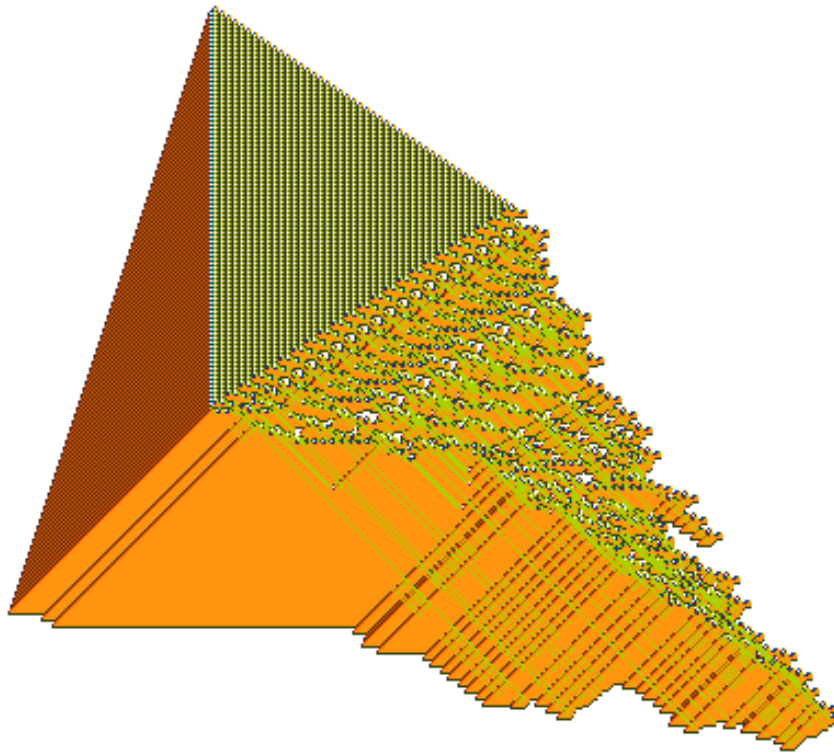


Prof. Dr. Alexander del Pino
Fachbereich Informatik

Genetische Algorithmen



10. Teil
Das
Schematheorem

Das Schematheorem

Worum geht es ?

Bisher haben wir uns hauptsächlich damit beschäftigt, zu verstehen **wie** ein genetischer Algorithmus funktioniert.

Dazu haben wir uns zuerst den grundsätzlichen Aufbau eines typischen genetischen Algorithmus angeschaut und anschließend Details betrachtet.

Das Schematheorem

Worum geht es ?

Bisher haben wir uns hauptsächlich damit beschäftigt, zu verstehen **wie** ein genetischer Algorithmus funktioniert.

Dazu haben wir uns zuerst den grundsätzlichen Aufbau eines typischen genetischen Algorithmus angeschaut und anschließend Details betrachtet.

Jetzt geht es um die Frage:

Warum funktioniert ein genetischer Algorithmus überhaupt ?

Das Schematheorem

Worum geht es ?

Bisher haben wir uns hauptsächlich damit beschäftigt, zu verstehen **wie** ein genetischer Algorithmus funktioniert.

Dazu haben wir uns zuerst den grundsätzlichen Aufbau eines typischen genetischen Algorithmus angeschaut und anschließend Details betrachtet.

Jetzt geht es um die Frage:

Warum funktioniert ein genetischer Algorithmus überhaupt ?

Dabei müssen wir folgendes beachten:

- Genetische Algorithmen sind große, komplexe Systeme und eignen sich zum Finden von Näherungslösungen für gerade solche Probleme, die oftmals selbst auch nur schwer zu beschreiben sind.

Das Schematheorem

Worum geht es ?

Bisher haben wir uns hauptsächlich damit beschäftigt, zu verstehen **wie** ein genetischer Algorithmus funktioniert.

Dazu haben wir uns zuerst den grundsätzlichen Aufbau eines typischen genetischen Algorithmus angeschaut und anschließend Details betrachtet.

Jetzt geht es um die Frage:

Warum funktioniert ein genetischer Algorithmus überhaupt ?

Dabei müssen wir folgendes beachten:

- Genetische Algorithmen sind große, komplexe Systeme und eignen sich zum Finden von Näherungslösungen für gerade solche Probleme, die oftmals selbst auch nur schwer zu beschreiben sind.
- Wegen der Verwendung von Zufallszahlen kann man allenfalls versuchen, mit Hilfe der Statistik das durchschnittliche Verhalten eines genetischen Algorithmus besser zu verstehen.

Das Schematheorem

Warum funktioniert ein genetischer Algorithmus ?

Es ist also eher *unwahrscheinlich*, für diese Frage ein exaktes Modell zu finden, welches das Verhalten eines genetischen Algorithmus für ein beliebiges Optimierungsproblem hinreichend genau beschreibt.

Das Schematheorem

Warum funktioniert ein genetischer Algorithmus ?

Es ist also eher *unwahrscheinlich*, für diese Frage ein exaktes Modell zu finden, welches das Verhalten eines genetischen Algorithmus für ein beliebiges Optimierungsproblem hinreichend genau beschreibt.

A. Eiben zieht hierzu einen interessanten Vergleich:

Moreover, while the field of EAs is fairly young, it is worth noting that the field of population genetics and evolutionary theory has a head start of more than a hundred years, and is still battling against the barrier of complexity.

Quelle: A. E. Eiben, J. E. Smith: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003

Das Schematheorem

Warum funktioniert ein genetischer Algorithmus ?

Es ist also eher *unwahrscheinlich*, für diese Frage ein exaktes Modell zu finden, welches das Verhalten eines genetischen Algorithmus für ein beliebiges Optimierungsproblem hinreichend genau beschreibt.

A. Eiben zieht hierzu einen interessanten Vergleich:

Moreover, while the field of EAs is fairly young, it is worth noting that the field of population genetics and evolutionary theory has a head start of more than a hundred years, and is still battling against the barrier of complexity.

Quelle: A. E. Eiben, J. E. Smith: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003

J. Holland schlug 1975 einen Erklärungsansatz für einen einfachen genetischen Algorithmus vor, der unter dem Begriff *Schematheorem* bekannt wurde.

Das Schematheorem

Ein einfacher genetischer Algorithmus

Betrachten wir einen einfachen genetischen Algorithmus.

- Die Population P besteht aus n Lösungskandidaten.

$$P = \{k_1, k_2, k_3, \dots, k_n\}$$

Das Schematheorem

Ein einfacher genetischer Algorithmus

Betrachten wir einen einfachen genetischen Algorithmus.

- Die Population P besteht aus n Lösungskandidaten.

$$P = \{k_1, k_2, k_3, \dots, k_n\}$$

- Die Lösungskandidaten werden als einfache Bitstrings der Länge l kodiert.

Das Schematheorem

Ein einfacher genetischer Algorithmus

Betrachten wir einen einfachen genetischen Algorithmus.

- Die Population P besteht aus n Lösungskandidaten.

$$P = \{k_1, k_2, k_3, \dots, k_n\}$$

- Die Lösungskandidaten werden als einfache Bitstrings der Länge l kodiert.
- Die Wahrscheinlichkeit, dass ein bestimmter Lösungskandidat k_i selektiert wird, ist proportional zu seiner Fitness.

$$p(k_i) = f(k_i) / \sum_{i=1}^n f(k_i)$$

Das Schematheorem

Ein einfacher genetischer Algorithmus

Betrachten wir einen einfachen genetischen Algorithmus.

