Über die Analyse randomisierter Suchheuristiken und den Entwurf spezialisierter Algorithmen im Bereich der kombinatorischen Optimierung

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Universität Dortmund

am Fachbereich Informatik

von

Carsten Witt

Dortmund

2004

Tag der mündlichen Prüfung:	6. Dezember 2004
Dekan:	Prof. Dr. Bernhard Steffen
Gutachter:	Prof. Dr. Berthold Vöcking, Prof. Dr. Ingo Wegener

Danksagung

Viele Personen haben direkt oder indirekt zum Entstehen dieser Dissertation beigetragen. Mein besonderer Dank gebührt Ingo Wegener für seine Betreuung und Zusammenarbeit. Darüber hinaus danke ich meinen Kollegen Stefan Droste, Oliver Giel, Jens Jägersküpper, Thomas Jansen und Tobias Storch, die mit zahlreichen Diskussionen und konstruktiver Kritik meine Forschung über randomisierte Suchheuristiken außerdem begleitet haben. Auch allen ehemaligen Mitarbeiterinnen und Mitarbeitern des Lehrstuhls für effiziente Algorithmen und Komplexitätstheorie, die mich in den vergangenen Jahren fachkundig beraten haben, gilt mein Dank.
Allen Mitarbeiterinnen und Mitarbeitern des Lehrstuhls danke ich für die freundliche und motivierende Arbeitsatmosphäre.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation und Überblick

Der Entwurf und die Analyse von Algorithmen zählen zu den grundlegenden Aufgaben im Bereich der Informatik. Algorithmen können einerseits als Verfahren konzipiert sein, die Entscheidungsfragen beantworten, andererseits versteht man sie vor allem als Verfahren zur Optimierung. Ein wichtiger Zweig der Informatik und diskreten Mathematik untersucht Grundlagen und Verfahren der *kombinatorischen Optimierung.* In allgemeiner Form dürfen wir darunter die Aufgabenstellung, eine Funktion $f\colon S \to \mathbb{R}$ mit einem diskreten, d. h. höchstens abzählbar unendlichen Definitionsbereich S und reellwertigem Bildbereich zu maximieren oder zu minimieren, verstehen (vgl. Papadimitriou und Steiglitz, 1982). Mit dem Begriff „kombinatorisch“ grenzen wir uns also von noch allgemeineren Szenarien, beispielsweise der Optimierung in reellwertigen Definitionsbereichen, ab. Auch die multikriterielle Optimierung, d. h. die Ermittlung von speziell gearteten Optima für Funktionen mit mehrdimensionalem Bildbereich, fällt nicht in diesen Rahmen.

Aus einem praktischen Blickwinkel fällt es nicht schwer, die Beschäftigung mit kombinatorischen Optimierungsproblemen zu motivieren. In der Gestalt von Lastverteilungsproblemen, Rundreiseproblemen, Teambildungsproblemen u. v. a. m. begegnen uns täglich zahlreiche Optimierungsaufgaben, die sich häufig in der Form von Modellen und idealisierten Beschreibungen nicht nur in Lehrbüchern der theoretischen Informatik wieder finden. Mit dem Hinweis auf die zumindest subjektiv empfundene Endlichkeit unserer Umwelt erklärt sich auch, weshalb kombinatorischen Optimierungsproblemen auf endlichen, aber von vornherein nicht größenbeschränkten Definitionsbereichen die größte Bedeutung beigemessen wird. Häufig setzt sich der nun endliche Definitionsbereich S von f aus dem kartesischen Produkt vieler kleiner Mengen zusammen und die Suche nach einem optimalen Element in S darf, wie auch der Begriff „kombinatorisch“ suggeriert, als geeignete Kombination der Belegung vieler Variablen gedeutet werden. Im äußersten Fall zerfällt S in das Produkt von binären Mengen, sodass uns das Szenario der Optimierung einer *pseudobooleschen Funktion* $f\colon \{0,1\}^n \to \mathbb{R}$ vorliegt. Die Optimierung pseudoboolescher Funktionen wird einer der dominierenden Grundgedanken dieser Dissertation sein. Zumindest grundsätzlich können wir jedes endliche kombinatorische Optimierungsproblem geeignet in eine pseudoboolesche Funktion verwandeln. Aus praktischer Sicht weitaus wichtiger ist jedoch der Umstand, dass wir ebendiese Transformation vornehmen, wenn wir Optimierungsprobleme mithilfe derzeit gebräuchlicher Rechner lösen wollen.

Wie gelangen wir aber zu Algorithmen, mit deren Hilfe wir ein in der Praxis vorliegendes Optimierungsproblem hoffentlich effizient behandeln können? Wenn wir Glück haben, ist das Problem gut verstanden, entspricht einer theoretischen Problembeschreibung, die wir bereits kennen, und es existieren effiziente Algorithmen, mit denen sich eine optimale Lösung schnell finden lässt. In diesem Fall müssen wir zwar immer noch zugestehen, dass wir möglicherweise nicht vernachlässigbare Fehler machen, indem wir das vorliegende Problem in eine theoretische Modellschablone pressen, aber befinden uns trotzdem in einer der besten denkbaren Situationen. Im Folgenden blenden wir diesen Aspekt der Modellbildung und die Frage, inwieweit ein Modell unsere Realität geeignet beschreibt, als eine eher philosophische Frage konsequent aus.

Oftmals sehen wir uns allerdings praktisch relevanten Problemen gegenüber, denen aus theoretischer Sicht das Stigma der NP-Vollständigkeit anhaftet. In solchen Fällen dürfen wir immer noch hoffen, dass Algorithmen verfügbar sind, die in akzeptabler Zeit zwar nicht unbedingt eine optimale, doch garantiert eine fast optimale Lösung liefern. Dieser Aspekt der Approximation wird ein zweiter Kerngedanke sein, auf dem viele Untersuchungen dieser Dissertation fußen. Der Entwurf und die Analyse effizienter Approximationsalgorithmen stellen eine der wichtigsten Antworten auf die Konsequenzen der NP-Vollständigkeit dar.

Häufig dürfen wir uns aber ganz und gar nicht glücklich schätzen und haben ein kombinatorisches Optimierungsproblem vorliegen, das wir weder gut verstehen noch unmittelbar mit bekannten Problemen in Verbindung bringen können. In einer solchen Situation würde eine Theoretikerin oder ein Theoretiker das Problem gerne geeignet modellieren, analysieren und effiziente (Approximations)algorithmen entwickeln. Doch diesen Weg kann man aus mindestens zwei Gründen nur im Idealfall einschlagen. Möglicherweise fehlt es in der Realität an wichtigen Ressourcen wie Geld, Zeit und Wissen, um das Problem auf diese Weise aufzuarbeiten. Vielleicht scheitert eine theoretische Modellbildung aber auch daran, dass die Struktur des Optimierungsproblems inhärent verborgen bleibt. Wenn die Qualität einer Lösung lediglich anhand des Ergebnisses eines Simulationslaufs oder eines Experiments, dessen Parameter oder Variablen wir geeignet setzen müssen, bestimmt ist, bleibt die der Qualität entsprechende, zu optimierende Funktion $f\colon S \to \mathbb{R}$ eine *Black Box*, deren Arbeitsweise im schlimmsten Fall undurchdringlich bleibt. In der Realität müssen wir oft diese Sichtweise, die wir Black-Box-Szenario nennen, einnehmen.

Aus vielfältigen Gründen müssen wir uns bei Optimierungsaufgaben also häufig mit Ansätzen zufrieden geben, von denen wir uns erhoffen, dass sie oft „schnell“ ablaufen und „gute“ Lösungen präsentieren. Ohne dass eine genaue Definition nötig wäre, bezeichnet man derartige Ansätze gemeinhin als *Heuristiken.* Manche Heuristiken, beispielsweise Branch-and-Bound-Ansätze, zeichnen sich dadurch aus, dass sie einerseits stets nach endlicher Zeit optimale Lösungen errechnen, andererseits aber grundsätzlich keine Garantie für eine effiziente Laufzeit besteht. Wir wenden solche Heuristiken trotzdem an, wenn sie auf typischen Probleminstanzen effizient erscheinen, und geben uns mit nicht optimalen Lösungen zufrieden, wenn das Verfahren zu lange dauert. Andere

Arten von Heuristiken, z. B. Verfahren der lokalen Suche, versprechen weder eine effiziente Laufzeit noch eine Mindestqualität der Lösung. Solange es aber keine beweisbar besseren und zudem auf praktischen Instanzen überlegenen Algorithmen gibt, greift man auch auf solche Heuristiken zurück. Sofern wir nicht mit dem Black-Box-Szenario konfrontiert sind, werden Heuristiken typischerweise regen Gebrauch vom vorhandenen Wissen über die vorliegende Problemstruktur machen und sollten deshalb nicht mit planlos vorgehenden Probieralgorithmen gleichgesetzt werden.

Wir sollten also bestrebt sein, gegebene Heuristiken theoretisch zu untersuchen, um zu verstehen, wie sie typischerweise ablaufen, wann und warum sie effizient sind und welche Annahmen über die Problemklassen, für die sie geeignet sind, beim Entwurf der Heuristiken implizit getroffen wurden. Solche Analysen wirken oft zurück auf ein besseres Verständnis der Optimierungsprobleme, die man mit einer Heuristik angeht. Besonders erfreulich gestalten sich solche Analysen aber vor allem dann, wenn es gelingt, Laufzeiten und Qualitätsgarantien für Heuristiken zu beweisen. Damit werten wir eine gegebene Heuristik für bestimmte Problemklassen eventuell zu einem effizienten Approximationsalgorithmus mit einer nicht trivialen Gütegarantie oder sogar zu einem effizienten exakten Algorithmus auf. Nichtsdestoweniger helfen oftmals auch negative Ergebnisse zu erkennen, wann und wieso eine Heuristik nicht effizient ist, und im Zuge dessen zu neuen Ideen und zu Verbesserungen zu gelangen. Wir sehen hier, wie der Entwurf und die Analyse effizienter Algorithmen einander befruchten und bedingen.

In dieser Dissertation werden wir uns zu einem überwiegenden Teil der Analyse spezieller Heuristiken verschreiben. Als besonders erfolgreich und populär gelten vor allem bei Praktiker(inne)n die so genannten *evolutionären Algorithmen.* Insbesondere in Abschwächungen des Black-Box-Szenarios greift man zwar mit wenig oder keiner Kenntnis der Problemstruktur, doch mit intuitiven Annahmen darüber auf evolutionäre Algorithmen zurück und erwartet sich davon vielversprechende Lösungen. Den Hintergrund und die Historie evolutionärer Algorithmen werden wir an geeigneter Stelle umreißen und erwähnen hier nur, dass sich evolutionäre Algorithmen die biologische Evolution als Optimierungsprozess zum Vorbild nehmen. Aus der Perspektive der Informatikerin oder des Informatikers viel spannender ist der Einsatz von Randomisierung in evolutionären Algorithmen. Wir wollen damit nicht behaupten, dass evolutionäre Algorithmen die einzigen Heuristiken seien, die sich zufälliger Entscheidungen bei ihrer Suche nach optimalen Lösungen bedienen. Nichtsdestotrotz bilden evolutionäre Algorithmen die vielleicht wichtigste Klasse randomisierter Heuristiken. Dem Umstand, dass wir mit den allgemein ausgelegten Heuristiken meist nach guten oder optimalen Punkten im Definitionsbereich der zu optimierenden Funktion $f\colon S \to \mathbb{R}$ „suchen", also ein Suchproblem vorliegt (vgl. Wegener, 2003), zollen wir Tribut, indem wir fortan von randomisierten Suchheuristiken reden und S als Suchraum bezeichnen. Einen ganz wesentlichen Aspekt dieser Dissertation darf man daher mit der Analyse randomisierter Suchheuristiken, speziell evolutionärer Algorithmen, überschreiben. Wir versprechen uns davon, die Theorie über die Analyse und auch den Entwurf von Algorithmen zu bereichern.

Diese Dissertation ist dreigeteilt angelegt. Im ersten Teil konzentrieren wir uns auf einfache randomisierte Suchheuristiken und evolutionäre Algorithmen und analysieren sie vor allem in Hinblick auf ihre Effizienz und Eignung zur Approximation. Charakteristisch für den Ablauf dieser Suchheuristiken ist, dass sie zu jedem Zeitpunkt nur einen Suchpunkt, d. i. ein Element des Suchraums S, verwalten. Dem gegenüber stehen so genannte *populationsbasierte* Suchheuristiken, bei denen die Bezeichnung „evolutionäre Algorithmen“ noch angemessener erscheint als bei den Suchheuristiken aus dem ersten Teil. Hier ist die theoretische Fundierung noch nicht so weit vorangeschritten wie bei den einfachen Suchheuristiken.

Im zweiten Teil der Dissertation widmen wir uns somit ausführlich der Analyse und auch Aspekten des Entwurfs populationsbasierter evolutionärer Algorithmen. In beiden Teilen behandeln wir als ganz zentrales Thema die Laufzeitanalyse randomisierter Suchheuristiken und legen dabei großes Gewicht auf die Vorstellung und Weiterentwicklung von Analysemethodiken.

In einem dritten Teil lösen wir uns schließlich von den in den ersten beiden Teilen vorherrschenden Aspekten der Analyse und der Randomisierung und schlagen eine Brücke zum Entwurf von Algorithmen, welcher untrennbar mit ihrer Analyse verzahnt ist. Anhand eines konkreten kombinatorischen Optimierungsproblems stellen wir dabei dar, wie man ausgehend von einfachen Heuristiken und Approximationsalgorithmen schließlich zu problemspezifischen exakten Algorithmen und verbesserten Approximationsalgorithmen gelangt, die sich, da wir fortwährend ihre Analyse im Auge behalten haben, schlussendlich alle als effizient erweisen. Anders als bei den zuvor betrachteten randomisierten Suchheuristiken geht hier die Theorie über Algorithmen also deren Anwendung voraus. Somit beleuchten wir letztendlich aus zwei entgegengesetzten, nichtsdestoweniger jeweils wohl motivierten Standpunkten, für wie unabdingbar wir eine fundierte Theorie halten.

Der Umfang dieser Dissertation wird mit der Gliederung in drei Teile besser beherrschbar. Diese Aufteilung profitiert aber andererseits von einer möglichst abgeschlossenen Darstellung der einzelnen Teile. Somit leiten wir jeden der drei Teile der Arbeit mit einem eigenen Kapitel ein. Dieses dient jeweils vor allem der Motivation des jeweiligen Themengebiets, der Aufbereitung von Vorarbeiten und des historischen Hintergrunds, der Übersicht und nicht zuletzt der Einordnung in den Gesamtkontext dieser Dissertation. Darüber hinaus schaffen wir mit dem einleitenden Kapitel jeweils einen Ausgangspunkt für die nachfolgenden Analysen und Rechnungen, indem wir grundlegende Definitionen und Methoden vorstellen. Aufgrund der verwandten Thematik kommen wir aber nicht umhin, insbesondere im zweiten Teil der Dissertation auf Methoden, die im ersten Teil vorgestellt werden, zurückzuverweisen.

Viele der in dieser Dissertation vorgestellten Analysen greifen auf verschiedene mathematische Hilfsmittel zurück. Während wir versuchen, neue und beispielsweise auf randomisierte Suchheuristiken zugeschnittene Analysemethoden direkt im Kontext ihrer Anwendung vorzustellen, gibt es einige Beziehungen insbesondere aus dem Bereich der Wahrscheinlichkeitstheorie, die wir regelmäßig anwenden und im Grunde als bekannt voraussetzen. Um die typischen und nützlichen Abschätzungen, Identitäten usw. aber an einer Stelle zu sammeln, besitzt diese Arbeit einen Anhang, der genau dieses leisten soll. Dort finden sich auch Hinweise auf die Literatur zur Mathematik und Wahrscheinlichkeitstheorie, an der sich unsere Notationen und Begriffe orientieren. An den Anhang schließt sich außerdem ein kurzes Symbolverzeichnis an. Die Leserin und der Leser sollten hier Erläuterungen für mathematische Symbole und Notationen, die eventuell nicht allgemein gebräuchlich sind, finden.

1.2 Überblick über die Veröffentlichungen

Diese Dissertation basiert auf den folgenden Arbeiten, die wir in der Reihenfolge ihrer Verwendung auflisten.

1. Wegener, I. und Witt, C. (2005b): On the optimization of monotone polynomials by simple randomized search heuristics, *Combinatorics, Probability and Computing*, im Druck; erweiterte Fassung von:

   Wegener, I. und Witt, C. (2003): On the optimization of monotone polynomials by the (1+1) EA and randomized local search, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03)*, Band 2723 von *Lecture Notes in Computer Science*, S. 622–633, Springer, ausgezeichnet mit einem *best paper award*

   Die Resultate der erstgenannten Version sind in Kapitel 3 dargestellt.

2. Witt, C. (2005b): Worst-case and average-case approximations by simple randomized search heuristics, in: *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS '05)*, Band 3404 von *Lecture Notes in Computer Science*, Springer, erscheint

   Die Ergebnisse dieser Arbeit sind Gegenstand von Kapitel 4.

3. Witt, C. (2003): Population size vs. runtime of a simple EA, in: *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, Band 3, S. 1996–2003, IEEE Press

   Kapitel 6 basiert auf der folgenden, erweiterten Version.

   Witt, C. (2005a): Population size vs. runtime of a simple evolutionary algorithm, *Theoretical Computer Science*, überarbeitete Version eingereicht

4. Witt, C. (2004): An analysis of the (μ+1) EA on simple pseudo-boolean functions, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 761–773, Springer, ausgezeichnet mit einem *best paper award*

Laut einer Ankündigung auf der GECCO-Konferenz soll der Verfasser eine erweiterte Version dieser Arbeit in der Zeitschrift *Evolutionary Computation* veröffentlichen dürfen. Kapitel 7 hat die erstgenannte Fassung zum Gegenstand.

5. Albers, S. und Witt, C. (2001): Minimizing stall time in single and parallel disk systems using multicommodity network flows, in: *Proceedings of the 4th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX '01)*, Band 2129 von *Lecture Notes in Computer Science*, S. 12–23, Springer, zu finden in Kapitel 9

1.3 Eigenanteil des Doktoranden

An gemeinsam erstellten Arbeiten war der Verfasser dieser Dissertation jeweils zur Hälfte beteiligt. Sowohl bei Arbeit 1 als auch bei Arbeit 5 aus der obigen Liste gab der Koautor bzw. die Koautorin den Anstoß zur Beschäftigung mit dem jeweiligen Thema. Andererseits erarbeitete der Doktorand bei beiden Arbeiten ebenfalls weiterführende Ideen. Auch bei der Ausarbeitung der mathematischen Beweise leisteten beide Autoren jeweils in etwa denselben Beitrag.

Vermutlich keine der Arbeiten wäre ohne die finanzielle Unterstützung vonseiten der Deutschen Forschungsgemeinschaft (DFG) möglich gewesen. Daher sei an dieser Stelle dafür ausdrücklich gedankt. Die ersten vier der genannten Arbeiten entstanden im Rahmen des Sonderforschungsbereiches „Design und Management komplexer technischer Systeme mit Methoden der Computational Intelligence“ (SFB 531) und die letzte Arbeit erfuhr Unterstützung aufgrund des DFG-Forschungsantrags „Effiziente Algorithmen für aktuelle Cachingprobleme“.

Teil I

Zur Analyse einfacher randomisierter Suchheuristiken

2 Evolutionäre Algorithmen und andere randomisierte Suchheuristiken

Dieses Kapitel dient als Grundlage speziell für die ersten beiden Teile dieser Dissertation. In Abschnitt 2.1 werden wir daher den Hintergrund der Verwendung von randomisierten Suchheuristiken und insbesondere evolutionären Algorithmen beleuchten. Im sich anschließenden Abschnitt 2.2 werden wir verdeutlichen, welche Ziele wir mit einer Analyse randomisierter Suchheuristiken verfolgen und unseren Ansatz gegenüber andersartigen Zugängen und Analysemethoden abgrenzen. In Abschnitt 2.3 schließlich legen wir den Grundstein für unsere Analysen. Neben fundamentalen Definitionen geben wir einen Überblick über Beweismethoden, die in der beschriebenen Form oder Abwandlungen bei der Analyse randomisierter Suchheuristiken konsequent wiederkehren werden. Somit dient der Abschnitt auch zur Referenz.

2.1 Hintergrund

Evolutionäre Algorithmen (EAs) sind randomisierte Suchheuristiken, deren Erfindung maßgeblich von der Vorstellung, die biologische Evolution nachzubilden, motiviert war. Die Ausführungen dieses Abschnitts sollen sicherlich nicht dazu dienen, einen allumfassenden Überblick über die kaum überschaubare Vielfalt von Klassen und Konzepten von EAs zu schaffen. Für solche Übersichten verweisen wir lieber auf Arbeiten, die genau dieses leisten wollen, beispielsweise Bäck (1996) und Bäck, Fogel und Michalewicz (1997). Gleichwohl müssen wir die wichtigsten Paradigmen evolutionärer Algorithmen vorstellen, um uns einerseits in die zunächst vielleicht ungewohnte Begriffswelt einzufinden und um uns andererseits später auf abstrakter, theoretischer Ebene mit einigen dieser Konzepte auseinander setzen und Analysen durchführen zu können.

Spätestens in den 60er-Jahren des vergangenen Jahrhunderts kam ausgehend von anwendungsspezifischen Problemen erstmals der Wunsch auf, ein Optimierungsverfahren zu entwerfen, das nach Prinzipien der biologischen Evolution vorgeht. Die Beweggründe bestanden allgemein gesagt darin, dass die biologische Evolution als besonders effizientes Optimierungsverfahren aufgefasst wurde. Weitgehend unabhängig voneinander entstanden daraus an verschiedenen Stellen die heutzutage noch gängigen Paradigmen evolutionärer Algorithmen, u. a. genetische Algorithmen (Goldberg, 1989), Evolutionsstrategien (Rechenberg, 1994; Schwefel, 1995), evolutionäre Programmierung (Fogel, 1995) und die etwas jüngere genetische Programmierung (Banzhaf, Nordin, Keller und Francone, 1998). Auf Besonderheiten einzelner dieser Grundsätze werden wir an gegebener Stelle noch eingehen. Hier verbleiben wir mit der Bemerkung, dass sich in der letzten Zeit die Bezeichnung „evolutionäre Algorithmen" oder *evolutionary computation* als Oberbegriff für alle genannten Paradigmen herauskristallisiert hat. An genauen Einordnungen der von uns betrachteten EAs sind wir auch gar nicht interessiert. Stattdessen werden wir darstellen, wie eine randomisierte Suchheuristik, die in die Klasse der evolutionären Algorithmen fällt, im Allgemeinen abläuft und die zugehörigen, zumeist von der Biologie inspirierten Nomenklaturen darstellen. Auch dieser Umriss kann nur die gebräuchlichsten Varianten darstellen.

Evolutionäre Algorithmen sollen im zugrunde liegenden Suchraum S anhand einer so genannten *Fitnessfunktion* $f\colon S \to V$, die wir meist nur *Zielfunktion* nennen, suchen. Wir beschränken uns bei unseren Analysen auf den Fall, dass $V \subseteq \mathbb{R}$ gilt, und nehmen o. B. d. A. an, dass ein Suchpunkt mit maximalem f-Wert zu finden sei. Dieses Szenario bezeichnet man häufig als monokriterielle Optimierung. In manchen Klassen von EAs sieht man S durchaus als stetigen Suchraum, z. B. $S = \mathbb{R}^n$, vor; im Rahmen dieser Dissertation hingegen betrachten wir bei Analysen, wie in der Einleitung erwähnt, durchweg den Fall der kombinatorischen Optimierung und den Suchraum $\{0,1\}^n$. Letzterer repräsentiert übrigens den Standard bei den oben erwähnten genetischen Algorithmen. Nur am Rande erwähnt sei, dass zuweilen zum Zwecke der Repräsentation von Suchpunkten noch eine dritte Menge, der so genannte Genotypraum, ins Spiel kommt. Wir ignorieren diesen Aspekt und gehen davon aus, dass ein EA stets Punkte aus dem Suchraum S verwaltet.

Die Menge von Suchpunkten, die ein EA zu einem Zeitpunkt vorhält, heißt *Population.* Die Größe dieser Population stellt die eigentliche Trennlinie zwischen dem Inhalt von Teil I und Teil II dieser Dissertation dar. Populationen der Größe 1 sind ebenso denkbar wie Populationen, deren Größe im Laufe des EAs variiert. Die zu Beginn des Laufs eines EAs vorliegende Population wird gemäß irgendeiner Regel initialisiert, wofür man oft in Ermangelung besseren Wissens eine zufällige und gleichverteilte Wahl aus dem Suchraum vorsieht. Anschließend läuft der typische EA in Runden ab, die meist *Generationen* heißen. Zu Beginn einer Generation wählt man aus der aktuellen Population eine Menge von Suchpunkten (dafür ist der Begriff *Individuen* geläufig) aus, auf Grundlage deren die Population der nächsten Generation erstellt werden soll. Das Verfahren, gemäß dem diese Individuen gezogen werden und in das häufig die f-Werte der Individuen einfließen, heißt *Selektion zur Reproduktion.* Die gewählten Individuen werden dann so genannten *Variationen* unterworfen, spezieller der *Mutation* und *Rekombination.* Hier fließt insbesondere der Aspekt der Randomisierung ein. Eine Mutation bildet ein Individuum auf ein anderes ab, eine Rekombination (auch *crossover* genannt) zieht zwei Individuen heran, um ein neues zu erzeugen. Zum Abschluss einer Generation muss noch die Population, die in der nächsten Generation aktuell sein soll, bestimmt werden. Hier gehen typischerweise die vorhandene Population, die Menge der durch Variation neu gebildeten Individuen und die f-Werte der Individuen ein. Wenn die Größe der Population, wie es häufig der Fall ist, konstant bleibt, müssen manche Individuen durch neue ersetzt werden, sodass die Regel zur Bestimmung der Nachfolgepopulation häufig *Selektion zur Ersetzung* heißt. Die Zahl der Runden, bis dass ein evolutionärer Algorithmus endet, bestimmt ein *Stoppkriterium.*

Wir sehen also, dass der grobe Ablauf eines evolutionären Algorithmus einem recht einfachen Schema folgt. Dies ist sicherlich ein Grund für die Beliebtheit evolutionärer Algorithmen. Sie sind leicht und schnell zu implementieren und auch einfach in Einzelheiten zu variieren. Darüber hinaus wurden bereits zahlreiche Baukastensysteme entwickelt, siehe z. B. Briest, Brockhoff, Degener, Englert, Gunia, Heering, Jansen, Leifhelm, Plociennik, Röglin, Schweer, Sudholt, Tannenbaum und Wegener (2004), mit deren Hilfe sich Programmkodes auch komplexerer EAs in kurzer Zeit zusammenstellen und simulieren lassen. In den letzten Jahrzehnten ist eine schier unüberschaubare Menge von Publikationen angewachsen, in denen der Erfolg evolutionärer Algorithmen bei anwendungsbezogenen Optimierungsaufgaben untermauert wird. Daher ist bei Ingenieur(inn)en auch ein enormes „praktisches Wissen“ vorhanden, also beispielsweise intuitives Verständnis davon, welche Klassen von EAs sich für welche Problemklassen gut eignen und welche Indizien es dafür gebe. Auf einen formalen Beweis der Effizienz ihres EAs können sich Anwender(innen) aber in den seltensten Fällen berufen. Auch wenn man traditionelle Ansätze einer theoretischen Manifestation des Erfolgs evolutionärer Algorithmen nicht verschweigen sollte, sind diese Ansätze aus unserer Sicht oft unbefriedigend. Daher werden wir uns im folgenden Abschnitt davon auch in Teilen abgrenzen.

Nicht zuletzt wegen dieser mangelnden theoretischen Fundierung ihrer Effizienz standen evolutionäre Algorithmen und verwandte Suchheuristiken im Ruf, die schwarzen Schafe in der Familie der Algorithmen darzustellen. Unser Ziel ist es, die evolutionären Algorithmen in ein anderes Licht zu rücken, indem wir sie unter den Mantelbegriff der randomisierten Suchheuristiken fassen und letztendlich Methoden zur Analyse randomisierter Algorithmen einsetzen.

2.2 Theoretische Erkenntnisse und theoretische Analyse

Schon recht bald nach der Präsentation der frühesten evolutionären Algorithmen kam der Versuch auf, das Verhalten spezieller EAs theoretisch zu analysieren, um damit ihre Arbeitsweise zu verstehen und ihre Effizienz zu untermauern. Bereits die ersten Buchveröffentlichungen über EAs (Rechenberg, 1973; Holland, 1975) enthielten daher Überschlagsrechnungen, um Anhaltspunkte für die Wahl spezieller Parameter zu geben, und Aussagen über lokale Fortschrittsmaße. Zu Letzteren zählt das im Zusammenhang mit einer Theorie über EAs immer wieder genannte Schema-Theorem (Holland, 1975). Weitere lokale Fortschrittsmaße beziehen sich auf den so genannten Qualitätsgewinn, d. i. die (erwartete) Verbesserung des Wertes der Zielfunktion in einer Generation, oder auch die so genannte Fortschrittsrate, die (erwartete) Verringerung des Wertes eines Distanzmaßes zu einem optimalen Punkt (für einen Überblick siehe Beyer und Schwefel, 2002; Beyer, Schwefel und Wegener, 2002). Aussagen über solche lokalen Maße lassen sich nur in Abhängigkeit eines aktuellen Suchpunkts treffen. Rückschlüsse auf Aussagen über globale Maße, beispielsweise die Zahl der Schritte, bis ein optimaler Punkt gefunden ist, lassen sich im Allgemeinen nicht daraus ziehen. Deshalb messen wir isolierten Analysen lokaler Maße keine große Bedeutung bei.

Dem Wunsch nach einer globalen Sichtweise werden wir gerecht, indem wir einen gegebenen EA als stochastischen Prozess auffassen. Wegen des oben erwähnten Ablaufs in Runden gelingt in den allermeisten Fällen auch eine genauere Charakterisierung, nämlich als Markoffprozess oder Markoffkette. In diesem Zusammenhang wird häufig der Aspekt der Konvergenz des stochastischen Prozesses zu Populationen, die einen optimalen Suchpunkt enthalten, untersucht (Rudolph, 1997). Solche Analysen können einerseits herausfordernd sein, lassen andererseits aber den Aspekt der Effizienz außen vor. Aufgrund des diskreten Zustandsraums $\{0,1\}^n$ werden wir uns bei kommenden Analysen zumeist mit Markoffketten konfrontiert sehen, deren Konvergenz trivial einzusehen ist. Die Frage ist nun, ob ein solcher spezieller stochastischer Prozess exakten analytischen Methoden zugänglich ist, um seine Effizienz zu bestimmen. Leider ist die Antwort in den meisten Fällen negativ. Vielleicht gelingt es, die Zustandsüberführungsfunktion der Markoffkette darzustellen. Beherrschbare exakte Ausdrücke über charakteristische Maße, beispielsweise der Verteilung der Population zu einem Zeitpunkt oder auch Erwartungswerte interessanter Größen wie der Zeit bis zur Erzeugung eines optimalen Suchpunktes, lassen sich daraus aber in den seltensten Fällen ableiten. Der Grund dafür liegt nicht zuletzt darin, dass man beim Entwurf evolutionärer Algorithmen eben nicht im Sinne hatte, sie auf eine Weise zu gestalten, die sie einer Analyse zugänglich macht. Das eingangs erwähnte Zusammenspiel von Entwurf und Analyse effizienter Algorithmen stand hier weit weniger im Vordergrund als das biologische Vorbild.

Um dennoch Aussagen über den komplexen stochastischen Prozess, den ein EA induziert, treffen zu können, kann man versuchen, Vergröberungen vorzunehmen. Indem man sich unendlich große Populationen vornimmt, gelingen allgemeine Aussagen über die Dynamik des zugehörigen stochastischen Prozesses (Vose, 1999). Die Relevanz der damit erzielten, zweifellos eleganten Ergebnisse über Konvergenzeigenschaften, Fixpunkte usw. bleiben aber für praktische Belange fragwürdig, solange es nicht gelingt, sie auf den realistischen Fall einer endlichen Populationsgröße zu übertragen. An dieser Stelle findet also eine Modellbildung des evolutionären Algorithmus statt, deren Folgen nicht abgeschätzt werden können. Noch weiter geht in dieser Hinsicht der der Physik entstammende Ansatz der so genannten statistischen Mechanik. Hier betrachtet man anstelle des tatsächlichen stochastischen Prozesses möglicherweise nur noch Kenngrößen wie Momente, beispielsweise spezielle Erwartungswerte, und schreibt deren Entwicklung über die Zeit fort. Viele der daraus resultierenden Ergebnisse beruhen auf nicht nachweisbaren Grundannahmen. Nichtsdestoweniger erlauben diese Modellbildungen zweifellos einen intuitiven und oftmals eleganten Einblick in die Arbeitsweise evolutionärer Algorithmen. Eine ausführlichere Widmung der meisten der bislang genannten Ansätze und Modellbildungen liefert das Buch von Reeves und Rowe (2003). Dort finden sich auch noch weitere Stoßrichtungen für einen möglichen theoretischen Zugang zu evolutionären Algorithmen erwähnt, beispielsweise Methoden zur Charakterisierung des Suchraums. Wir gehen hier nicht zur Gänze darauf ein, da sie nicht unseren bereits implizit postulierten Vorstellungen entsprechen, damit globale, mathematisch rigoros zu beweisende Aussagen über das Verhalten eines EAs ableiten zu können.

Die oben erwähnte Vergröberung innerhalb einer Analyse evolutionärer Algorithmen erscheint in vielen Fällen unumgänglich. Trotzdem sollte es möglich sein, bei mathematischen Abschätzungen den damit eingeführten Fehler zu kontrollieren. Dieses ist die Motivation dafür, Analysen der von einem EA oder einer anderen randomisierten Suchheuristik induzierten Markoffkette in einem asymptotischen Rahmen durchzuführen. Mithin folgen wir der aus der Analyse effizienter (randomisierter) Algorithmen (siehe z. B. Motwani und Raghavan, 1995; Cormen, Leiserson, Rivest und Stein, 2001) bekannten Grundhaltung, die Effizienz eines Algorithmus in Abhängigkeit von der Eingabegröße, wenn nötig, in O-Notation, zu messen und uns dabei eventuell mit oberen und unteren Schranken zufrieden zu geben. In unserem Szenario der Optimierung pseudoboolescher Funktionen $f\colon \{0,1\}^n \to \mathbb{R}$ fungiert die Dimension des Suchraums n als natürliches Maß für die Eingabe- bzw. Problemgröße. Mathematisch bewiesene Aussagen, die für alle (großen) n gelten, besitzen Gültigkeit auch für Problemgrößen, für die heutzutage die Aussagen noch gar nicht experimentell validiert werden können.

Wir beschreiben nun etwas genauer, an welchen Maßen wir konkret interessiert sind. Wenn eine randomisierte Suchheuristik A und eine zu optimierende Zielfunktion f vorliegen, beschäftigen wir uns häufig mit der Zufallsvariablen $T_{f,A}$, welche die Zeit repräsentiert, bis A erstmalig einen Suchpunkt mit optimalem (d. h. maximalem) f-Wert erzeugt hat; über das verwendete Zeitmaß werden wir gleich sprechen. Wir nennen $T_{f,A}$ auch die *Laufzeit von A auf f* und analysieren häufig ihren Erwartungswert $\mathrm{E}(T_{f,A})$. Natürlich kann es passieren, dass dieser Erwartungswert gar nicht definiert ist, aber wir betrachten hauptsächlich Beispiele, bei denen die Existenz von $\mathrm{E}(T_{f,A})$, also wiederum ein Aspekt der Konvergenz, trivial einzusehen ist. Zuweilen verdichtet dieser Erwartungswert aber zu viele Informationen. Vielleicht genügt uns eine Aussage der Form, dass $T_{f,A}$ mit einer Wahrscheinlichkeit von 99 % einen linearen Wert annimmt, während A mit der Restwahrscheinlichkeit beliebig lange braucht und somit einen riesigen Erwartungswert $\mathrm{E}(T_{f,A})$ provoziert. Daher betrachten wir häufig die *Erfolgswahrscheinlichkeit bis zur Zeit t*, d. i. $\mathrm{Prob}(T_{f,A} \leq t)$. Auf ähnliche Weise lassen sich Zufallsvariablen für Zeiten, bis die Suchpunkte der aktuellen Population eine gewisse Approximationsqualität besitzen, definieren. Wir überschreiben das in diesem Absatz beschriebene Konzept als *rigorose Laufzeitanalyse* oder kürzer *Zeitkomplexität* randomisierter Suchheuristiken.

Der Forschungsrichtung einer Laufzeitanalyse evolutionärer Algorithmen ist in den vergangenen Jahren auf großes Interesse gestoßen. Frühe Analysen aus diesem Blickwinkel stammen von Mühlenbein (1992) und Rudolph (1997). Um einen Anfang zu machen, konzentrierte man sich zunächst auf sehr einfache, aber auch heute noch spannende evolutionäre Algorithmen und sehr einfache Zielfunktionen. Bereits Rudolph (1997) gelang es aber, mit Aussagen über die Zeitkomplexität eines einfachen EAs Vermutungen, die andere Autoren bezüglich einer von ihnen konstruierten Zielfunktion hegten, rigoros zu widerlegen. Ab 1997 wurde insbesondere in Dortmund intensiv in dem Forschungsgebiet der Zeitkomplexität evolutionärer Algorithmen gearbeitet. In den folgenden Jahren gelangen zum einen weitere Resultate, die intuitive, zumeist aus der Praxis der EAs stammende Behauptungen widerlegen oder rigoros beweisen. Zum anderen wuchsen aber auch die Erkenntnisse über das Verhalten einfacher und komplexerer EAs auf mehreren interessanten Problemklassen. In diesem Zusammenhang sind bereits zwei Dissertationen (Droste, 2000; Jansen, 2000) erschienen, in denen ein Großteil der bis zum Jahr 2000 erzielten Ergebnisse zur Zeitkomplexität evolutionärer Algorithmen ausführlich beschrieben wird. Im Großen und Ganzen bauen die in der vorliegenden Dissertation behandelten Resultate zur Zeitkomplexität randomisierter Suchheuristiken, insbesondere evolutionärer Algorithmen, auf den Ergebnissen, die in jenen Arbeiten zusammengefasst sind, auf. An gegebener Stelle werden wir näher auf entsprechende Ergebnisse, die für uns wichtig sind, eingehen.

Indem wir uns ein wenig von der strengen Fixierung der pseudobooleschen Optimierung und evolutionärer Algorithmen entfernen, eröffnen sich uns – ohne einen Anspruch auf Vollständigkeit erheben zu wollen – noch einige weitere Arbeiten, deren Thema mit der Zeitkomplexität randomisierter Suchheuristiken überschrieben werden könnte. Sasaki und Hajek (1988) studieren beispielsweise die randomisierte Suchheuristik „Simulated Annealing“, die eher nicht als evolutionärer Algorithmus angesehen wird, bezüglich ihrer Zeitkomplexität auf einem kombinatorischen Optimierungsproblem. Jägersküpper (2003) hingegen untersucht die Zeitkomplexität eines althergebrachten EAs auf einer wohl untersuchten Zielfunktion in stetigen Suchräumen und gelangt in dieser Hinsicht zu fundamental neuen Resultaten. Im Bereich der multikriteriellen Optimierung, welche zu Funktionen mit nur noch partiell geordneten Bildbereichen führt, stellen beispielsweise Giel (2003) und Laumanns, Thiele und Zitzler (2004) Untersuchungen zur Zeitkomplexität gewisser auf diesen Problembereich zugeschnittener EAs an. Wir sehen also, dass die Theorie der Zeitkomplexität randomisierter Suchheuristiken ein spannendes und lebendiges Forschungsgebiet darstellt. Mit Beiträgen zu dieser Theorie erhoffen wir uns nicht nur ein besseres Verständnis über die Arbeitsweise randomisierter Suchheuristiken, sondern auch Beiträge zur praktischen Umsetzung und zur Lehre. Eine umfassende Theorie bleibt aber nach wie vor ein Ziel, das noch in weiter Ferne liegt.

Bislang haben wir nur angerissen, welche Probleme und Problemklassen im Bereich der kombinatorischen Optimierung Ausgangspunkt unserer Laufzeitanalysen werden sollen. Dazu ist es hilfreich, den bereits in der Einleitung angedeuteten Unterschied zwischen problemspezifischen und allgemeinen Suchverfahren aufzugreifen. Im Gegensatz zu problemspezifischen Heuristiken, beispielsweise für das Rundreiseproblem TSP (traveling salesperson problem), sind die erlaubten Eingaben der von uns untersuchten Suchheuristiken zunächst grundsätzlich nicht auf spezielle pseudoboolesche Funktionen oder Funktionenklassen festgelegt. Insbesondere bei evolutionären Algorithmen ist dieser Aspekt, für den man gelegentlich auch den Begriff der Robustheit verwendet, ausdrücklich erwünscht. Dieses wird vor dem Hintergrund, dass man in der Praxis oft mit dem in der Einleitung erwähnten Black-Box-Szenario (vgl. Droste, Jansen, Tinnefeld und Wegener, 2003) konfrontiert ist, motiviert. Das bedeutet, dass wir über eine gegebene (hier pseudoboolesche) Funktion $f\colon \{0,1\}^n \to \mathbb{R}$ anfangs gar keine Informationen besitzen und lediglich mithilfe von Auswertungen von f an Suchpunkten $x \in \{0,1\}^n$ Informationen gewinnen können. Die Kosten, die eine Suchheuristik zur Optimierung von f benötigt, messen wir dann in der Anzahl der Suchpunkte, deren f-Wert angefragt wurde. Diesem Maß werden wir bei der Analyse randomisierter Suchheuristiken generell folgen, zumal eine Auswertung der Zielfunktion in der Praxis tatsächlich ein teures Verfahren erfordern kann. Die oben erwähnte Zeit $T_{f,A}$ bemisst sich also anhand der Auswertungen von f, bis A einen für f optimalen Suchpunkt anfragt.

In diesem Black-Box-Szenario kann die Analyse spezieller Suchheuristiken uninteressant werden, sofern wir die Klasse der betrachteten Funktionen nicht einschränken. Das berühmte No-free-Lunch-Theorem von Wolpert und McReady (1997) besagt in unserem Fall nämlich, dass alle Suchheuristiken, gemittelt über alle denkbaren Funktionen $f\colon \{0,1\}^n \to W$ für einen endlichen, geordneten Wertebereich W, dieselbe Zahl von Funktionsauswertungen zur Optimierung benötigen, wenn man für jeden Suchpunkt nur die erste Anfrage daran zählt. Andererseits ist es klar, dass es aus praktischen Belangen nicht sinnvoll ist, über alle Funktionen zu argumentieren. Fast alle Funktionen scheiden allein aufgrund der Komplexität ihrer Beschreibung aus. Wir wollen die Thematik des No-free-Lunch-Theorems hier nicht weiter vertiefen, sondern verweisen stattdessen auf weiterführende Arbeiten, z. B. Droste, Jansen und Wegener (2002b). Wir halten jetzt fest, dass wir randomisierte Suchheuristiken auf speziellen Teilmengen aller denkbaren Funktionen untersuchen werden. Um einen Nutzen aus unserer Theorie ziehen zu können, versuchen wir damit auch praxisrelevante Klassen von Funktionen abzudecken. Im nächsten Abschnitt und im Verlauf der Arbeit werden wir allerdings auch sehen, inwieweit wir dabei zunächst Abstriche machen müssen.

2.3 Grundlegende Definitionen und Beweistechniken

In diesem Abschnitt wollen wir uns allmählich den interessanten randomisierten Suchheuristiken und zugehörigen Analysetechniken nähern. Wir haben bereits erwähnt, dass man bei neuen theoretischen Sichtweisen mit möglichst einfachen Gegenständen beginnen sollte. Als vielleicht einfachster, aber sicherlich in theoretischen Belangen meiststudierter evolutionärer Algorithmus für pseudoboolesche Funktionen gilt der folgende so genannte (1+1)-EA.

Definition 2.3.1 ((1+1)-EA) *Der* (1+1)-EA *für* $f\colon \{0,1\}^n \to \mathbb{R}$ *läuft wie folgt ab.*

Setze $t := 0$. *Wähle* $x_0 \in \{0,1\}^n$ *zufällig gleichverteilt. (Initialisierung)*
Wiederhole:

Erzeuge y *aus* x_t*, indem die Belegung jedes Bits von* x_t *unabhängig mit einer Wahrscheinlichkeit von* $1/n$ *durch ihr Komplement ersetzt wird. (Mutation)*
Falls $f(y) \geq f(x_t)$*, setze* $x_{t+1} := y$ *und* $x_{t+1} := x_t$ *sonst. (Selektion [zur Ersetzung])*
Setze $t := t + 1$.

Der (1+1)-EA verdankt seinen Namen den oben erwähnten Evolutionsstrategien, mit denen wir uns in Kapitel 7 noch des Weiteren auseinander setzen werden. Wie alle in diesem Teil der Dissertation auftretenden randomisierten Suchheuristiken arbeitet der (1+1)-EA mit einer Population der Größe 1 und hält an jedem *Zeitpunkt* t genau einen aktuellen Suchpunkt x_t vor. Daher benutzt der (1+1)-EA lediglich einen Mutationsoperator zur Variation. Der neue Suchpunkt y, der im t*-ten Schritt*, d. h. der zum Zeitpunkt $t - 1$ beginnenden Iteration der Schleife, erzeugt wird, heißt auch *Mutant*. Er weicht im Erwartungswert von x_t an genau einer Position ab. Andererseits ist es mit einer sehr kleinen Wahrscheinlichkeit möglich, dass in einem Schritt alle Bits *kippen*, d. h. durch ihr Komplement ersetzt werden. Darum kann jeder Schritt mit einer positiven Wahrscheinlichkeit einen optimalen Suchpunkt erzeugen, sodass der im letzten Abschnitt besprochene Erwartungswert $\mathrm{E}\bigl(T_{f,(1+1)\text{-EA}}\bigr)$ stets existiert. Da der Mutationsoperator weitere schöne Eigenschaften hat, später wiederkehren wird und auch in der Praxis geläufig ist, geben wir ihm einen Namen.

Definition 2.3.2 *Der Mutationsoperator des (1+1)-EA wird* $1/n$-Standardmutation *genannt.*

Wir besprechen noch die anderen Komponenten des (1+1)-EA. In der Initialisierung erfolgt eine gleichverteilte Wahl des *initialen Suchpunkts* x_0, was im Black-Box-Szenario die einzig sinnvolle Verteilung darstellt. Die Selektion zur Ersetzung weist eine Besonderheit auf. Sie ersetzt x_t durch y nur dann – wir sagen dazu, dass y *akzeptiert* wird –, wenn der f-Wert von y mindestens so groß wie der von x_t ist. Andernfalls sprechen wir davon, dass y *verworfen* werde. Wenn die genaue Gestalt von y nicht interessiert, sprechen wir häufig auch einfach von akzeptierten oder verworfenen Schritten bzw. Mutationen.

Mit anderen Worten sind die f-Werte der Folge der Suchpunkte, die der (1+1)-EA in seinem Lauf erzeugt, monoton wachsend. Im Vokabular aus dem Feld der Optimierung sagt man zu einem solchen Verfahren auch *Hillclimber*.

Definition 2.3.3 *Jede randomisierte Suchheuristik, die abgesehen von einem eventuell anderen Mutationsoperator der Beschreibung des (1+1)-EA folgt, heißt* randomisierter Hillclimber.

Wir verwenden auch für andere randomisierte Hillclimber als den (1+1)-EA die oben erwähnten, intuitiven Begriffe des Zeitpunkts t, eines Schrittes, eines Mutanten usw. In diesem Teil der Arbeit werden wir ausschließlich verschiedene randomisierte Hillclimber betrachten. Wir analysieren also in der Tat einfache randomisierte Suchheuristiken.

Um die Definition der randomisierten Hillclimber abzurunden, machen wir uns noch Gedanken über die Endlosschleife. In praktischen Implementierungen benötigt man ein Stoppkriterium, beispielsweise eine Maximallaufzeit. Für unsere theoretischen Analysen dürfen wir das Stoppkriterium fortlassen. Die oben erwähnte Zufallsvariable $T_{f,A}$ bezeichnet für randomisierte Hillclimber A und Zielfunktionen f die Zahl der Zielfunktionsauswertungen, bis erstmalig ein optimaler Suchpunkt angefragt wurde. In den kommenden Analysen sagen wir zu diesem Ereignis oft, dass *das Optimum erreicht/gefunden* wird. Weil auch der initiale Suchpunkt ausgewertet werden muss, ist $T_{f,A}$ um eins größer als der kleinste Zeitindex t von A, an dem das Optimum erreicht wird. Indem wir die zufälligen Suchpunkte von A an den diskreten Zeitpunkten t als Markoffkette ansehen, kann man $T_{f,A}-1$ in der zugehörigen Begriffswelt als *first hitting time* für die Menge der optimalen Zustände bezeichnen.

Trotz seiner Einfachheit enthält der (1+1)-EA viele typische Eigenschaften komplexerer evolutionärer Algorithmen und ergibt sich aus solchen manchmal auch als Spezialfall bei einer Populationsgröße von 1. Einen Einblick in die Analyse der (1+1)-EA auf pseudobooleschen Funktionen bietet die Arbeit von Droste, Jansen und Wegener (2002a), auf die wir uns im Folgenden häufig berufen werden.

Einen weiteren wichtigen randomisierten Hillclimber stellt die so genannte *randomisierte lokale Suche (RLS)* dar.

Definition 2.3.4 (RLS)
Die Suchheuristik RLS arbeitet wie der (1+1)-EA mit folgendem Mutationsoperator: Erzeuge y aus x_t, indem ein zufällig gleichverteilt gewähltes Bit von x_t gekippt wird.

Weil sich y von x_t nur in einem Bit unterscheidet, kann es nun passieren, dass $\mathrm{E}(T_{f,\mathrm{RLS}})$ auf manchen Funktionen f nicht existiert (unendlich wird). Dies hängt entscheidend davon ab, ob eine Funktion mehrere lokale Optima aufweist. In der folgenden Definition bezeichnet $H(x,y)$ den Hammingabstand, siehe Symbolverzeichnis.

Definition 2.3.5 *Sei $f\colon \{0,1\}^n \to \mathbb{R}$. Die Punkte $x, y \in \{0,1\}^n$ heißen* benachbart, *wenn $H(x,y) = 1$ gilt. Ein Punkt $x \in \{0,1\}^n$ heißt* lokales Maximum *von f, wenn für alle benachbarten $y \in \{0,1\}^n$ die Beziehung $f(y) \leq f(x)$ gilt.*

Naheliegenderweise heißt ein Suchpunkt x, sodass $f(x) \geq f(y)$ für alle $y \in \{0,1\}^n$ gilt, *global optimal*. Wenn der aktuelle Suchpunkt von RLS ein x ist, sodass für alle y mit $H(x,y) = 1$ ein geringerer f-Wert besteht, stagniert die Suche von RLS. Wenn x nicht global optimal ist, sagt man dann auch, dass RLS in einem lokalen Optimum stecken geblieben sei. Unter anderem deswegen kann es vorkommen, dass manche Praktiker(innen) RLS eher nicht als einen evolutionären Algorithmus ansehen.

Nichtsdestoweniger verkörpert RLS eine interessante randomisierte Suchheuristik. Da pro Schritt nur ein Bit kippt, sind Analysen von RLS oft einfacher zu bewerkstelligen als beim (1+1)-EA. Zumal der (1+1)-EA im Durchschnitt ein Bit kippt, gibt es Funktionenklassen, auf denen die asymptotischen Laufzeitverhalten beider Suchheuristiken einander gleichen. Daher erweist es sich manchmal als hilfreich, zunächst RLS auf einer Funktion(enklasse) zu analysieren, um eine Idee für das Verhalten des (1+1)-EA zu gewinnen. So werden wir u. a. im nächsten Kapitel verfahren.

An dieser Stelle bietet es sich auch an, gleich noch einen dritten randomisierten Hillclimber zu definieren, der sich bei dem soeben beschriebenen Ansatz der Übergangs von RLS auf den (1+1)-EA nützlich zeigt. Zuweilen ist die Varianz der Zahl der in einem Schritt des (1+1)-EA kippenden Bits für gute Laufzeitschranken noch zu groß. Um diese Varianz zu senken, stellen wir einen randomisierten Hillclimber vor, der sozusagen als Brücke zwischen dem (1+1)-EA und RLS fungiert. Er kippt in jedem Schritt mindestens ein Bit, um wertlose Schritte zu vermeiden.

Definition 2.3.6 ($\mathbf{RLS}_p$)
Die Suchheuristik RLS_p arbeitet wie der (1+1)-EA mit folgendem Mutationsoperator: Erzeuge y aus x_t, indem ein kippendes Bit zufällig gleichverteilt gewählt wird und zudem alle verbleibenden Bits unabhängig mit Wahrscheinlichkeit p gekippt werden.

Die erwartete Zahl kippender Bits pro Schritt beträgt bei RLS_p also $1 + p(n-1)$. Wenn wir über RLS_p reden, nennen wir das zunächst gewählte, mit Sicherheit kippende Bit das *erste kippende Bit* und die anderen kippenden Bits *zusätzlich kippende Bits.* Neben dem Aspekt, dass wertlose Schritte vermieden werden, ist RLS_p für unsere Analysen interessant, da RLS_p mit $p = 1/n$ dem (1+1)-EA sehr stark ähnelt und wir mit RLS, also RLS_p mit $p = 0$, als Ausgangspunkt über den Weg von RLS_p schließlich zu Aussagen über den populären (1+1)-EA kommen können.

Es ist jetzt an der Zeit, etwas näher auf die pseudobooleschen Funktionen, die wir betrachten werden, einzugehen. Zunächst möchte man mit selbst gewählten Funktionen ein typisches Verhalten der betrachteten randomisierten Suchheuristik und typische Auswirkungen ihrer Operatoren nachweisen. Als einfaches Beispiel kann man sich darunter eine konstruierte Funktion vorstellen, die dazu führt, dass RLS mit überwältigender Wahrscheinlichkeit $1 - 2^{-\Omega(n)}$ in einem lokalen Optimum stecken bleibt. Komplexere konstruierte Funktionen sollen beispielsweise den Nutzen von Populationen, siehe Teil II der Arbeit, nachweisen. Eine der frühesten konstruierten Funktionen für randomisierte Suchheuristiken stammt von Ackley (1987), der anhand der so genannten TRAP-Funktion ein Beispiel präsentiert, bei dem die von uns studierten randomisierten Hillclimber mit überwältigender Wahrscheinlichkeit eine exponentiell große (vgl. Definition A.16 im Anhang) Laufzeit besitzen; für einen Beweis im Hinblick auf den (1+1)-EA siehe Droste, Jansen und Wegener (2002a).

Bei solchen konstruierten Funktionen wird oft der Einwand erhoben, dass sie „künstlich“ seien und eher pathologische Instanzen für den EA darstellten. Diesem Einwand begegnen Droste, Jansen und Wegener (2000) mit der Diskussion über die „Natürlichkeit“ von Funktionen und anhand einer als natürlich angesehenen Funktion, die für evolutionäre Algorithmen schwierig erscheint. Wir greifen aber aus anderen Gründen immer noch häufig auf konstruierte Funktionen zurück. Zum einen muss man Eigenschaften von Funktionen, die das Verhalten einer randomisierten Suchheuristik beeinflussen, manchmal übertreiben, um in der O-Notation ins Gewicht fallende Laufzeitunterschiede zu zeigen. Oft sind wir ja sogar an exponentiell großen Laufzeitunterschieden interessiert. Wir glauben dann, dass andere Funktionen, die Züge unserer konstruierten Funktion tragen, immer noch zu praxisrelevanten Laufzeitunterschieden führen, aber (noch) keine rigoros beweisbaren Theoreme zulassen. Im Folgenden bezeichnen wir konstruierte Funktionen daher auch einfach als *Beispielfunktionen* oder *Testfunktionen.*

Noch pragmatischer ist zum anderen die Überzeugung, dass die ersten Schritte auf dem Weg zu einer theoretischen Analyse randomisierter Suchheuristiken bei ausgewählten Instanzen, die mit unseren analytischen Hilfsmitteln gut handhabbar sind, beginnen müssen. Es wäre vermessen, gleich zu Beginn komplexe EAs auf komplexen kombinatorischen Optimierungsproblemen theoretisch studieren zu wollen. Daher gingen die ersten Laufzeitanalysen auch anhand leicht zu begreifender Beispielfunktionen wie

$$\textsc{OneMax}(x_1, \ldots, x_n) \;:=\; \sum_{i=1}^{n} x_i$$

und

$$\textsc{LeadingOnes}(x_1, \ldots, x_n) \;:=\; \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$$

vonstatten. Die Funktion OneMax zählt also lediglich die Zahl der mit 1 belegten Bits im Suchpunkt und LeadingOnes zählt die Länge des längsten zusammenhängenden Blocks aus Einsbits, der bei der Position 1 beginnt. Beide Funktionen haben einen kleinen Wertebereich von nur $n+1$ Elementen und eine überschaubare Struktur. Sowohl der (1+1)-EA als auch RLS benötigen auf OneMax und LeadingOnes eine erwartete Laufzeit von $\Theta(n \log n)$ bzw. $\Theta(n^2)$. Die zugehörigen Beweise (Droste, Jansen und Wegener, 2002a) sind aus heutiger Sicht nicht allzu aufwendig. Mit Studien dieser einfachen Beispielfunktionen ist es aber gelungen, Analysemethoden für randomisierte Suchheuristiken zu entwickeln, die nun weiter tragen. Zum einen ist in den vergangenen Jahren ein Methodenreservoir geschaffen worden, das es zulässt, auch komplexere EAs zu analysieren; wir verweisen wieder auf den zweiten Teil der Dissertation, in dem uns übrigens auch OneMax und LeadingOnes wieder begegnen werden.

Zum anderen kommt uns das entstandene Methodenreservoir dabei zugute, randomisierte Suchheuristiken bezüglich ihres Verhaltens auf praxisrelevanten kombinatorischen Optimierungsproblemen zu analysieren. Seit kurzem ist eine Reihe von Arbeiten zu verzeichnen, die in diese Richtung gehen. Als Beispiel sei hier nur die Analyse des (1+1)-EA auf dem Problem der Berechnung maximaler Matchings erwähnt (Giel und Wegener, 2003), da wir in Kapitel 4 selbst diesen Ansatz verfolgen und entsprechende Literatur noch des Näheren erwähnen werden.

Aus einer eher mathematisch geprägten Perspektive verliert die Unterscheidung zwischen natürlichen und konstruierten Funktionen an Gewicht. Wir können jede pseudoboolesche Funktion $f\colon \{0,1\}^n \to \mathbb{R}$ als Polynom schreiben mit Koeffizienten $w_I \in \mathbb{R}$ für $I \subseteq \{1,\dots,n\}$, sodass für alle $x = (x_1,\dots,x_n) \in \{0,1\}^n$ die Identität

$$f(x) = \sum_{I\subseteq\{1,\dots,n\}} w_I \cdot \prod_{i\in I} x_i$$

gilt. Da der Definitionsbereich endlich ist und da eine Skalierung des f-Wertes um einen konstanten Faktor bei randomisierten Hillclimbern keine Rolle spielt, nimmt man übrigens oft $w_I \in \mathbb{Z}$ an. Jedenfalls eröffnet die Schreibweise als Polynom eine strukturelle Sicht auf pseudoboolesche Funktionen. Eine der wichtigsten Größen ist in diesem Zusammenhang ihr *Grad*, d. h. $\max_I\{|I| \mid w_I \neq 0\}$. Im Bereich der mathematischen Optimierung schreibt man beispielsweise linearen Funktionen, d. h. Funktionen vom Grad 1, und nicht linearen Funktionen entscheidende Unterschiede zu. Es ist somit spannend, inwieweit sich Struktureigenschaften pseudoboolescher Funktionen auf das Laufzeitverhalten randomisierter Suchheuristiken auswirken. Darauf werden wir in Kapitel 3 zu sprechen kommen.

Umriss wesentlicher Beweistechniken

Der Rest dieses Abschnitts widmet sich fundamentalen Beweismethoden, die bei der Laufzeitanalyse randomisierter Suchheuristiken immer wiederkehren werden. Damit ist es hilfreich, sie bereits an dieser Stelle zu sammeln. Die Leserin oder der Leser kann den Rest dieses Abschnitts daher zunächst überspringen und bei Anwendungen in späteren Kapiteln darauf zurückkommen.

Ganz zentral sind immer wieder die Wahrscheinlichkeiten für gewisse Mutationen des (1+1)-EA. Die Wahrscheinlichkeit, dass eine Mutation den Suchpunkt $x \in \{0,1\}^n$ in den Suchpunkt $y \in \{0,1\}^n$ verwandelt, beträgt offensichtlich

$$\left(\frac{1}{n}\right)^{H(x,y)} \left(1-\frac{1}{n}\right)^{n-H(x,y)}.$$

Im Fall $H(x,y) \geq 1$ schätzen wir den letzten Ausdruck gerne durch $\mathrm{e}^{-1}(1/n)^{H(x,y)}$ nach unten ab (Lemma A.1, vgl. auch für weitere Abschätzungen). Der Fall, dass der (1+1)-EA ein Replikat erzeugt, d. h. die Mutation in x kein Bit kippt, hat eine Wahrscheinlichkeit von $(1-1/n)^n \geq 1/(2\mathrm{e})$, da o. B. d. A. $n \geq 2$ ist.

Aus der Eigenschaft, dass die Anzahl kippender Bits in einem Schritt des (1+1)-EA binomialverteilt zu den Parametern n und $1/n$ und damit asymptotisch poissonverteilt zum Parameter 1 ist, erhalten wir, dass die Wahrscheinlichkeit für k kippende Bits exponentiell klein in k wird. Genauer gesagt ist die Wahrscheinlichkeit, dass in einem Schritt des (1+1)-EA mindestens (irgendwelche) k Bits kippen, höchstens

$$\binom{n}{k}\left(\frac{1}{n}\right)^k \leq \frac{n^k}{k!}\left(\frac{1}{n}\right)^k \leq \frac{1}{k!},$$

was gemäß der stirlingschen Formel (Lemma A.3) durch $2^{-\Omega(k\log k)}$ nach oben beschränkt ist. Wir brauchen häufig die Aussage, dass die Wahrscheinlichkeit von $\Omega(n)$ in einem Schritt kippenden Bits höchstens $2^{-\Omega(n\log n)}$ ist. Andererseits hat jede beliebige Mutation gemäß den Betrachtungen aus dem letzten Absatz eine Wahrscheinlichkeit von mindestens $(1/n)^n = 2^{-O(n\log n)}$.

Zuweilen interessiert uns auch die Gesamtzahl kippender Bits in k Schritten. Wir können dann mit unabhängigen Bernoulliversuchen argumentieren. Die Zufallsvariable $X_{i,j}$ sei 1 genau dann, wenn der (1+1)-EA im i-ten der k Schritte das j-te Bit des aktuellen Suchpunkts kippt, und 0 sonst. Alle $X_{i,j}$ sind unabhängig, da der (1+1)-EA einzelne Positionen unabhängig voneinander kippt und das Verhalten des Mutationsoperators in verschiedenen Schritten unabhängig ist. Für $X := \sum_{i=1}^{k}\sum_{j=1}^{n} X_{i,j}$ gilt $\mathrm{E}(X) = k$. Gemäß der Chernoffungleichung (Lemma A.13) können wir Aussagen wie $\mathrm{Prob}(X \geq 2k) = 2^{-\Omega(k)}$ ableiten.

Wenn wir Aussagen über die erwartete Laufzeit einer randomisierten Suchheuristik zeigen, möchten wir häufig unliebsame Ereignisse ausschließen, die zwar unwahrscheinlich sind, aber schwer kontrollierbare Folgen haben können. Die Beweismethode des *typischen Laufs* beruht darauf, in einem Schritt oder sogar für gewisse Zeitspannen solche unliebsamen Ereignisse, z. B. das oben erwähnte Kippen von $\Omega(n)$ Bits in einem Schritt, auszuschließen. Wir rechnen zunächst die Wahrscheinlichkeit des unliebsamen Ereignisses, das wir häufig einfach als *Fehler* bezeichnen, aus, nennen diese dann passenderweise *Fehlerwahrscheinlichkeit* und arbeiten anschließend unter der Annahme, dass dieser Fehler nicht eintritt. Sei E ein Ereignis, das wir im Laufe einer randomisierten Suchheuristik A betrachten wollen, beispielsweise die Wahrscheinlichkeit, dass A in $O(n)$ Schritten das Optimum findet. Sei außerdem F das Fehlerereignis. Gemäß dem Satz von der totalen Wahrscheinlichkeit (Lemma A.10) folgt

$$\mathrm{Prob}(E) \geq \mathrm{Prob}(E \mid \overline{F}) \cdot \mathrm{Prob}(\overline{F}) \geq \mathrm{Prob}(E \mid \overline{F}) - \mathrm{Prob}(F).$$

Also erhalten wir einen Wert, der höchstens um die Fehlerwahrscheinlichkeit verfälscht wird. Gelegentlich iteriert man diese Idee, indem man unter der Annahme, dass ein erster Fehler nicht eintritt, einen zweiten Fehler betrachtet, dessen Wahrscheinlichkeit abschätzt usw. Unter all diesen Annahmen versuchen wir dann für die randomisierte Suchheuristik eine Aussage zu treffen, die wir als typischen Lauf verstehen, wenn die

Summe aller Fehlerwahrscheinlichkeiten klein, am besten exponentiell klein, ist. Dieses Konzept wird uns sehr häufig begegnen.

Eng verwandt mit dem gerade Geschilderten sind Abschätzungen, die wir vornehmen, wenn wir am Durchschnitt von Ereignissen interessiert sind. Beispielsweise möchten wir zeigen, dass der Lauf einer randomisierten Suchheuristik mit hoher Wahrscheinlichkeit sowohl das Ereignis A als auch das Ereignis B erfüllt. Wenn diese nicht unabhängig sind, können wir versuchen, die Identität

$$\text{Prob}(A \cap B) \;=\; \text{Prob}(A \mid B) \cdot \text{Prob}(B)$$

auszunutzen und zunächst das Ereignis B zu betrachten. Das Ereignis $\overline{B}$ bezeichnen wir dann wieder als Fehler. Anschließend müssen wir das bedingte Ereignis $A \mid B$ untersuchen. Manchmal kann diese Bedingung aber stören. In einem solchen Fall können wir mit unteren Schranken für $\text{Prob}(A \cap B)$ arbeiten, indem wir die Abschätzung

$$\text{Prob}(A \cap B) \;=\; 1 - \text{Prob}(\overline{A} \cup \overline{B}) \;\geq\; 1 - \text{Prob}(\overline{A}) - \text{Prob}(\overline{B})$$

ausnutzen. Auch hier reden wir gerne von Fehlern und meinen damit die Ereignisse $\overline{A}$ und $\overline{B}$. Fazit ist, dass wir uns mit dem Aufsummieren von Fehlerwahrscheinlichkeiten und ohne bedingte Ereignisse zu betrachten oft die Beweise erleichtern können. Allerdings erhalten wir damit nur untere Schranken für die gesuchte Wahrscheinlichkeit.

Eine weitere, immer wiederkehrende Beweisidee steckt in der so genannten *Wiederholung unabhängiger Phasen*. Während $T_{f,A}$ die Laufzeit des randomisierten Hillclimbers A auf f bei gleichverteilter Wahl des initialen Suchpunkts angibt, gelingt es auch oft, Schranken für die Laufzeit unter Annahmen über den initialen Suchpunkt herzuleiten. Sei $T^p_{f,A}$ die Laufzeit von A auf f, wenn der initiale Suchpunkt gemäß der Verteilung p gezogen wird. Angenommen, es sind Aussagen über die Verteilung von $T^p_{f,A}$ bekannt, die für alle p gelten. Speziell seien eine Laufzeitschranke s und eine obere Schranke u gegeben, sodass für alle p

$$\text{Prob}(T^p_{f,A} > s) \;\leq\; u$$

gilt. Wenn A das Optimum nicht bis zum Zeitpunkt s erreicht, können wir mit der zufälligen Verteilung des s-ten Suchpunktes argumentieren. Da die vorgenannte Abschätzung für alle p gilt, erreicht A das Optimum innerhalb der Phase vom Zeitpunkt s bis zum Zeitpunkt $2s$ wiederum mit einer Wahrscheinlichkeit von mindestens $1-u$. In unserer Abschätzung ist das Ereignis, dass das Optimum in einer Phase gefunden wird, unabhängig vom Ereignis, dass es in der nachfolgenden Phase gefunden wird. Somit kann die zufällige Zahl der Phasen mit Länge s, die A bis zum Erreichen des Optimums durchläuft, mittels einer geometrisch zum Parameter u verteilten Zufallsvariablen nach oben abschätzt werden. Also gilt für alle p

$$\text{Prob}(T^p_{f,A} > ks) \;\leq\; u^k$$

und speziell $\text{Prob}(T_{f,A} > ks) \leq u^k$. Darüber hinaus beträgt die erwartete Zahl von Phasen, bis A das Optimum findet, maximal $1/u$ und wir erhalten

$$\text{E}(T_{f,A}) \leq \frac{s}{u}.$$

In den obigen Abschätzungen sind die Kosten der Anfrage an den initialen Suchpunkt, die $T_{f,A}$ auch zählen muss, mindestens einmal enthalten, da wir mit Phasen der Länge s arbeiten. Die Beweisidee der Wiederholung unabhängiger Phasen wird häufig in Verbindung mit einer Anwendung der Markoffungleichung (Lemma A.11) auftreten, welche dann die oberen Schranken u liefern wird.

Zum Abschluss dieses Kapitels kommen wir zu einer ersten formal detailliert beschriebenen Beweistechnik. Die einfache, aber erfolgreiche Technik liefert obere Schranken für die erwartete Laufzeit eines randomisierten Hillclimbers A und beruht auf der so genannten *Ranggruppenmethode*, die bereits implizit von Mühlenbein (1992) benutzt wurde und beispielsweise von Droste (2000) sowie Jansen (2000) ausführlich dargestellt wird. Ihr liegt die Idee zugrunde, den Suchraum $\{0,1\}^n$ anhand der Zielfunktion geeignet zu partitionieren. Da randomisierte Hillclimber keine Verschlechterungen des f-Wertes des aktuellen Suchpunkts akzeptieren, sollen die Mengen der Partition eine Verbesserung des f-Wertes widerspiegeln.

Definition 2.3.7 *Sei $f\colon \{0,1\}^n \to \mathbb{R}$ und seien L_1 und L_2 nicht leere Teilmengen von $\{0,1\}^n$. Wir schreiben $L_1 <_f L_2$ genau dann, wenn für alle $\ell_1 \in L_1$ und $\ell_2 \in L_2$ die Beziehung $f(\ell_1) < f(\ell_2)$ gilt.*

Wir benutzen die Relation $<_f$ im Folgenden für spezielle Partitionen, d. h. disjunkte Zerlegungen, des Suchraums $\{0,1\}^n$.

Definition 2.3.8 *Sei $f\colon \{0,1\}^n \to \mathbb{R}$, sei $f_{\max} := \max\{f(x) \mid x \in \{0,1\}^n\}$ und sei $L_1, \ldots, L_m$ eine Partition von $\{0,1\}^n$ mit $L_i <_f L_{i+1}$ für $1 \leq i \leq m-1$ sowie $f(a) = f_{\max}$ für $a \in L_m$. Dann heißt $L_1, \ldots, L_m$ eine f-*basierte Partition*. Die Mengen der Partition heißen* Ranggruppen.

Die Ranggruppen *(fitness levels)* $L_1, \ldots, L_m$ einer f-basieren Partition sind also bezüglich $<_f$ geordnet und L_m enthält nur (und damit alle) Suchpunkte mit maximalem f-Wert. Stehe A wieder für einen randomisierten Hillclimber. Die zugrunde liegende Idee für einfache obere Schranken beruht darauf, dass der Index der Ranggruppe, in der sich der aktuelle Suchpunkt von A befindet, nicht sinken kann. Damit kann A jede Ranggruppe nur einmal verlassen, sodass wir an der Wahrscheinlichkeit, eine Ranggruppe zu verlassen, interessiert sind.

Definition 2.3.9 *Sei $L_1, \ldots, L_m$ eine f-basierte Partition und A ein randomisierter Hillclimber. Ein Wert $s(i) > 0$, $1 \leq i \leq m-1$, heißt* Verlassenswahrscheinlichkeit *von L_i bezüglich A, wenn für alle $x \in L_i$, die Wahrscheinlichkeit, dass der Mutationsoperator von A, angewandt auf x, ein $y \in L_{i+1} \cup \cdots \cup L_m$ erzeugt, mindestens $s(i)$ ist.*

Die als Verlassenswahrscheinlichkeiten bezeichneten Werte sind im Allgemeinen natürlich untere Schranken. Weil wir $s(i) > 0$ fordern, braucht es nicht für jede f-basierte Partition Verlassenswahrscheinlichkeiten zu geben. Allerdings existieren immer positive Verlassenswahrscheinlichkeiten, wenn wir den (1+1)-EA betrachten.

Wir erhalten zwei einfache Anwendungen für f-basierte Partitionen.

Lemma 2.3.10 *Sei A ein randomisierter Hillclimber, sei $L_1, \dots, L_m$ eine f-basierte Partition und seien die Werte $s(i)$, $1 \leq i \leq m-1$, diesbezügliche Verlassenswahrscheinlichkeiten von A. Der aktuelle Suchpunkt von A auf f liege in der k-ten Ranggruppe. Für $\ell > k$ ist die erwartete Zahl weiterer Schritte, bis A einen Suchpunkt aus mindestens der ℓ-ten Ranggruppe erzeugt, höchstens $1/s(k) + \cdots + 1/s(\ell-1)$.*

Beweis: Wir wissen bereits, dass jede Ranggruppe höchstens einmal verlassen werden muss. Wenn sich der aktuelle Suchpunkt von A in der i-ten Ranggruppe befindet, ist die Wahrscheinlichkeit, diese Ranggruppe im folgenden Schritt zugunsten einer Ranggruppe mit höherem Index zu verlassen, mindestens $s(i)$. Die Zeit bis zum Verlassen kann somit mithilfe einer zum Parameter $s(i)$ geometrisch verteilten Zufallsvariablen abgeschätzt werden. Der zugehörige Erwartungswert beträgt $1/s(i)$. Wir summieren diese Erwartungswerte für alle Ranggruppen zwischen k und $\ell - 1$ auf. □

Lemma 2.3.11 *Sei $L_1, \dots, L_m$ eine f-basierte Partition und seien die Werte $s(i)$, $1 \leq i \leq m-1$, Verlassenswahrscheinlichkeiten eines randomisierten Hillclimbers A. Dann gilt*

$$\mathrm{E}(T_{f,A}) \;\leq\; 1 + \sum_{i=1}^{m-1} \frac{1}{s(i)}.$$

Beweis: Wir wenden Lemma 2.3.10 mit $\ell = m$ und der pessimistischen Annahme $k = 1$ an. Wir addieren 1, da ein optimaler Suchpunkt mindestens einmal angefragt werden muss. □

Gelegentlich bezeichnen wir eine Anwendung von Lemma 2.3.10 oder Lemma 2.3.11 einfach als Ranggruppenmethode. In anderen Zusammenhängen ist es manchmal besser, den zufälligen Index i der Ranggruppe, der ein aktueller Suchpunkt von A angehört, als *Potenzial* zu bezeichnen. Somit lässt sich i auch mit dem aus der Analyse effizienter Algorithmen gängigen Begriff einer *Potenzialfunktion* benennen, deren Wert im Laufe von A in diesem Fall nicht sinken kann. Diese alternative Sichtweise werden wir aber erst im zweiten Teil dieser Dissertation einnehmen.

3 Randomisierte Suchheuristiken auf monotonen Polynomen

3.1 Problemstellung und Hintergrund

Erste Laufzeitanalysen für den (1+1)-EA stützten sich auf einfache Beispielfunktionen, u. a. die im letzten Kapitel erwähnte Funktion OneMax, die als eine der einfachsten, aber nicht trivialen pseudobooleschen Funktionen gelten kann. Aus diesem Grunde ist OneMax die im Zusammenhang mit dem (1+1)-EA vielleicht am häufigsten studierte Funktion (siehe z. B. Mühlenbein, 1992; Rudolph, 1997; Garnier, Kallel und Schoenauer, 1999; Droste, Jansen und Wegener, 2002a). Obwohl man nicht hoffen kann, dass eine randomisierte Suchheuristik auf allen pseudobooleschen Funktionen effizient ist, ist es wünschenswert, Unterklassen K von pseudobooleschen Funktionen zu identifizieren, sodass eine gegebene randomisierte Suchheuristik, z. B. der (1+1)-EA, auf allen $f \in K$ effizient ist; die Funktion OneMax fällt beispielsweise in die Klasse der linearen Funktionen, für die der (1+1)-EA bereits analysiert wurde (Droste, Jansen und Wegener, 2002a). Auf der anderen Seite soll auch nachgewiesen werden, dass manche Unterklassen bereits zu groß sind, als dass sie generell effiziente Laufzeiten der betrachteten randomisierten Suchheuristik zuließen. Aus der Sicht von Anwender(inne)n, die eine unbekannte Funktion f optimieren möchten, kann eine solche Klassifikation hilfreich sein, wenn, anders als im Black-Box-Szenario gefordert, doch ein gewisses Verständnis von f vorhanden ist. Weiß man beispielsweise, dass f Züge einer linearen Funktion trägt, kann man sich beim Einsatz evolutionärer Algorithmen Erfolge erhoffen, da beispielsweise der (1+1)-EA auf allen linearen Funktionen eine kleine erwartete Laufzeit von $O(n \log n)$ hat.

Wegener und Witt (2005a) betrachten quadratische pseudoboolesche Funktionen, also Funktionen vom Grad 2. Obwohl aus komplexitätstheoretischer Sicht diese Funktionenklasse insoweit zu groß ist, als die Optimierung solcher Funktionen im Allgemeinen NP-hart ist (siehe z. B. Wegener, 2003), gelingt die Charakterisierung wichtiger Unterklassen quadratischer Funktionen. Auf manchen Unterklassen sind der (1+1)-EA oder zumindest polynomiell viele parallele Läufe davon (so genannte *multistarts*) effizient, wohingegen Beispiele existieren, auf denen der (1+1)-EA völlig ineffizient ist.

Nicht nur der Grad eines Polynoms, sondern auch die Nachbarschaftsstruktur einer Funktion kann als Klassifikation herhalten. In diesem Zusammenhang sind die *unimodalen Funktionen* von besonderem Interesse. Wir benötigen dazu den Begriff eines lokalen Maximums aus Definition 2.3.5.

Definition 3.1.1 *Eine Funktion $f\colon \{0,1\}^n \to \mathbb{R}$ heißt* unimodal*, wenn sie genau ein lokales Maximum besitzt.*

Droste, Jansen und Wegener (2002a) geben eine unimodale Funktion an, auf der der (1+1)-EA im Erwartungswert eine exponentielle Zeit benötigt. Im Rahmen des zweiten Teils dieser Dissertation werden auch unimodale Funktionen eine Rolle spielen.

Als weitere Unterklasse pseudoboolescher Funktionen bieten sich die so genannten *monotonen Polynome* (vgl. auch Wegener, 2001) an. Bei diesen stellen wir spezielle Anforderungen an die Vorzeichen der Koeffizienten.

Definition 3.1.2 *Eine Funktion $f\colon \{0,1\}^n \to \mathbb{R}$ heißt* monotones Polynom*, wenn sie für Variablen $z_1, \dots, z_n$ mit $z_i \in \{x_i, 1-x_i\}$ geschrieben werden kann als*

$$f(x) = \sum_{I \subseteq \{1,\dots,n\}} w_I \cdot \prod_{i \in I} z_i,$$

sodass $w_I \geq 0$ für alle I gilt.

Ein monotones Polynom enthält also ausschließlich nicht negative Koeffizienten, nachdem wir möglicherweise Variablen x_i durch ihr Komplement $1-x_i$ ersetzt haben. Damit ist die Menge der optimalen Suchpunkte verschiedener monotoner Polynome im Allgemeinen auch verschieden, sodass es keine trivialen Black-Box-Algorithmen (vgl. Droste, Jansen, Tinnefeld und Wegener, 2003) gibt. Die so genannte Black-Box-Komplexität der gesamten Klasse der monotonen Polynome ist, wie in der gerade erwähnten Arbeit gezeigt, tatsächlich nur durch $\Omega(2^n)$ nach unten beschränkt, da der Klasse Nadel-im-Heuhaufen-Funktionen wie $\prod_{i=1}^n x_i$ angehören.

Bezüglich jedes monotonen Polynoms f nehmen wir im Folgenden an, dass für alle i die Wahl $z_i = x_i$ in Definition 3.1.2 möglich ist. Letzteres lässt sich anhand folgender Symmetriegründe motivieren. Wir werden den (1+1)-EA und RLS_p betrachten, die den initialen Suchpunkt zufällig gleichverteilt aus $\{0,1\}^n$ wählen. Außerdem nehmen deren Mutationsoperatoren an einem zur Mutation ausgewählten Bit i lediglich solche Änderungen vor, die den Inhalt von Bit i mit Wahrscheinlichkeit 1 durch sein Komplement ersetzen, und die zufällige Auswahl von i ist unabhängig vom Suchpunkt. Also können wir formal von $\{0,1\}^n$ zu einem anderen Suchraum übergehen, in dem an manchen Bits die Namen 0 und 1 für ihre Belegungen gegeneinander ausgetauscht werden, ohne dass das zufällige Verhalten des (1+1)-EA und von RLS_p sich änderte. Obwohl in den nachfolgend betrachteten monotonen Polynomen also stets der nur aus Einsen bestehende Suchpunkt 1^n optimal ist, werden die folgenden Ergebnisse über den (1+1)-EA und RLS_p somit für alle monotonen Polynome gelten.

Wir haben bereits erwähnt, dass die Klasse der monotonen Polynome immer noch zu groß für randomisierte Suchheuristiken ist, wenn wir an einer effizienten Optimierung interessiert sind. Daher schränken wir nun die Klasse der monotonen Polynome anhand zweier Parameter ein. Erstens betrachten wir ihren Grad d. Zweitens ist in einem monotonen Polynom f die *Anzahl nicht verschwindender Terme* N wichtig. Diese ist definiert als die Zahl der Koeffizienten w_I, die in der Darstellung aus Definition 3.1.2 einen von 0 verschiedenen Wert annehmen.

Es ist intuitiv klar, dass monotone Polynome kleinen Grades und kleiner Anzahl nicht verschwindender Terme einfacher zu optimieren sind. Die folgenden Ausführungen werden daher zum Ziel haben, die Laufzeit und Erfolgswahrscheinlichkeit von RLS_p (einschließlich RLS) und des (1+1)-EA auf monotonen Polynomen in Abhängigkeit von den Parametern d und N sowie natürlich der Problemdimension n zu ermitteln. Wir beginnen dabei mit dem Fall $N = 1$, d. h. dem Fall, in dem ein monotones Polynom lediglich ein nicht verschwindendes Monom besitzt. In späteren Abschnitten werden wir auf die dazugehörigen Ergebnisse zurückgreifen können, um Resultate für komplexe monotone Polynome vorzustellen.

3.2 Die Optimierung von Monomen

Ein monotones Polynom mit $N = 1$ besteht nur aus einem Monom $m(x)$. Im vorigen Abschnitt haben wir bereits erläutert, dass wir o. B. d. A. die Gestalt $m(x) = w_m x_{i_1} \cdots x_{i_d}$ mit Indizes $i_k \in \{1, \dots, n\}$ annehmen dürfen. Weil der (1+1)-EA und RLS_p darüber hinaus per definitionem das gleiche zufällige Verhalten zeigen, wenn wir die Benennung der Bitpositionen $1, \dots, n$ permutieren, und weil die Größe des Koeffizienten bei nur einem Monom egal ist, gehen wir im Folgenden sogar davon aus, dass wir ein Monom der Gestalt $m(x) = x_1 \cdots x_d$ vorliegen haben. Also kann nur der Parameter d, der Grad von $m(x)$, einen Einfluss auf die Laufzeit der betrachteten randomisierten Suchheuristiken haben.

Der Lauf des (1+1)-EA und von RLS_p auf $m(x)$ lässt sich mithilfe relativ einfacher Markoffkettenmodelle beschreiben, sodass wir bekannte Ergebnisse über Übergangszeiten *(hitting times)* von Markoffketten (etwaige Begriffsbildungen sind an Motwani und Raghavan (1995) angelehnt) anwenden können. Allerdings ist ein kleiner Trick vonnöten. Zunächst halten wir die triviale Beobachtung fest, dass $m(x) = 0$ für alle x mit $(x_1, \dots, x_d) \neq 1^d$ gilt und die Suchheuristik damit jeden neuen Suchpunkt akzeptiert, sofern das Optimum noch nicht erreicht ist. Da wir lediglich an dem ersten Zeitpunkt interessiert sind, zu dem unsere Suchheuristik die optimale Belegung $(x_1, \dots, x_d) = 1^d$ für $m(x)$ gefunden hat, können wir es uns damit erlauben, den Selektionsoperator der Suchheuristiken so zu modifizieren, dass der Mutant stets akzeptiert wird. Damit erhalten wir eine ergodische Markoffkette auf dem Zustandsraum $\{0, 1\}^n$, deren Anfangsverteilung die Gleichverteilung ist. Weil die Übergangswahrscheinlichkeiten der betrachteten Suchheuristiken in diesem Zustandsraum symmetrisch sind und die Übergangsmatrix damit doppelt stochastisch ist, ist auch die eindeutige stationäre Verteilung dieser ergodischen Markoffkette die Gleichverteilung über $\{0, 1\}^n$ (für einen einfachen Beweis siehe Schickinger und Steger, 2001).

Wiederum aus den oben genannten Symmetriegründen können wir den Zustandsraum der Markoffkette weiter vereinfachen. Anstelle von $\{0,1\}^n$ betrachten wir den Zustandsraum $D = \{0, \dots, d\}$ und bilden den Zustand $(x_1, \dots, x_n)$ auf $x_1 + \cdots + x_d$ ab. Damit folgt für die stationäre Verteilung π auf D, dass $\pi(i) = \binom{d}{i} 2^{-d}$ für $0 \leq i \leq d$ gilt. Die Übergangswahrscheinlichkeiten der Markoffkette hängen vom eingesetzten Mutationsoperator ab. Im folgenden Theorem sehen wir, dass es aber asymptotisch keine Rolle spielt, ob ein oder mehrere Bits in einem Schritt kippen können. Aus Gründen, die im Zusammenhang mit komplizierten monotonen Polynomen in Abschnitt 3.5 klar werden, lassen wir auch die Bedingung zu, dass die Suchheuristik nur wenige Bits pro Schritt kippt.

Theorem 3.2.1 *Die erwartete Laufzeit des (1+1)-EA und von RLS$_p$ mit $p \leq 1/n$ auf einem Monom vom Grad d ist höchstens $3\mathrm{e}(n/d)2^d$ und mindestens $\Omega((n/d)2^d)$. Die obere Schranke gilt auch, wenn angenommen wird, dass der initiale Suchpunkt fest vorgegeben ist, oder angenommen wird, dass jeder Schritt höchstens zwei Bits kippt (oder beides angenommen wird).*

Beweis: Wir beginnen mit der oberen Schranke. Sei mit Y die aus dem Verhalten von RLS$_p$ oder des (1+1)-EA resultierende Markoffkette mit dem Zustandsraum D bezeichnet. (Zur Vereinfachung der Notation lassen wir die konkret betrachtete Suchheuristik im Moment unspezifiziert.) Mit $\tau_{i,j}$ bezeichnen wir den ersten Zeitpunkt $t \geq 1$, sodass $Y(t) = j$ gilt, wenn $Y(0) = i$ vorausgesetzt ist. Aus dem Fundamentalsatz für ergodische Markoffketten (Motwani und Raghavan, 1995) folgt, dass $\mathrm{E}(\tau_{i,i}) = 1/\pi(i) = 2^d / \binom{d}{i}$ gilt. Für das Theorem benötigen wir Aussagen über $\mathrm{E}(\tau_{i,d})$. Da die Suchheuristiken meist nur ein Bit pro Schritt kippen, nehmen wir für jedes i die Abschätzung

$$\mathrm{E}(\tau_{i,d}) \;\leq\; \sum_{j=0}^{d-1} \mathrm{E}(\tau_{j,j+1})$$

vor, woraus auch folgt, dass die obere Schranke ebenso für jeden fest vorgegebenen initialen Suchpunkt gilt.

Seien $P(i,j)$ die Übergangswahrscheinlichkeiten von Y. Laut dem Satz von der totalen Wahrscheinlichkeit folgt

$$\mathrm{E}(\tau_{j+1,j+1}) \;=\; 1 + \sum_{\substack{k=0 \\ k \neq j+1}}^{d} P(j+1,k) \cdot \mathrm{E}(\tau_{k,j+1}) \;\geq\; P(j+1,j) \cdot \mathrm{E}(\tau_{j,j+1})$$

und damit

$$\mathrm{E}(\tau_{j,j+1}) \;\leq\; \frac{2^d}{\binom{d}{j+1} \cdot P(j+1,j)}.$$

Nun benötigen wir untere Schranken für $P(j+1,j)$. Hier müssen wir zwischen RLS$_p$ und dem (1+1)-EA unterscheiden. RLS$_p$ mit $p \leq 1/n$ geht vom Zustand $j+1$ in den Zustand j über, wenn das erste kippende Bit als eines der $j+1$ Einsbits im Monom $x_1 \cdots x_d$ gewählt wird und kein anderes Bit kippt. Damit erhalten wir die untere Schranke $((j+1)/n)(1-p)^{n-1} \geq ((j+1)/n)(1-1/n)^{n-1} \geq (j+1)/(en)$. Beim (1+1)-EA ist die Wahrscheinlichkeit, dass genau ein Bit kippt und dieses eines der $j+1$ Einsbits im Monom ist, mindestens $(j+1)(1/n)(1-1/n)^{n-1} \geq (j+1)/(en)$. Beide unteren Schranken werden nur größer, wenn wir die Annahme treffen, dass in dem Schritt höchstens zwei Bits kippen. Aufgrund der Beziehung

$$\binom{d}{j+1}\frac{j+1}{en} = \binom{d-1}{j}\frac{d}{en}$$

folgt

$$E(\tau_{i,d}) \leq \sum_{j=0}^{d-1} \frac{2^d}{\binom{d-1}{j}\frac{d}{en}} \leq 2^d \cdot \frac{en}{d} \cdot \left(2 + \sum_{j=1}^{d-2} \frac{1}{\binom{d-1}{d-2}}\right) \leq \frac{3en2^d}{d}.$$

In der vorletzten Ungleichung haben wir die Symmetrie der Binomialkoeffizienten ausgenutzt und sie für $1 \leq j \leq d-2$ durch ihren kleinsten Wert nach unten abgeschätzt. Damit ist die obere Schranke gezeigt.

Zum Beweis der unteren Schranke beachten wir, dass die betrachteten Suchheuristiken Black-Box-Algorithmen darstellen, und wenden die Theorie der Black-Box-Komplexität an. Nach Droste, Jansen, Tinnefeld und Wegener (2003) benötigt jeder Black-Box-Algorithmus zur Optimierung eines Monoms vom Grad d (in dem wieder die Rolle von Nullen und Einsen vertauscht worden sein darf) im Erwartungswert mindestens $2^{d-1} + 1/2$ Zielfunktionsauswertungen. Diese Aussage zählt allerdings nur die Anzahl der verschiedenen angefragten Suchpunkte im Suchraum $\{0,1\}^d$. Wir brauchen also im Erwartungswert mindestens $\Omega(2^d)$ so genannte relevante Mutationen, die die Eigenschaft haben, mindestens ein Bit des betrachteten Monoms $x_1 \cdots x_d$ zu kippen. Die Wahrscheinlichkeit des Letzteren ist in Bezug auf den (1+1)-EA höchstens d/n. Für RLS_p können wir die Wahrscheinlichkeit durch $d/n + ((n-d)/n) \cdot d \cdot p \leq 2d/n$ nach oben abschätzen, indem wir mithilfe des Satzes von der totalen Wahrscheinlichkeit unterscheiden, ob das erste kippende Bit von RLS_p zum Monom gehört oder nicht. Also ist die erwartete Zeit, bis eine relevante Mutation eintritt, durch $\Omega(n/d)$ beschränkt. Da jede Mutation unabhängig mit einer festen Wahrscheinlichkeit relevant ist, ist die untere Schranke $\Omega((n/d)2^d)$ für die erwartete Laufzeit gezeigt. □

Theorem 3.2.1 stellt eine asymptotisch exakte Verallgemeinerung eines Theorems von Garnier, Kallel und Schoenauer (1999) dar, in dem die entsprechenden Schranken nur für den Fall $d = n$ und die Suchheuristiken RLS_0 und den (1+1)-EA gezeigt werden. Die von der Suchheuristik RLS_0 beschriebene Suche innerhalb von $\{0,1\}^n$, dem so genannten booleschen Hyperwürfel, bewegt sich in jedem Schritt nur lokal entlang einer Kante des Hyperwürfels, indem sie einen Hammingabstand von genau 1 zurücklegt. Für derartige Prozesse existieren in der Literatur Untersuchungen, die uns die obere Schranke aus Theorem 3.2.1 wiederum für RLS_0 und $d = n$ liefern

(Garcia und Palacios, 2002). Im Beweis von Theorem 3.2.1 haben wir mit einfachen Abschätzungen auch die globalen Suchheuristiken, d. h. RLS_p mit $p > 0$ und den (1+1)-EA, analysieren können.

Das Szenario der Optimierung eines Monoms $m(x) = x_1 \cdots x_d$ kann auch als Irrfahrt *(random walk)* auf einem *Plateau konstanter Fitness* verstanden werden. Die Suchpunkte $x \in \{0,1\}^n$ mit $m(x) = 0$ dürfen dann als Bereich mit konstantem Funktionswert 0, der bezüglich der in Definition 2.3.5 definierten Nachbarschaftsrelation zusammenhängend ist, verstanden werden. Bei randomisierten Suchheuristiken ist das Verhalten auf Plateaus von besonderem Interesse (Jansen und Wegener, 2001a). Für den (1+1)-EA ist u. a. bekannt, dass er spezielle Plateaus polynomieller Größe in erwarteter polynomieller Zeit durchsucht. Unsere Ergebnisse zeigen etwas Ähnliches, da die obere Schranke aus Theorem 3.2.1 für polynomiell große Werte von 2^d, d. h. die Zahl verschiedener Belegungen für $m(x)$, ebenfalls polynomiell ist. Auf andere polynomiell große Plateaus werden wir im zweiten Teil der Dissertation zurückkommen.

3.3 Das Verhalten einer lokalen Suchheuristik

In diesem Abschnitt werden wir das Verhalten der lokalen Suchheuristik RLS auf monotonen Polynomen analysieren. Wie sich im folgenden Abschnitt herausstellen wird, gelingt dabei die Herleitung einer asymptotisch exakten oberen Schranke für die erwartete Laufzeit. Darüber hinaus werden hier bereits einige Begriffe und Techniken eingeübt, die später in Abschnitt 3.5 bei der Analyse der beiden globalen Suchheuristiken wiederkehren.

Sei f ein monotones Polynom und $m(x) = w_m x_{i_1} \cdots x_{i_d}$ eines seiner Monome. Bezüglich einer Belegung $a \in \{0,1\}^n$ nennen wir m *aktiv*, wenn a alle Variablen x_{i_j}, $1 \le j \le d$, mit 1 belegt, das Monom also einen Beitrag w_m zum Funktionswert $f(a)$ leistet. Andernfalls sprechen wir davon, dass m *passiv* sei. Die angenehme Eigenschaft eines Laufes von RLS auf monotonen Polynomen besteht nun darin, dass Monome, die aktiv sind, im Folgenden nicht mehr deaktiviert werden können. Dieses gilt, da pro Schritt genau ein Bit kippt und eine solche Deaktivierung zwangsläufig zu einem geringeren f-Wert führte.

Zwei Monome m_1 und m_2 überlappen einander, wenn sie mindestens eine Variable gemeinsam enthalten. Denkbare komplizierte Überlappungsstrukturen von Monomen werden die Analyse der globalen Suchheuristiken erschweren, auch wenn Überlappungen die Optimierung monotoner Polynome intuitiv erleichtern sollten. Angenommen, m_1 und m_2 seien jeweils vom Grad d und überlappen einander in i Variablen. Wenn m_1 aktiv ist, sind bereits i Variablen von m_2 auf ihren richtigen Wert gesetzt und es genügt, eine „Verkürzung“ von m_2 mit Grad $d - i$ zu aktivieren. Nichtsdestoweniger betrachten wir zunächst eine spezielle Menge von Monomen, die keine Überlappungen aufweisen.

Lemma 3.3.1 *Sei f ein monotones Polynom mit einem Grad von höchstens d und sei M eine Teilmenge der Monome von f, sodass keine zwei Monome aus M einander überlappen. Dann ist die erwartete Zeit, bis RLS alle Monome aus M aktiviert hat, höchstens $O((n/d)\log(n/d+1)2^d)$.*

Beweis: Wenn ein Monom außerhalb der Menge M aktiviert wird, sind einige Variablen bereits dauerhaft auf den Wert 1 festgelegt, da RLS keine Monome deaktiviert. Damit kann die Aktivierung der Monome aus M nur beschleunigt werden. Im Folgenden nehmen wir daher o. B. d. A. an, dass f ausschließlich die Monome aus M enthält.

Laut Theorem 3.2.1 aktiviert RLS jedes Monom vom Grad i in höchstens $3\mathrm{e}(n/i)2^i$ Schritten. Wir behaupten, dass RLS mit einer Wahrscheinlichkeit von mindestens $1/2$ innerhalb von $s := 2c(n/d)(\log(n/d)+1)2^d = O((n/d)\log(n/d+1)2^d)$ Schritten alle Monome aus M aktiviert, wenn die Konstante $c > 0$ passend gewählt ist. Mit einer Wiederholung unabhängiger Phasen (vgl. Seite 22) folgt daraus das Lemma.

