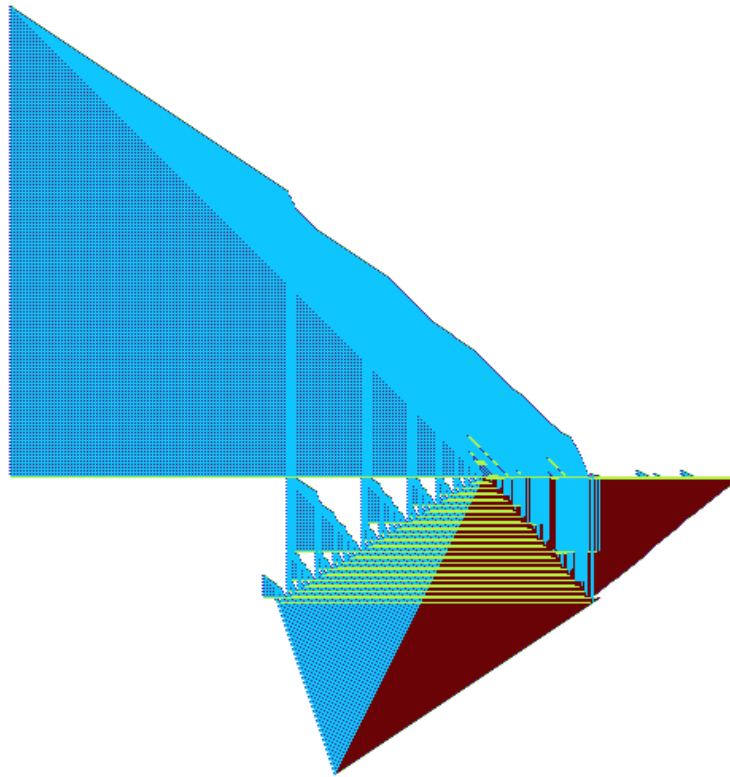


Prof. Dr. Alexander del Pino  
Fachbereich Informatik

Genetische Algorithmen



9. Teil

Koevolution und  
memetische  
Algorithmen

# Koevolution

## Das Gefangenendilemma - Beispiel

*Bonnie* und *Clyde* haben eine Bank überfallen.

Man kann ihnen allerdings nichts direkt nachweisen, deswegen würden beide wegen eines geringfügigen Delikts für 2 Jahre ins Gefängnis kommen.



Bildquelle: Wikipedia

# Koevolution

## Das Gefangenendilemma - Beispiel

*Bonnie* und *Clyde* haben eine Bank überfallen.

Man kann ihnen allerdings nichts direkt nachweisen, deswegen würden beide wegen eines geringfügigen Delikts für 2 Jahre ins Gefängnis kommen.

Die beiden werden in getrennte Zellen gebracht und können nicht miteinander kommunizieren. Dort bietet man beiden eine Kronzeugenregelung an.



Bildquelle: Wikipedia

# Koevolution

## Das Gefangenendilemma - Beispiel

*Bonnie* und *Clyde* haben eine Bank überfallen.

Man kann ihnen allerdings nichts direkt nachweisen, deswegen würden beide wegen eines geringfügigen Delikts für 2 Jahre ins Gefängnis kommen.

Die beiden werden in getrennte Zellen gebracht und können nicht miteinander kommunizieren. Dort bietet man beiden eine Kronzeugenregelung an.

Sagt einer der beiden aus Kronzeuge gegen den anderen aus, dann kommt der Kronzeuge auf Bewährung frei, und der andere kommt 5 Jahre ins Gefängnis.



Bildquelle: Wikipedia

# Koevolution

## Das Gefangenendilemma - Beispiel

*Bonnie* und *Clyde* haben eine Bank überfallen.

Man kann ihnen allerdings nichts direkt nachweisen, deswegen würden beide wegen eines geringfügigen Delikts für 2 Jahre ins Gefängnis kommen.

Die beiden werden in getrennte Zellen gebracht und können nicht miteinander kommunizieren. Dort bietet man beiden eine Kronzeugenregelung an.

Sagt einer der beiden aus Kronzeuge gegen den anderen aus, dann kommt der Kronzeuge auf Bewährung frei, und der andere kommt 5 Jahre ins Gefängnis.

Wenn allerdings beide als Kronzeuge auftreten, ist die Kronzeugenregelung hinfällig und beide kommen für 4 Jahre ins Gefängnis.



Bildquelle: Wikipedia

# Koevolution

## Das Gefangenendilemma

*Robert Axelrod* hat das in den 1950er Jahren von *Merrill Flood* und *Melvin Dresher* entwickelte Gefangenendilemma in Hinblick auf Rüstungswettläufe untersucht.

Under what conditions will cooperation emerge in a world of egoists without central authority ?

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 3



# Koevolution

## Das Gefangenendilemma

*Robert Axelrod* hat das in den 1950er Jahren von *Merrill Flood* und *Melvin Dresher* entwickelte Gefangenendilemma in Hinblick auf Rüstungswettläufe untersucht.

Under what conditions will cooperation emerge in a world of egoists without central authority ?

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 3

*Richard Dawkins* zum Gefangenendilemma:

Many influential people think it holds the key to strategic defence planning, and that we should study it to prevent a third world war. As a biologist, I agree with Axelrod and Hamilton that many wild animals and plants are engaged in ceaseless games of Prisoner's Dilemma, played out in evolutionary time.

Quelle: R. Dawkins. *The Selfish Gene*. Oxford University Press, 30<sup>th</sup> anniversary edition, 2006, S. 203

# Koevolution

## Das Gefangenendilemma

Für die beiden Spieler einem Gefangenendilemma ergeben sich folgende Handlungsmöglichkeiten:

		Spieler 2	
		Kooperiert	Verrät
Spieler 1	Kooperiert	S1+=K; S2+=K Gewinn bei beidseitiger Kooperation	S1+=D; S2+=V Dummgläubigkeit und Versuchung
	Verrät	S1+=V; S2+=D Versuchung und Dummgläubigkeit	S1+=S; S2+=S Strafe bei gegenseitigem Verrat

wobei hier

- $K = 3$  Gewinn bei beidseitiger **Kooperation**
- $D = 0$  Strafe wenn man wegen **Naivität** ausgenutzt wird
- $V = 5$  Gewinn wenn man den anderen Spieler **verraten** hat
- $S = 1$  **Strafe** wenn beide Spieler sich gegenseitig verraten



# Koevolution

## Das Gefangenendilemma



Wie soll man sich am besten verhalten, wenn man sich in einem Gefangenendilemma befindet ?

# Koevolution

## Das Gefangenendilemma



Wie soll man sich am besten verhalten, wenn man sich in einem Gefangenendilemma befindet ?

Im Gegensatz zu einem Nullsummenspiel wie z.B. Schach, wo die Schwäche eines Spielers immer ein Vorteil für den anderen Spielers ist, gibt es bei dem Gefangenendilemma durchaus Strategien, wo dies nicht der Fall ist.

# Koevolution

## Das Gefangenendilemma



Wie soll man sich am besten verhalten, wenn man sich in einem Gefangenendilemma befindet ?

Im Gegensatz zu einem Nullsummenspiel wie z.B. Schach, wo die Schwäche eines Spielers immer ein Vorteil für den anderen Spielers ist, gibt es bei dem Gefangenendilemma durchaus Strategien, wo dies nicht der Fall ist.

In fact, in the Prisoner's Dilemma the strategy that works best depends directly on what strategy the other player is using, and in particular, on whether this strategy leaves room for the development of mutual cooperation.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 15

# Koevolution

## Das Gefangenendilemma



Wie soll man sich am besten verhalten, wenn man sich in einem Gefangenendilemma befindet ?

Im Gegensatz zu einem Nullsummenspiel wie z.B. Schach, wo die Schwäche eines Spielers immer ein Vorteil für den anderen Spielers ist, gibt es bei dem Gefangenendilemma durchaus Strategien, wo dies nicht der Fall ist.

In fact, in the Prisoner's Dilemma the strategy that works best depends directly on what strategy the other player is using, and in particular, on whether this strategy leaves room for the development of mutual cooperation.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 15



Welche Annahme(n) muss man also treffen um mit dem höchsten Gewinn davon zu kommen ?

# Koevolution

## Das wiederholte Gefangenendilemma

Die Entscheidung für eine bestimmte Verhaltensweise wird unter anderem auch dadurch beeinflusst, ob dieser Zug der letzte ist oder nicht:

After all, if you are unlikely to meet the other person again, or if you care little about future payoffs, then you might as well defect now and not worry about the consequences for the future.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 15

Die Frage stellt sich also, wie man sich am besten bei einem **wiederholten Gefangenendilemma** (*iterated prisoner's dilemma*) verhält, wenn nicht bekannt ist, wie oft man sich noch einmal trifft.

# Koevolution

## Suche nach guten Strategien

This process simulates survival of the fittest. A rule that is successful on average with the current distribution of rules in the population will become an even larger proportion of the environment of the other rules in the next generation. [...]

This simulation provides an ecological perspective because there are no new rules of behavior introduced. It differs from an evolutionary perspective, which would allow mutations to introduce new strategies into the environment.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 51

# Koevolution

## Suche nach guten Strategien

This process simulates survival of the fittest. A rule that is successful on average with the current distribution of rules in the population will become an even larger proportion of the environment of the other rules in the next generation. [...]

This simulation provides an ecological perspective because there are no new rules of behavior introduced. It differs from an evolutionary perspective, which would allow mutations to introduce new strategies into the environment.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 51



Die Fitness einer Strategie kann in diesem Beispiel nicht *absolut*, sondern immer nur *relativ* zu anderen Strategien berechnet werden.



# Koevolution

## Suche nach guten Strategien

In dem Turnier wurden die Strategien (*decision rules*) 200 mal wiederholt.

- Wenn beide Spieler immer kooperieren, ergibt sich eine Fitness von 600.
- Wenn beide Spieler niemals kooperieren ergibt sich eine Fitness von 200.
- Die Fitness bewegt sich im Bereich [0 ... 1000].

# Koevolution

## Suche nach guten Strategien

In dem Turnier wurden die Strategien (*decision rules*) 200 mal wiederholt.

- Wenn beide Spieler immer kooperieren, ergibt sich eine Fitness von 600.
- Wenn beide Spieler niemals kooperieren ergibt sich eine Fitness von 200.
- Die Fitness bewegt sich im Bereich [0 ... 1000].
- Als beste Strategie hat sich *Wie Du mir, so ich Dir* (*Tit for tat*) mit einer Fitness von 504 herausgestellt. Sie funktioniert wie folgt:

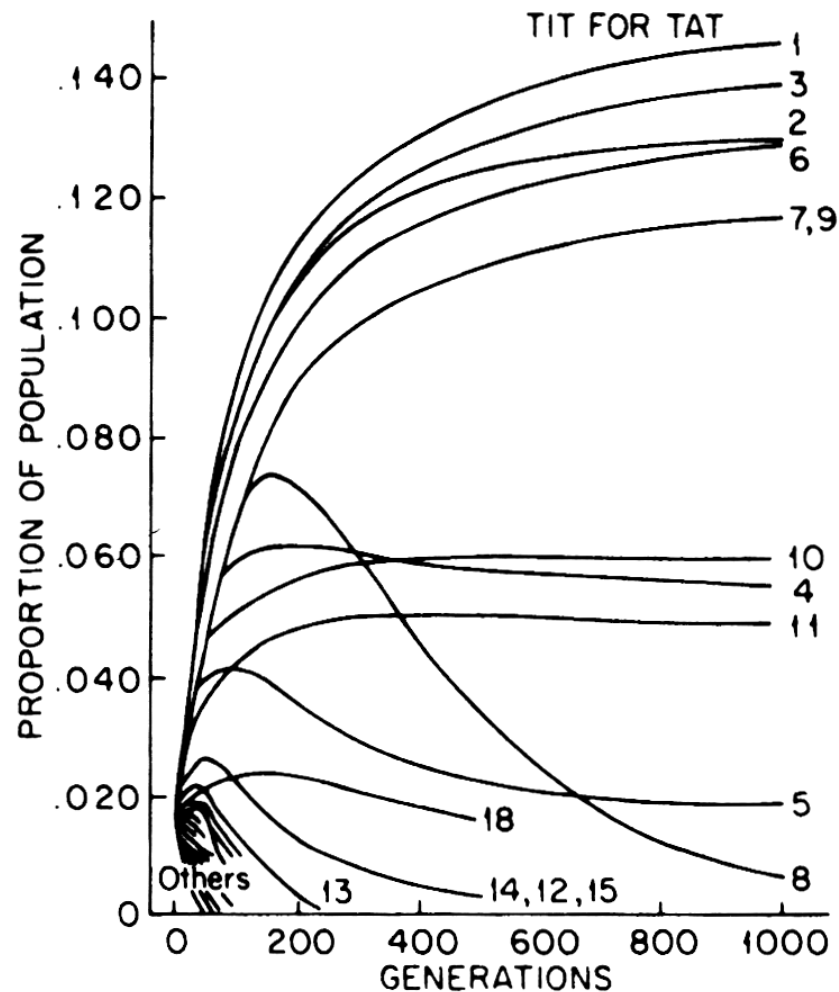
TIT FOR TAT, of course, starts with a cooperative choice, and thereafter does, what the other player did on the previous move.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 31

# Koevolution

## Ergebnisse

Veränderung der relativen Häufigkeit der Strategien.



Bildquelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 51

# Koevolution

## Weitere Ergebnisse

Strategien, welche im ersten Zug kooperieren werden **nette Strategien** (*nice strategies*) genannt.

