78308  42  32  26 30  0
20   32  32   68 5.3 2.3 3:02 36633 28212     -  109   91712  40  42  18 34  0
30   28  28   66 5.2 2.7 3:04 35961 28946     -   55   79688  58  30  12 36  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 200 K2 Ant2 39639 =================================================
 5    8   8   17 3.6 1.6 3:01 33598 18871     -  131   13670  24  42  34 46  0
10    8   8   20 3.3 1.4 3:01 34303 27021     -   62   14144  42  24  34 46  0
20    8   8   24 3.9 2.2 3:04 34528 27095     -   29   17739  40  40  20 46  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_gA 200d K2 Ant2 41238 ================================================
 5   92  92   21 3.5 1.6 3:01 43689 24858  1086  133   14085  60   8  32  4  0
10  100 100   21 3.3 2.0 3:02 44733 41626   279   54   14340  48   4  48  0  0
20   98  98   25 3.9 2.3 3:07 44358 37320   371   29   18375  66   6  28  1  0
30   96  96   28 4.0 2.5 3:11 43500 30219   653   13   14831  70   2  28  2  0
```

A.6.4 Szenarien „kR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. Parameters "Anzahl der Ameisen"  0=5, 1=10, ...
|   |   |   |    |   Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |   |    erreichte Note
|   |   |   |    |   |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |   |    |     |       |    |    Evaluationen
|   |   |   |    |   |   |    |     |       |    |    |        Anteil "Fast"
|   |   |   |    |   |   |    |     |       |    |    |        |  "Cheap"
|   |   |   |    |   |   |    |     |       |    |    |        |  |   "Appl."
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   ÜbLä
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  X
kR_kA 50 K2 Ant2 38705 ==================================================
 5  100 100  133 5.4 1.3 2:15 44122 42741   156  748  191768  46  32  22  0  0
10  100 100  147 5.8 1.5 2:38 44037 42075   197  473  223977  48  28  24  0  0
20   98  98  194 5.6 2.0 3:00 43866 26687   725  211  277403  42  40  18  1  0
30   98  98  196 5.9 2.2 3:01 43922 27436   693  151  255093  42  30  28  1  0
50  100 100  233 5.8 2.6 3:02 44182 43282   157   65  253189  64  20  16  0  0
70  100 100  254 5.9 3.4 3:04 43916 42340   182   46  255627  58  22  20  0  0
90  100 100  292 6.0 3.4 3:03 43824 42326   199   29  254939  58  22  20  0  0
120 100  96 1066 5.0 1.5 3:01 44748 43870   135   35  295620  46  24  30  2  0
150 100  92 1059 5.2 1.9 3:01 44634 42420   196   26  294050  46  22  32  4  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 100 K2 Ant2 36481 =================================================
10   98  98   54 5.0 1.6 3:01 40212 30256   457  162   54485  30  42  28  1  0
20   98  98   57 5.0 1.7 3:02 40252 30325   459   83   63263  36  26  38  1  0
30  100 100   64 4.9 2.0 3:05 40280 38910   208   49   56300  28  26  46  0  0
50   98  98   71 4.7 2.3 3:06 39898 30262   460   29   57067  36  18  46  1  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 200 K2 Ant2 36243 =================================================
10   72  72  149 1.2 1.1 3:00 36002 23670     -   75   12190  58  10  32 14  0
20   64  64   23 3.7 1.9 3:04 34570 24400     -   22   13340  60  16  24 18  0
30   84  84  146 1.2 1.1 3:02 36546 28208   700   23   12862  60  10  30  8  0
50   68  68  146 1.2 1.0 3:02 35011 27575     -   14   12932  50  10  40 16  0
|   |   |   |    |   |   |    |     |       |    |    |        |  |   |   |  |
kR_kA 200d K2 Ant2 39156 ================================================
 5   34  34   17 3.0 1.6 3:00 34010  7410     -  108   10370  24   6  70 33  0
10   10  10   16 3.4 2.2 3:03 31989 28491     -   41   10591  20   4  76 45  0
20   18  18   22 3.3 2.4 3:05 32898 28472     -   24   13448  36   4  60 41  0
30   12  12   23 3.7 2.4 3:20 31728 28462     -   12   11710  30   8  62 43  0
```

A.7 Benchmarkergebnisse für TRIV / K2 / pRep

A.7.1 Szenarien „gR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |    erreichte Note
|   |   |   |    |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |    |     |       |    |       Evaluationen
|   |   |   |    |   |    |     |       |    |       |     Anteil "Fast"
|   |   |   |    |   |    |     |       |    |       |     |  "Cheap"
|   |   |   |    |   |    |     |       |    |       |     |  |   "Appl."
