Diplomarbeit

Die Analyse des (1+1)-EA für fast lineare und spezielle quadratische Funktionen

Carsten Witt

Diplomarbeit
am Fachbereich Informatik
der Universität Dortmund

28. April 2000

Gutachter:

Prof. Dr. Ingo Wegener
PD Dr. Paul Fischer

Ich bedanke mich bei Ingo Wegener
für die Betreuung dieser Arbeit und
bei allen anderen Personen, die mich
währenddessen unterstützt haben.

Carsten Witt, Baroper Str. 335, D-44227 Dortmund, E-Mail: c.witt@web.de
Matrikelnummer 0059177, Fachbereich Informatik, Universität Dortmund

Inhaltsverzeichnis

Kapitel 1

Einleitung

In nahezu allen Bereichen der Naturwissenschaften, Ingenieurswissenschaften, aber auch z. B. Geisteswissenschaften und nicht zuletzt im täglichen Leben begegnen uns Optimierungsaufgaben. Bereits die Ermittlung einer kürzesten „Rundreise“, beispielsweise innerhalb des Bezirks eines Postboten bzw. einer Postbotin, verkörpert ein Problem, das in allgemeiner Form als schwierig (NP-hart) bekannt ist.

Um Zugang zu einer systematischen Lösung von Optimierungsproblemen zu gewinnen, fußen Optimierungsverfahren meist auf Modellen, die in der Sprache der Mathematik formuliert sind. Wir konzentrieren uns in dieser Arbeit auf folgendes formalisiertes Optimierungsproblem: Gesucht ist das Maximum einer von n freien Variablen abhängigen Funktion $f : M \to \mathbb{R}$, d. h. $f(x_1, \ldots, x_n) \to \max$, wobei $M \subseteq \mathbb{R}^n$ typischerweise mit der Menge der n-stelligen reellwertigen Vektoren $\mathbb{R}^n$ identisch ist oder die Menge der booleschen Vektoren $\{0, 1\}^n$ umfasst.[1] Dabei geben wir keine weiteren Restriktionen für die Werte der Variablen vor. Da die Minimierung einer Funktion f gleichbedeutend mit der Maximierung von $-f$ ist, setzen wir übrigens immer Maximierungsaufgaben voraus und gebrauchen die Begriffe „maximieren“ und „optimieren“ o. B. d. A. synonym.

Die Ermittlung eines Maximums einer wie zuvor beschriebenen Funktion f kann enormen Aufwand beinhalten und ist nur selten trivial. Insbesondere Definitionsbereiche höherer Dimensionen besitzen eine viel zu große Mächtigkeit, als dass sie systematisch an allen Punkten (sofern der Definitionsbereich überhaupt endlich ist) durchsucht werden könnten. Bereits die Mächtigkeit der Menge $\{0, 1\}^n$, also des n-fachen kartesischen Produktes der booleschen Menge $\{0, 1\}$, wächst mit 2^n exponentiell in n. Aufgrund dieses so genannten „Fluchs der Dimensionen“ (*curse of dimensionality*) stellen viele Optimierungsverfahren nur Heuristiken dar und können nicht immer eine befriedigende Lösung liefern.

[1] Diese Beschreibung kann natürlich noch verallgemeinert werden und birgt in dieser Form den Kritikpunkt, dass f nicht unbedingt Maxima besitzen muss. Allerdings brauchen wir im Rahmen dieser Arbeit nicht zwischen Suprema und Maxima von Funktionen zu unterscheiden, da wir ausschließlich endliche Definitionsbereiche betrachten.

1.1 Evolutionäre Algorithmen

In den sechziger Jahren kamen mehrere Personen unabhängig voneinander auf den Gedanken, das Element des Zufalls in Optimierungsheuristiken einzubringen. Im Zuge dessen entstand eine Unterklasse der randomisierten Algorithmen, nämlich die Klasse der *evolutionären Algorithmen.* Ihnen liegt der Gedanke zugrunde, das Element „Zufall“ mit der Absicht einzusetzen, natürliche Evolution nachzubilden.

Ohne Konzentration auf spezielle Anwendungen ist das Feld der evolutionären Algorithmen aber ungeheuer vielschichtig und verzweigt geworden. Es weist Verknüpfungen mit den Disziplinen der Biologie, des Operations Research, der numerischen Optimierung, der künstlichen Intelligenz und der Informatik im Allgemeinen auf. Unter evolutionäre Algorithmen können wir mindestens genetische Algorithmen [Gol89], die genetische Programmierung [BNKF98], die evolutionäre Programmierung [Fog95] und Evolutionsstrategien [Sch95] fassen. Einen ausführlichen Überblick über die meisten dieser Ansätze bietet die Dissertation von Bäck [Bäc94]. Dementsprechend vielfältig sind auch die Einsatzgebiete für evolutionäre Algorithmen. Im Bereich der der Informatik verwandten Fachgebiete werden evolutionäre Algorithmen in der künstlichen Intelligenz, Mustererkennung, Robotik, Simulation etc. genutzt. Eine umfassende Auflistung aller Einsatzgebiete ist vermutlich unmöglich; für einen tiefer gehenden, doch sicherlich nicht (mehr) vollständigen Einblick in die Vielfalt der Anwendungsgebiete sei auf [BHS92] verwiesen.

Wir wollen gleich konkrete, für uns interessante Instanzen evolutionärer Algorithmen beschreiben. Zuvor zählen wir jedoch einige öfter wiederkehrende Prinzipien evolutionärer Algorithmen kurz und informal auf (ohne zu postulieren, dass ein evolutionärer Algorithmus diese alle beinhalten muss oder nur darauf beschränkt zu sein braucht).

- Objekte der Suche sind so genannte *Individuen*, die man zu *Populationen* zusammenfasst.
- Einem Individuum ist ein *Fitnesswert* zugeordnet.
- Individuen können (zufällige) Änderungen durch *Mutation* erfahren.
- Durch *Selektion* werden Individuen (i. A. mit hoher Fitness) ausgewählt bzw. Individuen (geringer Fitness) entfernt.
- Infolge von *Rekombination (Crossover)*, dem Vereinigen oder Mischen von Individuen, entstehen möglicherweise neue Individuen.
- Man zählt die zu diskreten Zeitpunkten vorliegenden Populationen in *Generationen.*

1.2 Evolutionsstrategien

Die Erfindung der Evolutionsstrategien schreibt man gemeinhin Bienert, Rechenberg und Schwefel (vgl. [Rec94, Sch95]) zu. Sie erprobten in den späten sechziger Jahren den Einsatz

zufälliger Änderungen („Mutationen“) bei Optimierungen im Bereich der Strömungstechnik u. Ä. Rechenberg motiviert diese Vorgehensweise mit der These, dass die natürliche Evolution ein besonders effektives Optimierungsverfahren sei.

Evolutionsstrategien sollen eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ mit einem n-dimensionalen reellwertigen Definitionsbereich (hier der Einfachheit halber mit $\mathbb{R}^n$ identifiziert) maximieren, greifen also unsere Problemstellung auf. Die von natürlicher Evolution inspirierte Suche „lebt“ dabei hauptsächlich von Mutationen, die einen zufälligen Vektor $z \in \mathbb{R}^n$ zum Ausgangspunkt haben. Eine Mutation erzeugt aus einem Individuum $x \in \mathbb{R}^n$ ein $x' \in \mathbb{R}^n$ durch Addition des zufälligen Vektors z, d. h. $x' = x + z$. Der Vektor z ist im Allgemeinen normalverteilt und hat den Erwartungswert $\vec{0}$ (n-stelliger Nullvektor). Zudem ändern sich die Varianzen von z im Laufe des Algorithmus (Selbstanpassung bzw. *self adaption* genannt).

Schwefel hat eine gängige Nomenklatur zur Unterteilung der Selektionsmechanismen in Evolutionsstrategien entwickelt, die Unterscheidung von *(μ,λ)-Strategien* und *(μ+λ)-Strategien*. Der Parameter μ steht für die typischerweise konstante Populationsgröße, während λ die Zahl der durch Mutation (und Rekombination, hier außer Acht gelassen) erzeugten Nachfahren bezeichnet. In (μ+λ)-Strategien werden μ Nachfahren mit den besten Fitnesswerten in die nachfolgende Generation aufgenommen, wobei nur $\lambda > \mu$ sinnvoll ist. Bei (μ+λ)-Strategien hingegen werden μ Nachfahren mit den besten Fitnesswerten aus der Menge der μ Eltern und λ Nachfahren übernommen. Die Selektion erfolgt in Evolutionsstrategien (beispielsweise im Gegensatz zu genetischen Algorithmen) rein deterministisch[2].

1.3 Der (1+1)-EA

In dieser Arbeit konzentrieren wir uns auf eine konkrete (1+1)-Evolutionsstrategie, deren Suchraum der boolesche „Hyperwürfel“ $\{0,1\}^n$ ist. Diese spezielle Strategie heißt *(1+1)-EA*.

Definition 1.1 ((1+1)-EA) *Der (1+1)-EA erhält eine Funktion $f : \{0,1\}^n \to \mathbb{R}$ als Eingabe. Er erzeugt in einer Endlosschleife probabilistisch Vektoren $x \in \{0,1\}^n$ nach folgendem Ablauf:*

1. *Initialisiere zufällig gleichverteilt einen Bitstring $x \in \{0,1\}^n$, indem jedes Bit von x unabhängig mit Wahrscheinlichkeit $1/2$ auf 1 gesetzt wird.*
2. *Wiederhole:*
   - *Generiere x^* aus x, indem jedes Bit von x unabhängig mit Wahrscheinlichkeit $1/n$ negiert wird* (Mutation).
   - *Ersetze x durch x^* genau dann, wenn $f(x^*) \geq f(x)$* (Selektion).

[2]Eventuell bis auf Situationen, in denen ein „Unentschieden“ wegen verschiedener Möglichkeiten, μ beste Individuen zu wählen, eintritt.

Da diese Strategie die Optimierung *pseudoboolescher* Funktionen $f : \{0,1\}^n \to \mathbb{R}$ zum Ziel hat, können wir den (1+1)-EA als Evolutionsstrategie mit Elementen aus dem Bereich genetischer Algorithmen ansehen. Während die Repräsentation des Suchraums und der Mutationsoperator typisch für genetische Algorithmen sind, ist der deterministische Selektionsmechanismus aus der Vorgehensweise von Evolutionsstrategien übernommen.
Auf die Frage nach dem Sinn dieses einfachen evolutionären Algorithmus führen wir zunächst die immense Bedeutung der Optimierung pseudoboolescher Funktionen in der Informatik an. Wie eingangs angedeutet, lassen sich die Optimierungsvarianten vieler NP-vollständiger Probleme als Suche nach einem Optimum einer pseudobooleschen Funktion fassen; für eine Einführung in das Gebiet der so genannten kombinatorischen Optimierung und ihrer Anwendungen siehe z. B. [PS82].
Beim Aufbau einer theoretischen Grundlage für evolutionäre Algorithmen stellt es einen pragmatischen und vernünftigen Ansatz dar, zunächst möglichst einfache Instanzen zu analysieren. Mithin setzt die theoretische Erforschung von (μ+λ)-Strategien bei der Wahl $\mu = \lambda = 1$ an. Wir hoffen, dabei gewonnene Resultate verallgemeinern und beispielsweise auf größere Populationen übertragen zu können. Außerdem vermuten wir, dass ähnliche Effizienzsteigerungen wie durch größere Populationen auch durch mehrfache (evtl. parallele) Läufe des (1+1)-EA zu erreichen sind. Da Rekombination bei einer Populationsgröße von 1 sinnlos ist, ersparen wir uns überdies die Analyse der Auswirkungen von „Crossover“. (Diese erscheint äußerst komplex und konnte bisher kaum angegangen werden, vgl. beispielsweise [JW99] und [RRS98].)
Wir haben nun einen ersten Einblick in die Strukturen evolutionärer Algorithmen gewonnen und die Bedeutung des (1+1)-EA, der in dieser Arbeit betrachteten Instanz, herausgestellt. Nun wollen wir grundlegende theoretische Resultate anreißen.

1.4 Theorie

Gegenüber den mannigfachen Anwendungsgebieten evolutionärer Algorithmen erscheint deren theoretische Fundierung noch unterentwickelt. Erst in den letzten Jahren hat sich das Interesse an ihrer theoretischen Analyse verstärkt.
In Hinblick auf praktische Anwendungen interessieren neben Analysen der Einflüsse verschiedener Parameter evolutionärer Algorithmen (Wahl des Selektionsmechanismus etc.) besonders die Effizienz eines konkreten evolutionären Algorithmus auf einer konkreten Eingabe. Mögliche theoretische Maße dafür sind die *Konvergenzgeschwindigkeit* und *Erfolgswahrscheinlichkeit* abhängig von einem Punkt x des Suchbereiches. Bereits in den siebziger Jahren errechnete Rechenberg den erwarteten Fortschritt von einem Punkt $x \in \mathbb{R}^n$ für eine konkrete Funktion, das *Kugelmodell* $f(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i^2$, und eine (1+1)-Evolutionsstrategie. Unter dem Fortschritt (svw. Konvergenzgeschwindigkeit) ist die Verringerung der Distanz zum globalen Optimum der zu optimierenden Funktion zu verstehen. Der Konvergenzgeschwindigkeit steht die Erfolgswahrscheinlichkeit teilweise konfliktär[3] ge-

[3]Konflikt zwischen der Erforschung des Suchraumes gegenüber Ausnutzung von Fortschritten zum Optimum, *exploration vs. exploitation*, siehe [Bäc94]

genüber. Letztere gibt für einen Punkt x an, mit welcher Wahrscheinlichkeit eine Mutation erfolgt, die zu einem Punkt mit besserem Funktionswert führt. Für weitere Hinweise zu dieser Thematik empfehlen wir die Arbeiten von Beyer [Bey93] und Rudolph [Rud97].
Anders als bei dieser lokalen Betrachtungsweise, den Fortschritt oder die Erfolgswahrscheinlichkeit von einem festen Punkt des Suchraumes aus zu analysieren, beschäftigen wir uns hauptsächlich mit Fragen nach erwarteten Laufzeiten, wenn die Suche bei einem zufällig gewählten Punkt beginnt. Dieses soll nun für den (1+1)-EA näher erläutert werden.

1.4.1 Theoretische Analyse des (1+1)-EA

Bevor wir die bisherige „Geschichte“ der Analyse des (1+1)-EA ansatzweise beschreiben, nennen wir einige Voraussetzungen, die bei der theoretischen Analyse des (1+1)-EA immer im Hintergrund stehen werden. Zum einen haben wir den (1+1)-EA in Abschnitt 1.3 als Endlosschleife beschrieben, ohne ein Abbruchkriterium anzugeben. Der (1+1)-EA wird nämlich als *Black-Box-Optimierungsverfahren* angesehen, d. h. man macht keine weiteren Annahmen über die zu optimierende Funktion $f : \{0,1\}^n \to \mathbb{R}$. Bei einer Black-Box-Optimierung können wir über die unbekannte Funktion f ausschließlich Informationen gewinnen, indem wir sie auswerten. Der Wunsch, ein Optimierungsverfahren möglichst universell zu gestalten, motiviert diese Sichtweise; für nähere Hinweise zur Black-Box-Optimierung siehe z. B. [DJW99b].
Des Weiteren gehen wir bei der theoretischen Analyse des (1+1)-EA meist von einem Modell einer endlichen *Markoffkette* mit dem Zustandsraum $\{0,1\}^n$ aus. Diese Markoffkette hat mindestens einen absorbierenden Zustand, nämlich ein $x_{\max} \in \{0,1\}^n$, sodass $f(x_{\max})$ maximal ist. Unter Missachtung des fehlenden Stoppkriteriums wollen wir die erwartete Zeit bis zum Erreichen eines absorbierenden Zustandes abschätzen (*hitting time*) und nennen dies *erwartete Laufzeit*. Konkrete formale Beschreibungen liefern wir nach. Für den Augenblick genügen diese Hinweise, um einige bekannte Resultate zu präsentieren.
Erste Schritte zu einer theoretischen Analyse des (1+1)-EA auf *linearen Funktionen* (formale Definition in Kapitel 2.1) wurden Anfang der neunziger Jahre getan. Anhand der Funktion OneMax : $\{0,1\}^n \to \mathbb{N}_0$ mit $\textsc{OneMax}(x_1, \ldots, x_n) = x_1 + \cdots + x_n$ schufen Bäck und Mühlenbein [Bäc92, Müh92] die Grundlagen für die Analyse der zugrunde liegenden Markoffkette und für approximative Aussagen über die erwartete Laufzeit des (1+1)-EA auf der Funktion OneMax.
Während Bäck die erwartete Laufzeit wegen zu komplexer Formeln lediglich mit numerischen Berechnungen abschätzte, vereinfachte Mühlenbein die zugrunde liegende Markoffkette und präsentierte erste approximative Aussagen über die erwartete Laufzeit des (1+1)-EA auf der Funktion OneMax. Rudolph arbeitet diese Resultate in [Rud97] weiter aus und zeigt sehr ausführlich eine obere Schranke von $O(n \ln n)$ für die erwartete Laufzeit des (1+1)-EA auf OneMax.
Im Jahre 1998 erbrachten Droste, Jansen und Wegener wesentliche Fortschritte bei der theoretischen Analyse des (1+1)-EA. Sie zeigen in [DJW98a] eine obere Schranke von $O(n \ln n)$ für die erwartete Laufzeit auf allen linearen Funktionen und geben auf der anderen Seite Funktionen an, auf denen der (1+1)-EA versagt, d. h. exponentielle erwartete

Laufzeiten benötigt. Diese Diplomarbeit baut im Kern unmittelbar auf der genannten Arbeit von Droste, Jansen und Wegener auf.

1.5 Methoden und Konventionen dieser Arbeit

Um die in dieser Arbeit übliche Methodik der Analyse des (1+1)-EA leichter begreifen zu können, vertiefen wir den Blick auf die Markoffkette, mit der sich der (1+1)-EA modellieren lässt und erläutern einige Begriffe, die häufiger wiederkehren. Die Markoffkette des (1+1)-EA wird einerseits durch einen endlichen Zustandsraum $\Omega = \{0,1\}^n$ beschrieben, deren Elemente wir mit *Bitstrings* (Vektoren) $x = (x_1, \ldots, x_n)$ identifizieren. Diese Markoffkette ist aufgrund der statischen Mutations- und Selektionsmechanismen *homogen*, d. h. die Übergangswahrscheinlichkeiten zwischen zwei Zuständen hängen nicht von der (diskreten) Zeit ab. Die Zeit zählen wir zu Beginn jedes Durchlaufs der Endlosschleife in Schritt 2 des (1+1)-EA um eins weiter. Der erste Durchlauf (*Schritt*) beginnt zum Zeitpunkt 1, wenn der *Startvektor* (auch: *initialisierter Bitstring*) x^s erstmals mutiert wird.
Anders als bei einer bloßen Irrfahrt auf dem Zustandsraum hängen die Übergangswahrscheinlichkeiten der Markoffkette von der Funktion f ab, die Eingabe für den (1+1)-EA ist. So beträgt die Übergangswahrscheinlichkeit von allen $x \in \{0,1\}^n$ zu allen $x' \in \{0,1\}^n$ mit $f(x') < f(x)$ stets null. (Optimierungsverfahren mit solchen Eigenschaften heißen *Hillclimber*). Wenn wir die Übergangswahrscheinlichkeiten für ein fixiertes Paar $(x, x') \in \{0,1\}^n \times \{0,1\}^n$ betrachten, bezeichnen wir oft x als *aktuellen Bitstring* und verbinden damit das Ereignis, dass sich der (1+1)-EA im Zustand x befindet. Falls $f(x') \geq f(x)$, nimmt die Übergangswahrscheinlichkeit von x zu x' einen von null verschiedenen Wert an; die Mutation von x zu x' wird *akzeptiert*. Derartige Wahrscheinlichkeiten werden uns immer wieder begegnen, sodass wir einige Standardtechniken zu deren Abschätzung einbringen. Wir werden diese Abschätzungen bei konkreten Problemstellungen in dieser Arbeit erläutern; die wesentlichen Techniken sind nochmals in Anhang A.3 wiedergegeben. Für den Moment gewinnen wir ein Gefühl für Mutationswahrscheinlichkeiten – ob die Mutationen akzeptiert werden, lassen wir jetzt außer Acht –, indem wir festhalten, dass im Erwartungswert pro Schritt genau eine Position mutiert (*flippt*), vgl. Lemma A.5, und dass die Anzahl mutierter Bits pro Schritt für große n annähernd poissonverteilt ist (Lemma A.6).
Zwar ist es theoretisch möglich, die erwartete Zeit bis zum Erreichen eines absorbierenden Zustandes der Markoffkette mit analytischen Methoden (Matrizeninversionen) exakt zu errechnen, vgl. [Rud97, Theorem 3.6], doch in der Regel erweisen sich die dazu notwendigen Berechnungen aufgrund der Mächtigkeit des Zustandsraumes als kaum durchführbar. Wir nehmen daher oftmals Vergröberungen vor und ersetzen die eigentliche Markoffkette durch eine vereinfachte Markoffkette mit deutlich weniger Zuständen und nachweisbar nicht geringerer Absorbtionszeit.[4] Eine solche Methodik steht immer dann im Hintergrund, wenn wir so genannte *Fortschrittsmaße* aufstellen. Für eine explizite Erläuterung, wie Markoffketten sinnvoll manipuliert und damit vergröbert werden können, eignet sich Rudolphs Darstellung der oberen Schranke für OneMax [Rud97, Beispiel 5.2].

[4] Beyer [Bey95] nennt dies den *mesoskopischen Ansatz.*

Kapitel 2

Grundlagen

Im Rahmen einer theoretischen Analyse des (1+1)-EA versucht man Ergebnisse über dessen erwartete Laufzeit, d. h. die erwartete Zahl von Schritten bis zum Erreichen eines absorbierenden Zustandes, zu ermitteln. Dies ist vor dem Hintergrund einer Black-Box-Optimierung ein vernünftiges Maß für die Effizienz des (1+1)-EA, da wir die erwartete Zahl von Funktionsauswertungen in Schritt 2 des (1+1)-EA zählen.
Wie bei der Analyse von Algorithmen üblich, interessieren hauptsächlich asymptotische Aussagen. Wir suchen dabei für wachsendes $n \in \mathbb{N}$ geltende Schranken der erwarteten Laufzeit auf *Funktionenfolgen* $f_n : \{0,1\}^n \to \mathbb{R}$, wobei meist einfach von (pseudoboole-schen) *Funktionen* oder *Fitnessfunktionen* gesprochen wird und der Index n fortgelassen wird. Die Funktion OneMax aus Abschnitt 1.4.1 beispielsweise können wir auf natürliche Weise mit wachsendem n skalieren.
Noch allgemeiner möchten wir Funktionen(folgen) in *Klassen* von Funktionen einteilen. Eine wünschenswerte Klassenbildung fasst Funktionen mit möglichst ähnlichem erwarteten Laufzeitverhalten des (1+1)-EA zusammen. Die einfachste, aber völlig uninteressante Klasse von Funktionen $f : \{0,1\}^n \to \mathbb{R}$ ist die Klasse der *konstanten* Funktionen der Gestalt $f(x) = a$ für alle $x \in \{0,1\}^n$ und ein konstantes $a \in \mathbb{R}$. Da der Funktionswert solcher Funktionen überhaupt nicht vom aktuellen Bitstring des (1+1)-EA abhängt, ist jeder Bitstring optimal, und die erwartete Laufzeit ist durch $O(1)$ nach oben beschränkt.
Die in unseren Augen einfachste vernünftige Klasse von Funktionen ist die Klasse der *linearen Funktionen.*

2.1 Lineare Funktionen

Definition 2.1 (Lineare Funktionen) *Eine Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$, dargestellt durch $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{R}$ und $\tau \in \mathbb{R}$, heißt linear. Die Koeffizienten w_i werden auch als* Gewichte *bezeichnet.*

Lineare Funktionen heißen auch bitweise separierbar. Wir können den maximalen Wert solcher Funktionen trivialerweise „von Hand“ ermitteln, indem wir ein optimales x für $i \in \{1, \ldots, n\}$ auf folgende Weise erzeugen: Setze $x_i = 1$, falls $w_i \geq 0$, und $x_i = 0$ sonst.

Der (1+1)-EA als Black-Box-Optimierungsverfahren arbeitet ebenfalls effizient auf linearen Funktionen, wenn auch nicht in erwarteten $\Theta(n)$ Schritten. Für die erwartete Laufzeit gilt unabhängig von den Gewichten der linearen Funktion stets die obere und untere Schranke $\Theta(n \ln n)$. Für einen Beweis siehe [DJW98a] bzw. [DJW98b].
In dieser Arbeit wollen wir Fitnessfunktionen ohne Beschränkung der Allgemeinheit in gewissen „Normalformen“ vorliegen haben, um die Analyse des Verhaltens des (1+1)-EA auf solchen Funktionen etwas zu vereinfachen. Bei linearen Funktionen, deren Gewichte ursprünglich rational sind, nimmt sich diese Normalform wie folgt aus.

Lemma 2.1 *Gegeben sei eine lineare Fitnessfunktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{Q}$ und $\tau \in \mathbb{R}$, für die die erwartete Laufzeit des (1+1)-EA bestimmt werden soll. Es existiert eine lineare Funktion $f^*(x) = \sum_{i=1}^{n} w_i^* x_i$ mit $w_i^* \in \mathbb{N}$ und $w_1^* \geq \cdots \geq w_n^*$, durch die wir f bei der Analyse des Laufzeitverhaltens des (1+1)-EA o. B. d. A. ersetzen können.*

Beweis: Wir transformieren die Funktion f in eine Funktion $f^*(x) = \sum_{i=1}^{n} w_i^* x_i$ mit $w_i^* \in \mathbb{N}$, die für den (1+1)-EA im Hinblick auf seine erwartete Laufzeit äquivalent ist.
Zunächst wird o. B. d. A. angenommen, dass $\forall i \in \{1, \ldots, n\} : w_i \neq 0$ gilt; sonst wäre f von den x_i mit $w_i = 0$ unabhängig. Stelle w_i als $w_i = p_i/q_i$ mit $p_i, q_i \in \mathbb{Z}$ dar. Seien nun für $s := |\prod_{i=1}^{n} q_i|$ die Gewichte w_i' durch $w_i' := s w_i$ und die Funktion f' durch $f'(x) := \sum_{i=1}^{n} w_i' x_i + s\tau$ definiert. Also gilt $w_i' \in \mathbb{Z}$. Die Funktion f' ist für den (1+1)-EA zu f äquivalent, wenn die Ordnung der Funktionswerte aller Bitstrings erhalten bleibt. Zu zeigen bleibt also

$$\forall x, y \in \{0,1\}^n : f(x) \geq f(y) \Leftrightarrow f'(x) \geq f'(y).$$

Dazu seien x, y beliebig gewählt. Da $s > 0$, gilt

$$\begin{aligned} f(x) \geq f(y) &\Leftrightarrow \sum_{i=1}^{n} w_i x_i + \tau \geq \sum_{i=1}^{n} w_i y_i + \tau \Leftrightarrow s\Big(\sum_{i=1}^{n} w_i x_i + \tau\Big) \geq s\Big(\sum_{i=1}^{n} w_i y_i + \tau\Big) \\ &\Leftrightarrow \sum_{i=1}^{n} w_i' x_i + s\tau \geq \sum_{i=1}^{n} w_i' y_i + s\tau \Leftrightarrow f'(x) \geq f'(y). \end{aligned}$$

Aufgrund von

$$\sum_{\{i|w_i'>0\}} w_i' x_i + \sum_{\{i|w_i'<0\}} w_i' x_i + s\tau = \sum_{\{i|w_i'>0\}} w_i' x_i + \sum_{\{i|w_i'<0\}} (-w_i'(1 - x_i)) + \sum_{\{i|w_i'<0\}} w_i' + s\tau$$

lassen sich die ganzzahligen Gewichte in natürliche Gewichte transformieren. Dabei ist zu beachten, dass in den Variablen mit Indizes, bei denen die Gewichte negativ sind, eine Komplementbildung vorzunehmen ist, d. h. die Rollen von Nullen und Einsen vertauscht werden. Nach Ermittlung eines optimalen Bitstrings sind somit diese Variablen zu komplementieren, um einen für f optimalen Bitstring zu erhalten. Die erwartete Laufzeit des (1+1)-EA kann aber infolge der umgekehrten Bedeutung von $x_i = 1$ und $x_i = 0$ für die Indizes i mit $w_i' < 0$ nicht beeinflusst werden, da wir den Bitstring gleichverteilt initialisieren und Mutationen selbst auch lediglich Komplementbildungen vornehmen. Mit anderen

Worten können wir in der zugrunde liegenden Markoffkette mit dem Zustandsraum $\{0,1\}^n$ o. B. d. A. Paare von Zuständen durch Vertauschen von Nullen und Einsen umbenennen.
Zur Sortierung der Gewichte existiert darüber hinaus eine Permutation $\pi : \{1,\ldots,n\} \to \{1,\ldots,n\}$ mit der Eigenschaft, dass für alle $i,j \in \{1,\ldots,n\}$ gilt: $i < j \Rightarrow w_{\pi(i)} \geq w_{\pi(j)}$. (Ggf. müssen wir die Komponenten eines Optimums $x_{\max}$ gemäß π^{-1} umordnen, um einen optimalen Bitstring der ursprünglichen Funktion zu erhalten.) Da konstante Summanden keinen Einfluss auf die Ordnung der Funktionswerte ausüben, schließen für $i \in \{1,\ldots,n\}$ die Definitionen $w_i^* := w'_{\pi(i)}$, falls $w'_{\pi(i)} > 0$, und $w_i^* := -w'_{\pi(i)}$, falls $w'_{\pi(i)} < 0$, sowie $f^*(x) := \sum_{i=1}^n w_i^* x_i$ den Beweis ab. □

Sind die Gewichte der Fitnessfunktion irrational, lässt sich nicht wie im vorigen Beweis ein Hauptnenner finden. Jedoch gilt auch:

Lemma 2.2 *Sei $f : \{0,1\}^n \to \mathbb{R}$ eine lineare Fitnessfunktion $f(x) = \sum_{i=1}^n w_i x_i + \tau$ mit reellwertigen Koeffizienten $w_i \in \mathbb{R}$. Dann existiert eine lineare Fitnessfunktion $f^*(x) = \sum_{i=1}^n v_i^* x_i + \tau$ mit rationalen Koeffizienten $v_i^* \in \mathbb{Q}$, sodass gilt:*

$$\forall x,y \in \{0,1\}^n : f(x) \geq f(y) \Leftrightarrow f^*(x) \geq f^*(y). \qquad (*)$$

Beweis: Zunächst gelte wiederum, dass $\forall i \in \{1,\ldots,n\} : w_i \neq 0$. Wir konstruieren ein homogenes lineares Gleichungssystem $Av = \vec{0}$ aus maximal $\binom{2^n}{2} \leq 2^{2n-1}$ Gleichungen mit einer Matrix $A = (a_{ij}), a_{ij} \in \{-1,0,1\}$, und dem Vektor von Unbekannten $v = (v_1,\ldots,v_n,s_1,\ldots,s_r)$. Dabei sollen die Variablen s_l, $l \in \{1,\ldots,r\}$, mit

$$r := \#\{(x,y) \in \{0,1\}^n \times \{0,1\}^n | f(x) > f(y)\}$$

als „Schlupfvariablen“ fungieren, und daher wird $s_l > 0$ gefordert.
Die Matrix A enthält zunächst für alle ungeordneten Paare $x,y \in \{0,1\}^n$ mit $f(x) = f(y)$, d. h. $\sum_{\{j|x_j=1\}} w_j = \sum_{\{j|y_j=1\}} w_j$, in den ersten Zeilen der Matrix solche Einträge, dass für j mit $x_j = 1$ und $y_j = 0$ in Spalte j der Koeffizient 1, für $y_j = 1$ und $x_j = 0$ der Koeffizient -1 und sonst 0 eingetragen wird. Die verbleibenden Zeilen sollen in analoger Konstruktion für alle $x,y \in \{0,1\}^n$ mit der Eigenschaft $f(x) > f(y)$ die Gleichungen $\sum_{\{j|x_j=1\}} w_j - \sum_{\{j|y_j=1\}} w_j - s_l = 0$ ausdrücken. Offenbar erfüllt für eine Lösung $v^* = (v_1^*,\ldots,v_n^*,s_1^*,\ldots,s_r^*)$ des linearen Gleichungssystems genau dann die lineare Funktion $f^*(x) := \sum_{i=1}^n v_i^* x_i + \tau$ die Bedingung $(*)$, wenn $s_l^* > 0$ für alle $l \in \{1,\ldots,r\}$ gilt. Weiterhin gilt aufgrund der Annahme $w_i \neq 0$ für $i \in \{1,\ldots,n\}$, dass eine Lösung $v' = (w_1,\ldots,w_n,s'_1,\ldots,s'_r)$ mit reellen Komponenten und $s'_l > 0$ für $l \in \{1,\ldots,r\}$ existiert, die nicht die triviale Lösung ist. Da die Dimension des Lösungsraums größer als 0 ist, lässt die aus A mit elementaren Zeilenumformungen in Zeilenstufenform überführte Matrix $A' = (a'_{ij})$ mindestens eine Unbekannte frei wählbar. Es ist mindestens s_r frei wählbar, da sonst $s'_r = 0$ im Widerspruch zur existenten Lösung folgen würde.
Wird mit einem $s_r^* \in \mathbb{Q}$ eine Lösung $v^* = (v_1^*,\ldots,v_n^*,s_1^*,\ldots,s_r^*)$ per Rücksubstitution bestimmt – etwaige weitere frei wählbare Variablen werden ebenfalls auf rationale

Werte gesetzt –, so sind alle Komponenten im Folgenden ebenfalls rational, da A' auch nur rationale Koeffizienten enthält. Zu zeigen bleibt, dass es eine solche Wahl von s_r^* gibt, dass $s_l^* > 0$ für alle $l \in \{1, \ldots, r\}$ gilt. Dazu betrachten wir den Zeilenbereich $i_{\min} := \min\{i | \forall j \in \{1, \ldots, n\} : a'_{ij} = 0\}$ bis $i_{\max} := \max\{i | \exists j \in \{n+1, \ldots, n+r\} : a'_{ij} \neq 0\}$ von A', der diejenigen Einträge enthält, welche in die Rücksubstitution von $s_1, \ldots, s_r$ eingehen. Weiterhin definieren wir $a'_{\max} := \max\left\{|a'_{ij}|, i \in \{i_{\min}, i_{\max}\}, j \in \{n+1, \ldots, n+r\}\right\}$ und analog $a'_{\min}$ als betragsmäßig größte und kleinste Koeffizienten in dieser relevanten Submatrix und wählen ein $\epsilon > 0$. Wir setzen die frei wählbare Variable s_r^* auf einen rationalen Wert q mit $|s'_r - q| < \epsilon$. Dann existieren für die Variablen s_l, $l \in \{1, \ldots, r-1\}$, rationale Belegungen s_l^* mit $|s_l^* - s'_l| < (r-1)!(a'_{\max}/a'_{\min})^{r-1}\epsilon$. Diese Aussage ist für $r = 1$ trivialerweise richtig. Für $r > 1$ ist die Aussage nach Induktionsvoraussetzung für $l \in \{2, \ldots, r\}$ erfüllt, indem die Submatrix auf den Spaltenbereich von $n+2, \ldots, r$ eingeschränkt wird, während $a'_{\max}/a'_{\min}$ allenfalls kleiner wird. Für s_1^* gilt dann, sofern es nicht im trivialen Fall freier Wählbarkeit einen rationalen Wert entsprechend der Induktionsbehauptung erhält, aufgrund der Rücksubstitution

$$\begin{aligned} |s_1^* - s'_1| &= \left| \frac{-\sum_{l=2}^r a'_{i_{\min},n+l} s_l^*}{a'_{i_{\min},n+1}} - \frac{-\sum_{l=2}^r a'_{i_{\min},n+l} s'_l}{a'_{i_{\min},n+1}} \right| \leq \frac{\sum_{l=2}^r a'_{\max} |s_l^* - s'_l|}{a'_{\min}} \\ &\leq \frac{a'_{\max}}{a'_{\min}}(r-1)(r-2)! \left(\frac{a'_{\max}}{a'_{\min}}\right)^{r-2} \epsilon \leq (r-1)! \left(\frac{a'_{\max}}{a'_{\min}}\right)^{r-1} \epsilon. \end{aligned}$$

Mit einer Wahl $\epsilon < \min\{s'_l\}/((r-1)!(a'_{\max}/a'_{\min})^{r-1})$ werden alle s_l^* rational und positiv. Weil abzählbar unendlich viele rationale Zahlen zur Wahl der freien Parameter existieren, gibt es sogar abzählbar unendlich viele $f^*(x)$ mit den geforderten Eigenschaften. □

Damit ist die intuitiv klare Aussage, dass irrationale Gewichte durch rationale approximiert werden können, ohne die Ordnung der Funktionswerte zu beeinflussen, gezeigt. Die Kombination der vorangehenden Lemmata 2.1 und 2.2 liefert:

Korollar 2.3 *Bei einer linearen Funktion $f(x) = \sum_{i=1}^n w_i x_i + \tau$ nehmen wir o. B. d. A. an, dass $\tau = 0, w_i \in \mathbb{N}$ für alle $i \in \{1, \ldots, n\}$ und $w_1 \geq \cdots \geq w_n$ gelten.*

2.2 Quadratische Funktionen

Die zuvor behandelten linearen Funktionen heißen auch Funktionen vom Grad 1. Auch wenn vielfältige Kriterien zur Klassifikation von Fitnessfunktionen im Kontext evolutionärer Algorithmen herangezogen werden können (vgl. z. B. [Jan99]), unterscheiden wir in dieser Arbeit Funktionen hauptsächlich anhand ihres Grades. Um den Grad einer Fitnessfunktion näher fassen zu können, schließen wir einige Definitionen an (vgl. [DJW98a]).
Eine Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$ kann auf eindeutige Weise als Polynom mit Koeffizienten $c_f(I) \in \mathbb{R}$ geschrieben werden:

$$f(x_1, \ldots, x_n) = \sum_{I \subseteq \{1,\ldots,n\}} c_f(I) \cdot \prod_{i \in I} x_i.$$

Definition 2.2 (Grad einer Fitnessfunktion) *Der Grad von f wird definiert durch*

$$\deg(f) = \max\{i \in \{0,\ldots,n\} | \exists I \text{ mit } |I| = i \text{ und } c_f(I) \neq 0\}.$$

Definition 2.3 (Quadratische Funktionen) *Eine Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$ vom Grad 2, dargestellt durch $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} x_i x_j + \tau$, mit $w_i, w_{ij} \in \mathbb{R}$ und $\tau \in \mathbb{R}$ heißt quadratisch.*

Da das Verhalten des (1+1)-EA auf linearen Funktionen mittlerweile vollständig erforscht ist, erscheint es natürlich, sich nun mit der Analyse quadratischer Funktionen zu beschäftigen. Dies ist in der Tat das zentrale Thema dieser Arbeit. Wir versuchen quadratische Funktionen zu strukturieren und übertragen zunächst Ideen aus den Lemmata 2.2 und 2.1 auf quadratische Funktionen.

Lemma 2.4 *Bei einer quadratischen Fitnessfunktion können wir o. B. d. A. annehmen, dass f die Gestalt $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij} x_i x_j$ hat mit $w_i \in \mathbb{Z}, w_{ij} \in \mathbb{Z}$ für $1 \leq i < j \leq n$ und dass $w_1 \geq \cdots \geq w_n$ gilt.*

Beweis: Mit einem zu Lemma 2.2 analogen Beweis – es sind lediglich noch Spalten für die Gewichte w_{ij} in die Matrix A aufzunehmen – folgt die Rationalität und wie in Lemma 2.1 die Ganzzahligkeit aller Gewichte. Gleichsam zu Lemma 2.1 lassen sich $w_1,\ldots,w_n$ wie beschrieben ordnen sowie kann $\tau = 0$ angenommen werden. Weiterhin können die Gewichte w_{ij} und w_{ji} sowie w_i und w_{ii} infolge der Kommutativität der Multiplikation bzw. $x^2 = x$ für $x \in \{0,1\}$ zusammengefasst werden. □

Im Gegensatz zu linearen Funktionen kann aber für die Koeffizienten w_i quadratischer Funktionen nicht $w_i \neq 0$ gefordert werden, wie die Funktion $f(x_1,x_2) = x_1x_2$ zeigt. Darüber hinaus verbietet sich die Annahme $\forall i,j \in \{1,\ldots,n\} : w_i \geq 0, w_{ij} \geq 0$, da $f(x_1,x_2) = x_1 - x_2 + x_1x_2$ ein Gegenbeispiel ist. (Vertauschen von x_2 und $1-x_2$ wie in Lemma 2.1 erzwingt ein negatives Gewicht im letzten Summanden.) Zuletzt zeigt die Funktion $f(x_1,x_2) = -x_1 - x_2 + 2x_1x_2$, dass $w_i \in \mathbb{N}_0$ gleichermaßen nicht angenommen werden darf, denn der Übergang $f(x_1,x_2) \mapsto f(1-x_1,1-x_2)$ liefert $f(1-x_1,1-x_2) = f(x_1,x_2)$, d. h. die Komplementbildung von x_1 und x_2 ist hier ohne Bedeutung.

2.2.1 Eine untere Schranke für quadratische Funktionen

Bei linearen Funktionen ist einfach nachzuweisen (vgl. [DJW98a]), dass die erwartete Laufzeit des (1+1)-EA zu deren Optimierung durch $\Omega(n \ln n)$ nach unten beschränkt ist. Diese Schranke können wir unter einer realistischen Annahme auch bei quadratischen Funktionen auf nahezu identische Weise zeigen.