- Die Population P besteht aus n Lösungskandidaten.

$$P = \{k_1, k_2, k_3, \dots, k_n\}$$

- Die Lösungskandidaten werden als einfache Bitstrings der Länge l kodiert.
- Die Wahrscheinlichkeit, dass ein bestimmter Lösungskandidat k_i selektiert wird, ist proportional zu seiner Fitness.

$$p(k_i) = f(k_i) / \sum_{i=1}^n f(k_i)$$

- Die Rekombination erfolgt durch 1-Punkt-Crossover mit einer Wahrscheinlichkeit p_c .

Das Schematheorem

Ein einfacher genetischer Algorithmus

Betrachten wir einen einfachen genetischen Algorithmus.

- Die Population P besteht aus n Lösungskandidaten.

$$P = \{k_1, k_2, k_3, \dots, k_n\}$$

- Die Lösungskandidaten werden als einfache Bitstrings der Länge l kodiert.
- Die Wahrscheinlichkeit, dass ein bestimmter Lösungskandidat k_i selektiert wird, ist proportional zu seiner Fitness.

$$p(k_i) = f(k_i) / \sum_{i=1}^n f(k_i)$$

- Die Rekombination erfolgt durch 1-Punkt-Crossover mit einer Wahrscheinlichkeit p_c .
- Die Mutation erfolgt bitweise unabhängig mit einer Wahrscheinlichkeit p_m .

Das Schematheorem

Was ist ein Schema ?

Ein *Schema* ist die Beschreibung eines Bitmusters, wobei neben den Symbolen 0 und 1 auch noch das **-Symbol* für solche Bitstellen verwendet wird, die einen beliebigen Wert annehmen dürfen (*don't care*).

Das Schematheorem

Was ist ein Schema ?

Ein *Schema* ist die Beschreibung eines Bitmusters, wobei neben den Symbolen 0 und 1 auch noch das **-Symbol* für solche Bitstellen verwendet wird, die einen beliebigen Wert annehmen dürfen (*don't care*).

Eine *Instanz eines Schemas* ist ein Bitstring, der auf dieses Schema passt, wobei anstelle der *-Symbole beliebig 0 und 1 verwendet werden darf.

Das Schematheorem

Was ist ein Schema ?

Ein *Schema* ist die Beschreibung eines Bitmusters, wobei neben den Symbolen 0 und 1 auch noch das **-Symbol* für solche Bitstellen verwendet wird, die einen beliebigen Wert annehmen dürfen (*don't care*).

Eine *Instanz eines Schemas* ist ein Bitstring, der auf dieses Schema passt, wobei anstelle der *-Symbole beliebig 0 und 1 verwendet werden darf.

Beispiel

Zu dem Schema ****10*** gibt es genau acht verschiedene Instanzen.

Die rot markierten Bits sind hier durch das Schema vorgegeben, die grün markierten Bits ersetzen die *-Symbole:

00**1**00, 00**1**01, 01**1**00, 01**1**01, 10**1**00, 10**1**01, 11**1**00 und 11**1**01.

Das Schematheorem

Was ist ein Schema ?

Ein *Schema* ist die Beschreibung eines Bitmusters, wobei neben den Symbolen 0 und 1 auch noch das **-Symbol* für solche Bitstellen verwendet wird, die einen beliebigen Wert annehmen dürfen (*don't care*).

Eine *Instanz eines Schemas* ist ein Bitstring, der auf dieses Schema passt, wobei anstelle der *-Symbole beliebig 0 und 1 verwendet werden darf.

Beispiel

Zu dem Schema ***10** gibt es genau acht verschiedene Instanzen.

Die rot markierten Bits sind hier durch das Schema vorgegeben, die grün markierten Bits ersetzen die *-Symbole:

00100, 00101, 01100, 01101, 10100, 10101, 11100 und 11101.



Der Bitstring 01010 passt nicht zu diesem Schema. Warum ?

Das Schematheorem

Beschreibung von Schemen

Es gibt zwei Merkmale, die man bei einem Schema S betrachten kann.

- *Die Ordnung $o(S)$* des Schemas S ist die Anzahl der 0 und 1 Symbole im Schema (*order of the schema*).

Das Schematheorem

Beschreibung von Schemen

Es gibt zwei Merkmale, die man bei einem Schema S betrachten kann.

- *Die Ordnung $o(S)$* des Schemas S ist die Anzahl der 0 und 1 Symbole im Schema (*order of the schema*).
- *Die Distanz $d(S)$* des Schemas S ist der Index des am weitesten rechts stehenden 0 oder 1 Symbols minus dem Index des am weitesten links stehenden 0 oder 1 Symbols (*defining length of the schema*).

Das Schematheorem

Beschreibung von Schemen

Es gibt zwei Merkmale, die man bei einem Schema S betrachten kann.

- *Die Ordnung $o(S)$* des Schemas S ist die Anzahl der 0 und 1 Symbole im Schema (*order of the schema*).
- *Die Distanz $d(S)$* des Schemas S ist der Index des am weitesten rechts stehenden 0 oder 1 Symbols minus dem Index des am weitesten links stehenden 0 oder 1 Symbols (*defining length of the schema*).

Beispiel

$S = *1**110*0**$

$o(S) = 5$

$d(S) = 9 - 2 = 7$

Das Schematheorem

Aussagen und Beobachtungen zu Schemen

 Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

Das Schematheorem

Aussagen und Beobachtungen zu Schemen

 Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.

Das Schematheorem

Aussagen und Beobachtungen zu Schemen

 Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.
- Mit einem Bitstring der Länge k können 3^k Schemen definiert werden.

Das Schematheorem

Aussagen und Beobachtungen zu Schemen

 Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.
- Mit einem Bitstring der Länge k können 3^k Schemen definiert werden.
- Eine Population von n Bitstrings der Länge k enthält mindestens 2^k Instanzen von Schemen.

Das Schematheorem

Aussagen und Beobachtungen zu Schemen

 Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.
- Mit einem Bitstring der Länge k können 3^k Schemen definiert werden.
- Eine Population von n Bitstrings der Länge k enthält mindestens 2^k Instanzen von Schemen.
- Eine Population von n Bitstrings der Länge k enthält höchstens $n * 2^k$ Instanzen von Schemen.

Das Schematheorem

Aussagen und Beobachtungen zu Schemen



Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.
- Mit einem Bitstring der Länge k können 3^k Schemen definiert werden.
- Eine Population von n Bitstrings der Länge k enthält mindestens 2^k Instanzen von Schemen.
- Eine Population von n Bitstrings der Länge k enthält höchstens $n * 2^k$ Instanzen von Schemen.
- Zu einem Schema S sind bei einem Bitstring der Länge k genau $2^{k-o(S)}$ verschiedene Instanzen denkbar.

Das Schematheorem

Aussagen und Beobachtungen zu Schemen



Wie würden Sie folgende Aussagen zu Schemen plausibel begründen ?

- Ein Bitstring der Länge k ist immer eine Instanz von 2^k Schemen.
- Mit einem Bitstring der Länge k können 3^k Schemen definiert werden.
- Eine Population von n Bitstrings der Länge k enthält mindestens 2^k Instanzen von Schemen.
- Eine Population von n Bitstrings der Länge k enthält höchstens $n * 2^k$ Instanzen von Schemen.
- Zu einem Schema S sind bei einem Bitstring der Länge k genau $2^{k-o(S)}$ verschiedene Instanzen denkbar.