|   |   |   |    |   |    |     |       |    |       |     |  |   |   ÜbLä
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  X
gR_gA 50 K2 stoss 39850 ================================================
10  100  94  197 5.0 0:54 42036 41074   101  907     530   0  68  32  3  0
20  100  96  272 5.0 1:33 42084 41223   113  738     433   0  60  40  2  0
50  100  98  501 5.0 2:57 42239 41217   118  560     272   0  48  52  1  0
90  100  96  537 5.0 3:00 42430 41819    99  321     278   6  42  52  2  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_gA 100 K2 stoss 36917 ===============================================
10   28  26  356 5.0 2:05 30435 16849     - 1046     363   4  12  84 37  0
20   32  32  359 5.0 2:56 31180 17500     -  644     262   0  26  74 34  0
50   40  38  311 5.0 3:00 32801 21369     -  246     298   4  12  84 31  0
90   60  54  209 5.0 3:00 35864 22580     -  124     258   0  10  90 23  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_gA 200 K2 stoss 37745 ===============================================
10   94  88  242 5.0 3:00 39152 34133   374  599     420   0   4  96  6  0
20   94  84  162 5.0 3:00 39258 33111   414  248     303   0  14  86  8  0
50  100  92   78 5.0 3:00 39601 38538    86   79     348   0  18  82  4  0
90  100  84   54 5.0 3:00 39488 38595    97   41     362   0   8  92  8  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_gA 200d K2 stoss 37931 ==============================================
10  100  92  274 5.0 2:59 41608 40736   137  678     379 100   0   0  4  0
20  100  94  179 5.0 3:00 41923 41022   134  281     367 100   0   0  3  0
50  100  86   85 5.0 3:00 42029 41172   107   92     341 100   0   0  7  0
90  100  88   52 5.0 3:00 41928 41223    93   47     275 100   0   0  6  0
```

A.7.2 Szenarien „gR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |    erreichte Note
|   |   |   |    |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |    |     |       |    |       Evaluationen
|   |   |   |    |   |    |     |       |    |       |     Anteil "Fast"
|   |   |   |    |   |    |     |       |    |       |     |  "Cheap"
|   |   |   |    |   |    |     |       |    |       |     |  |   "Appl."
|   |   |   |    |   |    |     |       |    |       |     |  |   |   ÜbLä
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  X
gR_kA 50 K2 stoss 40793 ================================================
10  100  98  273 5.0 0:34 47637 45682   267  819     332  30  26  44  1  0
20  100 100  481 5.0 1:16 48134 46481   240  822     378  30  24  46  0  0
50  100  98  991 5.0 2:34 48772 47076   248  677     311  38  14  48  1  0
90  100  98 1093 5.0 3:00 49237 47575   163  433     306  56   2  42  1  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_kA 100 K2 stoss 38333 ===============================================
10  100  94  632 5.0 2:06 43607 41384   218 1408     310  32  10  58  3  0
20  100  90  786 5.0 2:52 43816 42325   196  904     339  48  16  36  5  0
50  100  92  544 5.0 3:00 44097 42073   231  310     327  42  12  46  4  0
90  100  88  360 5.0 3:00 44218 42457   185  147     317  20  10  70  6  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_kA 200 K2 stoss 37289 ===============================================
10  100  96  347 5.0 2:59 40227 38570   166  716     318   6  72  22  2  0
20  100  88  254 5.0 3:00 40435 38924   142  299     357  26  54  20  6  0
50  100  88  114 5.0 3:00 40710 39315   125   87     317  28  48  24  6  0
90  100  78   68 5.0 3:00 40428 39036   150   43     350  24  34  42 11  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
gR_kA 200d K2 stoss 39889 ==============================================
10  100  92  348 5.0 3:00 42411 41256   193  728     411  76   0  24  4  0
20  100 100  227 5.0 3:00 42630 41333   152  288     275  80   0  20  0  0
50  100  86  101 5.0 3:00 42855 41831   133   87     301  96   0   4  7  0
90  100  74   68 5.0 3:00 42566 41388   130   44     325  98   0   2 13  0
```

A.7.3 Szenarien „kR gA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |    erreichte Note
|   |   |   |    |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |    |     |       |    |       Evaluationen
|   |   |   |    |   |    |     |       |    |       |     Anteil "Fast"
|   |   |   |    |   |    |     |       |    |       |     |  "Cheap"
|   |   |   |    |   |    |     |       |    |       |     |  |   "Appl."