Eine kleine Rechnung (analog zu Lemma 3.3.3) zeigt, dass $\ell_i := (n/i)(\log(n/i)+1)2^i$ zumindest ab $i \geq 4$ monoton wachsend in i und $\ell_1 = O(\ell_4)$ ist. Folglich können wir die Konstante c so wählen, dass der betrachtete Zeitabschnitt von s Schritten für $1 \leq i \leq d$ mindestens $\lceil \log(n/i)+1 \rceil$ Phasen mit einer Länge von jeweils $\lceil 6\mathrm{e}(n/i)2^i \rceil$ enthält. Innerhalb einer Phase wird jedes beliebig gewählte Monom vom Grad i gemäß der Markoffungleichung mit einer Wahrscheinlichkeit von mindestens $1/2$ aktiv. Damit ist die Wahrscheinlichkeit, dass es nicht innerhalb von s Schritten aktiv wird, höchstens $(1/2)^{\lceil \log(n/i)+1 \rceil} \leq i/(2n)$. Sei c_i die Anzahl der Monome vom Grad i aus M. Dann ist die Wahrscheinlichkeit, dass es mindestens ein Monom aus M gibt, das nicht innerhalb der s Schritte aktiv wird, höchstens $\sum_{i=1}^{d} c_i \cdot i/(2n)$. Da keine zwei Monome aus M einander überlappen, ist $\sum_{i=1}^{d} c_i \cdot i \leq n$ und die betrachtete Fehlerwahrscheinlichkeit höchstens $1/2$. Damit ist die Behauptung und somit das Lemma gezeigt. □

Mit Lemma 3.3.1 kommen wir zur allgemeinen oberen Schranke für RLS auf monotonen Polynomen.

Theorem 3.3.2 *Die erwartete Laufzeit von RLS auf einem monotonen Polynom vom Grad d beträgt höchstens $O((n/d)\log(n/d+1)2^d)$.*

Beweis: Die Beweisidee besteht darin, Lemma 3.3.1 iterativ für speziell gewählte Mengen M_i anzuwenden. Sei $f_0 := f$ und sei M_0 eine maximale, d. h. nicht vergrößerbare Menge paarweise nicht überlappender Monome aus f_0. Nachdem wir $M_0, \ldots, M_{i-1}$ gewählt haben, betrachten wir die *Verkürzung* f_i von f, die wie folgt definiert ist. Wir erhalten f_i als diejenige Subfunktion von f, in der alle Variablen, die in mindestens einem der Monome aus $M_0 \cup \cdots \cup M_{i-1}$ enthalten sind, auf 1 gesetzt werden. Wenn damit f_i zu einer konstanten Funktion wird, ist das iterative Verfahren beendet. Es ist offensichtlich, dass dieses nach spätestens n Schritten der Fall ist.

Sei T_i die erwartete Zeit, bis RLS alle Monome aus M_i aktiviert. Da wir in jeder Iteration eine maximale Menge nicht überlappender Monome wählen, ersetzen wir jeweils mindestens eine Variable jedes noch vorhandenen Monoms durch 1. Damit folgt, dass der Grad jedes Monoms aus M_i höchstens $d-i$ beträgt und das Verfahren nach spätestens d Iterationen terminiert. Mit Lemma 3.3.1 folgt

$$T_i \;\le\; c \cdot \frac{n}{d-i} \log\left(\frac{n}{d-i}+1\right) 2^{d-i}$$

für eine Konstante $c > 0$ und die erwartete Laufzeit ist damit insgesamt höchstens

$$\sum_{i=0}^{d-1} c \cdot \frac{n}{d-i} \log\left(\frac{n}{d-i}+1\right) 2^{d-i}.$$

Laut Lemma 3.3.3 kann der Quotient der Summenglieder für i und $i+1$ im Fall $i \le d-5$ durch $6/7$ und sonst durch $5/2$ nach oben abgeschätzt werden. Also ist die erwartete Laufzeit höchstens

$$\sum_{i=0}^{d-4} c \cdot \frac{n}{d} \log\left(\frac{n}{d}+1\right) 2^d \left(\frac{6}{7}\right)^i + O\left(\frac{n}{d} \log\left(\frac{n}{d}+1\right) 2^d\right) \;=\; O\left(\frac{n}{d} \log\left(\frac{n}{d}+1\right) 2^d\right),$$

indem wir die geometrische Reihe durch ihren Grenzwert abschätzen. □

Bevor wir diesen Abschnitt beschließen, liefern wir die angekündigte kleine Rechnung nach.

Lemma 3.3.3 *Es gilt*

$$\frac{1}{2} \cdot \frac{i}{i-1} \cdot \frac{\log\left(\frac{n}{i-1}+1\right)}{\log\left(\frac{n}{i}+1\right)} \;\le\; \begin{cases} 5/2 & \text{für } 2 \le i \le n \text{ und} \\ 6/7 & \text{für } 5 \le i \le n. \end{cases}$$

Beweis: Wir betrachten zunächst den Quotienten der Logarithmen. Da $\ln(x)$ eine konkave Funktion ist, können wir für $0 < a < b$ die Steigung im Intervall $[a, b]$ abschätzen durch die Ableitung im Punkt a, d. h. $1/a$. Somit gelten für $a > 1$ die Implikationen

$$\frac{\ln(b) - \ln(a)}{b-a} \;\le\; \frac{1}{a} \quad\Rightarrow\quad \ln(b) \;\le\; \ln(a) + \frac{b-a}{a} \quad\Rightarrow\quad \frac{\ln(b)}{\ln(a)} \;\le\; 1 + \frac{b-a}{a\ln(a)}.$$

Es folgt

$$\frac{\log\left(\frac{n}{i-1}+1\right)}{\log\left(\frac{n}{i}+1\right)} \;=\; \frac{\ln\left(\frac{n}{i-1}+1\right)}{\ln\left(\frac{n}{i}+1\right)} \;\le\; 1 + \frac{n \cdot \frac{1}{i(i-1)}}{\left(\frac{n}{i}+1\right)\ln\left(\frac{n}{i}+1\right)} \;\le\; 1 + \frac{1}{(i-1)\ln\left(\frac{n}{i}+1\right)}.$$

Da $i \le n$ gilt, können wir den letzten Ausdruck durch $u(i) := 1 + 1/((\ln 2)(i-1))$ nach oben abschätzen. Es ist $u(i) \le 5/2$ für $i \ge 2$ und $u(i) \le 48/35$ für $i \ge 5$. Also ist

$$\frac{1}{2} \cdot \frac{i}{i-1} \cdot \frac{\log\left(\frac{n}{i-1}+1\right)}{\log\left(\frac{n}{i}+1\right)} \;\le\; \frac{i \cdot u(i)}{2(i-1)} \;\le\; \begin{cases} 5/2 & \text{für } i \ge 2 \text{ und} \\ (48/35) \cdot (5/8) = 6/7 & \text{für } i \ge 5 \end{cases}$$

wie behauptet. □

3.4 Ein Worst-Case-Beispiel

Im Rahmen dieses Abschnittes werden wir zeigen, dass die in Theorem 3.3.2 hergeleitete obere Schranke auf manchen Instanzen asymptotisch scharf ist. Im Fall $d = \Omega(n)$ entspricht die Schranke $O(2^d)$ und Theorem 3.2.1 zeigt sogar für alle in diesem Kapitel studierten Suchheuristiken eine asymptotisch scharfe untere Schranke $\Omega(2^d)$ für die erwartete Laufzeit. Sei nun der wesentlichere Fall $d = o(n)$ betrachtet. Wir interessieren uns für ein monotones Polynom, bei dem die erwartete Laufzeit für RLS (und auch für RLS_p mit $p \leq 1/n$ und den (1+1)-EA) durch den Ausdruck $\Omega((n/d)\log(n/d+1)2^d)$ nach unten beschränkt ist. Der Beweis der oberen Schranke legt nahe, dass der Fall vieler nicht überlappender Monome vom Grad d schwierig ist. Daher partitionieren wir die Variablenmenge $\{x_1, \ldots, x_n\}$ in k disjunkte Blöcke mit jeweils d Variablen und nehmen o. B. d. A. $n = kd$ an. In unserem Worst-Case-Beispiel soll der Funktionswert die Zahl der Blöcke angeben, in denen alle Variablen 1 sind. Es sei

$$\mathrm{RR}_d(x) \ := \ \sum_{i=0}^{k-1} x_{id+1} \cdots x_{id+d},$$

d. h., RR_d ist ein monotones Polynom mit k paarweise nicht überlappenden Monomen vom Grad d. Diese Funktion wurde in der Literatur über evolutionäre Algorithmen bereits in einem anderen Zusammenhang definiert und dort *Royal-Road-Funktion* genannt. Darauf werden wir in Kapitel 6 an passender Stelle zu sprechen kommen.

Theorem 3.4.1 *Sei $d = o(n)$. Die Wahrscheinlichkeit, dass RLS die Funktion RR_d innerhalb von höchstens $(n/d)\log(n/d)2^d/32$ Schritten optimiert, ist $o(1)$ und die erwartete Laufzeit ist gegeben durch $\Theta((n/d)\log(n/d+1)2^d)$.*

Beweis: Die obere Schranke der erwarteten Laufzeit folgt aus Theorem 3.3.2. Die entsprechende untere Schranke wird jeweils aus der ersten, stärkeren Aussage des Theorems folgen. Zunächst betrachten wir einen einfachen Fall, und zwar $d = O(1)$, wofür eine mit Wahrscheinlichkeit $1 - o(1)$ geltende untere Schranke $\Omega(n \log n)$ zu zeigen ist. Die Laufzeit ist mindestens so groß wie die Zeit, bis jedes mit 0 initialisierte Bit mindestens einmal gekippt ist. Dies ist das Szenario des Sammelbilderproblems (Lemma A.15). Da gemäß der Chernoffungleichung mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ mindestens $n/3$ Bits mit 0 initialisiert werden, gilt die untere Schranke $\Omega(n \log n)$ mit einer Wahrscheinlichkeit von $1 - o(1)$.

Im Folgenden gehen wir von den Beziehungen $d = \omega(1)$ und $d = o(n)$ aus. Sei der Zeitraum der ersten höchstens $k(\log k)2^d/32$ Schritte betrachtet. Uns interessiert die Wahrscheinlichkeit, dass RLS jedes Monom der Funktion RR_d innerhalb dieses Zeitraums aktiviert. Wir erinnern daran, dass die Suchheuristik RLS niemals Monome deaktiviert, da sie nur ein Bit pro Schritt kippt und keine schlechteren Suchpunkte akzeptiert. Wir betrachten ein beliebiges Monom m aus der Darstellung von RR_d

und bezeichnen mit X die Menge der Variablen, auf der m definiert ist. Ein Schritt von RLS heiße *essenziell* bezüglich m, wenn er die Belegung einer Variablen aus X ändert. Die Wahrscheinlichkeit, dass ein Schritt essenziell bezüglich m ist, beträgt genau $d/n = 1/k$. Auch wenn die Zahl essenzieller Schritte für verschiedene Monome nicht unabhängig ist, besteht die weitere Beweisidee darin zu zeigen, dass alle Monome in etwa dieselbe Anzahl essenzieller Schritte erhalten. Eine einfache, aber entscheidende Beobachtung ist, dass RLS alle für m essenziellen Schritte akzeptiert, solange m noch passiv ist. Dies folgt, da Variablen aus X in keinem anderen Monom auftreten. Wenn die Anzahl essenzieller Schritte für ein Monom vorgegeben ist, ist also das Ereignis, dass es innerhalb dieser Schritte aktiviert wird, unabhängig vom Verhalten der nicht essenziellen Schritte. Garnier, Kallel und Schoenauer (1999) haben für den Fall $d = \omega(1)$ gezeigt, dass die Wahrscheinlichkeit, dass ein Monom m vom Grad d innerhalb von $t2^d$ Schritten aktiviert wird, gegen den Ausdruck $1 - \mathrm{e}^{-t}$ konvergiert. Also ist für genügend großes d die Wahrscheinlichkeit, dass m innerhalb von 2^{d-1} essenziellen Schritten aktiv wird, höchstens $1/2$. Gleichzeitig haben die genannten Autoren gezeigt, dass die Zahl essenzieller Schritte τ bezüglich m, bis m aktiv ist, für den Grenzübergang $d \to \infty$ eine gedächtnislose Verteilung besitzt, d. h., für $t, t' \geq 0$ gilt

$$\mathrm{Prob}(\tau \geq t) \;=\; \mathrm{Prob}(\tau \geq t + t' \mid \tau \geq t').$$

Genauer gesagt haben sie sogar nachgewiesen, dass unabhängig von t und t' die Beziehung

$$\mathrm{Prob}(\tau \geq t) - \mathrm{Prob}(\tau \geq t + t' \mid \tau \geq t') \;=\; O(1/d)$$

gilt. Für genügend großes d können wir folgern, dass für alle t' selbst die bedingte Erfolgswahrscheinlichkeit $\mathrm{Prob}(\tau \leq 2^{d-1} + t' \mid \tau \geq t')$ höchstens $1/2$ ist. Diese Eigenschaft werden wir gleich ausnutzen.

Wir wenden jetzt die Beweismethode, einen typischen Lauf zu identifizieren (vgl. Seite 21), an. Wegen $d \geq 2$ ist die Wahrscheinlichkeit, dass ein Monom in der Initialisierung aktiv ist, höchstens $1/4$. Gemäß der Chernoffungleichung ist die Wahrscheinlichkeit, weniger als $k/2$ Monome passiv zu initialisieren, exponentiell klein in Bezug auf k, d. h. $2^{-\Omega(k)}$, und damit $o(1)$. Im Folgenden nehmen wir an, dass mindestens $k/2$ Monome passiv initialisiert werden und führen damit nur einen Fehlerterm der Größe $o(1)$ ein. Wir betrachten $k/2$ passive Monome und unterteilen die $k(\log k)2^d/32$ Schritte in $(\log k)/4$ Phasen mit einer Länge von jeweils $k2^d/8$. Ohne Beschränkung der Allgemeinheit handele es sich dabei um ganze Zahlen.

Sei p_i die zufällige Zahl passiver Monome nach der i-ten Phase. Gemäß unseren Annahmen gilt $p_0 \geq k/2$. Wir behaupten, dass die Wahrscheinlichkeit des Ereignisses, dass $p_i < p_{i-1}/8$ für irgendein $i \leq (\log k)/4$ gilt, exponentiell klein in $k^{1/4}$ ist. Wenn dieses Ereignis nicht eintritt, beträgt die Anzahl passiver Monome am Ende aller $(\log k)/4$ Phasen mindestens

$$p_0 \cdot \left(\frac{1}{8}\right)^{(\log k)/4} \;\geq\; \frac{1}{2} \cdot k \cdot k^{-3/4} \;\geq\; \frac{k^{1/4}}{2}.$$

Um die Behauptung zu zeigen, dürfen wir gemäß der letzten Rechnung unter der Annahme arbeiten, dass $p_{i-1} \geq k^{1/4}/2$ gilt. Die erwartete Zahl von Schritten der i-ten Phase, die essenziell für eins der zu Beginn der Phase passiven Monome sind, beträgt $(p_{i-1}/k) \cdot k2^d/8 = p_{i-1} \cdot 2^d/8$. Gemäß der Chernoffungleichung ist die Wahrscheinlichkeit von mehr als $p_{i-1} \cdot 2^d/4$ solcher Schritte exponentiell klein in p_{i-1} und damit in $k^{1/4}$. Damit ist auch die Wahrscheinlichkeit, dass dieses für mindestens eine Phase der $(\log k)/4$ Phasen eintritt, immer noch exponentiell klein in $k^{1/4}$. Wir nehmen im Folgenden an, dass dieses Ereignis nicht zutrifft. Gemäß unseren Annahmen enthält die i-te Phase höchstens $p_{i-1} \cdot 2^d/4$ essenzielle Schritte für alle p_{i-1} passiven Monome. Nach dem Schubfachprinzip gibt es höchstens $p_{i-1}/2$ passive Monome, die jeweils mindestens $2^d/2$ essenzielle Schritte erhalten. Wir nehmen pessimistisch an, dass diese Monome alle in der i-ten Phase aktiv werden. Gemäß der obigen Überlegung ist für jedes andere Monom die Wahrscheinlichkeit, in der i-ten Phase aktiv zu werden, höchstens 1/2. Damit liefert die Chernoffungleichung, dass die Wahrscheinlichkeit, mindestens einen Anteil von 3/4 der letzteren Monome zu aktivieren, exponentiell klein bezüglich $p_{i-1} \geq k^{1/4}/2$ ist. Also werden in der i-ten Phase insgesamt mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(k^{1/4})}$ höchstens $7p_{i-1}/8$ passive Monome aktiv und da wir nur $(\log k)/4$ Phasen betrachten, gilt dies auch für alle Phasen. $\square$

Auf der Funktion RR_d ist nicht nur die erwartete Laufzeit von RLS durch den in Theorem 3.3.2 genannten Ausdruck $O((n/d)(\log(n/d) + 1)2^d)$ beschränkt, sondern auch die von RLS_p mit $p \leq 1/n$ und die des (1+1)-EA. Ein spezieller Beweis der oberen Schranke für RLS auf RR_d wurde nämlich bereits von Mitchell, Holland und Forrest (1994) vorgestellt. Wir bemerken, dass sich die zugehörigen Beweismethoden leicht auf die anderen Suchheuristiken anwenden lassen. In den folgenden Abschnitten sind wir aber nur an oberen Schranken, die für alle monotonen Polynome vom Grad d gelten, interessiert.

Im Gegensatz zu den oberen Schranken ist es nicht allzu schwierig, die untere Schranke für RLS auf die anderen Suchheuristiken zu übertragen. Unter Berücksichtigung der Bemerkung aus dem vorigen Absatz werden damit monotone Polynome vom Grad d identifiziert, bei denen für alle betrachteten Suchheuristiken asymptotisch scharfe Schranken der erwarteten Laufzeit bekannt sind. Die Beweisidee für die weiteren unteren Schranken besteht darin, die Erkenntnisse aus Theorem 3.4.1 auszunutzen, indem man den Markoffprozess, den RLS auf RR_d beschreibt, durch den zugehörigen Prozess für die andere Suchheuristik ersetzt. Tatsächlich nehmen wir diese Ersetzung nicht unmittelbar vor, sondern führen noch einen „Zwischenprozess“ ein. Die Hauptaufgabe besteht dann darin, die Folgen dieser Ersetzungen in Bezug auf die Laufzeit oder Erfolgswahrscheinlichkeit sorgfältig abzuschätzen. Bei der Laufzeitanalyse evolutionärer Algorithmen geht man häufig auf diese Weise vor. Oft werden komplizierte Markoffprozesse durch einfache, besser zu analysierende Prozesse ersetzt. Beispielhafte Umsetzungen dieser Idee finden sich in den Arbeiten von Droste (2003, 2004). Zudem werden wir dieser Idee in Abschnitt 3.5 folgen, um die ausstehenden oberen Schranken zu zeigen.

Für das folgende Theorem führen wir vorübergehend eine hilfreiche Notation ein. Es bezeichne $T_d(A)$ die Laufzeit der Suchheuristik A auf der Funktion RR_d. Nun setzen wir die Verteilung von $T_d(\mathrm{RLS})$ in Beziehung mit $T_d(\mathrm{RLS}_p)$ und $T_d((1{+}1)\text{-EA})$, indem wir salopp gesagt zeigen, dass die Wahrscheinlichkeit, dass eine der letzteren Suchheuristiken auf RR_d viel schneller als RLS ist, recht klein ist.

Theorem 3.4.2 *Sei $\varepsilon > 0$ eine beliebige Konstante und $p \leq 1/n$. Für jedes $t \geq 0$ mit $t = 2^{o(n \log n)}$ gelten die Abschätzungen*

$$\begin{aligned} \mathrm{Prob}\big(T_d((1{+}1)\text{-EA}) \leq t\big) &\leq \mathrm{Prob}\left(T_d(\mathrm{RLS}) \leq t(1+\varepsilon)\right) + 2^{-\Omega(t)} + 2^{-\Omega(n \log n)} \\ \textit{und } \mathrm{Prob}\big(T_d(\mathrm{RLS}_p) \leq t\big) &\leq \mathrm{Prob}\left(T_d(\mathrm{RLS}) \leq t(2+\varepsilon)\right) + 2^{-\Omega(t)} + 2^{-\Omega(n \log n)}. \end{aligned}$$

Weil RLS laut Theorem 3.4.1 auf RR_d mit $d = o(n)$ mit einer Wahrscheinlichkeit von $1 - o(1)$ mindestens $\Omega((n/d)\log(n/d+1)2^d)$ Schritte braucht, erhalten wir aus Theorem 3.4.2 in diesem Fall die entsprechenden unteren Schranken für die anderen Suchheuristiken. Den Fall $d = \Omega(n)$ haben wir schon zu Beginn dieses Abschnitts diskutiert.

Beweis von Theorem 3.4.2: Zunächst zeigen wir die Aussage über den (1+1)-EA. Dazu ersetzen wir den (1+1)-EA durch einen auf RR_d beweisbar schnelleren Prozess (1+1)-EA* und vergleichen jenen mit RLS. Die Idee besteht dann darin, für jeden zufälligen Lauf des (1+1)-EA* mit Länge t einen äquivalenten und nur geringfügig langsameren Lauf von RLS zu identifizieren, der fast dieselbe Wahrscheinlichkeit hat. Bei der Beschreibung dieses Laufes sprechen wir von einer „Simulation“.

Die Konstruktion des (1+1)-EA* weist starke Ähnlichkeit mit einer Technik auf, die in der Literatur als *coupling* (Lindvall, 2002; Aldous und Fill, 2004) bekannt ist. Wir erhalten einen Schritt des (1+1)-EA* aus einem Schritt des (1+1)-EA mithilfe folgender Modifikationen. Sei x_0 der aktuelle Suchpunkt des (1+1)-EA* zu Beginn des Schrittes. Der (1+1)-EA* wählt zunächst die Zahl k der zu kippenden Bits gemäß derselben Verteilung, gemäß der die Zahl kippender Bits für einen Schritt des (1+1)-EA verteilt ist. Anschließend führt der (1+1)-EA* eine Folge von $k' \geq k$ Unterschritten mit aktuellen Suchpunkten $x_0, x_1, \ldots, x_{k'}$ durch. Dabei erzeugt der i-te Unterschritt den Suchpunkt x', indem genau ein zufällig gleichverteilt gewähltes Bit von x_i kippt, und setzt $x_{i+1} := x'$, wenn $\mathrm{RR}_d(x') \geq \mathrm{RR}_d(x_i)$ gilt, und sonst $x_{i+1} := x_i$. Die Zahl k' bestimmt sich aus dem kleinsten i, sodass der Hammingabstand von x_0 und x_i genau k beträgt. (Man beachte, dass $k' > k$ möglich ist, wenn mehrere Unterschritte dasselbe Bit kippen.) Mit anderen Worten wählt der (1+1)-EA* die Zahl k und die Menge zu kippender Bits wie der (1+1)-EA, kippt aber diese k Bits nicht auf einmal, sondern wendet wiederholt den Mutations- und den Selektionsoperator von RLS an, bis ein Hammingabstand von k erreicht ist. Diese Prozedur zählt nur als ein Schritt des (1+1)-EA*.

Angenommen, der aktuelle Suchpunkt sowohl des (1+1)-EA als auch des (1+1)-EA* seien durch denselben Wert fest vorgegeben und es seien x und x^* die zufälligen aktuellen Suchpunkte nach jeweils einem Schritt des (1+1)-EA bzw. des (1+1)-EA* auf RR_d. Es folgt nun für jedes $a \in \{0,1\}^n$, dass die Beziehung $\mathrm{Prob}(x^* \geq a) \geq \mathrm{Prob}(x \geq a)$ gilt, wenn „$\geq$“ die komponentenweise partielle Ordnung für Vektoren bezeichnet. Dieses gilt, da eine feste Auswahl von k zu kippenden Bits im (1+1)-EA auf RR_d eventuell verworfen wird, da mehr Monome deaktiviert als aktiviert werden, während der (1+1)-EA* auf RR_d sämtliche in der Auswahl auf 1 kippende Bits akzeptiert. Die Behauptung folgt dann induktiv für alle aktuellen Suchpunkte. Da der Suchpunkt 1^n optimal ist, erhalten wir, dass der (1+1)-EA* mit mindestens so hoher Wahrscheinlichkeit wie der (1+1)-EA innerhalb von t Schritten das Optimum erreicht. Um die Aussage des Theorems über $T_d((1{+}1)\text{-EA})$ zu zeigen, können wir statt des (1+1)-EA also den (1+1)-EA* analysieren.

Wir wollen nun mit RLS den (1+1)-EA* simulieren. Wir wissen bereits, dass sich RLS und der (1+1)-EA* nur darin unterscheiden, dass manchmal viele Schritte von RLS nötig sind, um einen Schritt des (1+1)-EA* zu simulieren. Um einen Schritt des (1+1)-EA*, der k Bits kippt, zu simulieren, können sogar mehr als k Schritte von RLS erforderlich sein. Um die Zahl nötiger Schritte abzuschätzen, halten wir zunächst fest, dass für jedes konstante $\delta > 0$ die Wahrscheinlichkeit, dass der (1+1)-EA* δn Bits in einem Schritt kippt, höchstens $2^{-\Omega(n \log n)}$ ist (vgl. Abschnitt 2.3). Die Wahrscheinlichkeit, dass so etwas innerhalb von $t = 2^{o(n \log n)}$ Schritten passiert, ist immer noch $2^{-\Omega(n \log n)}$. Wir nehmen an, dass dies nicht passiert, und führen damit einen Fehlerterm von $2^{-\Omega(n \log n)}$ ein.

Nun brauchen wir nur noch Schritte des (1+1)-EA*, die $k \leq \delta n$ Bits kippen, zu betrachten. Sei a der aktuelle Suchpunkt des (1+1)-EA* vor dem Schritt. Alle zu erreichenden Suchpunkte haben also einen Hammingabstand von höchstens δn zu a. Die Wahrscheinlichkeit, dass das nächste kippende Bit den Hammingabstand zu a erhöht, ist mindestens $1 - \delta$. In t Schritten von RLS erwarten wir $(1-\delta)t$ Schritte, die den Abstand zu a erhöhen, und nur δt Schritte, die ihn verringern, also einen Überschuss von $(1-2\delta)t$ erhöhenden Schritten. Gemäß der Chernoffungleichung haben wir mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(t)}$ einen Überschuss von $(1-3\delta)t$ erhöhenden Schritten. Nun bleibt die Zahl kippender Bits in t Schritten des (1+1)-EA* abzuschätzen. Ohne die Annahme, dass höchstens δn Bits kippen, gilt Folgendes: Das Verhalten des Mutationsoperators in verschiedenen Schritten ist unabhängig und jede Mutation kippt einzelne Bits unabhängig mit Wahrscheinlichkeit $1/n$. Damit können wir wie in Abschnitt 2.3 auf Seite 21 erläutert mit der Chernoffungleichung die Zahl insgesamt kippender Bits in t Schritten beschränken. Die Wahrscheinlichkeit von insgesamt mehr als $(1+\gamma)t$ kippenden Bits ist $2^{-\Omega(t)}$ für jede Konstante $\gamma > 0$ und die Annahme, dass höchstens δn Bits pro Schritt kippen, macht diese Wahrscheinlichkeit nur kleiner. Die Summe aller Fehlerwahrscheinlichkeiten ist bis hier durch $2^{-\Omega(t)} + 2^{-\Omega(n \log n)}$ beschränkt. Wenn kein Fehler eintritt, muss RLS insgesamt höchstens $(1+\gamma)t$ kippende Bits simulieren, wofür $(1+\gamma)t/(1-3\delta)$ Schritte von RLS ausreichen. Indem wir γ

und δ auf solche Weise wählen, dass $(1+\gamma)/(1-3\delta) = 1/(1+\varepsilon)$ gilt, folgt die erste Abschätzung des Theorems.

Für die zweite Abschätzung des Theorems gehen wir genauso vor. Die einzige Änderung besteht darin, dass RLS_p mit $p \leq 1/n$ pro Schritt ein Bit mit Sicherheit und jedes andere mit einer Wahrscheinlichkeit von höchstens $1/n$ kippt. Wir können die Chernoffungleichung auf diese anderen Bits anwenden und erhalten, dass in t Schritten mit einer Wahrscheinlichkeit von $1-2^{-\Omega(t)}$ insgesamt höchstens $(2+\gamma)t$ Bits kippen. □

Angesichts der bisherigen Ergebnisse können wir nun vermuten, dass auch die erwartete Laufzeit des (1+1)-EA und von RLS_p auf einem beliebigen monotonen Polynom vom Grad d durch $O((n/d)\log(n/d+1)2^d)$ beschränkt ist. Der Rest dieses Kapitels wird sich daher oberen Schranken für die beiden globalen Suchheuristiken widmen.

3.5 Das Verhalten globaler Suchheuristiken

3.5.1 Vorüberlegungen

Obere Schranken für RLS_p mit $p > 0$ und den (1+1)-EA herzuleiten ist deutlich schwieriger als für RLS. Der Grund liegt darin, dass die globalen Suchheuristiken mehrere Bits in einem Schritt kippen können. Damit können im Gegensatz zu RLS im Laufe der Zeit aktive Monome wieder deaktiviert werden, da an anderer Stelle passive Monome aktiv werden. Je nach Koeffizienten der Monome kann auch die Gesamtzahl aktiver Monome sinken. Selbst die Analyse der erwarteten Zeit, bis ein einzelnes Monom m aktiv wird, ist deutlich komplizierter als bei RLS. Beispielsweise kann die Asymmetrie eintreten, dass eine Mutation, die zwei Bits aus m von 0 auf 1 kippt und ein Bit aus m von 1 auf 0 kippt, verworfen wird, während der Fall, dass zwei Bits auf 0 kippen und eines auf 1 kippt, akzeptiert wird. Also entsprechen die zufälligen Belegungen vom m zu den Zeitpunkten der Suchheuristik nicht unbedingt der in Abschnitt 3.2 betrachteten einfachen Markoffkette.

Eine erste Überlegung, um die kompliziertere Markoffkette in den Griff zu bekommen, bezieht sich auf die Wahrscheinlichkeit, dass ein Schritt viele Bits eines Monoms kippt. Wir werden nämlich sehen, dass sich die im letzten Absatz angesprochene Asymmetrie bei wenigen kippenden Bits behandeln lässt. Die Wahrscheinlichkeit für viele kippende Bits ist natürlich umso geringer, je kleiner der Grad des Monoms ist. Die Theoreme 3.4.1 und 3.4.2 zeigen nun, dass polynomielle obere Schranken ohnehin nur für $d = O(\log n)$ möglich sind. Damit erscheint es sinnvoll, dass wir uns auf diesen Bereich für d beschränken. Als Erstes wollen wir RLS_p auf einem Monom m vom Grad $O(\log n)$ analysieren. Wie angekündigt, soll die dazugehörige Markoffkette untersucht und durch einfachere Markoffketten ersetzt werden. Die entsprechende Analyse fußt auf einigen technischen Definitionen und Lemmata. Im Folgenden stellen wir die Technik zunächst vor, verschieben aber einige Beweise auf Abschnitt 3.5.3, um einigermaßen rasch die eigentliche Anwendung bei den oberen Schranken für RLS_p präsentieren zu können.

Sei wie in Abschnitt 3.2 eine Markoffkette auf dem Zustandsraum $D = \{0, \dots, d\}$ betrachtet, welche die Zahl der Einsen in einem Monom m vom Grad d während des Laufes von RLS_p bzw. des (1+1)-EA widerspiegelt. Da wir Phasen betrachten wollen, in denen wenige Bits von m kippen, betrachten wir die Mutationsoperatoren unter speziellen Bedingungen. Sei $Q(i, j)$ die Übergangswahrscheinlichkeit von Zustand i nach Zustand j für den (1+1)-EA unter der Bedingung, dass pro Schritt höchstens zwei Bits von m kippen. Sei $R(i, j)$ die entsprechende Übergangswahrscheinlichkeit für RLS_p.

Die erste Eigenschaft der Markoffketten besteht darin, dass es wahrscheinlicher ist, vom Zustand i aus höhere Zustände zu erreichen als vom Zustand $i - 1$. Wir sprechen dann von einem *Vorteil.*

Definition 3.5.1 *Seien $P(i, j)$ die Übergangswahrscheinlichkeiten einer zeithomogenen Markoffkette auf $D = \{0, \dots, d\}$. Die Markoffkette hat einen ε-*Vorteil*, $\varepsilon > 0$, wenn für alle $0 \le i \le d - 2$ die folgenden Bedingungen erfüllt sind.*

1. $P(i, j) \;\ge\; (1 + \varepsilon) \cdot P(i + 1, j)$ *für* $j \le i$,

2. $P(i + 1, j) \;\ge\; (1 + \varepsilon) \cdot P(i, j)$ *für* $j > i$.

Borisovsky und Eremeev (2003) definieren den Begriff eines *monotonen Mutationsoperators*, um den (1+1)-EA auf speziellen Problemen in Bezug zu setzen zu komplexeren randomisierten Suchheuristiken. Wir weisen darauf hin, dass jener Begriff im Zustandsraum D mit einem Spezialfall eines 0-Vorteils zusammenfällt.

Die Beweise der in diesem Unterabschnitt folgenden Lemmata holen wir in Abschnitt 3.5.3 nach.

Lemma 3.5.2 *Sei $\varepsilon > 0$ und $d \le (n - 1)/(1 + \varepsilon)$. Dann besitzt die Markoffkette mit den Übergangswahrscheinlichkeiten $Q(i, j)$ einen ε-Vorteil.*

Lemma 3.5.3 *Sei $\varepsilon > 0$ und $d \le (n - 1)/(3 + 2\varepsilon)$. Dann besitzt die Markoffkette mit den Übergangswahrscheinlichkeiten $R(i, j)$ einen ε-Vorteil.*

Wir werden die Eigenschaft eines Vorteils jetzt mit den Übergangszeiten in Verbindung bringen. Sei $\tau^{(k)}$ der erste Zeitpunkt (die *first hitting time*), an dem sich eine zeithomogene Markoffkette Y mit Zustandsraum D im Zustand d befindet, wenn sie im Zustand k startet. Im Falle, dass Y einen 0-Vorteil besitzt, sollte es vorteilhaft sein, in einem „höheren“ Zustand zu starten. Dieses erfasst das folgende Lemma, in dem wir im Wesentlichen die stochastische Dominanz von $\tau^{(i)}$ bezüglich $\tau^{(i+1)}$ zeigen.

Lemma 3.5.4 *Seien $P(i,j)$ die Übergangswahrscheinlichkeiten einer zeithomogenen Markoffkette mit* 0*-Vorteil auf $D = \{0,\dots,d\}$. Dann gilt*

$$\mathrm{Prob}(\tau^{(i)} \geq t) \;\geq\; \mathrm{Prob}(\tau^{(i+1)} \geq t) \quad \textit{und} \quad \mathrm{E}\big(\tau^{(i)}\big) \;\geq\; \mathrm{E}\big(\tau^{(i+1)}\big)$$

für $0 \leq i \leq d-1$ und $t \geq 0$.

Eine wesentliche Überlegung der zu zeigenden oberen Schranken für die erwartete Laufzeit von RLS_p und des (1+1)-EA besteht darin, dass wir die komplizierte Markoffkette, die die Anzahl der Einsen in einem Monom eines monotonen Polynoms mit vielen Monomen beschreibt, mit der einfachen Markoffkette vergleichen, die das Verhalten auf einem Polynom mit nur einem Monom vorgibt. Wie bereits angekündigt, wollen wir damit die Resultate aus Abschnitt 3.2 übertragen. Seien für zwei zeithomogene Markoffketten Y_0 und Y_1 auf D mit $\tau_0^{(i)}$ bzw. $\tau_1^{(i)}$ die ersten Zeitpunkte bezeichnet, an denen die Markoffketten Y_0 bzw. Y_1 den Zustand d erreichen, gegeben, dass sie im Zustand i starten.

Definition 3.5.5 *Seien $P_0(i,j)$ und $P_1(i,j)$ die Übergangswahrscheinlichkeiten der zeithomogenen Markoffketten Y_0 und Y_1 auf $D = \{0,\dots,d\}$. Die Markoffkette Y_1 hat einen* Vorteil bezüglich Y_0*, wenn $P_1(i,j) \geq P_0(i,j)$ für $j \geq i+1$ und $P_1(i,j) \leq P_0(i,j)$ für $j \leq i-1$ gilt.*

Lemma 3.5.6 *Wenn Y_1 einen Vorteil bezüglich Y_0 hat und Y_0 einen* 0*-Vorteil hat, dann gelten die Beziehungen*

$$\mathrm{Prob}(\tau_1^{(i)} \geq t) \;\leq\; \mathrm{Prob}(\tau_0^{(i)} \geq t) \quad \textit{und} \quad \mathrm{E}\big(\tau_1^{(i)}\big) \;\leq\; \mathrm{E}\big(\tau_0^{(i)}\big)$$

für $0 \leq i \leq d-1$ und $t \geq 0$.

Tatsächlich werden wir etwas Schwächeres als in Definition 3.5.5 gefordert zeigen. In der komplizierten Markoffkette Y_1 wird gegenüber Y_0 die Wahrscheinlichkeit, vom Zustand i in den Zustand $i+j$ zu springen, um einen gewissen Faktor verringert sein. Andererseits wird auch die Wahrscheinlichkeit, von i nach $i-j$ zu springen, um denselben Faktor oder stärker verringert sein. Dem wollen wir mit folgender Definition Rechnung tragen.

Definition 3.5.7 *Seien $P_0(i,j)$ und $P_1(i,j)$ die Übergangswahrscheinlichkeiten der zeithomogenen Markoffketten Y_0 und Y_1 auf $D = \{0,\dots,d\}$ und seien die Werte $c(i,j)$ so gewählt, dass $P_1(i,j) = c(i,j)\cdot P_0(i,j)$ für $i,j \in D$ gilt. Dann hat Y_1 einen* relativen Vorteil bezüglich Y_0*, wenn die folgenden Beziehungen gelten:*

$$\begin{aligned} c(i,j) \;&\geq\; c(i,i+1) \quad \textit{für } j \geq i+1, \\ c(i,j) \;&\leq\; c(i,i+1) \quad \textit{für } j \leq i-1, \\ \textit{und} \quad 0 \;<\; c(i,i+1) \;&\leq\; 1 \quad \textit{für } 0 \leq i \leq d-1. \end{aligned}$$

Man beachte, dass die Werte $c(i,j)$ in Definition 3.5.7 beliebig gewählt sein können, falls $P_0(i,j) = P_1(i,j) = 0$ gilt.

Es folgt nun das wichtigste und letzte Lemma dieses Unterabschnitts. Seine intuitive Aussage besteht darin, dass die Laufzeit unter gewissen Bedingungen höchstens um den Kehrwert des kleinsten Faktors wachsen kann, um den bei der Definition eines relativen Vorteils die Übergangswahrscheinlichkeiten vom Zustand i zum nächsthöheren Zustand verringert wurden.

Lemma 3.5.8 *Angenommen, Y_1 hat einen relativen Vorteil bezüglich Y_0 und es ist $c^* := 1/\min\{c(i,i+1) \mid 0 \leq i \leq d-1\}$. Wenn Y_0 außerdem einen (c^*-1)-Vorteil hat, gilt $\mathrm{E}\big(\tau_1^{(i)}\big) \leq c^* \cdot \mathrm{E}\big(\tau_0^{(i)}\big)$ für $0 \leq i \leq d-1$.*

3.5.2 Eine obere Schranke für die Suchheuristik RLS$_p$

Im Folgenden werden wir eine obere Schranke für die erwartete Laufzeit von RLS$_p$ auf monotonen Polynomen zeigen. Als Erstes interessiert die erwartete Zeit, bis ein einzelnes Monom aktiv ist. Dabei können wir die bestmögliche obere Schranke von $O((n/d)2^d)$ zeigen, wenn p klein genug ist. Wie im vorigen Unterabschnitt motiviert, konzentrieren wir uns auf den Fall $d = O(\log n)$.

Lemma 3.5.9 *Sei f ein monotones Polynom vom Grad $d \leq c \log n$ für eine Konstante c und sei m eins seiner Monome. Dann existiert eine Konstante $\alpha > 0$, sodass RLS$_p$ mit*

$$p \;\leq\; \min\left\{\frac{1}{n}, \frac{\alpha}{n^{c/2}\log n}\right\}$$

das Monom m in einer erwarteten Zeit von $O((n/d)2^d)$ aktiviert.

Beweis: Die Beweisidee besteht darin zu zeigen, dass RLS$_p$ das Monom m mit einer konstanten Wahrscheinlichkeit von $\varepsilon > 0$ für eine geeignete Konstante $c' > 0$ innerhalb einer Phase von $c'(n/d)2^d$ Schritten aktiviert. Da unsere Analyse keine Annahmen über die anfängliche Belegung von m machen wird, erhalten wir mit einer Wiederholung unabhängiger Phasen eine obere Schranke von $c'(n/d)2^d/\varepsilon = O((n/d)2^d)$ für die erwartete Zeit, bis m aktiv ist.

Wir betrachten folgende unerwünschte Ereignisse als Fehler und schätzen die zugehörigen Fehlerwahrscheinlichkeiten ab:

- Es gibt einen Schritt in der Phase, in dem mindestens drei Bits von m kippen,
- unter der Bedingung, dass der erste Typ von Fehler nicht eintritt, erzeugt RLS$_p$ in der Phase keinen Suchpunkt, in dem m aktiv ist,
- der erste erzeugte Suchpunkt, in dem m aktiv ist, wird nicht akzeptiert.

Wenn keiner der drei Fehler eintritt, wird m in der Phase aktiv. Der erste und der dritte Fehlertyp werden sich mithilfe gängiger Abschätzungen behandeln lassen. Den zweiten Fehlertyp werden wir analysieren, indem wir RLS_p auf der Funktion f (unter der Annahme, dass jeder Schritt höchstens zwei Bits von m kippt) mit RLS_p auf dem einzelnen Monom m vergleichen. Wie angekündigt, ziehen wir deshalb die Techniken aus Abschnitt 3.5.1 heran. Im Folgenden gehen wir von der Gestalt $m(x) = x_1 \cdots x_d$ aus. Die oberen Schranken für die Fehlerwahrscheinlichkeiten werden nämlich nur geringer, wenn m einen geringeren Grad als d hat.

Der erste Typ von Fehler tritt ein, wenn das erste kippende Bit und zwei zusätzlich kippende Bits einer Mutation von RLS_p zu m gehören oder drei zusätzlich kippende Bits zu m gehören. Also ist die Fehlerwahrscheinlichkeit in einem Schritt höchstens

$$\frac{d}{n}\binom{d-1}{2}p^2 + \frac{n-d}{n}\binom{d}{3}p^3 \;\le\; d^3 \cdot \frac{p^2}{n}$$

und die Wahrscheinlichkeit mindestens eines solchen Fehlers in der Phase somit höchstens

$$c' \cdot \frac{n}{d} \cdot 2^d \cdot d^3 \cdot \frac{p^2}{n} \;=\; c' 2^d d^2 p^2 \;\le\; c' n^c c^2 (\log n)^2 \alpha^2 n^{-c} (\log n)^{-2}.$$

Indem wir α als genügend kleine Konstante wählen, können wir den letzten Ausdruck durch jede beliebige positive Konstante nach oben beschränken.

Damit ein Fehler des dritten Typs auftritt, ist es erforderlich, dass die Mutation, die m aktiviert, auch mindestens eins der so genannten Suffixbits $x_{d+1}, \ldots, x_n$ kippt. Weil wir unter der Annahme arbeiten, dass der Schritt m aktiviert, setzen wir voraus, dass genau die Bits von m, die vor dem Schritt mit 0 belegt sind, kippen. Ohne Beschränkung der Allgemeinheit seien diese Bits $x_1, \ldots, x_j$ mit $j \ge 1$. Wir können nun die Wahrscheinlichkeit q, dass der betrachtete Fehler nicht eintritt, geeignet abschätzen. Dazu sei A_i das Ereignis, dass das i-te Bit in dem Schritt kippt. Wir erhalten

$$\begin{aligned} q \;&=\; \mathrm{Prob}(\overline{A}_{d+1} \cap \cdots \cap \overline{A}_n \mid A_1 \cap \cdots \cap A_j \cap \overline{A}_{j+1} \cap \cdots \cap \overline{A}_d) \\ &=\; \frac{\mathrm{Prob}(A_1 \cap \cdots \cap A_j \cap \overline{A}_{j+1} \cap \cdots \cap \overline{A}_d \cap \overline{A}_{d+1} \cap \cdots \cap \overline{A}_n)}{\mathrm{Prob}(A_1 \cap \cdots \cap A_j \cap \overline{A}_{j+1} \cap \cdots \cap \overline{A}_d)}. \end{aligned}$$

Der Zähler dieses Bruches ist

$$\frac{j}{n} \cdot p^{j-1}(1-p)^{n-j} \;\ge\; \frac{j}{n} \cdot p^{j-1}\mathrm{e}^{-1}$$

und der Nenner ist

$$\frac{j}{n} \cdot p^{j-1}(1-p)^{d-j} + \frac{n-d}{n} \cdot p^j (1-p)^{d-j} \;\le\; 2 \cdot \frac{j}{n} \cdot p^{j-1}.$$

Folglich ist die Wahrscheinlichkeit, dass dieser Typ von Fehler nicht eintritt, mindestens $1/(2\mathrm{e})$.

Wir kommen nun zum zweiten Fehlertyp und damit zur Anwendung von Lemma 3.5.8. Sei mit RLS_p^* diejenige Suchheuristik bezeichnet, die aus RLS_p entsteht, wenn wir fordern, dass jeder Schritt höchstens zwei Bits von m kippt. Dann definieren wir Y_0 als die Markoffkette mit Zustandsraum $D = \{0, \ldots, d\}$, die sich aus dem Verhalten von RLS_p^* auf dem einzelnen Monom m ergibt und die Zahl der Einsbits in m beschreibt. Man beachte, dass es sich bei Y_0 um die Markoffkette mit Übergangswahrscheinlichkeiten $R(i, j)$ aus Abschnitt 3.5.1 handelt. Gleicherweise sei Y_1 für RLS_p^* auf dem monotonen Polynom f definiert. Es ergibt sich aber das Problem, dass Y_1 dann nicht mehr zeithomogen ist, da die Übergangswahrscheinlichkeiten von Y_1 zum Zeitpunkt t von der Belegung der Suffixbits des Suchpunktes, den RLS_p^* dann vorhält, abhängen. Gedanklich ersetzen wir zu jedem betrachteten Zeitpunkt daher den aktuellen Suchpunkt $a = (b, c^*)$ mit Präfix $(b_1, \ldots, b_d)$, also Belegung von m, und Suffix c^* durch einen Suchpunkt (b, c^{**}) mit eventuell modifiziertem Suffix c^{**}, von dem ausgehend die Wahrscheinlichkeit, m in der Phase zu aktivieren, am geringsten ist. Mit diesem Kunstgriff erhalten wir eine zeithomogene Markoffkette, die wir immer noch mit Y_1 bezeichnen.

Als Nächstes wollen wir folgende Bedingungen für Lemma 3.5.8 nachweisen:

- Y_1 hat einen relativen Vorteil bezüglich Y_0 mit $c(i, j)$-Werten, sodass die Abschätzung $\min_{i=0}^{d-1}\{c(i, i + 1)\} \geq 1/(2\mathrm{e})$ gilt,
- Y_0 hat einen $(2\mathrm{e} - 1)$-Vorteil,
- $\mathrm{E}\bigl(\tau_0^{(i)}\bigr) = O((n/d)2^d)$ für $0 \leq i \leq d$.

Wenn diese Behauptungen gelten, liefert Lemma 3.5.8 $\mathrm{E}\bigl(\tau_1^{(i)}\bigr) = O((n/d)2^d)$, sodass sich mit der Markoffungleichung die Wahrscheinlichkeit, dass der zweite Typ von Fehler innerhalb von $c'(n/d)2^d$ Schritten für eine geeignete Wahl von c' eintritt, durch jede beliebig kleine Konstante nach oben beschränken lässt.

Wir zeigen nun die drei Behauptungen. Die dritte Behauptung folgt aus Theorem 3.2.1. In Lemma 3.5.3 ist der als Zweites geforderte $(2\mathrm{e} - 1)$-Vorteil für $d \leq (n - 1)/(4\mathrm{e} + 1)$ enthalten, was für genügend große n wegen $d = O(\log n)$ gilt. Es bleibt die erste Behauptung nachzuweisen. Da höchstens zwei Bits von m kippen, müssen wir gemäß Definition 3.5.7 lediglich $c(i, j)$-Werte mit $c(i, j) = P_1(i, j)/P_0(i, j)$ für $j \in \{i - 2, i - 1, i + 1, i + 2\}$ betrachten und für $i \leq d - 1$ beweisen, dass

- $1/(2\mathrm{e}) \leq c(i, i + 1) \leq 1$,
- $c(i, i + 2) \geq c(i, i + 1)$,
- $c(i, i - 1) \leq c(i, i + 1)$ sowie
- $c(i, i - 2) \leq c(i, i + 1)$ (oder gar $c(i, i - 2) \leq c(i, i - 1)$).

Da RLS_p^* auf m jeden neuen Suchpunkt akzeptiert, solange das Optimum noch nicht erreicht ist, folgt $P_1(i,i+1) \le P_0(i,i+1)$ und damit $c(i,i+1) \le 1$. Weil RLS_p^* auf f jeden Schritt, der genau ein Nullbit von m kippt und das Suffix unverändert lässt, akzeptiert, können wir die obigen Rechnungen zum dritten Fehlertyp wieder verwerten und erhalten $c(i,i+1) = P_1(i,i+1)/P_0(i,i+1) \ge 1/(2\mathrm{e})$.

Wir kommen jetzt zu den letzten drei Ungleichungen für die $c(i,j)$-Werte. Sei $a = (b,c)$ ein Suchpunkt mit i Einsbits in m, also in b. Wenn a der aktuelle Suchpunkt von RLS_p^* ist, gibt $P_0(i,j)$ die Wahrscheinlichkeit an, dass RLS_p^* aus a einen Suchpunkt mit j Präfixeinsen, d. h. Einsbits im Präfix, erzeugt. Andererseits ist $P_1(i,j)$ die Wahrscheinlichkeit, dass RLS_p in dieser Situation einen Suchpunkt mit j Präfixeinsen erzeugt und diesen bezüglich der Zielfunktion f akzeptiert. Damit ist $c(i,j)$ die bedingte Wahrscheinlichkeit, dass RLS_p auf f einen zufälligen Suchpunkt a' akzeptiert, gegeben, dass dieser j Präfixeinsen hat und durch Mutation von a entstanden ist. Wir wenden den Satz von der totalen Wahrscheinlichkeit auf das Suffix c' von a' an, indem wir die drei Ungleichungen für jede feste Belegung von c' beweisen. Sei also eine beliebige Belegung c' für das Suffix von a' vorausgesetzt. Sei b_ℓ, $1 \le \ell \le d-i$, dasjenige Präfix, das wir aus b erhalten, indem wir dessen ℓ-tes Nullbit kippen. Dies ist die einzige Möglichkeit, in den Zustand $i+1$ zu gelangen, da höchstens zwei Bits von m kippen. Unter den vorgenommenen Bedingungen folgt $c(i,i+1) = k/(d-i)$, wenn k die Anzahl der ℓ ist, für die RLS_p^* auf f den Suchpunkt (b_ℓ, c') akzeptiert. Sei weiterhin $b_{\ell,\ell'}$ mit $1 \le \ell, \ell' \le d-i$ und $\ell \ne \ell'$ dasjenige Präfix, das wir aus b enthalten, wenn wir dessen ℓ-tes und ℓ'-tes Nullbit kippen. Dann sei k^* die Anzahl der Paare (ℓ,ℓ') mit $\ell < \ell'$, sodass RLS_p^* $(b_{\ell,\ell'}, c')$ akzeptiert. Wir erhalten $c(i,i+2) = k^*/\binom{d-i}{2}$.

Wenn RLS_p^* (b_ℓ, c') für ein ℓ akzeptiert, dann auch jedes $(b_{\ell,\ell'}, c')$ mit $\ell' \ne \ell$, denn der f-Wert des letzteren Suchpunktes kann nur größer sein. Also ist $k^* \ge k(d-i-1)/2$. Daraus folgt die Ungleichung $c(i,i+2) \ge c(i,i+1)$. Auf duale Weise zeigt man die Beziehung $c(i,i-2) \le c(i,i-1)$. Um schließlich die Ungleichung $c(i,i-1) \le c(i,i+1)$ zu zeigen, führen wir eine kleine Fallunterscheidung durch. Wenn bereits der Suchpunkt (b,c'), d. h. mit unverändertem Präfix, akzeptiert wird, gilt $c(i,i+1) = 1$, da zum Erreichen des Zustands $i+1$ genau ein Nullbit im Präfix kippt und alle Suchpunkte (b_ℓ, c') einen mindestens so hohen f-Wert wie (b,c') haben. Es folgt $c(i,i-1) \le 1 = c(i,i+1)$. Wenn (b,c') verworfen wird, gilt $c(i,i-1) = 0$, da zum Erreichen des Zustands $i-1$ genau ein Einsbit im Präfix kippt, was den f-Wert nur verringern kann. Es folgt $c(i,i-1) = 0 \le c(i,i+1)$. Damit sind die Behauptungen gezeigt. Weil die Summe aller Fehlerwahrscheinlichkeiten durch $1-\varepsilon$ beschränkt werden kann, haben wir insgesamt das Theorem bewiesen. □

Das Ergebnis aus Lemma 3.5.9 soll nun schließlich benutzt werden, um die erwartete Laufzeit von RLS_p auf monotonen Polynomen zu beschränken. Wir haben schon erwähnt, dass RLS_p bereits aktive Monome in Schritten, die mehr als ein Bit kippen, wieder deaktivieren kann und dass sogar die Zahl aktiver Monome sinken kann, da an anderer Stelle Monome mit hohem Gewicht aktiv werden. Wir möchten nun beispielsweise die Zahl aktiver Monome zum aktuellen Zeitpunkt als Potenzial auffassen, das den Zustand von RLS_p vergröbert und auf eine natürliche Zahl abbildet. Diese Sichtweise kann man, wie in Abschnitt 2.3 erwähnt, bei der Anwendung der Ranggruppenmethode einnehmen, indem man den Index der Ranggruppe, in der sich der aktuelle Suchpunkt befindet, als Potenzial bezeichnet. Bei unserem Beispiel der Zahl aktiver Monome sehen wir uns dem Problem gegenüber, dass das Potenzial auch sinken kann. Die Hoffnung lautet aber, dass wir dennoch für ein anderes Potenzial, das auch sinken kann, die erwartete Zeit, bis es um einen gewünschten Wert gestiegen ist, abschätzen können. Dies gelingt, wenn die erwartete Zunahme des Potenzials, die so genannte *Drift*, in jedem Schritt positiv ist. Die zugehörige Beweistechnik heißt daher auch *Driftanalyse*, siehe z. B. Hajek (1982) sowie Sasaki und Hajek (1988). Im Zusammenhang mit der Analyse evolutionärer Algorithmen setzen He und Yao (2001) Techniken der Driftanalyse ein, um obere und untere Schranken für die erwartete Laufzeit des (1+1)-EA zu zeigen; weitere Anwendungen der Technik liefern Giel und Wegener (2003).

Im Folgenden erarbeiten wir uns zunächst abermals ein technisches Lemma, welches die von uns gewünschte Driftanalyse gestattet. Es kann als Verallgemeinerung der so genannten waldschen Identität (für Martingale), siehe Feller (1968), verstanden werden. Unsere Version ist auch bei Aldous und Fill (2004) zu finden, doch da jene Monographie nur vorläufig ist, liefern wir hier einen vollständigen Beweis.

Lemma 3.5.10 *Sei D_i, $i \in \mathbb{N}$, eine Folge von Zufallsvariablen, sodass für eine Konstante c und alle i die Ungleichung $|D_i| \leq c$ gilt. Sei für $s > 0$ die Zufallsvariable τ_s das kleinste i, sodass $D_1 + \cdots + D_i = s$ gilt. Wenn $\mathrm{E}(\tau_s)$ existiert und es eine Konstante $\ell > 0$ gibt, sodass $\mathrm{E}(D_i \mid \tau_s \geq i) \geq \ell$ für alle i mit $\mathrm{Prob}(\tau_s \geq i) > 0$ gilt, dann ist $\mathrm{E}(\tau_s) \leq s/\ell$.*

Beweis: Da $|D_i| \leq c$, gilt $\mathrm{E}(D_i \mid \tau_s \geq i) < \infty$ für alle i. Gemäß der Definition von τ_s folgt

$$
\begin{aligned}
s &= \mathrm{E}\left(\sum_{i=1}^{\tau_s} D_i\right) \\
&= \sum_{j=1}^{\infty} \mathrm{Prob}(\tau_s = j) \cdot \mathrm{E}(D_1 + \cdots + D_j \mid \tau_s = j) \\
&= \sum_{j=1}^{\infty} \sum_{i=1}^{j} \mathrm{Prob}(\tau_s = j) \cdot \mathrm{E}(D_i \mid \tau_s = j) \\
&= \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j) \cdot \mathrm{E}(D_i \mid \tau_s = j).
\end{aligned}
$$

Dabei gilt die letzte Gleichung, weil die Eigenschaft $|D_i| \leq c$ impliziert, dass die Reihe $\sum_{j=1}^{\infty}\sum_{i=1}^{j} \mathrm{Prob}(\tau_s = j) \cdot \mathrm{E}(D_i \mid \tau_s = j)$ absolut konvergent ist (großer Umordnungssatz, vgl. Lemma A.7).

Da $j \geq i$, folgt aus $\tau_s = j$ trivialerweise $\tau_s \geq i$ und wir erhalten aus dem Satz von der totalen Wahrscheinlichkeit

$$\begin{aligned} s &= \sum_{i=1}^{\infty}\sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \geq i) \cdot \mathrm{Prob}(\tau_s \geq i) \cdot \mathrm{E}(D_i \mid \tau_s = j) \\ &= \sum_{i=1}^{\infty} \mathrm{Prob}(\tau_s \geq i) \cdot \left[\sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \geq i) \cdot \mathrm{E}(D_i \mid \tau_s = j \wedge \tau_s \geq i)\right]. \end{aligned}$$

Wenn wir den $[\cdot]$-Ausdruck durch ℓ nach unten beschränken können, erhalten wir die Beziehung $s \geq \mathrm{E}(\tau_s) \cdot \ell$ und damit das Lemma. Im $[\cdot]$-Ausdruck dürfen wir die Summanden für $1 \leq j \leq i-1$ hinzufügen, da diese wegen $\mathrm{Prob}(\tau_s = j \mid \tau_s \geq i) = 0$ alle null sind. Also lässt sich der $[\cdot]$-Ausdruck schreiben als

$$\sum_{j=1}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \geq i) \cdot \mathrm{E}(D_i \mid \tau_s = j \wedge \tau_s \geq i).$$

Per definitionem ist dies $\mathrm{E}(D_i \mid \tau_s \geq i)$ und dieser Erwartungswert ist wiederum laut Annahme mindestens ℓ. □

In Anwendungen von Lemma 3.5.10 wird $\mathrm{E}(D_i \mid \tau_s \geq i)$ die Drift repräsentieren. Nun haben wir alles zusammen, um die obere Schranke für RLS_p auf allgemeinen monotonen Polynomen zu zeigen. Wir zeigen eine obere Schranke von $O((n^2/d)2^d)$ für die erwartete Laufzeit, falls p klein genug ist. Diese Schranke ist nicht weit von der unteren Schranke $\Omega((n/d)\log(n/d+1)2^d)$ aus Theorem 3.4.2 entfernt.

Theorem 3.5.11 *Die erwartete Laufzeit von RLS_p auf einem monotonen Polynom f vom Grad $d \leq c\log n$ für eine Konstante c ist höchstens $O((n^2/d)2^d)$, wenn*

$$0 \ < \ p \ \leq \ \min\left\{\frac{1-\gamma}{3dn}, \ \frac{\alpha}{n^{c/2}\log n}\right\}$$

für die Konstante α aus Lemma 3.5.9 und eine beliebige Konstante $\gamma > 0$ gilt.

Beweis: Wir wählen zunächst eine geeignete Potenzialfunktion, mithilfe deren wir Lemma 3.5.10 anwenden können. Es bietet sich an, eine Funktion mit kleinem Wertebereich zu finden. Da dieses weder auf den Bildbereich von f noch auf die Anzahl aktiver Monome zutreffen muss, entscheiden wir uns für die Anzahl spezieller Einsbits im aktuellen Suchpunkt von RLS_p. Für eine Zielfunktion f und einen aktuellen Suchpunkt x nennen wir ein Einsbit von x essenziell bezüglich f, wenn dieses Einsbit in einem Monom enthalten ist, das im Suchpunkt x aktiv ist; Nullbits heißen niemals essenziell. Der Grund für diese Definition ist, dass inessenzielle Einsbits von x keinen Beitrag zu $f(x)$ leisten und dass RLS_p diese mithin zu 0 kippen kann, ohne den f-Wert zu verringern. Natürlich kann ein essenzielles Einsbit zu einem inessenziellen Einsbit werden, aber dafür muss im selben Schritt an anderer Stelle ein Monom aktiv werden. Andernfalls verringerte sich der f-Wert und der neue Suchpunkt würde verworfen.

Solange das Optimum noch nicht erreicht ist, gibt es mindestens ein passives Monom. Die Position und damit eventuell die Zahl der essenziellen Einsbits ändern sich erst, wenn ein passives Monom aktiv wird. Wir nennen entsprechende Schritte ebenfalls *essenziell*. Wenn es gelingt, die erwartete Zahl essenzieller Schritte, bis das Optimum erreicht wird, durch $O(n)$ zu beschränken, erhalten wir zusammen mit Lemma 3.5.9 die gewünschte obere Schranke für die erwartete Laufzeit.

Sei X_i die Anzahl der essenziellen Einsbits nach dem i-ten essenziellen Schritt, d. h., X_0 ist die Anzahl der essenziellen Einsbits des initialen Suchpunkts. Sei $D_0 := X_0$ und $D_i := X_i - X_{i-1}$ für $i \geq 1$. Uns interessiert τ, definiert als das kleinste i mit $\sum_{j=0}^{i} D_j = n$. Einige Bedingungen, die wir für Lemma 3.5.10 benötigen, lassen sich nun leicht nachweisen. Es ist $|D_i| \leq n$ für $i \geq 1$ und es gilt $\mathrm{E}(\tau) < \infty$, da in jedem Schritt die Wahrscheinlichkeit, den optimalen Suchpunkt 1^n zu erzeugen, mindestens p^n beträgt. Unser Ziel ist es nun, für eine Konstante $\varepsilon > 0$ und alle i die Drift gemäß $\mathrm{E}(D_i \mid \tau \geq i) \geq \varepsilon$ nachzuweisen, denn Lemma 3.5.10 liefert dann $\mathrm{E}(\tau) = O(n)$.

Da gemäß Definition jeder essenzielle Schritt mindestens ein Monom aktiviert, gibt es mindestens ein Bit, das in dem Schritt von einem inessenziellen zu einem essenziellen Bit wird. Um die Drift zu beschränken, schätzen wir die erwartete Zahl von Bits ab, die in einem solchen Schritt ihren Zustand von essenziell in inessenziell ändern. Dass ein Schritt essenziell ist, impliziert, dass der Mutant akzeptiert wird, also der f-Wert nicht sinkt. Wir lassen diese Annahme fallen und betrachten die Anzahl der kippenden Einsbits in einem Schritt, der genau ein Nullbit kippt. Damit überschätzen wir die Zahl der verlorenen essenziellen Bits in essenziellen Schritten höchstens. Man beachte, dass ein Einsbit, das kippt, mehrere, im schlimmsten Fall sogar $n-1$ essenzielle Bits in inessenzielle Bits verwandeln kann.

Sei Y die zufällige Zahl an Bits, die RLS_p außerdem kippt, unter der Annahme, dass ein spezielles Nullbit (das ein Monom aktiviert) kippt. Wir unterscheiden zwei Ereignisse in Bezug auf das Kippen dieses speziellen Bits i. Einerseits kann Bit i als erstes kippendes Bit von RLS_p gewählt worden sein (Wahrscheinlichkeit $1/n$) oder es wird als zusätzlich kippendes Bit von RLS_p gewählt (Wahrscheinlichkeit $(1-1/n)p$). Im ersten Fall ist die Zufallsvariable Y binomialverteilt zu den Parametern $n-1$ und p und besitzt einen Erwartungswert von $(n-1)p$. Im zweiten Fall gilt $Y = Z + 1$ für eine Zufallsvariable Z, die binomialverteilt zu den Parametern $n-2$ und p ist. In dem Fall gilt $\mathrm{E}(Y) = 1 + (n-2)p$. Nach dem Satz von der totalen Wahrscheinlichkeit gilt

$$\mathrm{E}(Y) \;=\; \frac{(n-1)\cdot p\cdot(1/n) + (1+(n-2)p)\cdot p\cdot(1-1/n)}{1/n + p\cdot(1-1/n)}$$

$$= \frac{p + (1 + (n-2)p) \cdot p}{1/(n-1) + p} \leq \frac{2p + p^2 n}{1/n + p} \leq \frac{1 - \varepsilon}{d}.$$

Die letzte Ungleichung folgt dabei, weil wir $p \leq (1-\gamma)/(3dn)$ für eine Konstante $\gamma > 0$ voraussetzen.

Nun sind wir noch an der erwarteten Zahl von Bits interessiert, die ihren Zustand von essenziell in inessenziell ändern, wenn $Y = i$ ist. Im schlimmsten Fall sind diese i kippenden Bits allesamt essenzielle Einsbits. Da wir immer noch vereinfachend annehmen, dass jede Mutation, die ein Nullbit kippt, akzeptiert wird, besitzt jede Teilmenge der essenziellen Einsbits, die i Elemente enthält, dieselbe Wahrscheinlichkeit, als Menge der i kippenden Einsbits gewählt zu werden. Sei A die aktuelle Menge der essenziellen Einsbits. Sei L die zufällige Zahl der Bits $j \in A$, die inessenziell werden, wenn ein zufällig gleichverteilt gewähltes Bit $k \in A$ zu 0 kippt. Wir werden unten zeigen, dass $\mathrm{E}(L) \leq d$ gilt. Es folgt nun, dass wir im Durchschnitt höchstens $i \cdot d$ essenzielle Einsbits verlieren, wenn wir gleichverteilt i der Bits $k \in A$ kippen. Da die durchschnittliche Zahl kippender essenzieller Einsbits bereits durch $(1-\varepsilon)/d$ beschränkt wurde, ist bei zufällig gleichverteilter Wahl der kippenden essenziellen Einsbits aus A die erwartete Anzahl inessenziell werdender Bits höchstens $1 - \varepsilon$. Da mindestens ein Bit in einem essenziellen Schritt essenziell wird, folgt $\mathrm{E}(D_i \mid \tau \geq i) \geq \varepsilon$ und damit das Theorem.

Es bleibt immer noch die Behauptung $\mathrm{E}(L) \leq d$ zu zeigen. Dazu wenden wir eine amortisierte Analyse (siehe Cormen, Leiserson, Rivest und Stein, 2001, zuweilen auch „Buchhaltermethode“ genannt) an, indem wir eine Matrix der Größe $|A| \times |A|$ betrachten. Im Folgenden sei o. B. d. A. $A = \{1, \ldots, |A|\}$. Der Matrixeintrag an Position (k, j) soll 1 sein, wenn das j-te Bit inessenziell wird, wenn das k-te Bit kippt. Andernfalls ist der Eintrag 0. Es sei L_k die Zahl der $j \in A$, die inessenziell werden, wenn das k-te Bit kippt. Offensichtlich ist die Zeilensumme in der k-ten Zeile genau L_k, es kann Zeilensummen größer als d geben und wir möchten die durchschnittliche Zeilensumme abschätzen. Die Idee ist nun, für eine feste Spalte $j \in A$ die Anzahl der positiven (k, j)-Einträge abzuschätzen. In der j-ten Spalte stehen Einsen genau an den Positionen k, sodass das Kippen des k-ten Bits das j-te Bit inessenziell macht. Die Spaltensumme ist höchstens d, da für jeden (k, j)-Eintrag, der 1 ist, gilt, dass die k-te Variable in jedem aktiven Monom auftaucht, das die j-te Variable enthält. Da also jede Spaltensumme höchstens d ist, gilt dies auch für die durchschnittliche Spaltensumme. Andererseits ist die durchschnittliche Zeilensumme gerade die durchschnittliche Spaltensumme und die Behauptung folgt. $\square$

Der Beweis der oberen Schranke für RLS_p wird erst mit den im nächsten Unterabschnitt nachzuholenden Beweisen der technischen Analysen aus den Lemmata 3.5.2, 3.5.3, 3.5.4, 3.5.6 und 3.5.8 vollständig sein. Bereits an dieser Stelle ziehen wir aber eine Zwischenbilanz. Theorem 3.5.11 zeigt uns, dass eine globale Suchheuristik auf monotonen Polynomen zumindest fast so effizient wie eine lokale Suchheuristik ist, wenn pro Schritt im Erwartungswert nicht zu viele Bits kippen. Wie in Abschnitt 2.3 erwähnt, können wir RLS_p als abgewandelten (1+1)-EA mit verringerter Mutationswahrscheinlichkeit verstehen. Da Theorem 3.5.11 nur für Werte p mit $p = O(1/(dn))$ gilt, haben wir eine gute obere Schranke für globale Suchheuristiken salopp gesagt bisher nur dann, wenn wir die Standardmutationswahrscheinlichkeit um einen Faktor $\Theta(1/d)$ absenken. Eine intuitive Erklärung dafür liegt bei der Beweistechnik für Theorem 3.5.11 darin, dass das Maß essenzieller Einsbits auf manchen Instanzen, z. B. der Funktion $\mathrm{RR}_d(x)$ aus Abschnitt 3.4, tatsächlich auf einen Schlag um den Wert d sinken kann, selbst wenn nur ein Einsbit kippt. Daher werden wir in Abschnitt 3.5.4 bei der Analyse des (1+1)-EA lediglich eine obere Schranke erhalten, die im Allgemeinen schlechter als $O((n^2/d)2^d)$ ist. Hier bieten sich also weitere Forschungsgegenstände an und wir werden am Ende dieses Kapitels über Erfolg versprechende Ansätze nachdenken.

Als positive Schlussfolgerung halten wir fest, dass es gelungen ist, den schädlichen Effekt, den Schritte von RLS_p, die mehrere Bits auf einmal kippen, haben können, zu kontrollieren. Außerdem haben wir die Vermutung erhärtet, dass die in Theorem 3.3.2 hergeleitete obere Schranke $O((n/d)\log(n/d+1)2^d)$ auch für RLS_p und den (1+1)-EA gilt. Den Nutzen der Techniken, mit denen wir nachweisen, dass eine Markoffkette eine andere in Bezug auf die erwartete Laufzeit dominiert, werden wir ebenfalls am Ende dieses Kapitels diskutieren.

3.5.3 Methoden zum Nachweis der Dominanz von Markoffketten

In diesem Unterabschnitt holen wir wie versprochen einige technische Analysen nach, die letztendlich zum Beweis von Lemma 3.5.8, das wir im letzten Unterabschnitt angewendet haben, erforderlich sind. Zunächst zeigen wir die auf den Seiten 39 ff. zu findenden Lemmata 3.5.2 und 3.5.3 mittels sorgfältiger Abschätzungen von Wahrscheinlichkeiten für spezielle Mutationen von RLS_p bzw. des (1+1)-EA.

Beweis von Lemma 3.5.2: Da $Q(i,j) = 0$ für $|i-j| \geq 3$ gilt, sind nur die Fälle $j = i-1 \geq 0$, $j = i$, $j = i+1$ und $j = i+2 \leq d$ nicht trivial. Sei α die Wahrscheinlichkeit, dass der (1+1)-EA in einem Schritt höchstens zwei der d betrachteten Bits, die den Zustandsraum bestimmen, kippt. Wir setzen $c := 1+\varepsilon$.

1. Fall: $j = i-1 \geq 0$. Dann ist

$$\begin{aligned}
Q(i,j) - c\cdot Q(i+1,j) &= Q(i,i-1) - c\cdot Q(i+1,i-1)\\
&= \frac{1}{\alpha}\cdot\binom{i}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1} - \frac{c}{\alpha}\cdot\binom{i+1}{2}\cdot\frac{1}{n^2}\cdot\left(1-\frac{1}{n}\right)^{d-2}\\
&= \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot i\cdot\left(1-\frac{1}{n}-\frac{c\cdot(i+1)}{2n}\right)\\
&\geq \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot i\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right) \geq 0,
\end{aligned}$$

da nach unseren Annahmen $i+1 \leq d$ und $d \leq (n-1)/c$ gilt.

2. Fall: $j = i$. Dann erhalten wir $Q(i,i) \geq (1/\alpha)\cdot(1-1/n)^d$, da es hinreichend (aber nicht notwendig) ist, keins der d betrachteten Bits zu ändern, um im Zustand i zu bleiben. Also erhalten wir die folgende untere Schranke:

$$\begin{aligned} Q(i,j) - c\cdot Q(i+1,j) &= Q(i,i) - c\cdot Q(i+1,i) \\ &\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^d - c\cdot\frac{1}{\alpha}\cdot\binom{i+1}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1} \\ &\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d-1}\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right) \geq 0. \end{aligned}$$

3. Fall: $j = i+1$. Dann ist

$$\begin{aligned} Q(i+1,j) - c\cdot Q(i,j) &= Q(i+1,i+1) - c\cdot Q(i,i+1) \\ &\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^d - c\cdot\frac{1}{\alpha}\cdot\binom{d-i}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1} \\ &\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d-1}\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right) \geq 0. \end{aligned}$$

4. Fall: $j = i+2 \leq d$. Dann ist

$$\begin{aligned} Q(i+1,j) - c\cdot Q(i,j) &= Q(i+1,i+2) - c\cdot Q(i,i+2) \\ &= \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot\left((d-i-1)\cdot\left(1-\frac{1}{n}\right) - \binom{d-i}{2}\cdot\frac{c}{n}\right) \\ &\geq \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot(d-i-1)\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right) \geq 0. \end{aligned}$$

□

Beweis von Lemma 3.5.3: Da $R(i,j) = 0$ für $|i-j| \geq 3$ gilt, sind nur die Fälle $j = i-1 \geq 0$, $j = i$, $j = i+1$ und $j = i+2 \leq d$ nicht trivial. Sei α die Wahrscheinlichkeit, dass RLS_p in einem Schritt höchstens zwei der d betrachteten Bits kippt. Wir setzen $c := 1+\varepsilon$.

Im Folgenden leiten wir untere Schranken für die $R(i,j)$-Werte her, indem wir lediglich den Fall betrachten, dass das erste kippende Bit einer Mutation von RLS_p nicht zu den d betrachteten Bits gehört. Dies tritt mit einer Wahrscheinlichkeit von $1-d/n$ ein. Um obere Schranken herzuleiten, wenden wir den Satz von der totalen Wahrscheinlichkeit an und unterscheiden danach, ob das erste kippende Bit von RLS_p zu den d betrachteten Bits gehört oder nicht. Die folgenden Rechnungen und Abschätzungen sind im Übrigen sehr ähnlich zu den entsprechenden Stellen im Beweis von Lemma 3.5.2.

1. Fall: $j = i - 1 \geq 0$. Dann ist

$$
\begin{aligned}
R(i,j) - c \cdot R(i+1,j) &= R(i,i-1) - c \cdot R(i+1,i-1) \\
&\geq \frac{1}{\alpha} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot \binom{i}{1} \cdot p \cdot (1-p)^{d-1} - c \cdot \frac{i+1}{n} \cdot \binom{i}{1} \cdot p \cdot (1-p)^{d-2} \right. \\
&\qquad \left. - c \cdot \frac{n-d}{n} \cdot \binom{i+1}{2} \cdot p^2 \cdot (1-p)^{d-2} \right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p) - c \cdot \frac{d}{n} - c \cdot \frac{n-d}{n} \cdot \frac{d}{2} \cdot p \right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(1 - \frac{d}{n} - p - \frac{cd}{n} - c \cdot d \cdot p\right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(1 - \frac{d}{n} - \frac{1}{n} - \frac{2cd}{n}\right) \geq 0.
\end{aligned}
$$

Die vorletzte Ungleichung gilt, da wir $p \leq 1/n$ voraussetzen. Die letzte gilt, da aus $d \leq (n-1)/(3+2\varepsilon)$ die Beziehung $n - d - 1 - 2cd \geq 0$ folgt.

2. Fall: $j = i$. Dann ist

$$
\begin{aligned}
R(i,j) - c \cdot R(i+1,j) &= R(i,i) - c \cdot R(i+1,i) \\
&\geq \frac{1}{\alpha} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p)^{d} - c \cdot \frac{i+1}{n} \cdot (1-p)^{d-1} \right. \\
&\qquad \left. - c \cdot \frac{n-d}{n} \cdot \binom{i+1}{1} \cdot p \cdot (1-p)^{d-1} \right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p) - c \cdot \frac{d}{n} - c \cdot \frac{n-d}{n} \cdot d \cdot p \right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(1 - \frac{d}{n} - p - \frac{cd}{n} - c \cdot d \cdot p\right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(1 - \frac{d}{n} - \frac{1}{n} - \frac{2cd}{n}\right) \geq 0.
\end{aligned}
$$

3. Fall: $j = i + 1$. Dieser Fall verläuft ganz analog zum zweiten Fall mit der einzigen Ausnahme, dass das kippende Bit aus den $d - i$ Nullbits der d betrachteten Bits stammen muss. Allerdings kann $d - i$ wieder durch d nach oben abgeschätzt werden, wie wir es im zweiten Fall für $i + 1$ gemacht haben.

4. Fall: $j = i + 2 \leq d$. Dieser Fall verläuft ganz analog zum ersten Fall mit der einzigen Ausnahme, dass das kippende Bit/die kippenden Bits aus den $d - i - 1$ bzw. $d - i$ Nullbits der d betrachteten Bits stammen muss/müssen. Wiederum schätzen wir $d - i$ durch d nach oben ab. □

In den Lemmata 3.5.4, 3.5.6 und 3.5.8, zu finden ab Seite 40, zeigen wir (zumindest implizit) jeweils, dass eine Folge von Zufallsvariablen eine andere stochastisch dominiert. Als Beweismethode geht zunächst der nahe liegende Ansatz der vollständigen Induktion, gepaart mit einer größeren Menge von Abschätzungen, ein. Im letzten Lemma schließlich greifen wir wiederum die auf Seite 36 erwähnte *Coupling-Technik* auf. Die Idee ist dabei, einen stochastischen Prozess (in diesem Fall eine Markoffkette) durch einen äquivalenten stochastischen Prozess, dessen Formulierung einer Analyse aber besser zugänglich ist, zu ersetzen. Der Begriff *coupling* drückt in diesem Fall aus, dass die äquivalenten Markoffketten insofern aneinander gekoppelt sind, als sie zu jedem Zeitpunkt dieselbe Wahrscheinlichkeitsverteilung besitzen.

Beweis von Lemma 3.5.4: Die zweite Behauptung folgt mit der Definition des Erwartungswertes aus der ersten. (Auch der Fall, dass $\sum_{t=0}^{\infty} \mathrm{Prob}(\tau^{(\cdot)} \geq t)$ divergiert, kann damit abgedeckt werden, indem wir $\mathrm{E}\big(\tau^{(\cdot)}\big) = \infty$ interpretieren.) Für die erste Behauptung führen wir eine Induktion über t durch und bemerken, dass der Induktionsanfang für $t = 0$ trivial ist. Wir brauchen im Induktionsschritt nur Werte $i \leq d-2$ zu betrachten, da $\mathrm{Prob}(\tau^{(d)} = 0) = 1$ gilt.

Die Werte $P(i,j)$ bezeichnen die Übergangswahrscheinlichkeiten der betrachteten Markoffkette. Für den Induktionsschritt bedingen wir gemäß dem ersten Schritt, den die Markoffkette im Zustand i bzw. $i+1$ ausführt, und erhalten mit dem Satz von der totalen Wahrscheinlichkeit

$$\mathrm{Prob}(\tau^{(i)} \geq t+1) - \mathrm{Prob}(\tau^{(i+1)} \geq t+1) \;=\; \sum_{k=0}^{d} \big(P(i,k) - P(i+1,k)\big) \cdot \mathrm{Prob}(\tau^{(k)} \geq t).$$

Gemäß der Annahme eines 0-Vorteils und der Induktionsvoraussetzung gelten die Beziehungen

$$P(i,k) - P(i+1,k) \geq 0 \;\text{ und }\; \mathrm{Prob}(\tau^{(k)} \geq t) \geq \mathrm{Prob}(\tau^{(i)} \geq t) \quad \text{für } k \leq i$$

sowie

$$P(i,k) - P(i+1,k) \leq 0 \;\text{ und }\; \mathrm{Prob}(\tau^{(k)} \geq t) \leq \mathrm{Prob}(\tau^{(i)} \geq t) \quad \text{für } k \geq i+1.$$

In der obigen Summe können demnach die Summanden für $k \neq i$ jeweils durch den Summanden für $k = i$ nach unten abgeschätzt werden. Wir erhalten

$$\mathrm{Prob}(\tau^{(i)} \geq t+1) - \mathrm{Prob}(\tau^{(i+1)} \geq t+1) \;\geq\; \mathrm{Prob}(\tau^{(i)} \geq t) \sum_{k=0}^{d} \big(P(i,k) - P(i+1,k)\big).$$

Der letzte Ausdruck ist 0. □

Beweis von Lemma 3.5.6: Wie im vorigen Beweis genügt es, die erste Behauptung durch Induktion über t zu zeigen, und der Induktionsanfang für $t = 0$ sowie der Fall $i = d$ sind trivial. Wir wenden den Satz von der totalen Wahrscheinlichkeit an und erhalten im Induktionsschritt für $i \leq d - 1$

$$\begin{aligned}
&\mathrm{Prob}(\tau_0^{(i)} \geq t+1) - \mathrm{Prob}(\tau_1^{(i)} \geq t+1) \\
&\qquad = \sum_{k=0}^{d} \bigl(P_0(i,k) \cdot \mathrm{Prob}(\tau_0^{(k)} \geq t) - P_1(i,k) \cdot \mathrm{Prob}(\tau_1^{(k)} \geq t)\bigr) \\
&\qquad \geq \sum_{k=0}^{d} \bigl(P_0(i,k) - P_1(i,k)\bigr) \cdot \mathrm{Prob}(\tau_0^{(k)} \geq t).
\end{aligned}$$

Nun können wir ganz analog zum Beweis von Lemma 3.5.4 fortfahren. Da Y_1 einen Vorteil bezüglich Y_0 hat und Y_0 einen 0-Vorteil hat, folgt

$$P_0(i,k) - P_1(i,k) \geq 0 \;\text{ und }\; \mathrm{Prob}(\tau_0^{(k)} \geq t) \geq \mathrm{Prob}(\tau_0^{(i)} \geq t) \quad \text{für } k \leq i-1$$

sowie

$$P_0(i,k) - P_1(i,k) \leq 0 \;\text{ und }\; \mathrm{Prob}(\tau_0^{(k)} \geq t) \leq \mathrm{Prob}(\tau_0^{(i)} \geq t) \quad \text{für } k \geq i+1.$$

Also ist

$$\begin{aligned}
&\mathrm{Prob}(\tau_0^{(i)} \geq t+1) - \mathrm{Prob}(\tau_1^{(i)} \geq t+1) \\
&\qquad\qquad \geq \mathrm{Prob}(\tau_0^{(i)} \geq t) \cdot \sum_{k=0}^{d} (P_0(i,k) - P_1(i,k)) \;=\; 0. \quad \square
\end{aligned}$$

Beweis von Lemma 3.5.8: Wir führen eine dritte zeithomogene Markoffkette Y_2 mit Übergangswahrscheinlichkeiten $P_2(i,j)$ und folgenden Eigenschaften ein:

1. $\mathrm{E}\bigl(\tau_2^{(i)}\bigr) \;\leq\; c^* \cdot \mathrm{E}\bigl(\tau_0^{(i)}\bigr)$ für $0 \leq i \leq d-1$,
2. Y_1 hat einen Vorteil bezüglich Y_2,
3. Y_2 hat einen 0-Vorteil.

Mit Lemma 3.5.6 liefern die zweite und die dritte Eigenschaft dann $\mathrm{E}\bigl(\tau_1^{(i)}\bigr) \leq \mathrm{E}\bigl(\tau_2^{(i)}\bigr)$ und dies impliziert zusammen mit der ersten Eigenschaft das Lemma.

Sei $P_2(i,j) := c(i,i+1) \cdot P_0(i,j)$ für $i \leq d-1$ und $j \neq i$ sowie $P_2(i,i) := 1 - \sum_{j \neq i} P_2(i,j)$. Außerdem setzen wir $P_2(d,j) := P_0(d,j)$ für $0 \leq j \leq d$. Da gemäß Definition 3.5.5 die Ungleichungen $c(i,i+1) \leq 1$ für alle $i \leq d-1$ gelten, haben wir damit tatsächlich legale Übergangswahrscheinlichkeiten definiert. Nun können wir zur oben angesprochenen Coupling-Technik kommen und einen Schritt der Markoffkette Y_2 auf folgende Weise darstellen: Solange wir noch nicht im Zustand d sind, bleiben wir mit einer Wahrscheinlichkeit von genau $1 - c(i, i+1)$ auf jeden Fall im aktuellen Zustand i. Andernfalls simulieren wir einen Schritt von Y_0 im Zustand i, was ebenfalls zur Folge haben kann, dass wir im Zustand i bleiben. Damit ist die erwartete Zeit, bis Y_2 im Zustand i einen Schritt von Y_0 simuliert, durch $1/c(i, i+1) \leq c^*$ gegeben. Es folgt die erste Eigenschaft.

Zum Beweis der zweiten Eigenschaft genügen folgende Abschätzungen. Für $j \geq i+1$ ist per definitionem

$$P_1(i,j) - P_2(i,j) \;=\; c(i,j) \cdot P_0(i,j) - c(i,i+1) \cdot P_0(i,j) \;\geq\; 0,$$

da hier $c(i,j) \geq c(i,i+1)$ gemäß Definition 3.5.5 gilt. Für $j \leq i-1$ beweist man die Beziehungen $P_1(i,j) \leq P_2(i,j)$ ganz analog. Es folgt der Vorteil von Y_1 bezüglich Y_2.

Um die dritte Eigenschaft zu zeigen, beobachten wir, dass aufgrund der speziellen Definition von Y_2 und der Schranke $c(i+1,i+2) \leq 1$ die Abschätzung

$$P_2(i+1,j) \;\geq\; c(i+1,i+2) \cdot P_0(i+1,j)$$

sogar für $j = i+1$ gilt. Wir erhalten für $j \geq i+1$

$$P_2(i+1,j) - P_2(i,j) \;\geq\; c(i+1,i+2) \cdot P_0(i+1,j) - c(i,i+1) \cdot P_0(i,j).$$

Da $1/c^* \leq c(i+1,i+2)$ und $c(i,i+1) \leq 1$ gilt und weil Y_0 gemäß Annahme einen (c^*-1)-Vorteil hat, folgt

$$P_2(i+1,j) - P_2(i,j) \;\geq\; P_0(i+1,j)/c^* - P_0(i,j) \;\geq\; 0.$$

Mit einer analogen Rechnung zeigt man $P_2(i+1,j) - P_2(i,j) \leq 0$ im Fall $j \leq i$. Also hat Y_2 einen 0-Vorteil. □

3.5.4 Eine obere Schranke für den (1+1)-EA

Nachdem wir in den Unterabschnitten 3.5.1–3.5.3 mit einigem Aufwand eine obere Schranke für die erwartete Laufzeit von RLS_p auf monotonen Polynomen gezeigt haben, möchten wir gerne möglichst viele der dortigen Techniken wieder verwerten und auf den (1+1)-EA übertragen. Dies gelingt glücklicherweise in großen Teilen unter anderem deshalb, da RLS_p im Fall $p = 1/n$ als kleine Modifikation des (1+1)-EA verstanden werden kann und im Fall $p < 1/n$ ähnlich zum (1+1)-EA bei verringerter Mutationswahrscheinlichkeit ist. Eine verringerte Mutationswahrscheinlichkeit sollte die Optimierung monotoner Polynome aber nicht erschweren, sofern wir nur die Zahl der Schritte betrachten, in denen etwas passiert, d. h. sich der aktuelle Suchpunkt ändert.

Wir haben am Ende von Abschnitt 3.5.2 bereits erwähnt, dass wir die erwartete Laufzeit des (1+1)-EA nicht durch den Ausdruck $O((n^2/d)2^d)$ aus Theorem 3.5.11 beschränken können. Jedoch ist es möglich, ein zu Lemma 3.5.9 korrespondierendes Ergebnis mit marginalen Abwandlungen der zugehörigen Beweistechniken zu zeigen.

Lemma 3.5.12 *Sei f ein monotones Polynom vom Grad d und sei m eins seiner Monome. Dann existiert eine Konstante $\alpha > 0$, sodass der (1+1)-EA das Monom m in einer erwarteten Zeit von $O((n/d)2^d)$ aktiviert, wenn $d \leq 2\log n - 2\log\log n - \alpha$ ist.*

Beweis: Der Beweis hat denselben Aufbau wie der Beweis von Lemma 3.5.9, sodass wir nur die Stellen nennen, an denen abweichende Argumente nötig sind.

Als Erstes ist die Wahrscheinlichkeit, dass eine Mutation des (1+1)-EA mindestens drei Bits von m kippt, nach oben beschränkt durch

$$\binom{d}{3}\left(\frac{1}{n}\right)^3 \leq \frac{d^3}{6n^3}.$$

Die Wahrscheinlichkeit, dass eine solche Mutation in einer Phase der Länge $c(n/d)2^d$ passiert, ist höchstens $(c/6)d^2 2^d/n^2$, was kleiner wird als eine Konstante kleiner als 1, wenn α groß genug ist.

Zweitens betrachten wir im Zusammenhang mit dem dritten Typ von Fehler die Wahrscheinlichkeit, dass ein Schritt kein Suffixbit kippt. Diese beträgt mindestens $(1-1/n)^{n-1} \geq 1/\mathrm{e}$ und geht auch in die Anwendung von Lemma 3.5.8 ein. Wir können jenes Lemma dann mit $c(i,j)$-Werten anwenden, für die $\min\{c(i,i+1)\} \geq 1/\mathrm{e}$ gilt. Weiterhin benötigen wir für Y_0 lediglich einen $(\mathrm{e}-1)$-Vorteil. Auch Theorem 3.2.1 kann weiterhin angewendet werden, da es auch für den (1+1)-EA gilt. Anstelle von Lemma 3.5.3 wenden wir Lemma 3.5.2 an. Hier genügt $d \leq (n-1)/\mathrm{e}$.

Zu guter Letzt kann das Argument, dass Y_1 einen relativen Vorteil bezüglich Y_0 für $c(i,j)$-Werte mit $c^* \leq \mathrm{e}$ hat, auf genau dieselbe Weise angewendet werden, d. h. mit einer Anwendung des Satzes von der totalen Wahrscheinlichkeit auf die möglichen Nachfolger eines aktuellen Suchpunkts $a = (b,c)$. □

Die folgende obere Schranke weist die Besonderheit auf, dass sie von der Zahl N nicht verschwindender Terme abhängt. Wenn $N = O(n)$ gilt, ist die Schranke nicht schlechter als die aus Abschnitt 3.5.2.

Theorem 3.5.13 *Die erwartete Laufzeit des (1+1)-EA auf einem monotonen Polynom f mit N Monomen und Grad $d \leq 2\log n - 2\log\log n - \alpha$ für die Konstante α aus Lemma 3.5.12 ist $O(N(n/d)2^d)$.*

Beweis: Wir wenden eine Abwandlung der Ranggruppenmethode (vgl. Lemma 2.3.11) an. Ohne Beschränkung der Allgemeinheit seien die positiven Koeffizienten der N nicht verschwindenden Monome gemäß $w_1 \geq w_2 \geq \cdots \geq w_N > 0$ geordnet. Damit stellen wir eine f-basierte Partition mit $N+1$ Ranggruppen auf. Sei

$$L_i := \{x \mid w_1 + \cdots + w_i \leq f(x) < w_1 + \cdots + w_{i+1}\}$$

für $i < N$ und $L_N := \{x \mid f(x) = w_1 + \cdots + w_N\}$. Wenn es uns gelingt, für $i < N$ eine obere Schranke von $O((n/d)2^d)$ für die erwartete Zeit, bis der (1+1)-EA die Ranggruppe L_i verlässt, zu zeigen, folgt unmittelbar das Theorem.

Sei x ein aktueller Suchpunkt des (1+1)-EA mit $x \in L_i$. Aus der Definition der Ranggruppen folgt dann, dass es ein $j \leq i + 1$ gibt, sodass das Monom m_j mit Koeffizient w_j in x passiv ist. Nach Lemma 3.5.12 ist die erwartete Zeit, bis m_j aktiv wird, höchstens $O((n/d)2^d)$. Wir nehmen pessimistisch an, dass L_i vorher nicht verlassen wird und dass der (1+1)-EA den Schritt, der m_j aktiviert, nicht akzeptiert, wenn der Schritt ein Bit außerhalb von m_j kippt. Da die Wahrscheinlichkeit, dass kein Bit außerhalb von m_j kippt, mindestens $(1 - 1/n)^{n-1} \geq \mathrm{e}^{-1}$ beträgt, wird L_i mit einer konstanten Wahrscheinlichkeit innerhalb von $O((n/d)2^d)$ Schritten verlassen. Wir können unabhängige Phasen wiederholen, und zwar eventuell mit anderen passiven Monomen m_k, $k \leq i + 1$. Da die erwartete Zahl benötigter Phasen $O(1)$ ist, ist das Theorem bewiesen. □

Die Laufzeitschranke aus dem vorigen Theorem bietet noch Raum für Verbesserungen. Bereits am Ende von Abschnitt 3.5.2 haben wir diskutiert, warum die für RLS_p durchgeführte Driftanalyse sich nicht unmittelbar auf den (1+1)-EA übertragen lässt. Jedoch könnte man versuchen, eine Driftanalyse anstelle mit der Zahl der essenziellen Einsen mit der Zahl der tatsächlich im aktuellen Suchpunkt vorhandenen Einsen vorzunehmen, also eine andere Potenzialfunktion zu betrachten. Sei X_i nun die Zahl aller Einsen im zufälligen Suchpunkt des (1+1)-EA zum Zeitpunkt i auf einem gegebenen monotonen Polynom vom Grad d. Zur Veranschaulichung stellen wir uns vor, dass dieses gegebene Polynom die Funktion $\mathrm{RR}_d(x)$ aus Abschnitt 3.4 ist. Wenn der Suchpunkt zum Zeitpunkt i viele Monome mit mehr als $d/2$ Einsen belegt, diese Monome aber nicht aktiviert, scheint $\mathrm{E}(X_{i+1} - X_i)$ negative Werte annehmen zu können, sodass eine Anwendung von Lemma 3.5.10 ausscheidet. Diesem sind wir mit der Definition essenzieller Einsen begegnet, welche Einsen in passiven Monomen ignorierte. Damit haben wir aber die sehr pessimistische Annahme, dass es keine Einsen außerhalb aktiver Monome gebe, vorgenommen.

Eine Driftanalyse anhand der tatsächlichen Zahl der Einsen scheint nicht zu funktionieren, wenn wir jede Verteilung der Einsen in passiven Monomen zulassen. Andererseits wissen wir aus Abschnitt 3.2, dass die Belegung in Monomen vom Grad d gleichverteilt über $\{0,1\}^d$ ist, sofern der (1+1)-EA jeden neuen Suchpunkt ungeachtet seines Funktionswertes akzeptiert. Man kann sich überlegen, dass die Belegung in einem Monom vom Grad d gleichverteilt über $\{0,1\}^d \setminus \{1^d\}$ ist, wenn wir den modifizierten (1+1)-EA unter der Annahme, das Monom nicht mit 1^d belegt zu haben, betrachten. Sei in diesem Szenario mit Y_i die Zahl der Einsen im Monom zum Zeitpunkt i bezeichnet. Die Drift lässt sich nach dem Satz von der totalen Wahrscheinlichkeit schreiben als

$$\mathrm{E}(Y_{i+1} - Y_i) \;=\; \sum_{j=0}^{d-1} \mathrm{E}(Y_{i+1} - Y_i \mid Y_i = j)\,\mathrm{Prob}(Y_i = j)$$

und damit vermutlich durch einen positiven Wert nach unten beschränken. Die positive untere Schranke scheint zu gelten, da die Verteilung ohne die Annahme, nicht im Optimum zu sein, gleichverteilt über $\{0,1\}^d$ ist, was dann eine Drift von genau 0 implizierte. Mit der Annahme eliminieren wir aber die negative Drift im Zustand d.

Es drängt sich die Vermutung auf, dass die Drift wie zuvor auch dann durch einen positiven Wert nach unten beschränkt ist, wenn wir die Zahl der Einsen in einem Monom eines beliebigen monotonen Polynoms betrachten. Jedoch benötigen wir für den speziellen Ansatz eine Aussage über die Verteilung dieser Zahl für jeden Zeitpunkt und müssen nachweisen, dass sich diese Verteilung gegenüber der von der Gleichverteilung induzierten Verteilung zu unseren Gunsten unterscheidet. Möglicherweise funktioniert dieses mit einer Abwandlung oder Verallgemeinerung der im Beweis von Lemma 3.5.9 angewandten Technik, die einen relativen Vorteil von Y_1 gegenüber Y_0 zeigt. Es scheint aber nun unumgänglich zu sein, auch Schritte, die mehr als zwei Bits des betrachteten Monoms kippen, zuzulassen. Auch wenn die zugelassenen Veränderungen der Zahl der Einsen damit noch schwerer zu durchschauen sind, gelingt es in Zukunft vielleicht, geeignete Schranken zu finden und insgesamt auf diesem Wege zu einer besseren oberen Schranke für die erwartete Laufzeit des (1+1)-EA, in die der Parameter N nicht mehr einfließt, zu kommen.