Der einfache genetische Algorithmus verarbeitet *explizit* zwar nur die n Bitstrings der aktuellen Population, aber *implizit* eine weitaus größere Anzahl an Schemen (*implicit parallelism*).

Das Schematheorem

Die echte durchschnittliche Fitness eines Schemas

Die echte durchschnittliche Fitness eines Schemas ist die durchschnittliche Fitness aller möglichen Instanzen des Schemas.

Das Schematheorem

Die echte durchschnittliche Fitness eines Schemas

Die echte durchschnittliche Fitness eines Schemas ist die durchschnittliche Fitness aller möglichen Instanzen des Schemas.

Beispiel

Bei Bitstrings der Länge $k=40$ gibt es 2^{32} verschiedene Instanzen eines Schemas S mit $o(S)=8$.

Das Schematheorem

Die echte durchschnittliche Fitness eines Schemas

Die echte durchschnittliche Fitness eines Schemas ist die durchschnittliche Fitness aller möglichen Instanzen des Schemas.

Beispiel

Bei Bitstrings der Länge $k=40$ gibt es 2^{32} verschiedene Instanzen eines Schemas S mit $o(S)=8$.

Um die echte durchschnittliche Fitness eines solchen Schemas zu berechnen müsste man also 2^{32} mal die Fitnessfunktion aufrufen.

Das Schematheorem

Die echte durchschnittliche Fitness eines Schemas

Die echte durchschnittliche Fitness eines Schemas ist die durchschnittliche Fitness aller möglichen Instanzen des Schemas.

Beispiel

Bei Bitstrings der Länge $k=40$ gibt es 2^{32} verschiedene Instanzen eines Schemas S mit $o(S)=8$.

Um die echte durchschnittliche Fitness eines solchen Schemas zu berechnen müsste man also 2^{32} mal die Fitnessfunktion aufrufen.

Wenn man berücksichtigt, dass es in diesem Beispiel $3^k \approx 1.2 * 10^{19}$ Schemen gibt, wird klar, dass man die echte durchschnittliche Fitness der Schemen aus praktischen Gründen nicht explizit berechnen kann.

Das Schematheorem

Die durchschnittliche Fitness eines Schemas

Stattdessen betrachtet man als Näherungswert die durchschnittliche Fitness derjenigen Lösungskandidaten, die sich zu dem Zeitpunkt t tatsächlich in der Population befinden und Instanzen des betrachteten Schemas sind.

Das Schematheorem

Die durchschnittliche Fitness eines Schemas

Stattdessen betrachtet man als Näherungswert die durchschnittliche Fitness derjenigen Lösungskandidaten, die sich zu dem Zeitpunkt t tatsächlich in der Population befinden und Instanzen des betrachteten Schemas sind.

Beispiel

Kandidat	Fitness
----------	---------


011001	45
--------	----

100101	50
--------	----

010100	15
--------	----

101100	25
--------	----

001110	60
--------	----

 Welche durchschnittliche Fitness besitzt das Schema *****10*** in dieser Population ?

Das Schematheorem


Die durchschnittliche Fitness eines Schemas

Stattdessen betrachtet man als Näherungswert die durchschnittliche Fitness derjenigen Lösungskandidaten, die sich zu dem Zeitpunkt t tatsächlich in der Population befinden und Instanzen des betrachteten Schemas sind.

Beispiel

Kandidat Fitness

011001	45
100101	50
010100	15
101100	25
001110	60

 Welche durchschnittliche Fitness besitzt das Schema *****10*** in dieser Population ?

Das Schema *****10*** besitzt drei Instanzen in dieser Population, nämlich **100101**, **010100** und **101100**. Die durchschnittliche Fitness dieses Schemas beträgt also $(50 + 15 + 25) / 3 = 90 / 3 = 30$.

Das Schematheorem

Wirkung der Selektion

Sei $n(S,t)$ die Anzahl der vorhandenen Instanzen des Schemas S zum Zeitpunkt t , und $x \in S$ eine Instanz von S . Die durchschnittliche Fitness von S beträgt dann wie in dem vorigen Beispiel gezeigt:

$$f(S, t) = \frac{\sum_{x \in S} f(x)}{n(S, t)} \quad (1)$$

Das Schematheorem

Wirkung der Selektion

Sei $n(S,t)$ die Anzahl der vorhandenen Instanzen des Schemas S zum Zeitpunkt t , und $x \in S$ eine Instanz von S . Die durchschnittliche Fitness von S beträgt dann wie in dem vorigen Beispiel gezeigt:

$$f(S, t) = \frac{\sum_{x \in S} f(x)}{n(S, t)} \quad (1)$$

Die zu erwartende Anzahl der Instanzen von S in der nächsten Generation $t+1$ ist abhängig von der Fitnessproportionalität der aktuellen Instanzen von S :

$$E(n(S, t+1)) = \sum_{x \in S} \frac{f(x)}{F(t)} \quad (2)$$

Das Schematheorem

Wirkung der Selektion

Sei $n(S,t)$ die Anzahl der vorhandenen Instanzen des Schemas S zum Zeitpunkt t , und $x \in S$ eine Instanz von S . Die durchschnittliche Fitness von S beträgt dann wie in dem vorigen Beispiel gezeigt:

$$f(S, t) = \frac{\sum_{x \in S} f(x)}{n(S, t)} \quad (1)$$

Die zu erwartende Anzahl der Instanzen von S in der nächsten Generation $t+1$ ist abhängig von der Fitnessproportionalität der aktuellen Instanzen von S :

$$E(n(S, t+1)) = \sum_{x \in S} \frac{f(x)}{F(t)} \quad (2)$$

wobei $F(t)$ die durchschnittliche Fitness in der aktuellen Population ist:

$$F(t) = \frac{1}{n} * \sum_{x=1}^n f(x) \quad (3)$$

Das Schematheorem

Wirkung der Selektion

(2) lässt sich wie folgt umschreiben:

$$E(n(S, t+1)) = \sum_{x \in S} \frac{f(x)}{F(t)} = \frac{1}{F(t)} * \sum_{x \in S} f(x) \quad (2')$$

Das Schematheorem

Wirkung der Selektion

(2) lässt sich wie folgt umschreiben:

$$E(n(S, t+1)) = \sum_{x \in S} \frac{f(x)}{F(t)} = \frac{1}{F(t)} * \sum_{x \in S} f(x) \quad (2')$$

(1) kann durch Multiplikation von $n(S, t)$ auf beide Seiten wie folgt umgestellt werden:

$$f(S, t) = \frac{\sum_{x \in S} f(x)}{n(S, t)} \Leftrightarrow \sum_{x \in S} f(x) = n(S, t) * f(S, t) \quad (4)$$

Das Schematheorem

Wirkung der Selektion

(2) lässt sich wie folgt umschreiben:

$$E(n(S, t+1)) = \sum_{x \in S} \frac{f(x)}{F(t)} = \frac{1}{F(t)} * \sum_{x \in S} f(x) \quad (2')$$

(1) kann durch Multiplikation von $n(S, t)$ auf beide Seiten wie folgt umgestellt werden:

$$f(S, t) = \frac{\sum_{x \in S} f(x)}{n(S, t)} \Leftrightarrow \sum_{x \in S} f(x) = n(S, t) * f(S, t) \quad (4)$$

Einsetzen von (4) in (2') liefert:

$$E(n(S, t+1)) = n(S, t) * \frac{f(S, t)}{F(t)} \quad (5)$$



Die erwartete Anzahl der Instanzen eines Schemas wächst also proportional zur durchschnittlichen Fitness des Schemas.

