Max-Planck-Institut für Mathematik in den Naturwissenschaften Leipzig

Schemen in genetischen Algorithmen

by

Susanne Schindler

Preprint no.: 117 2002

Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH)
Fachbereich Informatik, Mathematik und Naturwissenschaften
Postfach 300066
04251 Leipzig

Max-Planck-Institut für Mathematik in den Naturwissenschaften
Inselstraße 22
04103 Leipzig

Diplomarbeit

Schemen in genetischen Algorithmen

Susanne Schindler
Studiengang Wirtschaftsmathematik

Leipzig, den 13. Juni 2002

Betreuer:
Prof. Dr. rer. nat. Jürgen Jost (MPI)
Prof. Dr. rer. nat. habil. Helmut Rudolph (HTWK)

Inhaltsverzeichnis

Bezeichnungen

Allgemeine Bezeichnungen

$\mathbb{R}[a, b]$	Menge der reellen Zahlen, die im Intervall von a bis b liegen.
$D(z)$	Definitionsbereich einer Funktion z.
$P(A)$	Wahrscheinlicheit eines Ereignisses A.
$P(A/B)$	Bedingte Wahrscheinlicheit. Die Wahrscheinlichkeit des Ereignisses A unter der Bedingung, dass B eintritt.
EA	Erwartungswert eines Ereignisses A.
$\|M\|$	Mächtigkeit einer Menge M.
$M \backslash N$	Menge der Elemente aus M, die nicht in N liegen.
$\mathfrak{P}(M)$	Potenzmenge von M

Bezeichnungen für den einfachen genetischen Algorithmus

$\mathfrak{A}$	Suchraum. Die Menge aller möglichen Lösungen eines Optimierungsproblems.
$\mathbb{P}$	Menge aller möglichen Populationen.
$P_t \in \mathbb{P}$	Population der Generation t ($t \in \mathbb{N}_0$). Eine endliche Menge von Elementen des Suchraumes. Es gilt $P_t \not\subset \mathfrak{A}$, da ein $x \in \mathfrak{A}$ mehrmals in P_t auftreten kann.
O_S	Selektionsoperator
O_K	Kreuztauschoperator
O_M	Mutationsoperator

Bezeichnungen aus der Schematheorie

$\Delta = \{\delta_i : \mathfrak{A} \longrightarrow V_i,\ i = 1, \ldots, m,\ m \in \mathbb{N}\}$
Menge der Detektoren. Der Wertebereich des i-ten Detektors ist V_i.

$\star$ Symbol für „egal“, für den Fall, dass der Wert eines Detektors ohne Belang ist.

$H = (s_1, \ldots, s_m)$ mit $s_i \in \{V_i \cup \{\star\}\}$ und $i = 1, \ldots, m$
Schema. Die Menge von Elementen $x \in \mathfrak{A}$, für die gilt: $\delta_i(x) = s_i$ falls $s_i \neq \star\ \forall i = 1, \ldots, m$

$\Xi = \prod_{i=1}^{m} \{V_i \cup \{\star\}\}$ Menge aller Schemen

$o(H)$ Ordnung des Schemas H. Anzahl der fixierten Positionen von H.

$\delta(H)$ Länge des Schemas H. Differenz zwischen der ersten und der letzten fixierten Stelle von H.

$h_1, \ldots, h_{\delta(H)}$ Zeichenzwischenräume des Schemas H

Bezeichnungen für den Schemensatz

$2n$ Populationsgröße

l Zeichenketten- bzw. Chromosomenlänge und Anzahl der Detektoren

p_c Kreuztauschwahrscheinlichkeit

p_m Mutationswahrscheinlichkeit

$F(x)$ Fitnesswert des Elements $x \in \mathfrak{A}$

$\bar{F}_t = \frac{1}{2n} \sum_{x \in P_t} F(x)$ Mittlere Fitness der Population P_t

$f_t(x) = \frac{F(x)}{\sum_{y \in P_t} F(y)}$ Relative Fitness des Individuums $x \in P_t$

$m(H, t) = \left|\{P_t \cap H\}\right|$
Anzahl der Vertreter von H in P_t

$F_t(H) = \frac{1}{m(H,t)} \sum_{x \in P_t \cap H} F(x)$

Mittlere Fitness des Schemas H in P_t

$f_t(H) = \sum_{x \in P_t \cap H} f_t(x)$

Relative Fitness des Schemas H in P_t

H_k	Schema, dessen fixierte Positionen und dessen Allele an diesen Positionen mit denen von H bis zur Stelle $k \in \{h_1, \ldots, h_{\delta(H)}\}$ übereinstimmen.
$\hat{H}_k$	Schema, dessen fixierte Positionen und dessen Allele an diesen Positionen mit denen von H von der Stelle $k \in \{h_1, \ldots, h_{\delta(H)}\}$ an übereinstimmen.
$\bar{H}_i$	Klasse von Schemen, deren Allele an den fixierten Positionen an genau i Positionen verschieden von denen von H sind.
$E_H(H,t)$	Untere Schranke für die erwartete Anzahl an Vertretern eines Schemas in der Folgepopulation P_{t+1}. Gegeben im Schemensatz 2.3.
$E_S(H,t)$	Erwartete Anzahl an Vertretern eines Schemas in der Folgepopulation P_{t+1}. Gegeben im modifizierten Schemensatz 2.7.

Bezeichnungen für das Computerprogramm

$\lfloor a \rfloor$	Größte ganze Zahl, die nicht größer als a ist.
N	Maximale Generationenanzahl

1. Einleitung

In dem folgenden Abschnitt wird eine kurze Einführung in die Grundbegriffe und das Konzept der genetischen Algorithmen gegeben. Diese Einführung ist bewusst auf das Wesentliche reduziert und erhebt nicht den Anspruch vollständig zu sein. Ausführlichere Einführungen in das Thema „Genetische Algorithmen“ können in [6], [14], [15], [16] oder [17] nachgelesen werden. In Abschnitt 1.2 wird ein Ausblick auf den weiteren Inhalt und das Ziel dieser Diplomarbeit präsentiert.

1.1. Das Grundkonzept genetischer Algorithmen

Genetische Algorithmen bilden eine Untergruppe der evolutionären Algorithmen. Letztere werden seit den 60er Jahren vor allem als Alternative zu traditionellen Optimierungsverfahren diskutiert, weiterentwickelt und mathematisch modelliert.

Ein Optimierungsverfahren wird zur Lösung von Optimierungsproblemen verwendet. Ein Optimierungsproblem besteht in der Minimierung oder der Maximierung einer Funktion von mehreren Variablen. Diese Funktion wird **Zielfunktion** genannt. Die Menge aller möglichen Variablenbelegungen wird im Folgenden als **Suchraum** bezeichnet. Bei einem Optimierungsproblem können eine oder mehrere Bedingungen an die Variablen oder die Zielfunktionswerte gestellt werden. Als traditionelle Optimierungsverfahren werden in dieser Arbeit diejenigen deterministischen Verfahren zusammengefasst, die zumeist stark auf einen bestimmten Fall der Optimierung spezialisiert sind. Dazu gehören zum Beispiel der Simplexalgorithmus, das Newton-Verfahren, Varianten der Quasi-Newton-Verfahren und Trust-Region-Verfahren. Bei diesen Verfahren werden Stetigkeit, Differenzierbarkeit oder Konvexität der Zielfunktion vorausgesetzt. Sie bestimmen in der Regel entweder eine Folge von Variablenbelegungen, die gegen das Optimum konvergieren soll oder bestimmen durch Transformation des ursprünglichen Optimierungsproblems zu einem anderen das Optimum des Ausgangsproblems.

In der Literatur findet sich derzeit keine einheitliche Definition von einem evolutionären Algorithmus. Für diese Arbeit werden sie deshalb folgendermaßen definiert:

Definition 1.1 *Ein evolutionärer Algorithmus ist ein stochastisches Verfahren, welches Lösungsansätze für Optimierungsprobleme bietet und bestimmte Elemente aus der Evolutionstheorie modelliert.*

Evolutionäre Algorithmen basieren ähnlich der darwinschen Lehre auf dem Prinzip „Der Stärkere überlebt“. Das Ziel ist, Lösungen in einer Programmumgebung selbstständig entwickeln zu lassen, indem man die der Evolution zugrundeliegend geglaubten Prozesse (auch **Evolutionsfaktoren** genannt) modelliert.

Die zentralen Evolutionsfaktoren sind die Vererbung und Vermischung des Erbgutes, die Selektion der am besten an die Umwelt angepassten Individuen und die Mutation.

Evolutionäre Algorithmen arbeiten mit einer **Population** P_t, deren Elemente **Individuen** genannt werden (Seite 11). Diese Individuen repräsentieren Elemente des Suchraumes, also konkrete Variablenbelegungen. Dabei können zwei oder mehr Individuen auch dieselben Belegungen repräsentieren. Die Individuen werden im Folgenden mit den Elementen, die sie repräsentieren, identifiziert.

Durch Anwendung sogenannter **genetischer Operatoren** auf die Population P_t unterliegt diese Population generationsähnlichen[1] Veränderungen. Der Index $t \in \mathbb{N}_0$ der Population P_t bezeichnet dabei die Generation, in der sich die Population befindet. Genetische Operatoren modellieren die oben genannten Evolutionsfaktoren. Zu den elementarsten genetischen Operatoren gehören die Fitnesszuweisung, die Selektion, die Rekombination und die Mutation. Diese Operatoren werden zu einem späteren Zeitpunkt in diesem Abschnitt näher beschrieben.

An dieser Stelle soll zunächst der Ablauf eines evolutionären Algorithmus betrachtet werden. Dieser besteht aus dem aufeinander folgenden Anwenden der genetischen Operatoren. Es wird die Reihenfolge der genetischen Operatoren beschrieben, die in den evolutionären Algorithmen, die in dieser Arbeit untersucht werden, verwendet wird. Ausgehend von einer Anfangspopulation P_0 werden Folgepopulationen P_{t+1} in folgender Weise gebildet: Nach einer Fitnesszuweisung werden einige Individuen, ausgewählt (Selektionsoperator), um Klone, d. h. exakte Kopien ihrerselbst, zu erhalten. Die Auswahl dieser Individuen kann auf Grund der Erfüllung der an die Variablen gestellten Bedingungen erfolgen, oder durch einen besonders kleinen oder großen Wert beim Einsetzen in die Zielfunktion begründet sein. Anstelle von „Klonen“ spricht man häufiger von „**Nachkommen**“. Diese Nachkommen werden durch den Rekombinationsoperator und den Mutationsoperator transformiert. Darauffolgend werden die Individuen der Folgepopulation P_{t+1} aus den Individuen von P_t und den transformierten Nachkommen bestimmt. Auf welche Art die Folgepopulation zusammengesetzt wird, oder auf welche Weise die Transformationen des Rekombinations- und Mutationsoperators durchgeführt werden, ist problemabhängig und kann beliebig festgelegt werden. Die ursprüngliche Population wird dann durch P_{t+1} ersetzt. Der Zyklus vom Anwenden des Selektionsoperators bis zum Ersetzen der Population wird als **Gene-**

[1] Das Wort „generationsähnlich“ wird in dem Sinn gebraucht, dass es einen Zusammenhang zwischen den Elementen von P_t und P_{t+1} gibt. Die Individuen von P_{t+1} gehen aus Transformationen der Individuen von P_t hervor.

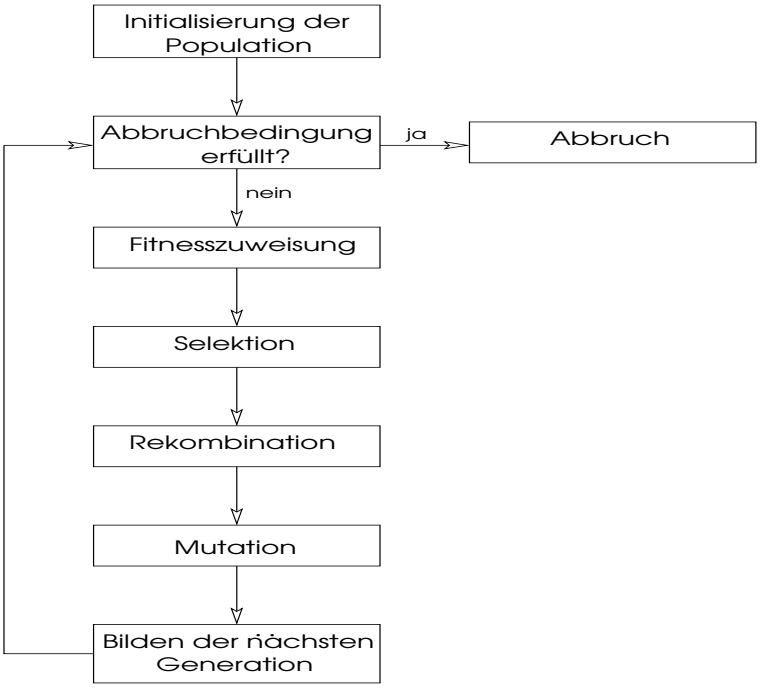

Abbildung 1.1.: Beispiel für den Ablauf eines evolutionären Algorithmus

ration bezeichnet. Anschließend werden wieder der Fitnesszuweisungs-, der Selektions-, der Rekombinations- und der Mutationsoperator auf die Population angewendet, solange bis ein bestimmtes Abbruchskriterium erfüllt ist. Das kann eine maximale Generationenanzahl sein oder eine ausbleibende signifikante Änderung des besten Zielfunktionswertes eines Individuums in der Population über mehrere Generationen. In Abbildung 1.1 wird der oben beschriebene Ablauf eines evolutionären Algorithmus graphisch dargestellt.

Durch das wiederholte Anwenden der genetischen Operatoren erhofft man sich eine schrittweise Annäherung der Individuen an das Optimum oder die Optima des zu Grunde liegenden Optimierungsproblems.

Für einen genetischen Algorithmus existiert ebensowenig wie für einen evolutionären Algorithmus eine einheitliche Definition. In der Mehrzahl der Veröffentlichung werden genetische Algorithmen als Verfahren betrachtet, welche mit binären Zeichenketten arbeiten und jeweils eine Form der Selektion, der Rekombination und der Mutation verwenden. Für diese Arbeit wird der genetische Algorithmus folgendermaßen definiert:

Definition 1.2 *Ein genetischer Algorithmus ist ein evolutionärer Algorithmus, bei dem die Individuen der Population binär kodiert sind und in dem die Fitnesszuweisung, die Selektion, die Rekombination und die Mutation implementiert sind.*

Handelt es sich bei dem Suchraum um den Raum der reellen Zah-

len, so können die Individuen der Population z. B. binäre Darstellungen der reellen Zahlen sein. Die Bezeichnungen der Objekte eines genetischen Algorithmus sind stark an die der Biologie angelehnt. So arbeitet ein genetischer Algorithmus mit **Chromosomen**, **Genen** und **Allelen**. In der Biologie ist die Erbinformation in den Chromosomen gespeichert, ein Gen entspricht einem Chromosomenabschnitt und die verschiedenen möglichen Ausprägungen eines Genes werden Allele genannt. Bei genetischen Algorithmen werden die aus der Kodierung resultierenden binären Zeichenketten als Chromosomen bezeichnet. Die Gene kodieren einzelne Eigenschaften der Elemente des Suchraumes und können mehrere Zeichenpositionen vereinigen. In dieser Arbeit werden jedoch die einzelnen Zeichenpositionen als Gene und die Bitbelegung an den einzelnen Zeichenpositionen als Allel bezeichnet.

Im Folgenden werden einige genetische Operatoren wie die Fitnesszuweisung, der Selektionsoperator, der Rekombinationsoperator und der Mutationsoperator detaillierter beschrieben und konkrete Beispiele dieser Operatoren angeführt. Dabei werden nur solche Beispieloperatoren vorgestellt, die auf binären Zeichenketten agieren können und somit für den Gebrauch in genetischen Algorithmen geeignet sind.

Fitnesszuweisung Die Fitnesszuweisung erfolgt meistens vor der Selektion. Jedem Individuum wird in Abhängigkeit von seinem Zielfunktionswert und der Erfüllung oder Nichterfüllung der Bedingungen, die an die Variablen gestellt werden ein Fitnesswert zugewiesen. Dieser ist ein Maß für die Anzahl an Nachkommen, die das Individuum bekommen würde, wenn es selektiert würde. In dieser Arbeit wird die ***proportionale Fitnesszuweisung***, welche vor der Selektion stattfindet, genutzt (Seite 11). Hier ist die Fitness eines Individuums proportional zu seinem Zielfunktionswert. Im Allgemeinen kann die Fitnesszuweisung auch nach der Mutation erfolgen. Dann ist die Fitness eines Individuums ein Maß dafür, wie wahrscheinlich es ist, dass es in der Population ersetzt oder in die Folgepopulation eingefügt wird.

Selektion Die Selektion kann vor oder nach der Fitnesszuweisung erfolgen. Sie legt fest, welche Individuen nun tatsächlich Nachkommen erhalten. Die Selektion kann aber auch nach der Anwendung der Rekombination und der Mutation stattfinden. In diesem Fall bestimmt sie, welche Individuen durch andere beim Bilden der Folgepopulation ersetzt oder eingefügt werden. Im Gegensatz zu anderen genetischen Operatoren kann die Selektion, dadurch dass sie bestimmte Individuen auswählt oder nicht auswählt, eine Richtung der Entwicklung der Population bestimmen. Die in dieser Arbeit gebrauchte Selektionsform ist die ***fitnessproportionale Selektion*** (Seite 13). Sie bestimmt, welche Individuen zur Rekombination ausgewählt werden. Die Wahrscheinlichkeit, dass ein Individuum ausgewählt wird, entspricht seiner relativen Fitness, welche sich aus dem Verhältnis seines Fitnesswertes zu der Summe der Fitnesswerte aller in

der Population befindlichen Individuen errechnet.

Rekombination Die Rekombination wirkt auf den Chromosomen von mindestens zwei Individuen und wird meist in einer Form des Überkreuzungsaustausches[2] (auch Crossing-over genannt) genutzt. Das Ziel der Rekombination ist das Einbringen von neuen Individuen in die Population, sowie die Kombination zweier guter Individuen zu hoffentlich noch besseren Nachkommen („Building block hypothesis“ [6]). Besonders häufig wird der **n-*Punkt Kreuztausch*** mit $n = 1$ verwendet (Seite 15). Mit der Kreuztauschwahrscheinlichkeit $\mathbf{p_c}$ werden gemäß einer gleichmäßigen Verteilung n Schnittstellen zwischen zwei Genen zweier Chromosomen ausgewählt und die dazwischenliegenden Abschnitte ausgetauscht.

Mutation Die Mutation operiert auf den Zeichenketten einzelner Individuen meistens nach dem Rekombinationsschritt. Der Zweck der Mutation liegt in dem Erhalt der Diversität der Population, denn als einziger genetischer Operator ändert die Mutation die Allelenhäufigkeit. Ist ein Allel an einem bestimmten Gen in allen Chromosomen verloren, also in keinem Chromosom der Individuen vorhanden, so kann dieses Allel nur durch die Mutation wieder in die Population gelangen. Bei der ***bitweisen Mutation*** wird jedes Allel mit der Mutationswahrscheinlichkeit $\mathbf{p_m}$ negiert, d. h. eine „0“ durch eine „1“ ersetzt und umgekehrt (Seite 17).

Einige Unterschiede zwischen evolutionären Algorithmen und traditionellen Optimierungsverfahren, und zwar der stochastische Charakter und die Verwendung von Populationen, wurden bereits angesprochen. Ein weiterer wesentlicher Unterschied besteht darin, dass evolutionäre Algorithmen häufig mit Kodierungen der Elemente aus dem Suchraum arbeiten. Die aus der Kodierung mit kleinen Alphabeten hervorgehenden Zeichenketten erlauben nicht nur zahlreichere Abänderungen, sondern bieten einen größeren Interpretationsspielraum bei der Suche nach optimalen Eigenschaften [7]. Folgendes Beispiel ist an [7] angelehnt:

Individuum	Element des Suchraumes $\mathbb{N}$	Chromosom	Fitness
x_1	22	10110	7
x_2	3	00011	8
x_3	26	11010	1
x_4	10	01010	2

Versucht man über den Grund der guten Fitnesswerte der ersten beiden Individuen x_1 und x_2 zu spekulieren, findet man wenig Hinweise, wenn man nur die natürlichen Zahlen betrachtet, die die Individuen repräsentieren. Schaut man jedoch auf die binäre Darstellung, so könnte man vermuten, dass sich die hohe Fitness von x_1 und x_2 in der „0“ am zweiten Gen

[2] Die Bezeichnung „Überkreuzungsaustausch“ wird im Folgenden mit „Kreuztausch“ abgekürzt werden.

in ihren Chromosomen begründet. Auch bei der Suche nach sinnvollen Kombinationen zweier Elemente schneiden Zeichenketten bezüglich der Anzahl verschiedener Kombinationsmöglichkeiten besser ab, als es einzelne Werte tun.

Durch das einfache und wenig spezialisierte Konzept (Abbildung 1.1), besitzen evolutionäre Algorithmen breite Anwendungsmöglichkeiten. Sie finden aber im Vergleich mit traditionellen Optimierungsverfahren nicht immer die Optimallösung. Ein Vorteil evolutionärer Algorithmen liegt in dem schnellen Finden einer Lösung. Besonders geeignet sind evolutionäre Algorithmen daher bei Optimierungsproblemen, bei denen möglichst rasch eine Lösung nahe des Optimums gefunden werden soll, es jedoch nicht unbedingt erforderlich ist, das exakte Optimum zu finden. Sie eignen sich weniger, wenn die Auswertung der Zielfunktion besonders zeitintensiv ist, denn die Verwendung von Populationen erfordert zahlreiche Zielfunktionswertberechnungen.

Die populationsbasierte Vorgehensweise bietet jedoch den Vorteil, dass mehrere Optima bei Vektoroptimierungsproblemen oder bei multimodalen Zielfunktionen gefunden werden können. Da keinerlei Informationen über die Zielfunktion, abgesehen von den Funktionswerten, benötigt und auch keine Bedingungen an die Zielfunktion gestellt werden, eignen sich evolutionäre Algorithmen auch bei nicht linearen und unstetigen Zielfunktionen.

Zum Abschluss dieser kurzen Einführung in genetische Algorithmen als eine Form evolutionärer Algorithmen soll Abbildung 1.2 einen Überblick zur Einordung und eine Zusammenfassung der gebräuchlichsten Arten der evolutionären Algorithmen bieten. Unabhängig voneinander wurden drei Gruppen von Algorithmen entwickelt, welche Evolutionsprozesse simulieren. Alle drei Gruppen werden heute als evolutionäre Algorithmen zusammengefasst.

1.2. Inhalt und Ziel der Diplomarbeit

Es gibt mehrere Versuche, das Verhalten genetischer Algorithmen mathematisch zu erfassen und zu begründen. Einen Ansatz bietet der Schemensatz von Holland [11]. Dieser bietet eine untere Schranke für die erwartete Anzahl von Individuen mit wünschenswerten Eigenschaften in der Folgepopulation. Der Kernpunkt dieser Arbeit wird die Herleitung eines modifizierten Schemensatzes sein, welcher eine Verbesserung der erwähnten Schranke bieten soll. Bevor dies getan werden kann ist es nötig, die genetischen Operatoren zu konkretisieren und mathematisch zu erfassen. Die recherchierte Literatur bietet für das Letztere keine oder keine zufrieden stellende Ansätze. Eine weitere Zielstellung dieser Arbeit ist daher die mathematische Beschreibung der Funktionsweise eines konkreten genetischen Algorithmus. Die Vorbereitung für die Herleitung der Modifikation des Schemensatzes von Holland, sowie derselbige befinden sich in Kapitel 2.

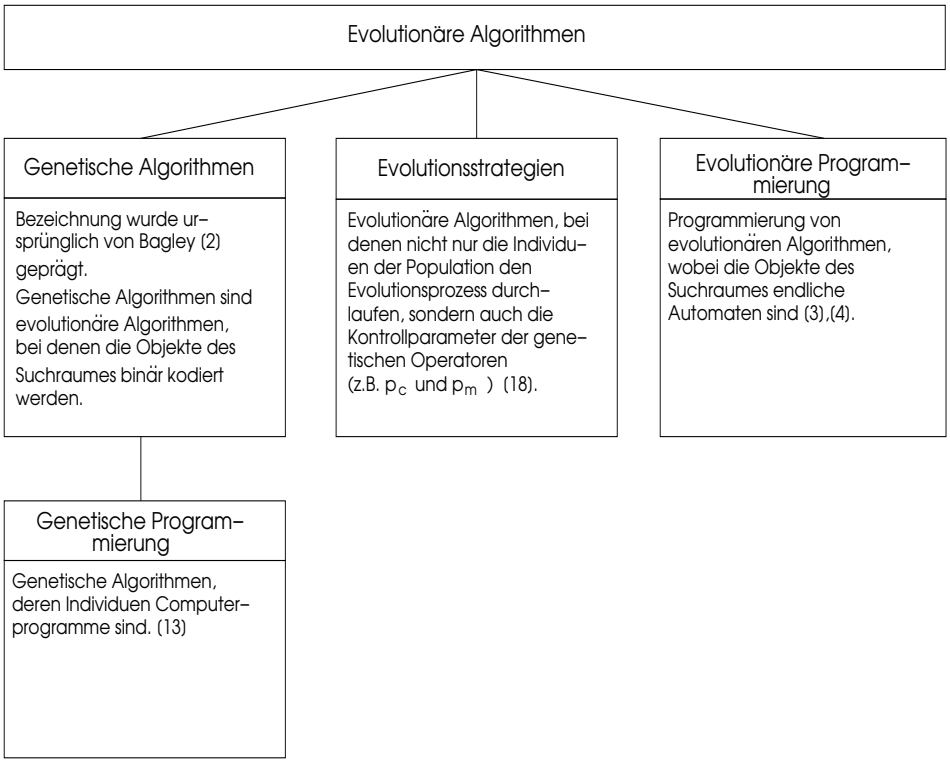

Abbildung 1.2.: Übersicht zu evolutionären Algorithmen

Kapitel 3 wird sich mit der numerischen Verifikation der beiden Sätze beschäftigen. Verschiedene Beziehungen zwischen den beiden Sätzen und Einflussfaktoren auf den genetischen Algorithmus werden untersucht und erklärt. Weiterhin wird die Mutationswahrscheinlichkeit für einen genetischen Algorithmus optimiert. Ein Optimierungsproblem dieser Art kommt sehr häufig in der Praxis vor. Die Leistung eines genetischen Algorithmus ist sehr stark von der Wahl der Mutationswahrscheinlichkeit abhängig. Ist diese Wahrscheinlichkeit ungünstig gewählt, kann es geschehen, dass keine sinnvolle Lösung für das untersuchte Problem gefunden wird.

Eine der unzähligen Anwendungsmöglichkeiten von evolutionären Algorithmen wird in Kapitel 4 präsentiert. Hier wird ein Projekt vorgestellt, welches einen evolutionären Algorithmus nutzt. Gleichzeitig demonstriert dieser Anwendungsfall die erstaunliche Leistungsfähigkeit dieser stochastischen Verfahren.