3.6 Fazit

In diesem Kapitel haben wir uns der strukturellen Frage nach der Zeitkomplexität einfacher randomisierter Suchheuristiken auf einer vorgegebenen Klasse von Funktionen gewidmet. Speziell haben wir die erwartete Laufzeit von RLS und des (1+1)-EA auf monotonen Polynomen vom Grad d betrachtet. Während es bei der lokalen Suchheuristik RLS gelungen ist, die erwartete Laufzeit durch die asymptotisch nicht verbesserbare Schranke $O((n/d)\log(n/d+1)2^d)$ einzugrenzen, haben wir für den (1+1)-EA lediglich eine Schranke erhalten, in die die Zahl der Monome des Polynoms eingeht. Da wir aber glauben, dass die für RLS gewonnene Schranke auch für den (1+1)-EA gilt, haben wir eine dritte Suchheuristik RLS_p untersucht, die als Brücke zwischen den vorgenannten Suchheuristiken fungiert. Wenn RLS_p pro Schritt ein Bit mit Sicherheit kippt und jedes weitere mit einer Wahrscheinlichkeit von höchstens $c/(dn)$ für eine geeignete Konstante $c > 0$, erhalten wir eine obere Schranke für die erwartete Laufzeit, die nicht mehr weit von der vermuteten Wahrheit entfernt ist. Diese Analysen können in Zukunft Startpunkt für mögliche Verbesserungen sein.

Eine interessante, allerdings nicht neue Schlussfolgerung aus den in diesem Kapitel vorgestellten Analysen besteht darin, dass der stochastische Prozess, den eine globale Suchheuristik wie RLS_p und der (1+1)-EA beschreibt, schwieriger zu analysieren ist als der Prozess einer lokalen Suchheuristik. Um den komplizierteren Prozess zu untersuchen, haben wir allgemeine Analysemethoden vorgestellt, mit deren Hilfe sich Resultate vom einfachen Prozess auf den komplizierten Prozess übertragen lassen. Neben diesen Analysemethoden haben wir außerdem noch ein Theorem zur Driftanalyse vorgestellt, mit dessen Hilfe sich die zufällige Änderung der Zahl spezieller Einsen in den aktuellen Suchpunkten von RLS_p fassen ließ und das allgemein genug formuliert ist, um hoffentlich auch in anderen Zusammenhängen weitere Anwendungen zu finden.

4 Randomisierte Suchheuristiken und Approximationen

Nachdem wir uns im vergangenen Kapitel mit der eher strukturellen Frage nach dem Verhalten einfacher randomisierter Suchheuristiken auf Funktionenklassen auseinander gesetzt haben, werden wir uns in diesem Kapitel einer Analyse randomisierter Suchheuristiken für ein konkretes kombinatorisches Optimierungsproblem zuwenden. Neben der Laufzeit werden wir uns mit einem weiteren Aspekt, nämlich der Approximationsfähigkeit der Suchheuristik, beschäftigen. Bevor wir jedoch in die konkreten Fragestellungen einsteigen, bietet sich ein kleiner Überblick in Hinblick auf eine mögliche Einordnung der Ergebnisse dieses Kapitels in die Forschung über die Zeitkomplexität evolutionärer Algorithmen und auch problemspezifischer Algorithmen für die kombinatorische Optimierung an.

Inzwischen existieren zahlreiche Arbeiten, in denen Resultate zur Laufzeit einfacher und mittlerweile auch komplexerer evolutionärer Algorithmen gezeigt werden. Wenn wir uns auf das Feld der kombinatorischen Optimierung, besonders der Optimierung pseudoboolescher Funktionen $f\colon \{0,1\}^n \to \mathbb{R}$, zurückziehen, können wir die genannten Arbeiten in mindestens drei verschiedene Klassen einordnen. Die folgende Kategorisierung erhebt allerdings keinen Anspruch darauf, erschöpfend zu sein. Zum Ersten besteht Interesse am Verhalten randomisierter Suchheuristiken auf Funktionen und Funktionenklassen, seien es lineare, quadratische, unimodale Funktionen, monotone Polynome (siehe Kapitel 3) usw. Dazu existiert inzwischen eine ganze Reihe von erfolgreichen Arbeiten, die zum Teil an die historisch frühesten Betrachtungen einer Zeitkomplexität evolutionärer Algorithmen anknüpfen, z. B. die Veröffentlichungen von Mühlenbein (1992) und Rudolph (1997). Einen guten Einblick in diese Sichtweise eröffnet die bereits häufig genannte Arbeit von Droste, Jansen und Wegener (2002a), indem sie einerseits nahe liegende Funktionenklassen vorstellt, andererseits aber konstruierte Beispielfunktionen heranzieht, um das typische Verhalten eines EAs auf speziellen Funktionen zu erhellen.

Zum Zweiten drängte sich angesichts der schier unüberschaubaren Vielfalt evolutionärer Algorithmen eine vergleichende Studie exemplarischer EAs auf. Als außerordentlich spannend gilt hier der Einfluss spezieller Operatoren der Mutation (Jansen und Wegener, 2000), der Rekombination (Jansen und Wegener, 2002) und weiterer Operatoren auf die Laufzeit eines EAs. Darüber hinaus fanden dynamische EAs, die beispielsweise ihre Mutationsstärke während der Laufzeit anpassen, große Beachtung (Droste, Jansen und Wegener, 2001). Nicht zuletzt gilt es als äußerst reizvoll, die Auswirkungen einer echten Population, also der Verwaltung einer Menge von aktuellen

Suchpunkten, auf die Laufzeit evolutionärer Algorithmen zu verstehen. Mit dem letztgenannten Thema werden wir uns daher eingehend in Teil II dieser Dissertation auseinander setzen. In vielen Fällen greifen Arbeiten aus dieser zweiten Klasse wiederum auf künstliche Beispielfunktionen zurück, um ein typisches Verhalten nachzuweisen und verallgemeinerbare Analysemethoden zu erarbeiten.

In eine dritte Klasse von Arbeiten fallen Studien, die zumeist nach dem Jahr 2002 zur Veröffentlichung gelangten und eine besonders populäre Motivation zum Ausgangspunkt haben. Evolutionäre Algorithmen kommen häufig dann zum Einsatz, wenn ein Optimierungsproblem vorliegt, zu dessen Lösung noch kein problemspezifischer Algorithmus bekannt ist und bei dem „gute“ Lösungen „in kurzer Zeit“ verlangt werden. Da der Entwurf problemspezifischer Algorithmen sehr aufwendig sein kann, wie in Teil III dieser Arbeit exemplarisch gezeigt werden wird, fehlen bei solchen Restriktionen dann möglicherweise Ressourcen wie Wissen und Zeit, um einen problemspezifischen Algorithmus zu entwerfen. Als Ausweg bieten sich dann randomisierte Suchheuristiken wie evolutionäre Algorithmen an. Diese sind einfach zu implementieren und gelten als robust, d. h. tolerant gegenüber Störungen und Fehlern.

Niemand behauptet, dass evolutionäre Algorithmen problemspezifische Algorithmen im Allgemeinen überflügeln könnten. Nichtsdestoweniger schreiben manche Studien evolutionären Algorithmen erstaunlichen Erfolg bei der Suche nach Lösungen für kombinatorische Optimierungsprobleme zu. Der dritte Typ von Arbeiten nimmt sich daher folgende Frage vor: Wie effizient ist ein evolutionärer Algorithmus auf einem vorgegebenen, bekannten kombinatorischen Optimierungsproblem? Dabei mag man an aus der Komplexitätstheorie und der Forschung über effiziente Algorithmen bekannte Optimierungsprobleme wie das Rucksackproblem, kürzeste Wege in Graphen usw. denken. Erkenntnisse über die Zeitkomplexität randomisierter Suchheuristiken, speziell evolutionärer Algorithmen, auf gegebenen kombinatorischen Optimierungsproblemen tragen weiter dazu bei zu verstehen, wann und wieso solche Heuristiken erfolgreich sind. Positive wie auch negative Resultate über die Effizienz der Heuristiken gestatten es einerseits, ihre Anwendung zu rechtfertigen und andererseits dem Nimbus, dass evolutionäre Algorithmen besonders effiziente Optimierungsverfahren seien, zu begegnen.

Eine der ersten Analysen evolutionärer Algorithmen auf kombinatorischen Optimierungsproblemen stammt von Giel und Wegener (2003). Die Autoren untersuchen eine Variante von RLS sowie den (1+1)-EA auf einem wohl bekannten Graphproblem, nämlich der Suche nach Matchings maximaler Größe. Dabei gelingt es, Graphklassen zu identifizieren, auf denen die Suchheuristiken effizient sind, und andererseits ein Beispiel vorzulegen, bei dem die Suchheuristiken sich extrem ineffizient verhalten. Auf eine weitere Besonderheit der Arbeit werden wir gleich zurückkommen. Neumann und Wegener (2004) analysieren randomisierte Suchheuristiken zur Berechnung minimaler Spannbäume in Graphen und Neumann (2004) beweist, dass randomisierte Suchheuristiken effizient Eulerkreise in Graphen finden. Weiterhin zeigen die Studien von Scharnow, Tinnefeld und Wegener (2002), dass evolutionäre Algorithmen effizient kürzeste Wege finden und effizient sortieren, wenn man das Problem zu sortieren geeignet als Optimierungsproblem kodiert. Auch das so genannte Ising-Modell, ein aus der Physik stammendes Modell für ferromagnetische Prozesse, kann als Optimierungsproblem verstanden werden. Aus diesem Blickwinkel stellen u. a. Fischer und Wegener (2004) und Fischer (2004) Beweise für die Laufzeit evolutionärer Algorithmen für verschiedene Modellvarianten vor. Es sei noch bemerkt, dass für die betrachteten Ising-Modelle zwar triviale deterministische Optimierungsverfahren existieren, dass aber gerade deswegen die Effizienz von Suchheuristiken besonders interessiert und darüber hinaus mit den Ising-Modellen eine Brücke zur historischen Bestimmung gewisser evolutionärer Algorithmen als Adaptationsschemata geschlagen wurde.

Wenn wir im Folgenden evolutionäre Algorithmen gegen problemspezifische Algorithmen antreten lassen, wollen wir sogar noch einen Schritt weiter gehen. Unser Ziel ist es, uns ein Optimierungsproblem vorzunehmen, für das zwar kein exakter Polynomialzeitalgorithmus bekannt ist, aber genügend problemspezifisches Wissen sowie die Möglichkeit einer Approximation in Polynomialzeit besteht. Auch in der Praxis ist die Approximation optimaler Lösungen mithilfe von Suchheuristiken das eigentliche Anliegen, weil man im Allgemeinen nicht auf eine optimale Lösung in kurzer Zeit hoffen kann. Bereits Giel und Wegener (2003) zeigen im Hinblick auf maximale Matchings, dass der (1+1)-EA zwar in Polynomialzeit mit hoher Wahrscheinlichkeit bei der Suche nach einem maximalen Matching scheitern kann, aber für jedes konstante $\varepsilon > 0$ mit hoher Wahrscheinlichkeit nach polynomiell vielen Schritten eine Lösung findet, deren Wert um höchstens den Faktor $1 + \varepsilon$ schlechter als optimal ist. Dieses Ergebnis darf man als randomisierte Variante eines so genannten polynomiellen Approximationsschemas auffassen, worauf wir im nächsten Abschnitt zu sprechen kommen werden. He und Yao (2003) stellen andererseits einige grundlegende notwendige Bedingungen dafür auf, dass ein gegebener EA ein randomisierter Approximationsalgorithmus mit nicht trivialen Gütegarantien sein kann.

Im Rahmen dieses Kapitels werden wir unser Hauptaugenmerk auf die Approximationsfähigkeit des (1+1)-EA und von RLS auf einem NP-harten Optimierungsproblem legen. Bei diesem Problem wird es sich um eine Optimierungsvariante des bekannten Problems PARTITION handeln, das vielleicht das einfachste NP-harte Optimierungsproblem verkörpert. Im folgenden Abschnitt werden wir eine genaue Formulierung dieses Optimierungsproblems vorstellen und wichtige problemspezifische Approximationsalgorithmen vorstellen. Gegen diese Approximationsalgorithmen lassen wir in den Abschnitten 4.3–4.4 den (1+1)-EA und RLS sowie parallele Läufe derselben antreten und erhalten Aussagen über die dazugehörige Approximationsgüte sowie Bezüge zu den problemspezifischen Algorithmen. Allerdings werden auch negative Resultate ersichtlich werden.

Wenn Theoretiker(innen) darauf hinweisen, dass ein vorliegendes Optimierungsproblem NP-hart und damit vermutlich nicht effizient lösbar sei, halten Praktiker(innen) ihnen manchmal die dieser Sichtweise innewohnende Worst-Case-Analyse vor. Vielleicht tauchen in der Praxis die schwierigen Instanzen, für die kein Polynomialzeitalgorithmus existiert, nur höchst selten oder nie auf. In den letzten Jahren haben Ansätze, diese Worst-Case-Sichtweise für (speziell NP-harte) Optimierungsprobleme abzumildern, daher regen Zulauf gefunden. Dabei rückte zunehmend auch der Wunsch einer *Average-Case-Analyse* von Algorithmen und Heuristiken für (nicht nur NP-harte) Optimierungsprobleme in den Vordergrund. Die Vorstellung dabei besteht darin, dass die Eingabeinstanzen eines Optimierungsproblems der Maßgabe einer speziellen Wahrscheinlichkeitsverteilung folgen und somit die Laufzeit des Algorithmus auf einer Instanz I zu gewichten sei mit der Wahrscheinlichkeit, dass die Instanz I auftritt. Bereits die altbekannte Average-Case-Analyse von deterministischen Quicksort-Algorithmen fußt auf der Annahme einer Gleichverteilung aller Permutationen. Wir errechnen dabei eine erwartete Laufzeit des Algorithmus bezüglich der zufälligen Instanzen.

Average-Case-Analysen bekannter Algorithmen für kombinatorische Optimierungsprobleme können in verschiedenen Szenarien aussagekräftig sein. Zum einen ist es wünschenswert, bei Polynomialzeitalgorithmen mithilfe einer Average-Case-Analyse eine erwartete Laufzeit zu erhalten, die asymptotisch viel geringer als die polynomielle Worst-Case-Laufzeit ist. In diese Richtung geht beispielsweise die Arbeit von Motwani (1994), der eine Average-Case-Analyse u. a. für Algorithmen zur Berechnung maximaler Matchings vorstellt. Zum anderen erscheint es reizvoll, einen exakten Algorithmus für ein NP-hartes Optimierungsproblem in Hinblick auf sein Average-Case-Verhalten zu analysieren, wenn bekannt ist, dass dieser Algorithmus zwar selbstverständlich eine superpolynomielle Worst-Case-Laufzeit besitzt, aber auf „natürlichen“ Instanzen erstaunlich effizient ist. Kürzlich haben Beier und Vöcking (2003) ein Resultat, das in diesen Rahmen passt, vorgestellt. Sie beweisen, dass ein altbekannter exakter Algorithmus für das Rucksackproblem polynomielle erwartete Laufzeiten für zahlreiche Klassen von Eingabeverteilungen zeigt. Darüber hinaus erläutern sie, dass kleine, zufällige Perturbationen von Worst-Case-Instanzen zu Instanzen führen, auf denen der betrachtete Algorithmus in erwarteter Polynomialzeit läuft. Diese so genannte geglättete Analyse *(smoothed analysis)*, siehe auch Spielman und Teng (2001), gilt zuweilen als Mischung aus Worst-Case- und Average-Case-Analyse und damit als in praktischer Hinsicht besonders aussagekräftig.

Nach den Worst-Case-Betrachtungen hinsichtlich der Approximationsgüten, die der (1+1)-EA und RLS für das PARTITION-Problem in polynomieller Zeit erreichen können, wollen wir, motiviert von den Ergebnissen für deterministische Algorithmen, uns auch an einer Average-Case-Analyse für randomisierte Suchheuristiken versuchen. In Bezug auf einfache, deterministische Approximationsalgorithmen für PARTITION existieren bereits Average-Case-Analysen für große Klassen von Verteilungen der Eingabeinstanzen (siehe z. B. Coffman und Whitt, 1995; Frenk und Rinnooy Kan, 1986). Da unsere Ergebnisse zu den ersten Average-Case-Analysen randomisierter Suchheuristiken zählen, beschränken wir uns in Abschnitt 4.5 auf zwei wohl bekannte Verteilungen für die Eingaben, nämlich die Gleichverteilung sowie die Exponentialverteilung. Wir vergleichen unsere Ergebnisse mit den erwähnten Average-Case-Ergebnissen für deterministische Algorithmen.

4.1 Das Optimierungsproblem PARTITION

Bei PARTITION handelt es sich um eines der am einfachsten zu formulierenden kombinatorischen Probleme. Wir geben es zunächst als Entscheidungsproblem an.

> *Seien $w_1, \ldots, w_n$ natürliche Zahlen. Gibt es eine Aufteilung $I \subseteq \{1, \ldots, n\}$, sodass $\sum_{i \in I} w_i = \sum_{i \notin I} w_i$ gilt?*

Es lassen sich natürlich leicht Spezialfälle von Instanzen identifizieren, bei denen es keine Aufteilung der beschriebenen Form geben kann. Andererseits ist seit Jahrzehnten die NP-Vollständigkeit dieses Entscheidungsproblems bekannt. Gleichwohl lässt sich mit einem Ansatz der dynamischen Programmierung leicht ein pseudopolynomieller Algorithmus finden, dessen Laufzeit durch $O\big(n \sum_{i=1}^{n} w_i\big)$ beschränkt ist (Garey und Johnson, 1979).

Der genannte pseudopolynomielle Algorithmus löst zugleich die kanonische Optimierungsvariante von PARTITION, indem er eine Indexmenge $I \subseteq \{1, \ldots, n\}$ ermittelt, sodass das Maximum der Ausdrücke $\sum_{i \in I} w_i$ und $\sum_{i \notin I} w_i$ minimal wird. Dieses kombinatorische Optimierungsproblem mit dem beschriebenen Maximum als Wert einer Lösung, d. h. Aufteilung I, sollen wir im Folgenden stets vor Augen haben, wenn wir vom Optimierungsproblem PARTITION oder – einfacher – vom Partitionsproblem reden. In diesem Zusammenhang bietet es sich an, die Aufgabenstellung als Lastverteilungsproblem aufzufassen, indem wir die w_i als Bearbeitungszeiten von n Aufgaben ansehen, die so auf zwei identische Maschinen (z. B. Prozessoren) zu verteilen sind, dass die maximale Laufzeit der beiden Maschinen minimal wird.

Bevor die NP-Härte des Partitionsproblems bekannt war, kamen dafür einfache, deterministische Heuristiken in der Gestalt von Lastverteilungsalgorithmen zum Vorschein. Ein einfacher Greedy-Algorithmus, der als *list scheduling algorithm* (Graham, 1969) bekannt ist, bearbeitet die Aufgaben mit Bearbeitungszeiten $w_1, \ldots, w_n$ in der vorliegenden Reihenfolge und lastet die i-te Aufgabe auf diejenige der beiden Maschinen ein, die nach der Verteilung der ersten $i - 1$ Aufgaben die geringere Gesamtbearbeitungszeit aufweist (im Falle gleicher Auslastung wird eine beliebige Maschine gewählt). Ebenso geht der Algorithmus LPT *(longest processing time)* vor, sortiert aber zuvor die Aufgaben gemäß absteigender Bearbeitungszeit. In der Sprechweise einfacher Greedy-Algorithmen, die für das verwandte Bin-Packing-Problem existieren, könnten wir die beiden Ansätze heutzutage als *worst fit* bzw. *worst fit decreasing* bezeichnen.

Graham (1969) gibt einen Überblick über die Approximationsfähigkeit der einfachen Heuristiken und beweist, dass der List-Scheduling-Algorithmus stets eine Approximationsgüte von 3/2 und LPT sogar eine von 7/6 erreicht. (Unter der Approximationsgüte verstehen wir bei dem vorliegenden Minimierungsproblem stets [eine obere Schranke für] den Quotienten aus dem Wert der Lösung der Heuristik und dem Wert einer optimalen Lösung.) Einfache Worst-Case-Instanzen zeigen, dass diese Approximationsgüten im Allgemeinen nicht zu verbessern sind. Ohne den Begriff zu verwenden, stellt

Graham (1969) zudem ein polynomielles Approximationsschema für das Partitionsproblem vor, das für jedes konstante $\varepsilon > 0$ in polynomieller Laufzeit bezüglich n eine Lösung berechnet, deren Approximationsgüte (mindestens so gut wie) $1 + \varepsilon$ ist. Seit längerem ist für das Problem sogar ein echt polynomielles Approximationsschema bekannt, das in einer Laufzeit, die durch ein Polynom sowohl in n als auch in ε^{-1} beschränkt ist, eine Lösung mit Approximationsgüte $1+\varepsilon$ ausgibt. Für weitere Einblicke in die Thematik der Approximationsschemata siehe beispielsweise Hochbaum (1997). Außerdem sollte nicht unerwähnt bleiben, dass das Partitionsproblem einen Spezialfall des Rucksackproblems darstellt und ebenfalls im Rahmen einer geglätteten Analyse in erwarteter Polynomialzeit gelöst werden kann (Beier und Vöcking, 2004).

All diese positiven Resultate und die einfache Beschreibung des Partitionsproblems motivieren, wieso wir die Approximationsfähigkeit einfacher randomisierter Suchheuristiken auf diesem Problem analysieren. Anders als in Kapitel 3 konzentrieren wir uns in diesem Kapitel auf den (1+1)-EA und betrachten RLS nebenbei, da es sich herausstellen wird, dass die Analyse der globalen Suchheuristik (1+1)-EA hier nicht viel schwieriger als die von RLS ist und darüber hinaus der (1+1)-EA in gewissem Sinne überlegen ist. Wir kommen daher nun zur Formulierung der Arbeitsweise der Suchheuristiken für unser kombinatorisches Optimierungsproblem. Sie benutzen als natürliche Kodierung einer Aufteilung, d. h. einer Indexmenge I, ihren charakteristischen Vektor und als Zielfunktion den Wert dieser Lösung. Damit arbeiten die Suchheuristiken im Black-Box-Szenario, können also nur anhand von Auswertungen der Zielfunktion Informationen über die konkrete Instanz gewinnen.

Definition 4.1.1 ((1+1)-EA/RLS für das Partitionsproblem) *Es seien natürliche Zahlen $w_1, \dots, w_n$ gegeben. Sei weiterhin $f_{w_1,\dots,w_n}\colon \{0,1\}^n \to \mathbb{R}$ definiert als*

$$f_{w_1,\dots,w_n}(x) \;:=\; \max\left\{\sum_{i=1}^{n} w_i x_i,\; \sum_{i=1}^{n} w_i(1-x_i)\right\}.$$

Der (1+1)-EA für das Partitionsproblem *ist dann der (1+1)-EA aus Definition 2.3.1 mit Zielfunktion $-f_{w_1,\dots,w_n}$. Analog wird* RLS für das Partitionsproblem *definiert.*

Die Funktion $f_{w_1,\dots,w_n}$ gibt für jede als charakteristischer Vektor kodierte Lösung $I \subseteq \{1,\dots,n\}$ für das Partitionsproblem per definitionem den Wert dieser Lösung an. Da die Zielfunktion gemäß Aufgabenstellung zu minimieren ist, setzen wir die Suchheuristiken formal auf $-f_{w_1,\dots,w_n}$ ein, betrachten aber für einen aktuellen Suchpunkt x stets den Wert $f_{w_1,\dots,w_n}(x)$ und tun so, als ob die Heuristik diese Funktion minimiere. Offensichtlich ist für alle $x \in \{0,1\}^n$ die Identität $f_{w_1,\dots,w_n}(x) = f_{w_1,\dots,w_n}(\overline{x})$ erfüllt, wenn $\overline{x}$ das bitweise Komplement von x bezeichnet. Zielfunktionen mit dieser Eigenschaft, die zuweilen als *spin-flip symmetry* bezeichnet wird, sind im Zusammenhang mit der Analyse evolutionärer Algorithmen von besonderem Interesse (siehe z. B. Dietzfelbinger, Naudts, van Hoyweghen und Wegener, 2003).

Wir einigen uns jetzt auf einige Konventionen, die sich im Zusammenhang mit dem (1+1)-EA und RLS für das Partitionsproblem als hilfreich erweisen. Zunächst gehen wir o. B. d. A. überall außer in Abschnitt 4.5 davon aus, dass die Zahlen $w_1, \ldots, w_n$ gemäß $w_1 \geq \cdots \geq w_n$ monoton fallend sortiert sind. Weiterhin führen wir für die Summe der w_i die Abkürzung $w := w_1 + \cdots + w_n$ ein. Um eine griffigere Beschreibung des Optimierungsproblems zu erhalten, sprechen wir oft von den Zahlen $w_1, \ldots, w_n$ als *Objekten* und reden davon, dass diese in zwei *Eimer* gepackt werden. So bezeichnen wir die Zahl w_i auch als *Volumen* des i-ten Objektes und nennen die Summe $\sum_{i=1}^{n} w_i x_i$ das *Volumen des ersten Eimers*; Analoges für den zweiten Eimer. Derjenige Eimer, dessen Volumen $f_{w_1,\ldots,w_n}$ maximiert, nennen wir gerne den *volleren Eimer*. Schließlich verzichten wir, falls Missverständnisse ausgeschlossen sind, gerne auf eine Indizierung, und bezeichnen die zu minimierende Funktion schlichtweg mit f.

4.2 Grundlegende Beweistechniken

Innerhalb dieses Kapitels werden wir im Wesentlichen zwei Beweistechniken einsetzen, um die Zeit, bis der (1+1)-EA (bzw. RLS) eine Lösung gewisser Güte erzeugt hat, zu beschränken. Im vorangehenden Abschnitt haben wir bereits angedeutet, dass die ersten Resultate lediglich auf lokalen Fortschritten der Suchheuristiken aufbauen. Hier betrachten wir zunächst hinreichende Bedingungen dafür, dass die Suchheuristiken für das Partitionsproblem den f-Wert mithilfe von Mutationen, die nur ein Bit kippen, verringern können. Mithin sind wir an einer Charakterisierung lokaler Optima (vgl. Definition 2.3.5), in diesem Fall lokaler Minima, interessiert. Die erste Überlegung ist, dass das Volumen des größten Objekts im volleren Eimer möglichst klein sein sollte. Wir definieren etwas Ähnliches, lassen aber Objekte, die den Eimer zum volleren Eimer machen, außen vor.

Definition 4.2.1 (kritisches Volumen) *Sei $w_1, \ldots, w_n$ eine Instanz für das Partitionsproblem, sei $\ell \geq w/2$ eine untere Schranke für den optimalen $f_{w_1,\ldots,w_n}$-Wert, sei $x \in \{0,1\}^n$ eine Aufteilung mit $f_{w_1,\ldots,w_n}(x) > \ell$ und seien $w_{i_1} \geq w_{i_2} \geq \cdots \geq w_{i_k}$ alle Objekte, die bez. x im volleren Eimer liegen, sortiert in monoton fallender Reihenfolge ihrer Volumina. Sei $r := i_j$ für das kleinste j, sodass $w_{i_1} + \cdots + w_{i_j} > \ell$ ist. Das Volumen w_r heißt dann* kritisches Volumen *(in Bezug auf $w_1, \ldots, w_n$, ℓ und x).*

Die einfache Idee hinter dieser komplizierten Definition ist wie folgt. Angenommen, x sei der aktuelle Suchpunkt des (1+1)-EA oder von RLS, x führe zu $f(x) > \ell$ für ein $\ell \geq w/2$ und wir kennen ein v, sodass v eine obere Schranke für das kritische Volumen entsprechend Definition 4.2.1 bezüglich der Instanz, ℓ und x darstellt. Sei r der kleinste Index $i \in \{1, \ldots, n\}$, sodass $w_i \leq v$ gilt. Aufgrund der Sortierung $w_1 \geq \cdots \geq w_n$ ist damit auch w_r eine obere Schranke für das kritische Volumen. Laut der Definition des kritischen Volumens befindet sich ein Objekt $w_{r'}$ mit $r' \geq r$, d. i. ein Objekt mit einem Volumen von höchstens w_r, im volleren Eimer. Gemäß Voraussetzung gilt $\ell \geq w/2$.

Wenn nun der Fall vorliegt, dass $f(x) \geq \ell + v$ ist, kann $w_{r'}$ vom volleren in den leereren Eimer gelegt werden, sodass sich der f-Wert um $w_{r'}$ verringert. Dies korrespondiert zu einer Mutation genau eines Bits, d. h., der Suchpunkt x ist lokal verbesserbar.

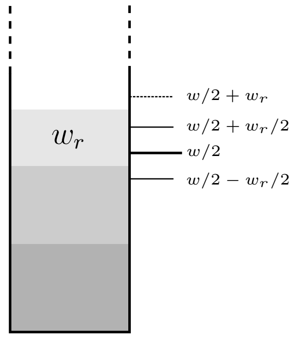

Abbildung 4.1: Sonderfall für den f-Wert beim Partitionsproblem

Mit einem weiteren Argument lässt sich die hinreichende Bedingung für lokal verbesserbare Suchpunkte sogar noch verstärken. Wenn der aktuelle Suchpunkt x immer noch ein kritisches Volumen von höchstens $w_r \leq v$ hat und die Bedingung $\ell + v/2 < f(x) < \ell + v$ erfüllt, gibt es immer noch ein $w_{r'}$ mit $r' \geq r$ im volleren Eimer; jedoch muss dessen Bewegung den f-Wert nicht notwendigerweise um $w_{r'}$ verringern, da das Objekt größer als $f(x) - w/2$ sein und sich infolgedessen der vollere Eimer ändern kann. Was können wir folgern, wenn $w_{r'}$ ein Volumen von mehr als $f(x) - w/2$, aber andererseits auch ein Volumen von höchstens w_r besitzt? Wenn $w_{r'}$ in den leereren Eimer gelegt wird, wird der vollere Eimer zum leereren Eimer, aber dessen Volumen beträgt anschließend immer noch mehr als $\ell + w_r/2 - w_r \geq w/2 - w_r/2$. Folglich ist das Volumen des volleren Eimers unter $w/2 + w_r/2 \leq \ell + v/2$ gesunken. Diesen Fall illustriert Abbildung 4.1 für $\ell = w/2$, $v = w_r$ und $r' = r$.

Wir fassen zusammen. Wenn $f(x) > \ell + v/2$ gilt, gibt es mindestens ein Objekt im volleren Eimer, dessen Bewegung den f-Wert um sein Volumen verringert oder den f-Wert unter $\ell + v/2$ sinken lässt. In den Anwendungen wird zu guter Letzt klar werden, wieso wir als untere Schranke ℓ nicht immer $w/2$ wählen.

Wenn wir nun wissen, dass das kritische Volumen immer durch einen (vorzugsweise kleinen) Wert nach oben beschränkt ist, können wir daraus Resultate herleiten, die sich auf die Approximationsgüte, die der (1+1)-EA und RLS in polynomieller Zeit erreichen können, beziehen. Im folgenden Lemma müssen wir außerdem den trivialen Fall, dass $w_1 \geq w/2$ gilt, abdecken.

Lemma 4.2.2 *Seien $w_1, \dots, w_n$ und ℓ wie in Definition 4.2.1. Angenommen, ab einem Zeitpunkt beträgt das kritische Volumen bez. $w_1, \dots, w_n, \ell$ und des aktuellen Suchpunktes des (1+1)-EA für das Partitionsproblem stets höchstens v. Dann erreicht der (1+1)-EA einen f-Wert von höchstens $\ell + v/2$, falls $w_1 < w/2$, und von höchstens w_1 andernfalls in einer erwarteten Zahl von weiteren $O(n^2)$ Schritten. Dieselbe Aussage gilt für RLS für das Partitionsproblem.*

Beweis: Sei r der kleinste Index $i \in \{1, \dots, n\}$, sodass $w_i \leq v$ gilt. Wir sehen uns den Lauf der Suchheuristik nur ab dem im Lemma genannten speziellen Zeitpunkt an und wenden die Ranggruppenmethode (vgl. Lemma 2.3.10) an. Zunächst betrachten wir den Fall $w_1 < w/2$. Sei $s := w_r + \cdots + w_n$, d. h., s ist die Summe der Objekte, die höchstens so groß sind wie w_r. Nach der Voraussetzung des Lemmas und der Definition des kritischen Volumens folgt, dass $f(x) \leq \ell + s$ für alle weiteren Suchpunkte der Suchheuristik gilt. Seien die Ranggruppen nun definiert gemäß

$$L_i := \left\{ x \;\middle|\; \ell + s - \sum_{j=r}^{r+i-1} w_j \;\geq\; f(x) \;>\; \ell + s - \sum_{j=r}^{r+i} w_j \right\}$$

für $0 \leq i \leq n - r$ und $L_{n-r+1} := \{x \mid f(x) = \ell\}$. (Möglicherweise sind die Mengen ab einem gewissen Index leer.) Sei nun ein aktueller Suchpunkt x mit $f(x) > \ell + w_r/2$ betrachtet. Wir haben uns oben überlegt, dass wegen der Definition des kritischen Volumens dann ein Objekt aus $w_r, \ldots, w_n$ im volleren Eimer liegt, das mithilfe einer Mutation, die genau ein Bit kippt (einer sog. *1-Bit-Mutation*), in den leereren Eimer gelegt werden kann, und dass der f-Wert damit um sein Volumen oder sogar sofort unter $\ell + w_r/2$ sinkt. Wir nehmen pessimistisch an, dass Letzteres nicht eintritt. Wenn $x \in L_i$, wissen wir sogar, dass ein solches Objekt unter $w_r, \ldots, w_{r+i}$ existiert. Wenn dieses in den leereren Eimer gelegt wird, entsteht aufgrund der Sortierung der Objekte ein $x' \in L_j$ mit $j > i$. Da eine spezielle 1-Bit-Mutation für RLS eine Wahrscheinlichkeit von $1/n$ und für den (1+1)-EA von mindestens $(1/n)(1 - 1/n)^{n-1} \geq 1/(en)$ hat, haben wir für alle L_i Verlassenswahrscheinlichkeiten von $\Omega(1/n)$. Da es höchstens $n - r + 2$ Ranggruppen gibt, folgt mit Lemma 2.3.10, dass die Suchheuristiken im Erwartungswert nach $O(n^2)$ weiteren Schritten zu einem f-Wert von höchstens $\ell + w_r/2$ kommen.

Wenn $w_1 \geq w/2$ gilt, können wir die vorstehenden Argumente mit den speziellen Werten $\ell := w_1$ und $r := 2$ anwenden. Die Besonderheit ist nun, dass, falls $f(x) > \ell$ gilt, ein Objekt mit einem Volumen von höchstens $f(x) - \ell$ im volleren Eimer sein muss. Also kann sich die Suchheuristik nicht in einem lokalen Optimum befinden und L_{n-r+1} mit 1-Bit-Mutationen erreichen. □

Wir kommen nun zur zweiten Technik. Wenn wir uns mit etwas höheren f-Werten als denjenigen, die Lemma 4.2.2 garantiert, begnügen, können sogar noch deutlich kleinere Schranken für die dazu benötigte erwartete Zeit gezeigt werden. Eine Idee des Beweises des folgenden Lemmas besteht darin, die erwartete Abnahme des f-Wertes in Schritten, die nur ein Bit kippen, zu untersuchen. Etwas Ähnliches führen auch Neumann und Wegener (2004) im Zusammenhang mit der Berechnung minimaler Spannbäume mithilfe des (1+1)-EA vor.

Lemma 4.2.3 *Seien $w_1, \ldots, w_n$ und ℓ wie in Definition 4.2.1. Angenommen, ab einem Zeitpunkt ist das kritische Volumen bez. $w_1, \ldots, w_n, \ell$ und des aktuellen Suchpunktes des (1+1)-EA (von RLS) für das Partitionsproblem stets höchstens v. Dann gilt für jedes $\gamma \geq 1$ und $0 < \delta \leq 1$ das Folgende. Mit einer Wahrscheinlichkeit von mindestens $1 - 1/\gamma$ erreicht der (1+1)-EA (RLS) einen f-Wert von höchstens $\ell + v/2 + \delta w/2$, falls $w_1 < w/2$, und von sogar höchstens $w_1 + \delta w/2$ andernfalls in höchstens $\lceil en \ln(\gamma/\delta) \rceil$ (höchstens $\lceil n \ln(\gamma/\delta) \rceil$) weiteren Schritten. Außerdem ist die erwartete Zahl dafür erforderlicher weiterer Schritte durch $2\lceil en \ln(2/\delta) \rceil$ (durch $2\lceil n \ln(2/\delta) \rceil$) beschränkt.*

Beweis: Sei r der kleinste Index $i \in \{1, \dots, n\}$, sodass $w_i \leq v$ gilt. Wir sehen uns den Lauf der Suchheuristik nur ab dem im Lemma genannten speziellen Zeitpunkt an. Zunächst betrachten wir den Fall $w_1 < w/2$. Im Folgenden sagen wir, dass ein Objekt einen Beitrag zum f-Wert leistet, wenn es sich im volleren Eimer befindet. Wenn für einen aktuellen Suchpunkt x die Beziehung $f(x) > \ell + w_r/2$ erfüllt ist, sind wir interessiert am Beitrag der so genannten kleinen Objekte $w_r, \dots, w_n$ zum aktuellen f-Wert. Wir wollen dessen erwartete Abnahme mithilfe der Potenzialfunktion $p(x) := \max\{f(x) - \ell - w_r/2, 0\}$ nach unten beschränken. Die entscheidende Beobachtung ist nun, dass $f(x) - \ell$ aufgrund der Definition des kritischen Volumens und der Beschränkung, die nach dem speziellen Zeitpunkt vorliegt, eine untere Schranke für den Beitrag der kleinen Objekte zum f-Wert darstellt. Damit ist auch der p-Wert eine solche untere Schranke. Weiterhin beobachten wir, dass, solange $p(x) > 0$ gilt, alle Schritte, die mit einer 1-Bit-Mutation ein kleines Objekt vom volleren in den leereren Eimer legen, akzeptiert werden und den p-Wert um das Volumen des bewegten Objektes verringern oder den f-Wert sofort unter $\ell + w_r/2$ sinken lassen. Wir nehmen pessimistisch an, dass der letzte Fall nicht eintritt. Sei $p_0 > 0$ ein aktueller p-Wert. Da jede spezielle 1-Bit-Mutation im (1+1)-EA (bei RLS) eine Wahrscheinlichkeit von mindestens $1/(en)$ (genau $1/n$) hat, beträgt die erwartete Verringerung des f-Wertes und somit des p-Wertes infolge des nächsten Schrittes mindestens $p_0/(en)$ (mindestens p_0/n). Danach ist der erwartete p-Wert höchstens $(1 - 1/(en))p_0$ bzw. $(1 - 1/n)p_0$. Die dritte wichtige Beobachtung ist, dass die vorstehende Betrachtung gültig bleibt, wenn p_0 lediglich ein Erwartungswert ist. Der Grund dafür liegt darin, dass das Verhalten der Mutationsoperatoren der Suchheuristiken in verschiedenen Schritten unabhängig ist und damit letztendlich der obige Faktor $1/(en)$ bzw. $1/n$ aus der Definition des Erwartungswertes ausgeklammert werden kann. Also ist der erwartete p-Wert p_t nach t Schritten höchstens $(1 - 1/(en))^t p_0$ bzw. $(1 - 1/n)^t p_0$.

Mit $t' := \lceil en \ln(\gamma/\delta) \rceil$ bzw. $t' := \lceil n \ln(\gamma/\delta) \rceil$ folgt nun

$$p_{t'} \leq \frac{\delta p_0}{\gamma} \leq \frac{\delta w}{2\gamma}.$$

Da die p-Werte nicht negativ sind, können wir die Markoffungleichung anwenden und erhalten, dass $p_{t'} \leq \delta w/2$ mit einer Wahrscheinlichkeit von mindestens $1 - 1/\gamma$ gilt. Da die vorangehenden Analysen keine Annahmen über p_0 treffen, erhalten wir mit einer Wiederholung unabhängiger Phasen und $\gamma = 2$ insgesamt die Aussagen des Lemmas für den Fall $w_1 < w/2$.

Falls $w_1 \geq w/2$, können wir die vorstehenden Argumente mit den speziellen Werten $\ell := w_1$ und $r := 2$ anwenden. Der einzige Unterschied besteht darin, dass, solange $f(x) > \ell$ ist, sich mindestens ein Objekt mit einem Volumen von höchstens $f(x) - \ell$ im volleren Eimer befinden muss. Also kann sich die Suchheuristik in keinem lokalen Optimum befinden. Indem wir den p-Wert gemäß $p(x) := f(x) - \ell$ redefinieren, folgt das Lemma auch in diesem Fall. □

4.3 Worst-Case-Approximationsgüten für einzelne Läufe

Im letzten Abschnitt haben wir anhand von Lemma 4.2.2 gezeigt, dass der (1+1)-EA bzw. RLS den optimalen f-Wert für manche trivialen Instanzen in erwarteter polynomieller Zeit findet. Es entstünde offenbar ein Widerspruch zu gängigen komplexitätstheoretischen Annahmen, wenn dies auch für allgemeine Instanzen gälte. Wir konzentrieren uns also nun auf die angekündigten Resultate zur Approximationsfähigkeit für beliebige Instanzen. Im Folgenden reden wir erneut von einer Approximationsgüte von g, wenn wir eine Approximationsgüte meinen, die mindestens so gut ist wie g, d. h., dass der Quotient aus dem Wert der ermittelten Lösung und dem Wert einer optimalen Lösung höchstens g beträgt.

Theorem 4.3.1 *Sei ε eine beliebige Konstante mit $0 < \varepsilon < 1$. Dann erzeugen der (1+1)-EA und RLS für das Partitionsproblem auf jeder Instanz eine Lösung mit einer Approximationsgüte von $4/3 + \varepsilon$ in einer erwarteten Zeit von $O(n)$ und eine Lösung mit einer Approximationsgüte von $4/3$ in einer erwarteten Zeit von $O(n^2)$.*

Beweis: Für die trivialen Instanzen mit $w_1 \geq w/2$ folgen sogar beide Aussagen mit $\delta := 1/3$ aus Lemma 4.2.3.

Sei nun $w_1 < w/2$. Wir setzen $\ell := w/2$ und unterscheiden zwei Fälle. Der erste Fall tritt bei $w_1 + w_2 > 2w/3$ ein. Dies impliziert $w_1 > w/3$ und damit $w - w_1 < 2w/3$. Wenn der initiale Suchpunkt der Suchheuristik also w_1 und w_2 in denselben Eimer legt, wird eine Mutation, die die Objekte trennt, indem sie w_2 in den leereren Eimer legt, akzeptiert und die beiden Objekte werden anschließend nie mehr im selben Eimer liegen. Eine 1-Bit-Mutation hat bei beiden Suchheuristiken eine Wahrscheinlichkeit von $\Theta(1/n)$, sodass die erwartete Zeit, bis w_1 und w_2 getrennt werden, durch $O(n)$ beschränkt ist.

Nach der Trennung ist sichergestellt, dass das kritische Volumen im Sinne von Definition 4.2.1 immer durch w_3 nach oben beschränkt ist. Wegen $w_3 + \cdots + w_n < w/3$ gilt $w_i \leq w/3$ für $i \geq 3$. Mittels Lemma 4.2.3 für $\delta := \varepsilon$ erhalten wir nach einer erwarteten Zeit von $O(n)$ eine Lösung mit einem f-Wert von höchstens

$$\frac{w}{2} + \frac{w/3}{2} + \varepsilon \cdot \frac{w}{2} \;=\; \frac{w}{2}\left(\frac{4}{3} + \varepsilon\right),$$

d. h. mit einer Approximationsgüte von $4/3 + \varepsilon$. Mit Lemma 4.2.2 folgt die Approximationsgüte von $4/3$ in $O(n^2)$ erwarteten Schritten.

Im verbleibenden Fall ist $w_1 + w_2 \leq 2w/3$. Zusammen mit der Sortierung der w_i folgt $w_i \leq w/3$ für $i \geq 2$. Da $w_1 < w/2$, ist das kritische Volumen also immer durch $w_2 \leq w/3$ nach oben beschränkt. Deshalb folgt das Theorem mit derselben Rechnung wie oben. □

Wir würden gerne zeigen, dass der Wert 4/3 aus Theorem 4.3.1 im Allgemeinen nicht verbessert werden kann. Wir zeigen, dass der Wert zumindest fast scharf ist, indem wir folgende Beinahe-Worst-Case-Instanz $W^*(\varepsilon)$ aufstellen.

Sei n gerade und sei $\varepsilon > 0$ eine beliebig kleine (und nicht zu große) Konstante. Dann enthält $W^*(\varepsilon)$ zwei Objekte w_1 und w_2 mit einem Volumen von jeweils $1/3 - \varepsilon/4$ und $n-2$ Objekte mit einem Volumen von je $(1/3+\varepsilon/2)/(n-2)$. Bei rationalen Werten von ε können wir durch Multiplikation mit einem Hauptnenner zu natürlichen Volumina kommen. Das Gesamtvolumen der Instanz beträgt bei $W^*(\varepsilon)$ zur Übersicht 1. Wir bemerken, dass es exponentiell viele perfekte Aufteilungen mit Wert 1/2 gibt.

Theorem 4.3.2 *Sei ε eine beliebige Konstante mit $0 < \varepsilon < 1/3$. Mit einer Wahrscheinlichkeit von $\Omega(1)$ benötigen der (1+1)-EA und RLS für das Partitionsproblem dann $n^{\Omega(n)}$ Schritte, um für $W^*(\varepsilon)$ eine Lösung zu finden, deren Approximationsgüte besser als $4/3 - \varepsilon$ ist.*

Beweis: Die Beweisidee besteht darin zu zeigen, dass die Suchheuristik in eine Situation gerät, in der w_1 und w_2 im selben Eimer liegen und mindestens $k := n-2-(n-2)\varepsilon/2$ der verbleibenden, so genannten kleinen Objekte sich im anderen Eimer befinden. Das Gesamtvolumen von mindestens k kleinen Objekten beträgt mindestens

$$\left(1 - \frac{\varepsilon}{2}\right)\left(\frac{1}{3} + \frac{\varepsilon}{2}\right) \;=\; \frac{1}{3} + \frac{\varepsilon}{3} - \frac{\varepsilon^2}{4} \;>\; \frac{1}{3} + \frac{\varepsilon}{4},$$

da $\varepsilon < 1/3$ gilt. Jede Lösung, die w_1 und w_2 in denselben Eimer legt, führt zu einem Volumen von mindestens $2/3 - \varepsilon/2$ im volleren Eimer und besitzt damit keine bessere Approximationsgüte als $4/3 - \varepsilon$. Wenn sich die Suchheuristik in der beschriebenen Situation befindet, muss sie aber in einem Schritt, der die beiden großen Objekte trennt, gleichzeitig kleine Objekte mit einem Gesamtvolumen von mindestens $\varepsilon/4$ bewegen, damit dieser Schritt akzeptiert wird. Andererseits werden in der Situation Schritte, die nur kleine Objekte bewegen und die Gesamtzahl kleiner Objekte im leereren Eimer verringern, trivialerweise nie akzeptiert.

Nach der obigen Rechnung ist das Gesamtvolumen von $(n-2)\varepsilon/2$ kleinen Objekten immer noch kleiner als $\varepsilon/4$. Um die beschriebene Situation zu verlassen, muss die Suchheuristik also in einem Schritt $\Omega(n)$ Bits kippen. Die zugehörige Wahrscheinlichkeit für den (1+1)-EA ist $n^{-\Omega(n)}$ und für RLS ist sie sogar 0. Die Wahrscheinlichkeit, eine solche Mutation innerhalb von n^{cn} Schritten zu erhalten, ist bei beiden Suchheuristiken also $n^{-\Omega(n)}$, wenn die Konstante c klein genug ist.

Wir müssen noch zeigen, dass beide Suchheuristiken mit Wahrscheinlichkeit $\Omega(1)$ in die Situation kommen. Dazu betrachten wir den initialen Suchpunkt der Heuristik. Dieser ordnet w_1 und w_2 mit einer Wahrscheinlichkeit von 1/2 demselben Eimer zu. Unter dieser Annahme schätzen wir die Wahrscheinlichkeit ab, dass genügend viele kleine Objekte in den leereren Eimer gelegt werden, bevor eines der Bits für die beiden großen Objekte kippt. Bei beiden Suchheuristiken ist für jede Konstante $c > 0$

die Wahrscheinlichkeit, dass innerhalb einer Phase der Länge cn niemals eines dieser Bits kippt, mindestens $(1 - 2/n)^{cn} = \Omega(1)$. Wir arbeiten jetzt unter der Annahme, dass eine solche Phase eintritt. In der Phase wird jede 1-Bit-Mutation, die ein kleines Objekt in den leereren Eimer legt, akzeptiert. Jetzt greifen wir auf ein ähnliches Argument wie im Beweis von Lemma 4.2.3 zurück und analysieren den (erwarteten) Beitrag der kleinen Objekte zum Volumen des volleren Eimers. Offenkundig genügt es, diesen Beitrag in der Phase auf einen Bruchteil von höchstens $\varepsilon/2$ seines anfänglichen Beitrags zu reduzieren, um zu mindestens k Objekten im leereren Eimer zu kommen. Jeder Schritt der Phase führt zu einer erwarteten Verringerung des Beitrags um einen Bruchteil der Größe mindestens $1/(en)$ in Bezug auf den (1+1)-EA und von mindestens $1/n$ in Bezug auf RLS. Da ε eine Konstante ist, reichen also bei beiden Suchheuristiken $O(n)$ Schritte aus, um den Beitrag im Erwartungswert auf höchstens $\varepsilon/4$ des anfänglichen Beitrags zu senken. Mit der Markoffungleichung folgt, dass $O(n)$ Schritte mit einer Wahrscheinlichkeit von mindestens $1/2$ ausreichen, um den Beitrag auf einen $\varepsilon/2$-Bruchteil zu senken. Da c beliebig groß gewählt werden kann, ist die Behauptung gezeigt. □

Abschließend vergleichen wir die Approximationsgüte $4/3 + \varepsilon$, die der (1+1)-EA und RLS für das Partitionsproblem innerhalb von $O(n)$ Schritten erreichen, mit den Ergebnissen für die in Abschnitt 4.1 vorgestellten einfachen, deterministischen Approximationsalgorithmen. Einerseits ist die Approximationsgüte besser als $3/2$, also die Güte des List-Scheduling-Algorithmus. Auf der anderen Seite kann man bereits mit dem LPT-Algorithmus eine Güte von $7/6$ garantieren, benötigt aufgrund der Sortierung der Objekte allerdings $\Omega(n \log n)$ Schritte.

Wir beenden diesen Abschnitt mit der Zusammenfassung, dass wir zum Nachweis oberer Schranken der Approximationsgüten bislang nur anhand von lokalen Schritten des (1+1)-EA argumentiert haben. Die untere Schranke aus Theorem 4.3.2 zeigt auch, dass wir auf keine viel besseren oberen Schranken hoffen können und damit die Eigenschaft des (1+1)-EA, global suchen zu können, bei unseren Worst-Case-Analysen nahezu wertlos ist. Im Gegensatz zu Kapitel 3 ist uns die Analyse der globalen Suchheuristik aber nicht schwerer als die der lokalen gefallen.

4.4 Ein polynomielles, randomisiertes Approximationsschema mit parallelen Läufen

Mit Theorem 4.3.2 haben wir gezeigt, dass es eine durch eine Konstante beschränkte Wahrscheinlichkeit gibt, sodass der (1+1)-EA und RLS in polynomieller Zeit nicht näher als um einen Faktor $4/3 - \varepsilon$ an das Optimum herankommen. Der Grund dafür lag darin, dass die Suchheuristik bereits zu Anfang die Objekte großen Volumens ungünstig platziert. Jedoch kann man sich im Hinblick auf die Instanz $W^*(\varepsilon)$ überlegen, dass der (1+1)-EA und RLS mit Wahrscheinlichkeit $\Omega(1)$ sogar eine perfekte

Partition in $O(n^2)$ Schritten finden können, wenn anfangs die beiden großen Objekte in verschiedene Eimer fallen. Also scheint das Worst-Case-Verhalten nicht mit einer Wahrscheinlichkeit $1 - o(1)$ einzutreten und somit erscheinen parallele Läufe *(multistarts)* hilfreich, um die Erfolgswahrscheinlichkeit in polynomiell vielen Schritten zu erhöhen.

In diesem Abschnitt wollen wir die vorangehende Überlegung auf beliebige Instanzen erweitern. Wenn wir in polynomieller Zeit eine $(1+\varepsilon)$-Approximation erhalten wollen, wäre es mit Blick auf eine Anwendung von Lemma 4.2.2 (im kommenden Theorem werden wir aber Lemma 4.2.3 benutzen) schön, wenn das kritische Volumen stets durch εw nach oben beschränkt wäre. Aufgrund der Sortierung $w_1 \geq \cdots \geq w_n$ sind alle Objekte mit einem Index von mindestens $s := \lceil 1/\varepsilon \rceil$ tatsächlich in ihrem Volumen durch εw beschränkt. Dies leitet uns zur entscheidenden Idee, dass die Suchheuristik die großen Objekte $w_1, \ldots, w_{s-1}$ so günstig verteilen sollte, dass das kritische Volumen stets höchstens w_s ist. Interessanterweise stellt dies im Wesentlichen dieselbe Idee dar, die auch das oben erwähnte, altbekannte polynomielle Approximationsschema von Graham (1969) benutzt.

Theorem 4.4.1 *Sei $4/n \leq \varepsilon < 1$. Dann erzeugt der (1+1)-EA für das Partitionsproblem mit einer Wahrscheinlichkeit von mindestens $2^{-(\mathrm{e}\log \mathrm{e}+\mathrm{e})(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))-\lceil 2/\varepsilon\rceil}$ innerhalb von $\lceil \mathrm{e}n \ln(2/\varepsilon)\rceil$ Schritten eine Lösung mit einer Approximationsgüte von $1+\varepsilon$. Dasselbe gilt für RLS bei einer Wahrscheinlichkeit von mindestens $2^{-(\log \mathrm{e}+1)(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))-\lceil 2/\varepsilon\rceil}$.*

Beweis: Sei $s := \lceil 2/\varepsilon \rceil \leq n/2+1$. Wegen $w_1 \geq \cdots \geq w_n$ gilt $w_i \leq \varepsilon w/2$ für $i \geq s$. Im Falle, dass $w_1 + \cdots + w_{s-1} \leq w/2$ ist, folgt unmittelbar, dass das kritische Volumen bezüglich der Instanz und $\ell := w/2$ stets höchstens w_s, also höchstens $\varepsilon w/2$, beträgt. Dann folgt das Theorem aus Lemma 4.2.3, indem wir $\delta := \varepsilon$ und $\gamma := 2$ wählen.

Sei im Folgenden $w_1 + \cdots + w_{s-1} > w/2$ angenommen. Wir betrachten alle denkbaren Aufteilungen lediglich der ersten $s-1$ Objekte. Sei ℓ^* das minimale Volumen des volleren Eimers, ermittelt über alle diese Aufteilungen. Damit ist $\ell := \max\{w/2, \ell^*\}$ eine untere Schranke für den optimalen f-Wert. Da komplementäre Suchpunkte bezüglich $f_{w_1,\ldots,w_n}$ denselben Funktionswert haben, gilt für die initialen Suchpunkte beider Suchheuristiken mit einer Wahrscheinlichkeit von mindestens 2^{-s+2} das Folgende. Der Suchpunkt korrespondiert zu einer Aufteilung aller Objekte mit der Eigenschaft, dass die ersten $s-1$ Objekte einen Beitrag von höchstens ℓ zum Volumen des volleren Eimers leisten. Solange diese Eigenschaft erhalten bleibt, ist sichergestellt, dass das kritische Volumen bezüglich des aktuellen Suchpunktes und ℓ höchstens $w_s \leq \varepsilon w/2$ beträgt, d. h., wir befinden uns in der Situation des ersten Absatzes.

Sei $t := \lceil \mathrm{e}n \ln(2/\varepsilon)\rceil$. Die Wahrscheinlichkeit, dass der (1+1)-EA in einer Phase von t Schritten niemals ein Bit an einer der ersten $s-1$ Positionen kippt, ist nach unten beschränkt durch

$$\left(1 - \frac{s-1}{n}\right)^{\mathrm{e}n \ln(2/\varepsilon)+1} = \left(1 - \frac{s-1}{n}\right)^{(s-1)\mathrm{e}\ln(2/\varepsilon)(n/(s-1)-1)+(1+(s-1)\mathrm{e}\ln(2/\varepsilon))}$$

$$\geq \ \mathrm{e}^{-\mathrm{e}\ln(2/\varepsilon)(s-1)} \left(1 - \frac{s-1}{n}\right)^{s\mathrm{e}\ln(2/\varepsilon)} \ \geq \ 2^{-(\mathrm{e}\log\mathrm{e}+\mathrm{e})(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))},$$

da $\varepsilon < 1$ und zudem $(1-(s-1)/n) \geq 1/2$ gilt. Wenn das kritische Volumen in der ganzen Phase wie gewünscht beschränkt ist, können wir Lemma 4.2.3 mit $\delta := \varepsilon$ und $\gamma := 2$ anwenden und erhalten, dass der (1+1)-EA eine (1+ε)-Approximation mit einer Wahrscheinlichkeit von mindestens 1/2 in t Schritten erreicht. Indem wir die übrigen Abschätzungen mit einbeziehen, erhalten wir, dass der (1+1)-EA in t Schritten die gewünschte Approximation mit einer Wahrscheinlichkeit von mindestens

$$\frac{1}{2} \cdot 2^{-\lceil 2/\varepsilon\rceil+2} \cdot 2^{-(\mathrm{e}\log\mathrm{e}+\mathrm{e})(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))} \ \geq \ 2^{-(\mathrm{e}\log\mathrm{e}+\mathrm{e})(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))-\lceil 2/\varepsilon\rceil}$$

erzielt. Das Ergebnis für RLS leiten wir mit einer analogen Rechnung für die neue Phasenlänge $t := \lceil n\ln(2/\varepsilon)\rceil$, die sich aus Lemma 4.2.3 für RLS ergibt, her. □

Der vorangehende Beweis enthält eine wichtige Erkenntnis. Obwohl die randomisierte Suchheuristik nichts von den algorithmischen Ideen zum Entwurf von Approximationsschemata für das Partitionsproblem weiß, hilft ihr der Zufall dabei, die Arbeitsweise des klassischen Approximationsschemas von Graham (1969) nachzuahmen. Letzteres probiert alle Aufteilungen von $O(1/\varepsilon)$ großen Objekten durch, findet damit eine bestmögliche Aufteilung dieser Objekte und verteilt die kleinen Objekte dann mit einem Greedy-Ansatz. Unsere Suchheuristik findet eine optimale Aufteilung der großen Objekte durch Zufall und arbeitet dann mit einer gewissen Wahrscheinlichkeit ähnlich dem Greedy-Ansatz weiter. Auch in anderen Zusammenhängen, beispielsweise bei EAs zur Berechnung maximaler Matchings (Giel und Wegener, 2003), darf man die Sichtweise vertreten, dass randomisierte Suchheuristiken manchmal mit guter Wahrscheinlichkeit algorithmische Ideen durch Zufall wieder entdecken.

Die in Theorem 4.4.1 genannte Erfolgswahrscheinlichkeit ist klein, aber beispielsweise für konstante ε immer noch in der Größenordnung $\Omega(1)$. Wir wollen im Folgenden beschreiben, wie wir parallele Läufe des (1+1)-EA und von RLS einsetzen können, um die Wahrscheinlichkeit so zu erhöhen, dass wir ein polynomielles, randomisiertes Approximationsschema (für die Definition vgl. Motwani und Raghavan, 1995) erhalten.

Definition 4.4.2 *Ein* polynomielles, randomisiertes Approximationsschema (PRAS) *für ein Optimierungsproblem ist ein randomisierter Algorithmus, der als Eingabe eine Probleminstanz I und ein $\varepsilon > 0$ erhält und in polynomieller Zeit bezüglich der Länge von I mit einer Wahrscheinlichkeit von mindestens* 3/4 *eine Lösung berechnet, deren Approximationsgüte* $1+\varepsilon$ *beträgt.*

Ein PRAS ist also ein Approximationsschema, bei dem die gewünschte Güte nur mit einer durch 1/4 beschränkten Fehlerwahrscheinlichkeit nicht in der genannten Zeit erreicht wird. Mit Standardargumenten kann man *probability amplification* betreiben und sich mit k Wiederholungen des PRAS die gewünschte Güte mit einer Wahrscheinlichkeit von mindestens $1-(1/4)^k$ sichern. Somit könnten wir die Konstante $3/4$ in Definition 4.4.2 durch jede positive Konstante ersetzen, was in der Definition von Motwani und Raghavan (1995) nicht möglich ist, da der Begriff eines PRAS dort noch weiter gefasst ist.

Es liegt nun auf der Hand, wie wir mit parallelen Läufen des (1+1)-EA (analog für RLS) ein PRAS im Sinne von Definition 4.4.2 erhalten. Seien eine Instanz für das Partitionsproblem und $\varepsilon > 0$ vorgegeben. Es bezeichne $\ell(n)$ eine untere Schranke für die Wahrscheinlichkeit, dass ein einzelner Lauf des (1+1)-EA in $t(n) = \lceil \mathrm{e}n \ln(2/\varepsilon)\rceil$ Schritten die gewünschte Approximation bezüglich der Instanz erzielt. Wir lassen nun $\lceil 2/\ell(n)\rceil$ Kopien des (1+1)-EA parallel laufen, brechen jede Kopie aber nach $t(n)$ Schritten ab. Die Wahrscheinlichkeit, dass mindestens eine Kopie die gewünschte Approximation erzielt, beträgt mindestens $1 - (1-\ell(n))^{\lceil 2/\ell(n)\rceil} \geq 1 - \mathrm{e}^{-2} > 3/4$ und der nach unserem Komplexitätsbegriff zu zählende Gesamtaufwand, d. h. die Zahl der Zielfunktionsauswertungen, beläuft sich auf höchstens $t(n)\lceil 2/\ell(n)\rceil$.

Wenn wir für $\ell(n)$ die unteren Schranken aus Theorem 4.4.1 einsetzen, passen die Schranken $t(n)$ dazu und der Gesamtaufwand der geschilderten parallelen Läufe ist höchstens

$$O(n \ln(1/\varepsilon)) \cdot 2^{(\mathrm{e}\log \mathrm{e}+\mathrm{e})(\lceil 2/\varepsilon\rceil \ln(2/\varepsilon))+O(1/\varepsilon)}.$$

Dieser Ausdruck ist für konstante Werte von ε durch $O(n)$ beschränkt und für $\varepsilon = \Omega(\log\log n/\log n)$ immerhin noch polynomiell beschränkt. Wir fassen zusammen.

Theorem 4.4.3 *Genügend viele parallele Läufe des (1+1)-EA und von RLS bilden ein PRAS für das Partitionsproblem.*

Theorem 4.4.3 ist das erste Resultat, das zeigt, dass ein evolutionärer Algorithmus ein polynomielles, randomisiertes Approximationsschema für ein NP-hartes Optimierungsproblem darstellt. Bislang war eine solche Charakterisierung lediglich für das in Polynomialzeit lösbare Problem der Berechnung maximaler Matchings bekannt (Giel und Wegener, 2003). Aus praktischer Hinsicht scheint eine Studie von Ruml, Ngo, Marks und Shieber (1996) zu unserem Ergebnis zu korrespondieren. Die Autoren untersuchen mehrere Kodierungen für Lösungen des Partitionsproblems und lassen einfache und spezialisierte evolutionäre Algorithmen auf verschiedenen Instanzen arbeiten. Dabei beobachten sie in vielen Fällen eine große Varianz der Approximationsgüte, die nach einer vorgegebenen Zeit erreicht wird, und schreiben parallelen Läufen daher eine besondere Bedeutung zu.

Wie im letzten Abschnitt haben wir auch bei den Analysen im Kontext dieses Abschnitts 4.4 festgestellt, dass wir lediglich von den lokalen Schritten des (1+1)-EA Gebrauch machen. Da wir alle Schritte mit mehr als einem kippenden Bit ignorieren, erhalten wir beim (1+1)-EA sogar eine schlechtere Schranke für die Erfolgswahrscheinlichkeit als bei RLS. Im nächsten Abschnitt wird allerdings ein Vorteil des globalen Suchoperators ans Tageslicht kommen.

4.5 Eine Average-Case-Analyse

Zu Beginn dieses Kapitels haben wir die mögliche Aussagekraft einer Average-Case-Analyse von Algorithmen für NP-harte Probleme motiviert. Ebenfalls bereits erwähnt haben wir Arbeiten, die sich Average-Case-Analysen von einfachen Approximationsalgorithmen für das Partitionsproblem, insbesondere des LPT-Algorithmus, widmen.

Die vielleicht einfachste Verteilung von Eingaben für das Partitionsproblem entsteht, wenn wir annehmen, dass jedes w_i, $1 \leq i \leq n$, gleichverteilt und unabhängig aus dem Intervall $[0, 1]$ gezogen wird. Da die Gleichverteilung als Modell für viele reale Prozesse, seien sie so einfach wie ein Münzwurf, herangezogen wird, darf diese Verteilung als natürlicher und nahe liegender Ausgangspunkt für eine Average-Case-Analyse gelten. Mit der Gleichverteilung auf dem Einheitsintervall betrachten wir in dieser Dissertation erstmals einen stetigen Wahrscheinlichkeitsraum, werden aber keine Aussagen über Wahrscheinlichkeitsdichten, sondern lediglich über Verteilungen benötigen. Die Wahl des Einheitsintervalls ist in diesem Zusammenhang gewissermaßen beliebig, da die von uns betrachteten Suchheuristiken für das Partitionsproblem invariant gegenüber einer Skalierung aller Volumina um denselben Faktor sind. Allerdings sind negative Werte für unsere Problemstellung unerwünscht, da sie zu einem anderen Typ von Problem führen können, insoweit als Bearbeitungszeiten von Lasten im Allgemeinen nicht negativ sind. Außerdem bemerken wir, dass in der Formulierung des Partitionsproblems als Entscheidungs- und als Optimierungsproblem (siehe Abschnitt 4.1) ursprünglich lediglich natürliche Zahlen w_i zugelassen sind. Dies ist in der üblichen komplexitätstheoretischen Arbeitsumgebung auch gar nicht anders möglich. Wir gehen dem Problem aus dem Weg, indem wir nun auch reellwertige, positive w_i zulassen, und stellen fest, dass reellwertige w_i aufgrund der oben erwähnten Skalierungseigenschaft in Instanzen mit natürlichen Zahlen beliebig genau approximiert werden können. Mit der Betrachtung stetiger Verteilungen erleichtern wir uns hoffentlich die Analyse, da „Ganzzahligkeitsbedingungen“, die bei allen diskreten Verteilungen auftreten müssen, wegfallen. Deshalb sind stetige Verteilungen oft Ausgangspunkt für Average-Case-Analysen, die man dann auf diskrete Verteilungen zu verallgemeinern sucht, vgl. auch Beier und Vöcking (2003).

Bereits in den 80er–Jahren des vergangenen Jahrhunderts untersuchten Coffman, Frederickson und Lueker (1984) einfache Greedy-Algorithmen für das Partitionsproblem auf der Grundlage des oben erwähnten gleichverteilten Eingabemodells. Dabei gelingt es ihnen unter anderem, obere und untere Schranken für das erwartete Volumen des volleren Eimers nach Ablauf des LPT-Algorithmus aufzustellen. Daraus erhalten sie als Konvergenzresultat die Aussage, dass die erwartete Differenz aus dem Volumen des volleren Eimers nach Ablauf des Algorithmus und dem Volumen in einer bestmöglichen Aufteilung mit n gegen 0 konvergiert. Dieses Art von Konvergenz heißt auch *Konvergenz im Erwartungswert.* Darauf aufbauend zeigen Frenk und Rinnooy Kan (1986), dass die erwähnte Differenz mit Wahrscheinlichkeit 1 in der Form $O(\log n/n)$ gegen 0 konvergiert, was als *fast sichere Konvergenz* bekannt ist.

Frenk und Rinnooy Kan (1986) unterziehen den LPT-Algorithmus auch für andere Verteilungen detaillierten Konvergenzanalysen. Als weitere populäre stetige Verteilung betrachten sie dabei die *Exponentialverteilung.* Die für uns wichtigen Eigenschaften dieser Verteilung werden wir in Abschnitt 4.5.2 nachliefern und motivieren die Beschäftigung mit dieser Verteilung hier, da sie als Modell für natürliche Zerfallsprozesse und Warteschlangenprozesse herhält und darüber hinaus als stetiges Analogon zur geometrischen Verteilung dient, siehe dazu beispielsweise Feller (1971).

Der Versuch einer Average-Case-Analyse randomisierter Suchheuristiken beschreitet in einem gewissen Sinne Neuland. Die Schwierigkeit liegt darin, dass wir hier mit zwei Quellen des Zufalls fertig werden müssen, nämlich den zufälligen Eingabeinstanzen und den zufälligen Entscheidungen der randomisierten Suchheuristik. Damit erscheint es vernünftig, wenn wir uns zunächst auf die beiden oben erwähnten Eingabeverteilungen beschränken.

4.5.1 Ergebnisse für gleichverteilte Koeffizienten

In diesem und im folgenden Unterabschnitt werden wir anstelle der Approximationsgüte, die der f-Wert des (1+1)-EA bzw. von RLS nach einer bestimmten Zeit erreicht, vornehmlich ein anderes Maß betrachten. Der Grund dafür liegt unter anderem darin, dass wir nur noch den Erwartungswert einer optimalen Lösung kennen und es unklar ist, auf welche Weise man die Approximationsgüte nun am besten definiert. Als Maß unserer Wahl analysieren wir im Folgenden die *Diskrepanz.* Diese gibt für eine Aufteilung der Objekte den Betrag der Differenz der Volumina der beiden Eimer an. Wir sind interessiert daran, wie nahe unsere Suchheuristiken an einen Diskrepanzwert von 0 herankommen.

Wir fassen noch einmal unser Modell zusammen. Seien die Zahlen w_i, $1 \leq i \leq n$, unabhängig und gleichverteilt aus $[0,1]$ gezogen. Der (1+1)-EA (RLS) für das Partitionsproblem mit einer solchen zufälligen Eingabe heißt im Folgenden *(1+1)-EA (RLS) im Gleichverteilungsmodell.*

Zunächst sehen wir uns die Suchheuristik im Gleichverteilungsmodell direkt nach der Initialisierung an. Man kann sich anhand von Symmetriegründen leicht überlegen, dass jeder der beiden Eimer dann ein erwartetes Volumen von $n/4$ besitzt und der Erwartungswert der Differenz des Volumens des ersten und des zweiten Eimers null ist. Das heißt natürlich nicht, dass der Erwartungswert der Diskrepanz dann ebenfalls gering ist.

Lemma 4.5.1 *Mit Wahrscheinlichkeit* $\Omega(1)$ *betragen die Diskrepanzen des (1+1)-EA und von RLS im Gleichverteilungsmodell nach der Initialisierung mindestens* $\Omega(\sqrt{n})$.

Beweis: Sei F die zufällige Anzahl von Objekten, die infolge der zufälligen Initialisierung im ersten Eimer landen. Da F binomialverteilt zu den Parametern n und $1/2$ ist, erhalten wir, dass für eine Konstante $c > 0$ die Beziehung $\text{Prob}(F \geq n/2 + c\sqrt{n}) = \Omega(1)$

gilt (für einen Beweis vgl. Jansen und Wegener, 2001a). Im Folgenden arbeiten wir unter der Annahme, dass dieses Ereignis eintritt, und beachten, dass dies die Verteilung der einzelnen Volumina nicht beeinflusst. Da das erwartete Volumen jedes Objekts 1/2 ist, haben die F Objekte dann ein erwartetes Gesamtvolumen von mindestens $n/4 + (c/2)\sqrt{n}$. Da die Verteilung einer Summe identisch verteilter, gleichverteilter Zufallsvariablen symmetrisch in Bezug auf ihren Erwartungswert ist, beträgt das Gesamtvolumen der F Objekte mit einer Wahrscheinlichkeit von mindestens 1/2 mindestens $n/4 + (c/2)\sqrt{n}$. Mit analogen Argumenten folgt, dass das Gesamtvolumen der $n - F$ Objekte im zweiten Eimer mit einer Wahrscheinlichkeit von mindestens 1/2 höchstens $n/4 - (c/2)\sqrt{n}$ beträgt. Insgesamt ist die Diskrepanz mit einer Wahrscheinlichkeit von $\Omega(1)$ also $\Omega(\sqrt{n})$. □

Man kann sich auch überlegen, dass die anfängliche Diskrepanz mit hoher Wahrscheinlichkeit nicht viel größer als $O(\sqrt{n})$ wird. Die bestmögliche Aufteilung einer Eingabe im Gleichverteilungsmodell besitzt andererseits eine erwartete Diskrepanz von höchstens $2^{-\Omega(n)}$ (Lueker, 1998). Wir interessieren uns nun dafür, wie weit der (1+1)-EA und RLS die Diskrepanz in polynomieller Zeit zu verringern vermögen.

Lemma 4.5.2 *Sei $c \geq 1$ eine beliebige Konstante. Mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ sind die Diskrepanzen des (1+1)-EA und von RLS im Gleichverteilungsmodell nach $O(n^2 \log n)$ Schritten durch 1 nach oben beschränkt. Außerdem ist die Diskrepanz nach einer erwarteten Zahl von $O(n^2)$ Schritten durch 1 beschränkt.*

Beweis: Mit w bezeichnen wir auch die Summe der w_i-Werte der zufälligen Instanz. Damit ist $\ell := w/2$ eine untere Schranke für den optimalen f-Wert. Da jedes Objekt ein Volumen von höchstens 1 besitzt, ist das kritische Volumen (Definition 4.2.1) in Bezug auf die Instanz und ℓ trivialerweise ebenfalls höchstens 1. Wir wenden Lemma 4.2.2 an. In der Sprache der Diskrepanz sagt es aus, dass diese nach einer erwarteten Zeit von höchstens $O(n^2)$ auf einen Wert von höchstens 1 fällt. Dies beweist bereits die zweite Aussage. Da Lemma 4.2.2 für jede Wahl der zufälligen Objekte und jeden Zeitpunkt gilt, dürfen wir die Markoffungleichung anwenden und mit einer Wiederholung unabhängiger Phasen arbeiten. Also sinkt die Diskrepanz nach $c \log n \cdot O(n^2) = O(n^2 \log n)$ Schritten mit einer Wahrscheinlichkeit von mindestens $1 - (1/2)^{c \log n} = 1 - 1/n^c$ auf einen Wert von höchstens 1. □

An dieser Stelle weisen wir noch darauf hin, dass man bei einem Versuch, im Gleichverteilungsmodell anhand von Lemma 4.2.3 zu argumentieren, ebenfalls im Auge behalten muss, dass die zufällige Instanz zu Beginn festgelegt wird und damit Lemma 4.2.3 nur für schlechtestmögliche Ausgänge der zufälligen Instanz greift.

Die Schranke aus Lemma 4.5.2 war leicht zu erhalten. Jedoch können wir zeigen, dass die Diskrepanz zumindest für den (1+1)-EA nach nur polynomiell vielen Schritten mit hoher Wahrscheinlichkeit viel kleiner als 1 wird. Dies wird das erste Mal in diesem

Kapitel sein, dass uns die Eigenschaft des (1+1)-EA, mehrere Bits auf einmal zu kippen, zugute kommt. Bevor wir das Ergebnis präsentieren, benötigen wir ein gewisses technisches Rüstzeug.

Definition 4.5.3 (Ordnungsstatistiken) *Seien $X_1, \ldots, X_n$ Zufallsvariablen. Wenn diese in absteigender Reihenfolge sortiert und als*

$$X_{(1)} \geq \ldots \geq X_{(n)}$$

geschrieben werden, dann heißt $X_{(i)}$ mit $1 \leq i \leq n$ die i-te Ordnungsstatistik *der Zufallsvariablen $X_1, \ldots, X_n$.*

Gegenüber der in der Literatur (siehe – auch für die weiteren Ergebnisse zu Ordnungsstatistiken – David und Nagaraja, 2003) vorherrschenden Definition haben wir in Definition 4.5.3 die Zufallsvariablen so angeordnet, dass $X_{(1)}$ die größte Zufallsvariable ist, um konsistent mit der üblichen Sortierung der w_i zu bleiben.

Ordnungsstatistiken von Zufallsvariablen sind grundsätzlich voneinander abhängige Zufallsvariablen. Im Allgemeinen folgt die Verteilung der i-ten Ordnungsstatistik einer Folge von Zufallsvariablen einer komplizierten Formel, die wir hier nicht angeben wollen. Selbst im Fall, dass $X_1, \ldots, X_n$ unabhängige und auf dem Einheitsintervall gleichverteilte Zufallsvariablen darstellen, existiert kein „schöner“ Ausdruck. Allerdings sind wir im Folgenden hauptsächlich an der Differenz $X_{(i)} - X_{(i+1)}$, also zweier aufeinander folgender Ordnungsstatistiken, interessiert. Diese Differenz und die kleinste Ordnungsstatistik folgen einer einfachen Verteilung.

Lemma 4.5.4 *Seien $X_1, \ldots, X_n$ unabhängige, auf $[0, 1]$ gleichverteilte Zufallsvariablen. Dann gelten für $1 \leq i \leq n-1$ und $0 < t < 1$ die Identitäten*

$$\mathrm{Prob}(X_{(i)} - X_{(i+1)} \geq t) \;=\; \mathrm{Prob}(X_{(n)} \geq t) \;=\; (1-t)^n.$$

Aus Lemma 4.5.4 folgt auch die interessante Identität $\mathrm{E}\big(X_{(i)} - X_{(i+1)}\big) = 1/(n+1)$. Weiterhin haben wir keine Probleme damit, von eindeutig bestimmten Ordnungsstatistiken der X_i zu reden, da mit Wahrscheinlichkeit 1 keine zwei der Zufallsvariablen denselben Wert annehmen.

Wie können uns die Ergebnisse zu den Ordnungsstatistiken helfen, um nachzuweisen, dass der (1+1)-EA eine kleinere Diskrepanz als 1 erreichen kann? Die Idee liegt darin, dass der (1+1)-EA mittels einer 2-Bit-Mutation die Objekte mit Volumina $X_{(i)}$ und $X_{(i+1)}$ vertauschen und den f-Wert verringern kann, wenn $X_{(i)}$ im volleren und $X_{(i+1)}$ im leereren Eimer liegt. Dies führt uns zu folgendem Theorem.

Theorem 4.5.5 *Sei $c \geq 1$ eine beliebige Konstante. Mit einer Wahrscheinlichkeit von mindestens $1 - O(1/n^c)$ beträgt die Diskrepanz des (1+1)-EA im Gleichverteilungsmodell nach $O(n^{c+4} \log n)$ Schritten höchstens $O(\log n/n)$. Außerdem ist die erwartete Diskrepanz nach $O(n^5 \log n)$ Schritten ebenfalls durch $O(\log n/n)$ beschränkt.*

Beweis: Wir wenden die Beweismethode des typischen Laufs an. Laut Lemma 4.5.2 sinkt die Diskrepanz nach $O(n^2 \log n)$ Schritten mit einer Wahrscheinlichkeit von $1 - O(1/n^2)$ auf einen Wert von höchstens 1. Da die Diskrepanz im Gleichverteilungsmodell höchstens n sein kann, trägt die Fehlerwahrscheinlichkeit lediglich einen Summanden von $n \cdot O(1/n^2) = o(\log n/n)$ zur erwarteten Diskrepanz nach $O(n^5 \log n)$ Schritten bei. Im Folgenden arbeiten wir unter der Annahme, dass die Diskrepanz für den aktuellen Suchpunkt des (1+1)-EA bereits höchstens 1 beträgt.

Seien $X_{(1)} \geq \cdots \geq X_{(n)}$ die Ordnungsstatistiken der n zufälligen w_i-Werte. Wir identifizieren die Objekte nun mit ihren Ordnungsstatistiken. Wenn es für den aktuellen Suchpunkt ein i gibt, sodass $X_{(i)}$ im volleren und $X_{(i+1)}$ im leereren Eimer liegt, verringert eine Mutation, die die beiden Objekte vertauscht, die Diskrepanz um $2(X_{(i)} - X_{(i+1)})$, wenn die Diskrepanz zuvor mindestens so groß ist. Wenn kein solches i existiert, müssen alle Objekte im leereren Eimer größer sein als jedes beliebige Objekt im volleren Eimer. Damit muss $X_{(n)}$ im volleren Eimer liegen und die Diskrepanz kann um $2X_{(n)}$ verringert werden, wenn sie zuvor mindestens $2X_{(n)}$ beträgt. Wir sind also an oberen Schranken für $X_{(i)} - X_{(i+1)}$, die für alle i gelten, sowie für $X_{(n)}$ interessiert.

Wir wenden Lemma 4.5.4 an. Sei $t^* := (c+1)(\ln n)/n$, d. h. $t^* = O(\log n/n)$. Es ist $(1-t^*)^n \leq n^{-c-1}$. Wir erhalten, dass mit einer Wahrscheinlichkeit von $1-O(1/n^c)$ für alle $1 \leq i \leq n-1$ die Beziehung $X_{(i)} - X_{(i+1)} \leq t^*$ gilt. Außerdem ist $\text{Prob}(X_{(n)} \leq t^*) = 1 - O(1/n^{c+1})$. Im Folgenden arbeiten wir unter der Annahme, dass $X_{(i)} - X_{(i+1)}$ und $X_{(n)}$ jeweils durch t^* beschränkt sind und führen damit nur eine Fehlerwahrscheinlichkeit von $O(1/n^c)$ ein. Im Fehlerfall beschränken wir die Diskrepanz nach $O(n^5 \log n)$ Schritten durch 1 und erhalten einen Beitrag von $O(1/n^c) = O(1/n)$ zur erwarteten Diskrepanz.

Solange die Diskrepanz noch mindestens $2t^*$ beträgt, gibt es nach unseren Annahmen immer eine 1-Bit- oder 2-Bit-Mutation, die die Diskrepanz verringert. Mit der auf Definition 4.2.1 folgenden Argumentation erhalten wir, dass dies sogar gilt, solange die Diskrepanz mehr als t^* beträgt. Die Mutation verringert die Diskrepanz wie oben beschrieben oder führt andernfalls zu einer Diskrepanz von weniger als t^*. Also benötigen wir auch untere Schranken für $X_{(i)} - X_{(i+1)}$ und $X_{(n)}$.

Sei $\ell^* := 1/n^{c+2}$. Mit Lemma 4.5.4 folgt

$$\text{Prob}(X_{(i)} - X_{(i+1)} \geq \ell^*) = \left(1 - \frac{1}{n^{c+2}}\right)^n \geq \mathrm{e}^{-2/n^{c+1}} \geq 1 - \frac{2}{n^{c+1}},$$

da $\mathrm{e}^{-x} \geq 1 - x$ (Lemma A.2), und gleicherweise $\text{Prob}(X_{(n)} \geq \ell^*) = 1 - 2/n^{c+1}$. Mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ ist dann $X_{(i)} - X_{(i+1)} \geq \ell^*$ für alle i sowie $X_{(n)} \geq \ell^*$. Wir nehmen an, dass diese Ereignisse eintreten, und behandeln den Fehlerfall wie oben.

Eine Mutation, die zwei spezielle Bits kippt, hat eine Wahrscheinlichkeit von mindestens $(1/n^2)(1-1/n)^{n-2} \geq 1/(\mathrm{e}n^2)$ und eine 1-Bit-Mutation ist noch wahrscheinlicher. Unter unseren Annahmen sinkt die Diskrepanz in jedem Schritt mit einer Wahrscheinlichkeit von $\Omega(1/n^2)$ um ℓ^* oder sie sinkt unter t^*. Also beträgt die erwartete Zeit, bis die Diskrepanz höchstens t^* ist, höchstens $O(n^2/\ell^*) = O(n^{c+4})$. Immer noch unter den Annahmen über die Eigenschaften der Ordnungsstatistiken können wir mit der Markoffungleichung und einer Wiederholung unabhängiger Phasen argumentieren und erhalten, dass die Diskrepanz mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ nach $O(n^{c+4} \log n)$ Schritten höchstens t^* beträgt. Da die Summe aller Fehlerwahrscheinlichkeiten $O(1/n^c)$ ist, folgt die erste Aussage des Theorems. Mit $c = 1$ und indem wir die erwartete Diskrepanz im Falle eines Fehlers jeweils wie beschrieben durch n oder 1 beschränken, folgt die zweite. □

Wir haben zum einen bewiesen, dass die Diskrepanz nach $O(n^5 \log n)$ Schritten des (1+1)-EA im Erwartungswert mit $n \to \infty$ gegen die geringstmögliche Diskrepanz konvergiert. Der Begriff der Konvergenz im Erwartungswert zählt allerdings zu den schwächsten Konvergenzbegriffen für Folgen von Zufallsvariablen (Chow und Teicher, 1997). Zum anderen zeigt Theorem 4.5.5 aber auch, dass die Diskrepanz nach einer polynomiell großen Anzahl von Schritten mit einer Wahrscheinlichkeit, die gegen 1 konvergiert, höchstens $O(\log n/n)$ ist, also gegen 0 konvergiert. Die Geschwindigkeit, mit der die Wahrscheinlichkeit gegen 1 konvergiert, ist mithilfe von c als jedes inverse Polynom einstellbar. Diese Form von Konvergenz ist aussagekräftiger als die Konvergenz im Erwartungswert. Wäre die erwähnte Wahrscheinlichkeit 1, könnten wir von fast sicherer Konvergenz reden, welche wie erwähnt für den deterministischen LPT-Algorithmus bei einer Konvergenzgeschwindigkeit von ebenfalls $O(\log n/n)$ bekannt ist. Es erscheint aber unmöglich, fast sichere Konvergenz für den (1+1)-EA nach polynomiell vielen Schritten nachzuweisen, da es immer eine positive Wahrscheinlichkeit gibt, dass während der ganzen Zeit kein Schritt etwas ändert.

4.5.2 Ergebnisse für exponentialverteilte Koeffizienten

Zum Abschluss unserer Average-Case-Analysen wollen wir uns mit Eingaben, deren Volumina der Exponentialverteilung folgen, auseinander setzen.

Definition 4.5.6 *Eine Zufallsvariable X folgt der Exponentialverteilung zum Parameter $\lambda \in \mathbb{R}^+$, wenn für alle $t \in \mathbb{R}$ die Beziehung*

$$\mathrm{Prob}(X \le t) = \begin{cases} 1 - \mathrm{e}^{-\lambda t}, & \text{falls } t \ge 0, \\ 0 & \text{sonst} \end{cases}$$

gilt.

Eine zum Parameter λ exponentialverteilte Zufallsvariable X hat einen Erwartungswert von $1/\lambda$. Sie weist die im Zusammenhang mit dem Beweis von Theorem 3.4.1 bereits erwähnte Gedächtnislosigkeit

$$\mathrm{Prob}(X > t + t' \mid X > t') = \mathrm{Prob}(X > t)$$

auf. Daher zieht man gerne exponentialverteilte Zufallsvariablen heran, um gedächtnislose Prozesse wie radioaktive Zerfallsprozesse, Ankunftsprozesse in Warteschlangensystemen u. Ä. zu modellieren. Weiterhin lässt sich mit der in Definition 4.5.6 vorgestellten Verteilungsfunktion gut rechnen.

Im Rest dieses Abschnitts werden wir das folgende Modell betrachten. Die Zahlen w_i, $1 \leq i \leq n$, seien unabhängige, zum Parameter 1 exponentialverteilte Zufallsvariablen. Der (1+1)-EA für das Partitionsproblem mit einer solchen zufälligen Eingabe heißt *(1+1)-EA im Exponentialverteilungsmodell.* Für RLS werden wir keine Ergebnisse mehr zeigen.

Der Beweis des kommenden Theorems wird einige Ideen aus dem Beweis von Theorem 4.5.5 aufgreifen. Insbesondere werden uns die Ordnungsstatistiken wieder begegnen.

Lemma 4.5.7 *Seien $X_1, \ldots, X_n$ unabhängige, zum Parameter 1 exponentialverteilte Zufallsvariablen und seien $X_{(1)} \geq \cdots \geq X_{(n)}$ ihre Ordnungsstatistiken. Dann existieren unabhängige, zum Parameter 1 exponentialverteilte Zufallsvariablen $Y_1, \ldots, Y_n$, sodass sich $X_{(i)}$ für $1 \leq i \leq n$ darstellen lässt als $\sum_{j=i}^{n} \frac{Y_j}{j}$.*

Die gerade erwähnte Identität, die zuweilen als Markoffkette für Ordnungsstatistiken bezeichnet wird, mag auf den ersten Blick verblüffen. Tatsächlich tauchen dieselben Zufallsvariablen Y_j in verschiedenen Ordnungsstatistiken $X_{(i)}$ auf. Andererseits kann man mit dem intuitiven Argument aushelfen, dass die Bedingungen für Ordnungsstatistiken verletzt sein könnten, wenn Y_j auch noch von i abhinge.

Wir zitieren noch ein letztes und allgemein bekanntes Ergebnis (Feller, 1971) für exponentialverteilte Zufallsvariablen. Die folgende Beziehung heißt auch, dass X der Gammaverteilung zu den Parametern 1 und n folgt.

Lemma 4.5.8 *Sei X die Summe von n unabhängigen, zum Parameter 1 exponentialverteilten Zufallsvariablen. Dann gilt für $t \geq 0$*

$$\text{Prob}(X \geq t) \;=\; \mathrm{e}^{-t}\left(1 + \frac{t}{1!} + \cdots + \frac{t^{n-1}}{(n-1)!}\right).$$

Jetzt haben wir genügend Hilfsmittel zusammen, um den (1+1)-EA im Exponentialverteilungsmodell zu analysieren.

Theorem 4.5.9 *Sei $c \geq 1$ eine beliebige Konstante. Mit einer Wahrscheinlichkeit von mindestens $1 - O(1/n^c)$ beträgt die Diskrepanz des (1+1)-EA im Exponentialverteilungsmodell nach $O(n^2 \log n)$ Schritten $O(\log n)$ und nach $O(n^{c+4} \log^2 n)$ Schritten $O(\log n/n)$. Außerdem beträgt die erwartete Diskrepanz nach $O(n^2 \log n)$ Schritten $O(\log n)$ und nach $O(n^6 \log^2 n)$ Schritten $O(\log n/n)$.*

Beweis: Zunächst beobachten wir, dass der Erwartungswert der Diskrepanz nach der Initialisierung durch n nach oben beschränkt ist, da jedes Objekt im Exponentialverteilungsmodell ein erwartetes Volumen von 1 hat. Wir werden zeigen, dass mit einer

Wahrscheinlichkeit von $1 - O(1/n^c)$ die folgenden beiden Eigenschaften gelten. Zum einen beträgt die Diskrepanz nach der Initialisierung höchstens $2n$. Zum anderen ist das kritische Volumen in Bezug auf die zufällige Instanz und den Wert $\ell := w/2$ (w ist wieder die Summe der zufälligen w_i) stets durch $O(\log n)$ beschränkt. Wenn mindestens eine der beiden Eigenschaften nicht erfüllt ist, sprechen wir von einem Fehler. Im Falle eines Fehlers beschränken wir die erwartete Diskrepanz nach $O(n^2 \log n)$ Schritten durch die anfänglich geltende Schranke n. Wenn kein Fehler eintritt, können wir wie im Beweis von Lemma 4.5.2 argumentieren und Lemma 4.2.2 anwenden. Mit der Markoffungleichung und einer Wiederholung unabhängiger Phasen sowie der gerade genannten Abschätzung im Fehlerfall erhalten wir beide Aussagen des Theorems bezüglich der Situation nach $O(n^2 \log n)$ Schritten.

Im Folgenden zeigen wir die beiden Eigenschaften und wenden sie anschließend an, um die Situationen nach $O(n^{c+4} \log^2 n)$ bzw. $O(n^6 \log^2 n)$ Schritten zu charakterisieren. Zum Nachweis der ersten Eigenschaft benutzen wir Lemma 4.5.8. Mit $t := 2n$ folgt

$$\mathrm{Prob}(w \geq 2n) \;=\; \mathrm{e}^{-2n} \left(1 + \frac{2n}{1!} + \cdots + \frac{(2n)^{n-1}}{(n-1)!}\right) \;\leq\; \frac{n\mathrm{e}^{-2n}(2n)^{n-1}}{(n-1)!},$$

da der Summand $(2n)^{n-1}/(n-1)!$ am größten ist. Gemäß der stirlingschen Formel ist der letzte Ausdruck nach oben beschränkt durch

$$\frac{n\mathrm{e}^{-2n+(n-1)}2^{n-1}n^{n-1}}{(n-1)^{n-1}} \;=\; \mathrm{e}^{-2n+(n-1)} \cdot 2^{n-1} \cdot n \cdot \left(1 - \frac{1}{n}\right)^{-(n-1)} \;=\; 2^{-\Omega(n)},$$

was beweist, dass die erste Eigenschaft sogar mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n)}$ gilt.

Zum Beweis der zweiten Eigenschaft betrachten wir wieder die Ordnungsstatistiken $X_{(1)} \geq \cdots \geq X_{(n)}$ der zufälligen w_i-Werte. Unser Ziel ist es zu zeigen, dass mit hoher Wahrscheinlichkeit die Abschätzung $X_{(1)} + \cdots + X_{(k)} \leq w/2$ für ein $k := \lceil \delta n \rceil$ mit einer Konstanten $\delta > 0$ erfüllt ist. Anschließend werden wir zeigen, dass $X_{(k)} = O(\log n)$ mit hoher Wahrscheinlichkeit gilt, und beschränken somit das kritische Volumen.

Um ein genügend großes k zu finden, ist es hilfreich, w auch nach unten zu beschränken. Da wir die Exponentialverteilung zum Parameter 1 betrachten, hat jedes Objekt mit einer Wahrscheinlichkeit von $\mathrm{e}^{-1} > 1/3$ ein Volumen von mindestens 1. Mit der Chernoffungleichung folgt $w \geq n/3$ mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$. Nun wenden wir Lemma 4.5.7 an und erhalten

$$\begin{aligned} X_{(1)} + \cdots + X_{(k)} \;&=\; \sum_{i=1}^{k} \sum_{j=i}^{n} \frac{Y_j}{j} \;=\; Y_1 + 2 \cdot \frac{Y_2}{2} + \cdots + k \cdot \frac{Y_k}{k} + k \sum_{i=k+1}^{n} \frac{Y_i}{i} \\ &\leq\; \sum_{j=1}^{k} Y_j + \sum_{i=1}^{\lceil n/k \rceil} \frac{1}{i} \sum_{j=ik+1}^{(i+1)k} Y_j \end{aligned}$$

mit $Y_j := 0$ für $j > n$. Im Wesentlichen sind wir mit $\lceil n/k \rceil + 1$ Summen, die je k exponentialverteilte Zufallsvariablen mit Parameter 1 enthalten, konfrontiert. Wenn wir $k := \lceil \delta n \rceil$ für irgendeine Konstante $\delta \in]0, 1[$ wählen, liefert die oben für $\text{Prob}(w \geq 2n)$ vorgeführte Abschätzungstechnik, dass eine einzelne Summe mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ durch $2k$ nach oben beschränkt ist. Da es höchstens n Summen gibt, ist mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ sogar jede Summe auf diese Weise beschränkt. Also ist mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$

$$X_{(1)} + \cdots + X_{(k)} \;\leq\; 2\lceil \delta n \rceil + \sum_{i=1}^{1/\delta} \frac{2\lceil \delta n \rceil}{i} \;\leq\; 2(\delta n + 1)\ln(1/\delta + 2),$$

indem wir wie üblich die harmonische Zahl abschätzen. Durch Nachrechnen folgt, dass der letzte Ausdruck kleiner wird als $n/6$, wenn wir $\delta \leq 1/50$ und n groß genug wählen. Also ist $X_{(1)} + \cdots + X_{(\lceil n/50 \rceil)} \leq w/2$ mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$.

Wie groß ist nun $X_{(\lceil n/50 \rceil)}$? Man sieht leicht, dass mit einer Wahrscheinlichkeit von mindestens $1 - ne^{-(c+1)\ln n} = 1 - n^{-c}$ alle Y_j höchstens $(c+1)\ln n$ sind. Folglich gilt mit mindestens derselben Wahrscheinlichkeit die Abschätzung

$$\begin{aligned} X_{(\lceil n/50 \rceil)} \;&=\; \sum_{j=\lceil n/50 \rceil}^{n} \frac{Y_j}{j} \;\leq\; (c+1)(\ln n)(H_n - H_{\lceil n/50 \rceil - 1}) \\ &\leq\; (c+1)(\ln n)\left(\ln n + 1 + \frac{1}{\lceil n/50 \rceil} - \ln(n/50)\right), \end{aligned}$$

deren letzter Ausdruck $(c+1)(\ln(50) + 1 + o(1))(\ln n) = O(\log n)$ ist. Da die Summe aller Fehlerwahrscheinlichkeiten $O(1/n^c)$ beträgt, haben wir das kritische Volumen wie gewünscht beschränkt und die zweite Eigenschaft gezeigt.