Satz 2.5 *Gegeben sei eine quadratische Funktion $f : \{0,1\}^n \to \mathbb{R}$. Nimmt f nur für ein $x^* \in \{0,1\}^n$ einen maximalen Wert an, benötigt der (1+1)-EA zur Optimierung von f erwartete Zeit $\Omega(n \ln n)$.*

Beweis: In der Initialisierungsphase (Schritt 1) des evolutionären Algorithmus wird ein zufälliger Vektor $x^s \in \{0,1\}^n$ erzeugt, anhand dessen wir Zufallsvariablen $X_1, \ldots, X_n$ definieren. Es ist $X_i = 1$, $i \in \{1, \ldots, n\}$, wenn $x_i^s = x_i^*$, und 0 sonst, d. h. X_i fungiert als Indikatorvariable und zeigt an, ob x_i^s „richtig" initialisiert wurde. Offenbar ist $\text{Prob}(X_i = 1) = 1/2$ sowie $E[X_i] = 1/2$. Auf $X := X_1 + \cdots + X_n$ mit $E[X] = n/2$ wenden wir die Chernoffschranke (vgl. Lemma A.1) an und wählen z. B. den Parameter $\delta = 0{,}2$. Dann ist

$$\text{Prob}(X > 0{,}6n) < 0{,}991^n = 2^{-\Omega(n)} \text{ sowie } \text{Prob}(X < 0{,}4n) < 0{,}991^n = 2^{-\Omega(n)},$$

also $\text{Prob}(0{,}4n \leq X \leq 0{,}6n) = 1 - 2^{-\Omega(n)}$. Die erwartete Laufzeit des (1+1)-EA muss daher von mindestens derselben Größenordnung wie die erwartete Laufzeit unter der Bedingung $0{,}4n \leq X \leq 0{,}6n$ sein. Für $c := 0{,}4$ ermitteln wir mit einem dem „Coupons Collector Problem" aus [MR95] ähnlichen Vorgehen die erwartete Zeit, bis cn Bits je mindestens einmal geflippt wurden, während jedes einzelne Bit mit Wahrscheinlichkeit $1/n$ geflippt wird. Da dieses zum Erreichen eines optimalen Bitstrings unerlässlich ist, zumal nicht jede dieser Mutationen akzeptiert werden muss, erhalten wir eine untere Schranke für die erwartete Laufzeit des (1+1)-EA. Die Zufallsvariable T soll nun die Zeit, bis jedes der cn Bits mindestens einmal geflippt wurde, angeben und hat den Erwartungswert

$$E[T] = \sum_{t=1}^{\infty} t \cdot \text{Prob}(T = t) = \sum_{t=1}^{\infty} \text{Prob}(T \geq t), \text{ da } T \text{ stets positiv ist.}$$

Die Wahrscheinlichkeit $\text{Prob}(T \geq t)$ lässt sich als Wahrscheinlichkeit, dass in $t-1$ Schritten mindestens eins der cn Bits niemals flippt, interpretieren. Das Gegenereignis dazu besteht darin, dass in $t-1$ Schritten alle cn Bits mindestens einmal flippen. Für die Wahrscheinlichkeit, in *einem* Schritt *ein* bestimmtes Bit mindestens einmal zu flippen, können wir wiederum die Gegenwahrscheinlichkeit, in einem Schritt dieses Bit nicht zu flippen, benutzen, nämlich $(1 - 1/n)$. Dann ist $(1 - 1/n)^{t-1}$ die Wahrscheinlichkeit, dieses Bit in $t-1$ (unabhängigen) Schritten nie zu flippen, $1 - (1 - 1/n)^{t-1}$ die Wahrscheinlichkeit, in $t-1$ Schritten dieses Bit mindestens einmal zu flippen, $(1-(1-1/n)^{t-1}))^{cn}$ die Wahrscheinlichkeit, alle cn Bits in $t-1$ Schritten mindestens einmal zu flippen und $1-(1-(1-1/n)^{t-1}))^{cn}$ schließlich die anfangs erwähnte Wahrscheinlichkeit $\text{Prob}(T \geq t)$. Insgesamt gilt dann

$$\begin{aligned}
E[T] &\geq \sum_{i=1}^{\infty} \left(1 - \left(1 - \left(1 - \frac{1}{n}\right)^{t-1}\right)^{cn}\right) \\
&\geq (n-1)(\ln n) \left(1 - \left(1 - \left(1 - \frac{1}{n}\right)^{(n-1)\ln n}\right)^{cn}\right), \; 1 - (1-1/n)^{t-1} \text{ mon. wachsend} \\
&\geq (n-1)(\ln n) \left(1 - \left(1 - e^{-\ln n}\right)^{cn}\right) = (n-1)(\ln n) \left(1 - \left(1 - \frac{1}{n}\right)^{cn}\right) \\
&\geq (n-1)(\ln n) \left(1 - e^{-c}\right) = \Omega(n \ln n),
\end{aligned}$$

wobei wir die Summe in der zweiten Zeile bei $(n-1)\ln n$ abgebrochen haben sowie in der dritten Zeile die Abschätzung $(1 - 1/n)^{n-1} \geq e^{-1}$, $n \geq 2$, und in der vierten die Abschätzung $(1 - 1/n)^n \leq e^{-1}$ herangezogen haben. □

Bemerkung 2.1 Die untere Schranke aus Satz 2.5 gilt nicht nur für quadratische Funktionen mit nur einem globalen Maximum, sondern für alle Funktionen $f : \{0,1\}^n \to \mathbb{R}$, die nur auf einem Element ihrer Definitionsmenge den maximalen Funktionswert annehmen.

Wir haben also eine untere Schranke, die für alle Elemente der Klasse der quadratischen Funktionen gilt. Dieser Schranke gegenüber steht die quadratische Funktion DISTANCE aus [DJW98a], denn die erwartete Laufzeit des (1+1)-EA auf DISTANCE ist exponentiell, nämlich durch $\Theta(n^n)$ nach unten (und oben) beschränkt. Dies zeigt, dass der (1+1)-EA auf quadratischen Funktionen keineswegs immer effizient arbeitet. Weiter ziehen wir die Schlussfolgerung, dass die Klasse sämtlicher quadratischer Funktionen für allgemeine Aussagen über das Verhalten des (1+1)-EA „zu groß“ ist. Wie wir sehen werden, gibt es im Gegensatz zu DISTANCE auch „gutartige“ quadratische Funktionen, auf denen der (1+1)-EA nur eine polynomielle erwartete Laufzeit benötigt.
Wir haben damit das Thema dieser Arbeit motiviert. Unsere Aufgabe ist es, die Klasse quadratischer Funktionen in Unterklassen mit möglichst ähnlichen Eigenschaften in Bezug auf die erwartete Laufzeit des (1+1)-EA zu unterteilen. Im Folgenden stellen wir einige nahe liegende Einschränkungen vor. Der näheren Analyse solcher Funktionen und weiteren Unterteilungen ist dann je ein eigenes Kapitel oder ein eigener Abschnitt dieser Arbeit gewidmet.

2.2.2 Quadrate linearer Funktionen

Die Funktion DISTANCE aus [DJW98a] entsteht, indem wir eine lineare Funktion (nämlich im Wesentlichen ONEMAX) quadrieren. Damit haben wir eine erste Unterklasse der quadratischen Funktionen gefunden.

Definition 2.4 (Quadrat einer linearen Funktion) *Gegeben sei eine lineare Fitnessfunktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{R}$ und $\tau \in \mathbb{R}$. Mit $f^2(x) := (f(x))^2$ wird das Quadrat der linearen Funktion bezeichnet.*

Offenbar ist f^2 eine quadratische Fitnessfunktion im Sinne von Definition 2.3. Wir erhalten ebenfalls quadratische Funktionen aus linearen Funktionen, indem wir Quadrate linearer Funktionen negieren, also aus einer linearen Funktion f die Funktion $-f^2$ bilden. Jedoch wird es sich später (in Abschnitt 3.4) zeigen, dass diese nicht allzu interessant sind. Für den Moment stellen wir Aussagen zu den Negationen von Quadraten linearer Funktionen zurück und beschränken uns auf die interessantere Klasse der Quadrate linearer Funktionen.
Es ist nicht offensichtlich, dass die bei der Analyse linearer Funktionen getroffenen Annahmen (natürliche und monoton fallende Werte der Koeffizienten sowie $\tau = 0$) auch bei deren Quadraten für das Verhalten des (1+1)-EA keine Einschränkungen darstellen. Allerdings lassen sich die zunächst erwähnten Forderungen realisieren.

Lemma 2.6 *Bei der Betrachtung des Quadrats f^2 einer linearen Fitnessfunktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ dürfen wir o. B. d. A. annehmen, dass $w_i \in \mathbb{N}$ für alle $i \in \{1, \ldots, n\}$ und $w_1 \geq \cdots \geq w_n$ gelten.*

Beweis: Zum Beweis, dass wir rationale Gewichte annehmen können, ziehen wir den Beweis zu Lemma 2.2 heran und nehmen einige Modifikationen vor. Es gilt offenbar $f^2(x) \geq f^2(y) \Leftrightarrow |f(x)| \geq |f(y)|$ für alle $x, y \in \{0,1\}^n$, wobei f hier die lineare Funktion bezeichnet, deren Quadrat untersucht wird. Die Zeilen der Matrix stellen nun für für alle ungeordneten Paare $x, y \in \{0,1\}^n$, für die $|f(x)| = |f(y)|$ gilt, die Gleichungen $|\sum_{\{j|x_j=1\}} w_j + \tau| - |\sum_{\{j|y_j=1\}} w_j + \tau| = 0$ sowie für alle geordneten Paare $x, y \in \{0,1\}^n$ mit $|f(x)| > |f(y)|$ die Gleichungen $|\sum_{\{j|x_j=1\}} w_j + \tau| - |\sum_{\{j|y_j=1\}} w_j + \tau| - s_l = 0$ dar. Die Einträge der Matrix $A = (a_{ij})$ berücksichtigen die Bildung des Betrages und werden ggf. negiert, d. h. $a_{ij} \in \{-2,-1,0,1,2\}$. Zudem ist eine Spalte für die Unbekannte v_τ aufzunehmen, die die Addition eines konstanten Summanden (der in f den Wert τ annimt) ausdrückt und in welcher infolge der Betragsbildung Werte aus $\{-2,0,2\}$ stehen können. Zuletzt werden weitere 2^n Zeilen und höchstens 2^n „Schlupfvariablen“ ergänzt, die anzeigen, ob bei der Betragsbildung ein Vorzeichenwechsel vorgenommen wurde, indem sie für alle $x \in \{0,1\}^n$ mit $f(x) = 0$ die Gleichung $\sum_{\{j|x_j=1\}} w_j + \tau = 0$ sowie für alle $x \in \{0,1\}^n$ mit $f(x) \neq 0$ die Gleichung $\sum_{\{j|x_j=1\}} w_j + \tau - s_l = 0$ bzw. $\sum_{\{j|x_j=1\}} (-w_j) - \tau - s_l = 0$ ausdrücken. Wir bezeichnen die Zahl aller Schlupfvariablen wieder mit r. Der Vektor von Unbekannten hat dann die Gestalt $(v_1, \ldots, v_n, v_\tau, s_1, \ldots, s_r)$ und kann wie im ursprünglichen Beweis mit einer Lösung $(v'_1, \ldots, v'_n, v'_\tau, s'_1, \ldots, s'_r)$ belegt werden, die $s'_l > 0$ für alle $l \in \{1, \ldots, r\}$ erfüllt, was äquivalent zu der Aussage ist, dass die Funktion $f'(x) := \sum_{i=1}^n v'_i x_i + v'_\tau$ die Eigenschaft

$$\forall x, y \in \{0,1\}^n : |f(x)| \geq |f(y)| \Leftrightarrow |f'(x)| \geq |f'(y)|$$

besitzt. Die Argumentation, eine Lösung $(v^*_1, \ldots, v^*_n, v^*_\tau, s^*_1, \ldots, s^*_r)$, in der die Werte s^*_l die gewünschten Eigenschaften aufweisen, finden zu können, läuft anschließend wie im ursprünglichen Beweis.
Nachdem nun alle Gewichte zu rationalen Werten transformiert wurden, kann wie in Lemma 2.1 durch Multiplikation mit einem positiven Hauptnenner, die offenkundig auch die Ordnung der Beträge der Funktionswerte nicht beeinflusst, die Ganzzahligkeit aller Gewichte erreicht werden. Auch die Möglichkeit, alle Gewichte als natürlich anzunehmen, ist wegen der Identität

$$\sum_{i=1}^n w_i x_i + \tau = \sum_{\{i|w_i>0\}} w_i x_i + \sum_{\{i|w_i<0\}} (-w_i(1-x_i)) + \sum_{\{i|w_i<0\}} w_i + \tau$$

und der Bemerkung im Beweis zu Lemma 2.1 gegeben. Ebenso stellt die Ordnung der Gewichte nach wie vor kein Problem dar. □

Ab sofort gehen wir also von natürlichen Gewichten aus. Die Annahme $\tau = 0$ bei der Analyse linearer Funktionen wird hier hingegen zu einer echten Einschränkung. Wie erwähnt, ist bei DISTANCE, dem Quadrat einer linearen Fitnessfunktion, bekannt, dass der (1+1)-EA eine exponentielle erwartete Zeit benötigt. Jedoch gilt:

Lemma 2.7 *Gegeben sei eine lineare Funktion $f(x) = \sum_{i=1}^n w_i x_i + \tau$ mit $w_i \in \mathbb{N}$ und $\tau \geq 0$. Der (1+1)-EA verhält sich auf $f^2(x)$ ebenso wie auf $f(x)$.*

Beweis: Wegen $w_i > 0$ für $i \in \{1, \ldots, n\}$ und $\tau \geq 0$ gilt für alle $x \in \{0,1\}^n$, dass $f(x) \geq 0$, und damit $\forall x, y \in \{0,1\}^n : f(x) \geq f(y) \Leftrightarrow f^2(x) \geq f^2(y)$, da $z \mapsto z^2$ auf $\mathbb{R}^+$ einen Ordnungsisomorphismus darstellt. □

Damit übertragen sich also die obere und untere Schranke der erwarteten Laufzeit des (1+1)-EA auf linearen Funktionen ($\Theta(n \ln n)$) im Fall $\tau \geq 0$ unmittelbar auf deren Quadrate. In ähnlicher Weise kann eine Wahl von τ, die $f(x)$ nicht positiv werden lässt, keine schlechtere Laufzeit erzwingen.

Lemma 2.8 *Gilt für eine lineare Fitnessfunktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$ und $\tau \leq 0$, dass $\sum_{i=1}^{n} w_i \leq -\tau$, so verhält sich der (1+1)-EA auf $f^2(x)$ ebenso wie auf $-f(x)$.*

Beweis: Wegen $w_i \geq 0$ und $\sum_{i=1}^{n} w_i + \tau \leq 0$ folgt $\forall x \in \{0,1\}^n : f(x) \leq \sum_{i=1}^{n} w_i + \tau \leq 0$. Zusammen mit den Eigenschaften von $z \mapsto z^2$ auf $\mathbb{R}^+$ erhalten wir

$$\forall x, y \in \{0,1\}^n : f(x) \geq f(y) \Leftrightarrow -f(y) \geq -f(x) \Leftrightarrow f^2(y) \geq f^2(x).$$

□

Natürlich ist auch $-f$ eine lineare Funktion. Folglich gilt für f^2 die Schranke $\Theta(n \ln n)$.
Falls τ die Voraussetzungen der Lemmata 2.7 bzw. 2.8 nicht erfüllt, können die Quadrate linearer Funktionen schwierig zu optimieren sein. Dies soll in Kapitel 3 näher beleuchtet werden.

2.2.3 Fast lineare Funktionen

Gegenstand dieser Untersuchungen sollen auch „fast lineare“ quadratische Fitnessfunktionen sein, welche den linearen im Wesentlichen ähneln. Wir erzeugen eine Ähnlichkeit von quadratischen Funktionen zu linearen Funktionen, indem wir nur konstant viele Monome mit dem Grad 2, d. h. konstant viele Ausnahmen gegenüber linearen Funktionen, erlauben. Somit ergibt sich die folgende formale Erfassung fast linearer Funktionen.

Definition 2.5 (Fast lineare Funktionen) *Eine quadratische Fitnessfunktion f, dargestellt durch $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$ mit $w_i \in \mathbb{Z}, w_{ij} \in \mathbb{Z}$ für $1 \leq i < j \leq n$ und $w_1 \geq \cdots \geq w_n$, wird als fast linear bezeichnet, wenn die Ungleichung $w_{ij} \neq 0$ nur für $O(1)$, d. h. konstant viele Koeffizienten w_{ij} zutrifft.*

In Abschnitt 4.3 erfolgt eine Analyse derartiger Funktionen.

2.2.4 Quadratische Funktionen ohne negative Gewichte

Die bisher bekannten Beispiele quadratischer Fitnessfunktionen mit erwarteter exponentieller Laufzeit weisen die Eigenschaft auf, dass mindestens linear viele der Gewichte negativ sind. Fordert man hingegen von allen w_i sowie w_{ij}, dass diese alle nicht negativ

sind, d. h. wir erhalten mithilfe der bekannten Transformationen aus Lemma 2.4 sogar $w_i \in \mathbb{N}_0, w_{ij} \in \mathbb{N}_0$ für $1 \leq i < j \leq n$ sowie $w_1 \geq \cdots \geq w_n$, drängt sich aufgrund der „Ähnlichkeit“ zu linearen Funktionen die Vermutung auf, hier keine exponentiellen erwarteten Laufzeiten erhalten zu können. Dies untersuchen wir in Abschnitt 4.1.

2.2.5 Weitere linearen Funktionen ähnliche Funktionen

Neben Quadraten linearer Funktionen, fast linearen Funktionen und quadratischen Funktionen ohne negative Gewichte können wir weitere Unterklassen von quadratischen Funktionen, die eine „Verwandtschaft“ zu linearen besitzen, aufstellen. Diese weiter gehende Unterteilung entwickeln wir im Verlauf von Kapitel 4.

Kapitel 3

Quadrate linearer Funktionen

Eine allgemeine Analyse der Quadrate beliebiger linearer Fitnessfunktionen scheint in mancherlei Hinsicht einen bedeutenden Aufwand zu beinhalten. Wäre bei den Lemmata 2.7 und 2.8 kein Rückzug auf die bekannten Resultate für lineare Funktionen möglich, hätte es eines Beweises bedurft, der mindestens denselben Schwierigkeitsgrad wie den des Beweises einer allgemeinen oberen Schranke für lineare Funktionen aufgewiesen hätte. Für den in Abschnitt 2.2.2 vorgestellten Bereich des Summanden τ, in dem keine unmittelbaren Ergebnisse aus den Eigenschaften linearer Funktionen folgen, ist kein geringerer Aufwand zur Gewinnung allgemeiner Resultate zu erwarten. Dennoch lassen sich für den Anfang Erkenntnisse über Quadrate linearer Funktionen festhalten, welche strukturelle Eigenschaften solcher Funktionen verdeutlichen. Wenn nicht anders erwähnt, gehen wir dabei stillschweigend davon aus, dass lineare Funktionen in der „Normalform“ gemäß Lemma 2.6 vorliegen.

3.1 Allgemeine Strukturierung

Eine intuitive, wenn auch nicht vollständig korrekte Erklärung für die einfache Handhabbarkeit linearer Funktionen liegt in deren Eigenschaft, nur ein *lokales Maximum* aufzuweisen, welches immer in $\vec{1} := (1, \ldots, 1)$ liegt. Der in diesem Kontext verwandte Begriff des Maximums (analog zu [DJW98a]) basiert auf dem *Hammingabstand*.

Definition 3.1 (Hammingabstand) *Für zwei Bitstrings $x, y \in \{0,1\}^n$ wird der Hammingabstand definiert durch*

$$H(x,y) := \sum_{i=1}^{n} |x_i - y_i|.$$

Definition 3.2 (Lokales Maximum) *Ein $x \in \{0,1\}^n$ heißt lokales Maximum einer Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$ genau dann, wenn gilt:*

$$\forall y \in \{0,1\}^n : H(x,y) = 1 \Rightarrow f(y) \leq f(x).$$

Offenbar ist mindestens ein lokales Maximum auch ein *globales Maximum*, in dem der maximale Funktionswert angenommen wird. Funktionen, die nur ein lokales und damit zugleich globales Maximum aufweisen, heißen *unimodal*.

Bemerkung 3.1 Lineare Fitnessfunktionen sind unimodal.

Wie in Abschnitt 2.2.2 vorgestellt, ist das Quadrat einer linearen Funktion unter gewissen Voraussetzungen für den (1+1)-EA nicht von der linearen Funktion selbst zu unterscheiden. Die genannten hinreichenden Bedingungen bezogen sich auf den Wertebereich der Funktion, was uns veranlasst, diesen zum Zwecke der besseren Strukturierung zu einer Zerlegung des Definitionsbereiches heranzuziehen.

Definition 3.3 *Gegeben sei eine lineare Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$. Wir definieren:*

$$\begin{aligned} N(f) &:= \{x \in \{0,1\}^n | f(x) < 0\}, \\ P(f) &:= \{x \in \{0,1\}^n | f(x) \geq 0\}. \end{aligned}$$

Bei den zuvor erbrachten Aussagen in den Lemmata 2.7 und 2.8 war die Menge $N(f)$ bzw. $P(f)$ leer. Des Weiteren haben wir folgende Eigenschaften herangezogen:

Lemma 3.1 *Gegeben sei eine lineare Funktion $f : \{0,1\}^n \to \mathbb{R}$. Für $x, y \in \{0,1\}^n$ gilt:*

$$\begin{aligned} x, y \in P(f) &: \; f(x) \geq f(y) \Leftrightarrow f^2(x) \geq f^2(y), \\ x, y \in N(f) &: \; f(x) \geq f(y) \Leftrightarrow f^2(y) \geq f^2(x). \end{aligned}$$

Bemerkung: Diese Aussagen beinhalten, dass anstelle von „$\geq$“ auch die Relation „$>$“ betrachtet werden kann.

Beweis: Folgt unmittelbar aus den Eigenschaften von $x \mapsto x^2$ auf $\mathbb{R}$. □

Unter Zuhilfenahme der Zerlegung des Definitionsbereiches können wir nun Bedingungen für die Existenz lokaler Maxima bei Quadraten linearer Fitnessfunktionen gewinnen, welche zum Beispiel in der Funktion DISTANCE aus [DJW98a] eine wesentliche Rolle spielen.

Satz 3.2 *Das Quadrat einer linearen Funktion f besitzt mindestens ein und maximal zwei lokale Maxima. Lokale Maxima werden nur in $\vec{0}$ oder $\vec{1}$ angenommen.*

Beweis: Sei $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$, $\tau \in \mathbb{R}$ und $w_1 \geq \cdots \geq w_n$ gegeben. Wir greifen zunächst Bemerkung 3.1 auf und zeigen, dass die lineare Funktion f unimodal ist. Für beliebige y mit $H(y, \vec{1}) = 1$ ist $f(y) \leq f(\vec{1})$, da alle w_i positiv sind. Wäre auch x mit $x \neq \vec{1}$ ein lokales Maximum, könnte man mittels Negieren eines Nullbits einen Bitstring mit besserem Funktionswert im Widerspruch zur Annahme konstruieren. Zur Sicherheit halten wir auch die analoge Aussage, dass $\forall y \in \{0,1\}^n : f(y) \geq f(\vec{0})$ gilt und dass für alle $y \in \{0,1\}^n \setminus \{\vec{0}\}$ ein x mit geringerem Funktionswert existiert, fest.

Ist $N(f) = \emptyset$, so besitzt $f^2(x)$ wegen Lemma 3.1 ebenfalls nur ein lokales Maximum in $\vec{1}$. Es gilt $\forall y \in \{0,1\}^n : -f(y) \leq -f(\vec{0})$. Im Fall $P(f) = \emptyset$ kann also mit Lemma 3.1 $\forall y \in \{0,1\}^n : f^2(y) \leq f^2(\vec{0})$ geschlossen werden.
Im Falle $P(f) \neq \emptyset \neq N(f)$ folgt wie zuvor, dass alle $x \in N(f)$ mit $x \neq \vec{0}$ und alle $x \in P(f)$ mit $x \neq \vec{1}$ für die Funktion f^2 nicht lokal optimal sein können. Von den verbleibenden Punkten $\vec{0}$ und $\vec{1}$ muss mindestens einer lokal optimal sein, da es sonst im Widerspruch zur Endlichkeit der Definitionsmenge kein globales Maximum gäbe. □

Korollar 3.3 *Falls $|f(\vec{0})| < |f(\vec{1})|$, liegt in $\vec{1}$ ein globales Maximum von f^2. Im Falle $|f(\vec{0})| > |f(\vec{1})|$ liegt in $\vec{0}$ ein globales Maximum. Wenn Gleichheit gilt, sind beide Punkte globale Maxima.*

Beweis: Bei den ersten beiden Aussagen wende man Satz 3.2 an. Im dritten Fall wird sogar in beiden möglichen Punkten für lokale Maxima ein maximaler Funktionswert angenommen. □

Beispiel 3.1 Das Quadrat der linearen Funktion $f(x) = (2n+1)x_1 + \sum_{i=2}^{n} x_i - n$ nimmt offenbar in $\vec{1}$ sein globales Maximum mit dem Funktionswert $4n^2$ an. Jedoch ist $\vec{0}$ kein lokales Maximum, da $f^2(\vec{0}) = n^2 < (n+1)^2 = f^2(1,0,\ldots,0)$, was auch verdeutlicht, dass die Mächtigkeit der Mengen $N(f)$ und $P(f)$ keine Rückschlüsse über die Existenz lokaler Maxima gestattet, da hier $\#N(f) = \#P(f) = 2^{n-1}$ ist.

Von dieser Funktion können wir nun leicht zeigen, dass sie für den (1+1)-EA „einfach“ ist.

Satz 3.4 *Sei f definiert wie in Beispiel* 3.1*. Der (1+1)-EA optimiert das Quadrat der Funktion f in erwarteter Zeit $O(n \ln n)$.*

Beweis: Zunächst analysieren wir das Verhalten des (1+1)-EA unter der Bedingung, dass x_1^s mit 0 initialisiert wurde. Da die Wahrscheinlichkeit, x_1 zu flippen, $1/n$ beträgt, ereignet sich im Erwartungswert nach n Schritten eine solche Mutation, woraufhin aufgrund der Eigenschaft $w_1 > n = |f(\vec{0})|$ keine Mutation nach $N(f)$ mehr stattfindet. Zumal jeder Vektor mit $x_1 = 1$ in $P(f)$ liegt, wird eine Mutation auf dem Quadrat von f im Folgenden daher genau dann akzeptiert, wenn sie auf der linearen Funktion f akzeptiert wird. Da nun die Resultate für erwartete Laufzeiten auf linearen Funktionen zum Tragen kommen (im Fall $x_1^s = 1$ ist dies also ohnehin unmittelbar nach der Initialisierung der Fall), wird f^2 insgesamt in erwarteter Zeit von höchstens $n + O(n \ln n) = O(n \ln n)$ optimiert. □

Bemerkung 3.2 Die untere Schranke für die Laufzeit des (1+1)-EA von $\Omega(n \ln n)$ aus Satz 2.5 gilt für die Quadrate linearer Funktionen mit nur einem globalen Maximum. Aus dem Beweis wird auch unmittelbar klar, dass er mühelos angepasst werden kann, um eine untere Schranke von $\Omega(n \ln n)$ für die Quadrate linearer Funktionen zu zeigen, die sowohl in $\vec{0}$ als auch in $\vec{1}$ ein globales Maximum besitzen. (Es sind mit exponentiell nahe an 1 liegender Wahrscheinlichkeit nach der Initialisierung mindestens $0{,}4n$ und höchstens $0{,}6n$

Bits gegenüber dem Maximum in $\vec{1}$ richtig eingestellt; ein Bit ist zum Maximum in $\vec{1}$ genau dann richtig eingestellt, wenn es zum Maximum in $\vec{0}$ falsch eingestellt ist, d. h. es sind mit exponentiell nahe an 1 liegender Wahrscheinlichkeit auch zwischen $0{,}4n$ und $0{,}6n$ Bits gegenüber dem Maximum in $\vec{0}$ falsch eingestellt, und es müssen mit exponentiell nahe an 1 liegender Wahrscheinlichkeit mindestens $0{,}4n$ Bits geflippt werden.) Wir halten diese untere Schranke bei allen Aussagen über die Quadrate linearer Funktionen in Gedanken und schreiben daher bei erwarteten Laufzeiten von $\Theta(n \ln n)$ statt dieses stärkeren Resultates meist $O(n \ln n)$, da wir nur noch obere Schranken zu zeigen brauchen.

Zum Abschluss dieses Abschnittes stellen wir noch eine Annahme vor, die im Weiteren die Analyse der Quadrate linearer Funktionen vereinfachen wird, insofern als „duale“ Aussagen nicht mehr bewiesen zu werden brauchen.

Lemma 3.5 *Beim Quadrat f^2 einer linearen Fitnessfunktion f dürfen wir o. B. d. A. annehmen, dass ein globales Maximum in $\vec{1}$ liegt.*

Beweis: Sei $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$, $\tau \in \mathbb{R}^{<0}$ und $w_1 \geq \cdots \geq w_n$ eine lineare Fitnessfunktion, wobei $|f(\vec{0})| > |f(\vec{1})|$, d. h. $|\tau| > |\sum_{i=1}^{n} w_i + \tau|$ ist. (Sonst ist nichts zu zeigen.) Im Beweis zu Lemma 2.1 haben wir begründet, dass die Rollen von Nullen und Einsen in beliebigen Positionen $i \in \{1, \ldots, n\}$ eines Bitstrings vertauscht werden dürfen, indem wir in allen Monomen der Polynomdarstellung der untersuchten Funktion x_i durch $(1 - x_i)$ ersetzen. Wir bilden daher vermöge der Abbildung $c : \{0,1\}^n \to \{0,1\}^n$, definiert durch $c(x) = \bar{x}$, die Funktion $f'(x) := (f \circ c)(x) = \sum_{i=1}^{n} -w_i x_i + \tau'$ mit $\tau' = \tau + \sum_{i=1}^{n} w_i$. Somit ist

$$\left|f'(\vec{0})\right| = \left|\tau'\right| = \left|\sum_{i=1}^{n} w_i + \tau\right| < \left|\tau\right| = \left|f'(\vec{1})\right|.$$

Da $|f'(x)| = |-f'(x)|$ für alle $x \in \{0,1\}^n$, haben wir mit der Funktion $f^*(x) := -f'(x) = \sum_{i=1}^{n} w_i x_i + (-\tau')$ eine lineare Fitnessfunktion in der üblichen Form gemäß Lemma 2.6 gefunden. Die Funktion $(f^*)^2$ nimmt nur in $\vec{1}$ (vgl. Satz 3.2) ihr globales Maximum an. □

Bemerkung 3.3 In diesem Abschnitt haben wir den interessanten Bereich des Parameters τ weiter eingeschränkt. In Verbindung mit den Lemmata 2.7 und 2.8 besagt Lemma 3.5, dass wir bei der Betrachtung des Quadrats einer linearen Funktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$ und $w_1 \geq \cdots \geq w_n$ o. B. d. A. davon ausgehen können, dass τ im Bereich $\left(-\sum_{i=1}^{n} w_i\right)/2 \leq \tau < 0$ liegt.

3.2 Die Funktion „OneSqr“

In diesem Abschnitt wollen wir auf das Verhalten des Quadrates der denkbar einfachsten linearen Funktion eingehen, d. i. das Quadrat der Funktion OneMax. Deren formale Definition liefern wir nun nach.

Definition 3.4 *Die Funktion* OneMax: $\{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\textsc{OneMax}(x) := \sum_{i=1}^{n} x_i.$$

Das mit OneSqr bezeichnete Quadrat der Funktion OneMax hat dann, da nun der Summand τ infolge der bisherigen Ergebnisse nicht vernachlässigt werden darf, die folgende Gestalt.

Definition 3.5 *Die Funktion* $\textsc{OneSqr}_\tau : \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\textsc{OneSqr}_\tau(x) := \Big(\sum_{i=1}^{n} x_i + \tau\Big)^2 \text{ mit } \tau \in \mathbb{R}.$$

Die Summe der Gewichte der Funktion $\textsc{OneSqr}_\tau$ beträgt n. Wir entnehmen Bemerkung 3.3, dass nur Werte von τ im Bereich $-n/2 \leq \tau < 0$ neue Erkenntnisse zu bringen vermögen.
Im Falle $\tau = -n/2$ existieren zwei globale Maxima (nämlich in $\vec{0}$ und in $\vec{1}$), und die Funktion ist für den (1+1)-EA leicht zu optimieren.

Lemma 3.6 *Die Funktion* $\textsc{OneSqr}_{-n/2}$ *wird vom (1+1)-EA in erwarteter Zeit* $O(n \ln n)$ *optimiert.*

Beweis: In diesem Beweis betrachten wir nur diejenigen (akzeptierten) Mutationen des (1+1)-EA, die höchstens ein Bit mutieren und berechnen die erwartete Zeit, bis mithilfe solcher Mutationen ein globales Optimum erreicht wurde. Bei diesem Vorgehen benutzen wir für die wiederum mit f benannte zugrunde liegende lineare Funktion die Identitäten $f(x) = n/2 - H(x,\vec{1})$, falls $x \in P(f)$, und $-f(x) = n/2 - H(x,\vec{0})$ für $x \in N(f)$. Damit verringern akzeptierte Mutationen, die mehr als ein Bit flippen und den Funktionswert von $\textsc{OneSqr}_{-n/2}$ erhöhen, den Hammingabstand zu einem der globalen Maxima um mindestens denselben Wert wie die Mutation eines einzigen Bits, sodass das Ignorieren solcher Mutationen die Anzahl noch erforderlicher 1-Bit-Mutationen und somit die erwartete Laufzeit allenfalls erhöhen kann. Falls eine akzeptierte Mutation mehr als eines Bits den Funktionswert unverändert lässt, kann ein Flipp von $P(f)$ nach $N(f)$ (bzw. umgekehrt) mit gleichem Funktionswert erfolgt sein. In diesen Fällen wird aber der Hammingabstand des generierten Bitstrings $x \in N(f)$ zum Maximum in $\vec{0}$ identisch zum Hammingabstand zu $\vec{1}$ vor der Mutation und umgekehrt, d. h. es sind noch ebenso viele 1-Bit-Mutationen wie zuvor zum Erreichen eines der globalen Maxima nötig und ausreichend, während diese im Falle $x \in P(f)$ Mutationen, die ein Bit von 0 nach 1 flippen, darstellen und umgekehrt. Wir geben bei der Ermittlung einer obereren Schranke mithin o. B. d. A. vor, dass nur 1-Bit-Mutationen und „Mutationen“, die überhaupt kein Bit flippen, akzeptiert werden können.
Falls der unmittelbar nach der Initialisierung erzeugte zufällige Bitstring x^s genau $n/2$ Einsen enthält, entscheidet die erste Mutation, ob der geänderte Algorithmus den Bitstring

$\vec{0}$ oder $\vec{1}$ als optimal ermittelt. Da die Wahrscheinlichkeit, genau ein Bit zu flippen, aufgrund der Beziehung $(1-1/n)^{n-1} \geq e^{-1}$ mindestens $\binom{n}{1}(1/n)(1-1/n)^{n-1} \geq e^{-1}$ beträgt, ist die erwartete Zeit bis zur ersten Mutation durch e nach oben beschränkt, d. h. konstant. Im Folgenden sind dann die erwarteten Laufzeiten im zweiten bzw. im dritten Fall hinzuzuzählen.

Im Fall 2, dass mehr als $n/2$ Bits von x^s Einsen sind, akzeptiert der geänderte (1+1)-EA nur Mutationen, die die Zahl der Einsen erhöhen. Die Wahrscheinlichkeit, einen Bitstring $x \in \{0,1\}^n$ mit $n/2 < c < n$ Einsen zu einem Bitstring mit $c+1$ Einsen zu mutieren, beträgt mindestens

$$\binom{n-c}{1}\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq e^{-1}\frac{n-c}{n}.$$

Die Zufallsvariable X_c bezeichnet die Zeit bis zum erstmaligen Eintreten einer solchen Mutation. Dann ist $E[X_c] \leq en/(n-c)$. Für die Zufallsvariable X, die die Zeit bis zum Erreichen von $\vec{1}$ angibt, gilt somit

$$E[X] = E[X_c + X_{c+1} + \cdots + X_{n-1}] \leq \sum_{i=c}^{n-1}\frac{en}{n-i} \leq en\sum_{i=1}^{n-1}\frac{1}{n-i} \leq enH(n) = O(n\ln n),$$

wobei $H(n)$ für die Harmonische Reihe bis zum Glied $1/n$ steht, deren asymptotisches Verhalten mit $H(n) = \ln n + \Theta(1)$ angegeben werden kann.

Der dritte Fall des Beweises – x^s enthält weniger als $n/2$ Einsen, d. h. mehr als $n/2$ Nullen – verläuft analog zum zweiten Fall. □

Zu untersuchen bleiben Werte von τ im Bereich $-n/2 < \tau < 0$. Die Funktion DISTANCE aus [DJW98a] ist der Funktion ONESQR_τ mit $\tau = -n/2+1/3$ bis auf Komplementbildung nach Lemma 3.5 identisch (zur Illustration vgl. auch Abbildung 3.1) und ruft eine exponentielle erwartete Laufzeit des (1+1)-EA hervor. Der folgende Satz zeigt, dass bei nur mit $n^{o(1)}$ und sogar noch mit $o(n/\log n)$ wachsenden Werten von $|\tau|$ keine exponentiellen erwarteten Laufzeiten möglich sind.

Satz 3.7 *Sei $-n/2 < \tau < 0$ und $|\tau| \leq (1-\epsilon)n/(2\log n)$ für ein $\epsilon > 0$. Dann optimiert der (1+1)-EA die Funktion ONESQR_τ in erwarteter Zeit $O(n\ln n)$.*

Beweis: Wir schreiben τ gemäß der Voraussetzung als $\tau \geq -cn/\log n$ für ein $c < 1/2$. Aus beweistechnischen Gründen gehen wir ferner davon aus, dass $cn/\log n \geq 2$ sowie $2cn/\log n \in \mathbb{N}$ gelten und dass sogar $-\tau = cn/\log n$ zutrifft, womit sich der Wert der oberen Schranke allenfalls vergrößert, weil sich die Wahrscheinlichkeit, in $N(f)$ zu starten oder zu gelangen, erhöht. Weiter wird mit $f(x) = \sum_{i=1}^{n} x_i + \tau$ die lineare Funktion, die ONESQR_τ zugrunde liegt, bezeichnet.

Unter der Bedingung, dass der Bitstring des EA mit mehr als $2|\tau|$ Einsen initialisiert wird, folgt die obere Schranke aus den Ergebnissen für ONEMAX, da der (1+1)-EA das gleiche Akzeptanzverhalten für Mutationen wie bei einem Vektor mit mehr als $2|\tau|$ Einsen auf

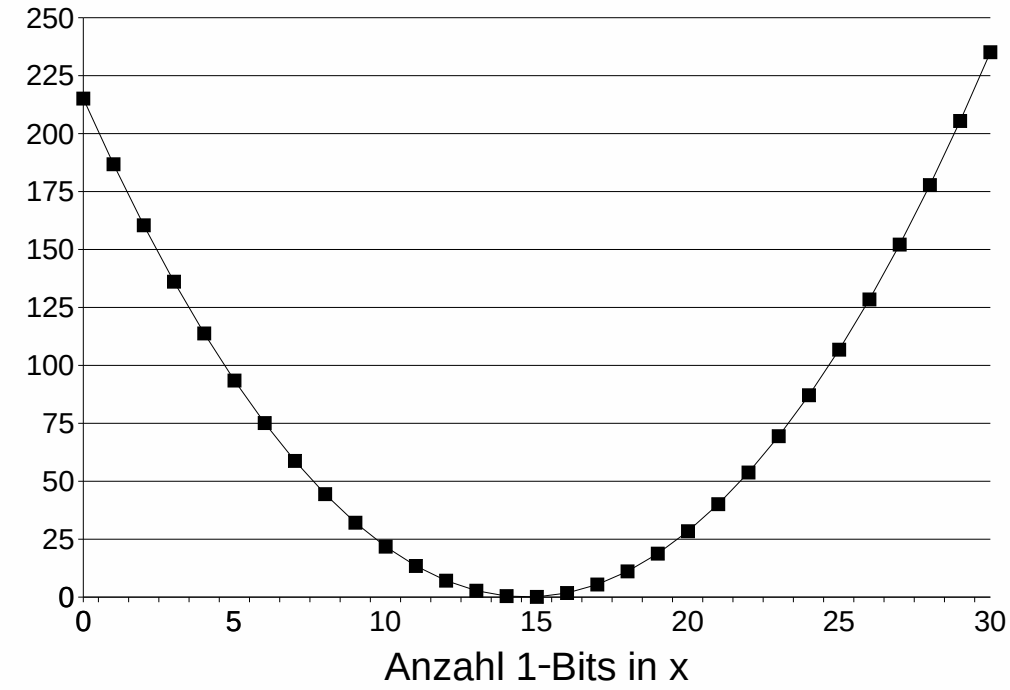

Abbildung 3.1: $\textsc{OneSqr}_{-n/2+1/3}$ für $n = 30$

der Funktion OneMax zeigt. Insgesamt erzeugt der EA in der Initialisierung mit einer Wahrscheinlichkeit von höchstens (bei den Abschätzungen für Binomialkoeffizienten vgl. Anhang A.2)

$$
\begin{aligned}
2^{-n}\sum_{i=0}^{2cn/\log n}\binom{n}{i} &\le 2^{-n}\left(\frac{2cn}{\log n}+1\right)\binom{n}{2cn/\log n} \le 2^{-n}\left(\frac{2cn}{\log n}+1\right)\left(\frac{ne}{2cn/\log n}\right)^{\frac{2cn}{\log n}} \\
&\le 2^{-n} n^{2cn/\log n} e^{2cn/\log n}\left(\frac{2cn}{\log n}\right)^{-2cn/\log n+2} \\
&= 2^{-n+(\log n)(2cn/\log n)+(\log e)(2cn/\log n)-\log(2cn/\log n)(2cn/\log n)+2\log(2cn/\log n)} \\
&= 2^{-n+2cn+(\log e)(2cn/\log n)-((\log n)+\log(2c/\log n))(2cn/\log n)+2\log(2cn/\log n)} \\
&= 2^{-n+2cn+(\log e)(2cn/\log n)-2cn-\log(2c/\log n)(2cn/\log n)+2\log(2cn/\log n)} \\
&= 2^{-n+(\log(e/2c)+\log\log n)(2cn/\log n)+2\log(2cn/\log n)} = 2^{-n+o(n)}
\end{aligned}
$$

einen Vektor mit höchstens $2|\tau|$ Einsen – dies nennen wir das Ereignis I – und gelangt unter dieser Bedingung mit positiver Wahrscheinlichkeit zum lokalen Maximum[1] in $\vec{0}$. Ist $\vec{0}$ der aktuelle Bitstring, werden nur Mutationen, die mindestens $2cn/\log n$ Bits gleichzeitig

[1]Ganz genau genommen ist der Vektor $\vec{0}$ gemäß Definition 3.2, falls $\tau > -1/2$ oder $n = 1$, kein lokales Maximum. Abgesehen davon, dass wir in der Beweisführung $|\tau| \ge 1/2$ voraussetzen, ist der Fall $\tau > -1/2$ ohnehin uninteressant, da dann jeder Vektor mit mindestens einem Einsbit der Menge $P(f)$ angehört und eine ähnliche Situation wie in Satz 3.4 eintritt. Den Fall $n = 1$ schließen wir ohnehin o. B. d. A. aus.