2. Der Schemensatz von J. Holland

Im Kapitel 1 wurde ein grober Überblick über die Arbeitsweise von genetischen Algorithmen gegeben. Zu Beginn dieses Kapitels sollen einige Grundbausteine eines solchen Algorithmus, also einige genetische Operatoren, konkretisiert und mathematisch formuliert werden. Das Ziel ist die Beschreibung der genetischen Operatoren als mathematische Operatoren. Durch die Konkretisierung der genetischen Operatoren erhält man einen speziellen genetischen Algorithmus, welcher in dieser Arbeit „einfacher genetischer Algorithmus“ genannt und in Abschnitt 2.1 beschrieben wird.

Um das Verhalten von genetischen Algorithmen besser zu verstehen und genauere Aussagen darüber treffen zu können, werden verschiedene Modellierungsansätze für genetische Algorithmen verfolgt. Zum Beispiel modelliert Vose [21] die Population, auf der die genetischen Operatoren wirken, als dynamisches System. Andere Ansätze nutzen Markov-Prozesse oder Methoden der statistischen Mechanik. Die hier vorliegende Arbeit beschäftigt sich jedoch mit der Schematheorie, welche Holland [11] entwickelt hat, um ein Verständnis für die Wirkungsweise von genetischen Algorithmen zu erhalten. Mit dem sogenannten Schemensatz zeigt Holland, dass sich bestimmte Klassen von Individuen mit wünschenswerten Eigenschaften exponentiell schnell in der Population ausbreiten und versucht damit, eine Erklärung für das in der Praxis bewährte schnelle Finden einer hinreichend guten Lösung zu geben. In Abschnitt 2.2 wird die Schematheorie vorgestellt, mit der die Demonstration des besagten Schemensatzes erleichtert wird. Die Herleitung des Schemensatzes von Holland befindet sich in Abschnitt 2.3. In Abschnitt 2.4 wird anschließend der Schemensatz modifiziert, um genauere Aussagen über die Entwicklung der erwähnten Klassen von Individuen zu erhalten.

2.1. Der einfache genetische Algorithmus

Genetische Operatoren wurden bereits in Abschnitt 1.1 vorgestellt. Sie modellieren die der Evolutionstheorie entnommenen Evolutionsfaktoren. Genetische Operatoren können beliebig variiert und an das zu untersuchende Problem angepasst werden. Daher ist es notwendig, sich vor Beginn der Untersuchungen auf ein konkretes Modell eines genetischen Algorithmus zu einigen, welches auf das Wesentliche reduziert ist. Die Schwerpunkte werden in dieser Arbeit auf die proportionale Fitnesszuweisung, die Selektion der am besten an die Umwelt angepassten Individuen ($\hat{=}$fitnessproportionale Selektion), die einfachste Form des Kreuztau-

sches (≙1-Punkt Kreuztausch) und die Mutation auf binären Zeichenketten (≙bitweise Mutation) gesetzt.

Definition 2.1 *Ein einfacher genetischer Algorithmus ist ein genetischer Algorithmus, bei dem die genetischen Operatoren proportionale Fitnesszuweisung, fitnessproportionale Selektion, 1-Punkt Kreuztausch und bitweise Mutation implementiert sind und diese in der genannten Reihenfolge auf die Population angewendet werden. Die Folgepopulation besteht ausschließlich aus den transformierten oder nicht transformierten Nachkommen.*

In dieser Arbeit wird der einfache genetische Algorithmus nur für Optimierungsprobleme genutzt, welchen eine Maximierungsaufgabe einer reellwertigen nichtnegativen Zielfunktion z zu Grunde liegt. Liegt eine Minimierungsaufgabe vor, so transformiert man diese durch Negierung der Zielfunktion in eine Maximierungsaufgabe. Besitzt eine Zielfunktion $\tilde{z}$ auch negative Werte, so betrachtet man

$$z(x) = \begin{cases} \tilde{z}(x) + K & : \text{ für } x \text{ mit } \tilde{z}(x) > -K \\ 0 & : \text{ sonst} \end{cases} \qquad K \in \mathbb{R}^+$$

als neue Zielfunktion, wobei K so groß sein sollte, dass mindestens ein Zielfunktionswert von z positiv ist. Ist die Zielfunktion $\tilde{z}$ nicht reellwertig, so muss auch hier eine Ersatzzielfunktion betrachtet werden, die reellwertig ist und deren Maxima mit denen von $\tilde{z}$ übereinstimmen.

Die Länge der Chromosomen wird mit l bezeichnet. Die Populationsgröße soll gerade sein, um die Durchführbarkeit des 1-Punkt Kreuztausches zwischen jeweils zwei Individuen zu gewährleisten. Die Populationsgröße wird mit $2n$ ($n \in \mathbb{N}$) bezeichnet. Weiterhin werden die Individuen nicht nur mit den Elementen des Suchraumes, welche sie repräsentieren, sondern auch mit ihren Chromosomen identifiziert. Die Chromosomenschreibweise für ein Individuum x ist $x = (x^1, x^2, \ldots, x^l)$.

2.1.1. Das mathematische Modell

Um den einfachen genetischen Algorithmus von dem mathematischen Verfahren, welches dem Algorithmus zu Grunde liegt und hier formuliert werden soll, zu unterscheiden, wird dem Verfahren der Name „genetisches Verfahren“ gegeben. Ein einfacher genetischer Algorithmus soll also eine Implementierung des genetischen Verfahrens sein. An dieser Stelle wird noch einmal daran erinnert, dass es sich um ein stochastisches Verfahren handelt. In der recherchierten Literatur befinden sich keine oder keine, für die Zwecke dieser Arbeit, brauchbaren mathematischen Beschreibungen eines genetischen Algorithmus. Daher werden im Folgenden mathematische Operatoren eingeführt, welche als Grundlage für die genetischen Operatoren des einfachen genetischen Algorithmus dienen können.

Das genetische Verfahren konstruiert, ausgehend von einer Population P_0 eine Folge von Populationen $\{P_t\}_{t\geq 1}$, also Mengen von Elementen aus $\mathfrak{A}$,

in denen ein Element mehrfach auftreten kann. Die Folgepopulation wird erzeugt, indem die einzelnen mathematischen Operatoren nacheinander auf die Population P_t angewendet werden.

Eine Population ist, wie bereits erwähnt wurde, eine ungeordnete $2n$-elementige Menge mit Elementen aus dem Suchraum $\mathfrak{A}$. Um die Population zur Generation t mathematisch zu formulieren, wird zunächst $\mathbb{P}$ als die Menge aller möglichen Populationen der Größe $2n$ definiert:

$$\mathbb{P} = \hat{\mathfrak{A}}^{2n}.$$

Dabei ist $\hat{\mathfrak{A}}^{2n}$ die Faktorgruppe oder die Menge der Äquivalenzklassen zu der Äquivalenzrelation $\sim$:

$$\hat{\mathfrak{A}}^{2n} = \mathfrak{A}^{2n} \Big/ \sim .$$

Für die Äquivalenzrelation gilt:

$$(a_1, \ldots, a_{2n}) \sim (a'_1, \ldots, a'_{2n}) \quad \Leftrightarrow \quad \exists \pi \in S_{2n} : \; a'_k = a_{\pi(k)} \quad \forall k \in \{1, \ldots, 2n\},$$

wobei S_{2n} die symmetrische Gruppe der Ordnung $2n$ ist. Nun kann eine Funktion $P : \mathbb{N}_0 \longrightarrow \mathbb{P}$ definiert werden, die jedem Zeitschritt t eine Population aus $\mathbb{P}$ zuordnet:

$$P(t) =: P_t.$$

Nun sind $\mathbb{P}$ und P_t genau definiert und können zur Beschreibung der genetischen Operatoren genutzt werden.

Der Operator der proportionalen Fitnesszuweisung wird folgendermaßen definiert:

Definition 2.2 *Die proportionale Fitnesszuweisung ist eine Abbildung F von dem Suchraum $\mathfrak{A}$ in die Menge der positiven reellen Zahlen, die folgendermaßen beschrieben ist:*

$$\begin{array}{rcl} F & : & \mathfrak{A} \longrightarrow \mathbb{R}_0^+ \\ F(x) & = & \begin{cases} a\, z(x) & : \quad x \in D(z) \\ 0 & : \quad \textit{sonst.} \end{cases} \qquad a > 0 \end{array}$$

Der Wert $F(x)$ heißt **Fitnesswert** des Elementes $x \in \mathfrak{A}$. Die proportionale Fitnesszuweisung ordnet jedem Element aus $\mathfrak{A}$ einen Fitnesswert zu, welcher gleich dem Zielfunktionswert von x multipliziert mit einem Proportionalitätsfaktor a ist. Der Fitnesswert eines Elementes ist also proportional zu seinem Zielfunktionswert. Durch die Beschränkung des Optimierungsproblems auf Maximierungsprobleme und die Definition von F ist gesichert, dass jedes Element einen nichtnegativen Fitnesswert besitzt. Es ist zu beachten, dass der Fitnesswert unabhängig von der Generation t ist.

Bezeichnung 2.3 *Der durchschnittliche Fitnesswert der Population P_t wird mit $\bar{F}_t$ bezeichnet:*

$$\bar{F}_t = \frac{1}{2n} \sum_{x \in P_t} F(x).$$

Definition 2.4 *Eine Funktion*

$$f_t : \mathfrak{P}(P_t) \longrightarrow \mathbb{R}[0,1]$$

heißt Fitnessfunktion zu einer Abbildung $F : \mathfrak{A} \longrightarrow \mathbb{R}_0^+$, wenn gilt:

$$(a) \qquad f_t(\{x\}) := f_t(x) = \begin{cases} \frac{1}{2n} & : \text{ wenn } F(y) = 0 \ \forall y \in P_t \\ \frac{F(x)}{\sum\limits_{y \in P_t} F(y)} & : \text{ wenn } \exists y \in P_t : F(y) > 0 \end{cases}$$

$$(b) \qquad f_t\left(\bigcup_{i=1}^{2^{2n}} A_i\right) = \sum_{i=1}^{2^{2n}} f_t(A_i) \quad \text{für paarweise disjunkte } A_i \in \mathfrak{P}(P_t).$$

Der Wert $f_t(x)$ wird **relative Fitness** oder auch **relativer Fitnesswert** von x genannt und ist von P_t abhängig.

Für die mathematische Beschreibung der fitnessproportionalen Selektion sind mehrere Schritte notwendig. Zum einen wird eine Bijektion $n_t^1 : P_t \longrightarrow \{1, 2, \ldots, 2n\} =: \Omega$ definiert, die dazu dient, die Individuen von P_t durchzunummerieren, und zum anderen wird ein Wahrscheinlichkeitsraum konstruiert.

Hilfssatz 2.1 *Das Tupel $(\Omega, \mathfrak{P}(\Omega), f_t)$ beschreibt einen Wahrscheinlichkeitsraum.*

Beweis: Damit $(\Omega, \mathfrak{P}(\Omega), f_t)$ einen Wahrscheinlichkeitsraum beschreibt, muss Folgendes gezeigt werden:

a) Ω ist eine nicht leere Menge.

b) $\mathfrak{P}(\Omega)$ ist eine σ-Algebra über Ω.

c) f_t ist ein Wahrscheinlichkeitsmaß.

zu a) $\Omega = \{1, 2, \ldots, 2n\}$ ist eine nicht leere Menge.
zu b) $\mathfrak{P}(\Omega)$ erfüllt die Bedingungen für eine σ-Algebra über Ω nach [22]. Der Beweis hierfür wird an dieser Stelle nicht durchgeführt.
zu c) aus [22]:
„Ist Ω eine Ergebnismenge und ist $\mathfrak{A}$ eine σ-Algebra von Ereignissen über Ω, so heißt eine Abbildung $P : \mathfrak{A} \longrightarrow \mathbb{R}$ ein *Wahrscheinlichkeitsmaß*, wenn gilt: (i) $P(A) \geq 0$ für alle $A \in \mathfrak{A}$, (ii) $P(\Omega) = 1$, (iii) $P\left(\bigcup\limits_{i=1}^{\infty} A_i\right) = \sum\limits_{i=1}^{\infty} P(A_i)$ für paarweise unvereinbare Ereignisse $A_i \in \mathfrak{A}$.“
Es muss gezeigt werden, dass f_t die Bedingungen (i) bis (iii) erfüllt.

zu (i) Nach Definition von $f_t : \mathfrak{P}(P_t) \longrightarrow \mathbb{R}[0,1]$ ist (i) erfüllt.
zu (iii) Die Bedingung (iii) ist ebenfalls durch die Definition erfüllt.
zu (ii) Die Abbildung n_t^1 ist eine Bijektion $n_t^1 : P_t \longrightarrow \Omega$. Somit kann geschrieben werden:

$$f_t(\Omega) = f_t(P_t) = f_t\left(\bigcup_{x \in P_t} x\right) \overset{(iii)}{=} \sum_{x \in P_t} f_t(x).$$

Nun sind zwei Fälle auf Grund der Definition von f_t zu unterscheiden:
1. Fall: $F(x) = 0 \quad \forall x \in P_t$, dann ist

$$\sum_{x \in P_t} f_t(x) = \sum_{x \in P_t} \frac{1}{2n} = 2n\frac{1}{2n} = 1, \quad \text{da } |P_t| = 2n.$$

2. Fall: $\exists x \in P_t : F(x) > 0$, dann ist

$$\sum_{x \in P_t} f_t(x) = \sum_{x \in P_t} \frac{F(x)}{\sum\limits_{y \in P_t} F(y)} = \frac{\sum\limits_{x \in P_t} F(x)}{\sum\limits_{y \in P_t} F(y)} = 1.$$

Somit ist (ii) erfüllt und der Hilfssatz bewiesen. □

Mit Hilfe der Bijektion n_t^1 kann eine Zufallsgröße $X_t : \Omega \longrightarrow \Omega$ über dem Wahrscheinlichkeitsraum $(\Omega, \mathfrak{P}(\Omega), f_t)$ definiert werden:

$$\begin{aligned} X_t(\omega) &= n_t^1(x), \text{ wenn das Individuum } x \text{ ausgewählt wurde} \\ &= n_t^1(x) \text{ mit der Wahrscheinlichkeit } f_t(x), x \in P_t, \omega \in \Omega. \end{aligned}$$

Das Ergebnis eines zufälligen Experimentes ist ω. Wird das Zufallsexperiment für X_t genau $2n$-mal wiederholt, so bezeichnet $X_t(\omega_i)$, $\omega_i \in \Omega$, das Ereignis, welches im i-ten Versuch eingetreten ist. Die Menge der Versuchsergebnisse wird mit S_t bezeichnet:

$$S_t = \{X_t(\omega_1), X_t(\omega_2), \ldots, X_t(\omega_{2n})\}.$$

Da n_t^1 eine Bijektion zwischen P_t und Ω definiert, beschreibt S_t eine $2n$-elementige Menge von Individuen aus P_t, in der ein Individuum auch mehrfach auftreten kann. Nun ist es möglich den Selektionsoperator zu definieren:

Definition 2.5 *Die fitnessproportionale Selektion O_S mit der Fitnessfunktion f_t ist eine Abbildung*

$$\begin{aligned} O_S &: \mathbb{P} \longrightarrow \mathbb{P} \qquad \textit{mit} \\ O_S(P_t) &= S_t. \end{aligned}$$

Die Elemente von $O_S(P_t)$ werden ebenfalls als Individuen bezeichnet. In Worten ausgedrückt ist $O_S(P_t)$ eine Menge von Individuen aus P_t, wobei

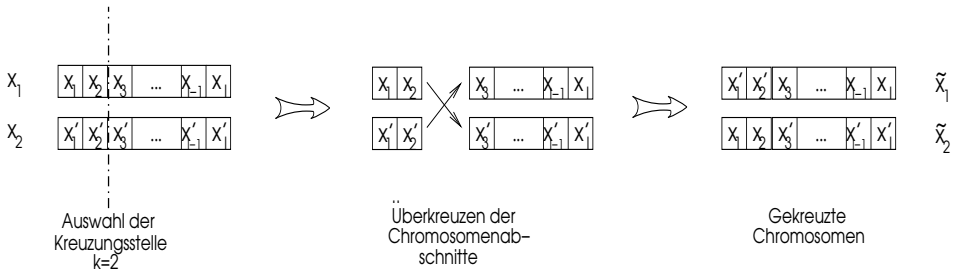

Abbildung 2.1.: Beispiel für einen Kreuztausch zwischen x_1 und x_2 mit der Kreuzungsstelle $k = 2$.

die Wahrscheinlichkeit, dass sich ein Individuum $x \in P_t$ auch in der Menge $O_S(P_t)$ befindet, $f_t(x)$ ist.

Ein Kreuztausch zwischen zwei Individuen aus $O_S(P_t)$ findet mit einer Wahrscheinlichkeit von p_c statt. Zwei Individuen x_1, x_2 werden gekreuzt, wenn ein Kreuztausch zwischen ihnen stattfindet. Dieser erfolgt, indem eine Kreuzungsstelle k in den Chromosomen der beiden Individuen ausgewählt wird und zwei neue Elemente gebildet werden, von denen eines aus den ersten k Allelen von x_1 und den letzten $(l-k)$ Allelen von x_2 besteht. Das andere Element besteht aus den ersten k Allelen von x_2 und den letzten $(l-k)$ Allelen von x_1. Abbildung 2.1 veranschaulicht den Vorgang.

Nun folgt die mathematische Beschreibung des Kreuztausches. Die Definition des Kreuztauschoperators wird nicht speziell für $O_S(P_t)$ hergeleitet, sondern für eine beliebige $2n$-elementige Menge $M \in \mathbb{P}$, deren Elemente in $\mathfrak{A}$ liegen, also mit der Chromosomenschreibweise dargestellt werden können.

Für die Anordnung der Elemente von M zu n Paaren wird die Bijektion $n_2 : M \longrightarrow \Omega$ definiert. Dann beschreibt die Menge

$$C = \big\{\{x_1, x_2\}, \{x_3, x_4\}, \ldots, \{x_{2n-1}, x_{2n}\}\big\} \qquad \text{mit } x = x_{n_2(x)} \quad \forall x \in M$$

eine Menge von n Paaren der Elemente von M. Dabei tritt jedes Element von M nur einmal auf.

Sei Ω_1 eine Ergebnismenge und $f_c : \mathfrak{P}(\Omega_1) \longrightarrow \mathbb{R}$ ein Wahrscheinlichkeitsmaß, welches die Kolmogoroffschen Axiome erfüllt und für welches gilt:

$$f_c(k) = \begin{cases} 1 - p_c & : \quad k = 0 \\ p_c \frac{1}{l-1} & : \quad k \neq 0 \end{cases} \qquad k \in \Omega_1.$$

Das Tupel $(\Omega_1, \mathfrak{P}(\Omega_1), f_c)$ bildet ebenfalls einen Wahrscheinlichkeitsraum. Eine Zufallsgröße $Y : \Omega_1 \longrightarrow \{0, 1, \ldots, l-1\}$ wird über diesen Wahrscheinlichkeitsraum definiert mit:

$$\begin{aligned} Y(\omega) &= \begin{cases} 0 & : \text{ es findet kein Kreuztausch statt} \\ k & : \text{ es findet ein Kreuztausch statt und die Kreuzungsstelle ist } k \in \{1, \ldots, l-1\} \end{cases} \\ &= k \text{ mit Wahrscheinlichkeit } f_c(k), \omega \in \Omega_1, k \in \{0, 1, \ldots, l-1\}. \end{aligned}$$

Das Wahrscheinlichkeitsmaß f_c wurde eingeführt, um die Wahrscheinlichkeit, dass zwei Elemente an der Kreuzungsstelle k gekreuzt werden, beschreiben zu können. Die Kreuzungsstelle kann zwischen 1 und $(l-1)$ variieren. Es wurde ein zusätzliches Ereignis $k = 0$ eingeführt, welches dasjenige Ereignis symbolisiert, bei dem kein Kreuztausch zwischen zwei Elementen stattfindet. Da die Kreuztauschwahrscheinlichkeit p_c beträgt, ist $f_c(0) = 1 - p_c$ die Wahrscheinlichkeit, dass zwei Elemente nicht gekreuzt werden. Findet ein Kreuztausch statt, so ist $f_c(k) = p_c \frac{1}{l-1}$ die Wahrscheinlichkeit, dass die Kreuzungsstelle an der Stelle k gewählt wurde. Damit ist gesichert, dass die Kreuzungsstelle gleichmäßig zwischen 1 und $(l-1)$ verteilt ist.

Ein Zufallsexperiment für Y wird n-mal (für alle n Paare) wiederholt. Das Ereignis im i-ten Versuch ist $Y(\omega_i)$.

Nun wird eine Abbildung κ definiert:

$$\begin{aligned} \kappa &: C \times \{0, 1, \ldots, l-1\} \longrightarrow \mathfrak{A} \times \mathfrak{A} \\ \kappa(\{x_{2i-1}, x_{2i}\}, k) &= \begin{cases} \{x_{2i-1}, x_{2i}\} & : \quad k = 0 \\ \{\tilde{x}_{2i-1}, \tilde{x}_{2i}\} & : \quad k \neq 0 \end{cases} \qquad i = 1, \ldots, n. \end{aligned}$$

Seien x_{2i-1} und x_{2i} in Chromosomenschreibweise

$$\begin{aligned} x_{2i-1} &= (x_{2i-1}^1, \ldots, x_{2i-1}^l) \\ x_{2i} &= (x_{2i}^1, \ldots, x_{2i}^l), \end{aligned}$$

dann ist

$$\begin{aligned} \tilde{x}_{2i-1} &= (x_{2i-1}^1, \ldots, x_{2i-1}^k, x_{2i}^{k+1}, \ldots, x_{2i}^l) \\ \tilde{x}_{2i} &= (x_{2i}^1, \ldots, x_{2i}^k, x_{2i-1}^{k+1}, \ldots, x_{2i-1}^l). \end{aligned}$$

Die Menge

$$K' = \left\{\kappa(\{x_1, x_2\}, Y(\omega_1)), \ldots, \kappa(\{x_{2n-1}, x_{2n}\}, Y(\omega_n))\right\}$$

ist eine Menge von Paaren aus Elementen von $\mathfrak{A}$. Ein Paar kann aus zwei Elementen von M bestehen (wenn kein Kreuztausch zwischen ihnen stattfand) oder aus zwei Elementen von $\mathfrak{A}$, die beim Kreuztausch zwischen zwei Elementen aus M entstanden sind. Nun werden die Paare von K' vereinigt, damit man mit einer Menge von Elementen aus $\mathfrak{A}$ arbeiten kann:

$$\begin{aligned} K &= \left\{\kappa(\{x_1, x_2\}, Y(\omega_1)) \cup \ldots \cup \kappa(\{x_{2n-1}, x_{2n}\}, Y(\omega_n))\right\} \\ &= \bigcup_{i=1}^{n} \kappa(\{x_{2i-1}, x_{2i}\}, Y(\omega_i)). \end{aligned}$$

Mit der Hilfe von K kann der Kreuztauschoperator definiert werden:

Definition 2.6 *Der 1-Punkt Kreuztauschoperator $\tilde{O}_K$ ist eine Abbildung*

$$\begin{aligned} \tilde{O}_K &: \mathbb{P} \longrightarrow \mathbb{P} \qquad \textit{mit} \\ \tilde{O}_K(M) &= K. \end{aligned}$$

Wird $\tilde{O}_K$ auf eine Menge M angewendet, so werden je zwei Elemente von M mit der Wahrscheinlichkeit p_c gekreuzt. Findet ein Kreuztausch zwischen zwei Elementen statt, so werden sie an einer gleichmäßig im Chromosom verteilten Stelle gekreuzt. Das Resultat des 1-Punkt Kreuztauschoperator ist eine Menge, die aus gekreuzten oder ungekreuzten Elementen von M besteht.

Nun soll der Mutationsoperator formuliert werden. Dieser wird ebenfalls allgemein für eine $2n$-elementige Menge $M \in \mathbb{P}$ mit Elementen aus $\mathfrak{A}$ definiert. Die Elemente von M müssen in Chromosomenschreibweise darstellbar sein. Jedes Allel eines Chromosoms wird mit der Mutationswahrscheinlichkeit p_m mutiert. Findet eine Mutation statt, so wird eine „1“ durch eine „0“ ersetzt und umgekehrt.

Zu Beginn werden die Elemente von M mit Hilfe der Bijektion $n_3 : M \longrightarrow \Omega$ nummeriert. Es wird der Ergebnisraum Ω_2 und das Wahrscheinlichkeitsmaß $f_m : \mathfrak{P}(\Omega_2) \longrightarrow \mathbb{R}$ eingeführt. Für f_m soll gelten:

$$f_m(q) = \begin{cases} p_m & : \quad q = 1 \\ 1 - p_m & : \quad q = 0. \end{cases}$$

Damit ist das Tupel $(\Omega_2, \mathfrak{P}(\Omega_2), f_m)$ ein Wahrscheinlichkeitsraum über dem die Zufallsgröße $Z : \Omega_2 \longrightarrow \{0, 1\}$ definiert wird:

$$\begin{aligned} Z(\omega) &= \begin{cases} 0 & : \text{ es findet keine Mutation statt} \\ 1 & : \text{ es findet eine Mutation statt} \end{cases} \\ &= \begin{cases} 0 & : \text{ mit der Wahrscheinlicheit } f_m(0) \\ 1 & : \text{ mit der Wahrscheinlicheit } f_m(1) \end{cases} \qquad \omega \in \Omega_2. \end{aligned}$$

Die Zufallsgröße Z beschreibt das Ereignis, ob eine Mutation an einem Allel stattfindet oder nicht. Das Zufallsexperiment für Z wird für jedes Element von Ω genau l-mal durchgeführt. Das Ereignis des j-ten Versuches für das i-te Element ist $Z^i(\omega_j)$.