Surprisingly, there is a single property which distinguishes the relatively high-scoring entries from the relatively low scoring entries. This is the property of being *nice*, which is to say never being the first to defect.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 33

und

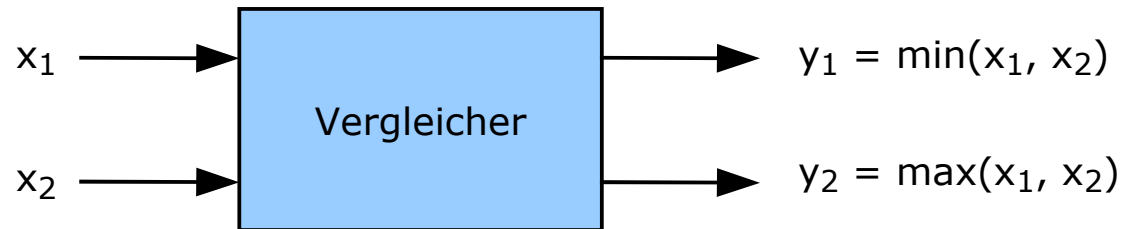
Not being nice may look promising at first, but in the long run it can destroy the very environment it needs for its own success.

Quelle: R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984, S. 52

# Koevolution

## Vergleicher

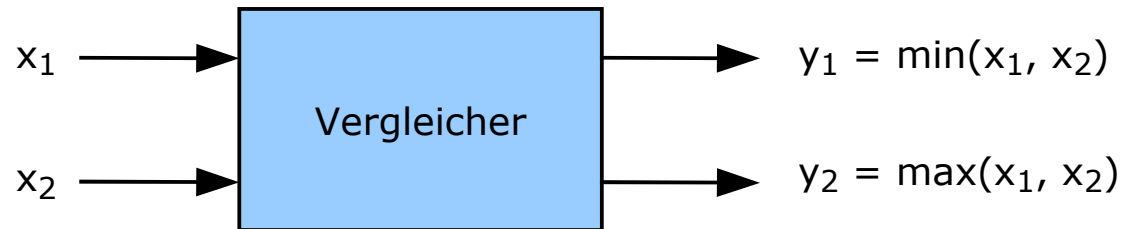
Ein **Vergleicher** (*comparator*) ist ein Element mit zwei Eingängen und zwei Ausgängen. Es vergleicht die beiden Werte an den Eingängen und bietet an einem Ausgang das Minimum, und am anderen Ausgang das Maximum an.



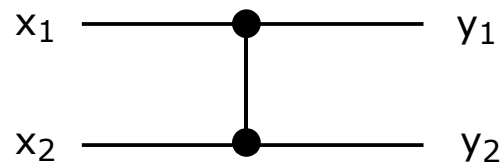
# Koevolution

## Vergleicher

Ein **Vergleicher** (*comparator*) ist ein Element mit zwei Eingängen und zwei Ausgängen. Es vergleicht die beiden Werte an den Eingängen und bietet an einem Ausgang das Minimum, und am anderen Ausgang das Maximum an.



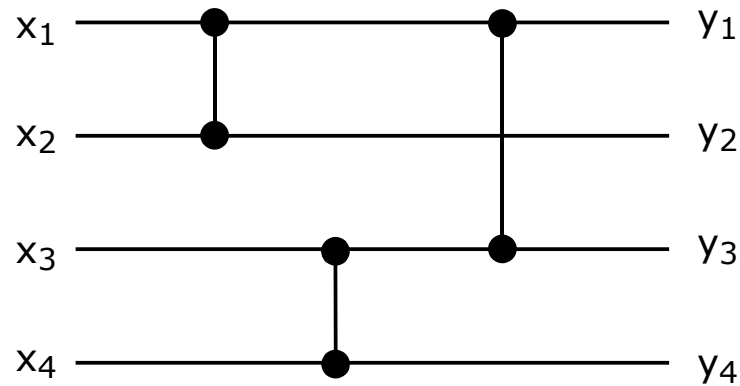
Vereinfacht kann ein Vergleicher wie folgt stilisiert dargestellt werden:



# Koevolution

## Vergleichsnetze

Mehrere Vergleiche können zu einem **Vergleichsnetz** hintereinander geschaltet werden, zum Beispiel:

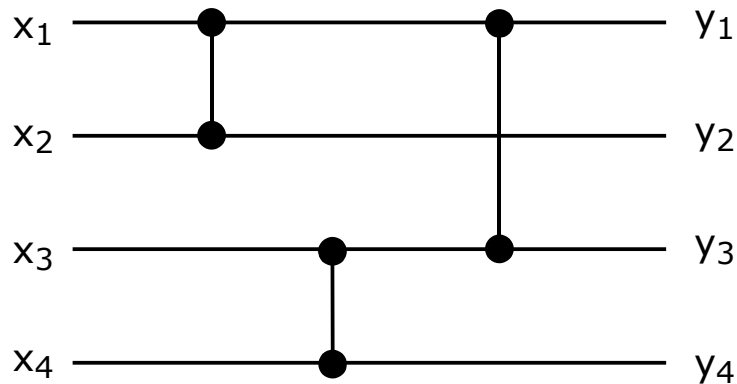




# Koevolution

## Vergleichsnetze

Mehrere Vergleiche können zu einem **Vergleichsnetz** hintereinander geschaltet werden, zum Beispiel:



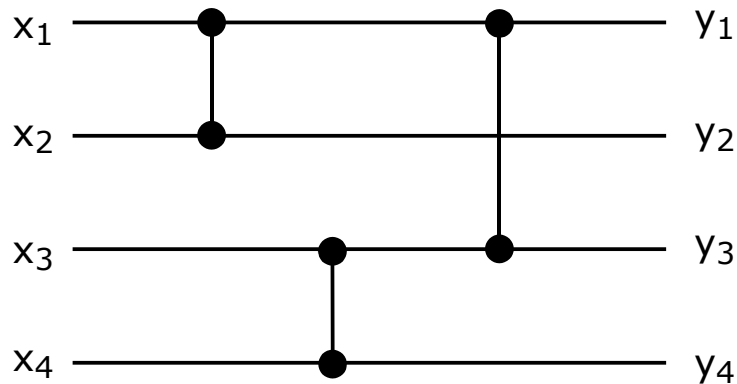
Die Vergleiche  $(x_1:x_2)$  und  $(x_3:x_4)$  können in diesem Beispiel parallel ausgeführt werden, da sie keine gemeinsamen Eingänge besitzen. Der Vergleich  $(x_1:x_3)$  muss nach  $(x_1:x_2)$  und  $(x_3:x_4)$  ausgeführt werden.

In diesem Beispiel hat das Vergleichsnetz also eine **Tiefe (*depth*)** von zwei **Stufen**.

# Koevolution


## Vergleichsnetze

Mehrere Vergleiche können zu einem **Vergleichsnetz** hintereinander geschaltet werden, zum Beispiel:



Die Vergleiche ( $x_1:x_2$ ) und ( $x_3:x_4$ ) können in diesem Beispiel parallel ausgeführt werden, da sie keine gemeinsamen Eingänge besitzen. Der Vergleich ( $x_1:x_3$ ) muss nach ( $x_1:x_2$ ) und ( $x_3:x_4$ ) ausgeführt werden.

In diesem Beispiel hat das Vergleichsnetz also eine **Tiefe (depth)** von zwei **Stufen**.

 Was ergibt sich an den Ausgängen, wenn an diesem Vergleichsnetz die Zahlenfolge 4 3 2 1 an den Eingängen anliegt ?

# Koevolution

## Sortiernetze

Ein **Sortiernetz** (*sorting network*) ist ein Vergleichsnetz, in welchem die Vergleiche so angeordnet sind, dass die Ausgänge *immer* sortiert sind, also

$$y_1 < y_2 < y_3 < \dots < y_n$$

für beliebige Permutationen der Eingangswerte

$$x_1, x_2, x_2, \dots, x_n.$$

# Koevolution

## Sortiernetze

Ein **Sortiernetz** (*sorting network*) ist ein Vergleichsnetz, in welchem die Vergleiche so angeordnet sind, dass die Ausgänge *immer* sortiert sind, also

$$y_1 < y_2 < y_3 < \dots < y_n$$

für beliebige Permutationen der Eingangswerte

$$x_1, x_2, x_2, \dots, x_n.$$

- Das Beispiel auf der vorigen Folie ist also kein Sortiernetz, da es zum Beispiel die Zahlenfolge 4 3 2 1 nicht sortiert.

# Koevolution

## Sortiernetze

Ein **Sortiernetz** (*sorting network*) ist ein Vergleichsnetz, in welchem die Vergleicher so angeordnet sind, dass die Ausgänge *immer* sortiert sind, also

$$y_1 < y_2 < y_3 < \dots < y_n$$

für beliebige Permutationen der Eingangswerte

$$x_1, x_2, x_2, \dots, x_n.$$

- Das Beispiel auf der vorigen Folie ist also kein Sortiernetz, da es zum Beispiel die Zahlenfolge 4 3 2 1 nicht sortiert.
- Es kann gezeigt werden, dass ein Sortiernetz alle Zahlenfolgen richtig sortiert, wenn es jede binäre Folge sortieren kann.

# Koevolution

## Sortiernetze

Ein **Sortiernetz** (*sorting network*) ist ein Vergleichsnetz, in welchem die Vergleiche so angeordnet sind, dass die Ausgänge *immer* sortiert sind, also

$$y_1 < y_2 < y_3 < \dots < y_n$$

für beliebige Permutationen der Eingangswerte

$$x_1, x_2, x_2, \dots, x_n.$$

- Das Beispiel auf der vorigen Folie ist also kein Sortiernetz, da es zum Beispiel die Zahlenfolge 4 3 2 1 nicht sortiert.
- Es kann gezeigt werden, dass ein Sortiernetz alle Zahlenfolgen richtig sortiert, wenn es jede binäre Folge sortieren kann.
- Wenn ein Sortiernetz mit  $n$  Eingängen alle  $2^n$  Bitstrings der Länge  $n$  richtig sortiert, dann sortiert es beliebige Permutationen von Werten an den Eingängen.

# Koevolution

## Sortiernetze für 16 Eingänge

Wie viele Vergleiche benötigt man mindestens um ein Sortiernetz für 16 Eingänge zu konstruieren ?

<b>Jahr</b>	<b>Autor</b>	<b>Vergleiche</b>
1962	<i>Bose, Nelson</i>	65



# Koevolution

## Sortiernetze für 16 Eingänge

Wie viele Vergleiche benötigt man mindestens um ein Sortiernetz für 16 Eingänge zu konstruieren ?

<b>Jahr</b>	<b>Autor</b>	<b>Vergleiche</b>
1962	<i>Bose, Nelson</i>	65
1964	<i>Floyd and Knuth, Batchner</i>	63

# Koevolution

## Sortiernetze für 16 Eingänge

Wie viele Vergleiche benötigt man mindestens um ein Sortiernetz für 16 Eingänge zu konstruieren ?

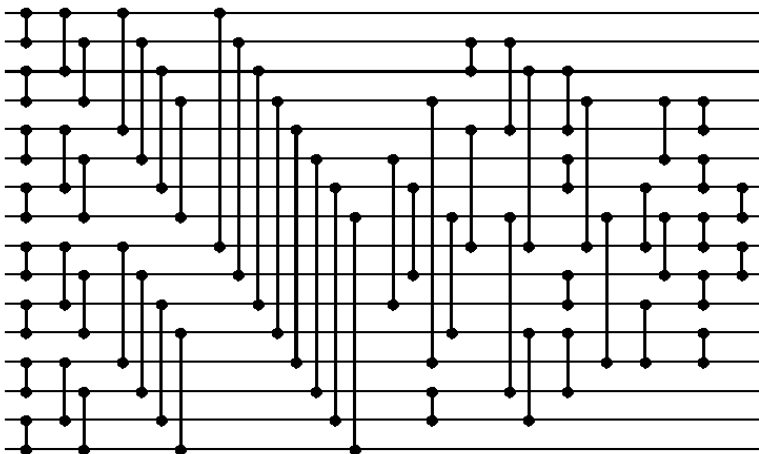
<b>Jahr</b>	<b>Autor</b>	<b>Vergleiche</b>
1962	<i>Bose, Nelson</i>	65
1964	<i>Floyd and Knuth, Batchner</i>	63
1969	<i>Shapiro</i>	62

# Koevolution

## Sortiernetze für 16 Eingänge

Wie viele Vergleiche benötigt man mindestens um ein Sortiernetz für 16 Eingänge zu konstruieren ?

<b>Jahr</b>	<b>Autor</b>	<b>Vergleicher</b>
1962	<i>Bose, Nelson</i>	65
1964	<i>Floyd and Knuth, Batcher</i>	63
1969	<i>Shapiro</i>	62
1969	<i>Green</i>	60



Das Sortiernetz von *Green* mit 60 Vergleichen und einer Tiefe von 10 Stufen.

# Koevolution

## Suche von minimalen Sortiernetzen (*W. D. Hillis, 1990*)

*Hillis'* Ziel war die Minimierung der benötigten Vergleiche, nicht Minimierung der Tiefe des Sortiernetzes.

# Koevolution

## Suche von minimalen Sortiernetzen (*W. D. Hillis, 1990*)

*Hillis'* Ziel war die Minimierung der benötigten Vergleiche, nicht Minimierung der Tiefe des Sortiernetzes.

Die Elemente in der Population sind Sortiernetze, die wie folgt kodiert sind:

- Der Genotyp besteht aus 15 diploiden Chromosomen mit jeweils 32 Bits.

# Koevolution

## Suche von minimalen Sortiernetzen (*W. D. Hillis, 1990*)

*Hillis'* Ziel war die Minimierung der benötigten Vergleiche, nicht Minimierung der Tiefe des Sortiernetzes.