Wir können nun die Aussage des Theorems über die Situation nach $O(n^{c+4}\log^2 n)$ Schritten zeigen. Da wir wiederum eine Fehlerwahrscheinlichkeit von $O(1/n^c)$ erhalten werden, können wir die erwartete Diskrepanz im Falle eines Fehlers durch die anfangs erwartete Diskrepanz von höchstens n beschränken und erhalten aus $c = 2$ einen Beitrag von $O(n) \cdot O(1/n^2) = O(1/n)$ zur erwarteten Diskrepanz nach $O(n^6 \log^2 n)$ Schritten. Jetzt nehmen wir unter Einführung eines Fehlers von höchstens $O(1/n^c)$ an, dass die oben erwähnten Eigenschaften zutreffen und die Diskrepanz innerhalb von $O(n^2 \log n)$ Schritten auf $O(\log n)$ reduziert wurde. Nun wollen wir wie im Beweis von Theorem 4.5.5 2-Bit-Mutationen, die $X_{(i)}$ und $X_{(i+1)}$ gegeneinander austauschen, oder 1-Bit-Mutationen, die $X_{(n)}$ bewegen, betrachten, nutzen aber aus, dass das kritische Volumen außerdem die Menge der möglichen 2-Bit-Mutationen beeinflusst. Wir haben oben gezeigt, dass mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ das kleinste Objekt im volleren Eimer immer die k-te Ordnungsstatistik mit $k \geq n/50$ ist. Also gibt es mit dieser Wahrscheinlichkeit immer ein Paar $(X_{(i)}, X_{(i+1)})$ mit $i \geq n/50$, sodass sich $X_{(i)}$ im volleren und $X_{(i+1)}$ im leereren Eimer befindet, oder aber $X_{(n)}$ befindet sich im volleren Eimer. Der Austausch der beiden Objekte verringert die Diskrepanz um $2(X_{(i)} - X_{(i+1)})$, wenn sie zuvor mindestens so groß war.

Da $X_{(i)} - X_{(i+1)} = Y_i/i$ ist, erhalten wir für die interessanten Werte von i die Abschätzung $X_{(i)} - X_{(i+1)} \leq 50Y_i/n$. Wir haben bereits ausgerechnet, dass alle Y_i mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ durch $(c+1)\ln n$ beschränkt sind. Insgesamt haben wir mit mindestens dieser Wahrscheinlichkeit $X_{(i)} - X_{(i+1)} \leq 50(c+1)\ln n =: t^*$ für alle interessanten i. Für $X_{(n)}$ nutzen wir die Identität $X_{(n)} = Y_n/n$ aus. Also ist auch $\text{Prob}(X_{(n)} \leq t^*)$ mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$. Im Folgenden nehmen wir an, dass die genannten oberen Schranken zutreffen. Also gibt es, solange die Diskrepanz größer als t^* ist, eine Mutation, die die Diskrepanz um $2(X_{(i)} - X_{(i+1)})$ bzw. $2X_{(n)}$ verringert oder unter t^* sinken lässt.

Wir müssen noch untere Schranken für $X_{(i)} - X_{(i+1)}$ bzw. $X_{(n)}$ bestimmen. Wir erhalten für jedes feste i die Abschätzung

$$\text{Prob}(X_{(i)} - X_{(i+1)} \geq 1/n^{c+2}) \;\geq\; \text{Prob}(Y_i/n \geq 1/n^{c+2}) \;=\; \mathrm{e}^{-1/n^{c+1}} \;\geq\; 1 - \frac{1}{n^{c+1}}$$

und analog $\text{Prob}(X_{(n)} \geq 1/n^{c+2}) \geq 1 - 1/n^{c+1}$. Insgesamt gelten alle diese unteren Schranken mit einer Wahrscheinlichkeit von $1-O(1/n^c)$ auf einmal und wir nehmen an, dass auch dies zutrifft. Mit denselben Argumenten wie im Beweis von Theorem 4.5.5 folgt, dass die Zeit bis zu einer Diskrepanz von höchstens $t^* = O(\log n/n)$, ausgehend von einer Diskrepanz von $O(\log n)$, mit einer Wahrscheinlichkeit von $1 - O(1/n^c)$ durch $O(n^2 \log n) \cdot O((\log n)n^{c+2}) = O(n^{c+4} \log^2 n)$ nach oben beschränkt ist. □

In Bezug auf die Konvergenzaussagen gelten für Theorem 4.5.9 dieselben Bemerkungen wie nach dem Beweis von Theorem 4.5.5. Auch im Falle des Exponentialverteilungsmodells konvergiert der absolute Fehler nach Ablauf des deterministischen LPT-Algorithmus mit Wahrscheinlichkeit 1 gemäß $O(\log n/n)$ gegen 0. Trotz der Ähnlichkeit der beiden Theoreme 4.5.5 und 4.5.9 war aber von vornherein nicht klar, ob die mit Wahrscheinlichkeit $1 - O(1/n^c)$ geltende Konvergenz des (1+1)-EA auch im Exponentialverteilungsmodell gezeigt werden kann. Aus Theorem 4.5.9 dürfen wir die Erkenntnis ziehen, dass es neben der beschränkten Gleichverteilung mit der Exponentialverteilung auch eine unbeschränkte Verteilung gibt, auf der der (1+1)-EA ein effizientes Average-Case-Verhalten an den Tag legt.

4.6 Fazit

In diesem Kapitel haben wir die randomisierten Suchheuristiken (1+1)-EA und RLS auf einem konkreten, wohl bekannten kombinatorischen Optimierungsproblem untersucht, nämlich der Optimierungsvariante des NP-vollständigen Entscheidungsproblems PARTITION. Unser Augenmerk lag dabei auf der Approximationsgüte der Lösungen, die die Suchheuristik typischerweise in einer polynomiell langen Zeitspanne berechnet.

Während man hier im Allgemeinen auf nicht viel bessere Approximationsgüten als 4/3 hoffen darf, zeigt ein weiter gehendes Resultat, dass der (1+1)-EA und RLS für das Partitionsproblem bei parallelen Läufen ein polynomielles, randomisiertes Approximationsschema darstellen können. Dies ist das erste solche Resultat für ein NP-hartes Optimierungsproblem.

Im Anschluss an diese Worst-Case-Untersuchungen haben wir Ansätze für eine Average-Case-Analyse der Suchheuristiken auf zufälligen Eingaben vorgestellt. Sowohl für gemäß der Gleichverteilung als auch gemäß der Exponentialverteilung gezogene Eingaben hat es sich herausgestellt, dass der (1+1)-EA die Diskrepanz einer Aufteilung nach polynomiell vielen Schritten mit einer Wahrscheinlichkeit von $1 - o(1)$ auf höchstens $O(\log n/n)$ gesenkt hat. Unter Inkaufnahme einer höheren polynomiellen Zeit ließ sich die im Ausdruck „$o(1)$“ enthaltene Fehlerwahrscheinlichkeit durch das Inverse eines Polynoms mit beliebigem Grad beschränken. Die Konvergenzaussage zeigt Bezüge zu Konvergenzaussagen eines problemspezifischen Approximationsalgorithmus in Average-Case-Modellen auf. Auch dort sind keine besseren Schranken als $O(\log n/n)$ für die nach Ablauf des Algorithmus geltenden Diskrepanzen bekannt. Mit den Average-Case-Analysen des (1+1)-EA haben wir die vermutlich erste Studie einer randomisierten Suchheuristik auf zufälligen Eingabeinstanzen vorgestellt und die Mächtigkeit der mittlerweile vorhandenen Methoden zur Laufzeitanalyse evolutionärer Algorithmen und anderer randomisierter Suchheuristiken unter Beweis gestellt.

Die Studien in diesem Kapitel zeigen im Gegensatz zu denen im vorangehenden Kapitel für das nun konkrete und praktische Optimierungsproblem einen Vorteil des global suchenden Mutationsoperators des (1+1)-EA auf, beschreiben jedoch immer noch viele Situationen, in denen eine lokale Suche erstaunlich weit führt. Mit den Ergebnissen haben wir außerdem unser allgemeines Verständnis von der Arbeitsweise randomisierter Suchheuristiken auf kombinatorischen Optimierungsproblemen erweitert. Im Zusammenhang mit der Entwicklung eines randomisierten Approximationsschemas haben wir nämlich ein Beispiel entdeckt, in dem eine randomisierte Suchheuristik durch Zufall auf eine algorithmische Idee eines problemspezifischen Algorithmus stößt und diese Idee nachahmt.

Teil II

Zur Analyse populationsbasierter evolutionärer Algorithmen

5 Grundlagen und Motivation von Populationen

Der erste Teil dieser Arbeit beschäftigte sich mit dem Verhalten einfacher randomisierter Suchheuristiken wie RLS, des (1+1)-EA und weiteren Variationen dieser Heuristiken. All diesen Heuristiken ist gemein, dass sie pro Schritt lediglich einen aktuellen Suchpunkt vorhalten. Diese Eigenschaft teilen sie mit weiteren Suchheuristiken wie dem Metropolis-Algorithmus (Metropolis, Rosenbluth, Rosenbluth, Teller und Teller, 1953) und Simulated Annealing (Kirkpatrick, Gelatt und Vecchi, 1983), die nicht primär als Instanzen evolutionärer Algorithmen verstanden werden.

Ebenfalls im ersten Teil der Arbeit erwähnt wurde der Begriff der Population eines evolutionären Algorithmus. Aus verschiedenen Gründen sahen die Erfinder der historisch unabhängig entwickelten Paradigmen evolutionärer Algorithmen die Möglichkeit vor, in ihrem EA mehrere, eventuell verschiedene Suchpunkte in einer so genannten Population vorzuhalten. Ohne einen Anspruch auf Vollständigkeit zu erheben, zeigen wir hier einige der vielleicht gängigsten Motivationen für den Einsatz von Populationen auf. Der sicherlich naheliegendste Grund ergibt sich aus dem Bestreben, die natürliche Evolution nachzubilden, indem eine Menge von Individuen einer Spezies modelliert und verwaltet wird. Gleichzeitig verspricht man sich eine gewisse Vielfalt *(Diversität)* der verwalteten Suchpunkte. In den von Holland (1975) begründeten genetischen Algorithmen ist eine Population mit mehr als einem Element sogar unerlässlich, um den obligatorischen Rekombinationsoperator (vgl. Abschnitt 2.1) anwenden zu können. In den einfachsten Ausprägungen von Evolutionsstrategien (Schwefel, 1995) hingegen ist kein Rekombinationsoperator vorgesehen, sodass auch sehr einfache Evolutionsstrategien mit einer Population der Größe 1, aus denen sich der (1+1)-EA ableitet, sinnvoll erscheinen. Trotzdem arbeiten komplexere Evolutionsstrategien aus vielfältigen Gründen mit Populationen, was in Kapitel 7 des Weiteren erläutert werden soll. Auf die Motivation von Populationen in anderen Ausprägungen evolutionärer Algorithmen, z. B. der genetischen Programmierung, wollen wir hier nicht eingehen.

5.1 Fragestellungen

Aus der Sicht der Informatik, speziell bei unserem Ansatz einer Theorie der Zeitkomplexität evolutionärer Algorithmen, interessiert der Zusammenhang zwischen der (erwarteten) Laufzeit eines gegebenen evolutionären Algorithmus und der Größe seiner Population. In diese Richtung gehen einerseits die Ansätze von Jansen und Wegener

(2001c, 2002) und Storch und Wegener (2003). In den genannten Arbeiten widmen sich die Autoren der Frage, ob ein populationsbasierter EA mit einem Rekombinationsoperator auf Beispielfunktionen deutlich effizienter als der (1+1)-EA sein kann. Letztendlich stellt sich heraus, dass der Einsatz des populationsbasierten EAs insgesamt selbst bei einer Populationsgröße von 2 zu einem exponentiell großen Laufzeitvorteil gegenüber dem (1+1)-EA führen kann, womit eine seit längerer Zeit diskutierte Frage (siehe z. B. Mitchell, Holland und Forrest, 1994) zufrieden stellend geklärt werden konnte. Die in den genannten Arbeiten durchgeführten Laufzeitanalysen sind auch insoweit erwähnenswert, als der Einsatz von Rekombination den Ruf hat, zu schwer durchschaubaren Systemen zu führen. Bei Verallgemeinerungen auf unendlich große Populationen gelangt man zu so genannten quadratischen dynamischen Systemen (vgl. Rabani, Rabinovich und Sinclair, 1998). Quadratische dynamische Systeme sind eine Verallgemeinerung von Markoffketten, welche aus komplexitätstheoretischer Sicht schwierige Fragen aufwirft (Arora, Rabani und Vazirani, 1994).

Jedoch kann man den Einwand erheben, dass die oben erwähnten Arbeiten nicht etwa den Einsatz einer Population, sondern vielmehr die Einführung eines Rekombinationsoperators rechtfertigen. Bis vor kurzem war es somit ungeklärt, ob die Population in evolutionären Algorithmen, die lediglich einen Mutationsoperator verwenden, an sich überhaupt sinnvoll sein kann. Für sinnvoll halten wir eine Population nämlich nur dann, wenn sie die Laufzeit des EAs beeinflussen und auch senken kann. In den folgenden beiden Kapiteln werden wir uns diesem Aspekt widmen. Kapitel 6 beschäftigt sich mit der Frage nach dem generellen Nutzen von Populationen in evolutionären Algorithmen. Aufbauend auf einer Arbeit von Jansen und Wegener (2001b) untersuchen wir dort, ob in einem populationsbasierten EA eine kleine Änderung der Populationsgröße zu einer großen Änderung der Laufzeit führen kann. Außerdem werden wir nachweisen, dass eine große Population einer effizienten Laufzeit abträglich sein kann, und zeigen, dass die Wahl der geeigneten Populationsgröße entscheidend sein kann. Insgesamt handelt es sich um strukturelle Fragen, die wir anhand eines geeignet gewählten populationsbasierten EAs zu beantworten suchen.

Kapitel 7 ist anders ausgerichtet, aber baut dennoch auf Kapitel 6 auf. Nachdem die genannten strukturellen Fragen zufrieden stellend beantwortet sind, untersuchen wir in Kapitel 7 den Einfluss der Populationsgröße auf die Laufzeit eines vorgegebenen, traditionellen evolutionären Algorithmus ohne Rekombination, des so genannten (μ+1)-EA. Wünschenswert ist nun der tatsächliche funktionale Zusammenhang zwischen Problemdimension, Populationsgröße und Laufzeit des EAs auf Funktionen und Funktionenklassen. Diese Funktionen(klassen) sollen sich an herkömmlichen, bei der Laufzeitanalyse des (1+1)-EA untersuchten Funktionen orientieren. Ziel des Ganzen soll ein Beitrag dazu sein, ein Methodenreservoir zur Laufzeitanalyse populationsbasierter EAs aufzubauen. Auch bei der Laufzeitanalyse des (1+1)-EA begann man mit einfachen, oft künstlich anmutenden Beispielfunktionen, um eine Methodik zu entwickeln. Inzwischen ist man, wie bereits in Kapitel 4 erwähnt und vorgeführt, in der Lage, den (1+1)-EA auch auf bekannten kombinatorischen Optimierungsproblemen zu analysieren. Somit schlagen wir auch die Brücke zum ersten Teil dieser Dissertation. Vielleicht ist es eines Tages möglich, mit den neuen Methoden einen populationsbasierten EA auf einem praktischen kombinatorischen Optimierungsproblem ähnlich gut wie den (1+1)-EA zu analysieren.

Über den Verzicht auf Rekombination hinaus können wir weitere Gemeinsamkeiten der in den Kapiteln 6 und 7 betrachteten EAs festhalten. Es hat sich bereits in den oben erwähnten Arbeiten zum Nutzen des Rekombinationsoperators als hilfreich erwiesen, pro Generation des EAs lediglich einen neuen Suchpunkt (ein Kind) zu erzeugen, sodass sich eine Population von ihrer nachfolgenden lediglich in einem Element unterscheiden kann. Dieses Konzept heißt *steady state selection* und wird häufig gerade mit der Ähnlichkeit zwischen zwei aufeinander folgenden Populationen und der Bewahrung „guter Suchpunkte" motiviert. Außerdem spart das Konzept Speicherplatz. Sowohl der in Kapitel 6 als auch der in Kapitel 7 betrachtete EA folgen diesem Schema, für das wir ab sofort den klareren Begriff des „einen Kindes" einsetzen. Daneben lässt sich eine weitere Übereinstimmung ausmachen, da beide EAs den aus Definition 2.3.2 bekannten $1/n$-Standardmutationsoperator benutzen. Die Verfahren, mit denen die EAs zur Reproduktion und zur Ersetzung auswählen, unterscheiden sich deutlich voneinander. Nichtsdestoweniger kristallisieren wir die Gemeinsamkeiten in Bezug auf diese Operatoren heraus, da es möglich ist, für einen allgemeinen Rahmenalgorithmus, dem die beiden EAs folgen, Begriffsbildungen einzuführen und sogar einfache Analysemethoden vorzustellen. Somit vermeiden wir es, in Kapitel 7 auf umständliche Weise Begriffe aus Kapitel 6 umzudefinieren und allgemeine Beweismethoden ein zweites Mal vorzustellen. Die in diesem Teil der Dissertation betrachteten populationsbasierten EAs werden in folgende Klasse von Suchheuristiken fallen.

Definition 5.1.1 (Ein-Kind-EA) *Es seien alle in dieser Definition auftretenden Mengen Multimengen. Eine randomisierte Suchheuristik heißt* Ein-Kind-EA*, wenn sie mit einer Populationsgröße $\mu \in \mathbb{N}$ und einer Zielfunktion $f\colon \{0,1\}^n \to \mathbb{R}$ parametrisiert ist und als folgender Endlosprozess abläuft.*

Setze $t := 0$. Initialisiere $x_i \in \{0,1\}^n$ für $1 \le i \le \mu$ unabhängig und zufällig gleichverteilt. $X^{(0)} := \{x_1, \dots, x_\mu\}$.

Wiederhole:

Die Selektion zur Reproduktion gibt jedem $x \in \operatorname{argmax}\{f(x) \mid x \in X^{(t)}\}$ eine Wahrscheinlichkeit von mindestens $1/\mu$ und wählt damit $y \in X^{(t)}$.

Der $1/n$-Standardmutationsoperator mutiert y und erzeugt y'. $X' := X^{(t)} \cup \{y'\}$.

Die Selektion zur Ersetzung wählt $x_{\max} \in \operatorname{argmax}\{f(x) \mid x \in X'\}$ gleichverteilt, gibt jedem $x \in \bigl(\operatorname{argmin}\{f(x) \mid x \in X'\} \setminus \{x_{\max}\}\bigr)$ eine Wahrscheinlichkeit von mindestens $1/\mu$ und wählt damit $y \in X' \setminus \{x_{\max}\}$. $X^{(t+1)} := X' \setminus \{y\}$.

$t := t + 1$.

In Definition 5.1.1 müssen wir mit Multimengen arbeiten, da die $X^{(t)}$ oder X' mehrere Elemente enthalten können, die demselben Suchpunkt $x \in \{0,1\}^n$ entsprechen. Im Folgenden bezeichnen wir die Multimengen $X^{(t)}$ sowie X' als *Populationen* und $X^{(0)}$ heißt *initiale Population*. Unter Populationen verstehen wir im Übrigen immer Multimengen. Die Elemente einer Population heißen *Individuen* und die Individuen aus $X^{(0)}$ heißen *initiale Individuen*. Gelegentlich bezeichnen wir auch einen gerade untersuchten Suchpunkt aus $\{0,1\}^n$ als Individuum, selbst wenn wir ihn gar keiner Population zuschreiben.

Wir sehen uns den Lauf eines Ein-Kind-EAs nun näher an. Die Selektion zur Reproduktion ist so beschaffen, dass Individuen, deren f-Wert maximal für die aktuelle Population ist, mit mindestens so hoher Wahrscheinlichkeit gewählt werden wie in dem Fall, dass alle Individuen denselben f-Wert haben. Die anschließende Mutation benutzt den Mutationsoperator aus Definition 2.3.2 und sucht damit global, d. h., aus jedem $y \in \{0,1\}^n$ wird jedes $y' \in \{0,1\}^n$ mit positiver Wahrscheinlichkeit erzeugt. Die nach der Mutation vorliegende Multimenge X' nennen wir häufig *die vergrößerte Population*, da sie $\mu + 1$ Elemente enthält. Die darauf folgende Selektion zur Ersetzung löscht ein Individuum aus der vergrößerten Population und stellt dabei zwei Anforderungen. Zum einen werden Individuen, deren f-Wert minimal für die vergrößerte Population ist, mit mindestens so hoher Wahrscheinlichkeit als zu löschendes Individuum gewählt wie in dem Fall, dass alle Individuen denselben f-Wert haben. Zum anderen aber wird sichergestellt, dass mindestens ein Individuum mit maximalem f-Wert in der vergrößerten Population keinesfalls gelöscht wird. Diese Eigenschaft heißt *Elitismus* und der Ein-Kind-EA heißt damit *elitär*. Mit dem Elitismus und der global suchenden $1/n$-Standardmutation ist garantiert, dass die Folge von Populationen $X^{(t)}$ für $t \to \infty$ gegen eine Menge von Populationen, die je mindestens ein optimales Individuum enthalten, strebt. Für evolutionäre Algorithmen, die nicht elitär sind, braucht dies nicht erfüllt zu sein (für eine Diskussion der Konvergenzeigenschaften populationsbasierter evolutionärer Algorithmen siehe wieder Rudolph, 1997).

Ebenso wie bei den in Teil I dieser Arbeit untersuchten Suchheuristiken verzichten wir in Ein-Kind-EAs auf ein Stoppkriterium. Bei der Variablen t aus der Definition des Ein-Kind-EAs sprechen wir wieder von einer *Zeit* und nennen $X^{(t)}$ häufig die *Population zum Zeitpunkt t*. Sei bei einem gegebenen Ein-Kind-EA A mit Populationsgröße μ und Zielfunktion f mit $\tau_{A,\mu,f}$ der zufällige kleinste Wert von t bezeichnet, sodass die Population $X^{(t)}$ mindestens ein Individuum mit dem global maximalen f-Wert enthält. Wie in Teil I der Arbeit sagen wir salopp, dass A *zum Zeitpunkt* $\tau_{A,\mu,f}$ (oder: *in* $\tau_{A,\mu,f}$ *Schritten*) *das Optimum erreicht*. Mit dem Wert $T_{A,\mu,f} := \mu + \tau_{A,\mu,f}$ bezeichnen wir die *Laufzeit* oder auch *Optimierungszeit* von A auf f bei Populationsgröße μ und beachten, dass diese der Zahl der Zielfunktionsauswertungen bis zum Ende der $\tau_{A,\mu,f}$-ten Iteration der „Wiederhole"-Schleife entspricht, da bereits die Initialisierung der Population μ Auswertungen von f nach sich zieht. Letzteres gilt, da alle Suchpunkte der initialen Population mindestens einmal angefragt werden müssen, um die anschließenden Selektionsoperatoren anwenden zu können. Das Maß für die Laufzeit eines Ein-Kind-EAs begründet sich mithin wiederum aus der Theorie der Black-Box-Komplexität, auch wenn man sich vor Augen führen sollte, dass die Berechnung der Auswahlwahrscheinlichkeiten u. Ä. bei den Selektionsoperatoren sicherlich einen größeren Aufwand als in einem Schritt des (1+1)-EA verkörpert.

Weiterhin benutzen wir den Begriff *Schritt* für die Menge der in einer Iteration der „Wiederhole"-Schleife durchgeführten Operationen. Als s-ten Schritt bezeichnen wir dabei wie bei den randomisierten Hillclimbern aus Abschnitt 2.1 diejenige Iteration, zu deren Beginn $t = s-1$ für die Variable t aus Definition 5.1.1 gilt, d. h., es gibt keinen 0-ten Schritt. Da in einem Schritt recht viel geschieht, müssen wir gelegentlich auch nur Teilschritte betrachten. Dies kann beispielsweise der Ablauf vom Beginn einer Iteration bis zur Erzeugung der vergrößerten Population X' oder auch nur die Generierung von y' aus y infolge des Mutationsoperators sein. Wenn wir von einer *Mutation* eines Ein-Kind-EAs reden, meinen wir im Gegensatz zum Begriff des Schrittes lediglich die Anwendung des Mutationsoperators. In Bezug auf einen randomisierten Hillclimber wie den (1+1)-EA haben wir diese beiden Begriffe oft synonym verwenden können, da dort das in einem Schritt zur Mutation gewählte Individuum eindeutig bestimmt ist. Das durch eine Mutation von y erzeugte Individuum y' nennen wir schließlich den *Mutanten* oder das *Kind* von y und y nennen wir den *Elter* von y'.

5.2 Beweistechniken

Ebenso wie bei der Analyse des (1+1)-EA kehren einige Beweistechniken zur Abschätzung von (erwarteten) Laufzeiten und Erfolgswahrscheinlichkeiten populationsbasierter EAs immer wieder. Einige dieser Techniken sind so allgemein, dass wir sie bereits an dieser Stelle anhand der Klasse der Ein-Kind-EAs definieren können. Wir werden sowohl Techniken für obere als auch für untere Schranken der (erwarteten) Laufzeit vorstellen.

5.2.1 Einfache obere Schranken

Wir sind an oberen Schranken für die erwartete Laufzeit populationsbasierter EAs, die in die Klasse der Ein-Kind-EAs fallen, interessiert. Als Erstes werden wir dazu die in Abschnitt 2.3 bereits angerissene Methode der Potenzialfunktionen für Populationen verdeutlichen. Sei $X = \{x_1, \ldots, x_\mu\}$ eine Population und sei $L\colon \{0,1\}^n \to \mathbb{R}$ eine Funktion, die also jedem Individuum $x \in \{0,1\}^n$ einen Wert $L(x)$ zuordnet. Oft werden wir Individuen, die für die Optimierung uninteressant sind, negative L-Werte geben. Sei weiterhin $L(X) := \max\{L(x) \mid x \in X\}$ das *L-Potenzial der Population X*. Wir nehmen implizit an, dass L zu maximieren sei und bezeichnen L als *monoton* bezüglich einer Zielfunktion $f\colon \{0,1\}^n \to \mathbb{R}$, wenn für alle $x, x' \in \{0,1\}^n$ die Implikation

$$f(x') \geq f(x) \;\Rightarrow\; L(x') \geq L(x)$$

gilt. Aus der oben als Elitismus bezeichneten Eigenschaft folgt dann für jede bezüglich der Zielfunktion monotone Funktion $L\colon \{0,1\}^n \to \mathbb{R}$, dass das L-Potenzial der Populationen eines Ein-Kind-EAs während des Laufes nicht sinken kann.

Betrachten wir eine aktuelle Population eines Ein-Kind-EAs, die nicht die initiale Population zu sein braucht. Oft ist es hilfreich, davon ausgehend die Zeit bis zum Erreichen von (Teil)zielen der Optimierung mithilfe einer monotonen Funktion L abzuschätzen. Sei daher $s(i) > 0$ eine untere Schranke für die Wahrscheinlichkeit, dass eine $1/n$-Standardmutation, angewandt auf ein Individuum mit L-Wert $i \geq 0$, ein Individuum mit einem größeren L-Wert erzeugt. Wir erhalten folgende obere Schranke.

Lemma 5.2.1 *Seien für einen Ein-Kind-EA A eine Zielfunktion f, eine bez. f monotone Funktion L, untere Schranken $s(i)$ wie oben und ein L-Potenzial der aktuellen Population von $j \geq 0$ gegeben. Bezeichne τ^* die zufällige Zahl von weiteren Schritten, bis sich das L-Potenzial von A insgesamt um einen Wert von mindestens k erhöht hat. Dann gilt $E(\tau^*) \leq \mu \sum_{i=j}^{j+k-1} 1/s(i)$.*

Wenn die Wahrscheinlichkeit, ein Individuum, dessen L-Wert dem L-Potenzial der Population gleicht, zur Mutation zu wählen, stets durch ein $p > 0$ nach unten beschränkt ist, gilt sogar $\mathrm{E}(\tau^*) \leq (1/p) \sum_{i=j}^{j+k-1} 1/s(i)$.

Beweis: Wenn i das L-Potenzial der aktuellen Population bezeichnet, ist i gemäß Definition das Maximum der L-Werte der Individuen der Population. Wir warten auf einen Schritt, der ein Individuum mit L-Wert i zur Mutation auswählt und einen Mutanten mit größerem L-Wert erzeugt. Aus der Monotonie von L folgt, dass der Mutant dann den maximalen f-Wert in der vergrößerten Population hat. Also löscht der betrachtete Schritt den Mutanten aufgrund des Elitismus keinesfalls. Wir werden zeigen, dass ein solcher Schritt stets Wahrscheinlichkeit mindestens $s(i)/\mu$ und im Fall, dass ein p mit den genannten Eigenschaften vorgegeben ist, mindestens $s(i) \cdot p$ hat. Das Lemma folgt dann durch Abschätzung der erwarteten Zeit für einen solchen Schritt mithilfe einer geometrisch verteilten Zufallsvariablen und durch Aufsummieren.

Aufgrund der Monotonie von L existiert ein Individuum mit maximalem L-Wert in der Population, das auch einen maximalen f-Wert besitzt. Also beträgt die Wahrscheinlichkeit, ein Individuum mit aktuell maximalem L-Wert zu wählen, gemäß Definition 5.1.1 mindestens $1/\mu$ und gegebenenfalls sogar mindestens p. Beide Abschätzungen für die Wahrscheinlichkeit des gewünschten Schritts folgen nun durch Multiplikation der Wahrscheinlichkeiten. $\square$

Wir verallgemeinern nun die in Abschnitt 2.3 vorgestellte Ranggruppenmethode. Ein denkbares monotones Potenzial kann der Index der Ranggruppe sein, in der sich ein bestes Individuum einer Population befindet. Für eine Zielfunktion $f\colon \{0,1\}^n \to \mathbb{R}$, eine Population X und eine f-basierte Partition $L_1, \ldots, L_m$ nehmen wir uns ein Individuum $x \in \operatorname{argmax}\{f(x) \mid x \in X\}$ vor und betrachten die Ranggruppe L_i, in der sich x befindet. Damit ordnen wir gedanklich der gesamten Population X die Ranggruppe L_i bzw. Potenzial i zu.

Aus Definition 2.3.9 übernehmen wir weiterhin den Begriff der Verlassenswahrscheinlichkeiten $s(i)$ für die f-basierte Partition. Hier rufen wir uns in Erinnerung, dass ein Ein-Kind-EA laut Definition 5.1.1 den $1/n$-Standardmutationsoperator einsetzt und anhand dessen stets für jedes i eine positive untere Schranke $s(i)$ für die Wahrscheinlichkeit, die i-te Ranggruppe zu verlassen, gefunden werden kann. Als Verallgemeinerung von Lemma 2.3.11 (bezogen für den (1+1)-EA) erhalten wir nun folgende obere Schranke.

Lemma 5.2.2 *Sei A ein Ein-Kind-EA mit Populationsgröße μ, sei $L_1, \ldots, L_m$ eine f-basierte Partition und seien $s(i)$, $1 \leq i \leq m-1$, entsprechende Verlassenswahrscheinlichkeiten. Für die Laufzeit $T_{A,\mu,f}$ gilt*

$$\mathrm{E}(T_{A,\mu,f}) \leq \mu + \mu \sum_{i=1}^{m-1} \frac{1}{s(i)}.$$

Beweis: Der Summand μ spiegelt die Zahl der Zielfunktionsauswertungen in der Initialisierung wider. Wir machen uns klar, dass der Index der Ranggruppe, in der sich ein Individuum x befindet, eine bezüglich f monotone Funktion ist. Damit folgt das Lemma aus Lemma 5.2.1. □

Die in diesem Unterabschnitt vorgestellten Obere-Schranken-Methoden werden in den Kapiteln 6 und 7 aufgegriffen und z. T. erweitert werden.

5.2.2 Die Methode der Analyse von Stammbäumen

Wir kommen nun zu einem Ansatz, der im Wesentlichen dazu dienen wird, untere Schranken für die Laufzeit populationsbasierter EAs herzuleiten. Unser Ansatz beruht auf einer Struktur, die Eigenschaften eines Laufs eines evolutionären Algorithmus modelliert. Die Namen dieser Struktur und zugehöriger Eigenschaften sind wiederum von Begriffsbildungen aus der Biologie inspiriert.

Definition 5.2.3 (Stammbaum) *Sei A ein Ein-Kind-EA und sei x ein Individuum der initialen Population von A. Der* Stammbaum $T_t(x)$ von x zum Zeitpunkt t, *$t \geq 0$, ist ein ungerichteter Baum (V_t, E_t) mit $V_t \subseteq \mathbb{N}_0$ und einer Abbildung $c_t \colon V_t \to \{0,1\}^n$ und wird induktiv definiert.*

Der Baum $T_0(x)$ erfüllt die Bedingungen $V_0 = \{0\}$, $E_0 = \emptyset$ und $c_0(0) = x$. Falls A im t-ten Schritt das Individuum x zur Mutation wählt (Fall 1) oder das im t'-ten Schritt, $t' < t$, erzeugte Individuum zur Mutation wählt und $t' \in V_t$ gilt (Fall 2), bezeichne y das Ergebnis der entsprechenden Mutation. Dann ist $V_t := V_{t-1} \cup \{t\}$, c_t entsteht aus c_{t-1} durch Erweiterung um die Vorschrift $c_t(t) := y$ und es ist $E_t := E_{t-1} \cup \{\{0,t\}\}$ im Fall 1 bzw. $E_t := E_{t-1} \cup \{\{t',t\}\}$ im Fall 2. Andernfalls ist $V_t := V_{t-1}$, $E_t := E_{t-1}$ und $c_t := c_{t-1}$. Der Knoten 0 ist stets der Wurzelknoten.

Ganz streng genommen ist das in Definition 5.2.3 auftretende Individuum x mehr als ein Suchpunkt aus $\{0,1\}^n$, da es zur Konstruktion des Stammbaums erforderlich ist, dass wir uns bei der Ausführung des Ein-Kind-EAs außerdem merken, welche Individuen der aktuellen Population noch welchen initialen Individuen entsprechen und in welchem Schritt die übrigen Individuen erzeugt wurden. Dies hätte aber die Definitionen unnötig verkompliziert. Aus demselben Grunde haben wir Stammbäume nur für initiale Individuen formuliert, werden aber später auch die Folge von Stammbäumen für ein (zufälliges) Individuum $x \in X^{(t^*)}$ aus einer Population zu einem speziellen Zeitpunkt t^* analysieren. Weil diese Stammbäume analog definiert werden können und es auf t^* nicht ankommen wird, tun wir dann so, als sei $t^* = 0$, und bezeichnen den Stammbaum von x zur Zeit $t^* + s$ trotzdem mit $T_s(x)$.

Im Folgenden identifizieren wir die Knoten eines Stammbaums oft mit Individuen aus speziellen Populationen. Wir behalten die Abbildung c_t aber im Hinterkopf, wenn wir davon reden, dass ein Knoten eines Stammbaums mit einem Individuum aus einer Population „beschriftet" sei oder ähnliche Formulierungen verwenden. Es ist wichtig festzuhalten, dass die in Definition 5.2.3 mit $\{t', t\}$ bezeichnete Kante selbst dann eingefügt wird, wenn A das Individuum y im t-ten Schritt unmittelbar aus der vergrößerten Population löscht. Wir sagen dann trotzdem, dass der Knoten im t-ten Schritt eingefügt wird und zum Zeitpunkt t erstmals vorhanden ist. Weiterhin gilt, dass $T_t(x)$ maximal $t+1$ Knoten besitzt. Die Knoten von $T_t(x)$ sind mit Individuen beschriftet, die zu irgendeinem Zeitpunkt $t' \leq t$ Elemente der aktuellen Population oder zumindest Mutant waren. Wenn ein Knoten aus einem Stammbaum $T_t(x)$ einem Individuum entspricht, das in der Population zum Zeitpunkt t noch vorhanden ist, bezeichnen wir den Knoten als *lebendig* und andernfalls als *tot*. Es ist klar, dass es zu jedem Zeitpunkt t mindestens ein x aus der initialen Population geben muss, in dessen Stammbaum $T_t(x)$ ein Knoten lebendig ist.

Bei einem Stammbaum $T_t(x)$ handelt es sich also um eine zufällige, ungerichtete Graphstruktur, die Eltern-Kind-Beziehungen von Individuen und sogar die gesamte Historie eines Individuums x' aus der Population zur Zeit t enthält. Mit Historie bezeichnen wir dabei die Folge der Individuen $x_0 = x, x_1, \ldots, x_k = x'$ mit $k \leq t$, sodass x_{i+1}, $0 \leq i \leq k-1$, durch (direkte) Mutation von x_i entstanden ist. In diesem Zusammenhang sprechen wir auch davon, dass x' ein Nachfahre von x sei. Wir finden diese Historie in $T_t(x)$ auf dem eindeutigen Pfad $0, t_1, \ldots, t_k$ von der Wurzel zu dem Knoten t_k, der mit x' beschriftet ist, wieder. Die Zahlen auf dem Pfad sagen uns dann, in welchen Schritten die Individuen, mit denen die Knoten beschriftet sind, erzeugt wurden; genauer gesagt wurde das Individuum am Knoten t_i im t_i-ten Schritt erzeugt. Die Betrachtung solcher Pfade wird im Folgenden eine entscheidende Rolle spielen. Wir werden dabei oft implizit eine Richtung für einen Pfad annehmen, die von Knoten geringer Tiefe (bezüglich der Wurzel) zu Knoten großer Tiefe verläuft.

Die historienbasierten Stammbäume können als nahe liegende, von evolutionären Algorithmen gebildete Struktur verstanden werden. Genauer gesagt induziert jeder Ein-Kind-EA für jedes initiale Individuum x einen stochastischen Prozesse $T(x) :=$

$(T_t(x))_{t\geq 0}$, der die unendliche, im Laufe des EAs entstehende Folge von Stammbäumen von x zu den Zeitpunkten t repräsentiert. Stammbäume wurden daher bereits erfolgreich als Analysetechniken eingesetzt. Rabani, Rabinovich und Sinclair (1998) betrachten sogar kompliziertere Stammbäume in mit Rekombinationsoperatoren versehenen EAs (aber ohne Mutation), deren Populationsgröße zunächst der Einfachheit halber auf unendlich gesetzt wird. Einige Eigenschaften dieser Stammbäume kommen ihnen dabei bei der Frage nach der so genannten *mixing time* des zugehörigen stochastischen Prozesses, also der Zeit, bis der Prozess genügend nahe an seiner stationären Verteilung ist, zugute. Zudem gelingt es den Autoren wiederum mithilfe der Stammbäume, die dabei gewonnenen Resultate zum Teil auf endliche Populationsgrößen zu übertragen. Kürzlich hat Ollivier (2003) die dazugehörigen Ergebnisse verbessert. Mit *mixing times* werden wir uns im Weiteren aber nicht beschäftigen.

Wir wollen uns nun überlegen, warum Stammbäume bei der Suche nach unteren Schranken für die (erwartete) Laufzeit von Ein-Kind-EAs eingesetzt werden können. Dazu arbeiten wir zunächst heraus, welche Eigenschaften ein vom Lauf des einfachen (1+1)-EA beschriebener Stammbaum aufweist. Offenkundig konstruiert der (1+1)-EA bis zum Zeitpunkt t genau einen Stammbaum $T_t(x)$ mit dem initialen Suchpunkt als Wurzel und mit $t+1$ Knoten. Wir unterteilen den Stammbaum $T_t(x)$ in Ebenen, sodass auf Ebene i genau die Knoten der Tiefe i liegen. Es folgt nun, dass höchstens ein Knoten auf Ebene i einen Nachfahren auf Ebene $i+1$ haben kann, da zu jedem Zeitpunkt nur ein Knoten im Stammbaum des (1+1)-EA lebendig sein kann. Wenn $T_t(x)$ nun eine geringe *Tiefe* d_t besitzt, bedeutet dies, dass viele Knoten im Baum einen hohen Grad haben, also viele Mutanten verworfen wurden. Alternativ können wir mit der Länge von Pfaden in $T_t(x)$ argumentieren. Die Tiefe d_t beschränkt zunächst die Länge jedes Pfades in $T_t(x)$. Andererseits entspricht ein Pfad von x nach x' wie oben erwähnt einer Folge von direkten Mutationen, die aus x über Zwischenindividuen das Individuum x' erzeugen. Bei geringer Länge kann dies implizieren, dass sich x' von x mit hoher Wahrscheinlichkeit nicht allzu stark unterscheidet. Wenn wir nun wissen, dass das initiale Individuum x einen hohen Hammingabstand zu jedem optimalen Suchpunkt besitzt, können wir schlussfolgern, dass es unwahrscheinlich ist, in einem Stammbaum geringer Tiefe das Optimum zu finden.

Die vorangehenden Überlegungen möchten wir auch auf allgemeinere Ein-Kind-EAs, insbesondere solche mit $\mu > 1$, übertragen. Bezüglich solcher allgemeiner EAs lässt sich die Struktur von Stammbäumen zwar nicht mehr so gut wie beim (1+1)-EA fassen, aber der Zusammenhang zwischen der Tiefe und der Optimierung bleibt prinzipiell bestehen. Gleichwohl sollte man genau beachten, welche Bedingungen man bei der Untersuchung eines Stammbaums stellt. In Anwendungen werden wir häufig für ein fest gewähltes initiales Individuum x seinen Stammbaum $T_t(x)$ zu irgendeinem Zeitpunkt t betrachten. Oft untersuchen wir dann einen Pfad $p = (0, t_1, \ldots, t_k)$ aus dem Stammbaum $T_t(x)$. Selbst wenn wir keine weiteren Anforderungen an p stellen, also p beispielsweise gleichverteilt aus der Menge der im Stammbaum vorliegenden Pfade wählen, stellen wir aufgrund der Beobachtung die Anforderung, dass p entstehen konnte. Welche Pfade aber mit welchen Wahrscheinlichkeiten entstehen, hängt vom Zusammenspiel der Mutations- und Selektionsoperatoren des zugrunde liegenden Ein-Kind-EAs ab. Auch wenn das Verhalten des $1/n$-Mutationsoperators, d. h. die Auswahl der zu kippenden Bits, in einem Schritt eines Ein-Kind-EAs zunächst unabhängig vom Verhalten in anderen Schritten ist, können wir nicht folgern, dass die Beschriftungen auf p infolge unabhängiger Anwendungen des Mutationsoperators entstanden seien. Dies funktioniert nur, wenn die Selektionsoperatoren überhaupt nicht vom Ergebnis der Mutationen abhängen.

Beobachtung 5.2.1 Sei A ein Ein-Kind-EA, dessen Selektion zur Reproduktion und Selektion zur Ersetzung stets gleichverteilt aus der vorliegenden Population wählen. Sei $0, t_1, \dots, t_k$ ein Pfad in einem Stammbaum $T_t(x)$ für ein Individuum x aus einer Population von A. Dann verhält sich der Mutationsoperator von A im t_i-ten Schritt, $1 \le i \le k$, unabhängig als $1/n$-Standardmutationsoperator.

Die Situation aus Beobachtung 5.2.1 trifft laut Definition 5.1.1 beispielsweise auf konstante Zielfunktionen zu. Natürlich ist die Betrachtung solcher nicht interessant, sodass Beobachtung 5.2.1 lediglich einen Idealfall darstellt, zu dem wir später mithilfe von Abschätzungen kommen werden. Wenn diese Abschätzungen gelingen, können wir die Chernoffungleichung auf die verschiedenen Mutationen, die entlang des Pfades beschrieben werden, anwenden. In anderen Analysen können wir zwar nicht mit Abschätzungen die Situation aus Beobachtung 5.2.1 herstellen, aber trotzdem über die Folge abhängiger Mutationen auf einem Pfad argumentieren.

Wir halten nun die folgende, intuitive Deutung fest. Wenn der Stammbaum eines jeden Individuums aus der initialen Population nach einer gegebenen Zeit noch flach ist, ist es unwahrscheinlich, dass eine (nicht zu einfache) Funktion optimiert wurde. Im Laufe dieses Teils der Arbeit werden wir daher obere Schranken für die Tiefe von Stammbäumen für vorgegebene Ein-Kind-EAs und Zielfunktionen erarbeiten. Als Vorgriff sei erwähnt, dass wir dabei in manchen Fällen sogar vom konkreten Mutationsoperator des EAs abstrahieren können. Damit stellt sich die Frage, inwieweit sich die Technik der Analyse von Stammbäumen auf kompliziertere populationsbasierte EAs und andere Mutationsoperatoren oder gar Suchräume erweitern lässt. Diesem wollen wir uns in einer abschließenden Diskussion am Ende dieses Teils der Arbeit widmen.

6 Der Nutzen von Populationen in evolutionären Algorithmen

In diesem Kapitel wollen wir die in Abschnitt 5.1 angerissene Frage nach dem Nutzen von Populationen in evolutionären Algorithmen näher beleuchten. Während einerseits mit einer Reihe von Arbeiten (Jansen und Wegener, 2001c, 2002; Storch und Wegener, 2003) der Nutzen von Rekombinationsoperatoren belegt wurde, sind andererseits einige Szenarien bekannt, in denen eine einfache Suchheuristik, die ähnlich wie RLS arbeitet, einem populationsbasierten EA mit Rekombination zumindest in empirischer Hinsicht (Juels und Wattenberg, 1996) überlegen ist. Darüber hinaus konstruierten Mitchell, Forrest und Holland (1992) Beispielfunktionen, die aus Kapitel 3 bekannten Royal-Road-Funktionen, mit der Absicht, einen buchstäblichen Königsweg für genetische Algorithmen mit Rekombinationsoperatoren zu eröffnen. Zur Überraschung derselben Autoren (Mitchell, Holland und Forrest, 1994) erwies sich auf diesen Funktionen aber RLS als überlegene Suchheuristik, was die Autoren auch mit einer theoretischen Analyse, die der in Kapitel 3 für RLS vorgestellten Laufzeitanalyse ähnelt, zu untermauern vermochten.

Es stellt sich also die Frage, ob ein populationsbasierter EA, der noch nicht einmal über einen Rekombinationsoperator verfügt, den einfachen randomisierten Suchheuristiken aus Teil I überlegen sein kann. Schließlich sind aus dem Feld der Evolutionsstrategien populationsbasierte EAs bekannt, in denen ganz bewusst auf Rekombination verzichtet wird. Dieses als Frage nach dem Nutzen von Populationen bezeichnete Problem sollte, um zufrieden stellend gelöst werden zu können, in folgendem Szenario geklärt werden.

Da RLS keine globale Suchheuristik ist, der (1+1)-EA aber als einfachster evolutionärer Algorithmus gilt, wählen wir den (1+1)-EA als diejenige Suchheuristik, der ein populationsbasierter evolutionärer Algorithmus A ohne Rekombination gegenübergestellt wird. Damit der zu zeigende Vorteil des EAs A tatsächlich seiner Population zugeschrieben werden kann, sollte A denselben Mutationsoperator wie der (1+1)-EA, d. h. die $1/n$-Standardmutation, nutzen. Weiterhin sollte A in gewisser Weise als natürlicher, an praktischen EAs angelehnter populationsbasierter EA gelten. Auch wenn dies eine eher „weiche“ Anforderung verkörpert, halten wir fest, dass A nach Möglichkeit bereits bekannte Selektionsoperatoren nutzen sollte. Des Weiteren wünschen wir, dass wir den (1+1)-EA gerade als Spezialfall von A erhalten, wenn wir die Populationsgröße auf 1 setzen.

Zu guter Letzt konkretisieren wir, was wir darunter verstehen, dass A dem (1+1)-EA auf einer noch zu wählenden Zielfunktion f überlegen sein soll. Einerseits heißt dies, dass die erwartete Laufzeit von A, gemessen in der Zahl der Zielfunktionsauswertungen, geringer als die des (1+1)-EA ist. Aus komplexitätstheoretischer Sicht sind wir aber üblicherweise an einer Unterscheidung zwischen polynomiellen und nicht polynomiellen Laufzeiten interessiert. Daher soll A eine polynomielle erwartete Laufzeit besitzen, während die des (1+1)-EA superpolynomiell sein soll. Außerdem haben wir in Teil I der Arbeit (vgl. beispielsweise Theorem 4.4.3) gesehen, dass parallele Läufe des (1+1)-EA manchmal deutliche Effizienzsteigerungen bewirken. Daher stellen wir die zusätzliche Anforderung, dass auch polynomiell viele parallele Läufe des (1+1)-EA auf f eine exponentielle erwartete Laufzeit an den Tag legen sollen.

Bereits Jansen und Wegener (2001b) sind im Wesentlichen dem beschriebenen Ansatz gefolgt und haben somit den Nutzen von Populationen in evolutionären Algorithmen gezeigt. In ihrer Arbeit konstruieren sie eine Beispielfunktion f, sodass der (1+1)-EA mit einer Wahrscheinlichkeit von $1 - o(1/\text{poly}(n))$ eine superpolynomielle Laufzeit benötigt, wohingegen der von ihnen definierte populationsbasierte EA eine erwartete polynomielle Laufzeit besitzt. Die intuitive Erklärung besteht darin, dass die Funktion f ein lokales Optimum besitzt, in das der (1+1)-EA mit hoher Wahrscheinlichkeit gelangt und zu dessen Verlassen eine Mutation erforderlich ist, die viele Bits auf einmal kippt. Auf der anderen Seite besitzen die Punkte in der Nachbarschaft des lokalen Optimums einen sehr ähnlichen, aber natürlich geringfügig geringeren f-Wert als das lokale Optimum. Der populationsbasierte EA besitzt dann eine genügend große Wahrscheinlichkeit, seine Individuen in der Nachbarschaft des lokalen Optimums zu verteilen. Da manche Punkte in der Nachbarschaft näher an global optimalen Suchpunkten sind, kann der populationsbasierte EA über den Umweg solcher Punkte besser zum globalen Optimum gelangen. In einem Satz gesagt beruht der Vorteil des populationsbasierten EAs darauf, dass geringfügig schlechtere Suchpunkte in die Population aufgenommen werden können und damit ein kleines Tal mit geringen f-Werten besser verlassen werden kann. Es wurde nicht untersucht, ob ein Suchverfahren wie Simulated Annealing, das geringfügige Verschlechterungen des f-Wertes in einer Mutation mit nicht zu kleiner Wahrscheinlichkeit akzeptiert, ebenfalls einen Laufzeitvorteil gegenüber dem (1+1)-EA auf der Beispielfunktion brächte.

Auch wenn Jansen und Wegener (2001b) die grundsätzliche Frage nach dem Nutzen von Populationen in Bezug auf die Laufzeit beantwortet haben, bleiben offene Fragen. Zum einen wäre es überzeugender, wenn auch ein exponentiell großer Laufzeitunterschied anstelle eines nur superpolynomiell großen Laufzeitunterschieds zwischen dem (1+1)-EA und dem populationsbasierten EA gezeigt werden könnte. Zum anderen erhalten wir den (1+1)-EA als Spezialfall verschiedener populationsbasierter EAs. Wünschenswert wäre es damit, eine Beispielfunktion zu finden, bei der ein populationsbasierter EA bei einer Populationsgröße μ_1 einen exponentiellen Laufzeitvorteil gegenüber demselben EA bei einer Populationsgröße μ_2 mit $1 < \mu_2 < \mu_1$ hätte. Drittens steht die Frage im Raum, ob der Bereich der Populationsgröße, in dem die (erwartete) Laufzeit von einem exponentiellen auf einen polynomiellen Wert wechselt, eingegrenzt oder sogar durch die Definition der Zielfunktion festgelegt werden kann.

Viertens darf man überlegen, ob sich auch der „Schaden durch Populationen“ rigoros beweisen lässt, d. h., eine Erhöhung der Populationsgröße um einen polynomiell kleinen Faktor die erwartete Laufzeit um einen exponentiell großen Faktor ansteigen lässt.

Schließlich weisen wir noch auf ein Manko hin, das der populationsbasierte EA von Jansen und Wegener (2001b) in Bezug auf die oben genannten Anforderungen aufweist. Jener EA bringt einen Operator zur so genannten Diversitätserhaltung ein, der dafür sorgt, dass sich ein Mutant auf jeden Fall von seinem Elter unterscheidet. Damit fällt es schwer, den EA als Verallgemeinerung des (1+1)-EA anzusehen. Im Rahmen dieses Kapitels wollen wir versuchen, diese Anforderung zu erfüllen, und uns der vier oben genannten Fragestellungen annehmen.

6.1 Ein einfacher, populationsbasierter EA

Wir definieren hier einen sehr einfachen populationsbasierten EA ohne Rekombination, der stark dem von Jansen und Wegener (2001b) untersuchten Algorithmus ähnelt. Der im Folgenden betrachtete EA zeichnet sich durch eine so genannte fitnessproportionale Selektion zur Reproduktion und zur Ersetzung aus. Wie anhand der Diskussion in Abschnitt 5.1 zu erwarten, nutzen wir die $1/n$-Standardmutation, sorgen dafür, dass der EA elitär ist, und erzeugen einen Ein-Kind-EA.

Definition 6.1.1 (Beispiel-GA) *Seien alle in dieser Definition auftretenden Mengen Multimengen. Der* Beispiel-GA *mit Populationsgröße μ für $f\colon \{0,1\}^n \to \mathbb{R}^+$ läuft wie folgt ab.*

Setze $t := 0$. Initialisiere $x_i \in \{0,1\}^n$ für $1 \le i \le \mu$ unabhängig und zufällig gleichverteilt. Setze $X^{(0)} := \{x_1, \dots, x_\mu\}$.

Wiederhole

Sei $X^{(t)} = \{x_1, \dots, x_\mu\}$. Wähle $y \in X^{(t)}$, sodass $\mathrm{Prob}(y = x_i) = \frac{f(x_i)}{\sum_{j=1}^{\mu} f(x_j)}$ *gilt.*

Erzeuge $x_{\mu+1}$ aus y durch $1/n$-Standardmutation.

Benenne $x_1, \dots, x_{\mu+1}$ mit einer Indexpermutation derart um, dass $f(x_1)$ maximal und $f(x_{\mu+1})$ minimal ist. Falls es mehrere Kandidaten für x_1 oder $x_{\mu+1}$ gibt, seien x_1 bzw. $x_{\mu+1}$ zufällig gleichverteilt aus diesen gewählt.
Wähle $y \in \{x_2, \dots, x_{\mu+1}\}$, sodass $\mathrm{Prob}(y = x_i) = \frac{f(x_1)+f(x_{\mu+1})-f(x_i)}{\sum_{j=2}^{\mu+1}(f(x_1)+f(x_{\mu+1})-f(x_j))}$ *gilt.*

Setze $X^{(t+1)} := \{x_1, \dots, x_{\mu+1}\} \setminus \{y\}$ und $t := t+1$.

Wir nennen den Algorithmus aus Definition 6.1.1 „Beispiel-GA“, da er viele Merkmale eines **g**enetischen **A**lgorithmus (Holland, 1975; Goldberg, 1989) in sich trägt, nämlich die fitnessproportional arbeitenden Selektionsoperatoren sowie den Suchraum $\{0,1\}^n$. Das Attribut „Beispiel“ soll andeuten, dass sich viele der kommenden Ergebnisse dieses Kapitels auch für Varianten des Beispiel-GAs zeigen ließen. Mit den ausgesuchten

Operatoren wollen wir jedoch die eingangs erwähnte Anforderung, einen möglichst „natürlichen“ populationsbasierten EA ohne Rekombination zu analysieren, erfüllen. Es stellt keine Einschränkung dar, dass der Beispiel-GA nur für Funktionen mit positivem Bildbereich definiert ist, da jede pseudoboolesche Funktion aufgrund der Endlichkeit des Suchraums in eine solche überführt werden kann. Auf die Auswirkungen einer solchen Transformation auf die Optimierungszeit gehen wir hier nicht ein.

Zur Beschreibung unseres Beispiel-GAs fassen wir zusammen, dass die Wahrscheinlichkeit, ein Individuum y zur Mutation auszuwählen, proportional zu seinem f-Wert ist. Das aus der vergrößerten Population mit $\mu+1$ Elementen zu löschende Individuum wählt der Beispiel-GA im Wesentlichen im umgekehrten Verhältnis zu seinem f-Wert, behält aber auf jeden Fall ein bestes Individuum und ist damit elitär. Damit erhalten wir auch, dass unser Beispiel-GA ein Ein-Kind-EA ist. Es sei hier angemerkt, dass der von Jansen und Wegener (2001b) betrachtete EA sich aus dem Beispiel-GA ergibt, indem der oben erwähnte Operator zur Diversitätserhaltung ergänzt wird.

Der Beispiel-GA mit $\mu = 1$ und der (1+1)-EA unterscheiden sich nur an einer unbedeutenden Stelle. Wenn in einem Schritt der Mutant denselben f-Wert wie sein Elter besitzt, löscht der Beispiel-GA eines dieser beiden Individuen gleichverteilt, wohingegen der (1+1)-EA gemäß Definition 2.3.1 auf jeden Fall den Mutanten übernimmt. Dieser Unterschied wird aber im Folgenden kaum eine Rolle spielen und man kann sich überlegen, dass die beiden Varianten stets asymptotisch identische erwartete Laufzeiten haben. Auf manchen einfachen Beispielfunktionen, z. B. OneMax, sind die Varianten mit $\mu = 1$ hinsichtlich ihrer Laufzeit sogar austauschbar. Die Laufzeit des Beispiel-GAs messen wir schließlich wie auf Seite 92 anhand des Ein-Kind-EAs erläutert, sodass nur Werte $\mu = \text{poly}(n)$ vernünftig erscheinen.

6.2 Eine Beispielfunktion

Wenden wir uns nun der bereits in den Raum gestellten Frage zu, ob der Beispiel-GA mit Populationsgröße μ_1 deutlich besser sein kann als der Beispiel-GA mit Populationsgröße $\mu_2 < \mu_1$. Wir suchen daher eine beispielhafte Zielfunktion f, für die dieses Verhalten gilt. Diese Beispielfunktion wird eine relativ komplexe Gestalt aufweisen und insbesondere mehrere lokale Optima besitzen. Andererseits glauben wir, dass sich der Nutzen von Populationen auf unimodalen Funktionen (vgl. Abschnitt 3.1) nicht zeigen kann, sondern dass große Populationen auf unimodalen Funktionen sich als eher hinderlich erweisen. Dies wird in Kapitel 7 zumindest für den dort untersuchten (μ+1)-EA klar werden.

6.2.1 Idee und Definition der Beispielfunktion

Die noch zu definierende Beispielfunktion f beruht auf folgendem, zunächst sehr informal skizziertem Grundgedanken. Angenommen, wir kennen zwei pseudoboolesche

Funktionen f_1 und f_2. Es sei nun bekannt, dass der Beispiel-GA mit kleinen Werten von μ die Funktion f_1 (im Erwartungswert) schneller optimiert als f_2. Andererseits soll auch gelten, dass zumindest die Laufzeit auf der Funktion f_1 mit der Populationsgröße μ wächst; die Laufzeit auf f_2 hingegen soll nicht mit μ oder zumindest weniger stark als bei f_1 wachsen. Dann erhalten wir f durch geschickte „Kombination" von f_1 und f_2 auf folgende Weise. Wenn der Beispiel-GA auf f eine optimale Belegung für den f_2-Bestandteil von f vor einer optimalen Belegung für den f_1-Bestandteil erzeugt, soll ein global optimaler Suchpunkt gefunden sein. Tritt allerdings ein, dass der f_1-Bestandteil vor dem f_2-Bestandteil optimiert ist, soll sich die „Fitnesslandschaft" dergestalt ändern, dass der Beispiel-GA mit großer Wahrscheinlichkeit für lange Zeit in einem lokalen Optimum hängen bleibt.

Wenn das beschriebene lokale Optimum nicht in polynomieller Zeit verlassen werden kann, scheint es möglich zu sein, mit einer der genannten Idee folgenden Funktion f den Nutzen von Populationen nachweisen zu können. Unter den gewünschten Voraussetzungen erhalten wir dann nämlich, dass der Beispiel-GA bei kleinen Populationen (mit hoher Wahrscheinlichkeit) gerade den f_1-Bestandteil vor dem f_2-Bestandteil optimiert, damit das schwer zu verlassende lokale Optimum betritt und eine lange Optimierungszeit erforderlich wird. Wenn auf der anderen Seite eine Vergrößerung der Population die Optimierungszeit von f_1 stärker erhöht als die von f_2, lässt dies darauf schließen, dass ab einer kritischen Populationsgröße der f_2-Bestandteil (mit hoher Wahrscheinlichkeit) vor dem f_1-Bestandteil optimiert wird und somit ein globales Optimum gefunden ist.

Tatsächlich ist es möglich, diese allgemeine Idee in die Tat umzusetzen. Die „Kombination" von f_1 und f_2 innerhalb von f realisieren wir durch Definition der beiden Funktionen auf disjunkten Teilen des Suchpunktes und anschließende Addition der Bestandteile, müssen aber die daraus resultierenden Folgen später sorgfältig abschätzen. Die Forderung, dass die Optimierungszeit von f_2 von der Populationsgröße möglichst wenig beeinflusst wird, erreichen wir, indem wir f_2 deutlich größere Werte als f_1 geben und die fitnessproportionalen Selektionsoperatoren des Beispiel-GAs ausnutzen. Als f_1 werden wir letztendlich eine Variante der bekannten Funktion OneMax, als f_2 eine Variante von LeadingOnes verwenden.

Wir kommen nun zur Definition von f. Sei $\ell := \lceil n^{1/2}/45 \rceil$ und $m := n - \ell$. Dabei sei o. B. d. A. m durch 3 teilbar. Wir unterteilen Individuen $x \in \{0,1\}^n$, also Suchpunkte mit n Bits, in das Präfix $(x_1, \ldots, x_m)$ der Länge m und das Suffix $(x_{m+1}, \ldots, x_n)$ der Länge ℓ. Außerdem sei

$$\mathrm{PE}(x) := \sum_{i=1}^{m} x_i$$

die Zahl der **P**räfix**e**insen sowie

$$\mathrm{FSE}(x) := \sum_{i=0}^{\ell-1} \prod_{j=0}^{i} x_{m+1+j}$$

die Zahl der **f**ührenden **S**uffix**e**insen. Aus Gründen der Übersicht wollen wir erreichen, dass das Suffix eines Individuums immer von der Gestalt $1^i0^{\ell-i}$ ist. Wir sagen daher, dass $x \in \{0,1\}^n$ *wohlgeformt* ist, wenn $(x_{m+1},\ldots,x_n) = 1^i0^{\ell-i}$ für ein $i \in \{0,\ldots,\ell\}$ gilt. Aus noch zu erläuternden Gründen soll im Präfix zwar im Allgemeinen die Zahl der Einsen erhöht werden, sie soll aber doch nicht zu nahe bei m liegen. Daher definieren wir $b := \lceil n^{1/2}/(90\log n)\rceil$ und die Funktion

$$\mathrm{PE}^*(x) \;:=\; \frac{2m}{3} + b - \left|\frac{2m}{3} + b - \mathrm{PE}(x)\right|.$$

Damit erhalten wir $\mathrm{PE}^*(x) = \mathrm{PE}(x)$ für $\mathrm{PE}(x) \leq 2m/3 + b$ und andernfalls $\mathrm{PE}^*(x) = 2m/3 + b - (\mathrm{PE}(x) - 2m/3 - b) \geq m/3 + 2b$. Sei nun

$$f(x) \;:=\; \begin{cases} \mathrm{PE}(x) + n^{2n(\mathrm{FSE}(x)+1)}, & \text{falls } x \text{ wohlgeformt und } \mathrm{PE}(x) \leq \frac{2m}{3},\\ n^{2n\ell+\mathrm{PE}^*(x)} + \mathrm{FSE}(x), & \text{falls } x \text{ wohlgeformt und } \frac{2m}{3} < \mathrm{PE}(x),\\ n^{1+\ell^2-\ell\sum_{i=m+1}^{n} x_i} & \text{sonst.}\end{cases}$$

Wir müssen uns nun die Struktur dieser Funktion f klar machen. Zuallererst halten wir fest, dass $f(x)$ stets positiv ist. Die ersten beiden Fälle aus der Definition sind am wichtigsten und gelten für wohlgeformte Eingaben. Zudem ist die Funktion jeweils in gewissen Teilen separierbar, d. h., sie lässt sich als Summe einer Funktion auf den Präfixbits und einer Funktion auf den Suffixbits schreiben. Betrachten wir nun den ersten Fall, der eintritt, wenn die Zahl der Einsen im Präfix noch klein ist, nämlich bei höchstens $2m/3$ liegt. Auf den Präfixbits liegt lediglich $\mathrm{PE}(x)$, also OneMax bezüglich der Präfixbits, vor. Auf den Suffixbits entspricht die Funktion einem exponentiell skalierten $\mathrm{FSE}(x)$-Wert, also einem exponentiell skalierten LeadingOnes-Wert bezüglich der Suffixbits. Somit finden wir hier die oben erwähnten Funktionen f_1 und f_2 wieder. Aufgrund der exponentiellen Skalierung beeinflusst der FSE-Wert den f-Wert deutlich stärker als der PE-Wert. Für alle x mit $\mathrm{PE}(x) = 2m/3$ und $\mathrm{FSE}(x) = \ell$ erhalten wir den (global) maximalen Wert $f(x) = 2m/3 + n^{2n(\ell+1)}$.

Der zweite Fall der Funktion f trifft auf wohlgeformte Eingaben mit einem PE-Wert von mehr als $2m/3$ zu. Aufgrund der Definition von $\mathrm{PE}^*(x)$ erhalten wir zwei Unterfälle. Den Grund werden wir gleich erläutern. Zuvor bemerken wir aber, dass der f-Wert im zweiten Fall mindestens $n^{2n\ell+m/3+2b}$ beträgt und damit größer als jeder f-Wert von Individuen, die zum ersten Fall gehören, aber einen FSE-Wert von maximal $\ell - 1$ haben, ist. Mit anderen Worten scheint es für den Beispiel-GA leicht zu sein, durch Erhöhung des PE-Wertes vom ersten Fall in den zweiten Fall zu gelangen, sofern das Suffix noch nicht die optimale Belegung mit ℓ Einsen aufweist. Weiterhin steigt der f-Wert im Unterfall $\mathrm{PE}(x) \leq 2m/3+b$ des zweiten Falls exponentiell mit dem PE-Wert an, während der FSE-Wert nur linear eingeht. Also kehrt sich hier die Gewichtung von Präfix und Suffix gegenüber dem ersten Fall um. Für x mit $\mathrm{PE}(x) = 2m/3 + b$ und $\mathrm{FSE}(x) = \ell$ erhalten wir dann aufgrund der Definition von $\mathrm{PE}^*(x)$ einen lokal optimalen Wert $f(x) = n^{2n\ell+2m/3+b} + \ell$. Besser als solche x sind nur Eingaben mit einem

PE-Wert von höchstens $2m/3$. Damit beträgt der Hammingabstand vom betrachteten lokalen Optimum zu einem besseren Punkt mindestens $b = \Omega(n^{1/2}/\log n)$, was nahe legt, dass dieses lokale Optimum nicht leicht verlassen werden kann. Nun können wir auch erwähnen, dass b geschickt gewählt wurde, um die Überwindung des lokalen Optimums doch so wahrscheinlich zu machen, dass der Beispiel-GA bei genügend großem μ eine polynomielle erwartete Laufzeit besitzt. Deshalb wird $\mathrm{PE}^*(x)$ den Beispiel-GA im zweiten Fall von f zum lokalen Optimum führen, selbst wenn der zweite Unterfall eintritt, d. h. Individuen mit $\mathrm{PE}(x) > 2m/3 + b$ entstehen.

Der dritte Fall von f schließlich gilt für Eingaben, die nicht wohlgeformt sind. In diesem Fall sinkt der f-Wert exponentiell in der Zahl der Einsen im Suffix. Damit soll der Beispiel-GA nach der Initialisierung zur Erzeugung eines wohlgeformten Individuums mit wenigen Einsen im Suffix geführt werden. Der f-Wert im dritten Fall beträgt höchstens $n^{1+\ell^2} \leq n^{n+1}$, ist also (relativ gesehen) sehr klein in Hinblick auf den minimalen f-Wert n^{2n} im ersten und zweiten Fall.

Als abschließende Bemerkung in unserer Präsentation von f halten wir noch fest, dass die Funktion selbst im logarithmischen Kostenmaß (vgl. Wegener, 2003) noch in polynomieller Zeit ausgewertet werden kann, da maximal exponentielle Zielfunktionswerte entstehen. Somit bleibt die im Black-Box-Szenario betrachtete Zahl der Zielfunktionsauswertungen eine realistische Größe für die Effizienz des Beispiel-GAs, wenn wir im Folgenden exponentielle Laufzeitunterschiede betrachten werden.

6.2.2 Grundlegende Beobachtungen für die Beispielfunktion

Im Folgenden werden wir einige grundlegende Eigenschaften für das Verhalten des Beispiel-GAs auf der Zielfunktion f erarbeiten. Zunächst führen wir dazu den Begriff einer *normalen Population* ein. Eine Population mit μ oder eine vergrößerte Population mit $\mu + 1$ Individuen heiße normal, wenn sie mindestens ein wohlgeformtes Individuum enthält und alle ihre wohlgeformten Individuen einen PE-Wert von höchstens $2m/3$ haben. Mit anderen Worten fallen alle wohlgeformten Individuen in den ersten Fall der Definition von f. Wir werden sehen, dass der Beispiel-GA auf f typischerweise für lange Zeit normale Populationen enthält, bevor er das Optimum erreicht.

Die erste Aufgabe besteht darin zu formalisieren, dass der PE-Wert von Individuen in normalen Populationen einen vernachlässigbar kleinen Einfluss auf die Selektionswahrscheinlichkeiten hat. Dazu sei bezüglich einer aktuellen Population $\{x_1, \dots, x_\mu\}$ und der oben definierten Zielfunktion f die Zufallsvariable M definiert als der Index $i \in \{1, \dots, \mu\}$ des Individuums, das im anstehenden Schritt zur Mutation gewählt wird. (Damit hängt M implizit von einem Zeitpunkt t ab, den wir aus Gründen der Übersicht nicht angeben.) Gleicherweise bezeichne für eine vergrößerte Population die Zufallsvariable D den Index $i \in \{1, \dots, \mu + 1\}$ desjenigen Individuums, das zum Löschen gewählt wird. Da Populationen Multimengen sind, kann die Indizierung der Elemente jeweils beliebig gewählt werden. Wir erhalten dann folgende einfache Beziehungen.

Lemma 6.2.1 *Sei $X = \{x_1, \ldots, x_\mu\}$ eine normale Population und seien x_i und x_j zwei wohlgeformte Individuen aus X mit* $\mathrm{FSE}(x_i) = \mathrm{FSE}(x_j)$*. Dann gilt*

$$1 - n^{-\Omega(n)} \;\leq\; \frac{\mathrm{Prob}(M = i)}{\mathrm{Prob}(M = j)} \;\leq\; 1 + n^{-\Omega(n)}.$$

Beweis: Gemäß der Definition des Beispiel-GAs (Definition 6.1.1) gilt

$$\frac{\mathrm{Prob}(M = i)}{\mathrm{Prob}(M = j)} \;=\; \frac{f(x_i)}{f(x_j)}.$$

Weil X eine normale Population ist, gilt für alle wohlgeformten $x \in X$, dass $f(x) = \mathrm{PE}(x) + n^{2n(\mathrm{FSE}(x)+1)}$. Da $\mathrm{FSE}(x_i) = \mathrm{FSE}(x_j)$, erhalten wir $f(x_i)/f(x_j) = 1 \pm n^{-\Omega(n)}$. □

Sei nun für eine normale Population X das *FSE-Potenzial* $\mathrm{FSE}(X)$ definiert als der maximale FSE-Wert der wohlgeformten Individuen in X. (Im Sinne von Abschnitt 5.2.1 benutzen wir also eine Funktion L mit $L(x) := \mathrm{FSE}(x)$ für wohlgeformte x und $L(x) := -1$ sonst. Man sieht leicht, dass es sich für normale Populationen aufgrund der Definition von f dabei um eine bezüglich f monotone Funktion handelt.) Wir wollen zeigen, dass das FSE-Potenzial eine Menge nicht zur Mutation wählbarer Individuen und eine Menge nicht zum Löschen wählbarer Individuen mit hoher Wahrscheinlichkeit festlegt.

Lemma 6.2.2 *Sei $X = \{x_1, \ldots, x_\mu\}$ eine normale und $Y = \{y_1, \ldots, y_{\mu+1}\}$ eine vergrößerte normale Population. Seien L_X und L_Y die FSE-Potenziale von X bzw. Y. Außerdem existiere mindestens ein wohlgeformtes $y_i \in Y$ mit* $\mathrm{FSE}(y_i) < L_Y$*. Falls* $\mu = \mathrm{poly}(n)$*, gelten die Beziehungen*

$$\mathrm{Prob}(\mathrm{FSE}(x_M) = L_X) \;=\; 1 - n^{-\Omega(n)} \quad \textit{und} \quad \mathrm{Prob}(\mathrm{FSE}(y_D) = L_Y) \;=\; n^{-\Omega(n)}.$$

Beweis: Für die erste Beziehung betrachten wir die Auswahlwahrscheinlichkeit zur Mutation aus Definition 6.1.1. Offensichtlich wird hier die Wahrscheinlichkeit, ein wohlgeformtes Individuum mit maximalem FSE-Wert zu wählen, minimal, falls $\mu - 1$ wohlgeformte Individuen x_i der Bedingung $\mathrm{FSE}(x_i) = L_X - 1$ und $\mathrm{PE}(x_i) = 2m/3$ genügen. Also gilt

$$\mathrm{Prob}(\mathrm{FSE}(x_M) = L_X) \;\geq\; \frac{n^{2n(L_X+1)}}{(\mu - 1) \cdot (2m/3 + n^{2nL_X}) + n^{2n(L_X+1)}} \;=\; 1 - n^{-\Omega(n)},$$

da $\mu = \mathrm{poly}(n)$.

Für die zweite Beziehung argumentieren wir auf ähnliche Weise und betrachten den Ausdruck für die Löschwahrscheinlichkeit. Offenbar wird $\mathrm{Prob}(\mathrm{FSE}(y_D) = L_Y)$ am größten, wenn Y genau μ wohlgeformte Individuen mit FSE-Wert L_Y enthält. Im

Nenner des Ausdrucks für die Löschwahrscheinlichkeit zählen wir nun nur den Term für das schlechteste Individuum, das einen FSE-Wert von höchstens $L_Y - 1$ besitzt, und erhalten

$$\text{Prob}(\text{FSE}(y_D) = L_Y) \;\leq\; \mu \cdot \frac{n^{2nL_Y} + 4m/3}{n^{2n(L_Y+1)}} \;=\; n^{-\Omega(n)},$$

da $\mu = \text{poly}(n)$. □

Nun geben wir der in Lemma 6.2.2 implizit betrachteten Menge einen Namen. Ein wohlgeformtes Individuum einer normalen Population heiße *FSE-maximal*, wenn sein FSE-Wert gerade dem FSE-Potenzial der Population gleicht.

Lemma 6.2.3 *Sei $\mu = \text{poly}(n)$ und sei $X = \{x_1, \ldots, x_\mu\}$ eine normale Population mit FSE-Potenzial L_X. Sei zudem $B := \{x \in X \mid x$ wohlgeformt und $\text{FSE}(x) = L_X\}$ die Menge der FSE-maximalen Individuen. Dann gilt für alle $x \in B$, dass*

$$\text{Prob}(x_M = x) \;=\; \frac{1}{|B|} \pm n^{-\Omega(n)}.$$

Beweis: Folgt wegen $\mu = \text{poly}(n)$ unmittelbar aus Lemma 6.2.1 und Lemma 6.2.2. □

6.3 Untere Schranken für kleine Populationen

In diesem Abschnitt wollen wir zeigen, dass die Beispielfunktion f aus Abschnitt 6.2.1 tatsächlich schwierig zu optimieren ist, wenn der Beispiel-GA nur kleine Populationen benutzt. Darüber hinaus werden wir beweisen, dass auch polynomiell viele parallele Läufe des Beispiel-GAs mit überwältigender Wahrscheinlichkeit nicht effizient sind.

Theorem 6.3.1 *Sei $\mu = O(1)$. Mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ ist die Laufzeit des Beispiel-GAs auf f dann mindestens $2^{\Omega(n^{1/2})}$.*

Natürlich impliziert Theorem 6.3.1 dann auch eine untere Schranke von $2^{\Omega(n^{1/2})}$ für die erwartete Laufzeit. Die grundlegende Beweisidee wird sein, dass der Beispiel-GA mit überwältigender Wahrscheinlichkeit ein Individuum mit einem PE-Wert von $2m/3 + b$ erzeugt, bevor jemals ℓ Einsen in irgendeinem Suffix generiert werden. Sei ein wohlgeformtes Individuum mit einem PE-Wert von $2m/3 + b$ als *präfixoptimal* bezeichnet. Wir wissen bereits, dass alle präfixoptimalen Individuen ähnliche Funktionswerte haben, ein präfixoptimales Individuum mit FSE-Wert ℓ lokal optimal ist und alle präfixoptimalen Individuen einen Hammingabstand von mindestens b zu jedem global optimalen Individuum besitzen. Dies legt nahe, dass sich an die Erzeugung eines präfixoptimalen Individuums eine lange Wartezeit anschließt, wenn zuvor kein global optimales Individuum gesehen wurde. Zum formalen Beweis werden wir die Beweismethode des typischen Laufs anwenden und zeigen, dass die Wahrscheinlichkeit, einen Lauf des Beispiel-GAs zu beobachten, der von noch zu nennenden Eigenschaften abweicht, exponentiell klein ist.

Der typische Lauf des Beispiel-GAs mit $\mu = O(1)$ auf f zerfällt in drei aufeinander folgende *Epochen.* Eine Epoche zeichnet sich dadurch aus, dass ihr Ende vom Auftreten eines speziellen Ereignisses vorgegeben wird. Die erste Epoche soll mit der Initialisierung beginnen und zum Zeitpunkt des ersten Schritts enden, der ein wohlgeformtes Individuum erzeugt. Die zweite Epoche soll in dem ersten Schritt enden, der ein präfixoptimales Individuum erzeugt. Die dritte Epoche schließlich soll im ersten Schritt enden, der zur Erzeugung eines global optimalen Individuums führt. Wenn bereits in der ersten oder zweiten Epoche das Optimum gefunden wird, definieren wir die Länge der verbleibenden Epochen als 0. Wir werden zeigen, dass allein die dritte Epoche mit überwältigender Wahrscheinlichkeit die genannten $2^{\Omega(n^{1/2})}$ Schritte in Anspruch nimmt.

Um den Beweis von Theorem 6.3.1 übersichtlich zu halten und in Abschnitt 6.4 wieder verwertbare Hilfsaussagen zu gewinnen, analysieren wir die drei Epochen in einzelnen Lemmata. Die darin auftretende Fehlerwahrscheinlichkeit der Form $2^{-2\ell/3+o(\ell)}$ kann als $2^{-2\ell/3+O(\log \ell)}$ gelesen werden; wir haben den etwas schlechteren, ersten Ausdruck zur Vereinfachung der Notation gewählt.

Lemma 6.3.2 *Sei $\mu = \mathrm{poly}(n)$ und sei x^* das erste wohlgeformte Individuum im Lauf des Beispiel-GAs auf f. Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-2\ell/3+o(\ell)}$ entsteht x^* nach höchstens $2e\ell n \ln(e\ell)$ Schritten, ist einziges wohlgeformtes Individuum der Population und genügt $\mathrm{PE}(x^*) \leq 7m/12$ und $\mathrm{FSE}(x^*) \leq 3\ell/4$. Außerdem ist die erwartete Zahl von Schritten bis zur Erzeugung von x^* höchstens $O(n \log n)$.*

Lemma 6.3.3 *Sei $\mu = \mathrm{poly}(n)$. Angenommen, die aktuelle Population des Beispiel-GAs auf f enthält mindestens ein wohlgeformtes Individuum und alle ihre wohlgeformten Individuen haben einen FSE-Wert von höchstens $3\ell/4$. Dann entsteht ein präfixoptimales Individuum, bevor das Optimum erreicht wird, mit einer Wahrscheinlichkeit von mindestens $\max\{2^{-O(\mu)} - 2^{-\Omega(n)}, 1 - 2^{-n^{1/2}/\exp(O(\mu))} - 2^{-\Omega(n^{1/2})}\}$.*

Lemma 6.3.4 *Sei $\mu = \mathrm{poly}(n)$. Angenommen, die aktuelle Population des Beispiel-GAs auf f enthält mindestens ein präfixoptimales und kein global optimales Individuum. Dann beträgt die Laufzeit mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ mindestens $2^{\Omega(n^{1/2})}$.*

Beweis von Lemma 6.3.2: Wir beginnen mit einer Analyse der initialen Population. Da es in wohlgeformten Individuen nur $\ell + 1$ verschiedene Belegungen für das Suffix gibt und die initiale Population zufällig gleichverteilt und unabhängig erzeugt wird, ist jedes initiale Individuum nur mit einer Wahrscheinlichkeit von höchstens $(\ell+1)2^{-\ell}$ wohlgeformt. Die Wahrscheinlichkeit für mindestens ein wohlgeformtes initiales Individuum ist höchstens $\mu(\ell + 1)2^{-\ell}$, was aufgrund von $\mu = \mathrm{poly}(n)$ höchstens $2^{-\ell+o(\ell)}$ ist. Wir nehmen im Folgenden an, dass der Beispiel-GA mit einer Population ohne wohlgeformte Individuen initialisiert, und erhalten unmittelbar die Aussage des Lemmas, dass x^* alleiniges wohlgeformtes Individuum seiner Population ist. Der Schritt, der zu seiner Erzeugung führt, sei als Ende der Epoche bezeichnet. Der f-Wert aller Individuen bis zum Ende der Epoche ist stets gegeben durch $g(x) := n^{1+\ell^2-\ell\sum_{i=m+1}^{n} x_i}$.

Das Ende der Epoche ist spätestens erreicht, wenn ein Individuum mit einem g-Wert von $n^{1+\ell^2}$ erzeugt wird, da dieser Wert ein Suffix der Form 0^ℓ impliziert. Wir werden zunächst die Zeit bis zum Erreichen eines g-Werts von $n^{1+\ell^2}$ durch Anwendung der Technik aus Lemma 5.2.1 nach oben abschätzen, obwohl wir insgesamt an einer unteren Schranke für die Laufzeit des Beispiel-GAs auf f interessiert sind. Die genannte obere Schranke soll helfen, die Wahrscheinlichkeit unerwünschter Ereignisse, die zwischenzeitlich auftreten können, klein zu halten. Wir werden daher mithilfe der oberen Schranke zeigen, dass mit überwältigender Wahrscheinlichkeit während der Zeit bis zur Erzeugung von x^* niemals ein FSE-Wert von mehr als $3\ell/4$ auftritt.

Sei nun $L(x) := \ell - \sum_{i=m+1}^{n} x_i$. Aus der Definition von g folgt, dass L monoton in Bezug auf g ist (vgl. Abschnitt 5.2.1). Betrachten wir also das L-Potenzial der aktuellen Population, d. h. den maximalen L-Wert ihrer Individuen. Ein Individuum mit L-Wert i enthält im Suffix $\ell - i$ Einsen. Die Wahrscheinlichkeit, genau eine dieser Einsen zu kippen, beträgt mindestens $((\ell-i)/n)(1-1/n)^{n-1} \geq (\ell-i)/(en)$. Darüber hinaus können wir die Wahrscheinlichkeit, ein Individuum zur Mutation zu wählen, dessen L-Wert nicht dem L-Potenzial entspricht, mit einer Technik ähnlich wie im Beweis von Lemma 6.2.2 nach oben abschätzen als

$$\mu \cdot \frac{n^{\ell\cdot(L-1)}}{n^{\ell\cdot L}} \;=\; n^{-\Omega(\ell)} \;=\; 2^{-\Omega(n^{1/2}\log n)}.$$

Das L-Potenzial kann höchstens ℓ-mal steigen. Folglich ist die erwartete Zahl von Schritten bis zum Erreichen eines L-Potenzials von ℓ und damit bis zum Ende der Epoche mithilfe von Lemma 5.2.1 und nicht zu kleine Werte von n nach oben beschränkt durch

$$\left(1+2^{-\Omega(n^{1/2}\log n)}\right)\sum_{i=0}^{\ell-1}\frac{en}{\ell-i} \;=\; \left(1+2^{-\Omega(n^{1/2}\log n)}\right)enH_\ell \;\leq\; en(\ln\ell+1) \;=\; en\ln(e\ell).$$

Die letzte Ungleichung gilt für genügend großes n, indem wir $H_\ell(1+2^{-\Omega(n^{1/2}\log n)})-\ln\ell$ anhand von Lemma A.6 durch 1 nach oben abschätzen. Die betrachtete Zeit ist nach der Markoffungleichung höchstens doppelt so groß mit einer Wahrscheinlichkeit von mindestens 1/2. Mit einer Wiederholung unabhängiger Phasen folgt insgesamt, dass die Zeit bis zur Erzeugung von x^* für genügend großes n mit einer Wahrscheinlichkeit von mindestens $1-2^{-\ell}$ durch $2e\ell n\ln(e\ell)$ nach oben beschränkt ist.

Wir müssen jetzt nur noch die Aussage über PE(x^*) und FSE(x^*) zeigen. Das Präfix eines initialen Individuums ist gleichverteilt über $\{0,1\}^m$. Weil die Belegung des Präfixes überhaupt nicht in den g-Wert eingeht, ist auch das Präfix des ersten wohlgeformten Individuums x^* gleichverteilt über $\{0,1\}^m$. Nach Anwendung der Chernoffungleichung folgt mithilfe der Beziehung $m = n - o(n)$ nun, dass das Präfix von x^* mit einer Wahrscheinlichkeit von sogar $1 - 2^{-\Omega(n)}$ höchstens $7m/12$ Einsen enthält.

Es bleibt die Aussage über $\mathrm{FSE}(x^*)$ zu zeigen. Zunächst halten wir fest, dass aufgrund von $\mu = \mathrm{poly}(n)$ mit einer Wahrscheinlichkeit von $1 - 2^{-2\ell/3+o(\ell)}$ kein initiales Individuum mehr als $2\ell/3$ führende Einsen im Suffix erhält, d. h. einen L-Wert von weniger als $\ell/3$ hat. Die Wahrscheinlichkeit, mindestens einmal innerhalb von höchstens $2\mathrm{e}\ell n \ln(\mathrm{e}\ell) = \mathrm{poly}(n)$ Schritten ein Individuum zu wählen, dessen L-Wert nicht dem L-Potenzial gleicht, ist immer noch $2^{-\Omega(n^{1/2} \log n)}$. Die Wahrscheinlichkeit, mindestens einmal in $\mathrm{poly}(n)$ Schritten mindestens $\ell/12$ Bits auf einmal zu kippen, ist ebenfalls $2^{-\Omega(n^{1/2} \log n)}$. Damit besitzt das erste wohlgeformte Individuum einen L-Wert von immer noch mindestens $\ell/3 - \ell/12 = \ell/4$, d. h. einen FSE-Wert von höchstens $3\ell/4$. Das Lemma folgt insgesamt, da die Summe aller Fehlerwahrscheinlichkeiten höchstens $2^{-2\ell/3+o(\ell)}$ beträgt. □

Beweis von Lemma 6.3.3: Der aktuelle Zeitpunkt definiere den Beginn einer Epoche, die zum Zeitpunkt des Schrittes endet, der das erste präfixoptimale Individuum erzeugt. Zunächst schätzen wir wieder die Wahrscheinlichkeiten einiger unerwünschter Ereignisse ab. Aufgrund des Elitismus wird vom Beginn der Epoche an stets ein wohlgeformtes Individuum in der Population sein. Damit ist die Wahrscheinlichkeit, ein nicht wohlgeformtes Individuum zur Mutation zu wählen, durch $\mu n^{1+\ell^2}/n^{2n} = n^{-\Omega(n)}$, da $\mu = \mathrm{poly}(n)$, nach oben beschränkt. Im Folgenden nehmen wir an, dass in einer Epoche polynomieller Länge niemals ein nicht wohlgeformtes Individuum zur Mutation gewählt wird und führen damit nur einen Fehlerterm von $n^{-\Omega(n)}$ ein.

Nun wollen wir die Epoche in eine noch zu definierende Anzahl $p(\mu)$ disjunkter Phasen noch zu definierender Länge $s(\mu)$ einteilen. Unser Ziel ist es, eine Phase zu finden, sodass zum einen zu Beginn der Phase das FSE-Potenzial, d. h. der maximale FSE-Wert der wohlgeformten Individuen der aktuellen Population, höchstens $\ell - 1$ beträgt und zum anderen während der Phase keine Mutation den FSE-Wert des ausgewählten Individuums erhöht. Sei eine solche Phase *gut* genannt und sei L ihr FSE-Potenzial. Es folgt nun, dass in guten Phasen die Funktion P mit

$$P(x) := \begin{cases} \mathrm{PE}^*(x), & \text{falls } x \text{ wohlgeformt und } (\mathrm{FSE}(x) = L \text{ oder } \mathrm{PE}(x) > 2m/3), \\ -1 & \text{sonst} \end{cases}$$

monoton in Bezug auf f ist. Hier ist zu beachten, dass wir auch den zweiten Fall der Definition von f einbeziehen. Sobald das P-Potenzial einer Population in einer guten Phase den Wert $2m/3 + b$ annimmt, ist ein präfixoptimales Individuum gefunden. Dies beendet die Phase eventuell sogar in weniger als $s(\mu)$ Schritten.

Um die Zeit bis zum Erreichen eines P-Potenzials von $2m/3 + b$ abzuschätzen, argumentieren wir ähnlich wie im Beweis von Lemma 5.2.1. Um das P-Potenzial der Population zu erhöhen, genügt in einer guten Phase Folgendes. Der Beispiel-GA wählt ein Individuum x mit aktuell maximalem f-Wert zur Mutation und fügt genau eine

Eins in seinem Präfix hinzu (wenn $\mathrm{PE}(x) < 2m/3 + b$) oder löscht genau eine Eins im Präfix (wenn $\mathrm{PE}(x) > 2m/3+b$). Im Fall $\mathrm{PE}(x) < 2m/3+b$ ist die Wahrscheinlichkeit für einen solchen Schritt durch

$$\frac{1}{\mu}\left(\frac{m}{3} - b\right)\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} \;\geq\; \frac{c}{\mu}$$

für eine Konstante $c > 0$ nach unten beschränkt, da $m/3 - b = \Omega(n)$ gilt. Im anderen Fall ist die Wahrscheinlichkeit mindestens ebenso groß, da der Beispiel-GA dann unter mindestens $2m/3 + b$ Einsen im Präfix wählen kann. Höchstens $2m/3 + b$ Erhöhungen des P-Potenzials um 1 sind hinreichend, um den gewünschten Wert zu erreichen. Mit der Chernoffungleichung folgt, dass in einer guten Phase der Länge $s(\mu) := \lceil \mu m/c \rceil$ mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ der gewünschte Wert erreicht und somit ein präfixoptimales Individuum erzeugt wird.

Es gilt nun zu zeigen, dass es mit genügend großer Wahrscheinlichkeit mindestens eine gute Phase gibt. Dazu heiße ein Schritt *schlecht*, wenn in ihm das spezielle Bit kippt, das zur Erhöhung der Zahl der führenden Suffixeinsen führt. Eine Phase ist gut, wenn zu ihrem Beginn das FSE-Potenzial kleiner als ℓ ist und sie keine schlechten Schritte enthält. Die Wahrscheinlichkeit für Letzteres ist mindestens $(1 - 1/n)^{s(\mu)}$, da die Wahrscheinlichkeit, ein spezielles Bit zu kippen, $1/n$ beträgt. Der letzte Ausdruck kann wegen $m \leq n - n^{1/2}/45$ durch

$$\left(1 - \frac{1}{n}\right)^{\mu(n-1)/c} \;\geq\; \mathrm{e}^{-\mu/c}$$

nach unten abgeschätzt werden. Da nach Voraussetzung alle wohlgeformten Individuen zu Beginn der Epoche einen FSE-Wert von höchstens $3\ell/4$ haben und alle übrigen Fehlerwahrscheinlichkeiten $2^{-\Omega(n)}$ sind, folgt mit der Betrachtung dieser einen guten Phase bereits das Lemma für den Fall einer Erfolgswahrscheinlichkeit von $2^{-O(\mu)} - 2^{-\Omega(n)}$.

Es bleibt nun zu zeigen, dass auch die untere Schranke $1 - 2^{-n^{1/2}/\exp(O(\mu))} - 2^{-\Omega(n^{1/2})}$ für die Erfolgswahrscheinlichkeit zutrifft. Dazu setzen wir $p(\mu) := \lceil \ell c/(8\mu) \rceil$. Die Wahrscheinlichkeit, $p(\mu)$ unabhängige, aufeinander folgende Phasen der Länge $s(\mu)$ zu beobachten, sodass alle Phasen mindestens einen schlechten Schritt enthalten, ist nach oben beschränkt durch

$$\left(1 - \frac{1}{\mathrm{e}^{\mu/c}}\right)^{\ell c/(8\mu)} \;=\; \left(1 - \frac{1}{\mathrm{e}^{\mu/c}}\right)^{\exp(\mu/c)\cdot \ell c/(8\mu \exp(\mu/c))} \;=\; 2^{-n^{1/2}/\exp(O(\mu))}.$$

Andererseits enthalten $p(\mu) - 1$ Phasen der Länge $s(\mu)$ im Erwartungswert insgesamt höchstens

$$\frac{\ell c}{8\mu} \cdot \left\lceil \frac{\mu m}{c} \right\rceil \cdot \frac{1}{n} \;\leq\; \frac{\ell}{8}$$

schlechte Schritte. Aufgrund der Unabhängigkeit der Schritte und mithilfe der Chernoffungleichung erhalten wir, dass in $p(\mu)$ Phasen, darunter mindestens eine ohne schlechte Schritte, mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ insgesamt höchstens $\ell/6$ schlechte Schritte auftreten. Wie auf Seite 21 können wir Zufallsvariablen $X_{i,j}$ für die Bernoulliversuche, dass der i-te schlechte Schritt das j-te Bit kippt, einführen. Davon sind diejenigen Zufallsvariablen, die sich auf die Bits rechts der linkesten Null im Suffix im jeweiligen Schritt beziehen, unabhängig. Es kippen in $\ell/6$ schlechten Schritten im Erwartungswert auch nur höchstens $\ell(\ell-1)/(6n)$ Bits an anderen Stellen außer der kippenden linkesten Null im Suffix. Mit der Chernoffungleichung erhalten wir schließlich, dass in $\ell/6$ schlechten Schritten mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ insgesamt höchstens $\ell/30$ solche Bits kippen. Unter dieser Voraussetzung wird das FSE-Potenzial in den $p(\mu)$ Phasen höchstens $3\ell/4+\ell/6+\ell/30<\ell$. Durch Aufsummieren der Fehlerwahrscheinlichkeiten ergibt sich, dass die Wahrscheinlichkeit, keine gute Phase in den $p(\mu)$ Phasen zu haben, insgesamt höchstens $2^{-n^{1/2}/\exp(O(\mu))}+2^{-\Omega(n^{1/2})}$ ist. Da eine gute Phase mit Wahrscheinlichkeit $1-2^{-\Omega(n)}$ ein präfixoptimales Individuum erzeugt, ist auch die zweite Schranke für die Erfolgswahrscheinlichkeit gezeigt. □

Beweis von Lemma 6.3.4: Aufgrund des Elitismus implizieren die Voraussetzungen des Lemmas, dass auch vor dem aktuellen Zeitpunkt das Optimum nicht erreicht worden sein kann. Wir bezeichnen die Zeit vom aktuellen Zeitpunkt bis zum ersten Schritt, der ein optimales Individuum erzeugt, wieder als Epoche. Die Definition von f und der Elitismus erzwingen, dass während der gesamten Epoche mindestens ein präfixoptimales Individuum in der Population vorhanden ist.

Die Beweisidee ist es nun zu zeigen, dass Schritte, die ein global optimales Individuum erzeugen können, aus verschiedenen Gründen unwahrscheinlich sind. Zunächst ist die Wahrscheinlichkeit, mindestens $b/2$ Bits in einem einzigen Schritt zu kippen, mithilfe der stirlingschen Formel durch

$$\binom{n}{b/2}\left(\frac{1}{n}\right)^{b/2} \leq \frac{1}{(b/2)!} = 2^{-\Omega(b\log b)} = 2^{-\Omega\left((n^{1/2}/\log n)\log(n^{1/2}/\log n)\right)} = 2^{-\Omega(n^{1/2})}$$

nach oben beschränkt. Wir betrachten jetzt eine Phase von $s:=\lfloor 2^{\varepsilon n^{1/2}}\rfloor$ Schritten für ein noch zu wählendes $\varepsilon>0$.

Wir nehmen nun an, dass ein Schritt höchstens $b/2$ Bits kippt. Um ein global optimales Individuum in einem solchen Schritt zu erzeugen, ist dann aber notwendig, ein Individuum mit einem PE-Wert von höchstens $2m/3+b/2$ zu mutieren. Die Wahrscheinlichkeit, ein solches Individuum zur Mutation zu wählen, ist jedoch wegen der Existenz eines präfixoptimalen Individuums und $\mu=\text{poly}(n)$ durch

$$\mu\cdot\frac{\ell+n^{2n\ell+2m/3+b/2}}{n^{2n\ell+2m/3+b}} = n^{-\Omega(b)} = 2^{-\Omega(n^{1/2})}$$

nach oben beschränkt. Insgesamt ist die Wahrscheinlichkeit, innerhalb von $\lfloor 2^{\varepsilon n^{1/2}}\rfloor$ Schritten mindestens einmal ein Ereignis zu beobachten, das zur Erzeugung eines optimalen Individuums führt, damit durch $2^{\varepsilon n^{1/2}}\cdot 2^{-\Omega(n^{1/2})}$ nach oben beschränkt. Wenn

ε eine genügend kleine Konstante ist, beträgt dieser Ausdruck immer noch $2^{-\Omega(n^{1/2})}$. Also beträgt die Länge der Epoche mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ mindestens $2^{\Omega(n^{1/2})}$. □

Nun folgt der Beweis von Theorem 6.3.1 fast unmittelbar.