Es wird eine Abbildung m_j definiert, die einem Element dasjenige Chromosom zuordnet, welches bei Stattfinden der Mutation dieses Elementes am j-ten Allel entsteht:

$$\begin{aligned} m_j &: \mathfrak{A} \times \{0, 1\} \longrightarrow \mathfrak{A} \\ m_j(x, q) &= \begin{cases} (x^1, \ldots, x^j, \ldots, x^l) & : \quad q = 0 \\ (x^1, \ldots, x^{j-1}, 1 - x^j, x^{j+1}, \ldots, x^l) & : \quad q = 1 \end{cases} \end{aligned}$$

Nun wird die Abbildung m eingeführt, welche einem Element dasjenige Element zuordnet, welches nach l Zufallsexperimenten für Z entsteht:

$$\begin{aligned} m &: \mathfrak{A} \longrightarrow \mathfrak{A} \\ m(x_i) &= m_l\Big(\cdots m_2\big(m_1(x_i, Z^i(\omega_1)), Z^i(\omega_2)\big) \cdots, Z^i(\omega_l)\Big) \end{aligned}$$

Damit kann nun die Menge M' als Menge der Elemente aus $\mathfrak{A}$, die nach dem Stattfinden oder Nicht-Stattfinden von Mutationen an den Allelen der Chromosomen von M entstanden ist beschrieben werden:

$$M' = \{m(x_1), m(x_2), \ldots, m(x_{2n})\}$$

Mit der Hilfe von M' kann nun der Mutationsoperator definiert werden:

Definition 2.7 *Der bitweise Mutationsoperator O_M ist eine Abbildung*

$$\begin{aligned} O_M &: \mathbb{P} \longrightarrow \mathbb{P} \\ O_M(M) &= M'. \end{aligned}$$

Nach der Definition der Operatoren F, O_S, $\tilde{O}_K$ und O_M kann nun das genetische Verfahren beschrieben werden: Ist eine Anfangspopulation P_0 gegeben so erzeugt es eine Folge von Populationen $\{P_t\}_{t\geq 1}$ mit folgender Iterationsvorschrift:

$$P_{t+1} = O_M(\tilde{O}_K(O_S(P_t))) \qquad t \in \mathbb{N}_0.$$

Zu Beginn einer Iteration werden also die Individuen von P_t selektiert und auf diese der Kreuztauschoperator angewendet. Auf die resultierenden Elemente wird der Mutationsoperator angewendet.

Mit den in diesem Abschnitt eingeführten Bezeichnungen und Operatoren ist es in späteren Abschnitten dieses Kapitels möglich, den Schemensatz von Holland herzuleiten und zu modifizieren.

2.1.2. Anmerkungen zum einfachen genetischen Algorihmus

Die Werte für die Wahrscheinlichkeiten p_m und p_c werden bei dem einfachen genetischen Algorithmus konstant gesetzt, sind also nicht von der Generation t abhängig. In der Praxis werden häufig p_c nahe 1 und p_m nahe 0 gewählt, da für diese Belegung gute Ergebnisse erzielt wurden.

Beim einfachen genetischen Algorithmus werden die genetischen Operatoren proportionale Fitnesszuweisung, fitnessproportionale Selektion, 1-Punkt Kreuztausch und bitweise Mutation auf die Population P_t solange angewendet, bis ein bestimmtes Abbruchkriterium erfüllt ist.

Möchte man die Wahrscheinlichkeit untersuchen, dass zwei Individuen aus P_t miteinander gekreuzt werden, so benötigt man die Wahrscheinlichkeit, mit der diese zwei Individuen durch die Selektion ausgewählt und zusammen als Paar angeordnet werden. Hierfür wird folgender Satz hergeleitet:

Hilfssatz 2.2 *Bildet man n Paare aus $2n$ Elementen, so hat man hierfür*

$$\begin{aligned} T(2n) &= (2n-1)T(2n-2) \\ T(0) &= 1 \end{aligned}$$

verschiedene Möglichkeiten.

Ist $n = 2$, so gibt es drei verschiedene Möglichkeiten, vier Elemente a, b, c und d zu zwei Paaren zu ordnen:

1.Möglichkeit : (a,b) und (c,d)
2.Möglichkeit : (a,c) und (b,d)
3.Möglichkeit : (a,d) und (b,c)

Beweis zu 2.2 Der Beweis erfolgt durch Induktion.
Induktionsanfang: n=1

$T(2) = 1$ ist korrekt, da zwei Elemente nur auf eine Art ein Paar bilden können.
Induktionsannahme: $T(2n) = (2n-1)T(2n-2)$ sei richtig für $n = k$.
Induktionsbeweis für $k+1$:
Betrachtet wird ein beliebiges Element x_0 aus den $2(k+1)$ Elementen. So gibt es für x_0 insgesamt $2(k+1)-1$ mögliche Partner, mit denen es als Paar angeordnet werden kann. Es wird ein Paar mit x_0 gebildet. So bleiben $2(k+1)-2 = 2k$ Elemente, die zu k Paaren angeordnet werden sollen. Für sie gibt es $T(2k)$ mögliche Paarbildungen. Also ist $T\big(2(k+1)\big) = (2(k+1)-1)T(2k)$.

□

Es interessierte die Wahrscheinlichkeit, mit der zwei Individuen x_1, x_2 aus P_t als Paar bei der Anwendung des 1-Punkt Kreuztausches angeordnet werden. Der Kreuztauschoperator wird auf die Individuen von $O_S(P_t)$ angewendet, also müssen x_1 und x_2 in $O_S(P_t)$ sein. Ein Element $x \in P_t$ befindet sich mit einer Wahrscheinlichkeit von $f_t(x)$ in $O_S(P_t)$. Daher ist für $x_1, x_2 \in P_t$ die Wahrscheinlichkeit

$$\begin{aligned} P^*(x_1 \text{ und } x_2 \text{ bilden ein Paar}) &= \frac{T(2n-2)}{T(2n)} f_t(x_1) f_t(x_2) \\ &= \frac{T(2n-2)}{(2n-1)T(2n-2)} f_t(x_1) f_t(x_2) \\ &= \frac{1}{2n-1} f_t(x_1) f_t(x_2). \end{aligned} \tag{2.1}$$

Die Wahrscheinlichkeit P^* ist das Produkt der Wahrscheinlichkeiten, dass x_1 und x_2 selektiert und als Paar angeordnet werden. Die Anzahl an Möglichkeiten, bei denen x_1 und x_2 ein Paar sind ist $T(2n-2)$ und $T(2n)$ ist die Gesamtanzahl an möglichen Paarbildungen mit $2n$ Individuen.

Um die Untersuchungen zu vereinfachen, wird statt (2.1) die Wahrscheinlichkeit

$$P(x_1 \text{ und } x_2 \text{ bilden ein Paar}) := f_t(x_1) f_t(x_2) \tag{2.2}$$

betrachtet. Dies würde bedeuten, dass die Selektion implizit in der Paarbildung beim Kreuztausch enthalten ist. Mit anderen Worten werden die Individuen der Population in Paare angeordnet, wobei die Wahrscheinlichkeit eines Individuums x als Teil eines Paares ausgewählt zu werden, $f_t(x)$ beträgt. Der Vorgang des Auswählens und anschließenden Paarbilden wird also abgekürzt.

Für den Kreuztausch wird mit (2.2) $\tilde{O}_K\big(O_S(P_t)\big)$ ersetzt durch $O_K(P_t)$. Der Operator O_K ist dabei ein Kreuztauschoperator mit impliziter Selektion. Er unterscheidet sich von $\tilde{O}_K$ in der Bildung der Menge C, die aus Paaren von Elementen der Menge, auf die der Operator angewendet wird, besteht. Die Menge C hatte die Form

$$C = \big\{\{x_1, x_2\}, \{x_3, x_4\}, \dots, \{x_{2n-1}, x_{2n}\}\big\} \qquad \text{mit } x = x_{n_2(x)} \quad \forall x \in M.$$

Bei der Anwendung von $\tilde{O}_K$ auf M wird C derart gebildet, dass jedes Element von M in C vertreten ist. Bei dem Operator O_K soll die Bildung von C durch die Zufallsgröße X_t erfolgen. Diese Zufallsgröße wurde für die Selektion über dem Wahrscheinlichkeitsraum $(\Omega, \mathfrak{P}(\Omega), f_t)$ definiert:

$$\begin{aligned} X_t(\omega) &= n_t^1(x), \text{ wenn das Individuum } x \text{ ausgewählt wurde} \\ &= n_t^1(x) \text{ mit der Wahrscheinlichkeit } f_t(x), x \in P_t, \omega \in \Omega. \end{aligned}$$

Damit kann C folgendermaßen gebildet werden:

$$C = \big\{\{X_t(\omega_1), X_t(\omega_2)\}, \{X_t(\omega_3), X_t(\omega_4)\}, \ldots, \{X_t(\omega_{2n-1}), X_t(\omega_{2n})\}\big\}.$$

Der Kreuztauschoperator O_K bildet somit die Grundlage für den Kreuztausch mit impliziter Selektion. Wird im Folgenden von „Selektion“ gesprochen, so handelt es sich dabei um ein Konstrukt, das die Menge der Individuen, die zum Kreuztausch ausgewählt werden beschreibt.

2.2. Einführung in die Schematheorie

In diesem Abschnitt wird eine kurze Einführung in die Schematheorie gegeben. Die Bezeichnungen sind an [11] angelehnt. Die Schematheorie beschreibt die Konstruktion von Schemen. Diese sind von der verwendeten Kodierung abhängig. Zunächst wird die allgemeine Konstruktion von Schemen beschrieben und im Anschluss daran die Spezialisierung auf Schemen in genetischen Algorithmen durch die binäre Kodierung.

Der Ausgangspunkt der Schematheorie ist die Suche nach einer Möglichkeit, die Elemente des Suchraumes miteinander zu vergleichen. Dabei wird vom direkten Vergleich der Zielfunktionswerte abgesehen.

Um bestimmte Eigenschaften der Elemente quantifizieren zu können, werden Detektoren definiert.

Definition 2.8 *Ein Detektor δ ist eine Abbildung des Suchraumes $\mathfrak{A}$ in den Wertevorrat V. Der Definitionsbereich ist $\mathfrak{A}$.*

$$\delta : \mathfrak{A} \longrightarrow V \qquad mit \qquad D(\delta) = \mathfrak{A}.$$

Die Menge von Detektoren wird mit Δ bezeichnet:

$$\Delta = \{\delta_i : \mathfrak{A} \longrightarrow V_i,\ i = 1, \ldots, m\} \quad m \in \mathbb{N}.$$

Dabei ist V_i der Wertevorrat des i-ten Detektors und m die Anzahl der Detektoren.

Mit Hilfe der Detektoren erhält jedes Element $x \in \mathfrak{A}$ eine Darstellung mit dem geordnetem Tupel $\big(\delta_1(x), \delta_2(x), \ldots, \delta_m(x)\big)$.

Es ist nicht immer von Interesse, den Wert eines bestimmten Detektors zu kennen. Für diesen Fall wird das Symbol „$\star$“ eingeführt. Damit

beschreibt das Tupel $(v_1, \star, \ldots, \star, v_l)$ mit $v_1 \in V_1$ und $v_l \in V_l$ die Menge aller Elemente $x \in \mathfrak{A}$, für die gilt:

$$\begin{aligned} \delta_1(x) &= v_1 \\ \delta_l(x) &= v_l \ . \end{aligned}$$

Die Werte aller anderen Detektoren $\delta_2(x), \ldots, \delta_{l-1}(x)$ sind hierbei nicht von Belang.

Definition 2.9 *Ein Schema ist ein m-Tupel aus der Menge*

$$\prod_{i=1}^{m} \{V_i \cup \{\star\}\}.$$

Ist $H = (s_1, s_2, \ldots, s_m)$, so beschreibt H die Menge aller Elemente aus $\mathfrak{A}$, für welche gilt:

$$\delta_i(x) = s_i \text{ falls } s_i \neq \star \quad \forall i = 1, \ldots, m \ .$$

Das Schema wird mit der Menge, die es beschreibt identifiziert.

Bezeichnung 2.10 *Die Menge aller Schemen wird mit Ξ bezeichnet:*

$$\Xi = \prod_{i=1}^{m} \{V_i \cup \{\star\}\}.$$

Jedes Schema unterteilt den Suchraum $\mathfrak{A}$ in zwei Teilmengen, in H und $\mathfrak{A} \backslash H$. Ein Element $x \in \mathfrak{A}$ heißt **Vertreter** eines Schemas H, falls $x \in H$. Eine Position k $(k = 1, \ldots, m)$ eines Schemas $H = (s_1, s_2, \ldots, s_m)$ heißt **fixiert**, wenn $s_k \neq \star$.

Bezeichnung 2.11 *Die Anzahl der Vertreter eines Schemas H in der Population P_t wird mit $m(H,t) = |\{P_t \cap H\}|$ bezeichnet.*

Definition 2.12 *Die durchschnittliche Fitness $F_t(H)$ eines Schemas in der Population P_t ist*

$$F_t(H) = \frac{1}{m(H,t)} \sum_{x \in P_t \cap H} F(x).$$

Definition 2.13 *Die relative Fitness $f_t(H)$ eines Schemas H ist die Summe der relativen Fitnesswerte der Vertreter des Schemas, die sich in der Population P_t befinden:*

$$f_t(H) = \sum_{x \in P_t \cap H} f_t(x).$$

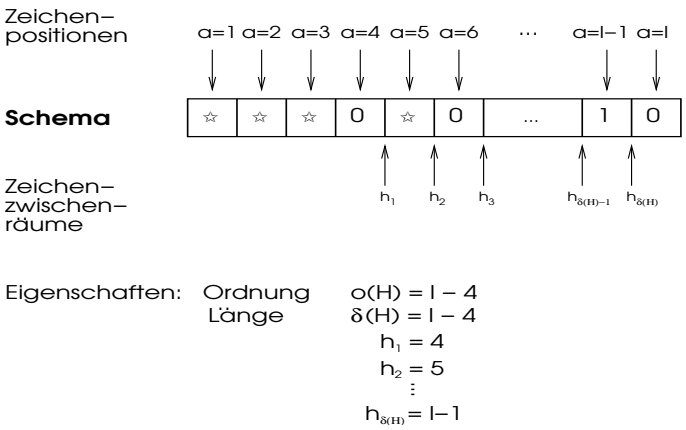

Abbildung 2.2.: Bezeichnungen an einem Schema.

Definition 2.14 *Die Ordnung $o(H)$ eines Schemas $H \in \Xi$ ist die Anzahl der fixierten Stellen von H.*

Definition 2.15 *Die Länge $\delta(H)$ eines Schemas $H \in \Xi$ ist die Differenz zwischen der ersten und der letzten fixierten Stelle von H.*

Beginnend bei der ersten fixierten Position eines Schemas H werden die Zeichenzwischenräume mit h_1 bis $h_{\delta(H)}$ bezeichnet.

Für den einfachen genetischen Algorithmus werden die Detektoren folgendermaßen spezialisiert: Die Anzahl m der Detektoren ist identisch mit der Chromosomenlänge l und der Wertevorrat jedes Detektors ist die Menge der möglichen Allele an einem Gen:

$$V_i = \{0, 1\} \quad i = 1, \ldots, l \; .$$

Weiterhin wird festgelegt, dass der i-te Detektor das Allel am i-ten Gen wiedergeben soll:

$$\delta_i(x) = \begin{cases} 0 & : \text{ das Allel am } i\text{-ten Gen von } x \text{ ist eine } 0 \\ 1 & : \text{ das Allel am } i\text{-ten Gen von } x \text{ ist eine } 1 \end{cases} \qquad x \in \mathfrak{A}.$$

Abbildung 2.2 veranschaulicht die Bezeichnungen an einem Schema für den einfachen genetischen Algorithmus.

In diesem Abschnitt wurde gezeigt, wie ein Schema mit Hilfe von Detektoren konstruiert werden kann. Ein Detektor wird dabei als Hilfsmittel benutzt, mit dem gewisse Eigenschaften der Elemente von $\mathfrak{A}$ quantifiziert werden können. Ein Schema vereinigt die Elemente zu einer Menge, die dieselben Ausprägungen bestimmter Eigenschaften besitzen. Somit ermöglichen Schemen eine Unterteilung des Suchraumes. Mit dieser Unterteilung ist es wiederum möglich, die Elemente untereinander bezüglich ihrer Ausprägung bestimmter Eigenschaften zu vergleichen.

Beim einfachen genetischen Algorithmus werden zur Konstruktion der Schemen genau so viele Detektoren benutzt, wie die Chromosomen lang

sind. Der i-te Detektor beschreibt dabei das Allel am i-ten Gen eines Chromosomes. Ein Schema teilt dann den Suchraum in zwei Mengen, wovon die eine aus Elementen besteht, deren Chromosomen an bestimmten Genen dieselben Allele besitzen.

2.3. Herleitung von Hollands Schemensatz

In diesem Abschnitt soll Hollands Schemensatz vorgestellt, hergeleitet und anschließend interpretiert werden. Betrachtet wird dabei ein einfacher genetischer Algorithmus mit der Population P_t.

2.3.1. Hollands Schemensatz

Satz 2.3 (Hollands Schemensatz) *In einem einfachen genetischen Algorithmus gilt für die erwartete Anzahl von Vertretern eines Schemas H in der Folgegeneration $(t+1)$ [11]:*

$$\begin{aligned} E(m(H,t+1)) &= E\left|\left\{P_{t+1} \cap H\right\}\right| \\ &\geq m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big)\right)(1-p_m)^{o(H)} \end{aligned} \tag{2.3}$$

Bezeichnung 2.16

$$E_H(H,t) := m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big)\right)(1-p_m)^{o(H)}$$

Im Folgenden soll der Satz 2.3 hergeleitet werden. In einem einfachen genetischen Algorithmus ist die fitnessproportionale Selektion implementiert. Bei diesem Selektionsoperator ist die Wahrscheinlichkeit eines Individuums x zum Kreuztausch ausgewählt zu werden gleich $f_t(x)$. Damit ein Schema einen Vertreter in der Folgepopulation besitzt, muss ein Vertreter von H zum Kreuztausch ausgewählt werden. Die Wahrscheinlichkeit dafür ist die Summe der relativen Fitnesswerte seiner Vertreter in P_t, also $f_t(H)$. Es werden $2n$ Individuen für den Kreuztausch ausgewählt und somit ist die erwartete Anzahl der ausgewählten Individuen, die Vertreter von H sind:

$$
\begin{aligned}
2nf_t(H) &= 2n \sum_{x \in H \cap P_t} f_t(x) \\
&= 2n \sum_{x \in H \cap P_t} \frac{F(x)}{\sum\limits_{y \in P_t} F(y)} \quad \text{wenn } \exists z \in P_t : F(z) > 0^1 \\
&= 2n \frac{\sum\limits_{x \in H \cap P_t} F(x)}{\sum\limits_{y \in P_t} F(y)} \frac{m(H,t)}{m(H,t)} \\
&= m(H,t) \frac{\frac{1}{m(H,t)} \sum\limits_{x \in H \cap P_t} F(x)}{\frac{1}{2n} \sum\limits_{y \in P_t} F(y)} \\
&= m(H,t) \frac{F_t(H)}{\bar{F}_t}. \qquad (2.4)
\end{aligned}
$$

Werden zwei Individuen gekreuzt, wovon wenigstens eines ein Vertreter von H ist, so interessiert, wie wahrscheinlich es ist, dass von den beiden aus der Kreuzung resultierenden Elemente wenigstens einer ein Vertreter von H ist. Dazu wird das komplementäre Ereignis betrachtet: Keines der aus dem Kreuztausch resultierenden Elemente ist ein Vertreter von H. Dieses Ereignis tritt ein, wenn ein Vertreter von H mit einem Individuum $x \notin H$ gekreuzt wird und die Kreuzungsstelle zwischen die fixierten Positionen von H fällt. Die Wahrscheinlichkeit hierfür ist:

$$
\begin{aligned}
P(\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } \{x_{2i-1}, x_{2i}\} \cap (\mathfrak{A} \backslash H) \neq \emptyset \text{ und} \\
k \in \{h_1, \ldots, h_{\delta(H)}\}) = p_c \frac{\delta(H)}{l-1} \big(1 - f_t(H)\big).
\end{aligned}
$$

Damit ist die Wahrscheinlichkeit, dass bei der Kreuzung zweier Individuen an der Kreuzungsstelle k wenigstens ein Vertreter von H resultiert:

$$
P\big(\kappa(\{x_{2i-1}, x_{2i}\}, k) \cap H \neq \emptyset\big) \geq 1 - p_c \frac{\delta(H)}{l-1} \big(1 - f_t(H)\big). \qquad (2.5)
$$

Man erhält eine untere Schranke in (2.5), da die tatsächliche Wahrscheinlichkeit für Schemen nach einer Kreuzung Vertreter zu haben größer ist, als der in (2.5) gegebene Term. Eine Kreuzung kann auch zwischen einem Vertreter von H und einem Individuum stattfinden, welches Vertreter eines Teilschemas ist, das bis zur Kreuzungsstelle mit H übereinstimmt. In diesem Fall würden Vertreter von H aus dem Kreuztausch resultieren, doch dieser Fall wird in (2.5) nicht beachtet.

Findet eine Mutation in einem Element statt, welches ein Vertreter von H ist, so ist das aus der Mutation resultierende Element ein Vertreter

[1] Ist $F(x) = 0$ für alle Individuen von P_t, so erhält man $2nf_t(H) = m(H,t)$. Dieser Fall ist in (2.4) für $F_t(H) = \bar{F}_t$ enthalten und braucht damit nicht weiter explizit untersucht zu werden.

von H, wenn keine Mutation an den fixierten Stellen von H stattfand. Die Wahrscheinlichkeit, dass $o(H)$-mal keine Mutation in dem Chromosom des Elementes stattfindet ist

$$\begin{aligned} P(o(H)\text{-mal keine Mutation}) &= P\left(\sum_{i=1}^{l} Z(\omega_i) \leq l - o(H)\right) \\ &= \binom{l}{o(H)}(1-p_m)^{o(H)}, \end{aligned}$$

da es $\binom{l}{o(H)}$ Möglichkeiten gibt, $o(H)$ Positionen in einem Chromosom der Länge l auszuwählen. Der Ausdruck

$$\sum_{i=1}^{l} Z(\omega_i) \leq l - o(H)$$

bedeutet, dass bei der Summation der Ereignisse, die beim l-maligen Durchführen des Zufallsexperimentes für Z eintreten, die Summe nicht größer als $l - o(H)$ sein darf, es muss also wenigstens $o(H)$-mal $Z(\omega_i) = 0$ eingetreten sein. In dem Chromosom gibt es nur eine Auswahl von $o(H)$ Positionen, so dass die fixierten Positionen von H ausgewählt werden. Folglich ist die Wahrscheinlichkeit, dass keine Mutation an den fixierten Positionen auftritt[2]:

$$\begin{aligned} P(m(x) \in H) &= \binom{l}{o(H)}(1-p_m)^{o(H)} \frac{1}{\binom{l}{o(H)}} \\ &= (1-p_m)^{o(H)}. \end{aligned} \tag{2.6}$$

Ein Schema H kann in der Folgepopulation Vertreter besitzen, wenn Vertreter von H, die sich in der Population befinden, bei der Selektion ausgewählt werden und auch nach dem Anwenden des Kreuztausch- und des Mutationsoperators Vertreter von H vorhanden sind. Eine untere Schranke für die Wahrscheinlichkeit, dass ein Vertreter von H aus dem Kreuztausch eines Vertreters von H mit einem anderen Element resultiert, ist mit (2.5) gegeben. Die Wahrscheinlichkeit, dass aus der Mutation eines Vertreters von H ein Element resultiert, welches ebenfalls in H liegt, ist mit (2.6) gegeben. Gleichung (2.4) beschreibt die erwartete Anzahl von Vertretern von H, die zum Kreuztausch ausgewählt werden. Multipliziert man diese drei Terme miteinander, so erhält man eine untere Schranke für den Erwartungswert der Anzahl an Vertretern eines Schemas H in der Folgepopulation P_{t+1}:

[2] Es ist somit auch dieselbe Wahrscheinlichkeit, dass an $o(H)$ beliebig gewählten Stellen keine Mutation auftritt.

$$\begin{aligned} E(m(H,t+1)) &\geq m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1-p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)} \\ &= 2nf_t(H)\left(1-p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)} \end{aligned} \tag{2.7}$$

Somit ist Satz 2.3 bewiesen. □

2.3.2. Interpretation von Hollands Schemensatz

Hollands Schemensatz 2.3 schätzt die Anzahl der in der Folgegeneration zu erwartenden Vertreter eines Schemas nach unten ab. Die eigentliche Erkenntnis aus dem Schemensatz wird jedoch an einer schwächeren Form der in (2.3) gegebenen unteren Schranke ersichtlich. Diese schwächere Form ist gegeben durch:

$$\begin{aligned} E\big(m(H,t+1)\big) &\geq E_H(H,t) \\ &= m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1-p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)} \\ &\geq m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1-p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)}, \end{aligned}$$

da $(1-f_t(H)) \geq 0$.

In der Praxis werden p_c nahe 1 und p_m nahe 0 gewählt:

$$p_m \approx 0, \qquad p_c \approx 1.$$

Somit hängt die Vertreteranzahl eines Schemas H in der Folgepopulation im Wesentlichen von dem Verhältnis seiner durchschnittlichen Fitness zu der durchschnittlichen Populationsfitness $\frac{F_t(H)}{\bar{F}_t}$, seiner Länge $\delta(H)$ und seiner Ordnung $o(H)$ ab. Ist $F_t(H)$ größer als $\bar{F}_t$, $\frac{\delta(H)}{l-1}$ und $o(H)$ klein, dann wird die Anzahl der Vertreter von H vergrößert. Ist $F_t(H)$ kleiner oder gleich $\bar{F}_t$, so wird die Anzahl dezimiert. So vermutet man, dass kleine ($o(H)$ gering), kompakte ($\frac{\delta(H)}{l-1}$ gering) und überdurchschnittlich fitte Schemen im Laufe des Algorithmus dominieren, weil für diese Schemen der Ausdruck

$$m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1-p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)}$$

größer ist, als für große, nicht kompakte oder unterdurchschnittlich fitte Schemen.

Bleibt ein kleines und kompaktes Schema über mehrere Generationen mit einem bestimmten Betrag $c\bar{F}_t$ über der durchschnittlichen Populationsfitness $\bar{F}_t$ ($c > 0$):

$$F_t(H) = (1+c)\bar{F}_t,$$

so kann man zeigen, dass es in den Folgepopulation eine exponentiell steigende Anzahl von Vertretern besitzt. Ist das Schema H klein und kompakt so kann die destruktive Wirkung vom Kreuztausch und von der Mutation vernachlässigen werden:

$$1 - p_c\frac{\delta(H)}{l-1} \approx 1 \qquad\qquad (1-p_m)^{o(H)} \approx 1.$$

Und sofern $m(H,0) \neq 0$ sieht man:

$$\begin{aligned} E\big(m(H,t+1)\big) &\geq m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1 - p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)} \\ &\approx m(H,t)\frac{F_t(H)}{\bar{F}_t} \\ &= m(H,t)(1+c) \\ &= m(H,t-1)(1+c)^2 \\ &\vdots \\ &= m(H,0)(1+c)^t. \end{aligned}$$

Kleine, kompakte und überdurchschnittlich fit bleibende Schemen erhalten also eine exponentiell steigende Anzahl an Vertretern in der Population, da die untere Schranke für die erwartete Anzahl an Vertretern dieser Schemen mit einer Rate von $(1+c)$ exponentiell wächst. In einer endlichen Population kann natürlich kein Schema fortlaufend überdurchschnittlich fit bleiben, da es mit steigender Vertreteranzahl auch die durchschnittliche Popultionsfitness $\bar{F}_t$ erhöht. Aus diesem Grund und wegen der endlichen Populationsgröße kann somit auch nicht die Vertreteranzahl eines Schemas fortlaufend exponentiell ansteigen. Doch die Hauptaussage des Schemensatzes bleibt bestehen: In einem einfachen genetischen Algorithmus erhalten überdurchschnittlich fitte Schemen mit bestimmten Eigenschaften (klein und kompakt) über einige Generationen exponentiell steigende Vertreteranzahl in der Population. Da sich die durchschnittliche Fitness eines Schemas aus den Fitnesswerten der Individuen zusammensetzt, die Vertreter des Schemas sind, deutet ein überdurchschnittlich fittes Schema auch immer auf überdurchschnittlich fitte Vertreter in P_t hin. Steigt also die Vertreteranzahl eines Schemas, welches überdurchschnittlich fit ist, steigt auch die Anzahl an überdurchschnittlich fitten Individuen. Somit kann der Schemensatz 2.3 eine Erklärung für das schnelle Finden einer Lösung nahe am Optimum oder einer Lösung, deren Zielfunktionswert nahe am Optimalzielfunktionswert liegt durch einfache genetische Algorithmen bieten.