Das Schematheorem

Wirkung der Rekombination

Bei Bitstrings der Länge l kann der Crossover-Operator an $l-1$ Stellen wirken.

Die Wahrscheinlichkeit $p_z(S)$ dass eine Instanz des Schemas S durch Crossover zerstört wird, hängt dabei maßgeblich von der Distanz $d(S)$ des Schemas ab:

$$p_z(S) = \frac{d(S)}{l-1} \quad (6)$$

Das Schematheorem

Wirkung der Rekombination

Bei Bitstrings der Länge l kann der Crossover-Operator an $l-1$ Stellen wirken.

Die Wahrscheinlichkeit $p_z(S)$ dass eine Instanz des Schemas S durch Crossover zerstört wird, hängt dabei maßgeblich von der Distanz $d(S)$ des Schemas ab:

$$p_z(S) = \frac{d(S)}{l-1} \quad (6)$$

Allerdings wird der Crossover-Operator ja nicht auf jeden Bitstring angewendet, sondern der Bitstring wird mit einer Wahrscheinlichkeit p_c dafür ausgewählt:

$$p_z(S) = p_c * \frac{d(S)}{l-1} \quad (7)$$

Das Schematheorem

Wirkung der Rekombination

Bei Bitstrings der Länge l kann der Crossover-Operator an $l-1$ Stellen wirken.

Die Wahrscheinlichkeit $p_z(S)$ dass eine Instanz des Schemas S durch Crossover zerstört wird, hängt dabei maßgeblich von der Distanz $d(S)$ des Schemas ab:

$$p_z(S) = \frac{d(S)}{l-1} \quad (6)$$

Allerdings wird der Crossover-Operator ja nicht auf jeden Bitstring angewendet, sondern der Bitstring wird mit einer Wahrscheinlichkeit p_c dafür ausgewählt:

$$p_z(S) = p_c * \frac{d(S)}{l-1} \quad (7)$$

Die Wahrscheinlichkeit $p_{\bar{u}c}(S)$ dass eine Instanz des Schemas S den Crossover-Operator überlebt, also nicht auseinander geschnitten wird, beträgt somit:

$$p_{\bar{u}c}(S) = 1 - p_c * \frac{d(S)}{l-1} \quad (8)$$

Das Schematheorem

Wirkung der Rekombination

Weiterhin besteht eine - wenn auch kleine - Chance, dass die Schemainstanz nicht durch Crossover zerstört wird, weil beide Elternteile Instanzen dieses Schemas sind. Somit gilt also:

$$p_{\ddot{u}c}(S) \geq 1 - p_c * \frac{d(S)}{l-1} \quad (9)$$

Das Schematheorem

Wirkung der Rekombination

Weiterhin besteht eine - wenn auch kleine - Chance, dass die Schemainstanz nicht durch Crossover zerstört wird, weil beide Elternteile Instanzen dieses Schemas sind. Somit gilt also:

$$p_{\text{üc}}(S) \geq 1 - p_c * \frac{d(S)}{l-1} \quad (9)$$

Aus (5) und (9) können wir die erwartete Anzahl der Instanzen eines Schemas in Abhängigkeit von dessen aktueller Anzahl, dessen Fitness und dessen Distanz wie folgt ausdrücken:

$$E(n(S, t+1)) \geq n(S, t) * \frac{f(S, t)}{F(t)} * \left(1 - p_c * \frac{d(S)}{l-1}\right) \quad (10)$$



(10) zeigt, dass Instanzen von Schemen mit überdurchschnittlicher Fitness und mit kurzer Distanz bevorzugt weiterkommen. Umgekehrt tendieren Schemen mit unterdurchschnittlicher Fitness und/oder einer hohen Distanz dazu, auszusterben.

Das Schematheorem

Wirkung der Mutation

Die Wahrscheinlichkeit, dass ein Bit durch Mutation verändert wird, beträgt p_m .

Somit beträgt die Wahrscheinlichkeit $p_{\text{üm}}(S)$ dass keines der $o(S)$ Bits der Instanz des Schemas S durch die Mutation verändert wird:

$$p_{\text{üm}}(S) = (1 - p_m)^{o(S)} \quad (11)$$

Das Schematheorem

Wirkung der Mutation

Die Wahrscheinlichkeit, dass ein Bit durch Mutation verändert wird, beträgt p_m .

Somit beträgt die Wahrscheinlichkeit $p_{\text{üm}}(S)$ dass keines der $o(S)$ Bits der Instanz des Schemas S durch die Mutation verändert wird:

$$p_{\text{üm}}(S) = (1 - p_m)^{o(S)} \quad (11)$$

Wenn wir also den Einfluß der Selektion, Rekombination und Mutation zusammen betrachten, erhalten wir:

$$E(n(S, t+1)) \geq n(S, t) * \frac{f(S, t)}{F(t)} * (1 - p_c * \frac{d(S)}{l-1}) * (1 - p_m)^{o(S)} \quad (12)$$



Dieser Ausdruck ist nun das **Schematheorem** von *John Holland*. Es besagt, dass sich Schemen mit überdurchschnittlicher Fitness, kleiner Ordnung und kurzer Distanz besonders gut weiterentwickeln.

Das Schematheorem

Die Baustein-Hypothese

Aus dem Schematheorem lässt sich die Baustein-Hypothese (building block hypothesis) ableiten.

Das Schematheorem

Die Baustein-Hypothese

Aus dem Schematheorem lässt sich die **Baustein-Hypothese** (**building block hypothesis**) ableiten.

Die Baustein-Hypothese besagt, dass ein genetischer Algorithmus versucht, Schemen mit überdurchschnittlicher Fitness, kleiner Ordnung und kurzer Distanz - sogenannte **Bausteine** (**building blocks**) - so zusammenzusetzen, dass damit eine gute Lösung gefunden wird.

Das Schematheorem

Die Baustein-Hypothese

Aus dem Schematheorem lässt sich die **Baustein-Hypothese** (**building block hypothesis**) ableiten.

Die Baustein-Hypothese besagt, dass ein genetischer Algorithmus versucht, Schemen mit überdurchschnittlicher Fitness, kleiner Ordnung und kurzer Distanz - sogenannte **Bausteine** (**building blocks**) - so zusammenzusetzen, dass damit eine gute Lösung gefunden wird.

David Goldberg beschreibt dies sehr treffend:

Because highly fit schemata of low defining length and low order play such an important role in the action of genetic algorithms, we have already given them a special name: building blocks.

Quelle: D. Goldberg: *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley, 1989, S. 41

Das Schematheorem

Die Baustein-Hypothese

Aus dem Schematheorem lässt sich die **Baustein-Hypothese** (**building block hypothesis**) ableiten.

Die Baustein-Hypothese besagt, dass ein genetischer Algorithmus versucht, Schemen mit überdurchschnittlicher Fitness, kleiner Ordnung und kurzer Distanz - sogenannte **Bausteine** (**building blocks**) - so zusammenzusetzen, dass damit eine gute Lösung gefunden wird.

David Goldberg beschreibt dies sehr treffend:

Because highly fit schemata of low defining length and low order play such an important role in the action of genetic algorithms, we have already given them a special name: building blocks.

Just as a child creates magnificent fortresses through the arrangement of simple blocks of wood, so does a genetic algorithm seek near optimal performance through the juxtaposition of short, low-order, high-performance schemata, or building blocks.