|   |   |   |    |   |    |     |       |    |       |     |  |   |   ÜbLä
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  X
kR_gA 50 K2 stoss 40386 ================================================
10  100  96   67 4.8 0:26 44679 44283    92  652    2719  32  18  50  2  0
20  100 100  100 5.0 1:01 44822 44283    91  694     668  52  12  36  0  0
50  100 100  259 5.0 2:13 45026 44283    94  611     247  40   8  52  0  0
90  100  98  348 5.0 3:00 45219 44700    85  460     253  42   0  58  1  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_gA 100 K2 stoss 38497 ===============================================
10   82  80  255 5.0 1:28 44342 33086  1401 1045     404  40  26  34 10  0
20   82  78  313 5.0 2:24 44437 32836  1449  777     278  48  20  32 11  0
50   94  92  375 5.0 3:00 45916 33091   907  384     276  64  16  20  4  0
90  100  96  311 5.0 3:00 47006 45238   144  203     269  66   6  28  2  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_gA 200 K2 stoss 39639 ===============================================
10   58  58  324 5.0 2:54 41862 35548     -  913     322  38  24  38 21  0
20   86  80  277 5.0 3:00 44793 36527   806  431     343  32  26  42 10  0
50   88  78  127 5.0 3:00 44926 36188   861  135     308  34  20  46 11  0
90   92  78   75 5.0 3:00 45148 36116   645   70     316  38  28  34 11  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_gA 200d K2 stoss 41238 ==============================================
10  100 100  347 5.0 2:31 47130 45283   274  993     389  64   2  34  0  0
20  100  98  358 5.0 2:59 47383 45664   181  551     367  66   6  28  1  0
50  100 100  209 5.0 3:00 47876 46737   177  184     285  58   2  40  0  0
90  100  92  122 5.0 3:00 47813 46758   165   91     261  60   2  38  4  0
```

A.7.4 Szenarien „kR kA“

```
Populationsgröße
|   Erfolgsrate erreichte Mindestnote
|   |   Erfolgsrate Straffreiheit
|   |   |   Anzahl der Adaptionen
|   |   |   |    Level d. W'keit von "LSV für ganze Gen." 0=0, 1=0,2...
|   |   |   |    |   Zeit bis zur Konvergenz (Max. 3:00 Minuten)
|   |   |   |    |   |    erreichte Note
|   |   |   |    |   |    |     Minimalwert erreichte Note
|   |   |   |    |   |    |     |       Breite des 95%-KI der Note
|   |   |   |    |   |    |     |       |    Anzahl der Generationen
|   |   |   |    |   |    |     |       |    |       Evaluationen
|   |   |   |    |   |    |     |       |    |       |     Anteil "Fast"
|   |   |   |    |   |    |     |       |    |       |     |  "Cheap"
|   |   |   |    |   |    |     |       |    |       |     |  |   "Appl."