Der folgende Satz wird für den Beweis eines anderen Satzes benötigt.

Hilfssatz 2.4 *Ist $p_m < 1$, so ist die untere Schranke für die erwartete Anzahl von Vertretern eines Schemas in der Folgepopulation genau dann Null, wenn die relative Fitness des Schemas Null ist:*

$$E_H(H,t) = 0 \Leftrightarrow f_t(H) = 0.$$

Beweis ($\Leftarrow$): Sei $f_t(H) = 0$ dann ist

$$\begin{aligned} E_H(H,t) &= 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big)\right)(1-p_m)^{o(H)} \\ &= 0. \end{aligned}$$

Ist die relative Fitness eines Schemas Null, so ist untere Schranke für die erwartete Anzahl von Vertretern in P_{t+1} gleich Null: $f_t(H) = 0 \Rightarrow E_H(H,t) = 0$.
($\Rightarrow$): Sei $E_H(H,t) = 0$. Es wird angenommen, dass die relative Fitness eines Schemas ungleich Null ist. Mit einigen Äquivalenzumformungen wird ein Widerspruch zur Annahme gezeigt. Annahme: $f_t(H) \neq 0$

$$\begin{aligned} & E_H(H,t) = 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big)\right)(1-p_m)^{o(H)} = 0 \\ \Leftrightarrow\ & 1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big) = 0, \text{ da } f_t(H) > 0 \text{ und } p_m < 1 \\ \Leftrightarrow\ & p_c\frac{\delta(H)}{l-1}f_t(H) = p_c\frac{\delta(H)}{l-1} - 1 \\ \Leftrightarrow\ & 0 = p_c\frac{\delta(H)}{l-1}f_t(H) = p_c\frac{\delta(H)}{l-1} - 1, \end{aligned}$$

denn $\quad p_c\frac{\delta(H)}{l-1}f_t(H) \in [0,1] \quad$ und $\quad p_c\frac{\delta(H)}{l-1} - 1 \in [-1,0]$

$$\begin{aligned} & 0 = p_c\frac{\delta(H)}{l-1}f_t(H) = p_c\frac{\delta(H)}{l-1} - 1 \\ \Leftrightarrow\ & p_c\frac{\delta(H)}{l-1} = 1 \quad \text{und} \quad f_t(H) = 0 \end{aligned}$$

Das ist ein Widerspruch zur Annahme. Also folgt aus $E_H(H,t) = 0$, dass die relative Fitness des Schema ebenfalls Null ist: $E_H(H,t) \Rightarrow f_t(H) = 0$.
Mithin: $E_H(H,t) \Leftrightarrow f_t(H) = 0$. □

2.4. Modifikation von Hollands Schemensatz

2.4.1. Der modifizierte Schemensatz

In Hollands Schemensatz 2.3 wird nur der destruktive Einfluss des Kreuztausches und der Mutation auf die Vertreteranzahl eines Schemas in der Folgepopulation betrachtet. Weiterhin bietet der Schemensatz nur eine untere Schranke für eben jene Vertreteranzahl. Da jedoch für die Analyse des einfachen genetischen Algorithmus eine präzisere Angabe wünschenswert

ist, wird für die erwartete Vertreteranzahl eines Schemas in der Folgepopulation im Folgendem eine exakte Formel hergeleitet.

Bei dem einfachen genetischen Algorithmus können durch folgende Ursachen Vertreter eines Schemas H in der Folgepopulation vorhanden sein:

(1a) Aus der Kreuzung zweier Individuen, wovon wenigstens eines in H liegt, resultiert wenigstens ein Element, das ebenfalls Vertreter von H ist.

(1b) Bei der Kreuzung zweier Individuen, welche nicht Vertreter von H sind, resultiert ein Element, das in H liegt.

(2) Aus der Mutation von einem Vertreter von H resultiert ein Element, welches ebenfalls in H liegt.

(3) Aus der Mutation von einem Element, welches kein Vertreter von H ist, resultiert ein Element, welches in H liegt.

Diese vier Szenarien sollen im Folgenden betrachtet und hinsichtlich ihres Einflusses auf die Vertreteranzahl eines Schemas in der Folgepopulation analysiert werden. Es wurde nur (2) und teilweise (1a) in Hollands Schemensatz 2.3 betrachtet.

Zunächst wird der Fall betrachtet, dass einige Elemente nach dem Anwenden des Kreuztauschoperators Vertreter von H sind:

$$O_K(P_t) \cap H \neq \emptyset.$$

(1a) Es wird zuerst das Ereignis untersucht, bei dem wenistens ein Vertreter von H an der Kreuzung zwischen zwei Individuen teilnimmt. Bei dem Kreuztausch zwischen zwei Individuen wird eine Kreuzungsstelle ausgewählt. Es werden zwei Fälle unterschieden. Die Kreuzungsstelle $k \in \{1, \ldots, l-1\}$ kann außerhalb des Schemas H oder zwischen die fixierten Positionen des Schemas fallen. Formell geschrieben ist das erste Ereignis:

$$\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } k \notin \{h_1, \ldots, h_{\delta(H)}\}. \tag{2.8}$$

Die Wahrscheinlichkeit, dass zwischen zwei Individuen ein Kreuztausch stattfindet und die Kreuzungsstelle zwischen die fixierten Positionen fällt, ist:

$$P(\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } k \in \{h_1, \ldots, h_{\delta(H)}\}) = p_c \frac{\delta(H)}{l-1}. \tag{2.9}$$

Dann ist die Wahrscheinlichkeit für das Ereignis (2.8):

$$P(\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } k \notin \{h_1, \ldots, h_{\delta(H)}\}) = 1 - p_c \frac{\delta(H)}{l-1}.$$

Bei einer Kreuzung zweier Individuen, bei der wenigstens ein Vertreter von H teilnimmt, können entweder beide oder nur ein Individuum Vertreter von H sein. Die Auswahl eines Vertreters von H und eines Individuums, welches kein Vertreter von H ist, kann dabei auf zwei Arten erfolgen. Der Vertreter von H kann als erster oder als zweiter Kreuzungspartner gewählt werden. Sind beide Individuen Vertreter von H, so gibt es nur eine Auswahlmöglichkeit. Damit ist die Wahrscheinlichkeit, dass wenigstens eines von zwei zu kreuzenden Individuen ein Vertreter von H ist:

$$P(\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } \{x_{2i-1}, x_{2i}\} \cap H \neq \emptyset) = f_t(H)^2 + 2f_t(H)(1 - f_t(H)).$$

Die Anzahl der Vertreter von H nach einem Kreuztausch zwischen zwei Individuen ist abhängig davon, wieviele Vertreter von H, ob zwei oder nur einer, am Kreuztausch beteiligt waren. Werden zwei Vertreter von H miteinander gekreuzt, liegen beide, durch den Kreuztausch entstehenden Elemente in H. Ist jedoch nur ein Individuum Vertreter von H, so liegt auch nur eines von zwei resultierenden Elementen in H. Werden zwei Individuen gekreuzt, wobei wenigstens eines Element von H ist, so ist die zu erwartende Anzahl von Vertretern von H demnach:

$$E\left|\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } \{x_{2i-1}, x_{2i}\} \cap H \neq \emptyset \text{ und } k \notin \{h_1, \ldots, h_{\delta(H)}\}\right| =$$

$$\begin{aligned} &= \left(1 - p_c\frac{\delta(H)}{l-1}\right)\left(\mathbf{2}f_t(H)^2 + \mathbf{1} * 2f_t(H)(1 - f_t(H))\right) \\ &= 2f_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right). \end{aligned} \tag{2.10}$$

Damit ist Szenario (1a) vollständig untersucht.

(1b) Nun wird das Ereignis betrachtet, bei dem die Kreuzungsstelle zwischen die fixierten Positionen von H fällt. Die Wahrscheinlichkeit für dieses Ereignis wurde bereits erwähnt (2.9):

$$P(\kappa(\{x_{2i-1}, x_{2i}\}, k) \text{ mit } k \in \{h_1, \ldots, h_{\delta(H)}\}) = p_c\frac{\delta(H)}{l-1}.$$

Nach dem Kreuztausch zwischen zwei Individuen, bei dem wenigstens eines der resultierenden Elemente in H liegt und die Kreuzungsstelle zwischen die fixierten Positionen fällt, können ein oder zwei Elemente in H liegen. Das ist davon abhängig, ob zwei Vertreter von H miteinander gekreuzt werden oder ob ein Vertreter von H mit einem Individuum gekreuzt wird, welches entweder bis zur Kreuzungsstelle oder von der Kreuzungsstelle an den fixierten Positionen von H mit dessen Allelen übereinstimmt.

Definition 2.17 *Das Schema $H_k \in \Xi$ ist ein Schema, dessen fixierte Positionen und dessen Allele an diesen fixierten Positionen mit denen von H bis zur Zeichenposition $k \in \{h_1, \ldots, h_{\delta(H)}\}$ übereinstimmen.*

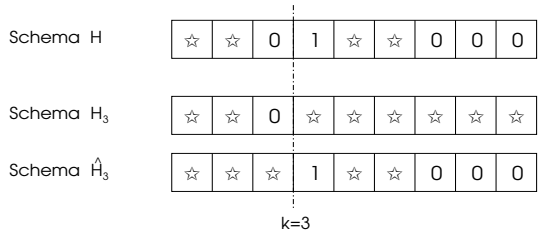

Abbildung 2.3.: Ein Beispiel für die Schemen H, H_k und $\hat{H}_k$ mit $k = 3$.

Abgesehen von diesen erwähnten fixierten Positionen, sind keine weiteren Positionen von H_k fixiert. Ist $H = (s_1, \ldots, s_k, \ldots, s_l)$ dann ist also $H_k = (s_1, \ldots, s_k, \star, \ldots, \star)$.

Definition 2.18 *Das Schema $\hat{H}_k \in \Xi$ ist ein Schema, dessen fixierte Positionen und dessen Allele an diesen fixierten Positionen mit denen von H von der Zeichenposition $k \in \{h_1, \ldots, h_{\delta(H)}\}$ an übereinstimmen. Abgesehen von diesen erwähnten fixierten Positionen, sind keine weiteren Positionen von $\hat{H}_k$ fixiert. Ist $H = (s_1, \ldots, s_k, \ldots, s_l)$ dann ist also $\hat{H}_k = (\star, \ldots, \star, s_{k+1}, \ldots, s_l)$.*

Abbildung 2.3 zeigt ein Beispiel für H, H_k und $\hat{H}_k$. Es sind folgende Beziehungen erkennbar:

Hilfssatz 2.5 *Für H, H_k und $\hat{H}_k$ gelten folgende Beziehungen:*

a) $H \subseteq H_k$

b) $H \subseteq \hat{H}_k$

c) $H_k \cap \hat{H}_k = H$.

Beweis Sei $H = (s_1, \ldots, s_k, \ldots, s_l)$, $H_k = (s_1, \ldots, s_k, \star, \ldots, \star)$ und $\hat{H}_k = (\star, \ldots, \star, s_{k+1}, \ldots, s_l)$. Sei $x \in H$, also ist der Detektor $\delta_i(x) = s_i$ für $s_i \neq \star$, $i = 1, \ldots, l$.
zu a) Damit ist x auch Element von H_k. Es folgt $H \subseteq H_k$. Sei $x \in H_k$ und es gelte für einen Detektor $\delta_{i_0}(x) = 1 - s_{i_0}$ für ein $i_0 \in \{k+1, \ldots, l\}$ mit $s_{i_0} \neq \star$, das heißt x unterscheide sich an einer fixierten Position i_0 von dem s_{i_0} von H. So ist $x \notin H$. Also gilt: $x \in H_k \not\Rightarrow x \in H$
$\Rightarrow H_k \not\subset H$
Es sind Suchräume konstruierbar, in denen $H = H_k$ gilt, also ist $H \subseteq H_k$
zu b) Um b) zu beweisen geht man entsprechend a) vor, nur dass H_k durch $\hat{H}_k$ ersetzt wird und $i_0 \in 1, \ldots, k$ gewählt wird.
zu c) Sei $x \in H_k$ und $x \in \hat{H}_k$, also gilt für die Detektoren $\delta_i(x) = s_i$ für $i = 1, \ldots, k$ und $\delta_i(x) = s_i$ für $i = k+1, \ldots, l$.
$\Rightarrow x \in H$

$\Rightarrow H = H_k \cap \hat{H}_k$ $\square$

Es gilt also $H \subseteq H_{h_{\delta(H)}} \subseteq H_{h_{\delta(H)-1}} \subseteq \ldots \subseteq H_{h_2} \subseteq H_{h_1}$.

Folgerung 2.6 *Für die relativen Fitnesswerte von H, H_k und $\hat{H}_k$ gelten folgende Beziehungen:*

$a_1)$ $f_t(H_k) \geq f_t(H)$ $\qquad$ $a_2)$ $f_t(H_k \backslash H) = f_t(H_k) - f_t(H)$

$b_1)$ $f_t(\hat{H}_k) \geq f_t(H)$ $\qquad$ $b_2)$ $f_t(\hat{H}_k \backslash H) = f_t(\hat{H}_k) - f_t(H)$

$c)$ $f_t(H_k \cup \hat{H}_k) = f_t(H_k) + f_t(\hat{H}_k) - f_t(H)$.

Beweis zu a_1) Nach dem Hilfssatz 2.5 ist H eine Teilmenge von H_k. Damit besitzt H_k wenigstens so viele Vertreter, wie H. Dies gilt auch für die Anzahl der Vertreter in der Population. Da sich die relative Fitness eines Schemas aus der Summe der relativen Fitnesswerte seiner Vertreter in der Population errechnet, ist der relative Fitnesswert von H_k notwendigerweise mindestens so groß, wie der von H. Der Beweis wird jedoch auch formell durchgeführt, damit a_2) leichter zu beweisen ist.

$$
\begin{aligned}
f_t(H_k) &= \sum_{x \in P_t \cap H_k} f_t(x) \\
&= \sum_{x \in P_t \cap \left((H_k \backslash H) \cup H\right)} f_t(x) \qquad \text{mit Hilfssatz 2.5 a)} \\
&= \sum_{x \in \left(P_t \cap (H_k \backslash H)\right) \cup \left(P_t \cap H\right)} f_t(x) \\
&= \sum_{x \in P_t \cap (H_k \backslash H)} f_t(x) + \sum_{x \in P_t \cap H} f_t(x) \\
&\geq \sum_{x \in P_t \cap H} f_t(x) \\
&= f_t(H)
\end{aligned}
$$

zu a_2)

$$
\begin{aligned}
f_t(H_k) &= \sum_{x \in P_t \cap (H_k \backslash H)} f_t(x) + \sum_{x \in P_t \cap H} f_t(x) \\
&= f_t(H_k \backslash H) + f_t(H)
\end{aligned}
$$

Also ist $f_t(H_k \backslash H) = f_t(H_k) - f_t(H)$
zu b_1) und b_2) Man verfährt genauso wie in a_1) und a_2) mit der Ausnahme,

dass H_k durch $\hat{H}_k$ zu ersetzen ist.
zu c)

$$\begin{aligned} f_t(H_k \cup \hat{H}_k) &= \sum_{x \in P_t \cap (H_k \cup \hat{H}_k)} f_t(x) \\ &= \sum_{x \in (P_t \cap H_k) \cup (P_t \cap \hat{H}_k)} f_t(x) \\ &= \sum_{x \in (P_t \cap H_k)} f_t(x) + \sum_{x \in (P_t \cap H)} f_t(x) - \sum_{x \in (P_t \cap H_k) \cap (P_t \cap \hat{H}_k)} f_t(x) \\ &= f_t(H_k) + f_t(\hat{H}_k) - \sum_{x \in (P_t \cap H)} f_t(x) \quad \text{mit Satz 2.5 c)} \\ &= f_t(H_k) + f_t(\hat{H}_k) - f_t(H). \end{aligned}$$

□

Somit gilt also

$$f_t(H) \leq f_t(H_{h_{\delta(H)}}) \leq f_t(H_{h_{\delta(H)-1}}) \leq \ldots \leq f_t(H_{h_2}) \leq f_t(H_{h_1}).$$

Es gibt vier Kombinationen, in denen Vertreter von H, H_k und $\hat{H}_k$ miteinander gekreuzt werden können, so dass wenigstens ein aus dem Kreuztausch zwischen zwei Individuen hervorgehendes Element Vertreter von H ist. Folgende Übersicht gibt einen Überblick über diese Kombinationen, die Anzahl der Möglichkeiten, diese Kombinationen auszuwählen und die Anzahl der resultierenden Elemente, die bei diesen Kombinationen in H liegen:

Kombination	Anzahl der Arten, diese Kombination auszuwählen	Resultierende Anzahl von Vertretern von H
H und H	1	2
H und $H_k \backslash H$	2	1
H und $\hat{H}_k \backslash H$	2	1
$H_k \backslash H$ und $\hat{H}_k \backslash H$	2	1

Die Wahrscheinlichkeit, dass Vertreter von H, H_k oder $\hat{H}_k$ in einer derartigen Konstellation gekreuzt werden ist demnach

$$\begin{aligned} &P\big(\kappa(\{x_{2i-1}, x_{2i}\}, k_0) \cap H \neq \emptyset, k_0 \in \{h_1, \ldots, h_{\delta(H)}\}\big) = \\ &= P\Big(k_0 \in \{h_1, \ldots, h_{\delta(H)}\}\Big) P(k_0 = k) * \Big(P(x_{2i-1} \in H, x_{2i} \in H) + \\ &\quad P(x_{2i-1} \in H, x_{2i} \in H_k) + P(x_{2i-1} \in H, x_{2i} \in \hat{H}_k) + \\ &\quad P(x_{2i-1} \in H_k, x_{2i} \in \hat{H}_k)\Big) \\ &= p_c \frac{\delta(H)}{l-1} \frac{1}{\delta(H)} \Big(f_t(H)^2 + 2 f_t(H)[f_t(H_k) - f_t(H)] + \\ &\quad 2 f_t(H)[f_t(\hat{H}_k) - f_t(H)] + 2[f_t(H_k) - f_t(H)][f_t(\hat{H}_k) - f_t(H)]\Big). \end{aligned}$$

Der Faktor $p_c \frac{\delta(H)}{l-1}$ ist die Wahrscheinlichkeit, dass die Kreuzungsstelle zwischen die fixierten Positionen des Schemas fällt. Der Faktor $\frac{1}{\delta(H)}$ ist die Wahrscheinlichkeit, dass die Kreuzungsstelle k_0 genau an die gewünschte Stelle k fällt.

Der Wert von k kann zwischen h_1 und $h_{\delta(H)}$ variieren. Da die Kreuzungsstelle nur einmal ausgewählt wird, sind die Ereignisse $k = h_i$ und $k = h_j$ für $i \neq j$, $(i, j = h_1, \ldots, h_{\delta(H)})$ disjunkt und die Wahrscheinlichkeiten der Ereignisse

$$P\big(\kappa(\{x_{2i-1}, x_{2i}\}, h_1) \cap H \neq \emptyset\big), \ldots, P\big(\kappa(\{x_{2i-1}, x_{2i}\}, h_{\delta(H)}) \cap H \neq \emptyset\big)$$

können addiert werden. Also:

$$P\big(\kappa(\{x_{2i-1}, x_{2i}\}, k) \cap H \neq \emptyset, \quad k \in \{h_1, \ldots, h_{\delta(H)}\}\big) =$$

$$\begin{aligned} = \; & p_c \frac{1}{l-1} \sum_{k=h_1}^{h_{\delta(H)}} \Big(f_t(H)^2 + \\ & 2 f_t(H)[f_t(H_k) - f_t(H)] + \\ & 2 f_t(H)[f_t(\hat{H}_k) - f_t(H)] + \\ & 2[f_t(H_k) - f_t(H)][f_t(\hat{H}_k) - f_t(H)]\Big). \end{aligned}$$

Nun wird die zu erwartende Anzahl von Vertretern von H bei diesem Ereignis errechnet. Nur wenn zwei Vertreter von H miteinander gekreuzt werden, sind liegen die beiden resultierenden Elemente in H. Bei den anderen drei Konstellationen kann nur eines der entstehenden Elemente in H liegen:

$$E\left|\kappa(\{x_{2i-1}, x_{2i}\}, k) \cap H \neq \emptyset, k \in \{h_1, \ldots, h_{\delta(H)}\}\right| =$$

$$\begin{aligned} & \frac{p_c}{l-1} \sum_{k=h_1}^{h_{\delta(H)}} \Big(\mathbf{2} f_t(H)^2 + 2 f_t(H)[f_t(H_k) - f_t(H)] + \\ & \qquad 2 f_t(H)[f_t(\hat{H}_k) - f_t(H)] + \\ & \qquad 2[f_t(H_k) - f_t(H)][f_t(\hat{H}_k) - f_t(H)]\Big) \\ = \; & 2 \frac{p_c}{l-1} \sum_{k=h_1}^{h_{\delta(H)}} f_t(H_k) f_t(\hat{H}_k). \end{aligned} \tag{2.11}$$

Es ist zu bemerken, dass die relative Fitness eines Teilschemas H_{h_m} genau m-mal in die Summe

$$\sum_{k=h_1}^{h_{\delta(H)}} f_t(H_k) f_t(\hat{H}_k)$$

eingeht, denn es gilt

$$H_{h_m} \subseteq \ldots \subseteq H_{h_2} \subseteq H_{h_1}.$$

Damit ist H_{h_m} in m Schemen vertreten und die relative Fitness von H_{h_m} trägt zur Fitness von m Schemen bei, denn

$$f_t(H_{h_m}) \leq f_t(H_{h_{m-1}}) \leq \ldots \leq f_t(H_{h_2}) \leq f_t(H_{h_1}).$$

Die Ereignisse $k \in \{h_1, \ldots, h_{\delta(H)}\}$ und $k \notin \{h_1, \ldots, h_{\delta(H)}\}$ sind ebenfalls disjunkt. Daher können die Wahrscheinlichkeiten und damit die Erwartungswerte für die zu erwartende Vertreteranzahl von H bei diesen Ereignissen ((2.10) und (2.11)) addiert werden. Damit erhält man einen Ausdruck für die zu erwartende Anzahl von Vertretern von H nach dem Anwenden des Kreuztauschoperators. Auf Grund von verbesserter Lesbarkeit variiert der Wert der Kreuzungsstelle k nun zwischen 1 und $\delta(H)$[3]. Weiterhin werden insgesamt n Paare gekreuzt. Damit ist der Erwartungswert für die Anzahl der Vertreter von H nach dem Anwenden des Kreuztauschoperators:

$$\begin{aligned} E\left|\{O_K(P_t) \cap H\}\right| &= 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right) + \\ &\quad 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k) \\ &= 2n\left[f_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right) + \frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k)\right]. \end{aligned} \tag{2.12}$$

(2) Betrachtet wird nun das Szenario (2). Nach der Mutation eines Vertreters von H soll das resultierende Element ebenfalls in H liegen. Es wird angenommen, es existiert ein Vertreter von H nach dem Kreuztausch. Dieser wird mit $x_H \in O_K(P_t) \cap H$ bezeichnet. Die Wahrscheinlichkeit, dass aus der Mutation von x_H ein Vertreter von H entsteht, wurde bereits hergeleitet. Nach (2.6) ist

$$P(m(x_H) \in H) = (1 - p_m)^{o(H)}. \tag{2.13}$$

(3) Abschließend ist der letzte Fall zu betrachten. Aus der Mutation eines Elementes, welches nicht in H liegt, soll ein Vertreter von H resultieren. Um dieses Ereignis zu untersuchen, wird eine Klasse von Schemen $\bar{H}_i$ eingeführt.

Definition 2.19 *Die Menge $\bar{H}_i$ ist eine Menge von Schemen, die sich an genau i Stellen von den fixierten Positionen von H unterscheiden. Ist also*

[3] Innerhalb der Chromosomen ist die Kreuzungsstelle dann $(h_1 - 1) + k$.

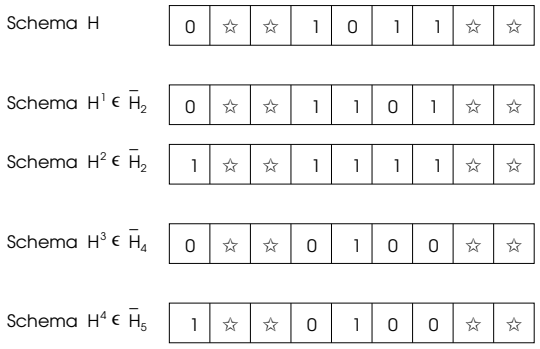

Abbildung 2.4.: Beispiele für Schemen aus den Klassen $\bar{H}_i$ für $i = 2, 4$ und 5.

$H = (s_1, \ldots, s_l)$, dann gilt:
$H' = (s'_1, \ldots, s'_l) \in \bar{H}_i \Leftrightarrow$ für $s_k \neq \star$ gilt $s'_k = 1 - s_k$ für $k = k_1, \ldots, k_i$ und $s'_m = s_m$ für $m = k'_1, \ldots, k'_{o(H)-i}$.

Ein Schema $H' \in \bar{H}_i$ kann also aus H konstruiert werden, indem beliebige i Allele an den fixierten Positionen von H negiert werden. In Abbildung 2.4 sind einige Beispiele für Schemen, die in $\bar{H}_i$ liegen graphisch dargestellt. Das Schema H' besitzt die gleiche Ordnung und Länge wie das Schema H:

$$\begin{aligned} o(H') &= o(H) \\ \delta(H') &= \delta(H). \end{aligned}$$

Es wird angenommen, dass nach dem Anwenden des Kreuztauschoperators ein Element $x_{H_i} \in H' \in \bar{H}_i$ existiert, welches sich an genau i Stellen von den Allelen der fixierten Positionen des Schemas H unterscheidet.