Quelle: D. Goldberg: *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley, 1989, S. 41

Das Schematheorem

3-SAT

Beispiel: 256 Variablen in 1024 Klauseln. Zustandsraum 2^{256} . Wann ist diese Formel erfüllbar (alle Klauseln sind wahr) ?

(1 17 2) (1 -17 -2) (-1 17 -2) (-1 -17 2) (17 33 18) (17 -33 -18) (-17 33 -18) (33 49 34) (33 -49 -34) (-33 49 -34) (-33 -49 34) (49 65 50) (49 -65 -50) (-49 65 -50) (-49 -65 50) (65 81 66) (65 -81 -66) (-65 81 -66) (-65 -81 66) (81 97 82) (81 -97 -82) (-81 97 -82) (-81 -97 82) (97 113 98) (97 -113 -98) (-97 113 -98) (-97 -113 98) (113 129 114) (113 -129 -114) (-113 129 -114) (-113 -129 114) (129 145 130) (129 -145 -130) (-129 145 -130) (-129 -145 130) (145 161 146) (145 -161 -146) (-145 161 -146) (-145 -161 146) (161 177 162) (161 -177 -162) (-161 177 -162) (-161 -177 162) (177 193 178) (177 -193 -178) (-177 193 -178) (-177 -193 178) (193 209 194) (193 -209 -194) (-193 209 -194) (-193 -209 194) (209 225 210) (209 -225 -210) (-209 225 -210) (-209 -225 210) (225 241 226) (225 -241 -226) (-225 241 -226) (-225 -241 226) (241 1 242) (241 -1 -242) (-241 1 -242) (-241 -1 242) (2 18 3) (2 -18 -3) (-2 18 -3) (-2 -18 3) (18 34 19) (18 -34 -19) (-18 34 -19) (-18 -34 19) (34 50 35) (34 -50 -35) (-34 50 -35) (-34 -50 35) (50 66 51) (50 -66 -51) (-50 66 -51) (-50 -66 51) (66 82 67) (66 -82 -67) (-66 82 -67) (-66 -82 67) (82 98 83) (82 -98 -83) (-82 98 -83) (-82 -98 83) (98 114 99) (98 -114 -99) (-98 114 -99) (-98 -114 99) (114 130 115) (114 -130 -115) (-114 130 -115) (-114 -130 115) (130 146 131) (130 -146 -131) (-130 146 131) (-130 -146 131) (146 162 147) (146 -162 -147) (-146 162 -147) (-146 -162 147) (162 178 163) (162 -178 -163) (-162 178 -163) (-162 -178 163) (178 194 179) (178 -194 -179) (-178 194 -179) (-178 -194 179) (194 210 195) (194 -210 -195) (-194 210 -195) (-194 -210 195) (210 226 211) (210 -226 -211) (-210 226 -211) (-210 -226 211) (226 242 227) (226 -242 -227) (-226 242 -227) (-226 -242 227) (242 2 243) (242 -2 -243) (-242 2 -243) (-242 -2 243) (3 19 4) (3 -19 -4) (-3 19 -4) (-3 -19 4) (19 35 20) (19 -35 -20) (-19 35 -20) (-19 -35 20) (35 51 36) (35 -51 -36) (-35 51 -36) (-35 -51 36) (51 67 52) (51 -67 -52) (-51 67 -52) (-51 -67 52) (67 83 68) (67 -83 -68) (-67 83 -68) (-67 -83 68) (83 99 84) (83 -99 -84) (-83 99 -84) (-83 -99 84) (99 115 100) (99 -115 -100) (-99 115 -100) (-99 -115 100) (115 131 116) (115 -131 -116) (-115 131 -116) (-115 -131 116) (131 147 132) (131 -147 -132) (-131 147 -132) (-131 -147 132) (147 163 148) (147 -163 -148) (-147 163 -148) (-147 -163 148) (163 179 164) (163 -179 -164) (-163 179 -164) (-163 -179 164) (179 195 180) (179 -195 -180) (-179 195 -180) (-179 -195 180) (195 211 196) (195 -211 -196) (-195 211 -196) (-195 -211 196) (211 227 212) (211 -227 -212) (-211 227 -212) (-211 -227 212) (227 243 228) (227 -243 -228) (-227 243 -228) (-227 -243 228) (243 3 244) (243 -3 -244) (-243 3 -244) (-243 -3 244) (4 20 5) (4 -20 -5) (-4 20 -5) (-4 -20 5) (20 36 21) (20 -36 -21) (-20 36 -21) (-20 -36 21) (36 52 37) (36 -52 -37) (-36 52 -37) (-36 -52 37) (52 68 53) (52 -68 -53) (-52 68 -53) (-52 -68 53) (68 84 69) (68 -84 -69) (-68 84 -69) (-68 -84 69) (84 100 85) (84 -100 -85) (-84 100 -85) (-84 -100 85) (100 116 101) (100 -116 -101) (-100 116 -101) (-100 -116 101) (116 132 117) (116 -132 -117) (-116 132 -117) (-116 -132 117) (132 148 133) (132 -148 -133) (-132 148 -133) (-132 -148 133) (148 164 149) (148 -164 -149) (-148 164 -149) (-148 -164 149) (164 180 165) (164 -180 -165) (-164 180 -165) (-164 -180 165) (180 196 181) (180 -196 -181) (-180 196 -181) (-180 -196 181) (196 212 197) (196 -212 -197) (-196 212 -197) (-196 -212 197) (212 228 213) (212 -228 -213) (-212 228 -213) (-212 -228 213) (228 244 229) (228 -244 -229) (-228 244 -229) (-228 -244 229) (244 4 245) (244 -4 -245) (-244 4 -245) (-244 -4 245) (5 21 6) (5 -21 -6) (-5 21 -6) (-5 -21 6) (21 37 22) (21 -37 -22) (-21 37 -22) (-21 -37 22) (37 53 38) (37 -53 -38) (-37 53 -38) (-37 -53 38) (53 69 54) (53 -69 -54) (-53 69 -54) (-53 -69 54) (69 85 70) (69 -85 -70) (-69 85 -70) (-69 -85 70) (85 101 86) (85 -101 -86) (-85 101 -86) (-85 -101 86) (101 117 102) (101 -117 -102) (-101 117 -102) (-101 -117 102) (117 133 118) (117 -133 -118) (-117 133 -118) (-117 -133 118) (133 149 134) (133 -149 -134) (-133 149 -134) (-133 -149 134) (149 165 150) (149 -165 -150) (-149 165 -150) (-149 -165 150) (165 181 166) (165 -181 -166) (-165 181 -166) (-165 -181 166) (181 197 182) (181 -197 -182) (-181 197 -182) (-181 -197 182) (197 213 198) (197 -213 -198) (-197 213 -198) (-197 -213 198) (213 229 214) (213 -229 -214) (-213 229 -214) (-213 -229 214) (229 245 230) (229 -245 -230) (-229 245 -230) (-229 -245 230) (245 5 246) (245 -5 -246) (-245 5 -246) (-245 -5 246) (6 22 7) (6 -22 -7) (-6 22 -7) (-6 -22 7) (22 38 23) (22 -38 -23) (-22 38 -23) (-22 -38 23) (38 54 39) (38 -54 -39) (-38 54 -39) (-38 -54 39) (54 70 55) (54 -70 -55) (-54 70 -55) (-54 -70 55) (70 86 71) (70 -86 -71) (-70 