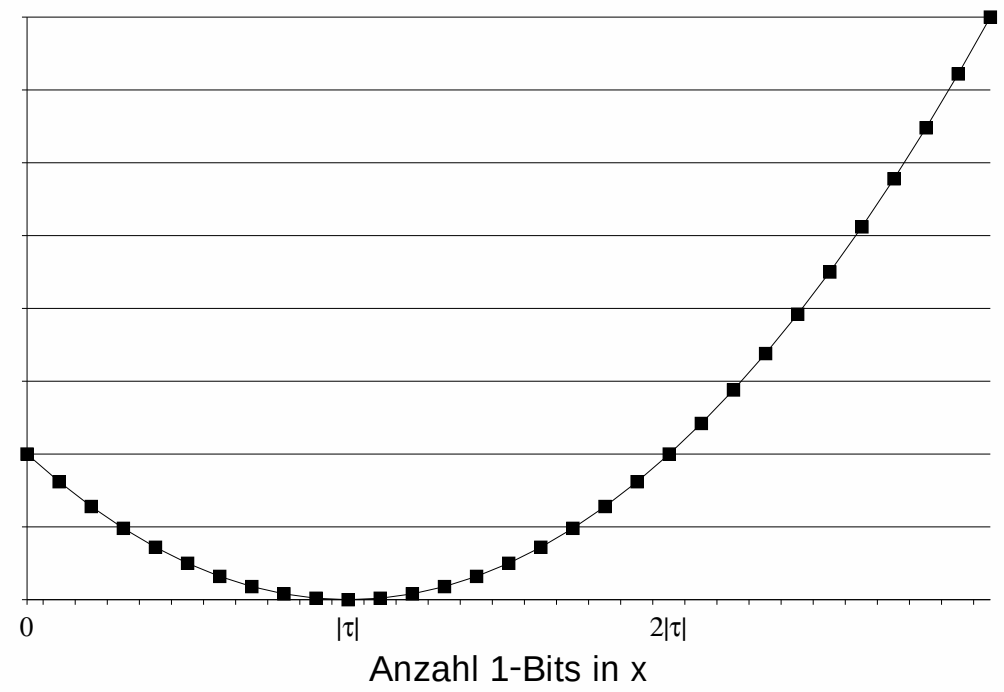

Abbildung 3.2: $\textsc{OneSqr}_\tau$ für $\tau \gg -n/2$

flippen, akzeptiert (zur Illustration vgl. Abbildung 3.2). Auf der anderen Seite ist das Eintreten einer Mutation, die in einem Schritt mindestens $k := 2cn/\log n + 1$ mit 0 belegte Bits und keine mit 1 belegten Bits flippt, eine hinreichende Bedingung dafür, dass alle aktuellen Bitstrings im Folgenden mindestens $2|\tau| + 1$ Einsen enthalten und dass sich der (1+1)-EA danach wie auf f verhält.
Die Wahrscheinlichkeit, mindestens k Nullbits und kein Einsbit zu mutieren, ist nach unten beschränkt durch die Wahrscheinlichkeit, genau k ausgewählte Nullbits zu mutieren, d. h.

$$\left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} \geq \left(\frac{1}{n}\right)^k e^{-1} = e^{-1} n^{-k}.$$

Unter der Bedingung I schätzen wir die erwartete Zeit bis zur einer der betrachteten Mutationen mit en^k nach oben ab, woraufhin (wegen äquivalenten Verhaltens zu OneMax, s. o.) noch erwartete $O(n \ln n)$ Schritte bis zum Erreichen des globalen Maximums in $\vec{1}$ nötig sind. Fassen wir diesen bedingten Erwartungswert und den Erwartungswert $O(n \ln n)$, der sonst gilt, zum Erwartungswert der Laufzeit (Zufallsvariable X) zusammen, ergibt sich

$$\begin{aligned} E[X] &\leq E[X|I] \cdot \text{Prob}(I) + E[X|\bar{I}] \\ &\leq 2^{-n+o(n)} \cdot \left(e \cdot n^{2cn/\log n+1}\right) + O(n \ln n) \\ &\leq e \cdot 2^{-n+o(n)+(\log n)(2cn/\log n+1)} + O(n \ln n) \\ &= e \cdot 2^{-n+2cn+o(n)} + O(n \ln n) = 2^{-\Omega(n)} + O(n \ln n) = O(n \ln n), \end{aligned}$$

da $c < 1/2$ vorausgesetzt wird. □

Auf der anderen Seite können wir auch hinreichende Bedingungen für eine erwartete exponentielle Laufzeit errechnen.

Satz 3.8 *Sei $-n/2 < \tau < 0$ und $|\tau| \geq (1+\epsilon)n/(2\log n)$ für ein $\epsilon > 0$. Dann benötigt der (1+1)-EA zur Optimierung der Funktion* $\textsc{OneSqr}_\tau$ *erwartete Zeit $2^{\Omega(n)}$.*

Beweis: Zunächst schreiben wir $\tau \leq -cn/\log n$ für ein $c > 1/2$ und nehmen wiederum an, dass $cn/\log n$ ganzzahlig ist, um eventuelle Auf- und Abrundungen zu vermeiden und setzen sogar $|\tau| = cn/\log n$ voraus; f bezeichne die $\textsc{OneSqr}_\tau$ zugrunde liegende lineare Funktion. Wir analysieren ausschließlich die Situation, während der Initialisierung den Vektor $\vec{0}$ zu erzeugen (Ereignis I), woraufhin eine Mutation von mindestens $2|\tau|$ Bits zugleich vonnöten ist, um einen Vektor aus $P(f)$ zu erzeugen und anschließend zum globalen Maximum zu gelangen (vgl. Abbildung 3.2). Die Wahrscheinlichkeit, den Startvektor mit $\vec{0}$ zu initialisieren, beträgt 2^{-n}. Demgegenüber ist die Wahrscheinlichkeit, mindestens $k := 2cn/\log n$ Bits zugleich zu flippen, höchstens $\binom{n}{k}(1/n)^k \leq (n^k/k!) \cdot n^{-k} = 1/k!$ (siehe auch Lemma A.8), d. h. die erwartete Zeit bis zu einer solchen Mutation beträgt mindestens $k!$ Schritte. Somit trägt das Ereignis I aufgrund der Stirlingformel einen Summanden von mindestens

$$\begin{aligned}
&2^{-n} \cdot e^{-2cn/\log n}(2cn/\log n)^{2cn/\log n} = 2^{-n-(\log e)(2cn/\log n)+\log(2cn/\log n)(2cn/\log n)} \\
&= 2^{-n-(\log e)(2cn/\log n)+(\log(2c/\log n)+\log n)(2cn/\log n)} = 2^{-n+(-\log e+\log(2c/\log n))(2cn/\log n)+2cn} \\
&= 2^{-n+2cn+\log(2c/(e\log n))(2cn/\log n)} = 2^{\Omega(n)}, \text{ da } c > 1/2,
\end{aligned}$$

zur erwarteten Laufzeit bei. □

Bemerkung 3.4 Bei $\tau = -n/(2\log n)$ scheint die erwartete Laufzeit von $O(n\ln n)$ auf $2^{\Omega(n)}$ umzuschlagen. Bei der Errechnung der oberen und unteren Schranke konnten wir die Bereiche von $|\tau|$, in denen die erwartete Laufzeit $O(n\ln n)$ bzw. $2^{\Omega(n)}$ beträgt, sehr genau einander angleichen, weil sich die erwartete Laufzeit für $\tau = -cn/\log n$, $c > 0$, in beiden Fällen durch $2^{-n+2cn\pm o(n)}$ darstellen lässt. Interessanterweise braucht bei der Ermittlung der unteren Schranke lediglich das Ereignis, in $\vec{0}$ zu starten, betrachtet zu werden, da unter dieser Bedingung $|2\tau|$ Bits zu flippen haben und der Summand $2cn$ im Exponenten infolge der Stirlingformel entsteht.
Am Rande sei bemerkt, dass sich für $\tau = -n/(2\log n)$ eine untere Schranke für die Laufzeit von

$$2^{-(\log e)(n/\log n)+\log(1/\log n)(n/\log n)} = 2^{\log(1/(e\log n))(n/\log n)} = 2^{-\Omega(n/\log n)}$$

ergibt. Die obere Schranke nimmt dagegen den Wert

$$e \cdot 2^{\log(e+\log\log n)(n/\log n)+2\log(n/\log n)+\log n} + O(n\ln n) = 2^{\Omega(n/\log n)+o(n/\log n)} = 2^{\Omega(n/\log n)}$$

an, d. h. bei diesem Wert von τ liefern die Rechnungen aus Satz 3.7 und 3.8 keine sinnvollen Erkenntnisse über die erwartete Laufzeit.

Bemerkung 3.5 Im Falle $|\tau| = \Omega(n/\log\log n)$ können wir eine noch größere untere Schranke zeigen. Die erwartete Zeit bis zu einem Flipp von $cn/\log\log n$ Bits (für ein beliebiges $c > 0$) ist nach unten beschränkt durch

$$(cn/\log\log n)! = 2^{\Omega((cn/\log\log n)\log(cn/\log\log n)-(cn\log e)/\log\log n)} = 2^{\Omega(n\log n/\log\log n)},$$

und selbst die minimale Wahrscheinlichkeit von 2^{-n} für einen Start in $N(f)$ belässt das asymptotische Verhalten der erwarteten Laufzeit bei $2^{\Omega(n\log n/\log\log n)}$. Ohne größere Mühe erkennen wir hieran auch einen Erwartungswert von $2^{\Omega(n\log n)}$, falls $|\tau| = \Theta(n)$ zutrifft.

Die Funktion DISTANCE veranlasst den (1+1)-EA sogar zu der noch schlechteren erwarteten Laufzeit $\Omega(n^n)$. Dies kann jedoch nur unter der Bedingung $-\tau = n/2 - o(n)$ eintreten. Wir betrachten dazu lediglich den „Sprung“ von einem Vektor mit o Einsen aus zu einem Vektor mit mindestens $o + 2|\tau| + 1$ Einsen, da die erwartete Zeit bis zum Erreichen von $\vec{1}$ ohne den Sprung ohnehin durch $O(n\ln n)$ beschränkt ist. Die Wahrscheinlichkeit für eine Mutation von mindestens k ausgewählten Nullbits zugleich ist im Falle $1 \le k \le n$ durch $e^{-1}n^{-k}$ nach unten beschränkt. Selbst wenn der (1+1)-EA mit Wahrscheinlichkeit 1 den Sprung ausführen müsste, würde dies zur erwarteten Laufzeit höchstens en^k beitragen. Die obere Schranke en^k kann zwar für alle $k \in \{1,\dots,n\}$ als $2^{\Theta(k\log k)}$ geschrieben werden, jedoch nur für $k = n$ als $\Theta(n^n)$.
Abgesehen von der im Allgemeinen unüblichen Unterscheidung zwischen verschiedenen exponentiellen Größen sind diese Betrachtungen der erwarteten Laufzeit jedoch auch insofern lediglich theoretischer Natur, als ONESQR_τ selbst für den „Extremfall“ $\tau = -n/2+o(n)$ mit Wahrscheinlichkeit $1/2 - \epsilon$ in der Zeit $O(n\ln n)$ maximiert wird (vgl. [DJW98a]). Wächst $|\tau|$ proportional zu n, können meist noch bessere Wahrscheinlichkeiten für polynomielle Laufzeiten gezeigt werden. Im folgenden Satz halten wir im Hinterkopf, dass die erwartete Laufzeit aber wegen Satz 3.8 immer noch exponentiell ist.

Satz 3.9 *Lässt sich τ als $\tau = -\delta n/2$ für ein $\delta \in]0,1[$ darstellen, optimiert der (1+1)-EA die Funktion ONESQR_τ für ein beliebiges $\epsilon > 0$ mit Wahrscheinlichkeit von mindestens $1-\epsilon$ in $O(n\ln n)$ Schritten.*

Beweis: Es bezeichne f wiederum die ONESQR_τ zugrunde liegende lineare Funktion. Unter Verwendung der Chernoffschranke (vgl. Lemma A.1) kann die Wahrscheinlichkeit, dass der (1+1)-EA mit einem Bitstring mit mindestens $(1-(1-\delta)/2)\cdot n/2$ Einsen startet, mit $1 - 2^{-\Omega(n)}$ nach unten abgeschätzt werden. Unter dieser Bedingung müssten mindestens $k := (1-\delta)\cdot n/2$ Bits zugleich flippen, um einen Bitstring aus $N(f)$ zu erzeugen, der während der Optimierung von ONESQR_τ akzeptiert werden könnte. Sonst verhält sich der (1+1)-EA wie auf der linearen Funktion f, und die erwartete Laufzeit beträgt maximal $c'n\ln n$ für ein $c' \in \mathbb{R}^{>0}$. Die Wahrscheinlichkeit, in $\lceil cc'n\ln n\rceil$ Schritten, $c \in \mathbb{R}^{>0}$, niemals mindestens k Bits zugleich zu mutieren, liegt ebenfalls exponentiell nahe bei 1. Die Wahrscheinlichkeit, mindestens k Bits auf einmal zu flippen, ist nämlich höchstens $1/k!$ (Lemma A.8). Eine obere Schranke dafür, dass eine Mutation von mindestens k Bits mindestens

einmal in $\lceil cc'n \ln n \rceil$ Schritten eintritt, ist dann $(1/k!)(\lceil cc'n \ln n \rceil)$, da die Wahrscheinlichkeit einer Vereinigung von Ereignissen höchstens die Summe der Wahrscheinlichkeiten der Einzelereignisse ist. Dieser Ausdruck ist wegen $k = \Omega(n)$ und der Stirlingformel durch

$$\frac{\lceil cc'n \ln n \rceil}{e^{-k}k^k} = 2^{\log(\lceil cc'n \ln n \rceil)} \cdot e^{O(n)} \cdot 2^{-\Omega(n \log n)} = 2^{-\Omega(n \log n)}$$

nach oben beschränkt.
Aufgrund der Markoffungleichung (siehe Lemma A.3) optimiert der (1+1)-EA die lineare Funktion f mit einer Wahrscheinlichkeit von mindestens $1 - 1/c$ in höchstens $\lceil cc'n \ln n \rceil$ Schritten. Wir wählen c so groß, dass $1/c$ zuzüglich der beiden exponentiell kleinen Wahrscheinlichkeiten durch ϵ nach oben beschränkt ist. Dann tritt nur mit Wahrscheinlichkeit von höchstens ϵ mindestens einer der beschriebenen „Fehler“ ein, d. h. dass der EA mit Wahrscheinlichkeit von mindestens $1 - \epsilon$ in höchstens $\lceil cc'n \ln n \rceil = O(n \ln n)$ Schritten die Funktion $\textsc{OneSqr}_\tau$ maximiert. □

Trotz erwarteter exponentieller Laufzeit können wir unter den Bedingungen von Satz 3.9 sogar schließen, dass nur mit exponentiell kleiner Wahrscheinlichkeit auch exponentielle Laufzeiten beobachtet werden. Die Wahrscheinlichkeit, zur Optimierung einer linearen Funktion statt $O(n \ln n)$ exponentielle Zeit zu benötigen, kann nämlich mithilfe der Markoffungleichung als exponentiell klein nachgewiesen werden, da die beobachtete Laufzeit um einen exponentiell großen Faktor von der erwarteten abweichen müsste. Im vorigen Satz 3.9 sind also die Wahrscheinlichkeiten der „Fehler“, weniger als $((1 + \delta)/2) \cdot n/2$ Bits mit 1 initialisiert zu haben, mindestens einmal in $O(n \ln n)$ Schritten mindestens $(1-\delta) \cdot n/2$ Bits zugleich zu flippen und bei der Optimierung linearer Funktionen exponentiell lange zu brauchen, als exponentiell klein nachgewiesen.

3.3 Allgemeines Laufzeitverhalten

Wir untersuchen nun Quadrate linearer Funktionen, denen Koeffizienten nicht notwendigerweise alle gleich sind, betrachten also die Quadrate von Funktionen $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$ für alle $i \in \{1, \ldots, n\}$ und $w_1 \geq \cdots \geq w_n$, wobei nur der Fall $-w/2 \leq \tau < 0$ mit $w := \sum_{i=1}^{n} w_i$ interessant ist (vgl. Bemerkung 3.3). Im folgenden Abschnitt gehen wir wiederum stillschweigend davon aus, dass lineare Funktionen in dieser Form vorliegen. Die Variable w bezeichne immer die Summe der Gewichte der betrachteten Funktion.

3.3.1 Erwartete Laufzeiten bis zum Erreichen von $\vec{0}$ oder $\vec{1}$

Im Falle $\tau = -w/2$ besitzt f^2, das Quadrat einer linearen Funktion f, zwei globale Maxima, was sich bei $\textsc{OneSqr}_{-n/2}$ in Lemma 3.6 als hinreichende Bedinung für eine effiziente Optimierung erwiesen hat. Dies wollen wir auch für den allgemeinen Fall nachweisen. Dazu zeigen wir sogar für beliebige τ im Bereich $-w/2 \leq \tau < 0$, dass der (1+1)-EA in effizienter erwarteter Zeit zu einem der zwei lokalen Maxima von f^2 gelangt. (Wir geben jetzt aus

ähnlichen Gründen wie im Beweis von Satz 3.7 vor, dass $\vec{0}$ ein lokales Maximum ist.) Im Fall $\tau = -w/2$ erhalten wir dann die beabsichtigte Aussage als Korollar.

Zur Vorbereitung des Beweises definieren wir ein Fortschrittsmaß.

Definition 3.6 *Für eine lineare Funktion f und ein $x \in \{0,1\}^n$ werden der maximal (von x für den (1+1)-EA) zu $\vec{1}$ erreichbare Hammingabstand $H_1^*(x)$ und der maximal (von x für den (1+1)-EA) zu $\vec{0}$ erreichbare Hammingabstand $H_0^*(x)$ definiert durch*

$$H_1^*(x) := n - \max\Big\{j \in \{0,\ldots,n\} | f(x) \geq \sum_{i=1}^{j} w_i + \tau\Big\} \text{ und}$$

$$H_0^*(x) := n - \max\Big\{j \in \{0,\ldots,n\} | f(x) \leq w - \sum_{i=1}^{j} w_i + \tau\Big\}.$$

Wie beschrieben, hängen H_1^* und H_0^* auch von f ab. Aus Gründen der Übersichtlichkeit verzichten wir aber auf einen Index f, da im Folgenden aus dem Zusammenhang klar wird, für welche Funktionen wir die Maße nutzen. Dies werden wir auch bei weiteren Verwendungen von Fortschrittsmaßen in dieser Arbeit (z. B. in Kapitel 4) so handhaben.

Bei der Betrachtung allgemeiner linearer Funktionen besteht anders als bei OneMax normalerweise keine Bijektion zwischen dem Funktionswert und der Zahl der Einsen im aktuellen Bitstring. Am Beispiel der Funktion BinVal aus [DJW98a] kann man leicht einsehen, dass der Hammingabstand zum optimalen Bitstring im Laufe der Optimierung vorübergehend steigen kann. Eine Mutation von $(0,1,\ldots,1)$ zu $(1,0,\ldots,0)$ würde während der Optimierung der linearen Funktion akzeptiert, wohingegen der Hammingabstand zu $\vec{1}$ von 1 auf $n-1$ steigt. Ähnliche Situationen können auch bei der Optimierung der Quadrate linearer Funktionen eintreten, wobei beide lokalen Optima zu betrachten sind.

Mit den aus dem üblichen Hammingabstand abgeleiteten Maßen $H_1^*(x)$ und $H_0^*(x)$ messen wir, wie groß der Hammingabstand zum jeweiligen lokalen Optimum höchstens werden kann, nachdem der (1+1)-EA einen Bitstring x erzeugt hat. Das erläutern wir formal im folgende Lemma.

Lemma 3.10 *Sei f eine lineare Funktion und $x \in \{0,1\}^n$ ein Vektor ihrer Definitionsmenge. Für alle $x' \in \{0,1\}^n$ mit $f(x') \geq f(x)$ ist $H(x', \vec{1}) \leq H_1^*(x)$, und für alle $x' \in \{0,1\}^n$ mit $f(x') \leq f(x)$ gilt $H(x', \vec{0}) \leq H_0^*(x)$.*

Beweis: Zum Beweis der ersten Aussage ziehen wir $i := n - H_1^*(x)$, $i \in \{0,\ldots,n\}$, heran. Da $f(x') \geq f(x)$, folgt aus der Definition des maximal erreichbaren Hammingabstandes, dass $f(x') \geq \sum_{j=1}^{i} w_j + \tau$ ist. Wegen $w_1 \geq \cdots \geq w_n$ müssen mindestens i Bits von x' Einsen sein, d. h. $H(x', \vec{1}) \leq n - i = H_1^*(x)$.

Um die zweite Aussage zu zeigen, nutzen wir $i := n - H_0^*(x)$. Analog zum vorigen Absatz folgt $f(x') \leq w - \sum_{j=1}^{i} w_j + \tau$, also sind wegen der Ordnung der Gewichte mindestens i Bits von x' Nullen, sodass $H(x', \vec{0}) \leq n - i = H_0^*(x)$. □

Hat der (1+1)-EA während der Optimierung des Quadrates einer linearen Funktion also einen Bitstring $x \in P(f)$ mit $H_1^*(x) = k$ erreicht, wird für weitere im Verlaufe des Algorithmus erzeugte $x' \in P(f)$ stets $H_1^*(x') \leq k$ zutreffen. Damit gibt uns das Maß an, dass die Mutation von bestimmten maximal k Bits zum Erreichen von $\vec{1}$ genügt; offensichtlich gelten analoge Aussagen für $\vec{0}$. Zur weiteren Vorbereitung einer Aussage über die Laufzeit des (1+1)-EA auf Quadraten linearer Funktionen stellen wir im folgenden Lemma fest, dass eine Mutation eines eventuell genau bestimmten Bits den maximal erreichbaren Hammingabstand zu $\vec{0}$ oder $\vec{1}$ verringert.

Lemma 3.11 *Sei f eine lineare Funktion. Für alle $x \in \{0,1\}^n \setminus \{\vec{0}, \vec{1}\}$ existieren ein $x' \in \{0,1\}^n$ mit $H(x,x') = 1$, $f(x') > f(x)$ und $H_1^*(x') \leq H_1^*(x) - 1$ sowie ein $x^* \in \{0,1\}^n$ mit $H(x,x^*) = 1$, $f(x^*) < f(x)$ und $H_0^*(x^*) \leq H_0^*(x) - 1$.*

Beweis: Für ein $x \in \{0,1\}^n$ setzen wir $o := n - H_1^*(x)$ und $z := n - H_0^*(x)$. Aus der Definition des maximal erreichbaren Hammingabstandes folgt $\sum_{i=1}^{o+1} w_i + \tau > f(x) \geq \sum_{i=1}^{o} w_i + \tau$. Wären alle Positionen $k \in \{1, \ldots, o+1\}$ von x mit 1 belegt, gälte, da $w_i \in \mathbb{N}$, die Ungleichung $f(x) \geq \sum_{i=1}^{o+1} w_i + \tau$. Also existiert ein $k \in \{1, \ldots, o+1\}$ mit $x_k = 0$. Durch eine Mutation an der Position k wird der Funktionswert von f um mindestens w_{o+1} erhöht (denn $w_1 \geq \cdots \geq w_n$). Mittels der Definitionen $x'_j := x_j$ für $j \in \{1, \ldots, n\} \setminus \{k\}$ und $x'_k := \bar{x}_k$ ist daher ein x' mit den Eigenschaften $H(x,x') = 1$, $f(x') \geq \sum_{i=1}^{o+1} w_i + \tau > f(x)$ und damit $H_1^*(x') < H_1^*(x)$ gefunden.
Den Beweis für die Existenz von x^* können wir mit vertauschten Rollen von Nullen und Einsen und den Ungleichungen $w - \sum_{i=1}^{z+1} w_i + \tau < f(x) \leq w - \sum_{i=1}^{z} w_i + \tau$ führen. Wären alle Positionen $k \in \{1, \ldots, z+1\}$ von x mit Nullen belegt, wäre dies ein Widerspruch zur Beziehung $f(x) > w - \sum_{i=1}^{z+1} w_i + \tau$. Wir können also einen Index von höchstens $z+1$ finden, an dem sich eine Komplementbildung eines Einsbits vornehmen lässt, um ein x^* mit den gewünschten Eigenschaften aus x zu konstruieren. □

Somit haben wir gezeigt, dass der maximal erreichbare Hammingabstand ein vernünftiges Maß ist, um den Fortschritt des (1+1)-EA bei der Optimierung der Quadrate linearer Funktionen zu messen. Mit den zurechtgelegten Hilfsmitteln zeigen wir nun eine Laufzeitschranke $O(n^2)$, da die vermutlich gültige Schranke $O(n \ln n)$ eines äußerst umfangreichen Beweises bedürfte.

Lemma 3.12 *Nach erwarteten $O(n^2)$ Schritten erreicht der (1+1)-EA auf dem Quadrat einer linearen Funktion f das globale Maximum in $\vec{1}$ oder den Vektor $\vec{0}$.*

Beweis: Wir unterteilen den Verlauf des Algorithmus in Phasen $i = 1, \ldots, 2n$ noch zu bestimmender Länge, zu deren Beginn wir jeweils für den aktuellen Bitstring x^i des (1+1)-EA anhand von

$$z_i := \begin{cases} H_0^*(x^i), & \text{wenn } x^i \in N(f), \\ z_{i-1}, & \text{wenn } x^i \in P(f), \end{cases} \quad \text{und} \quad o_i := \begin{cases} H_1^*(x^i), & \text{wenn } x^i \in P(f), \\ o_{i-1}, & \text{wenn } x^i \in N(f), \end{cases}$$

den maximalen erreichbaren Hammingabstand zum Nullstring oder zum Einsstring neu messen und während der Phasen i des Algorithmus das Paar (z_i, o_i) als Fortschrittsmaß in Bezug auf die Bitstrings $\vec{0}$ und $\vec{1}$ mitführen. Ferner setzen wir $z_0 := o_0 = n$.
Die erste Phase beginnt mit dem ersten Durchlauf der Schleife in Schritt 2 des (1+1)-EA; Phase i endet, unmittelbar bevor infolge einer Mutation ein Vektor x^{i+1} erzeugt wurde, für den $H_0^*(x^{i+1}) < z_i$, falls $x^{i+1} \in N(f)$, oder $H_1^*(x^{i+1}) < o_i$, falls $x^{i+1} \in P(f)$, zutrifft. Somit sinkt zu Beginn jeder Phase entweder der maximal erreichbare Hammingabstand zu $\vec{0}$ oder $\vec{1}$, d. h. nach höchstens $2n$ Phasen stimmt der aktuelle Bitstring des (1+1)-EA mit $\vec{0}$ oder $\vec{1}$ überein, wonach im Falle von $\vec{0}$ weitere Mutationen zu anderen Bitstrings mit gleichem oder besserem Funktionswert zwar möglich, aber zum Beweis des Lemmas irrelevant sind. Weil die Werte o_i bzw. z_i ggf. aktualisiert werden, wenn der aktuelle Bitstring aus $N(f)$ bzw. $P(f)$ stammt, gibt immer eine Komponente des Paares (z_i, o_i) den maximal erreichbaren Hammingabstand zu einem der Bitstrings $\vec{0}$ und $\vec{1}$ an, während die andere Komponente (o_i im Falle $x \in N(f)$ und umgekehrt) uninteressant ist.
Für jede Phase $i \in \{1, \ldots, 2n\}$, in der noch $z_i \neq 0$ und $o_i \neq 0$ gilt, bleibt nun zu zeigen, dass sie nach erwarteten $O(n)$ Schritten endet. Dieses folgt, da für jeden aktuellen Bitstring x während der Phase i nur $x' \in P(f)$ mit $H_1^*(x') \leq o_i$ (folgt aus dem Akzeptanzverhalten des (1+1)-EA und Lemma 3.10) sowie $x' \in N(f)$ mit $H_0^*(x') \leq z_i$ erreichbar sind. Daher genügt gemäß Lemma 3.11 stets die Mutation eines ausgewählten Bits im suboptimalen aktuellen Bitstring, um entweder z_i oder o_i zu verringern. Also sinkt nach erwarteter Zeit von höchstens en (siehe Lemma A.9) entweder z_i oder o_i. □

Daraus können wir im Fall $\tau = -w/2$ eine Folgerung ziehen.

Korollar 3.13 *Nach erwarteten $O(n^2)$ Schritten erreicht der (1+1)-EA auf dem Quadrat einer linearen Funktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$, $w_1 \geq \cdots \geq w_n$ und $\tau = -(\sum_{i=1}^{n} w_i)/2$ eines der globalen Maxima in $\vec{0}$ und $\vec{1}$.*

Unter den Bedingungen von Korollar 3.13 gilt für die erwartete Laufzeit vermutlich sogar eine Schranke von $O(n \ln n)$. Bei OneSqr$_{-n/2}$ war dies in Lemma 3.6 leicht zu sehen, doch bei den Quadraten allgemeiner linearer Funktionen mit $\tau = -w/2$ ist dies mindestens ebenso schwierig wie die allgemeine obere Schranke $O(n \ln n)$ für lineare Funktionen.
Bisher haben wir bewiesen, dass der (1+1)-EA auf den Quadraten linearer Funktionen effizient das globale Maximum in $\vec{1}$ oder den Vektor $\vec{0}$, der bei interessanten Werten von τ und n ein lokales Maximum ist, erreicht. Gleichzeitig ist der Fall $\tau = -w/2$ abgehandelt. Zu untersuchen bleiben die Quadrate linearer Funktionen mit $-w/2 < \tau < 0$. Da wir damit die Klasse linearer Funktionen einschränken, geben wir solchen linearen Funktionen einen Namen.

3.3.2 Potenziell schwierige Funktionen und Wahrscheinlichkeiten polynomieller Laufzeiten

Definition 3.7 *Eine lineare Funktion der Gestalt $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{N}$ für $i \in \{1, \ldots, n\}$, $w := \sum_{i=1}^{n} w_i$, $-w/2 < \tau < 0$ und $w_1 \geq \cdots \geq w_n$ heißt potenziell schwierig.*

Der Begriff „potenziell schwierig“ resultiert aus dem Verhalten des (1+1)-EA auf den Quadraten der Elemente dieser Klasse linearer Fitnessfunktionen, wie es z. B. in Abschnitt 3.2 deutlich wurde. Anhand der Funktion $\textsc{OneSqr}_\tau$ konnten wir das Entstehen exponentieller Laufzeiten in Abhängigkeit vom Parameter τ beobachten. Nun ist es nicht erstaunlich, dass bei allgemeinen linearen Funktionen bzw. der interessanten Unterklasse der potenziell schwierigen (linearen) Funktionen Werte von τ nahe bei $-w/2$ zu exponentiellen erwarteten Laufzeiten auf deren Quadraten führen.

Lemma 3.14 *Sei f eine potenziell schwierige Funktion. Ist die Differenz von τ und $-w/2$ so gering, dass $\forall x \in \{0,1\}^n \setminus \{\vec{0},\vec{1}\} : |f(x)| < |f(\vec{0})|$ gilt, benötigt der (1+1)-EA zur Optimierung von f^2 erwartete $2^{\Omega(n \log n)}$ Schritte.*

Beweis: Hat der (1+1)-EA den Vektor $\vec{0}$ zum aktuellen Bitstring, ist eine Mutation von n Bits zugleich erforderlich, um eine Mutation zu $\vec{1}$, dem einzigen Bitstring mit mindestens ebenso gutem Funktionswert unter f^2, zu vollführen. Wegen der Wahrscheinlichkeit n^{-n} für dieses Ereignis beträgt die erwartete Zeit dafür n^n. Zusammen mit der Wahrscheinlichkeit von 2^{-n} für einen Start in $\vec{0}$ trägt dies einen Summanden von mindestens $2^{-n+\Omega(n \log n)} = 2^{\Omega(n \log n)}$ zur erwarteten Laufzeit bei. □

Bemerkung 3.6 Es ist ohne weiteres möglich, den Parameter τ jeder beliebigen gegebenen potenziell schwierigen Funktion f so zu ändern, dass die Bedingung in Lemma 3.14 zutrifft. Wählen wir nämlich $\tau := -w/2 + \epsilon$ für $0 < \epsilon < 1/2$, so gilt

$$\left|f(\vec{1})\right| = \left|-\frac{w}{2} + w + \epsilon\right| = \frac{w}{2} + \epsilon \text{ und } \left|f(\vec{0})\right| = \left|-\frac{w}{2} + \epsilon\right| = \frac{w}{2} - \epsilon \text{ (da } w \geq 1),$$
$$\text{und daraus folgt } |f(\vec{1})| - |f(\vec{0})| = 2\epsilon < 1, \text{ also } |f(\vec{0})| > |f(\vec{1})| - 1.$$

Da alle Gewichte natürliche Zahlen sind, gilt für alle $x \in P(f) \setminus \{\vec{1}\}$ die Ungleichung $f(x) \leq f(\vec{1}) - 1$, d. h. für die Beträge der Funktionswerte folgt $|f(x)| \leq |f(\vec{1})| - 1 < |f(\vec{0})|$; für $x \in N(f) \setminus \{\vec{0}\}$ ist $|f(x)| < |f(\vec{0})|$ ohnehin klar. Wir erhalten somit für $n > 1$ (vgl. Definition 3.2) ein lokales Maximum von f^2 in $\vec{0}$, das einen Hammingabstand von n zum globalen Maximum besitzt.

Trotz exponentieller Laufzeiten wollen wir jetzt zeigen, dass potenziell schwierige Funktionen tatsächlich nie „absolut schwierig“ sind, d. h. dass trotz möglicherweise exponentieller Werte für die erwartete Laufzeit (auf den Quadraten) mit einer durch eine Konstante nach unten beschränkten Wahrscheinlichkeit polynomielle Laufzeiten eintreten. Einen ersten Hinweis darauf erhalten wir aus dem folgenden Lemma.

Lemma 3.15 *Sei f eine potenziell schwierige Funktion. Für die Zufallsvariable $X := \sum_{i=1}^{n} w_i x_i^s + \tau$, die den zufälligen Funktionswert des in Schritt 1 des (1+1)-EA initialisierten Vektors $x^s \in \{0,1\}^n$ angibt, gilt $E[X] = w/2 + \tau$.*

Beweis: Wir drücken den Erwartungswert der Zufallsvariablen $X_i := w_i x_i^s$, $i \in \{1,\ldots,n\}$, durch $E[X_i] = w_i \cdot \text{Prob}(X_i = w_i) + 0 \cdot \text{Prob}(X_i = 0) = w_i/2$ aus. Aus der Linearität des Erwartungswertes (vgl. Lemma A.4) für $X = \sum_{i=1}^{n} X_i + \tau$ folgt die Behauptung. □

Da $\tau > -w/2$, bedeutet dies, dass der erwartete Funktionswert nach der Initialisierung positiv ist. Wir können zudem eine einfache Aussage über die Verteilung der obigen Zufallsvariablen X festhalten.

Lemma 3.16 *Sei f eine lineare Funktion mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $\tau \in \mathbb{R}$, $w_i \in \mathbb{N}$ für alle $i \in \{1, \ldots, n\}$, $w := \sum_{i=1}^{n} w_i$ und $w_1 \geq \cdots \geq w_n$. Für die Zufallsvariable $X := \sum_{i=1}^{n} w_i x_i^s + \tau$, die den zufälligen Funktionswert des in Schritt 1 des (1+1)-EA initialisierten Vektors $x^s \in \{0,1\}^n$ angibt, gilt* $\text{Prob}(X \geq w/2 + \tau) \geq 1/2$.

Beweis: Wir zählen alle 2^{n-1} paarweise disjunkten Mengen $\{x, \bar{x}\}$, gebildet aus Vektoren $x \in \{0,1\}^n$ und ihrem Komplement, in einer beliebigen Reihenfolge auf. Aus

$$\sum_{i=1}^{n} w_i x_i < w/2 \;\Rightarrow\; w - \sum_{i=1}^{n} w_i x_i > w/2 \;\Rightarrow\; \sum_{i=1}^{n} w_i (1 - x_i) > w/2 \;\Rightarrow\; \sum_{i=1}^{n} w_i \bar{x}_i \geq w/2$$

erhalten wir $\sum_{i=1}^{n} w_i x_i + \tau < w/2 + \tau \;\Rightarrow\; \sum_{i=1}^{n} w_i \bar{x}_i + \tau \geq w/2 + \tau$. Damit liefert je mindestens ein Element der 2^{n-1} zweielementigen Mengen und somit die mindestens die Hälfte der 2^n Vektoren unter f einen Funktionswert von mindestens $w/2 + \tau$. □

Korollar 3.17 *Sei f eine potenziell schwierige Funktion. Der in Schritt 1 des (1+1)-EA initialisierte Vektor $x^s \in \{0,1\}^n$ gehört mit einer Wahrscheinlichkeit von mindestens* $1/2$ *der Menge $P(f)$ an.*

Beweis: Nach Lemma 3.16 gilt mit einer Wahrscheinlichkeit von mindestens $1/2$ die Ungleichung $f(x^s) \geq w/2 + \tau$. Da $\tau > -w/2$ aufgrund der Definition potenziell schwieriger Funktionen, tritt $f(x^s) \geq 0$ mit einer Wahrscheinlichkeit von mindestens $1/2$ ein. □

Bemerkung 3.7 In den in diesem Abschnitt beschriebenen Lemmata 3.16, 3.20, 3.21 und 3.22 machen wir statt über potenziell schwierige Funktionen Aussagen über die Obermenge alle linearen Funktionen, die wir in der üblichen „Normalform“ mit monoton fallenden Gewichten etc. annehmen. Dies hat technische Gründe, die bei der Beweisführung in Satz 3.24, auf den wir hinarbeiten, relevant werden.

Nun liegt die Vermutung nahe, dass der (1+1)-EA auf dem Quadrat einer beliebigen potenziell schwierigen Funktionen wie bei der Funktion DISTANCE mit einer Wahrscheinlichkeit nahe bei $1/2$ das globale Maximum $\vec{1}$ in $O(n \ln n)$ Schritten erreicht. Im Beweis, dass dies bei DISTANCE zutrifft (siehe [DJW98a]), macht man sich die Gleichheit aller Gewichte zunutze und kann unter einer Annahme, die mit Wahrscheinlichkeit $1/2 - o(1)$ zutrifft, zeigen, dass die Wahrscheinlichkeit, im Laufe des Algorithmus aus einem Vektor, auf dem die zugrunde liegende lineare Funktion einen positiven Wert annimmt, durch Mutationen einen Vektor, auf dem die lineare Funktion einen negativen Wert annimmt, zu erzeugen, exponentiell klein ist. Leider kann dieser Trick bei der Betrachtung der Quadrate potenziell schwieriger linearer Funktionen im Allgemeinen nicht angewandt werden, da ein Bit ein so großes Gewicht haben kann, dass ein 1-0-Flipp dieses Bits selbst bei einem Hammingabstand von 1 zum Vektor $\vec{1}$ einen besseren Funktionswert des Quadrates erzeugt.

Beispiel 3.2 Die potenziell schwierige Funktion $f(x) = \sum_{i=1}^{n} 2^{n-i}x_i - 2^{n-1} + 3/4$ entspricht der Funktion BinVal aus [DJW98a], zu der $\tau = -w/2 + 1/4$ addiert wird. Hier gilt

$$|f(1,0,1,\ldots,1)| = |2^{n-2} - 1/4| < |-2^{n-2} - 1/4| = |f(0,0,1,\ldots,1)|.$$

Der mit der Wahrscheinlichkeit $1/n$ eintretende Flipp von x_1 ist zudem in $\Theta(n \ln n)$ Schritten absolut nicht auszuschließen.

Hier können wir mit einer Annahme über die beiden „gewichtigsten“ Bits allerdings eine Mutation von $P(f)$ nach $N(f)$ unwahrscheinlich genug machen.

Lemma 3.18 *Mit einer Wahrscheinlichkeit von $1/4 - \epsilon$ für ein beliebig kleines $\epsilon > 0$ optimiert der (1+1)-EA das Quadrat der in Beispiel 3.2 definierten Funktion f in $O(n \ln n)$ Schritten.*

Beweis: Mit Wahrscheinlichkeit $1/4$ werden die Bits x_1 und x_2 mit 1 initialisiert. Wir rechnen nach, dass $(1,1,*,\ldots,*) \in P(f)$ sowie

$$|f(1,1,*,\ldots,*)| > |f(1,0,*,\ldots,*)| \quad \text{und} \quad |f(1,1,*,\ldots,*)| > |f(0,1,*,\ldots,*)|$$

gelten, wenn die mit $*$ bezeichneten Bits beliebig belegt werden. Die Wahrscheinlichkeit, in $\lceil cc'n \ln n \rceil$ Schritten, $c, c' \in \mathbb{R}^{>0}$, mindestens einmal x_1 und x_2 zugleich zu mutieren, beträgt höchstens $\lceil cc'n \ln n \rceil / n^2$ und konvergiert gegen null. Wie üblich, wenden wir die Markoffungleichung an, um die Wahrscheinlichkeit, nicht in $\lceil cc'n \ln n \rceil$ Schritten das globale Maximum der linearen Funktion f und somit von f^2 zu finden, durch $1/c$ nach oben zu beschränken und damit die Gesamtwahrscheinlichkeit $1/4 - 1/c - o(1)$ für eine Optimierung in $O(n \ln n)$ Schritten für genügend großes n auf mindestens $1/4 - \epsilon$ zu bringen. □

Diese Technik scheint im Allgemeinen aber auch nicht zu funktionieren. Bei der obigen Funktion f nutzen wir die Besonderheit, dass $\forall i \in \{1,\ldots,n-1\} : w_i > \sum_{j=i+1}^{n} w_j$ gilt und außerdem $w_1 + w_2 > (3/4) \cdot w$ ist, d. h. $f(1,1,*,\ldots,*) > w/4$. Dies hat zur Folge, dass nach Eintreten des Ereignisses $x_1^s = x_2^s = 1$ um Erreichen eines Bitstrings aus $N(f)$ mit betragsmäßig mindestens ebenso hohem Funktionswert zwingend der Funktionswert der linearen Funktion um mehr als $w/2$ verringert werden muss, wozu ein Flipp von x_1 noch nicht ausreicht, sondern $x_1 = x_2 = 0$ erforderlich ist. Auf die folgende Funktion lässt sich dieser Gedanke aber nicht übertragen.

Definition 3.8 *Die Funktion* $\mathrm{FW} : \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\mathrm{FW}(x) := \sum_{i=1}^{n} (n - i + 1) \cdot x_i - \frac{n^2 + n}{4} + \frac{1}{4}.$$

Soll der (1+1)-EA das Quadrat dieser Fitnessfunktion[2] optimieren, vereitelt das Ereignis $x_1^s = x_2^s = 1$ noch lange nicht die Akzeptanz einer Mutation, die höchstens vier Einsbits

[2] FW steht für „falling weights“.

mit Indizes aus 3 bis etwa $n/2$ zugleich flippt, selbst wenn die Bits $x_3^s, \ldots, x_n^s$ derart initialisiert wurden, dass sie zusammen ihren erwarteten Anteil $\left(\sum_{i=3}^n w_i\right)/2$ zum Funktionswert beitragen. Eine Mutation von 4 Bits innerhalb linear vieler Bits tritt jedoch selbst in einem Schritt mit konstanter Wahrscheinlichkeit ein (siehe Lemma A.12).
Möchten wir erreichen, dass eine Mutation von $\omega(1)$ Bits notwendig ist, um von $P(\mathrm{FW})$ nach $N(\mathrm{FW})$ zu gelangen, stellen wir fest, dass der bei der Funktion DISTANCE eingesetzte Trick (vgl. [DJW98a]) hier zwar Erfolg versprechend erscheint, aber ein Problem mit sich bringt. Die Annahme, in der Initialisierung würden $n/2 + n^{1/4}$ der n Bits gleichverteilt mit 1 initialisiert, erhöht zwar den erwarteten Beitrag von $\sum_{i=1}^n w_i x_i^s$ nach der Initialisierung von $w/2$ auf $w/2 + (1/2)(n^{5/4} + n^{1/4})$, da unter dieser Annahme jedes Bit mit Wahrscheinlichkeit $1/2 + n^{-3/4}$ mit 1 initialisiert wird, doch es ist nicht klar, mit welcher Wahrscheinlichkeit dieser Erwartungswert tatsächlich erreicht oder überschritten wird. Stattdessen beschränken wir uns wiederum auf die Wahrscheinlichkeit 1/4, indem wir die Menge der Gewichte zerlegen, zwei Annahmen treffen und die erwähnte Technik aus [DJW98a] abwandeln.