Das Element x_{H_i} mutiert zu einem Vertreter von H, wenn an genau den i Positionen, an denen H' sich von H unterscheidet, Mutationen stattfinden. Die restlichen Positionen von x_{H_i}, deren Allele mit denen von H übereinstimmen, dürfen dann nicht mutiert werden. Die Wahrscheinlichkeit, dass i Mutationen stattfinden und $\big(o(H) - i\big)$ Mutationen nicht, ist:

$$P(i \text{ Mutationen an best. fixierten Pos. von } H) = {p_m}^i (1 - p_m)^{o(H)-i}. \tag{2.14}$$

Es wurde angenommen, dass ein solches Element x_{H_i} in der Menge der Individuen nach Anwenden des Kreuztauschoperators vorhanden ist. Die Wahrscheinlichkeit $P(x_{H_i} \in O_K(P_t))$ ist jedoch nicht nur abhängig von der Fitness $f_t(x_{H_i})$, der Länge und Ordnung von H', sondern auch von der derzeit vorhandenen Population[4]. Weiterhin ist $P(x_{H_i} \in O_K(P_t))$ von der

[4]Betrachtet man das Schema H=(1,1, ... ,1) und besteht die Population P nur aus Individuen, welche die Chromosomen (0,0, ... ,0) besitzen, so wird $P(x_{H_1} \in C(P))$ sicherlich geringer sein, als in einer Population, deren Individuen mehr über den Suchraum verteilt sind.

Verteilung der i Positionen über die $o(H)$ Positionen abhängig. Für die Wahrscheinlichkeit, dass nach dem Anwenden des Kreuztauschoperators Vertreter von H in $O_K(P_t)$ sind, die sich an genau i Stellen von H unterscheiden, müssen daher alle $\binom{o(H)}{i}$ Schemen $H' \in \bar{H}_i$ betrachtet werden:

$$P(x_{H_i} \in O_K(P_t)) = \sum_{H' \in \bar{H}_i} P\big(O_K(P_t) \cap H' \neq \emptyset\big).$$

Nun kann die Wahrscheinlichkeit angeben werden, dass ein Vertreter eines Schemas $H' \in H_i$ vorhanden ist und zu einem Vertreter von H mutiert:

$$\begin{aligned} P\big(m(x_{H_i}) \in H\big) &= P(i \text{ Mut. best. Pos. von } H) P(x_{H_i} \in O_K(P_t)) \\ &= {p_m}^i (1-p_m)^{o(H)-i}\, P(x_{H_i} \in O_K(P_t)) \\ &= {p_m}^i (1-p_m)^{o(H)-i} \sum_{H' \in \bar{H}_i} P\big(O_K(P_t) \cap H' \neq \emptyset\big). \end{aligned} \tag{2.15}$$

Der Wert von i kann zwischen 1 und $o(H)$ variieren. So erhält man die Wahrscheinlichkeit, dass ein Vertreter von H nach der Mutation eines Elementes, welches nicht in H liegt resultiert:

$$\begin{aligned} &P\big(m(x) \in H \text{ mit } x \notin H\big) \quad = \\ &\sum_{i=1}^{o(H)} {p_m}^i (1-p_m)^{o(H)-i} \sum_{H' \in \bar{H}_i} P\big(O_K(P_t) \cap H' \neq \emptyset\big). \end{aligned} \tag{2.16}$$

Da aus der Mutation von einem Element nur ein Element resultiert, kann maximal ein Vertreter von H nach der Mutation vorhanden sein. Der Mutationsoperator O_M wird auf $2n$ Elemente angewendet, daher ist die erwartete Anzahl von Vertretern von H, die aus der Mutation eines Elementes, das nicht in H liegt resultiert:

$$\begin{aligned} &E\Big|\big\{x : x \notin H \text{ und } m(x) \in H\big\}\Big| \quad = \\ &\qquad 2n \sum_{i=1}^{o(H)} {p_m}^i (1-p_m)^{o(H)-i} \sum_{H' \in \bar{H}_i} P\big(O_K(P_t) \cap H' \neq \emptyset\big). \end{aligned} \tag{2.17}$$

Nun können (2.12), (2.13) und (2.17) zusammengesetzt werden. Man erhält folgenden Satz:

Satz 2.7 (Modifizierter Schemensatz) *In einem einfachen genetischen Algorithmus ist die erwartete Anzahl von Vertretern eines Schemas H in der Folgegeneration $(t+1)$:*

$$E\Big|\big\{P_{t+1} \cap H\big\}\Big| =$$

$$
\begin{aligned}
&2nf_t(H)\left(1-p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)} + \\
&2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k)(1-p_m)^{o(H)} + \\
&2n\sum_{i=1}^{o(H)} p_m^i(1-p_m)^{o(H)-i}\sum_{H'\in\bar{H}_i} P\big(O_K(P_t)\cap H'\neq\emptyset\big).
\end{aligned}
\tag{2.18}
$$

Bezeichnung 2.20 *Es werden folgende Bezeichnungen der Terme in dem modifizierten Schemensatz vergeben:*

$$
\begin{aligned}
E_{Co_1} := E_{Co_1}(H,t) &:= 2nf_t(H)\left(1-p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)} \\
E_{Co_2} := E_{Co_2}(H,t) &:= 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k)(1-p_m)^{o(H)} \\
E_{Mut} := E_{Mut}(H,t) &:= 2n\sum_{i=1}^{o(H)} p_m^i(1-p_m)^{o(H)-i}\sum_{H'\in\bar{H}_i} P\big(O_K(P_t)\cap H'\neq\emptyset\big) \\
E_S := E_S(H,t) &:= E_{Co_1}+E_{Co_2}+E_{Mut}.
\end{aligned}
$$

2.4.2. Interpretation des modifizierten Schemensatzes

Der modifizierte Schemensatz bietet einen exakten Ausdruck für den Erwartungswert für die Anzahl der Vertreter eines Schemas in der Folgepopulation. Somit sind nun genauere Untersuchungen an der Entwicklung der Population eines genetischen Algorithmus möglich als bisher. Es könnten zum Beispiel Abschätzungen für das erstmalige Auftreten des Optimums in der Population bei gegebenen Parameterbelegungen berechnet werden.

Weiterhin kann der modifizierte Schemensatz für die Optimierung der Parameter eines einfachen genetischen Algorithmus genutzt werden. Bisher war dies mit dem Schemensatz nur eingeschränkt möglich.

Es soll formal bewiesen werden, dass durch E_S die untere Schranke aus dem Schemensatz 2.3 verbessert wurde.

Hilfssatz 2.8 *In einem einfachen genetischen Algorithmus gilt:*

$$E_S(H,t) \geq E_H(H,t).$$

Beweis Nach Hilfssatz 2.6 gilt:

$$
\begin{aligned}
f_t(H_k) &\geq f_t(H) \\
f_t(\hat{H}_k) &\geq f_t(H),
\end{aligned}
$$

und daraus folgt

$$\begin{aligned} E_{Co_2} &= 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k)(1-p_m)^{o(H)} \\ &\geq 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H)^2(1-p_m)^{o(H)} \\ &= 2np_c\frac{\delta(H)}{l-1}f_t(H)^2(1-p_m)^{o(H)}. \end{aligned}$$

Somit:

$$\begin{aligned} E_S &= E_{Co_1} + E_{Co_2} + E_{Mut} \\ &\geq E_{Co_1} + E_{Co_2} \\ &= 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)} + \\ &\quad 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f_t(H_k)f_t(\hat{H}_k)(1-p_m)^{o(H)} \\ &\geq 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)} + 2np_c\frac{\delta(H)}{l-1}f_t(H)^2(1-p_m)^{o(H)} \\ &= 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)}. \end{aligned}$$

Und mit (2.4) folgt:

$$\begin{aligned} E_S &\geq 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)} \\ &= m(H,t)\frac{F_t(H)}{\bar{F}_t}\left(1 - p_c\frac{\delta(H)}{l-1}\big(1-f_t(H)\big)\right)(1-p_m)^{o(H)} \\ &= E_H. \end{aligned}$$

□

Es ist zu bemerken, dass E_S, also der Erwartungswert der Anzahl an Vertretern eines Schemas in der Folgepopulation, immer positiv ist. Für den Beweis wird folgender Hilfssatz benötigt.

Hilfssatz 2.9 *In einem einfachen genetischen Algorihtmus mit $p_m < 1$ ist der Wert für $E_{Mut}(H,t)$ genau dann Null, wenn alle Individuen von P_t Vertreter des Schemas H sind. Also:*

$$E_{Mut} = 0 \Leftrightarrow f_t(H,t) = 1.$$

Beweis Sei $H' \in \bar{H}_i$ ein Schema, welches sich an i Stellen von H unterscheidet.

$$
\begin{aligned}
& E_{Mut} = 0 \\
\Leftrightarrow \quad & 2n \sum_{i=1}^{o(H)} p_m^i (1-p_m)^{o(H)-i} \sum_{H' \in \bar{H}_i} P\big(O_K(P_t) \cap H' \neq \emptyset\big) = 0 \\
\Leftrightarrow \quad & 2n \sum_{i=1}^{o(H)} {p_m}^i (1-p_m)^{o(H)-i} \; P(x_{H_i} \in O_K(P_t)) = 0 \\
& \qquad \text{mit } x_{H_i} \in H' \quad \text{für alle } H' \in \bar{H}_i \\
\Leftrightarrow \quad & P(x_{H_i} \in O_K(P_t)) = 0 \text{ mit } x_{H_i} \in H' \quad \forall H' \in \bar{H}_i \quad \forall i = 1, \ldots, o(H).
\end{aligned}
$$

Das bedeutet $E_{Mut} = 0$ ist nur erfüllt, wenn nach dem Anwenden des Kreuztauschoperators kein Individuum vorhanden ist, das sich an irgendeinem Gen von H unterscheidet. Also:

$$
\begin{aligned}
& P(x_{H_i} \in O_K(P_t)) = 0 \text{ mit } x_{H_i} \in H' \quad \forall H' \in \bar{H}_i \quad \forall i = 1, \ldots, o(H) \\
\Leftrightarrow \quad & E_{Co_1}(H', t) + E_{Co_2}(H', t) = 0 \quad \forall H' \in \bar{H}_i \quad \forall i = 1, \ldots, o(H).
\end{aligned}
$$

Dies ist nur erfüllt, wenn $f_t(H') = 0$ und $f_t(H'_k) f(\hat{H}'_k) = 0$ für alle $k \in \{1, \ldots, \delta(H)\}$ für alle $H' \in \bar{H}_i$ und für alle $i = 1, \ldots, o(H)$. Also dürfte schon vor dem Kreuztausch kein Individuum vorhanden sein, welches Vertreter von H' ist. Dies tritt genau dann ein, wenn alle Indivduen Vertreter von H sind. Also:

$$
\begin{aligned}
& E_{Co_1}(H', t) + E_{Co_2}(H', t) = 0 \quad \forall H' \in \bar{H}_i \\
\Leftrightarrow \quad & \sum_{x \in P_t \cap H} f_t(x) = f_t(H) = 1 \quad \text{nach Satz 2.1 c)}
\end{aligned}
$$

Somit wurde gezeigt, dass $E_{Mut} = 0$ genau dann erfüllt ist, wenn $f_t(H) = 1$ gilt. □

Nun kann gezeigt werden, dass der Wert für E_S für jedes Schema positiv ist.

Hilfssatz 2.10 *In einem einfachen genetischen Algorithmus mit $p_m < 1$ gilt:*

$$E_S(H, t) = E_{Co_1} + E_{Co_2} + E_{Mut} > 0 \qquad \forall H \in \Xi \, .$$

Beweis Fallunterscheidung:
1. Fall: Sei $f_t(H) > 0$:

$$E_S(H, t) \geq E_H(H, t) \qquad \text{siehe Hilfssatz 2.8}$$

Nach Hilfssatz 2.4 gilt $E_H(H, t) = 0 \Leftrightarrow f_t(H) = 0$. Hier ist $f_t(H) > 0$ gefordert, also ist auch $E_H(H, t) > 0$. Und somit: $E_S(H, t) \geq E_H(H, t) > 0$.
2. Fall: $f_t(H) = 0$: Nach Hilfsatz 2.9 ist $E_{Mut}(H, t) > 0$, da $f_t(H) \neq 1$. Damit ist auch $E_S(H, t) = E_{Co_1} + E_{Co_2} + E_{Mut} > 0$.

Mithin $E_S(H,t) > 0 \qquad \forall H \in \Xi.$ □

Es ist interessant zu bemerken, dass die erwartete Vertreteranzahl für jedes Schema positiv ist. Das heißt jedes Schema besitzt eine positive Wahrscheinlichkeit, in der nächsten Generation Vertreter zu besitzen. Sind die Schemen derart konstruiert, dass jedes Element des Suchraumes einziger Vertreter von wenigstens einem Schema ist, so kann jedes Element des Suchraumes erreicht werden.

Im Folgenden soll gezeigt werden, dass eine Erhöhung der relativen Fitness eines Schemas unweigerlich zu einer Erhöhung seiner erwarteten Vertreteranzahl in der Folgepopulation führt.

Hilfssatz 2.11 *Die Funktionen $E_H(H,t)$ und $E_S(H,t)$ sind streng monoton wachsend bezüglich der relativen Fitness von H, wenn*

$$f_t(H) > 0 \quad \textit{und} \quad p_m < 1.$$

Beweis $E_H(H,t)$ und $E_S(H,t)$ werden als Funktionen von $f_t(H)$ betrachtet. Bildet man die Ableitung von $E_H(H,t)$ nach $f_t(H)$, so sieht man:

$$\begin{aligned}
\frac{\partial E_H(H,t)}{\partial f_t(H)} &= \frac{\partial}{\partial f_t(H)} \left(2nf_t(H) \left(1 - p_c \frac{\delta(H)}{l-1}(1 - f_t(H)) \right) (1-p_m)^{o(H)} \right) \\
&= \frac{\partial}{\partial f_t(H)} \left(2nf_t(H) - 2nf_t(H)p_c\frac{\delta(H)}{l-1} + \right. \\
&\quad \left. 2np_c\frac{\delta(H)}{l-1}f_t(H)^2 \right) (1-p_m)^{o(H)} \\
&= \left(2n - 2np_c\frac{\delta(H)}{l-1} + 4np_c\frac{\delta(H)}{l-1}f_t(H) \right) (1-p_m)^{o(H)} \\
&\geq 4np_c\frac{\delta(H)}{l-1}f_t(H)(1-p_m)^{o(H)} \quad \text{da} \quad 2n - 2np_c\frac{\delta(H)}{l-1} \geq 0 \\
&> 0 \qquad \text{für} \quad f_t(H) > 0 \quad \text{und} \quad p_m < 1.
\end{aligned}$$

$E_H(H,t)$ ist also für $f_t(H) > 0$ streng monoton wachsend. Da $E_S(H,t) \geq E_H(H,t)$, ist somit auch $E_S(H,t)$ streng monoton wachsend. □

In diesem Abschnitt wurde der Schemensatz von Holland 2.3 modifiziert, indem die untere Schranke für die erwartete Vertreteranzahl eines Schemas in der Folgepopulation verbessert wurde. Es ist nun möglich, diesen Erwartungswert exakt anzugeben.

Weiterhin konnten analytische Betrachtungen zur Entwicklung der Population eines einfachen genetischen Algorithmus durchgeführt werden. So wurde mit der Hilfe des modifizierten Schemensatzes bewiesen, dass die erwartete Vertreteranzahl für jedes Schema positiv ist. Sind die Schemen derart konstruiert, dass jedes Element Vertreter eines Schemas ist, welches an jeder Position fixiert ist, so bedeutet diese Erkenntnis, dass jedes Element des Suchraumes erreichbar ist, also in der Folgepopulation auftreten kann.

Wird der Erwartungswert für die Vertreteranzahl eines Schemas als Funktion von der relativen Fitness eines Schemas aufgefasst, so ist diese Funktion streng monoton wachsend. Dieses Ergebnis wird in einem späteren Kapitel genutzt.

3. Computersimulationen

Es werden in diesem Kapitel die Ergebnisse von numerischen Untersuchungen an Hollands Schemensatz (2.3) und dem modifizierten Schemensatz (2.7) vorgestellt. So wird die Genauigkeit, mit der E_H und E_S die Vertreteranzahl eines Schemas in der Folgepopulation vorhersagen untersucht. Damit erhält man einen Eindruck, wie sehr die Genauigkeit der Vorhersage erhöht wird, wenn E_S benutzt wird. Weiterhin wird die durchschnittliche Differenz zwischen E_H und E_S betrachtet und die Auswirkungen der Wahl der Zielfunktion auf die Entwicklung des einfachen genetischen Algorithmus beleuchtet. Zum Abschluß wird die Mutationswahrscheinlichkeit für den einfachen genetischen Algorithmus optimiert.

Um diese Untersuchungen zu ermöglichen, wurde ein einfacher genetischer Algorithmus programmiert, welcher zusammen mit dieser Arbeit eingereicht wurde. Die wesentlichen Komponenten und Berechnungsvorschriften des Programmes werden in Abschnitt 3.1 beschrieben.

3.1. Das Programm

In diesem Abschnitt wird das, für die Diplomarbeit programmierte Programm beschrieben, welches zur Erlangung numerischer Daten diente. Es wird dabei nur auf die Komponenten näher eingegangen, welche zum Verständnis des Programmablaufes notwendig und zum Erlangen der numerischen Daten von direkter Bedeutung sind.

Zu Beginn wird der Ablauf des Programmes vorgestellt. Darauf folgt eine detailliertere Beschreibung, wie die Auswahl der Individuen für den Kreuztausch erfolgt. Anschließend wird kurz beschrieben, wie die numerischen Werte berechnet und ausgewertet wurden.

Ablauf des Computerprogrammes

Die Kreuztausch- und Mutationswahrscheinlichkeit (p_c und p_m) sowie die Populationsgröße ($2n$) und die Chromosomenlänge (l) werden zu Beginn des Programmes mit Werten belegt.

Das Programm startet mit der Initialisierung der Individuen der Anfangspopulation. Die Chromosomen der Individuen werden bei der Initialisierung derart erzeugt, dass jedes Gen, also jede Zeichenposition, mit einer Wahrscheinlichkeit von 50% mit „0“ oder mit „1“ belegt wird. Dann werden die Chromosomen dekodiert, so dass die Individuen nicht nur eine binäre Darstellung besitzen, sondern auch den Punkt im Suchraum

zugewiesen bekommen, den sie mit ihren Chromosomen repräsentieren. Für dieses Programm wurde eine Kodierung von ganzen Zahlen in binäre Zeichenketten gewählt, bei der das erste Zeichen der binären Zeichenkette das Vorzeichen der Zahl kodiert und die restlichen Zeichen die binäre Darstellung einer positiven ganzen Zahl sind. Eine „1“ als erstes Zeichen bedeutet dabei, dass die Zahl ein negatives Vorzeichen besitzt. Ist also $a_1 a_2 \dots a_l$ $(a_i \in \{0,1\}\ i = 1, \dots, l)$ das Chromosom von x, so repräsentiert x den Punkt $y \in \mathfrak{A}$ mit

$$y = (-1)^{a_1} * \sum_{k=0}^{l-2} a_{l-k} 2^k.$$

Das Abbruchkriterium des Programmes ist eine maximale Generationenanzahl N. In einer Schleife werden folgende Schritte N-mal durchlaufen, wobei das einmalige Durchlaufen der Schleife einer Generation entspricht.

Zu Beginn werden die Zielfunktionswerte der Individuen berechnet. Anschließend wird jedem Individuum seine relative Fitness zugewiesen. Hierfür wird zuerst die Summe aller positiven Zielfunktionswerte errechnet. Die relative Fitness $f_t(x)$ für ein Individuum x wird dann in Abhängigkeit von dieser Summe und seinem Zielfunktionswert $z(x)$ berechnet:

$$f_t(x) = \begin{cases} \frac{z(x)}{\sum\limits_{y \in P_t} F(y)} & : \text{für } z(x) > 0 \text{ und } \sum\limits_{y \in P_t} F(y) > 0 \\ 0 & : \text{für } z(x) \leq 0 \text{ und } \sum\limits_{y \in P_t} F(y) > 0 \\ \frac{1}{2n} & : \text{sonst} \end{cases}$$

mit

$$\sum_{y \in P_t} F(y) := \sum_{\substack{y \in P_t \\ z(y) > 0}} z(y).$$

Es kann gezeigt werden, dass die Fitnessfunktion f_t die in Satz 2.1 c) gestellte Bedingung

$$\sum_{x \in P_t} f_t(x) = 1$$

erfüllt. Der Ausdruck $F(x)$ kann als Fitnesswert des Individuums x betrachtet werden, wird aber als solcher weder explizit berechnet noch benötigt.

Nach der Zuweisung der relativen Fitnesswerte werden n Paare von Individuen gebildet, auf die der 1-Punkt Kreuztausch angewendet wird. Dabei ist $f_t(x)$ die Wahrscheinlichkeit, dass das Individuum x als Partner in einem Paar ausgewählt wird. Eine genauere Beschreibung, wie diese Auswahl und Paarbildung erfolgt, wird in dem nächsten Unterabschnitt gegeben.

Für jedes Paar wird dann mit der Kreuztauschwahrscheinlichkeit p_c entschieden, ob der Kreuztausch auf dieses Paar angewendet wird. Wird er angewendet, so wird mit einer gleichmäßig verteilten Wahrscheinlichkeit eine Kreuzungsstelle zwischen 1 und $(l-1)$ ausgewählt, an der die Chromosomen der Individuen des Paares gekreuzt werden. Die gekreuzten Chromosomen ersetzen dann die beiden Eltern-Chromosomen.

Die Mutation wird auf die aus dem Kreuztausch resultierenden Individuen angewendet, unabhängig davon, ob sie gekreuzt wurden oder nicht. Bei diesen Individuen wird jedes Gen mit der Mutationswahrscheinlichkeit p_m mutiert. Bei einer Mutation wird das entsprechende Allel an diesem Gen negiert. Fand eine Mutation statt, so ersetzt das mutierte Chromosom das ursprüngliche Chromosom.

Anschließend werden die gekreuzten und mutierten Chromosomen wieder dekodiert, so dass jedem Chromosom der Punkt des Suchraumes zugewiesen wird, den es mit seinen geänderten Allelen repräsentiert. Danach werden die Individuen der ursprünglichen Population durch die gekreuzten und mutierten Individuen ersetzt und der Generationenzähler um Eins erhöht. Ist der Generationenzähler kleiner als die maximale Generationenanzahl N, so wird die Schleife von der Zielfunktionszuweisung bis zum Ersetzen der Population und Erhöhung des Generationenzählers noch einmal durchlaufen. Das einmalige Abarbeiten des gesamten Programmes wird **Simulation** genannt.

Die Auswahl der Individuen für den Kreuztausch

Nachdem jedem Individuum x der Population ein relativer Fitnesswert $f_t(x)$ zugewiesen wurde, wird nun für jedes Individuum eine Nachkommenszahl $b(x)$ berechnet:

$$b(x) = \lfloor 2nf_t(x) \rfloor \qquad \forall x \in P_t.$$

Die Nachkommenszahl eines Individuums ist der ganzzahlige Anteil seiner relativen Fitness multipliziert mit der Populationsgröße. Die relative Fitness von x wird aktualisiert auf $\tilde{f}_t(x)$:

$$\tilde{f}_t(x) := \frac{2nf_t(x) - b(x)}{2n}.$$

Der Wert von $\tilde{f}_t(x)$ entspricht dem Wert, um den $2nf_t(x)$ bei Berechnung der Nachkommenszahl abgerundet wird. Die Aktualiserung der relativen Fitness auf $\tilde{f}_t(x)$ ist notwendig, damit die eventuelle Zuweisung von weiteren Nachkommen, wenn $\sum_{x \in P_t} b(x) \neq 2n$ eintritt, gerecht erfolgt. Das heißt, dass die Zuordnung der Nachkommen entsprechend der relativen Fitness der Individuen erfolgt.

Wurde für jedes Individuum eine Nachkommenszahl berechnet, so wird eine Matrix, der Dimension $2 \times n$ generiert. In den einzelnen Matrixfeldern sollen die Nachkommen zu n Paaren angeordnet werden. Für jedes Individuum x werden $b(x)$ Nachkommen zufällig auf die Matrixfelder verteilt,

so dass jedem Nachkommen der Matrix maximal ein anderer Nachkomme zugeordnet ist.

Sind nicht alle Matrixfelder belegt, nachdem jedes Individuum entsprechend seiner relativen Fitness Nachkommen erhalten hat und Letztere in der Matrix angeordnet worden, so werden die Individuen bezüglich ihrer aktualisierten Fitness $\tilde{f}_t$ sortiert. Sind m Felder im Matrixfeld unbelegt, so bekommen die besten m Individuen einen weiteren Nachkommen, welcher ebenfalls zufällig in der Matrix angeordnet wird.

Berechnung und Auswertung von $\mathbf{E_H(H,t)}$ und $\mathbf{E_S(H,t)}$

Um eine möglichst genaue Auswertung zu erhalten, werden die Berechnungen für möglichst viele Schemen durchgeführt.

Bezeichnung 3.1 *Das Schema* $H^\star = (s_1, \ldots, s_l) \in \Xi$ *ist das Schema mit*

$$s_i = \star \quad \forall i = 1, \ldots, l.$$

Bei einer Chromosomenlänge von l gibt es 3^l verschiedene Schemen. Die folgenden Berechnungen werden für alle Schemen bis auf $H^\star$ durchgeführt. Die Anzahl der betrachteten Schemen beträgt also $3^l - 1$. Die Schemen $H \in \Xi \backslash H^\star$ werden nach der Initialisierung der Population erzeugt.

Nachdem die relativen Fitnesswerte den Indivdiduen zugewiesen wurden, werden die relativen Fitnesswerte der Schemen berechnet. Die relative Fitness $f_t(H)$ eines Schemas ist dabei die Summe der relativen Fitnesswerte seiner Vertreter (siehe Definition 2.13):

$$f_t(H) = \sum_{x \in P_t \cap H} f_t(x).$$

Anschließend wird die Anzahl der Vertreter der einzelnen Schemen in der Population bestimmt und $E_H(H,t)$ und $E_S(H,t)$ errechnet.

Für die Berechnung von $E_H(H,t)$ braucht dabei nur der Ausdruck

$$E_H(H,t) = 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\big(1 - f_t(H)\big)\right)(1-p_m)^{o(H)}$$

ausgewertet werden.

Für $E_S(H,t) = E_{Co_1} + E_{Co_2} + E_{Mut}$ sind jedoch mehrere Berechnungen notwendig. Zunächst wird E_{Co_1} berechnet:

$$E_{Co_1} = 2nf_t(H)\left(1 - p_c\frac{\delta(H)}{l-1}\right)(1-p_m)^{o(H)}.$$

Dann werden für jedes $k = h_1, \ldots, h_{\delta(H)}$ die Schemen H_k und $\hat{H}_k$ erzeugt. Die relative Fitness der Schemen H_k und $\hat{H}_k$ wird ermittelt, so dass die Summe

$$\sum_{k=1}^{\delta(H)} f(H_k) f(\hat{H}_k)$$

gebildet werden kann. Damit wird

$$E_{Co_2} = 2n\frac{p_c}{l-1}\sum_{k=1}^{\delta(H)} f(H_k)f(\hat{H}_k)(1-p_m)^{o(H)}$$

berechnet.