86 -71) (-70 -86 71) (86 102 87) (86 -102 -87) (-86 102 -87) (-86 -102 87) (102 118 103) (102 -118 -103) (-102 118 -103) (-102 -118 103) (118 134 119) (118 -134 -119) (-118 134 -119) (-118 -134 119) (134 150 135) (134 -150 -135) (-134 150 -135) (-134 -150 135) (150 166 151) (150 -166 -151) (-150 166 -151) (-150 -166 151) (166 182 167) (166 -182 -167) (-166 182 -167) (-166 -182 167) (182 198 183) (182 -198 -183) (-182 198 -183) (-182 -198 183) (198 214 199) (198 -214 -199) (-198 214 -199) (-198 -214 199) (214 230 215) (214 -230 -215) (-214 230 -215) (-214 -230 215) (230 246 231) (230 -246 -231) (-230 246 -231) (-230 -246 231) (246 6 247) (246 -6 -247) (-246 6 -247) (-246 -6 247) (7 23 8) (7 -23 -8) (-7 23 -8) (-7 -23 8) (23 39 24) (23 -39 -24) (-23 39 -24) (-23 -39 24) (39 55 40) (39 -55 -40) (-39 55 -40) (-39 -55 40) (55 71 56) (55 -71 -56) (-55 71 -56) (-55 -71 56) (71 87 72) (71 -87 -72) (-71 87 -72) (-71 -87 72) (87 103 88) (87 -103 -88) (-87 103 -88) (-87 -103 88) (103 119 104) (103 -119 -104) (-103 119 -104) (-103 -119 104) (119 135 120) (119 -135 -120) (-119 135 -120) (-119 -135 120) (135 151 136) (135 -151 -136) (-135 151 -136) (-135 -151 136) (151 167 152) (151 -167 -152) (-151 167 -152) (-151 -167 152) (167 183 168) (167 -183 -168) (-167 183 -168) (-167 -183 168) (183 199 184) (183 -199 -184) (-183 199 -184) (-183 -199 184) (199 215 200) (199 -215 -200) (-199 215 -200) (-199 -215 200) (215 231 216) (215 -231 -216) (-215 231 -216) (-215 -231 216) (231 247 232) (231 -247 -232) (-231 247 -232) (-231 -247 232) (247 7 248) (247 -7 -248) (-247 7 -248) (-247 -7 248) (8 24 9) (8 -24 -9) (-8 24 -9) (-8 -24 9) (24 40 25) (24 -40 -25) (-24 40 -25) (-24 -40 25) (40 56 41) (40 -56 -41) (-40 56 -41) (-40 -56 41) (56 72 57) (56 -72 -57) (-56 72 -57) (-56 -72 57) (72 88 73) (72 -88 -73) (-72 88 -73) (-72 -88 73) (88 104 89) (88 -104 -89) (-88 104 -89) (-88 -104 89) (104 120 105) (104 -120 -105) (-104 120 -105) (-104 -120 105) (120 136 121) (120 -136 -121) (-120 136 -121) (-120 -136 121) (136 152 137) (136 -152 -137) (-136 152 -137) (-136 -152 137) (152 168 153) (152 -168 -153) (-152 168 -153) (-152 -168 153) (168 184 169) (168 -184 -169) (-168 184 -169) (-168 -184 169) (184 200 185) (184 -200 -185) (-184 200 -185) (-184 -200 185) (200 216 201) (200 -216 -201) (-200 216 -201) (-200 -216 201) (216 232 217) (216 -232 -217) (-216 232 -217) (-216 -232 217) (232 248 233) (232 -248 -233) (-232 248 -233) (-232 -248 233) (248 8 249) (248 -8 -249) (-248 8 -249) (-248 -8 249) (8 25 10) (8 -25 -10) (-8 25 -10) (-8 -25 10) (25 41 26) (25 -41 -26) (-25 41 -26) (-25 -41 26) (41 57 42) (41 -57 -42) (-41 57 -42) (-41 -57 42) (57 73 58) (57 -73 -58) (-57 73 -58) (-57 -73 58) (73 89 74) (73 -89 -74) (-73 89 -74) (-73 -89 74) (89 105 90) (89 -105 -90) (-89 105 -90) (-89 -105 90) (105 121 106) (105 -121 -106) (-105 121 -106) (-105 -121 106) (121 137 122) (121 -137 -122) (-121 137 -122) (-121 -137 122) (137 153 138) (137 -153 -138) (-137 153 -138) (-137 -153 138) (153 169 154) (153 -169 -154) (-153 169 -154) (-153 -169 154) (169 185 170) (169 -185 -170) (-169 185 -170) (-169 -185 170) (185 201 186) (185 -201 -186) (-185 201 -186) (-185 -201 186) (201 217 202) (201 -217 -202) (-201 217 -202) (-201 -217 202) (217 233 218) (217 -233 -218) (-217 233 -218) (-217 -233 218) (233 249 234) (233 -249 -234) (-233 249 -234) (-233 -249 234) (249 9 250) (249 -9 -250) (-249 9 -250) (-249 -9 250) (10 26 11) (10 -26 -11) (-10 26 -11) (-10 -26 11) (26 42 27) (26 -42 -27) (-26 42 -27) (-26 -42 27) (42 58 43) (42 -58 -43) (-42 58 -43) (-42 -58 43) (58 74 59) (58 -74 -59) (-58 74 -59) (-58 -74 59) (74 90 75) (74 -90 -75) (-74 90 -75) (-74 -90 75) (90 106 91) (90 -106 -91) (-90 106 -91) (-90 -106 91) (106 122 107) (106 -122 -107) (-106 122 -107) (-106 -122 107) (122 138 123) (122 -138 -123) (-122 138 -123) (-122 -138 123) (138 154 139) (138 -154 -139) (-138 154 -139) (-138 -154 139) (154 170 155) (154 -170 -155) (-154 170 -155) (-154 -170 155) (170 186 171) (170 -186 -171) (-170 186 -171) (-170 -186 171) (186 202 187) (186 -202 -187) (-186 202 -187) (-186 -202 187) (202 218 203) (202 -218 -203) (-202 218 -203) (-202 -218 203) (218 234 219) (218 -234 -219) (-218 234 -219) (-218 -234 219) (234 250 235) (234 -250 -235) (-234 250 -235) (-234 -250 235) (250 10 251) (250 -10 -251) (-250 10 -251) (-250 -10 251) (11 27 12) (11 -27 -12) (-11 27 -12) (-11 -27 12) (27 43 28) (27 -43 -28) (-27 43 -28) (-27 -43 28) (43 59 44) (43 -59 -44) (-43 59 -44) (-43 -59 44) (59 75 60) (59 -75 -60) (-59 75 -60) (-59 -75 60) (75 91 76) (75 -91 -76) (-75 91 -76) (-75 -91 76) (91 107 92) (91 -107 -92) (-91 107 -92) (-91 -107 92) (107 123 108) (107 -123 -108) (-107 123 -108) (-107 -123 108) (123 139 124) (123 -139 -124) (-123 139 -124) (-123 -139 124) (139 155 140) (139 -155 -140) (-139 155 -140) (-139 -155 140) (155 171 156) (155 -171 -156) (-155 171 -156) (-155 -171 156) (171 187 172) (171 -187 -172) (-171 187 -172) (-171 -187 172) (187 