Lemma 3.19 *Mit einer Wahrscheinlichkeit von mindestens* $1/4-\epsilon$ *für ein beliebig kleines* $\epsilon > 0$ *optimiert der (1+1)-EA das Quadrat der Funktion* FW *in* $O(n \ln n)$ *Schritten.*

Beweis: O. B. d. A. ist n eine Quadratzahl und gerade. Wir betrachten die Gewichte $w_1, \ldots, w_{\sqrt{n}}$ mit $h := \sum_{i=1}^{\sqrt{n}} w_i$ sowie $w_{\sqrt{n}+1}, \ldots, w_n$ mit $l := \sum_{i=\sqrt{n}+1}^n w_i$. Der initialisierte Bitstring $x^s \in \{0,1\}^n$ besitzt mit Wahrscheinlichkeit von mindestens 1/2 die Eigenschaft $\sum_{i=\sqrt{n}+1}^n w_i x_i^s \geq l/2$; dies folgt aus Lemma 3.16, wenn wir die von den erwähnten „hinteren“ Gewichten induzierte lineare Funktion $\sum_{i=1}^{n-\sqrt{n}} w_i' x_i$ mit $w_1' := w_{\sqrt{n}+1}, \ldots, w_{n-\sqrt{n}}' := w_n$ betrachten. Unabhängig davon tritt in den „vorderen“ Positionen mit Wahrscheinlichkeit $1/2 - o(1)$ das Ereignis $\sum_{i=1}^{\sqrt{n}} w_i x_i^s \geq h/2 + d$, wobei $d := n^{9/8} - n^{5/8} - (n+\sqrt{n})/4$, ein. Um dies zu zeigen, weisen wir nach, dass der initialisierte Bitstring x^s mit einer Wahrscheinlichkeit von $1/2 - o(1)$ mindestens $n/2 + n^{1/4}$ Einsen enthält. Wenn wir bei der Anwendung der Stirlingformel n so groß wählen, dass der durch $1 + 1/(12n) + O(n^{-2})$ darstellbare relative Fehler im Ausdruck $n! \approx \sqrt{2\pi} e^{-n} n^{n+1/2}$ geringer als $\sqrt{\pi/2}$ wird, sehen wir, dass es höchstens (vgl. [GKP89], Übungsaufgabe 5.60)

$$\binom{n}{n/2} \leq \frac{(\sqrt{2\pi}e^{-n}n^{n+1/2})\sqrt{\pi/2}}{(\sqrt{2\pi}e^{-n/2}(n/2)^{n/2+1/2})^2} = \frac{n^{n+1/2}\sqrt{\pi/2}}{\sqrt{2\pi}(n/2)^{n+1/2}(n/2)^{1/2}} = \frac{2^{n+1}n^{-1/2}}{\sqrt{4}} = 2^n n^{-1/2}$$

Vektoren $x \in \{0,1\}^n$ mit $n/2$ Einsen gibt, indem wir den Ausdruck $n!$ nach oben und $(n/2)!$ nach unten abschätzen. Für alle anderen $k \in \{0, \ldots, n\}$ gibt es also ebenfalls höchstens $2^n n^{-1/2}$ Vektoren mit k Einsen, denn der Binomialkoeffizient $\binom{n}{k}$ nimmt für $k = n/2$ sein Maximum an. Es besitzen damit höchstens $2^n n^{-1/4}$ Vektoren mindestens $n/2$ und weniger als $n/2 + n^{1/4}$ Einsen, d. h. es verbleiben $(1/2 - o(1)) \cdot 2^n$ Vektoren aus $\{0,1\}^n$ mit mindestens $n/2 + n^{1/4}$ Einsen.
Angewandt auf die Bits $x_1^s, \ldots, x_{\sqrt{n}}^s$ heißt dies, dass mit Wahrscheinlichkeit $1/2 - o(1)$ mindestens $\sqrt{n}/2 + n^{1/8}$ von diesen nach der Initialisierung mit 1 belegt sind und daher

mindestens $(\sqrt{n}/2+n^{1/8})(n-\sqrt{n}) = (n^{3/2}-n)/2+n^{9/8}-n^{5/8}$ zum Funktionswert beitragen, indem wir das geringstwertige Gewicht in diesen Positionen mit $n-\sqrt{n}$ nach unten abschätzen. Die Differenz zu $h/2 = \frac{1}{2}\sum_{i=1}^{\sqrt{n}}(n-i+1) = n^{3/2}/2-n/4+\sqrt{n}/4$ ist genau der oben erwähnte Wert d.
Weil alle Positionen von x^s unabhängig initialisiert werden, beträgt der Wert der Summe $\sum_{i=1}^{n} w_i x_i^s$ mit Wahrscheinlichkeit von mindestens $1/4-o(1)$ mindestens $(h+l)/2+d = w/2+d$. Damit müssen $\Omega(n^{1/8})$ Bits, deren zugeordnetes Gewicht höchstens n beträgt, zugleich flippen, um den Funktionswert der (linearen) Funktion FW um $\Omega(n^{9/8})$ zu vermindern und einen Bitstring aus $N(\mathrm{FW})$ mit einem betragsmäßig mindestens ebenso großen Funktionswert zu erzeugen (notwendige Bedingung für die Akzeptanz einer solchen Mutation bei der Optimierung des Quadrates). Die Wahrscheinlichkeit für dieses Ereignis von höchstens $1/(\Omega(n^{1/8}))!$ konvergiert auch nach einer Multiplikation mit der Laufzeit $\lceil cc'n\ln n\rceil$, $c, c' \in \mathbb{R}^+$, gegen null (vgl. Lemma A.11), woraus nach der üblichen Anwendung der Markoffungleichung die Aussage des Lemmas folgt. □

Wir wollen diesen Ansatz verallgemeinern und auf die Quadrate beliebiger potenziell schwieriger Funktionen anwenden. Daher zeigen wir im zentralen Hilfssatz dieses Abschnittes (Lemma 3.21) eine gegenüber der im Beweis von Lemma 3.19 erläuterten Technik von [DJW98a] verallgemeinerte Aussage. Damit stellen wir dann fest, dass der Funktionswert einer potenziell schwierigen Funktion unmittelbar nach der Initialisierung mit einer Wahrscheinlichkeit nahe bei 1/2 bereits $w/2+\tau$ um einen gewissen Wert übersteigt, d. h. dass der das Quadrat einer potenziell schwierigen Funktion f optimierende (1+1)-EA in der Initialisierung einen bereits „deutlich“ in $P(f)$ liegenden Bitstring x^s erzeugt, für den eine Mutation zu einem $x' \in N(f)$ unwahrscheinlich wird. Zur Vorbereitung des Beweises von Lemma 3.21 greifen wir in einem weiteren Hilfssatz Ideen aus Lemma 3.16 nochmals auf.

Lemma 3.20 *Sei f eine lineare Funktion mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $\tau \in \mathbb{R}$, $w_i \in \mathbb{N}$ für alle $i \in \{1,\ldots,n\}$, $w := \sum_{i=1}^{n} w_i$ und $w_1 \geq \cdots \geq w_n$. Es seien außerdem die Mengen G_k, $k \in \{0,\ldots,n\}$, definiert als*

$$G_k := \{x \in \{0,1\}^n | x_1 + \cdots + x_n = k \wedge f(x) \geq w/2+\tau\}.$$

Dann gilt für alle $k \in \{0,\ldots,n\}$: $\#G_k = j \Rightarrow \#G_{n-k} \geq \binom{n}{k} - j$.

Beweis: Die Menge G_k umfasst alle Vektoren $x \in \{0,1\}^n$ mit k Einsen, für die $f(x) \geq w/2+\tau$ zutrifft. Besitzen genau j Vektoren $x \in G_k$ die Eigenschaft $f(x) \geq w/2+\tau$, also $\sum_{i=1}^{n} w_i x_i \geq w/2$, verbleiben in der Menge der Vektoren mit k Einsen genau $\binom{n}{k} - j$ Vektoren x' mit $\sum_{i=1}^{n} w_i x'_i < w/2$. Für deren Komplemente gilt $\sum_{i=1}^{n} w_i \bar{x}'_i \geq w/2$ und damit $f(\bar{x}) \geq w/2+\tau$, und dies sind $\binom{n}{k} - j$ Vektoren mit $n-k$ Einsen. □

Lemma 3.21 *Sei f eine lineare Funktion mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $\tau \in \mathbb{R}$, $w_i \in \mathbb{N}$ für alle $i \in \{1,\ldots,n\}$, $w := \sum_{i=1}^{n} w_i$ und $w_1 \geq \cdots \geq w_n$. Sei weiterhin $d \in \mathbb{N}$ eine natürliche Zahl. Dann existieren mindestens $(1/2 - 2dn^{-1/2})2^n$ verschiedene $x \in \{0,1\}^n$ mit $f(x) \geq w/2+\tau+d$.*

Beweis: Wir zerlegen den Definitionsbereich von f in $n+1$ Gruppen, nämlich die $n+1$ Mengen S_k, $k \in \{0,\ldots,n\}$, von Vektoren, die $\binom{n}{k}$ Einsen enthalten. Für solch eine Menge bezeichnet die rationale Zahl $q_k \in [0,1]$ den Wert $q_k := \#\{x \in S_k | f(x) \geq w/2+\tau\}/\#S_k$, d. i. der Anteil der Vektoren der Menge S_k, die einen Funktionswert von mindestens $w/2+\tau$ liefern. Dabei ist z. B. $q_0 = 0$ sowie $q_n = 1$; aus Lemma 3.20 folgt die wesentliche Beziehung $q_k \geq 1 - q_{n-k}$. Darüber hinaus benötigen wir die Werte $q'_k \in [0,1]$ mit $q'_k := \#\{x \in S_k | f(x) \geq w/2+\tau+d\}/\#S_k$, für die wir die Eigenschaft $q'_k \geq q_{k-d}$ für $k \in \{d,\ldots,n\}$ mithilfe eines Gedankenexperimentes beweisen. Dazu stellen wir uns eine Urne mit n Gewichten $w_1,\ldots,w_n$ mit $w_i \in \mathbb{N}$ für $i \in \{1,\ldots,n\}$ und $w := \sum_{i=1}^{n} w_i$ vor, von denen wir $k-d$ Stück ohne Zurücklegen ziehen. Die Wahrscheinlichkeit für das Ereignis, dass die gezogenen Gewichte zusammen mindestens $w/2$ wiegen, ist eine untere Schranke für die Wahrscheinlichkeit, nach k gezogenen Gewichten das Gesamtgewicht $w/2+d$ zu erreichen, da jedes Gewicht mindestens 1 beträgt. Indem wir diese Wahrscheinlichkeiten als Anteile q_{k-d} und q'_k an den Mengen S_{k-d} bzw. S_k interpretieren, folgt die Behauptung. Mittels Abschätzen von $\sum_{k=0}^{n} q'_k\binom{n}{k}$ erhalten wir schließlich eine untere Schranke für die Anzahl der $x \in \{0,1\}^n$ (n o. B. d. A. gerade) mit $f(x) \geq w/2+\tau+d$ gemäß

$$\begin{aligned}
\sum_{k=0}^{n} q'_k\binom{n}{k} &\geq \sum_{k=d}^{n} q_{k-d}\binom{n}{k} \geq \sum_{k=2d}^{n/2} q_{k-d}\binom{n}{k} + \sum_{k=n/2+2d}^{n} q_{k-d}\binom{n}{k} \\
&\geq \sum_{k=2d}^{n/2} q_{k-d}\binom{n}{k-2d} + \sum_{k=n/2}^{n-2d} q_{k+d}\binom{n}{k+2d} \text{ (Monotonie von } \tbinom{n}{k} \text{ für } k \leq n/2) \\
&\geq \sum_{k=2d}^{n/2} q_{k-d}\binom{n}{k-2d} + \sum_{k=n/2}^{n-2d} (1-q_{n-k-d})\binom{n}{k+2d} \text{ (denn } q_{k+d} \geq 1-q_{n-k-d}) \\
&= \sum_{k=2d}^{n/2} q_{k-d}\binom{n}{k-2d} + \sum_{k=2d}^{n/2} (1-q_{k-d})\binom{n}{n-k+2d} \\
&= \sum_{k=2d}^{n/2} q_{k-d}\binom{n}{k-2d} + \sum_{k=2d}^{n/2} (1-q_{k-d})\binom{n}{k-2d} \text{ (Symmetrie von } \tbinom{n}{k}) \\
&= \sum_{k=2d}^{n/2} \binom{n}{k-2d} = \sum_{k=0}^{n/2-2d} \binom{n}{k},
\end{aligned}$$

wobei wir in der ersten und letzten mit Bemerkungen versehenen Zeile die Monotonie $k \leq k' \Leftrightarrow \binom{n}{k} \leq \binom{n}{k}$ sowie die Symmetrie $\binom{n}{k} = \binom{n}{n-k}$ für $k,k' \in \{0,\ldots,n/2\}$ ausgenutzt haben.
Dem Beweis von Lemma 3.19 entnehmen wir zuletzt die Abschätzung $\sum_{k=n/2-2d+1}^{n/2}\binom{n}{k} \leq 2dn^{-1/2}2^n$, sodass der Wert der letzten Summe in den obigen Abschätzungen aufgrund von $\sum_{k=0}^{n/2}\binom{n}{k} \geq \frac{1}{2}2^n$ mindestens $(1/2-2dn^{-1/2})2^n$ beträgt. □

Bemerkung 3.8 Indem wir im Beweis zu Lemma 3.21 zusätzlich die dort vernachlässigten

Summen

$$\sum_{k=n/2+1}^{n/2+d-1} q_{k-d}\binom{n}{k} + \sum_{k=n/2+d+1}^{n/2+2d-1} q_{k-d}\binom{n}{k} = \sum_{k=n/2-d+1}^{n/2-1} q_k\binom{n}{k+d} + \sum_{k=n/2-d+1}^{n/2-1} q_{n-k}\binom{n}{n-k+d}$$

betrachten und geeignet abschätzen sowie zusammenfassen, können wir sogar für mindestens $(1/2-(d+1)n^{-1/2})2^n$ Vektoren die gewünschten Eigenschaften nachweisen. Dies würde jedoch einen unübersichtlicheren Beweis erfordern und ist für uns irrelevant, da lediglich ein konstanter Faktor von etwa 2 gewonnen würde, der im Folgenden keine Rolle spielen wird.

Wir benötigen auch eine Variante von Lemma 3.21 mit einer stärkeren Aussage.

Lemma 3.22 *Sei f eine lineare Funktion mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $\tau \in \mathbb{R}$, $w_i \in \mathbb{N}$ für alle $i \in \{1,\ldots,n\}$, $w := \sum_{i=1}^{n} w_i$ und $w_1 \geq \cdots \geq w_n$. Sei weiterhin $d \in \mathbb{N}$ eine natürliche Zahl. Dann existieren mindestens $(1/2 - 2dn^{-1/2})2^n$ verschiedene $x \in \{0,1\}^n$ mit $f(x) \geq w/2 + \tau + \sum_{i=0}^{d-1} w_{n-i}$.*

Beweis: Der Beweis zu Lemma 3.21 kann bis auf das Gedankenexperiment übernommen werden. Dieses wandeln wir mit der Feststellung ab, dass die d zusätzlich gezogenen Gewichte sogar mindestens $\sum_{i=0}^{d-1} w_{n-i}$ aufgrund ihrer monoton fallenden Ordnung wiegen müssen. □

Da die Beweisstrategien beim Quadrat der oben behandelten Fitnessfunktion FW, der Funktion OneSqr und dem Quadrat der BinVal ähnelnden Funktion aus Beispiel 3.2 stark von den Gewichten der zugrunde liegenden potenziell schwierigen Funktion abhängen, klären wir vor der Formulierung des zentralen Satzes zunächst, welche Eigenschaften und Abhängigkeiten die Gewichte aufweisen können und legen damit weitere strukturelle Eigenschaften von potenziell schwierigen Funktionen offen. Hier wird es entscheidend, sich noch einmal vor Augen zu führen, dass wir bei der Analyse des asymptotischen Laufzeitverhaltens des (1+1)-EA auf Fitnessfunktionen im Grunde immer *Folgen* $(f_n)_{n\in\mathbb{N}}$, $f_n : \{0,1\}^n \to \mathbb{R}$, von Fitnessfunktionen betrachten. Besonders deutlich wird dies bei Funktionenfolgen, deren Gewichte von n abhängen, z. B. bei FW, deren Gewicht $w_1 = n$ für alle Glieder der Funktionenfolge verschieden ist. Auf der anderen Seite können alle Gewichte wie bei OneSqr in allen Gliedern der Funktionenfolge durch eine Konstante beschränkt sein, was sich als günstige Eigenschaft in Bezug auf die Wahrscheinlichkeit, in effizienter Zeit optimiert zu werden, erweisen wird. Aus Gründen der Übersichtlichkeit sprechen wir weiterhin statt von *Folgen von potenziell schwierigen Funktionen* einfach von *potenziell schwierigen Funktionen* und verstehen darunter immer die gesamte Funktionenfolge.

Lemma 3.23 *Sei f potenziell schwierige Funktion, deren Gewichte $w_1,\ldots,w_n$ für alle $n \in \mathbb{N}$ durch eine Konstante $c \in \mathbb{N}$ nach oben beschränkt sind. Der (1+1)-EA optimiert das Quadrat der Funktion f mit einer Wahrscheinlichkeit von mindestens $1/2 - \epsilon$ für ein beliebig kleines $\epsilon > 0$ in der Zeit $O(n \ln n)$.*

Beweis: Lemma 3.21 zufolge gilt für den Funktionswert des initialisierten Bitstrings x^s mit einer Wahrscheinlichkeit von mindestens $1/2 - 2n^{1/4-1/2} = 1/2 - o(1)$ die untere Schranke $f(x^s) \geq w/2 + \tau + n^{1/4}$. Da c konstant, können wir ein $N \in \mathbb{N}$ finden, sodass $N^{1/8} > c$ und damit $N^{1/4} > cN^{1/8}$ gilt. Indem wir $n \geq N$ voraussetzen, gilt dann $\forall n \geq N : cn^{1/8} < n^{1/4}$. Unter der mit einer Wahrscheinlichkeit von $1/2 - o(1)$ eintretenden Bedingung $f(x^s) \geq w/2 + \tau + n^{1/4}$ muss der (1+1)-EA während der Optimierung von f^2 dann für einen „Sprung“ von $P(f)$ nach $N(f)$ mehr als $\Omega(n^{1/8})$ Bits mit einem durch c nach oben beschränkten Gewicht bzw. sogar $\Omega(n^{1/4})$ Bits mit Gewicht 1 zugleich flippen, was mit gegen 1 konvergierender Wahrscheinlichkeit (vgl. Lemma A.11) in $O(n \ln n)$ Schritten nicht eintritt. Der übliche Einsatz der Markoffungleichung mit Berücksichtigung des Parameters ϵ beschließt den Beweis. □

Nachdem wir Folgen potenziell schwieriger Funktionen, in denen die Gewichte nicht mit n wachsen, gesondert behandelt haben und bei ihnen eine Wahrscheinlichkeit nahe $1/2$ für das gewünschte Laufzeitverhalten auf deren Quadraten nachgewiesen haben, zeigen wir nun eine allgemeine Aussage über das Verhalten des (1+1)-EA auf den Quadraten potenziell schwieriger Funktionen, deren Gewichte nicht durch eine Konstante für alle Glieder der Funktionenfolge nach oben beschränkt zu sein brauchen.

Satz 3.24 *Der (1+1)-EA optimiert das Quadrat einer potenziell schwierigen Funktion f mit einer Wahrscheinlichkeit von mindestens $1/8 - \epsilon$ für ein beliebig kleines $\epsilon > 0$ in der Zeit $O(n \ln n)$.*

Beweis: Wir möchten in ähnlicher Vorgehensweise wie bei den bisher betrachteten Beispielen potenziell schwieriger Funktionen nachweisen, dass der initialisierte Bitstring x^s mit einer durch eine Konstante nach unten beschränkten Wahrscheinlichkeit aus $P(f)$ stammt und darüber hinaus bereits insofern „weit“ in $P(f)$ liegt, als eine Mutation des aktuellen Bitstrings $x \in P(f)$ zu einem $x' \in N(f)$ mit einem betragsmäßig mindestens ebenso hohen Funktionswert $|f(x')| \geq |f(x)|$ – nur dann kann eine solche Mutation während der Optimierung des Quadrates akzeptiert werden – mit gegen 1 konvergierender Wahrscheinlichkeit während der Optimierung von f^2 in $O(n \ln n)$ Schritten nicht eintritt. Dann verhält sich der (1+1)-EA wie auf der linearen Funktion f, und die erwartete Laufzeit ist durch $c'n \ln n$ für ein konstantes $c' \in \mathbb{R}^{>0}$ nach oben beschränkt.

In der Beweisführung gehen wir pessimistisch davon aus, dass sogar $\tau = -w/2$ gilt, auch wenn dies in der Definition potenziell schwieriger Funktionen nicht vorgesehen ist, und Mutationen, die von einem $x \in P(f)$ mit $\sum_{i=1}^{n} w_i x_i = w/2 + \delta$, $\delta > 0$, zu einem x' mit $\sum_{i=1}^{n} w_i x'_i = w/2 - \delta$ führen, schädlich sind und akzeptiert werden. Eine solche Mutation, die den Funktionswert von f um mindestens 2δ verringert, nennen wir *schlecht.* Bei Betrachtungen von Werten $\delta > 0$ sprechen wir von einem *Überschuss* (in Bezug auf den erwarteten Wert von $\sum_{i=1}^{n} w_i x_i^s$ nach der Initialisierung). Diese Sichtweise hat den Vorteil, den Parameter τ bei nachfolgenden Abschätzungen nicht mitführen zu müssen, denn wir werden Aussagen über Teilsummen $\sum_{i=l}^{h} w_i x_i^s$, also lineare Funktionen ohne zusätzlichen

Summanden, für noch zu erläuternde $l, h \in \{1, \ldots, n\}$ treffen und dabei die Lemmata 3.16 und 3.22 nutzen.[3]

Da in Folgen potenziell schwieriger Funktionen für jedes $n \in \mathbb{N}$ andere Situationen gelten können, was die Werte der Gewichte und deren Verhältnis betrifft, müssen wir eine Fallunterscheidung anhand der Gewichte $w_1, \ldots, w_n$ für jedes betrachtete $n \in \mathbb{N}$ durchführen. Wir werden zeigen, dass in jedem Fall Annahmen über x^s und das Verhalten des (1+1)-EA während $cc'n \ln n$, Schritten, $c \in \mathbb{R}^{>0}$, getroffen werden können, die mit einer Wahrscheinlichkeit von mindestens $1/8 - o(1)$ eintreffen. Damit konvergieren die unteren Schranken der Wahrscheinlichkeiten für die noch zu beschreibenden Annahmen für wachsendes n gegen $1/8$ unabhängig von der Beschaffenheit der Gewichte bei dem in der Fallunterscheidung betrachteten $n \in \mathbb{N}$. Aus Gründen der Übersichtlichkeit verzichten wir in den folgenden Schlüssen auf Auf- und Abrundungen mittels Gaußklammern.

1. Fall: Wenn es ein $k \in \{1, \ldots, n\}$, k ist von n abhängig, mit der Eigenschaft

$$\sum_{i=0}^{k^{1/4}-1} w_{k-i} \geq (\log n) w_1$$

gibt, nutzen wir Ideen aus dem Beweis zu Lemma 3.19. Zunächst benötigen wir die Aussage $k \geq \log n$, die aus der monoton fallenden Ordnung der Gewichte $w_1, \ldots, w_k$ mit der Konsequenz $k^{1/4} \geq \log n$ folgt. Die Bits an den Positionen $1, \ldots, k$ der linearen Teilsumme $\sum_{i=1}^{k} w_i x_i$ werden gemäß Lemma 3.22 (setze $d := k^{1/4}$ und $\tau := 0$) mit einer Wahrscheinlichkeit von $1/2 - 2k^{1/4}k^{-1/2} = 1/2 - 1/\Omega((\log n)^{1/4})$ derart initialisiert, dass

$$\sum_{i=1}^{k} w_i x_i^s \geq \frac{1}{2} \sum_{i=1}^{k} w_i + \sum_{i=0}^{k^{1/4}-1} w_{k-i}$$

gilt. Mit Wahrscheinlichkeit von mindestens $1/2$ tritt gemäß Lemma 3.16 außerdem das Ereignis $\sum_{i=k+1}^{n} w_i x_i^s \geq \frac{1}{2} \sum_{i=k+1}^{n} w_i$ ein. Da diese Ereignisse disjunkte, d. h. unabhängig initialisierte Positionen im Bitstring betreffen, können wir die Wahrscheinlichkeiten der Einzelereignisse multiplizieren, um die Wahrscheinlichkeit der Konjunktion dieser Ereignisse zu erhalten. Folglich beträgt der Wert der Summe $\sum_{i=1}^{n} w_i x_i^s$ nach der Initialisierung mit einer Wahrscheinlichkeit von sogar mindestens $(1/2)(1/2 - 1/\Omega((\log n)^{1/4})) = 1/4 - o(1)$ bereits mindestens

$$\frac{1}{2} \sum_{i=1}^{k} w_i + \sum_{i=0}^{k^{1/4}-1} w_{k-i} + \frac{1}{2} \sum_{i=k+1}^{n} w_i = \frac{w}{2} + \sum_{i=0}^{k^{1/4}-1} w_{k-i} \geq \frac{w}{2} + (\log n) w_1.$$

Der eingangs beschriebene Überschuss δ beträgt mit Wahrscheinlichkeit von mindestens $1/4 - o(1)$ nach der Initialisierung also mindestens $(\log n) w_1$. Weil jedes Gewicht maximal w_1 beträgt, müssen für eine schlechte Mutation mindestens einmal während $cc'n \ln n$

[3] Da bei diesen Teilsummen kein zusätzlicher Summand $\tau < 0$ auftritt, mussten wir in den genannten Lemmata allgemeine lineare Funktionen berücksichtigen, vgl. Bemerkung 3.7.

Schritten mindestens $2 \log n$ Bits zugleich flippen. Dies passiert mit gegen 1 konvergierender Wahrscheinlichkeit in $cc'n \ln n$ Schritten nicht, denn die Wahrscheinlichkeit einer solchen Mutation ist durch $1/(2 \log n)! = 2^{-\Omega(\log n \log \log n)}$ nach oben beschränkt und konvergiert auch nach einer Multiplikation mit $cc'n \ln n$ gemäß $2^{-\Omega(\log n \log \log n)+O(\log n)} = 2^{-\Omega(\log n \log \log n)}$ gegen null. (Diese Berechnung ist in Lemma A.10 ausführlicher dargestellt.)

2. Fall: Es gibt kein k mit den im ersten Fall beschriebenen Eigenschaften, d. h.

$$\forall k \in \{1, \dots, n\} : \sum_{i=0}^{k^{1/4}-1} w_{k-i} < (\log n) w_1.$$

Für $k = (\log n)^8$ besagt die vorangehende Ungleichung, dass $(\log n)^{(1/4)\cdot 8} = (\log n)^2$ Gewichte mit Indizes aus $\{(\log n)^8 - (\log n)^2 + 1, \dots, (\log n)^8\}$ summiert noch nicht $(\log n) w_1$ ergeben. Aus $w_1 \geq \cdots \geq w_n$ folgt dann, dass ab der Position $t := (\log n)^8 - \log(n) + 1$ je $\log n$ Gewichte mit aufeinander folgenden Indizes und erst recht mit beliebigen Indizes von mindestens t addiert noch nicht w_1 erreichen.

Wir benutzen die Position t nun bei der „Konstruktion“ eines Überschusses nach der Initialisierung in Form einer Schranke. Zum einen tritt mit Wahrscheinlichkeit von $1/2$ das Ereignis $x_1^s = 1$ ein. Zum anderen gilt für die Positionen $2, \dots, t$ gemäß Lemma 3.22 (setze $d := (t-1)^{1/4}$ und $\tau := 0$)

$$\sum_{i=2}^{t} w_i x_i^s \geq \frac{1}{2} \sum_{i=2}^{t} w_i + \sum_{i=0}^{(t-1)^{1/4}} w_{t-i} \geq \frac{1}{2} \sum_{i=2}^{t} w_i + (t-1)^{1/4} w_t$$

mit einer Wahrscheinlichkeit von mindestens

$$\frac{1}{2} - 2(t-1)^{1/4}(t-1)^{-1/2} = \frac{1}{2} - 2(t-1)^{-1/4} = \frac{1}{2} - \frac{2}{((\log n)^8 - \log n)^{1/4}} = \frac{1}{2} - o(1).$$

Drittens tritt mit einer Wahrscheinlichkeit von mindestens $1/2$ gemäß Lemma 3.16 das Ereignis $\sum_{i=t+1}^{n} w_i x_i^s \geq \frac{1}{2} \sum_{i=t+1}^{n} w_i$ ein. Diese drei Ereignisse sind wie im ersten Fall wegen der Disjunktheit der betrachteten Positionen unabhängig. Setzen wir nun die drei (Un)gleichungen zusammen, folgt, dass

$$\sum_{i=1}^{n} w_i x_i^s \geq \frac{w}{2} + \frac{w_1}{2} + (t-1)^{1/4} w_t \qquad (*)$$

mit einer Wahrscheinlichkeit von mindestens $1/8 - o(1)$ zutrifft. Der Überschuss δ nimmt dann einen Wert von mindestens $w_1/2 + (t-1)^{1/4} w_t$ an.

Zu zeigen bleibt nun, dass bei Eintreten dieses Überschusses die Wahrscheinlichkeit mindestens einer schlechten Mutation in $cc'n \ln n$ Schritten gegen null konvergiert. Wir betrachten dazu Mutationen anhand der Schranke t. Es sind erstens Mutationen möglich, die nur Bits an Positionen mit Indizes von weniger als t ändern, zweitens Mutationen, die nur Bits mit

Indizes von mindestens t ändern, und drittens Mutationen, die sowohl Bits mit Indizes von weniger als t als auch Bits mit Indizes von mindestens t ändern.

Über die Bits mit Indizes von weniger als t wissen wir wegen $2\delta > w_1$, dass mindestens einmal in $cc'n \ln n$ Schritten mindestens zwei von ihnen gleichzeitig flippen müssen, um den Funktionswert von f um mindestens 2δ zu verringern und eine schlechte Mutation zu vollführen. Da $t \leq (\log n)^8$, beträgt die Wahrscheinlichkeit für dieses Ereignis in einem Schritt des (1+1)-EA maximal $\binom{(\log n)^8}{2}(1/n)^2 \leq (\log n)^{16}/n^2$; selbst nach einer Multiplikation mit $cc'n \ln n$ konvergiert dieser Ausdruck gegen 0.

Beschränkt auf die Bits mit Indizes von mindestens t haben wir festgestellt, dass die Gewichte an beliebigen $\log n$ Positionen summiert immer noch geringer als w_1 sind. Es müssen für eine Verringerung des Funktionswertes von f um mindestens 2δ, d. h. eine schlechte Mutation, wegen der Ungleichung $2\delta > w_1$ mehr als $\log n$ Bits zugleich flippen. Bereits im ersten Fall haben wir gezeigt, dass mit gegen 1 konvergierender Wahrscheinlichkeit in $cc'n \ln n$ Schritten sogar niemals $\log n$ Bits mit beliebigen Indizes gleichzeitig flippen.

Zuletzt betrachten wir die dritte Möglichkeit, eine schlechte Mutation zu vollführen, d. h. die Mutation von Bits mit Indizes von weniger als t und Bits mit Indizes von mindestens t. Wir wissen, dass die Wahrscheinlichkeit, in $cc'n \ln n$ Schritten mindestens einmal zwei oder mehr Bits mit Indizes von weniger als t zu mutieren, gegen null konvergiert. Also verbleiben die Mutationen, die genau ein Bit mit einem Index von weniger als t und weitere Bits (z. B. konstant viele) Bits mit Indizes von mindestens t flippen. Die Mutation des Bits mit einem Index von weniger als t kann maximal eine Senkung des Funktionswertes von f um w_1 bewirken und nur den Überschuss $2\delta_1$ für $\delta_1 := w_1/2$ überwinden. Da $w_1 \geq \cdots \geq w_n$, greift nun der zusätzliche Überschuss $\delta_2 := (t-1)^{1/4} w_t$ aus der Ungleichung $(*)$. Es müssten mindestens weitere $2(t-1)^{1/4} = \Omega((\log n)^2)$ Bits mit Indizes von mindestens t flippen, um den Funktionswert von f darüber hinaus um mindestens $2\delta_2 = 2(t-1)^{1/4} w_t$ zu verringern und insgesamt eine schlechte Mutation auszuführen, die den Funktionswert von f um mindestens $2\delta = 2\delta_1 + 2\delta_2$ verringert. Die Wahrscheinlichkeit für letzteres Ereignis konvergiert auch nach einer Multiplikation mit $cc'n \ln n$ gegen 0, was den zweiten Fall beschließt.

Gemäß dem Wert des Parameters ϵ setzen wir schließlich wie gewohnt die Markoffungleichung ein und wählen c genügend groß. □

In einem Korollar halten wir fest, was wir aufgrund unserer Beweistechnik außerdem gezeigt haben.

Korollar 3.25 *Sei f eine potenziell schwierige Funktion, deren Quadrat Eingabe für den (1+1)-EA ist. Mit einer Wahrscheinlichkeit von mindestens $1/8 - o(1)$ findet in $O(n \ln n)$ Schritten keine akzeptierte Mutation von einem $x \in P(f)$ zu einem $x' \in N(f)$ statt.*

Beweis: Folgt unmittelbar aus dem Beweis zu Satz 3.24. □

Am folgenden Beispiel sehen wir, dass beide Fälle der Fallunterscheidung im Beweis zu Satz 3.24 eintreten können.

Beispiel 3.3 Die Funktion

$$f(x_1,\ldots,x_n) = \begin{cases} \sum_{i=1}^{n} 2^{n-i}x_i - \frac{2^n-1}{2} + \frac{1}{4} & \text{für ungerade } n \\ \sum_{i=1}^{n}(n-i+1)x_i - \frac{n^2+n}{4} + \frac{1}{4} & \text{für gerade } n \end{cases}$$

entspricht für gerade n der Funktion FW, für ungerade n hingegen bis auf den passend gewählten Summanden $-2^{n-1}+1/2+1/4$ der Funktion BinVal. Bei der Betrachtung des Verhaltens des (1+1)-EA auf f^2 befinden wir uns für gerade n im ersten Fall des Beweises und nutzen die Lemmata 3.22 und 3.16, für ungerade n treffen wir die Annahmen des zweiten Falls über den initialisierten Bitstring x^s, indem wir für das höchstwertige Bit $x_1^s = 1$ annehmen und uns für die übrigen Bits darüber hinaus die Lemmata 3.22 und 3.16 zunutze machen.

Bemerkung 3.9 Im Grunde können wir glauben, dass der (1+1)-EA auf dem Quadrat einer beliebigen potenziell schwierigen Funktion f mit einer Wahrscheinlichkeit von sogar $1/2-\epsilon$ für ein beliebig kleines $\epsilon > 0$ in $O(n \ln n)$ Schritten zum globalen Maximum in $\vec{1}$ gelangt, ähnlich wie es bei Distance der Fall ist. Die Beweistechnik im vorstehenden Satz macht sich aber das äquivalente Verhalten auf dem Quadrat von f wie auf f selbst zunutze, das gegeben ist, solange alle aktuellen Bitstrings aus $P(f)$ stammen. Wir mussten daher zusätzliche Annahmen treffen, um den „Fehler“ einer Mutation von $P(f)$ nach $N(f)$ unwahrscheinlich zu machen und konnten deshalb lediglich eine Wahrscheinlichkeit nahe 1/8 beweisen. Beispiel 3.2 zusammen mit Lemma 3.18 liefert übrigens einen deutlichen Hinweis darauf, dass die Wahrscheinlichkeit für einen Start mit einem x^s in einer Teilmenge von $P(f)$, sodass für alle in dieser Menge liegenden aktuellen Bitstrings die Wahrscheinlichkeit einer Mutation zu einem $x' \in N(f)$ für wachsendes n gegen 0 konvergiert, bei der dort betrachteten Funktion nicht über 1/4 zu liegen scheint.

Bemerkung 3.10 Wir glauben, dass auf den Quadraten potenziell schwieriger Funktionen, deren Parameter τ sehr nahe bei $-w/2$ liegen, auch mit einer Wahrscheinlichkeit von mindestens $1/8-\epsilon$ für beliebig kleine $\epsilon > 0$ exponentielle Laufzeiten beobachtet werden können. Zum einen existiert nämlich bei den Quadraten potenziell schwieriger Funktionen f mit Werten von τ im Bereich $-w/2 < \tau < -w/2+1/2$ gemäß Bemerkung 3.6 ein lokales Maximum in $\vec{0}$, zu dessen Überwindung eine erwartete Zahl von $\Omega(n^n)$ Schritten erforderlich ist. Zum anderen liegt es nahe, dass duale Aussagen zu Lemma 3.22 etc. existieren, die untere Schranken für die Wahrscheinlichkeit des Ereignisses, dass $\sum_{i=1}^{n} w_i x_i^s \leq w/2-\delta$, $\delta > 0$, nach der Initialisierung gilt, angeben. Liegt x^s in $N(f)$, müsste der Wert der Summe $\sum_{i=1}^{n} w_i x_i$ wegen der Ungleichung $\tau - w/2 < 1/2$ um mindestens $2\delta - 1$ geändert werden, um von einem $x \in N(f)$ nach $P(f)$ zu springen. Der Beweis zu Satz 3.24 dürfte also mit kleinen Anpassungen geführt werden können und würde aussagen, dass der (1+1)-EA nach einem Start in $N(f)$ mit einer Wahrscheinlichkeit von mindestens $1/8-\epsilon$ während $O(n \ln n)$ Schritten keine Mutation nach $P(f)$ ausführt, sich also wie auf der Funktion $-f$ verhält und nach erwarteten $O(n \ln n)$ Schritten zum lokalen Maximum in $\vec{0}$ gelangt. Dies bedeutet eine exponentielle Wartezeit mit Wahrscheinlichkeit von mindestens $1/8-\epsilon$.

Wir ziehen aus Satz 3.24 einige Konsequenzen zur Komplexität der Optimierung der Quadrate potenziell schwieriger Funktionen. Während die erwartete Laufzeit des (1+1)-EA auf dem Quadrat einer gegebenen potenziell schwierigen Funktion f zwar nicht durch ein Polynom in n beschränkt werden kann, ist es hingegen beispielsweise möglich, $\epsilon = 1/72$ zu wählen und dann für genügend große n mit einer Wahrscheinlichkeit von mindestens $1/9$ eine Laufzeit von $O(n \ln n)$ zu beobachten. Lassen wir 9 unabhängige Instanzen des (1+1)-EA parallel je auf f^2 laufen – dies bezeichnet man als *Multistartstrategie*, vgl. z. B. [Müh91] –, optimiert im Erwartungswert mindestens eine Instanz die Funktion f^2 in $O(n \ln n)$ Schritten. Allgemeiner stellen wir zu $k \in \mathbb{N}$ unabhängigen Instanzen des (1+1)-EA Zufallsvariablen $X_1, \ldots, X_k$ mit dem Wertebereich $\{0, 1\}$ auf, wobei X_i den Wert 1 annimmt, wenn Instanz i in $O(n \ln n)$ Schritten das Optimum von f^2 findet. Für die Zufallsvariable $X := X_1 + \cdots + X_k$ mit $E[X] \geq k/9$ können wir mithilfe der Chenernoffschranke beispielsweise die Wahrscheinlichkeit, dass weniger als $k/18$ Instanzen in $O(n \ln n)$ Schritten das Optimum finden, in der Form c^k für eine Konstante $0 < c < 1$ nach oben abschätzen. Es ist also einfach, durch eine Erhöhung von k die Wahrscheinlichkeit, dass keine Instanz in $O(n \ln n)$ Schritten das Optimum von f^2 findet, beliebig klein (mit sogar exponentiell in k sinkenden Werten) zu machen. Bei n Instanzen (dies ist zugegebenermaßen in der Praxis schwer zu realisieren) erhalten wir sogar eine exponentiell in n gegen 0 konvergierende Wahrscheinlichkeit für das Ereignis, dass keine Instanz f^2 in $O(n \ln n)$ Schritten optimiert. Insgesamt sehen wir, dass die Quadrate potenziell schwieriger Funktionen für Varianten des (1+1)-EA, nämlich Multistartstrategien, nicht wirklich problematisch sind.

3.4 Folgerungen für beliebige Potenzen linearer Funktionen

Wir haben bei allen bisherigen Analysen der Quadrate linearer Funktionen f niemals die tatsächlichen Funktionswerte des Quadrates f^2 betrachten müssen, sondern haben mithilfe von Lemma 3.1 stets anhand der Beträge der Funktionswerte von f argumentieren können. Es scheint daher keinen großen Aufwand zu beinhalten, die Resultate dieses Kapitels in Form eines kleinen Exkurses auf beliebige (natürliche) Potenzen linearer Funktionen zu erweitern.

Zunächst halten wir fest, dass lineare Funktionen f mit $N(f) = \emptyset$ absolut uninteressant sind, da beliebige Potenzen f^k mit $f^k(x) := (f(x))^k$, $k \in \mathbb{N}$, für den (1+1)-EA von f selbst nicht zu unterscheiden sind. (Der Beweis zu Lemma 2.7 kann mit der Abbildung $z \mapsto z^k$, die auf $\mathbb{R}^+$ streng monoton ist, analog geführt werden.) Ebenso wenig spannend sind die Potenzen f^k, $k \in \mathbb{N}$, linearer Funktionen f mit $P(f) = \emptyset$. Bei diesen ist die Anpassung von Lemma 2.8 mittels der Abbildung $z \mapsto z^k$ für gerade k ebenso trivial wie zuvor; bei ungeraden k ist $z \mapsto z^k$ sogar auf ganz $\mathbb{R}$ streng monoton. Drittens folgt, dass auch die Negationen der Potenzen, d. h. Funktionen der Gestalt $-f^k$, für den (1+1)-EA von $-f$ oder f nicht zu unterscheiden sind, falls $N(f) = \emptyset$ oder $P(f) = \emptyset$ gilt. Insgesamt erhalten wir für beliebige Potenzen nicht negativer und nicht positiver linearer Funktionen sowie

für die Negationen dieser Potenzen die Schranken der erwarteten Laufzeit des (1+1)-EA unmittelbar aus der Schranke $O(n \ln n)$ für alle lineare Funktionen.
Zu untersuchen bleiben die Potenzen f^k, $k \in \mathbb{N}$, linearer Funktionen f mit der Eigenschaft $N(f) \neq \emptyset \neq P(f)$. Hier ist eine genauere Unterscheidung zwischen geraden und ungeraden Werten von k erforderlich.