Beweis von Theorem 6.3.1: Wegen $\mu = \text{poly}(n)$ dürfen wir die Lemmata 6.3.2–6.3.4 anwenden. Wir nehmen daher an, dass das erste im Lauf des Beispiel-GAs auf f erzeugte wohlgeformte Individuum den Aussagen von Lemma 6.3.2 genügt (die Aussage über $\text{PE}(x^*)$ brauchen wir noch nicht einmal) und führen damit einen Fehlerterm von höchstens $2^{-\Omega(n^{1/2})}$ ein. Gemäß unserer Annahme gelten die Voraussetzungen von Lemma 6.3.3. Da wir sogar $\mu = O(1)$ gegeben haben, führen wir wiederum nur einen Fehlerterm von höchstens $2^{-\Omega(n^{1/2})}$ ein, indem wir annehmen, dass die dort beschriebenen Ereignisse eintreten. Damit sind die Voraussetzungen von Lemma 6.3.4 erfüllt. Das Theorem folgt, da die Summe aller Fehlerwahrscheinlichkeiten $2^{-\Omega(n^{1/2})}$ ist. □

Theorem 6.3.1 sagt uns, dass der Beispiel-GA mit konstanten Populationsgrößen auf f extrem ineffizient ist. Da $\mu = 1$ zugelassen ist, gilt die Aussage auch für den (1+1)-EA. Da die Erfolgswahrscheinlichkeit in polynomiell vielen Schritten exponentiell klein ist, erhalten wir auch eine Folgerung für parallele Läufe des Beispiel-GAs mit $\mu = O(1)$ sowie des (1+1)-EA. Sofern die Anzahl der parallelen Läufe nämlich polynomiell beschränkt ist, beträgt die Wahrscheinlichkeit, dass mindestens ein Lauf in polynomiell vielen Schritten erfolgreich ist, immer noch $2^{-\Omega(n^{1/2})}$. Damit sind auch parallele Läufe höchst ineffizient.

Der Fall einer nicht mit der Problemgröße wachsenden Population ist in der Praxis vielleicht weniger üblich. Doch selbst für manche Werte mit $\mu = \omega(1)$ gelten die Aussagen des vorangehenden Absatzes zumindest qualitativ weiterhin.

Theorem 6.3.5 *Es gibt eine Konstante $c > 0$, sodass für $\mu \le c \ln n$ folgende Aussage gilt: Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n^{1/4})}$ ist die Laufzeit des Beispiel-GAs auf f mindestens $2^{\Omega(n^{1/2})}$.*

Beweis: Die Voraussetzungen des Theorems implizieren $\mu = \text{poly}(n)$. Wir wenden nun bis auf eine Ausnahme dieselben Argumente wie im Beweis von Theorem 6.3.1 an. Die Ausnahme besteht darin, dass die Erfolgswahrscheinlichkeit bei der Anwendung von Lemma 6.3.3 anders abgeschätzt werden muss. Es gilt $e^{O(\mu)} \le n^{1/4}$, falls $\mu \le c \ln n$ für eine genügend kleine Konstante $c > 0$ gilt und n groß genug ist. Damit folgt die untere Schranke $1 - 2^{-\Omega(n^{1/4})}$ für die Wahrscheinlichkeit einer Laufzeit von $2^{\Omega(n^{1/2})}$. □

Für noch größere Werte von μ können wir nicht mehr zeigen, dass die Erfolgswahrscheinlichkeit in polynomiell vielen Schritten noch exponentiell klein ist. Allerdings gibt es Werte von μ mit $\mu = \Omega(n^{1/2})$, sodass die erwartete Laufzeit des Beispiel-GAs auf f immer noch exponentiell ist.

Theorem 6.3.6 *Es gibt eine Konstante $c > 0$, sodass für $\mu \leq cn^{1/2}$ die erwartete Laufzeit des Beispiel-GAs auf f mindestens $2^{\Omega(n^{1/2})}$ ist.*

Beweis: Die Voraussetzungen des Theorems implizieren $\mu = \text{poly}(n)$. Wir wenden wiederum bis auf eine Ausnahme dieselben Argumente wie im Beweis von Theorem 6.3.1 an. Bei der Anwendung von Lemma 6.3.3 benutzen wir diesmal die untere Schranke von $2^{-O(\mu)} - 2^{-\Omega(n^{1/2})}$ für die Erfolgswahrscheinlichkeit. Insgesamt folgt dann, dass die Laufzeit mit einer Wahrscheinlichkeit von immer noch mindestens $2^{-O(\mu)} - 2^{-\Omega(n^{1/2})}$ mindestens $2^{\Omega(n^{1/2})}$ beträgt. Da $\mu \leq cn^{1/2}$, lässt sich einerseits der erste Ausdruck für eine Konstante $c_1 > 0$ gemäß $2^{-c_1 cn^{1/2}}$ nach unten abschätzen, sofern c klein genug und n groß genug ist. Für eine genügend kleine Konstante $c_2 > 0$ können wir andererseits den zweiten Ausdruck durch $2^{c_2 n^{1/2}}$ nach unten abschätzen. Wir betrachten im Produkt der Schranken der beiden Ausdrücke den Exponenten $-cc_1 n^{1/2} + c_2 n^{1/2}$. Wenn c klein genug gewählt wird, gilt $-cc_1 n^{1/2} + c_2 n^{1/2} = \Omega(n^{1/2})$. Das Theorem folgt dann durch eine Anwendung des Satzes von der totalen Wahrscheinlichkeit auf die erwartete Laufzeit. □

Im folgenden Abschnitt werden wir sehen, dass für manche Werte μ mit $\mu = O(n^{1/2})$ die erwartete Laufzeit bereits polynomiell ist. Dies impliziert, dass die kritische Populationsgröße, bei der die erwartete Laufzeit von einem exponentiellen auf einen polynomiellen Wert umschlägt, asymptotisch scharf ermittelt wurde. Eine solche Aussage bezeichnet man zuweilen als *scharfes Threshold-Resultat.*

6.4 Obere Schranken für große Populationen

In diesem Abschnitt wollen wir beweisen, dass die erwartete Laufzeit des Beispiel-GAs auf f tatsächlich bei Verwendung einer genügend großen Population nur polynomiell groß ist. Dazu formulieren wir das zugehörige Theorem vorab und erarbeiten uns sodann dessen Beweisidee und schließlich den formalen Beweis.

6.4.1 Die obere Schranke für die Laufzeit

Theorem 6.4.1 *Sei $\mu \geq 45n^{1/2}$ und $\mu = \text{poly}(n)$. Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n^{1/2})}$ ist die Laufzeit des Beispiel-GAs auf f dann höchstens $5e\ell n \ln(e\ell) + 4\mu n$. Die erwartete Laufzeit ist ebenfalls $O(n^{3/2} \log n + \mu n)$.*

In Verbindung mit Theorem 6.3.6 erhalten wir aus Theorem 6.4.1, dass die Wahl eines μ mit $\mu \geq 45n^{1/2}$ und $\mu = O(n^{1/2})$ asymptotisch optimal sein könnte. Die erwartete Laufzeit des Beispiel-GAs kann für $\mu = \lceil 45n^{1/2} \rceil$ durch den in einem kleinen Polynom beschränkten Ausdruck $(5e/45 + o(1))n^{3/2} \ln(e\ell)$ nach oben abgeschätzt werden. Auf der anderen Seite erhalten wir jedenfalls bereits für $\mu = \omega(n^{3/2} \log n)$, dass die Initialisierungskosten des Beispiel-GAs die vorgenannte obere Schranke übersteigen.

Im Folgenden werden wir diskutieren, warum Theorem 6.4.1 gilt. Aus der Diskussion der Idee der Funktion f, beschrieben in Abschnitt 6.2.1, und dem formalen Beweis von Theorem 6.3.1 wissen wir bereits, dass der Beispiel-GA mit hoher Wahrscheinlichkeit in der Umgebung eines lokalen Optimums hängen bleibt, wenn sich in einem Individuum der PE-Wert auf über $2m/3$ erhöht, bevor jemals ein FSE-Wert von ℓ erreicht wurde. Also scheint es hilfreich zu sein, dass der FSE-Wert in einem Individuum auf ℓ steigt, ohne dass jemals ein größerer PE-Wert als $2m/3$ gesehen wurde.

Warum sollte eine große Population das zuletzt beschriebene Verhalten erleichtern? Um dies zu erläutern, greifen wir auf die Lemmata 6.2.1 und 6.2.2 zurück. Solange die aktuelle Population des Beispiel-GAs normal ist, können wir also fast mit Sicherheit davon ausgehen, dass der nächste Schritt ein FSE-maximales Individuum zur Mutation wählt. Darüber hinaus ermöglicht die $1/n$-Standardmutation bekanntlich die Bildung eines Replikats, nämlich mit einer konstanten Wahrscheinlichkeit von mindestens $1/(2e)$ (vgl. Seite 20). Dieses Replikat verdrängt dann gemäß Lemma 6.2.2 fast mit Sicherheit ein Individuum mit geringerem FSE-Wert, es sei denn, alle Individuen der aktuellen Population besitzen (bereits) denselben FSE-Wert.

Solange das FSE-Potenzial einer normalen Population unverändert bleibt (es kann aufgrund des Elitismus nicht sinken), scheint sich also die Zahl der FSE-maximalen Individuen schnell zu erhöhen. Unter solchen Individuen wählt der Beispiel-GA laut Lemma 6.2.1 nahezu gleichverteilt zur Mutation aus. Nehmen wir zunächst vereinfachend an, dass in der aktuellen Population für lange Zeit alle Individuen denselben FSE-Wert aufweisen. Mutationen, die den PE-Wert erhöhen, „verteilen" sich dann auf μ Individuen. Mit hoher Wahrscheinlichkeit muss aufgrund der gleichverteilten Initialisierung in allen Individuen der PE-Wert um $\Omega(n)$ erhöht werden, ehe er $2m/3$ erreicht. Um in einem Individuum auf einen PE-Wert von $2m/3$ zu kommen, sind mit hoher Wahrscheinlichkeit $\Omega(n)$ Mutationen notwendig. Wenn sich Mutationen, die den PE-Wert erhöhen, gleichmäßig auf μ Individuen verteilen, könnte sich die für einen PE-Wert von mindestens $2m/3$ erforderliche erwartete Zeit t_1 auf $\Omega(\mu n)$ erhöhen. Andererseits bleibt die erwartete Zeit bis zum Erreichen eines optimalen FSE-Wertes von der Populationsgröße nahezu unbeeinflusst, da immer Individuen mit maximalem FSE-Wert zur Mutation gewählt werden. Da die Wahrscheinlichkeit, den FSE-Wert zu erhöhen, mindestens $\Omega(1/n)$ beträgt, scheint eine erwartete Zeit t_2 von $O(\ell n) = O(n^{3/2})$ zu genügen, um auf einen FSE-Wert von ℓ zu kommen. Für genügend großes, aber immer noch durch $O(n^{1/2})$ beschränktes μ könnte die untere Schranke für t_1 größer werden als die obere Schranke für t_2. Somit könnte die Wahrscheinlichkeit, einen FSE-Wert von ℓ zu erreichen, bevor jemals ein PE-Wert von mindestens $2m/3$ gesehen wurde, dann sehr groß werden.

Die bislang beschriebene Idee hat zunächst einen entscheidenden Nachteil. Selbst wenn der Beispiel-GA für lange Zeit (nahezu) gleichverteilt aus der Population zur Mutation wählt, kann man nicht so tun, als würden sich Mutationen so auf μ unabhängige Individuen verteilen, als ob diese stets verschiedene Vorfahren in der initialen Population hätten. Damit entspräche der Beispiel-GA eher μ parallelen Läufen des

(1+1)-EA. Tatsächlich ist es aber beispielsweise nicht allzu unwahrscheinlich, dass ein zur Zeit t und ein zur Zeit $t+1$ erzeugtes Individuum einen gemeinsamen Elter besitzen. Die vielleicht überraschende Erkenntnis wird im Folgenden sein, dass für den Fall, in dem der Beispiel-GA (nahezu) gleichverteilt zur Mutation wählt, das Bild der parallelen Läufe nichtsdestoweniger asymptotisch korrekte Werte für Laufzeiten liefern kann. Diese Beziehung wird uns in Abschnitt 7.4.1 bei der Analyse des (μ+1)-EA wieder begegnen.

Um nachzuweisen, dass der maximale PE-Wert der Population bei großem μ in der Tat langsamer wächst als bei kleinem μ, werden wir die in Abschnitt 5.2.2 beschriebene Technik der Analyse von Stammbäumen anwenden. Es sollte nun nicht mehr überraschen, dass diese Technik, die als Werkzeug zum Nachweis von unteren Schranken für Laufzeiten vorgestellt wurde, im Beweis einer oberen Schranke für die Laufzeit des Beispiel-GAs Anwendung findet. Die wesentliche Idee bei der Definition der Funktion f bestand schließlich darin, dass eine „Komponente“ der Funktion, nämlich der PE-Wert, bei großer Population langsamer wächst. Bevor wir aber in die Analyse des Wachstums von Stammbäumen einsteigen können, greifen wir auf Lemma 6.3.2 zurück. Wir erhalten dann, dass es mit überwältigender Wahrscheinlichkeit vor dem Erreichen des Optimums eine erste normale Population gibt, die nur aus einem wohlgeformten Individuum x mit $\mathrm{PE}(x) \leq 7m/12$, also $\mathrm{PE}(x) = 2m/3 - \Omega(n)$, besteht. Sei angenommen, dass dies eintritt. Somit ziehen wir wieder die Beweismethode des typischen Laufes heran.

Zur Vereinfachung der Notation sei 0 derjenige Zeitpunkt, zu dem x erstmalig in der Population ist. In einer mit dem Zeitpunkt 0 beginnenden Phase noch festzulegender Länge s wollen wir das Wachstum des Stammbaums $T(x) := (T_t(x))_{t\geq 0}$ analysieren und dabei Bezüge zu bereits bekannten Modellen zufälliger Bäume aufstellen, um die Tiefe zum Zeitpunkt s nach oben zu beschränken. Das FSE-Potenzial der Population zum Zeitpunkt 0 ist gerade $\mathrm{FSE}(x)$ und die Anzahl FSE-maximaler Individuen ist 1. Mit überwältigender Wahrscheinlichkeit wählt der Schritt zum Zeitpunkt 0 also gemäß Lemma 6.2.2 das Individuum x zur Mutation. Wir nehmen bis zum Ende dieses Absatzes zunächst vereinfachend an, dass alle (auch vergrößerten) Populationen normal sind. Sei nun x' der Mutant von x. Jetzt interessiert hauptsächlich das Verhältnis von $\mathrm{FSE}(x')$ zu $\mathrm{FSE}(x)$. Falls $\mathrm{FSE}(x') < \mathrm{FSE}(x)$, wird x' im nachfolgenden Schritt und auch in polynomiell vielen Schritten laut Lemma 6.2.3 mit überwältigender Wahrscheinlichkeit nie zur Mutation ausgewählt werden. Das Gleiche gilt, falls x' nicht wohlgeformt ist. In den letztgenannten beiden Fällen bezeichnen wir den Knoten im Stammbaum, der x' enthält, und den Schritt, der x' erzeugt, dann jeweils als *irrelevant*. Die übrigen Knoten heißen relevant. Sei nun angenommen, dass x' wohlgeformt ist. Wenn $\mathrm{FSE}(x') = \mathrm{FSE}(x)$ gilt, liefert uns Lemma 6.2.2, dass x' im betrachteten Schritt mit überwältigender Wahrscheinlichkeit nicht gelöscht wird. In der nächsten Population ist die Anzahl FSE-maximaler Individuen also um 1 angewachsen und der kommende Schritt wählt gemäß Lemma 6.2.3 jedes der beiden Individuen x und x' mit einer Wahrscheinlichkeit von mindestens $1/2 - n^{-\Omega(n)}$. Wenn auch der darauf folgende

Schritt ein wohlgeformtes Individuum x'' mit $\mathrm{FSE}(x'') = \mathrm{FSE}(x)$ erzeugt, wird in der nachfolgenden Population jedes Individuum aus $\{x, x', x''\}$ mit einer Wahrscheinlichkeit von mindestens $1/3 - n^{-\Omega(n)}$ gewählt usw. (Hier nehmen wir natürlich $\mu \geq 3$ an.) Den Fall $\mathrm{FSE}(x') > \mathrm{FSE}(x)$ werden wir später behandeln.

Zusammengefasst haben wir motiviert, wieso wir für die nächsten $\mu - 1$ Schritte den Prozess, der die relevanten Knoten von $T(x)$ erzeugt, unter der Voraussetzung, dass sich das FSE-Potenzial nicht erhöht und kein Individuum einen größeren PE-Wert als $2m/3$ hat, wie folgt modellieren wollen.

Definition 6.4.2 (zufälliger rekursiver Baum (ZRB)) *Ein* ZRB T_0 *zum Zeitpunkt* 0 *besteht nur aus seiner Wurzel. Ein ZRB* T_t *zum Zeitpunkt* $t > 0$ *entsteht aus dem ZRB* T_{t-1}*, indem zufällig gleichverteilt einer seiner Knoten gewählt und ein neues Blatt angehängt wird.*

Wir beachten, dass ein ZRB zum Zeitpunkt t genau $t + 1$ Knoten enthält. Die englischsprachige Bezeichnung für einen ZRB nach Definition 6.4.2 lautet *uniform random recursive tree*, siehe z. B. Pittel (1994) sowie Smythe und Mahmoud (1995). In unserer Übersetzung verzichten wir auf das Wort *uniform*, da die in Definition 6.4.2 geforderte gleichverteilte Wahl eines Elterknotens eine natürliche Entscheidung darstellt. Außerdem weisen wir darauf hin, dass wir mit dem Begriff eines ZRBs eigentlich einen stochastischen (Markoff)prozess definieren. Allerdings sollten keine Missverständnisse auftreten, wenn man von einem ZRB zum Zeitpunkt t spricht und eigentlich den Zustand des Prozesses zu diesem Zeitpunkt meint.

Wie der Übersichtsartikel von Smythe und Mahmoud (1995) belegt, sind bereits mannigfache Eigenschaften von ZRBs untersucht worden. Besonders interessant ist dabei natürlich die erwartete Tiefe zum Zeitpunkt t, die genau $\mathrm{e} \ln(t + 1)$ beträgt. Wir sehen uns aber dem Problem gegenüber, dass die enge Beziehung zwischen dem Prozess, der relevante Knoten des Stammbaums $T(x)$ erzeugt, und dem Prozess eines ZRBs verschwindet, sobald alle Individuen der Population denselben FSE-Wert haben. Bis zum Ende dieses Unterabschnitts betrachten wir wiederum nur normale (vergrößerte) Populationen. Mit Lemma 6.2.3 folgt dann, dass der nächste Schritt jedes Individuum mit einer Wahrscheinlichkeit von $1/\mu \pm n^{-\Omega(n)}$ wählt. Falls die zugehörige Mutation ein wohlgeformtes Individuum mit unverändertem FSE-Wert erzeugt, wird eines der temporär $\mu + 1$ Individuen gelöscht. Wir können nicht voraussagen, welcher Knoten im zugehörigen Stammbaum dann zu einem toten Knoten wird. Solange sich das FSE-Potenzial nicht erhöht, wissen wir für die nachfolgenden Schritte jedoch, dass die Anzahl der FSE-maximalen Individuen mit überwältigender Wahrscheinlichkeit bei μ bleibt und jedes Individuum mit einer Wahrscheinlichkeit von $1/\mu \pm n^{-\Omega(n)}$ zur Mutation gewählt wird. Daher definieren wir eine Verallgemeinerung von ZRBs, die in den ersten $u - 1$ Schritten wie ZRBs funktionieren und dann auf Auswahlwahrscheinlichkeiten von $1/u$ für einen Parameter u, der später eine untere Schranke für μ sein wird, umschalten. Dabei verfälschen wir die Wahrscheinlichkeiten also nur um einen additiven Term in der Größenordnung $n^{-\Omega(n)}$, was wir anschließend korrigieren werden. Der nachfolgend auftretende Begriff einer Trägermenge ist in Definition A.9 erläutert.

Definition 6.4.3 (p-zufälliger und $1/u$-Baum) *Sei $p := (p_t)_{t\geq 0}$ eine Folge von Wahrscheinlichkeitsverteilungen, sodass die Trägermenge von p_t eine Teilmenge von $\{0,\ldots,t\}$ ist. Ein p-*zufälliger Baum* T_0 zum Zeitpunkt 0 besteht nur aus seiner Wurzel. Ein p-zufälliger Baum T_t zum Zeitpunkt $t > 0$ entsteht aus dem p-zufälligen Baum T_{t-1}, indem t^* gemäß der Verteilung p_{t-1} gezogen und ein neues Blatt an den zum Zeitpunkt t^* eingefügten Knoten angehängt wird.*

Ein p-zufälliger Baum heißt $1/u$-Baum*, falls es ein $u \in \mathbb{N}$ gibt, sodass p_t jedem Element aus $\{0,\ldots,t\}$ eine Wahrscheinlichkeit von höchstens $1/(t+1)$, falls $t \leq u-1$, und andernfalls von höchstens $1/u$ zuordnet.*

Wiederum enthält ein Baum zum Zeitpunkt t genau $t+1$ Knoten. Mit $1/u$-Bäumen bezeichnen wir nun aber eine ganze Klasse von stochastischen Prozessen. Anstelle von $1/u$-Bäumen betrachten wir auch noch so genannte *Beinahe-$1/u$-Bäume* . Diese sind analog zu Definition 6.4.3 definiert, nur sind im letzten Satz statt $1/(t+1)$ und $1/u$ die Schranken $1/(t+1)+n^{-\Omega(n)}$ bzw. $1/u+n^{-\Omega(n)}$ einzusetzen. Der Parameter n verkörpert immer die Dimension des Suchraums einer gerade untersuchten Zielfunktion.

Die bisherigen Überlegungen nutzen nicht aus, dass x das einzig wohlgeformte Individuum zum Zeitpunkt 0 ist, sondern nur, dass die Anzahl FSE-maximaler Individuen zum Zeitpunkt 0 eins ist. Also haben wir die folgende Beziehung bewiesen.

Lemma 6.4.4 *Sei X eine normale Population mit genau einem FSE-maximalen Individuum x. Es gilt mit einer Wahrscheinlichkeit von $1-n^{-\Omega(n)}$: Solange der Beispiel-GA auf f nur normale Populationen erzeugt und sich das FSE-Potenzial nicht erhöht, ist der Prozess, der die relevanten Knoten des Stammbaums $T(x)$ erzeugt, für jedes $u \leq \mu$ der Prozess eines Beinahe-$1/u$-Baums.*

Die in Lemma 6.4.4 auftretende Fehlerwahrscheinlichkeit der Größenordnung $n^{-\Omega(n)}$ bezieht sich darauf, dass ein Schritt ein Individuum wählt, das nicht FSE-maximal ist. Weil wir ein solches Ereignis ignorieren möchten, führen wir einen weiteren Begriff, der sich auf ganze Zeitabschnitte bezieht, ein.

Definition 6.4.5 *Sei eine Phase von s Schritten des Beispiel-GAs auf der Funktion f betrachtet. Die Phase heißt* normal*, wenn sie die folgenden Bedingungen erfüllt:*

1. *Zu Beginn der Phase liegt eine normale Population vor, die genau ein FSE-maximales Individuum enthält.*

2. *In allen Schritten der Phase liegen normale Populationen vor.*

3. *Wenn die Zahl der FSE-maximalen Individuen in einem Schritt der Phase kleiner als μ ist, löscht der Schritt kein FSE-maximales Individuum.*

Eine normale Phase beginnt also mit genau einem FSE-maximalen Individuum. Wir betrachten zunächst den Fall, dass der Beispiel-GA in einer normalen Phase das FSE-Potenzial unverändert lässt. Sei $u := \mu$. Dann erzeugt der Beispiel-GA nach unseren Annahmen in den folgenden $u-1$ relevanten Schritten einen Beinahe-$1/u$-Baum mit u Knoten, der fast wie ein ZRB funktioniert, und zugleich eine Population, die nur aus FSE-maximalen Individuen besteht. Wir sagen dazu auch, dass der Beispiel-GA den Beinahe-$1/u$-Baum *einschwingt*. Dies kann bei großen Werten von u die gesamte normale Phase in Anspruch nehmen. Gegebenenfalls wählt er anschließend bis zum Ende der Phase beinahe gleichverteilt zur Mutation und damit jeden Knoten im bereits vorhandenen Beinahe-$1/u$-Baum mit einer Wahrscheinlichkeit von höchstens $1/u + n^{-\Omega(n)}$.

Wir kommen schließlich zu dem Fall, dass ein relevanter Schritt einer normalen Phase den FSE-Wert eines FSE-maximalen Individuums erhöht. Aufgrund des Elitismus erhöht dieser Schritt dann mit Sicherheit das FSE-Potenzial. Sei t^* der nachfolgende Zeitpunkt und x^* das neue FSE-maximale Individuum. Bis zur nächsten Erhöhung des FSE-Potenzials und unter der Voraussetzung, in einer normalen Phase zu sein, gilt für die Schritte ab t^* dasselbe wie das oben für die Schritte ab 0 Diskutierte. Insbesondere kann der relevante Teil des Stammbaums von x^* gemäß Lemma 6.4.4 modelliert werden. Ein Blatt im Baum $T_{t^*}(x)$ wird also zur Wurzel eines neuen Beinahe-$1/u$-Baums. Bei der nächsten Erhöhung des FSE-Potenzials wiederholt sich dieses Phänomen. Der relevante Teil des Stammbaums $T_s(x)$ am Ende der oben erwähnten Phase der Länge s ergibt sich (wenn die Phase normal ist) also, indem unabhängige Beinahe-$1/u$-Bäume aneinander gehängt werden, vgl. Abbildung 6.1. Wir nehmen pessimistisch an, dass

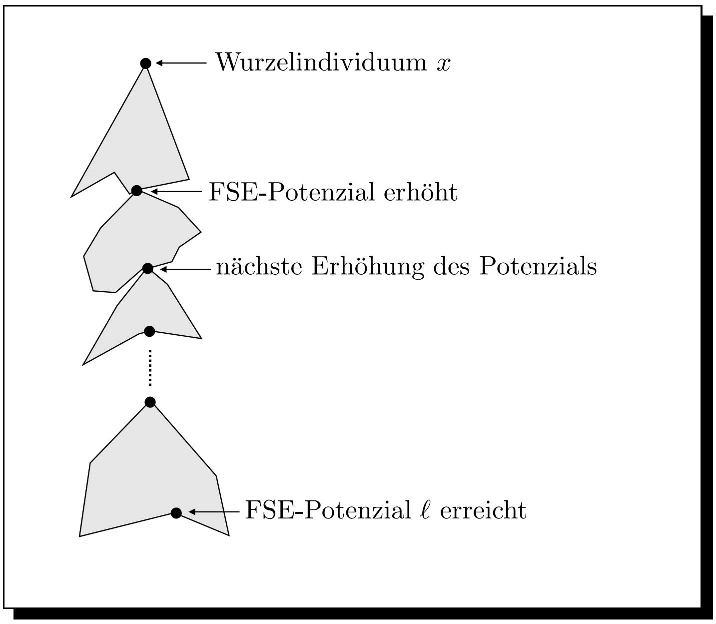

Abbildung 6.1: Stammbaum als Folge von Bäumen

die Wurzel des nächsten Beinahe-$1/u$-Baums ein tiefstes Blatt im vorherigen Beinahe-$1/u$-Baum ist. Letztendlich sind wir daher an der Summe der Tiefen von Beinahe-$1/u$-Bäumen interessiert. Die Zahl dieser Bäume wird höchstens ℓ betragen, da sich das FSE-Potenzial nicht öfter erhöhen kann.

6.4.2 Beweis der oberen Schranke

Wie im vorigen Unterabschnitt motiviert, betrachten wir die Summe der Tiefen mehrerer Beinahe-$1/u$-Bäume. Um nicht immer mit den exponentiell kleinen Abweichungen der Form $n^{-\Omega(n)}$ bei den Auswahlwahrscheinlichkeiten in Beinahe-$1/u$-Bäumen arbeiten zu müssen, führen wir im Folgenden einige Rechnungen für $1/u$-Bäume anstelle von Beinahe-$1/u$-Bäumen durch. Weil wir nur Werte von u mit $u = \text{poly}(n)$ betrachten werden, führen wir in polynomiell vielen Schritten damit bei allen Aussagen über Beinahe-$1/u$-Bäume nur eine Fehlerwahrscheinlichkeit der Größenordnung $n^{-\Omega(n)}$ ein. Wir können die Auswahlen in Beinahe-$1/u$-Bäumen (wegen $u = \text{poly}(n)$) nämlich modellieren als Auswahlen, die zunächst die Forderungen für $1/u$-Bäume einhalten und anschließend mit einer exponentiell kleinen Wahrscheinlichkeit $n^{-\Omega(n)}$ unabhängig ihre Entscheidung revidieren (wiederum eine Art *coupling*, vgl. Abschnitt 3.5.3). Die Wahrscheinlichkeit mindestens einer Revision in polynomiell vielen Schritten ist durch $n^{-\Omega(n)}$ beschränkt.

Letztendlich wollen wir auf folgende Aussage hinaus.

Lemma 6.4.6 *Sei $D(k,t)$ die Summe der Tiefen von k unabhängigen $1/u$-Bäumen mit einer Gesamtzahl von t Knoten. Es gilt*

$$\text{Prob}(D(k,t) \geq 4t/u + 3kH_u) \;\leq\; \left(\frac{\mathrm{e}}{3}\right)^{t/u - O(k\log t)} \;=\; 2^{-\Omega(t/u)+O(k\log t)}.$$

Wir verschieben den technischen, aber methodisch interessanten Beweis von Lemma 6.4.6 auf Abschnitt 6.4.3. Wenn ein Stammbaum nicht tief wird, besagt unsere Intuition, dass auf keinem Pfad im Baum Individuen entstehen, die sich in den Präfixbits, die in normalen Phasen unwichtig sind, stark von der Wurzel unterscheiden. Dies konkretisiert das folgende Lemma.

Lemma 6.4.7 *Sei eine normale Phase des Beispiel-GAs auf f mit Länge s betrachtet und sei x das zu Beginn der Phase FSE-maximale Individuum. Angenommen, das FSE-Potenzial erhöht sich in der Phase k-mal. Für jedes $d \geq 0$ gilt, dass am Ende der Phase mit einer Wahrscheinlichkeit von höchstens*

$$\mathrm{e}^{-dm/(3n)} \cdot k \cdot (\mu \cdot (n+1))^k \cdot \left(1 + \frac{2}{\mu}\right)^s \;+\; n^{-\Omega(n)}$$

im Stammbaum von x ein Knoten existiert, der eine Tiefe von höchstens d hat und für dessen Beschriftung x' die Eigenschaft $\text{PE}(x') - \text{PE}(x) \geq 3d$ gilt.

Auch den Beweis des letzten Lemmas verschieben wir, nämlich ans Ende dieses Unterabschnitts, und widmen uns jetzt dem Beweis von Theorem 6.4.1.

Beweis von Theorem 6.4.1: Wir arbeiten mit der Beweismethode des typischen Laufes. Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-2\ell/3+o(\ell)}$ gibt es laut Lemma 6.3.2 einen ersten Zeitpunkt $t^* \leq 2\mathrm{e}\ell n \ln(\mathrm{e}\ell)$, an dem der Beispiel-GA eine Population mit genau einem wohlgeformten Individuum x erzeugt hat, für das $\mathrm{PE}(x) \leq 7m/12$ gilt. Wir nehmen im Folgenden an, dass ein solches t^* existiert, und führen damit nur einen Fehlerterm von höchstens $2^{-2\ell/3+o(\ell)}$ ein.

Beginnend ab dem Zeitpunkt t^* betrachten wir eine Phase fester Länge $s := \lceil 3\mathrm{e}\ell n \rceil$. Wir zeigen dann, dass der Lauf des Beispiel-GAs mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ die folgenden beiden Eigenschaften erfüllt; die Phase heiße dann *erfolgreich.*

1. Die Phase ist normal gemäß Definition 6.4.5.

2. Es gibt darin einen Schritt, der ein Individuum mit FSE-Wert ℓ erzeugt.

Wenn die zweite Bedingung vor dem s-ten Schritt erfüllt ist, lassen wir die Phase vorzeitig enden. Wir beachten, dass am Ende einer erfolgreichen Phase gemäß der Definition normaler Phasen mindestens ein Individuum mit einem PE-Wert von höchstens $2m/3$ und FSE-Wert ℓ existiert. Wir werden später zeigen, dass es für den Beispiel-GA sehr leicht ist, aus dieser Ausgangsposition das Optimum zu erreichen.

Um die Wahrscheinlichkeit des Schnitts der beiden Eigenschaften abzuschätzen, betrachten wir zunächst die zweite Eigenschaft unter der Bedingung, dass die erste zutrifft. Nun können wir wieder mit einer Potenzialfunktion arbeiten. Naheliegenderweise betrachten wir das FSE-Potenzial der Population, das aufgrund der Definition von f und der ersten Eigenschaft während der Phase nicht sinken kann.

In einer normalen Phase beträgt die Wahrscheinlichkeit, ein FSE-maximales Individuum zur Mutation zu wählen, 1. Die Wahrscheinlichkeit, seinen FSE-Wert zu erhöhen, ist mindestens $(1/n)(1-1/n)^{n-1} \geq 1/(\mathrm{e}n)$, da es hinreichend ist, allein die linkeste Null im Suffix zu kippen. Diese untere Schranke bleibt auch unter der Annahme, dass die erste Eigenschaft gilt, gültig. Somit enthält die Phase der Länge s im Erwartungswert mindestens 3ℓ Schritte, die das FSE-Potenzial erhöhen, und gemäß der Chernoffungleichung mit einer Wahrscheinlichkeit von mindestens $1 - \mathrm{e}^{-2\ell/3} = 1 - 2^{-\Omega(n^{1/2})}$ mindestens ℓ solcher Schritte. (Die etwas genauere Abschätzung $2\ell/3$ für den Exponenten wird später relevant werden.) Damit ist bereits die zweite Eigenschaft gezeigt.

Zum Nachweis der ersten Eigenschaft werden wir die Charakterisierung (der relevanten Knoten) des Stammbaums $T_t(x)$ des obigen Individuums x als Folge von Beinahe-$1/u$-Bäumen (vgl. Lemma 6.4.4 und die nachfolgende Argumentation) anwenden. Damit werden wir zum einen zeigen, dass $T_s(x)$, der Baum am Ende der Phase, mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ eine Tiefe von höchstens $n/36$ hat. Zum anderen werden wir mit Lemma 6.4.7 nachweisen können, dass mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ an keinem Knoten x' mit einer Tiefe von höchstens $n/36$ in $T_s(x)$ die Beziehung $\mathrm{PE}(x') \geq \mathrm{PE}(x)+m/12$ erfüllt ist. Also ist mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ sowohl das letzte Ereignis als auch das vorletzte Ereignis erfüllt. Da $\mathrm{PE}(x) \leq 7m/12$ gilt, folgt daraus, dass mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ kein Nachfahre von x in der Phase einen PE-Wert von mehr als $7m/12+m/12=2m/3$ erhält. Dann ist die Phase mit einer Wahrscheinlichkeit von $1-n^{-\Omega(n)}$ normal.

Wir betrachten ab sofort nur diejenigen Knoten in $T_s(x)$, die eine Tiefe von höchstens $n/36$ haben, und wollen zu der Behauptung über die PE-Werte kommen. Weil wir insgesamt das Ereignis betrachten, eine normale Phase zu haben, können wir Lemma 6.4.7 anwenden. Sei $u := 45n^{1/2}$, also die untere Schranke für μ aus dem Theorem. Das Lemma zeigt nun für $k := \ell = \lceil n^{1/2}/45 \rceil$, $d := n/36$ und genügend großes n, dass mit einer Wahrscheinlichkeit von höchstens

$$\begin{aligned} &\mathrm{e}^{-m/108} \cdot \ell \cdot (\mu \cdot (n+1))^{\ell} \cdot \left(1+\frac{2}{u}\right)^{s} + n^{-\Omega(n)} \\ \leq\ & 2^{O(n^{1/2}\log n)} \mathrm{e}^{-m/108} \left(1+\frac{2}{45n^{1/2}}\right)^{3\mathrm{e}n^{3/2}/45+1} + n^{-\Omega(n)} \\ \leq\ & 2^{O(n^{1/2}\log n)} \mu \mathrm{e}^{-m/108+n/124+1} + n^{-\Omega(n)} \;=\; 2^{-\Omega(n)} \end{aligned}$$

(da $\mu = \mathrm{poly}(n)$) die Behauptung über die PE-Werte nicht erfüllt ist.

Wir müssen nun die Tiefe von $T_s(x)$ wie gewünscht beschränken. Dazu nehmen wir an, dass der Prozess, der die relevanten Knoten von $T_s(x)$ zwischen zwei Erhöhungen des FSE-Potenzials erzeugt, der Prozess eines $1/u$-Baums ist. Weil wir insgesamt das Ereignis betrachten, eine normale Phase zu haben, führen wir damit nur einen Fehlerterm der Größe $n^{-\Omega(n)}$ ein, der dem Übergang von Beinahe-$1/u$-Bäumen auf $1/u$-Bäume zuzuschreiben ist. Da maximal ℓ FSE-Erhöhungen möglich sind, beschränken wir die Tiefe von $T_s(x)$ schließlich wie folgt. Wir addieren die Tiefen von ℓ unabhängigen $1/u$-Bäumen mit einer Gesamtzahl von s Knoten und addieren anschließend noch 1, um die irrelevanten Knoten zu erfassen.

Wir rufen Lemma 6.4.6 mit $t := s = \lceil 3\mathrm{e}\ell n \rceil$, $k := \ell = \lceil n^{1/2}/45 \rceil$ auf und erhalten

$$\frac{4t}{u} + 3kH_u \;\leq\; \frac{12\mathrm{e}\ell n + 4}{45n^{1/2}} + 3\ell(\ln u + 1) \;\leq\; \left(\frac{12\mathrm{e}}{2025} + O(n^{-1/2}\log n)\right) n,$$

was höchstens $n/36-1$ ist, falls n groß genug ist. (Der Summand -1 ist nötig, da wir irrelevante Knoten erfassen wollen.) In der Notation von Lemma 6.4.6 erhalten wir

$$\mathrm{Prob}(D(k,t) \geq n/36-1) \;=\; 2^{-\Omega(t/u)+O(k\log t)} \;=\; 2^{-\Omega(n)}.$$

Damit ist die Behauptung bezüglich der Tiefe von $T_s(x)$ gezeigt. Da die Summe aller bis hier betrachteten Fehlerwahrscheinlichkeiten $2^{-\Omega(n^{1/2})}$ beträgt, ist auch die erste der obigen Eigenschaften gezeigt.

Ein Individuum x mit $\mathrm{FSE}(x) = \ell$ und $\mathrm{PE}(x) < 2m/3$ ist fast optimal. Der Elitismus des Beispiel-GAs impliziert, dass der maximale PE-Wert der aktuellen Population nach Erzeugung eines solchen Individuums nicht mehr sinken kann. Daher können wir mit der Funktion

$$P(x) := \begin{cases} \mathrm{PE}(x), & \text{falls } \mathrm{FSE}(x) = \ell \text{ und } \mathrm{PE}(x) \leq 2m/3, \\ -1 & \text{sonst} \end{cases}$$

argumentieren. Solange das P-Potenzial der Population noch nicht $2m/3$ erreicht hat, ist die Wahrscheinlichkeit einer Erhöhung mindestens $(1/\mu)(m/3)(1/n)$. Also genügt gemäß der Chernoffungleichung eine Phase von $3\mu n$ Schritten mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$, um das P-Potenzial auf $2m/3$ zu erhöhen und damit das Optimum zu erreichen. Die Summe aller bislang betrachteten Fehlerwahrscheinlichkeiten ist durch $2^{-\Omega(n)} + 2^{-2\ell/3+o(\ell)} + \mathrm{e}^{-2\ell/3} = 2^{-2\ell/3+o(\ell)}$ beschränkt. Im Folgenden bezeichnen wir einen Lauf, in dem kein Ereignis eintritt, das in die genannten Fehlerwahrscheinlichkeiten eingeht, als typisch. Die Laufzeit bis zum Erreichen des Optimums ist in einem typischen Lauf insgesamt durch $\mu + 2\mathrm{e}\ell n \ln(\mathrm{e}\ell) + \lceil 3\mathrm{e}\ell n \rceil + 3\mu n \leq 5\mathrm{e}\ell n \ln(\mathrm{e}\ell) + 4\mu n$ nach oben beschränkt. Damit ist die erste Aussage des Theorems gezeigt.

Es bleibt die Aussage über die erwartete Laufzeit des Beispiel-GAs zu zeigen. Dazu müssen wir nun zusätzlich den Fall betrachten, dass Populationen erzeugt werden, die wohlgeformte Individuen enthalten, aber nicht normal sind. Aus Lemma 6.3.2 wissen wir, dass nach einer erwarteten Zeit von $O(n \log n)$ eine Population mit mindestens einem wohlgeformten Individuum entsteht. Außerdem liefert uns die obige Analyse der beiden Eigenschaften, da wir unabhängige Phasen wiederholen dürfen, Folgendes. Nach einer erwarteten Zeit von $O(\ell n)$ entsteht entweder eine normale Population mit FSE-Potenzial ℓ oder es entsteht ein wohlgeformtes Individuum mit einem PE-Wert von mehr als $2m/3$. Im ersten der beiden Fällen können wir wie im letzten Absatz argumentieren und erhalten eine erwartete Zeit von insgesamt $O(\ell n \log n + \mu n)$, um das Optimum zu finden. Im zweiten Fall ziehen wir die Funktion P' mit $P'(x) := \mathrm{PE}^*(x)$, falls x wohlgeformt und $\mathrm{PE}(x) > 2m/3$ ist, und andernfalls $P'(x) := -1$ heran. Offensichtlich ist P' monoton bezüglich f (wenn wir pessimistisch Individuen mit FSE-Wert ℓ ignorieren) und eine Mutation, die ein Individuum mit nicht negativem P'-Wert wählt, erhöht mit einer konstanten Wahrscheinlichkeit den P'-Wert, vgl. auch den Beweis von Lemma 6.3.3. Nach Lemma 5.2.1 beträgt die erwartete Zeit bis zum Erreichen eines P'-Potenzials von $2m/3 + b$ höchstens $O(\mu n)$. Anschließend betrachten wir den Wert L mit $L(x) := \mathrm{FSE}(x)$, falls $\mathrm{PE}(x) = 2m/3 + b$ gilt und x wohlgeformt ist, und andernfalls $L(x) := -1$. Die Funktion L ist monoton (wiederum unter der pessimistischen Annahme) und die erwartete Zeit bis zum Erreichen eines L-Potenzials von ℓ ist mit den üblichen Abschätzungen für Mutationswahrscheinlichkeiten und gemäß Lemma 5.2.1 beschränkt durch $O(\mu \ell n)$. Also ist ein lokal optimales Individuum x mit $\mathrm{PE}(x) = 2m/3 + b$ und $\mathrm{FSE}(x) = \ell$ nach erwarteten $O(\mu \ell n)$ Schritten gefunden.

Zuletzt betrachten wir die Wahrscheinlichkeit, aus einem lokal optimalen Individuum ein global optimales Individuum zu erzeugen. Offensichtlich genügt eine Mutation, die genau b Einsen im Präfix kippt. Die Wahrscheinlichkeit, ein lokal optimales Individuum zu wählen und eine solche Mutation zu vollführen, ist mindestens

$$\frac{1}{\mu}\left(\frac{1}{n}\right)^{b}\left(1-\frac{1}{n}\right)^{n-b} \;=\; 2^{-(\log n)\cdot b - O(\log \mu)} \;=\; 2^{-\ell/2-o(\ell)},$$

da $b \leq \ell/(2\log n)+1$ und $\mu = \mathrm{poly}(n)$. Die erwartete Zeit bis zu einem solchen Schritt kann dann durch $2^{\ell/2+o(\ell)}$ nach oben abgeschätzt werden. Also ist die erwartete Laufzeit des Beispiel-GAs auf f durch $2^{\ell/2+o(\ell)} + O(\mu\ell n) = 2^{\ell/2+o(\ell)}$ beschränkt, wenn kein typischer Lauf beobachtet wird. Jedoch ist die Laufzeit im Falle eines typischen Laufes sogar $O(\ell n \log n + \mu n)$ und ein typischer Lauf hat eine Wahrscheinlichkeit von mindestens $1-2^{-2\ell/3+o(\ell)}$. Wir wenden den Satz von der totalen Wahrscheinlichkeit an. Die erwartete Laufzeit ist insgesamt $O(\ell n \log n + \mu n)$, da das Produkt von $2^{-2\ell/3+o(\ell)}$ und $2^{\ell/2+o(\ell)}$ gegen 0 konvergiert. □

Im letzten Absatz des Beweises von Theorem 6.4.1 haben wir illustriert, dass der Wert b in der Definition von f geschickt gewählt wurde, um die erwartete Laufzeit des Beispiel-GAs zu beschränken. Die Erkenntnis über die polynomielle erwartete Laufzeit ist zwar aus struktureller Hinsicht wichtig, für praktische Belange ist aber die erste Aussage des Theorems interessanter. Dort haben wir nicht nur die Laufzeit – abgesehen von einer exponentiell kleinen Fehlerwahrscheinlichkeit – durch ein Polynom nach oben beschränkt, sondern auch eine Abschätzung für die Leitkoeffizienten des Polynoms präsentiert. Obschon recht grob, demonstrieren die Abschätzungen, dass die genannten Koeffizienten nicht abstrus groß werden. Demzufolge haben wir die Vermutung erhärtet, dass der Beispiel-GA auf f auch im praktischen Sinne „effizient" ist, also eine akzeptable Laufzeit besitzt, wohingegen eine in O-Notation gefasste Laufzeitschranke darüber keinen Aufschluss liefern könnte. Zusätzlich ist die Populationsgröße, die eine polynomielle erwartete Laufzeit ermöglicht, mit $45n^{1/2}$ durch einen im praktischen Sinne nicht allzu abwegigen Wert beschränkt.

Nachdem bis hierhin Intuition für das Verständnis von Theorem 6.4.1 und letztendlich auch sein formaler Beweis unter der Voraussetzung, dass die Lemmata 6.4.6 und 6.4.7 gelten, vermittelt wurde, müssen wir uns jetzt die noch ausstehenden Beweise vornehmen. Als Erstes führen wir den Beweis von Lemma 6.4.7 durch. Dazu benötigen wir das folgende Lemma, welches in Kapitel 7 erneut von entscheidender Bedeutung sein wird. Die dahinter stehende Intuition lautet, dass wir in Stammbäumen unter gewissen Umständen die Fehler abschätzen können, die uns unterlaufen, wenn wir annehmen, dass ein Pfad in einem Stammbaum durch eine Folge von unabhängigen Mutationen entsteht, vgl. Beobachtung 5.2.1. Dabei werden wir zwei Fälle unterscheiden. Entweder hängt die Selektion zur Reproduktion gar nicht von der Gestalt der Individuen der Population ab oder dies gilt zumindest für Teile, nämlich Präfixe, der Individuen. Daher bezeichne für $m \leq n$ die Funktion $\pi_m(x)$, angewandt auf $x = (x_1,\ldots,x_n) \in \{0,1\}^n$, das Präfix der Länge m von x, d. h. $\pi_m(x) = (x_1,\ldots,x_m)$.

Lemma 6.4.8 *Sei ein Ein-Kind-EA A gegeben, der stets gleichverteilt aus der aktuellen Population zur Mutation wählt. Sei weiterhin ein $m \leq n$ gegeben. Betrachtet seien ein Individuum x aus einer Population von A und sein Stammbaum $T_s(x)$ nach s Schritten. Für jedes $d \geq 0$ ist die Wahrscheinlichkeit, dass es in $T_s(x)$ einen Knoten gibt, der eine Tiefe von höchstens d hat und dessen Beschriftung x' die Beziehung $H(\pi_m(x), \pi_m(x')) \geq 2dm/n$ erfüllt, höchstens $\mathrm{e}^{-dm/(3n)}\mu(1+2/\mu)^s$.*

Die Aussage bleibt gültig, wenn A in den ersten $s' \leq s$ Schritten anders zur Mutation wählt, aber dann weder die Selektion zur Reproduktion noch die Selektion zur Ersetzung von den π_m-Werten der Individuen der aktuellen Population abhängen.

Beweis: Zur Vereinfachung der Notation sei x aus der Population $X^{(0)}$. Ein Knoten in $T_s(x)$, der die beschriebene Eigenschaft hat, heiße *schlecht* und $T_s(x)$ heiße schlecht, wenn es darin mindestens einen schlechten Knoten gibt. Wir werden die Wahrscheinlichkeit abschätzen, dass $T_s(x)$ schlecht ist, und zeigen zunächst den Fall gleichverteilter Wahl zur Reproduktion, d. h. $s' = 0$. Die Beweisidee dafür besteht darin, A mit einem populationsbasierten EA A^* zu vergleichen, in dem kein Selektionsoperator mehr von den Individuen abhängt. Die Besonderheit ist, dass A^* niemals Individuen löscht und daher mit einer monoton wachsenden Populationsgröße arbeitet.

Der EA A^* arbeitet wiederum mit Multimengen, und zwar auf folgende Weise. Die Initialisierung ist wie bei einem Ein-Kind-EA. Zum Zeitpunkt t, $t \geq 0$, wählt er aus der aktuellen Population $X^{(t)}$ jedes Individuum mit einer Wahrscheinlichkeit von mindestens $1/\mu$, indem er $k := \lceil |X^{(t)}|/\mu \rceil$ Individuen $y_1, \ldots, y_k$ zufällig gleichverteilt aus $X^{(t)}$ (mit Zurücklegen) zieht. Jedes y_i mit $1 \leq i \leq k$ wird durch unabhängige Anwendung des $1/n$-Standardmutationsoperators zu einem y_i' mutiert. Anschließend erzeugt A^* die Nachfolgepopulation gemäß $X^{(t+1)} := X^{(t)} \cup \{y_1', \ldots, y_i'\}$.

Auch der Lauf von A^* induziert für x bis zum Zeitpunkt s eine Folge von Stammbäumen $T_t^*(x)$, $0 \leq t \leq s$, die bis auf die Knotenmenge analog zu Definition 5.2.3 definiert sind. Die Knotenmenge enthält jetzt Paare (t, i), um anzugeben, dass der Knoten in der i-ten Mutation des t-ten Schritts eingefügt wurde. Sei τ_s ein Stammbaum, der eine mögliche Ausprägung für $T_s(x)$ ist. Wir sagen, dass ein Teilgraph $(\tilde{V}, \tilde{E})$ einer Ausprägung von $T_s^*(x)$ isomorph zu τ_s ist, wenn die Graphen im üblichen Sinne isomorph sind, jeder (und damit nur ein) Knoten der Gestalt $(t, i) \in \tilde{V}$ auf den Knoten t aus τ_s abgebildet wird und die Beschriftungen von (t, i) und t jeweils übereinstimmen. Die entscheidende Beobachtung ist nun, dass mit einer Wahrscheinlichkeit von mindestens $\mathrm{Prob}(T_s(x) = \tau_s)$ der Baum $T_s^*(x)$ einen zu τ_s isomorphen Teilgraphen besitzt. Der Grund ist, dass in den Bäumen $T_t^*(x)$ keine Knoten gelöscht werden, jeder aktuell vorhandene Knoten wie bei $T_t(x)$ mit einer Wahrscheinlichkeit von mindestens $1/\mu$ gewählt und derselbe Mutationsoperator eingesetzt wird. Wir betrachten einen beliebigen Pfad in $T_s^*(x)$, der an der Wurzel startet und dessen Länge höchstens d beträgt. Wenn es gelingt, die Wahrscheinlichkeit, dass der Pfad einen schlechten Knoten enthält, und die Gesamtzahl der Pfade abzuschätzen, können wir also eine obere Schranke für die Wahrscheinlichkeit, dass $T_s(x)$ schlecht ist, gewinnen.

Auf den Baum $T_s^*(x)$ trifft Beobachtung 5.2.1 zu. Jeder Pfad mit Individuen $x_0 = x, x_1, \ldots, x_k$ in $T_s^*(x)$ entspricht einer Folge von Anwendungen des $1/n$-Standardmutationsoperators, sodass x_{i+1} für $0 \leq i \leq k-1$ jeweils durch unabhängige Anwendung des Mutationsoperators auf x_i entsteht. Zudem kippt jede Mutation einzelne Bits unabhängig voneinander. Damit können wir wieder unabhängige Zufallsvariablen $X_{i,j}$ für die Bernoulliversuche, dass die i-te Mutation das j-te Bit, $j \leq m$, von x_i kippt, einführen. Weil wir nur Pfade einer Länge von höchstens d betrachten, wenden wir die Chernoffungleichung auf die Zahl insgesamt kippender Präfixbits in d Schritten an. Ihr Erwartungswert ist dm/n und die Wahrscheinlichkeit von insgesamt mindestens $2dm/n$ kippenden Bits ist höchstens $\mathrm{e}^{-dm/(3n)}$. Also gibt es mit einer Wahrscheinlichkeit von höchstens $\mathrm{e}^{-dm/(3n)}$ auf einem beliebigen Pfad einer Länge von maximal d einen Knoten, dessen Präfix zum Präfix von x_0 einen Hammingabstand von mindestens $2dm/n$ hat.

Wir müssen noch die Zahl der Pfade in $T_s^*(x)$ nach oben abschätzen. Wir beschränken diese durch $|X^{(s)}|$. Da $|X^{(0)}| = \mu$, gilt $\lceil |X^{(t)}|/\mu \rceil \leq 2|X^{(t)}|/\mu$ für $t \geq 0$. Also ist $|X^{(t+1)}| \leq |X^{(t)}|(1+2/\mu)$ und damit $|X^{(s)}| \leq \mu(1+2/\mu)^s$. Folglich ist die Wahrscheinlichkeit, dass es einen schlechten Knoten in $T_s(x)$ gibt, wie gewünscht beschränkt.

Wir müssen noch die Aussage zeigen, wenn A an den Zeitpunkten $0, \ldots, s'-1$ anders zur Mutation wählt, aber die Präfixbits weder zur Reproduktion noch zur Ersetzung verwendet. Wir modifizieren A^* dahin gehend, dass A^* an den Zeitpunkten $0, \ldots, s'-1$ wie A arbeitet. Im Baum $T_{s'}^*(x)$ gilt jetzt immer noch Beobachtung 5.2.1, wenn wir lediglich die Präfixe der Individuen betrachten. Da ab dem Zeitpunkt s' nicht mehr gelöscht wird, gilt die Beobachtung für Präfixe auf Pfaden im gesamten Baum. Das Lemma folgt, da wir nicht mehr Knoten als im Fall $s' = 0$ erzeugen. □

Beweis von Lemma 6.4.7: Sei x wieder das zu Beginn der normalen Phase, o. B. d. A. Zeitpunkt 0, vorliegende FSE-maximale Individuum. Der Stammbaum $T_s(x)$ enthält eine Folge von Beinahe-$1/u$-Bäumen mit $u := \mu$ und zerfällt in relevante und irrelevante Knoten (vgl. Seite 116). Alle irrelevanten Knoten haben jedoch einen relevanten Elter. Wir müssen auch die PE-Werte an irrelevanten Knoten im Auge behalten.

Wir betrachten zunächst die Subphase vom Anfang der normalen Phase bis zu dem Zeitpunkt t^*, der erstmalig das FSE-Potenzial erhöht. Die Charakterisierung der relevanten Knoten von $T_{t^*}(x)$ als Beinahe-$1/u$-Baum lässt folgende zwei Beobachtungen bezüglich der Belegung der Präfixbits der Individuen der Populationen in der Subphase zu. Während der Baum einschwingt (vgl. Seite 119), haben diese Bits fast überhaupt keinen Einfluss auf die Auswahlwahrscheinlichkeiten. Während des Einschwingens nehmen wir daher nun vereinfachend an, dass der f-Wert gar nicht vom PE-Wert abhinge. Wir wissen bereits, dass wir damit insgesamt die gesuchte Wahrscheinlichkeit nur um einen Term der Größenordnung $n^{-\Omega(n)}$ überschätzen. Nach dem Einschwingen spielt der PE-Wert eine Rolle, da der Beispiel-GA elitär ist. Dies wirkt sich auf die Wahl der gelöschten Individuen aus, während wir für die zur Mutation gewählten Individuen wieder vereinfachend die Gleichverteilung annehmen. Wir können also Lemma 6.4.8

auf die Schritte bis t^*, einschließlich der irrelevanten Schritte, anwenden. Zudem dürfen wir das Lemma unabhängig auf alle in der Phase entstehenden $1/u$-Bäume anwenden. Im Folgenden sprechen wir nur noch von Bäumen anstelle von $1/u$-Bäumen, wenn Missverständnisse ausgeschlossen sind.

Sei für $1 \leq i \leq k$ mit s_i die Anzahl der Schritte der Phase, die den i-ten Baum und die daran hängenden irrelevanten Knoten erzeugen, bezeichnet. Die Wurzel des $(i+1)$-ten Baums ist ein Blatt im i-ten Baum; sei für $i < k$ mit d_i die Tiefe dieses Blattes im i-ten Baum bezeichnet. Wir betrachten die s_i-Werte und d_i-Werte zunächst als beliebig gegeben. Relevante Knoten in $T_s(x)$ können genau einem der Bäume zugeordnet werden, wenn wir die Wurzel des i-ten Baums nur dem i-ten Baum zuschreiben. Ebenso können wir jeden irrelevanten Knoten genau einem Baum zuordnen, indem wir seinen Elter betrachten. Sei ein beliebiger Knoten v^* mit Tiefe $d' \leq d$ in $T_s(x)$ betrachtet und sei j die Nummer des $1/u$-Baums, dem v^* zugeordnet wird. Auf dem Pfad von x zum Knoten v^* befinden sich jeweils d_i Knoten aus den Bäumen mit Index $i < j$ und innerhalb des j-ten Baums hat v^* eine Tiefe von $d' - d_1 - \cdots - d_{j-1}$. Den Fall, dass v^* nicht im k-ten Baum liegt, werden wir im Folgenden behandeln, indem wir für alle $k' < k$ auch das Ereignis betrachten, dass $T_s(x)$ nur aus k' $1/u$-Bäumen besteht. Also betrachten wir zunächst nur noch Knoten v^* in $T_s(x)$, die eine Tiefe von $d' \leq d$ haben und im k-ten $1/u$-Baum liegen. Wir setzen $d_k := d' - d_1 - \cdots - d_{k-1}$.

Der Pfad zu v^* läuft über Knoten mit einer Tiefe von jeweils höchstens d_i im i-ten $1/u$-Baum. Die Wahrscheinlichkeit, dass es einen Knoten gibt, dessen Beschriftung x' die Eigenschaft $\mathrm{PE}(x') - \mathrm{PE}(x) \geq 3d$ erfüllt, wird nur größer, wenn wir Pfade verlängern. Deswegen betrachten wir ab sofort nur noch d_i-Werte mit $d_1 + \cdots + d_k = d$. Für Knoten w im i-ten Baum bezeichne $h(w)$ den Hammingabstand zwischen dem Präfix (der Beschriftung) von w und dem Präfix der Wurzel des i-ten Baums. Es bezeichne die Zufallsvariable H_i das Maximum der $h(w)$-Werte über alle Knoten w, die im i-ten $1/u$-Baum auf einer Tiefe von höchstens d_i liegen. Sei $H := H_1 + \cdots + H_k$. Offensichtlich ist H eine obere Schranke für den Hammingabstand der Präfixe von v^* und x. Für jede feste Ausprägung $(h_1, \ldots, h_k)$ der H_i-Werte, sodass $h_1 + \cdots + h_k \geq 3d$ gilt, werden wir die zugehörige Wahrscheinlichkeit beschränken und anschließend die Anzahl insgesamt möglicher Ausprägungen nach oben abschätzen.

Sei $(h_1, \ldots, h_k)$ irgendeine Ausprägung mit $h_1 + \cdots + h_k \geq 3d$. Außerdem sei k^* die Anzahl der h_i mit $h_i \geq 2d_i$ und es sei o. B. d. A. angenommen, dass $h_i \geq 2d_i$ für $1 \leq i \leq k^*$ gilt. Gemäß Definition der h_i und d_i gilt $h_{k^*+1} + \cdots + h_k \leq 2d$. Letzteres impliziert $k^* \geq 1$ und $h_1 + \cdots + h_{k^*} \geq d$. Aufgrund der Unabhängigkeit der $1/u$-Bäume ist die Wahrscheinlichkeit, dass $h_i \geq 2d_i$ für $i \leq k^*$ gilt, damit laut Lemma 6.4.8 höchstens

$$\prod_{i=1}^{k^*} \left(\mathrm{e}^{-h_i m/(3n)} \mu \left(1 + \frac{2}{\mu}\right)^{s_i} \right) \;\leq\; \mathrm{e}^{-dm/(3n)} \mu^k \left(1 + \frac{2}{\mu}\right)^{s},$$

was insgesamt die Wahrscheinlichkeit jeder betrachteten Ausprägung der h_i-Werte beschränkt. Der Ausdruck ist monoton wachsend in k, sodass eine Multiplikation der

Schranke mit k genügt, um den oben erwähnten Fall, dass v^* nicht im k-ten Baum liegt, zu behandeln. Die Anzahl der insgesamt möglichen Ausprägungen der h_i-Werte ist trivialerweise durch $(n+1)^k$ beschränkt. Die Rechnungen gelten für alle möglichen s_i- und d_i-Werte. Folglich erhalten wir zusammen mit dem oben erwähnten Ausdruck $n^{-\Omega(n)}$ die im Lemma genannte Abschätzung. □

Im letzten Unterabschnitt dieses Abschnitts 6.4 werden wir den noch ausstehenden Beweis der Tiefenbeschränkung von $1/u$-Bäumen, d. h. Lemma 6.4.6, liefern.

6.4.3 Methoden zur Analyse der Tiefe zufälliger Bäume

Um die Tiefe zunächst eines $1/u$-Baums zu beschränken, greifen wir auf einen kombinatorischen Ansatz zurück, der stark an die Ausführungen von Arya, Golin und Mehlhorn (1999), welche die Tiefe zufälliger Schaltkreise abschätzen, angelehnt ist. Als ein Nebenprodukt werden unsere Rechnungen auch die erwartete Tiefe eines ZRBs mit t Knoten beschränken. Diesen Erwartungswert kann Pittel (1994) mithilfe von Poissonprozessen exakt berechnen. Unser kombinatorischer Ansatz hat aber den Vorteil, für die mit asymptotischer Analyse vertrauten Leser(innen) besser zugänglich zu sein und auch eine Aussage über die Verteilung der Tiefe von ZRBs und $1/u$-Bäumen zu liefern. Letzteres ist von entscheidender Bedeutung, um exponentiell kleine Fehlerwahrscheinlichkeiten zu zeigen.

Definition 6.4.9 *Sei $L(t,d)$ die Anzahl der Knoten mit Tiefe d in einem $1/u$-Baum zum Zeitpunkt t. Bezeichne $E(t,d) := \mathrm{E}(L(t,d))$ den zugehörigen Erwartungswert.*

Die Notation aus Definition 6.4.9 ist an der entsprechenden Notation von Arya, Golin und Mehlhorn (1999) orientiert. Zunächst betrachten wir den Fall $t \le u$, d. h., der $1/u$-Baum verhält sich wie ein ZRB. Wir stellen einige grundlegende Beziehungen für das Maß $E(t,d)$ auf.

Lemma 6.4.10 *Sei ein $1/u$-Baum betrachtet. Dann gilt*

$$\begin{aligned} E(t,0) &= 1 && \text{für } t \ge 0,\\ E(0,d) &= 0 && \text{für } d \ge 1,\\ E(t,d) &\le E(t-1,d) + \frac{E(t-1,d-1)}{t} && \text{für } 1 \le t \le u \text{ und } d \ge 1. \end{aligned}$$

Beweis: Die erste und die zweite Beziehung gelten gemäß Definition 6.4.3. Zum Beweis der dritten Beziehung halten wir fest, dass $L(t,d)$ sich von $L(t-1,d)$ genau dann unterscheidet, falls zum Zeitpunkt t ein Knoten auf Tiefe $d-1$ ausgewählt wird, um Elter des neu eingefügten Knotens zu sein. Wenn $L(t-1,d-1)=i$, passiert dies gemäß Definition 6.4.3 mit einer Wahrscheinlichkeit von höchstens i/t, da wir $t \leq u$ betrachten. Laut dem Satz von der totalen Wahrscheinlichkeit erhalten wir

$$E(t,d) \;\leq\; E(t-1,d) + \sum_{i=1}^{\infty} \mathrm{Prob}(L(t-1,d-1) = i) \cdot \frac{i}{t}.$$

Da die letzte Summe die Definition von $E(t-1,d-1)$ enthält, entspricht die rechte Seite gerade $E(t-1,d) + E(t-1,d-1)/t$ wie oben behauptet. □

Mithilfe der vorigen Beziehungen können wir $E(t,d)$ beschränken.

Lemma 6.4.11 *Für $0 \leq t \leq u$ und $d \geq 0$ gilt*

$$E(t,d) \;\leq\; \frac{(H_t)^d}{d!}.$$

Beweis: Wir nehmen zunächst an, dass die dritte Beziehung aus Lemma 6.4.10 eine Gleichheit ist und sogar für $t > u$ zutrifft. Obere Schranken für das modifizierte $E(t,d)$ werden offensichtlich obere Schranken für das ursprüngliche $E(t,d)$ im Fall $t \leq u$ liefern. Wir führen nun die erzeugende Funktion $E_d(x) := \sum_{t=0}^{\infty} E(t,d)x^t$ für die (modifizierten) $E(t,d)$-Werte ein und behandeln diese als Potenzreihe unter der impliziten Annahme, dass x stets innerhalb des Konvergenzradius liegt. (Für einen Überblick über das Hilfsmittel der erzeugenden Funktionen siehe z. B. Steger (2001).) Dann gilt

$$\begin{aligned} E_0(x) &= 1 + x + x^2 + \ldots \;=\; \frac{1}{1-x}, \\ E_d(x) &= \frac{1}{1-x} \int_0^x E_{d-1}(y)\,\mathrm{d}y \quad \text{für } d \geq 1. \end{aligned} \tag{6.1}$$

Die erste Gleichung folgt unmittelbar aus der Beziehung $E(t,0) = 1$ aus Lemma 6.4.10. Um die Gleichung (6.1) zu zeigen, ziehen wir die zweite Gleichung aus Lemma 6.4.10 heran und erhalten zunächst $E_d(x) = \sum_{t=1}^{\infty} E(t,d)x^t$, falls $d \geq 1$. Zusammen mit der dritten Beziehung (die als Gleichheit angenommen wird) folgt für $d \geq 1$

$$\begin{aligned} E_d(x) &= \sum_{t=1}^{\infty} E(t,d)x^t \;=\; \sum_{t=1}^{\infty} \left(E(t-1,d) + \frac{E(t-1,d-1)}{t} \right) \cdot x^t \\ &= x \sum_{t=0}^{\infty} E(t,d)x^t + \sum_{t=0}^{\infty} E(t,d-1) \frac{x^{t+1}}{t+1} \\ &= xE_d(x) + \sum_{t=0}^{\infty} E(t,d-1) \int_0^x y^t \,\mathrm{d}y \;=\; xE_d(x) + \int_0^x E_{d-1}(y)\,\mathrm{d}y. \end{aligned}$$

Die letzte Gleichheit folgt hier aus dem Satz von Beppo Levi (vgl. Lemma A.8), den wir anwenden können, da die $E(t,d)$-Werte nicht negativ sind. Indem wir auf beiden Seiten $xE_d(x)$ subtrahieren und dann jeweils durch $1-x$ teilen, kommen wir schließlich zur gewünschten Gleichung (6.1).

Als Nächstes beweisen wir die Gleichung

$$E_d(x) \;=\; \frac{(-\ln(1-x))^d}{d!(1-x)}$$

durch Induktion über d. Der Fall $d=0$ ist trivial. Für den Induktionsschritt benutzen wir (6.1) und weiterhin die Induktionsvoraussetzung. Wir erhalten

$$\begin{aligned} E_d(x) \;&=\; \frac{1}{1-x}\int_0^x E_{d-1}(y)\,\mathrm{d}y \;=\; \frac{1}{1-x}\int_0^x \frac{(-\ln(1-y))^{d-1}}{(d-1)!(1-y)}\,\mathrm{d}y \\ &=\; \frac{1}{(d-1)!(1-x)}\left[\frac{(-\ln(1-y))^d}{d}\right]_0^x \;=\; \frac{(-\ln(1-x))^d}{d!(1-x)}. \end{aligned}$$

Aus dieser geschlossenen Formel wollen wir $E(t,d)$, den Koeffizienten von x^t in der Reihendarstellung für $E_d(x)$, extrahieren. Dazu führen wir die Taylorreihendarstellung $-\ln(1-x) = \sum_{i=1}^{\infty} x^i/i$ ein und berechnen

$$(-\ln(1-x))^d \;=\; \left(\sum_{i=1}^{\infty}\frac{x^i}{i}\right)^d \;=\; \sum_{i=1}^{\infty}\left(\sum_{\substack{i_1,\dots,i_d\ge 1\\ i_1+\dots+i_d=i}}\frac{1}{i_1 i_2\cdots i_d}\right)x^i.$$

Die letzte Gleichung folgt dabei durch Faltung der Reihen. Es folgt, dass der Faktor $(-\ln(1-x))^d/(1-x) = (-\ln(1-x))^d \cdot \sum_{i=0}^{\infty} x^i$ aus der obigen geschlossenen Formel geschrieben werden kann als

$$\left(\sum_{i=1}^{\infty}\left(\sum_{\substack{i_1,\dots,i_d\ge 1\\ i_1+\dots+i_d=i}}\frac{1}{i_1 i_2\cdots i_d}\right)x^i\right)\left(\sum_{j=0}^{\infty}x^j\right) \;=\; \sum_{i=1}^{\infty}\left(\sum_{j=0}^{i}\sum_{\substack{j_1,\dots,j_d\ge 1\\ j_1+\dots+j_d=j}}\frac{1}{j_1 j_2\cdots j_d}\right)x^i,$$

worin die letzte Gleichung wiederum durch Faltung folgt. Damit ist $E(t,d)$, also der Koeffizient von x^t in der Reihendarstellung von $E_d(x)$, gegeben durch

$$E(t,d) \;=\; \frac{1}{d!}\cdot\sum_{\substack{i_1,\dots,i_d\ge 1\\ i_1+\dots+i_d\le t}}\frac{1}{i_1 i_2\cdots i_d} \;\le\; \frac{1}{d!}\cdot\sum_{1\le i_1,\dots,i_d\le t}\frac{1}{i_1 i_2\cdots i_d} \;=\; \frac{1}{d!}\left(\sum_{i=1}^{t}\frac{1}{i}\right)^d.$$

Der letzte Ausdruck entspricht $\frac{(H_t)^d}{d!}$. Dies liefert die gewünschte obere Schranke für das ursprüngliche $E(t,d)$ im Fall $t\le u$. $\square$

Nun kommen wir zu den Zeitpunkten $t \geq u+1$, d. h. zu den Zeitpunkten, an denen die Wahrscheinlichkeit, einen bestimmten Knoten aus einem $1/u$-Baum zu wählen, nur durch $1/u$ beschränkt werden kann.

Lemma 6.4.12 *Sei ein $1/u$-Baum betrachtet. Falls $t \geq u+1$ und $d \geq 1$, gilt*

$$E(t,d) \;\leq\; E(t-1,d) + \frac{E(t-1,d-1)}{u}.$$

Beweis: Der Beweis folgt mit denselben Argumenten wie der Beweis der dritten Beziehung innerhalb von Lemma 6.4.10. Aufgrund der Voraussetzung $t \geq u+1$ ist die Wahrscheinlichkeit, zum Zeitpunkt t einen Elter zu wählen, nun gemäß Definition 6.4.3 durch $1/u$ beschränkt. □

Wir können die bislang beschriebenen Analysen jetzt benutzen, um eine Abschätzung für $E(t,d)$, die für alle t gilt, herzuleiten.

Lemma 6.4.13 *Sei ein $1/u$-Baum betrachtet. Für $t \geq 0$ und $d \geq 0$ gilt dann*

$$E(t,d) \;\leq\; \frac{1}{d!}\left(H_u + \frac{t}{u}\right)^d.$$

Beweis: Wir beweisen die Ungleichung durch Induktion über d und t. Zunächst betrachten wir den Fall $d=0$. Da die zu zeigende obere Schranke dann 1 ist, folgt dieser Fall mithilfe von Lemma 6.4.10.

Für den Induktionsschritt von $d-1$ nach d mit $d \geq 1$ müssen wir alle Werte von t berücksichtigen und daher eine weitere Induktion nach t durchführen. Für $t \leq u$ folgt die Ungleichung aus Lemma 6.4.11. Für den Induktionsschritt von $t-1$ nach t nehmen wir $t \geq u+1$ an und benutzen die Rekurrenz aus Lemma 6.4.12. Gemäß der Induktionsvoraussetzung für $t-1$ erhalten wir dann

$$\begin{aligned}
E(t,d) &\leq \frac{1}{d!}\left(H_u + \frac{t-1}{u}\right)^d + \frac{1}{u}\cdot\frac{1}{(d-1)!}\left(H_u + \frac{t-1}{u}\right)^{d-1} \\
&= \frac{1}{d!}\left(\left(H_u + \frac{t-1}{u}\right)^d + \frac{d}{u}\left(H_u + \frac{t-1}{u}\right)^{d-1}\right) \\
&\leq \frac{1}{d!}\left(H_u + \frac{t}{u}\right)^d.
\end{aligned}$$

Die letzte Beziehung folgt mithilfe des binomischen Satzes (Lemma A.5), indem wir $H_u + t/u$ als $(H_u + (t-1)/u) + 1/u$ schreiben und in der Darstellung der d-ten Potenz nur die ersten beiden Terme berücksichtigen. Dies beschließt den Induktionsschritt für t und damit auch für d. □

Mit dem Ergebnis aus Lemma 6.4.13 können wir auf einfache Weise eine Schranke für die Verteilung der Tiefe herleiten.

Lemma 6.4.14 *Bezeichne $D(t)$ die Tiefe eines $1/u$-Baums zum Zeitpunkt t und sei $d \geq 3t/u + 3H_u$. Dann ist* $\mathrm{Prob}(D(t) \geq d) \leq (\mathrm{e}/3)^d$.

Beweis: Ohne Beschränkung der Allgemeinheit ist $d \in \mathbb{N}$. Damit $D(t) \geq d$ eintritt, ist $L(t,d) \geq 1$ notwendig (und hinreichend). Da nach der Markoffungleichung die Beziehung $\mathrm{Prob}(L(t,d) \geq 1) \leq E(t,d)$ zutrifft, wird das Lemma folgen, indem wir die Ungleichung $E(t,d) \leq (\mathrm{e}/3)^d$ für beliebige t zeigen.

Durch Anwendung der stirlingschen Formel auf die rechte Seite der Ungleichung aus Lemma 6.4.13 folgt

$$E(t,d) \;\leq\; \frac{\mathrm{e}^d}{d^d}\left(H_u + \frac{t}{u}\right)^d.$$

Gemäß unseren Voraussetzungen ist dies nach oben beschränkt durch

$$\frac{\mathrm{e}^d}{(3H_u + 3t/u)^d}\left(H_u + \frac{t}{u}\right)^d \;\leq\; \left(\frac{\mathrm{e}}{3}\right)^d$$

wie behauptet. □

Wir können nun endlich zum Beweis von Lemma 6.4.6 kommen. Eine etwas schlechtere, aber ausreichende Abschätzung für die Fehlerwahrscheinlichkeit, als dort formuliert ist, kann man bereits herleiten, indem man den Wertebereich der $D(t)$ mit Lemma 6.4.14 einschränkt und dann mithilfe der Aussagen über ihren Erwartungswert aus Lemma 6.4.13 die hoeffdingsche Ungleichung benutzt (siehe Hoeffding, 1963). Wir gehen hier aber einen bereits bekannten, direkteren Weg.

Beweis von Lemma 6.4.6: Seien für $1 \leq i \leq k$ die Tiefe des i-ten $1/u$-Baums mit D_i und die Anzahl seiner Knoten mit t_i bezeichnet. Indem wir $d_i := 3(t_i - 1)/u + 3H_u$ wählen (da t_i Knoten zur Zeit $t_i - 1$ vorhanden sind), liefert uns Lemma 6.4.14 für alle $d \geq d_i$ die Aussage

$$\mathrm{Prob}(D_i \geq d) \;\leq\; \left(\frac{\mathrm{e}}{3}\right)^d.$$

Wir gehen jetzt ähnlich wie im Beweis von Lemma 6.4.7 vor. Für jede feste Ausprägung der D_i-Werte, sodass $D(k,t) \geq 4t/u + 3kH_u$ gilt, beschränken wir die Wahrscheinlichkeit der Ausprägung durch $(\mathrm{e}/3)^{t/u}$. Anschließend schätzen wir die Anzahl der möglichen Ausprägungen ab, um das Lemma zu zeigen.

Sei $(e_1, \ldots, e_k)$ eine Ausprägung mit $e_1 + \cdots + e_k \geq 4t/u + 3kH_u$. Außerdem sei k^* die Anzahl der e_i mit $e_i \geq d_i$ und es sei o. B. d. A. angenommen, dass $e_i \geq d_i$ für $1 \leq i \leq k^*$ gilt. Gemäß Definition der d_i und e_i gilt $e_{k^*+1} + \cdots + e_k \leq 3t/u + 3kH_u$.

Letzteres impliziert $k^* \geq 1$ und $e_1 + \cdots + e_{k^*} \geq t/u$. Aufgrund der Unabhängigkeit der $1/u$-Bäume ist die Wahrscheinlichkeit, dass $e_i \geq d_i$ für $i \leq k^*$ gilt, damit höchstens

$$\left(\frac{\mathrm{e}}{3}\right)^{e_1} \cdot \left(\frac{\mathrm{e}}{3}\right)^{e_2} \cdots \left(\frac{\mathrm{e}}{3}\right)^{e_{k^*}} = \left(\frac{\mathrm{e}}{3}\right)^{e_1+\cdots+e_{k^*}} \leq \left(\frac{\mathrm{e}}{3}\right)^{t/u},$$

was insgesamt die Wahrscheinlichkeit jeder betrachteten Ausprägung der e_i-Werte beschränkt.

Es verbleibt die Abschätzung der Zahl der Ausprägungen. Offensichtlich kann $D(k,t)$ höchstens t Werte annehmen. Trivialerweise haben wir höchstens t^k Möglichkeiten, einen einzelnen Wert von $D(k,t)$ als Summe k nicht negativer ganzer Zahlen darzustellen. Insgesamt sind dies höchstens t^{k+1} Ausprägungen. Also lässt sich die Wahrscheinlichkeit des Ereignisses $D(k,t) \geq 4t/u + 3kH_u$ insgesamt durch

$$t^{k+1} \cdot \left(\frac{\mathrm{e}}{3}\right)^{t/u} = \left(\frac{\mathrm{e}}{3}\right)^{t/u - O(k \log t)},$$

nach oben beschränken. Dies ist $2^{-\Omega(t/u)+O(k \log t)}$, da $\mathrm{e}/3$ eine Konstante kleiner als 1 ist. $\square$

Die obere Schranke aus Lemma 6.4.6 ließe sich noch um einen Faktor $2^{-\Omega(k \log k)}$ verbessern, indem wir im vorigen Beweis die Anzahl der Ausprägungen besser abschätzten. Allerdings ist k bei allen Anwendungen so klein, dass dies im asymptotischen Rahmen keine Rolle spielt.

Mit dem nun vollständigen, technisch aufwendigen Beweis von Lemma 6.4.6 haben wir die obere Schranke für die erwartete Laufzeit des Beispiel-GAs auf f insgesamt bewiesen. Damit dürfte der Beweis dieser oberen Schranke insgesamt als der längste in dieser Dissertation geführte Beweis gelten. Dennoch ist die zugrunde liegende Beweisidee, die auf einer Analyse des Wachstums von Stammbäumen beruht, immer noch anschaulich und vielleicht auch wegweisend bei der Laufzeitanalyse populationsbasierter evolutionärer Algorithmen. Jedenfalls werden uns einige der Beweistechniken und Modelle zufälliger Bäume in Kapitel 7 bei der Analyse des (μ+1)-EA wieder begegnen. Auch kann man sich überlegen, dass die bisherigen Ergebnisse ebenso für Variationen des Beispiel-GAs gültig bleiben, beispielsweise, wenn die Selektion zur Ersetzung gleichverteilt ein Individuum mit geringstem f-Wert löscht. Darüber hinaus eignen sich die bislang entwickelten Beweistechniken dazu, auf nun einfache Weise das Verhalten des Beispiel-GAs auf Verallgemeinerungen und Modifikationen von f zu analysieren. Dies soll uns in den nächsten Abschnitten beschäftigen.

6.5 Eine Hierarchie von Funktionen

In den vorigen Abschnitten haben wir in den Theoremen 6.3.1, 6.3.5 und 6.3.6 sowie in Theorem 6.4.1 gezeigt, dass die Wahl der Populationsgröße des Beispiel-GAs von entscheidender Bedeutung sein kann. Besonders interessant ist dabei, dass wir die *kritische Populationsgröße*, bei der die erwartete Laufzeit von einem exponentiellen Wert auf einen polynomiellen Wert umschlägt, asymptotisch exakt als $\Theta(n^{1/2})$ ermittelt haben.

Man darf sich nun die Frage stellen, ob die kritische Populationsgröße des Beispiel-GAs auch andere, größere Werte annehmen kann oder ob sich die Laufzeit des Beispiel-GAs ab einer gewissen Populationsgröße nur noch verschlechtern kann. Aus diesem Grunde suchen wir nach einem so genannten Hierarchieresultat, d. h. nach einer Klasse von Funktionen f_k mit $k \geq 1$, sodass die Funktion f_{k+1} eine größere kritische Populationsgröße des Beispiel-GAs als f_k erfordert und damit in einem gewissen Sinne schwieriger als f_k ist. Tatsächlich wird sich herausstellen, dass die kritische Populationsgröße von f_k in der Größenordnung $\Theta(n^{k-1/2})$ liegen wird.

Um das Hierarchieresultat zu zeigen, verallgemeinern wir f mithilfe einer nahe liegenden Verallgemeinerung der Funktion LeadingOnes, die gelegentlich als LeadingOnesBlocks (siehe Jansen und Wiegand, 2003) bezeichnet wird und in einem anderen Zusammenhang als *royal staircase function* bekannt ist (van Nimwegen und Crutchfield, 2001). Die Struktur der Funktion f_k wird analog zu f sein und die Kernidee wird darin bestehen, dass der Beispiel-GA im Erwartungswert $\Theta(n^k)$ Schritte benötigt, um einen Erfolg im Suffix eines mutierten Individuums zu erreichen. Ohne weitere große Vorbereitungen können wir jetzt zur Definition von f_k kommen. Seien die Größen ℓ, b, $\mathrm{PE}(x)$ und $\mathrm{PE}^*(x)$ wie in Abschnitt 6.2.1 definiert. Sei $k \geq 1$ als konstant angenommen. Ohne Beschränkung der Allgemeinheit genüge ℓ der Eigenschaft $\ell = k \cdot \ell_k$ für ein $\ell_k \in \mathbb{N}$. Als Nächstes unterteilen wir das Suffix der Länge ℓ eines Individuums $x \in \{0,1\}^n$ in ℓ_k aufeinander folgende Blöcke mit einer Länge von jeweils k. Ein Individuum heiße genau dann *wohlgeformt*, wenn sein Suffix von der Gestalt $1^{ik}0^{\ell-ik}$ für ein $i \geq 0$ ist, d. h., dessen erste i Blöcke bestehen nur aus Einsen und der Rest nur aus Nullen. Die Anzahl **f**ührender **S**uffix-Eins**b**löcke ist gegeben durch

$$\mathrm{FSB}_k(x) := \sum_{i=1}^{\ell_k} \prod_{j=0}^{ik-1} x_{m+1+j},$$

was im Fall $k = 1$ gerade $\mathrm{FSE}(x)$ entspricht. Sei nun

$$f_k(x) := \begin{cases} \mathrm{PE}(x) + n^{2n(k \cdot \mathrm{FSB}_k(x)+1)}, & \text{falls } x \text{ wohlgeformt und } \mathrm{PE}(x) \leq \frac{2m}{3}, \\ n^{2n\ell + \mathrm{PE}^*(x)} + \mathrm{FSB}_k(x), & \text{falls } x \text{ wohlgeformt und } \frac{2m}{3} < \mathrm{PE}(x), \\ n^{1+\ell^2-\ell \sum_{i=m+1}^{n} x_i} & \text{sonst.} \end{cases}$$

Offensichtlich gilt $f_1(x) = f(x)$. Wir betrachten nun den Fall $k > 1$ und nehmen an, dass x wohlgeformt ist mit $\mathrm{PE}(x) \leq 2m/3$. Wenn $\mathrm{FSB}_k(x) < \ell_k$ gilt, ist eine Mutation von x, die mindestens k Bits gleichzeitig kippt, notwendig, um den f_k-Wert zu erhöhen; andererseits ist eine Mutation, die genau die k linkesten Nullen im Suffix kippt, dafür auch hinreichend. Die Mengen der lokal und global optimalen $x \in \{0,1\}^n$ sind dieselben wie bei der Funktion f. Demzufolge werden wir die Lemmata und Theoreme aus den Abschnitten 6.3 und 6.4 auf einfache Weise erweitern können. Zunächst verallgemeinern wir aber die Hilfsaussagen aus Abschnitt 6.2.2. Die Definition einer normalen Population sowie die Zufallsvariablen M und D seien von dort übernommen und das FSB_k-Potenzial sowie der Begriff eines FSB_k-maximalen Individuums seien analog zum FSE-Potenzial bzw. FSE-maximalen Individuum definiert. Wir erinnern daran, dass k als Konstante angenommen wird.

Lemma 6.5.1 *Sei $X = \{x_1, \ldots, x_\mu\}$ eine normale Population und seien x_i und x_j zwei wohlgeformte Individuen aus X mit $\mathrm{FSB}_k(x_i) = \mathrm{FSB}_k(x_j)$. Dann gilt*

$$1 - n^{-\Omega(n)} \;\leq\; \frac{\mathrm{Prob}(M = i)}{\mathrm{Prob}(M = j)} \;\leq\; 1 + n^{-\Omega(n)}.$$

Lemma 6.5.2 *Sei $X = \{x_1, \ldots, x_\mu\}$ eine normale und $Y = \{y_1, \ldots, y_{\mu+1}\}$ eine vergrößerte normale Population. Seien L_X und L_Y die FSB_k-Potenziale von X bzw. Y. Außerdem existiere mindestens ein wohlgeformtes $y_i \in Y$ mit $\mathrm{FSB}_k(y_i) < L_Y$. Falls $\mu = \mathrm{poly}(n)$, gelten die Beziehungen*

$$\mathrm{Prob}(\mathrm{FSB}_k(x_M) = L_X) \;=\; 1 - n^{-\Omega(n)} \quad \textit{und} \quad \mathrm{Prob}(\mathrm{FSB}_k(y_D) = L_Y) \;=\; n^{-\Omega(n)}.$$

Lemma 6.5.3 *Sei $\mu = \mathrm{poly}(n)$ und sei $X = \{x_1, \ldots, x_\mu\}$ eine normale Population mit FSB_k-Potenzial L_X. Sei zudem $B := \{x \in X \mid x \text{ wohlgeformt und } \mathrm{FSB}_k(x) = L_X\}$ die Menge der FSB_k-maximalen Individuen. Dann gilt für alle $x \in B$, dass*

$$\mathrm{Prob}(x_M = x) \;=\; \frac{1}{|B|} \pm n^{-\Omega(n)}.$$

Für $k = 1$ entsprechen die Lemmata 6.5.1–6.5.3 den Lemmata 6.2.1–6.2.3. Für größere k funktionieren die Beweise aus dem Fall $k = 1$ weiterhin, da die dort verwandten Abschätzungen für den Quotienten aus dem f_k-Wert eines nicht FSB_k-maximalen Individuums und eines FSB_k-maximalen Individuums für alle betrachteten k gültig bleiben.

Im Folgenden formulieren wir die Analoga zu den Lemmata 6.3.2–6.3.4. Der Begriff eines präfixoptimalen Individuums sei dabei aus Abschnitt 6.3 übernommen.

Lemma 6.5.4 *Sei $\mu = \mathrm{poly}(n)$ und sei x^* das erste wohlgeformte Individuum im Lauf des Beispiel-GAs auf f_k. Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-2\ell/3+o(\ell)}$ entsteht x^* nach höchstens $2e\ell n \ln(e\ell)$ Schritten, ist einziges wohlgeformtes Individuum der Population und genügt $\mathrm{PE}(x^*) \leq 7m/12$ und $\mathrm{FSB}_k(x^*) \leq 3\ell/4$. Außerdem ist die erwartete Zahl von Schritten bis zur Erzeugung von x^* höchstens $O(n \log n)$.*

Lemma 6.5.5 *Sei $\mu = \mathrm{poly}(n)$. Angenommen, die aktuelle Population des Beispiel-GAs auf f_k enthält mindestens ein wohlgeformtes Individuum und alle ihre wohlgeformten Individuen haben einen FSB_k-Wert von höchstens $3\ell/4$. Dann entsteht ein* *präfixoptimales Individuum, bevor das Optimum erreicht wird, mit einer Wahrscheinlichkeit von mindestens* $\max\{2^{-O(\mu/n^{k-1})} - 2^{-\Omega(n)}, 1 - 2^{-n^{k-1/2}/\exp(O(\mu/n^{k-1}))} - 2^{-\Omega(n^{1/2})}\}$.

Lemma 6.5.6 *Sei* $\mu = \text{poly}(n)$. *Angenommen, die aktuelle Population des Beispiel-GAs auf* f_k *enthält mindestens ein präfixoptimales und kein global optimales Individuum. Dann beträgt die Laufzeit mit einer Wahrscheinlichkeit von* $1 - 2^{-\Omega(n^{1/2})}$ *mindestens* $2^{\Omega(n^{1/2})}$.

Für den Beweis von Lemma 6.5.4 ist gegenüber dem Beweis von Lemma 6.3.2 nur eine Änderung vonnöten. Die Wahrscheinlichkeit, dass ein initiales Individuum ein wohlgeformtes Suffix besitzt, ist nun durch $(\ell_k + 1)2^{-\ell}$ gegeben. Wegen $\ell_k \leq \ell$ funktioniert aber wiederum die Abschätzung aus dem Beweis von Lemma 6.3.2.

Um Lemma 6.5.5 zu zeigen, nehmen wir am Beweis von Lemma 6.3.3 folgende Änderungen vor. An allen Stellen, die FSE-Werte verwenden, sind entsprechende FSB_k-Werte zu ersetzen. Die Wahrscheinlichkeit eines schlechten Schrittes ist nun sogar durch $1/n^k$ nach oben beschränkt, da die k linkesten Nullen im Suffix kippen müssen, um den FSB_k-Wert zu erhöhen. Somit hat eine Phase der Länge $s(\mu)$ das Attribut, gut zu sein, mit einer Wahrscheinlichkeit von mindestens $(1 - 1/n^k)^{\mu(n-1)/c} \geq e^{-\mu/(cn^{k-1})}$, womit die erste Aussage des Lemmas gezeigt ist. Für die zweite Aussage redefinieren wir $p(\mu) := \lceil \ell n^{k-1} c/(8\mu) \rceil$ und erhalten, dass $p(\mu)$ aufeinander folgende Phasen mit einer Wahrscheinlichkeit von höchstens $2^{-n^{k-1/2}/\exp(O(\mu/n^{k-1}))}$ allesamt mindestens einen schlechten Schritt enthalten. Durch Kürzen mit n^{k-1} ergibt sich wiederum dieselbe Abschätzung dafür, dass diese Phasen insgesamt höchstens $\ell/6$ schlechte Schritte enthalten. Damit folgt auch die zweite Aussage des Lemmas.

Zu guter Letzt kann der Beweis von Lemma 6.3.4 unmittelbar als Beweis für Lemma 6.5.6 eingesetzt werden, da wir hier gar nicht mit dem Einfluss des FSE-Wertes argumentieren. Schließlich erhalten wir folgende Erweiterung von Theorem 6.3.1.

Theorem 6.5.7 *Sei* $\mu = O(1)$. *Mit einer Wahrscheinlichkeit von* $1 - 2^{-\Omega(n^{1/2})}$ *ist die Laufzeit des Beispiel-GAs auf* f_k *dann mindestens* $2^{\Omega(n^{1/2})}$.

Der Beweis von Theorem 6.5.7 wird mithilfe der neuen Lemmata 6.5.4–6.5.6 analog zum Beweis von Theorem 6.3.1 (zu finden ab Seite 113) geführt. Auch ein Analogon zu Theorem 6.3.5 folgt unmittelbar.

Theorem 6.5.8 *Es gibt eine Konstante* $c > 0$, *sodass für* $\mu \leq cn^{k-1} \ln n$ *folgende Aussage gilt: Mit einer Wahrscheinlichkeit von mindestens* $1 - 2^{-\Omega(n^{1/4})}$ *ist die Laufzeit des Beispiel-GAs auf* f_k *mindestens* $2^{\Omega(n^{1/2})}$.

Für den Beweis von Theorem 6.5.9 ziehen wir den Beweis von Theorem 6.3.5 heran, setzen dort die neuen Lemmata 6.5.4–6.5.6 ein und schätzen nur an einer Stelle etwas anders ab. Für genügend kleines c ist $e^{O(\mu/n^{k-1})} \leq n^{1/4}$. Ebenso gilt folgendes Analogon zu Theorem 6.3.6, das uns eine untere Schranke für die erwartete Laufzeit angibt.

Theorem 6.5.9 *Es gibt eine Konstante $c > 0$, sodass für $\mu \leq cn^{k-1/2}$ die erwartete Laufzeit des Beispiel-GAs auf f_k mindestens $2^{\Omega(n^{1/2})}$ ist.*

Für den Beweis von Theorem 6.5.9 greifen wir auf den Beweis von Theorem 6.3.6 zurück, setzen dort erneut die neuen Lemmata ein und schätzen an einer Stelle etwas anders ab. Nach der Anwendung von Lemma 6.5.5 folgt, dass die Laufzeit mit einer Wahrscheinlichkeit von mindestens $2^{-O(\mu/n^{k-1})} - 2^{-\Omega(n^{1/2})}$ mindestens $2^{\Omega(n^{1/2})}$ beträgt. Bei genügend kleiner Wahl von c erhalten wir wiederum den Term $2^{\Omega(n^{1/2})}$.

Auch der Beweis, dass Werte von $\mu \geq cn^{k-1/2}$ für geeignete Konstanten $c > 0$ eine polynomielle erwartete Laufzeit ermöglichen, weist starke Analogien zu den Überlegungen aus den vorigen Abschnitten auf.

Theorem 6.5.10 *Sei $\mu \geq 45n^{k-1/2}$ und $\mu = \mathrm{poly}(n)$. Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n^{1/2})}$ ist die Laufzeit des Beispiel-GAs auf f_k dann höchstens $5en^k\ell\ln(e\ell) + 4\mu n$. Die erwartete Laufzeit ist ebenfalls $O(n^{k+1/2}\log n + \mu n)$.*

Zum Beweis von Theorem 6.5.10 ersetzen wir im Beweis von Theorem 6.4.1 (zu finden auf Seite 121 ff.) alle vorkommenden FSE-Werte durch FSB_k-Werte. Um die so genannte zweite Eigenschaft zu zeigen, schätzen wir die Wahrscheinlichkeit, den FSB_k-Wert zu erhöhen, durch $1/(en^k)$ nach unten ab, da es hinreichend ist, die k linkesten Nullen im Suffix zu kippen. Mit der neuen Phasenlänge $s := \lceil 3e\ell n^k \rceil$ folgt die zweite Eigenschaft mit einer Erfolgswahrscheinlichkeit von erneut mindestens $1 - e^{-2\ell/3+o(\ell)}$. Zudem spielt die neue Phasenlänge eine Rolle bei der Anwendung von Lemma 6.4.7 und Lemma 6.4.6. Bei der Rechnung bezüglich des ersten Lemmas haben sich sowohl der Wert s als auch die untere Schranke $u := 45n^{k-1/2}$ (abgesehen von Gaußklammern) um denselben Faktor n^{k-1} erhöht. Da $\mu = \mathrm{poly}(n)$, erhalten wir immer noch eine obere Schranke von $2^{-\Omega(n)}$ für die Wahrscheinlichkeit, dass die Behauptung über die PE-Werte nicht zutrifft. Der neue Wert von t bei der Anwendung des zweiten Lemmas ist nun $t := s = \lceil 3e\ell n^k \rceil$. Also ist $4t/u$ (abgesehen von Gaußklammern) wiederum durch denselben Ausdruck wie im Beweis von Theorem 6.4.1 beschränkt. Damit gilt für genügend großes n immer noch $4t/u + 3\ell H_u \leq n/36 - 1$, denn es ist $H_u = O(\log n)$ wegen $\mu = \mathrm{poly}(n)$.

Dieselbe Art von Abschätzungen wie im Beweis von Theorem 6.4.1 liefert dann die obere Schranke $5e\mu\ell n^k \ln(e\ell) + 4\mu n$ für die Laufzeit, die mit exponentiell nahe an 1 liegender Wahrscheinlichkeit angenommen wird. Bei der Abschätzung der erwarteten Laufzeit schließlich ist eine neue Schranke $O(\mu\ell n^k)$ für die erwartete Zeit bis zur Maximierung des L-Potenzials ausreichend. Alle übrigen Abschätzungen bleiben asymptotisch unberührt, da $\mu = \mathrm{poly}(n)$ gilt.

Wir haben im Beweis von Theorem 6.5.10 u. a. davon profitiert, dass wir in Abschnitt 6.4.3 einigen Aufwand bei der Abschätzung der Tiefe von $1/u$-Bäumen betrieben haben. Die für unsere Überlegungen entscheidende obere Schranke $O(s/u + \ell H_u)$ für die Tiefe von $T_s(x)$ ließe es bei noch sorgfältigeren Abschätzungen in den Beweisen von Theorem 6.5.7 und Theorem 6.5.10 (einschließlich des Falls $k = 1$) sogar zu, mit einer Suffixlänge $\ell = \Theta(n/\log n)$ zu arbeiten. Damit verringerten sich die Fehlerwahrscheinlichkeiten in den Theoremen von $2^{-\Omega(n^{1/2})}$ auf $2^{-\Omega(n/\log n)}$. Eine so genannte echt exponentiell kleine Fehlerwahrscheinlichkeit von $2^{-\Omega(n)}$ ließe sich mit unseren Methoden aber allenfalls dann zeigen, wenn die Definition von f dahin gehend modifiziert würde, dass $\ell = \Omega(n)$ wäre. In diesem Fall aber träten Probleme beim Beweis einer oberen Laufzeitschranke auf, welche darin begründet sind, dass ein ZRB mit n Knoten bereits eine erwartete Tiefe von $\Theta(\log n)$ besitzt und damit die Summe der Tiefen von $\ell = \Theta(n)$ Bäumen durch einen unzureichend großen Ausdruck $O(n \log n)$ beschränkt wäre. Da $\ell = \Theta(n)$ nicht zu funktionieren scheint, haben wir uns letztendlich für die Wahl $\ell = \Theta(n^{1/2})$ entschieden, da diese die Theoreme und ihre Beweise etwas übersichtlicher gestaltet.