|   |   |   |    |   |    |     |       |    |       |     |  |   |   ÜbLä
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  X
kR_kA 50 K2 stoss 38705 ================================================
10  100  96  181 5.0 0:22 44988 44264   108  837     699  50  24  26  2  0
20  100 100  308 5.0 0:47 45043 44186   112  822     354  42  24  34  0  0
50  100 100  692 5.0 1:58 45228 44435   108  797     301  46  20  34  0  0
90  100 100 1006 5.0 2:51 45364 44703    84  640     286  56  14  30  0  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_kA 100 K2 stoss 36481 ===============================================
10  100 100  605 5.0 1:18 41972 40959   158 1531     571  40  28  32  0  0
20  100 100  791 5.0 2:24 42249 41270   143 1180     383  30  26  44  0  0
50  100  96  792 5.0 3:00 42460 41481   111  537     314  30  28  42  2  0
90  100 100  603 5.0 3:00 42551 41839   102  271     320  22  30  48  0  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_kA 200 K2 stoss 36243 ===============================================
10   92  86  613 5.0 2:59 40122 30223   618 1380     367  54  22  24  7  0
20   94  88  392 5.0 3:00 40439 31078   549  552     329  50  12  38  6  0
50  100  84  155 5.0 3:00 40886 40365    83  163     335  48  14  38  8  0
90  100  80   97 5.0 3:00 40490 40017    77   84     342  50  16  34 10  0
|   |   |   |    |   |    |     |       |    |       |     |  |   |   |  |
kR_kA 200d K2 stoss 39156 ==============================================
10   96  90  633 5.0 2:54 41455 31704   482 1426     385  52   4  44  5  0
20   98  96  461 5.0 3:00 41737 31908   421  611     343  60   4  36  2  0
50  100  80  203 5.0 3:00 42056 41190   129  179     323  58   4  38 10  0
90  100  72  119 5.0 3:00 41845 41064    99   90     301  58   2  40 14  0
```

Literaturverzeichnis

[AL97] Aarts, E. ; Lenstra, J.K.: *Local Search in Combinatorial Optimization*. John Wiley and Sons Ltd, 1997

[BBHS00] Bauer, A. ; Bullnheimer, B. ; Hartl, R. ; Strauss, C.: Minimizing total tardiness on a single machine using ant colony optimization. In: *Central European Journal of Operations Research* 8 (2000), Nr. 2, S. 125–141

[BDT99] Bonabeau, E. ; Dorigo, M. ; Theraulaz, G.: *Swarm Intelligence: From Natural to Artificial Systems*. Oxford, UK : Oxford University Press, 1999

[Bey01] Beyer, H.-G.: *The theory of evolution strategies*. Springer, 2001

[BJ02] Blume, C. ; Jakob, W.: GLEAM - An Evolutionary Algorithm for Planning and Control Based on Evolution Strategy. In: *GECCO 2002: Late Breaking papers at the Genetic and Evolutionary Computation Conference 2002*, AAAI, 2002, S. 31–38

[Blu90] Blume, C.: GLEAM - A System for Simulated 'Intuitive Learning'. In: Männer, R. (Hrsg.) ; Schwefel, H.-P. (Hrsg.): *Parallel Problem Solving from Nature I*. Berlin : Springer, 1990, S. 48–54

[Blu02] Blume, C.: ACO Applied to Group Shop Scheduling: A Case Study on Intensification and Diversification. In: *ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms*. London, UK : Springer-Verlag, 2002, S. 14–27

[Blu04] Blume, Christian: An Ant Colony Optimization Algorithm for Shop Scheduling Problems. In: *Journal of Mathematical Modelling and Algorithms* 3 (2004), Nr. 3, S. 285–308