203 188) (187 -203 -188) (-187 203 -188) (-187 -203 188) (203 219 204) (203 -219 -204) (-203 219 -204) (-203 -219 204) (219 235 220) (219 -235 -220) (-219 235 -220) (-219 -235 220) (235 251 236) (235 -251 -236) (-235 251 -236) (-235 -251 236) (251 11 252) (251 -11 -252) (-251 11 -252) (-251 -11 252) (12 28 13) (12 -28 -13) (-12 28 -13) (-12 -28 13) (28 44 29) (28 -44 -29) (-28 44 -29) (-28 -44 29) (44 60 45) (44 -60 -45) (-44 60 -45) (-44 -60 45) (60 76 61) (60 -76 -61) (-60 76 -61) (-60 -76 61) (76 92 77) (76 -92 -77) (-76 92 -77) (-76 -92 77) (92 108 93) (92 -108 -93) (-92 108 -93) (-92 -108 93) (108 124 109) (108 -124 -109) (-108 124 -109) (-108 -124 109) (124 140 125) (124 -140 -125) (-124 140 -125) (-124 -140 125) (140 156 141) (140 -156 -141) (-140 156 -141) (-140 -156 141) (156 172 157) (156 -172 -157) (-156 172 -157) (-156 -172 157) (172 188 173) (172 -188 -173) (-172 188 -173) (-172 -188 173) (188 204 189) (188 -204 -189) (-188 204 -189) (-188 -204 189) (204 220 205) (204 -220 -205) (-204 220 -205) (-204 -220 205) (220 236 221) (220 -236 -221) (-220 236 -221) (-220 -236 221) (236 252 237) (236 -252 -237) (-236 252 -237) (-236 -252 237) (252 12 253) (252 -12 -253) (-252 12 -253) (-252 -12 253) (13 29 14) (13 -29 -14) (-13 29 -14) (-13 -29 14) (29 45 30) (29 -45 -30) (-29 45 -30) (-29 -45 30) (45 61 46) (45 -61 -46) (-45 61 -46) (-45 -61 46) (61 77 62) (61 -77 -62) (-61 77 -62) (-61 -77 62) (77 93 78) (77 -93 -78) (-77 93 -78) (-77 -93 78) (93 109 94) (93 -109 -94) (-93 109 -94) (-93 -109 94) (109 125 110) (109 -125 -110) (-109 125 -110) (-109 -125 110) (125 141 126) (125 -141 -126) (-125 141 -126) (-125 -141 126) (141 157 142) (141 -157 -142) (-141 157 -142) (-141 -157 142) (157 173 158) (157 -173 -158) (-157 173 -158) (-157 -173 158) (173 189 174) (173 -189 -174) (-173 189 -174) (-173 -189 174) (189 205 190) (189 -205 -190) (-189 205 -190) (-189 -205 190) (205 221 206) (205 -221 -206) (-205 221 -206) (-205 -221 206) (221 237 222) (221 -237 -222) (-221 237 -222) (-221 -237 222) (237 253 238) (237 -253 -238) (-237 253 -238) (-237 -253 238) (253 13 254) (253 -13 -254) (-253 13 -254) (-253 -13 254) (14 30 15) (14 -30 -15) (-14 30 -15) (-14 -30 15) (30 46 31) (30 -46 -31) (-30 46 -31) (-30 -46 31) (46 62 47) (46 -62 -47) (-46 62 -47) (-46 -62 47) (62 78 63) (62 -78 -63) (-62 78 -63) (-62 -78 63) (78 94 79) (78 -94 -79) (-78 94 -79) (-78 -94 79) (94 110 95) (94 -110 -95) (-94 110 -95) (-94 -110 95) (110 126 111) (110 -126 -111) (-110 126 -111) (-110 -126 111) (126 142 127) (126 -142 -127) (-126 142 -127) (-126 -142 127) (142 158 143) (142 -158 -143) (-142 158 -143) (-142 -158 143) (158 174 159) (158 -174 -159) (-158 174 -159) (-158 -174 159) (174 190 175) (174 -190 -175) (-174 190 -175) (-174 -190 175) (190 206 191) (190 -206 -191) (-190 206 -191) (-190 -206 191) (206 222 207) (206 -222 -207) (-206 222 -207) (-206 -222 207) (222 238 223) (222 -238 -223) (-222 238 -223) (-222 -238 223) (238 254 239) (238 -254 -239) (-238 254 -239) (-238 -254 239) (254 14 255) (254 -14 -255) (-254 14 -255) (-254 -14 255) (15 31 16) (15 -31 -16) (-15 31 -16) (-15 -31 16) (31 47 32) (31 -47 -32) (-31 47 -32) (-31 -47 32) (47 63 48) (47 -63 -48) (-47 63 -48) (-47 -63 48) (63 79 64) (63 -79 -64) (-63 79 -64) (-63 -79 64) (79 95 80) (79 -95 -80) (-79 95 -80) (-79 -95 80) (95 111 96) (95 -111 -96) (-95 111 -96) (-95 -111 96) (111 127 112) (111 -127 -112) (-111 127 -112) (-111 -127 112) (127 143 128) (127 -143 -128) (-127 143 -128) (-127 -143 128) (143 159 144) (143 -159 -144) (-143 159 -144) (-143 -159 144) (159 175 160) (159 -175 -160) (-159 175 -160) (-159 -175 160) (175 191 176) (175 -191 -176) (-175 191 -176) (-175 -191 176) (191 207 192) (191 -207 -192) (-191 207 -192) (-191 -207 192) (207 223 208) (207 -223 -208) (-207 223 -208) (-207 -223 208) (223 239 224) (223 -239 -224) (-223 239 -224) (-223 -239 224) (239 255 240) (239 -255 -240) (-239 255 -240) (-239 -255 240) (255 15 256) (255 -15 -256) (-255 15 -256) (-255 -15 256) (16 32 1) (16 -32 -1) (-16 32 -1) (-16 -32 1) (32 48 17) (32 -48 -17) (-32 48 -17) (-32 -48 17) (48 64 33) (48 -64 -33) (-48 64 -33) (-48 -64 33) (64 80 49) (64 -80 -49) (-64 80 -49) (-64 -80 49) (80 96 65) (80 -96 -65) (-80 96 -65) (-80 -96 65) (96 112 81) (96 -112 -81) (-96 112 -81) (-96 -112 81) (112 128 97) (112 -128 -97) (-112 128 -97) (-112 -128 97) (128 144 113) (128 -144 -113) (-128 144 -113) (-128 -144 113) (144 160 129) (144 -160 -129) (-144 160 -129) (-144 -160 129) (160 176 145) (160 -176 -145) (-160 176 -145) (-160 -176 145) (176 192 161) (176 -192 -161) (-176 192 -161) (-176 -192 161) (192 208 177) (192 -208 -177) (-192 208 -177) (-192 -208 177) (208 224 193) (208 -224 -193) (-208 224 -193) (-208 -224 193) (224 240 209) (224 -240 -209) (-224 240 -209) (-224 -240 209) (240 256 225) (240 -256 -225) (-240 256 -225) (-240 -256 225) (256 16 241) (256 -16 -241) (-256 16 -241) (-256 -16 241) (256 17 241) (256 -17 -241) (-256 17 -241) (-256 -17 241)