6.6 Ein Beispiel für den Schaden durch Populationen

Die Funktion f und die Funktionen f_k aus den vorangehenden Abschnitten dienten dazu, rigoros zu beweisen, dass eine Erhöhung der Populationsgröße in einem evolutionären Algorithmus ohne Rekombination dazu beitragen kann, die (erwartete) Laufzeit drastisch zu senken. Es ist nun nahe liegend, danach zu fragen, ob eine Erhöhung der Populationsgröße auch das umgekehrte Verhalten zur Folge haben kann, d. h. die Laufzeit von einem polynomiellen auf einen exponentiellen Wert steigern kann. Informal gesagt stellen wir die Frage, ob eine Population schaden kann.

In diesem Abschnitt wollen wir eine Beispielfunktion vorstellen, bei der die Population des Beispiel-GAs schadet. Ihre Idee basiert wiederum auf der in Abschnitt 6.2.1 bei der Motivation von f beschriebenen Kombination zweier „verschieden schneller“ Optimierungsprozesse. Tatsächlich wird in der folgenden Funktion $g_c\colon \{0,1\}^n \to \mathbb{R}$ gegenüber f im Wesentlichen die Rolle lokaler und globaler Optima vertauscht sein. Um zu vermeiden, bei der kommenden Analyse der erwarteten Laufzeit erneut konstante Faktoren in Fehlertermen abschätzen zu müssen, verändern wir nur die Lage lokaler Optima gegenüber Abschnitt 6.2.1. Sei $c \geq 1$ eine Konstante. Wir redefinieren

$$\mathrm{PE}^*(x) := \frac{2m}{3} + \left\lceil \frac{b}{c} \right\rceil - \left| \frac{2m}{3} + \left\lceil \frac{b}{c} \right\rceil - \mathrm{PE}(x) \right|.$$

Die Definition eines wohlgeformten Individuums sowie die übrigen Notationen seien aus Abschnitt 6.2.1 übernommen. Damit definieren wir

$$g_c(x) := \begin{cases} \mathrm{PE}(x) + n^{2n(\mathrm{FSE}(x)+1)}, & x \text{ wohlgeformt und } \mathrm{PE}(x) \leq \frac{2m}{3}, \\ n^{2n\ell + \mathrm{PE}^*(x)} + \mathrm{FSE}(x), & x \text{ wohlgeformt und } \frac{2m}{3} < \mathrm{PE}(x) < \frac{2m}{3} + \lceil \frac{b}{c} \rceil, \\ n^{2n(\ell+2)}, & x \text{ wohlgeformt und } \mathrm{PE}(x) = \frac{2m}{3} + \lceil \frac{b}{c} \rceil, \\ n^{1+\ell^2 - \ell \sum_{i=m+1}^{n} x_i} & \text{sonst.} \end{cases}$$

Für Eingaben x, die nicht wohlgeformt sind, gilt also $f(x) = g_c(x)$ unabhängig von c. Auch im ersten Fall der Definition von g_c, d. h. für wohlgeformte x mit $\mathrm{PE}(x) \leq$

$2m/3$, gleichen die Funktionswerte einander. Darüber hinaus gilt wegen $c \geq 1$ sogar $f(x) = g_c(x)$ im Falle, dass $\mathrm{PE}(x) \leq 2m/3 + \lceil b/c \rceil$ ist. Für wohlgeformte x mit $\mathrm{PE}(x) = 2m/3 + \lceil b/c \rceil$ zeigt der dritte Fall der Definition von g_c aber den entscheidenden Unterschied zu f auf: Solche Eingaben sind global optimal mit einem g_c-Wert von $n^{2n(\ell+2)}$. Wohlgeformte x mit $\mathrm{PE}(x) = 2m/3$ und $\mathrm{FSE}(x) = \ell$ führen nur zum zweitbesten Funktionswert von $2m/3 + n^{2n(\ell+1)}$ und sind zudem lokal optimal.

Es erscheint nun nicht zu abwegig, dass der Beispiel-GA mit $\mu = O(1)$ mit hoher Wahrscheinlichkeit in kurzer Zeit den PE-Wert auf $2m/3 + \lceil b/c \rceil$ erhöht, bevor jemals ein FSE-Wert von ℓ entsteht. Dieses erfasst das folgende Theorem.

Theorem 6.6.1 *Sei $\mu = O(1)$ und $c \geq 1$ eine Konstante. Mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ ist die Laufzeit des Beispiel-GAs auf g_c dann $O(n^{3/2} \log n)$. Für genügend großes c ist auch die erwartete Laufzeit $O(n^{3/2} \log n)$.*

Beweis: Wir argumentieren zunächst wie im Beweis von Theorem 6.3.1 und machen uns zunutze, dass f und g_c auf Individuen mit PE-Werten von weniger als $2m/3 + \lceil b/c \rceil$ und auf nicht wohlgeformten Individuen übereinstimmen. Mit Lemma 6.3.2 können wir annehmen, dass nach $O(\ell n \log n)$ Schritten ein wohlgeformtes Individuum mit einem FSE-Wert von höchstens $3\ell/4$ entsteht, und führen damit einen Term $2^{-\Omega(n^{1/2})}$ für die Fehlerwahrscheinlichkeit ein. Weil $c \geq 1$ angenommen wird, liefert die Analyse aus dem Beweis von Lemma 6.3.3, dass mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ nach $O(\ell)$ Phasen der Länge $O(n)$, d. h. nach $O(n^{3/2})$ Schritten, ein wohlgeformtes Individuum mit einem PE-Wert von $2m/3 + \lceil b/c \rceil$ entsteht. Da dieses global optimal ist, beschließt dies den Beweis der ersten Aussage des Theorems. Im Folgenden heiße ein Lauf mit den bis jetzt beschriebenen Eigenschaften *typisch*.

Um die Behauptung über die erwartete Laufzeit zu zeigen, ziehen wir ähnliche Ideen wie im entsprechenden Abschnitt im Beweis von Theorem 6.4.1 heran. Als Erstes halten wir fest, dass laut Lemma 6.3.2 nach $O(n \log n)$ erwarteten Schritten ein wohlgeformtes Individuum entsteht. Wenn nur normale Populationen, d. h. ohne wohlgeformte Individuen mit einem PE-Wert von mehr als $2m/3$, entstehen, können wir wieder mit dem in Abschnitt 6.2.2 definierten FSE-Potenzial arbeiten. Wir nehmen im Folgenden an, dass nur normale Populationen entstehen, da wir sonst mit den Argumenten aus dem ersten Absatz die erwartete Laufzeit durch $O(n^{3/2} \log n)$ abschätzen können. Die erwartete Zeit, bis ein FSE-Potenzial von ℓ erreicht wird, beträgt dann nach Lemma 5.2.1 und mit den mittlerweile bekannten Abschätzungen höchstens $O(\mu \ell n) = O(\ell n)$. Anschließend betrachten wir das Potenzial P aus dem Beweis von Theorem 6.4.1. Nach einer erwarteten Zeit von $O(\mu n) = O(n)$ ist dieses auf $2m/3$ gestiegen, sodass sich ein lokal optimales Individuum x mit $\mathrm{PE}(x) = 2m/3$ und $\mathrm{FSE}(x) = \ell$ in der Population befindet. Die Wahrscheinlichkeit, ein solches Individuum zur Mutation zu wählen und in ein global optimales zu mutieren, ist dann durch

$$\frac{1}{\mu}\left(\frac{1}{n}\right)^{b/c+1}\left(1-\frac{1}{n}\right)^{n-b/c} = 2^{-b(\log n)/c-O(1)} = 2^{-\ell/(2c)-O(1)}$$

nach unten beschränkt und die erwartete Zeit bis zu einem solchen Schritt ist höchstens $2^{\ell/(2c)+O(1)}$. Also ist die erwartete Laufzeit des Beispiel-GAs auf g_c höchstens $\text{poly}(n) + 2^{\ell/(2c)+O(1)}$, wenn kein typischer Lauf beobachtet wird. Wenn wir c groß genug wählen, ist das Produkt aus dieser oberen Schranke und der Schranke $2^{-\Omega(n^{1/2})}$ für die im ersten Absatz betrachteten Fehlerwahrscheinlichkeiten insgesamt $o(1)$. Somit folgt die Aussage über die erwartete Laufzeit insgesamt mit dem Satz von der totalen Wahrscheinlichkeit. □

Mit Theorem 6.6.1 haben wir gezeigt, dass der Beispiel-GA auf g_c bei einer kleinen Population effizient ist. Des Weiteren gilt diese Aussage für den (1+1)-EA, da er bekanntlich aus dem Beispiel-GA mit $\mu = 1$ hervorgeht. Es bleibt nun nachzuweisen, dass große Populationen schädlich sind, d. h. mit überwältigender Wahrscheinlichkeit exponentielle Laufzeiten auftreten.

Theorem 6.6.2 *Seien $\mu \geq 45n^{1/2}$, $\mu = \text{poly}(n)$ und $c \geq 1$ eine beliebige Konstante. Mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ ist die Laufzeit des Beispiel-GAs auf g_c dann mindestens $2^{\Omega(n^{1/2})}$.*

Beweis: Die entscheidende Beobachtung besteht erneut darin, dass der Beispiel-GA sich auf f und g_c gleich verhält, solange kein so genannter Fehler der Gestalt, dass ein Individuum mit wohlgeformten Suffix und mit einem PE-Wert von mehr als $2m/3$ zur Mutation gewählt wird, eintritt. Wir nehmen uns den Beweis von Theorem 6.4.1 vor. Dort haben wir gezeigt, dass der Beispiel-GA mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2})}$ ein wohlgeformtes Individuum mit FSE-Wert ℓ erzeugt, ohne dass ein Fehler passiert. Wir nehmen nun an, dass ein solches Individuum erzeugt worden ist und kein Fehler eingetreten ist. Nun können wir ein ähnliches Argument wie im Beweis von Lemma 6.3.4 anbringen.

Nur Individuen mit einem PE-Wert von $2m/3 + \lceil b/c \rceil$ sind besser als wohlgeformte Individuen x mit $\text{FSE}(x) = \ell$ und $\text{PE}(x) \leq 2m/3$. Wir nennen letztere Individuen *suffixoptimal*. Aufgrund des Elitismus des Beispiel-GAs enthält die aktuelle Population bis zum Erreichen des Optimums stets ein suffixoptimales Individuum. Die Wahrscheinlichkeit, ein global optimales Individuum mit einer direkten Mutation eines suffixoptimalen Individuums zu erzeugen, ist höchstens $1/\lceil b/c \rceil ! = 2^{-\Omega(n^{1/2})}$ aufgrund der stirlingschen Formel und da c konstant ist. Andererseits ist in der vorliegenden Situation die Wahrscheinlichkeit, ein Individuum, das nicht suffixoptimal ist, zur Mutation zu wählen, aufgrund der Definition von g_c höchstens

$$\mu \cdot \frac{n^{2n\ell+2m/3+b-1}}{n^{2n(\ell+1)}} = n^{-\Omega(n)} = 2^{-\Omega(n \log n)}.$$

Damit ist die Wartezeit bis zum Erreichen des Optimums mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n^{1/2})}$ durch $2^{\Omega(n^{1/2})}$ nach unten beschränkt. □

Bemerkung: Natürlich drängt sich die Frage nach der kritischen Populationsgröße, bei der die erwartete Laufzeit des Beispiel-GAs auf g_c von polynomiellen auf exponentielle Werte umschlägt, auf. Insbesondere ist es interessant, die erwartete Laufzeit im Fall $\mu = \omega(1)$ zu untersuchen. In diesem Fall scheint aber aus folgenden Gründen keine polynomielle erwartete Laufzeit mehr einzutreten. Seien die im Beweis von Lemma 6.3.3 als schlecht bezeichneten Schritte erneut betrachtet. Der FSE-Wert im ersten Fall der Definition von g_c (und auch f) geht exponentiell in den Funktionswert ein. Damit scheint die Verteilung der Zahl der Einsen im Präfix von Individuen, deren PE-Wert nicht dem Potenzial P entspricht, „fast“ binomialverteilt zu den Parametern m und $1/2$ zu sein. Wenn ein schlechter Schritt also nicht gerade ein Individuum mit aktuell maximalem PE-Wert wählt, ist nach dem schlechten Schritt also das neue P-Potenzial mit hoher Wahrscheinlichkeit nahe bei $m/2$. Wenn nun $\mu = \omega(1)$ ist, scheint ein schlechter Schritt also mit gegen 1 konvergierender Wahrscheinlichkeit ein Individuum zu erzeugen, dessen PE-Wert nahe bei $m/2$ ist. Mit anderen Worten scheint mit gegen 1 konvergierender Wahrscheinlichkeit tatsächlich eine gute Phase erforderlich zu sein, um den PE-Wert auf $2m/3 + \lceil b/c \rceil$ zu bringen. Wir wissen bereits, dass die Wahrscheinlichkeit einer guten Phase gegen 0 konvergiert, falls $\mu = \omega(1)$ ist.

Nichtsdestoweniger scheint es möglich zu sein, wiederum und analog zu Abschnitt 6.5 eine Hierarchie von Funktionen $g_{c,k}$ für konstante $k \geq 1$ aufzustellen, sodass der Beispiel-GA mit $\mu = O(n^{k-1})$ effizient ist, während er bei $\mu \geq 45n^{k-1/2}$ bereits extrem ineffizient ist.

6.7 Fazit

In diesem Kapitel haben wir anhand von Beispielen untersucht, inwieweit die Population eines evolutionären Algorithmus ohne Rekombinationsoperator dazu beitragen kann, die Laufzeit zu beeinflussen. Zum ersten Mal ist es gelungen, in diesem Szenario eine exponentiell große Lücke zu zeigen. Für konstante Populationsgrößen ist die Laufzeit unseres Algorithmus und auch des (1+1)-EA mit exponentiell nahe an 1 liegender Wahrscheinlichkeit exponentiell groß. Dies schließt auch eine polynomielle erwartete Laufzeit von parallelen Läufen der Algorithmen aus. Auf der anderen Seite lässt eine moderate Populationsgröße von $45\sqrt{n}$ bereits eine polynomielle erwartete Laufzeit zu. Gegenüber dem Ergebnis von Jansen und Wegener (2001b) zeigt unser Ergebnis nicht nur einen exponentiellen Laufzeitvorteil anstelle eines superpolynomiellen Vorteils, sondern zeichnet sich auch dadurch aus, dass die exponentielle Laufzeit nachweisbar nicht nur bei einer Populationsgröße von 1 eintritt.

Die Definition der Beispielfunktion, zu deren Optimierung eine Population von Nutzen ist, basiert auf einer recht allgemeinen Idee. Das Konzept beruht darauf, zwei Funktionen f_1 und f_2 zu identifizieren, sodass ein EA bei kleiner Populationsgröße deutlich schneller f_1 als f_2 optimiert, während dieser EA mit einer großen Population das umgekehrte Verhalten zeigt. Diese Funktionen werden dann zu einer neuen

Funktion f zusammengesetzt, die den EA leicht das globale Optimum erreichen lässt, wenn der f_2-Bestandteil vor dem f_1-Bestandteil optimiert wird, und mit hoher Wahrscheinlichkeit in einem lokalen Optimum stecken bleiben lässt, wenn er sich umgekehrt verhält.

Dieses neue Konzept, um exponentielle Laufzeitvorteile zu beweisen, gestattet es, durch Manipulationen der Komponenten f_1 und f_2 so genannte Hierarchieresultate zu zeigen. In Abschnitt 6.5 haben wir daher eine Klasse von Funktionen f_k definiert, sodass erst Populationen der Größe $\Omega(n^{k-1/2})$ eine polynomielle erwartete Laufzeit zulassen. Zudem ermöglichte eine weitere einfache Manipulation der Komponenten den Beweis, dass eine große Population in dem Sinne schädlich sein kann, als sich die erwartete Laufzeit auf einen exponentiellen Wert steigert. Alles in allem führt das neue Konzept zu Funktionen, die auf einer großen Teilmenge des Definitionsbereichs zumindest teilweise separierbar sind, was die Hoffnung nährt, mit den theoretischen Resultaten die Bedeutung der Populationsgröße auch für praktisch relevante separierbare Funktionen unterstreichen zu können. Jedenfalls haben wir insoweit einen Beitrag zum Entwurf evolutionärer Algorithmen geleistet, als wir die grundsätzliche Bedeutung der Populationsgröße unter Beweis gestellt haben. Zuletzt hat das geschilderte Beweiskonzept übrigens auch Anwendung gefunden, um einen exponentiellen Laufzeitvorteil eines koevolutionären Algorithmus gegenüber dem (1+1)-EA theoretisch zu begründen (Jansen und Wiegand, 2005).

Um zu beweisen, dass eine große Population auf der genannten Funktion f hilfreich ist, mussten wir implizit nachweisen, dass die Optimierungszeit der Komponente f_1 mit der Populationsgröße wächst. Zu diesem Zweck wurde erstmalig die Tiefe der Stammbäume von Individuen abgeschätzt und anhand einer Schranke für die Tiefe eine Aussage über die Laufzeit abgeleitet. Als äußerst nützlich erwiesen sich dabei bereits aus der Wahrscheinlichkeitstheorie bekannte Modelle zufälliger Bäume, beispielsweise zufälliger rekursiver Bäume, welche in Abwandlungen Teile von Stammbäumen zu modellieren vermochten. Die mit diesen Methoden gefundenen Tiefenbeschränkungen für Stammbäume sind so genau, dass wir einerseits die kritische Populationsgröße, bei der die erwartete Laufzeit des Beispiel-GAs auf der Funktion f von einem exponentiellen auf einen polynomiellen Wert wechselt, asymptotisch exakt festlegen konnten. Zum anderen trugen die aufwendigen Abschätzungen dazu bei, das genannte Hierarchieresultat und das Beispiel für den Schaden durch Populationen ohne weitere große Mühen zeigen zu können. Im folgenden Kapitel werden wir unter anderem weitere Überlegungen zu Stammbäumen anstellen.

7 Die Analyse einer $(\mu+1)$-Evolutionsstrategie

Nachdem die hauptsächliche Motivation des letzten Kapitels darin bestand, den Nutzen von Populationen in evolutionären Algorithmen nachzuweisen, wollen wir in diesem Kapitel eher das Verhalten vorgegebener populationsbasierter evolutionärer Algorithmen studieren. Der Beispiel-GA aus Definition 6.1.1 kann zwar als typischer genetischer Algorithmus ohne Rekombination verstanden werden; gleichwohl wurde der Algorithmus doch ausgesucht, um im Zusammenspiel mit der Beispielfunktion f den gewünschten Nachweis des Nutzens von Populationen erbringen zu können.

In diesem Kapitel wollen wir populationsbasierte evolutionäre Algorithmen aus einem anderen Blickwinkel betrachten. Wie in Kapitel 2 erwähnt, existieren seit Jahrzehnten verschiedene Typen evolutionärer Algorithmen, deren Gemeinsamkeit darin besteht, mit einer echten Population, d. h. mit einer Population mit mehr als einem Element, arbeiten zu können. Der (1+1)-EA kann als ein Spezialfall verschiedener populationsbasierter EAs, deren Populationsgröße auf 1 gesetzt ist, aufgefasst werden. Während bei der Laufzeitanalyse des (1+1)-EA in den vergangenen Jahren bedeutsame Fortschritte erzielt wurden, sind bislang wenige Aussagen über den funktionalen Zusammenhang zwischen der Populationsgröße und der (erwarteten) Laufzeit bekannt. Mit anderen Worten suchen wir Laufzeitschranken, in die sowohl die Populationsgröße μ als auch die Problemdimension n einfließen. Damit soll ein weiterer Schritt auf dem Weg zu einer Theorie über die Zeitkomplexität evolutionärer Algorithmen gemacht werden. Wenn die gewünschten Zusammenhänge zwischen Populationsgröße und Laufzeit erhellt werden können, geben uns diese nicht nur Aufschluss darüber, ob eine Population nutzen kann, sondern ggf. auch, in welchem Umfang sie hilfreich ist.

7.1 Einordnung von $(\mu+\lambda)$-Evolutionsstrategien

Zu den ursprünglich konkurrierenden Paradigmen evolutionärer Algorithmen zählen die um das Jahr 1970 entwickelten Evolutionsstrategien. Ohne näher auf den technischen und historischen Hintergrund eingehen zu wollen, lässt sich festhalten, dass Evolutionsstrategien seit jeher hauptsächlich zur Optimierung reellwertiger Funktionen in reellwertigen Suchräumen, d. h. Funktionen der Gestalt $f\colon \mathbb{R}^n \to \mathbb{R}$, angewandt werden. Tatsächlich arbeiteten frühe Instanzen von Evolutionsstrategien mit degenerierten Populationen der Größe 1. Allerdings schlug bereits Rechenberg (1973) eine so genannte Steady-State-Evolutionsstrategie mit dem Namen $(\mu+1)$-ES vor, die auf einer echten Population basiert und sowohl Mutation als auch Rekombination benutzt. Der entscheidende Unterschied zum Beispiel-GA aus Definition 6.1.1 besteht darin, dass die Menge zu rekombinierender und zu mutierender Individuen hier gleichverteilt aus der Population ausgesucht wird. Auch die Selektion zur Ersetzung, d. h. die Bestimmung der Nachfolgepopulation, folgt einem anderen Grundgedanken. Hier wird ein Individuum mit schlechtestem Funktionswert aus der vergrößerten Population, d. h. der Vereinigung der Vorgängerpopulation mit dem neu erzeugten Individuum, gelöscht, sodass außer im Falle mehrerer schlechtester Individuen eine deterministische Auswahl erfolgt.

In allgemeinen so genannten (μ+λ)-Evolutionsstrategien ist eine feste Populationsgröße μ vorgegeben. Pro Generation wird dann λ-mal ein Nachfahre erzeugt, indem man gleichverteilt ein Individuum aus der aktuellen Population wählt und mutiert. Aus der vergrößerten Population mit $\mu + \lambda$ Individuen sind dann μ Individuen mit bestem Funktionswert beizubehalten. Nur erwähnen wollen wir, dass Evolutionsstrategien auch Rekombinationsoperatoren verwenden können und einen abgewandelten Selektionsmechanismus, die so genannte Kommaselektion, enthalten können. Im Rahmen dieser Arbeit beschränken wir uns auf die ältesten und einfachsten Varianten, nämlich die (μ+λ)-Evolutionsstrategien. Um zu ermitteln, wann diese den einfachen randomisierten Suchheuristiken aus Teil I dieser Arbeit überlegen sind, sind Aussagen über die erwartete Laufzeit solcher (μ+λ)-Strategien auf gegebenen Problemen gewünscht. Aus historischer Sicht wird der Nutzen der Elternpopulation der Größe μ mit mindestens zwei Argumenten motiviert. Zum einen erhofft man sich, oft eine genügend diverse Population vorzuhalten, die Individuen aus verschiedenen Regionen des Suchraums enthält. Damit besteht die Chance, dass, obschon manche Individuen lokal optimal sind, Individuen aus anderen Regionen des Suchraums verhindern, dass der EA für lange Zeit in einem lokalen Optimum stecken bleibt. Zum anderen kam für Evolutionsstrategien in reellwertigen Suchräumen recht früh die Idee auf, verschiedenen Individuen verschiedene so genannte *Strategieparameter* mitzugeben. Als beispielhaften Strategieparameter können wir die Mutationsstärke, kodiert durch Varianzen der Standardnormalverteilung, nennen, siehe z. B. Beyer und Schwefel (2002). Mithin versucht man verschiedene Mutationswahrscheinlichkeiten parallel auszuprobieren. Die Nachkommenpopulation der Größe λ schließlich soll dazu dienen, einen so genannten *Selektionsdruck* zu erzeugen, d. h. pro Generation einen vorteilhaften Fortschritt zum Optimum zu erzielen. Darüber hinaus lässt sich das Verfahren, die λ Nachfahren zu erzeugen, in üblichen Evolutionsstrategien leicht parallelisieren.

Jansen und De Jong (2002) haben erste Laufzeitanalysen für einen (1+λ)-EA auf pseudobooleschen Funktionen durchgeführt. In ihrer Arbeit übertragen sie den bereits bei der Analyse des (1+1)-EA eingeschlagenen Ansatz, das Verhalten der Suchheuristik auf einfachen Beispielfunktionen zu analysieren, auf den (1+λ)-EA. Da sich herausstellt, dass die Wahl $\lambda = 1$ auf den einfachen unimodalen (vgl. Definition 3.1.1) Beispielfunktionen ONEMAX und LEADINGONES zu asymptotisch optimalen Laufzeiten führt, definieren die Autoren des Weiteren eine Beispielfunktion mit speziellen

Eigenschaften. Auf der Beispielfunktion benötigt der (1+1)-EA mit exponentiell nahe an 1 liegender Wahrscheinlichkeit eine superpolynomielle Optimierungszeit, während der (1+λ)-EA bei einer polynomiellen Populationsgröße mit hoher Wahrscheinlichkeit mit einer polynomiellen Zeit auskommt. Außerdem sei noch einmal der Aspekt der Parallelisierung angesprochen. Während auf einfachen Funktionen eine Erhöhung von λ zwar nicht die erwartete Zahl der Zielfunktionsauswertungen senken kann, verringert sich aber oft die Zahl der parallelisierbaren Generationen, die zur Optimierung benötigt werden. Ganz konkret kann es sein, dass der (1+1)-EA beispielsweise auf LeadingOnes eine Mutation mit Wahrscheinlichkeit $\Theta(1/n)$ durchführen muss, um den Zielfunktionswert zu erhöhen. Die erwartete Zeit dafür beträgt $\Theta(n)$. Andererseits folgt mit einfachen Argumenten, dass im Falle $\lambda = \Theta(n)$ unabhängig erzeugter Nachfahren mit konstanter Wahrscheinlichkeit mindestens einmal die gewünschte Mutation eintritt, sodass sich die erwartete Zahl der Generationen zur Erhöhung des Funktionswertes auf $O(1)$ verringert. Ähnliche Effekte lassen sich beobachten, wenn man einen (1+λ)-EA auf einfachen Ising-Modellen (Fischer und Wegener, 2004) und zur Berechnung minimaler Spannbäume (Neumann und Wegener, 2004) heranzieht.

Es ist nun nahe liegend, auch bei der Laufzeitanalyse von (μ+λ)-EAs dem im letzten Absatz beschriebenen Muster zu folgen. Storch (2004) untersucht eine Variante eines (μ+1)-EAs mit einem Operator zur Diversitätserhaltung und erzielt damit Hierarchieresultate bezüglich der Populationsgröße. Allerdings konzentrieren wir uns auf traditionelle (μ+λ)-EAs, in denen solche Operatoren nicht auftauchen, und sind, wie oben erwähnt, am funktionalen Zusammenhang zwischen μ, λ, n und der Laufzeit interessiert. In diese Richtung versuchen He und Yao (2002) zu gehen, aber betrachten letztendlich nur geeignet gewählte Varianten von (μ+μ)-EAs. Insbesondere beschränken sie sich häufig auf lokale Varianten eines (μ+μ)-EAs und zeigen lediglich obere Schranken für ihre erwartete Laufzeit. Im Gegensatz dazu wollen wir hier global suchende (μ+λ)-EAs, die als Verallgemeinerung des (1+1)-EA angesehen werden dürfen, analysieren und obere und untere Schranken für ihre erwartete Laufzeit herleiten. Da zu diesem Zweck zum Teil neue Analysemethoden entwickelt werden müssen, beschränken wir uns hier auf den Fall $\lambda = 1$ und definieren den folgenden (μ+1)-EA.

Definition 7.1.1 ((μ+1)-EA) *Es seien alle in dieser Definition auftretenden Mengen Multimengen. Der* (μ+1)-EA *für* $f\colon \{0,1\}^n \to \mathbb{R}$ *läuft wie folgt ab.*

Setze $t := 0$. *Initialisiere* $x_i \in \{0,1\}^n$ *für* $1 \le i \le \mu$ *unabhängig und zufällig gleichverteilt. Setze* $X^{(0)} := \{x_1, \dots, x_\mu\}$.

Wiederhole:

Wähle $y \in X^{(t)}$ *zufällig gleichverteilt.*

Erzeuge $x_{\mu+1}$ *aus* y *durch* $1/n$*-Standardmutation.*

Wähle $y \in \operatorname{argmin}\bigl\{f(x) \mid x \in X^{(t)} \cup \{x_{\mu+1}\}\bigr\}$ *zufällig gleichverteilt und setze* $X^{(t+1)} := \bigl(X^{(t)} \cup \{x_{\mu+1}\}\bigr) \setminus \{y\}$ *und* $t := t+1$.

Offensichtlich ist der vorgenannte (μ+1)-EA ein Ein-Kind-EA im Sinne von Definition 5.1.1. Man beachte, dass der (μ+1)-EA das zu löschende Individuum sogar deterministisch wählt, falls das Individuum mit geringstem f-Wert eindeutig bestimmt ist. Weiterhin beobachten wir, dass der f-Wert von Individuen nur die Selektion zur Ersetzung betrifft, nicht aber die Selektion zur Reproduktion. Die gleichverteilte Wahl des zu mutierenden Individuums ist aber tatsächlich charakteristisch für eine Evolutionsstrategie und steht im Gegensatz zu genetischen Algorithmen, vgl. auch den Beispiel-GA aus Definition 6.1.1. Wir sehen den (μ+1)-EA als kanonische (μ+1)-Evolutionsstrategie für den Suchraum $\{0,1\}^n$ an, da die Selektionsoperatoren der traditionellen Definition für reellwertige Suchräume folgen und der Mutationsoperator der vielleicht gebräuchlichste global in $\{0,1\}^n$ suchende Operator ist.

Wenn wir in Definition 7.1.1 für μ den Wert 1 einsetzen, erhalten wir eine Suchheuristik, die folgenden unbedeutenden Unterschied gegenüber dem (1+1)-EA aufweist. Wenn in einem Schritt der Mutant denselben f-Wert wie sein Elter besitzt, löscht der (μ+1)-EA eines dieser beiden Individuen gleichverteilt, wohingegen der (1+1)-EA gemäß Definition 2.3.1 auf jeden Fall den Mutanten übernimmt. Es gilt hier dieselbe Bemerkung wie auf Seite 102 bezüglich des Beispiel-GAs. Beiden Varianten werden asymptotisch stets identische erwartete Laufzeiten haben. Für ein Ergebnis, das den (μ+1)-EA und den (1+1)-EA trennt und in Abschnitt 7.6 vorgestellt werden wird, wird es vollkommen unerheblich sein, welche der Varianten zugrunde gelegt wird.

Die Laufzeit des (μ+1)-EA messen wir schließlich wie auf Seite 92 anhand des Ein-Kind-EAs erläutert, sodass nur Werte $\mu = \mathrm{poly}(n)$ vernünftig erscheinen.

7.2 Beispielfunktionen

Wir werden die erwartete Laufzeit und Erfolgswahrscheinlichkeit des (μ+1)-EA auf drei Funktionen analysieren. Zunächst betrachten wir die bekannten pseudobooleschen Funktionen OneMax und LeadingOnes, zumal diese auch für Jansen und De Jong (2002) Basis bei der Analyse des (1+λ)-EA waren. Außerdem beschäftigen wir uns mit einer von Jansen und Wegener (2001a) eingeführten Beispielfunktion namens SPC, was für *short path with constant fitness* steht. Sei nun für $x \in \{0,1\}^n$

$$\mathrm{SPC}(x) := \begin{cases} n+1, & \text{falls } x = 1^i 0^{n-i} \text{ mit } 0 \le i \le n-1, \\ 2n, & \text{falls } x = 1^n, \\ n - \textsc{OneMax}(x) & \text{sonst.} \end{cases}$$

Ähnlich wie die Beispielfunktion f aus Abschnitt 6.2 führt $\mathrm{SPC}(x)$ nur auf Eingaben einer speziellen Gestalt zu großen Funktionswerten. Individuen der Gestalt $1^i 0^{n-i}$ mit $0 \le i \le n-1$ führen zum zweitgrößten Funktionswert $n+1$ und ein Individuum der Gestalt 1^n ist global optimal. Die Eingaben mit Funktionswert $n+1$ bilden bezüglich der Hammingnachbarschaft einen zusammenhängenden Teil des Suchraums, auf dem

SPC überall denselben Funktionswert annimmt. Eine zusammenhängende Teilmenge des Suchraums, auf der der Funktionswert konstant ist, heißt, wie in Abschnitt 3.2 vorgestellt, *Plateau*. Die Bezeichnung *Pfad*, die im Namen SPC steckt, werden wir hier vermeiden, da im Folgenden häufig Pfade in Stammbäumen auftreten.

Das Verhalten randomisierter Suchheuristiken auf Plateaus ist, wie bereits in Kapitel 3 gesehen, von besonderem Interesse. Auf der Funktion SPC benötigt der (1+1)-EA eine erwartete Laufzeit von $O(n^3)$ (Jansen und Wegener, 2001a) und die untere Schranke $\Omega(n^3)$ für seine erwartete Laufzeit ist auch leicht einzusehen (sie wird aber auch aus Abschnitt 7.5 folgen). Besonders interessant ist damit die Frage, ob die Erforschung des Plateaus von der Population des (μ+1)-EA profitiert.

7.3 Obere Schranken

Wir beginnen mit zwei relativ leicht zu beweisenden oberen Schranken für die erwartete Laufzeit des (μ+1)-EA, nämlich auf den Funktionen OneMax und LeadingOnes. Dazu können wir auf die Methoden aus Abschnitt 5.2.1 zurückgreifen, aber werden die Schranke aus Lemma 5.2.2 verfeinern. Es erscheint nahe liegend, dass wir die erwartete Laufzeit des (μ+1)-EA mit Lemma 5.2.2 im Allgemeinen überschätzen, da wir in seinem Beweis pessimistisch davon ausgehen, dass immer nur ein Individuum auch tatsächlich der Ranggruppe angehört, in der sich die aktuelle Population befindet (vgl. Seite 94 für die Definition der Ranggruppe einer Population). Wenn wir nun die Ranggruppennummer der aktuellen Population wieder als ein Potenzial auffassen, können wir in Phasen, in denen dieses Potenzial nicht steigt, ein Unterpotenzial betrachten. Dieses soll dann die Anzahl der Individuen der Population angeben, die zur Ranggruppe der aktuellen Population zählen. Da der (μ+1)-EA nur Individuen mit geringstem f-Wert löscht, kann das Unterpotenzial in den genannten Phasen nicht sinken. Es stellt sich außerdem heraus, dass das Unterpotenzial dann verhältnismäßig schnell wächst.

Ein hohes Unterpotenzial impliziert eine hohe Wahrscheinlichkeit, ein Individuum zu wählen, das der Ranggruppe der aktuellen Population angehört. Im folgender Erweiterung von Lemma 5.2.2 „warten" wir dann darauf, dass wir in der i-ten Ranggruppe ein Unterpotenzial von r_i erhalten. Da wir später auch Teilziele der Optimierung betrachten werden, bezieht sich das folgende Lemma nicht nur auf die erwartete Laufzeit.

Lemma 7.3.1 *Sei $L_1, \dots, L_m$ eine f-basierte Partition mit Verlassenswahrscheinlichkeiten $s(i)$ bez. des (μ+1)-EA und seien weiterhin Werte $r_i \in \mathbb{R}$ mit $1 \le r_i \le \mu$ für $1 \le i \le \mu - 1$ gegeben. Dann ist die erwartete Zahl von Schritten des (μ+1)-EA auf f, bis die aktuelle Population der mindestens k-ten Ranggruppe angehört, höchstens*

$$t(k) \ := \ \mu \sum_{i=1}^{k-1} \left(\frac{1}{r_i \cdot s(i)} + 2\mathrm{e}H_{\lceil r_i \rceil - 1} \right)$$

und die erwartete Laufzeit höchstens $t(m) + \mu$.

Beweis: Sei für die aktuelle Population mit i ihre Ranggruppennummer und mit R die Anzahl ihrer Individuen, die der i-ten Ranggruppe angehören, bezeichnet. Solange sich die Ranggruppennummer der Population nicht erhöht, kann R nicht sinken, da der (μ+1)-EA nur Individuen mit geringstem f-Wert löscht. Wenn $R \geq r_i$ gilt, können wir die Argumentation aus dem Beweis von Lemma 5.2.1 mit einer speziellen unteren Schranke für die Auswahlwahrscheinlichkeit anwenden. Es folgt aufgrund der gleichverteilten Selektion zur Reproduktion im (μ+1)-EA, dass die Wahrscheinlichkeit, ein Individuum aus der i-ten Ranggruppe zu wählen, dann mindestens r_i/μ beträgt.

Wenn $R < r_i$ gilt, warten wir darauf, bis $R \geq \lceil r_i \rceil$ eintritt oder sich die Ranggruppennummer der Population erhöht. Die Wartezeit dafür schätzen wir mit einer Argumentation, die ähnlich zum Beweis der Aussage des Sammelbilderproblems (Lemma A.15) funktioniert, wie folgt ab. Wenn ein Schritt des (μ+1)-EA ein Individuum der i-ten Ranggruppe zur Mutation wählt und ein Replikat erzeugt, erhöht sich der R-Wert. Die entsprechende Auswahlwahrscheinlichkeit beträgt mindestens R/μ und die Wahrscheinlichkeit eines Replikats bekanntermaßen mindestens $1/(2\mathrm{e})$. Wenn wir die zugehörigen oberen Schranken für die Wartezeiten aufsummieren, erhalten wir $\sum_{r=1}^{\lceil r_i \rceil - 1} 2\mathrm{e}\mu/r = 2\mathrm{e}\mu H_{\lceil r_i \rceil - 1}$ als obere Schranke für die erwartete Zeit, bis die i-te Ranggruppe verlassen wurde oder mindestens r_i Individuen daraus vorhanden sind. Die Aussage über die Laufzeit folgt schließlich direkt aus deren Definition. □

Wir erkennen an der Formulierung von Lemma 7.3.1, dass sowohl zu kleine als auch zu große r_i-Werte zu großen oberen Schranken für die erwartete Laufzeit führen. Eine Anwendung von Lemma 7.3.1 wird daher nur dann gute Schranken liefern, wenn die r_i-Werte sorgfältig gewählt sind.

Wir bemerken noch am Rande, dass man bei der Analyse populationsbasierter EAs gelegentlich daran interessiert ist, wie lange es dauert, bis nicht nur ein optimales Individuum in der Population ist, sondern die gesamte Population nur aus optimalen Individuen besteht. Die Zeit von einem bis zu allen optimalen Individuen bezeichnet man dann auch als *Übernahmezeit (takeover time)*, siehe z. B. Rudolph (2000). Sie kann für den (μ+1)-EA mit den Methoden aus dem Beweis von Lemma 7.3.1 abgeschätzt werden.

Mithilfe von Lemma 7.3.1 folgen die oberen Schranken für die erwartete Laufzeit des (μ+1)-EA auf OneMax und LeadingOnes nun auf relativ leichte Weise.

Theorem 7.3.2 *Sei $\mu = \mathrm{poly}(n)$. Dann beträgt die erwartete Laufzeit des (μ+1)-EA auf* LeadingOnes *höchstens $\mu + 3\mathrm{e}n \cdot \max\{\mu \ln(\mathrm{e}n), n\} = O(\mu n \log n + n^2)$.*

Beweis: Zur Vereinfachung der Notation sei in diesem Beweis $f :=$ LeadingOnes. Um Lemma 7.3.1 anzuwenden, stellen wir die folgende f-basierte Partition auf. Sei $L_i := \{x \mid f(x) = i\}$ für $0 \leq i \leq n-1$ und $L_n := \{x \mid f(x) = n\} = \{1^n\}$. Es gilt $s(i) \geq (1/n)(1-1/n)^{n-1} \geq 1/(\mathrm{e}n)$, da es hinreichend ist, genau die linkeste Null eines Individuums aus Ranggruppe L_i zu kippen, um ein Individuum aus Ranggruppe L_{i+1} zu erzeugen.

Wir unterscheiden nun zwei Fälle. Im ersten Fall ist $\mu < n$. Wir setzen dann $r_i := \mu$ für $0 \le i \le n-1$. Nach Lemma 7.3.1 ist die erwartete Laufzeit beschränkt durch

$$\mu + \mu \sum_{i=0}^{n-1} \left(\frac{\mathrm{e}n}{\mu} + 2\mathrm{e}\mu H_{\mu-1} \right) \;\le\; \mu + \mathrm{e}n^2 + 2\mathrm{e}n\mu(\ln \mu + 1) \;\le\; \mu + \mathrm{e}n^2 + 2\mathrm{e}n\mu(\ln n + 1).$$

Im anderen Fall, d. h. für $\mu \ge n$, setzen wir $r_i := n/(\ln n + 1)$ für $0 \le i \le n-1$. Dann liefert Lemma 7.3.1

$$\mu + \mu \sum_{i=0}^{n-1} \left(\frac{\mathrm{e}n}{n/(\ln n + 1)} + 2\mathrm{e}\mu H_{\lceil n/(\ln n+1)\rceil - 1} \right) \;\le\; \mu + \mathrm{e}n\mu(\ln n + 1) + 2\mathrm{e}n\mu(\ln n + 1)$$

als obere Schranke für die erwartete Laufzeit. Indem wir nach dem Maximum der Ausdrücke n und $\mu(\ln n + 1) = \mu \ln(\mathrm{e}n)$ unterscheiden, folgt insgesamt die obere Schranke $\mu + 3\mathrm{e}n \max\{\mu \ln(\mathrm{e}n), n\}$ wie behauptet. □

Wir kommen nun zur zweiten einfachen Testfunktion. Es stellt sich heraus, dass die Argumente, die die erwartete Laufzeit des (μ+1)-EA auf OneMax beschränken, etwas komplizierter als bei LeadingOnes sind. Interessanterweise gilt Ähnliches für entsprechende Analysen des (1+λ)-EA (Jansen und De Jong, 2002). Ein Grund dafür liegt darin, dass die Wahrscheinlichkeit, den Funktionswert eines Individuums mit Funktionswert i zu erhöhen, bei OneMax anders als bei LeadingOnes nicht für alle i in der Größenordnung $\Theta(1/n)$ ist.

Theorem 7.3.3 *Sei $\mu = \mathrm{poly}(n)$. Dann beträgt die erwartete Laufzeit des (μ+1)-EA auf* OneMax *höchstens $\mu + 5\mathrm{e}\mu n + \mathrm{e}n \ln(\mathrm{e}n) = O(\mu n + n \log n)$.*

Beweis: Wir gehen ähnlich wie im Beweis von Theorem 7.3.2 vor, leiten aber untere Schranken $s(i)$ und Werte r_i her, die jeweils von i abhängen. Zur Vereinfachung der Notation schreiben wir $f :=$ OneMax und stellen erneut die f-basierte Partition $L_i := \{x \mid f(x) = i\}$ für $0 \le i \le n-1$ und $L_n := \{x \mid f(x) = n\} = \{1^n\}$ auf. Es gilt $s(i) \ge (n - i/n)(1 - 1/n)^{n-1} \ge (n-i)/(\mathrm{e}n)$, da es hinreichend ist, genau eine der $n-i$ Nullen eines Individuums aus Ranggruppe L_i zu kippen, um ein Individuum aus Ranggruppe L_{i+1} zu erzeugen. Wir wählen $r_i := \min\{\mu, n/(n-i)\}$ für $0 \le i \le n-1$.

Wir wenden Lemma 7.3.1 an und erhalten als obere Schranke für die erwartete Laufzeit

$$\mu + \mu \sum_{i=0}^{n-1} \left(\frac{\mathrm{e}n}{n-i} \cdot \frac{1}{\min\{\mu, n/(n-i)\}} + 2\mathrm{e}H_{\lceil n/(n-i)\rceil - 1} \right),$$

indem wir im letzten Summanden stets $n/(n-i)$ als obere Schranke für r_i nehmen. Wir schätzen nun die Summe der H-Werte ab und erhalten gemäß der stirlingschen Formel

$$\sum_{i=0}^{n-1} H_{\lceil n/(n-i)\rceil - 1} \;\le\; \sum_{i=0}^{n-1} \ln\!\left(\frac{\mathrm{e}n}{n-i} \right) \;=\; \ln\!\left(\frac{\mathrm{e}^n n^n}{n!} \right) \;\le\; \ln\!\left(\frac{\mathrm{e}^n n^n}{(n/\mathrm{e})^n} \right) \;=\; 2n.$$

Es ist $1/\min\{a,b\} \le 1/a + 1/b$ für $a, b > 0$. Also ist die erwartete Laufzeit insgesamt nach oben beschränkt durch

$$\mu + \sum_{i=0}^{n-1} \left(\frac{\mu e n}{n-i} \left(\frac{1}{\mu} + \frac{n-i}{n} \right) \right) + 2e\mu \cdot 2n \;=\; \mu + \sum_{i=0}^{n-1} \left(\frac{en}{n-i} + e\mu \right) + 4e\mu n,$$

was wiederum höchstens $\mu + en\ln(en) + 5e\mu n$ ist. □

Die erwarteten Laufzeiten des (1+1)-EA auf OneMax und LeadingOnes betragen bekanntlich $\Theta(n \log n)$ bzw. $\Theta(n^2)$. Im Vergleich zu den trivialen oberen Schranken $O(\mu n \log n)$ bzw. $O(\mu n^2)$, die für die Laufzeit des (μ+1)-EA aus Lemma 5.2.2 folgten, enthalten die Theoreme 7.3.2 und 7.3.3 asymptotisch bessere Aussagen. Obschon die zugehörige Beweismethode recht grobe Abschätzungen vornimmt, wird sich in den nächsten Abschnitten zeigen, dass die Schranken aus den beiden Theoremen im asymptotischen Sinne optimal bzw. fast optimal sind.

Bei der Analyse einer oberen Schranke für die erwartete Laufzeit des (μ+1)-EA auf SPC können wir gegenüber der entsprechenden Schranke $O(n^3)$ für den (1+1)-EA nur eine um einen scheinbar auf triviale Weise zu erhaltenden Faktor $\Theta(\mu)$ erhöhte Abschätzung nachweisen. Trotzdem wird sich herausstellen, dass diese Abschätzung im asymptotischen Sinne zumindest fast optimal ist. Zum Beweis des folgenden Theorems werden wir erstmalig Stammbäume auch im Zusammenhang mit oberen Schranken für die erwartete Laufzeit einsetzen.

Theorem 7.3.4 *Sei $\mu = \mathrm{poly}(n)$. Dann beträgt die erwartete Laufzeit des (μ+1)-EA auf* SPC *höchstens $O(\mu n^3)$.*

Bevor wir uns dem Beweis von Theorem 7.3.4 widmen, führen wir einige Begriffe ein, die sich alle auf den Lauf des (μ+1)-EA auf SPC beziehen. Darin gibt es einen ersten Zeitpunkt, an dem alle Individuen der aktuellen Population von der Gestalt $1^i 0^{n-i}$ für $0 \le i \le n$ sind. (Hier ist $i = n$ zugelassen.) Wir nennen den genannten Zeitpunkt t_{PLA} und die Individuen der genannten Gestalt *Plateauindividuen* oder kürzer *Plateau(such)punkte*. Man beachte, dass 1^n nicht zum Plateau konstanten Funktionswertes aus der Funktion SPC gehört und die Bezeichnung *Pfadindividuen* wegen der Struktur $1^i 0^{n-i}$ daher eigentlich passender wäre. Allerdings bleiben wir bei den vorgenannten Bezeichnungen, um Verwechselungen mit Pfaden innerhalb von Stammbäumen zu vermeiden.

Man macht sich leicht klar, dass $\mathrm{E}(t_{\mathrm{PLA}})$ existiert und dass t_{PLA} sogar den Wert 0 annimmt, falls die initiale Population nur aus Plateaupunkten besteht. Allerdings gilt, wie wir für eine untere Schranke in Abschnitt 7.5 ausnutzen werden, typischerweise $t_{\mathrm{PLA}} > 0$ und der dann einzige Plateaupunkt in der Population zur Zeit t_{PLA} ist nicht optimal. Schritte, die nach Zeit t_{PLA} passieren und keinen Plateaupunkt, d. h. ein Individuum, das nicht als $1^i 0^{n-i}$ geschrieben werden kann, erzeugen, löschen einen solchen Mutanten noch im selben Schritt. Nun betrachten wir die in Abschnitt 5.2.2

eingeführten Stammbäume. Sei $t \geq t_{\mathrm{PLA}}$ ein beliebiger Zeitpunkt, sei x ein Individuum aus der Population $X^{(t)}$ und sei v ein Knoten im Stammbaum von x zu einer weiteren Zeit $s \geq t$. Falls der Knoten v zu dieser Zeit bereits tot ist (vgl. Abschnitt 5.2.2), nennen wir den Pfad von x nach v ebenfalls tot und andernfalls nennen wir ihn lebendig.

Wir kommen jetzt auf die im Zusammenhang mit Beobachtung 5.2.1 diskutierte Abhängigkeit von Mutationen zurück, die sich ergibt, wenn man entstehende Pfade aus Stammbäumen betrachtet. Wenn wir das Ereignis betrachten, dass der Knoten v zur Zeit s lebendig ist, stellt dies gewisse Bedingungen an die Mutationen, die die Knoten auf dem Pfad von x nach v erzeugen. Beispielsweise werden Mutanten, die keine Plateaupunkte sind, nach t_{PLA} aufgrund ihres geringen Funktionswertes sofort gelöscht. Im Folgenden werden wir diese Bedingungen näher studieren, um nachzuweisen, dass wir auf lebendigen Pfaden gewissermaßen den Lauf des (1+1)-EA auf SPC wieder finden.

Lemma 7.3.5 *Es sei der (μ+1)-EA auf* SPC *betrachtet. Sei x_0 ein Individuum aus der Population zu einer Zeit $t \geq t_{\mathrm{PLA}}$ und sei p ein Pfad im Stammbaum von x_0 zu einer Zeit $s \geq t$. Seien mit $x_0, \ldots, x_\ell$ die Individuen entlang p bezeichnet. Falls p lebendig ist und $x_i \neq 1^n$ für $0 \leq i \leq \ell$ gilt, haben die Elemente der Folge $x_0, \ldots, x_\ell$ dieselbe Verteilung wie die Suchpunkte des (1+1)-EA auf der Funktion* SPC *zu den jeweiligen Zeitpunkten $0, \ldots, \ell$, gegeben, dass der (1+1)-EA mit dem Suchpunkt x_0 startet und in den Mutationen bis zum Zeitpunkt ℓ ausschließlich Suchpunkte der Gestalt $1^i 0^{n-i}$ mit $0 \leq i < n$ erzeugt.*

Beweis: Wir nehmen o. B. d. A. $t = 0$ an, um Indextransformationen zu vermeiden. Seien die Knoten des Pfades p mit $0, t_1, \ldots, t_\ell$ bezeichnet. Wir werden die Behauptung mit Induktion über $i \in \{0, \ldots, \ell\}$ zeigen; der Induktionsanfang $i = 0$ ist dabei trivial, da der (1+1)-EA gemäß Voraussetzung mit x_0 startet. Nun sei $0 \leq i < \ell$ und es sei angenommen, dass der Knoten t_i, der den Suchpunkt x_i trägt, gerade erzeugt wurde und lebendig ist. Eine solche Situation muss eintreten, damit p zur Zeit s lebendig sein kann. Das Lemma stellt nun genau die folgenden Bedingungen an die Ereignisse, die den Pfad $0, t_1, \ldots, t_{i+1}$, den wir im Induktionsschritt analysieren, erzeugen.

1. Es ereignet sich ein Schritt, der t_i zur Mutation wählt und einen Knoten t', für dessen Beschriftung x' die Eigenschaft $x' = 1^k 0^{n-k}$ mit $k \neq n$ gilt, erzeugt.

2. Falls $i + 1 < \ell$, wird t' nicht gelöscht, bevor ein Schritt geschieht, der t' wählt und t_{i+2} erzeugt; falls $i + 1 = \ell$, bleibt t' bis zum Zeitpunkt s lebendig.

3. $t' = t_{i+1}$.

Die Bedingungen (2) und (3) sind trivial einzusehen. Bedingung (1) gilt, da wir zum einen $x_i \neq 1^n$ annehmen und zum anderen einen Zeitpunkt nach t_{PLA} betrachten. Wäre x' kein Plateaupunkt, würde das Individuum x' sofort gelöscht, womit Bedingung (2) unmöglich würde. Sei mit A der Durchschnitt der beschriebenen drei Bedingungen bezeichnet. Wenn A eintritt, trägt x' den Namen x_{i+1}.

Im Lemma nehmen wir analog zu Bedingung (1) für die aktuellen Suchpunkte des (1+1)-EA bis zur Zeit ℓ an, dass ausschließlich von 1^n verschiedene Plateaupunkte erzeugt werden. Diese Mutanten werden aufgrund der Definition des (1+1)-EA stets akzeptiert. Also gelten sowohl für den x_{i+1} erzeugenden Schritt des (μ+1)-EA und denjenigen Schritt des (1+1)-EA, der seinen Suchpunkt zur Zeit $i+1$ erzeugt, dass die zugehörigen Mutationen der Bedingung gehorchen, einen Suchpunkt $1^k 0^{n-k}$ für $k \neq n$ zu produzieren.

Wir müssen die Verteilung von x_{i+1} noch genauer untersuchen. Hier ist es entscheidend, dass der (μ+1)-EA das zu löschende Individuum gleichverteilt aus der Menge der Individuen mit geringstem Funktionswert wählt. Da alle Individuen der Gestalt $1^j 0^{n-j}$ mit $j \neq n$ denselben SPC-Wert haben, folgt, dass das Ereignis, dass ein solches Individuum gelöscht wird, unabhängig von dem Wert ist, den j annimmt. Damit ist auch das als Bedingung (2) formulierte Ereignis unabhängig davon, welchen Wert j annimmt. Sei k die Zahl der Einsen von x'. Gemäß den vorigen Überlegungen hat k mit Bedingung (1) allein dieselbe Verteilung wie mit allen drei Bedingungen. Der (1+1)-EA besitzt denselben Mutationsoperator wie der (μ+1)-EA und gemäß der Überlegung im vorigen Absatz erfüllt die Mutation, die den Suchpunkt des (1+1)-EA zur Zeit i wählt, die zu Bedingung (1) analoge Bedingung. Wenn x_i und der Suchpunkt des (1+1)-EA zur Zeit i denselben Wert $1^j 0^{n-j}$ annehmen, folgen ihre Nachfolger also derselben Verteilung. Da x_i und der Suchpunkt des (1+1)-EA zur Zeit i gemäß Induktionsvoraussetzung derselben Verteilung folgen, ist der Induktionsschritt gezeigt. □

Wir können auf einem lebendigen Pfad, der die Bedingungen von Lemma 7.3.5 erfüllt, also den Lauf des (1+1)-EA auf SPC wieder finden, wenn der (1+1)-EA mit einem Plateaupunkt startet, nur akzeptierte Mutationen des (1+1)-EA betrachtet werden und das Optimum noch nicht erreicht ist. Dies erlaubt uns zunächst, Theorem 7.3.4 zu beweisen. In Abschnitt 7.5 werden wir auf Lemma 7.3.5 erneut zurückgreifen, um untere Schranken für die erwartete Laufzeit des (μ+1)-EA auf SPC zu zeigen.

Beweis von Theorem 7.3.4: Sei x ein beliebig gewähltes Individuum aus der Population zum Zeitpunkt t_{PLA} und bezeichne $T(x)$ seinen Stammbaum. Wir wollen die folgende Eigenschaft E nachweisen: Die erwartete Zahl von Schritten, bis mindestens ein Pfad in $T(x)$ Länge k erreicht oder bis alle Pfade in $T(x)$ tot sind oder bis das Optimum erreicht wird, ist für jedes $k \geq 1$ durch $4e\mu k$ nach oben beschränkt. Unter Zuhilfenahme dieser Eigenschaft zeigen wir das Theorem dann auf folgende Weise. Zum einen beweisen wir $\mathrm{E}(t_{\text{PLA}}) = O(\mu n \log n)$. Zum anderen werden wir Lemma 7.3.5 wie folgt anwenden. Wir betrachten das Ereignis, dass ein spezieller Pfad in $T(x)$ entsteht, der mit dem Suchpunkt 1^n endet. Seien $x_0 = x, \ldots, x_{k-1}, x_k = 1^n$ die Individuen entlang dieses Pfades. Die Folge $x_0, \ldots, x_{k-1}$ tritt laut Lemma 7.3.5 mit derselben Wahrscheinlichkeit in den ersten $k-1$ akzeptierten Schritten des (1+1)-EA auf SPC mit initialem Suchpunkt x_0 auf. Eine Mutation, die x_{k-1} zu 1^n mutiert, hat trivialerweise im (μ+1)-EA und im (1+1)-EA dieselbe Wahrscheinlichkeit. Also ist die Wahrscheinlichkeit, dass der Suchpunkt 1^n auf einem Pfad vorgegebener Länge k innerhalb des Stammbaums $T(x)$ erzeugt wird, ebenso groß wie die Wahrscheinlichkeit, dass der (1+1)-EA mit initialem Suchpunkt x das Optimum 1^n innerhalb von k akzeptierten Schritten erzeugt.

Die Analysen der erwarteten Laufzeit des (1+1)-EA auf SPC (Jansen und Wegener, 2001a) liefern, dass die erwartete Zeit bis zum Erreichen des Optimums durch $O(n^3)$ beschränkt ist, wenn der (1+1)-EA mit irgendeinem Plateaupunkt startet. Hier werden sogar nicht akzeptierte Schritte gezählt. Gemäß der Markoffungleichung genügt eine Zeit von $O(n^3)$ mit einer Wahrscheinlichkeit von mindestens 1/2 und wir erhalten gemäß den vorigen Überlegungen, dass ein Pfad in $T(x)$ mit Länge cn^3 für eine geeignete Konstante c mit einer Wahrscheinlichkeit von mindestens 1/2 zum Optimum führt. Da es immer mindestens einen lebendigen Pfad im Stammbaum irgendeines Individuums aus der Population zur Zeit t_{PLA} gibt, erhalten wir zusammen mit der Eigenschaft E, dass der (μ+1)-EA das Optimum mit einer Wahrscheinlichkeit von mindestens 1/2 in $O(\mu n^3)$ Schritten erreicht. Im Falle eines Fehlers können wir diese Argumentation wiederholen, indem wir den Stammbaum für einen geeigneten Plateaupunkt aus der dann vorliegenden Population betrachten. Die erwartete Zahl von Wiederholungen ist höchstens 2, sodass die erwartete Laufzeit des (μ+1)-EA insgesamt durch $O(\mu n^3)$ beschränkt ist.

Wir wollen nun Eigenschaft E nachweisen. Dazu nehmen wir an, dass bis zum Zeitpunkt $t_{\text{PLA}} + 4\text{e}\mu k$ stets ein lebendiger Pfad in $T(x)$ vorliegt, denn sonst ist nichts zu zeigen. Nun benutzen wir eine Potenzialfunktion L, die für jeden Zeitpunkt $t \geq t_{\text{PLA}}$ auf folgende Weise definiert ist. Sei S die Menge der aktuell lebendigen Knoten in $T(x)$, die außerdem bis zum Zeitpunkt $t_{\text{PLA}}+4\text{e}\mu k$ stets einen lebendigen Nachfahren haben. Dann ist L definiert als die maximale Tiefe der Knoten aus S und entspricht der Länge eines augenblicklich längsten Pfades, für den feststeht, bis zum genannten Zeitpunkt stets eine lebendige Verlängerung zu haben. Man beachte, dass L zwar insoweit ein (zufälliges) Potenzial im üblichen Sinne ist, als jedem Zeitpunkt zwischen t_{PLA} und $t_{\text{PLA}} + 4\text{e}\mu k$ ein Wert zugeordnet wird. Im Gegensatz zu den Potenzialen für Populationen, die wir in Abschnitt 5.2.1 kennen gelernt haben, hängt der Wert von L aber auch von zukünftigen Zeitpunkten ab und ist nicht über den Funktionswert von Individuen definiert.

Per definitionem kann L in dem Zeitraum, für den es definiert ist, nicht sinken. Außerdem besteht folgende hinreichende Bedingung für eine Erhöhung des aktuellen L-Wertes: Ein Individuum x', das den aktuellen L-Wert bestimmt, wird zur Mutation gewählt, der Mutant ist ein Plateaupunkt und x' wird gelöscht, bevor der Mutant gelöscht wird. Dadurch erhöht sich der L-Wert, weil x' stets einen lebendigen Nachfahren im betrachteten Zeitraum hat. Die Wahrscheinlichkeit, x' zu wählen, beträgt $1/\mu$, die Wahrscheinlichkeit eines geeigneten Mutanten mindestens $1/(2\text{e})$, da es genügt, ein Replikat zu erzeugen, und die Wahrscheinlichkeit, dass x' vor seinem Mutanten gelöscht wird, ist genau 1/2, da die beiden Individuen denselben SPC-Wert haben und daher in jedem Schritt mit derselben Wahrscheinlichkeit zum Löschen gewählt werden. Also ist die erwartete Zeit, um L zu erhöhen, höchstens $4\mathrm{e}\mu$ und damit erreicht mindestens ein Pfad in $T(x)$ eine Länge von mindestens k in höchstens $4\mathrm{e}\mu k$ erwarteten Schritten, was E beweist.

Es verbleibt die Aussage $\mathrm{E}(t_{\mathrm{PLA}}) = O(\mu n \log n)$ zu zeigen. Falls die aktuelle Population $i \geq 1$ Plateaupunkte enthält, erhöht sich die Zahl der Plateaupunkte, indem ein Plateaupunkt gewählt wird und ein Replikat erzeugt wird. Dieses hat eine Wahrscheinlichkeit von mindestens $(i/\mu)(1/2\mathrm{e})$. Durch Aufsummieren der erwarteten Zeiten und der üblichen Abschätzung der μ-ten harmonischen Zahl erhalten wir eine erwartete Zeit von höchstens $O(\mu \log \mu) = O(\mu \log n)$ (wegen $\mu = \mathrm{poly}(n)$), bis wir eine Population, die nur Plateaupunkte enthält, bekommen. (Diese Methode findet sich auch im Beweis von Lemma 7.3.1.) Falls die Population noch keinen Plateaupunkt enthält, ist der Funktionswert aller ihrer Individuen gegeben durch $n - \textsc{OneMax}(x)$, auch ZeroMax genannt. Da der $(\mu{+}1)$-EA immer noch dasselbe Verhalten zeigt, wenn Nullen ins Einsen umbenannt werden und umgekehrt (vgl. auch die Bemerkungen in Abschnitt 3.1), folgt mit Theorem 7.3.2, dass nach einer erwarteten Zeit von $O(\mu n + n \log n) = O(\mu n \log n)$ mindestens ein Plateaupunkt entsteht. □

7.4 Eine Untere-Schranken-Technik

Bereits in Abschnitt 6.4, speziell im Beweis von Theorem 6.4.1, haben wir für die Funktion f und den Beispiel-GA implizit eine untere Schranke für die Zeit, bis viele Einsen im Präfix entstehen, nachgewiesen. Als entscheidendes Werkzeug, das wir dort benutzt haben, erwies sich die Möglichkeit, den stochastischen Prozess, der Stammbäume von Individuen erzeugte, zu erfassen. Anschließend lieferte eine Studie dieses stochastischen Prozesses einen nicht trivialen Zusammenhang zwischen der Zahl der Knoten eines Stammbaumes und seiner Tiefe. Letztendlich lief eine damit gewonnene obere Schranke für die Tiefe auf das Argument hinaus, dass sämtliche Knoten im Stammbaum mit hoher Wahrscheinlichkeit noch so ähnlich zur Wurzel sind, dass eine spezielle Eigenschaft, dort viele Präfixeinsen, noch hinreichend unwahrscheinlich ist. Im Folgenden soll die spezielle Eigenschaft, die ein Knoten im Stammbaum aufweisen muss und die für die Wurzel typischerweise nicht erfüllt ist, eng verbunden sein mit dem Ereignis, dass das Optimum erreicht wird.

7.4.1 Beschreibung der Technik

Wie soeben motiviert, versuchen wir den stochastischen Prozess, der einen Stammbaum, der im Laufe des $(\mu{+}1)$-EA entsteht, zu charakterisieren. Es stellt sich heraus, dass eine recht allgemeine Aussage über diesen Prozess möglich ist, da bei der Selektion zur Reproduktion gleichverteilt und somit unabhängig von der gegebenen Zielfunktion gewählt wird.

Definition 7.4.1 *Sei $p := p_{t,u}$, $0 \leq u \leq t$, eine Folge von Wahrscheinlichkeitsverteilungen, sodass die Trägermenge von $p_{t,u}$ eine Teilmenge von $\{-1, 0, \ldots, u\}$ ist. Ein* p-zufälliger Baum *zum Zeitpunkt* 0 *besteht nur aus seiner Wurzel. Ein p-zufälliger Baum T_t zum Zeitpunkt $t > 0$ entsteht aus dem p-zufälligen Baum T_{t-1} auf folgende Weise. Sei u die Zahl der Knoten von T_{t-1} und werde t^* gemäß $p_{t-1,u-1}$ gezogen. Falls $t^* \geq 0$, wird ein neues Blatt an den zum Zeitpunkt t^* eingefügten Knoten angehängt; andernfalls ist $T_t := T_{t-1}$.*

Ein p-zufälliger Baum heißt $1/\mu$-Baum, *falls $p_{t,u}$ jedem Element $t^* \geq 0$ eine Wahrscheinlichkeit von höchstens $1/\mu$ zuordnet.*

Da wir in Definition 7.4.1 einige Begriffe aus Definition 6.4.3 wieder verwenden, sollten wir jetzt sorgfältig den Unterschied zwischen beiden Definitionen herausarbeiten. Während der Begriff eines p-zufälligen Baums in Definition 6.4.3 auf einer Folge von Verteilungen p_t, $t \in \mathbb{N}_0$, beruhte, erweitert Definition 7.4.1 diesen Begriff auf Verteilungen, die von zwei Parametern, t und u, abhängen. Der p-zufällige Baum aus Definition 6.4.3 enthält zum Zeitpunkt t genau $t + 1$ Knoten, wohingegen für den p-zufälligen Baum aus Definition 7.4.1 dann nur eine obere Schranke von $t + 1$ für die Zahl seiner Knoten gilt. Dies begründet sich daraus, dass wir bei Letzterem nicht sicher sein können, ob der (μ+1)-EA aus dem mithilfe des p-zufälligen Baums modellierten Stammbaum wählt. Damit wird auch verständlich, dass wir -1 als Wert der Zufallsvariablen $p_{t,u}$ erlauben, um das Ereignis, dass ein Stammbaum in einem Schritt nicht wächst, zu erfassen.

Auch die Begriffe eines $1/u$-Baums und eines $1/\mu$-Baums unterscheiden sich. Während bei einem $1/u$-Baum erst ab dem Zeitpunkt $u - 1$ sichergestellt ist, dass die Auswahlwahrscheinlichkeit aller Knoten durch $1/u$ beschränkt ist, gilt in einem $1/\mu$-Baum bereits ab dem Zeitpunkt 0, dass jeder Knoten mit einer Wahrscheinlichkeit von höchstens $1/\mu$ gewählt wird. Demzufolge muss übrigens $p_{t,u}$ für $t < \mu - 1$ mit positiver Wahrscheinlichkeit den Wert -1 annehmen. Da die Begriffe aus Definition 6.4.3 im Folgenden keine Rolle mehr spielen werden, sollten die ähnlichen Namensgebungen gemäß Definition 7.4.1 keine Probleme bereiten.

Wir müssen noch begründen, wieso das Wachstum eines Stammbaums im Lauf des (μ+1)-EA von Definition 7.4.1 korrekt modelliert wird. Zunächst machen wir uns klar, dass die Folge von Verteilungen $p_{t,u}$ wohldefiniert ist, wenn wir den Prozess, der den Stammbaum eines Individuums x des (μ+1)-EA auf einer gegebenen Zielfunktion erzeugt, vor Augen haben. Das Wurzelindividuum x des Stammbaums darf sowohl zufällig sein als auch aus einer fest vorgegebenen Population sein; zudem braucht x nicht aus der initialen Population zu stammen. Während die Selektion zur Reproduktion des (μ+1)-EA alle Individuen der aktuellen Population gleich behandelt, können die Verteilungen $p_{t,u}$ den u Knoten des aktuellen p-zufälligen Baums aber durchaus verschiedene Werte zuweisen. Beispielsweise haben Knoten, deren Funktionswert gering ist, eine geringere Wahrscheinlichkeit, zum betrachteten Zeitpunkt noch lebendig zu sein, als Knoten mit hohem Funktionswert. Die Selektion zur Ersetzung, die ein Individuum mit geringstem Funktionswert löscht, schlägt sich mithin in $p_{t,u}$-Werten nieder, die auch zwischen 0 und $1/\mu$ liegen können.

Obwohl es bei gegebener Zielfunktion also theoretisch möglich ist, das Wachstum des Stammbaums eines Individuums durch einen p-zufälligen Baum für eine wohldefinierte Folge von Verteilungen $p_{t,u}$ zu modellieren, ist es im Allgemeinen zu kompliziert, diese Verteilungen exakt anzugeben. Wir arbeiten daher mit Schranken, die der (μ+1)-EA trivialerweise einhält, und fassen p-zufällige Bäume, deren Verteilungen $p_{t,u}$ diese Schranken beachten, zu einer neuen Klasse von stochastischen Prozessen zusammen, die wir $1/\mu$-Bäume nennen. Auch hier erlauben wir uns, sowohl den stochastischen Prozess „$1/\mu$-Baum“ als auch seinen Zustand zu einem Zeitpunkt t jeweils mit demselben Namen zu bezeichnen. Dies sollte ebenso wenig wie in Abschnitt 6.4 Schwierigkeiten bereiten.

Trotz der trivialen oberen Schranke $1/\mu$, die als einziges Wissen über die Auswahlwahrscheinlichkeiten für Knoten in Stammbäumen des (μ+1)-EA in die Definition der $1/\mu$-Bäume einfließt, lassen sich erstaunlich gute obere Schranken für die Tiefe Letzterer herleiten.

Lemma 7.4.2 *Es bezeichne $D(t)$ die Tiefe eines $1/\mu$-Baums zum Zeitpunkt t. Für alle $t, d \geq 0$ gilt*

$$\text{Prob}(D(t) \geq d) \;\leq\; \frac{1}{d!}\left(\frac{t}{\mu}\right)^d .$$

Außerdem gilt $\text{Prob}(D(t) \geq 3t/\mu) = 2^{-\Omega(t/\mu)}$.

Bevor wir den Beweis von Lemma 7.4.2 angehen, wollen wir die dort formulierte Schranke $3t/\mu$, die die Tiefe eines $1/\mu$-Baums mit überwältigender Wahrscheinlichkeit einhält, interpretieren. Wenn wir μ parallele Läufe des (1+1)-EA zum Zeitpunkt t betrachten, wurden genau $\mu(t+1)$ Auswertungen der Zielfunktion vorgenommen. Dies ist andererseits auch die Zahl der Zielfunktionsauswertungen im (μ+1)-EA bis zum Zeitpunkt $t\mu$. Die maximale Tiefe eines Stammbaums beträgt in beiden Fällen mit mindestens überwältigender Wahrscheinlichkeit $O(t)$. Wenn wir also an oberen Schranken für die Tiefe von Stammbäumen interessiert sind, besteht ein bemerkenswerter Bezug zwischen dem (μ+1)-EA und μ parallelen Läufen des (1+1)-EA. In Fällen, in denen alle parallelen Läufe auch zu Stammbäumen der Tiefe $\Omega(t)$ zum Zeitpunkt t führen, können wir in Bezug auf diese Tiefe sagen, dass der (μ+1)-EA zum Zeitpunkt $t\mu$ nicht effizienter als die parallelen Läufe ist. Natürlich lässt sich die Effizienz der evolutionären Algorithmen nicht allein anhand der Tiefe von Stammbäumen festmachen; gleichwohl werden wir im nächsten Unterabschnitt ein Beispiel diskutieren, bei dem der (μ+1)-EA aufgrund von Lemma 7.4.2 tatsächlich in gewissem Sinne ebenso ineffizient wie μ parallele Läufe des (1+1)-EA ist.

Um Lemma 7.4.2 zu beweisen, ist abermals eine technische Analyse vonnöten. Diese greift aber sehr viele Ideen aus Abschnitt 6.4.3 wieder auf.

Definition 7.4.3 *Sei $L(t, d)$ die Anzahl der Knoten mit Tiefe d in einem $1/\mu$-Baum zum Zeitpunkt t. Bezeichne $E(t, d) := \mathrm{E}(L(t, d))$ ihren Erwartungswert.*

Es gelten die folgenden einfachen und grundlegenden Beziehungen.

Lemma 7.4.4 *Es sei ein $1/\mu$-Baum betrachtet. Dann gilt*

$$\begin{aligned} E(t,0) &= 1 && \text{für } t \geq 0, \\ E(0,d) &= 0 && \text{für } d \geq 1, \\ E(t,d) &\leq \sum_{i=0}^{t-1} \frac{E(i,d-1)}{\mu} && \text{für } t \geq 1,\ d \geq 1. \end{aligned}$$

Beweis: Die erste und die zweite Beziehung gelten gemäß Definition 7.4.1. Zum Beweis der dritten Beziehung halten wir fest, dass $L(t, d)$ sich von $L(t-1, d)$ genau dann unterscheidet, wenn zum Zeitpunkt t ein Knoten auf Tiefe $d-1$ ausgewählt wird, um Elter des neu eingefügten Knotens zu sein. Wenn $L(t-1, d-1) = i$, passiert dies gemäß Definition 7.4.1 mit einer Wahrscheinlichkeit von höchstens $1/\mu$. Indem wir $E(t, d)$ als Teleskopsumme schreiben, $E(0, d) = 0$ berücksichtigen und den Satz von der totalen Wahrscheinlichkeit anwenden, erhalten wir

$$E(t,d) \;=\; \sum_{i=0}^{t-1} \mathrm{E}(L(i+1,d) - L(i,d)) \;\leq\; \sum_{i=0}^{t-1} \sum_{j=1}^{\infty} \mathrm{Prob}(L(i,d-1) = j) \cdot \frac{j}{\mu}.$$

Da die innere Summe gerade $E(i, d-1)/\mu$ ist, folgt auch die dritte Beziehung. □

Beweis von Lemma 7.4.2: Das Ereignis $D(t) \geq d$ ist äquivalent zu $L(t, d) \geq 1$. Laut der Markoffungleichung gilt $\mathrm{Prob}(L(t, d) \geq 1) \leq E(t, d)$, sodass es genügt, die Ungleichung

$$E(t,d) \;\leq\; \frac{1}{d!}\left(\frac{t}{\mu}\right)^d$$

für alle $t \geq 0$ und $d \geq 0$ zu zeigen. Dazu dient folgende Induktion über d.

Für $d = 0$ gilt die Ungleichung wegen Lemma 7.4.4. Für den Induktionsschritt von d nach $d + 1$ wenden wir die Ungleichung aus Lemma 7.4.4 sowie die Induktionsvoraussetzung an und erhalten

$$E(t,d+1) \;\leq\; \frac{1}{\mu}\sum_{i=0}^{t-1} \frac{1}{d!}\left(\frac{i}{\mu}\right)^d \;\leq\; \frac{1}{d!\mu^{d+1}} \int_0^t i^d \mathrm{d}i \;=\; \frac{1}{(d+1)!}\left(\frac{t}{\mu}\right)^{d+1}.$$

Die Abschätzung mithilfe des Integrals ist möglich, da die Untersumme $\sum_{i=0}^{t-1} i^d$ vorliegt. Damit ist die erste Aussage des Lemmas gezeigt.

Die zweite Aussage ist eine einfache Folgerung aus der ersten unter Zuhilfenahme der stirlingschen Formel. Mit $d := 3t/\mu$ erhalten wir

$$\mathrm{Prob}(D(t) \geq d) \;\leq\; \frac{1}{d!}\left(\frac{d}{3}\right)^{d} \;\leq\; \left(\frac{\mathrm{e}}{d}\right)^{d}\left(\frac{d}{3}\right)^{d} \;=\; 2^{-\Omega(d)},$$

da $\mathrm{e}/3$ eine Konstante kleiner als 1 ist. □

An dieser Stelle bemerken wir, dass es möglich zu sein scheint, die vorstehenden Analysen auf $(\mu{+}\lambda)$-EAs mit $\lambda > 1$ zu verallgemeinern. Es drängt sich die Definition eines λ/μ-Baums als Verallgemeinerung des $1/\mu$-Baums auf und erscheint plausibel, dass für Werte von λ mit $\lambda = O(1)$ asymptotisch dieselben Schranken für die Tiefe von Stammbäumen wie in Lemma 7.4.2 bestehen. Dies geht jedoch über den Rahmen dieser Arbeit hinaus.

Wie soll nun Lemma 7.4.2 eingesetzt werden, um untere Schranken für die (erwartete) Laufzeit des $(\mu{+}1)$-EA herzuleiten? Die zugrunde liegende Beweisidee wird bei allen folgenden Schranken gleich sein. Angenommen, wir möchten zeigen, dass der $(\mu{+}1)$-EA mit einer hohen Wahrscheinlichkeit mindestens t^* Schritte zur Optimierung einer gegebenen Funktion f benötigt. Zunächst überlegen wir uns anhand des Lemmas, dass mit einer Wahrscheinlichkeit von mindestens $1 - \mu 2^{-\Omega(t^*/\mu)}$ in der vorgegebenen Zeit keiner der μ Stammbäume des $(\mu{+}1)$-EA die Tiefe $d(t^*) := 3t^*/\mu$ erreicht. Als Nächstes versuchen wir zu belegen, dass außerdem mit einer Wahrscheinlichkeit von mindestens p Folgendes gilt. Innerhalb von t^* Schritten erzeugt ein Lauf des $(\mu{+}1)$-EA in jedem Stammbaum auf allen Knoten mit einer Tiefe von weniger als $d(t^*)$ nur Beschriftungen, die nicht optimal bezüglich f sind. Das Ereignis, das Optimum nicht innerhalb von t^* Schritten zu erreichen, ist mindestens so wahrscheinlich wie der Schnitt des Ereignisses, innerhalb von t^* Schritten keinen Stammbaum mit einer Tiefe von mindestens $d(t^*)$ zu erzeugen, mit dem Ereignis, auf keinem Knoten mit einer Tiefe von weniger als $d(t^*)$ eine optimale Beschriftung zu erzeugen. Wenn die erwähnten Abschätzungen gelingen, haben wir also nachgewiesen, dass die Laufzeit mit einer Wahrscheinlichkeit von mindestens $1 - p - \mu 2^{-\Omega(t^*/\mu)}$ mindestens t^* beträgt. Zugleich folgt durch Multiplikation dieser Wahrscheinlichkeitsschranke mit dem Faktor t^* eine untere Schranke für die erwartete Laufzeit.

7.4.2 Eine allgemeine untere Schranke

In diesem Unterabschnitt werden wir die neue Untere-Schranken-Methode auf eine recht allgemeine Funktionenklasse anwenden, nämlich eine Klasse von Funktionen mit nicht allzu vielen global optimalen Eingaben. Viele der in dieser Arbeit betrachteten Funktionen besitzen sogar nur eine global optimale Eingabe, darunter OneMax, LeadingOnes und SPC. Auch bei den monotonen Polynomen aus Kapitel 3 trifft dies zu, sofern das Polynom von allen Variablen essenziell abhängt. Zudem verfügen auch die schwierigsten Funktionen wie die Nadel-im-Heuhaufen-Funktion $\prod_{i=1}^{n} x_i$ sowie die Funktion Trap (siehe Ackley, 1987; Droste, Jansen und Wegener, 2002a) lediglich über eine global optimale Eingabe. Umgekehrt können wir sagen, dass Funktionen, die sehr viele global optimale Eingaben enthalten, äußerst einfache Funktionen sind.

In Abschnitt 3.1 haben wir den Begriff einer unimodalen Funktion definiert. Unimodale Funktionen besitzen lediglich ein lokales und damit auch nur ein globales Maximum, werden also von unserer unteren Schranke erfasst. Unimodale Funktionen sind bei der theoretischen Analyse evolutionärer Algorithmen stets von besonderem Interesse gewesen. Schon früh äußerte Mühlenbein (1992) die Vermutung, dass der (1+1)-EA sich auf solchen Funktionen effizient verhalte. Dies konnten Droste, Jansen und Wegener (2002a) widerlegen, indem sie eine unimodale Funktion konstruierten, zu deren Optimierung der (1+1)-EA eine exponentielle erwartete Laufzeit benötigt. Nichtsdestoweniger galten unimodale Funktionen weiterhin als im Allgemeinen „einfache" Zielfunktionen für randomisierte Suchheuristiken. Diese Vermutung konnten Droste, Jansen, Tinnefeld und Wegener (2003) allerdings endgültig ausräumen, indem sie den Beweis einer exponentiellen unteren Schranke für die Black-Box-Komplexität unimodaler Funktionen erbrachten. Unser Ergebnis über die erwartete Laufzeit des (μ+1)-EA auf unimodalen Funktionen passt insoweit in diese Diskussion, als wir zeigen können, dass zumindest eine untere Schranke für seine erwartete Laufzeit proportional zu μ ist. Bei unimodalen Funktionen, für die diese Schranke scharf ist, dürfen wir also informal sagen, dass die Funktionen zumindest in dem Sinne „einfach" sind, als eine Population der Größe 1 für den (μ+1)-EA hier asymptotisch optimal ist.

Nach dieser längeren Vorrede kommen wir zur angekündigten allgemeinen unteren Schranke. Wir erfassen damit sogar manche Funktionen, bei denen die Menge der global optimalen Suchpunkte eine exponentielle Größe besitzt. Im Wesentlichen stellen wir die Anforderung, dass die Menge der global optimalen Eingaben durch $2^{o(n)}$ beschränkt ist oder dass alle global optimalen Eingaben einen linearen Hammingabstand zu $n/2$ aufweisen. Im folgenden Theorem benutzen wir die Schreibweise $|x| := x_1 + \cdots + x_n$ für $x \in \{0,1\}^n$.

Theorem 7.4.5 *Seien $f\colon \{0,1\}^n \to \mathbb{R}$, $S_{\text{opt}} := \text{argmax}\{f(x) \mid x \in \{0,1\}^n\}$, $e_{\min} := \min\{|x| \mid x \in S_{\text{opt}}\}$ sowie $e_{\max} := \max\{|x| \mid x \in S_{\text{opt}}\}$. Sei außerdem $\mu = \text{poly}(n)$. Wenn mindestens eine der drei Bedingungen*

$$|S_{\text{opt}}| = 2^{o(n)}, \quad e_{\min} = n/2 + \Omega(n), \quad e_{\max} = n/2 - \Omega(n)$$

gilt, beträgt die erwartete Laufzeit des (μ+1)-EA auf f mindestens $\Omega(\mu n)$ und die Erfolgswahrscheinlichkeit in $c\mu n$ Schritten ist $2^{-\Omega(n)}$, falls die Konstante $c > 0$ klein genug ist. Wenn $|S_{\text{opt}}| = 1$ gilt, beträgt die erwartete Laufzeit des (μ+1)-EA auf f sogar $\Omega(\mu n + n \log n)$.

Beweis: Die untere Schranke $\Omega(n \log n)$ im Fall $|S_{\text{opt}}| = 1$ braucht nur für Werte μ mit $\mu \le c' \log n$ für eine beliebige Konstante $c' > 0$ gezeigt zu werden. Sei daher im Folgenden $\mu \le (\log n)/2$ und $|S_{\text{opt}}| = 1$ angenommen. Wir zeigen die untere Schranke nun mit einer Verallgemeinerung des Sammelbilderproblems, die Droste, Jansen und Wegener (2002a, Lemma 10) in sehr ähnlicher Weise für den (1+1)-EA und lineare Funktionen aufgestellt haben. Sei $y \in S_{\text{opt}}$ das (eindeutige) global optimale Individuum und sei für $1 \leq i \leq n$ mit y_i die Belegung des i-ten Bits von y bezeichnet. Die Wahrscheinlichkeit, dass das i-te Bit eines beliebigen initialen Individuums von y_i abweicht, beträgt genau 1/2. Wegen $\mu \leq (\log n)/2$ und der unabhängigen Wahl der initialen Individuen ist die Wahrscheinlichkeit, dass das i-te Bit in allen initialen Individuen des (μ+1)-EA von y_i abweicht, mindestens $(1/2)^{(\log n)/2} = n^{-1/2}$. Da verschiedene Bits unabhängig initialisiert werden, können wir mit der Chernoffungleichung die Zahl der Bits i, die anfänglich in allen initialen Individuen von y_i abweichen, mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(\sqrt{n})}$ durch $\sqrt{n}/2$ nach unten beschränken. Wir nehmen nun an, dass diese untere Schranke zutrifft. Die Wahrscheinlichkeit, dass mindestens eines der genannten Bits innerhalb von $t := (n-1)(\ln n)/2$ Schritten nie gekippt wird, ist mindestens

$$1 - \left(1 - \left(1 - \frac{1}{n}\right)^t\right)^{\sqrt{n}/2} \geq 1 - \left(1 - \frac{1}{\sqrt{n}}\right)^{\sqrt{n}/2} \geq 1 - \mathrm{e}^{-1/2},$$

womit die untere Schranke t für die Laufzeit mit einer Wahrscheinlichkeit von mindestens $1 - \mathrm{e}^{-1/2} - 2^{-\Omega(\sqrt{n})} = \Omega(1)$ gilt und die untere Schranke $\Omega(n \log n)$ für die erwartete Laufzeit folgt.

Zum Beweis der unteren Schranke $\Omega(\mu n)$ unter einer der drei Bedingungen stellen wir eine Phase der Länge $s := \lfloor c\mu n \rfloor$ für eine Konstante $c > 0$ auf und zeigen, dass der (μ+1)-EA mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ mindestens s Schritte benötigt, falls c klein genug gewählt wird. Wenn diese Behauptung gilt, folgt auch die Aussage über die erwartete Laufzeit und insgesamt das Lemma. Um die Behauptung zu zeigen, wenden wir die im vorigen Unterabschnitt erläuterte Beweisidee auf der Grundlage von Lemma 7.4.2 an. Sei x ein beliebiges initiales Individuum. Gemäß dem genannten Lemma erreicht der Stammbaum $T_s(x)$ mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ die Tiefe von mindestens $3cn$ nicht. Also erreicht er mit mindestens dieser Wahrscheinlichkeit auch die Tiefe von mindestens $7cn$ nicht.

Wir werden zeigen, dass mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ Folgendes gilt: Es gibt in $T_s(x)$ keinen Knoten, der eine Tiefe von weniger als $7cn$ besitzt und den optimalen f-Wert n hat. Dies impliziert gemäß den genannten Überlegungen, dass die Laufzeit mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(s/\mu)} - 2^{-\Omega(n)} = 1 - 2^{-\Omega(n)}$ mindestens s beträgt.

Da der (μ+1)-EA stets gleichverteilt zur Mutation wählt, können wir Lemma 6.4.8, siehe Seite 125, mit $m := n$ und dem Stammbaum $T_s(x)$ anwenden. Wir erhalten, dass es mit einer Wahrscheinlichkeit von höchstens

$$\mathrm{e}^{-7cn/3}\mu \left(1 + \frac{2}{\mu}\right)^{c\mu n} \leq \mathrm{e}^{-7cn/3}\mu \mathrm{e}^{2cn} = 2^{-\Omega(n)}$$

(da $\mu = \text{poly}(n)$) einen Knoten mit einer Tiefe von weniger als $7cn$ in $T_s(x)$ gibt, sodass dieser Knoten einen Hammingabstand von mindestens $14cn$ zu x hat.

Wir unterscheiden nun nach den drei Bedingungen. Im Fall $|S_{\max}| = 2^{o(n)}$ argumentieren wir, dass das zufällige initiale Individuum x im Erwartungswert einen Hammingabstand von $n/2$ zu einem festen $y \in S_{\text{opt}}$ hat und gemäß der Chernoffungleichung ist dieser Hammingabstand mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ mindestens $n/3$. Wegen $|S_{\text{opt}}| = 2^{o(n)}$ ist der Hammingabstand von x zu allen $y \in S_{\text{opt}}$ mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ ebenfalls mindestens $n/3$. Wenn $e_{\min} = n/2+\Omega(n)$, wenden wir die Chernoffungleichung bezüglich des Erwartungswertes $\text{E}(|x|) = n/2$ an. Also ist der Hammingabstand von x zu jedem optimalen Individuum mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ mindestens $\Omega(n)$; Analoges gilt im Fall $e_{\max} = n/2-\Omega(n)$. Für eine genügend kleine Wahl der Konstanten c gilt in allen drei Fällen, dass $14cn$ kleiner ist als die untere Schranke $\Omega(n)$ für den Hammingabstand. Das heißt, dass es mit einer Wahrscheinlichkeit von höchstens $2^{-\Omega(n)}$ in $T_s(x)$ auf einer Tiefe von weniger als $7cn$ einen Knoten mit maximalem f-Wert gibt. Da es nur μ Wahlen für x gibt und $\mu = \text{poly}(n)$ gilt, gibt es mit einer Wahrscheinlichkeit von höchstens $2^{-\Omega(n)}$ nach s Schritten mindestens einen Stammbaum, der einen optimalen Knoten auf einer Tiefe von weniger als $7cn$ enthält. □

Indem wir Theorem 7.4.5 und Theorem 7.3.3 in Verbindung bringen, wird deutlich, dass es zum einen gelungen ist, die erwartete Laufzeit des (μ+1)-EA auf OneMax asymptotisch scharf zu ermitteln. Zweitens zeigt Theorem 7.3.3, dass im Allgemeinen keine asymptotisch größere untere Schranke als in Theorem 7.4.5 angegeben möglich ist. Drittens führen wir nochmals die Schlussfolgerung an, dass zumindest für OneMax die Wahl $\mu = 1$ asymptotisch optimal ist. Zudem zeigt Theorem 7.4.5 für eine große Klasse von Funktionen einschließlich der unimodalen Funktionen, dass große Werte von μ zu großen unteren Schranken der erwarteten Laufzeit führen.

Wir kommen nun auf das im vorangehenden Unterabschnitt angesprochene Beispiel zurück, bei dem sich der (μ+1)-EA ebenso ineffizient wie μ parallele Läufe des (1+1)-EA verhält. Wir definieren dazu die Funktion $f(x) := \min\{2n/3, \text{OneMax}(x)\}$. Man überlegt sich leicht, dass die Laufzeit des (1+1)-EA auf f mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ in der Größenordnung $\Theta(n)$ liegt (dies folgt mit der Chernoffungleichung, da die Wahrscheinlichkeit, dass eine Mutation den f-Wert eines suboptimalen Suchpunkts erhöht, durch eine Konstante nach unten beschränkt ist.) Für $\mu = \text{poly}(n)$ parallele Läufe des (1+1)-EA gilt, dass die parallele Zeit, bis mindestens einer der Läufe das Optimum von f findet, ebenfalls mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ in der Größenordnung $\Theta(n)$ liegt, also mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ die Zahl der Zielfunktionsauswertungen $\Theta(\mu n)$ ist. Mit Lemma 5.2.2 erhalten wir eine obere Schranke von $O(\mu n)$ für die erwartete Laufzeit des (μ+1)-EA auf f. Wir können Theorem 7.4.5 anwenden, da f die Bedingung $e_{\min} = 2n/3$ erfüllt, und erhalten die untere Schranke $\Omega(\mu n)$ für die erwartete Laufzeit des (μ+1)-EA. Folglich handelt es sich bei f um ein konstruiertes Beispiel, bei dem die erwartete Laufzeit von μ parallelen Läufen des (1+1)-EA und die des (μ+1)-EA tatsächlich in derselben Größenordnung liegen.

7.5 Spezielle untere Schranken

Nachdem im vorigen Abschnitt die allgemeine Untere-Schranken-Technik vorgestellt und auf eine große Funktionenklasse, die die Beispielfunktion OneMax enthält, angewandt wurde, werden wir in diesem Abschnitt die Beweistechnik erneut benutzen, um untere Schranken für die beiden verbleibenden Beispielfunktionen nachzuweisen. In beiden Fällen werden wir in Verbindung mit den oberen Schranken asymptotisch fast scharfe Abschätzungen erhalten.

Theorem 7.5.1 *Sei $\mu = \mathrm{poly}(n)$. Dann beträgt die erwartete Laufzeit des (μ+1)-EA auf* LeadingOnes *mindestens $\Omega(\mu n + n^2)$. Außerdem ist die Erfolgswahrscheinlichkeit in $c\mu n$ Schritten höchstens $2^{-\Omega(n)}$, falls die Konstante $c > 0$ klein genug ist.*

Beweis: Im gesamten Beweis gelte zwecks Vereinfachung der Notation die Vereinbarung $f :=$ LeadingOnes. Da f eine Funktion mit einem eindeutigen global optimalen Suchpunkt ist, brauchen wir wegen Theorem 7.4.5 nur die untere Schranke $\Omega(n^2)$ zu zeigen.

Diese Schranke $\Omega(n^2)$ für die erwartete Laufzeit folgt fast unmittelbar aus dem Beweis der entsprechenden unteren Schranke für den (1+1)-EA von Droste, Jansen und Wegener (2002a, Theorem 17). Anstelle des f-Werts des aktuellen Individuums des (1+1)-EA betrachten wir nun das f-Potenzial der Population des (μ+1)-EA, d. h. den maximalen f-Wert ihrer Individuen. Die Wahrscheinlichkeit, das f-Potenzial zu erhöhen, ist stets höchstens $1/n$, da es notwendig ist, dass die linkeste Null des ausgewählten Individuums kippt. Dies ist dieselbe obere Schranke wie für die Wahrscheinlichkeit, dass der (1+1)-EA den f-Wert erhöht. Die nächste entscheidende Beobachtung besteht darin, dass bei einem f-Potenzial von ℓ jedes Individuum der aktuellen Population die Eigenschaft besitzt, dass darin (mindestens) die Bits $\ell + 1, \ldots, n$ gleichverteilt über $\{0, 1\}^{n-\ell}$ sind. Dieses folgt, da der (μ+1)-EA wie der (1+1)-EA zufällig gleichverteilt initialisiert und denselben Mutationsoperator wie der (1+1)-EA besitzt. Damit gelten dieselben Abschätzungen wie im (1+1)-EA für die Zahl der so genannten Trittbrettfahrer, d. i. die zufällige Zahl aufeinander folgender Einsen rechts von der linkesten Null eines Individuums. Insgesamt folgt daraus die untere Schranke $\Omega(n^2)$ mit denselben Argumenten wie im genannten Theorem von Droste, Jansen und Wegener (2002a). □

Die untere Schranke aus Theorem 7.5.1 zeigt in Verbindung mit Theorem 7.3.2, dass die Wahl $\mu = 1$ bei LeadingOnes asymptotisch zur geringsten erwarteten Laufzeit führt, die Population also in diesem Sinne keinen Nutzen hat. Allerdings sind die Schranken der beiden Theoreme um einen Faktor $\Theta(\log n)$ voneinander entfernt, wenn μ nicht zu klein ist. Man darf vermuten, dass der im Beweis der oberen Schranke beschriebene Prozess einem typischen Lauf des (μ+1)-EA auf LeadingOnes ähnelt. Wir könnten dann versuchen, eine untere Schranke $\Omega(\mu n \log n)$ für seine erwartete Laufzeit zu zeigen. Um die in Abschnitt 7.4.1 vorgestellte Technik anwenden zu können, müssten wir nachweisen, dass zur Optimierung der Funktion jeder Stammbaum mit hoher Wahrscheinlichkeit eine Tiefe von $\Omega(n \log n)$ benötigte. Dies mutet schwieriger als im letzten Abschnitt an, da wir dann nicht nur über den Hammingabstand, vgl. Lemma 6.4.8, argumentieren müssten. Als größtes Problem erscheint die bislang fehlende Charakterisierung der Struktur von Pfaden in Stammbäumen und der Bedingungen, unter denen neue Knoten erzeugt werden. Dies ist mit Lemma 7.3.5 zwar teilweise für die Funktion SPC gelungen, aber die dazugehörigen Argumente lassen sich nicht ohne weiteres auf den Fall der Zielfunktion LeadingOnes übertragen.

Wir kommen jetzt zur letzten der drei Beispielfunktionen. Wie in Abschnitt 7.2 bereits angesprochen, könnte man zunächst mutmaßen, dass die Erforschung des Plateaus der Funktion SPC bei großen Werten von μ zu einer geringeren erwarteten Laufzeit des (μ+1)-EA führt. Da das Plateau anhand seines Funktionswertes ohnehin keine Hinweise gibt, wäre damit auch eine Rechtfertigung für den gleichverteilt wählenden Selektionsoperator des (μ+1)-EA gefunden.

Um diese Frage zu klären, stellen wir im Folgenden auch eine untere Schranke für die erwartete Laufzeit des (μ+1)-EA auf SPC auf. Tatsächlich beruht die Beweisidee wiederum darauf, Situationen zu analysieren, in denen die gesamte Population aus Plateaupunkten (diesen Begriff haben wir im Zusammenhang mit Theorem 7.3.4 auf Seite 150 eingeführt) besteht. Die hier gewonnene untere Schranke, die von der Größenordnung $\Omega(\mu n^3/\log \mu)$ sein wird, ist leider wiederum nur fast asymptotisch scharf. Jedoch gilt $\Omega(\mu n^3/\log \mu) = \Omega(n^3)$, während der (1+1)-EA auf SPC im Erwartungswert $O(n^3)$ Schritte benötigt. Also hilft auch hier die Population bezüglich der asymptotischen erwarteten Laufzeit nicht.