[BMK96] Bierwirth, C. ; Mattfeld, D. ; Kopfer, H.: On permutation representations for scheduling problems. In: Voigt, et a. H.-M. (Hrsg.): *Parallel Problem Solving from Nature IV*. Berlin : Springer, 1996, S. 310—-318

[Box65] Box, M.J.: A New Method of Constrained Optimization and a Comparison with Other Methods. In: *The Computer Journal* 8 (1965), Nr. 1, S. 42–52

[Bru04] Brucker, P.: *Scheduling Algorithms*. Springer, 2004

[Bru06] Brucker, P: *Complex scheduling*. Springer, 2006

[BW93] Blanton, J. L. ; Wainwright, R. L.: Multiple vehicle routing with time and capacity constraints using genetic algorithms. In: Forrest, S. (Hrsg.): *Proceedings of the Fifth International Conference on Genetic Algorithms*. San Mateo, California : Morgan Kaufmann, 1993, S. 452—459

[Dar59] Darwin, C.: *The Origin of Species*. John Murray, 1859

[Dav92] Davis, L.: *Handbook Of Genetic Algorithms*. New York : Van Nostrand Reinhold, 1992

[DeJ06] DeJong, K.A.: *Evolutionary computation : a unified approach*. Cambridge, Mass. : MIT Press, 2006

[DM02] Daniel Merkle, Martin M.: Ant Colony Optimization with the Relative Pheromone Evaluation Method. In: *3rd European Workshop on Scheduling and Timetabling and 3rd European Workshop on Evolutionary Methods for AI Planning (EvoSTIM/EvoPLAN-2002)* Bd. 2279, Springer, 2002 (LNCS), S. 325–333

[DS04] Dorigo, M. ; Stützle, T.: *Ant Colony Optimization*. Cambridge, USA : MIT Press, 2004

[ES03] Eiben, A.E. ; Smith, J.E.: *Introduction to evolutionary computing*. Berlin : Springer, 2003

[Fis05] Fischer, Layna: *Workflow Handbook*. The Workflow Management Coalition, 2005

[FKNT02] Foster, I. ; Kesselman, C. ; Nick, J.M. ; Tuecke, S.: Grid Services for Distributed System Integration. In: *IEEE Computer Journal* 35 (2002), Nr. 6, S. 37–46

[FKT01] Foster, I. ; Kesselman, C. ; Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organisations. In: *Intern. Journal of Supercomputer Applications* 15 (2001), Nr. 3, S. 200–222

[Fos99] Foster, I.: *The grid: blueprint for a new computing infrastructure*. Morgan Kaufmann Publ., 1999

[Fos02] Foster, I.: What Is The Grid? - A Three Point Checklist. In: *GRIDtoday* 1 (2002), Nr. 6

[FOW66] Fogel, L.J. ; Owens, A.J. ; Walsh, M.J.: *Artificial Intelligence through Simulated Evolution*. New York : John Wiley & Sons, 1966

[FS01] Forman, E.H. ; Selly, M.A.: *Decision by objectives : How To Convince Others That You Are Right*. River Edge, USA : World Scientific, 2001

[GD79] Garey, M.R. ; D.S., Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979

[Gol04] Goldberg, D.E.: *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 2004

[GS90] Gorges-Schleuter, M.: *Genetic Algorithms and Population Structures - A Massively Parallel Algorithm*. Dormund, University of Dortmund, Computer Science Department, Diss., August 1990

[HKKS03] Hovestadt, M. ; Kao, O. ; Keller, A. ; Streit, A.: Scheduling in HPC Resource Management Systems: Queuing vs. Planning. In: Feitelson, D. G. (Hrsg.) ; Rudolph, L. (Hrsg.) ; Schwiegelshohn, U. (Hrsg.): *Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP)* Bd. 2862. Berlin / Heidelberg : Springer, 2003 (LNCS), S. 1–20

[Hol75] Holland, J.H.: *Adaptation in Natural and Artificial Systems.* Cambridge : University of Michigan Press, 1975