Die Elemente in der Population sind Sortiernetze, die wie folgt kodiert sind:

- Der Genotyp besteht aus 15 diploiden Chromosomen mit jeweils 32 Bits.
- Jedes 32-Bit Wort kodiert acht Codons je vier Bit. Ein Codon entspricht einem Eingang eines Sortiernetzes mit 16 Eingängen.

# Koevolution

## Suche von minimalen Sortiernetzen (*W. D. Hillis, 1990*)

*Hillis'* Ziel war die Minimierung der benötigten Vergleiche, nicht Minimierung der Tiefe des Sortiernetzes.

Die Elemente in der Population sind Sortiernetze, die wie folgt kodiert sind:

- Der Genotyp besteht aus 15 diploiden Chromosomen mit jeweils 32 Bits.
- Jedes 32-Bit Wort kodiert acht Codons je vier Bit. Ein Codon entspricht einem Eingang eines Sortiernetzes mit 16 Eingängen.
- Der Phänotyp ist bei Hillis Verfahren eine geordnete Liste von 60–120 Zahlenpaaren. Ein Zahlenpaar gibt an, welche Eingänge verglichen und ggf. vertauscht werden.



# Koevolution

## Suche von minimalen Sortiernetzen (*W. D. Hillis, 1990*)

*Hillis'* Ziel war die Minimierung der benötigten Vergleiche, nicht Minimierung der Tiefe des Sortiernetzes.

Die Elemente in der Population sind Sortiernetze, die wie folgt kodiert sind:

- Der Genotyp besteht aus 15 diploiden Chromosomen mit jeweils 32 Bits.
- Jedes 32-Bit Wort kodiert acht Codons je vier Bit. Ein Codon entspricht einem Eingang eines Sortiernetzes mit 16 Eingängen.
- Der Phänotyp ist bei Hillis Verfahren eine geordnete Liste von 60–120 Zahlenpaaren. Ein Zahlenpaar gibt an, welche Eingänge verglichen und ggf. vertauscht werden.

Mit dieser Kodierung kann also kein Sortiernetz mit weniger als 60 Vergleichen gefunden werden.

# Koevolution

## Suche von minimalen Sortiernetzen

Die Dekodierung geschieht wie folgt:

- Homozygot - Wenn ein Paar von zwei Codons auf beiden Exemplaren eines Chromosomenpaars identisch ist, dann wird nur das dadurch definierte Zahlenpaar zum Phänotyp hinzugefügt.
- Heterozygot – Andernfalls werden beide Zahlenpaare dem Phänotyp hinzugefügt.

# Koevolution

## Kodierung bei Hillis

Beispiel

```

10110101011110011110010010101001 01001100101101001111010001100011 01100111100000001101001101000111
10110101001001110011110010101001 01010101011101001100011111001100 01101011011110011000000110100100
11010100111101010011110111011110 00011101001011110000101010110111 01100111101100110111100111001111
10111011010001010000000100010010 11000110100001010010111111111100 11100000101101011000011100101111
01011000101000110001110110111001 11011011001110101001001101010110 11010011011101001011000000010010
11111000111001001110010101001001 00110000110011001010111110000110 00011111101111100100110110111001
11010100111101010011110111011110 10111001100010010010101011011000 01010111001011011001110101101000
10111011010001010000000100010010 0001000001110101110101010100011 01010001000001100001101011111111
10100010110111100011101001010011 01100111100000001101001101000111 01011101001000000010111111111111
00010111111100001100000001010010 01101011011110011000000110100100 00101100000110100000000110010110
  
```

(a)

*codons:* 1011 0101 0111 1001 1110 0100 1010 1001

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

*integers:* 11 5 7 9 14 4 10 9

↓

*comparisons to insert in phenotype:* (11, 5) (7, 9) (14, 4) (10, 9)

(b)

<i>chrom. A:</i>	1011 0101	0111 1001	1110 0100	1010 1001
<i>chrom. B:</i>	1011 0101	0010 0111	0011 1100	1010 1001
	(11, 5)	(7, 9), (2, 7)	(14, 4), (3, 12)	(10, 9)

(c)

Bildquelle: M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998, S. 24

# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.

# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.
- Jeder Lösungskandidat wurde auf einem zweidimensionalen Gitter mit Torus-artigen Randbedingungen platziert. Die Koordinaten eines Partners werden mit einer binomialen Approximation einer Normalverteilung ermittelt.

# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.
- Jeder Lösungskandidat wurde auf einem zweidimensionalen Gitter mit Torus-artigen Randbedingungen platziert. Die Koordinaten eines Partners werden mit einer binomialen Approximation einer Normalverteilung ermittelt.
- Die Berechnung erfolgte auf einer *Connection Machine* mit 65536 Prozessoren.

# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.
- Jeder Lösungskandidat wurde auf einem zweidimensionalen Gitter mit Torus-artigen Randbedingungen platziert. Die Koordinaten eines Partners werden mit einer binomialen Approximation einer Normalverteilung ermittelt.
- Die Berechnung erfolgte auf einer *Connection Machine* mit 65536 Prozessoren.
- Verwendung von sehr großen Populationen mit 512 bis eine Million Lösungskandidaten.

# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.
- Jeder Lösungskandidat wurde auf einem zweidimensionalen Gitter mit Torus-artigen Randbedingungen platziert. Die Koordinaten eines Partners werden mit einer binomialen Approximation einer Normalverteilung ermittelt.
- Die Berechnung erfolgte auf einer *Connection Machine* mit 65536 Prozessoren.
- Verwendung von sehr großen Populationen mit 512 bis eine Million Lösungskandidaten.
- Die Simulationen dauerten bis zu 5000 Generationen an.



# Koevolution

## Weitere Besonderheiten / Details

- Die meisten guten Sortiernetze mit 16 Eingängen haben das gleiche Muster bei den ersten 32 Vergleichen. Hillis hat dieses Muster bei der Initialisierung übernommen.
- Jeder Lösungskandidat wurde auf einem zweidimensionalen Gitter mit Torus-artigen Randbedingungen platziert. Die Koordinaten eines Partners werden mit einer binomialen Approximation einer Normalverteilung ermittelt.
- Die Berechnung erfolgte auf einer *Connection Machine* mit 65536 Prozessoren.
- Verwendung von sehr großen Populationen mit 512 bis eine Million Lösungskandidaten.
- Die Simulationen dauerten bis zu 5000 Generationen an.
- Die Fitness eines Sortiernetzes wurde anhand seiner Fähigkeit, ein paar ausgewählte (Zufalls-)Zahlenfolgen zu sortieren, ermittelt.

# Koevolution

## Ergebnisse

Der beste gefundene Lösungskandidat benötigte 65 Vergleiche. Dieses Ergebnis war respektabel, jedoch nicht überragend. Was war passiert ?

After the first few generations, most of the tests performed were sorted successfully by almost all viable networks, so they provided little information about differential fitness. Many of the tests were too "easy." Unfortunately, the discriminative value of a test depends on the solutions that initially evolve [...]

Quelle: W. Daniel Hillis. *Co-evolving parasites improve simulated evolution as an optimization procedure*. Physica D, Vol. 42 (1990), pg. 228-234



Was schlagen Sie vor ?

# Koevolution

## Koevolution von Netzwerken und Testfällen

In the improved procedure, there are two independent gene pools [...]. One population, the "**hosts**", represents sorting networks, while the other population, the "**parasites**", represents test cases. (These two populations might also be considered as "prey" and "predator", since their evolution rates are comparable.) Both populations evolve on the same grid, and their interaction is through their fitness functions.

Quelle: W. Daniel Hillis. *Co-evolving parasites improve simulated evolution as an optimization procedure*. Physica D, Vol. 42 (1990), pg. 228-234

# Koevolution

## Koevolution von Netzwerken und Testfällen

- Der Phänotyp eines Parasiten entspricht 10-20 Testfälle.

# Koevolution

## Koevolution von Netzwerken und Testfällen

- Der Phänotyp eines Parasiten entspricht 10-20 Testfälle.
- Die Fitness eines Parasiten steigt mit der Anzahl der als fehlerhaft erkannten Sortiernetze.

# Koevolution

## Koevolution von Netzwerken und Testfällen

- Der Phänotyp eines Parasiten entspricht 10-20 Testfälle.
- Die Fitness eines Parasiten steigt mit der Anzahl der als fehlerhaft erkannten Sortiernetze.
- Die Fitness eines Netzwerks steigt mit der Anzahl der erfolgreich bestandenen Testfälle.



Was erwarten Sie als Ergebnis ?

# Koevolution

## Koevolution von Netzwerken und Testfällen

- Der Phänotyp eines Parasiten entspricht 10-20 Testfälle.
- Die Fitness eines Parasiten steigt mit der Anzahl der als fehlerhaft erkannten Sortiernetze.
- Die Fitness eines Netzwerks steigt mit der Anzahl der erfolgreich bestandenen Testfälle.



Was erwarten Sie als Ergebnis ?



Eine Art Wettrüsten zwischen Netzwerken und Testfällen findet statt. Das beste auf diese Weise gefundene Sortiernetz benötigte nur 61 (statt 65) Vergleiche.

# Koevolution

## Eigenschaften von Koevolutionsalgorithmen (KEA)

Die Berechnung der Fitness eines Kandidaten in einem KEA basiert auf der Interaktion mit anderen Kandidaten.



Die Fitness hängt davon ab, mit welchen anderen Kandidaten diese Interaktion stattfindet und ist somit immer subjektiv.



# Koevolution

## Eigenschaften von Koevolutionsalgorithmen (KEA)

Die Berechnung der Fitness eines Kandidaten in einem KEA basiert auf der Interaktion mit anderen Kandidaten.

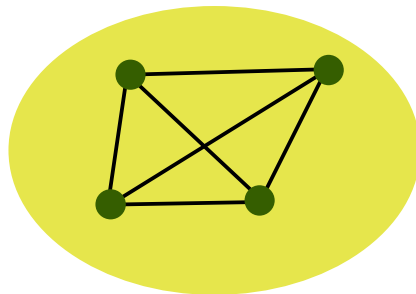


Die Fitness hängt davon ab, mit welchen anderen Kandidaten diese Interaktion stattfindet und ist somit immer subjektiv.

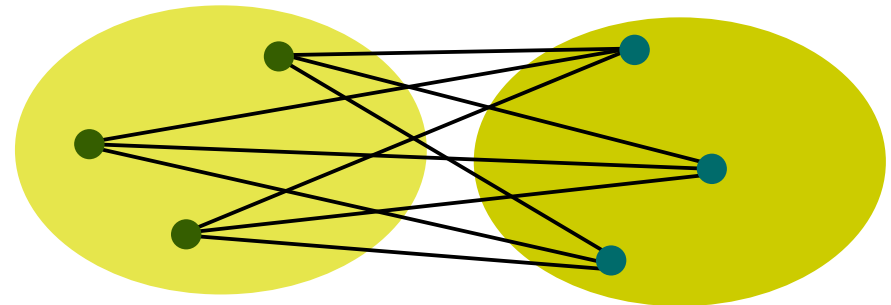
Ein KEA kann eine oder mehrere Populationen benutzen. Im letzteren Fall findet die Interaktion zwischen Individuen in verschiedenen Populationen statt.

Beispiel

KEA mit einer Population



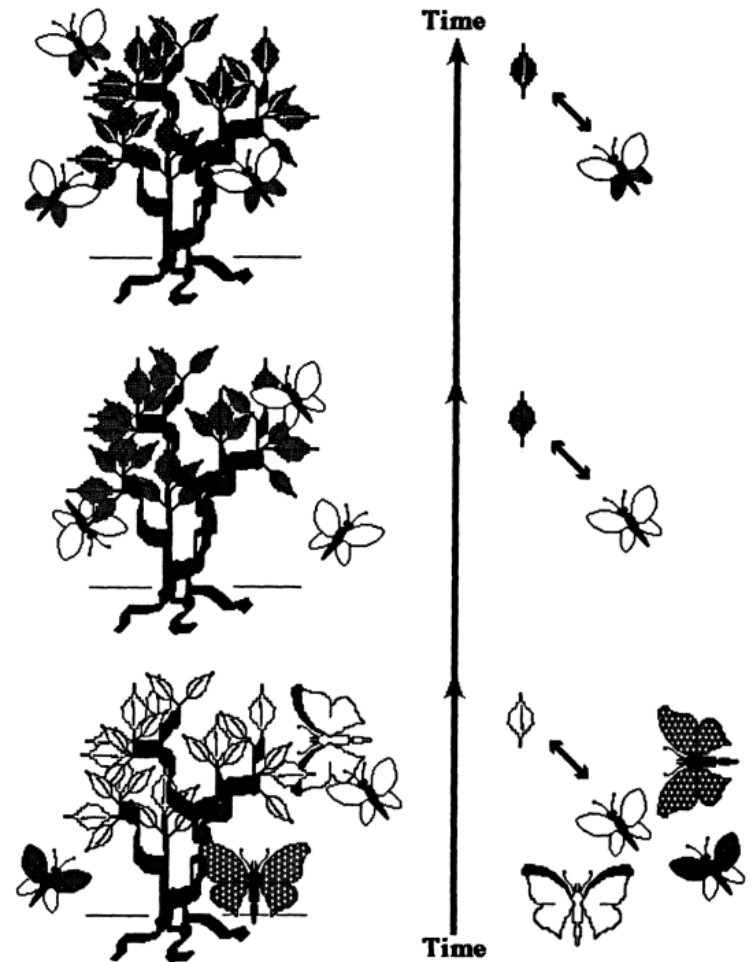
KEA mit  $n=2$  Populationen



# Koevolution

## Konkurrierender KEA

Man kann einen KEA danach unterscheiden, ob er **konkurrierend** (*competitive*) oder **kooperativ** (*cooperative*) ist.