Lemma 3.26 *Sei f eine lineare Funktion mit $N(f) \neq \emptyset \neq P(f)$ und $k \in \mathbb{N}$ eine ungerade Zahl. Der (1+1)-EA optimiert die Funktion f^k in erwarteter Zeit $O(n \ln n)$.*

Beweis: Für alle $x, x' \in \{0,1\}^n$ gilt $f^k(x) \geq f^k(x') \Leftrightarrow f(x) \geq f(x')$, da $z \mapsto z^k$ auf $\mathbb{R}$ wegen ungeradem k einen Ordnungsisomorphismus darstellt. Der (1+1)-EA kann f und f^k also nicht unterscheiden. □

Korollar 3.27 *Sei f eine lineare Funktion mit $N(f) \neq \emptyset \neq P(f)$ und $k \in \mathbb{N}$ eine ungerade Zahl. Der (1+1)-EA optimiert die Funktion $-f^k$ in erwarteter Zeit $O(n \ln n)$.*

Beweis: Da $(-z)^k = -z^k$ für ungerade k und beliebige $z \in \mathbb{R}$ gilt, folgt mit Lemma 3.26, dass $-f^k$ und $-f$ vom (1+1)-EA nicht unterschieden werden können. □

Zur Betrachtung verbleiben für gerade k die Potenzen f^k linearer Funktionen sowie deren Negationen $-f^k$. Die erste Klasse ist für den (1+1)-EA ebenso schwierig wie die Klasse der Quadrate linearer Funktionen, da Satz 3.24 bzw. Korollar 3.13 einfach umformuliert werden dürfen.

Korollar 3.28 *Sei f eine lineare Funktion mit $N(f) \neq \emptyset \neq P(f)$ und $k \in \mathbb{N}$ eine gerade Zahl. Besitzt f^k ein eindeutiges globales Maximum, optimiert der (1+1)-EA die Funktion f^k mit einer Wahrscheinlichkeit von mindestens $1/8 - \epsilon$ für ein beliebig kleines $\epsilon > 0$ in der Zeit $O(n \ln n)$. Sonst besitzt f^k zwei globale Maxima, und der (1+1)-EA optimiert f^k in erwarteter Zeit von höchstens $O(n^2)$.*

Beweis: Für alle $x, x' \in \{0,1\}^n$ und gerade $k \in \mathbb{N}$ gilt

$$f^k(x) \geq f^k(x') \Leftrightarrow |f(x)| \geq |f(x')|, \text{ also } f^k(x) \geq f^k(x') \Leftrightarrow f^2(x) \geq f^2(x'),$$

und f^k und f^2 sind somit für den (1+1)-EA äquivalent. Wir ersetzen f^k durch f^2. Damit können wir sämtliche Ergebnisse über die Quadrate linearer Funktionen anwenden (z. B. die Transformation nach Lemma 3.5) und insbesondere die Aussagen von Satz 3.24 sowie Korollar 3.13 übernehmen. □

Ein wenig Arbeit erfordert eine Aussage über das Verhalten des (1+1)-EA auf den Negationen $-f^k$ der Potenzen linearer Funktionen f für gerade k, weshalb wir in Abschnitt 2.2.2 noch nicht näher darauf eingehen konnten. Wir haben aber mit der Beweisführung zu Lemma 3.12 geeignete Techniken vorgestellt, mit denen es sich nachweisen lässt, dass derartige Funktionen $-f^k$ für den (1+1)-EA nicht schwierig sind.

Lemma 3.29 *Sei f eine lineare Funktion mit $N(f) \neq \emptyset \neq P(f)$ und $k \in \mathbb{N}$ eine gerade Zahl. Der (1+1)-EA optimiert die Funktion $-f^k$ in erwarteter Zeit $O(n^2)$.*

Beweis: Da $-f^k$ zu $-f^2$ für den (1+1)-EA äquivalent ist, nehmen wir o. B. d. A. $k = 2$ an und wenden Lemma 2.6 an, das besagt, dass wir f in der Form $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_1 \geq \cdots \geq w_n$, $w_i \in \mathbb{N}$ für $i \in \{1, \ldots, n\}$ und $\tau \in \mathbb{R}$ voraussetzen dürfen.
Sei $m := \min\{f^2(x) | x \in \{0,1\}^n\}$ der minimale Funktionswert der Funktion f^2 und $M := \{x \in \{0,1\}^n | f^2(x) = m\}$ die Menge aller Punke, an denen die Funktion f^2 ihren minimalen Funktionswert annimmt. Es gilt für alle $x \in \{0,1\}^n \setminus M$ und alle $x_m \in M$ wegen der Negation die Ungleichung $-f^2(x) < -f^2(x_m)$. Die Menge $\{0,1\}^n \setminus M$ suboptimaler Bitstrings zerfällt in die disjunkten Mengen $N' := N(f) \setminus M$ und $P' := P(f) \setminus M$. Wir stellen ähnlich wie in Lemma 3.12 je ein Fortschrittsmaß auf, das für suboptimale aktuelle Bitstrings $x \in N'$ bzw. $x \in P'$ zum Zuge kommt. Weil für alle $x, x' \in N(f)$ die Beziehung $-f^2(x) \geq -f^2(x') \Leftrightarrow f(x) \geq f(x')$ beziehungsweise für alle $x, x' \in P(f)$ die Beziehung $-f^2(x) \geq -f^2(x') \Leftrightarrow -f(x) \geq -f(x')$ gilt, formulieren wir die für alle $x \in \{0,1\}^n$ definierten, aber speziell für suboptimale $x \in N'$ respektive $x \in P'$ genutzten Fortschrittsmaße H_P^* und H_N^* als

$$H_N^*(x) := n - \max\Big\{j \in \{0, \ldots, n\} | f(x) \geq \sum_{i=1}^{j} w_i + \tau\Big\} \text{ und}$$

$$H_P^*(x) := n - \max\Big\{j \in \{0, \ldots, n\} | f(x) \leq w - \sum_{i=1}^{j} w_i + \tau\Big\}.$$

Diese Maße sind im Wesentlichen analog zu Definition 3.6 definiert. Allerdings sind die Rollen von $P(f)$ und $N(f)$ vertauscht, um zu berücksichtigen, dass in der Menge M statt in $\vec{0}$ bzw. $\vec{1}$ die für $-f^2$ optimalen Vektoren liegen. Auch kann für $x \in M$ trotz der Optimalität $H_N^*(x) \neq 0$ und $H_P^*(x) \neq 0$ gelten, was aber unerheblich für die folgenden Aussagen ist. Wir dürfen nun nämlich analog zu Lemma 3.12 verfahren:
Im Verlaufe der Optimierung von $-f^2$ durch den (1+1)-EA notieren wir für alle aktuellen Bitstrings $x \in N'$ die Folge der Werte von H_N^* und für alle aktuellen Bitstrings $x \in P'$ die Folge der Werte von H_P^* jeweils in zeitlicher Reihenfolge. Weil die Ordnung von Funktionswerten von Bitstrings aus N' bezüglich $-f^2$ der Ordnung unter der Funktion f und die Ordnung von Funktionswerten von Bitstrings aus P' bezüglich $-f^2$ der Ordnung unter der Funktion $-f$ entspricht, sind beide Folgen monoton fallend. Zudem existiert analog zu Lemma 3.11 für alle $x \in N'$ ein $x' \in \{0,1\}^n$ mit $H(x, x') = 1$ und $H_N^*(x') \leq H_N^*(x) - 1$ und für alle $x \in P'$ ein $x' \in \{0,1\}^n$ mit $H(x, x') = 1$ und $H_P^*(x') \leq H_P^*(x) - 1$, d. h. die Maße können infolge einer 1-Bit-Mutation sinken. Wir zeigen dies exemplarisch für H_N^* und fixieren dazu ein $x \in N'$. Sei $j := n - H_N^*(x)$. Aus der Definition von H_N^* folgt

$$\sum_{i=1}^{j} w_i + \tau \leq f(x) < \sum_{i=1}^{j+1} w_i + \tau.$$

Wären alle Positionen $i \in \{1, \ldots, j+1\}$ von x mit 1 belegt, gälte wegen $w_i \in \mathbb{N}$ die Ungleichung $f(x) \geq \sum_{i=1}^{j+1} w_i + \tau$. Also existiert ein $i \in \{1, \ldots, j+1\}$ mit $x_i = 0$. Wir konstruieren x' aus x durch eine Mutation an Position i. Wegen $w_1 \geq \cdots \geq w_n > 0$ gilt dann $f(x') \geq \sum_{i=1}^{j+1} w_i + \tau$ und damit $H_N^*(x') \leq n - (j+1) = H_N^*(x) - 1$. Die analoge Aussage für H_P^* ist mit vertauschten Rollen von Nullbits und Einsbits zu führen.

Da die Menge suboptimaler Bitstrings disjunkt in N' und P' zerfällt und die erwartete Zeit bis zum Flipp an einer evtl. eindeutig bestimmten Position durch $O(n)$ beschränkt ist, sinkt nach erwarteten $O(n)$ Schritten eines der Fortschrittsmaße H_N^* bzw. H_P^*. Da jedes Fortschrittsmaß maximal n-mal sinken kann, ist nach erwarteten $2n \cdot O(n) = O(n^2)$ Schritten mindestens ein Fortschrittsmaß auf seinen minimal möglichen Wert gesunken. Dies ist äquivalent dazu, dass der (1+1)-EA einen Bitstring $x \in M$ erreicht hat, d. h. $-f^2$ optimiert hat. □

Wiederum glauben wir, dass sogar eine obere Schranke von $O(n \ln n)$ gilt. Leider konnten wir uns bei diesem Beweis aber ebenso wenig wie in Lemma 3.12 auf die allgemeine obere Schranke für lineare Funktionen zurückziehen, da die aktuellen Bitstrings des (1+1)-EA während der Optimierung von $-f^2$ sowohl aus N' und P' stammen können und sich bei einem Wechsel von N' zu P' die Ordnung der Funktionswerte umkehrt, als ob zwischen der Optimierung der linearen Funktion f und der Optimierung der linearen Funktion $-f$ gewechselt würde, und umgekehrt.

Mit diesem Exkurs, in dem wir Auskunft über das Verhalten des (1+1)-EA auf beliebigen Potenzen linearer Funktionen zu geben vermochten, beenden wir dieses Kapitel. Im Folgenden lösen wir uns weiter von linearen Funktionen und stellen Untersuchungen zu den Strukturen quadratischer Funktionen an.

Kapitel 4

Fast lineare Funktionen

In diesem Kapitel beschäftigen wir uns mit quadratischen Fitnessfunktionen, die gewisse Ähnlichkeiten zu linearen Funktionen aufweisen und daher eventuell vom (1+1)-EA effizient optimiert werden. Bei den Quadraten potenziell schwieriger Funktionen stellte es sich in Kapitel 3 heraus, dass zwar nicht alle von diesen in erwarteter polynomieller Zeit optimiert werden, dass sie aber ähnlich genug zu linearen Funktionen sind, um mit einer Wahrscheinlichkeit nahe bei 1/8 in der Zeit $O(n \ln n)$ optimiert zu werden. In diesem Kapitel wollen wir analysieren, wie stark quadratische Funktionen linearen Funktionen zu ähneln haben, um in erwarteter effizienter, d. h. polynomieller Zeit optimiert zu werden. Zu den eigentlichen fast linearen Funktionen im Sinne von Definition 2.5 kommen wir dabei in Abschnitt 4.3.
Zunächst überlegen wir uns eine Darstellungsform für quadratische Funktionen.

Definition 4.1 (Koeffizientenmatrix quadratischer Fkt.) *Die $(n \times n)$-Matrix $W = (w_{ij})$, induziert durch eine quadratischen Fitnessfunktion $f : \{0,1\}^n \to \mathbb{R}$ der Gestalt $f(x) = \sum_{i=1}^{n} w_{ii}x_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij}x_ix_j$, heißt Koeffizientenmatrix der quadratischen Funktion.*

Die Matrix W ist eine obere Dreiecksmatrix, da $xy = yx$ im Körper $\mathbb{Z}_2$, aus dem die Multiplikationsverknüpfung für $x, y \in \{0,1\}$ resultiert, stets zutrifft; alternativ wäre eine Darstellung von W als eventuell vollständig besetzte Matrix mit Gewichten w_{ij} und w_{ji}, die beide nicht notwendig 0 sind, möglich, was aber keinen weiteren Vorteil bringt, sondern der Übersichtlichkeit eher abträglich ist. Daher nehmen wir bei quadratischen Funktionen stets $w_{ij} = 0$ für $j < i$ an.
Bereits in Abschnitt 2.2 haben wir in Lemma 2.4 gesehen, dass wir neben der Annahme $\tau = 0$, die bereits in die Darstellung als Koeffizientenmatrix einging, für die Koeffizienten w_{ii}, dort mit w_i bezeichnet, sowie w_{ij}, $i < j$, ohne Beschränkung der Allgemeinheit Ganzzahligkeit annehmen dürfen sowie von einer monoton fallenden Ordnung der Gewichte w_{ii}, $i \in \{1, \ldots, n\}$, also $w_{11} \geq \cdots \geq w_{nn}$, ausgehen können. Wir schreiben nachfolgend weiterhin kurz w_i statt w_{ii}, wenn wir Einträge auf der Hauptdiagonalen bezeichnen wollen und nennen diese zuweilen kurz *lineare Gewichte/Koeffizienten*, wohingegen für die w_{ij} mit $i < j$ der Begriff *quadratische Gewichte/Koeffizienten* eingesetzt wird.

Bemerkung 4.1 Lineare Funktionen als Spezialfall quadratischer Funktionen haben in ihrer Koeffizientenmatrix nur in der Hauptdiagonalen von 0 verschiedene Einträge. Alle anderen Positionen der Koeffizientenmatrix sind mit Nullen besetzt.

Quadrate linearer Funktionen hingegen weisen im Allgemeinen in allen Positionen ihrer Koeffizientenmatrix Nichtnulleinträge auf – statt „Nichtnulleinträgen“ sagen wir kurz, die Matrix habe *Einträge* an bestimmten Positionen. Auffällig sind dabei negative Einträge, die infolge des Quadrierens einer potenziell schwierigen linearen Funktion (vgl. Definition 3.7) entstehen. Dieses soll am Beispiel von $\textsc{OneSqr}_{-n/2+1/3}$ aus Kapitel 3 näher untersucht werden:

$$\begin{aligned}\left(\sum_{i=1}^{n} x_i - \frac{n}{2} + \frac{1}{3}\right)^2 &= \left(\sum_{i=1}^{n} x_i\right)^2 - 2\left(\frac{n}{2} - \frac{1}{3}\right)\sum_{i=1}^{n} x_i + \frac{n^2}{4} - \frac{n}{3} + \frac{1}{9} \\ &= \left(\sum_{i=1}^{n} x_i \sum_{j=1}^{n} x_j\right) - \sum_{i=1}^{n}\left(n - \frac{2}{3}\right)x_i + \frac{n^2}{4} - \frac{n}{3} + \frac{1}{9} \\ &= \sum_{i=1}^{n}\left(-n + \frac{5}{3}\right)x_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} 2x_i x_j + \frac{n^2}{4} - \frac{n}{3} + \frac{1}{9}, \text{ denn } x_i^2 = x_i.\end{aligned}$$

Die Koeffizientenmatrix der Funktion $\textsc{OneSqr}_{-n/2+1/3}$ ist also in und oberhalb der Hauptdiagonalen voll besetzt mit der Besonderheit, dass alle Einträge der Hauptdiagonalen negative und alle übrigen Einträge positive rationale Zahlen sind. (Per Multiplikation mit einem Hauptnenner sind diese ohne weiteres ganzzahlig zu machen.) Wir erkennen leicht, dass beim Quadrieren einer linearen Funktion mit $\tau \geq 0$ und o. B. d. A. natürlichen Gewichten generell nur positive Einträge in der Koeffizientenmatrix der entstehenden quadratischen Funktion auftreten. Bereits aus Lemma 2.7 ist bekannt, dass der (1+1)-EA derartige quadratische Fitnessfunktionen effizient zu optimieren vermag, da sie von der linearen Funktion selbst nicht zu unterscheiden sind. Allgemeiner wollen wir nun untersuchen, ob eine quadratische Funktion, deren Gewichte alle nicht negativ sind, für den (1+1)-EA überhaupt schwierig sein kann.

4.1 Quadratische Funktionen ohne negative Gewichte

Bei einer quadratischen Funktion, deren Gewichte alle mindestens 0 betragen, kann eine Mutation eines Nullbits im aktuellen Bitstring für sich alleine betrachtet offensichtlich nicht zu einem geringeren Funktionswert führen und wird immer akzeptiert, wenn keine Mutationen von Einsbits zu Nullbits an anderen Positionen stattfinden. Bei linearen Funktionen nehmen wir o. B. d. A. an, dass alle Gewichte nicht negativ sind, um ein globales Maximum in $\vec{1}$ zu erhalten. Da die Funktion von Bits an Positionen mit einem zugeordneten Gewicht 0 unabhängig wäre, dürfen wir sogar von positiven, also natürlichen Gewichten ausgehen und erhalten damit ein eindeutiges globales Maximum in $\vec{1}$. Diese Art von „Normalform“ wollen wir nun auch bei quadratischen Funktionen ohne negative Gewichte finden, um bei nachfolgenden Beweisen die Übersichtlichkeit zu erhöhen.

Lemma 4.1 *Sei f eine quadratische Fitnessfunktion, deren Gewichte ausschließlich in der Menge N_0 enthalten sind. Wir können o. B. d. A. annehmen, dass f nur ein globales Maximum in $\vec{1}$ besitzt.*

Beweis: Sei

$$I_0 := \{i \in \{1, \ldots, n\} | w_i = 0 \wedge \forall j \in \{i+1, \ldots, n\} : w_{ij} = 0\}$$

die Menge aller Indizes, sodass für $i \in I_0$ die Monome $w_i x_i$ und $w_{ij} x_i x_j$, $i < j \leq n$, stets 0 sind. Die Funktion f hängt von den x_i mit $i \in I_0$ nicht ab; jede Belegung der x_i mit $i \in I_0$ führt bei einer Belegung $x'_i := 1$ für $i' \notin I_0$ wegen nicht negativer Gewichte zu einem optimalen Funktionswert. Wir können daher annehmen, dass die Menge I_0 leer ist und damit alle Bits des aktuellen Bitstrings für einen optimalen Funktionswert mit 1 belegt sein müssen. □

Im Folgenden nehmen wir nun natürlich an, dass die Koeffizientenmatrix einer betrachteten quadratischen Funktion f mit nur nicht negativen Gewichten so beschaffen ist, dass die Funktion von allen n Variablen abhängig ist und das einzige Maximum in $\vec{1}$ vorliegt. (Dann enthält für jedes $i \in \{1, \ldots, n\}$ die Zeile oder Spalte i der Matrix mindestens einen Eintrag.) Aufgrund der Nichtnegativität der Gewichte der in diesem Abschnitt betrachteten quadratischen Funktionen möchten wir nun eine polynomielle obere Schranke für die erwartete Laufzeit des (1+1)-EA zeigen, indem wir ähnlich wie in Lemma 3.12 Mutationen, die den Funktionswert möglichst stark erhöhen und den maximal für den (1+1)-EA erreichbaren Hammingabstand zum Optimum verringern, betrachten. Nehmen wir uns alle Gewichte der Koeffizientenmatrix vor, so können wir uns alle von 0 verschiedenen Gewichte absteigend geordnet vorstellen. Solange nicht alle dieser Gewichte „aktiviert“ sind, d. h. dass im Falle eines linearen Gewichtes die zugehörige Position im Bitstring noch 0 bzw. im Falle eines quadratischen Gewichtes mindestens eine der beiden zugehörigen Positionen im Bitstring noch 0 ist, kann mithilfe einer Mutation mindestens eines und höchstens zweier Bits der Funktionswert der quadratischen Funktion um mindestens den Wert des/eines größten noch nicht aktivierten Gewichtes erhöht werden. Das Wort „mindestens“ im vorigen Satz zeigt allerdings an, dass hier die Analogie zu linearen Funktionen aufgrund von Seiteneffekten – die Aktivierung eines linearen Gewichtes mittels einer Mutation eines Nullbits kann neben einem linearen noch $n-1$ weitere quadratische und die Mutation zweier Nullbits sogar 2 lineare und $(n-1)+(n-2) = 2n-3$ quadratische Gewichte aktivieren – endet, denn gegenüber n Positionen im Bitstring existieren bis zu $(n^2+n)/2$ von 0 verschiedene Gewichte. Aus diesen Seiteneffekten folgt zunächst, dass wir infolge der Aktivierung eines Gewichtes mit maximalem Wert anders als bei linearen Funktionen den maximal erreichbaren Hammingabstand zu $\vec{1}$ nicht unbedingt um 1 verringern müssen, vgl. folgendes Beispiel.

Beispiel 4.1 Die quadratische Funktion

$$f(x) = \sum_{i=1}^{5} 6x_i + \sum_{i=6}^{9} \sum_{j=i+1}^{9} 5x_i x_j$$

besitzt nur nicht negative Gewichte, die sich (abgesehen von denen mit Wert 0) in der monoton fallenden Ordnung $w_1 = w_2 = w_3 = w_4 = w_5 > w_{67} = w_{68} = w_{69} = w_{78} = w_{79} = w_{89}$ notieren lassen. Erwarten wir dann, dass von $\vec{0}$ ausgehend zunächst die Bits $x_1, \ldots, x_5$ zu 1 flippen, da sie die maximalen Gewichte aktivieren, besitzt der aktuelle Bitstring zwar 5 Einsen, doch der maximal (für den (1+1)-EA) erreichbare Hammingabstand zu $\vec{1}$ ist nicht $9 - 5 = 4$, sondern $9 - 4 = 5$, denn der Bitstring x mit $x_1 = \cdots = x_5 = 0$ und $x_6 = \cdots = x_9 = 1$ hat mit 4 Einsen einen Funktionswert von bereits 30 und könnte durch Mutation aus dem zunächst erwähnten Bitstring entstehen, da jener ebenfalls den Funktionswert 30 besitzt.

Allerdings kann eine Mutation wie im vorigen Beispiel die Zahl der Einsen im aktuellen Bitstring nicht beliebig (das hieße z. B. auf 1) verringern, wenn zunächst diejenigen Nullbits geflippt sind, die jeweils maximale noch nicht aktivierte Gewichte aktivieren. Anhand der Ordnung der Gewichte definieren wir das folgende Fortschrittsmaß.

Definition 4.2 *Für eine quadratische Funktion f mit ausschließlich nicht negativen Koeffizienten sei für $N := (n^2+n)/2$ die Folge $w_1^*, \ldots, w_N^*$ eine monoton fallende Umordnung der Gewichte w_i, w_{ij}, $1 \leq i < j \leq n$, von f. Daraus definieren wir für ein $x \in \{0,1\}^n$ das Maß $O^*(x)$ als*

$$O^*(x) := \max\Big\{j \in \{0, \ldots, N\} | f(x) \geq \sum_{i=1}^{j} w_i^*\Big\}.$$

Wir können dieses Maß als minimale Zahl von Einsen, die ein Bitstring mit einem Funktionswert von mindestens $f(x)$ auf der imaginären linearen Funktion $\sum_{i=1}^{N} w_i^* x_i$ (eventuell nehmen hier einige Gewichte ab einem Index N' mit $n/2 < N' \leq N$ den Wert 0 an) haben müsste, interpretieren. Auf einer quadratischen Funktion mit nicht negativen Gewichten erhalten wir mithilfe dieses Maßes eine untere Schranke für die Zahl der Einsen im aktuellen Bitstring.

Lemma 4.2 *Sei f eine quadratische Funktion mit ausschließlich nicht negativen Koeffizienten. Für alle $x \in \{0,1\}^n$ gilt: $O^*(x) = i \Rightarrow$ x enthält mindestens $\sqrt{i}$ Einsen.*

Beweis: Da $O^*(x) = i$, enthält x so viele Einsen, dass mindestens i Gewichte, die alle mindestens 1 betragen, aus der Folge w_j^*, $j \in \{1, \ldots, N\}$, addiert werden müssen, um einen Wert von mindestens $f(x)$ zu erreichen. Um i Gewichte zu aktivieren, muss x mindestens $\sqrt{i}$ Einsen besitzen, denn mit $k := \sqrt{i}$ Einsen werden k lineare und $\sum_{i=1}^{k} \sum_{j=i+1}^{k} = (k^2 - k)/2$ quadratische Gewichte aktiviert (von denen eventuell einige sogar null sind). Also aktivieren $\sqrt{i}$ Einsen höchstens $k^2/2 + k/2 = i/2 + \sqrt{i}/2 \leq i$ Gewichte. □

Im Gegensatz zu Lemma 3.11 gibt unser Fortschrittsmaß $O^*(x)$ keine obere Schranke für den maximal erreichbaren Hammingabstand an, sondern ist „umgekehrt“ definiert, und liefert mit Lemma 4.2 eine untere Schranke für die Zahl der Einsen im Bitstring x. Da die Zahl der Einsen in x, sofern $x \neq \vec{1}$, zur Erreichung des Optimums steigen muss, fehlt zur

Vorbereitung eines Satzes über quadratische Funktionen mit nur nicht negativen Gewichten noch eine formale Aussage über die Bedingungen, unter denen sich das Fortschrittsmaß $O^*(x)$ erhöht.

Lemma 4.3 *Sei f eine quadratische Funktion mit ausschließlich nicht negativen Koeffizienten. Für alle $x \in \{0,1\}^n \setminus \{\vec{1}\}$ existiert ein $x' \in \{0,1\}^n$ mit $H(x,x') \leq 2$ und $O^*(x') \geq O^*(x) + 1$.*

Beweis: Sei $j := O^*(x)$, also $j \in \{0,\ldots,N-1\}$. Wir entnehmen der Definition von $O^*(x)$, dass $\sum_{i=1}^{j} w_i^* \leq f(x) < \sum_{i=1}^{j+1} w_i^*$ gilt. Daraus folgt $w_{j+1}^* > 0$. Wenn x alle Werte w_i^* mit $i \in \{1,\ldots,j+1\}$ aktivierte, gälte $f(x) \geq \sum_{i=1}^{j+1} w_i^*$. Wir können mithin ein noch nicht aktiviertes w_i^* mit einem Index $i \in \{1,\ldots,j+1\}$ aktivieren, indem wir die dazu gehörenden maximal 2 Positionen – das Gewicht kann quadratisch sein – in x von 0 nach 1 flippen und somit ein x' mit $H(x,x') \leq 2$ konstruieren. Aufgrund der Ordnung $w_1^* \geq \cdots \geq w_N^*$ folgt dann $f(x') \geq \sum_{i=1}^{j+1} w_i^*$, also $O^*(x') \geq O^*(x) + 1$. □

Trivialerweise sinkt $O^*(x)$ im Laufe des Algorithmus niemals. Mit den vorangehenden Lemmata lässt sich nun eine polynomielle obere Schranke für die betrachtete Klasse von Funktionen zeigen.

Satz 4.4 *Sei f eine quadratische Funktion, deren Koeffizientenmatrix nur nicht negative Einträge enthält. Der (1+1)-EA optimiert die Funktion f in erwarteter Zeit $O(n^4)$.*

Beweis: Für den initialisierten Bitstring x^s messen wir $j := O^*(x^s)$, $j \in \{0,\ldots,N\}$, und schätzen die erwartete Zeit, bis der aktuelle Bitstring x mit $\vec{1}$ übereinstimmt, nach oben ab. Gemäß Lemma 4.2 ist nach maximal n^2 Erhöhungen (es genügen aufgrund der Definition von $O^*(x)$ sogar $N - j = (n^2+n)/2 - j = O(n^2)$ Erhöhungen) von $O^*(x)$ der optimale Bitstring erreicht. Eine hinreichende Bedingung für diese Erhöhung ist laut Lemma 4.3 die Mutation mindestens eines und maximal zweier Bits an eventuell genau bestimmten Positionen, was mit einer Wahrscheinlichkeit von mindestens $n^{-2}(1-1/n)^{n-2} \geq e^{-1}n^{-2}$ eintritt, sodass die erwartete Zeit bis zu einer solchen Mutation durch en^2 nach oben beschränkt ist. Somit genügen $n^2 \cdot en^2 = O(n^4)$ erwartete Schritte. □

Bemerkung 4.2 In Satz 4.4 ist der Sonderfall linearer Funktionen, falls $w_{ij} = 0$ für alle $i,j \in \{1,\ldots,n\}$ mit $i < j$ gilt, enthalten, für den sogar die obere Schranke $O(n \ln n)$ bekannt ist. Wir vermuten, dass die Schranke $O(n^4)$ aus Satz 4.4 weiter verbessert werden kann. Insbesondere wenn die Koeffizientenmatrix einer quadratischen Funktion mit ausschließlich nicht negativen Gewichten in der Diagonalen voll besetzt ist, d. h. wenn alle linearen Gewichte positiv sind, kann der Funktionswert, solange das Optimum nicht erreicht ist, mithilfe einer 1-Bit-Mutation erhöht werden. Dies stellt eine Analogie zu linearen Funktionen dar. Wir verfolgen diesen Gedanken wegen des Umfangs des Beweises der allgemeinen oberen Schranke für lineare Funktionen aus [DJW98a] jedoch nicht weiter.

Wenn die quadratischen Gewichte w_{ij}, $i < j$, gegenüber den übrigen linearen Gewichten kleine Werte besitzen, können wir aus der quadratischen Funktion sogar eine für den (1+1)-EA äquivalente lineare Funktion konstruieren.

Beispiel 4.2 Die quadratische Funktion f mit

$$f(x) = \sum_{i=1}^{n} 2^{2\log n+n-i+1} x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i x_j$$

ist für den (1+1)-EA von der linearen Funktion BinVal nicht zu unterscheiden. Dies machen wir uns anhand der Ordnung der Funktionswerte der 2^n Vektoren des Definitionsbereiches klar, die bei beiden Funktionen identisch ist, denn die quadratischen Gewichte tragen in ihrer Summe von weniger als n^2 einen geringeren Wert als die minimale positive Differenz zwischen den linearen Gewichten bei, und alle Funktionswerte sind unterschiedlich. An diesem Beispiel erkennen wir auch die Schwierigkeit bzw. Unmöglichkeit, eine lineare Funktion zu finden, auf der der (1+1)-EA dasselbe Akzeptanzverhalten wie auf einer gegebenen quadratischen Funktion aufweist. Will man dies einfach durch Entfernen der quadratischen Gewichte erreichen, müssen diese zum einen so klein sein, dass sie die Ordnung bezüglich „<“ erhalten, zum anderen müssen Vektoren mit demselben Funktionswert unter der quadratischen Funktion auch auf einen jeweils gleichen Funktionswert unter der linearen Funktion abgebildet werden. Dieses funktioniert vermutlich nur bei sehr wenigen quadratischen Funktionen.

Bemerkung 4.3 Analog zu den vorigen Beweisen lässt sich vermutlich auch eine polynomielle obere Schranke für die erwartete Laufzeit des (1+1)-EA auf Funktionen konstanten Grades mit ausschließlich nicht negativen Koeffizienten zeigen. Eine Funktion $f : \{0,1\}^n \to \mathbb{R}$ vom Grad k, $k \in \mathbb{N}$ konstant, kann als Polynom aus weniger als n^k Monomen dargestellt werden, d. h. wir haben maximal $O(n^k)$ Gewichte. Um ein Gewicht mit maximalem Wert zu aktivieren, genügt eine Mutation von maximal k ausgewählten Bits zugleich, die nach erwarteten $O(n^k)$ Schritten eintritt. Insgesamt optimiert der (1+1)-EA also Funktionen vom Grad k, die nur nicht negative Koeffizienten besitzen, vermutlich in höchstens $O(n^{2k})$ Schritten.
Andererseits glauben wir auch, dass Funktionen vom Grad k mit ausschließlich nicht negativen Koeffizienten existieren, auf denen der (1+1)-EA im Erwartungswert mindestens $\Omega(n^{k/2})$ Schritte benötigt. In der Funktion $f_k(x) = \sum_{i=0}^{n/k-1} \prod_{j=1}^{k} x_{ik+j}$ beispielsweise sind Bitstrings in n/k Blöcke unterteilt, innerhalb deren die Zahl der Einsen genau k betragen muss, um das zugehörige Monom vom Grad k zu aktivieren. Die Zahl der Einsen innerhalb eines solchen Blockes hat keinen Einfluss auf das Akzeptanzverhalten des (1+1)-EA, solange sie nicht k beträgt. Somit scheint die Zahl der Einsen innerhalb von Blöcken, die noch nicht k Einsen enthalten, ein rein zufälliger Wert zu sein. Wir vermuten, dass solche Blöcke im Erwartungswert $k/2$ Einsen enthalten. Eine hinreichende Bedingung für eine optimale Belegung eines Blocks mit ursprünglich $k/2$ Einsen ist das Eintreten einer oder mehrerer Mutationen, die insgesamt genau $k/2$ Nullbits zu 1 flippen und bereits vorhandene Einsbits unverändert lassen. Egal, ob dieses in einem oder mehreren Schritten geschieht, scheint

die Wahrscheinlichkeit dafür durch $O(n^{-k/2})$ nach oben beschränkt zu sein. Wir vermuten daher, dass sich die Zahl der Einsen innerhalb eines Blockes erst nach erwarteten $\Omega(n^{k/2})$ Schritten auf k erhöht.
Funktionen nicht konstanten Grades schließlich können schwierig sein, obwohl alle Koeffizienten nicht negativ sind. Ist $k = n$, also nicht mehr konstant, entspricht die vorgestellte Funktion f_k genau der Funktion $\textsc{Peak}(x) = \prod_{i=1}^{n} x_i$, welche in [DJW98a] behandelt wird. Zur Optimierung dieser „Nadel-im-Heuhaufen-Funktion“ benötigt der (1+1)-EA (sogar alle Black-Box-Optierungsverfahren) erwartete exponentielle Zeit.

Nachdem wir einen Spezialfall quadratischer Funktionen mit eventuell voll besetzten Koeffizientenmatrizen behandelt haben, beschäftigen wir uns nun mit quadratischen Funktionen, deren Koeffizientenmatrizen höchstens linear viele Einträge besitzen.

4.2 Linear viele quadratische Gewichte

Die Definition fast linearer Funktionen (vgl. Definition 2.5) erlaubt konstant viele Einträge außerhalb der Hauptdiagonalen der Koeffizientenmatrix. Allerdings zeigt sich, dass auch spezielle quadratische Funktionen mit sogar linear vielen quadratischen Gewichten für den (1+1)-EA „einfach“ sind. Dabei werden wir uns wiederum Resultate über lineare Funktionen zunutze machen, die wir allerdings nicht mehr in der üblichen Normalform mit natürlichen Gewichten etc. annehmen können. Deshalb stellen wir ein zu Definition 3.6 etwas abgewandeltes Fortschrittsmaß auf.

Definition 4.3 *Für eine lineare Funktion $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$ mit $w_i \in \mathbb{Z}$ für alle $i \in \{1, \ldots, n\}$ und $\tau \in \mathbb{R}$ definieren wir die Folge $w_i^*, \ldots, w_n^*$ als eine monoton fallende Umordnung der Folge der Beträge der Gewichte $|w_1|, \ldots, |w_n|$. Daraus definieren wir für ein $x \in \{0,1\}^n$ den maximal (von x für den (1+1)-EA) erreichbaren Hammingabstand zu einem Maximum von f als*

$$H^*(x) := n - \max\left\{ j \in \{0, \ldots, n\} | f(x) \geq \sum_{i=1}^{j} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau \right\}.$$

Wir müssen den Sinn dieses Maßes erläutern.

Lemma 4.5 *Für eine lineare Funktion f mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $w_i \in \mathbb{Z}$, $\tau \in \mathbb{R}$, das dafür nach Definition 4.3 gebildete Fortschrittsmaß H^* und ein beliebiges $x_{\max} \in \{0,1\}^n$ mit maximalem Funktionswert gilt*

$$\forall x \in \{0,1\}^n,\ x \text{ nicht optimal}: H^*(x) = j \ \Rightarrow\ H(x, x_{\max}) \leq j.$$

Beweis: Der minimale Funktionswert von f ist $\sum_{\{i|w_i<0\}} w_i + \tau$. Dieser wird auf einem Bitstring angenommen, der an den Bits mit positiven Gewichten mit 0 und an den Bits

mit negativen Gewichten mit 1 belegt ist. Eine Erhöhung des Funktionswertes um $|w_i|$, $i \in \{1, \ldots, n\}$, ist infolge eines 0-1-Flipps an Bit i, falls w_i positiv, bzw. infolge eines 1-0-Flipps an Bit i, falls w_i negativ, möglich. Offensichtlich sind in $x_{\max}$ Bits mit positiven Gewichten mit 1 und Bits mit negativen Gewichten mit 0 belegt, während die Belegung von Bits mit zugeordnetem Gewicht 0 irrelevant ist.
Aus der Definition von H^* folgt nun, dass für $m := n - H^*(x)$ die Ungleichung $f(x) \geq \sum_{i=1}^{m} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau$ gilt. Weil x nicht optimal ist, müssen aufgrund der Ordnung $w_1^* \geq \cdots \geq w_n^* \geq 0$ die Werte $w_1^*, \ldots, w_{m+1}^*$ noch von 0 verschieden sein. Damit folgt wiederum aus der monoton fallenden Ordnung der w_i^*, dass m eine untere Schranke für die Anzahl der Bits mit von 0 verschiedenen Gewichten angibt, die entsprechend der Belegung im Maximum zu belegen sind, um zum geringstmöglichen Funktionswert $\sum_{\{i|w_i<0\}} w_i + \tau$ einen Wert $\sum_{i=1}^{m} w_i^*$ zu addieren, sodass die Summe der Werte $f(x)$ betragen kann.
Also können maximal $n-m = H^*(x)$ Gewichte gegenüber einem optimalen Bitstring falsch belegt sein, d. h. $H(x, x_{\max}) \leq H^*(x)$. □

Ähnlich wie in Lemma 3.11 beschreiben wir nun eine Möglichkeit, dieses Fortschrittsmaß zu senken.

Lemma 4.6 *Sei f eine lineare Funktion mit $f(x) = \sum_{i=1}^{n} w_i x_i + \tau$, $w_i \in \mathbb{Z}$, $\tau \in \mathbb{R}$ und sei H^* das nach Definition 4.3 für f definierte Fortschrittsmaß. Für alle nicht optimalen $x \in \{0,1\}^n$ existiert ein $x' \in \{0,1\}^n$ mit $H(x, x') = 1$, sodass $H^*(x') \leq H^*(x) - 1$.*

Beweis: Betrachte $m := n - H^*(x)$ Aus der Definition von H^* folgt

$$\sum_{i=1}^{m} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau \leq f(x) < \sum_{i=1}^{m+1} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau.$$

Trivialerweise gilt dann $w_{m+1}^* > 0$. Bei den Werten w_i^*, $i \in \{1, \ldots, n\}$, sprechen wir nun wiederum davon, dass sie von einem Bitstring x *aktiviert* werden, wenn für ein i das zugehörige Bit x_j, $j \in \{1, \ldots, n\}$, in der noch nicht umsortierten Folge $w_1, \ldots, w_n$ im Falle eines negativen Gewichtes w_j mit 0 und im Falle eines positiven Gewichtes mit 1 belegt wird. Gewichte w_i^* mit $w_i^* = 0$ betrachten wir nicht. Jetzt können wir ähnlich wie im Beweis zu Lemma 4.3 argumentieren.
Wenn x alle Werte w_i^* mit $i \in \{1, \ldots, m+1\}$ aktivierte, würde daraus die Ungleichung $f(x) \geq \sum_{i=1}^{m+1} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau$ folgen. Es wird also einer der Werte $w_1^*, \ldots, w_{m+1}^*$ von x nicht aktiviert, d. h. wir können mithilfe einer 1-Bit-Mutation den Bitstring x' aus x erzeugen und sodann aufgrund der monoton fallenden Ordnung der w_i^* zu der Ungleichung $f(x') \geq \sum_{i=1}^{m+1} w_i^* + \sum_{\{i|w_i<0\}} w_i + \tau$ kommen. Damit folgt $H^*(x') \leq n-(m+1) = H^*(x)-1$.
□

Lemma 4.7 *Sei f eine quadratische Funktion. Wenn es einen Index $l \in \{1, \ldots, n\}$ gibt, dass $\forall i, j \in \{1, \ldots, n\} \setminus \{l\} : i \neq j \Rightarrow w_{ij} = 0$ gilt, optimiert der (1+1)-EA f in erwarteter Zeit $O(n^2)$.*

Beweis: Mittels Umordnung nehmen wir o. B. d. A. an, dass lediglich die quadratischen Gewichte in Zeile 1 der Koeffizientenmatrix, d. h. w_{1i}, $i \in \{2, \ldots, n\}$, von 0 verschiedene Werte haben, womit wir die Annahme $w_1 \geq \cdots \geq w_n$ für die linearen Gewichte fallen lassen.
Der (1+1)-EA wechselt nun abhängig vom Wert von x_1 zwischen den linearen Funktionen $f_1(x_2, \ldots, x_n) := \sum_{i=2}^{n} w_i x_i$ und $f_2(x_2, \ldots, x_n) := w_1 + \sum_{i=2}^{n} w'_i x_i$ mit $w'_i := w_i + w_{1i}$ für $i \in \{2, \ldots, n\}$ eventuell hin und her (nach n erwarteten Schritten). Würden die Funktionen f_1 und f_2 isoliert betrachtet, könnten wir Aussagen über das Verhalten des (1+1)-EA auf linearen Funktionen heranziehen. Weil aber eventuell zwischen f_1 und f_2 während der Optimierung von f gewechselt wird, dürfen wir die obere Schranke von $O(n \ln n)$ für lineare Funktionen auf die Funktion f nicht unmittelbar anwenden. Stattdessen benutzen wir wieder ein Fortschrittsmaß wie in Definition 3.6 und zeigen eine obere Schranke von $O(n^2)$. Für f_1 nutzen wir das obige nach Definition 4.3 gegebene Maß und bezeichnen es mit H_1^*, für f_2 bezeichnen wir das nach Definition 4.3 gegebene Maß mit H_2^*.
Wenn wir die Folgen der Werte von $H_1^*(x)$ für $x_1 = 0$ bzw. von $H_2^*(x)$ für $x_1 = 1$ in der durch den Verlauf des (1+1)-EA bestimmten Reihenfolge notieren, ist es aufgrund dessen Akzeptanzverhaltens klar, dass die beiden Folgen monoton fallen. Gemäß Lemma 4.6 führt ein Flipp eines falsch eingestellten Bits, dessen Gewicht einen maximalen Betrag hat, zu einer Verringerung von $H_1^*(x)$, falls $x_1 = 0$, bzw. von $H_2^*(x)$, falls $x_1 = 1$. Dabei handelt es sich um eine Mutation von 0 nach 1, falls dieses betragsmäßig maximale Gewicht größer als 0 ist, und von 1 nach 0, falls es geringer als 0 ist. Die Belegung von Bits, deren Gewicht 0 beträgt, ist für die Maximierung der entsprechenden Funktion irrelevant. Somit beträgt die Erfolgswahrscheinlichkeit für eine Mutation, die den maximalen Hammingabstand einer linearen Funktion zu einem ihrer Maxima (nicht unbedingt in $\vec{1}$) um mindestens 1 verringt, mindestens $e^{-1}n^{-1}$ (siehe Lemma A.9), sodass nach maximal erwarteten $2n \cdot en = O(n^2)$ Schritten ein für mindestens eine der linearen Funktionen f_1 und f_2 optimaler Bitstring erreicht ist. Anschließend sind möglicherweise noch weitere $n + O(n \ln n) = o(n^2)$ erwartete Schritte bis zu einer Mutation von x_1 und zur Optimierung derjenigen linearen Funktion, deren maximaler Funktionswert mit dem von f übereinstimmt, erforderlich. □

Beispiel 4.3 Die Funktion $f(x) = nx_1 + \sum_{i=2}^{n} x_i + \sum_{i=2}^{n} -2x_1 x_i$ wechselt in Abhängigkeit vom Wert von x_1 zwischen der linearen Funktion

$$f_1(x_2, \ldots, x_n) = 1 + \textsc{OneMax}(x_2, \ldots, x_n)$$

und der Funktion

$$f_2(x_2, \ldots, x_n) = n - \textsc{OneMax}(x_2, \ldots, x_n)$$

hin und her und besitzt sowohl in $(0, 1, \ldots, 1)$ als auch in $(1, 0, \ldots, 0)$ ein globales Maximum mit dem Funktionswert n sowie zwei globale Minima in $\vec{0}$ und $\vec{1}$. Wenn der aktuelle Bitstring im Bereich von x_2 bis x_n weniger als $(n-1)/2$ Einsen besitzt, ist ein 0-1-Flipp von x_1 denkbar, woraufhin keine Mutation mehr, die nur x_1 flippt, akzeptiert wird, da der aktuelle Bitstring mit mehr als $(n-1)/2$ Nullen im angegebenen Bereich unter f_2 näher am maximalen Funktionswert als unter f_1 läge. Analog kann bei mehr als $(n-1)/2$ Einsen

im Bereich x_2 bis x_n ein Flipp, der ausschließlich x_1 mutiert, eintreten, der in diesem Fall von $x_1 = 1$ zu $x_1 = 0$ führt. Lediglich bei bei exakt $(n-1)/2$ Einsen unter den Bits x_2 bis x_n und ungeradem n sind mehrere alleinige Flipps von x_1 denkbar, bis sich die Zahl der Einsen unter den Bits x_2 bis x_n nach konstanter[1] erwarteter Zeit ändert.