[HS05] Hoos, H. H. ; Stützle, T.: *Stochastic Local Search, Foundations and Applications.* Elsevier, 2005

[HSS04] Halstenberg, S. ; Stucky, K.-U. ; Süss, W.: A Grid Environment for Simulation and Optimization and a First Implementation of a Biomedical Application. In: Meersman, R. (Hrsg.) ; Tari, Z. (Hrsg.) ; Corsaro, A. (Hrsg.): *On the Move to Meaningful Internet Systems OTM 2004: Workshops* Bd. 3292. Agia Napa, Cyprus : Springer, October 2004 (LNCS), S. 59–67

[Jak02] Jakob, W.: HyGLEAM - An Approach to Generally Applicable Hybridization of Evolutionary Algorithms. In: Adamidis, P. (Hrsg.) ; Beyer, H.-G. (Hrsg.) ; Fernandez-Villacanas, J.-L. (Hrsg.) ; Guervós, J.J.M. (Hrsg.) ; Schwefel, H.-P. (Hrsg.): *Parallel Problem Solving from Nature VII.* Berlin : Springer, 2002, S. 527–536

[Jak04a] Jakob, W.: HyGLEAM: Hybrid GeneraL-purpose Evolutionary Algorithm and Method. In: *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference, Part I*, Springer, 2004, S. 187–192

[Jak04b] Jakob, W.: *Eine neue Methodik zur Erhöhung der Leistungsfähigkeit Evolutionärer Algorithmen durch die Integration lokaler Suchverfahren.* Karlsruhe, Forschungszentrum Karlsruhe, Institut für Angewandte Informatik, Diss., März 2004. – Wissenschaftliche Berichte, FZKA 6965

[Jak05] Jakob, W.: Eine neue Methode zur Leistungssteigerung Evolutionärer Algorithmen. In: *at-Automatisierungstechnik* 53 (2005), Nr. 6

[Jak06] Jakob, W.: Towards an Adaptive Multimeme Algorithm for Parameter Optimisation Suiting the Engineers' Needs. In: Runarsson, T.P. (Hrsg.) ; Beyer, H.-G. (Hrsg.) ; Burke, E. (Hrsg.) ; Merelo-Guervos, J.J. (Hrsg.) ; Whitley, L.D. (Hrsg.) ; Yao, X. (Hrsg.): *Parallel Problem Solving from Nature IX*. Berlin : Springer, 2006, S. 132–141

[JBB04] Jakob, W. ; Blume, C. ; Bretthauer, G.: Towards a Generally Applicable Self-Adapting Hybridization of Evolutionary Algorithms. In: *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference, Part I*, Springer, 2004, S. 790–791

[JGSB92] Jakob, W. ; Gorges-Schleuter, M. ; Blume, C.: Application of Genetic Algorithms to Task Planning and Learning. In: *Parallel Problem Solving from Nature II*, Elsevier, 1992, S. 293–302

[JQSS05] Jakob, W. ; Quinte, A. ; Stucky, K.-U. ; Süss, W.: Optimised Scheduling of Grid Resources Using Hybrid Evolutionary Algorithms. In: Wyrzykowski, R. (Hrsg.) ; Dongarra, J. (Hrsg.) ; Meyer, N. (Hrsg.) ; Wasniewski, J. (Hrsg.): *Parallel Processing and Applied Mathematics PPAM 2005* Bd. 3911. Poznan, Poland : Springer, September 2005 (LNCS), S. 406–413

[Kos03] Kost, B.: *Optimierung mit Evolutionsstrategien*. Deutsch, 2003

[KT05] Kleinberg, J. ; Tardos, E.: *Algorithm Design*. Pearson / Addison-Wesley, 2005

[LK73] Lin, s. ; Kernighan, B.W.: An Effective Heuristic Algorithm for the Traveling-Salesman Problem. In: *Operations Research* 21 (1973), April, Nr. 2, S. 498–516

[Man99] Manola, F.: *Characterizing Computer-Related Grid Concepts*. Website, 1999. – http://www.objs.com/agility/tech-reports/9903-grid-report-fm.html

[Mic96] Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3. Berlin : Springer Verlag, 1996

[MM00] Merkle, D. ; Middendorf, M.: An Ant Algorithm with a New Pheromone Evaluation Rule for Total Tardiness Problems. In: *EvoWorkshops*, 2000, S. 287–296