Quelle: Code implementiert nach H. Jia, C. Moore, B. Selman: „*From Spin Glasses to Hard Satisfiable Formulas*“, Proceedings of SAT 2004

Das Schematheorem

Täuschungsprobleme

Wenn sich die Lösung eines genetischen Algorithmus tatsächlich durch Anordnen von Bausteinen ergibt, dann stellt sich folgende Frage:

Wie reagiert ein genetischer Algorithmus, wenn die Lösung sich nicht aus solchen Bausteinen zusammensetzt ?

Das Schematheorem

Täuschungsprobleme

Wenn sich die Lösung eines genetischen Algorithmus tatsächlich durch Anordnen von Bausteinen ergibt, dann stellt sich folgende Frage:

Wie reagiert ein genetischer Algorithmus, wenn die Lösung sich nicht aus solchen Bausteinen zusammensetzt ?

Beispiel

Der Bitstring 00000000 repräsentiert die beste Lösung, besitzt also die höchste Fitness: $f(00000000) = \max$.

Das Schematheorem

Täuschungsprobleme

Wenn sich die Lösung eines genetischen Algorithmus tatsächlich durch Anordnen von Bausteinen ergibt, dann stellt sich folgende Frage:

Wie reagiert ein genetischer Algorithmus, wenn die Lösung sich nicht aus aus solchen Bausteinen zusammensetzt ?

Beispiel

Der Bitstring 00000000 repräsentiert die beste Lösung, besitzt also die höchste Fitness: $f(00000000) = \max$.

Die Fitness der anderen Bitstrings wird so definiert, dass sie für die Schemen mit 0 kleiner ist als für solche Schemen, wo die 0 durch 1 ersetzt wurde, z.B:

$$f(\text{*****}0) < f(\text{*****}1)$$

$$f(\text{**}0\text{***}0\text{*}) < f(\text{**}1\text{***}1\text{*})$$

Das Schematheorem

Täuschungsprobleme

Wenn sich die Lösung eines genetischen Algorithmus tatsächlich durch Anordnen von Bausteinen ergibt, dann stellt sich folgende Frage:

Wie reagiert ein genetischer Algorithmus, wenn die Lösung sich nicht aus aus solchen Bausteinen zusammensetzt ?

Beispiel

Der Bitstring 00000000 repräsentiert die beste Lösung, besitzt also die höchste Fitness: $f(00000000) = \max$.

Die Fitness der anderen Bitstrings wird so definiert, dass sie für die Schemen mit 0 kleiner ist als für solche Schemen, wo die 0 durch 1 ersetzt wurde, z.B:

$$f(\text{*****}0) < f(\text{*****}1)$$

$$f(\text{**}0\text{***}0\text{*}) < f(\text{**}1\text{***}1\text{*})$$

Solche Probleme nennt man **Täuschungsprobleme** (deception problems).

Das Schematheorem

Übungsaufgabe

Konstruieren Sie ein Beispiel für ein 4-Bit Täuschungsproblem.

Wie hoch ist die *echte durchschnittliche Fitness* der Schemen $*00*$ und $*01*$?

Betrachten Sie nun eine Population mit den folgenden fünf Lösungskandidaten:

1010

1001

0101

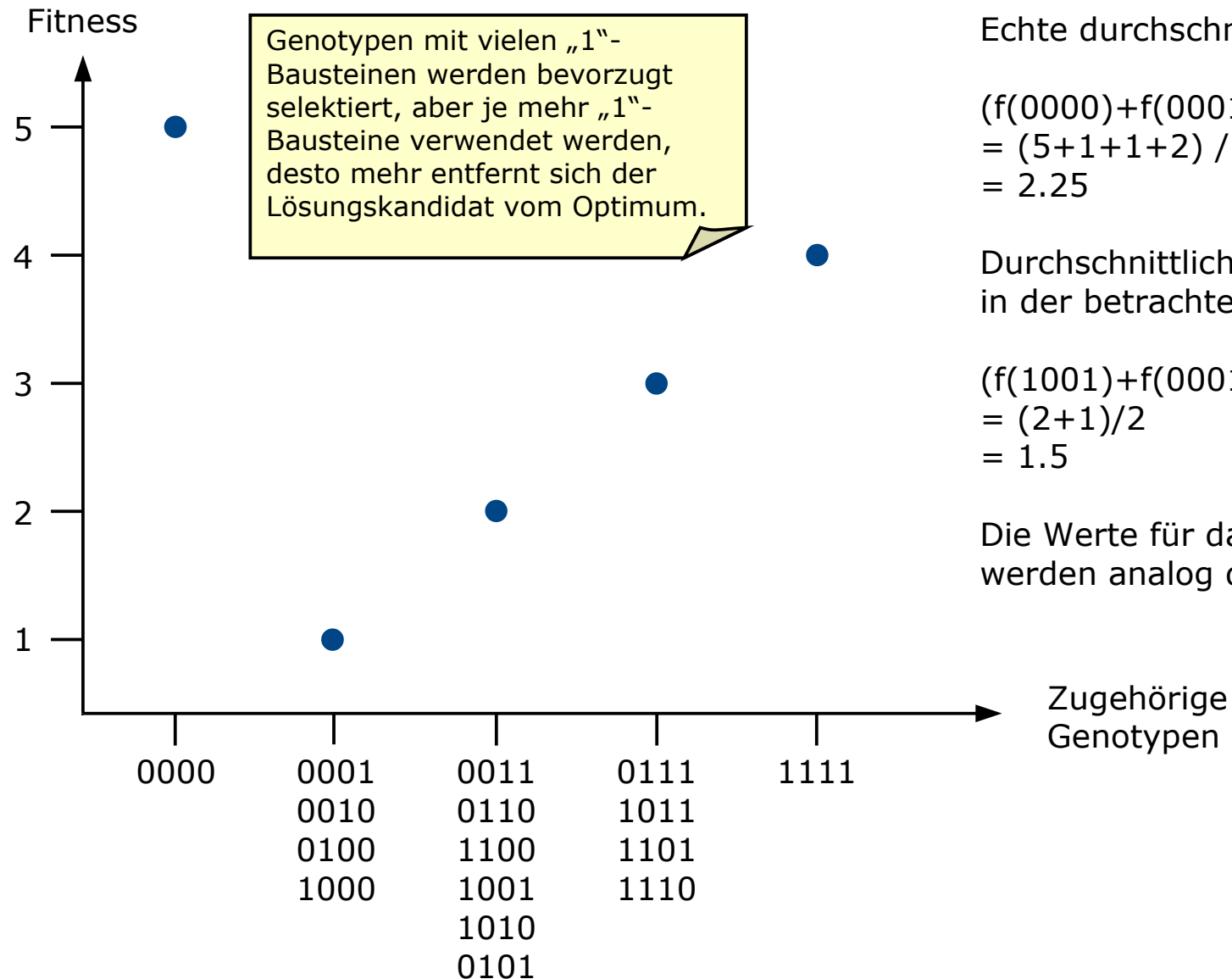
1010

0001

Wie sieht die *durchschnittliche Fitness* der beiden Schemen $*00*$ und $*01*$ in dieser Population aus ?

Das Schematheorem

Beispiel: 4-Bit Täuschungsproblem



Echte durchschnittliche Fitness von *00*

$$\begin{aligned} & (f(0000) + f(0001) + f(1000) + f(1001)) / 4 \\ &= (5 + 1 + 1 + 2) / 4 \\ &= 2.25 \end{aligned}$$

Durchschnittliche Fitness von *00* in der betrachteten Population

$$\begin{aligned} & (f(1001) + f(0001)) / 2 \\ &= (2 + 1) / 2 \\ &= 1.5 \end{aligned}$$

Die Werte für das Schema *01* werden analog dazu berechnet.