Bemerkung 4.4 In Lemma 4.7 können wir vermuten, dass sogar eine obere Schranke von $O(n \ln n)$ gilt. Leider kann bei einer Mutation desjenigen Bits x_i, $i \in \{1, \ldots, n\}$, mittels dessen zwischen zwei linearen Funktionen f_1, f_2 gewechselt wird, der Hammingabstand zu einem Maximum einer Funktion $f^* \in \{f_1, f_2\}$ größer geworden sein, als er zum letzten vorhergegehenden Zeitpunkt war, zu dem die Funktion f^* „eingeschaltet“ war, und somit der Fortschritt zu einem Maximum der Funktion f^* verringert worden sein. Auch wenn die Möglichkeit einer den Hammingabstand vergrößernden Mutation im Beweis, dass jede lineare Funktion in $O(n \ln n)$ Schritten maximiert wird (vgl. [DJW98a]), berücksichtigt wird, tritt solch ein Ereignis vermutlich mit anderen Wahrscheinlichkeiten als ein Flipp von x_i ein.

In Lemma 4.7 haben wir ein Resultat für quadratische Funktionen mit linear vielen Gewichten, die allerdings (nach einer Umordnung der Indizes) in nur einer Zeile der Koeffizientenmatrix auftreten dürfen, aufgestellt. Der grundlegende Gedanke hat dabei darin bestanden, dass der (1+1)-EA zwischen zwei linearen Funktionen wechselt. Wenn nun in maximal k Zeilen der Koeffizientenmatrix außerhalb der Diagonalen Gewichte erlaubt werden, vermuten wir, dass der (1+1)-EA dann bei der Optimierung zwischen 2^k Funktionen hin und her wechseln kann. Der folgende Satz zeigt in einer kanonischen Erweiterung zu Lemma 4.7, dass die erwartete Laufzeit des (1+1)-EA dann ebenfalls polynomiell ist.

Satz 4.8 *Sei f eine quadratische Funktion. Wenn eine Menge von Indizes $I \subseteq \{1, \ldots, n\}$ mit konstanter Größe $k := \#I$ existiert, sodass $\forall i, j \in \{1, \ldots, n\} \setminus I : i \neq j \Rightarrow w_{ij} = 0$ gilt, optimiert der (1+1)-EA f in erwarteter Zeit $O(n^k + n^2)$.*

Beweis: Per Umordnung der Indizes nehmen wir wiederum o. B. d. A. an, dass für alle i, j mit $k+1 \leq i < j \leq n$ die Gleichung $w_{ij} = 0$ gültig ist, d. h. dass nur in den ersten k Zeilen der Koeffizientenmatrix außerhalb der Diagonalen Einträge auftreten, und lassen erneut die Monotonie $w_1 \geq \cdots \geq w_n$ fallen. Außerdem gelte $I \neq \emptyset$. Nun betrachten wir $K := 2^k$ lineare Funktionen f_i, $i \in \{1, \ldots, K\}$, die von den Variablen $x_{k+1}, \ldots, x_n$ abhängen und Subfunktionen von f verkörpern. Es bezeichne für ein $i \in \{1, \ldots, K\}$ der Vektor $b_i \in \{0,1\}^k$ mit $b_i = (b_{i1}, \ldots, b_{ik})$ die Binärdarstellung von i, d. h. $\sum_{j=1}^{k} b_{ij} \cdot 2^{j-1} = i$. Damit können wir die Funktionen f_i in der Gestalt

$$f_i(x_{k+1}, \ldots, x_n) = \sum_{j=1}^{k} w_j b_{ij} + \sum_{j=1}^{k} \sum_{l=j+1}^{k} w_{jl} b_{ij} b_{il} + \sum_{j=k+1}^{n} w_j x_j + \sum_{j=1}^{k} \sum_{l=k+1}^{n} w_{jl} b_{ij} x_l$$

[1] Eine hinreichende Bedingung ist das Eintreten einer Mutation, die genau ein Bit aus $x_2, \ldots, x_n$ ändert und x_1 unverändert lässt. Die Wahrscheinlichkeit dafür ist durch $\binom{n-1}{1}(1/n)(1-1/n)^{n-1} \geq e^{-1}(n-1)/n$ nach unten beschränkt.

darstellen, denn die quadratischen Monome der Summe $\sum_{j=k+1}^{n}\sum_{l=j+1}^{k} w_{jl}x_jx_l$ sind nach Voraussetzung stets 0. Die Funktionen f_i sind linear, denn die ersten beiden Summen in der obigen Darstellung sind nicht von $x_{k+1},\dots,x_n$ abhängig, und die letzten beiden Summen enthalten nur lineare Terme. Wir sprechen davon, dass die lineare Funktion f_i, $i \in \{1,\dots,K\}$ *aktiv* ist, wenn $\sum_{j=1}^{k} x_j 2^{j-1} = i$ gilt. Außerdem ziehen wir für jede der Funktionen $f_i : \{0,1\}^{n-k} \to \mathbb{Z}$ ein Fortschrittsmaß H_i^* gemäß Definition 4.3 heran. Da die f_i Subfunktionen von f sind, d. h. durch Konstantsetzungen der ersten k Variablen konstruiert wurden, entspricht der Funktionswert von $f(x_1,\dots,x_n)$ immer dem Wert der aktiven Funktion $f_i(x_{k+1},\dots,x_n)$, sodass die an den Zeitpunkten, zu denen f_i aktiv ist, betrachteten Werte der Maße H_i^* im Laufe des (1+1)-EA monoton fallen.

Nun wenden wir Lemma 4.6 an. Unabhängig davon, welche Funktion gerade aktiv ist, beträgt die Wahrscheinlichkeit, den maximal erreichbaren Hammingabstand einer Funktion f_i um mindestens 1 zu senken, mindestens $e^{-1}n^{-1}$ (Lemma A.9). Somit sinkt nach erwarteten en Schritten mindestens eines der Maße H_i^* (nämlich mindestens das Maß der zu dem Zeitpunkt aktiven Funktion) um mindestens 1. Nimmt die aktive Funktion bereits einen optimalen Wert an, der aber für die quadratische Funktion f noch nicht optimal ist, muss eine Mutation an den Bits $x_1,\dots,x_k$ stattfinden, um eine andere Funktion zu aktivieren. Mit einer Wahrscheinlichkeit von mindestens $n^{-k}(1-1/n)^{n-k} \geq e^{-1}n^{-k}$ wird eine noch nicht optimierte lineare Funktion aktiviert, sodass der Funktionswert von f durch Mutationen an den Bits $x_{k+1},\dots,x_n$ weiter erhöht werden kann. Dies ist nach maximal erwarteten en^k Schritten der Fall und kann höchstens K-mal erforderlich sein. Außerdem können maximal $K(n-k)$ Senkungen von Fortschrittsmaßen vorgenommen werden, wozu maximal erwartete $(K(n-k))en = Ken^2 - Kken$ Schritte ausreichen. Somit sind nach maximal erwarteten $(Ken^k) + (Ken^2 - Kken) = O(n^k + n^2)$ Schritten alle Funktionen f_i, die aktiv werden mussten, aktiv gewesen und ihre Fortschrittsmaße jeweils auf 0 gesunken. Damit erhalten wir eine obere Schranke für die erwartete Laufzeit des (1+1)-EA auf f. □

Bemerkung 4.5 Wir können eine Funktion, die die Voraussetzungen von Satz 4.8 erfüllt und für die auch die untere Schranke von $\Omega(n^k)$ für die erwartete Laufzeit bewiesen wird, explizit angeben. Im Vorgriff auf den nächsten Abschnitt erwähnen wir die mit der Konstanten k parametrisierte Funktion

$$\textsc{Dist}_k(x) = \sum_{i=1}^{k} (5n - 3kn)\, x_i + \sum_{i=k+1}^{n} x_i + \sum_{i=1}^{k}\sum_{j=i+1}^{k} 6n \cdot x_i x_j,$$

die nur in den ersten k Zeilen ihrer Koeffizientenmatrix außerhalb der Hauptdiagonalen Einträge besitzt, aber laut Satz 4.12 mindestens erwartete $\Omega(n^k)$ Schritte zur Optimierung durch den (1+1)-EA benötigt.

Wir fragen uns in Anbetracht der Ergebnisse von Satz 4.8 nun, ob eine quadratische Funktion, bei der die Anzahl quadratischer Koeffizienten linear ist, überhaupt exponentielle erwartete Laufzeiten des (1+1)-EA hervorrufen kann. (Das Standardbeispiel für exponentielle Laufzeiten DISTANCE besitzt nämlich $\Omega(n^2)$ quadratische Gewichte.) In der folgenden

Definition stellen wir als Antwort darauf eine Funktion vor, die nur n quadratische Koeffizienten hat, aber „schwierig“ ist.

Definition 4.4 *Die Funktion* CHAIN $: \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\text{CHAIN}(x) := \sum_{i=1}^{n}(1-2n)x_i + \sum_{i=1}^{n} 2nx_i x_{(i \bmod n)+1}.$$

Lemma 4.9 *Die Funktion* CHAIN *hat in* $\vec{0}$ *ein lokales Maximum mit der Eigenschaft* $\forall x \in \{0,1\}^n \setminus \{\vec{0},\vec{1}\} : \text{CHAIN}(x) < \text{CHAIN}(\vec{0})$.

Beweis: Es ist $\text{CHAIN}(\vec{0}) = 0$ und $\text{CHAIN}(\vec{1}) = n(1-2n) + n(2n) = n$. Wir betrachten ein beliebiges $x \in \{0,1\}^n \setminus \{\vec{0},\vec{1}\}$. Sei $o := x_1 + \cdots + x_n$ die Zahl der Einsen in x. Es gilt $\sum_{i=1}^{n}(1-2n)x_i = (1-2n)o = o-2no$. Anhand der Menge $Z := \{i \in \{1,\ldots,n\} | x_i = 0\}$ der Indizes der Nullbits zeigen wir nun, dass mindestens $n-o+1$ Terme $x_i x_{(i \bmod n)+1}$ den Wert 0 annehmen. Zum einen gilt für alle $i \in Z : x_i x_{(i \bmod n)+1} = 0$. Dies sind $n-o$ verschiedene Terme. Zum anderen existiert wegen $\vec{0} \neq x \neq \vec{1}$ ein $j \in \{1,\ldots,n\}$ mit $x_j = 1$ und $x_{(j \bmod n)+1} = 0$. Der Term $x_j x_{(j \bmod n)+1}$ nimmt ebenfalls den Wert 0 an und ist von den $n-o$ aufgezählten Termen verschieden. Es nehmen also mindestens $n-o+1$ der n quadratischen Terme den Wert 0 an, sodass $\sum_{i=1}^{n} 2nx_i x_{(i \bmod n)+1} \leq 2n(n-(n-o+1)) = 2no-2n$ und somit

$$\text{CHAIN}(x) \leq (o-2no) + (2no-2n) = o-2n < \text{CHAIN}(\vec{0}) \text{ (denn } o < n)$$

gilt. □

Informal gesagt existiert bei dieser Funktion das lokale Maximum in $\vec{0}$ mit dem Hammingabstand n zu $\vec{1}$ trotz der größeren Beträge der quadratischen gegenüber den linearen Gewichten, weil bei nur einem Nullbit im Bitstring schon zwei quadratische Monome „wegfallen“. Mittlerweile bekannt ist, dass sich solch große Hammingabstände negativ auf die erwarte Laufzeit des (1+1)-EA auswirken. Wie in Lemma 3.14 zeigen wir:

Lemma 4.10 *Der (1+1)-EA benötigt zur Optimierung der Funktion* CHAIN *erwartete* $2^{\Omega(n \log n)}$ *Schritte.*

Beweis: Hat der (1+1)-EA den Vektor $\vec{0}$ zum aktuellen Bitstring, ist eine Mutation von n Bits zugleich erforderlich, um eine Mutation zu $\vec{1}$, dem einzigen Bitstring mit mindestens ebenso gutem Funktionswert, zu vollführen. Wegen der Wahrscheinlichkeit n^{-n} für dieses Ereignis beträgt die erwartete Zeit dafür n^n. Zusammen mit der Wahrscheinlichkeit von 2^{-n} für einen Start in $\vec{0}$ trägt dies einen Summanden von mindestens $2^{-n+\Omega(n \log n)} = 2^{\Omega(n \log n)}$ zur erwarteten Laufzeit bei. □

Bemerkung 4.6 Es wäre interessant zu wissen, ob die Funktion CHAIN ähnlich „trügerisch“ für den (1+1)-EA wie ein Quadrat einer linearen Funktion, z. B. $\text{ONESQR}_{-n/2+1/3}$, ist, d. h. dass die Wahrscheinlichkeit, nach der Initialisierung zunächst zum lokalen Optimum in $\vec{0}$ zu gelangen, durch eine Konstante nach unten beschränkt ist. Leider ist die Funktion im Gegensatz zu ONESQR nicht symmetrisch[2], denn $\text{CHAIN}(1,1,0,\ldots,0) = 2-4n+2n = 2-2n$, wohingegen $\text{CHAIN}(1,0,1,0,\ldots,0) = 2-4n$ gilt. Dies erschwert die Analyse des Verhaltens des (1+1)-EA. Im Rahmen dieses Kapitels nehmen wir von weiteren Betrachtungen von CHAIN Abstand. Allerdings versuchen wir in Abschnitt 5.2 in einem anderen Zusammenhang etwas mehr über CHAIN in Erfahrung zu bringen.

Bemerkung 4.7 Jansen stellt in [Jan99] verschiedene Modelle zur Klassifikation von Fitnessfunktionen vor. Die Funktion CHAIN können wir in das *NK*-Modell mit $K = 1$ und aufeinander folgenden Positionen (*adjacent NK model*) einordnen, für das Thompson und Wright in [TW96] einen Polynomialzeitalgorithmus zur Optimierung angeben. Wir schlussfolgern daraus, dass der simple (1+1)-EA bereits auf einer Klasse von Fitnessfunktionen, die als einfach bekannt ist, versagen kann. In Kapitel 5 werden wir uns näher mit der Komplexität der Optimierung von quadratischen Funktionen beschäftigen.

Zusammenfassend haben wir in diesem Abschnitt gesehen, dass quadratische Funktionen mit linear vielen quadratischen Koeffizienten je nach deren Anordnung in der Koeffizientenmatrix entweder noch in polynomieller erwarteter Zeit des (1+1)-EA optimiert werden oder sogar eine exponentielle erwartete Laufzeit erfordern. Die Funktion CHAIN besitzt in jeder Zeile ihrer Koeffizientenmatrix einen Eintrag außerhalb der Hauptdiagonalen und ist schwierig, während durchaus linear viele Koeffizienten, die in nur konstant vielen „Zeilen“ (bis auf Umordnung) auftreten, noch keine exponentiellen erwarteten Laufzeiten erzwingen können.
Im nächsten Abschnitt werden wir die Auswirkungen einer Beschränkung auf konstant viele quadratische Koeffizienten näher analysieren.

4.3 Konstant viele quadratische Gewichte

In diesem Abschnitt wollen wir uns auf fast lineare Funktionen gemäß Definition 2.5, d. h. Funktionen mit konstant vielen quadratischen Gewichten, konzentrieren und die Sichtweise annehmen, dass deren quadratische Gewichte „Störeinflüsse“ darstellen und die Linearität der Funktion beeinflussen.

4.3.1 Von der Zahl quadratischer Gewichte abhängige Schranken

Wir halten in einer einfachen Folgerung aus den Ergebnissen von Abschnitt 4.2 eine allgemeine obere Schranke für fast lineare Funktionen fest, in die nur die Anzahl quadratischer Gewichte eingeht.

[2]Die Werte symmetrischer Funktionen hängen nur von der Zahl der Einsen im Bitstring ab.

Lemma 4.11 *Sei f eine quadratische Funktion, deren Koeffizientenmatrix außerhalb der Diagonalen nur k Einträge, $k \in \mathbb{N}$ konstant, hat. Der (1+1)-EA optimiert f in erwarteter polynomieller Zeit von höchstens $O(n^k + n^2)$.*

Beweis: Wir wollen Satz 4.8 anwenden und eine Menge I möglichst geringer Mächtigkeit finden, sodass für alle $i, j \in \{1, \ldots, n\} \setminus I$ gilt: $i \neq j \Rightarrow w_{ij} = 0$. Aus der Menge $I^* := \{(i,j) | 1 \leq i < j \leq n \wedge w_{ij} \neq 0\}$, die die Indexpaare der quadratischen Gewichte enthält, konstruieren wir I, indem wir in I die Projektionen aller Elemente von I^* auf deren erste Komponente aufnehmen.[3] Gäbe es ein $i \in \{1, \ldots, n\} \setminus I$ und ein $j \in \{i+1, \ldots, n\} \setminus I$ mit der Eigenschaft $w_{ij} \neq 0$, wäre $(i,j) \in I^*$ und damit $i \in I$; Widerspruch. Also erfüllt I die oben angegebenen Voraussetzungen. Da es nur k quadratische Gewichte gibt, gilt $\#I^* = k$, also $\#I \leq k$, und damit folgt die Behauptung aus Satz 4.8. □

Andererseits können wir anhand des Beispiels DISTANCE eine Klasse fast linearer Funktionen konstruieren, bei denen die erwartete Laufzeit des (1+1)-EA durch ein von der Zahl quadratischer Gewichte abhängiges Polynom nach unten beschränkt ist.

Definition 4.5 *Die quadratische Funktion* $\text{DIST}_k : \{0,1\}^n \to \mathbb{R}$*, wird für eine Konstante* $k \in \mathbb{N}$*,* $k < n$*, definiert durch*

$$\text{DIST}_k(x) = \sum_{i=1}^{k} (5n - 3kn)\, x_i + \sum_{i=k+1}^{n} x_i + \sum_{i=1}^{k} \sum_{j=i+1}^{k} 6n \cdot x_i x_j.$$

Eingeschränkt auf die Bits $x_{k+1}, \ldots, x_n$ und für eine feste Belegung $\hat{x}_1, \ldots, \hat{x}_k$ von $x_1, \ldots, x_k$ entspricht DIST_k der linearen Funktion

$$f(x_{k+1}, \ldots, x_n) = \text{ONEMAX}(x_{k+1}, \ldots, x_n) + \sum_{i=1}^{k} (5n - 3kn)\, \hat{x}_i + \sum_{i=1}^{k} \sum_{j=i+1}^{k} 6n \cdot \hat{x}_i \hat{x}_j,$$

also ONEMAX plus einer additiven Konstante. Die auf die Bits $x_1, \ldots, x_k$ eingeschränkte Funktion mit einer festen Belegung $\hat{x}_1, \ldots, \hat{x_k}$ von $x_1, \ldots, x_k$ hingegen ist aus einer Ausprägung eines Quadrats einer linearen Funktion[4], nämlich

$$f(x_1, \ldots, x_k) = 3n \left(\sum_{i=1}^{k} x_i - \frac{k}{2} + \frac{1}{3} \right)^2 - 3n \left(\frac{k^2}{4} - \frac{k}{3} + \frac{1}{9} \right) + \sum_{i=k+1}^{n} \hat{x}_i,$$

entstanden, ähnlich zu $\text{ONESQR}_{-n/2+1/3}$, vgl. dazu die Rechnung zu Beginn des Kapitels. Die hier vorgenommene Multiplikation mit 3 macht die Gewichte nach der Quadrierung

[3] Wir konstruieren gewissermaßen ein „Vertex Cover“ auf dem Graphen $G = (V, E)$ mit $V = \{1, \ldots, n\}$ und $E = \{\{i,j\} | w_{ij} \neq 0\}$.

[4] Für die in der Definition nicht vorgesehene Belegung $k := n$ erhielten wir eine Funktion, die DISTANCE bis auf Komplementbildung nach Lemma 3.5, Multiplikation mit einem Faktor und eine additive Konstante identisch ist.

ganzzahlig, der zusätzliche Faktor n hat den Zweck, die geringstmögliche positive Differenz der Werte von $f(x_1, \ldots, x_k)$ für eine feste Belegung von $x_{k+1}, \ldots, x_n$ größer als die größtmögliche positive Differenz der Werte von $f(x_{k+1}, \ldots, x_n)$ für eine feste Belegung von $x_1, \ldots, x_k$ zu machen. Erstere beträgt mindestens n. Es gilt nämlich zunächst für die lineare Funktion $\sum_{i=1}^{k} x_i - k/2 + 1/3$, dass die minimale positive Differenz der Beträge von Funktionswerten genau $1/3$ ist (tritt z. B. bei Vektoren mit $k/2$ Einsen gegenüber Vektoren mit $k/2 - 1$ Einsen ein), und für die Quadrate der Funktionswerte folgt somit

$$a - b \geq \frac{1}{3} \Rightarrow (a+b)(a-b) \geq (a+b)\frac{1}{3} \Rightarrow a^2 - b^2 \geq \frac{a+b}{3},$$

wobei für a, b mit $a \neq b$ beliebige Beträge von Funktionswerten der betrachteten linearen Funktion eingesetzt werden dürfen. Weil $a + b \geq 1$ (es gibt einen Funktionswert mit dem Betrag $1/3$, alle übrigen verschiedenen Funktionswerte sind vom Betrag her mindestens $2/3$), folgt für jede positive Differenz von Funktionswerten der oben erwähnten Funktion $f(x_1, \ldots, x_k)$ nach einer Multiplikation mit $3n$ ein Wert von mindestens n. Andererseits sehen wir bei der auf $x_{k+1}, \ldots, x_n$ eingeschränkten Funktion sofort, dass sie für eine feste Belegung von $x_1, \ldots, x_k$ eine Änderung von höchstens $n - k$ im Funktionswert erfahren kann.
Im Bereich $x_1, \ldots, x_k$ kann es bei der Optimierung der Funktion $\textsc{Dist}_k$ zu dem bekannten Phänomen kommen, dass die Zahl der Einsen dort auf null sinkt und ein Flipp von k Bits gleichzeitig erforderlich ist, was im folgenden Satz eine untere polynomielle Schranke bewirkt.

Satz 4.12 *Der (1+1)-EA benötigt zur Optimierung der Funktion* $\textsc{Dist}_k$ *mindestens erwartete* $\Omega(n^k)$ *Schritte.*

Beweis: Mit konstanter Wahrscheinlichkeit 2^{-k} werden die Bits $x_1^s, \ldots, x_k^s$ des initialisierten Bitstrings x_s mit 0 belegt. Um innerhalb dieser Positionen infolge einer akzeptierten Mutation Einsbits zu erzeugen, müssen hier alle k Bits zugleich flippen, während selbst ein Flipp von $n - k$ Nullbits an den übrigen Positionen nicht zusammen mit dem Flipp von weniger als k Bits aus $\{x_1, \ldots, x_k\}$ akzeptiert werden kann, da sich der Funktionswert wegen der hohen Gewichte an den Positionen $x_1, \ldots, x_k$ verringern würde. Die Wahrscheinlichkeit, dass die Bits $1, \ldots, k$ zugleich flippen, ist nach oben beschränkt durch n^{-k}, d. h. die erwartete Zeit bis zu einer solchen Mutation beträgt mindestens n^k Schritte, was mit einem konstanten Faktor 2^{-k} in den Erwartungswert der Laufzeit eingeht. □

Auch eine von k abhängige polynomielle obere Schranke ist rasch einzusehen.

Satz 4.13 *Der (1+1)-EA benötigt zur Optimierung der Funktion* $\textsc{Dist}_k$ *höchstens erwartete* $O(n^k + n \ln n)$ *Schritte.*

Beweis: Im Fall $k = 1$ ist $\textsc{Dist}_k$ eine lineare Funktion, für die die in diesem Zusammenhang uninteressante Schranke $O(n \ln n)$ gilt.

Indem wir $I := \{1, \ldots, k\}$ setzen, folgt die oberere Schranke im Fall $k \neq 1$ direkt aus Satz 4.8. Um hinreichende Bedingungen für eine Optimierung von $\textsc{Dist}_k$ durch den (1+1)-EA vorzustellen, präsentieren wir außerdem eine abweichende Beweisstrategie.

Ähnlich wie in Satz 4.12 halten wir fest, dass nach erwarteten $O(n^k)$ Schritten die Bits $x_1, \ldots, x_k$ alle zu ihrer optimalen Belegung mit 1 flippen, denn die Wahrscheinlichkeit einer solchen Mutation ist durch $n^{-k} \cdot (1 - 1/n)^{n-k} \geq e^{-1}n^{-k}$ nach unten beschränkt. Da im Folgenden keine Mutation, die Nullbits an den Positionen $x_1, \ldots, x_k$ erzeugt, mehr akzeptiert wird, ist nur noch die erwartete Zeit, bis die Bits $x_{k+1}, \ldots, x_n$ zu 1 geflippt sind, nach oben abzuschätzen. Wir befinden uns bis auf die Beständigkeit der Einsen in den ersten k Positionen in derselben Situation, als ob bei OneMax bereits mindestens k Einsen im aktuellen Bitstring stehen (deren Positionen können sich hingegen ändern, was für uns aber unerheblich ist), sodass wir die erwartete Zeit, bis höchstens $n - k$ Nullbits zu 1 geflippt sind, durch $O(n \ln n)$ beschränken können. □

4.3.2 Von der Struktur abhängige Schranken

Trotz der gerade bewiesenen Schranke $O(n^k)$ (wenn $k \geq 2$) für $\textsc{Dist}_k$ liefert Lemma 4.11 wegen $\binom{k}{2}$ quadratischer Gewichte die deutlich schlechtere Schranke $O(n^{(k^2-k)/2} + n^2)$. Dies begründet sich daraus, dass in Lemma 4.11 keine Voraussetzungen über die Anordnung der quadratischen Gewichte in der Koeffizientenmatrix eingehen, sondern lediglich die Anzahl quadratischer Gewichte. Wenn wir etwas über die Struktur der Funktion, nämlich die Anzahl der „Zeilen“ (bis auf Umordnung der Indizes) der Koeffizientenmatrix, in der quadratische Gewichte auftreten, wissen, erhalten wir mithilfe des komplexeren Satzes 4.8 asymptotisch bessere Schranken und für $\textsc{Dist}_k$ in der Tat die asymptotisch korrekte Schranke $O(n^k)$, wenn $k \geq 2$.

Allerdings ist es leicht möglich, eine fast lineare Funktion anzugeben, bei der sowohl Lemma 4.11 als auch Satz 4.8 obere Schranken angeben, die wir nicht wirklich für exakt halten. Eine Funktion wie $f(x) = \sum_{i=1}^{n} n^3 x_i - \sum_{i=1}^{k} \sum_{j=i+1}^{k} x_i x_j$ mit großen linearen Gewichten ist für den (1+1)-EA zu linearen Funktionen (hier ähnlich wie OneMax, da die Zahl der Einsen im aktuellen Bitstring im Verlaufe der Optimierung nicht sinken kann) äquivalent. Die folgende fast lineare Funktion erscheint dem (1+1)-EA zwar nicht mehr linear, sieht aber dennoch einfach aus.

Definition 4.6 *Die Funktion* $\textsc{Pair}_k : \{0,1\}^n \to \mathbb{R}$ *wird für eine Konstante* $k \in \mathbb{N}$, $k < n$, *definiert durch*

$$\textsc{Pair}_k(x) := \sum_{i=1}^{n}(-x_i) + \sum_{i=1}^{k} 3x_{2i-1}x_{2i}.$$

Die Menge $I := \{2i | i \in \{1, \ldots, k\}\}$ erfüllt für $\textsc{Pair}_k$ die Voraussetzungen von Satz 4.8 und hat die Mächtigkeit $\#I = k$; Mengen geringerer Mächtigkeit erfüllen die Voraussetzungen des Satzes offenbar nicht. Mithin gibt Satz 4.8 eine obere Schranke $O(n^k + n^2)$ ebenso wie Lemma 4.11 an. Wir glauben aber, dass die erwartete Laufzeit des (1+1)-EA auf

$\textsc{Pair}_k$ durch höchstens $O(n^2)$ für alle konstanten k nach oben beschränkt ist. (Diese obere Schranke hängt also nicht einmal von k ab.) Eine Intuition gewinnen wir dafür, indem wir sehen, dass die Bits $x_1, \ldots, x_{2k}$ mit höchstens k aufeinander folgenden akzeptierten 2-Bit-Mutationen[5] zu einer optimalen Belegung geführt werden können und dass die Funktion eingeschränkt auf die übrigen Bits linear ist.

Bei $\textsc{Pair}_k$ sind je zwei Paare $(2i-1, 2i)$ von Indizes für $i \in \{1, \ldots, k\}$ wegen $w_{ij} \neq 0$ miteinander „verknüpft“; alle anderen Paare von Indizes sind in diesem Sinne nicht verknüpft. Bei $\textsc{Dist}_k$ hingegen sind jede zwei verschiedene Indizes $i, j \in \{1, \ldots, k\}$, $i < j$, wegen $w_{ij} \neq 0$ miteinander verknüpft. Dies bringt uns auf den Gedanken, anhand eines „Verknüpfungsgrades“ den Grad des Polynoms, durch das die erwartete Laufzeit auf einer fast linearen Funktion nach oben beschränkt ist, auszumachen.

Wir deuten zunächst an, dass eine fast lineare Funktion gefunden werden kann, bei der es genügt, dass k quadratische Gewichte k Indizes *transitiv* miteinander verknüpfen, um erwartete Laufzeiten von $\Omega(n^k)$ zu hervorzurufen. (Die obere Schranke $O(n^k)$ gilt wegen Lemma 4.11 sowieso.) Dazu greifen wir die Funktion $\textsc{Chain}$ aus Abschnitt 4.2 auf.

Definition 4.7 *Die Funktion* $\textsc{Chain}_k : \{0,1\}^n \to \mathbb{R}$ *wird für eine Konstante* $k \in \mathbb{N}$, $k < n$, *definiert durch*

$$\textsc{Chain}_k(x) := \sum_{i=1}^{k} n(1-2k)x_i + \sum_{i=1}^{k} 2nkx_i x_{(i \bmod k)+1} + \sum_{i=k+1}^{n} x_i.$$

Eingeschränkt auf die ersten k Bits entspricht $\textsc{Chain}_k$ der Funktion $\textsc{Chain}$ auf k Bits, die mit n multipliziert wird und zu der ein Summand zwischen 0 und $n-k$ addiert wird. Wir sehen an der Ganzzahligkeit der Gewichte, dass die geringstmögliche positive Differenz verschiedener Funktionswerte von $\textsc{Chain}$ mindestens 1 beträgt. Also macht die Multiplikation mit n die geringstmögliche positive Differenz der Funktionswerte von $\textsc{Chain}_k$ bei einer Änderung der Belegung der ersten k Bits größer als die größtmögliche positive Differenz bei einer Änderung der Bits $x_{k+1}, \ldots, x_n$. Dann kann Satz 4.12 statt für $\textsc{Dist}_k$ für $\textsc{Chain}_k$ formuliert werden und ein analoger Beweis geführt werden. (Wir erinnern uns, dass $\textsc{Chain}$ in $\vec{0}$ ein lokales Optimum hat.)

Ohne die Beweise für $\textsc{Dist}_k$ nochmals in analoger Form für $\textsc{Chain}_k$ zu führen, erkennen wir, dass fast lineare Funktionen mit k quadratischen Koeffizienten schon $\Theta(n^k)$ erwartete Schritte erfordern können, indem die quadratischen Gewichte k Indizes transitiv miteinander verknüpfen. Zum einen haben wir damit eine Funktion gefunden, für die die obere Schranke aus Lemma 4.11 asymptotisch optimal ist. Zum anderen legt diese Funktion die folgende formale Erfassung der „Verknüpfung“ mithilfe einer Äquivalenzrelation nahe. Daran zeigen wir dann eine obere Schranke für fast lineare Funktionen, die z. B. auch für $\textsc{Pair}_k$ realistisch erscheint.

[5] Andererseits besitzt $\textsc{Pair}_k$ auch lokale Maxima mit einem Hammingabstand von 2 zu Bitstrings mit mindestens ebenso hohem Funktionswert, z. B. von $\vec{0}$ zu $(1, 1, 0, \ldots, 0)$.

Definition 4.8 *Zwei Indizes $i, j \in \{1, \ldots, n\}$, $i \neq j$, heißen bezüglich einer quadratischen Funktion $f : \{0,1\}^n \to \mathbb{R}$ mit $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$* quadratisch verknüpft*, Notation $i \equiv_f j$, wenn eine Folge $i_1, \ldots, i_k$ mit $i_1 = i$ und $i_k = j$ von paarweise verschiedenen Indizes existiert, sodass für alle $l \in \{1, \ldots, k-1\}$ eine der Ungleichungen $w_{i_l i_{l+1}} \neq 0$ bzw. $w_{i_{l+1} i_l} \neq 0$ erfüllt ist. Außerdem gilt für $i \in \{1, \ldots, n\}$ die Relation $i \equiv_f i$.*

Bemerkung 4.8 Die Relation $\equiv_f$ ist eine Äquivalenzrelation. Wenn wir die Symmetrie der Relation ignorieren, können wir die quadratische Verknüpfung anhand eines aus der quadratischen Funktion konstruierten ungerichteten Graphen $G = (V, E)$ mit $V = \{1, \ldots, n\}$ und $E = \{\{i, j\} | 1 \leq i < j \leq n, w_{ij} \neq 0\}$ veranschaulichen. Stehen zwei Indizes i und j, $i \neq j$, bezüglich $\equiv_f$ in Relation, ist dies äquivalent dazu, dass im Graphen G ein ungerichteter Pfad vom Knoten i zum Knoten j existiert. Somit lässt sich für alle Paare $\{i, j\}$ mit $i \neq j$ leicht berechnen, ob $i \equiv_f j$ gilt, indem wir den transitiven Abschluss der Adjazenzmatrix des Graphen, gegeben durch $A = (a_{ij})$ mit $a_{ij} = 1$, falls $w_{ij} \neq 0$ oder $w_{ji} \neq 0$, und $a_{ij} = 0$ sonst für alle $i, j \in \{1, \ldots, n\}$, $i \neq j$, bilden.

Definition 4.9 *Gegeben sei eine quadratische Funktion $f : \{0,1\}^n \to \mathbb{R}$ mit $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$. Für einen Index $i \in \{1, \ldots, n\}$ bezeichnet $[i]$ die Äquivalenzklasse bezüglich der für f definierten Relation $\equiv_f$, d. h. $[i] = \{j \in \{1, \ldots, n\} | i \equiv_f j\}$.*

Definition 4.10 *Gegeben sei eine quadratische Funktion $f : \{0,1\}^n \to \mathbb{R}$ mit $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$. Für einen Index $i \in \{1, \ldots, n\}$ definieren wir die von $[i]$ induzierte Subfunktion $f_{[i]} : \{0,1\}^{\#[i]} \to \mathbb{R}$ als*

$$f_{[i]}\left(\{x_j | j \in [i]\}\right) := \sum_{j \in [i]} w_j x_j + \sum_{\substack{j,k \in [i] \\ j<k}} w_{jk} x_j x_k.$$

Bei den Funktionen $f_{[i]}$ handelt es sich um Subfunktionen, die nur von Variablen mit Indizes aus $[i]$ abhängen, während alle übrigen Variablen gleich 0 gesetzt werden. Wir zeigen nun, dass jede quadratische Funktion f in k Subfunktionen $f_{[i]}$ separiert werden kann, wobei k die Zahl der Äquivalenzklassen bezüglich $\equiv_f$ ist.

Lemma 4.14 *Gegeben sei eine quadratische Funktion $f : \{0,1\}^n \to \mathbb{R}$ mit $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$. Sei außerdem $R \subseteq \{1, \ldots, n\}$ ein minimales vollständiges Repräsentantensystem für die bezüglich f definierte Äquivalenzrelation $\equiv_f$. Dann lässt sich f wie folgt in $\#R$ Funktionen separieren:*

$$f(x_1, \ldots, x_n) = \sum_{i \in R} f_{[i]}\left(\{x_j | j \in [i]\}\right).$$

Beweis: Wir zeigen die Identität der Polynomdarstellungen

$$S_1 := \sum_{i=1}^{n} w_i x_i + \sum_{j=1}^{n} \sum_{k=j+1}^{n} w_{jk} x_j x_k \quad \text{und} \quad S_2 := \sum_{i \in R} \sum_{j \in [i]} w_j x_j + \sum_{i \in R} \sum_{\substack{j,k \in [i] \\ j<k}} w_{jk} x_j x_k,$$

indem wir für jedes Monom in S_1, das nicht konstant 0 ist, nachweisen, dass es genau einmal in S_2 enthalten ist. Für die linearen Monome $w_i x_i$ folgt dies sofort, da R ein vollständiges minimales Repräsentantensystem für die Äquivalenzklassen bezüglich $\equiv_f$ ist. Sei nun $w_{jk}x_jx_k$ mit $1 \leq j < k \leq n$ und $w_{jk} \neq 0$ ein beliebiges nicht konstantes quadratisches Monom aus S_1. Aufgrund der Definition von $\equiv_f$ folgt, dass $j \equiv_f k$ gilt. Sei $i \in R$ der Repräsentant der Äquivalenzklasse, die j und k enthält. Da aufgrund der Minimalität von R in S_2 nur genau einmal über die Indizes einer Äquivalenzklasse summiert wird, tritt $w_{jk}x_jx_k$ in S_2 genau einmal, nämlich bei der Summenbildung über $[i]$, auf. □

Nach diesen technischen Vorbereitungen können wir schließlich anhand der Äquivalenzrelation $\equiv_f$ eine obere Schranke für die Komplexität von quadratischen Fitnessfunktionen für den (1+1)-EA festhalten.

Satz 4.15 *Sei $f : \{0,1\}^n \to \mathbb{R}$ mit $f(x) = \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j$ eine quadratische Funktion. Es bezeichne $m := \max\{\#[i] | i \in \{1,\ldots,n\}\}$ die maximale Mächtigkeit der Äquivalenzklassen bezüglich der für f definierten Relation $\equiv_f$. Dann gilt: Der (1+1)-EA optimiert f in erwarteter Zeit $O(2^m n^{m+1})$.*

Beweis: Wir ziehen ein minimales vollständiges Repräsentantensystem $R \subseteq \{1,\ldots,n\}$ für die Äquivalenzrelation $\equiv_f$ heran und setzen $r := \#R$. Nach Lemma 4.14 können wir f in der Form $f(x_1,\ldots,x_n) = \sum_{i \in R} f_{[i]}(\{x_j | j \in [i]\})$ separieren. Im Fall $r = n$ ist f eine lineare (bitweise separierbare) Funktion, für die wir schon in verschiedenen Fällen, z. B. in Lemma 4.7, mithilfe von Fortschrittsmaßen obere Schranken gezeigt haben. Die Besonderheit im Fall $r = n$ liegt darin, dass jede Subfunktion nur zwei verschiedene Werte annehmen kann, da sie nur von einem Bit abhängt. Hängt eine Subfunktion hingegen von $k > 1$ Bits ab, kann sie maximal 2^k verschiedene Funktionswerte annehmen. Dies erschwert die Übertragung der üblichen Argumentation bei der Nutzung von Fortschrittsmaßen für lineare Funktionen auf Funktionen, die nur in r Funktionen, $r < n$, separiert werden können. Wir können die Idee des Fortschrittsmaßes aber verallgemeinern.
Für jede Subfunktion $f_{[i]}$, $i \in R$, definieren wir $l_{[i]} := \min\{f_{[i]}(x) | x \in \{0,1\}^{\#[i]}\}$ und $h_{[i]} := \max\{f_{[i]}(x) | x \in \{0,1\}^{\#[i]}\}$ als minimalen bzw. maximalen Funktionswert der Subfunktion und $V_{[i]} := \{f_{[i]}(x) | x \in \{0,1\}^{\#[i]}\}$ als Menge der Funktionswerte von $f_{[i]}$. Zu jeder Menge $V_{[i]}$ stellen wir die Folge $v^j_{[i]}$, $j \in \{1,\ldots,\#V_{[i]}\}$, als monoton fallende Umordnung der Elemente von $V_{[i]}$ auf und definieren daraus wiederum die (nicht unbedingt monoton fallende) nicht negative Folge $d^j_{[i]}$ mit $d^j_{[i]} := v^j_{[i]} - v^{j+1}_{[i]}$ für $j \in \{1,\ldots,\#V_{[i]} - 1\}$ und $d^0_{[i]} := 0$, die die Differenzen der Folgenglieder $v^j_{[i]}$ darstellt. Es gilt $\sum_{j=0}^{\#V[i]-1} d^j_{[i]} = h_{[i]} - l_{[i]}$, da es sich hier um eine Teleskopsumme handelt. Aus der Menge $V := \{(i, d^j_{[i]}) | i \in R, j \in \{0,\ldots,\#V_{[i]} - 1\}\}$ konstruieren wir schließlich mit $v := \#V$ die monoton fallende nicht negative Folge v^*_i, $i \in \{1,\ldots,v\}$, aus einer Umordnung der Familie, die wir aus den Projektionen aller Elemente von V auf die zweite Komponente erhalten. Zuletzt definieren wir anhand der Folge v^*_i mithilfe von $f_{\min} := \min\{f(x) | x \in \{0,1\}^n\}$ das Fortschrittsmaß P^* für $x \in \{0,1\}^n$ als

$$P^*(x) := \max\left\{ j \in \{0,\ldots,v\} | f(x) \geq \sum_{i=1}^{j} v^*_i + f_{\min} \right\}.$$

Wegen der Separierbarkeit von f gilt offensichtlich $f_{\min} = \sum_{i \in R} l_{[i]}$ und $f_{\max} = \sum_{i \in R} h_{[i]}$. Da die Folge v_i^* nur eine Umordnung der Projektionen der Elemente von V ist, folgt

$$\sum_{i=1}^{v} v_i^* + f_{\min} = \sum_{i=1}^{r} \sum_{j=0}^{\#V[i]-1} d_{[i]}^j + f_{\min} = \sum_{i=1}^{r} (h_{[i]} - l_{[i]}) + f_{\min} = f_{\max}.$$

Aus $P^*(x) = v$ folgt also, dass x optimal ist. Trivial einzusehen ist, dass P^* aufgrund des Akzeptanzverhaltens des (1+1)-EA niemals sinken kann. Wie üblich, brauchen wir noch eine hinreichende Bedingung für eine Erhöhung des Maßes.