[MM03] Merkle, D. ; Middendorf, M.: Ant Colony Optimization with Global Pheromone Evaluation for Scheduling a Single Machine. In: *Applied Intelligence* 18 (2003), January, Nr. 1, S. 105–111

[MM05] Merkle, D. ; Middendorf, M.: On solving permutation scheduling problems with ant colony optimization. In: *Intern. J. Syst. Sci.* 36 (2005), Nr. 5, S. 255–266

[MR92] Milgrom, P. ; Roberts, J.: *Economics, Organization and Management*. Prentice Hall, 1992

[NSW04] Nabrzyski, J. ; Schopf, J.M. ; Weglarz, J.: *Grid Resource Management: State of the Art and Future Trends*. Kluwer Academic Publ., 2004

[Par02] Pardalos, P.M.: *Handbook of applied optimization*. Oxford University Press, 2002

[Pin95] Pinedo, M.: *Scheduling - Theory, Algorithms, and Systems*. Prentice Hall, 1995

[Poh00] Pohlheim, H.: *Evolutionäre Algorithmen*. Springer, 2000

[PS98] Papadimitriou, Christos H. ; Steiglitz, Kenneth: *Combinatorial optimization*. Corr., unabridged republ. of 1982. Dover Publ., 1998

[Rec94] Rechenberg, I.: *Evolutionsstrategie '94*. Stuttgart : Frommann-Holzboog, 1994

[Ros60] Rosenbrock, H.H.: An automatic method for finding the greatest or least value of a function. In: *The Computer Journal* 3 (1960), Nr. 3, S. 175–184

[Sch95] Schwefel, H.-P.: *Evolution and optimum seeking.* Wiley, 1995

[Sch05] Schwindt, C.: *Resource allocation in project management.* Springer, 2005

[SJQS05] Süss, W. ; Jakob, W. ; Quinte, A. ; Stucky, K.-U.: GORBA: A Global Optimising Resource Broker Embedded in a Grid Resource Management System. In: Zheng, S.Q. (Hrsg.): *Parallel and Distributed Computing Systems 2005.* Phoenix : ACTA Press, November 2005, S. 137–145

[SJQS06a] Süss, W. ; Jakob, W. ; Quinte, A. ; Stucky, K.-U.: Resource Brokering in Grid Environments using Evolutionary Algorithms. In: Fahringer, T. (Hrsg.): *Parallel and Distributed Computing and Networks 2006.* Innsbruck, Austria : ACTA Press, february 2006, S. 94–104

[SJQS06b] Stucky, K.-U. ; Jakob, W. ; Quinte, A. ; Süss, W.: Solving Scheduling Problems in Grid Resource Management Using an Evolutionary Algorithm. In: Meersman, R. (Hrsg.) ; Tari, Z. (Hrsg.): *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops* Bd. 4276. Montpellier, France : Springer, November 2006 (LNCS), S. 1252–1262

[SMM00] Schmeck, H. ; Merkle, D. ; Middendorf, M.: Ant Colony Optimization for Resource-Constrained Project Scheduling. In: Whitley, D. (Hrsg.) ; Goldberg, D. (Hrsg.) ; Cantu-Paz, E. (Hrsg.) ; Spector, L. (Hrsg.) ; Parmee, I. (Hrsg.) ; Beyer, H.-G. (Hrsg.): *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000).* Las Vegas, Nevada, USA : Morgan Kaufmann, 2000, S. 893–900

[War62] Warshall, S.: A Theorem on Boolean Matrices. In: *J. ACM* 9 (1962), Nr. 1, S. 11–12

[WH04] Watanabe, K. ; Hashem, M.M.A.: *Evolutionary computations.* Springer, 2004

[WM97] Wolpert, D.H. ; Macready, W.G.: No Free Lunch Theorems for Optimization. In: *IEEE Transactions On Evolutionary Computation* 1 (1997), April, Nr. 1, S. 67–82

Erklärung

Ich versichere hiermit wahrheitsgemäß, die Arbeit bis auf die dem Aufgabesteller bereits bekannte Hilfe selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, 31. März 2007, Björn Hahnenkamp