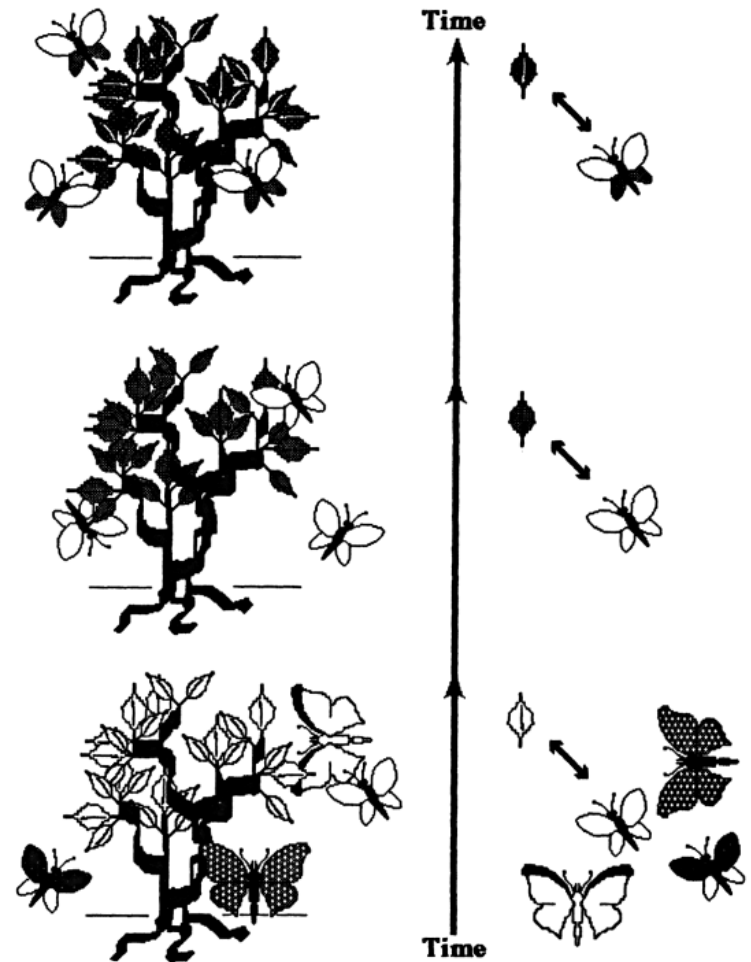
Bildquelle: J. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, S. 30

# Koevolution

## Konkurrierender KEA

Man kann einen KEA danach unterscheiden, ob er **konkurrierend** (*competitive*) oder **kooperativ** (*cooperative*) ist.

Bei einem **konkurrierenden KEA** repräsentieren die Kandidaten Lösungen und Tests (oder Räuber und Beute) für diese Lösungen.



Bildquelle: J. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, S. 30

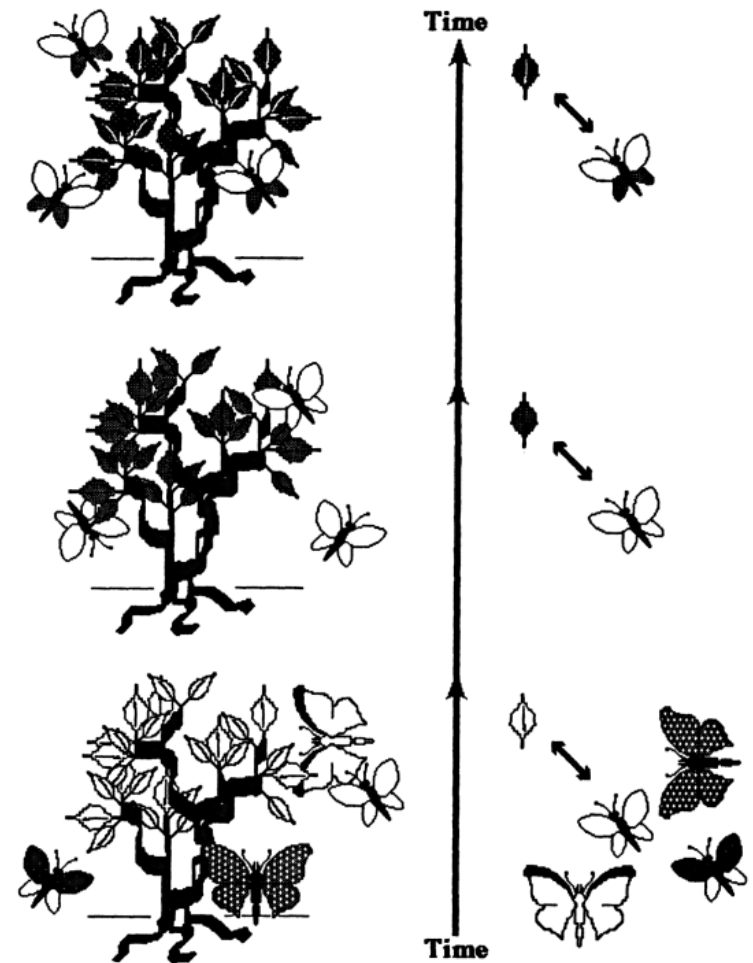
# Koevolution

## Konkurrierender KEA

Man kann einen KEA danach unterscheiden, ob er **konkurrierend** (*competitive*) oder **kooperativ** (*cooperative*) ist.

Bei einem **konkurrierenden KEA** repräsentieren die Kandidaten Lösungen und Tests (oder Räuber und Beute) für diese Lösungen.

Das Ziel eines konkurrierenden KEA ist es, ein **Wettrüsten** (*arms race*) zwischen den Lösungen und ihren Tests hervorzurufen.



Bildquelle: J. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, S. 30

# Koevolution

## Kooperativer KEA

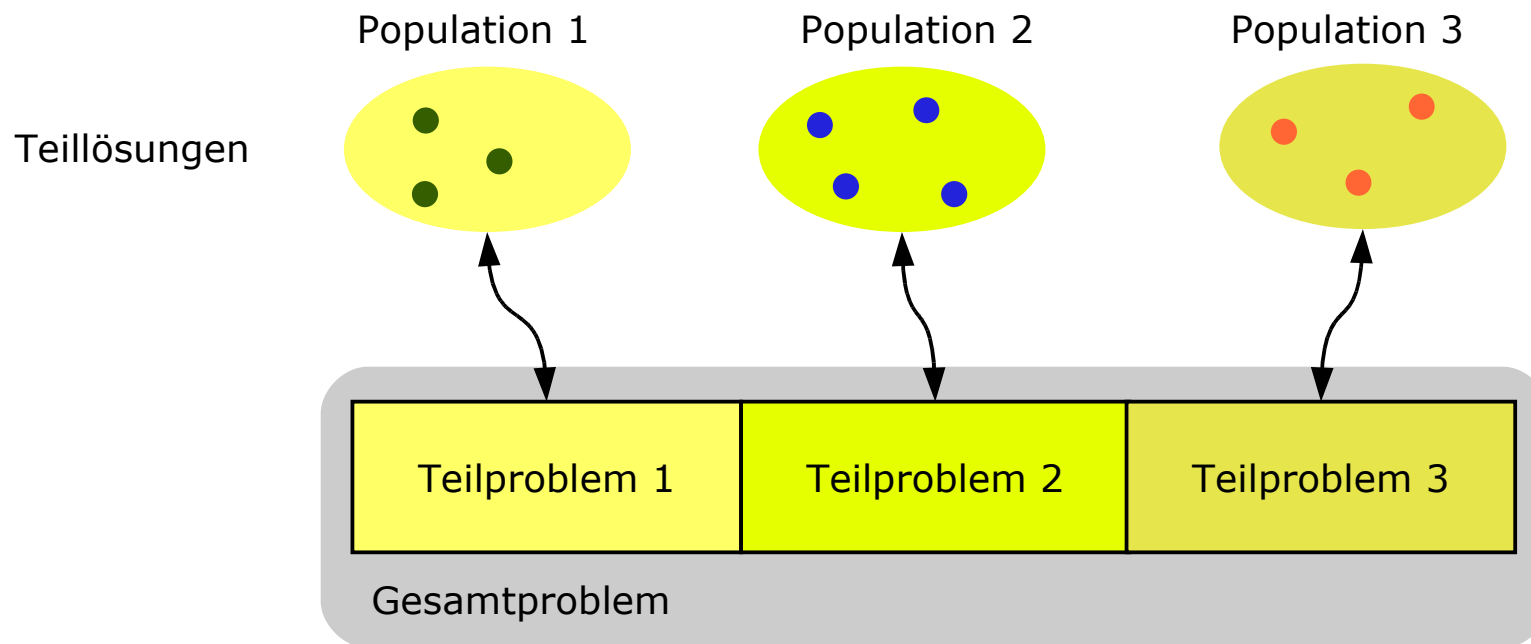
Bei einem **kooperativen KEA** werden Lösungen aus mehreren Kandidaten zusammengesetzt. Ein Kandidat ist somit eine Lösungskomponente oder eine Teillösung.

# Koevolution

## Kooperativer KEA

Bei einem **kooperativen KEA** werden Lösungen aus mehreren Kandidaten zusammengesetzt. Ein Kandidat ist somit eine Lösungskomponente oder eine Teillösung.

Das Ziel eines kooperativen KEA ist es, die Robustheit von zusammengesetzten Lösungen zu verbessern.

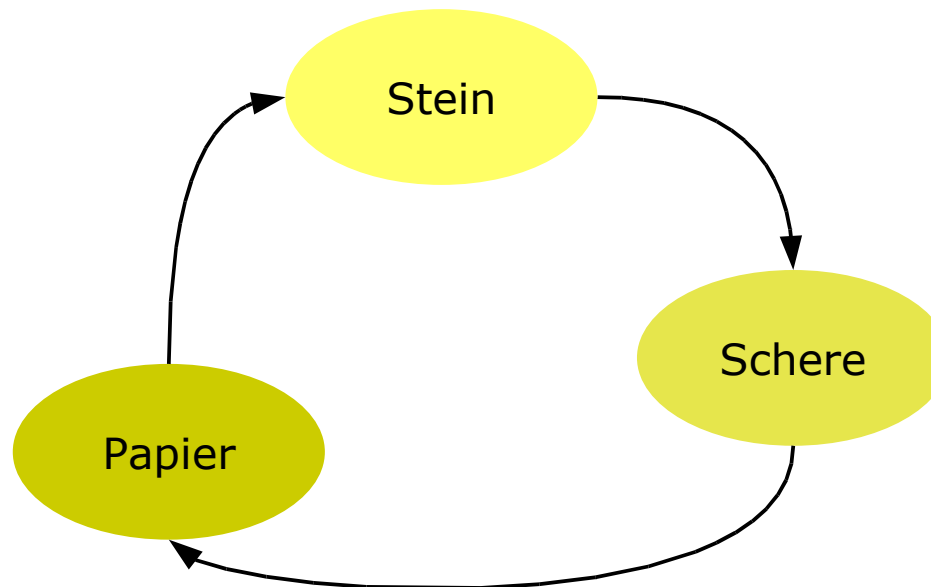


# Koevolution

## Oszillieren

Man fand ziemlich schnell heraus, dass KEAs mit ihren eigenen Problemen behaftet sind. Vor allem bei intransitiven Problemen kann ein **Oszillieren** der Fitness über die Zeitachse (*cycling*) auftreten.

Das Spiel **Stein, Schere, Papier** ist ein Beispiel für ein intransitives Problem.



# Koevolution

## Der rote Königin Effekt (*van Valen, 1973*)

Als den **rote Königin Effekt** bezeichnet man in der Biologie die Beobachtung, dass sich trotz kontinuierlicher genetischer Veränderungen die Überlebenswahrscheinlichkeit einer Spezies nicht verändert, weil gleichzeitig auch Änderungen in der Umgebung stattfinden.



# Koevolution

## Der rote Königin Effekt (*van Valen, 1973*)

Als den **rote Königin Effekt** bezeichnet man in der Biologie die Beobachtung, dass sich trotz kontinuierlicher genetischer Veränderungen die Überlebenswahrscheinlichkeit einer Spezies nicht verändert, weil gleichzeitig auch Änderungen in der Umgebung stattfinden.

Bei einem KEA bedeutet der rote Königin Effekt, dass sich die Selektionswahrscheinlichkeit eines Individuums trotz Verbesserungen wegen den Änderungen von anderen Individuen die sich ebenfalls verändern, nicht erhöht.

# Koevolution

## Der rote Königin Effekt (*van Valen, 1973*)

Als den **rote Königin Effekt** bezeichnet man in der Biologie die Beobachtung, dass sich trotz kontinuierlicher genetischer Veränderungen die Überlebenswahrscheinlichkeit einer Spezies nicht verändert, weil gleichzeitig auch Änderungen in der Umgebung stattfinden.

Bei einem KEA bedeutet der rote Königin Effekt, dass sich die Selektionswahrscheinlichkeit eines Individuums trotz Verbesserungen wegen den Änderungen von anderen Individuen die sich ebenfalls verändern, nicht erhöht.



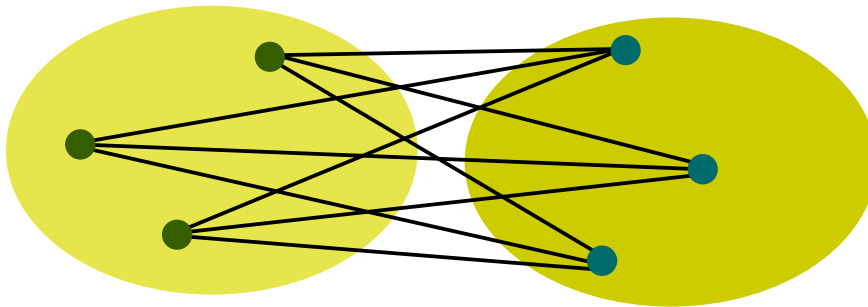
Der rote Königin Effekt ist problematisch, weil man dadurch ein erwünschtes Wettrüsten nicht von dem unerwünschten Oszillieren der Fitness unterscheiden kann.