Für E_{Mut} wird die Wahrscheinlichkeit berechnet, dass Schemen $H' \in \bar{H}_i$, die sich an genau i Positionen von den fixierten Positionen von H unterscheiden, nach dem Kreuztausch Vertreter in der Population besitzen. Es gibt genau $\binom{o(H)}{i}$ dieser Schemen. Für jedes Schema $H' \in \bar{H}_i$ werden die Terme $E_{Co_1}(H',t)$ und $E_{Co_2}(H',t)$ berechnet, so dass

$$P(O_K(P_t) \cap H' \neq \emptyset) = \frac{E_{Co_1}(H',t)}{2n} + \frac{E_{Co_2}(H',t)}{2n}$$

angegeben werden kann. Damit wird E_{Mut} berechnet:

$$E_{Mut} = 2n\sum_{i=1}^{o(H)} p_m^i(1-p_m)^{o(H)-i}\sum_{H'\in\bar{H}_i} P(O_K(P_t)\cap H' \neq \emptyset).$$

Es folgt die Berechnung von $E_S(H,t)$:

$$E_S(H,t) = E_{Co_1} + E_{Co_2} + E_{Mut}.$$

Nach der Mutation wird die tatsächliche Vertreteranzahl $m(H,t+1)$ für alle $H \in \Xi\backslash H^\star$ bestimmt. Für jedes Schema $H \in \Xi\backslash H^\star$ wird $g_H := g_H\big(E_H(H,t), m(H,t+1)\big)$ als Maß, wie gut der Term $E_H(H,t)$ die tatsächliche Vertreteranzahl eines Schemas $m(H,t+1)$ angenähert hat, berechnet:

$$g_H := \begin{cases} 1+\frac{E_H(H,t)}{2n} & : \quad m(H,t+1)=0 \\ 1-\frac{m(H,t+1)}{2n} & : \quad m(H,t+1)\neq 0 \quad \text{und} \quad E_H(H,t)=0 \\ \frac{E_H(H,t)}{m(H,t+1)} & : \quad m(H,t+1)\neq 0 \quad \text{und} \quad E_H(H,t)\neq 0. \end{cases}$$

Ebenso wird $g_S := g_S\big(E_S(H,t), m(H,t+1)\big)$ als Maß für die Approximation von $m(H,t+1)$ durch $E_S(H,t)$ berechnet:

$$g_S := \begin{cases} 1+\frac{E_S(H,t)}{2n} & : \quad m(H,t+1)=0 \\ 1-\frac{m(H,t+1)}{2n} & : \quad m(H,t+1)\neq 0 \quad \text{und} \quad E_S(H,t)=0 \\ \frac{E_S(H,t)}{m(H,t+1)} & : \quad m(H,t+1)\neq 0 \quad \text{und} \quad E_S(H,t)\neq 0. \end{cases}$$

Es kann gezeigt werden, dass g_H und g_S mit steigender relativer Fitness eines Schemas wachsen.

Hilfssatz 3.1 *Die Funktionen g_H und g_S sind monoton wachsend bezüglich $f_t(H)$.*

Beweis a) Zunächst wird gezeigt, dass g_H monoton wachsend ist. Die Funktion $g_H(E_H(H,t), m(H,t+1))$ ist definiert durch:

$$g_H = \begin{cases} 1 + \frac{E_H(H,t)}{2n} & : \quad m(H,t+1) = 0 \\ 1 - \frac{m(H,t+1)}{2n} & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) = 0 \\ \frac{E_H(H,t)}{m(H,t+1)} & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) \neq 0. \end{cases}$$

Es wird die Ableitung von $g_H(E_H(H,t), m(H,t+1))$ nach $f_t(H)$ gebildet:

$$\frac{\partial g_H}{\partial f_t(H)} = \frac{\partial g_H}{\partial E_H(H,t)} \frac{\partial E_H(H,t)}{\partial f_t(H)} =$$

$$\begin{cases} \frac{1}{2n}\left(\frac{\partial E_H(H,t)}{\partial f_t(H)}\right) & : \quad m(H,t+1) = 0 \\ 0 & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) = 0 \\ \frac{1}{m}\left(\frac{\partial E_H(H,t)}{\partial f_t(H)}\right) & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) \neq 0 \end{cases}$$

$$= \begin{cases} > 0 & : \quad m(H,t+1) = 0 \qquad \text{mit Hilfssatz 2.11} \\ = 0 & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) = 0 \\ > 0 & : \quad m(H,t+1) \neq 0 \quad \text{und} \quad E_H(H,t) \neq 0 \text{ mit Hilfssatz 2.11} \end{cases}$$

$\Rightarrow \frac{\partial g_H}{\partial f_t(H)} \geq 0$. Also ist g_H monoton wachsend bezüglich $f_t(H)$.
b) Der Beweis, dass g_S bezüglich $f_t(H)$ monoton wachsend ist, erfolgt entsprechend der Vorgehensweise in a). Dabei wird E_H durch E_S ersetzt. □

Mit diesem Hilfssatz wurde bewiesen, dass die Vorhersagegenauigkeit von E_H und E_S mit steigender relativen Fitness eines Schemas wächst. Dies ist ein interessantes Ergebnis, da es bedeutet, dass, je fitter ein Schema ist, desto besser seine zukünftige Vertreteranzahl durch E_H und E_S approximiert wird.

Mit g_H (bzw. g_S) können G_H (bzw. G_S) als ein Maß für die durchschnittliche Güte der Approximation von $m(H,t+1)$ durch E_H (bzw. E_S) berechnet werden:

$$G_H = \frac{1}{3^l - 1} \sum_{H \in \Xi \setminus H^\star} g_H$$

$$G_S = \frac{1}{3^l - 1} \sum_{H \in \Xi \setminus H^\star} g_S.$$

Abgesehen von g_H und g_S wird der Term $v := v(g_H, g_S, m(H,t+1))$ berechnet, welcher ein Maß für die verbesserte Approximation von $m(H,t+1)$ ist, die erreicht wird, wenn $E_S(H,t)$ anstelle von $E_H(H,t)$ verwendet wird. Für ein Schema H berechnet sich v folgendermaßen:

$$v := \begin{cases} g_H - g_S & : \quad \text{für } m(H,t+1) = 0 \text{ oder } g_H > 1 \\ 2 - (g_H + g_S) & : \quad \text{für } g_H < 1 \text{ und } g_S > 1 \\ g_S - g_H & : \quad \text{für } g_H < 1 \text{ und } g_S < 1. \end{cases} \tag{3.1}$$

Die durchschnittliche Verbesserung V der Approximationsgüte kann dann wie folgt berechnet werden:

$$V = \frac{1}{3^l - 1} \sum_{H \in \Xi \setminus H^\star} v.$$

Weiterhin wird die Anzahl der Fälle $\ddot{u}_H$ (bzw. $\ddot{u}_S$) gezählt, in denen $E_H(H,t)$ (bzw. $E_S(H,t)$) den Wert $m(H,t+1)$ überschreitet, wenn also

$$E_H(H,t) > m(H,t+1) \quad \big(\text{bzw. } E_S(H,t) > m(H,t+1)\big)$$

eintritt. Mit $\ddot{u}_H$ und $\ddot{u}_S$ kann dann die durchschnittliche Anzahl der Überschreitungen $\ddot{U}_H$ und $\ddot{U}_S$ berechnet werden:

$$\ddot{U}_H = \frac{\ddot{u}_H}{3^l - 1} \quad \text{bzw.} \quad \ddot{U}_S = \frac{\ddot{u}_S}{3^l - 1}.$$

3.2. Allgemeine Ergebnisse

In diesem Abschnitt werden die ersten numerischen Ergebnisse vorgestellt. Es wird die Optimierungsaufgabe

$$z(x) = \sin(x) \longrightarrow max \qquad \text{mit} \quad x \in \mathbb{Z}$$

betrachtet. Abhängig von dem interessierenden Ergebniss werden unterschiedlich viele Simulationen mit variierenden N durchgeführt. Wird nichts anders lautendes erwähnt, so werden folgende Parameterbelegungen in den Simulationen verwendet:

$$\begin{aligned} p_c &= 0.9 \\ p_m &= 0.01 \\ l &= 5 \\ 2n &= 10. \end{aligned}$$

Die Wahl der Werte für p_c und p_m sind durch die Erfahrungswerte aus der Praxis und theoretischer Untersuchungen motiviert, welche $p_c \approx 1$ und $p_m \approx 0$ empfehlen. Für repräsentable Simulationsergebnisse sollen möglichst viele Schemen untersucht werden. Mit wachsender Chromosomenlänge steigt jedoch die Schemenanzahl derart, dass die Laufzeit der Simulationen erheblich verlängert wird. In vorangegangenen Testsimulationen hat sich $l = 5$ als zeitlich akzeptabel profiliert und bietet mit $3^5 - 1 = 242$ Schemen eine ausreichend große Anzahl an Werten, um aussagekräftige Ergebnisse zu liefern.

Die Laufzeit einer Simulation wird nicht nur durch eine große Anzahl an Schemen verlängert, sondern auch durch eine große Anzahl an Individuen. Je mehr Individuen, desto größer die Anzahl an Berechnungen. Jedoch erhöht eine geringe Anzahl an Individuen die Wahrscheinlichkeit, das der genetische Algorithmus in einem Submaximum „hängen“ bleibt.

G_H	G_S	V	$\ddot{U}_H$	$\ddot{U}_S$
94.29%	96.60%	0.56%	7.23	206.39

Tabelle 3.1.: Durchschnittswerte für G_H, G_S, V, $\ddot{U}_H$ und $\ddot{U}_S$ aus 20 Simulation mit $N = 500$. Das Optimum wurde in 7 Simulationen gefunden.

Möchte man mit einer Wahrscheinlichkeit von 99%, dass bei der Initialisierung der Populations jedes Allel an jedem Gen vertreten ist, um ein „Hängenbleiben“ zu vermeiden, dann muss nach einer in [20] gegebenen Formel

$$2n \geq 6.21 \qquad \text{also} \qquad n > 4$$

gewählt werden. Um die Gefahr des „Hängenbleibens“ noch weiter zu verringern wird $n = 5$ gewählt.

Durchschnittliche Approximationsgüte von $\mathbf{E_H}$ und $\mathbf{E_S}$

Um erste Werte für G_H, G_S, V, $\ddot{U}_H$ und $\ddot{U}_S$ zu erhalten, wurden 20 Simulationen mit jeweils 500 Generationen durchgeführt. Die Ergebnisse sind in Tabelle 3.1 aufgelistet.

Der Schemensatz 2.3 liefert auch ohne die vollständige Betrachtung aller Szenarien, in denen ein Schema in der Folgepopulation Vertreter besitzt eine sehr gute Annäherung an $m(H, t+1)$. Die durchschnittliche Approximationsgüte von $E_S(H, t)$ liegt jedoch mehr als 2% über der von $E_H(H, t)$ und die tatsächliche Verbesserung der Approximation V beträgt 0.56%. Zu beachten ist weiterhin, dass $E_S(H, t)$ durchschnittlich in 206.39 von 242 Fällen über dem Wert $m(H, t+1)$ liegt.

Korrelation zwischen $\mathbf{E_H}$ (bzw. $\mathbf{E_S}$) und $\mathbf{m(H, t+1)}$

Zwischen E_H und $m(H, t+1)$ bzw. zwischen E_S und $m(H, t+1)$ sollte ein linearer Zusammenhang erkennbar sein. Je höher die Voraussage der Vertreteranzahl eines Schemas, desto höher sollte die tatsächlich eingetretene Vertreteranzahl sein. Es wurden Daten aus einer Simulation mit 10 Generationen extrahiert und Werte für die Regressionsgeraden r_H und r_S

$$r_H : y_1 = a_1 x_1 + b_1$$

$$r_S : y_2 = a_2 x_2 + b_2$$

berechnet. Die Regressionsgeraden sollen die lineare Abhängigkeit von $m(H, t+1)$ von E_H bzw. E_S beschreiben. Tabelle 3.2 fasst die Ergebnisse und Berechnungen zusammen.

Die Werte der Korrelationskoeffizienten $\rho_1 = 0.98$ und $\rho_2 = 0.99$ bestätigen, dass $m(H, t+1)$ als linear abhängig von $E_H(H, t)$ und als linear abängig von $E_S(H, t)$ betrachtet werden kann. Es besteht also tatsächlich ein linearer Zusammenhang zwischen $m(H, t+1)$ und E_H bzw. E_S. Je größer E_H und E_S, desto größer wird der Wert von $m(H, t+1)$ sein.

	Summen		Anzahl der Werte, die in die Summen eingehen	Durchschnitt	
$m(H,t+1)$ $= k$	${S_H}^k = \sum_{\substack{H: \\ m(H,t+1)=k}} E_H$	${S_S}^k = \sum_{\substack{H: \\ m(H,t+1)=k}} E_S$	A_k	$\frac{{S_H}^k}{A_k} = E_H(H,t)$	$\frac{{S_S}^k}{A_k} = E_S(H,t)$
0	22.02	209.88	1530	0.014	0.137
1	56.21	188.98	293	0.192	0.645
2	148.33	304.09	218	0.680	1.395
3	119.09	213.09	72	1.654	2.96
4	139.90	200.53	44	3.18	4.557
5	119.12	144.03	23	5.179	6.262
6	285.28	422.71	72	3.962	5.871
7	169.09	210.03	30	5.636	7.001
8	320.98	352.71	45	7.133	7.838
9	564.67	608.42	68	8.304	8.947
10	245.52	245.53	25	9.821	9.821

Berechnung der Regressionsgeraden r_H und r_S:

i	$y_i = a_i x_i + b_i$	a_i	b_i	ρ_i	ρ_i^2
1	r_H	0.96	0.99	0.98	0.96
2	r_S	0.97	0.09	0.99	0.98

Tabelle 3.2.: Korrelation zwischen $m(H,t+1)$ und $E_H(H,t)$ (bzw. $E_S(H,t)$). Es wurde eine Simulation mit $N = 10$ durchgeführt.

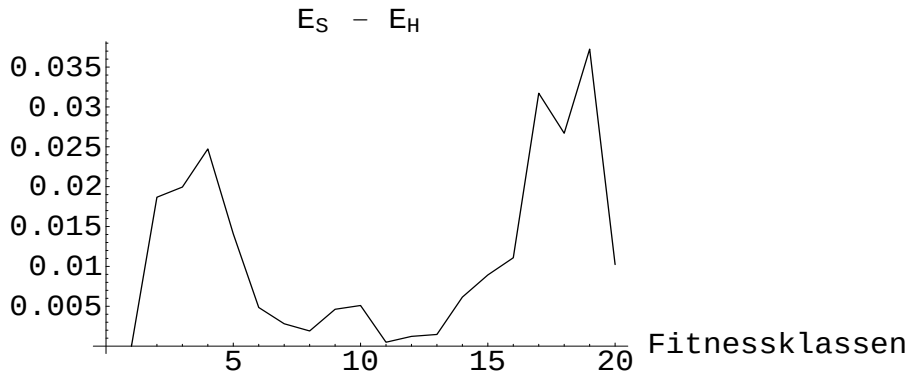

Abbildung 3.1.: Die Differenz $E_S(H,t) - E_H(H,t)$ in Abhängigkeit von der Fitness eines Schemas. Es wurde eine Simulation mit $N = 200$ durchgeführt. Die Schemen sind in 20 Fitnessklassen eingeteilt.

Differenz $\mathbf{E_S - E_H}$

Weiterhin ist es interessant zu untersuchen, ob die durchschnittliche absolute Differenz $(E_S(H,t) - E_H(H,t))$ möglicherweise von der Fitness des Schemas H abhängig ist, z. B. in der Art, dass die Differenz mit steigender Fitness wächst. Wäre dies der Fall, so würde es darauf hindeuten, dass $E_S(H,t)$ mit einer größeren Rate monoton wächst als $E_H(H,t)$. Um dies numerisch zu untersuchen, wurden die Schemen in 20 Klassen $K_i \quad (i = 1, \ldots, 20)$ bezüglich ihrer relativen Fitness eingeteilt:

$$H \in K_i \Leftrightarrow f_t(H) \in \left[\frac{i-1}{20}, \frac{i}{20}\right) \qquad i = 1, \ldots, 20 \quad \forall H \in \Xi \backslash H^\star.$$

Eine Simulation mit 200 Generationen wurde durchgeführt. Die Ergebnisse sind in Abbildung 3.1 dargestellt. Die Abbildung zeigt die verbundenen Werte der Differenz $(E_H(H,t) - E_S(H,t))$ in Abhängigkeit von den Fitnessklassen. Sie zeigt, dass die Differenz bezüglich $f_t(H)$ nicht monoton ist. Das bedeutet, dass der Abstand zwischen E_H und E_S nicht mit steigender relativer Fitness eines Schemas wächst.

Einfluss der einzelnen Terme $\mathbf{E_{Co_1}}$, $\mathbf{E_{Co_2}}$ und $\mathbf{E_{Mut}}$ auf $\mathbf{E_S}$

In diesem Unterabschnitt soll numerisch untersucht werden, welchen Einfluss die Terme E_{Co_1}, E_{Co_2} und E_{Mut} auf E_S besitzen. Es wird dabei der prozentuale Anteil dieser Terme an dem Wert von E_S betrachtet. Aus den Ergebnissen können Rückschlüsse gezogen werden, wie wahrscheinlich die

Angaben in [%]	E_{Co_1}	E_{Co_2}	E_{Mut}
Durchschnittlicher Anteil an E_S	8.2	11.3	80.5
Durchschnittlicher Anteil an E_S, wenn $E_{Co_1} + E_{Co_2} > 0$	41.6	55.6	2.8

Tabelle 3.3.: Durchschnittliche Werte für den Anteil von E_{Co_1}, E_{Co_2} und E_{Mut} an E_S. Es wurden 20 Simulationen mit $N = 500$ durchgeführt.

in Abschnitt 2.4.1 untersuchten Szenarien sind. Das heißt, es wäre erkennbar, wie wahrscheinlich es ist, dass ein Vertreter von H aus dem Kreuztausch zweier Vertreter der Teilschemen H_k und $\hat{H}_k$ resultiert (E_{Co_2}) und wie wahrscheinlich es ist, dass ein Vertreter von H aus dem Kreuztausch mit mindestens einem Vertreter von H resultiert (E_{Co_1}). Aus dem prozentualem Anteil von E_{Mut} ist dann ersichtlich, wie häufig ein Vertreter von H durch die Mutation in die Population gelangt. Um dies zu untersuchen, werden 20 Simulationen durchgeführt. Die Ergebnisse sind in Tabelle 3.3 aufgelistet.

Auf Grund der meist klein gewählten Mutationswahrscheinlichkeit ist es sehr unwahrscheinlich, dass ein Element, welches kein Vertreter von H ist, zu einem Vertreter von H mutiert wird. Daher schätzt man den Anteil von E_{Mut} als sehr gering ein. Deshalb ist es verwunderlich, dass der durchschnittliche prozentuale Anteil von E_{Mut} mit 80.5% sehr groß ist. Um dieses Phänomenen zu erklären, erfolgt ein Vorgriff auf Simulationen, die zu einem späteren Zeitpunkt in der Arbeit erwähnt werden (Tabelle 3.6). Anhand der Ergebnisse dieser Simulationen ist bekannt, dass die Anzahl der Schemen, die keine Vertreter in der Population P_t besitzen, sehr hoch ist. Besitzt ein Schema keine Vertreter in der Population P_t, so ist seine relative Fitness $f_t(H)$ notwendigerweise gleich Null. Nach dem Hilfssatz 2.9 muss dann aber $E_{Mut} > 0$ gelten. Der Term E_{Co_1} wird ebenfalls Null. Über den Wert von $E_{Co_2}(H,t)$, wenn $f_t(H,t) = 0$ gilt ist nichts bekannt. Somit setzt sich E_S nur additiv aus E_{Co_2} und E_{Mut} zusammen, was wiederum den prozentualen Anteil von beiden gegenüber E_{Co_2} erhöht. Dadurch ist der große durchschnittliche Anteil von E_{Mut} an E_S von über 80% erklärbar. Zum Vergleich wurden die Simulationen nochmals durchgeführt, wobei die prozentualen Anteile von E_{Co_1}, E_{Co_2} und E_{Mut} nur gemessen wurden, wenn $E_{Co_1} + E_{Co_2} > 0$ galt. Die Ergebnisse sind ebenfalls in Tabelle 3.3 zu sehen. Sie bestätigen, dass der Anteil von E_{Mut} an E_S, wie vermutet, sehr gering ist. Weiterhin ist ersichtlich, dass für ein Schema mit positiver relativer Fitness es am wahrscheinlichsten ist, durch den Kreuztauschoperator Vertreter in der Population zu erhalten.

3.3. Nutzung von verschiedenen Zielfunktionen

In diesem Abschnitt wird der Einfluss, den die Wahl der Zielfunktion auf $E_H(H,t)$ und $E_S(H,t)$ hat, untersucht. Im Allgemeinen hat der An-

wender von genetischen Algorithmen keine Wahlmöglichkeit, sondern die Zielfunktion ist ihm gegeben. Die Leistung eines genetischen Algorithmus wird wesentlich dadurch bestimmt, wie gut die Kodierung der Elemente des Suchraumes auf die Zielfunktion abgestimmt ist. Die Kodierung sollte in irgendeiner Art den Charakter der Zielfunktion widerspiegeln. Ist die Zielfunktion monoton, so sollte eine wachsende Anzahl von Nullen in den Chromosomen mit einem wachsenden oder fallenden Zielfunktionswert korrespondieren. Sind die Zielfunktionswerte zufällig gewählt, gäbe es zwischen der in dieser Arbeit verwendeten Kodierung und der Zielfunktion keinen Zusammenhang und der genetische Algorithmus würde einer zufälligen Suche entsprechen. Dies ist dadurch begründet, dass die Kombination zweier fitter Individuen nicht notwendig ein mindestens genauso fittes Individuum erzeugt. Mit diesem Abschnitt soll gezeigt werden, wie ausschlaggebend das Zusammenspiel zwischen Zielfunktion und Kodierung auf die Leistung eines genetischen Algorithmus ist.

Es werden zwei verschiedene Zielfunktionen betrachtet, die mehr oder weniger mit der vorliegenden Kodierung korrespondieren. Zum einen die Zielfunktion $z_1(x) = \sin(x)$ und zum anderen wurde die Königswegfunktion[1] $z_2(x)$ betrachtet. Die Königswegfunktion arbeitet mit einer Menge von Schemen

$$\{S_1, S_2, \ldots, S_m\} \quad \text{mit} \quad S_i \in \Xi \backslash H^\star \;\; \forall i \in \{1, \ldots, m\}$$

und ist folgendermaßen definiert:

$$z_2(x) = \sum_{i=1}^{m} \omega_i \nu_i(x) \qquad \text{mit} \qquad \nu_i(x) = \begin{cases} 1 & : \quad x \in S_i \\ 0 & : \quad \text{sonst} \end{cases} .$$

Die ω_i $(i = 1, \ldots, m)$ sind nichtnegative Gewichte. Gilt für alle Gewichte $\omega_i = 1$, so ist der Zielfunktionswert $z_2(x)$ die Anzahl der Schemen, für die x Vertreter ist.

Das Optimum von $z_1(x)$ ist das Individuum $x_{opt} = -11$ mit dem Chromosom $(1, 1, 0, 1, 1)$. Die Wahl der Schemen und Gewichte für die Königwegsfunktion $z_2(x)$ wurde derart getroffen, dass $x_{opt} = -11$ auch das Optimum von $z_2(x)$ ist. Die Menge der Schemen und die zugehörigen Gewichte sind in der folgenden Tabelle aufgelistet.

i	Schema S_i	Gewicht ω_i
1	$(1, \star, \star, \star, \star)$	1
2	$(\star, 1, \star, \star, \star)$	1
3	$(\star, \star, 0, \star, \star)$	1
4	$(\star, \star, \star, 1, \star)$	1
5	$(\star, \star, \star, \star, 1)$	1

[1] aus dem Englischen: „Royalroad function“

	$z_1(x)$	$z_2(x)$
Anzahl der Simulationen, in denen das Optimum gefunden wurde	39	100
Anzahl der Simulationen, in denen das Optimum in der Anfangspopulation P_0 enthalten war	24	26
Durchschnittliche Anzahl von Generationen, die benötigt wurden das Optimum zu finden, wenn es überhaupt gefunden wurde und dabei nicht in P_0 vorhanden war	87.9333	5.0541

Tabelle 3.4.: Ergebnisse aus 100 Simulationen mit $N_{max} = 500$ zum Vergleich von $z_1(x)$ und $z_2(x)$ bzgl. der benötigten Generationsanzahl zum erstmaligen Auftretens des Optimums.

Anzahl der Generationen, die zum Finden des Optimums benötigt werden

Zu Beginn ist es interessant zu untersuchen, ob die Zielfunktion einen Einfluss auf die Anzahl der Generationen hat, die benötigt werden, um das Optimum zu finden. Das Optimum gilt als gefunden, wenn wenigstens ein Individuum der Population das Optimum repräsentiert. Dabei zählt nur das erstmalige Auftreten des Optimums in der Population, es bleibt außer Acht, ob es in der Population über mehrere Generationen bleibt. Da jedes Schema, welches der Königswegfunktion übergeben wurde, das Optimum x_{opt} als Vertreter besitzt, wird vermutlich bei der Königswegfunktion das Optimum schneller gefunden werden, als bei der Zielfunktion $z_1(x)$. Dies liegt darin begründet, dass die Königswegfunktion einen Weg zum Optimum bereit legt, da nur diejenigen Individuen einen positiven Fitnesswert zugewiesen bekommen, die mit wenigstens einem Allel mit dem Allel am betreffenden Gen des Optimums übereinstimmen. Die Vermutung, dass das Optimum bei einem genetischen Algorithmus mit $z_2(x)$ als Zielfunktion schneller gefunden wird, als bei der Nutzung von $z_1(x)$ als Zielfunktion soll untersucht werden. Dafür wurden jeweils 100 Simulationen pro Zielfunktion durchgeführt. Die Ergebnisse sind in Tabelle 3.4 zusammengefasst.