Theorem 7.5.2 *Sei $\mu = \text{poly}(n)$. Dann beträgt die erwartete Laufzeit des (μ+1)-EA auf* SPC *mindestens $\Omega(\mu n^3/\log \mu)$. Außerdem ist die Erfolgswahrscheinlichkeit innerhalb von $c\mu n^3/\log \mu$ Schritten $2^{-\Omega(\log^2 \mu)}$, falls die Konstante $c > 0$ klein genug ist.*

Zum Beweis von Theorem 7.5.2 wenden wir wiederum die in Abschnitt 7.4.1 vorgestellte Technik an. Wir betrachten eine Phase geeigneter, polynomieller Länge s und die Stammbäume $T_s(x)$ für die initialen Individuen x. Das Lemma wird folgen, indem wir zeigen, dass mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\log^2 \mu)}$ in keinem $T_s(x)$ ein Knoten mit maximalem SPC-Wert $2n$ auf einer Tiefe von weniger als $3cn^3/\log \mu$ für eine geeignete Konstante $c > 0$ liegt.

Um die Gültigkeit dieser Aussage nachzuweisen, betrachten wir für ein beliebiges x einen beliebigen bei x startenden Pfad p in $T_s(x)$ und argumentieren über dessen Beschriftungen. Aus technischen Gründen sind wir an Knoten auf dem Pfad interessiert, die nicht nur suboptimal, sondern auch weit entfernt von 0^n und 1^n sind. Daher analysieren wir lediglich einen Teilpfad $\tilde{p}$ von p, der die folgenden Eigenschaften hat: Alle

Knoten auf $\tilde{p}$ sind mit Suchpunkten von der Gestalt 1^i0^{n-i} mit $n/4 \leq i \leq 3n/4$ beschriftet und kein Vorgänger des Startknotens von $\tilde{p}$ ist optimal. Natürlich besteht die Möglichkeit, dass kein solcher Teilpfad existiert. Allerdings wollen wir zeigen, dass $\tilde{p}$ mit hoher Wahrscheinlichkeit existiert. Außerdem wollen wir nachweisen, dass $\tilde{p}$ mit hoher Wahrscheinlichkeit eine Länge von mindestens $3cn^3/\log\mu$ für genügend kleines c hat. Dabei nehmen wir pessimistisch an, dass p entsprechend lang ist.

Die Aussage über die Länge von $\tilde{p}$ wird in zwei Schritten bewiesen. Erstens zeigen wir, dass sich mit einer hohen Wahrscheinlichkeit die Suchpunkte entlang $\tilde{p}$ mindestens $\Omega(n^2/\log\mu)$-mal ändern. Zweitens ist der Abstand zweier Knoten mit unterschiedlichen Suchpunkten auf $\tilde{p}$ mit einer Wahrscheinlichkeit von $\Omega(1)$ mindestens $\Omega(n)$.

Wie schon erwähnt, konzentrieren wir uns auf Zeitpunkte, zu denen die gesamte Population des (μ+1)-EA aus Plateaupunkten, d. h. Individuen der Gestalt 1^i0^{n-i} mit $0 \leq i \leq n$, besteht. Es bezeichne t_{PL1} den ersten Zeitpunkt, an dem mindestens ein Plateaupunkt in der aktuellen Population ist. Außerdem bezeichnet t_{PLA} immer noch den ersten Zeitpunkt, zu dem μ Plateaupunkte in der Population sind. Im Folgenden zeigen wir zunächst, dass es sehr unwahrscheinlich ist, dass der (μ+1)-EA vor dem Zeitpunkt t_{PLA} das Optimum findet.

Lemma 7.5.3 *Mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\sqrt{n})}$ enthalten alle Populationen bis zum Zeitpunkt t_{PLA} (einschließlich t_{PLA}) nur Individuen mit höchstens $3n/5$ Einsen.*

Beweis: Als Erstes zeigen wir, dass alle bis zur Zeit t_{PL1} erzeugten Individuen mit einer Wahrscheinlichkeit von $1-2^{-\Omega(n)}$ höchstens $4n/7$ Einsen haben. Da die initialen Individuen gleichverteilt gezogen werden und wir $\mu = \text{poly}(n)$ annehmen, folgt mithilfe der Chernoffungleichung, dass die Wahrscheinlichkeit eines initialen Individuums mit mehr als $5n/9$ Einsen durch $2^{-\Omega(n)}$ nach oben beschränkt ist. Im letzten Absatz des Beweises von Theorem 7.3.4 haben wir begründet, dass $\text{E}(t_{\text{PL1}}) = O(\mu n \log n)$ gilt. Durch Anwendung der Markoffungleichung und durch Wiederholung unabhängiger Phasen gelangen wir zu der Aussage, dass $t_{\text{PLA}} = O(\mu n^2 \log n)$ mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ gilt. Da wegen $\mu = \text{poly}(n)$ das Ereignis, dass mindestens einmal innerhalb von $O(\mu n^2 \log n)$ Schritten mindestens $\Omega(n)$ Bits gleichzeitig kippen, eine Wahrscheinlichkeit von $2^{-\Omega(n \log n)}$ hat, besitzt das (einzige) Plateauindividuum zur Zeit t_{PL1} mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ höchstens $4n/7$ Einsen und alle übrigen Individuen der Population höchstens $5n/9$ Einsen mit mindestens derselben Wahrscheinlichkeit. Im Folgenden nehmen wir an, dass dieses Ereignis eintritt.

Im letzten Absatz des Beweises von Theorem 7.3.4 wurde auch gezeigt, dass der Erwartungswert von $t_{\text{PLA}} - t_{\text{PL1}}$ durch $O(\mu \log \mu)$ beschränkt ist. Außerdem ist die Zeit zwischen t_{PL1} und t_{PLA} gemäß der Markoffungleichung und der Wiederholung unabhängiger Phasen mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\sqrt{n})}$ durch $O(\mu\sqrt{n}\log\mu)$ beschränkt. Da laut Lemma 7.4.2 mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\sqrt{n}\log\mu)}$ innerhalb von $O(\mu\sqrt{n}\log\mu)$ Schritten jeder Stammbaum eine Tiefe von höchstens

$O(\sqrt{n}\log\mu)$ erhält, können wir die Zahl der Einsen der Individuen zur Zeit t_{PLA} wiederum mit Lemma 6.4.8 wie folgt beschränken. Da $(1+2/\mu)^{O(\mu\sqrt{n}\log\mu)} = 2^{O(\sqrt{n}\log n)}$ und $\mu = \mathrm{poly}(n)$ gilt, gibt es mit einer Wahrscheinlichkeit von höchstens $2^{-\Omega(n)}$ einen Stammbaum, der einen Knoten enthält, der eine Tiefe von höchstens $n/70$ und mindestens $n/35$ Einsen mehr als die Wurzel hat. Folglich besitzt mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(\sqrt{n})}$ kein Individuum zum Zeitpunkt t_{PLA} mehr als $4n/7 + n/35 = 3n/5$ Einsen. □

Nun können wir $\tilde{p}$, den Teilpfad von p mit einer schönen Struktur, analysieren. Zur Beschreibung der Mutationen, die die Beschriftungen seiner Knoten erzeugen, greifen wir auf die Charakterisierung aus Lemma 7.3.5 zurück. Um den Fall, dass μ nicht mit n wächst, in den Griff zu bekommen, müssen wir im folgenden Lemma vorsichtig sein, wenn wir die O-Notation bezüglich μ einsetzen wollen. Wir wiederholen noch einmal, dass wir pessimistisch davon ausgehen, dass der Pfad p die Mindestlänge, die für die folgende Aussage nötig ist, auch besitzt.

Lemma 7.5.4 *Mit einer Wahrscheinlichkeit von mindestens* $\max\{1-2^{-\Omega(\log^2\mu)}, \Omega(1)\}$ *existiert ein Teilpfad* $\tilde{p}$ *von* p*, sodass alle Knoten von* $\tilde{p}$ *zum Zeitpunkt* t_{PLA} *oder später erzeugt werden, kein Vorgänger des Startknotens von* $\tilde{p}$ *optimal ist,* $\tilde{p}$ *nur Suchpunkte der Gestalt* 1^i0^{n-i} *mit* $n/4 \le i \le 3n/4$ *enthält und sich die Suchpunkte entlang* $\tilde{p}$ $\Omega(n^2/\log\mu)$*-mal ändern.*

Beweis: Sei v der letzte Suchpunkt auf p, der bis zur (einschließlich der) Zeit t_{PLA} erzeugt wird. Gemäß Lemma 7.5.3 enthält v mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(\sqrt{n})}$ höchstens $3n/5$ Einsen. Beginnend ab dem Zeitpunkt t_{PLA} wird jedes Individuum, das kein Plateaupunkt ist, unmittelbar nach seiner Erzeugung gelöscht. Wir identifizieren die Knoten in Stammbäumen wieder mit ihren Beschriftungen. Ohne Beschränkung der Allgemeinheit sind v und alle Nachfolger von v auf p Plateaupunkte, da der Pfad sonst vorzeitig endet. Die Wahrscheinlichkeit, dass der (μ+1)-EA mindestens einmal in polynomiell vielen Schritten $\Omega(n)$ Bits auf einmal kippt, ist durch $2^{-\Omega(n\log n)}$ nach oben beschränkt. Folglich erhalten wir, dass es mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\sqrt{n})}$ einen Nachfolger v^* für v auf p gibt, der mindestens $n/3$ und immer noch höchstens $2n/3$ Einsen besitzt. Also haben wir mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\sqrt{n})}$ einen Startpunkt v^* für $\tilde{p}$ identifiziert, welcher zur Zeit t_{PLA} oder später erzeugt wird und der keinen optimalen Vorgänger auf p hat. Der Pfad $\tilde{p}$ endet vor dem ersten Suchpunkt auf p, der nicht von der Gestalt 1^i0^{n-i} mit $n/4 \le i \le 3n/4$ ist. Nun verbleibt zu zeigen, dass es mit der gewünschten Wahrscheinlichkeit mindestens $\Omega(n^2/\log\mu)$ Knoten gibt, die ein anderer Plateaupunkt als ihr Elter sind.

Der Suchpunkt 1^i0^{n-i}, $n/3 \le i \le 2n/3$, des Startknotens v^* hat einen linearen Hammingabstand zum optimalen Suchpunkt 1^n. Wir wollen zeigen, dass die Irrfahrt, die die verschiedenen Suchpunkte der Knoten auf $\tilde{p}$ beschreibt, eine große Ähnlichkeit zu der eindimensionalen, symmetrischen Irrfahrt auf der Linie der Länge $\Theta(n)$, bei der mit einer Wahrscheinlichkeit von jeweils 1/2 nach links oder rechts gegangen wird, aufweist. Auf dieser Idee beruht auch der Beweis der oberen Schranke $O(n^3)$ für die erwartete Laufzeit des (1+1)-EA auf SPC, siehe Jansen und Wegener (2001a). Für letztere Irrfahrt ist wohl bekannt (einfache Anwendung der Chernoffungleichung), dass sie sich mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\log^2 \mu)}$ innerhalb von $O(n^2/\log \mu)$ Schritten nicht um eine Distanz $\Omega(n)$ vom Startpunkt entfernt.

Da wir Pfade betrachten, die zu mindestens einem Zeitpunkt lebendig sind, können wir Lemma 7.3.5, angefangen mit dem Stammbaum von v^*, anwenden und über die Verteilung der Suchpunkte, mit denen die Nachfolger von v^* beschriftet sind, argumentieren. Daher betrachten wir die ersten höchstens $s := \lfloor n^2/(50(\log \mu + 1)) \rfloor$ Knoten auf p, die ein anderer Suchpunkt als ihr Elter sind. Jeder dieser speziellen Knoten ist das Ergebnis einer Mutation, die einen Suchpunkt 1^i0^{n-i} zu einem Suchpunkt 1^j0^{n-j} mit $j \neq i$ verändert hat. Wir betrachten auch diese s Mutationen. Im Gegensatz zur genannten Irrfahrt kann $|j - i| > 1$ sein, es können also Schritte auf dem Plateau mit einer Schrittweite von mehr als 1 stattfinden.

Wir betrachten das Ereignis, dass der (μ+1)-EA das i-te Bit eines Suchpunktes 1^i0^{n-i} kippt. Unter der Bedingung, dass diese Mutation zu einem anderen Plateaupunkt führt, ist es am wahrscheinlichsten, dass $1^{i-1}0^{n-i+1}$ erzeugt wird. Um nämlich den Plateaupunkt $1^{i-1-k}0^{n-i+1+k}$ für ein $k \geq 1$ zu erzeugen, müssen die Bits $i - k, \ldots, i - 1$, so genannte zusätzlich kippende Bits, gleichzeitig gekippt werden. Die dazugehörige Wahrscheinlichkeit ist für den Mutationsoperator des (μ+1)-EA durch $1/n^k$ nach oben beschränkt. Analoges gilt für den Fall, dass das $(i + 1)$-te Bit gekippt wird. Daher ist die erwartete Zahl zusätzlich kippender Bits in s Mutationen nach oben beschränkt durch $\sum_{i=1}^{n} s/n^i \leq 2n/(50(\log \mu + 1))$ und die Wahrscheinlichkeit von mindestens $n/24$ zusätzlich kippenden Bits ist gemäß der Chernoffungleichung höchstens $2^{-\Omega(n)}$. Darauf werden wir im folgenden Absatz zurückkommen.

Im Folgenden nehmen wir an, dass keine der s Mutationen mindestens $n/4$ Bits auf einmal kippt, und führen damit nur einen Fehlerterm der Größenordnung $2^{-\Omega(n \log n)}$ ein. Nun gilt dank dieser Annahme Folgendes. Gegeben, dass der Suchpunkt 1^i0^{n-i} mit $n/4 \leq i \leq 3n/4$ in einen anderen Plateaupunkt mutiert wird, sind die Wahrscheinlichkeit, dass das i-te Bit kippt, und die Wahrscheinlichkeit, dass das $(i + 1)$-te Bit kippt, gleich. Also ist die Wahrscheinlichkeit, einen Suchpunkt mit mehr Einsen zu erzeugen, ebenso groß wie die Wahrscheinlichkeit, einen mit weniger Einsen zu erzeugen, d. h., wir haben die Verbindung zur oben erwähnten symmetrischen Irrfahrt hergestellt. In s Mutationen erwarten wir genau $s/2$ Mutationen, die die Zahl der Einsen erhöhen, und genau $s/2$ Mutationen, die die Zahl der Einsen verringern, sofern es nicht passiert, dass ein Suchpunkt mit mehr als $3n/4$ oder weniger als $n/4$ Einsen dabei entsteht. Das letztere Ereignis nennen wir einen Fehler. Die Wahrscheinlichkeit, in s Mutationen einen Überschuss von mindestens $n/25$ erhöhenden oder verringernden Mutationen zu haben, ist, sofern kein Fehler passiert, gemäß der Chernoffungleichung für $\delta = (2/n)(\log \mu + 1)$ sowohl durch $2^{-\Omega(\log^2 \mu)}$ als auch durch $1 - \Omega(1)$ nach oben beschränkt. Wir erinnern uns, dass der Suchpunkt von v^* mindestens $n/3$ und höchstens

$2n/3$ Einsen enthält. Indem wir die obere Schranke für die Anzahl zusätzlich kippender Bits hinzuaddieren, erhalten wir insgesamt, dass die Wahrscheinlichkeit, innerhalb der s Mutationen einen Suchpunkt der Gestalt 1^i0^{n-i} mit $i < n/4$ oder $i > 3n/4$ zu erzeugen, höchstens $\max\{2^{-\Omega(\log^2 \mu)}, 1-\Omega(1)\}$ beträgt. Damit wird auch die Annahme, dass kein Fehler passiert, gerechtfertigt. □

Nun muss noch der Abstand zweier Knoten auf $\tilde{p}$, an denen sich gegenüber dem Elter der Suchpunkt ändert, abgeschätzt werden. Wir nennen solche Knoten *Wechselknoten*. Um den Abstand von Wechselknoten nach unten zu beschränken, greifen wir erneut auf Lemma 7.3.5 zurück.

Lemma 7.5.5 *Mit einer Wahrscheinlichkeit von $\Omega(1)$ beträgt der Abstand zweier aufeinander folgender Wechselknoten v, v' auf $\tilde{p}$ mindestens $\Omega(n)$. Dies gilt unabhängig von den Vorgängern von v auf $\tilde{p}$.*

Beweis: Zunächst betrachten wir anstelle von $\tilde{p}$ den Oberpfad p, und zwar an Knoten, die weder mit 1^n beschriftet sind noch Nachfolger eines solchen Knotens sind. Sei w ein beliebiger zur Zeit t_{PLA} oder später erzeugter Knoten auf p und sei w' der Nachfolger von w auf dem betrachteten Teilstück von p. Nach Voraussetzung ist der Pfad von w nach w' zu mindestens einem Zeitpunkt lebendig, sodass wir Lemma 7.3.5 mit $x_0 := w$ anwenden können. Wir identifizieren die Knoten wieder mit ihren Beschriftungen. Also wissen wir, dass w' das Ergebnis einer Mutation unter der Bedingung ist, dass w, ein Plateaupunkt, in einen von 1^n verschiedenen Plateaupunkt mutiert wird.

Die unbedingte Wahrscheinlichkeit, einen Plateaupunkt 1^i0^{n-i} für $0 \leq i \leq n-1$ in einen anderen Plateaupunkt zu mutieren, ist durch $2/n$ nach oben beschränkt, da das i-te oder das $(i+1)$-te Bit kippen muss. Die Wahrscheinlichkeit, einen Plateaupunkt in einen Plateaupunkt zu mutieren, ist mindestens $(1-1/n)^n \geq 1/(2\mathrm{e})$, da es reicht, ein Replikat zu erzeugen. Folglich beträgt die Wahrscheinlichkeit, dass sich w' von w unterscheidet, höchstens $4\mathrm{e}/n$. Wenn sich w' nicht unterscheidet, wenden wir Lemma 7.3.5 mit $x_0 := w'$ an. Also gilt mit einer Wahrscheinlichkeit von mindestens $1/2$, dass die ersten $n/(8\mathrm{e})$ Nachfolger von w auf p derselbe Suchpunkt wie w sind.

In Wirklichkeit müssen wir die Knoten auf dem Teilpfad $\tilde{p}$ betrachten. Per definitionem werden diese nicht vor dem Zeitpunkt t_{PLA} erzeugt. Aber $\tilde{p}$ folgt gemäß seiner Definition der Bedingung, dass nur Knoten der Gestalt 1^i0^{n-i} mit $n/4 \leq i \leq 3n/4$ darauf liegen, und der Bedingung, dass sich $\Omega(n^2/\log \mu)$-mal der Suchpunkt ändert. Die erste Bedingung erhöht lediglich die Wahrscheinlichkeit, ein Replikat zu erzeugen, zu unseren Gunsten und die zweite Bedingung beeinflusst den Abstand von Wechselknoten auf $\tilde{p}$ nicht. Zuletzt machen wir uns klar, dass wir für die oben betrachteten Knoten w auch Wechselknoten einsetzen dürfen, da die Definition eines Wechselknotens lediglich Bedingungen an dessen Vorgänger stellt. Damit ist auch der Abstand zweier aufeinander folgender Wechselknoten v, v' auf $\tilde{p}$ mit einer Wahrscheinlichkeit von mindestens $1/2$ mindestens $n/(8\mathrm{e})$. Dass dies unabhängig von den Vorgängern von v gilt, folgt aus der Unabhängigkeit einzelner Anwendungen des Mutationsoperators des (μ+1)-EA, die entlang $\tilde{p}$ erhalten bleibt, auch wenn manche Mutationen ausgeschlossen sind. $\square$

Nun können wir die drei vorangehenden Lemmata zusammensetzen.

Beweis von Theorem 7.5.2: Indem wir im gesamten Beweis annehmen, dass das in Lemma 7.5.3 beschriebene Ereignis eintritt, führen wir nur einen Term $2^{-\Omega(\sqrt{n})}$ für die Fehlerwahrscheinlichkeit ein. Als Erstes betrachten wir den trivialen Fall $\mu = O(1)$. Hier liefert uns Lemma 7.5.4, dass für eine Konstante $c > 0$ mit Wahrscheinlichkeit $\Omega(1)$ mindestens cn^2 Mutationen, die Plateaupunkte zu anderen Plateaupunkten mutieren, erforderlich sind. Jede dieser Mutationen hat eine Wahrscheinlichkeit von nur $O(1/n)$. Damit folgt das Theorem in diesem Fall mit einer Anwendung der Chernoffungleichung.

Sei nun $\mu = \omega(1)$. Sei p immer noch ein beliebiger Pfad im Stammbaum $T_s(x)$ für ein beliebiges initiales Individuum x. Wir nehmen nun an, dass ein Teilpfad $\tilde{p}$ wie in Lemma 7.5.4 beschrieben existiert. Zusammen mit Lemma 7.5.5 folgt, dass die erwartete Zahl von Wechselknoten auf $\tilde{p}$ mit einem Abstand von $\Omega(n)$ mindestens $\Omega(n^2/\log\mu)$ beträgt. Aufgrund der in Lemma 7.5.5 beschriebenen Unabhängigkeit können wir die Chernoffungleichung anwenden. Es folgt, dass die erwähnte Zahl mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^2/\log\mu)}$ mindestens $\Omega(n^2/\log\mu)$ und damit die Länge von $\tilde{p}$ mindestens $\Omega(n^3/\log\mu)$ beträgt. Zusammen mit den Fehlerwahrscheinlichkeiten aus den Lemmata 7.5.3 und 7.5.4 erhalten wir, dass mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(\log^2\mu)}$ die Länge von $\tilde{p}$ mindestens $\Omega(n^3/\log\mu)$ beträgt oder p vorzeitig stirbt, also mit einem nicht optimalen Knoten endet.

Die Wahrscheinlichkeit, dass irgendein Stammbaum innerhalb von $\lfloor c\mu n^3/\log\mu\rfloor$ Schritten eine Tiefe von $u := 3cn^3/\log\mu$ erreicht, ist wegen $\mu = \text{poly}(n)$ beschränkt durch $2^{-\Omega(n^3/\log\mu)} = 2^{-\Omega(\log^2\mu)}$. Die Zahl der lebendigen Pfade ist über alle Stammbäume trivialerweise stets durch μ nach oben (und unten) beschränkt. Also ist die Wahrscheinlichkeit, dass in irgendeinem Stammbaum mindestens ein Pfad mit einem optimalen Knoten und einer Länge von weniger als $\ell := c'n^3/\log\mu$, $c' > 0$ eine geeignete Konstante, existiert, höchstens $\mu 2^{-\Omega(\log^2\mu)} = 2^{-\Omega(\log^2\mu)}$. Wenn c klein genug und n groß genug gewählt wird, ist u kleiner als ℓ und das Theorem folgt aufgrund der allgemeinen Beweistechnik aus Abschnitt 7.4.1. $\square$

Da die obere Schranke aus Theorem 7.3.4 und die untere Schranke aus Theorem 7.5.2 um einen asymptotischen Faktor $\Theta(\log\mu)$ voneinander abweichen, sollten wir diskutieren, wo die asymptotisch wahre Laufzeit liegen könnte. Zu diesem Zweck ist es zunächst hilfreich, parallele Läufe des (1+1)-EA auf SPC zu betrachten. Im Beweis der oberen Schranke für den (1+1)-EA zeigen Jansen und Wegener (2001a) im Wesentlichen, dass die erwartete Laufzeit beschränkt ist durch $O(n \cdot s(n))$, wenn $s(n)$ die Zahl der unabhängigen Bernoulliversuche mit Erfolgswahrscheinlichkeit 1/2 ist, die man durchführen muss, um mit konstanter Wahrscheinlichkeit $s(n)/2 + n$ Erfolge zu erzielen.

Wir brauchen also im Gegensatz zur Chernoffungleichung eine untere Schranke für die Wahrscheinlichkeit, dass eine binomialverteilte Zufallsvariable um einen gewissen Wert von ihrem Erwartungswert abweicht.

Laut Lemma A.14 können wir die Wahrscheinlichkeit von $s(n)/2+n$ Erfolgen in $s(n)$ Versuchen der genannten Gestalt für $s(n) := \lceil cn^2/\sqrt{\log \mu}\rceil$ und eine geeignete Konstante $c > 0$ durch $\Omega(1/\mu)$ nach unten beschränken. Also ist die Erfolgswahrscheinlichkeit eines Laufes des (1+1)-EA innerhalb von $cn^3/\sqrt{\log \mu}$ Schritten immerhin durch $\Omega(1/\mu)$ abgeschätzt. Daraus folgt, dass μ parallele Läufe des (1+1)-EA nach einer erwarteten parallelen Zeit von $O(n^3/\sqrt{\log \mu})$ das Optimum erreichen, d. h., dass die erwartete Zahl von Zielfunktionsauswertungen durch $O(\mu n^3/\sqrt{\log \mu})$ beschränkt ist. Deshalb könnte man vermuten, dass die erwartete Laufzeit des (μ+1)-EA auf SPC ebenfalls durch $O(\mu n^3/\text{polylog}(\mu))$ nach oben beschränkt ist.

Auf der anderen Seite könnte man sich an alternativen Beweismethoden für untere Schranken für die erwartete Laufzeit des (μ+1)-EA auf SPC versuchen, um die Lücke zu schließen. Wir schlagen einen Ansatz vor, der die Lebensdauer von Stammbäumen mit einbezieht. Damit wollen wir Teilbäume von Stammbäumen identifizieren, die ziemlich schnell vollständig aussterben, d. h. keine lebendigen Knoten mehr enthalten. Solche Teilbäume enthalten dann mit hoher Wahrscheinlichkeit keinen optimalen Knoten. Wir betrachten in unserem Ansatz eine Population P_t zu einem Zeitpunkt $t \geq t_{\text{PLA}}$, nehmen an, dass alle Individuen in P_t höchstens $3n/4$ Einsen besitzen und wählen $x \in P_t$ beliebig. Nun untersuchen wir den Prozess, der den Stammbaum $T(x)$ erzeugt, ignorieren aber Knoten, die keine Plateaupunkte sind. Die Zahl der lebendigen Knoten in $T(x)$ lässt sich als der Zustand einer Markoffkette auffassen, sodass uns eine Irrfahrt auf den Zahlen $\{0, \ldots, \mu\}$ mit Startzustand 1 vorliegt. Bei dieser Irrfahrt zählen wir lediglich diejenigen Schritte des (μ+1)-EA, die die Zahl der lebendigen Knoten innerhalb von $T(x)$ beeinflussen. Wenn diese Zahl aktuell größer als 1 ist, impliziert die Voraussetzung, nur Knoten, die Plateaupunkte sind, zu betrachten, dass sowohl die Wahrscheinlichkeit, einen lebendigen Knoten zu $T(x)$ hinzuzufügen, als auch die Wahrscheinlichkeit, einen lebendigen Knoten aus $T(x)$ zu löschen, durch den Ausdruck $(i/\mu)((\mu - i)/(\mu + 1))$ gegeben ist. (Dieses kann man sich klar machen, indem man einfach die Auswahlwahrscheinlichkeiten zur Reproduktion und zur Ersetzung heranzieht.) Demzufolge liegt eine zumindest in den Zuständen $\{2, \ldots, \mu - 1\}$ symmetrische Irrfahrt vor. Der Zustand 1 stellt zunächst eine Ausnahme dar, da der (μ+1)-EA immer mindestens einen lebendigen Knoten in irgendeinem Stammbaum vorhält. Dies ignorieren wir jetzt und nehmen vereinfachend an, dass die Übergänge auch im Zustand 1 jeweils mit Wahrscheinlichkeit 1/2 in den Zustand 0 bzw. 2 führen. Wir bezeichnen die erhaltene Markoffkette mit A und beachten, dass sie zwei absorbierende Zustände enthält. In der Literatur tritt diese einfache und symmetrische Markoffkette häufig als einführendes Beispiel auf, siehe beispielsweise Grinstead und Snell (1997).

Als Nächstes interessieren wir uns für die Zeit, bis die Markoffkette A in einen absorbierenden Zustand gerät. Dies ist genau die Fragestellung nach der Zeit bis zum

Ende des Spiels im fairen Gambler's-Ruin-Problem (Feller, 1968) bei einem Startkapital von 1 und Gesamtkapital von μ. Die erwartete Länge des Spiels beträgt μ und gemäß der Markoffungleichung ist die Länge mit einer Wahrscheinlichkeit von mindestens $1-1/\mu$ höchstens μ^2. Wenn die Markoffkette A im absorbierenden Zustand 0 landet, ist der Stammbaum $T(x)$ vollständig tot, und wenn sie im Zustand μ landet, sind die Stammbäume aller $x' \in P_t$ mit $x' \neq x$ vollständig tot. Somit erhalten wir unter der oben erwähnten vereinfachenden Annahme, dass jeder Stammbaum $T(x)$ für $x \in P_t$ mit einer Wahrscheinlichkeit von mindestens $1-1/\mu$ nach $O(\mu^2)$ Schritten vollständig tot ist. Allerdings haben wir ignoriert, dass mindestens ein Stammbaum nie vollständig sterben kann. Für jeden Stammbaum ist die Wahrscheinlichkeit, dass er nicht vollständig aussterben kann, höchstens $1/\mu$. Indem wir dies mit einbeziehen, folgt, dass jeder Stammbaum $T(x)$ mit einer Wahrscheinlichkeit von höchstens $2/\mu$ nach $O(\mu^2)$ Schritten noch nicht vollständig tot ist.

Nun unterscheiden wir zwei Fälle in Bezug auf μ. Sei $\mu = O(n^{3/2-\varepsilon})$ für eine Konstante $\varepsilon > 0$. Dann ist μ^2 asymptotisch kleiner als $\Omega(n^3/\log \mu)$. Den letzten Ausdruck haben wir im Beweis von Theorem 7.5.1 als eine mit hoher Wahrscheinlichkeit geltende untere Schranke für die Tiefe eines Stammbaums, der das Optimum enthält, ermittelt. Dies heißt, dass mit einer Wahrscheinlichkeit von $1-O(1/\mu)$ ein Stammbaum $T(x)$ ausstirbt, bevor ein optimaler Knoten 1^n darin entstehen kann. Dies würde bedeuten, dass, solange die aktuelle Population weit vom Optimum entfernt ist, ein erwarteter Anteil von $\Omega((\mu-1)/\mu)$ aller Schritte vergeudet wird. Dieses alles deutet darauf hin, dass jetzt eine Schranke von $\Omega(n^3)$ für die erwartete Laufzeit des (μ+1)-EA gilt.

Leider bricht diese Argumentationskette zusammen, falls $\mu = \omega(n^{3/2}/\log \mu)$ gilt. Aber für große Werte von μ, z. B. $\mu = \Omega(n^{3+\varepsilon})$ für eine Konstante $\varepsilon > 0$, gilt vielleicht eine kleinere obere Schranke für die Laufzeit. Gemäß der Theorie für das Gambler's-Ruin-Problem gilt, dass der Stammbaum $T(x)$ mit einer Wahrscheinlichkeit von mindestens $1/\mu^{3+2\varepsilon/3}$ mindestens einmal $\mu^{3+2\varepsilon/3}$ lebendige Knoten enthält, bevor er ausstirbt. Damit die Markoffkette A vom Zustand $\mu^{3+2\varepsilon/3}$ zum Zustand 0 gelangen kann, sind mit einer Wahrscheinlichkeit von $\Omega(1)$ mindestens $\Omega(\mu^{6+4\varepsilon/3})$ Schritte nötig, was asymptotisch größer als $O(\mu n^3)$, unsere obere Schranke für die erwartete Laufzeit aus Theorem 7.3.4, ist. Indem wir alle Individuen $x \in P_t$ betrachten, erhalten wir, dass ein erwarteter Anteil von $\Omega(\mu^{\varepsilon/3})$ seiner Individuen in $O(\mu n^3)$ Schritten stets einen lebendigen Knoten in seinem Stammbaum zu haben scheint. Natürlich sind die Irrfahrten, die die Zahl der lebendigen Knoten in verschiedenen Stammbäumen $T(x)$ beschreiben, nicht unabhängig voneinander, da eine hohe Zahl lebendiger Knoten in einem Stammbaum eine geringe Zahl in einem anderen Stammbaum begünstigt. Wenn allerdings μ groß genug ist, scheint es so zu sein, dass mit hoher Wahrscheinlichkeit ein polynomiell großer Teil der initialen Population stets einen lebendigen Nachfahren innerhalb von $O(\mu n^3)$ Schritten besitzt. Dies ist ein bemerkenswerter Bezug zum oben erwähnten parallelen Lauf. Wenn diese Vermutung gilt, kann man anscheinend mithilfe der Argumente aus dem Beweis von Theorem 7.3.4 eine obere Schranke von $O(\mu n^3/\text{polylog}(\mu))$ für die erwartete Laufzeit des (μ+1)-EA auf SPC und große μ zeigen.

Insgesamt lassen die vorangehenden Argumente vermuten, dass die erwartete Laufzeit des (μ+1)-EA auf SPC von der Größenordnung $\Theta(\mu n^3/\max\{\log(\mu/f(n)), 1\})$ für eine Funktion $f(n)$, die vielleicht ähnlich zu n^3 ist, sein könnte.

7.6 Ein Beispiel, bei dem $\mu > 1$ essenziell ist

Nachdem wir in den letzten Abschnitten die erwartete Laufzeit des bereits bekannten Algorithmus (μ+1)-EA auf bereits bekannten Beispielfunktionen untersucht haben, wenden wir uns in diesem Abschnitt einer strukturellen Frage zu. Bei allen drei Beispielfunktionen stellte sich nämlich heraus, dass $\mu = 1$ asymptotisch zu einer minimalen Laufzeit führt, also die Population des (μ+1)-EA keinen Nutzen zu zeigen scheint. Darüber hinaus erhalten wir in allen Fällen beliebig große polynomielle Laufzeiten, indem wir μ immer größer wählen.

Bislang ist es also immer noch nicht ersichtlich, ob die Population des bereits seit langem bekannten (μ+1)-EA in unserem Szenario der Optimierung pseudoboolescher Funktionen überhaupt einen Nutzen haben kann. Der Nachweis, dass dieser Nutzen tatsächlich besteht, soll das strukturelle Resultat sein, dem wir uns im Folgenden widmen werden. Im Großen und Ganzen greifen wir dabei auf dieselbe Idee wie in Abschnitt 6.2.1 zurück und konstruieren eine Beispielfunktion, bei der eine Erhöhung der Populationsgröße des (μ+1)-EA um einen polynomiell kleinen Faktor zu einer exponentiellen Abnahme der erwarteten Laufzeit führt. Als Populationsgröße, bei der eine exponentielle Laufzeit eintritt, betrachten wir lediglich $\mu = 1$ und untersuchen den (1+1)-EA, der sich auf der Beispielfunktion ebenso wie der (μ+1)-EA mit $\mu = 1$ verhält. Mit einer komplizierteren Funktion und komplizierteren Argumenten ist es allem Anschein nach gleichfalls möglich, eine exponentielle Laufzeit des (μ+1)-EA im Fall $\mu = O(1)$ nachzuweisen. Darauf verzichten wir aber, da wir bereits in Kapitel 6 einigen Aufwand getrieben haben, um ein Hierarchieresultat für die Populationsgröße zu zeigen.

Um die allgemeine Idee aus Abschnitt 6.2.1 wieder zu verwerten, benötigen wir zwei Funktionen f_1 und f_2, sodass der (1+1)-EA auf f_1 schneller als auf f_2 ist, wohingegen für den (μ+1)-EA das Umgekehrte gilt. Nun kommen uns die Analysen für die Beispielfunktionen OneMax und LeadingOnes aus den vorangehenden Abschnitten zugute. Indem wir LeadingOnes lediglich auf $\Theta(\sqrt{n})$ führenden Bits zugrunde legen, werden wir aus einer Modifikation von Theorem 7.3.2 erhalten, dass sich eine obere Schranke für die erwartete Laufzeit des (μ+1)-EA auf der beschriebenen LeadingOnes-Variante als $O(n^{3/2} + \mu n^{1/2} \log n)$ ergibt. Die erwartete Laufzeit auf OneMax hingegen ist mindestens $\Omega(\mu n)$. Für nicht zu kleine μ ist dies größer als die zuvor erwähnte obere Schranke. Andererseits beträgt die erwartete Laufzeit des (1+1)-EA auf OneMax $\Theta(n \log n)$ und ist geringer als die erwartete Laufzeit $\Theta(n^{3/2})$, die für die LeadingOnes-Variante vorherrscht.

Nachdem die „Komponenten" der zu konstruierenden Funktion nun bekannt sind, kommen wir bereits zu ihrer Definition. Wir unterteilen Suchpunkte $x \in \{0,1\}^n$ in das Präfix $(x_1, \ldots, x_m)$ der Länge m und das Suffix $(x_{m+1}, \ldots, x_n)$ der Länge ℓ. Es sei $\ell := \lceil n^{1/2}/55 \rceil$, d. h., es gilt $m = n - o(n)$. Ohne Beschränkung der Allgemeinheit sei m durch 3 teilbar. Wie in Abschnitt 6.2.1 definieren wir

$$\mathrm{PE}(x) \ := \ \sum_{i=1}^{m} x_i$$

sowie

$$\mathrm{FSE}(x) \ := \ \sum_{i=0}^{\ell-1} \prod_{j=0}^{i} x_{m+1+j}.$$

Schließlich sei $b := 2m/3 + \lceil n^{1/2}/(60 \log^2 n) \rceil$. Damit definieren wir

$$f(x) \ := \ \begin{cases} \mathrm{PE}(x) + n^2 \cdot \mathrm{FSE}(x), & \text{falls } \mathrm{PE}(x) \leq 2m/3, \\ n^2\ell - n \cdot |\,\mathrm{PE}(x) - b| + \mathrm{FSE}(x) - \ell - 1 & \text{sonst.} \end{cases}$$

Als Nächstes wollen wir die Struktur von f erläutern. Im ersten Fall ist der PE-Wert gering und der f-Wert wird stark vom FSE-Wert beeinflusst. Im Fall, dass $\mathrm{FSE}(x) = \ell$ und $\mathrm{PE}(x) = 2m/3$ gilt, erreichen wir den maximalen f-Wert von $n^2\ell + 2m/3$. Im Fall, dass $\mathrm{PE}(x) \leq 2m/3$ und $\mathrm{FSE}(x) < \ell$ gilt, beträgt der f-Wert jedoch höchstens $n^2(\ell - 1) + 2m/3$, was kleiner ist als $n^2\ell - nb - \ell - 1$, eine untere Schranke für den f-Wert im zweiten Fall, d. h. für $\mathrm{PE}(x) > 2m/3$. Falls $\mathrm{PE}(x) = b$ und $\mathrm{FSE}(x) = \ell$ gilt, haben wir einen lokal optimalen Suchpunkt vorliegen, dessen Hammingabstand zu jedem Suchpunkt mit höherem f-Wert mindestens $b - 2m/3 = \Omega(n^{1/2}/\log^2 n)$ beträgt. Alles in allem können wir festhalten, dass wir sehr viele Ideen der in Abschnitt 6.2.1 definierten Funktion wieder verwertet haben.

Theorem 7.6.1 *Mit einer Wahrscheinlichkeit von* $1 - 2^{-\Omega(n^{1/2}/\log n)}$ *beträgt auf* f *sowohl die Laufzeit des* $(\mu+1)$*-EA mit* $\mu = 1$ *als auch die des (1+1)-EA* $2^{\Omega(n^{1/2}/\log n)}$.

Beweis: Die Beweisidee folgt der allgemeinen Idee hinter der Konstruktion von f. Wir zeigen, dass der betrachtete EA mit hoher Wahrscheinlichkeit ein Individuum mit einem PE-Wert von b erzeugt. Anschließend muss der Hammingabstand von $b - 2m/3$ in einem einzigen Schritt überwunden werden, damit die zugehörige Mutation akzeptiert werden kann. Dies erfordert mit hoher Wahrscheinlichkeit $2^{\Omega(n^{1/2}/\log n)}$ Schritte. Im Folgenden werden wir den Beweis lediglich für den (1+1)-EA führen. Es lässt sich leicht überprüfen, dass alle Argumente auch für den $(\mu+1)$-EA mit $\mu = 1$ Gültigkeit besitzen, da es keine Rolle spielt, ob Individuen mit gleichem Funktionswert akzeptiert oder verworfen werden.

Mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\ell/2} = 1 - 2^{-\Omega(n^{1/2}/\log n)}$ hat der initiale Suchpunkt des (1+1)-EA einen FSE-Wert von höchstens $\ell/2$. Wir nehmen an, dass dies zutrifft. Im Folgenden wenden wir wieder die Beweismethode des typischen Laufs an und unterteilen den Lauf des (1+1)-EA in aufeinander folgende Phasen der Länge $\lceil 4\mathrm{e}(n-1)\rceil$. Dabei haben wir im Sinne, dass der (1+1)-EA in einer der Phasen den PE-Wert des aktuellen Suchpunktes auf b erhöht. Allerdings sehen wir uns wie im Beweis von Theorem 6.3.1 dem Problem gegenüber, dass der PE-Wert in Schritten, die den FSE-Wert erhöhen, sinken kann. Wir bezeichnen einen Schritt, der den FSE-Wert erhöht, als *schlecht* und eine Phase als *gut*, wenn sie keine schlechten Schritte enthält. Wir betrachten nun eine gute Phase der Länge $\lceil 4\mathrm{e}(n-1)\rceil$ und nehmen dabei an, dass der FSE-Wert noch kleiner als ℓ ist. In diesem Fall sind höchstens b Verringerungen des Abstands $|\,\mathrm{PE}(x)-b|$ hinreichend, um einen Suchpunkt mit PE-Wert b zu erzeugen. Die Wahrscheinlichkeit, den genannten Betrag zu verringern, ist mindestens $((m-b)/n)(1-1/n)^{n-1} \geq 1/(4\mathrm{e})$ (für genügend großes n), da es immer mindestens $m-b \geq n/4$ Bits gibt, deren Mutation (als 1-Bit-Mutation) den Betrag $|\,\mathrm{PE}(x)-b|$ verringert und somit den f-Wert erhöht. Folglich enthält die Phase gemäß der Chernoffungleichung mit einer Wahrscheinlichkeit von mindestens $1-2^{-\Omega(n)}$ mindestens b solcher Verringerungen.

Es verbleibt die Aufgabe, die Auswirkungen schlechter Schritte zu kontrollieren. Die Wahrscheinlichkeit, dass ein Schritt schlecht ist, ist durch $1/n$ nach oben beschränkt, da es notwendig ist, dass die linkeste Null im Suffix kippt. Damit ist die Wahrscheinlichkeit, dass eine Phase gut ist, durch $(1-1/n)^{\lceil 4\mathrm{e}(n-1)\rceil} \geq \mathrm{e}^{-4\mathrm{e}-1} = \Omega(1)$ nach unten beschränkt. Demzufolge treten vor einer guten Phase mit einer Wahrscheinlichkeit von $(1-\mathrm{e}^{-4\mathrm{e}-1})^{\ell/(32\mathrm{e})} = 2^{-\Omega(\ell)}$ höchstens $\ell/(32\mathrm{e})$ schlechte Phasen, d. h. Phasen mit schlechten Schritten, auf. Die Gesamtzahl der Schritte in dieser Zahl von Phasen ist durch $\ell n/8$ nach oben beschränkt. Da in $\ell n/8$ Schritten insgesamt mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ höchstens $\ell/6$ Bits kippen, nehmen wir nun an, dass wir höchstens $\ell/8$ schlechte Phasen und insgesamt höchstens $\ell/6$ schlechte Schritte vor dem Auftreten einer guten Phase beobachten. Durch diese Annahme wird die Wahrscheinlichkeit, dass eine Phase schlecht ist, nicht erhöht.

Anders als im Beweis von Lemma 6.3.3 kann es passieren, dass ein initialer FSE-Wert von $\ell/2$ und $\ell/6$ schlechte Schritte zu einem FSE-Wert von ℓ führen. Wir müssen hier die im Beweis von Theorem 7.5.1 erwähnten Trittbrettfahrer, die Einsen hinter der linkesten Null im Suffix, betrachten. Die Analyse von Droste, Jansen und Wegener (2002a) liefert nun, dass die Zahl der Trittbrettfahrer in $\ell/6$ Schritten mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ durch $\ell/3$ nach oben beschränkt ist. Insgesamt kommen wir also mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ bis zur guten Phase zu einem FSE-Wert, der noch kleiner als $\ell/2+\ell/3+\ell/6=\ell$ ist.

Wir wissen jetzt, dass der (1+1)-EA mit einer Wahrscheinlichkeit von $1-2^{-\Omega(\ell)}$ einen Suchpunkt mit einem PE-Wert von b erzeugt, bevor er das Optimum erreicht. Wenn der aktuelle Suchpunkt von dieser Gestalt ist, wird kein Suchpunkt mit einem PE-Wert von mehr als $2m/3$ (abgesehen von b) akzeptiert. Folglich müssen mindestens $2m/3-b \geq n^{1/2}/(60\log^2 n)$ Bits in einem Schritt des (1+1)-EA kippen. Die entsprechende Wahrscheinlichkeit ist mithilfe der stirlingschen Formel nach oben beschränkt durch

$$\binom{n}{2m/3-b}\left(\frac{1}{n}\right)^{2m/3-b} \leq \frac{1}{(n^{1/2}/(60\log^2 n))!} = 2^{-\Omega(n^{1/2}/\log n)}.$$

Also ist die Wahrscheinlichkeit, eine solche Mutation in $2^{\varepsilon n^{1/2}/\log n}$ Schritten zu vollführen, höchstens $2^{-\Omega(n^{1/2}/\log n)}$, wenn die Konstante ε klein genug ist. Das Theorem folgt, da die Summe aller Fehlerwahrscheinlichkeiten $2^{-\Omega(n^{1/2}/\log n)}$ ist. □

Wir halten fest, dass der (1+1)-EA und der (μ+1)-EA mit $\mu = 1$ sowie parallele Läufe dieser Suchheuristiken auf f äußerst ineffizient sind. Eine Erhöhung der Populationsgröße um einen weniger als linear großen Faktor lässt hingegen eine effiziente erwartete Laufzeit zu. Im Beweis des folgenden Theorems finden wiederum die Stammbäume Anwendung, weil implizit auch eine untere Schranke gezeigt wird. Der Grund dafür ist derselbe wie im Beweis von Theorem 6.4.1.

Theorem 7.6.2 *Sei $\mu \geq n/\ln(en)$ und $\mu = \text{poly}(n)$. Mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n^{1/2}/\log n)}$ ist die Laufzeit des (μ+1)-EA auf f durch $O(\mu n)$ beschränkt. Die erwartete Laufzeit ist ebenfalls $O(\mu n)$.*

Beweis: Für die erste Behauptung wenden wir zunächst die Idee aus dem Beweis von Theorem 7.3.2 an. Für den Augenblick nehmen wir an, dass in einem Zeitraum noch zu definierender Länge alle Individuen der aktuellen Population stets einen PE-Wert von höchstens $2m/3$ besitzen, also immer der erste Fall der Definition von f zutrifft. Als f-basierte Partition eignet sich

$$L_i := \left\{x \mid in^2 \leq f(x) < (i+1)n^2\right\}$$

für $0 \leq i \leq \ell - 1$,

$$L_\ell := \left\{x \mid n^2\ell \leq f(x) < n^2\ell + 2m/3\right\}$$

und $L_{\ell+1} := \{x \mid f(x) = n^2\ell + 2m/3\}$. Ein Individuum, das der i-ten Ranggruppe für $i \leq \ell - 1$ angehört, besitzt aufgrund der Struktur von f einen FSE-Wert von genau i. Also können wir mit den Abschätzungen aus dem Beweis von Theorem 7.3.2 zeigen, dass die erwartete Zahl von Schritten (nicht die erwartete Laufzeit), bis die Population der mindestens ℓ-ten Ranggruppe angehört, aufgrund der Wahl von μ durch $3e\mu\ell\ln(en)$ nach oben beschränkt ist. Hier verwenden wir den Parameter k aus der Formulierung von Lemma 7.3.1. Die Ranggruppe $L_{\ell+1}$ hat hier somit lediglich formalen Charakter.

Gemäß der Markoffungleichung ist die Zeit bis zu einem Erfolg, d. h., bis mindestens die ℓ-te Ranggruppe erreicht ist, mit einer Wahrscheinlichkeit von mindestens $1/2$ durch $t^* := \lfloor 6e\mu\ell\ln(en)\rfloor$ nach oben beschränkt. Wiederum können wir unabhängige Phasen wiederholen. Für jede beliebige Konstante $c > 0$ folgt, dass die Wahrscheinlichkeit für keinen Erfolg nach spätestens $\lfloor cn^{1/2}/\ln(en)\rfloor$ Phasen der Länge t^*, also in insgesamt höchstens $6ce\mu\ell n^{1/2}$ Schritten, durch $2^{-cn^{1/2}/\log n}$ nach oben beschränkt ist (für genügend großes n, da wir die Beziehung $\ln(en) < \log n$ ausnutzen, um Gaußklammern zu umgehen).

Nach einem Erfolg enthält die aktuelle Population aufgrund der Definition von f stets ein Individuum mit einem FSE-Wert von ℓ. Wir definieren $P(x) := \mathrm{PE}(x)$, falls $\mathrm{FSE}(x) = \ell$ und $\mathrm{PE}(x) \leq 2m/3$, und $P(x) := -1$ sonst. Es handelt sich bei P um eine bezüglich f monotone Funktion und die Zeit bis zu einem P-Potenzial von $2m/3$, d. h. zum Erreichen des Optimums, ist gemäß der Chernoffungleichung und den üblichen Abschätzungen für Mutationswahrscheinlichkeiten mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ durch $O(\mu n)$ beschränkt.

Wir müssen nun die Wahrscheinlichkeit abschätzen, dass es in $s := \lfloor 6ce\mu\ell n^{1/2} \rfloor$ Schritten nicht passiert, dass ein Individuum mit einem PE-Wert von mehr als $2m/3$ entsteht. Zu diesem Zweck verwenden wir im Wesentlichen wieder die Untere-Schranken-Technik aus Abschnitt 7.4.1 und gehen ähnlich wie im Beweis von Theorem 7.4.5 vor, sodass wir nun etwas verkürzt argumentieren. Gemäß Lemma 7.4.2 erreicht mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ kein Stammbaum in der Phase eine Tiefe von mindestens $18ce\ell n^{1/2}$. Also hat mit dieser Wahrscheinlichkeit auch kein Stammbaum eine Tiefe von mindestens $d := 42ce\ell n^{1/2}$. Zudem hat mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(n)}$ kein initiales Individuum einen PE-Wert von mindestens $7m/12$. Indem wir beispielsweise $c \leq 1/55$ und n groß genug wählen, erhalten wir $2 \cdot 42ce\ell n^{1/2} < m/12$. Gemäß Lemma 6.4.8 ist also die Wahrscheinlichkeit, im Stammbaum $T_s(x)$ für ein beliebiges initiales Individuum x einen Knoten x' mit $\mathrm{PE}(x') - \mathrm{PE}(x) \geq m/12$ auf einer Tiefe von höchstens d zu erhalten, höchstens

$$\mathrm{e}^{-14ce\ell n^{1/2} \cdot m/n} \cdot \mu \cdot \left(1 + \frac{2}{\mu}\right)^{6ce\mu\ell n^{1/2}} = 2^{-\Omega(n)}$$

Insgesamt ist die Wahrscheinlichkeit eines PE-Wertes von über $2m/3$ innerhalb der s Schritte durch $2^{-\Omega(n)}$ beschränkt. Die Wahrscheinlichkeit, dass ein Fehler eintritt, ist insgesamt durch $2^{-n^{1/2}(c-o(1))/\log n}$ beschränkt worden. Damit ist die erste Aussage des Theorems gezeigt.

Um die Aussage über die erwartete Laufzeit nachzuweisen, orientieren wir uns an der entsprechenden Passage im Beweis von Theorem 6.4.1. Hier müssen wir den Fall betrachten, dass Individuen mit einem PE-Wert von mehr als $2m/3$ entstehen. Oben haben wir gezeigt, dass die erwartete Zeit, bis der $(\mu{+}1)$-EA in eine solche Situation gerät oder das Optimum erreicht, durch $O(\mu n)$ beschränkt ist. Mit einem Potenzial, das den PE-Wert von Individuen mit einem PE-Wert über $2m/3$ beschreibt, folgt anhand von Lemma 5.2.1, dass anschließend im Erwartungswert $O(\mu n)$ Schritte ausreichen, um ein Individuum mit einem PE-Wert von b zu erzeugen. Wiederum mithilfe der üblichen Anwendung eines Potenzials sieht man leicht, dass anschließend $O(\mu\ell n)$ erwartete Schritte bis zur Erzeugung eines lokal optimalen Individuums reichen. Mit der Argumentation aus dem Beweis von Lemma 7.3.1 erhalten wir, dass nach weiteren $O(\mu \log \mu) = O(\mu \log n)$ (wegen der Voraussetzung $\mu = \mathrm{poly}(n)$) Schritten im Erwartungswert eine Population bestehend aus ausschließlich lokal optimalen Individuen vorliegt. Dabei nehmen wir überall pessimistisch an, dass das Optimum zwischenzeitlich nicht gefunden wird.

Der Hammingabstand eines lokal optimalen und eines global optimalen Individuums ist $h := b - 2m/3 \leq n^{1/2}/(60 \log^2 n) + 1$. Also genügt es, dass der (μ+1)-EA höchstens $n^{1/2}/(60 \log^2 n) + 1$ spezielle Bits eines lokal optimalen Individuums kippt, um das Optimum zu erreichen. In der vorliegenden Situation, dass die Population nur aus lokal optimalen Individuen besteht, ist die zugehörige Wahrscheinlichkeit durch

$$\left(\frac{1}{n}\right)^h \left(1 - \frac{1}{n}\right)^{n-h} = 2^{(-\log n)(n^{1/2})/(60 \log^2 n) - O(\log n)} = 2^{-n^{1/2}/(60 \log n) - O(\log n)}$$

nach unten beschränkt. Die Wartezeit auf einen solchen Schritt lässt sich daher durch $2^{n^{1/2}/(60 \log n) + O(\log n)}$ nach oben abschätzen. Im ersten Teil des Beweises haben wir gezeigt, dass bei unserer Wahl von c nur mit einer Wahrscheinlichkeit von höchstens $2^{-n^{1/2}(1-o(1))/(55 \log n)}$ die Laufzeit des (μ+1)-EA nicht durch $O(\mu n)$ beschränkt werden kann. Das Produkt dieser oberen Schranke und der oberen Schranke für die erwartete Laufzeit im Falle eines Fehlers ist $o(1)$. Mithilfe des Satzes von der totalen Wahrscheinlichkeit folgt insgesamt, dass die erwartete Laufzeit $O(\mu n)$ ist. $\square$

Die in den Theoremen 7.6.1 und 7.6.2 enthaltenen Resultate zum Nutzen der Population des (μ+1)-EA sind nicht so stark wie diejenigen aus Kapitel 6. Das war, wie oben erwähnt, auch gar nicht unser Ziel. Es ist aber bemerkenswert, dass wir die beiden erwähnten Theoreme ohne viele neue Ideen zeigen konnten. Denn anders als beim Beispiel-GA aus Definition 6.1.1 erscheint es beim (μ+1)-EA wegen der gleichverteilten Selektion zur Reproduktion nicht so leicht möglich, das typische Verhalten durch Konstruktion einer Beispielfunktion mit sehr großen Funktionswerten (vgl. die Funktion f aus Abschnitt 6.2.1) zu lenken.

7.7 Fazit und Ausblick

In diesem Kapitel haben wir uns mit den aus der Literatur bekannten Evolutionsstrategien beschäftigt. Unser Ziel war es dabei, die für den (1+1)-EA bekannten Ansätze zur Laufzeitanalyse auf pseudobooleschen Funktionen auf allgemeine (μ+λ)-Strategien zu übertragen. Als Ausgangspunkt untersuchten wir deshalb den (μ+1)-EA auf drei aus der Literatur bekannten Beispielfunktionen. Um obere Schranken für seine erwartete Laufzeit zu ermitteln, haben wir im Wesentlichen mit Potenzialfunktionen und Erweiterungen der Ranggruppenmethode gearbeitet. Für untere Schranken bedienten wir uns wieder der Stammbäume, die der Lauf des (μ+1)-EA beschreibt, und entwickelten daraus eine sehr allgemeine untere Schranke. Diese gilt für Funktionen mit nicht allzu vielen globalen Optima, manche Funktionen mit exponentiell vielen globalen Optima und insbesondere für die wohl untersuchte Klasse der unimodalen Funktionen. Bei einer der drei Beispielfunktionen ist es gelungen, die erwartete Laufzeit asymptotisch scharf festzulegen, und bei den übrigen Funktionen weichen die obere und die untere Schranke lediglich um einen Faktor $\Theta(\log \mu)$ voneinander ab. In allen drei Fällen, darunter bei einer Funktion, die ein Plateau konstanten Funktionswertes enthält, führt die Wahl $\mu = 1$ asymptotisch zur geringsten Laufzeit.

Da die Population des (μ+1)-EA auf den betrachteten Beispielfunktionen bewiesenermaßen asymptotisch keinen Nutzen bringt, sollte die Frage diskutiert werden, ob Beispiele existieren, bei denen eine moderate Erhöhung der Populationsgröße des (μ+1)-EA eine drastische Verringerung der erwarteten Laufzeit bedingen kann. Obschon ein entsprechendes Resultat bereits in Kapitel 6 für den EA aus Definition 6.1.1 gezeigt wurde, stellte dies in dem Sinne in Bezug auf den (μ+1)-EA eine neue Herausforderung dar, da jener EA nicht fitnessproportional zur Mutation wählt. Nichtsdestoweniger erhielten wir das gewünschte Beispiel mithilfe der allgemeinen, bereits in Abschnitt 6.2.1 vorgestellten Idee auf erstaunlich einfache Weise. Indem wir auf die Resultate für die zuvor untersuchten Beispielfunktionen zurückgriffen, wurde die Konstruktion einer Funktion f möglich, bei der der (1+1)-EA und dessen parallele Läufe mit exponentiell nahe an 1 liegender Wahrscheinlichkeit eine exponentielle Laufzeit benötigen, wohingegen der (μ+1)-EA bereits bei einer weniger als linear großen Population mit einer erwarteten polynomiellen Laufzeit auskommt. Wiederum kann die Beispielfunktion für weite Teile des Definitionsbereichs als Summe zweier auf disjunkten Teilen des Suchpunktes definierter Funktionen geschrieben werden, ist also teilweise separierbar.

Zur Herleitung unterer Schranken für die Laufzeit des (μ+1)-EA sind uns die in Kapitel 6 eingeführten Stammbäume wieder begegnet. Trotz des andersartigen Selektionsoperators trugen die in Abschnitt 6.4.3 entwickelten Methoden dazu bei, eine allgemeine obere Schranke für die Tiefe eines Stammbaums des (μ+1)-EA zu gewinnen. Obwohl in den zugehörigen Beweis scheinbar grobe Abschätzungen eingingen, war die Schranke gut genug, um die erwähnten asymptotisch scharfen bzw. fast scharfen Laufzeitschranken zu ermitteln. Es besteht die Hoffnung, dass sich die Ansätze zur Analyse der Stammbäume des (μ+1)-EA auch auf (μ+λ)-EAs mit $\lambda > 1$ verallgemeinern lassen. Zumindest für Werte mit $\lambda = O(1)$ scheint dies möglich zu sein.

Wir beschließen dieses Kapitel und damit auch diesen Teil der Dissertation mit einigen Gedanken, die sich auf den gesamten Teil II der Arbeit stützen. Während in Kapitel 6 die strukturelle Frage nach dem Nutzen von Populationen in evolutionären Algorithmen betrachtet wurde, untersuchten wir in Kapitel 7 mit dem (μ+1)-EA einen traditionellen populationsbasierten evolutionären Algorithmus. In beiden Fällen gelang es, die Laufzeit des jeweiligen evolutionären Algorithmus nach oben und nach unten zu beschränken. Insbesondere mit der Analyse der zufälligen Stammbäume haben wir mithin Methoden entwickelt, um das Verhalten der Population in manchen als „Ein-Kind-EA“ bezeichneten evolutionären Algorithmen zu verstehen und zu kontrollieren. Auch wenn diese Analysen jeweils auf den speziellen EA zugeschnitten worden waren, bergen sie viele Ansätze für eine mögliche Verallgemeinerung. Man sollte nicht unerwähnt lassen, dass die Schranken für die Tiefe von Stammbäumen in allen hier betrachteten Fällen völlig unabhängig vom zugrunde liegenden Suchraum $\{0,1\}^n$ sind. Somit könnte eine Aufgabe für die Zukunft darin bestehen, die zufälligen Stammbäume eines populationsbasierten evolutionären Algorithmus für den Suchraum $\mathbb{R}^n$ zu analysieren und damit nach Aussagen über die Laufzeit zu suchen.

Teil III

Zum Entwurf und zur Analyse spezialisierter Algorithmen

8 Die Historie eines kombinatorischen Optimierungsproblems

In den vergangenen Teilen haben wir reichlich Aufwand getrieben, um selbst einfachste randomisierte Suchheuristiken und evolutionäre Algorithmen, z. B. RLS und den (1+1)-EA, analysieren zu können. Die Schwierigkeiten, die sich bei der Analyse auftaten, standen unserer Ansicht nach vor allem damit im Zusammenhang, dass man beim Entwurf evolutionärer Algorithmen und verwandter Suchheuristiken weniger ihre Analysierbarkeit als beispielsweise Analogien zu Konzepten aus der Biologie im Blick hatte.

In diesem letzten Teil der Dissertation werden wir uns daher mit dem Entwurf von Algorithmen speziell unter dem Aspekt ihrer Analysierbarkeit auseinander setzen. Anhand von konkreten kombinatorischen Optimierungsproblemen werden wir illustrieren, wie eng Analyse und Entwurf miteinander verzahnt sein können, indem wir einfache Heuristiken und Approximationsalgorithmen aufarbeiten sowie komplexere exakte Algorithmen und schließlich wieder Approximationsalgorithmen für verallgemeinerte Problemstellungen entwerfen und untersuchen.

8.1 Problemstellung und Hintergrund

Der Ausgangspunkt aller im Folgenden betrachteten kombinatorischen Optimierungsprobleme liegt in der Architektur moderner Rechner. Bekanntlich steigt die Leistungsfähigkeit gegenwärtiger Rechnerarchitekturen zumindest zurzeit noch rapide und stetig weiter. Wenn wir konkrete Leistungsmerkmale wie Taktraten von Prozessoren, Speicherkapazitäten usw. betrachten, prognostiziert das so genannte mooresche Gesetz in etwa alle 18 Monate eine Verdoppelung der entsprechenden Kennzahl.

Leider scheint aber die Entwicklung der Leistung verschiedener Komponenten auseinander zu laufen. Ein Zugriff auf den Hauptspeicher einer modernen Rechnerarchitektur benötigt im Durchschnitt ein Vielfaches der Länge eines Taktzyklus des Prozessors. Noch ungünstiger nimmt sich das Verhältnis der Dauer eines Speicherzugriffs und der Dauer für einen Zugriff auf Hintergrundspeicher wie Festplatten aus.

Eine klassische Lösung, um die Latenzzeiten bei Speicherzugriffen abzumildern oder ihnen zu entgehen, besteht in der Einführung von schnellen, aber teuren und somit kleineren Zwischenspeichern *(caches)*. Solche Caches können sowohl dem Hauptspeicher als auch dem Hintergrundspeicher vorgeschaltet sein. Der Hauptspeicher kann wiederum als Zwischenspeicher für den Hintergrundspeicher fungieren, selbst wenn man den Hauptspeicher eher nicht als Cache bezeichnen wollte. Für unsere theoretischen Modellbildungen sind solche feinen Unterscheidungen ohne Bedeutung. Wir bezeichnen alle Formen von Zwischenspeichern als Caches und gehen davon aus, dass eine im Rechner benötigte Information, z. B. der Inhalt eines Speicherblocks, von einem langsameren, aber umfangreicheren Hintergrundspeicher in den Cache eingebracht werden muss, bevor diese verarbeitet werden kann.

Sobald ein Speicherblock angefragt wird, der sich nicht im Cache befindet, wird eine teure Leseoperation, die den Block in den Cache einbringt, fällig. Weitere Anfragen an einen Speicherblock sind billig, solange sich der Block im Cache befindet. Wenn irgendwann die Kapazität des Caches erschöpft ist, resultiert eine weitere Anfrage an einen nicht im Cache vorhandenen Block darin, dass ein anderer Block aus dem Cache zugunsten des neuen Blocks zu entfernen ist. Die Wahl des zu entfernenden Blockes stellt eine einfache Variante des allgemeineren kombinatorischen Optimierungsproblems, dem wir uns in diesem Teil der Dissertation verschreiben, dar. Informal gesagt besteht das Ziel darin, den Cache so zu verwalten, dass sich Anfragen möglichst häufig auf bereits im Cache vorhandene Blöcke beziehen.

Formal liegt uns eine Menge S von Speicherblöcken aus einem Hintergrundspeicher und ein anfangs leerer Cache, der mit k Zellen je einen Speicherblock fassen kann, vor. Das System muss eine in zeitlicher Reihenfolge vorgegebene Anfragesequenz $\sigma = \sigma(1), \ldots, \sigma(n)$ bedienen, in der jede Anfrage $\sigma(i)$, $1 \leq i \leq n$, ein Element aus S referenziert. Befindet sich Block $\sigma(i)$ zum Zeitpunkt seiner Anfrage nicht im Cache, verursacht diese Anfrage eine Kosteneinheit, indem sie den Block in den Cache einbringt und dafür bei aktuell k Blöcken im Cache einen anderen Block aus dem Cache entfernt. Gesucht ist eine kostenminimale Wahl der zu den Zeitpunkten zu entfernenden Blöcke.

Gebräuchliche Heuristiken für dieses Modell, so genannte Cacheersetzungsstrategien, finden sich in wohl jedem Lehrbuch über Rechnerarchitektur. Einfache Greedy-Algorithmen, wie beispielsweise FIFO *(first in, first out)*, waren lange vor der Beschreibung eines optimalen Algorithmus (oft LFD oder MIN genannt, siehe Belady, 1966) bekannt. Mittlerweile ist klar, dass FIFO einen Approximationsalgorithmus mit einer nicht konstanten Güte von lediglich k darstellt. Trotzdem ist die FIFO-Heuristik im so genannten Online-Szenario (siehe z. B. Borodin und El-Yaniv, 1998) von Interesse, da zu jedem Zeitpunkt nur das nächste Element der Anfragesequenz bekannt zu sein braucht. Insbesondere im Online-Szenario existiert eine Vielfalt von Verallgemeinerungen und Spezialisierungen des von uns beschriebenen Caching-Modells.

Ein weiteres Konzept, das in Rechnersystemen leistungssteigernd wirken soll und in zunächst nur losem Zusammenhang mit Caching steht, wird gemeinhin als *Prefetching* bezeichnet. Prefetching liegt dann vor, wenn ein Element aus der Menge S, das sich nicht im Cache befindet, bereits vor dem Zeitpunkt seiner nächsten Referenzierung vorsorglich in den Cache eingebracht wird. Damit modellieren wir Latenzzeiten zum Einholen von Blöcken, die insbesondere bei langsamen Zugriffen auf Festplatten entstehen. Die Bearbeitung eines Elements $\sigma(i)$ der Anfragesequenz soll eine Zeiteinheit beanspruchen, während das Einbringen F Zeiteinheiten nach sich zieht. Wird das Einbringen erst zum Zeitpunkt der Anfrage gestartet, muss das gesamte System F Zeiteinheiten warten. Startet man das Einbringen eines Blocks jedoch $i \leq F$ Zeiteinheiten vor seiner Anfrage, entsteht eine Wartezeit von lediglich $F - i$.

Vor einer theoretischen Fassung des Konzeptes des Prefetchings existierte bereits eine Reihe experimenteller Arbeiten, die Systeme, welche Speicherblöcke bereits vor ihrer unmittelbaren Verwendung einbringen, vorschlagen und evaluieren, siehe z. B. Griffioen und Appleton (1994) sowie Patterson, Gibson, Ginting, Stodolsky und Zelenka (1995). Diese Ansätze zum Prefetching verwenden allerdings jeweils eine feste Cachingstrategie, die isoliert von der Strategie zum Prefetching funktioniert. Somit wenden die Systeme zwar auf der einen Seite heuristisches Wissen darüber an, welche Speicherblöcke demnächst benötigt und deshalb vorsorglich eingebracht werden, bauen aber die Strategien zum Entfernen von Blöcken aus dem Cache auf unter Umständen andersartigen und konfliktären Vorstellungen auf. Ein weiteres, offensichtliches Problem liegt darin, dass ein vorsorglich eingebrachter Block bereits eine Zelle des Caches blockiert. Unser Wunsch ist es daher, Prefetching und Caching integrativ zu betrachten. Im Rahmen dieser Arbeit bleiben wir dabei in der Welt eines üblichen kombinatorischen Optimierungsproblems, dessen Instanz wir als vollständig bekannt annehmen. Eine Erweiterung der Ansätze auf das oben erwähnte Online-Szenario erscheint erst dann sinnvoll, wenn die klassische Formulierung untersucht und gut verstanden wurde.

8.2 Das kombinatorische Optimierungsproblem

Cao, Felten, Karlin und Li (1995) haben als Erste ein Modell, das Prefetching und Caching integriert, vorgestellt. Wiederum liegt eine Menge S von Speicherblöcken, ein anfangs leerer Cache der Größe k sowie eine Anfragesequenz $\sigma = \sigma(1), \ldots, \sigma(n)$ an Blöcke aus S vor. Die Bedienung einer Anfrage $\sigma(i)$ benötigt eine Zeiteinheit und kann nur stattfinden, wenn der zugehörige Block im Cache ist. Es fallen jeweils $F \in \mathbb{N}$ Zeiteinheiten an, um einen Block in den Cache einzubringen. Wenn das Einbringen eines Blocks erst unmittelbar vor seiner Anfrage begonnen wird, erfährt das System eine Wartezeit von F Zeiteinheiten, während lediglich $\max\{F - i, 0\}$ Einheiten Wartezeit anfallen, wenn das Einbringen des Blockes bereits i Zeiteinheiten vor seiner Anfrage gestartet wird. Selbst für $i = 0$ sprechen wir von einer Prefetchoperation. Sobald eine Prefetchoperation gestartet wurde, ist dafür eine Zelle des Caches reserviert, d. h., es darf daraus nicht mehr gelesen werden, und es muss deshalb ggf. ein anderer Block aus dem Cache entfernt werden. Ein Prefetching-und-Caching-Plan entscheidet,

- wann eine Prefetchoperation gestartet wird,
- welcher Block eingebracht wird,
- und welcher Block aus dem Cache für den einzubringenden zu entfernen ist.

Unser Ziel ist es, einen Plan zu finden, der die *Summe der Wartezeiten* nach Befriedigung der Anfragesequenz minimiert. Gelegentlich betrachtet man auch die *abgelaufene Zeit*, die sich zusammensetzt aus der Summe der Wartezeiten und der Zeit n, die für die Bedienung der Anfragen anfällt. Wir werden aber im Folgenden stets die Summe der Wartezeiten minimieren. Im Gegensatz zum reinen Caching kann es beim so beschriebenen integrierten Prefetching und Caching (IPC) also der Fall sein, dass das Einbringen von Elementen in den Cache in fast allen Fällen kostenlos bleibt.

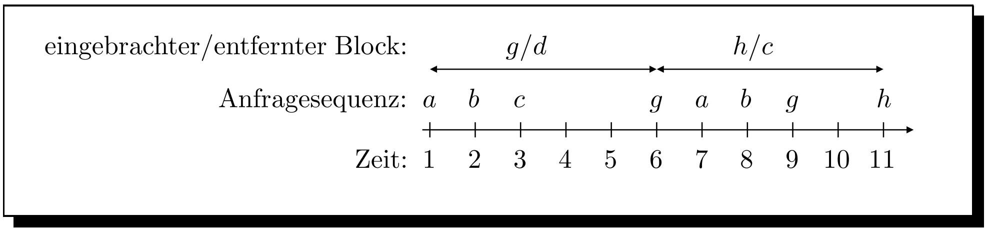

Abbildung 8.1: Prefetching und Caching: Beispiel für eine Platte

In Anlehnung an die Arbeit von Albers, Garg und Leonardi (2000) zeigt Abbildung 8.1 ein Beispiel für den Fall $F = 5$, $k = 4$, die Anfragesequenz a, b, c, g, a, b, g, h und einen anfänglichen Cacheinhalt von $\{a, b, c, d\}$. (In den Beispielen weichen wir der Einfachheit halber von der Annahme eines anfänglich leeren Caches ab.) Die minimale Wartezeit, die wir bei dieser Sequenz in Kauf nehmen müssen, beträgt 3 und die abgelaufene Zeit damit 11. Mit der Bedienung der ersten Anfrage an a entfernen wir d aus dem Cache und starten eine Prefetchoperation für g. Da für Letzteres 5 Zeiteinheiten benötigt werden, entsteht zwischen der Bedienung von c und der von g eine Wartezeit von zwei Zeiteinheiten. Mit der Bedienung von g beginnen wir eine Prefetchoperation für h, entfernen c und müssen nach der Bedienung des vorletzten Elements der Anfragesequenz eine Zeiteinheit warten, ehe sich h im Cache befindet.

Da Prefetching insbesondere das Lesen von Blöcken langsamer Festplattenspeicher modelliert, schlugen Kimbrel und Karlin (2000) erstmals im Jahre 1996 eine Verallgemeinerung integrierten Prefetchings und Cachings auf Systeme mit mehreren Plattenspeichern vor. Sei $D \in \mathbb{N}$. Mit S_d, $1 \leq d \leq D$, bezeichnen wir die Menge der Speicherblöcke, die auf der d-ten Platte liegen. Die Menge S aller Blöcke zerfällt jetzt disjunkt in die Mengen S_d. Die Erweiterung besteht nun darin, dass Blöcke, die auf verschiedenen Platten liegen, gleichzeitig eingebracht werden dürfen, während für jede Platte nur höchstens eine Prefetchoperation aktiv sein darf. Deshalb spricht man hier auch vom *parallelen* Prefetching und Caching. Jede gestartete Prefetchoperation benötigt wie oben F Zeiteinheiten und benutzt eine Zelle des Caches, sodass ggf. wieder ein Block dafür zu entfernen ist. Wir fordern nicht, dass der zu entfernende Block auf derselben Platte liegen muss wie derjenige, für den wir die Prefetchoperation starten, und modellieren somit ein Nur-Lesen-System, in dem keine modifizierten Speicherblöcke zurückgeschrieben zu werden brauchen. Das Ziel besteht wie oben in der Minimierung der Summe der Wartezeiten zur Bedienung einer Anfragesequenz. Infolge der Möglichkeit, Blöcke verschiedener Platten gleichzeitig einzubringen, unterscheidet sich der Fall $D > 1$ gegenüber dem oben erwähnten Spezialfall $D = 1$ insbesondere darin, dass unser Problem jetzt einen Aspekt der Lastverteilung enthält. Unter dem Problem IPC verstehen wir im Folgenden die erarbeitete Verallgemeinerung mit variablem D. Wir nehmen stets $1 \leq F \leq n$ an, da das Problem sonst trivial wird.

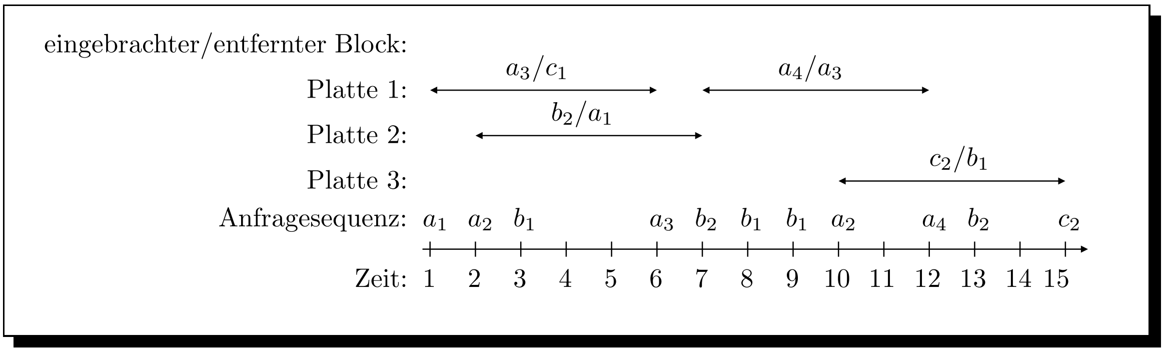

Abbildung 8.2: Prefetching und Caching: Beispiel für drei Platten

Abbildung 8.2 zeigt ein ebenfalls der oben erwähnten Arbeit entnommenes Beispiel für den Fall $D = 3$ mit $S_1 = \{a_1, a_2, a_3, a_4\}$, $S_2 = \{b_1, b_2\}$ und $S_3 = \{c_1, c_2\}$. Wiederum nehmen wir $F = 5$ und $k = 4$ an. Anfangs seien die Blöcke a_1, a_2, b_1 und c_1 im Cache. Um die Anfragesequenz $a_1, a_2, b_1, a_3, b_2, b_1, b_1, a_2, a_4, b_2, c_2$ zu bedienen, genügen insgesamt vier Einheiten Wartezeit. Während der Wartezeiten zwischen den Zeitpunkten 4 und 6 sowie zwischen 11 und 12 laufen zwei Prefetchoperationen zugleich.

8.3 Stand der Forschung

Anhand ihrer theoretischen Fassung von IPC (für den Fall $D = 1$) arbeiteten Cao, Felten, Karlin und Li (1995) am Entwurf effizienter Algorithmen zur Lösung dieses Optimierungsproblems. Zunächst gelang ihnen eine Charakterisierung von vier Regeln, die jeder optimale Algorithmus für IPC mit $D = 1$ o. B. d. A. einhält.

Regel 1: optimales Prefetching Jede Prefetchoperation bringt den nächsten Block aus der Anfragesequenz, der nicht im Cache vorhanden ist, ein.

Regel 2: optimale Ersetzung Jede Prefetchoperation entfernt denjenigen Block aus dem Cache, dessen nächste Anfrage am weitesten in der Zukunft liegt.

Regel 3: keinen Schaden anrichten Block a darf nicht zum Einbringen von b aus dem Cache entfernt werden, wenn die nächste Anfrage an a vor der an b liegt.

Regel 4: erste Gelegenheit nutzen Wenn Block a zum Einbringen von b aus dem Cache entfernt wird, war dies zum vorhergehenden Zeitpunkt nicht möglich.

Die Korrektheitsbeweise dieser Regeln geben wir nicht wieder. Die ersten beiden Regeln sind insoweit interessant, als sie zu jedem Zeitpunkt festlegen, welcher Block ggf. einzubringen und welcher dafür zu entfernen ist. Allerdings geben sie nicht an, ob eine Operation gestartet werden soll.

Es blieb für mehrere Jahre ungeklärt, ob das Problem IPC mit $D = 1$ in Polynomialzeit gelöst werden kann. Aufbauend auf den oben erwähnten vier Regeln gelang es Cao, Felten, Karlin und Li (1995) aber, zwei einfache Polynomialzeitalgorithmen vorzustellen und nachzuweisen, dass sie eine nicht triviale Approximationsgüte einhalten. Sie nennen diese Algorithmen *conservative* und *aggressive.* Grob gesagt baut *conservative* auf einer Strategie für das oben erwähnte „reine" Caching auf. Anhand des MIN-Algorithmus von Belady (1966) legt der Algorithmus zunächst fest, welche Blöcke in welcher zeitlicher Reihenfolge aus dem Cache zu entfernen sind. Anschließend bestimmt er anhand dieser Vorgaben die Prefetchoperationen, indem er für den nächsten zu entfernenden Block e den frühestmöglichen Zeitpunkt bestimmt, der in Einklang mit den vier Regeln ist und vorschreibt, e zu entfernen. Aufgrund der ersten Regel ist der einzubringende Block damit festgelegt. Der Algorithmus heißt konservativ, weil er die geringstmögliche Zahl an Prefetchoperationen durchführt.

Der als *aggressive* bezeichnete Algorithmus geht nach einem Greedy-Ansatz vor. Er startet für das nächste nicht im Cache befindliche Element zum frühestmöglichen Zeitpunkt eine Prefetchoperation und entfernt dafür gemäß der zweiten Regel das Element, dessen Anfrage am weitesten in der Zukunft liegt, aus dem Cache. Der frühestmögliche Zeitpunkt ergibt sich aus der dritten Regel. Der Algorithmus wartet also zumindest mit dem Start einer Prefetchoperation ab, bis es einen Block im Cache gibt, dessen nächste Anfrage hinter der des Blockes liegt, der eingebracht werden muss.

Die oben genannten Autoren der beiden Algorithmen konnten nachweisen, dass *aggressive* einen Approximationsalgorithmus mit einer Güte von $\min\{2, 1 + F/k\}$ und *conservative* einen Approximationsalgorithmus mit Güte 2 darstellt. Interessanterweise gelang Albers und Büttner (2003a) erst einige Jahre später der Nachweis einer verbesserten oberen und fast scharfen unteren Schranke für die Güte von *aggressive.* In den nachfolgenden Jahren konzentrierte sich die Forschung stärker auf die Entwicklung verbesserter Approximationsalgorithmen und möglicherweise exakter Algorithmen. Es folgten auch umfangreichere Studien der praktischen Effizienz der erwähnten einfachen Algorithmen (Cao, Felten, Karlin und Li, 1996).

Mit einer 1998 vorgestellten Arbeit gelang Albers, Garg und Leonardi (2000) endlich der Nachweis, dass das Problem IPC im Falle $D = 1$ effizient gelöst werden kann. Die Autoren formulieren IPC als binäres Optimierungsproblem (auch unter dem Namen *0/1-integer programming* bekannt) und führen eine lineare Relaxation durch. Eine optimale Lösung des auf diese Weise erhaltenen linearen Optimierungsproblems (synonym für „lineares Programm") lässt sich in Polynomialzeit bestimmen. Die entscheidende Leistung besteht dann darin zu erkennen, dass die somit gewonnene gebrochene Lösung für das Optimierungsproblem eine Konvexkombination von polynomiell vielen zulässigen und ganzzahligen Lösungen darstellt. Mithilfe eines Verfahrens, das eine ganzzahlige und optimale Lösung aus der gebrochenen Lösung extrahiert, ergibt sich schließlich der gesuchte Polynomialzeitalgorithmus. Unschön daran ist lediglich das Erfordernis, einen Polynomialzeitalgorithmus zur Lösung eines linearen Optimierungsproblems einsetzen zu müssen. In Abschnitt 9.1 werden wir Nachteile dieses Aspekts diskutieren und Alternativen vorschlagen.

Mit diesem kurzen Ausflug in die Historie der Forschung über das Problem IPC mit $D = 1$ haben wir exemplarisch gesehen, wie der Entwurf effizienter Algorithmen von einfachen Greedy-Heuristiken, die sogar als Approximationsalgorithmen nicht trivialer Güte fungieren, bis hin zu komplexen, aber exakten Polynomialzeitalgorithmen führen kann. Im Fall $D > 1$, also beim so genannten parallelen Prefetching und Caching, war eine ähnlich spannende Entwicklung zu verzeichnen. Um unsere eigenen Beiträge in diesen Kontext einzuordnen, werden wir nachfolgend auch jene Historie aufarbeiten.

Kimbrel und Karlin (2000) stellen in ihrer Arbeit aus dem Jahre 1996 nicht nur als Erste die Erweiterung auf den Fall $D > 1$ vor, sondern analysieren in diesem Szenario außerdem die oben erwähnten einfachen Algorithmen *conservative* und *aggressive*. Als Ergebnisse erhalten sie im Wesentlichen die Aussage, dass die für den Fall $D = 1$ bekannten Approximationsgüten mit einem Faktor von ungefähr D multipliziert werden müssen. Dies begründet sich teilweise daraus, dass drei der obigen vier Regeln im Fall $D > 1$ nicht mehr gelten müssen. Weiterhin schlagen Kimbrel und Karlin (2000) einen verbesserten Approximationsalgorithmus vor, der auf der rückwärts gelesenen Anfragesequenz arbeitet und auf eine Approximationsgüte von $1 + DF/k$ kommt. Albers, Garg und Leonardi (2000) wiederum geben erneut auf der Basis eines linearen Optimierungsproblems einen Polynomialzeitalgorithmus an, der auf jeden Fall eine Güte von D einhält, sofern ein zusätzlicher Speicherplatz von $D - 1$ Zellen im Cache reserviert ist. Albers und Büttner (2003a) schließlich stellen einen exakten Polynomialzeitalgorithmus für den Fall $D > 1$ vor, betrachten aber wiederum eine Variante des Problems, in der sogar bis zu $3(D - 1)$ Extrazellen im Cache zur Verfügung gestellt sind.

Die Komplexität des ursprünglichen Problems IPC blieb für den Fall $D > 1$ bis zum Jahr 2004 ungeklärt. Kimbrel (1997) konnte zunächst lediglich einen Algorithmus angeben, der in Polynomialzeit die eingeschränkte Entscheidungsfrage, ob eine Anfragesequenz (bei anfangs gefülltem Cache) ohne Wartezeit bedient werden kann, beantwortet. Ambühl und Weber (2004) schließlich weisen nach, dass IPC im Fall $D > 1$ ein NP-hartes Optimierungsproblem repräsentiert und unter der P$\neq$NP-Hypothese in Polynomialzeit nicht besser als auf einen Faktor von $25/24 - \varepsilon$ approximiert werden kann, es also dann kein polynomielles Approximationsschema geben kann. Passend zu diesem Resultat werden wir in Abschnitt 9.2 weitere Ansätze für Approximationsalgorithmen konstanter Güte im Fall $D > 1$ erarbeiten. Inzwischen konzentriert sich die Forschung auf Erweiterungen des parallelen Prefetchings und Cachings und die Suche nach passenden Approximationsalgorithmen (Albers und Büttner, 2003b).

9 Entwurf und Analyse problemspezifischer Algorithmen

In diesem Kapitel werden wir alternative exakte Algorithmen und Approximationsalgorithmen für das in Abschnitt 8.2 vorgestellte Problem IPC erarbeiten. In Abschnitt 9.1 werden wir uns einem exakten Algorithmus für den Fall $D = 1$ widmen, wohingegen wir in Abschnitt 9.2 auf das parallele Prefetching und Caching, d. h. den Fall $D > 1$, zu sprechen kommen werden.

9.1 Ein kombinatorischer Ansatz

Die in Abschnitt 8.3 erwähnte Arbeit von Albers, Garg und Leonardi (2000) enthält einen Polynomialzeitalgorithmus für IPC im Fall $D = 1$, der auf einer Formulierung als lineares Optimierungsproblem basiert. Derartige Algorithmen gelten zuweilen aus verschiedenen Gründen als etwas unbefriedigend. Zum einen setzen sie eine Implementierung voraus, die in Polynomialzeit ein lineares Optimierungsproblem löst. Der einfach zu implementierende und als praktisch effizient geltende Ansatz des Simplex-Algorithmus scheidet hier aufgrund seiner exponentiellen Worst-Case-Rechenzeit aus. Um das lineare Optimierungsproblem in Polynomialzeit zu lösen, ist stattdessen ein komplexerer Algorithmus aus dem Bereich der linearen Programmierung erforderlich, beispielsweise Ellipsoidmethoden (Schrijver, 1986). Abgesehen von der komplexen Beschreibung dieser Algorithmen fällt es außerdem als Nachteil auf, dass eine gute obere Schranke für ihre Laufzeit schwer zu ermitteln zu sein scheint. Wir wissen zwar eine polynomielle Worst-Case-Laufzeit, können aber nur schwer erkennen, ob diese für die speziellen Instanzen von linearen Optimierungsproblemen, die wir aus einer Formulierung von IPC erhalten, ebenfalls Worst-Case-Laufzeiten darstellen. Weiterhin mögen wir den Rückgriff auf lineare Optimierungsprobleme nicht, weil deren Lösung wenig intuitives Verständnis darüber ermöglicht, wie der Algorithmus die spezielle Problemstruktur von IPC ausnutzt, um zu einer Lösung zu gelangen.

Eine Herausforderung in Bezug auf das Problem IPC bestand also darin, für den Fall $D = 1$ einen Polynomialzeitalgorithmus zu finden, der ohne die Lösung eines linearen Optimierungsproblems auskommt. Man spricht in solchen Zusammenhängen gerne von so genannten *kombinatorischen Algorithmen*, ohne dass es möglich wäre, diesen Begriff exakt zu definieren. Unter kombinatorischen Algorithmen verstehen wir hauptsächlich Algorithmen, die bereits in ihrer abstrakten Beschreibung lediglich auf diskreten Strukturen, z. B. Graphen, ganzzahligen Objekten usw., arbeiten. Beschreibungen von linearen Optimierungsproblemen und zugehöriger Algorithmen wie Ellipsoidmethoden arbeiten in ihrer allgemeinen Form nicht mit diskreten Strukturen, sondern beispielsweise mit Hyperebenen im $\mathbb{R}^n$. Wir halten fest, dass wir Algorithmen, die auf Ansätze zur Lösung linearer Optimierungsprobleme und allgemeinerer Verfahren der mathematischen Programmierung ausweichen und damit die Struktur des kombinatorischen Optimierungsproblems fallen lassen, auf keinen Fall kombinatorisch nennen.

Wie könnte das Problem IPC geeignet formuliert werden, um es mit einem Algorithmus, den wir kombinatorisch nennen können, anzugehen? Ein erster Gedanke könnte sein, den Ab- und Zugang von Blöcken im Cache als einen Fluss in einem Netzwerk aufzufassen. Die Forderung, die Summe der Wartezeiten bei einer Bedienung der Sequenz zu modellieren, ließe sich vielleicht erreichen, indem wir in einem speziellen Netzwerk einen Fluss mit minimalen Kosten bei einer Mindestanforderung an den Wert des Flusses berechneten (Minimalkostenflussproblem, *min-cost flow*). Für solche Problemstellungen existieren kombinatorische Algorithmen (siehe – auch zu weiteren Flussproblemen – Ahuja, Magnanti und Orlin, 1993).

Leider ist es bislang nicht gelungen, IPC mithilfe eines solchen einfachen Flussproblems darzustellen. Im Folgenden wird dies aber für eine Erweiterung des herkömmlichen Flussmodells gelingen, das so genannte *Mehrgüterflussproblem.* Wir definieren eine spezielle, für unsere Zwecke geeignete Variante, indem wir die herkömmlichen Begriffe eines (zulässigen) Flusses in einem kostenbehafteten, kapazitätsbeschränkten Netzwerk sowie seines Wertes und seiner Kosten als bekannt voraussetzen.

Definition 9.1.1 (Mehrgüterfluss) *Gegeben sei ein gerichteter Graph $G = (V, E)$ mit Kapazitätsfunktion $u\colon E \to \mathbb{R}^+$, Kostenfunktion $c\colon E \to \mathbb{R}^+$, einer Menge von Gütern $M = \{1, \ldots, g\}$ und ausgezeichneten Knotenpaaren $(s_i, t_i) \in V^2$ sowie Anforderungen $d_i \in \mathbb{R}^+$ für $i \in M$. Ein zulässiger Fluss für ein Gut i ist eine Abbildung $f_i\colon E \to \mathbb{R}_0^+$, die einen zulässigen Fluss bez. G und u mit Quelle s_i und Senke t_i darstellt. Ein* Mehrgüterfluss *$f = (f_1, \ldots, f_g)$ heißt zulässig, wenn alle f_i zulässig sind, der Wert jedes Flusses f_i mindestens d_i beträgt und für alle $e \in E$ die Beschränkung $f_1(e) + \cdots + f_g(e) \leq u(e)$ eingehalten wird. Die Kosten von f ergeben sich aus der Summe der Kosten der f_i.*

Ziel ist dann natürlich die Minimierung der Kosten von f. Dieses Mehrgüterflussproblem heißt in der Literatur *min-cost multicommodity flow problem.* Der Spezialfall $g = 1$ entspricht dem oben erwähnten Minimalkostenflussproblem für einen Wert von mindestens d_1.

Mehrgüterflussprobleme haben vielfältige Anwendungen, z. B. zur Modellierung von Kommunikationsnetzwerken mit ausgewählten Sender-Empfänger-Paaren. Daher wurde unser Mehrgüterflussproblem bereits vor einigen Jahrzehnten erstmals betrachtet. Inzwischen existiert eine ganze Reihe exakter Polynomialzeitalgorithmen, in die das Buch von Ahuja, Magnanti und Orlin (1993) einen Einblick bietet. Aus unserer Sicht ärgerlich ist aber der Umstand, dass bislang kein effizienter kombinatorischer Algorithmus zur Lösung eines Mehrgüterflussproblems (mit $g > 1$) bekannt ist. Selbst wenn es gelingt, IPC auf solche Flussprobleme zu reduzieren, scheinen wir auf der Suche nach kombinatorischen Algorithmen abermals gescheitert.

Zu unserem Glück existieren kombinatorische Approximationsalgorithmen für das oben beschriebene Mehrgüterflussproblem. Mit deren Hilfe könnte es gelingen, einen neuen, kombinatorischen Approximationsalgorithmus für IPC zu entwerfen. Erstaunlicherweise ist es aber sogar möglich, einen Approximationsalgorithmus für das Mehrgüterflussproblem so zu parametrisieren, dass wir zu einem exakten und kombinatorischen Polynomialzeitalgorithmus für IPC gelangen. Wir erhalten in diesem Zusammenhang eine interessante Verknüpfung zwischen dem Entwurf von Approximationsalgorithmen und exakten Algorithmen.

Es ist nun an der Zeit, etwas näher auf das Verhalten gängiger Approximationsalgorithmen für das Mehrgüterflussproblem gemäß Definition 9.1.1 einzugehen. Viele der Algorithmen weichen nicht nur in den Kosten der berechneten Lösungen vom optimalen Wert ab, sondern verletzen möglicherweise außerdem die Mindestanforderungen d_i für die einzelnen Güter. Der im Folgenden angewandte Approximationsalgorithmus von Kamath, Palmon und Plotkin (1995) ist mit zwei Parametern $\delta, \varepsilon > 0$ ausgestattet. Er berechnet in einer Zeit von $O^*(\varepsilon^{-3}\delta^{-3}g|E||V|^2)$ einen Mehrgüterfluss, der für jedes Gut i mindestens einen Fluss von $(1-\varepsilon)d_i$ vorsieht und Kosten von höchstens des $(1+\delta)$-Fachen des Optimums besitzt. Die Laufzeitschranke ist nach wie vor die asymptotisch beste dem Verfasser dieser Dissertation bekannte Schranke für Approximationsalgorithmen, die das Gleiche wie der von Kamath, Palmon und Plotkin (1995) leisten.

9.1.1 Ein Flussnetzwerk als Lösungsmodell

In diesem Unterabschnitt werden wir aus einer beliebigen Instanz für IPC mit $D = 1$ ein Flussnetzwerk konstruieren. Im nächsten Unterabschnitt werden wir dann illustrieren, wie man aus einer approximativen Lösung des Mehrgüterflussproblems auf diesem Netzwerk einen optimalen Prefetching-und-Caching-Plan ablesen kann.

Sei eine Instanz für IPC mit $D = 1$ gegeben. Anhand der Anfragesequenz $\sigma = \sigma(1), \ldots, \sigma(n)$ konstruieren wir ein Netzwerk $G = (V, E)$ mit Kosten, Kapazitäten und $n+1$ Gütern. Zu jeder Anfrage $\sigma(i)$ gibt es ein Gut i, $1 \le i \le n$, mit einem eigenen Quelle-Senke-Paar (s_i, t_i) und einer Anforderung $d_i = 1$. Sei $a_i \in S$ der von $\sigma(i)$ angefragte Block. Für jede Anfrage $\sigma(i)$ erzeugen wir zwei Knoten x_i und x'_i. Wir verbinden diese Knoten in linearer Reihenfolge, d. h., wir führen Kanten (x_i, x'_i), $1 \le i \le n$, und Kanten (x'_i, x_{i+1}), $1 \le i \le n-1$, mit einer Kapazität von jeweils k und Kosten 0 ein. Intuitiv soll diese Folge von Knoten den Cache darstellen. Wenn Gut i über die Kante (x_j, x'_j) fließt, assoziieren wir damit, dass sich der Block a_i im Cache befindet, während Anfrage $\sigma(j)$ bedient wird. Um sicherzustellen, dass sich a_i im Cache befindet, während $\sigma(i)$ bedient wird, fügen wir die Kante (x'_i, t_i) mit Kapazität 1 und Kosten 0

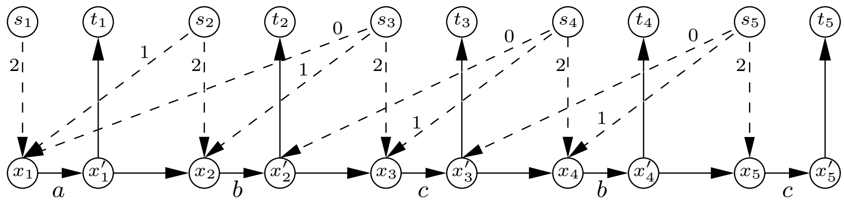

Abbildung 9.1: Skizze des Flussnetzwerks für eine beispielhafte Eingabe

hinzu. Da es keine weiteren Kanten mit Endpunkt t_i oder x_i' gibt, muss Gut i über die Kante (x_i, x_i') fließen.