# Koevolution

## Berechnung der Fitness

Bei der Berechnung der Fitness gibt es verschiedene Vorgehensweisen:

- Jeder mit/gegen jedem (ggf. zu rechenintensiv).

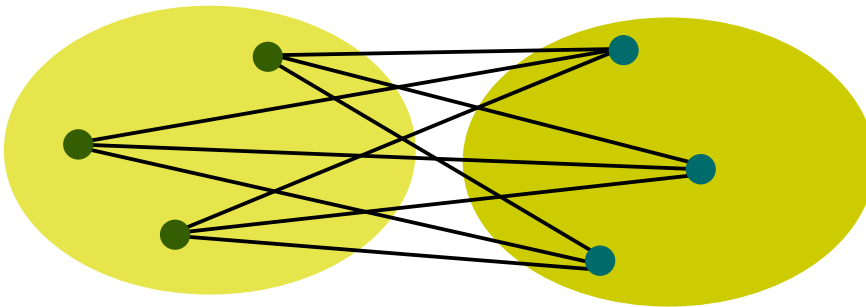


# Koevolution

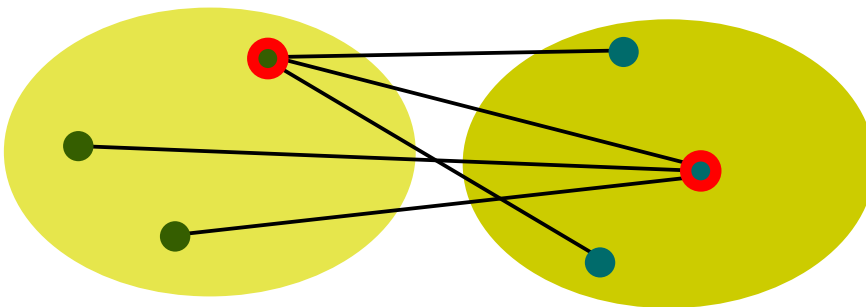
## Berechnung der Fitness

Bei der Berechnung der Fitness gibt es verschiedene Vorgehensweisen:

- Jeder mit/gegen jedem (ggf. zu rechenintensiv).



- Jeder gegen den jeweils Besten (benötigt Elitismus).



# Koevolution

## Literatur

### Gefangenendilemma:

- R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984
- H.Y. Quek, K.C. Tan, C.K. Goh, H.A. Abbass. *Evolution and Incremental Learning in the Iterated Prisoner's Dilemma*. IEEE Transactions on Evolutionary Computation, Vol. 13 No. 2 (April 2009), pp. 303-320

### Sortiernetze:

- W. Daniel Hillis. *Co-evolving parasites improve simulated evolution as an optimization procedure*. Physica D, Vol. 42 (1990), pp. 228-234
- Siehe auch: M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998, pp. 21-27

### Koevolutionäre Algorithmen:

- E. de Jong, K. Stanley, R. Paul Wiegand. *Introductory Tutorial on Coevolution*. Proceedings of GECCO '07, pp. 3133-3157
- S. G. Ficici, A. Bucci. *Advanced Tutorial on Coevolution*. Proceedings of GECCO '07, pp. 3172-3204

# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

Die Fitness eines Lebewesens wird allerdings nicht ausschließlich durch dessen genetische Anlagen festgelegt, sondern auch stark durch die **Erfahrungen und Lernprozesse**, welche dieses Lebewesen durchläuft.

# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

Die Fitness eines Lebewesens wird allerdings nicht ausschließlich durch dessen genetische Anlagen festgelegt, sondern auch stark durch die **Erfahrungen und Lernprozesse**, welche dieses Lebewesen durchläuft.

**Memetische Algorithmen** stellen eine Variante von genetischen Algorithmen dar, bei denen das auch Konzept einer **kulturellen Evolution** berücksichtigt wird.



# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

Die Fitness eines Lebewesens wird allerdings nicht ausschließlich durch dessen genetische Anlagen festgelegt, sondern auch stark durch die **Erfahrungen und Lernprozesse**, welche dieses Lebewesen durchläuft.

**Memetische Algorithmen** stellen eine Variante von genetischen Algorithmen dar, bei denen das auch Konzept einer **kulturellen Evolution** berücksichtigt wird.

Der Begriff **memetischer Algorithmus** wurde etwa 1989 von *Pablo Moscato* geprägt und basiert auf dem Konzept der **Meme** von *Richard Dawkins*.

# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

Die Fitness eines Lebewesens wird allerdings nicht ausschließlich durch dessen genetische Anlagen festgelegt, sondern auch stark durch die **Erfahrungen und Lernprozesse**, welche dieses Lebewesen durchläuft.

**Memetische Algorithmen** stellen eine Variante von genetischen Algorithmen dar, bei denen das auch Konzept einer **kulturellen Evolution** berücksichtigt wird.

Der Begriff **memetischer Algorithmus** wurde etwa 1989 von *Pablo Moscato* geprägt und basiert auf dem Konzept der **Meme** von *Richard Dawkins*.

Daneben wird manchmal auch der Begriff **hybrider genetischer Algorithmus** verwendet.

# Memetische Algorithmen

## Worum geht es ?

Bisher haben wir uns mit Algorithmen beschäftigt, welche nach den Konzepten der **biologischen Evolution** – Selektion, Mutation und Rekombination - konzipiert sind.

Die Fitness eines Lebewesens wird allerdings nicht ausschließlich durch dessen genetische Anlagen festgelegt, sondern auch stark durch die **Erfahrungen und Lernprozesse**, welche dieses Lebewesen durchläuft.

**Memetische Algorithmen** stellen eine Variante von genetischen Algorithmen dar, bei denen das auch Konzept einer **kulturellen Evolution** berücksichtigt wird.

Der Begriff **memetischer Algorithmus** wurde etwa 1989 von *Pablo Moscato* geprägt und basiert auf dem Konzept der **Meme** von *Richard Dawkins*.

Daneben wird manchmal auch der Begriff **hybrider genetischer Algorithmus** verwendet.

Memetische Algorithmen sind für viele Probleme schneller als vergleichbare klassische genetische Algorithmen und stellen oftmals den *state-of-the-art* dar.

# Memetische Algorithmen

## Kulturelle Evolution (cultural evolution)

*Richard Dawkins:*

Most of what is unusual about man can be summed up in one word: 'culture'. I use the word not in its snobbish sense, but as a scientist uses it. Cultural transmission is analogous to genetic transmission in that, although basically conservative, it can give rise to a form of evolution.

[...]

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Kulturelle Evolution (cultural evolution)

*Richard Dawkins:*

Most of what is unusual about man can be summed up in one word: 'culture'. I use the word not in its snobbish sense, but as a scientist uses it. Cultural transmission is analogous to genetic transmission in that, although basically conservative, it can give rise to a form of evolution.

[...]

It is our own species that really shows, what cultural evolution can do. Language is only one example out of many. Fashions in dress and diet, ceremonies and customs, art and architecture, engineering and technology, all evolve in historical time in a way that looks like highly speeded up genetic evolution, but has really nothing to do with genetic evolution.

[...]

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Replikatoren und Meme

*Richard Dawkins:*

What, after all, is so special about genes? The answer is that they are **replicators**. [...] But do we have to go to distant worlds to find other kinds of replicator and other, consequent, kinds of evolution? I think that a new kind of replicator has recently emerged on this very planet. [...]

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Replikatoren und Meme

*Richard Dawkins:*

What, after all, is so special about genes? The answer is that they are **replicators**. [...] But do we have to go to distant worlds to find other kinds of replicator and other, consequent, kinds of evolution? I think that a new kind of replicator has recently emerged on this very planet. [...]

We need a name for the new replicator, a noun that conveys the idea of a unit of cultural transmission, or a unit of imitation. 'Mimeme' comes from a suitable greek root, but I want a monosyllable that sounds a bit like 'gene'. I hope my classicist friends will forgive me if I abbreviate mimeme to **meme**. [...]

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Replikatoren und Meme

*Richard Dawkins:*

What, after all, is so special about genes? The answer is that they are **replicators**. [...] But do we have to go to distant worlds to find other kinds of replicator and other, consequent, kinds of evolution? I think that a new kind of replicator has recently emerged on this very planet. [...]

We need a name for the new replicator, a noun that conveys the idea of a unit of cultural transmission, or a unit of imitation. 'Mimeme' comes from a suitable greek root, but I want a monosyllable that sounds a bit like 'gene'. I hope my classicist friends will forgive me if I abbreviate mimeme to **meme**. [...]

Examples of memes are tunes, ideas, catch-phrases, clothes fashions, ways of making pots, or of building arches.

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).



# Memetische Algorithmen

## Meme

*Richard Dawkins:*

When you plant a fertile meme in my mind, you literally **parasitize** my brain, turning it into a vehicle for the meme's propagation in just the way that a virus may parasitize the genetic mechanism of a host cell. [...]

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Meme

*Richard Dawkins:*

When you plant a fertile meme in my mind, you literally **parasitize** my brain, turning it into a vehicle for the meme's propagation in just the way that a virus may parasitize the genetic mechanism of a host cell. [...]

The computers in which memes live are human brains.

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Meme

*Richard Dawkins:*

When you plant a fertile meme in my mind, you literally **parasitize** my brain, turning it into a vehicle for the meme's propagation in just the way that a virus may parasitize the genetic mechanism of a host cell. [...]

The computers in which memes live are human brains.

Time is possibly a more important limiting factor than storage space, and it is the subject of heavy competition. The human brain, and the body that it controls, cannot do more than one or a few things at once.

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Meme

*Richard Dawkins:*

When you plant a fertile meme in my mind, you literally **parasitize** my brain, turning it into a vehicle for the meme's propagation in just the way that a virus may parasitize the genetic mechanism of a host cell. [...]

The computers in which memes live are human brains.

Time is possibly a more important limiting factor than storage space, and it is the subject of heavy competition. The human brain, and the body that it controls, cannot do more than one or a few things at once.

If a meme is to dominate the attention of a human brain, it must do so at the expense of 'rival' memes.

Other commodities for which memes compete are radio and television time, billboard space, newspaper column-inches, and library shelf-space.

Quelle: R. Dawkins: *The Selfish Gene*. Oxford University Press, 30th anniversary edition, 2006, (first published 1976).

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
}
```



# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
}
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
    Mutation;  
}
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
    Mutation;  
    Verbesserung durch lokale Suche;  
}
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
    Mutation;  
    Verbesserung durch lokale Suche;  
    Selektion der Nachkommen;  
}
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
    Mutation;  
    Verbesserung durch lokale Suche;  
    Selektion der Nachkommen;  
    generation++;  
}
```

# Memetische Algorithmen

## Ein einfacher memetischer Algorithmus

Ein einfacher memetischer Algorithmus ist ein genetischer Algorithmus, welcher die Fähigkeit hat, durch eine lokale Suche zu lernen.

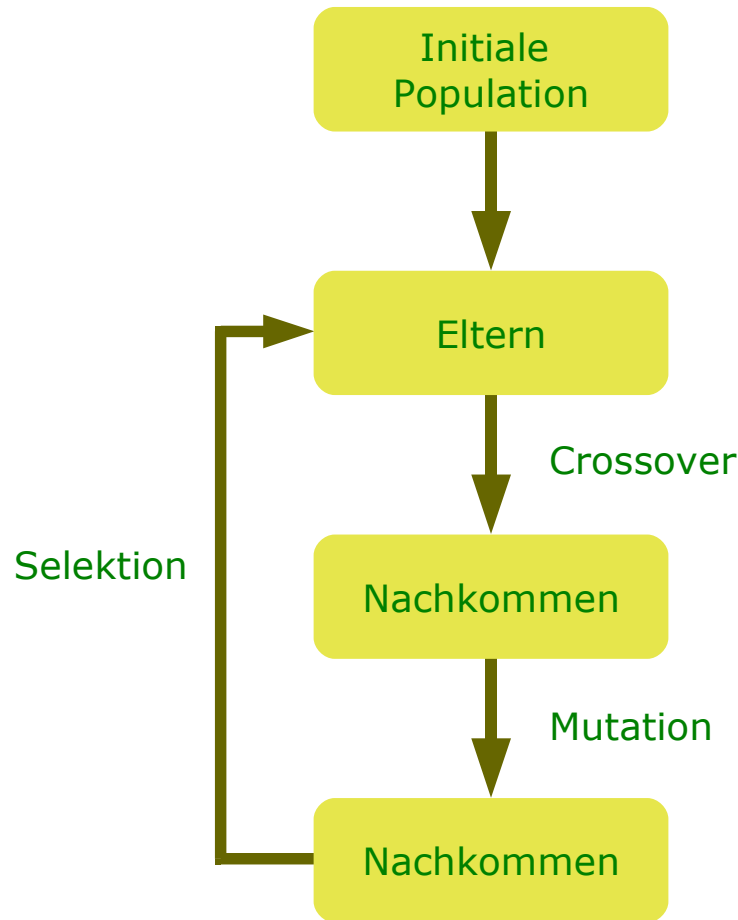
Beispiel

```
generation = 0;  
Erzeuge eine initiale Population P;  
while (Abbruchbedingung nicht erreicht) {  
    Rekombination;  
    Mutation;  
    Verbesserung durch lokale Suche;  
    Selektion der Nachkommen;  
    generation++;  
}  
return beste gefundene Lösung;
```

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:

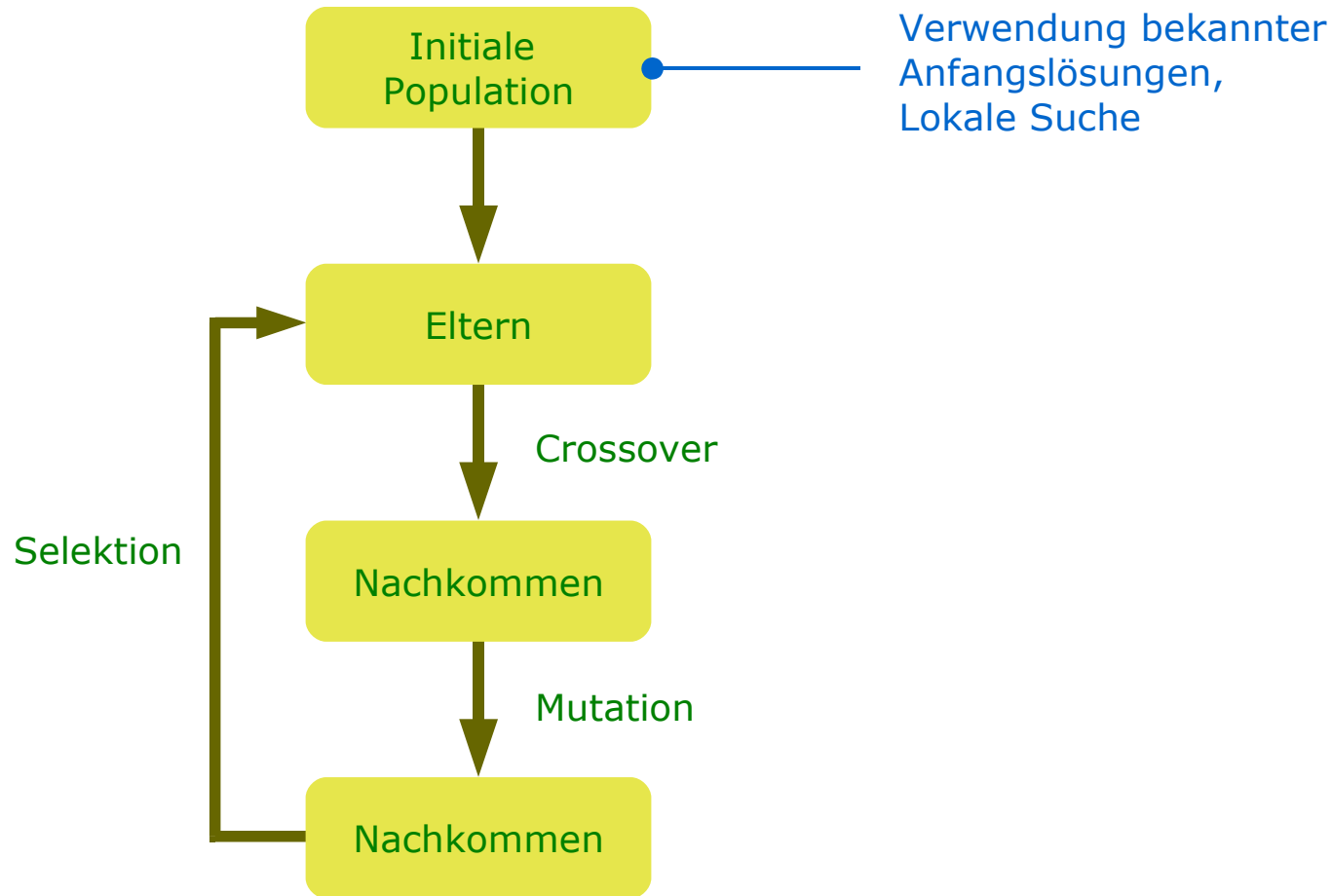


Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:



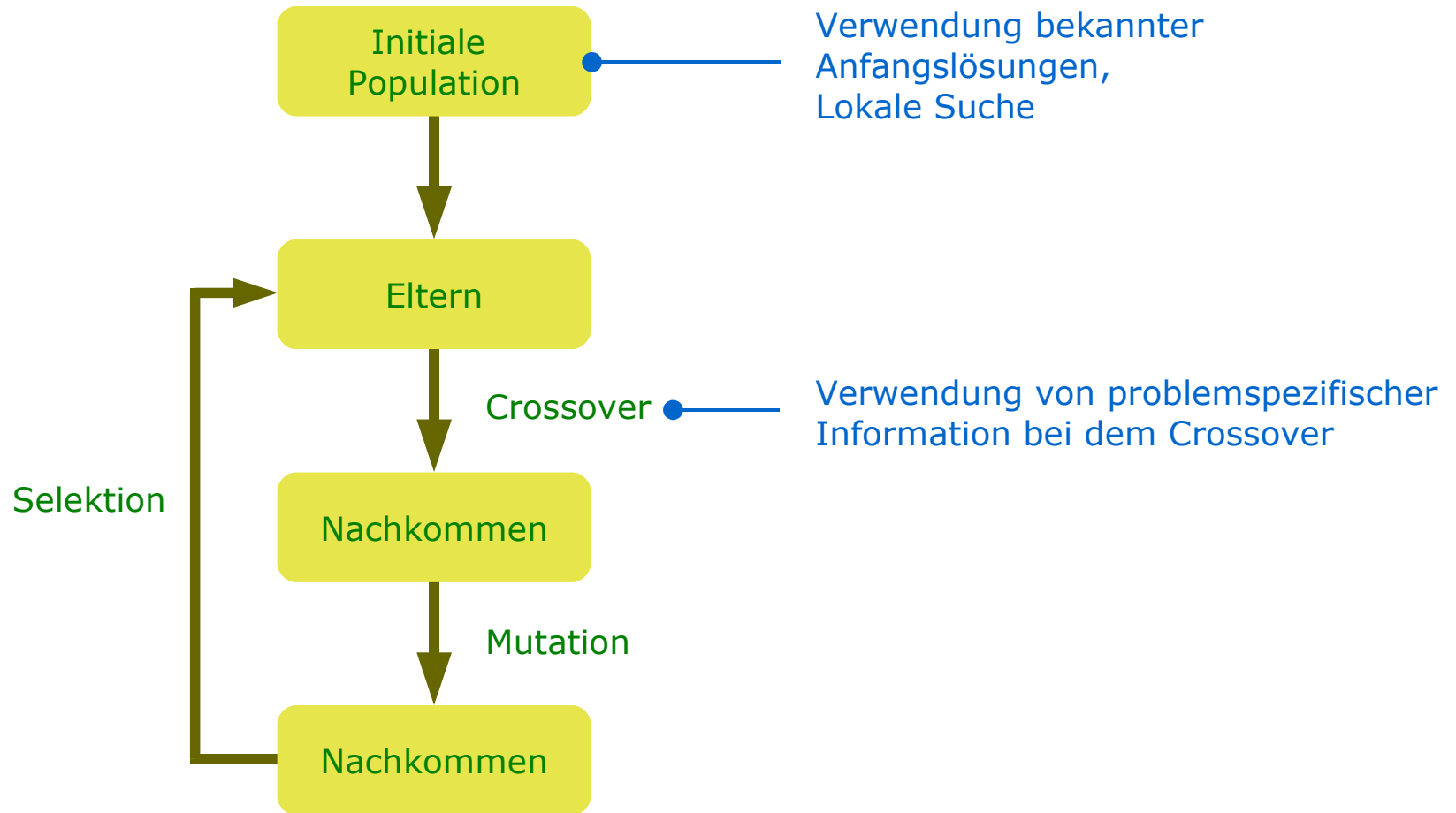
Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179



# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:

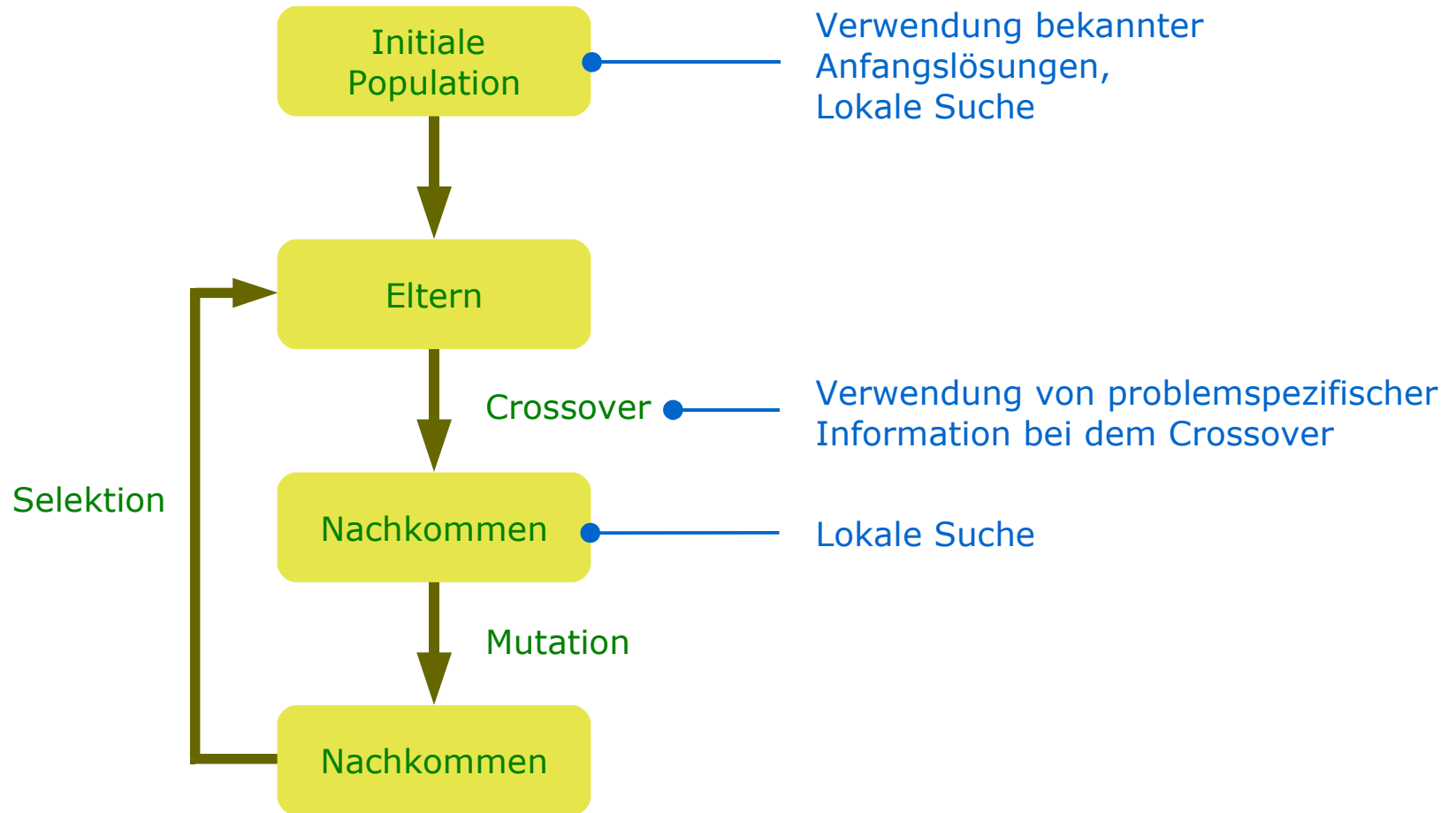


Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:

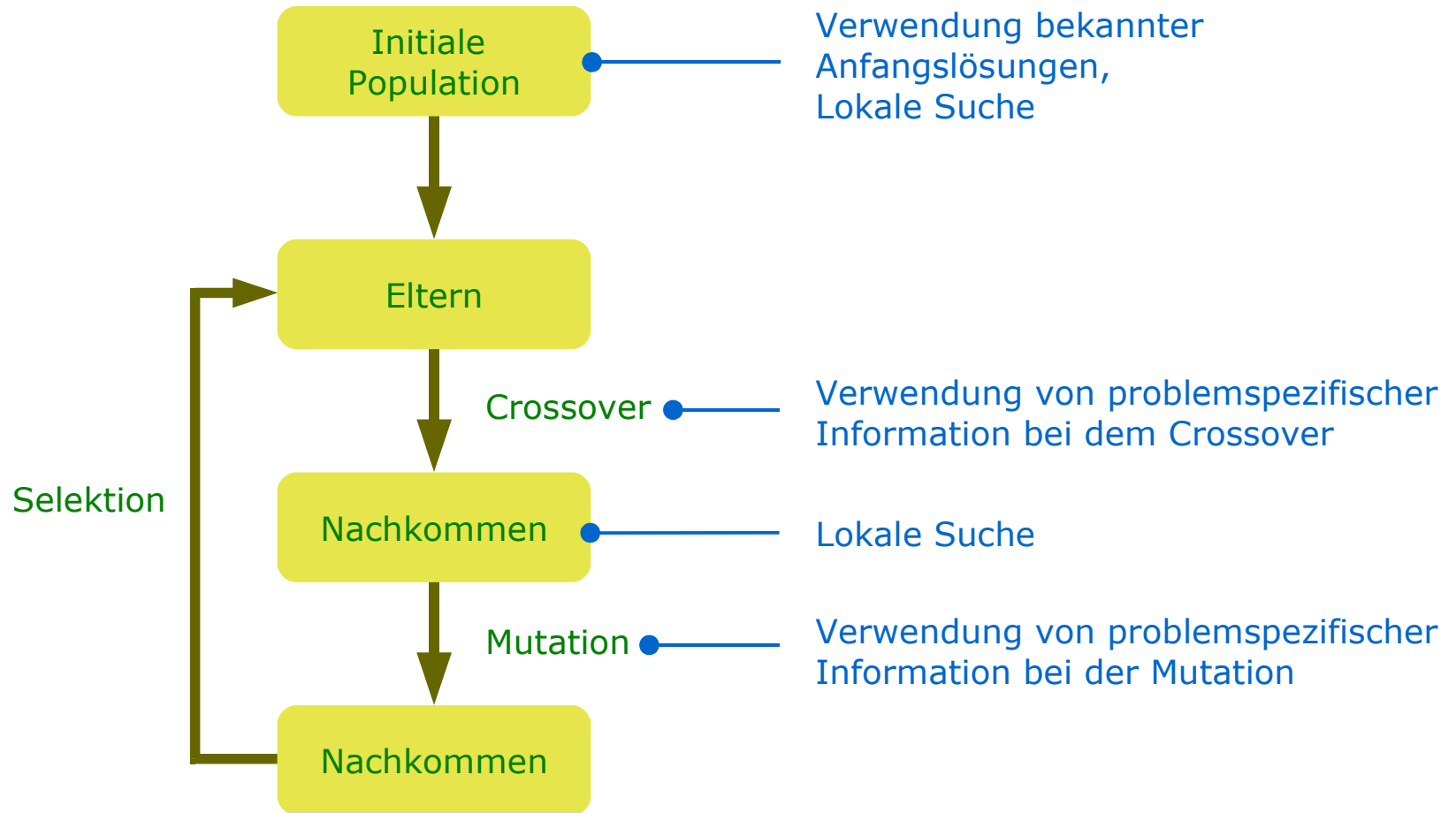


Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:

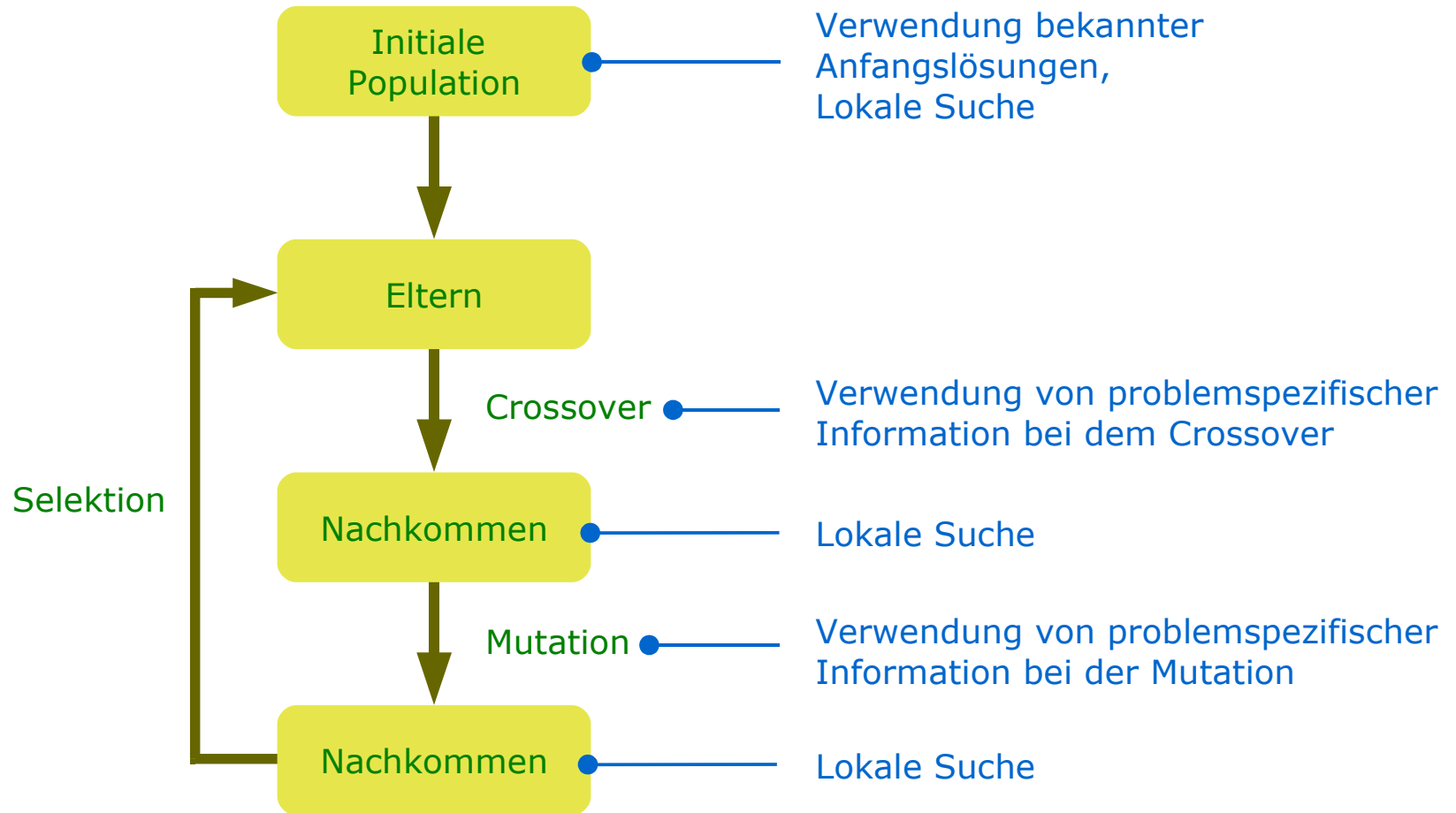


Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:

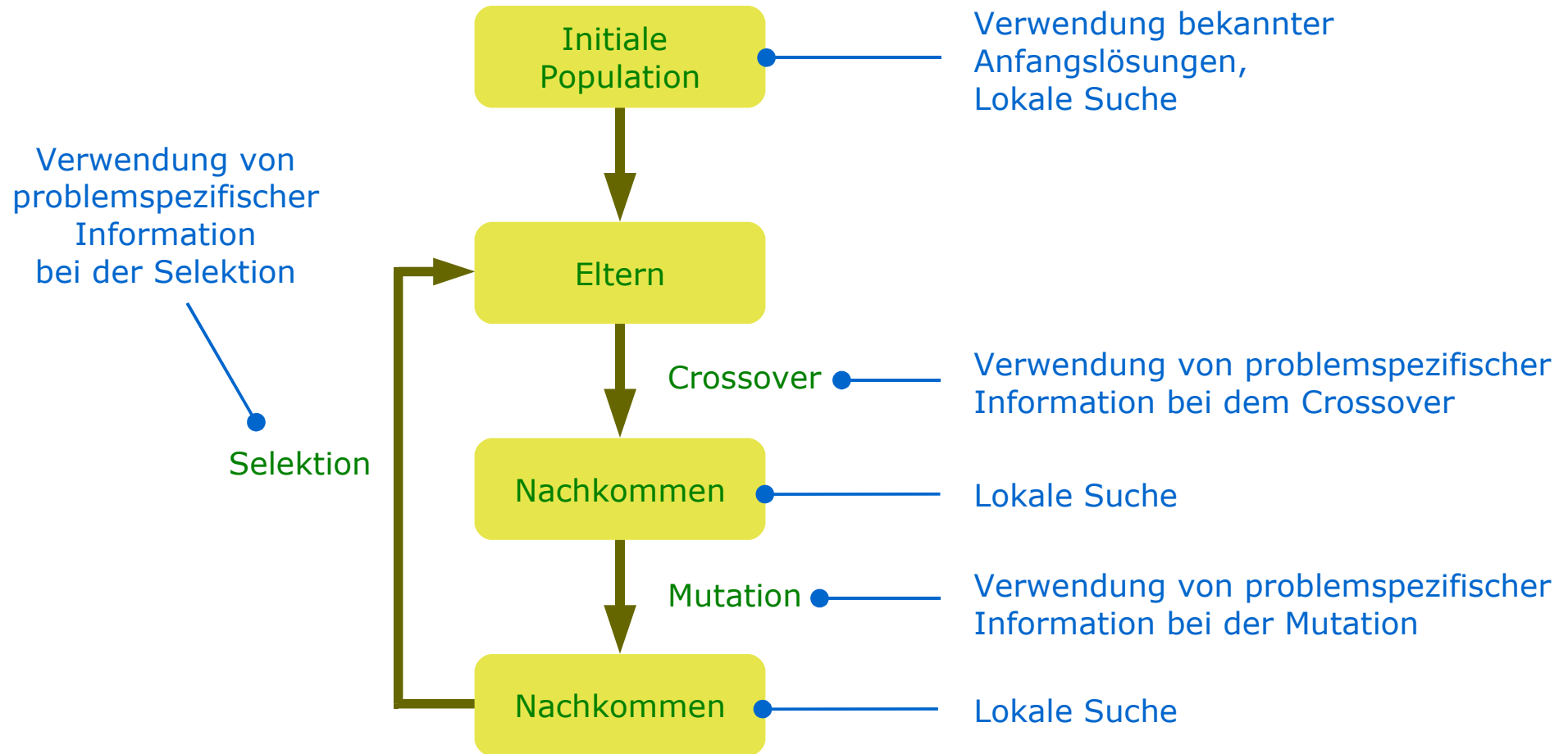


Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Genetische vs. memetische Algorithmen

Es gibt mehrere Stellen, wo man in einen genetischen Algorithmus mit der Fähigkeit zum Lernen erweitern kann:



Vgl. mit A. Eiben: *Introduction to Evolutionary Computing*. Springer-Verlag, 2003, S. 179

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;  
for (maximale Anzahl der lokalen Suchschritte) {  
}
```



# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;  
for (maximale Anzahl der lokalen Suchschritte) {  
    N = Nachbarn(bestesAktuellesElement);  
}
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
for (maximale Anzahl der lokalen Suchschritte) {
    N = Nachbarn(bestesAktuellesElement);
    foreach (n ∈ N) {
    }
}
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
for (maximale Anzahl der lokalen Suchschritte) {
    N = Nachbarn(bestesAktuellesElement);
    foreach (n ∈ N) {
        if (fitness(n) > fitness(bestesAktuellesElement) {
            }
        }
    }
}
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
for (maximale Anzahl der lokalen Suchschritte) {
    N = Nachbarn(bestesAktuellesElement);
    foreach (n ∈ N) {
        if (fitness(n) > fitness(bestesAktuellesElement) {
            bestesAktuellesElement = n;
        }
    }
}
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
for (maximale Anzahl der lokalen Suchschritte) {
    N = Nachbarn(bestesAktuellesElement);
    foreach (n ∈ N) {
        if (fitness(n) > fitness(bestesAktuellesElement) {
            bestesAktuellesElement = n;
        }
    }
}
return bestesAktuellesElement;
```

# Memetische Algorithmen

## Lokale Suche

Bei der lokalen Suche in einem Raum wird davon ausgegangen, dass jedes Element Nachbarschaftsbeziehungen zu anderen Elementen besitzt. Diese Nachbarelemente werden bis zu einer gewissen Suchtiefe nach Elementen abgesucht, die eine bessere Fitness als das Ausgangselement besitzen.

Beispiel

```
bestesAktuellesElement = StartElement;
for (maximale Anzahl der lokalen Suchschritte) {
    N = Nachbarn(bestesAktuellesElement);
    foreach (n ∈ N) {
        if (fitness(n) > fitness(bestesAktuellesElement) {
            bestesAktuellesElement = n;
        }
    }
}
return bestesAktuellesElement;
```



Was genau kann man mit einer lokalen Suche erreichen ? Was nicht ?

# Memetische Algorithmen

## Wie wirkt sich die lokale Suche aus ?

Bei memetischen Algorithmen gibt es zwei Möglichkeiten, wie sich das Ergebnis einer lokalen Suche auswirken kann.

# Memetische Algorithmen

## Wie wirkt sich die lokale Suche aus ?

Bei memetischen Algorithmen gibt es zwei Möglichkeiten, wie sich das Ergebnis einer lokalen Suche auswirken kann.

Bei dem **lamarckischen memetischen Algorithmus** werden Verbesserungen, die durch die lokale Suche gefunden wurden, an die Nachkommen weitergegeben.



# Memetische Algorithmen

## Wie wirkt sich die lokale Suche aus ?

Bei memetischen Algorithmen gibt es zwei Möglichkeiten, wie sich das Ergebnis einer lokalen Suche auswirken kann.

Bei dem **lamarckischen memetischen Algorithmus** werden Verbesserungen, die durch die lokale Suche gefunden wurden, an die Nachkommen weitergegeben.

Hierzu werden die gefundenen Verbesserungen in den Genotyp des Kandidaten hineinkodiert. Dies ist nur möglich, wenn die Abbildung vom Phänotyp zum Genotyp mit vertretbaren Kosten berechnet werden kann.

# Memetische Algorithmen

## Wie wirkt sich die lokale Suche aus ?

Bei memetischen Algorithmen gibt es zwei Möglichkeiten, wie sich das Ergebnis einer lokalen Suche auswirken kann.

Bei dem **lamarckischen memetischen Algorithmus** werden Verbesserungen, die durch die lokale Suche gefunden wurden, an die Nachkommen weitergegeben.

Hierzu werden die gefundenen Verbesserungen in den Genotyp des Kandidaten hineinkodiert. Dies ist nur möglich, wenn die Abbildung vom Phänotyp zum Genotyp mit vertretbaren Kosten berechnet werden kann.

Dies steht im **Gegensatz** zu dem zentralen Dogma der Biologie, aber im Rahmen von memetischen Algorithmen sind wir nicht daran gebunden.

# Memetische Algorithmen

## Wie wirkt sich die lokale Suche aus ?

Bei memetischen Algorithmen gibt es zwei Möglichkeiten, wie sich das Ergebnis einer lokalen Suche auswirken kann.