Die Simulationsergebnisse scheinen die Vermutung zu bestätigen. Ein genetischer Algorithmus mit der Königswegfunktion $z_2(x)$ als Zielfunktion findet das Optimum schneller als ein genetischer Algorithmus mit $z_1(x)$ als Zielfunktion. Das liegt daran, dass ein Algorithmus mit $z_1(x)$ das Optimum aus der Menge der Elemente mit hohen Fitnesswerten herausfinden, beim Kreuztausch zusammenfügen oder entdecken muss. Dies geschieht nur mit der Hilfe der Fitnesswerte der Individuen, der zufällig gewählten Kreuzungstelle und mit dem zufälligen Stattfinden einer Mutation. Bei

	G_H	G_S	V	$\ddot{U}_H$	$\ddot{U}_S$
$z_1(x)$	94.38%	96.34%	0.54%	7.19	207
$z_2(x)$	76.32%	94.53%	8.02%	21.56	165.26

Tabelle 3.5.: Simulationsergebnisse zum Vergleich von $z_1(x)$ und $z_2(x)$ bezüglich der Genauigkeit von E_H und E_S. Es wurde pro Zielfunktion eine Simulation mit $N = 100$ durchgeführt.

einem genetischen Algorithmus, welcher jedoch $z_2(x)$ verwendet, besitzt jedes Individuum mit positiven Fitnesswert wenigstens ein Allel des Optimums am richtigen Gen im Chromosom. Werden zwei Individuen mit positiven Fitnesswerten beim Kreuztausch gekreuzt, so entstehen Individuen, wovon wenigstens einer mindestens den Fitnesswert des schlechtesten Elternteils besitzt. Eine gleichzeitige Verschlechterung der Fitnesswerte beider gekreuzten Nachkommen ist dadurch, im Gegensatz zu Simulationen mit der Zielfunktion $z_1(x)$, nicht möglich. Wird $z_2(x)$ als Zielfunktion verwendet, so ist es auch möglich, dass das Optimum nicht erreicht wird. Dies kann geschehen, wenn $p_m = 0$ gesetzt wird und alle Individuen an einem bestimmten Gen das gleiche Allel besitzen, welches nicht mit dem Allel am betreffenden Gen des Optimums übereinstimmt.

Die Simulationsergebnisse verdeutlichen die Notwendigkeit einer guten Zusammenarbeit zwischen Zielfunktion und Kodierung. Mit der Zielfunktion z_1 wurde bei der verwendeten Kodierung nur in 15% der Simulationen das Optimum gefunden und dann erst in der 88. Generation. Im Vergleich zu z_2 ist das ein enttäuschendes Ergebnis. Die Königswegfunktion z_2 kann durch ihre Definition sehr genau an die zu Grunde liegende Kodierung angepasst werden. Dies erklärt die guten Simulationsergebnisse mit z_2.

Einfluss auf G_H und G_S

Da in Simulationen, in denen die Königswegfunktion $z_2(x)$ als Zielfunktion benutzt wird, das Optimum öfter und erheblich schneller gefunden wird, liegt es nahe zu vermuten, dass Vertreter des Optimums die Population zu einem früheren Zeitpunkt dominieren, als bei der Nutzung von $z_1(x)$. Das ist dadurch begründet, dass in den Simulationen mit der letzteren Zielfunktion das Optimum erst sehr spät oder gar nicht gefunden wird (Tabelle 3.4). Man kann also vermuten, dass in genetischen Algorithmen mit $z_2(x)$ mehr Vertreter von Schemen mit hohen durchschnittlichen Fitnesswerten vorhanden sind, als in Simulationen, in denen $z_1(x)$ verwendet wird.

Nach dem Hilfssatz 3.1 sind g_H und g_S monoton wachsend bezüglich der relativen Fitness eines Schemas. Dadurch sind vermutlich auf Grund der größeren Anzahl an sehr fitten Schemen die Werte von G_H und G_S bei der Verwendung von $z_2(x)$ im Vergleich mit $z_1(x)$ höher. Um diese Vermutung zu bestätigen, wurde pro Zielfunktion eine Simulation mit jeweils 100 Generationen durchgeführt. Die Ergebnisse sind in Tabelle 3.5 aufgelistet.

	$z_1(x)$			$z_2(x)$		
	K_1	K_2	K_3	K_1	K_2	K_3
Anzahl von Schemen in den einzelnen Klassen	211.08	0.19	30.73	206.76	24.96	10.28
davon mit Vertretern in der Population	22.54	0.19	30.73	84.62	24.96	10.28
G_H in [%] pro Klasse	82.02	0.06	12.3	65.69	6.81	3.81
G_S in [%] pro Klasse	83.08	0.08	13.18	78.47	11.52	4.54
Durchschnittswerte von G_H in [%] pro Klasse	94	75	97	77	66	90
Durchschnittswerte von G_S in [%] pro Klasse	95	105	104	92	112	107
$\ddot{U}_H$ pro Klasse	0.69	0.03	6.47	15.69	2.5	3.37
$\ddot{U}_S$ pro Klasse	189.9	0.1	17.09	142.15	15.09	8.02

Tabelle 3.6.: Simulationsergebnisse zum Vergleich von $z_1(x)$ und $z_2(x)$ bzgl. der Genauigkeit von E_H und E_S, wobei die Schemen in die Fitnessklassen K_1, K_2 und K_3 eingeteilt sind. $N = 100$

Entgegen der Vermutung sind die Werte von G_H und G_S bei der Nutzung der Königswegfunktion $z_2(x)$ geringer. Dabei erweist sich E_S stabiler als E_H gegenüber $z_2(x)$, da G_S nur um 1.81%, G_H aber um 18.06% sinkt. Die Verbesserung V der Approximationsgüte, die erreicht wird, wenn E_S anstelle von E_H verwendet wird, steigt bei der Nutzung von $z_2(x)$ von 0.54% auf 8.02%.

Um einen Hinweis auf den Grund der unerwartet geringen Werte von G_H und G_S bei der Verwendung der Königswegfunktion zu erhalten, werden die beiden Simulationen noch einmal durchgeführt und dabei die Schemen in drei Fitnessklassen (K_1, K_2 und K_3) eingeteilt:

$$\begin{array}{lcll} H \in K_1 & \Leftrightarrow & f_t(H) < \frac{1}{3} & \\ H \in K_2 & \Leftrightarrow & \frac{1}{3} \leq f_t(H) < \frac{2}{3} & \forall H \in \Xi \backslash H^\star. \\ H \in K_3 & \Leftrightarrow & f_t(H) \geq \frac{2}{3}. & \end{array}$$

Nun kann man die Anzahl der Schemen in jeder Klasse und den Beitrag jeder Klasse zu G_H und G_S aufschlüsseln. Die Ergebnisse der Simulationen sind in Tabelle 3.6 zusammengefasst.

Überraschenderweise ist die Anzahl an Schemen mit hoher Fitness in der Simulation mit der Zielfunktion $z_1(x)$ dreimal so hoch, als in der Simulation mit $z_2(x)$. Damit war die Vermutung zu Beginn des Unterabschnittes falsch, welche beinhaltete, dass die Anzahl an sehr fitten Schemen bei genetischen Algorithmen mit der Königswegfunktion höher sein müsste, als bei Algorithmen mit $z_1(x)$. Doch dadurch, dass die Mehrzahl an Schemen in Algorithmen mit $z_2(x)$ eine geringe bis mittlere Fitness besitzen, sind die gesunkenen Werte von G_H und G_S erklärbar. Denn je kleiner die Fitness eines Schemas H, desto kleiner sind g_H und g_S (siehe Hilfssatz 3.1) und somit auch G_H und G_S.

Einfluss auf V

Es wurde festgestellt, dass der Wert für V bei Benutzung von $z_2(x)$ anstelle von $z_1(x)$ von 0.54% auf 8.02% steigt (Tabelle 3.5). In diesem Unterabschnitt soll ein Erklärungsversuch für dieses überraschende Ergebnis gegeben werden.

Aus den Daten von Tabelle 3.6 ist ersichtlich, dass die durchschnittliche Anzahl der Überschreitungen $\ddot{U}_H$, also die Anzahl der Fälle in denen E_H größer als die tatsächlich eingetretene Anzahl von Vertretern eines Schemas H in der Population war, beim Übergang von $z_1(x)$ zu $z_2(x)$ steigt. Der Wert für $\ddot{U}_S$ sinkt jedoch stark. Jede Überschreitung ($g_H > 1$ oder $g_S > 1$) geht negativ in den Term für v ein (3.1). Wird $z_2(x)$ als Zielfunktion verwendet, geht also, wegen dem gesunkenen Wert für $\ddot{U}_S$, ein bestimmter Betrag nicht mehr negativ in den Wert für v und somit in den Wert für V ein. Dies könnte ein Grund für den größeren Wert für V bei der Benutzung von $z_2(x)$ anstelle von $z_1(x)$ sein.

Offene Fragen

Die Simulationsergebnisse in Tabelle 3.5 und Tabelle 3.6 werfen eine Reihe weiterer interessanter Fragen auf, die in der hier vorliegenden Arbeit entweder nicht untersucht oder keine Hinweise auf deren Antworten gefunden werden konnten. Einige dieser Fragen sind im Folgenden aufgelistet:

- Warum sinkt G_H stärker als G_S bei der Verwendung von $z_2(x)$?
- Warum sinkt bei der Verwendung von $z_2(x)$ der Wert für $\ddot{U}_S$ in K_1 und K_2 so stark, und warum steigt $\ddot{U}_H$ in K_1 und K_2?
- Warum ist die Anzahl von Schemen mit geringer bis mittlerer Fitness so groß, wenn $z_2(x)$ verwendet wird?
- Warum ist die Gesamtanzahl an Schemen, die bei der Zielfunktion $z_2(x)$ in der Population vorhanden sind, gegenüber der Simulation mit $z_1(x)$ mehr als doppelt so groß?
- Wie groß ist die Differenz zwischen E_H und $E_{Co_1} + E_{Co_2}$? Daraus könnte eine Vorstellung gewonnen werden, wie oft ein Vertreter eines Schemas durch die Kreuzung zweier Individuen, die Vertreter von $H_k \backslash H$ oder $\hat{H}_k \backslash H$ sind, entsteht.

3.4. Optimierung der Mutationswahrscheinlichkeit

Der einfache genetische Algorithmus besitzt die Kontrollparameter Kreuztausch- und Mutationswahrscheinlichkeit, welche einen großen Einfluss auf die Folge der Populationen, und somit auf das Finden der Optimallösung, besitzen.

Dieser Abschnitt wird sich mit der optimalen Wahl der Mutationswahrscheinlichkeit p_m in einem einfachen genetischen Algorithmus beschäftigen.

Zunächst wird das Optimierungsproblem formuliert und ein Lösungsansatz vorgestellt. Anschließend wird die Optimierung für einen Spezialfall durchgeführt.

Die Mutationswahrscheinlichkeit hat Einfluss auf die Geschwindigkeit, damit ist die benötigte Anzahl an Generationen gemeint, mit der das Optimum gefunden wird und auf die Rate, mit der sich die Anzahl der Vertreter eines Schemas mit hoher Fitness vergrößert. Ist die Mutationswahrscheinlichkeit sehr groß gewählt, so werden sich die Individuen der Population auf Grund der hohen Zahl von Mutationen untereinander mehr unterscheiden, als bei einer kleiner Mutationswahrscheinlichkeit. Durch die erhöhte Heterogenität der Population kann das Optimum schneller gefunden werden. Jedoch ist es auch wahrscheinlicher, dass aus der Mutationtion eines Vertreters des optimalen Schemas nur Elemente resultieren, die nicht Vertreter des optimalen Schemas sind. Mit „optimalen Schema“ ist dabei Folgendes gemeint:

Bezeichnung 3.2 *Das optimale Schema H_{opt} ist das Schema, welches als einzigen Vertreter das Optimum des zu Grunde liegenden Problems besitzt.*

Einige Auswirkungen der Mutationswahrscheinlichkeit sind im Folgenden aufgelistet. Sie besitzt Einfluss auf:

- die Geschwindigkeit, mit der die Anzahl der Individuen mit hoher Fitness in der Population wächst.
- die Geschwindigkeit, mit der das Optimum gefunden wird.
- die Wahrscheinlichkeit, mit der das gefundene Optimum in der Population bleibt.

Diese Merkmale werden als Funktionen von p_m formuliert:

f_1 : $\mathbb{R}[0,1] \longrightarrow \mathbb{R}$
$\hat{=}$ Anzahl der zu erwartenden Vertreter eines guten Schemas in der Folgepopulation, beschrieben durch $E_H(H,t)$ oder $E_S(H,t)$.

f_2 : $\mathbb{R}[0,1] \longrightarrow \mathbb{N}$
$\hat{=}$ Anzahl der Generationen, die der einfache genetische Algorithmus benötigt, um das Optimum zu finden.

f_3 : $\mathbb{R}[0,1] \longrightarrow \mathbb{R}[0,1]$
$\hat{=}$ Wahrscheinlichkeit, mit der das gefundene Optimum während der Simulation in der Population bleibt.

Dabei sollen f_1 und f_3 maximiert und f_2 minimiert werden, da die Vertreteranzahl von Schemen mit hohen Fitnesswerten in der Population möglichst schnell steigen und die Wahrscheinlichkeit, dass das gefundene Optimum in Population bleibt möglichst groß sein soll. Die Anzahl an Generationen bis zum erstmaligen Auftreten des Optimums in der Population sollte dagegen möglichst gering sein.

Der Schemensatz 2.3 kann für die Optimierung der Mutationswahrscheinlichkeit des einfachen genetischen Algorithmus verwendet werden. Es ist zu vermuten, dass eine qualitativ bessere Lösung bei der Optimierung erreicht wird, wenn der modifizierte Schemensatz 2.7 verwendet wird, da dieser die Vertreteranzahl eines Schemas in der Folgepopulation genauer vorhersagt.

Zu Beginn des Abschnittes wurde vermutet, dass f_2 und f_3 monoton fallend sind, dass also das Optimum mit steigender Mutationswahrscheinlichkeit zwar schneller gefunden wird, jedoch weniger wahrscheinlich in der Population bleibt. Um diese Vermutung zu bestätigen und zusätzliche Hinweise auf den Verlauf von f_2 und f_3 und auf weitere Auswirkungen der Mutationswahrscheinlichkeit zu erhalten, werden für verschiedene Mutationswahrscheinlichkeiten jeweils 10 Simulationen durchgeführt. Die Ergebnisse sind in Abbildung 3.2 dargestellt.

Es ist erkennbar, dass f_2 und f_3 tatsächlich für größere Werte von p_m sinken. Wird f_2 mit (-1) multipliziert, so erhält man das Vektormaximierungsproblem

$$\begin{bmatrix} f_1(p_m) \\ -f_2(p_m) \\ f_3(p_m) \end{bmatrix} \longrightarrow \max \qquad \text{für} \quad p_m \in [0,1]. \tag{3.2}$$

Die Zahlenbereiche der Wertevorräte von f_1, f_2 und f_3 sind nicht identisch. Weiterhin muss angenommen werden, dass die Funktionen im Allgemeinen nichtlinear sind.

Eine Optimallösung von (3.2) kann mit den folgenden Schritten berechnet werden:

1. Approximation von f_2 und f_3 durch geeignete Kurven.
2. Lösen des Vektormaximierungsproblems (3.2) zum Beispiel durch die Methode der schrittweisen Konzessionen [8] oder die Methode der ϵ-Beschränkungen [10].

In dieser Arbeit wird die Methode der schrittweisen Konzessionen verwendet und soll daher hier beschrieben werden. Für die Vektormaximierungsaufgabe über den zulässigen Bereich B

$$f_i(x) \longrightarrow \max_{x \in B} \qquad i = 1, \ldots, m$$

wird für die Methode der schrittweisen Konzessionen eine Rangfolge

$$(f_{i_1}, \ldots, f_{i_m})$$

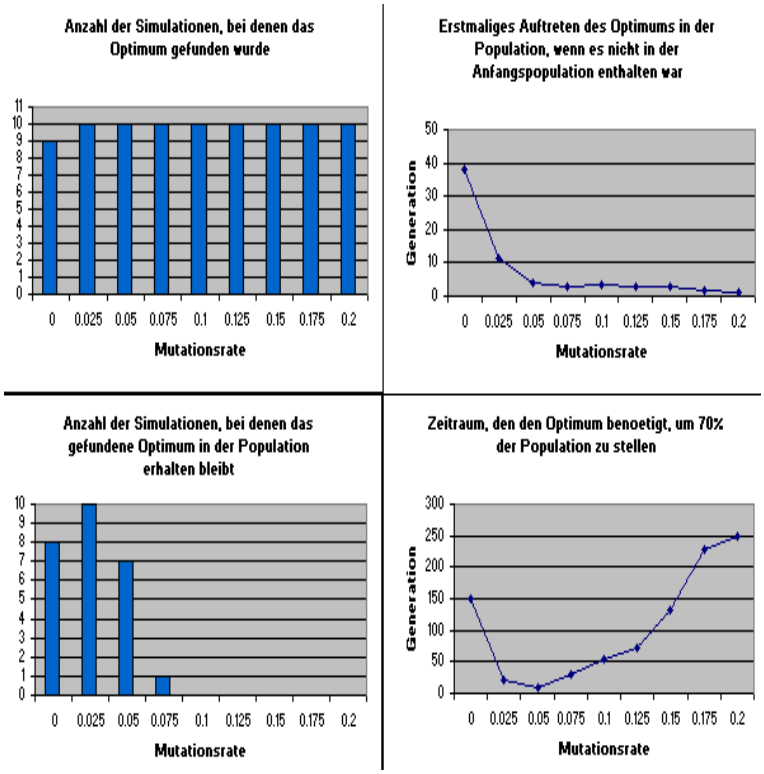

Abbildung 3.2.: Ergebnisse aus jeweils 10 Simulationen für $p_m = 0.0,\ 0.025,\ 0.05,\ 0.075,\ 0.1,\ 0.125,\ 0.15,\ 0.175$ und 0.2.

der Funktionen f_i gebildet. Dann wird die erste Funktion f_{i_1} der Rangfolge maximiert und der zulässige Bereich B verkleinert. Der Maximalwert von f_{i_1} wird mit $f_{i_1}^{max}$ bezeichnet. Der zulässige Bereich wird beschränkt, indem mit einem geeigneten α_{i_1} eine untere Schranke $\alpha_{i_1} f_{i_1}^{max}$ für den Funktionswert $f_{i_1}^{\star}(x_{opt})$ der Optimallösung x_{opt} bestimmt wird, die nicht unterschritten werden darf. Anschließend wird die nächste Funktion f_{i_2} der Rangfolge über den eingeschränkten zulässigen Bereich maximiert und der Funktionswert $f_{i_2}^{\star}$ durch $\alpha_{i_2} f_{i_2}^{max}$ nach unten beschränkt. Dieses Verfahren wird bis zur Maximierung von f_{i_m} fortgeführt.

Im Folgenden soll die Optimierung der Mutationswahrscheinlichkeit beim einfachen genetischen Algorithmus durchgeführt werden. Dabei wird (3.2) mit $f_1 = E_H(H_{opt}, t)$ optimiert und eine weitere Optimierungsaufgabe konstruiert, indem f_1 in (3.2) durch $f_1' = E_S(H_{opt}, t)$ ersetzt wird:

$$\begin{bmatrix} f_1'(p_m) \\ -f_2(p_m) \\ f_3(p_m) \end{bmatrix} \longrightarrow \max \qquad \text{für} \quad p_m \in [0,1]. \tag{3.3}$$

Anschließend wird mit einigen Simulationen überprüft, ob $f_1' = E_S(H,t)$ tatsächlich eine qualitativ bessere Lösung als $f_1 = E_H(H,t)$ ermöglicht.

Bei der Optimierung von (3.2) und (3.3) wird nur das optimale Schema H_{opt} betrachtet, also f_1 auf $f_1 = E_H(H_{opt},t)$ und f_1' auf $f_1' = E_S(H_{opt},t)$ beschränkt. Diese Beschränkung ist für die Aufwandsbegrenzung notwendig, da die Maximierung von f_1 und f_1' für alle Schemen zu umfangreich ist. Das Schema H_{opt} wird als Repräsentant für Schemen mit hohen Fitnesswerten gewählt. Da H_{opt} nur einen einzigen Vertreter besitzt, ist die Länge und Ordnung von H_{opt} bestimmt:

$$\begin{aligned} o(H_{opt}) &= l \\ \delta(H_{opt}) &= l-1. \end{aligned}$$

Besitzt das optimale Schema erstmalig Vertreter in der Population P_t, so handelt es sich meist nur um einen Vertreter, also $m(H_{opt},t) = 1$. Da außerdem $F_t(H_{opt}) \geq \bar{F}_t$ gilt, erhält man eine untere Schranke für die relative Fitness von H_{opt}:

$$\begin{aligned} 2n f_t(H_{opt}) &= m(H_{opt},t)\frac{F_t(H_{opt})}{\bar{F}_t} \geq 1 \qquad \text{mit (2.4)} \\ \Rightarrow f_t(H_{opt}) &\geq \frac{1}{2n} = 0.1 \end{aligned}$$

Für die Untersuchungen wird $f_t(H_{opt}) = 0.1$ gesetzt. Zu der vollständigen Berechnung von E_S für ein konkretes p_m werden weiterhin Werte für

$$f_{sum} := \sum_{k=1}^{\delta(H)} f_t(H_{opt_k}) f_t(\hat{H}_{opt_k})$$

und

$$f_{mut}^i := P\big(H' \cap O_K(P_t) \neq \emptyset\big) \quad \forall H' \in H_{opt_i} \quad \forall i = 1,\ldots,o(H_{opt})$$

benötigt. Diese müssen aus Messungen gewonnen werden. Messwerte für f_{sum} und f_{mut}^i $(i = 1,\ldots,5)$ sind in Tabelle 3.7 zu finden. In diesen Untersuchungen werden, wie bereits erwähnt, die Werte für $m(H_{opt},t) = 1$ verwendet.

Die Funktionswerte von $f_1(p_m)$ und $f_1'(p_m)$ können nun berechnet werden. Abbildung 3.3 zeigt die Graphen der Funktionen $f_1(p_m) = E_H(p_m)$ und $f_1'(p_m) = E_S(p_m)$ mit

$$\begin{aligned} E_H(p_m) &= 0.19 - 0.945p_m + 1.9p_m^2 - 1.9p_m^3 + 0.95p_m^4 - 0.19p_m^5 \\ E_S(p_m) &= 4.671 - 20.030p_m + 34.661p_m^2 - 30.197p_m^3 + 13.179p_m^4 - 2.284p_m^5. \end{aligned}$$

$m(H_{opt}, t)$	f_{sum}	f^1_{mut}	f^2_{mut}	f^3_{mut}	f^4_{mut}	f^5_{mut}
1	2.032	0.333	0.125	0.032	0.0003	0
2	2.921	0.150	0.103	0.013	0.00005	0
3	3.116	0.172	0.041	0.006	0.00003	0
4	3.458	0.081	0.045	0.007	0.00001	0
5	3.644	0.064	0.02	0.004	0.000006	0
6	3.712	0.056	0.013	0.003	0.0000001	0
7	3.892	0.018	0.008	0.0002	0	0
8	3.936	0.015	0.001	0.00002	0	0
9	3.949	0.008	0.004	0	0	0
10	4	0	0	0	0	0

Tabelle 3.7.: Messwerte für f_{sum} und f^i_{mut} $i = 1, \ldots, 5$ aus jeweils 20 Simulation pro $m(H_{opt}, t)$ mit $N = 1$. $m(H_{opt}, t) = 1, 2, \ldots, 10$

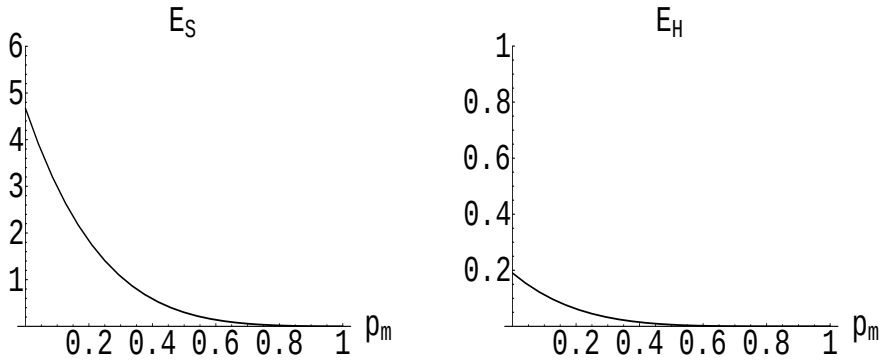

Abbildung 3.3.: Graph der Funktionen $E_H(p_m)$ und $E_S(p_m)$.

Als Approximation für f_2 und f_3 werden die Regressionskurven $r_2(p_m)$ und $r_3(p_m)$ berechnet und verwendet:

$$\begin{aligned} -f_2(p_m) &\approx r_2(p_m) = -16.363\ e^{-13.833 p_m} \\ f_3(p_m) &\approx r_3(p_m) = 0.419 + 0.536 \sin(18.782 p_m + 1.9). \end{aligned}$$

Die Korrelationskoeffizienten ρ_2 für f_2 und ρ_3 für f_3 sind

$$\begin{aligned} \rho_2 &= -0.89133 \\ \rho_3 &= -0.93756. \end{aligned}$$

In Abbildung 3.4 sind die Graphen der Regressionskurven $r_2(p_m)$ und $r_3(p_m)$ dargestellt. Da die Regressionskurven für Werte $p_m \in [0, 0.2]$ berechnet wurden, wird für die Optimierung der zulässige Bereich ebenfalls auf $p_m \in [0, 0.2]$ beschränkt. Dieses Vorgehen ist akzeptabel, da die Graphen in Abbildungen 3.2 und 3.3 darauf hindeuten, dass f_1, f_1' und f_3 mit steigenden p_m gegen Null streben, wogegen f_2 sich nur gering verkleinert.

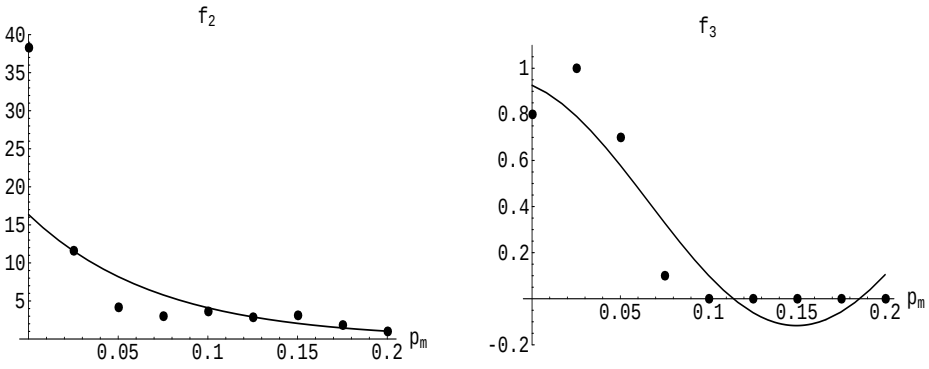

Abbildung 3.4.: Regressionskurven für f_2 und f_3.

Die Rangfolge der zu maximierenden Funktionen für die Methode der schrittweisen Konzessionen wird auf (f_1, f_2, f_3) festgelegt. Die Maximierung der Vertreteranzahl von H_{opt} in der Folgepopulation wird also primär wichtiger als die Minimierung der benötigten Generationen bis zum erstmaligen Auftretens des Optimums und der Maximierung der Wahrscheinlichkeit für den Verbleib des Optimums in der Population erachtet. Das ist dadurch motiviert, dass das Finden des Optimumss bei einer Mutationswahrscheinlichkeit von wenigstens $p_m = 0.025$ schon innerhalb der ersten 10 Generationen erfolgt (Abbildung 3.2). Dies ist für die Praxis ausreichend. Die letzte Funktion der Rangfolge, und damit die als unwichtigste betrachtete, ist f_3. Ein genetischer Algorithmus kann derart modifiziert werden, dass das beste Individuum der Population unverändert in die Folgepopulation übernommen wird. Mit diesem Vorgehen könnte man (3.2) und (3.3) auf Optimierungsprobleme mit zwei Funktionen reduzieren.