Es bezeichne p_i den Zeitpunkt der letzten Anfrage an a_i, d. h., p_i ist das größte $j < i$, sodass $\sigma(j)$ den Block a_i referenziert. Wenn a_i zum ersten Mal in σ auftritt, setzen wir $p_i := 0$. Um die Anfrage $\sigma(i)$ zu bedienen, darf der Block a_i entweder von der Anfrage $\sigma(p_i)$ bis zur Anfrage $\sigma(i)$ im Cache bleiben (sofern $p_i > 0$) oder er kann zu einem Zeitpunkt vor $\sigma(i)$ erneut eingebracht werden. Um Ersteres zu modellieren, fügen wir, sofern $p_i > 0$, die Kante (s_i, x'_{p_i}) mit Kapazität 1 und Kosten 0 hinzu. Um Letzteres zu modellieren, erzeugen wir im Wesentlichen Kanten (s_i, x_j) für $j = p_i + 1, \dots, i$, um zu symbolisieren, dass eine Prefetchoperation für a_i direkt mit dem Beginn der Bedienung von $\sigma(j)$ gestartet wird. Im Spezialfall $j = i$ stellt die Kante eine Prefetchoperation dar, die erst nach der Bedienung von $\sigma(i-1)$ gestartet wird. Wenn $i - j < F$ gilt, muss das System $F - (i - j)$ Zeiteinheiten warten, sodass wir der Kante (s_i, x_j) Kosten von $\max\{0, F - (i - j)\}$ zuweisen. In Abbildung 9.1 stellen wir diese Konstruktion für die beispielhafte Anfragesequenz $\sigma = a, b, c, b, c$ und den Wert $F = 2$ bildlich dar. Diejenigen Kanten, die aus einer der Quellen s_i, $1 \le i \le 5$, herauslaufen, sind mit ihren Kosten beschriftet.

Bislang lässt es unsere Konstruktion zu, dass ein Flussalgorithmus mehr als eine der Kanten, die zu parallel laufenden Prefetchoperationen korrespondieren, saturiert. (Man betrachte beispielsweise die Kanten (s_i, x_{i-1}) und (s_{i-1}, x_{i-2}) für ein i, sodass $\sigma(i-2)$, $\sigma(i-1)$ und $\sigma(i)$ paarweise verschieden sind.) Um IPC korrekt zu modellieren, müssen wir garantieren, dass zu jedem Zeitpunkt höchstens eine Prefetchoperation im Gange ist. In unserem Netzwerk haben wir uns das Leben in dieser Hinsicht insoweit etwas einfacher gemacht, als wir Knoten x_i und x_i' erzeugt haben, obwohl es so aussieht, als ob x_i' mit x_{i+1} jeweils verschmolzen werden könne. Kanten (s_i, x_j') sollen aber Fluss tragen, wenn ein Block im Cache bleibt, wohingegen Kanten (s_i, x_j) für Prefetchoperationen stehen. Deshalb unterteilen wir jetzt genau die Kanten (s_i, x_j) in mehrere Teilstücke und bezeichnen (s_i, x_j) im Folgenden nur noch als „Superkante“. Sei für beliebiges $\ell \in \{1, \dots, n-1\}$ mit $[\ell, \ell+1)$ das Zeitintervall bezeichnet, das mit der Bedienung von $\sigma(\ell)$ beginnt und unmittelbar vor der Bedienung von $\sigma(\ell+1)$ endet. Das Intervall $[0, 1)$ ist das Intervall vor der Bedienung von $\sigma(1)$.

Wir müssen für alle Werte von ℓ alle Prefetchoperationen betrachten, die zu irgendeinem Zeitpunkt im Zeitintervall $[\ell, \ell + 1)$ aktiv sein können. Eine Operation, die a_i

einbringt und mit der Bedienung von $\sigma(j)$ beginnt, $j < i$, ist aktiv während der Intervalle $[\ell, \ell+1)$ mit $\ell = j, \ldots, \min\{j+F, i\} - 1$. Für jedes i und j mit $1 \leq i \leq n$ und $p_i + 1 \leq j < i$ führen wir die Knoten v_{ij}^{ℓ} und w_{ij}^{ℓ} mit $\ell = j, \ldots, \min\{j+F, i\} - 1$ ein. Diese Knoten werden durch Kanten mit Kapazität 1 und Kosten 0 verbunden. Genauer gesagt existieren Kanten $(v_{ij}^{\ell}, w_{ij}^{\ell})$ mit $\ell = j, \ldots, \min\{j+F, i\} - 1$ sowie Kanten $(w_{ij}^{\ell}, v_{ij}^{\ell-1})$ mit $\ell = j+1, \ldots, \min\{j+F, i\} - 1$. Der letzte Knoten dieser Folge, w_{ij}^{j}, ist durch eine Kante mit Kosten 0 und Kapazität 1 mit x_j verbunden. Schließlich ergänzen wir noch die Kante (s_i, v_{ij}^{ℓ}) mit $\ell = \min\{j+F, i\} - 1$ zum ersten Knoten dieser Folge mit Kosten $F - (i-j)$ und Kapazität 1. In dieser Beschreibung haben wir bislang den Fall $j = i$ außen vor gelassen, da eine Prefetchoperation, die a_i einbringt und erst mit dem Versuch der Bedienung von $\sigma(i)$ gestartet wird, einen kleinen Sonderfall darstellt. Die Prefetchoperation wird nämlich vollständig in dem Zeitraum nach der Bedienung von $\sigma(i-1)$ und vor der Bedienung von $\sigma(i)$ ausgeführt und überschneidet sich daher mit keiner Anfrage. Außerdem erzeugt sie eine Wartezeit von F Zeiteinheiten und ist im Intervall $[i-1, i)$ aktiv. Wir führen zwei durch eine Kante mit Kapazität 1 und Kosten 0 verbundene Knoten v_{ii}^{i-1} und w_{ii}^{i-1} ein. Der Knoten w_{ii}^{i-1} ist mit x_i ebenfalls durch eine Kante mit Kapazität 1 und Kosten 0 verbunden. Schließlich gibt es noch die Kante (s_i, v_{ii}^{i-1}) mit Kapazität 1 und Kosten F.

Als Nächstes beschreiben wir die Rolle des $(n+1)$-ten Gutes, welches sicherstellen soll, dass keine zwei Prefetchoperationen zum selben Zeitpunkt aktiv sind, indem es genügend viele Kanten „verstopft“. Genauer gesagt garantieren wir, dass in jedem Intervall $[\ell, \ell+1)$ mit $1 \leq \ell \leq n-1$ höchstens eine Prefetchoperation ausgeführt wird. Sei f_ℓ die Anzahl der Prefetchoperationen, deren Ausführung sich mit dem Intervall $[\ell, \ell+1)$ überschneidet, d. h.

$$f_\ell := \left|\left\{v_{ij}^{\ell} \mid 1 \leq i \leq n,\ p_i + 1 \leq j \leq i\right\}\right| .$$

Gut $n+1$ erhält die Quelle s_{n+1}, die Senke t_{n+1} und eine Anforderung von $d_{n+1} := \sum_{\ell=1}^{n-1}(f_\ell - 1)$. Wir leiten den Fluss von s_{n+1} nach t_{n+1} über die Kanten $(v_{ij}^{\ell}, w_{ij}^{\ell})$ und neu eingeführte „Untersenken“ t_{n+1}^{ℓ}, $1 \leq \ell \leq n-1$. Für jedes Paar von Knoten v_{ij}^{ℓ} und w_{ij}^{ℓ} fügen wir die Kanten (s_{n+1}, v_{ij}^{ℓ}) und $(w_{ij}^{\ell}, t_{n+1}^{\ell})$ mit Kapazität 1 und Kosten 0 hinzu. Außerdem ergänzen wir Kanten $(t_{n+1}^{\ell}, t_{n+1})$ mit Kapazität $f_\ell - 1$ und Kosten 0. Abbildung 9.2 veranschaulicht diese Unterteilung der (s_i, x_j)-Kanten und die Rolle des $(n+1)$-ten Gutes anhand des Netzwerkes aus Abbildung 9.1.

Man muss sich jetzt klar machen, dass sich das Netzwerk dazu eignet, IPC mit $D = 1$ zu modellieren. Dazu untersuchen wir Eigenschaften von Flüssen des $(n+1)$-ten Gutes. Wir betrachten ein beliebiges, aber festes Intervall $[\ell, \ell+1)$ mit $1 \leq \ell \leq n-1$ und eine Prefetchoperation, die während des Intervalls $[\ell, \ell+1)$ aktiv ist. Wenn die Prefetchoperation a_i einbringt und zu Beginn der Bedienung von $\sigma(j)$ gestartet wird, wird dies mit der Superkante (s_i, x_j) modelliert, die wiederum eine Kante $(v_{ij}^{\ell}, w_{ij}^{\ell})$ enthält. Für jedes ℓ gibt es f_ℓ solcher Kanten. Wir beschränken uns jetzt vorübergehend auf ganzzahlige Flüsse, d. h. Flüsse, die für jedes Gut und jede Kante der Kante einen

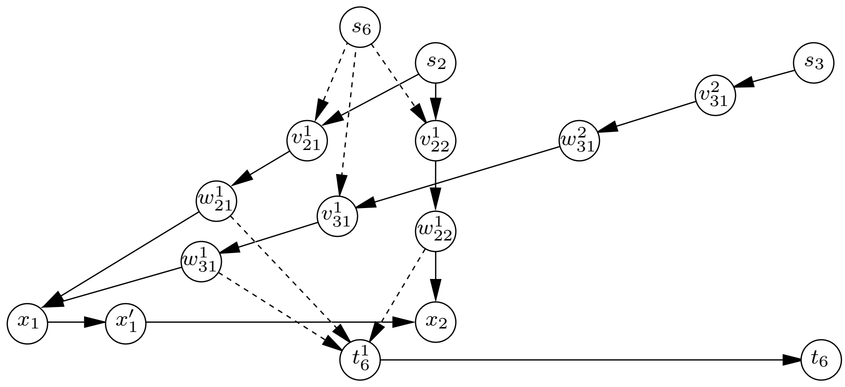

Abbildung 9.2: Vergrößerung des Flussnetzwerks mit unterteilten Kanten

ganzzahligen Flusswert des jeweiligen Gutes zuordnen. Dann garantieren für jedes ℓ die Kapazitäten $f_\ell - 1$ der Kanten (t^ℓ_{n+1}, t_{n+1}), dass höchstens eine der Kanten $(v^\ell_{ij}, w^\ell_{ij})$ einen Fluss eines Gutes $i \leq n$ tragen kann. Trügen zwei oder mehr solcher Kanten Fluss eines Gutes $i \leq n$, wäre die Kapazitätsbeschränkung an irgendeiner Kante $(t^{\ell'}_{n+1}, t_{n+1})$ mit $\ell' \neq \ell$ verletzt oder die Anforderung d_{n+1} würde nicht geleistet.

Unmittelbar aus der Konstruktion des Netzwerks folgt nun, dass eine Prefetchoperation, die a_i einbringt und zu Beginn der Bedienung von $\sigma(j)$ oder nach der Bedienung von $\sigma(j-1)$ gestartet wird, anhand eines Flusses des Gutes i mit Wert 1 von s_i nach x_j auf solche Weise modelliert werden kann, dass die Kosten dieses Teilflusses der Wartezeit für die Prefetchoperation entsprechen. Wir müssen uns aber klar machen, dass wir tatsächlich Fluss über diesen Weg schicken können, ohne die Anforderung an das $(n+1)$-te Gut zu verletzen. Da keine zwei Prefetchoperationen gleichzeitig möglich sind, kann es nicht passieren, dass in einem legalen Prefetching-und-Caching-Plan mehr als eine Prefetchoperation ein ganzes Intervall $[i, j)$, $j \geq i+1$, schneidet. Allerdings haben wir mit der Modellierung des oben erwähnten Sonderfalls von Prefetchoperationen der Länge F die Menge der erlaubten Pläne sogar weiter eingeschränkt. Während im Netzwerk modellierte Prefetchoperationen der Länge F weitere Operationen in einem bestimmten Intervall $[i, i+1)$ verbieten, könnte es sein, dass in einem erlaubten Plan eine erste Prefetchoperation O_1 nach der Bedienung von $\sigma(i)$ endet und eine zweite Operation O_2 komplett zwischen den Bedienungen von $\sigma(i)$ und $\sigma(i+1)$ abläuft. Damit benutzen zwei Prefetchoperationen das Intervall $[i, i+1)$. In einem solchen Fall bringt O_2 den Block a_{i+1} ein und O_1 bringt einen Block $a_{i'}$ mit $i' \geq i+1$ ein. Indem wir O_1 stattdessen den Block a_{i+1} einbringen lassen und statt O_2 eine neue Anfrage O'_2 komplett zwischen den Anfragen $\sigma(i'-1)$ und $\sigma(i')$ ablaufen und den Block $a_{i'}$ einbringen lassen, ändern wir den Plan, ohne die Kosten zu ändern. Indem man das Verfahren iteriert, kommt man zu einem ebenso teuren Plan, in dem der Sonderfall nicht mehr auftritt. Wir machen uns jetzt noch klar, dass die Kapazitäten der anderen Kanten ausreichen, um einen zulässigen Fluss zu erzeugen. Wir können also einen optimalen Plan geeignet in einen zulässigen Mehrgüterfluss mit denselben Kosten übersetzen.

Andererseits können wir aus einem zulässigen Mehrgüterfluss die Prefetchoperationen und das Entfernen von Blöcken in einem Prefetching-und-Caching-Plan ablesen: Wir haben gezeigt, dass für jedes Intervall $[\ell, \ell + 1)$ höchstens eine der das Intervall schneidenden Superkanten (s_i, x_j) Fluss tragen kann. Wenn ein Block zwischen zwei seiner Anfragen im Cache bleibt, wird eine (s_i, x'_j)-Kante benutzt. Die Kapazität der Kanten (x_i, x'_i) und (x'_i, x_{i+1}) garantiert, dass nicht mehr als k Blöcke gleichzeitig im Cache sein können. Jedoch erzwingen die Anforderungen d_i, $i \leq n$, dass jeder Block bei seiner Anfrage im Cache ist. Wir erhalten also folgende Korrektheitsaussage.

Lemma 9.1.2 *Jeder zulässige und ganzzahlige Mehrgüterfluss mit Kosten C in G entspricht einem zulässigen Prefetching-und-Caching-Plan mit Gesamtwartezeit C für σ und umgekehrt.*

Bislang haben wir verschwiegen, dass es im Szenario von Definition 9.1.1 noch nicht einmal dann einen kostenminimalen, ganzzahligen Mehrgüterfluss zu geben braucht, wenn alle Anforderungen und Kapazitäten ganzzahlig sind. Dies steht im Gegensatz zu den herkömmlichen Flussproblemen mit lediglich einem Gut, da man dort mithilfe der gängigen Algorithmen sogar einen optimalen ganzzahligen Fluss berechnen kann, sofern die Parameter des Netzwerks ganzzahlig sind.

Wir müssen also damit rechnen, dass auch ein exakter Algorithmus zur Lösung des Mehrgüterflussproblems auf dem Netzwerk G eine Lösung mit gebrochenen (rationalen) Werten ausgibt. Genau dies kann auch passieren, wenn man das lineare Optimierungsproblem aus der Arbeit von Albers, Garg und Leonardi (2000) löst. Dort wird deshalb auch folgende Erweiterung von Lösungen für das Problem IPC vorgeschlagen.

Definition 9.1.3 (gebrochene Lösung) *Es sei eine Instanz für IPC gegeben. Die Menge der* gebrochenen Lösungen *für die Instanz ist eine Obermenge der ganzzahligen Lösungen. Eine gebrochene Lösung darf von einer ganzzahligen auf folgende Weise abweichen:*

- *Der Anteil, zu dem ein Block im Cache ist, darf gebrochene Werte zwischen* 0 *und* 1 *annehmen. Allerdings muss sich jeder Block zum Zeitpunkt seiner Anfrage vollständig im Cache befinden.*
- *Gebrochene Anteile von Blöcken im Cache entstehen infolge von Operationen, die Blöcke teilweise in den Cache einbringen oder teilweise aus dem Cache entfernen. In jedem Zeitintervall $[\ell, \ell+1)$ darf der Gesamtanteil eingebrachter Blöcke höchstens so groß sein wie der Gesamtanteil entfernter Blöcke und der Gesamtanteil eingebrachter Blöcke darf höchstens* 1 *sein.*
- *Wartezeiten werden wie folgt angerechnet: Wenn mit der Bearbeitung von $\sigma(i)$ eine Prefetchoperation gestartet wird, die $\delta \in [0, 1]$ Einheiten von Block $\sigma(j)$ einbringt und $j - i < F$ gilt, entsteht eine Wartezeit von $\delta(F - (j - i))$ Zeiteinheiten.*

Salopp gesagt besteht der wesentliche Unterschied zwischen ganzzahligen und gebrochenen Lösungen für IPC darin, dass es möglich wird, Prefetchoperationen zu unterbrechen und Teile eines Blockes zwischen aufeinander folgenden Anfragen an diesen Block im Cache zu lassen. Im Netzwerk G entspricht ein teilweises Entfernen von δ Einheiten des Blockes $\sigma(j)$ einem Fluss, der nur $1-\delta$ Einheiten über eine Kante (s_i, x'_j), wenn $\sigma(i)$ die nächste Anfrage ist, schickt. Also modelliert unser Netzwerk nur gebrochene Lösungen, bei denen Blöcke nach ihrer Anfrage erst (teilweise) entfernt und dann nur noch zurückgebracht werden. Es ist klar, dass jede gebrochene Lösung, die diese Eigenschaft nicht erfüllt, also Einbringen und Entfernen zwischen den Anfragen mischt, in eine gebrochene Lösung transformiert werden kann, die sie erfüllt und höchstens dieselbe Gesamtwartezeit besitzt. Wir können also jetzt auch mit gebrochenen Mehrgüterflüssen argumentieren.

Lemma 9.1.4 *Jeder zulässige Mehrgüterfluss mit Kosten C in G entspricht einem zulässigen, möglicherweise gebrochenen Prefetching-und-Caching-Plan mit Gesamtwartezeit C für σ. Es gibt einen zulässigen Mehrgüterfluss, dessen Kosten denen eines optimalen (gebrochenen) Prefetching-und-Caching-Plans für σ entsprechen.*

Die oben erwähnte, wesentliche Erkenntnis von Albers, Garg und Leonardi (2000) besteht in einem Verfahren, mit dessen Hilfe wir eine ganzzahlige Lösung aus einer gebrochenen Lösung für IPC extrahieren können.

Lemma 9.1.5 *Sei L eine gebrochene Lösung für eine IPC-Instanz. Es gibt einen kombinatorischen Polynomialzeitalgorithmus, der aus L einen ganzzahligen Prefetching-und-Caching-Plan berechnet, dessen Gesamtwartezeit höchstens so groß wie die von L ist.*

Es würde den Rahmen dieser Arbeit bei weitem sprengen, den Algorithmus, der Lemma 9.1.5 zugrunde liegt, samt seinem Korrektheitsbeweis vorzustellen. Wir werden aber später (Abschnitt 9.2) ohnehin auf die von Albers, Garg und Leonardi (2000) präsentierte Idee der Formulierung von IPC als lineares Optimierungsproblem zu sprechen kommen und umreißen daher im Folgenden ein paar wesentliche Punkte der Transformation einer gebrochenen Lösung.

Die Autoren repräsentieren gebrochene Lösungen für IPC mit $D = 1$ und Anfragesequenzen σ der Länge n mithilfe von Variablen $x(I)$, die sich auf Intervalle $I = (i, j)$ mit ganzzahligen Grenzen $1 \leq i < j \leq n$ beziehen. Eine solche Variable erhält den Wert δ, wenn in dem Intervall, das nach (!) der Bedienung von $\sigma(i)$ beginnt und vor der von $\sigma(j)$ endet, ein δ-Anteil von bestimmten Blöcken (die durch andere Variablen kodiert sind) eingebracht oder entfernt wird. Die erste Idee ist es dann, eine Lösung, d. h. Belegung der $x(I)$ und anderer Variablen, so zu transformieren, dass relevante Intervalle einer natürlichen totalen Ordnung folgen. Dies geschieht durch Elimination folgenden Typs von Intervallen.

Definition 9.1.6 *Seien $I_1 = (i_1, j_1)$ und $I_2 = (i_2, j_2)$ zwei Intervalle. Das Intervall I_1 ist im Intervall I_2* echt enthalten*, wenn $i_1 > i_2$ und $j_1 < j_2$ gilt. Das Paar (I_1, I_2) heißt dann* verschachteltes Paar.

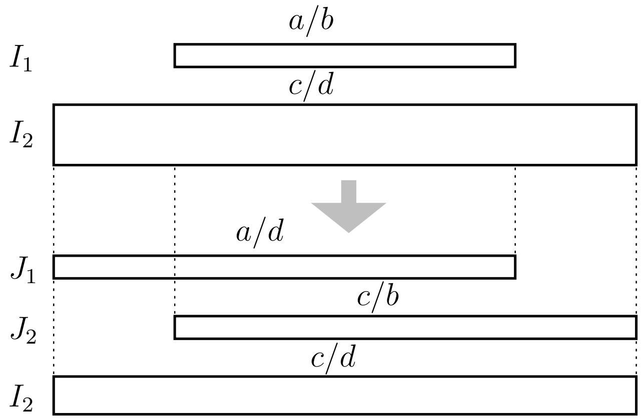

Abbildung 9.3: Elimination verschachtelter Intervalle

Wir betrachten ein verschachteltes Paar (I_1, I_2), für das eine gebrochene Lösung sowohl $x(I_1)$ als auch $x(I_2)$ mit positiven Werten belegt. Sei $x := \min\{x(I_1), x(I_2)\}$. Indem wir sowohl $x(I_1)$ als auch $x(I_2)$ um den Betrag x reduzieren, wird mindestens eine der beiden Variablen 0 und wir können das zugehörige Intervall oder die zugehörigen Intervalle wegfallen lassen. Dafür manipulieren wir die Variablen der Intervalle $J_1 = (i_2, j_1)$ und $J_2 = (i_1, j_2)$, indem wir sowohl $x(J_1)$ als auch $x(J_2)$ um den Wert x erhöhen. In den Intervallen J_1 und J_2 bringen wir jetzt außerdem den x-Betrag von Blöcken ein, die in I_1 (bzw. I_2) eingebracht werden und entfernen den x-Betrag von Blöcken, die in I_2 (bzw. I_1) entfernt werden. Da die Endindizes von I_1 und J_1 übereinstimmen, kommen die Blöcke, die ursprünglich in I_1 eingebracht wurden, jetzt immer noch zur selben Zeit wie zuvor im Cache an. Eine analoge Aussage gilt für die entfernten Blöcke. Dann macht man sich noch klar, dass nach der Transformation in keinem Intervall zu viel eingebracht oder entfernt wird. Zuletzt überlegt man sich, dass der Wert der Zielfunktion unverändert bleibt. Abbildung 9.3 (in Anlehnung an Albers, Garg und Leonardi, 2000) illustriert dieses Vorgehen.

Nach dem Entfernen verschachtelter Intervalle werden die neuen bzw. verbleibenden Intervalle nach aufsteigendem Startpunkt (und bei Gleichheit nach aufsteigendem Endpunkt) sortiert. Anschließend finden einige weitere Manipulationen statt. Aufgrund der Ordnung der Intervalle ist es möglich, diese in einer Art zeitlicher Reihenfolge zu betrachten und für diese Art von Zeit Verallgemeinerungen der in Abschnitt 8.3 vorgestellten vier Regeln für optimale Pläne aufzustellen. Diese neuen Regeln gestatten es nun, die gebrochene Lösung, ohne dass sich der Wert der Lösung erhöht, so umzugestalten, dass Prefetchoperationen nicht mehr unterbrochen werden. Die nächste Schwierigkeit besteht darin, dass es nicht gelingt, auf analoge Weise den Prozess der Entfernung von Blöcken aus dem Cache ohne Unterbrechungen zu gestalten. Als wichtige Beobachtung fällt aber auf, dass solche Unterbrechungen (im zugrunde gelegten Zeitbegriff) stets ganzzahlige Zeiteinheiten lang sind. Damit wird es möglich, für jeden festen, gebrochenen Wert $t \in [0,1]$ die eingebrachten/entfernten Blöcke an den Zeitpunkten $t+i$, $i \in \mathbb{N}$, zu beobachten und nachzuweisen, dass diese zu einem legalen und ganzzahligen Prefetching-und-Caching-Plan korrespondieren. Im Folgenden nennen wir dieses Vorgehen das *Abgreifen* einer ganzzahligen Lösung mithilfe des Wertes t.

Zu guter Letzt weisen Albers, Garg und Leonardi (2000) nach, dass die Menge verschiedener ganzzahliger Lösungen, die man abgreifen kann, indem man mit dem zuvor beschriebenen t das Intervall $[0,1]$ durchläuft, in ihrer Mächtigkeit durch Fn^2 nach oben beschränkt ist. Es ist dann möglich, das Intervall $[0,1]$ disjunkt in q Intervalle $[t_{p-1}, t_p]$ mit $1 \leq p \leq q$, $t_0 = 0$ und $t_q = 1$ zu zerlegen, sodass wir für festes p bei allen Werten $t \in [t_{p-1}, t_p]$ dieselbe ganzzahlige Lösung abgreifen. Es bezeichne $\mathrm{val}(p)$ den Wert der p-ten ganzzahligen Lösung, d. h. die Wartezeit, die bei Realisierung des entsprechenden Prefetching-und-Caching-Plans entsteht. Die allerletzte benötigte Beobachtung besagt, dass diese Lösung einen Beitrag von genau $t_p - t_{p-1}$ zum Wert der gebrochenen Lösung, die wir vor dem Abgreifen hatten, leistet. Da sich die Differenzen $t_p - t_{p-1}$ zu 1 summieren, können wir sagen, dass der Wert der gebrochenen Lösung eine Konvexkombination der q abgegriffenen, ganzzahligen Lösungen darstellt. Es folgt, dass mindestens eine der ganzzahligen Lösungen in ihrem Wert durch den der gebrochenen Lösung beschränkt ist. Wenn die gebrochene Lösung sogar optimal ist, stellt jede abgegriffene Lösung eine optimale Lösung dar und wir können sofort an $t = 0$ abgreifen. Nachfolgend werden wir aber bei einer Anwendung von Lemma 9.1.5 die Menge der abgreifbaren ganzzahligen Lösungen durchsuchen müssen, was wegen $q \leq nF^2$ in Polynomialzeit möglich ist.

9.1.2 Der kombinatorische Polynomialzeitalgorithmus

Wir haben im vergangenen Abschnitt gesehen, dass aus einer exakten, optimalen, aber möglicherweise gebrochenen Lösung des Mehrgüterflussproblems auf dem Netzwerk G in Polynomialzeit eine optimale und ganzzahlige Lösung für IPC abgeleitet werden kann. Andererseits wollen wir keinen exakten Algorithmus zur Lösung des Mehrgüterflussproblems einsetzen, der nicht kombinatorisch ist, und weichen auf den auf Seite 191 erwähnten, kombinatorischen Ansatz von Kamath, Palmon und Plotkin (1995) aus. Wir erinnern daran, dass dieser mit ε und δ parametrisiert ist, alle Anforderungen an die Güter zu einem Anteil von mindestens $(1-\varepsilon)$ erfüllt und in den Kosten nur um einen Faktor $(1+\delta)$ vom Optimum abweicht.

Theorem 9.1.7 *Sei eine Instanz für IPC mit $D = 1$ gegeben und sei G das zugehörige Netzwerk gemäß Abschnitt 9.1.1. Es ist möglich, durch Anwendung eines Approximationsalgorithmus für das Mehrgüterflussproblem auf G mit kombinatorischen Metho-*

den in Polynomialzeit zu einer optimalen, ganzzahligen Lösung für die IPC-Instanz zu kommen.

Beweis: Der Beweis beruht an verschiedenen Stellen auf Argumenten über Ganzzahligkeitsbedingungen. Wir nehmen das Netzwerk G als gegeben an, setzen $\varepsilon := 1/(4F^2n^3)$, $\delta := 1/(3nF)$ und lassen den kombinatorischen Approximationsalgorithmus von Kamath, Palmon und Plotkin (1995) laufen. Dieser gibt eine Lösung aus, die für die Güter $1 \leq i \leq n$ mindestens einen Fluss von $1 - \varepsilon$ durch das Netzwerk leitet.

Man sieht leicht, dass für die Anforderung an Gut $n + 1$ die Beziehung $d_{n+1} \leq Fn^2$ erfüllt ist. Also beträgt der Fluss von Gut $n + 1$ mindestens $d_{n+1} - \varepsilon Fn^2$. Da die Anforderung an Gut $n + 1$ nicht zur Gänze erfüllt ist, kann ein Fluss von insgesamt mehr als 1 über diejenigen Kanten fließen, die Prefetchoperationen in einem Intervall $[\ell, \ell+1)$ symbolisieren. Allerdings ist dieser Fluss für jedes Intervall immer noch durch $1 + \varepsilon Fn^2$ nach oben beschränkt.

Sei $\rho := 1 - \varepsilon - \varepsilon Fn^2$. Wir wollen erreichen, dass jedes Gut einen Fluss von genau ρ durch das Netzwerk schickt. Dies ist möglich, indem wir auf jeder Kante den Fluss von Gut i durch Multiplikation mit einem Proportionalitätsfaktor von höchstens 1 geeignet reduzieren. Wir gelangen damit zu einem gleichförmigen Mehrgüterfluss, der über jedes Intervall nur noch einen Fluss von höchstens 1 leitet und für alle Güter einen Fluss von genau ρ von der jeweiligen Quelle zur Senke schickt. Sei dieser neue Fluss mit ϕ' bezeichnet.

Im Hinblick auf Definition 9.1.3 und Lemma 9.1.4 haben wir es bei ϕ' mit einem Fluss zu tun, der eine gebrochene Lösung für IPC darstellt, in der alle Speicherblöcke eine Größe von ρ haben und der Cache eine Größe von k/ρ besitzt. Im Folgenden nennen wir eine solche Lösung eine *ρ-Lösung*. Laut Lemma 9.1.5 dürfen wir die ρ-Lösung, die dem Fluss ϕ' entspricht, als (konvexe) Kombination von *ganzzahligen* ρ-Lösungen auffassen.

Um die Güte der beschriebenen Kombination ganzzahliger ρ-Lösungen zu analysieren, benötigen wir eine untere Schranke für ρ. Es ist $\rho \geq 1 - \varepsilon - \varepsilon Fn^2 \geq 1 - 1/(2nF)$ aufgrund der Wahl von ε. Als Nächstes schätzen wir die Kosten C der Konvexkombination von ρ-Lösungen ab. Da der Approximationsalgorithmus eine Lösung mit Kosten von höchstens $(1 + \delta)\,\mathrm{OPT}$ ausgibt, wenn OPT die Kosten eines optimalen Mehrgüterflusses sind, und da die oben beschriebene Reduktion der Flüsse der Güter $1, \ldots, n$ die Kosten nicht erhöhen kann, folgt wegen $\mathrm{OPT} \leq nF$ die Abschätzung

$$C \;\leq\; (1 + \delta)\,\mathrm{OPT} \;=\; \mathrm{OPT} + \frac{\mathrm{OPT}}{3nF} \;\leq\; \mathrm{OPT} + \frac{1}{3}.$$

Weiterhin haben wir die Kosten C der Konvexkombination von ρ-Lösungen um einen additiven Term von höchstens $n(1 - \rho)F$ unterschätzt. Dies liegt daran, dass jeder Block, der zu einem Gut korrespondiert, in Wahrheit eine Größe von 1, aber eine Größe von ρ in der Konvexkombination hat. Indem wir die Blockgröße (oder, gleicherweise, den Fluss des entsprechenden Gutes) erhöhen, können die Kosten um höchstens $(1-\rho)F$ ansteigen. Also betragen die Kosten C' der Konvexkombination ganzzahliger ρ-Lösungen höchstens

$$C' \leq C + n(1-\rho)F \leq \mathrm{OPT} + \frac{1}{3} + \frac{nF}{2nF} < \mathrm{OPT} + 1.$$

Aus der Ungleichung $C' < \mathrm{OPT} + 1$ schließen wir, dass wir anhand von Lemma 9.1.5 aus der Konvexkombination mindestens eine ganzzahlige Lösung abgreifen können, die kostenoptimal ist. Allerdings beruht eine solche ganzzahlige Lösung auf einer ρ-Lösung, d. h., die Größe des Caches beträgt k/ρ. Da die tatsächliche Cachegröße k beträgt, müssen wir nachweisen, dass keine ganzzahlige Lösung, die Bestandteil der Konvexkombination ist, jemals zu mehr als k Blöcken im Cache führt.

Da IPC für $k \geq n$ und für $F = 1$ trivial wird, nehmen wir nun $k/(nF) < 1$ an. Damit ist die Zahl der Blöcke der Größe ρ, die sich in der gebrochenen ρ-Lösung zugleich im Cache befinden, höchstens

$$\frac{k}{\rho} \leq \frac{k}{1 - 1/(2nF)} \leq k\left(1 + \frac{1}{nF}\right) < k + 1.$$

Die zweite Ungleichung folgt dabei aus der Ungleichung $(1-\gamma)^{-1} \leq 1 + 2\gamma$ für $\gamma \in [0, 1/2]$. Also haben wir $(k+1)\rho > k$ gezeigt und erhielten einen Widerspruch, falls eine ganzzahlige Lösung zu irgendeinem Zeitpunkt mehr als k Blöcke im Cache hielte. Somit haben wir die Ausgabe des Approximationsalgorithmus zu einem zulässigen, ganzzahligen und optimalen Prefetching-und-Caching-Plan umgewandelt. Die Gesamtlaufzeit des Approximationsalgorithmus beträgt $O^*(\varepsilon^{-3}\delta^{-3}g|E||V|^2)$, kann also wegen $|V| = O(n^2)$ und $|E| = O(n^2)$ gemäß $O^*((nF)^3(n^3F^2)^3(n+1)n^2n^4) = O^*(n^{19}F^9)$ nach oben abgeschätzt werden. Da der in Lemma 9.1.5 genannte Algorithmus ebenfalls kombinatorisch ist und in Polynomialzeit arbeitet, ist das Theorem gezeigt. □

Wir haben gezeigt, dass IPC mit $D = 1$ in Polynomialzeit gelöst werden kann, ohne auf eine Formulierung als lineares Optimierungsproblem auszuweichen. Zweifellos ist die polynomielle obere Schranke, die sich im Beweis von Theorem 9.1.7 versteckt, ein so großes Polynom, dass man dem Algorithmus von vornherein vermutlich keine praktische Effizienz beimessen will. Bislang ist jedenfalls auch keine Implementierung des Ansatzes bekannt. Jedoch haben unsere Untersuchungen einen strukturellen Wert. Es ist gelungen, das Zusammenspiel von Prefetching und Caching so gut zu verstehen, dass eine Formulierung als Mehrgüterflussproblem möglich wurde. Diese Formulierung könnte Ausgangspunkt für noch besser handhabbare Netzwerke sein. Zudem hat sich unser Verständnis von der Struktur der von Albers, Garg und Leonardi (2000) vorgestellten gebrochenen Lösungen für IPC verbessert. Damit wurde es möglich, Ganzzahligkeitsbedingungen so geschickt auszunutzen, dass wir aus dem Lauf eines Approximationsalgorithmus für das Flussnetzwerk eine optimale, ganzzahlige Lösung ablesen konnten.

9.2 Neue Ansätze für Approximationsalgorithmen

In diesem Abschnitt betrachten wir das parallele Prefetching und Caching, d. h. IPC mit $D > 1$. Da wir bereits wissen, dass die Berechnung einer exakten Lösung NP-hart ist, versuchen wir uns an verbesserten Approximationsalgorithmen. Dies gelingt, indem wir die Formulierung von IPC als binäres Optimierungsproblem, vorgestellt von Albers, Garg und Leonardi (2000), aufbereiten. Während die genannten Autoren lediglich eine D-Approximation zeigen konnten, werden wir die Approximationsgüte für konstante Werte von D auf 2 verbessern.

Wir haben in Abschnitt 9.1.1 bereits einige Ideen der Formulierung von IPC als binäres Optimierungsproblem (vgl. Albers, Garg und Leonardi, 2000) vorgestellt. Im Folgenden wollen wir ein binäres Optimierungsproblem erarbeiten, bei dem wir später die Ganzzahligkeitsbedingungen relaxieren werden. Sei n wieder die Länge der Anfragesequenz σ in einer Instanz für IPC. Wir arbeiten mit offenen Intervallen $I = (i, j)$ mit $0 \leq i < j \leq n$. Ein solches Intervall bezieht sich auf die Zeitspanne, die nach der Bedienung von $\sigma(i)$ beginnt und vor der Bedienung von $\sigma(j)$ endet. Seine Länge beträgt $|I| = j - i - 1$. Falls $|I| < F$ gilt und innerhalb des Intervalls eine Prefetchoperation aktiv sein soll (die auch in dem Intervall endet), dann müssen $\max\{0, F - |I|\}$ Zeiteinheiten Wartezeit entstehen. Auf der anderen Seite können wir uns deswegen auch auf Intervalle mit einer Maximallänge von F beschränken. Für jedes verbleibende Intervall I führen wir nun Kopien I^d für jede Platte $d \in \{1, \ldots, D\}$ ein. Sei nun $\mathcal{I}$ die so erhaltene Menge von Intervallen. Die erwähnte Formulierung als Optimierungsproblem legt fest, in welchem der Intervalle aus $\mathcal{I}$ Prefetchoperationen ausgeführt werden sollen. Damit werden Wartezeiten getrennt für Intervalle und Platten gezählt und deshalb die Gesamtwartezeit im schlimmsten Fall um einen Faktor von ungefähr D überschätzt.

Unsere Idee einer Erweiterung der Formulierung von Albers, Garg und Leonardi (2000) besteht im Wesentlichen darin, *Bündel* von Intervallen zu erzeugen und jedes Bündel als eine Einheit zu betrachten. In jedem Bündel können dann nur entweder in allen zugehörigen Intervallen oder in keinem der Intervalle Prefetchoperationen laufen. Aus technischen Gründen können wir keine Intervalle gebrauchen, die im Sinne von Definition 9.1.6 echt ineinander enthalten sind.

Lemma 9.2.1 *Ein optimaler, möglicherweise gebrochener Prefetching-und-Caching-Plan für den Fall $D > 1$ kann in Polynomialzeit in einen optimalen, möglicherweise gebrochenen Plan transformiert werden, sodass in keinen zwei Intervallen, die echt ineinander enthalten sind, gleichzeitig Prefetchoperationen ausgeführt werden.*

Die Beweisidee für Lemma 9.2.1 haben wir bereits in Abschnitt 9.1.1 vorgestellt. Es ist dabei egal, ob die beiden betrachteten Intervalle mit derselben oder mit verschiedenen Platten assoziiert sind.

Wir kommen nun zur Definition eines Bündels. Ein einfaches Beispiel wäre die Menge von Intervallen $\{(i-1,i+1),(i,i+2)\}$, wenn die beiden Intervalle zu verschiedenen Platten gehören. Die Wartezeit von $F-1$, die entsteht, wenn im Intervall $(i-1,i+1)$ eine Prefetchoperation ausgeführt wird, braucht am Ende des zweiten Intervalls nicht erneut veranschlagt zu werden, da Prefetchoperationen für verschiedene Platten parallel ausgeführt werden können.

Definition 9.2.2 *Zwei Intervalle $I=(i_1,j_1)$ und $J=(i_2,j_2)$ mit $i_1 \le i_2$ heißen* überlappend, *wenn entweder $i_2 < j_1 - 1$ gilt oder $i_2 = j_1 - 1$ und zusätzlich $|I| < F$ gilt. Eine Menge von Intervallen B heißt* überlappend, *wenn sie keine verschachtelten Paare enthält und für alle bis auf ein $I=(i_1,j_1) \in B$ ein $J=(i_2,j_2) \in B$ mit $J \ne I$ enthält, sodass $i_1 \le i_2$ gilt und J mit I überlappt. Eine nicht leere Menge B von Intervallen heißt* Bündel, *wenn sie für jedes $d \in \{1,\dots,D\}$ höchstens ein Intervall enthält und überlappend ist.*

Eine einelementige Menge von Intervallen ist also immer ein Bündel. Wenn D nicht konstant ist, wird die Menge aller möglichen Bündel exponentiell groß. Wir wählen daher ein $z \le D$, das im Folgenden häufig als Konstante angesehen wird. Weiterhin nehmen wir der Einfachheit halber an, dass $D/z \in \mathbb{N}$ ist. Nun partitionieren wir die Menge der Platten in D/z Mengen

$$\{1,\dots,z\},\{z+1,\dots,2z\},\dots,\{D-z+1,\dots,D\}.$$

Sei $\mathcal{B}_z$ die Menge aller Bündel, die ausschließlich Intervalle enthalten, die aus $\mathcal{I}$ sind und sich jeweils auf dieselbe Teilmenge der oben erwähnten Partition beziehen. Da jedes Intervall höchstens $F+1$ verschiedene Längen annehmen kann, es höchstens n Möglichkeiten für den kleinsten Startindex im Bündel gibt, wir D/z Mengen betrachten und es $z!$ Wahlen für die Zugehörigkeit von Intervallen zu verschiedenen Platten gibt, folgt

$$|\mathcal{B}_z| \;\le\; n(F+1)^{2z} z! \cdot \frac{D}{z},$$

sodass wir für konstante z nur polynomiell viele Bündel erzeugen.

Um in einem linearen Optimierungsproblem mit den Bündeln zu arbeiten, müssen wir uns überlegen, wie Wartezeiten anzurechnen sind. Sei irgendein $B \in \mathcal{B}_z$ betrachtet. Wir sortieren die Intervalle, die in B enthalten sind, gemäß aufsteigendem Endindex und bei Gleichheit nach aufsteigendem Startindex; wenn beide gleich sind, ist die Ordnung beliebig. Sei $(a_1,b_1),\dots,(a_m,b_m)$ das Ergebnis dieser Sortierung. Seien $i_1 \le \cdots \le i_{m'}$ alle Indizes aus der Folge $b_1,\dots,b_m$ mit der Eigenschaft $b_{i_j+1} > b_{i_j}$ und sei $i_{m'+1} := m$. Für $j=1,\dots,m'+1$ gilt nun, dass das Intervall (a_{i_j},b_{i_j}) ein kürzestes Intervall mit Endindex b_{i_j} darstellt und damit die Haltezeit bestimmt, die vor der Anfrage $\sigma(b_{i_j})$ entsteht. Wir definieren uns nun induktiv die Funktion $v(j)$ mit $1 \le j \le m'+1$. Diese soll die Wartezeit vor der Bedienung von $\sigma(b_{i_j})$ nach oben beschränken, wenn in allen Intervallen des Bündels Prefetchoperationen aktiv sind und keine weiteren Prefetchoperationen existieren. Sei nun

$$v(j) \;:=\; \max\left\{0, F - (b_{i_j} - a_{i_j} - 1) - \sum_{r<j \;|\; a_{i_j}+1 \le b_{i_r} \le b_{i_j}-1} v(r)\right\}.$$

Schließlich setzen wir $w(B) := \sum_{j=1}^{m'+1} v(j)$.

Lemma 9.2.3 *Wenn ein Prefetching-und-Caching-Plan ausschließlich in allen Intervallen eines Bündels B Prefetchoperationen vorsieht und für jedes Intervall (i,j) das Einbringen von $\sigma(j)$ direkt nach der Bedienung von $\sigma(i)$ startet, beträgt die Summe der Wartezeiten genau $w(B)$.*

Beweis: Wir zeigen, dass bei einem solchen Plan vor der Bedienung von $\sigma(b_{i_j})$ tatsächlich Wartezeiten in Höhe von $v(j)$ anfallen. Da wir im Plan keine Prefetchoperationen in Intervallen außerhalb des Bündel zulassen, können zwischen zwei Anfragen $\sigma(i), \sigma(i+1)$ nur dann Wartezeiten anfallen, wenn $i+1$ Endpunkt eines Intervalls aus dem Bündel ist. Der Fall $j = 1$ folgt nun, da b_{i_1} der kleinste Endindex der Intervalle im Bündel ist und (a_{i_1}, b_{i_1}) ein kürzestes Intervall mit Endpunkt b_{i_1} ist.

Wenn vor $\sigma(b_{i_1}), \ldots, \sigma(b_{i_j})$ jeweils Wartezeiten von $v(1), \ldots, v(j)$ anfallen, fällt vor $\sigma(b_{i_{j+1}})$ schließlich eine Wartezeit von $v(j+1)$ an. Dies folgt, da wir wieder ein kürzestes Intervall mit diesem Endpunkt betrachten und die Definition von $v(j+1)$ die Wartezeit der Prefetchoperation, die nach $\sigma(a_{i_{j+1}})$ beginnt, um die anderen Wartezeiten, mit denen die Operation überlappt, korrigiert. □

Man beachte, dass die Voraussetzungen von Lemma 9.2.3 unerfüllbar sein könnten, wenn ein Bündel mehrere Intervalle derselben Platte enthielte.

Wir definieren jetzt die Variablen für unsere Formulierung als binäres Optimierungsproblem. Für jedes Bündel $B \in \mathcal{B}_z$ gibt es eine Variable $x(B)$, die 1 ist, wenn in allen Intervallen des Bündels eine Prefetchoperation ausgeführt werden soll, und 0 ist, wenn dies in keinem Intervall der Fall sein soll. Um zu spezifizieren, welche Blöcke eingebracht und entfernt werden sollen, führen wir die Variablen $f_{I^d,a}$ und $e_{I^d,a}$ für alle $I^d \in \mathcal{I}$ und alle Blöcke $a \in S_d$ (S_d ist die Menge der Blöcke, die auf der d-ten Platte liegen) ein. Die Variable $f_{I^d,a}$ ($e_{I^d,a}$) soll genau dann 1 sein, wenn Block a im Intervall I^d eingebracht (entfernt) wird. Weiterhin benötigen wir für $1 \le d \le D$ die Projektionen $\pi_d \colon \mathcal{B}_z \to \mathcal{I}$. Dabei soll $\pi_d(B) = I$ sein, wenn $I \in B$ ist und das Intervall I sich auf Platte d bezieht, und $\pi_d(B) = \emptyset$ sonst. Weil jedes Bündel für jede Platte höchstens ein Intervall enthält, ist π_d wohldefiniert.

Wir führen jetzt noch eine Inklusionsrelation für Intervalle ein, indem wir $(a,b) \subseteq (c,d)$ schreiben, wenn $c \le a$ und $d \ge b$ gilt. Nun können wir zur Idee des binären Optimierungsproblems kommen. Um zu garantieren, dass keine Prefetchoperationen derselben Platte gleichzeitig aktiv sind, fügen wir für alle $1 \leq i \leq n$ und alle $1 \leq d \leq D$ die Nebenbedingungen

$$\sum_{B \in \mathcal{B}_z:\ \pi_d(B) \supseteq (i-1,i)} x(B) \ \leq \ 1 \tag{9.1}$$

ein. Für jedes $d \in D$ und jedes Intervall $I^d \in \mathcal{I}$ müssen wir sicherstellen, dass höchstens ein Block der Platte in dem Intervall entfernt und eingebracht wird. Dies soll nur möglich sein, wenn eine zugehörige x-Variable auf 1 gesetzt ist. Wir erhalten die Nebenbedingungen

$$\sum_{a \in S} f_{I^d,a} \ = \ \sum_{a \in S} e_{I^d,a} \ \leq \ \sum_{B \in \mathcal{I}_z:\ \pi_d(B) = I^d} x(B). \tag{9.2}$$

Die folgende, dritte Menge von Bedingungen bezieht sich auf die Forderung, dass ein Block zum Zeitpunkt seiner Anfrage im Cache sein muss. Wir stellen zunächst für jeden Block a sicher, dass er zwischen zwei aufeinander folgenden Anfragen an ihn vor der nächsten Anfrage wieder eingebracht wird, wenn er entfernt wurde. Seien $a_1, \ldots, a_{n_a}$ die Anfragen an a aus der Sequenz σ. (Wir setzen $a_{n_a} = 0$ für Blöcke, die nie angefragt werden.) Also fordern wir für alle $a \in S$ und alle $i \in \{1, \ldots, n_a\}$

$$\sum_{I \in \mathcal{I}:\ I \subseteq (a_i, a_{i+1})} f_{I,a} \ = \ \sum_{I \in \mathcal{I}:\ I \subseteq (a_i, a_{i+1})} e_{I,a} \ \leq \ 1. \tag{9.3}$$

Da ein Block zum Zeitpunkt seiner ersten Anfrage im Cache sein muss, ergänzen wir für alle $a \in S$ die Bedingungen

$$\sum_{I \in \mathcal{I}:\ I \subseteq (0, a_1)} f_{I,a} \ = \ 1 \quad \text{und} \quad \sum_{I \in \mathcal{I}:\ I \subseteq (0, a_1)} e_{I,a} \ = \ 0. \tag{9.4}$$

Da nach der letzten Anfrage an einen Block nicht mehr als eine Einheit des Blockes aus dem Cache entfernt werden kann, schreiben wir für alle $a \in S$

$$\sum_{I \in \mathcal{I}:\ I \subseteq (a_{n_a}, n)} e_{I,a} \ \leq \ 1. \tag{9.5}$$

Schließlich ist sicherzustellen, dass jeder Block für die Dauer seiner Anfrage im Cache bleibt. Für alle $a \in S$ und alle $i \in \{1, \ldots, n_a\}$ muss

$$\sum_{I \in \mathcal{I}:\ I \supseteq (a_i - 1, a_i + 1)} f_{I,a} \ = \ \sum_{I \in \mathcal{I}:\ I \supseteq (a_i - 1, a_i + 1)} e_{I,a} \ = \ 0 \tag{9.6}$$

gelten. Hinzu kommen noch die Bedingungen

$$f_{I,a}, e_{I,a} \in \{0,1\} \quad \text{und} \quad x(B) \in \{0,1\} \tag{9.7}$$

für alle $I \in \mathcal{I}$, für alle $a \in S$ und alle $B \in \mathcal{B}_z$.

Ziel ist es nun, die Funktion

$$\sum_{B \in \mathcal{B}_z} w(B)x(B)$$

unter der Menge von Bedingungen (9.1)–(9.7) zu minimieren.

Wir müssen die Korrektheit dieser Formulierung nachweisen.

Lemma 9.2.4 *Eine Lösung für das binäre Optimierungsproblem mit Kosten C korrespondiert zu einem zulässigen Prefetching-und-Caching-Plan mit Kosten von höchstens C.*

Beweis: Sei eine beliebige Lösung für das binäre Optimierungsproblem gegeben. Die Belegungen der Variablen $f_{I,a}$ und $e_{I,a}$ sagen uns, in welchem Intervall welcher Block einzubringen bzw. zu entfernen ist, und die Nebenbedingungen stellen sicher, dass kein unzulässiger Plan möglich ist. Allerdings geben die Belegungen der Variablen lediglich an, zwischen welchen Anfragen eine Prefetchoperation zu starten ist, lassen aber den genauen Zeitpunkt des Beginns der Prefetchoperation offen, wenn zwischen den Anfragen Wartezeit besteht. Wir konstruieren daher auf induktive Weise einen Prefetching-und-Caching-Plan, dessen Wartezeit durch den Wert der Lösung des binären Optimierungsproblems beschränkt ist.

Zunächst sortieren wir die Bündel B, für die $x(B) = 1$ gilt, nach aufsteigendem maximalem Endindex (der gebündelten Intervalle) und bei Gleichheit nach aufsteigendem minimalem Startindex. Sei $B_1, \ldots, B_m$ die damit erhaltene Folge von Bündeln. Angenommen, wir haben bereits für die Bündel $B_1, \ldots, B_{r-1}$ einen Plan erstellt. Für das Bündel B_r müssen wir das Einbringen und Entfernen für diejenigen Blöcke planen, deren Variablen $f_{I,a}$ und $e_{I,a}$ auf 1 gesetzt wurden und für die $I \in B_r$ gilt. Wir greifen jetzt die auf Seite 202 im Zusammenhang mit der Definition von $w(B)$ eingeführte Notation auf. Sei $(a_1, b_1), \ldots, (a_m, b_m)$ die entsprechende Folge von Intervallen aus B_r. Als Erstes fügen wir $v(\ell)$ Zeiteinheiten Wartezeit vor den Anfragen $\sigma(b_{i_\ell})$ mit $1 \leq \ell \leq m' + 1$ ein und planen die Prefetchoperationen in den Intervallen (a_i, b_i), $1 \leq i \leq m$, wie folgt. Die Prefetchoperation im Intervall (a_1, b_1) wird zum spätestmöglichen Zeitpunkt gestartet, d. h., die Operation wird unmittelbar mit Bedienung der Anfrage $\sigma(a_1 + 1)$ gestartet. Die Prefetchoperationen in den Intervallen (a_i, b_i) mit $i \geq 2$ hingegen werden zum frühestmöglichen Zeitpunkt nach a_i gestartet, an dem die zugehörige Platte verfügbar ist. Die Definition der v-Werte stellt nun sicher, dass wir für jede Prefetchoperation mindestens F Zeiteinheiten reservieren. Diese Reservierung geschieht sogar unabhängig von Prefetchoperationen, die in zu den Bündeln $B_1, \ldots, B_{r-1}$ gehörenden Intervallen ablaufen und erst vor Anfragen enden, die zu Intervallen unseres Bündels gehören. Tatsächlich kann es also passieren, dass wir damit sogar zu viel Zeit für Prefetchoperationen reservieren. Wenn dies passiert, lassen wir die Platte für den Rest der Zeit einfach außer Betrieb. Somit ist der konstruierte Prefetching-und-Caching-Plan zulässig und wir fügen genau $\sum_{j=1}^{m} w(B_j)$ Zeiteinheiten Wartezeit ein. □

Wir haben also gezeigt, dass wir mit der Formulierung des binären Optimierungsproblems die anfallenden Kosten höchstens überschätzen. Nun zeigen wir eine obere Schranke für diese Überschätzung. Im Folgenden bezeichne OPT wiederum die Kosten einer optimalen Lösung der Instanz für IPC.

Lemma 9.2.5 *Das binäre Optimierungsproblem für D Platten hat eine ganzzahlige Lösung mit Kosten von höchstens* $(2D/z)\,\mathrm{OPT}$.

Beweis: Angenommen, uns liegt ein ganzzahliger Prefetching-und-Caching-Plan mit einer Gesamtwartezeit von OPT vor. Wir betrachten eine beliebige Teilmenge der oben erwähnten Partition von $\{1,\ldots,D\}$. Ohne Beschränkung der Allgemeinheit handele es sich dabei um die Menge $\{1,\ldots,z\}$. Wir betrachten nun lediglich die Wartezeiten, die zu Intervallen gehören, die mit den Platten $\{1,\ldots,z\}$ verknüpft sind.

Im Folgenden geben wir eine Belegung derjenigen Variablen an, die zu den Platten aus $\{1,\ldots,z\}$ gehören, sodass die Wartezeiten, die infolge nur derjenigen Prefetchoperationen entstehen, die sich auf die Platten $\{1,\ldots,z\}$ beziehen, höchstens zweimal im Wert der Zielfunktion des binären Optimierungsproblems gezählt werden. Indem wir dieses Vorgehen für alle D/z Teilmengen aus der Partition von $\{1,\ldots,D\}$ wiederholen, gelangen wir zu einer Belegung aller Variablen $x(B)$ für $B \in \mathcal{B}_z$. Weil die Zielfunktion des Optimierungsproblems bezüglich der Bündel aus $\mathcal{B}_z$ und damit bezüglich der D/z Teilmengen der Partition separierbar ist, zählen wir jede beliebige Wartezeit in einem optimalen Prefetching-und-Caching-Plan höchstens $2D/z$-mal.

Es bezeichne $\mathcal{I}' \subseteq \mathcal{I}$ die Menge aller Intervalle, die zu der Teilmenge $\{1,\ldots,z\}$ der Platten gehören und in denen nach dem vorgegebenen Plan Prefetchoperationen durchgeführt werden. Laut Lemma 9.2.1 dürfen wir annehmen, dass $\mathcal{I}'$ keine verschachtelten Paare enthält. Damit können wir die Intervalle aus $\mathcal{I}'$ gemäß aufsteigenden Startpunkten (bei Gleichheit gemäß aufsteigenden Endpunkten) sortieren. Sei $I_1,\ldots,I_m$ das Ergebnis dieser Sortierung. Wir erzeugen jetzt eine Partition von $\mathcal{I}'$, sodass die Teilmengen aus der Partition Bündel sind, mithilfe folgenden Greedy-Ansatzes.

$B := \emptyset$
Für $j = 1,\ldots,m$:
  Wenn $I_j \cup B$ ein Bündel ist, dann setze $B := B \cup \{I_j\}$,
  sonst gib B als Teilmenge der Partition aus und setze $B := \emptyset$.

Sei $B_1 \cup \cdots \cup B_\ell$, $\ell \le m$, die Partition von $\mathcal{I}'$, die wir mit dem beschriebenen Greedy-Algorithmus erhalten. Unsere Lösung des binären Optimierungsproblems setzt $x(B_j) := 1$ für $1 \le j \le \ell$. Die Variablen $f_{I,a}$ und $e_{I,a}$ werden anhand der Blöcke gesetzt, die der Plan in den Intervallen des Bündels einbringt bzw. entfernt. Anschließend wiederholen wir das Ganze für alle verbleibenden Teilmengen der oben erwähnten Partition von D. Alle nicht gesetzten Variablen sind 0.

Wir kehren jetzt zu der Teilmenge $\{1,\ldots,z\}$ der Partition von D zurück und bezeichnen mit OPT^* die Wartezeiten, die im optimalen Plan infolge von Prefetchoperationen, die Blöcke von Platten aus $\{1,\ldots,z\}$ einbringen, entstehen. Für die Bündel B_j,

$1 \le j \le \ell$, bezeichne $w'(B_j)$ die Summe der Wartezeiten, die im optimalen Plan zwischen dem Start der ersten Prefetchoperation in Intervallen aus B_j und der Beendigung der letzten Prefetchoperation in Intervallen aus B_j anfallen. Es ist $w(B_j) \le w'(B_j)$, da $w(B_j)$ laut Lemma 9.2.3 die minimale Wartezeit darstellt, die für die Prefetchoperationen in B_j erforderlich ist. Um zu zeigen, dass der Beitrag der Bündel B_j zum Wert der Zielfunktion höchstens $2\,\mathrm{OPT}^*$ beträgt, zeigen wir $\sum_{j=1}^{\ell} w'(B_j) \le 2\,\mathrm{OPT}^*$.

Wir betrachten eine beliebige, aber feste Zeiteinheit Wartezeit im optimalen Plan. Es seien $I_1, \ldots, I_\ell$, $\ell \le D$, diejenigen Intervalle im Plan, in denen diese Einheit Wartezeit auftritt. Man beachte, dass sich diese Intervalle auf verschiedene Platten beziehen müssen. Unser Greedy-Ansatz zur Bündelung von Intervallen kann höchstens ein Bündel fertig stellen, während er $I_1, \ldots, I_\ell$ als eine Teilfolge bearbeitet, da sich alle diese Intervalle auf verschiedene Platten beziehen und überlappend sind. Damit ist die Teilfolge eine Teilmenge der Vereinigung höchstens zweier Bündel und die betrachtete Wartezeit wird höchstens zweimal in der Summe $\sum_{j=1}^{\ell} w'(B_j)$ gezählt. Damit ist die Ungleichung gezeigt. □

Lemma 9.2.5 impliziert, dass es auch eine gebrochene Lösung mit Kosten von höchstens $(2D/z)\,\mathrm{OPT}$ geben muss, wenn wir das binäre Optimierungsproblem dahin gehend relaxieren, dass wir für alle Variablen Werte im Intervall $[0, 1]$ erlauben. Wir können dann eine Technik von Albers, Garg und Leonardi (2000), die im Wesentlichen eine Verallgemeinerung des am Ende von Abschnitt 9.1.1 umrissenen Ansatzes darstellt, anwenden. Damit wandeln wir eine gebrochene Lösung des nun linearen Optimierungsproblems mit Wert C in einen ganzzahligen Prefetching-und-Caching-Plan mit einer Gesamtwartezeit von höchstens C um, sofern wir $D - 1$ zusätzliche Zellen im Cache zur Verfügung haben. Also haben wir letztendlich einen Approximationsalgorithmus mit Güte $2D/z$ gefunden.

Theorem 9.2.6 *Es gibt ein Polynom $p(n)$, sodass für jedes $z \in \{1, \ldots, D\}$ mit $D/z \in \mathbb{N}$ ein Algorithmus mit Laufzeit $O(p(n) \cdot n \cdot (F + 1)^{2z} \cdot D \cdot z!)$ existiert, der für jede IPC-Instanz eine $2D/z$-Approximation berechnet, sofern $D - 1$ zusätzliche Zellen im Cache vorhanden sind.*

Für konstante D erhalten wir also eine 2-Approximation. Für $D = 2$ enthält Theorem 9.2.6 allerdings keine bessere Aussage, als bereits aus der Arbeit von Albers, Garg und Leonardi (2000) bekannt ist. Jedoch können wir unsere Analyse verbessern. Indem wir verschiedene Weisen betrachten, auf die Bündel gebildet werden können, erhalten wir eine Verbesserung um den Faktor $(z - 1)/z$. Im folgenden Lemma begnügen wir uns mit einer Darstellung für den Fall $D = 2$.

Lemma 9.2.7 *Im Fall $D = 2$ hat das binäre Optimierungsproblem eine ganzzahlige Lösung mit Kosten von höchstens* $(3/2)\,\mathrm{OPT}$.

Beweis: Angenommen, uns liegt ein ganzzahliger Prefetching-und-Caching-Plan mit einer Gesamtwartezeit von OPT vor. Im Folgenden werden wir zwei verschiedene ganzzahlige Lösungen für das binäre Optimierungsproblem erzeugen. Die Belegung der $f_{I,a}$- und $e_{I,a}$-Variablen ist in beiden Lösungen gleich und wird direkt vom Plan vorgegeben. Um die $x(B)$-Variablen zu belegen, nehmen wir die im Beweis von Lemma 9.2.5 benutzte Folge $I_1, \ldots, I_m$ von Intervallen als gegeben an. Lösung 1 belegt die Variablen $x(B)$, $B \in \mathcal{B}_2$, auf die Weise, die der Greedy-Ansatz im Beweis von Lemma 9.2.5 vorgibt. Lösung 2 ist in gewissem Sinne komplementär dazu. Seien $i_1 \leq i_2 \leq \cdots \leq i_\ell$ alle Indizes, sodass Lösung 1 die Intervalle I_{i_j} und I_{i_j+1} mit $1 \leq j \leq \ell$ in einem Bündel zusammenfasst. Ein Algorithmus zur Ermittlung von Lösung 2 erstellt eine Verfeinerung der folgenden Partition:

$$\{I_1, \ldots, I_{i_1}\}, \{I_{i_1+1}, \ldots, I_{i_2}\}, \ldots, \{I_{i_\ell+1}, \ldots, I_m\}.$$

Dazu geht der Algorithmus für jede Menge dieser Partition gemäß dem Greedy-Ansatz vor, der Lösung 1 erzeugt. Anhand der erhaltenen Partition belegt Lösung 2 die Variablen $x(B)$, $B \in \mathcal{B}_2$. In Abbildung 9.4 zeigen wir anhand einer Menge von Intervallen exemplarisch die Bündelungen aus den zwei verschiedenen Lösungen. Intervalle derselben Farbe (verschieden von Weiß) sind dabei als Bündel aufzufassen.

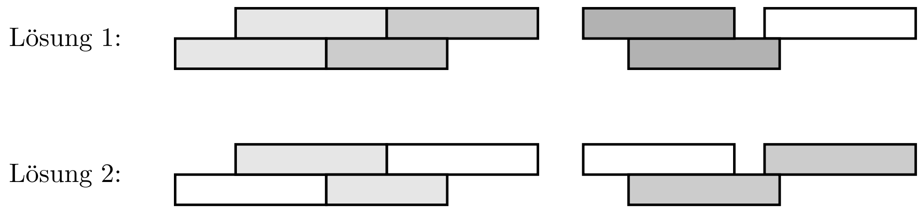

Abbildung 9.4: Zwei komplementäre Bündelungen

Seien s_1 und s_2 die Zielfunktionswerte von Lösung 1 bzw. Lösung 2. Wir überlegen uns, wie oft eine feste Zeiteinheit Wartezeit, die im optimalen Plan auftritt, in beiden Lösungen zusammen gezählt wird. Eine Einheit Wartezeit am Ende eines Intervalls wird insgesamt höchstens zweimal gezählt, wenn es für die andere Platte keine Prefetchoperation gibt, die mit dem Intervall überlappt. Also genügt es, dass wir uns auf den Fall zweier Intervalle I_j und I_{j+1}, $1 \leq j \leq m-1$, aus der obigen Folge konzentrieren, die einander überlappen und in denen jeweils Prefetchoperationen ausgeführt werden. Wenn Lösung 1 die Intervalle nicht bündeln kann, weil I_j mit I_{j-1} gebündelt ist, bündelt Lösung 2 die Intervalle I_j und I_{j+1}. Also wird die Einheit Wartezeit höchstens in einer Lösung doppelt gezählt und in der anderen Lösung nur einfach. Wir erhalten $s_1 + s_2 \leq 3\,\mathrm{OPT}$ und folgern, dass eine der beiden Lösungen die Wartezeit höchstens um einen Faktor von $3/2$ überschätzt. $\square$

Wir erhalten also im Fall $D = 2$ in Polynomialzeit eine Approximation mit Güte $3/2$, sofern eine Zelle Extraplatz im Cache verfügbar ist.

9.3 Fazit und Ausblick

In diesem Kapitel haben wir uns vornehmlich mit dem Entwurf und der anschließenden Analyse problemspezifischer Algorithmen für das kombinatorische Optimierungsproblem IPC beschäftigt. Im Falle $D = 1$ ist der Nachweis gelungen, dass das Problem mit rein kombinatorischen Mitteln in Polynomialzeit gelöst werden kann. Anhand einer Modellierung des Problems als Flussproblem haben wir unser Verständnis von der Struktur des Optimierungsproblems erweitert und eine mögliche Basis für verbesserte kombinatorische Polynomialzeitalgorithmen geschaffen.

Im Fall $D > 1$ konnten wir neue Approximationsalgorithmen entwerfen. Wir erhielten die ersten Polynomialzeit-Approximationsalgorithmen mit konstanter Güte 2 für den Fall, dass D konstant ist. Im Fall $D = 2$ konnte die Analyse verbessert werden, um sogar eine Approximationsgüte von $3/2$ zu zeigen. Alle Approximationsalgorithmen kommen mit einem Extraplatz der Größe $D - 1$ im Cache aus.

Wir haben auch gesehen, wie der Entwurf und die Analyse von Algorithmen ineinander greifen. Leider besitzt man nicht immer genügend Ressourcen, um konsequent nach spezialisierten Algorithmen zu forschen. Dann dürfen wir aber immer noch hoffen, mit randomisierten Suchheuristiken wie evolutionären Algorithmen gute Lösungen zu erzielen. In den ersten beiden Teilen dieser Dissertation sind wir auf dem Weg, solche Suchheuristiken zu verstehen, ein Stück weit vorangekommen.

Anhang und Verzeichnisse

Anhang

Einige mathematische Hilfsmittel

In diesem Anhang fassen wir grundlegende Beziehungen aus der Wahrscheinlichkeitstheorie, einige Abschätzungen und Identitäten sowie Sprechweisen zusammen, die bei verschiedenen Analysen zahlreiche Anwendungen finden. Die folgenden Ausführungen orientieren sich an Motwani und Raghavan (1995), Schickinger und Steger (2001), Feller (1968, 1971), Graham, Knuth und Patashnik (1994) sowie Jansen (2000).

Lemma A.1 *Für $x \in \mathbb{R}$ mit $x > 1$ ist*

$$\left(1 - \frac{1}{x}\right)^x \;\leq\; \frac{1}{\mathrm{e}} \;\leq\; \left(1 - \frac{1}{x}\right)^{x-1}.$$

Lemma A.2 *Für $x \in \mathbb{R}$ ist*

$$1 + x \;\leq\; \mathrm{e}^x.$$

Lemma A.3 (stirlingsche Formel) *Für $n \in \mathbb{N}$ ist*

$$\sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n \;\leq\; n! \;\leq\; \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n \mathrm{e}^{1/(12n)}.$$

Daraus folgt

$$n! \;\geq\; \left(\frac{n}{\mathrm{e}}\right)^n.$$

Lemma A.4 (Binomialkoeffizienten) *Für $n, k \in \mathbb{N}$ mit $k \leq n$ ist*

$$\binom{n}{k} \;\leq\; \frac{n^k}{k!} \quad \text{und} \quad \binom{n}{k} \;\geq\; \frac{n^k}{k^k}.$$

Lemma A.5 (binomischer Satz) *Für $n \in \mathbb{N}$ und $x, y \in \mathbb{R}$ ist*

$$(x + y)^n \;=\; \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}.$$

Lemma A.6 (Abschätzung der harmonischen Zahl) *Für $n \in \mathbb{N}$ ist*

$$\ln n \;\leq\; H_n \;\leq\; \ln n + 0{,}578 + O\left(\frac{1}{n}\right).$$

Lemma A.7 (großer Umordnungssatz) *Sei J abzählbar unendlich und sei $a_j \in \mathbb{R}$ für $j \in J$. Sei weiterhin $J = \cup_{i=0}^{\infty} I_i$ für paarweise disjunkte I_i, $i \in \mathbb{N}_0$. Wenn $\sum_{j\in J} a_j$ absolut konvergent ist, gilt*

$$\sum_{j\in J} a_j \;=\; \sum_{i=0}^{\infty} \sum_{j\in I_i} a_j.$$

Das nächste Lemma heißt auch *monotone convergence theorem*, siehe Kingman und Taylor (1966).

Lemma A.8 (Satz von Beppo Levi) *Seien $f_i\colon \mathbb{R} \to \mathbb{R}_0^+$, $i \in \mathbb{N}$, Lebesgue-integrierbare Funktionen und sei die Reihe $\sum_{i=1}^{\infty} \int_{\mathbb{R}} f_i(x)\,\mathrm{d}x$ konvergent. Dann konvergiert $\sum_{i=1}^{\infty} f_i(x)$ und es gilt*

$$\int_{\mathbb{R}} \left(\sum_{i=1}^{\infty} f_i(x) \right) \mathrm{d}x \;=\; \sum_{i=1}^{\infty} \int_{\mathbb{R}} f_i(x)\,\mathrm{d}x.$$

Wir listen jetzt einige Hilfsmittel aus dem Bereich der Wahrscheinlichkeitstheorie auf. Stillschweigend sei dabei jeweils ein diskreter Wahrscheinlichkeitsraum mit Ergebnismenge Ω und Wahrscheinlichkeitsfunktion Prob() vorausgesetzt.

Definition A.9 (Trägermenge) *Sei $X\colon \Omega \to \mathbb{R}$ eine Zufallsvariable für einen diskreten Wahrscheinlichkeitsraum. Die Elemente des Bildbereichs $\mathbb{R}$, die X mit positiver Wahrscheinlichkeit annimmt, bilden die* Trägermenge *von X.*

Lemma A.10 (Satz von der totalen Wahrscheinlichkeit) *Seien die Ereignisse $A_1, \ldots, A_n$ paarweise disjunkt, sei $\mathrm{Prob}(A_i) > 0$ für $1 \le i \le n$ und sei B ein Ereignis mit $B \subseteq A_1 \cup \cdots \cup A_n$. Dann ist*

$$\mathrm{Prob}(B) \;=\; \sum_{i=1}^{n} \mathrm{Prob}(B \mid A_i) \cdot \mathrm{Prob}(A_i).$$

Wenn darüber hinaus $\bigcup_{i=1}^{n} A_i = \Omega$ ist, gilt für jede Zufallsvariable $X\colon \Omega \to \mathbb{R}$ die Beziehung

$$\mathrm{E}(X) \;=\; \sum_{i=1}^{n} \mathrm{E}(X \mid A_i) \cdot \mathrm{Prob}(A_i),$$

sofern alle Erwartungswerte existieren und die Summe $\sum_{i=1}^{n} |\mathrm{E}(X \mid A_i)| \cdot \mathrm{Prob}(A_i)$ konvergiert.

Lemma A.11 (Markoffungleichung) *Sei X eine Zufallsvariable, die ausschließlich nicht negative Werte annehmen kann, mit Erwartungswert $\mathrm{E}(X)$. Für $t \in \mathbb{R}^+$ gilt dann*

$$\mathrm{Prob}(X \ge t \cdot \mathrm{E}(X)) \;\le\; \frac{1}{t}.$$

Lemma A.12 (Tschebyscheffungleichung) *Sei X eine Zufallsvariable mit Erwartungswert* $\mathrm{E}(X)$ *und Varianz* $\mathrm{Var}(X)$*. Für $t \in \mathbb{R}^+$ gilt dann*

$$\mathrm{Prob}\Big(|X - \mathrm{E}(X)| \geq t \cdot \sqrt{\mathrm{Var}(X)}\Big) \;\leq\; \frac{1}{t^2}.$$

Lemma A.13 (Chernoffungleichung) *Seien $X_1, \ldots, X_n$ unabhängige Zufallsvariablen mit* $\mathrm{Prob}(X_i = 1) = p_i$ *und* $\mathrm{Prob}(X_i = 0) = 1 - p_i$ *für $1 \leq i \leq n$. Sei $X := \sum_{i=1}^n X_i$ und $\mu := \mathrm{E}(X) = \sum_{i=1}^n p_i$. Dann gelten die folgenden Ungleichungen.*

$$\begin{array}{lcll}
\mathrm{Prob}(X \geq (1+\delta)\mu) & \leq & \left(\frac{\mathrm{e}^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu} & \textit{für } \delta \in \mathbb{R}^+, \\
\mathrm{Prob}(X \leq (1-\delta)\mu) & \leq & \left(\frac{\mathrm{e}^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu} & \textit{für } 0 < \delta < 1, \\
\mathrm{Prob}(X \geq (1+\delta)\mu) & \leq & \mathrm{e}^{-\mu\delta^2/3} & \textit{für } 0 < \delta \leq 1, \\
\mathrm{Prob}(X \leq (1-\delta)\mu) & \leq & \mathrm{e}^{-\mu\delta^2/2} & \textit{für } 0 < \delta \leq 1, \\
\mathrm{Prob}(X \geq t) & \leq & 2^{-t} & \textit{für } t \geq 2\mathrm{e}\mu.
\end{array}$$

Die folgende Variante des Theorems von de Moivre/Laplace findet sich in Raab und Steger (1998).

Lemma A.14 (de Moivre/Laplace) *Sei X eine zu den Parametern n und $1/2$ binomialverteilte Zufallsvariable. Für $\alpha(n)$ mit $\alpha(n) = \omega(1)$ und $\alpha(n) = o(n^{1/6})$ gilt*

$$\mathrm{Prob}(X \geq n/2 + \alpha(n)\sqrt{n}) \;=\; \frac{1 + o(1)}{\sqrt{2\pi} \cdot \alpha(n)} \cdot \mathrm{e}^{-\frac{(\alpha(n))^2}{2}}.$$

Das folgende Szenario heißt in der englischsprachigen Literatur gewöhnlich *coupon collector's problem*.

Lemma A.15 (Sammelbilderproblem) *Angenommen, es gibt n verschiedene Typen von Sammelbildern. Es bezeichne X die Zeit, die man benötigt, um von jedem Typ mindestens ein Sammelbild zu haben, wenn man zu jedem Zeitpunkt unabhängig ein Sammelbild erhält, das jedem Typ mit Wahrscheinlichkeit $1/n$ entspricht. Dann gilt*

$$\mathrm{E}(X) \;=\; nH_n$$

und

$$\lim_{n\to\infty} \mathrm{Prob}(X \leq n(\ln n - c)) \;=\; \mathrm{e}^{-\mathrm{e}^{c}}$$

für jede Konstante $c \in \mathbb{R}$.

Wir beschließen diesen Anhang mit gängigen Sprechweisen.

Definition A.16 (exponentiell groß/klein) *Sei $f\colon \mathbb{N} \to \mathbb{R}^+$. Wir nennen f* exponentiell groß in Bezug auf *n, wenn $f(n) = 2^{\Omega(n^{\varepsilon})}$ für eine Konstante $\varepsilon > 0$ gilt. Analog heißt f* exponentiell klein in Bezug auf *n, wenn $f(n) = 2^{-\Omega(n^{\varepsilon})}$ gilt.*

Symbolverzeichnis

$\mathbb{N}$	Menge der natürlichen Zahlen: $1, 2, \ldots$
$\mathbb{N}_0$	Menge der natürlichen Zahlen, vereinigt mit $\{0\}$
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{R}^+$	Menge der positiven reellen Zahlen
$\mathbb{R}_0^+$	Menge der nicht negativen reellen Zahlen
$\mathbb{Z}$	Menge der ganzen Zahlen
$[a, b]$	abgeschlossenes, reellwertiges Intervall mit Grenzen a, b
$]a, b[$	offenes, reellwertiges Intervall mit Grenzen a, b
e	eulersche Zahl (2,71...)
$H(x, y)$	$\sum_{i=1}^{n} \|x_i - y_i\|$ für $x, y \in \{0, 1\}^n$ *(Hammingabstand)*
H_n	$\sum_{i=1}^{n} 1/i$ *(n-te harmonische Zahl)*
$\text{argmax}\{f(x) \mid x \in X\}$	$\{x' \in X \mid f(x') \geq f(x) \text{ für alle } x \in X\}$
$\text{argmin}\{f(x) \mid x \in X\}$	$\{x' \in X \mid f(x') \leq f(x) \text{ für alle } x \in X\}$
$\exp(x)$	e^x
$\ln(x)$	Logarithmus zur Basis e
$\log(x)$	Logarithmus zur Basis 2
$\text{poly}(n)$	$n^{O(1)}$ *(polynomiell in n beschränkt)*
$\text{polylog}(n)$	$(\log n)^{O(1)}$ *(polylogarithmisch in n beschränkt)*
$O^*(f(n))$	$O(f(n) \, \text{polylog}(n))$
$\mathrm{E}(X)$	Erwartungswert der Zufallsvariablen X
$\text{Prob}(F)$	Wahrscheinlichkeit des Ereignisses F
$\text{Var}(X)$	Varianz der Zufallsvariablen X

Abbildungsverzeichnis

Literaturverzeichnis

Ackley, D. (1987): *A Connectionist Machine for Genetic Hillclimbers*, Kluwer

Ahuja, R., Magnanti, L. und Orlin, J. (1993): *Network Flows*, Prentice Hall

Albers, S. und Büttner, M. (2003a): Integrated prefetching and caching in single and parallel disk systems, in: *Proceedings of the 15th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '03)*, S. 109–117, ACM Press

Albers, S. und Büttner, M. (2003b): Integrated prefetching and caching with read and write requests, in: *Proceedings of the 8th International Workshop on Algorithms and Data Structures (WADS '03)*, Band 2748 von *Lecture Notes in Computer Science*, S. 162–173, Springer

Albers, S., Garg, N. und Leonardi, S. (2000): Minimizing stall time in single and parallel disk systems, *Journal of the ACM*, **47**(6), S. 969–986, vorläufige Version erschien in STOC '98

Albers, S. und Witt, C. (2001): Minimizing stall time in single and parallel disk systems using multicommodity network flows, in: *Proceedings of the 4th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX '01)*, Band 2129 von *Lecture Notes in Computer Science*, S. 12–23, Springer

Aldous, D. und Fill, J. (2004): *Reversible Markov Chains and Random Walks on Graphs*, Monographie in Vorbereitung, Entwurf war am 14. Dezember 2004 verfügbar unter `http://stat-www.berkeley.edu/users/aldous/RWG/book.html`

Ambühl, C. und Weber, B. (2004): Parallel prefetching and caching is hard, in: *Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS '04)*, Band 2996 von *Lecture Notes in Computer Science*, S. 211–221, Springer

Arora, S., Rabani, Y. und Vazirani, U. V. (1994): Simulating quadratic dynamical systems is PSPACE-complete, in: *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC '94)*, S. 459–467, ACM Press

Arya, S., Golin, M. J. und Mehlhorn, K. (1999): On the expected depth of random circuits, *Combinatorics, Probability and Computing*, **8**(3), S. 209–228

Bäck, T. (1996): *Evolutionary Algorithms in Theory and Practice*, Oxford University Press

Bäck, T., Fogel, D. B. und Michalewicz, Z. (Hrsg.) (1997): *Handbook of Evolutionary Computation*, Institute of Physics Publishing

Banzhaf, W., Nordin, P., Keller, R. E. und Francone, F. D. (1998): *Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and its Applications*, Morgan Kaufmann

Beier, R. und Vöcking, B. (2003): Random knapsack in expected polynomial time, in: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC '03)*, S. 232–241, ACM Press

Beier, R. und Vöcking, B. (2004): Typical properties of winners and losers in discrete optimization, in: *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC '04)*, S. 343–352, ACM Press

Belady, L. A. (1966): A study of replacement algorithms for a virtual storage-computer, *IBM Systems Journal*, **5**(2), S. 78–101

Beyer, H.-G. und Schwefel, H.-P. (2002): Evolution strategies – a comprehensive introduction, *Natural Computing*, **1**, S. 3–52

Beyer, H.-G., Schwefel, H.-P. und Wegener, I. (2002): How to analyse evolutionary algorithms, *Theoretical Computer Science*, **287**(1), S. 101–130

Borisovsky, P. A. und Eremeev, A. V. (2003): A study on performance of the (1+1)-evolutionary algorithm, in: *Proceedings of the 7th Foundations of Genetic Algorithms Conference (FOGA '02)*, S. 271–287, Morgan Kaufmann

Borodin, A. und El-Yaniv, R. (1998): *Online Computation and Competitive Analysis*, Cambridge University Press

Briest, P., Brockhoff, D., Degener, B., Englert, M., Gunia, C., Heering, O., Jansen, T., Leifhelm, M., Plociennik, K., Röglin, H., Schweer, A., Sudholt, D., Tannenbaum, S. und Wegener, I. (2004): Evolutionäre Algorithmen zwischen experimenteller und theoretischer Analyse, Endbericht der Projektgruppe 427, Fachbereich Informatik, Universität Dortmund, Bericht und zugehörige Software waren am 14. Dezember 2004 verfügbar unter `http://ls2-www.cs.uni-dortmund.de/lehre/pg427/`

Cao, P., Felten, E. W., Karlin, A. R. und Li, K. (1995): A study of integrated prefetching and caching strategies, in: *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '95)*, S. 188–197, ACM Press

Cao, P., Felten, E. W., Karlin, A. R. und Li, K. (1996): Implementation and performance of integrated application-controlled file caching, prefetching and disk scheduling, *ACM Transactions on Computer Systems*, **14**(4), S. 311–343

Chow, Y. S. und Teicher, H. (1997): *Probability Theory – Independence, Interchangeability, Martingales*, 3. Auflage, Springer

Coffman, E. G., Jr., Frederickson, G. N. und Lueker, G. S. (1984): A note on expected makespans for largest-first sequences of independent tasks on two processors, *Mathematics of Operations Research*, **9**(2), S. 260–266

Coffman, E. G., Jr. und Whitt, W. (1995): Recent asymptotic results in the probabilistic analysis of schedule makespans, in: Chrétienne, P., Coffman, E. G., Jr., Lenstra, J. K. und Liu, Z. (Hrsg.): *Scheduling Theory and its Applications*, S. 15–31, Wiley

Cormen, T. H., Leiserson, C. E., Rivest, R. L. und Stein, C. (2001): *Introduction to Algorithms*, 2. Auflage, MIT Press

David, H. A. und Nagaraja, H. N. (2003): *Order Statistics*, 3. Auflage, Wiley

Dietzfelbinger, M., Naudts, B., van Hoyweghen, C. und Wegener, I. (2003): The analysis of a recombinative hill-climber on H-IFF, *IEEE Transactions on Evolutionary Computation*, **7**(5), S. 417–423

Droste, S. (2000): *Zu Analyse und Entwurf evolutionärer Algorithmen*, Dissertation, Universität Dortmund

Droste, S. (2003): Analysis of the (1+1) EA for a dynamically bitwise changing OneMax, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03)*, Band 2723 von *Lecture Notes in Computer Science*, S. 909–921, Springer

Droste, S. (2004): Analysis of the (1+1) EA for a noisy OneMax, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 1088–1099, Springer

Droste, S., Jansen, T., Tinnefeld, K. und Wegener, I. (2003): A new framework for the valuation of algorithms for black-box optimization, in: *Proceedings of the 7th Foundations of Genetic Algorithms Conference (FOGA '02)*, S. 253–270, Morgan Kaufmann, erscheint in Theory of Computing Systems unter dem Titel *Upper and lower bounds for randomized search heuristics in black-box optimization*

Droste, S., Jansen, T. und Wegener, I. (2000): A natural and simple function which is hard for all evolutionary algorithms, in: *Proceedings of the 26th Annual Conference* *of the IEEE Industrial Electronics Society on Industrial Electronics, Control, and Instrumentation (IECON '00)*, S. 2704–2709, IEEE Press

Droste, S., Jansen, T. und Wegener, I. (2001): Dynamic parameter control in simple evolutionary algorithms, in: *Proceedings of the 6th Foundations of Genetic Algorithms Conference (FOGA '00)*, S. 275–294, Morgan Kaufmann

Droste, S., Jansen, T. und Wegener, I. (2002a): On the analysis of the (1+1) evolutionary algorithm, *Theoretical Computer Science*, **276**, S. 51–81

Droste, S., Jansen, T. und Wegener, I. (2002b): Optimization with randomized search heuristics – the (A)NFL theorem, realistic scenarios, and difficult functions, *Theoretical Computer Science*, **287**(1), S. 131–144

Feller, W. (1968): *An Introduction to Probability Theory and its Applications*, Band 1, 3. Auflage, Wiley

Feller, W. (1971): *An Introduction to Probability Theory and its Applications*, Band 2, 2. Auflage, Wiley

Fischer, S. (2004): A polynomial upper bound for a mutation-based algorithm on the two-dimensional ising model, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 1100–1112, Springer

Fischer, S. und Wegener, I. (2004): The Ising model on the ring: Mutation versus recombination, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 1113–1124, Springer

Fogel, D. B. (1995): *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press

Frenk, J. B. G. und Rinnooy Kan, A. H. G. (1986): The rate of convergence to optimality of the LPT rule, *Discrete Applied Mathematics*, **14**, S. 187–197

Garcia, N. und Palacios, J. L. (2002): On mixing times for stratified walks on the d-cube, *Random Structures and Algorithms*, **20**(4), S. 540–552

Garey, M. R. und Johnson, D. S. (1979): *Computers and Intractability – A Guide to the Theory of NP-Completeness*, Freeman

Garnier, J., Kallel, L. und Schoenauer, M. (1999): Rigorous hitting times for binary mutations, *Evolutionary Computation*, **7**(2), S. 173–203

Giel, O. (2003): Expected runtimes of a simple multi-objective evolutionary algorithm, in: *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, S. 1918–1925, IEEE Press

Giel, O. und Wegener, I. (2003): Evolutionary algorithms and the maximum matching problem, in: *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS '03)*, Band 2607 von *Lecture Notes in Computer Science*, S. 415–426, Springer

Goldberg, D. E. (1989): *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley

Graham, R. L. (1969): Bounds on multiprocessing timing anomalies, *SIAM Journal on Applied Mathematics*, **17**(2), S. 416–429

Graham, R. L., Knuth, D. E. und Patashnik, O. (1994): *Concrete Mathematics*, 2. Auflage, Addison-Wesley

Griffioen, J. und Appleton, R. (1994): Reducing file system latency using a predictive approach, in: *Proceedings of the USENIX Summer 1994 Technical Conference*, S. 197–207, ISBN 1-880446-62-6

Grinstead, C. M. und Snell, J. L. (1997): *Introduction to Probability*, 2. Auflage, American Mathematical Society

Hajek, B. (1982): Hitting-time and occupation-time bounds implied by drift analysis with applications, *Advances in Applied Probability*, **14**, S. 502–525

He, J. und Yao, X. (2001): Drift analysis and average time complexity of evolutionary algorithms, *Artificial Intelligence*, **127**(1), S. 57–85

He, J. und Yao, X. (2002): From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms, *IEEE Transactions on Evolutionary Computation*, **6**(5), S. 495–511

He, J. und Yao, X. (2003): An analysis of evolutionary algorithms for finding approximation solutions to hard optimisation problems, in: *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, S. 2004–2010, IEEE Press

Hochbaum, D. S. (Hrsg.) (1997): *Approximation Algorithms for NP-Hard Problems*, Wadsworth Publishing Company

Hoeffding, W. (1963): Probability inequalities for sums of bounded random variables, *American Statistical Association Journal*, **58**(301), S. 13–30

Holland, J. H. (1975): *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, MI, USA

Jägersküpper, J. (2003): Analysis of a simple evolutionary algorithm for minimization in Euclidean spaces, in: *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP '03)*, Band 2719 von *Lecture Notes in Computer Science*, S. 1068–1079, Springer

Jansen, T. (2000): *Theoretische Analyse evolutionärer Algorithmen unter dem Aspekt der Optimierung in diskreten Suchräumen*, Dissertation, Universität Dortmund

Jansen, T. und De Jong, K. (2002): An analysis of the role of offspring population size in EAs, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '02)*, S. 238–246, Morgan Kaufmann

Jansen, T. und Wegener, I. (2000): On the choice of the mutation probability for the (1+1) EA, in: *Proceedings of the 6th Conference on Parallel Problem Solving from Nature (PPSN '00)*, Band 1917 von *Lecture Notes in Computer Science*, S. 89–98, Springer

Jansen, T. und Wegener, I. (2001a): Evolutionary algorithms – how to cope with plateaus of constant fitness and when to reject strings of the same fitness, *IEEE Transactions on Evolutionary Computation*, **5**(6), S. 589–599

Jansen, T. und Wegener, I. (2001b): On the utility of populations, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '01)*, S. 1034–1041, Morgan Kaufmann

Jansen, T. und Wegener, I. (2001c): Real royal road functions – where crossover provably is essential, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '01)*, S. 375–382, Morgan Kaufmann

Jansen, T. und Wegener, I. (2002): The analysis of evolutionary algorithms – a proof that crossover really can help, *Algorithmica*, **34**, S. 47–66

Jansen, T. und Wiegand, R. P. (2003): Exploring the explorative advantage of the cooperative coevolutionary (1+1) EA, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03)*, Band 2723 von *Lecture Notes in Computer Science*, S. 310–321, Springer

Jansen, T. und Wiegand, R. P. (2005): The cooperative coevolutionary (1+1) EA, *Evolutionary Computation*, erscheint

Juels, A. und Wattenberg, M. (1996): Stochastic hillclimbing as a baseline method for evaluating genetic algorithms, in: *Proceedings of the 8th Conference on Advances in Neural Information Processing Systems (NIPS '95)*, S. 430–436, MIT Press

Kamath, A., Palmon, O. und Plotkin, S. (1995): Fast approximation algorithm for minimum cost multicommodity flow, in: *Proceedings of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '95)*, S. 493–501, ACM Press

Kimbrel, T. (1997): *Parallel Prefetching and Caching*, Dissertation, University of Washington

Kimbrel, T. und Karlin, R. (2000): Near-optimal parallel prefetching and caching, *SIAM Journal on Computing*, **29**(4), S. 1051–1082, vorläufige Version erschien in FOCS '96

Kingman, J. F. C. und Taylor, S. J. (1966): *Introduction to Measure and Probability*, Cambridge University Press

Kirkpatrick, S., Gelatt, C. D. und Vecchi, M. P. (1983): Optimization by simulated annealing, *Science*, **220**, S. 671–680

Laumanns, M., Thiele, L. und Zitzler, E. (2004): Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions, *IEEE Transactions on Evolutionary Computation*, **8**(2), S. 170–182

Lindvall, T. (2002): *Lectures on the Coupling Method*, Dover Publications

Lueker, G. S. (1998): Exponentially small bounds on the expected optimum of the partition and subset sum problems, *Random Structures and Algorithms*, **12**(1), S. 51–62

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. und Teller, E. (1953): Equation of state calculation by fast computing machines, *Journal of Chemical Physics*, **21**, S. 1087–1092

Mitchell, M., Forrest, S. und Holland, J. H. (1992): The royal road for genetic algorithms: Fitness landscapes and GA performance, in: *Proceedings of the 1st European Conference on Artificial Life (ECAL '91)*, S. 245–254, MIT Press

Mitchell, M., Holland, J. H. und Forrest, S. (1994): When will a genetic algorithm outperform hill climbing, in: *Proceedings of the 7th Conference on Advances in Neural Information Processing Systems (NIPS '93)*, S. 51–58, Morgan Kaufmann

Motwani, R. (1994): Average-case analysis of algorithms for matchings and related problems, *Journal of the ACM*, **41**(6), S. 1329–1356

Motwani, R. und Raghavan, P. (1995): *Randomized Algorithms*, Cambridge University Press

Mühlenbein, H. (1992): How genetic algorithms really work: I. Mutation and hillclimbing, in: *Proceedings of the 2nd Conference on Parallel Problem Solving from Nature (PPSN '92)*, S. 15–26, Elsevier

Neumann, F. (2004): Expected runtimes of evolutionary algorithms for the Eulerian cycle problem, in: *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, Band 1, S. 904–910, IEEE Press

Neumann, F. und Wegener, I. (2004): Randomized local search, evolutionary algorithms, and the minimum spanning tree problem, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 713–724, Springer

Ollivier, Y. (2003): Rate of convergence of crossover operators, *Random Structures and Algorithms*, **23**(1), S. 58–72

Papadimitriou, C. und Steiglitz, K. (1982): *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall

Patterson, R. H., Gibson, G. A., Ginting, E., Stodolsky, D. und Zelenka, J. (1995): Informed prefetching and caching, in: *Proceedings of the 15th ACM Symposium on Operating System Principles (SOSP '95)*, S. 79–95, ACM Press

Pittel, B. (1994): Note on the heights of random recursive trees and random m-ary search trees, *Random Structures and Algorithms*, **5**(2), S. 337–348

Raab, M. und Steger, A. (1998): Balls into bins – a simple and tight analysis, in: *Proceedings of the 2nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM '98)*, Band 1518 von *Lecture Notes in Computer Science*, S. 159–170, Springer

Rabani, Y., Rabinovich, Y. und Sinclair, A. (1998): A computational view of population genetics, *Random Structures and Algorithms*, **12**(4), S. 313–334

Rechenberg, I. (1973): *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog, Stuttgart

Rechenberg, I. (1994): *Evolutionsstrategie '94*, Frommann-Holzboog, Stuttgart

Reeves, C. R. und Rowe, J. E. (2003): *Genetic Algorithms – Principles and Perspectives*, Kluwer

Rudolph, G. (1997): *Convergence Properties of Evolutionary Algorithms*, Dissertation, Universität Dortmund, Verlag Dr. Kovač, Hamburg

Rudolph, G. (2000): Takeover times and probabilities of non-generational selection rules, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00)*, S. 903–910, Morgan Kaufmann

Ruml, W., Ngo, J. T., Marks, J. und Shieber, S. M. (1996): Easily searched encodings for number partitioning, *Journal of Optimization Theory and Applications*, **89**(2), S. 251–291

Sasaki, G. H. und Hajek, B. (1988): The time complexity of maximum matching by simulated annealing, *Journal of the ACM*, **35**, S. 387–403

Scharnow, J., Tinnefeld, K. und Wegener, I. (2002): Fitness landscapes based on sorting and shortest paths problems, in: *Proceedings of the 7th Conference on Parallel Problem Solving from Nature (PPSN '02)*, Band 2439 von *Lecture Notes in Computer Science*, S. 54–63, Springer, erweiterte Version erscheint in Journal of Mathematical Modeling and Algorithms (Special Issue on Evolutionary Computation in Combinatorial Optimization)

Schickinger, T. und Steger, A. (2001): *Diskrete Strukturen*, Band 2, Springer

Schrijver, A. (1986): *Theory of Linear and Integer Programming*, Wiley

Schwefel, H.-P. (1995): *Evolution and Optimum Seeking*, Wiley

Smythe, R. T. und Mahmoud, H. M. (1995): A survey of recursive trees, *Theory of Probability and Mathematical Statistics*, **51**, S. 1–27

Spielman, D. A. und Teng, S.-H. (2001): Smoothed analysis: Why the simplex algorithm usually takes polynomial time, in: *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC '01)*, S. 296–305, ACM Press

Steger, A. (2001): *Diskrete Strukturen*, Band 1, Springer

Storch, T. (2004): On the choice of the population size, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 748–760, Springer

Storch, T. und Wegener, I. (2003): Real royal road functions for constant population size, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03)*, Band 2724 von *Lecture Notes in Computer Science*, S. 1406–1417, Springer

van Nimwegen, E. und Crutchfield, J. P. (2001): Optimizing epochal evolutionary search: population-size dependent theory, *Machine Learning*, **45**(1), S. 77–114

Vose, M. D. (1999): *The Simple Genetic Algorithm: Foundations and Theory*, MIT Press

Wegener, I. (2001): Theoretical aspects of evolutionary algorithms (invited paper), in: *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP '01)*, Band 2076 von *Lecture Notes in Computer Science*, S. 64–78, Springer

Wegener, I. (2003): *Komplexitätstheorie – Grenzen der Effizienz von Algorithmen*, Springer

Wegener, I. und Witt, C. (2003): On the optimization of monotone polynomials by the (1+1) EA and randomized local search, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03)*, Band 2723 von *Lecture Notes in Computer Science*, S. 622–633, Springer, ausgezeichnet mit einem *best paper award*

Wegener, I. und Witt, C. (2005a): On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions, *Journal of Discrete Algorithms*, erscheint

Wegener, I. und Witt, C. (2005b): On the optimization of monotone polynomials by simple randomized search heuristics, *Combinatorics, Probability and Computing*, im Druck

Witt, C. (2003): Population size vs. runtime of a simple EA, in: *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, Band 3, S. 1996–2003, IEEE Press

Witt, C. (2004): An analysis of the (μ+1) EA on simple pseudo-boolean functions, in: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '04)*, Band 3102 von *Lecture Notes in Computer Science*, S. 761–773, Springer, ausgezeichnet mit einem *best paper award*

Witt, C. (2005a): Population size vs. runtime of a simple evolutionary algorithm, *Theoretical Computer Science*, überarbeitete Version eingereicht

Witt, C. (2005b): Worst-case and average-case approximations by simple randomized search heuristics, in: *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS '05)*, Band 3404 von *Lecture Notes in Computer Science*, Springer, erscheint

Wolpert, D. H. und McReady, W. G. (1997): No free lunch theorems for optimization, *IEEE Transactions on Evolutionary Computation*, **1**(1), S. 67–82