Wir behaupten, dass eine Mutation von maximal m ausgewählten Bits zugleich das Maß P^* um mindestens eins erhöht. Dazu betrachten wir einen beliebigen Zeitpunkt im Laufe des (1+1)-EA mit x als aktuellem Bitstring und $P^*(x) = k$, wobei $k < v$. Es muss eine Subfunktion $f_{[i]}$, $i \in R$, existieren, deren Funktionswert um mindestens v_{k+1}^* erhöht werden kann. Ließe sich keine Subfunktion um v_{k+1}^* erhöhen, gälte für alle $i \in R$ die Ungleichung $f_{[i]}(\{x_j | j \in [i]\}) > h_{[i]} - v_{k+1}^*$. Durch Summenbildung über die Äquivalenzklassen würde $f(x) \geq f_{\max} - \sum_{j=k+2}^{v} v_j^*$ folgen, da nur die Werte $v_{k+2}^*, \ldots, v_v^*$ kleiner als v_{k+1}^* sein können. (Wir erinnern uns, dass diese Werte v_i^* alle möglichen Differenzen von aufeinander folgenden Funktionswerten von Subfunktionen angeben.) Da $f_{\max} - \sum_{j=k+2}^{v} v_j^* = f_{\min} + \sum_{j=1}^{k+1} v_j^*$ gilt, hieße dies $P^*(x) = k + 1$, ein Widerspruch zu $P^*(x) = k$. Also kann der Funktionswert mindestens einer Subfunktion um mindestens v_{k+1}^* erhöht werden und damit das Fortschrittsmaß erhöht werden. Da dazu höchstens m Bits gleichzeitg flippen müssen, geschieht dies wegen einer nach unten beschränkten Wahrscheinlichkeit von $e^{-1}n^{-m}$ nach erwarteten $O(n^m)$ Schritten. Weil es $r = O(n)$ Subfunktionen mit je maximal 2^m Funktionswerten gibt, gilt $v = O(n2^m)$. Somit genügen $O(n2^m)$ Erhöhungen von P^*, sodass die erwartete Laufzeit zur Optimierung von f durch $O(n2^m n^m) = O(2^m n^{m+1})$ nach oben beschränkt ist. □

Wir haben damit für den Spezialfall linearer Funktionen (dann ist $m = 1$) die einfache obere Schranke von $O(n^2)$ gezeigt. Falls m konstant ist, gilt folgendes Korollar.

Korollar 4.16 *Sei f wie in Satz 4.15 gegeben und sei m dem Satz entsprechend definiert. Gilt $m = \Theta(1)$, optimiert der (1+1)-EA f in erwarteten $O(n^{m+1})$ Schritten.*

Bemerkung 4.9 Es gibt quadratische Funktionen, die die Voraussetzungen von Korollar 4.16 erfüllen, aber linear viele quadratische Gewichte haben, also nicht mehr unter die Definition fast linearer Funktionen fallen. Als Beispiel führen wir die Verallgemeinerung von $\textsc{Pair}_k$ an, nämlich (n sei gerade)

$$\textsc{Pair}(x) = \sum_{i=1}^{n} (-x_i) + \sum_{i=1}^{n} 3x_{2i-1}x_{2i},$$

für die wir $O(n^3)$ als obere Schranke erhalten.

Nur für fast lineare Funktionen betrachtet, ist die Schranke in der Form von Korollor 4.16 spezieller als in Lemma 4.11 formuliert, insofern als Struktureigenschaften der Funktionen, nämlich deren Separierbarkeit, und nicht nur die Anzahl quadratischer Gewichte eingehen. Bei $\textsc{Pair}_k$ beträgt die Mächtigkeit der größten Äquivalenzklasse 2, und Korollar 4.16 zeigt wegen der Schranke $O(n^3)$ für $k \geq 4$ die Unzulänglichkeit der Schranken von Satz 4.8 und von Lemma 4.11, aus denen wir nur $O(n^k)$ erhalten.
Wir haben sogar noch mehr gezeigt. Da wir im Beweis zu Satz 4.15 an keiner Stelle davon Gebrauch machen, dass f eine quadratische Fitnessfunktion ist, gilt Folgendes.

Korollar 4.17 *Sei $f : \{0,1\}^n \to \mathbb{R}$ eine beliebige Fitnessfunktion, die in folgender Form separierbar ist: Es existiert für ein $i \in \{1,\dots,n\}$ eine Partition der Menge $I := \{1,\dots,n\}$ in i paarweise disjunkte Mengen $I_1,\dots,I_i$ mit $I := \bigcup_{j=1}^{i} I_j$, und es existieren i Fitnessfunktionen $f_1,\dots f_i$ mit $f_j : \{0,1\}^{\#I_j} \to \mathbb{R}$ für $j \in \{1,\dots,i\}$, sodass $f(x_1,\dots,x_n) = \sum_{j=1}^{i} f_j(\{x_k | k \in I_j\})$ gilt. Es bezeichne $m := \max\{\#I_j | j \in \{1,\dots,i\}\}$ die maximale Mächtigkeit der Mengen I_j. Dann gilt: Der (1+1)-EA optimiert f in erwarteter Zeit $O(2^m n^{m+1})$.*

Beweis: Wir können den Beweis zu Satz 4.15 mit einer abgeänderten Bedeutung der Äquivalenzklassen bezüglich $\equiv_f$ übernehmen. Die Rolle der Äquivalenzklassen übernehmen die Mengen I_j, $j \in \{1,\dots,i\}$, d. h. zwei Indizes stehen in Relation, wenn sie in derselben Menge I_j liegen. □

Der (1+1)-EA optimiert also eine Funktionen f, deren Grad größer als 2 sein kann, noch in polynomieller Zeit, wenn sich f in Subfunktionen separieren lässt, deren Definitionsbereiche jeweils konstante Dimensionen haben.
Wir beschließen nun diesen Exkurs und ziehen aus Satz 4.15 Konsequenzen für allgemeine quadratische Funktionen. Offenbar optimiert der (1+1)-EA quadratische Funktionen für $m = O(\log^d n)$, $d \in \mathbb{N}$, noch in pseudopolynomieller Zeit. Da wir in Kapitel 5 eine Funktion konstruieren wollen, die „besonders schwierig“ ist, insofern als die erwartete Laufzeit exponentiell sein soll und diese auch mit hoher Wahrscheinlichkeit (im Gegensatz zu Quadraten linearer Funktionen) eintreten soll, wissen wir damit, dass für m, die Mächtigkeit der größten Äquivalenzklasse bezüglich $\equiv_f$, dann auf jeden Fall $m = \omega(\log^d n)$ gelten muss.

4.4 Zusammenfassung

In diesem Kapitel haben wir einige Kriterien entdeckt, an denen sich die Komplexität quadratischer Funktionen in Bezug auf den (1+1)-EA ausmachen lässt. Zum einen werden quadratische Funktionen ohne negative Gewichte auf jeden Fall in erwarteter polynomieller Zeit optimiert. Dieses Resultat lässt sich vermutlich sogar auf Funktionen konstanten Grades ohne negative Gewichte übertragen.
Im zweiten Abschnitt haben wir gesehen, dass bereits linear viele quadratische Gewichte zu erwarteten exponentiellen Laufzeiten führen können. Je nach deren Anordnung in der

Koeffizientenmatrix können Funktionen mit linear vielen quadratischen Gewichten aber auch noch einfach sein.

Im dritten Abschnitt haben wir unseren Blick auf die so genannten fast linearen Funktionen mit konstant vielen quadratischen Gewichten gerichtet. Bei diesen haben wir nachweisen können, dass sie keine exponentiellen erwarteten Laufzeiten des (1+1)-EA hervorrufen. Darüber hinaus haben wir eine Klasse fast linearer Funktionen vorgestellt, die mit einer Konstanten k parametrisiert sind, um verschiedene polynomielle erwartete Laufzeiten scharf einzustellen. (Man kann dies als kleines Hierarchieresultat ansehen.) Indem wir schließlich gezeigt haben, wie fast lineare Funktionen zu separieren sind, haben wir beweisen können, dass nicht nur die Anzahl quadratischer Gewichte, sondern auch das Ausmaß, in dem sie die Separierbarkeit der Funktion beinflussen, Auswirkungen auf die erwartete Laufzeit des (1+1)-EA hat. Diese Resultate zur Separierbarkeit gelten nicht nur für fast lineare bzw. quadratische Funktionen.

Kapitel 5

Komplexitätstheoretische Aspekte

5.1 NP-Vollständigkeit

Obschon die bis heute bekannten Resultate über die Effizienz evolutionärer Algorithmen bei weitem nicht befriedigend erscheinen, ist die Komplexität der zugrunde liegenden Aufgabe vieler evolutionärer Algorithmen, nämlich der Optimierung pseudoboolescher Funktionen $f : \{0,1\}^n \to \mathbb{R}$, seit längerem bekannt. Mit Bezug auf genetische Algorithmen zeigen Belew und Hart in [BH91], dass die Optimierung einer beliebigen pseudobooleschen Funktion ein NP-hartes Problem ist und dass darüber hinaus vermutlich keine effizienten Approximationsalgorithmen für dieses Problem existieren. Intuitiv werden uns beide Aussagen klar, wenn wir eine „Nadel-im-Heuhaufen-Funktion“ wie $\textsc{Needle}_{n,m}(x_1, \ldots, x_n) = m \prod_{i=1}^{n} x_i$ mit $m \in \mathbb{N}$ betrachten. Ein Black-Box-Optimierungsverfahren erhält bei einer Auswertung der Funktion auf einem zufälligen Bitstring mit einer Wahrscheinlichkeit von $1 - 2^{-n}$ weder einen Hinweis auf das Optimum in $\vec{1}$ noch auf den maximalen Funktionswert.

Auch die Komplexität der Optimierung pseudoboolescher Funktionen mit beschränktem Grad ist relativ leicht zu erkennen. Bei quadratischen Funktionenen, d. h. Funktionen mit einem „kleinen“ Grad 2, weiß man, dass das zur Optimierung der Funktion gehörende Entscheidungsproblem NP-vollständig bleibt. Wir formalisieren dieses Entscheidungsproblem.

Definition 5.1 (Problem Deg-2-Opt) *Gegeben seien eine in polynomieller Zeit auszuwertende quadratische Fitnessfunktion $f : \{0,1\}^n \to \mathbb{N}$ und ein $k \in \mathbb{N}$. Es ist zu entscheiden, ob ein $x \in \{0,1\}^n$ mit $f(x) \geq k$ existiert.*

Bereits in Lemma 2.4 haben wir gesehen, dass wir bei der Betrachtung des (1+1)-EA o. B. d. A. annehmen dürfen, dass quadratische Fitnessfunktionen in die ganzen Zahlen abbilden. Offensichtlich lässt sich der Wertebereich dann auch auf natürliche Zahlen einschränken, indem man einen genügend großen konstanten Term addiert. Damit wird deutlich, dass die Klasse der quadratischen Funktionen der Gestalt $f : \{0,1\}^n \to \mathbb{N}$ zumindest für den (1+1)-EA sämtliche quadratische Funktionen $f : \{0,1\}^n \to \mathbb{R}$ beinhaltet. Abgesehen davon ist es klar, dass das Problem Deg-2-Opt im Sinne der Komplexitätstheorie

nicht einfacher werden kann, wenn wir in dessen Definition statt quadratischer Funktionen $f : \{0,1\}^n \to \mathbb{N}$ sogar quadratische Funktionen $f : \{0,1\}^n \to \mathbb{R}$ erlauben.
Wir wollen die NP-Vollständigkeit von DEG-2-OPT nachweisen und benötigen dazu ein Entscheidungsproblem, das wir auf DEG-2-OPT reduzieren. Als geeignet erweist sich MAX-2-SAT.

Definition 5.2 (Problem MAX-2-SAT) *Gegeben seien m Klauseln $c_1, \ldots, c_m$ mit je genau 2 Literalen aus n Variablen $x_1, \ldots, x_n$ und eine Zahl $k \in \mathbb{N}$. Es ist zu entscheiden, ob es eine Belegung $(\hat{x}_1, \ldots, \hat{x}_n)$ der Variablen $x_1, \ldots, x_n$ gibt, die mindestens k Klauseln gleichzeitig erfüllt.*

Seit längerem ist die NP-Vollständigkeit von MAX-2-SAT bekannt [GJ76]. Damit zeigen wir die NP-Vollständigkeit von DEG-2-OPT.

Lemma 5.1 DEG-2-OPT *ist NP-vollständig.*

Beweis: Es ist DEG-2-OPT $\in$ NP: Wir können ein $x \in \{0,1\}^n$ raten. Dann kann $f(x)$ in polynomieller Zeit berechnet und $f(x) \geq k$ überprüft werden.
Seien die Klauseln $c_1, \ldots, c_m$, die Variablenmenge $x_1, \ldots, x_n$ und $k \in \mathbb{N}$ eine Eingabe für MAX-2-SAT. Zu einer Klausel c_l, $l \in \{1, \ldots, m\}$, definieren wir die „Projektionen“ $\pi_1 : \{c_1, \ldots, c_m\} \to \{1, \ldots, n\}$ und $\pi_2 : \{c_1, \ldots, c_m\} \to \{1, \ldots, n\}$. Es gilt $\pi_1(c_l) = i$ für eine Klausel c_l genau dann, wenn i der Index der Variablen des ersten Literals von c_l ist; analog ist π_2 definiert. Außerdem geben die Indikatorfunktionen $s_1 : \{c_1, \ldots, c_m\} \to \{0,1\}$ und $s_2 : \{c_1, \ldots, c_m\} \to \{0,1\}$ an, ob das erste bzw. zweite Literal positiv (Wert 1) oder negativ (Wert 0) ist. Für $c_l = (x_i \vee \bar{x}_j)$ haben wir beispielsweise $\pi_1(c_l) = i$ und $\pi_2(c_l) = j$ sowie $s_1(c_l) = 1$ und $s_2(c_l) = 0$. Damit definieren für Klauseln c_l die Funktionen

$$p_1(c_l) = \begin{cases} x_{\pi_1(c_l)}, & \text{falls } s_1(c_l) = 0 \\ 1 - x_{\pi_1(c_l)}, & \text{falls } s_1(c_l) = 1 \end{cases} \quad \text{und} \quad p_2(c_l) = \begin{cases} x_{\pi_2(c_l)}, & \text{falls } s_2(c_l) = 0. \\ 1 - x_{\pi_2(c_l)}, & \text{falls } s_2(c_l) = 1. \end{cases}$$

Schließlich konstruieren wir die quadratische (in polynomieller Zeit auszuwertende) pseudoboolesche Funktion f in der Gestalt

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{m} (1 - p_1(c_i)p_2(c_i))$$

und setzen den Parameter k der Eingabe für DEG-2-OPT mit dem Wert von k in der Eingabe für MAX-2-SAT gleich. Das ist offensichtlich in polynomieller Zeit möglich.
Wir müssen nun nachweisen, dass in der Eingabe für MAX-2-SAT genau dann mindestens k Klauseln zugleich erfüllt werden können, wenn ein $x \in \{0,1\}^n$ mit $f(x) \geq k$ existiert. Wir zeigen dazu, dass eine Belegung $(\hat{x}_1, \ldots, \hat{x}_n) \in \{0,1\}^n$ genau dann eine Klausel $c_l = (l_i \vee l_j)$ erfüllt, wenn $(1 - p_1(c_l)p_2(c_l))$ den Wert 1 annimmt. Dies folgt mit der de-Morgan'schen Regel $c_l = \overline{\bar{l}_i \wedge \bar{l}_j}$ und den Identitäten $ab = a \wedge b$ und $\bar{a} = 1 - a$ für $a, b \in \{0,1\}$ dann unmittelbar aus der Definition von p_1 und p_2. Weil jeder Ausdruck $1 - p_1(c_i)p_2(c_i)$ nur den Wert 0 oder 1 annehmen kann, folgt insgesamt die Behauptung. □

Bezüglich polynomieller Reduktion sind also alle Probleme aus NP nicht schwieriger als DEG-2-OPT. Daher ist es nicht verwunderlich, dass sich viele als NP-vollständig bekannte Probleme auf natürliche Weise als das Problem der Optimierung einer quadratischen pseudobooleschen Funktion beschreiben lassen. Hammer und Simeone stellen in [HS89] eine Reihe solcher Reduktionen vor, insbesondere aus dem Bereich von Graphproblemen. Es ist beispielsweise leicht, das Problem „Independent Set“ als Maximierungsproblem einer quadratischen Funktion umzuformulieren. Auch muss sich das Problem, eine beliebige pseudoboolesche Funktion zu maximieren, auf den quadratischen Fall reduzieren lassen.[1] Über die rein pragmatische Einführung quadratischer Funktionen in Abschnitt 2.2 hinaus – wir haben die quadratischen Funktionen als „nächste“ Klasse nach den linearen vorgestellt – gewinnen wir in diesem Licht ein wichtiges Argument für die Relevanz der Betrachtung quadratischer Funktionen.

In Abschnitt 3.3 haben wir gesehen, dass der (1+1)-EA die Quadrate beliebiger linearer Funktionen mit einer durch eine Konstante nach unten beschränkten Wahrscheinlichkeit in polynomieller Laufzeit optimiert. Nach dem obigen NP-Vollständigkeitsbeweis können wir dies nicht bei allen quadratischen Funktionen annehmen. Um dies zu erläutern, fragen wir uns, was folgen würde, wenn der (1+1)-EA unabhängig von der zu optimierenden quadratischen Funktion f immer mit einer Wahrscheinlichkeit von mindestens c, c konstant, in $t(n)$ Schritten, t ist polynomiell in n, ein Optimum fände. Mit Multistartstrategien, nämlich konstant vielen Instanzen des (1+1)-EA, ließe sich die Wahrscheinlichkeit, dass mindestens eine Instanz f in polynomieller Zeit optimierte, über 1/2 bringen. Für das Entscheidungsproblem DEG-2-OPT könnte daraus ein BPP-Algorithmus (vgl. [Weg93]) konstruiert werden, der mit einer Wahrscheinlichkeit von über 1/2 in polynomieller Zeit die richtige Antwort ausgäbe und sonst abbräche und eine beliebige Antwort gäbe. Wir hätten also einen BPP-Algorithmus für ein NP-vollständiges Problem. Dies wäre ein Widerspruch zu der i. A. für wahr gehaltenen Hypothese, dass NP $\not\subseteq$ BPP gilt (vgl. [Weg97]). Die vorigen Gedanken gelten natürlich auch dann noch, wenn c nicht konstant ist, sondern durch $\Omega(n^{-k})$, k konstant, nach unten beschränkt ist.

Wir bauen also auf der Vermutung auf, dass eine „besonders schwierige“ quadratische Funktion existieren muss. „Besonders schwierig“ bedeutet dann, dass der (1+1)-EA mit einer Wahrscheinlichkeit $1-\epsilon$, $\epsilon = o(n^{-k})$ für alle konstanten k, superpolynomielle Laufzeit benötigt. Es besteht Hoffnung, solche Eigenschaften bei einer quadratischen Funktion nachweisen zu können, da der (1+1)-EA gut zu analysieren ist und eine besonders schwierige Funktion vom Grad 3 bekannt ist (siehe [DJW99a]).

In den nächsten beiden Abschnitten stellen wir zunächst zwei Funktionen vor, die eventuell besonders schwierig sind. Dies können wir zurzeit aber nicht beweisen. In Abschnitt 5.4 untersuchen wir eine dritte Funktion; bei ihr gelingt der Nachweis, besonders schwierig zu sein. Dabei stellt sich heraus, dass die oben erwähnte Wahrscheinlichkeit ϵ sogar exponentiell klein sein kann.

[1] Dies werden wir in Abschnitt 5.4 ausnutzen.

5.2 Einige Gedanken zu „Chain“

In Bemerkung 4.6 haben wir darauf hingewiesen, dass die Funktion CHAIN zwar eine erwartete exponentielle Laufzeit des (1+1)-EA hervorruft, es aber nicht offensichtlich ist, ob diese mit einer hohen Wahrscheinlichkeit eintritt. Wir stellen im Folgenden einige Untersuchungen an, die die Vermutung erhärten, dass die Wahrscheinlichkeit exponentieller Laufzeiten auf CHAIN gering ist. Dabei führen wir zur Vereinfachung zwei Begriffe ein.

Definition 5.3 *Sei $x \in \{0,1\}^n$ ein Bitstring. Der Bitstring x enthält eine Kette der Länge l, $2 \leq l \leq n$, wenn ein Index $k \in \{1,\ldots,n-l+1\}$ existiert, sodass $\forall i \in \{k,\ldots,k+l-1\}$: $x_i = 1$ und $k \neq 1 \Rightarrow x_{k-1} = 0$ sowie $k \neq n \Rightarrow x_{k+1} = 0$ zutrifft.*

Natürlich ist für ein gegebenes $x \in \{0,1\}^n$ die Anzahl der Ketten in x eindeutig bestimmt.

Definition 5.4 *Sei $x \in \{0,1\}^n$ ein Bitstring. Bit i, $i \in \{1,\ldots,n\}$, heißt isoliert, wenn $x_i = 1$ und $i \neq 1 \Rightarrow x_{i-1} = 0$ sowie $i \neq n \Rightarrow x_{i+1} = 0$ zutrifft.*

Damit können wir eine interessante Aussage über den Funktionswert von CHAIN festhalten.

Lemma 5.2 *Sei $x \in \{0,1\}^n$ ein Bitstring. Es sei außerdem k die Zahl der Ketten in x, l die Summe der Längen aller Ketten und i die Zahl isolierter Bits in x. Dann gilt*

$$\textsc{Chain}(x) \geq l - 2nk + i(1-2n).$$

Beweis: Aus der Definition von CHAIN folgt, dass eine Kette der Länge l' genau l' lineare Gewichte und mindestens $l'-1$ quadratische Gewichte aktiviert. Der Beitrag einer solchen Kette zum Funktionswert ist also $l'(1-2n)+(l'-1)2n = -2n+l'$. Über alle Ketten summiert folgt, dass alle Ketten einen Summanden von mindestens $l-2nk$ zum Funktionswert beitragen. Die isolierten Bits hingegen tragen $i(1-2n)$ zum Funktionswert bei. Da wir bei dieser Argumentation alle aktivierten negativen Gewichte berücksichtigen, folgt die untere Schranke. Wenn $x_1 = 0$ oder $x_n = 0$ gilt, tritt sogar Gleichheit ein, denn wir haben nur das quadratische Gewicht w_{1n} unberücksichtigt gelassen. □

Anhand von Lemma 5.2 wollen wir die Beschaffenheit von akzeptierten Mutationen des (1+1)-EA, die aus einem $x \in \{0,1\}^n$ ein $x' \in \{0,1\}^n$ mit $\textsc{Chain}(x') \geq \textsc{Chain}(x)$ erzeugen, analysieren. Zur Vereinfachung unterstellen wir, dass Lemma 5.2 statt einer unteren Schranke für den Funktionswert von CHAIN immer den tatsächlichen Funktionswert angibt.

Welche Mutationen, die nur ein Bit flippen, können akzeptiert werden? Erstens sind dies natürlich Mutationen, die isolierte Bits zu 0 flippen. Zum anderen werden aber auch 1-Bit-Mutationen akzeptiert, die die Gesamtlängen der Ketten erhöhen und damit die Zahl der Einsen erhöhen.

Eine Mutation mehrerer Bits zugleich kann akzeptiert werden, wenn sie mindestens ein isoliertes Bit flippt. Dabei kann sich die Gesamtlänge aller Ketten oder die Anzahl der Ketten verringern.

Aufschlussreich ist die Situation, dass der aktuelle Bitstring x keine isolierten Bits enthält. Wir betrachten eine akzeptierte Mutation zu einem $x' \in \{0,1\}^n$ anhand von Lemma 5.2. Diese kann einerseits die Gesamtlänge l aller Ketten erhöhen. Weil andererseits der Funktionswert von CHAIN nicht sinken darf, kann sich l, die Gesamtlänge aller Ketten, notwendigerweise nur dann verringern, wenn k sinkt. Dazu muss entweder mindestens eine Kette in einem Schritt durch Mutation aller Bits der Kette „verschwinden“ oder es muss sich die Anzahl der Ketten verringern, indem zwei Ketten „verschmelzen“, also Nullbits zwischen Ketten zu 1 flippen.

Isolierte Bits können bei einer akzeptierten Mutation nicht allein durch Erhöhung der Gesamtlänge aller Ketten entstehen, weil $l \leq n$ ist, ein isoliertes Bit aber $1 - 2n$ zum Funktionswert von CHAIN beiträgt. Es ist also zudem eine Verringerung der Gesamtzahl der Ketten notwendig, d. h. wie zuvor durch „Verschwinden“ oder „Verschmelzen“ von Ketten.

Die Wahrscheinlichkeit, dass isolierte Bits nach der Initialisierung schnell verschwinden, scheint recht hoch zu sein, weil dazu 1-Bit-Mutationen ausreichen. Demgegenüber scheint die Wahrscheinlichkeit für die Entstehung isolierter Bits gering zu sein, weil während einer solchen Mutation Ketten verschwinden oder verschmelzen müssen. Wir vermuten, dass der (1+1)-EA auf CHAIN eine deutliche Tendenz, die Gesamtlänge der Ketten zu erhöhen, besitzt. Deshalb stellen wir die Behauptung auf, dass die Funktion CHAIN mit deutlich über 1/2 liegender Wahrscheinlichkeit in polynomieller Zeit (evtl. $O(n^2)$) optimiert wird. Weil die Zahl der Einsen im aktuellen Bitstring sinken oder steigen kann, würde eine nähere Analyse vermutlich in einem Modell eines Markoffprozesses, der eine Irrfahrt (*random walk*) auf dem Zustandsraum vollführt, resultieren. Dies mutet sehr komplex an.

Interessant ist auch die folgende Funktion, die wir mit einer geringfügigen Modifikation aus CHAIN erhalten.

Definition 5.5 *Die Funktion* MISSINGLINK : $\{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\text{MISSINGLINK}(x) := \sum_{i=1}^{n}(1-2n)x_i + \sum_{i=1}^{n-1} 2nx_ix_{i+1}.$$

Wir haben also aus CHAIN das Monom $2nx_nx_1$ entfernt. Damit hat MISSINGLINK ein lokales Maximum in $\vec{1}$ mit dem Funktionswert $-n$; der Hammingabstand zum globalen Maximum $\vec{0}$ beträgt n. Die Abschätzung für den Funktionswert von CHAIN aus Lemma 5.2 wird für MISSINGLINK zu einer echten Gleichheit (vgl. Beweis des Lemmas), und die obigen Überlegungen bezüglich der Wahrscheinlichkeit, dass der (1+1)-EA zu $\vec{1}$ gelangt, behalten ihre Gültigkeit.

Die Funktion MISSINGLINK ist ein Kandidat für eine besonders schwierige quadratische Funktion, die nur mit kleiner Wahrscheinlichkeit in polynomieller Zeit optimiert wird. Wir formulieren es als offenes Problem zu zeigen oder zu widerlegen, dass der (1+1)-EA nur mit exponentiell kleiner Wahrscheinlichkeit in polynomiell vielen Schritten das Optimum von MISSINGLINK findet.

5.3 Einige Gedanken zu „MultiDist“

Nachdem wir uns in dieser Arbeit wiederholt mit der Funktion DISTANCE bzw. dem dualen Pendant $\text{ONESQR}_{-n/2+1/3}$ beschäftigt haben, erscheint es natürlich, darauf aufbauend eine besonders schwierige quadratische Funktion zu konstruieren zu versuchen.

Definition 5.6 *Sei $n = k^2$ für ein $k \in \mathbb{N}$. Die Funktion* $\text{MULTIDIST} : \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\text{MULTIDIST}(x) = \sum_{i=0}^{k-1} \left(\sum_{j=1}^{k} x_{ik+j} - \frac{k}{2} + \frac{1}{3} \right)^2 .$$

Diese Funktion entsteht durch Addition von $k = \sqrt{n}$ Funktionen mit paarweise disjunkten Variablenmengen, wobei jede Funktion eine Instanz der Funktion ONESQR, und zwar $\text{ONESQR}_{-k/2+1/3}$, ist. Wir wissen, dass der (1+1)-EA auf $\text{ONESQR}_{-n/2+1/3}$ mit einer Wahrscheinlichkeit von $1/2 - \epsilon$ eine exponentielle Laufzeit benötigt (vgl. die Ausführungen zu DISTANCE in [DJW98a]). Daraus leitet sich die Vermutung ab, dass für MULTIDIST mit einer Wahrscheinlichkeit von $1 - 2^{-\Omega(k)}$ eine exponentielle Laufzeit erforderlich ist. Wir geben einen ersten Hinweis darauf.

Lemma 5.3 *Sei $x^s \in \{0,1\}^n$ der initialisierte Bitstring des (1+1)-EA. Mit einer Wahrscheinlichkeit von mindestens $1 - (3/4)^k$ existiert ein $i \in \{0, \ldots, k-1\}$, sodass die Ungleichung $\sum_{j=1}^{k} x^s_{ik+j} \leq k/2 - \sqrt{k}/4$ gilt.*

Beweis: Die Wahrscheinlichkeit, dass ein Block von unabhängig gleichverteilt initialisierten k Bits höchstens $k/2 - \sqrt{k}/4$ Einsbits enthält, beträgt mindestens $1/2 - (\sqrt{k}/4)/\sqrt{k}) = 1/4$ (siehe Beweis zu Lemma 3.19). Damit ist die Wahrscheinlichkeit, dass keiner der k unabhängig initialisierten Blöcke der Länge k höchstens $k/2 - \sqrt{k}/4$ Einsbits enthält, durch $(3/4)^k$ nach oben beschränkt. Durch Übergang zur Gegenwahrscheinlichkeit folgt die Behauptung. □

Mit exponentiell nahe an 1 liegender Wahrscheinlichkeit wird also o. B. d. A. der Block $x_1, \ldots, x_k$ so initialisiert, dass er höchstens $k/2 - \sqrt{k}/4$ Einsen enthält (Ereignis I). Wir betrachten die auf $x_1, \ldots, x_k$ definierte Funktion

$$f(x_1, \ldots, x_k) = \left(\sum_{i=1}^{k} x_i - \frac{k}{2} + \frac{1}{3} \right)^2 ,$$

die ein Summand von MULTIDIST ist. Ein bekanntes Resultat für DISTANCE besagt, dass bei Eintreten von I die Wahrscheinlichkeit für polynomielle Laufzeiten auf f exponentiell klein in k ist, vgl. [DJW98a]. Dies begründet sich daraus, dass der (1+1)-EA mindestens $\sqrt{k}/2$ Bits gleichzeitig flippen muss, um zum globalen Maximum in $\vec{1}$ gelangen zu können. Auch wenn bei MULTIDIST das Ereignis I mit exponentiell nahe an 1 liegender Wahrscheinlichkeit eintritt, können wir die Laufzeit mit der Argumentation für DISTANCE nicht

durch $2^{\Omega(k)} = 2^{\Omega(\sqrt{n})}$ nach unten beschränken. Bereits eine Mutation von 2 Bits – nämlich eine Mutation, die eines der Nullbits unter $x_1, \ldots, x_k$ zu 1 flippt und in einem anderen Block ein weiteres Bit flippt – kann unter Umständen akzeptiert werden und damit die Zahl der Einsen im betrachteten Block erhöhen. Auf der anderen Seite wird eine 1-Bit-Mutation, die im Block $x_1, \ldots, x_k$ ein Einsbit flippt, bei Eintreten des Ereignisses I immer akzeptiert.

Wie schon in Abschnitt 5.2 scheint eine nähere Untersuchung von MULTIDIST in der Analyse einer Irrfahrt zu münden. Wir könnten versuchen, eine Zufallsvariable X, die die Zahl der Einsen im Block $x_1, \ldots, x_k$ angibt, aufzustellen und unter Voraussetzung von I die Wahrscheinlichkeit, dass X während polynomiell vielen Schritten auf k steigt (notwendige Bedingung für die Optimierung), abzuschätzen. Dabei scheinen aber unsere Methoden zu versagen. Eine Anwendung der Chernoffschranke scheidet zunächst wegen des Wertebereiches von X aus; zum anderen liegt keine Zerlegung von X in Zufallsvariablen mit dem Wertebereich $\{0, 1\}$ nahe. Da die Varianz von X gewiss nicht exponentiell klein ist, scheint auch eine Abschätzung mithilfe der Tschebyscheff-Ungleichung nicht zum Erfolg zu führen.

Wiederum belassen wir ein offenes Problem. Während es leicht zu sehen ist, dass die erwartete Laufzeit des (1+1)-EA auf MULTIDIST exponentiell ist (Beweis analog zu Lemma 4.10), bleibt zu untersuchen, mit welcher Wahrscheinlichkeit exponentielle Laufzeiten eintreten.

5.4 Die Funktion „QTrap“

Rosenberg gibt in [Ros75] eine Reduktion an, die das Problem, eine beliebige pseudoboolesche Funktion (ohne Restriktionen) zu maximieren, in das Problem, eine quadratische pseudoboolesche Funktion zu maximieren, transformiert. Um diese Arbeit abgeschlossen zu gestalten, geben wir in den folgenden Definitionen und Lemmata zunächst Rosenbergs Konstruktion wieder.

Definition 5.7 (Polynomdarstellung) *Sei $f : \{0,1\}^n \to \mathbb{R}$ gegeben. Die Polynomdarstellung von f wird durch diejenige (eindeutige) Wahl der Menge $I \subseteq \mathfrak{P}(\{1, \ldots, n\})$, einer Menge von Teilmengen von $\{1, \ldots, n\}$, und der Menge $c_f(A)$, $A \in I$, einer Menge von Koeffizienten mit $c_f(A) \in \mathbb{R} \setminus \{0\}$ für alle $A \in I$, induziert, die die Identität*

$$f(x_1, \ldots, x_n) = \sum_{A \in I} c_f(A) \prod_{i \in A} x_i$$

(dabei ist $\prod_{i \in \emptyset} x_i := 1$) für alle $x \in \{0,1\}^n$ erfüllt.

Definition 5.8 (Rosenbergs Methode) *Es sei $f(x) = \sum_{A \in I} c_f(A) \prod_{i \in A} x_i$ die Polynomdarstellung einer Funktion $f : \{0,1\}^n \to \mathbb{R}$. Weiter sei $f^* := \sum_{\{A \in I | c_f(A) > 0\}} c_f(A)$ die Summe aller positiven Koeffizienten von f und $q \in \mathbb{R}$ eine Zahl mit der Eigenschaft $q < \max\{f(x) | x \in \{0,1\}^n\}$. Es sei ein $F \in \mathbb{R}$, das $F \leq q - f^*$ erfüllt, gewählt. Durch*

Wahl eines Paares (i,j), $1 \leq i < j \leq n$, *konstruiert man aus einer Menge* $A \subseteq \{1,\ldots,n\}$ *die Menge* $A' \subseteq \{1,\ldots,n+1\}$ *gemäß*

$$A' = \begin{cases} A \setminus \{i,j\} \cup \{n+1\}, & \textit{falls } \{i,j\} \subseteq A, \\ A & \textit{sonst.} \end{cases}$$

Damit definiert man die Menge $I' := \{A' | A \in I\}$ *durch Anwendung dieser Konstruktion auf alle* $A \in I$ *und setzt für alle* $A' \in I'$ *den Koeffizienten* $c_f(A')$ *mit dem ursprünglichen Koeffizienten* $c_f(A)$ *gleich. Schließlich entsteht die Funktion* $f' : \{0,1\}^{n+1} \to \mathbb{R}$ *aus* f *als*

$$f'(x_1,\ldots,x_{n+1}) := F(x_ix_j + (3-2x_i-2x_j)x_{n+1}) + \sum_{A' \in I'} c_f(A') \prod_{i \in A'} x_i.$$

Mit anderen Worten entsteht f' aus f durch Ersetzung des Produktes x_ix_j durch x_{n+1} in allen Monomen und anschließender Addition eines Strafterms $F(x_ix_j+(3-2x_i-2x_j)x_{n+1})$ zu dem auf diese Weise gewonnenen Polynom. Den Sinn dieser Konstruktion entnehmen wir dem folgenden Lemma.

Lemma 5.4 *Sei* $f' : \{0,1\}^{n+1} \to \mathbb{R}$ *aus* $f : \{0,1\}^n \to \mathbb{R}$ *durch Anwendung von Rosenbergs Methode entstanden. Es bezeichnen* M_f *und* $M_{f'}$ *die Menge aller für* f *bzw.* f' *optimalen Bitstrings. Dann gelten die drei folgenden Aussagen:*

1. $f'(x_1,\ldots,x_n,x_ix_j) = f(x_1,\ldots,x_n)$
2. $\max\{f'(x) | x \in \{0,1\}^{n+1}\} = \max\{f(x) | x \in \{0,1\}^n\}$
3. $(x_1,\ldots,x_{n+1}) \in M_{f'} \Leftrightarrow (x_1,\ldots,x_n) \in M_f$ *und* $x_{n+1} = x_ix_j$

Beweis: Wir nutzen die Abkürzung $h(x,y,z) := xy + (3-2x-2y)z$, $x,y,z \in \{0,1\}$. Durch Überprüfung aller 2^3 Belegungen für x,y,z sehen wir, dass $h(x,y,z) = 0$ ist, wenn $z = xy$ gilt, dass $h(0,0,1) = 3$ ist und dass sonst $h(x,y,z) = 1$ gilt. Für $z \neq xy$ heißt dies $h(x,y,z) \geq 1$. Die erste Aussage des Lemmas folgt aus der Konstruktion von f' gemäß

$$\begin{aligned} f'(x_1,\ldots,x_n,x_ix_j) &= Fh(x_i,x_j,x_ix_j) + \sum_{A' \in I'} c_f(A') \prod_{i \in A'} x_i \\ &= \sum_{A \in I} c_f(A) \prod_{i \in A} x_i = f(x_1,\ldots,x_n), \end{aligned}$$

denn $h(x_i,x_j,x_ix_j) = 0$. Insbesondere nimmt f' im Falle $x_{n+1} = x_ix_j$ keinen größeren Wert als den maximalen Wert von f an. Andererseits nimmt f' auf Bitstrings, deren erste n Komponenten einen optimalen Bitstring für f darstellen, dann auch diesen maximalen Wert an. Wenn wir nachweisen, dass f' im Falle $x_{n+1} \neq x_ix_j$ nur Werte, die geringer als der maximale Funktionswert von f sind, annehmen kann, folgt zunächst die zweite Aussage

und damit unmittelbar die dritte Aussage. Also sei $(x_1, \ldots, x_n) \in \{0,1\}^n$ beliebig und $x_{n+1} \neq x_i x_j$. Dies impliziert zunächst $h(x_i, x_j, x_{n+1}) \geq 1$ und dann wegen $F < 0$

$$\begin{aligned} f'(x_1, \ldots, x_{n+1}) &= Fh(x_i, x_j, x_{n+1}) + \sum_{A' \in I'} c_f(A') \prod_{i \in A'} x_i \\ &\leq F + f^* \leq q < \max\{f(x) | x \in \{0,1\}^n\}. \end{aligned}$$

□

Weil es $\binom{n}{2}$ Variablenpaare (x_i, x_j), $1 \leq i < j \leq n$, gibt, genügen offensichtlich höchstens $\binom{n}{2}$ iterierte Anwendungen von Rosenbergs Methode, um eine beliebige pseudoboolesche Funktion in eine quadratische Funktion zu transformieren. Wir erhalten dann unmittelbar den folgenden Satz.

Satz 5.5 *Sei $f : \{0,1\}^n \to \mathbb{R}$ eine beliebige pseudoboolesche Funktion. Dann existieren ein $l \in \mathbb{N}$ mit $0 \leq l \leq \binom{n}{2}$ und eine Funktion $f : \{0,1\}^{n+l} \to \mathbb{R}$ mit einem Grad von höchstens 2, sodass gilt:*

1. *Ist $(x_1, \ldots, x_{n+l}) \in \{0,1\}^{n+l}$ für f' optimal, so ist $(x_1, \ldots, x_n)$ für f optimal.*
2. *Die optimalen Funktionswerte von f und f' stimmen überein.*
3. *Ist $(x_1, \ldots, x_n) \in \{0,1\}^n$ für f optimal, gibt es eindeutig bestimmte $x_{n+1}, \ldots, x_{n+l} \in \{0,1\}$, sodass $(x_1, \ldots, x_{n+l}) \in \{0,1\}^{n+l}$ für f' optimal ist.*

Rosenberg reduziert mit seiner Transformation das Entscheidungsproblem zur Maximierung beliebiger pseudoboolescher Funktionen (dies ist abgesehen von der Gradbeschränkung wie DEG-2-OPT zu formalisieren) auf das Problem DEG-2-OPT. Am Rande sei noch erwähnt, dass die Reduktion gewisse Wahlfreiheiten bei der Elimination von Variablen lässt, denn in Definition 5.8 ist (i, j) praktisch beliebig. Damit können aus einer festen Funktion f verschiedene quadratische Funktionen f' entstehen, und die Zahl der einzuführenden Hilfsvariablen l kann ebenfalls variieren. Dies spielt für uns aber keine weitere Rolle.
Bevor wir Rosenbergs Reduktion nutzen, nehmen wir daran eine kleine Modifikation vor. Als etwas störend erweisen sich bei einer iterierten Anwendung seiner Methode die neu entstehenden Monome $-2Fx_ix_{n+1}$ und $-2Fx_jx_{n+1}$, die wir durch Ausmultiplizieren von $F(x_ix_j + (3 - 2x_i - 2x_j)x_{n+1})$ erhalten. Da F negativ ist, haben die neuen Monome positive Koeffizienten, die den Wert von f^* und damit auch von F bei einer weiteren Anwendung der Methode beeinflussen können. Wir wollen zeigen, dass eine erneute Festlegung von F bei iterierter Anwendung der Methode nicht wirklich nötig ist. Dazu müssen wir fordern, dass Rosenbergs Methode sinnvoll angewandt wird und Variablenpaare (x_i, x_{n+1}), die bereits in einem Strafterm enthalten sind und wie oben ein Monom $-2Fx_ix_{n+1}$ erzeugen, nicht erneut durch eine Hilfsvariable ersetzt werden. Dies ist vernünftig, da das Produkt x_ix_{n+1} aufgrund von Rosenbergs Methode nur in (ausmultiplizierten) Straftermen, nicht aber in den modifizierten Monomen der ursprünglichen Funktion auftreten kann. Im folgenden Lemma verstehen wir unter einer *sinnvollen* Anwendung der Methode den Verzicht auf derartige Wahlen von zu ersetzenden Variablenpaaren.