Weiterhin werden die Werte für α_1 und α_2 gesetzt. Es wird $\alpha_1 = 0.8$ und $\alpha_2 = 0.9$ gewählt. Der Wert von $f_1(p_m^{opt})$ und $f_1'(p_m^{opt})$ soll also wenigstens 80% des Maximalwertes f_1^{max} und $f_1^{'max}$ betragen, und der Wert von $f_2(p_m^{opt})$ wenigstens 90% von f_2^{max}. Es muss beachtet werden, dass f_2 für die Optimierungsaufgabe negiert wurde. Der Wert von $-f_2(p_m)$ muss daher wenigstens $\big(1 + (1 - \alpha_2)\big)(-f_2^{max}) = (2 - \alpha_2)(-f_2^{max})$ betragen.

Da $f_1(p_m)$, $f_1'(p_m)$, und $-f_2(p_m)$ streng monoton sind, ist die Maximierung dieser Funktionen und die anschließende Beschränkung des zulässigen Bereichs sehr einfach. Das Maximum von $f_1 = E_H(p_m)$ und $f_1' = E_S(p_m)$ über $\mathbb{R}[0, 0.2]$ wird an der Stelle $p_m = 0$ angenommen:

$$\begin{array}{rclrcl} f_1^{max} & = & E_H(0) = 0.19 & f_1^{'max} & = & E_S(0) = 4.67127 \\ \alpha_1 f_1^{max} & = & 0.152 & \alpha_1 f_1^{'max} & = & 3.73702 \\ E_H(p_m^0) & = & \alpha_1 f_1^{max} \text{ für} & E_S(p_m^1) & = & \alpha_1 f_1^{'max} \text{ für} \\ p_m^0 & = & 0.0436475 & p_m^1 & = & 0.0509372 \end{array}$$

Der zulässige Bereich für (3.2) wird nun auf $\mathbb{R}[0, p_m^0]$ und der von (3.3)

	$p_m = 0.0368$	$p_m = 0.0436$
Anzahl der Simulationen, in denen das Optimum gefunden wurde	20	20
Anzahl der Simulationen, in denen das Optimum in der Population blieb, wenn es gefunden wurde	18	16
Durchschnittliche Anzahl der Generationen, die benötigt wurden, das Optimum zu finden, wenn es nicht in der Anfangspopulation enthalten war	3.429	3.357
Durchschnittliche Anzahl der Generationen, die benötigt wurden, bis das Optimum die Population dominierte, d.h. wenigstens von 70% der Individuen repräsentiert wurde	14.1	13.95

Tabelle 3.8.: Ergebnisse aus jeweils 20 Simulationen für $p_m^{opt_1} = 0.0368$ und $p_m^{opt_2} = 0.0436$ mit $N = 250$. Als Zielfunktion wurde $z(x) = x_3$ verwendet.

auf $\mathbb{R}[0, p_m^1]$ beschränkt. Das Maximum von $-f_2$ wird am rechten Intervallendpunkt p_m^0 bzw. p_m^1 angenommen, also:

$$\begin{array}{rclcrcl} f_2^{max} & = & E_H(p_m^0) = -8.94654 & & f_2^{max} & = & E_S(p_m^1) = -8.08841 \\ (2-\alpha_2)f_2^{max} & = & -9.84119 & & (2-\alpha_2)f_2^{max} & = & -8.89725 \\ E_H(p_m^2) & = & (2-\alpha_2)f_2^{max} \text{ für} & & E_S(p_m^3) & = & (2-\alpha_2)f_2^{max} \text{ für} \\ p_m^2 & = & 0.0367572 & & p_m^3 & = & 0.0436 \end{array}$$

Der zulässige Bereich wird für (3.2) auf $\mathbb{R}[p_m^2, p_m^0]$ und der von (3.3) auf $\mathbb{R}[p_m^3, p_m^1]$ beschränkt. Das Maximum von f_3 wird am linken Intervallendpunkt p_m^2, bzw. an p_m^3, angenommen, also

$$p_m^{opt_1} = p_m^2 \qquad p_m^{opt_2} = p_m^3.$$

Nun werden die beiden Optimallösungen $p_m^{opt_1}$ und $p_m^{opt_2}$ miteinander verglichen, um zu überprüfen, ob bei der Optimierung mit $f_1'(p_m) = E_S(p_m)$ tatsächlich eine qualitativ bessere Optimallösung gefunden werden konnte, als mit $f_1 = E_H$. Es werden jeweils 20 Simulationen für $p_m^{opt_1}$ und $p_m^{opt_2}$ durchgeführt. Die Ergebnisse sind in Tabelle 3.8 zu sehen.

Wird $p_m^{opt_2} = 0.0436$ als Mutationswahrscheinlichkeit verwendet, wird das Optimum tatsächlich schneller gefunden und der Algorithmus benötigt weniger Generationen bis wenigstens 70% der Individuen das Optimum repräsentieren. Dafür beträgt die Wahrscheinlichkeit, dass das Optimum in der Population bleibt bei $p_m^{opt_2}$ nur 80% im Vergleich zu 90% bei der Verwendung von $p_m^{opt_1} = 0.0368$. Das Problem des Verbleibs des Optimums

in der Population kann dadurch umgangen werden, dass das jeweils beste Individuum in die Folgepopulation übernommen wird. Somit erweist sich $p_m^{opt_2}$, welches für der Optimierung der Mutationswahrscheinlichkeit mit Hilfe von E_S berechnet wurde, unter den gegebenen Bedingungen als bessere Wahl der Mutationswahrscheinlichkeit.

4. Anwendungsbeispiel für einen evolutionären Algorithmus

4.1. Vorstellung des Projektes

Unter der Leitung von Prof. Dr. Jost wird am Max-Planck-Institut für Mathematik in den Naturwissenschaften ein Projekt verfolgt, in welchem u. a. versucht wird, Kommunikation zwischen zwei oder mehreren Objekten während eines Computerprogrammes entstehen zu lassen.

Da Bienen die einfachste bekannte direkte Kommunikationsform verwenden, soll ein Programm entwickelt werden, in welchem ein Modell von Bienenschwärmen implementiert ist. Einige Komponenten dieses Programmes wurden während des Zeitraumes September 2000 bis August 2001 von der Diplomandin erstellt. Das Programm ist in Java mit Hilfe der Programmbibliothek SWARM [19] programmiert. Zu dem jetzigen Zeitpunkt ist die Umgebung, in der die Bienenschwärme leben, sich ernähren und fortpflanzen, implementiert. Die Kommunikation zwischen den Bienen wird in dem Programm jedoch noch nicht angeboten.

4.2. Das Modell

In diesem Abschnitt soll das Modell, welches dem Programm zu Grunde liegt, soweit es implementiert ist, näher beschrieben werden.

Die Komponenten des Modells sind die Bienenstöcke mit Bienen, die Nahrungsquellen und die Umgebung, in der sich Stöcke, Bienen und Nahrungsquellen befinden. Die Umgebung ist für die Bienen abgeschlossen und torusförmig, also kann keine Biene aus der Umgebung verschwinden, plötzlich auftauchen oder herausfliegen.

In der Umgebung befinden sich Nahrungsquellen, die räumlich gleichmäßig verteilt sind. Sie besitzen unterschiedliche Mengen an Futtereinheiten. Erreicht eine Biene eine Futterquelle, so nimmt sie genau eine Futtereinheit auf und verringert somit die Futtermenge der Nahrungquelle. Besitzt eine Nahrungsquelle keine Futtereinheiten mehr, so verschwindet sie aus der Umgebung. Die Anzahl der Nahrungsquellen in der Umgebung ist normalverteilt.

Es befindet sich eine feste Anzahl von Bienenstöcken in der Umgebung. Jeder der Stöcke hat eine feste Anzahl an Bienen und einen imaginären Futterhaufen, auf dem die Futtereinheiten, welche die Bienen in den Stock tragen, gesammelt werden.

Jede Biene ist mit einer Menge von reellwertigen Koeffizienten ausgestattet, die derzeit 1200 beträgt. Diese Koeffizienten steuern das Verhalten einer Biene. Sie hat die Optionen zu fliegen, im Stock zu bleiben, einer anderen Biene zu folgen, auf einer Futterquelle zu landen und eine Futtereinheit zu ihrem Stock zu tragen. Die Bienen sind in Arbeitsbienen und Königinnen unterteilt. Eine Königin unterscheidet sich von den Arbeitsbienen dadurch, dass sie ihr Verhalten, also ihre Koeffizienten, vererben kann.

In dem Programm ist ein evolutionärer Algorithmus implementiert. Dieser arbeitet mit einer Population, die aus den einzelnen Bienenschwärmen mit den dazugehörigen Stöcken besteht. Die Zielfunktion ist auf der Menge der Bienenschwärme definiert und der Zielfunktionswert eines Schwarmes ist die gesammelte, in dem Stock befindliche, Futtermenge. Die Zielfunktion soll maximiert werden. Es wird die proportionale Fitnesszuweisung und die fitnessproportionale Selektion benutzt. Je höher also die Futtermenge eines Schwarmes, desto höher sein Fitnesswert und somit die Wahrscheinlichkeit, dass er zur Rekombination ausgewählt wird. Bei der Rekombination erhalten alle durch die Selektion ausgewählten Schwärme einen Klon und anschließend werden die Klone und die ausgewählten Schwärme zufällig zu Paaren geordnet. Jedes Paar erzeugt einen neuen Schwarm, indem für jede Biene jeder Koeffizient mit einer Wahrscheinlichkeit von 50% von der Königin des einen und zu 50% von der Königin des anderen Elternschwarmes stammt. Diese Rekombinationart wird „gleichmäßiges Kreuzen“[1] genannt.

Bei der Mutation wird jeder Koeffizient einer Biene mit der Mutationswahrscheinlichkeit p_m mutiert. Findet eine Mutation an dem Koeffizienten k_0 statt, so wird der mutierte Koeffizient $\tilde{k}_0$ in folgender Weise ermittelt:

$$\tilde{k}_0 = k_0 * 2^{\pm\sigma},$$

wobei das Vorzeichen von σ mit einer Wahrscheinlichkeit von 50% positiv oder negativ ist. Die Parameter p_m und σ werden zu Beginn des Programmes festgelegt.

Die entstandenen Bienenschwärme ersetzen die bisherigen Schwärme der Population. Innerhalb der nächsten festgelegten Anzahl an Zeitschritten können die Bienen dieser Schwärme herumfliegen und Futter sammeln. Nach dem Ablauf dieser Zeitschritte, findet wieder die Selektion, die Rekombination und die Mutation statt. Das Abbruchskriterium des Programms ist eine maximale Generationenanzahl.

Die Menge an Futtereinheiten, die ein Schwarm sammeln kann, ist nicht nur von dem Verhalten seiner Bienen, sondern auch von der Lage des Stockes in der Umgebung abhängig. So kann ein Schwarm, dessen Bienen ein sehr gutes Verhalten besitzen, nur wenig Futtereinheiten sammeln, wenn er direkt neben einen anderen Schwarm gelegen ist. Ebenso kann ein Schwarm mit schlechten Bienen überdurchschnittlich viele Futtereinheiten sammeln, wenn sich sein Stock genau neben einer Nahrungsquelle

[1] aus dem Englischen: „uniform crossover“

befindet. Solche zufälligen Einflüsse, wie eine begünstigte Lage oder eine erhöhte Konkurrenz, sollen möglichst ausgeschlossen werden. Daher wird in den Simulationen ein ausgewogener Gitterblockplan, hier speziell ein 2-(49,7,1)-Design, angewendet.

Es befinden sich 49 Schwärme in der Population, die in 56 Blöcken mit jeweils sieben Schwärmen angeordnet werden. Dabei tritt jeder Schwarm genau acht Mal in den Blöcken auf. Jeder Block von Schwärmen hat eine gewisse Anzahl an Zeitschritten zur Verfügung, in denen die sieben Schwärme des Blockes um die Futtereinheiten in der Umgebung konkurrieren können. Die Bienen können also für ihren Schwarm Futter sammeln. Die Schwärme sind derart in die Blöcke eingeteilt, dass jeder Schwarm genau acht Mal mit sechs jeweils anderen Schwärmen um die Futtereinheiten konkurriert. Es tritt also jeder Schwarm mit jedem anderen Schwarm genau einmal in Konkurrenz. Nachdem alle 56 Blöcke abgearbeitet wurden, werden Selektions-, Rekombinations- und Mutationsoperator auf die 49 Schwärme angewendet.

4.3. Bisherige Ergebnisse

Durch die Verwendung eines Designs wird zwar der Einfluss der Zufallskomponenten in den Simulationen verringert, jedoch verlängert sich auch die Laufzeit des evolutionären Algorithmus. So dauert eine Simulation mit sechs Generationen zwischen zwei und drei Tagen. Allerdings hat sich in den meisten Fällen nach sechs Generationen schon ein außerordentlich gutes Sammel- und Flugverhalten der Bienen entwickelt. Zu Beginn werden die Koeffizienten zufällig gesetzt. In Anbetracht der Masse an Koeffizienten, welche aufeinander abgestimmt sein müssen, um ein gutes Verhalten einer Biene zu ermöglichen, ist dies ein beeindruckendes Ergebnis. Abbildung 4.1 zeigt den typischen Verlauf einer Simulation. Der Graph beschreibt die von allen Bienenschwärmen gesammelte Futtermenge pro Generation.

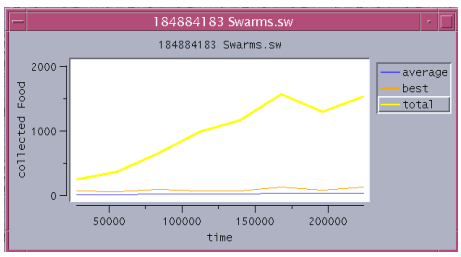

Abbildung 4.1.: Simulationsergebnis nach acht Generationen.

5. Zusammenfassung

Diese Arbeit beschäftigte sich mit der Schematheorie für genetische Algorithmen. Ein Schwerpunkt war die mathematische Formulierung des Ablaufes von genetischen Algorithmen und die Beschreibung der Wirkungsweise genetischer Operatoren. Für die Zwecke dieser Arbeit war eine zufrieden stellende mathematische Formulierung nicht gegeben.

Mit diesem eingeführten Formalismus wurde Hollands Schemensatz 2.3 modifiziert. Mit dem modifizierten Schemensatz 2.7, dessen Herleitung einen weiteren Schwerpunkt darstellt, sind nun genauere Aussagen über die Entwicklung der Population in genetischen Algorithmen möglich. Der modifizierte Schemensatz bietet eine Formel, mit der die erwartete Anzahl von Vertretern eines Schemas in der Folgepopulation exakt angegeben werden kann. Bisher war es nur möglich eine untere Schranke für den Erwartungswert der zukünftigen Vertreteranzahl eines Schemas zu bestimmen.

Mit der exakten Formel können nun zum Beispiel die Kontrollparameter eines einfachen genetischen Algorithmus (Kreuztausch- und Mutationswahrscheinlichkeit) optimiert werden. Die optimale Wahl der Kontrollparameter ist für praktische Zwecke von enormer Bedeutung. In dieser Arbeit wurde eine Optimierung der Mutationswahrscheinlichkeit mit der Hilfe des Schemensatzes und des modifizierten Schemensatzes durchgeführt. Dabei hat sich gezeigt, dass eine qualitativ bessere Lösung des Optimierungsproblems erreicht wurde.

Weiterhin wurden mehrere numerische Untersuchungen an einem einfachen genetischen Algorithmus durchgeführt und interpretiert. Dabei stellte sich heraus, dass die durchschnittliche Approximationsgüte mit der der modifizierte Schemensatz die tatsächlich eingetretene Vertreteranzahl eines Schemas annähert mehr als 2% über der Approximationsgüte des Schemensatzes von Holland liegt.

Abschließend wurde ein evolutionärer Algorithmus, der zu Forschungszwecken implementiert wurde, vorgestellt. Damit wurde die breite Einsetzbarkeit und die Leistungsfähigkeit von evolutionären Algorithmen demonstriert.

A. Übersicht über die Klassen und die Methoden des Programms

Die Klassen

Das Programm wurde in der Programmiersprache Java verfasst und mit der Version 1.3 getestet.

Hauptprogramm: `StartGA`

	Oberklassen	Unterklassen
1	`Coding`	`BinaryCoding`
		`BinaryCodingOfFloatingPointNumbers`
2	`ControlInstances-OfSchemata`	`ControlBinarySchemata`
3	`Crossover`	`OnePointCrossover`
4	`Element`	`BinaryElement`
5	`Evolutionary-Algorithm`	`SimpleGeneticAlgorithm`
6	`FitnessAssignment`	`ProportionalFitnesAssignment`
7	`FitnessAssignment-ForSchemata`	`NumberOfInstancesFitnessForSchemata`
		`ProportionalFitnessForSchemata`
		`RoyalRoadForSchemata`
8	`Mutation`	`BitwiseMutation`
9	`MyMath`	
10	`Scaling`	`LinearScaling`
11	`Schema`	`BinarySchema`
12	`Selection`	`PropotionalSelection`
13	`ValueAssignment`	`ObjectiveScore`
		`RandomGoalFunction`
		`RoyalRoad`

Auflistung der Methoden der einzelnen Klassen

1 Coding

- `code(LinkedList population)`

- `decode(LinkedList population)`

BinaryCoding

- code(LinkedList population)
- code(int dim, double[] point, int strgL, boolean flag)
- code(int point, int strgL)
- decode(LinkedList population)
- decode(int dim, String coding, boolean flag)
- decode(String coding)
- getSign (double x)

BinaryCodingOfFloatingPointNumbers

- code(LinkedList population)
- decode(LinkedList population)
- getSign(double x)
- getExponentN(double x)
- truncate(String oldString)

2 ControlInstancesOfSchemata

- checkNumberOfInstances(LinkedList population)
- checkNumberOfInstancesWithoutBounds(LinkedList population, BinarySchema optSchema)
- assignActualNumberOfInstances(LinkedList population)
- outputNumbers(int popSize)
- outputNumbersByClasses(int popSize)
- getSum_Fhk_times_Fhkroof(BinarySchema schema, LinkedList population)
- getMutArray(BinarySchema schema,LinkedList population)

ControlBinarySchemata

- checkNumberOfInstances(LinkedList population)
- checkNumberOfInstancesWithoutBounds(LinkedList population, BinarySchema optSchema)
- assignActualNumberOfInstances(LinkedList population)
- outputNumbers(int popSize)

- outputNumbersByClasses(int popSize)

- getSum_Fhk_times_Fhkroof(BinarySchema schema, LinkedList population)

- getMutArray(BinarySchema schema,LinkedList population)

- getLowerBoundHolland(BinarySchema schema)

- getLowerBoundHollandOriginal(BinarySchema schema)

- getLowerBoundSchindler(BinarySchema schema, LinkedList population)

- getLowerBoundSchindlerWithSingleTerms(BinarySchema schema, LinkedList population)

3 Crossover

- crossover(LinkedList pop)

- crossoverWithImplicitSelection(LinkedList pop)

OnePointCrossover

- crossover(LinkedList pop)

- crossoverWithImplicitSelection(LinkedList pop)

- doubleElements(LinkedList population)

- getMatingArray(int dim, LinkedList pop2)

- sortList(LinkedList population)

- crossoverWithImplicitSelection(LinkedList pop)

- getMatingArrayWithImplicitSelection(int dim, LinkedList pop2)

- repairNumbers(int diff, LinkedList pop, Element[][] array)

4 Element

- clone()

BinaryElement

- clone()

- reset()

5 EvolutionaryAlgorithm

- setDefaults()
- setProbabilities(double mprob, double coprob)
- setRandomSeeds(long rSeed, long mSeed, long cSeed)
- buildObjects()
- run()
- createPopulation ()
- outputPopulation()

SimpleGeneticAlgorithm

- buildObjects()
- run()
- createPopulation ()
- createStrings()
- createStrings(int noOptElements)
- outputPopulation()
- outputPopulation(LinkedList population)
- getAllBinarySchemata()
- getSpecialSchemata()
- checkIfOptimaIsFound(String optSchema, LinkedList pop)

6 FitnessAssignment

- assignFitness(LinkedList population)
- getSumValues(LinkedList population)
- getSumValues(LinkedList population, boolean flag)
- min(LinkedList population)
- avg(LinkedList population)
- max(LinkedList population)

ProportionalFitness

- assignFitness(LinkedList population)

7 FitnessAssignmentForSchemata

- assignFitness(LinkedList population)
- getFitness(BinarySchema schema, LinkedList population)
- getProbOfSchemaWith$differentPositions(BinarySchema schema, int i, LinkedList population)
- averageValue(LinkedList schemata)

NumberOfInstancesFitnessForSchemata

- assignFitness(LinkedList population)
- getFitness(BinarySchema schema, LinkedList population)
- getProbOfSchemaWith$differentPositions(BinarySchema schema, int i, LinkedList population)

ProportionalFitnessForSchemata

- assignFitness(LinkedList population)
- getFitness(BinarySchema schema, LinkedList population)
- getProbOfSchemaWith$differentPositions(BinarySchema schema, int i, LinkedList population)

RoyalRoadForSchemata

- assignFitness(LinkedList population)
- getFitness(BinarySchema schema, LinkedList population)
- getProbOfSchemaWith$differentPositions(BinarySchema schema, int i, LinkedList population)

8 Mutation

- mutate(LinkedList population)

BitwiseMutation

- mutate(LinkedList population)

9 MyMath

- binomialCoeff(int n, int k)
- faculty(int n)
- getArrayOfPermutations(int n, int k)
- permsCorrect(int dimN, int dimM, int[][] array,int k)

10 Scaling

- `scale(LinkedList population, double fmin, double favg, double fmax)`
- `prescale(double fmin, double favg, double fmax)`

LinearScaling

- `scale(LinkedList population, double fmin, double favg, double fmax)`
- `prescale(double fmin, double favg, double fmax)`
- `scale(double oldFitness)`

11 Schema

- `clone()`
- `getNumberOfInstances(LinkedList population)`
- `isRepresentative(Element elem)`

BinarySchema

- `clone()`
- `setDefiningLength()`
- `getNumberOfInstances(LinkedList population)`
- `getFirstDefinedPosition()`
- `getLastDefinedPosition()`
- `isRepresentative(BinaryElement elem)`
- `getStringForm()`
- `isBinarySchema(int order, int[][] array)`
- `toArray(String strg)`
- `getValue(LinkedList population)`
- `getFitness()`
- `getHk(int k)`
- `getHkRoof(int k)`
- `sortFixedPos()`
- `fixedPosSorted()`
- `findPositiveMin()`
- `isSubSchemaOf(BinarySchema schema)`

12 Selection

- `select(LinkedList population)`
- `sortList(LinkedList population)`

ProportionalSelection

- `select(LinkedList population)`
- `repairNumbersSelected(int diff, LinkedList pop)`

13 ValueAssignment

- `assignValues(LinkedList population)`
- `averageValue(LinkedList population)`

ObjectiveScore

- `assignValues(LinkedList population)`

RandomGoalFunction

- `assignValues(LinkedList population)`
- `createFunction1D()`
- `createFunction2D()`

RoyalRoad

- `assignValues(LinkedList population)`

Literaturverzeichnis

[1] Altenberg Lee, *The Schema Theorem and Price's Theorem*, Foundations of Genetic Algorithms 3, ed. Darrell Whitley and Michael Vose. pp. 23-49. Morgan Kaufmann, San Francisco, 1995

[2] Bagley J.D., *The behavior of adaptive systems which employ genetic and correlation algorithms*, Doctorial dissertation, University of Michigan, Dissertation Abstracts International, 28(12), 5106B, University Microfilms No. 68-7556, 1967

[3] Fogel L.J., Owens A.J., Walsh M.J., *Artificial Intelligence through Simulated Evolution*, New York, John Wiley, 1966

[4] Fogel, Lawrence J., *Intelligence through simulated evolution: forty years of evolutionary programming*, New York, John Wiley, 1999

[5] Forrest Stephanie, Holland John H., Mitchell Melanie, *The Royal Road for Genetic Algorithms: Fitness Landscapes and GA Performance*, Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life, 1991

[6] Goldberg David E., *Genetic Algorithms in Search, Optimization, and Machine learning*, Reading Massachusetts, Addison Wesley, 1989

[7] Goldberg David E., *Real coded Genetic Algorithms, Virtual Alphabets, and Blocking*, Complex Systems 5, 139-167, 1991

[8] Grosche G., Zeidler E., Ziegler D., Ziegler V., *Teubner-Taschenbuch der Mathematik*, Stuttgart, Leipzig, Teubner, 1995

[9] Haefner James W., *Modeling biological systems: principles and applications*, New York, Chapman & Hall, 1996

[10] Hillermeier Claus, *Nonlinear multiobjective optimization: a generalized homotopy approach*, Berlin, Birkhäuser, 2001 (International series of numerical mathematics Vol. 135)

[11] Holland John H., *Adaption in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*, Cambridge Massachusetts, MIT Press, 1975

[12] Kopka Helmut, *LaTeX: Einführung*, Bonn, Addison-Wesley, 1996

[13] Koza John R., *Genetic programming: On the programming of computers by means of natural selection*, Cambridge Massachusetts, MIT Press, 1992

[14] Michalewicz Zbigniew, *Genetic algorithms + data structures = evolution programs*, New York, Springer-Verlag, 1996

[15] Mitchell Melanie, *An introduction to genetic algorithms*, Cambridge Massachusetts, MIT Press, 1996

[16] Nissen Volker, *Einführung in Evolutionäre Algorithmen: Optimierung nach dem Vorbild der Evolution*, Braunschweig, Wiesbaden, Vieweg, 1997

[17] Pohlheim Hartmut, *Evolutionäre Algorithmen: Verfahren, Operatoren und Hinweise für die Praxis*, Berlin, Heidelberg, Springer-Verlag, 2000

[18] Rechenberg I., *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Stuttgart, Frommann-Holzboog, 1973

[19] Swarm Development Group, 624 Agua Fria, Suite 2, Santa Fe, New Mexico, 87501 USA

[20] Reeves Colin R., *Using Genetic Algorithms With Small Populations*, Proceedings of the Fifth International Conference on Genetic Algorithms, ed. S. Forrest, Morgan Kaufmann, San Mateo California, 1993

[21] Vose Michael D., *The simple genetic algorithm: foundations and theory*, Cambridge Massachusetts, MIT Press, 1999

[22] Wegmann Helmut, Lehn Jürgen, *Einführung in die Statistik*, Stuttgart, Teubner, 1992

Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit vollständig von mir und nur mit der Hilfe meiner Betreuer Prof. Jost und Prof. Rudolph und unter Verwendung der angegebenen Literatur angefertigt wurde.

Leipzig, den 13. Juni 2002