Lemma 5.6 *Sei $f : \{0,1\}^n \to \mathbb{R}$ eine beliebige pseudoboolesche Funktion und habe der Parameter F einen gültigen Wert für die Anwendung von Rosenbergs Methode (Def. 5.8) auf f. Lemma 5.4 und Satz 5.5 behalten ihre Gültigkeit, wenn $f' : \{0,1\}^{n+l} \to \mathbb{R}$ durch l-fache sinnvolle Anwendung der Methode konstruiert wird und bei allen Anwendungen der Wert von F unverändert bleibt.*

Beweis: Der Fall $l = 1$ ist trivial. Für den Induktionsschluss von $l-1$ auf l betrachten wir die durch $l-1$ Anwendungen von Rosenbergs Methode entstandene Funktion f_{l-1} und die dann in der l-ten Anwendung konstruierte Funktion f_l. Wir zeigen, dass der bei der ersten Anwendung gewählte Wert von q dann auch $q < \max\{f_l(x)|x \in \{0,1\}^{n+l}\}$ erfüllt, dass $\max\{f_{l-1}(x)|x \in \{0,1\}^{n+l-1}\} = \max\{f_l(x)|x \in \{0,1\}^{n+l}\}$ gilt und dass Lemma 5.4 bei der l-ten Anwendung der Methode insgesamt gültig bleibt.
Für f_{l-1} gilt Lemma 5.4 nach Induktionsvoraussetzung; insbesondere stimmt der maximale Funktionswert von f_{l-1} mit dem von f überein, und q erfüllt die Voraussetzungen der Methode. Beim Beweis des Lemmas folgt sofort, dass die erste Aussage unabhängig von F gültig bleibt. Insbesondere kann f_l den maximalen Funktionswert von f_{l-1} annehmen.
Der Wert von F ist nur für die zweite und dritte Aussage des Lemmas relevant. Sogar ohne Induktion sehen wir, dass f_{l-1} niemals größer als f^*, die Summe der positiven Koeffizienten von f, werden kann, da die $l-1$ bereits entstandenen Strafterme niemals positive Werte annehmen können. Auch die Ersetzung eines Variablenpaares (x_i, x_j), $1 \le i < j \le n+l-1$, durch die Hilfsvariable x_{n+l} bewahrt die obere Schranke, da in Straftermen keine Ersetzungen vorgenommen werden.
Im l-ten Schritt wird der Strafterm $Fh(x_i, x_j, x_{n+1})$ addiert. Damit gilt wieder im Falle $x_{n+l} \neq x_i x_j$ die Ungleichung $f_l(x) \le F + f^* \le q$. Weil q geringer als der optimale Funktionswert von f_{l-1} ist, nimmt f_l dann also keinen optimalen Funktionswert an. Somit ist die zweite und dritte Aussage des Lemmas auch für f_l bewiesen, und $q < \max\{f_l(x)|x \in \{0,1\}^{n+l}\}$ folgt aus $\max\{f_l(x)|x \in \{0,1\}^{n+l}\} = \max\{f(x)|x \in \{0,1\}^n\}$. □

Beispiel 5.1 Um zu zeigen, dass Lemma 5.6 nicht mehr gilt, wenn in Rosenbergs Methode Variablenpaare „sinnlos“ ersetzt werden, nehmen wir uns die Funktion $f(x_1, x_2, x_3) = x_1 x_2 x_3$ vor. Hier ist $f^* = 1$ und $\max\{f(x)|x \in \{0,1\}^3\} = 1$. Wir wählen $q = 0$ und $F = -2$. Durch Anwendung von Rosenbergs Methode auf das Variablenpaar (x_1, x_2) erhalten wir

$$\begin{aligned} f'(x_1, \ldots, x_4) &= x_3 x_4 - 2(x_1 x_2 + (3 - 2x_1 - 2x_2)x_4) \\ &= x_3 x_4 - 2x_1 x_2 - 6x_4 + 4x_1 x_4 + 4x_2 x_4. \end{aligned}$$

Eine weitere Anwendung der Methode, diesmal auf sinnlose Weise mit dem Variablenpaar (x_2, x_4), liefert

$$f''(x_1, \ldots, x_5) = x_3 x_4 - 2x_1 x_2 - 6x_4 + 4x_1 x_4 + 4x_5 - 2(x_2 x_4 + (3 - 2x_2 - 2x_4)x_5).$$

Dann ist $f''(0,1,0,0,1) = 4 - 2(0 + (3-2)\cdot 1) = 2$, d. h. der maximale Funktionswert von f'' ist größer als der von f. Lemma 5.6 gilt bei sinnlosen Anwendungen der Methode also nicht mehr.

Nun wollen wir Rosenbergs Methode endlich einsetzen. Sie ermöglicht es nämlich, beliebige pseudoboolesche Funktionen, die für den (1+1)-EA als besonders schwierig bekannt sind, in quadratische Funktionen umzuwandeln. Für eine solche Transformation wählen wir die Funktion TRAP.

Definition 5.9 *Die Funktion* $\text{TRAP}_n : \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\text{TRAP}_n(x) = \sum_{i=1}^{n}(-x_i) + (n+1)\prod_{i=1}^{n} x_i.$$

Es ist in den folgenden Ausführungen zuweilen erforderlich, die Länge der Bitstrings anzugeben, für die TRAP_n definiert ist. Geht dies aus dem Zusammenhang hervor, lassen wir den Index n meist weg.

Definition 5.9 weicht von der Definition von TRAP aus [DJW98a] ab, denn das globale Maximum „unserer“ TRAP-Funktion liegt in $\vec{1}$, wobei $\text{TRAP}(\vec{1}) = 1$. Damit wollen wir Konformität mit unserer Konvention, das globale Optimum in $\vec{1}$ zu legen, herstellen und Verwirrungen vermeiden. Nichtsdestoweniger gelten die von Droste, Jansen und Wegener vorgestellten Resultate auch für die TRAP-Funktion aus Definition 5.9, da nur eine Komplementbildung vorgenommen und der Wertebereich von $\{1,\ldots,n+1\}$ auf $\{-n+1,-n+2,\ldots,1\}$ geändert wurde.

Nun wenden wir Rosenbergs Reduktion auf die Funktion TRAP_m, $m := n/2+1$, an. Indem wir den dazu benötigten Parameter F in Einklang mit der Reduktion wählen als $F := -(m+1)$, bilden wir die Funktion QTRAP („quadratic trap“).

Definition 5.10 *Sei* $n \in \mathbb{N}$ *gerade und* $m := n/2+1$. *Die Funktion* $\text{QTRAP} : \{0,1\}^n \to \mathbb{R}$ *wird definiert durch*

$$\begin{aligned}\text{QTRAP}(x) = &\sum_{i=1}^{m}(-x_i) + (m+1)x_1x_n \\ &+ \sum_{i=1}^{m-2}(-(m+1)(x_{m-i}x_{m+i-1} + (3 - 2x_{m-i} - 2x_{m+i-1})x_{m+i})).\end{aligned}$$

Es stellt keine echte Einschränkung dar, dass diese Definition nur für gerade n gilt. Wir weisen noch darauf hin, dass die Definition wegen $m + (m-2) = n$ sinnvoll ist, d. h. dass QTRAP von genau n Variablen abhängt.

Nun machen wir uns den Zusammenhang zu Rosenbergs Reduktion klar.

Lemma 5.7 *Die Funktion* $\text{QTRAP} : \{0,1\}^n \to \mathbb{R}$ *entsteht durch sukzessive Anwendung von Rosenbergs Methode (Def.* 5.8*) aus der Funktion* TRAP_m, *wobei* $m := n/2+1$.

Beweis: Gemäß den Notationen in Definition 5.8 ist $f^* = m+1$. Wegen des optimalen Funktionswertes $\text{TRAP}_m(\vec{1}) = 1$ wählen wir $q := 0$. Der Wert $F := -(m+1)$ erfüllt die

Voraussetzung $F \leq q - f^*$. Auf die Funktion $\textsc{Trap}_m(x) = \sum_{i=1}^{m} -x_i + (m+1)\prod_{i=1}^{m} x_i$ wenden wir sukzessive Rosenbergs Methode an, indem wir den Grad des positiven Monoms durch Elimination der Variablen mit dem größten und zweitgrößten Index und anschließender Ergänzung einer Hilfsvariablen um eins verringern. Die erste Elimination führt zu der Funktion

$$\sum_{i=1}^{m}(-x_i) + (m+1)\left(\prod_{i=1}^{m-2} x_i\right) x_{m+1} - (m+1)(x_{m-1}x_m + (3 - 2x_{m-1} - 2x_m)x_{m+1}).$$

Laut Lemma 5.6 dürfen wir Rosenbergs Methode mit $F := -(m+1)$ iterieren. Weil $m-2$ Schritte ausreichen, um den Grad des positiven Monoms auf 2 zu verringern, folgt dann induktiv die Gestalt aus Definition 5.10. □

Mit Satz 5.5 erhalten wir dann, dass der optimale Funktionswert von QTrap wie bei $\textsc{Trap}_m$ zum einen 1 beträgt und dass zum anderen nur der n-stellige Einsvektor $\vec{1}$ diesen Funktionswert liefert. Im folgenden Lemma zeigen wir, dass die Funktion QTrap zu $\textsc{Trap}_m$ noch weitere Ähnlichkeiten aufweist.

Lemma 5.8 *Gegeben sei ein beliebiges $x \in \{0,1\}^n$. Gilt für $i := \sum_{j=1}^{m} x_j$, dass $i < m$, existiert ein $k \in \{0, \ldots, 3(m-2)\}$, sodass gilt:* $\textsc{QTrap}(x) = -i - k(m+1)$.

Beweis: Lemma 5.4 sagt aus, dass die Ausdrücke $x_{m-i}x_{m+i-1} + (3 - 2x_{m-i} - 2x_{m+i-1})x_{m+i}$ lediglich Werte aus $\{0,1,3\}$ annehmen können. Nach einer Multiplikation mit $-(m+1)$ folgt für die Strafterme in QTrap, dass sie nur die Werte $0, -(m+1)$ oder $-3(m+1)$ annehmen. Weil das Monom $(m+1)x_1x_n$ den Koeffizienten $m+1$ besitzt, die linearen Monome -1 als Koeffizienten haben und es $m-2$ Strafterme gibt, folgt zunächst $\textsc{QTrap}(x) = -i-k(m+1)$ für ein $k \in \{-1, \ldots, 3(m-2)\}$. Zu zeigen bleibt $k \geq 0$. Da $(x_1, \ldots, x_m)$ für $\textsc{Trap}_m$ wegen $i < m$ nicht optimal ist, folgt aus Satz 5.5 die Eigenschaft, dass x auch für QTrap nicht optimal sein kann. Wir erhalten somit die Ungleichung $\textsc{QTrap}(x) < 1$. Dies schließt negative Werte von k aus, denn $i < m+1$. □

Damit haben wir bereits alle Aussagen zusammen, um zu beweisen, dass die quadratische Funktion QTrap für den (1+1)-EA besonders schwierig ist.

Satz 5.9 *Der (1+1)-EA benötigt zur Optimierung der Funktion* QTrap *mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-\Omega(n)}$ mindestens $2^{\Omega(n \log n)}$ Schritte.*

Beweis: Wir unterteilen Bitstrings $x \in \{0,1\}^n$ in diesem Beweis mithilfe von $m := n/2+1$ in einen „vorderen“ Block $(x_1, \ldots, x_m)$ und einen „hinteren“ Block $(x_{m+1}, \ldots, x_n)$, d. h. den Block der Variablen der ursprünglichen Trap-Funktion (vgl. Lemma 5.7) und den Block der zusätzlichen, durch die Reduktion entstandenen Variablen. Wir werden zeigen, dass sich die Zahl der Einsen im vorderen Block nach der Initialisierung mit hoher Wahrscheinlichkeit nicht auf m erhöht, bevor ein Zustand erreicht ist, in dem sich die Zahl der Einsen nicht

mehr erhöhen kann, es sei denn, dass mindestens alle verbleibenden Nullbits im vorderen Block gleichzeitig zu 1 flippen.

Wir treffen unter Anwendung der Chernoffschranke zwei Annahmen über den initialisierten Bitstring x^s. Für den vorderen Block nutzen wir die Zufallsvariablen X_i, $i \in \{1, \ldots, m\}$, mit $X_i := 1 - x_i^s$. Dies sind natürlich unabhängige Zufallsvariablen mit dem Wertebereich $\{0, 1\}$, sodass auf $X := X_1 + \cdots + X_m$ mit $E[X] = \sum_{i=1}^{m} \text{Prob}(X_i = 1) = m/2$ die Chernoffschranke angewendet werden darf. Es ist

$$\text{Prob}(X < 0{,}4m) = \text{Prob}(X < (1 - 0{,}05)E[X]) < e^{-(0{,}05)^2(n/2+1)/4} \text{ (Lemma A.2)},$$

d. h. mit exponentiell nahe an 1 liegender Wahrscheinlichkeit enthält der initialisierte Bitstring mindestens $0{,}4m$ Nullen im vorderen Block.

Für den hinteren Block ziehen wir weitere $m-2$ 0-1-Zufallsvariablen Y_i, $i \in \{1, \ldots, m-2\}$, heran. Es ist $Y_i = 1$, falls $x^s_{m+i} = x^s_{m-i}x^s_{m+i-1}$ gilt, und 0 sonst. Für Aussagen über diese Zufallsvariablen wenden wir das *Prinzip der verzögerten Entscheidungen* (vgl. [MR95, Abschnitt 3.5]) an. Dies heißt, dass wir für $i \in \{2, \ldots, n\}$ davon ausgehen, dass die Bits $x^s_1, \ldots, x^s_{i-1}$ des initialisierten Bitstrings bereits einen festen Wert haben, wenn x^s_i „ausgewürfelt“ wird, d. h. mit Wahrscheinlichkeit 1/2 mit 1 belegt wird. Diese Annahme ist zulässig, da verschiedene Bits von x^s unabhängig initialisiert werden. So gesehen nimmt x^s_{m+i} mit Wahrscheinlichkeit 1/2 den *festen* Wert $x^s_{m-i}x^s_{m+i-1}$ an, d. h. $\text{Prob}(Y_i = 1) = 1/2$. Die Zufallsvariable Y_i hängt also nur von der Position x^s_{m+i} ab. Folglich sind auch alle Y_i unabhängig voneinander. Wir wenden auf $Y := Y_1 + \cdots + Y_{m-2}$ mit $E[Y] = (m-2)/2$ wie oben die Chernoffschranke an und erhalten

$$\text{Prob}(X < 0{,}4(m-2)) < e^{-(0{,}05)^2(m-2)/4} = e^{-(0{,}05)^2(n/2-1)/4},$$

d. h. mit exponentiell nahe an 1 liegender Wahrscheinlichkeit nehmen mindestens $0{,}4(m-2)$ Zufallsvariablen Y_i den Wert 1 an. Das Ereignis $Y_i = 1$ ist gleichbedeutend damit, dass der zu Y_i gehörende Strafterm

$$h(x_{m-i}, x_{m+i-1}, x_{m+i}) := -(m+1)(x_{m-i}x_{m+i-1} + (3 - 2x_{m-i} - 2x_{m+i-1})x_{m+i})$$

den Wert 0 annimmt (siehe dazu Lemma 5.4). Mit exponentiell nahe an 1 liegender Wahrscheinlichkeit nehmen also mindestens $0{,}4(m-2)$ Strafterme den Wert 0 an.

Nun setzen wir die beiden Annahmen zusammen. Da X und Y unabhängig voneinander sind, tritt das Ereignis $I :=$„$X \geq 0{,}4m$ und $Y \geq 0{,}4(m-2)$“ mit einer Wahrscheinlichkeit von mindestens

$$(1 - e^{-(0{,}05)^2(n/2+1)/4})(1 - e^{-(0{,}05)^2(n/2-1)/4}) \geq 1 - e^{-(0{,}05)^2(n/2+1)/4} - e^{-(0{,}05)^2(n/2-1)/4},$$

also ebenfalls mit exponentiell nahe an 1 liegender Wahrscheinlichkeit, ein. Wir setzen im Folgenden voraus, dass I eingetreten ist. Wir setzen $o^* := x^s_1 + \cdots + x^s_m$; es gilt dann $o^* \leq 0{,}6m$. Nehmen wir außerdem pessimistisch an, dass $0{,}6(m-2)$ (o. B. d. A. eine natürliche

Zahl) Strafterme ihren betragsmäßig maximal möglichen Wert $-3(m+1)$ annehmen, folgt mit Lemma 5.8, dass für den Funktionswert nach der Initialisierung die untere Schranke

$$f(x^s) \geq -o^* - 1{,}8(m-2)(m+1)$$

gilt.

Bei der Betrachtung der Optimierung der Funktion QTrap führen wir für suboptimale Bitstrings die Variablenpaare (o,k), $o \in \{0,\ldots,m\}$ und $k \in \{0,\ldots,1{,}8(m-2)\}$, mit. (Diese hängen eigentlich vom Zeitpunkt $t \in \mathbb{N}$ ab, d. h. der Zahl der Schritte, die der (1+1)-EA bereits gemacht hat; wir lassen den Index t der Übersichtlichkeit halber aber fort.) Die Variablen zeigen an, dass der Funktionswert des aktuellen, noch nicht optimalen Bitstrings $-o-k(m+1)$ (vgl. Lemma 5.8) beträgt. Im Laufe der Optimierung kann k aufgrund des Akzeptanzverhaltens des (1+1)-EA und wegen der Ungleichung $o < m+1$ niemals größer werden. Wir überlegen uns, welche Ereignisse notwendig sind, damit sich o erhöhen kann. In Phasen des (1+1)-EA, während deren sich k nicht ändert, kann dies offensichtlich nicht der Fall sein. Solange das Optimum von QTrap nicht erreicht ist, kann höchstens $1{,}8(m-2)$ Mal eine akzeptierte Mutation stattfinden, die den Wert von k verringert. Infolge einer solchen Mutation kann o steigen, eventuell um jeden beliebigen Wert aus $\{1,\ldots,m-o\}$. Ist k hingegen auf null gesunken, müssen mindestens alle verbleibenden $m-o$ Nullbits im vorderen Block gleichzeitig zu 1 flippen, um das Optimum von QTrap zu erreichen.

Wir nehmen pessimistisch an, dass $t := 1{,}8(m-2)$ Mutationen, die den Wert von k verringern, stattfinden. Um nachzuweisen, dass es höchst unwahrscheinlich ist, dass diese t Mutationen „nebenbei“ zu einem optimalen Bitstring führen, zeigen wir, dass mit exponentiell nahe an 1 liegender Wahrscheinlichkeit während dieser t Mutationen ein Anteil von 1/20 der ursprünglichen mit 0 belegten Bits im vorderen Block nicht mutiert werden. Dazu setzen wir z als Zahl der Nullbits im vorderen Block von x^s – dann gilt $z = m - o^*$ – und nummerieren diese Nullbits mit der streng monoton wachsenden Abbildung $\phi : \{1,\ldots,z\} \to \{1,\ldots,m\}$, sodass $\forall i \in \{1,\ldots,z\} : x^s_{\phi(i)} = 0$ gilt. Außerdem ziehen wir $t \cdot z$ Zufallsvariablen Z_{ij}, $i \in \{1,\ldots,z\}$, $j \in \{1,\ldots,t\}$, heran. Die Zufallsvariable Z_{ij} nimmt den Wert 1 an, wenn das Bit $\phi(i)$ bei der j-ten Mutation geflippt wird. Weil in verschiedenen Schritten vorgenommene Mutationen des (1+1)-EA unabhängig voneinander sind und verschiedene Positionen unabhängig voneinander mutiert werden, sind auch alle Zufallsvariablen Z_{ij} unabhängig. Damit können wir ein drittes Mal die Chernoffschranke anwenden. Wegen $\text{Prob}(Z_{ij} = 1) = 1/n$ folgt mit $Z := \sum_{i=1}^{z}\sum_{j=1}^{t} Z_{ij}$ und $\delta := 19/18 - 1$ (vgl. Lemma A.1)

$$\text{Prob}(Z > (1+\delta)E[Z]) = \text{Prob}\left(Z > \frac{19}{18}\frac{tz}{n}\right) < \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^{\frac{tz}{n}} < 0{,}9985^{\frac{tz}{n}}$$

und daraus wegen $t := 1{,}8(m-2) = \Omega(n)$ und $z \geq 0{,}4m = \Omega(n)$

$$\text{Prob}\left(Z > \frac{19}{18}\frac{1{,}8(m-2)z}{2(m-1)}\right) < 0{,}9985^{\Omega(n)}$$

und damit $\text{Prob}(Z > \frac{19}{20}z) = 2^{-\Omega(n)}$. Dies bedeutet, dass mit Wahrscheinlichkeit $1 - 2^{-\Omega(n)}$ während t Mutationen mindestens $(1/20)z$ Nullen nicht geflippt werden.

Insgesamt erzeugt der (1+1)-EA also während der Optimierung von QTrap mit exponentiell nahe an 1 liegender Wahrscheinlichkeit nur suboptimale Bitstrings mit höchstens $o^* + (19/20)z$ Einsen im vorderen Block. Solange kein Optimum erreicht ist, müssen dann mindestens $m - o^* - (19/20)z = z - (19/20)z = (1/20)z$ Nullen im vorderen Block gleichzeitig zu 1 flippen, um einen optimalen Bitstring zu erreichen. Wegen $z \geq 0{,}4m$ sind dies mindestens $(1/20)(0{,}4m) = (1/50)(n/2 + 1) \geq n/100 = \Omega(n)$ Nullen.

Wir wissen (vgl. Lemma A.8), dass die Wahrscheinlichkeit, mindestens $n/100$ Bits gleichzeitig zu flippen, durch $1/((n/100)!)$ nach oben beschränkt ist. Die erwartete Zeit bis zu einer solchen Mutation beträgt also unter Zuhilfenahme der Stirlingformel mindestens $(n/100)! = 2^{\Omega(n \log n) - O(n)} = 2^{\Omega(n \log n)}$. Indem wir auf $(n/100)!$ Schritte ein letztes Mal die Chernoffschranke anwenden, sehen wir, dass die Wahrscheinlichkeit, in höchstens $((n/100)!)/2$ Schritten mindestens eine solche Mutation auszuführen, exponentiell klein ist. Dies beschließt den Beweis. □

Anders als bei Quadraten linearer Funktionen sind Multistartstrategien zur Optimierung von QTrap also kein Ausweg. Selbst bei polynomiell vielen parallelen Instanzen des (1+1)-EA ist die Wahrscheinlichkeit, dass mindestens eine Instanz in polynomieller Zeit einen optimalen Bitstring findet, noch exponentiell klein.

5.5 Zusammenfassung

Wir haben gesehen, dass die Optimierung quadratischer Funktionen ein NP-hartes Problem ist. Das überzeugt uns von der Existenz „besonders schwieriger“ quadratischer Funktionen, die ein randomisierter Algorithmus wie der (1+1)-EA mit sehr hoher Wahrscheinlichkeit nicht in polynomieller Zeit optimieren kann.

Im zweiten und dritten Abschnitt haben wir zwei Kandidaten für besonders schwierige Funktionen vorgestellt. Es bleibt aber ein offenes Problem, deren Schwierigkeit nachzuweisen oder zu widerlegen.

Die im vierten Abschnitt vorgestellte Funktion QTrap ist beweisbar besonders schwierig. Zum einen unterstützt dieser Beweis natürlich die gängigen Annahmen der Komplexitätstheorie. Zum anderen widerlegt er die intuitiv nahe liegende Hypothese, dass Funktionen „kleinen“ Grades einfach zu optimieren seien.

Anhang A

Häufig verwendete Symbole und Abschätzungen

A.1 Wichtige und eventuell nicht allgemein gebräuchliche Symbole

$\mathbb{N}$	natürliche Zahlen: $\{1, 2, \dots\}$
$\mathbb{N}_0$	$\mathbb{N} \cup \{0\}$
$\mathbb{Z}$	ganze Zahlen
$\mathbb{Z}_p$	ganze Zahlen modulo p
$\mathbb{Q}$	rationale Zahlen
$\mathbb{R}$	reelle Zahlen
$\mathbb{R}^+$	nicht negative reelle Zahlen
$\mathbb{R}^-$	nicht positive reelle Zahlen
$\mathbb{R}^{>0}$	positive reelle Zahlen
$\mathbb{R}^{<0}$	negative reelle Zahlen
$[a, b]$	abgeschlossenes Intervall $(a, b \in \mathbb{R})$
$]a, b[$	offenes Intervall $(a, b \in \mathbb{R})$
$\vec{0}$	Nullvektor
$\vec{1}$	Einsvektor
$\subseteq$	Teilmenge
$\subset$	echte Teilmenge
$\#M$	Kardinalität der Menge M
$\|a\|$	Betrag von $a \in \mathbb{R}$
$E[X]$	Erwartungswert der Zufallsvariablen X
$E[X\|E]$	bedingter Erwartungswert von X unter Ereignis E
Prob	Wahrscheinlichkeitsmaß
$\text{Prob}(E_1\|E_2)$	bedingte Wahrscheinlichkeit von E_1 unter E_2
$H(n)$	Harmonische Reihe bis zum Glied $1/n$

ln	natürlicher Logarithmus
log	Logarithmus zur Basis 2
x^s	Bitstring nach der Initialisierung
$\bar{x}$	Komplement von $x \in \{0,1\}^n$
$\equiv_f$	Relation „quadratische Verknüpfung“
$[i]$	Äquivalenzklasse von $i \in \mathbb{N}$ bezüglich $\equiv_f$

Tabelle A.1: Häufig verwendete Symbole

A.2 Gebräuchliche Abschätzungen und Identitäten

In der folgenden Auflistung finden sich häufig genutzte Abschätzungen und Identitäten sowie wahrscheinlichkeitstheoretische Lemmata. Sie sind in Anlehnung an den Anhang von „Randomized Algorithms“ [MR95] dargestellt.

1. $n! = \sqrt{2\pi}e^{-n}n^{n+1/2}\left(1 + \frac{1}{12n} + O\left(\frac{1}{n^2}\right)\right)$ *(Stirlingformel)*

2. $\binom{n}{k} = \binom{n}{n-k}$ für $n \geq k \geq 0$

3. $\binom{n}{k} \leq \binom{n}{k'} \Longleftrightarrow k \leq k'$ für $n \in \mathbb{N}$, $k, k' \in \{0, \ldots, n/2\}$

4. $\binom{n}{k} \leq \frac{n^k}{k!}$ für $n \geq k \geq 0$

5. $\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$ für $n \geq k \geq 0$

6. $\binom{n}{k} \geq \left(\frac{n}{k}\right)^k$ für $n \geq k \geq 0$

7. $\left(1 - \frac{1}{n}\right)^n \leq e^{-1}$ für $n \in \mathbb{N}$

8. $\left(1 - \frac{1}{n}\right)^{n-1} \geq e^{-1}$ für $n \in \mathbb{N} \setminus \{1\}$

9. $\sum_{i=1}^{n} \frac{1}{i} = \ln n + \Theta(1)$ für $n \in \mathbb{N}$ *(Harmonische Reihe)*

Lemma A.1 (Chernoffschranke) *Seien $X_1, \ldots, X_n$ unabhängige Zufallsvariablen mit dem Wertebereich $\{0,1\}$ und $\text{Prob}(X_i = 1) = p_i$, $\text{Prob}(X_i = 0) = 1 - p_i$ und $p_i \in]0,1[$ für alle $i \in \{1, \ldots, n\}$. Mit $X := X_1 + \cdots + X_n$ und $\mu := p_1 + \cdots + p_n$ gilt dann für $0 < \delta \leq 1$*

$$\text{Prob}(X < (1-\delta)\mu) < \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu}$$

und

$$\text{Prob}(X > (1+\delta)\mu) < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}.$$

Im Rahmen dieser Arbeit sind bei allen Anwendungen die Werte p_i für alle i identisch.

Für Abweichungen unter den Erwartungswert wie in der ersten Ungleichung aus Lemma A.1 existiert eine vereinfachte Abschätzung (siehe [MR95, Theorem 4.2]):

Lemma A.2 *Seien $X_1, \ldots, X_n$ unabhängige Zufallsvariablen mit dem Wertebereich $\{0,1\}$ und $\text{Prob}(X_i = 1) = p_i$, $\text{Prob}(X_i = 0) = 1 - p_i$ und $p_i \in]0,1[$ für alle $i \in \{1, \ldots, n\}$. Mit $X := X_1 + \cdots + X_n$ und $\mu := p_1 + \cdots + p_n$ gilt dann für $0 < \delta \leq 1$*

$$\text{Prob}(X < (1-\delta)\mu) < e^{-\mu\delta^2/2}.$$

Lemma A.3 (Markoffungleichung) *Sei X eine nicht negative Zufallsvariable. Für alle $t \in \mathbb{R}^{>0}$ gilt*

$$\text{Prob}(X \geq tE[X]) \leq \frac{1}{t}.$$

Lemma A.4 (Linearität des Erwartungswertes) *Seien $X_1, \ldots, X_n$ beliebige Zufallsvariablen und $X := X_1 + \cdots + X_n$. Es gilt*

$$E[X] = \sum_{i=1}^{n} E[X_i].$$

A.3 Wahrscheinlichkeiten für Mutationen

Definition A.1 ((1+1)-EA) *Der (1+1)-EA erhält eine Funktion $f : \{0,1\}^n \to \mathbb{R}$ als Eingabe. Er erzeugt in einer Endlosschleife probabilistisch Vektoren $x \in \{0,1\}^n$:*

1. *Initialisiere zufällig gleichverteilt einen Bitstring $x \in \{0,1\}^n$.*
2. *Wiederhole:*
   (a) *Erzeuge x^* aus x, indem jedes Bit von x unabhängig mit Wkt. $1/n$ negiert wird.*
   (b) *Ersetze x durch x^*, gdw. $f(x^*) \geq f(x)$.*

Eine konsekutive Ausführung der Anweisungen (a) und (b) heißt Schritt.

Lemma A.5 *Für die Zufallsvariable X, definiert als die Anzahl mutierter Bits in einer Ausführung von Anweisung (a) des (1+1)-EA (Def.* A.1*), gilt $E[X] = 1$.*

Beweis: Nutze die Indikatorvariablen X_i, $i \in \{1, \ldots, n\}$, mit $X_i = 1$, falls Bit i geflippt wird, und $X_i = 0$ sonst. Es ist $E[X_i] = \text{Prob}(X_i = 1) = 1/n$ aufgrund der konstanten Mutationswahrscheinlichkeit. Wegen $X = X_1 + \cdots + X_n$ folgt die Aussage aus der Linearität des Erwartungswertes (Lemma A.4). □

Lemma A.6 *Bezeichne die Zufallsvariable X die Anzahl mutierter Bits in einer Ausführung von Anweisung (a) des (1+1)-EA (Def.* A.1*). Für alle $k \in \{0, \ldots, n\}$ gilt:*

$$\lim_{n\to\infty} \text{Prob}(X = k) = e^{-1}\frac{1}{k!}.$$

Beweis: Die Zufallsvariable X ist aufgrund der konstanten Mutationswahrscheinlichkeit binomialverteilt mit den Parametern n (Anzahl der Experimente) und $1/n$ (Erfolgswahrscheinlichkeit). Für wachsendes n konvergiert die Verteilung von X gegen die Poissonverteilung. Wegen $E[X] = 1$ gilt

$$\lim_{n\to\infty} \text{Prob}(X = k) = e^{-E[X]}\frac{(E[X])^k}{k!} = e^{-1}\frac{1}{k!}.$$

□

Lemma A.7 *Die Wahrscheinlichkeit für eine Mutation von k, $k \in \{1, \ldots, n\}$, Bits in Anweisung (a) des (1+1)-EA (Def.* A.1*) beträgt mindestens $e^{-1}k^{-k}$.*

Beweis: Die Wahrscheinlichkeit, in einem Schritt k Bits zu flippen, lässt sich durch

$$\binom{n}{k}\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{n-k} \geq \left(\frac{n}{k}\right)^k\left(\frac{1}{n}\right)^k e^{-1} = \left(\frac{1}{k}\right)^k e^{-1}$$

nach unten abschätzen. □

Lemma A.8 *Die Wahrscheinlichkeit für eine Mutation von mindestens k, $k \in \{1, \ldots, n\}$, Bits in einem Schritt des (1+1)-EA (Def.* A.1*) beträgt höchstens $1/k!$.*

Beweis: Die Wahrscheinlichkeit, mindestens k Bits zu flippen, lässt sich mit dem Ereignis „k Bits flippen, die übrigen $n-k$ Positionen bleiben mit Wahrscheinlichkeit 1 unverändert oder flippen ebenfalls“ gemäß

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

nach oben abschätzen. □

Lemma A.9 *Die Wahrscheinlichkeit, dass in Anweisung (a) des (1+1)-EA (Def.* A.1*) genau ein eindeutig bestimmtes Bit flippt, beträgt mindestens* $e^{-1}n^{-1}$.

Beweis: Die Wahrscheinlichkeit, dass genau ein eindeutig bestimmtes Bit flippt, lässt sich durch

$$\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq e^{-1}n^{-1}$$

nach unten abschätzen. □

Lemma A.10 *Die Wahrscheinlichkeit, dass in* $cn\ln n$ *Schritten des (1+1)-EA (Def.* A.1*) mindestens einmal eine Mutation von mindestens* $\log n$ *Bits zugleich eintritt, konvergiert für jedes konstante* $c \in \mathbb{R}^{>0}$ *und wachsendes* $n \in \mathbb{N}$ *gegen null.*

Beweis: O. B. d. A. ist $s := cn\ln n$ eine natürliche Zahl. Sei E_i, $i \in \{1,\dots,s\}$, das Ereignis, dass in Schritt i des EA mindestens $\log n$ Bits zugleich flippen. Nach Lemma A.8 ist $\text{Prob}(E_i) \leq 1/(\log n)!$ für alle $i \in \{1,\dots,s\}$. Die Wahrscheinlichkeit des Ereignisses $E := \bigcup_{i=1}^{s} E_i$ ist höchstens die Summe der Wahrscheinlichkeiten der Einzelereignisse E_i, also folgt mit der Stirlingformel

$$\begin{aligned}\text{Prob}(E) \leq& \frac{cn\ln n}{(\log n)!} \leq \frac{cn\ln n}{e^{-\log n}(\log n)^{(\log n)}} = 2^{\log(cn\ln n)+(\log e)(\log n)-(\log\log n)(\log n)}\\ &= 2^{O(\log n)-\Omega((\log\log n)(\log n))} = 2^{-\Omega((\log\log n)(\log n))} \xrightarrow{n\to\infty} 0.\end{aligned}$$

□

Lemma A.11 *Die Wahrscheinlichkeit, dass in* $cn\ln n$ *Schritten des (1+1)-EA (Def.* A.1*) mindestens einmal eine Mutation von mindestens* n^d *Bits zugleich eintritt, konvergiert für konstante* $c, d \in \mathbb{R}^{>0}$ *und wachsendes* $n \in \mathbb{N}$ *gegen null.*

Beweis: Da $n^d = \omega(\log n)$ für jedes $d \in \mathbb{R}^{>0}$, folgt die Behauptung direkt aus Lemma A.10. □

Lemma A.12 *Die Wahrscheinlichkeit, in Anweisung (a) des (1+1)-EA (Def.* A.1*) eine Mutation von mindestens* $k \in \mathbb{N}$ *Bits,* k *konstant, zugleich auszuführen, ist durch eine Konstante nach unten beschränkt.*

Beweis: Die Wahrscheinlichkeit, dass mindestens k Bits zugleich flippen, ist durch die Wahrscheinlichkeit, dass genau k Bits zugleich flippen, nach unten beschränkt, die mindestens

$$\binom{n}{k}\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{n-k} \geq \left(\frac{n}{k}\right)^k\left(\frac{1}{n}\right)^k e^{-1} = k^{-k}e^{-1} = \Theta(1)$$

beträgt. □

Literaturverzeichnis

[Bäc92] Bäck, T.: *The interaction of mutation rate, selection, and self-adaption within a genetic algorithm.* In: Männer, R. und B. Manderick (Herausgeber): *Parallel Problem Solving from Nature II*, Seiten 85–94, Amsterdam, 1992. North Holland.

[Bäc94] Bäck, T.: *Evolutionary Algorithms in Theory and Practice.* Dissertation, Universität Dortmund, 1994.

[Bey93] Beyer, H.-G.: *Toward a theory of evolution strategies: Some asymptotical results from the* $(1\overset{+}{,}\mu)$*-theory.* Evolutionary Computation, 1(2):163–188, 1993.

[Bey95] Beyer, H.-G.: *How GAs do NOT work: Understanding GAs without schemata and building blocks.* Technischer Bericht SYS-2/95, Universität Dortmund, Lehrstuhl für Systemanalyse, Apr. 1995.

[BH91] Belew, R. K. und W. E. Hart: *Optimizing an arbitrary function is hard for the genetic algorithm.* In: Belew, R. K. und L. B. Booker (Herausgeber): *Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications*, Seiten 190–195, San Diego, USA, 1991. University of California, Morgan Kaufmann Publishers.

[BHS92] Bäck, T., F. Hoffmeister und H.-P. Schwefel: *Applications of evolutionary algorithms.* Technischer Bericht SYS-2/92, Universität Dortmund, Lehrstuhl für Systemanalyse, 1992.

[BNKF98] Banzhaf, W., P. Nordin, R. Keller und F. Francone: *Genetic Programming.* Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[DJW98a] Droste, S., T. Jansen und I. Wegener: *On the analysis of the (1+1) Evolutionary Algorithm.* Technischer Bericht, SFB 531 CI-21/98, Universität Dortmund, 1998.

[DJW98b] Droste, S., T. Jansen und I. Wegener: *A rigorous complexity analysis of the (1+1) Evolutionary Algorithm for linear functions with Boolean inputs.* In: *Proceedings of the IEEE International Conference on Evolutionary Computation ICEC '98*, Seiten 499–504, Piscataway, NJ, 1998. IEEE Press.

[DJW99a] Droste, S., T. Jansen und I. Wegener: *A natural and simple function which is hard for all evolutionary algorithms.* Technischer Bericht, SFB 531, Universität Dortmund, 1999.

[DJW99b] Droste, S., T. Jansen und I. Wegener: *Perhaps not a free lunch but at least a free appetizer.* In: Banzhaf, W., J. Daida, A. E. Eiben, M. H. Garzon, V. Honovar, M. Jakiela und R. E. Smith (Herausgeber): *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99)*, Seiten 833–839, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[Fog95] Fogel, D. B.: *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press, Piscataway, NJ, 1995.

[GJ76] Garey, M. R. und D. S. Johnson: *Some simplified NP-complete graph problems.* Theoretical Computer Science, 1:237–267, 1976.

[GJ79] Garey, M. R. und D. S. Johnson: *Computers and Intractability – A Guide to the Theory of NP-Completeness.* W. H. Freeman And Company, San Francisco, 1979.

[GKP89] Graham, R. L., D. E. Knuth und O. Patashnik: *Concrete Mathematics.* Addison-Wesley, 6. Auflage, 1989.

[Gol89] Goldberg, D. E.: *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, 1989.

[HS89] Hammer, P. L. und B. Simeone: *Quadratic functions of binary variables.* In: Simeone, B. (Herausgeber): *Combinatorial Optimization*, Lecture Notes in Mathematics, Seiten 1–56. Springer, Berlin, Heidelberg, 1989.

[Jan99] Jansen, T.: *On classifications of fitness functions.* Technischer Bericht, SFB 531 CI-76/99, Universität Dortmund, Nov. 1999.

[JW99] Jansen, T. und I. Wegener: *On the analysis of evolutionary algorithms – a proof that crossover really can help.* In: Nešetřil, J. (Herausgeber): *Proceedings of the 7th Ann. European Symposium on Algorithms (ESA '99)*, Seiten 184–193, Berlin, 1999. Springer.

[Müh91] Mühlenbein, H.: *Evolution in time and space – the parallel genetic algorithm.* In: Rawlins, G. J. E. (Herausgeber): *Foundations of Genetic Algorithms*, Seiten 316–337. Morgan Kaufmann Publishers, San Mateo, CA, 1991.

[Müh92] Mühlenbein, H.: *How genetic algorithms really work I. Mutation and hillclimbing.* In: Männer, R. und B. Manderick (Herausgeber): *Parallel Problem Solving from Nature II*, Seiten 15–25, Amsterdam, 1992. North Holland.

[MR95] Motwani, R. und P. Raghavan: *Randomized Algorithms*. Cambridge University Press, 1995.

[PS82] Papadimitriou, C. H. und K. Steiglitz: *Combinatorial Optimization - Algorithms and Complexity*. Prentice Hall, 1982.

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

[Ros75] Rosenberg, I. G.: *Reduction of bivalent maximization to the quadratic case*. Cahiers du Centre d'Etudes de Recherche Operationnelle, 17:71–74, 1975.

[RRS98] Rabani, Y., Y. Rabinovich und A. Sinclair: *A computational view of population genetics*. Random Structures and Algorithms, 12(4):313–334, 1998.

[Rud97] Rudolph, G.: *Convergence Properties of Evolutionary Algorithms*. Dissertation, Universität Dortmund, 1997. Verlag Dr. Kovač, Hamburg.

[Sch95] Schwefel, H.-P.: *Evolution and Optimum Seeking*. John Wiley and Sons, 1995.

[TW96] Thompson, R. K. und A. H. Wright: *Additively decomposable fitness functions*. Technischer Bericht, University of Montana, CS Department, 1996.

[Weg93] Wegener, I.: *Theoretische Informatik*. B. G. Teubner, Stuttgart, 1993.

[Weg97] Wegener, I.: *Komplexitätstheorie*. Vorlesungsskript WS 1997/98, Universität Dortmund, 1997.