Bei dem **lamarckischen memetischen Algorithmus** werden Verbesserungen, die durch die lokale Suche gefunden wurden, an die Nachkommen weitergegeben.

Hierzu werden die gefundenen Verbesserungen in den Genotyp des Kandidaten hineinkodiert. Dies ist nur möglich, wenn die Abbildung vom Phänotyp zum Genotyp mit vertretbaren Kosten berechnet werden kann.

Dies steht im **Gegensatz** zu dem zentralen Dogma der Biologie, aber im Rahmen von memetischen Algorithmen sind wir nicht daran gebunden.

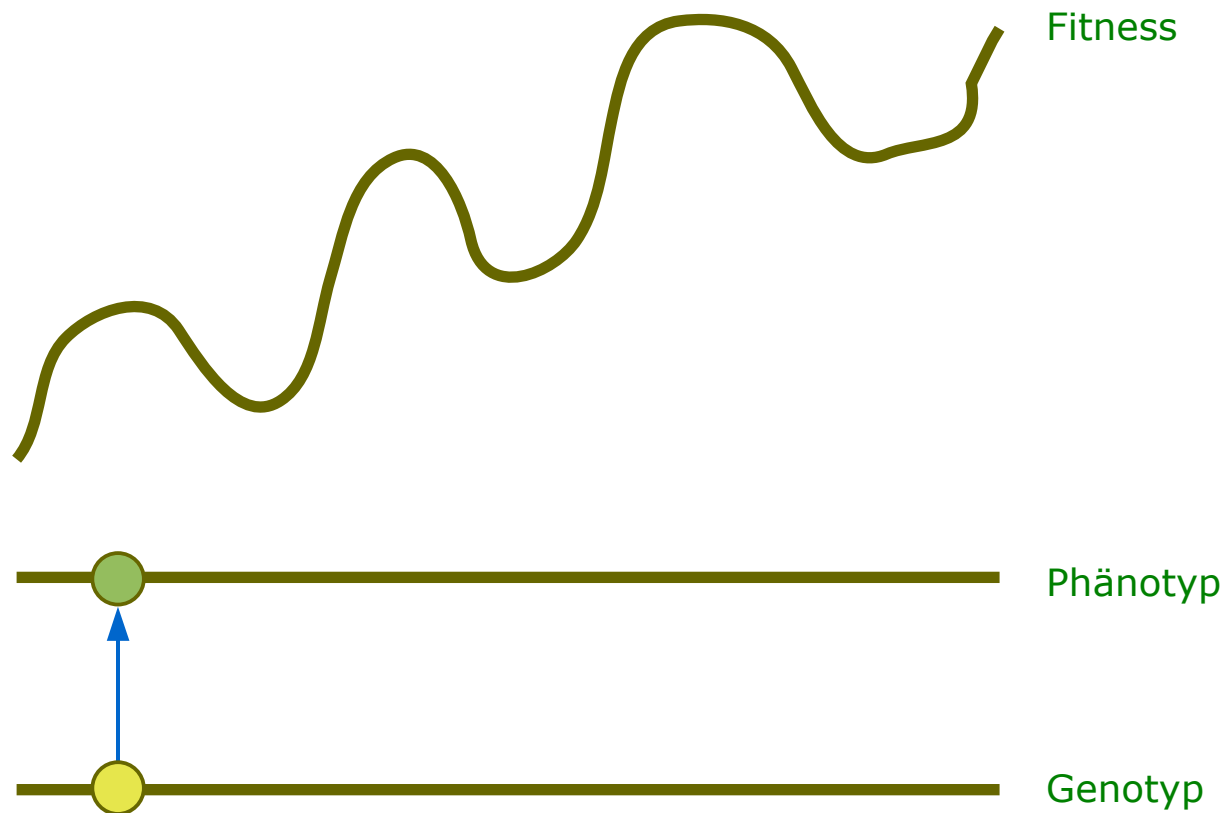
Aus dem **zentralen Dogma der Biologie** folgt, dass erworbene Eigenschaften nicht in die DNS zurückgeschrieben werden können.



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

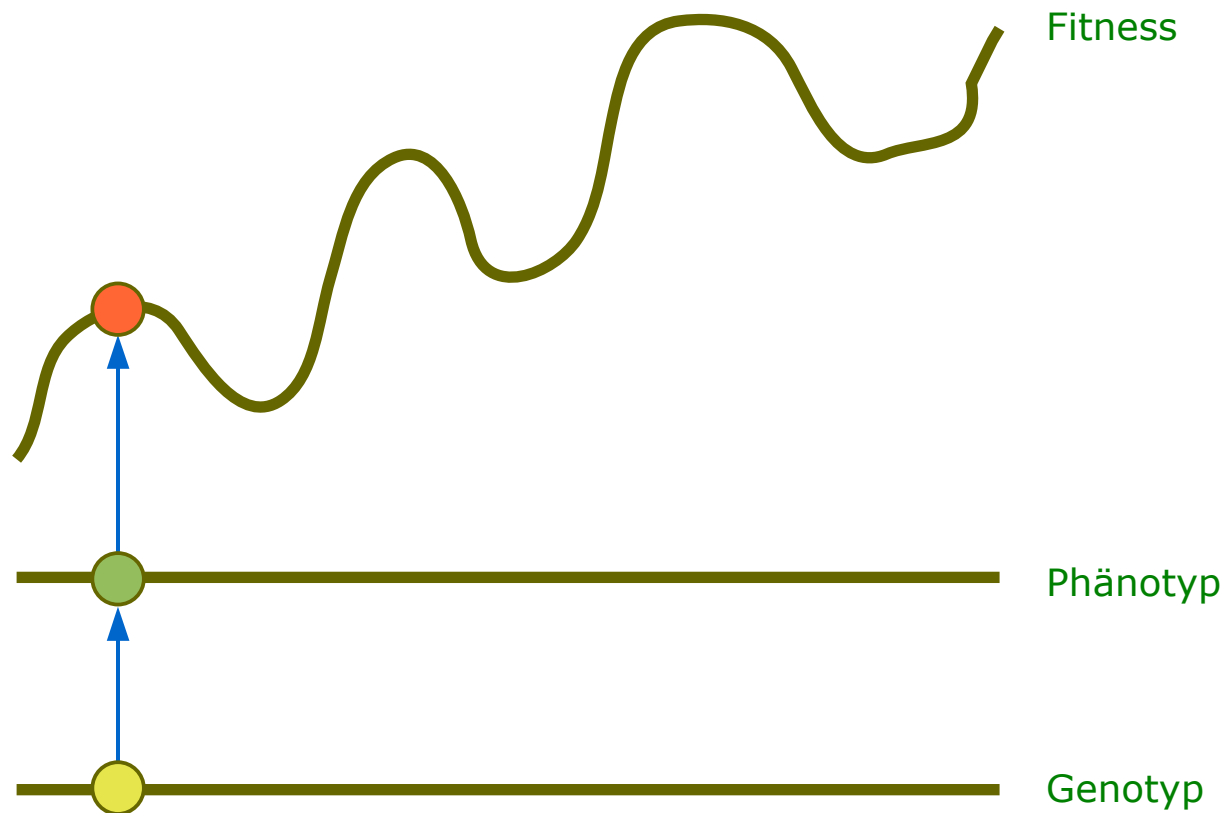
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

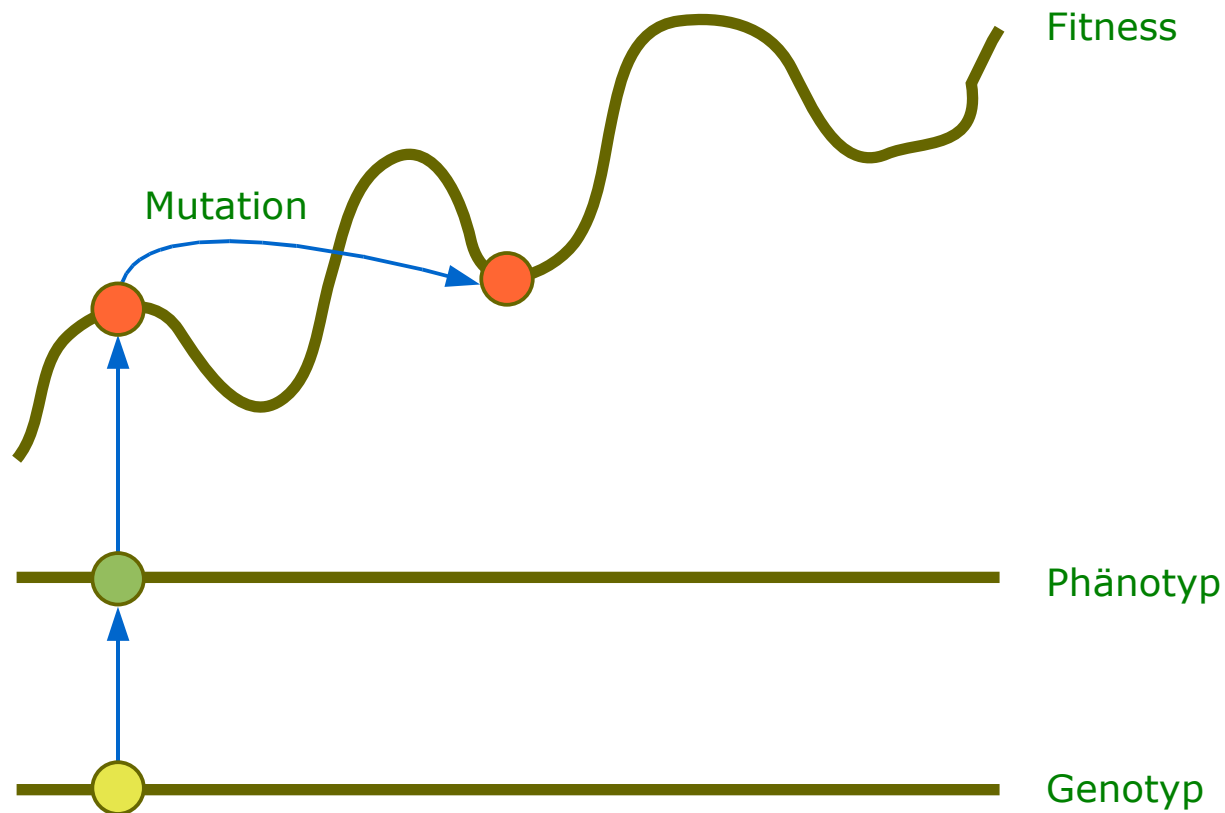
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

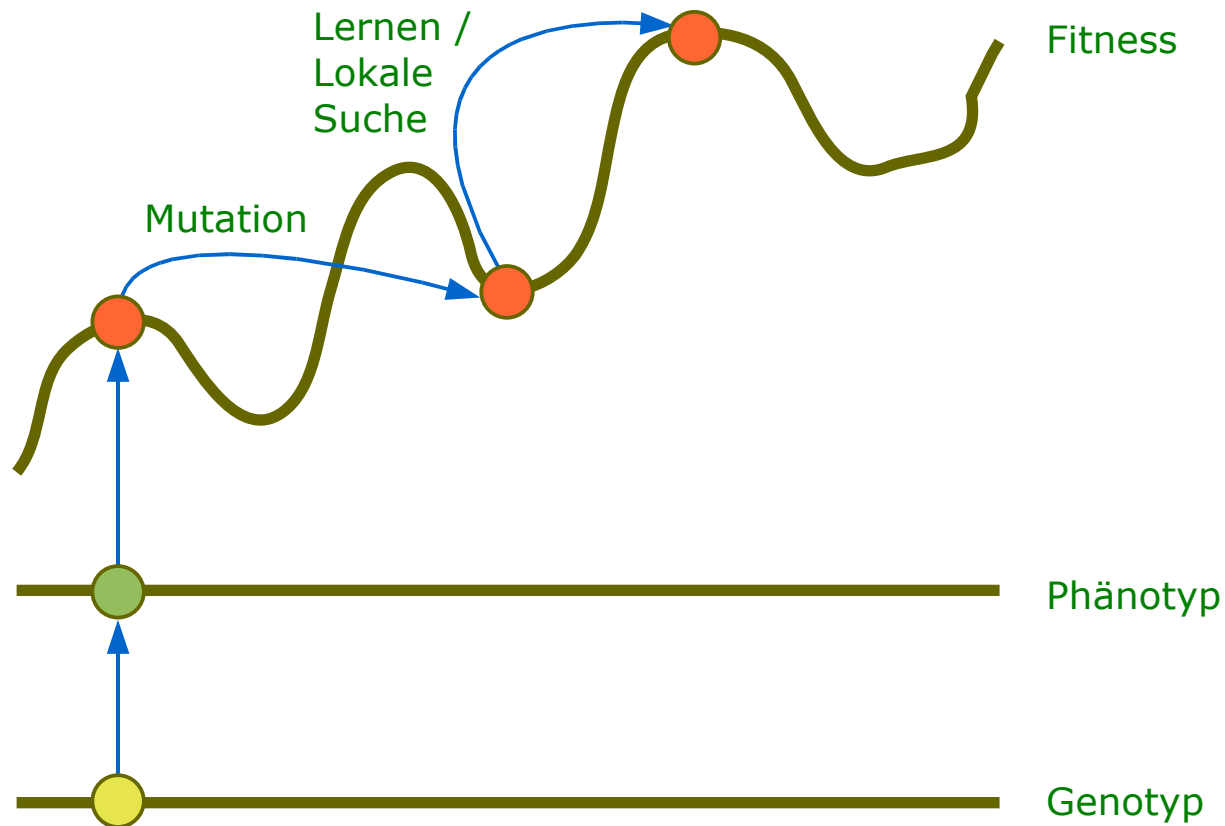
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

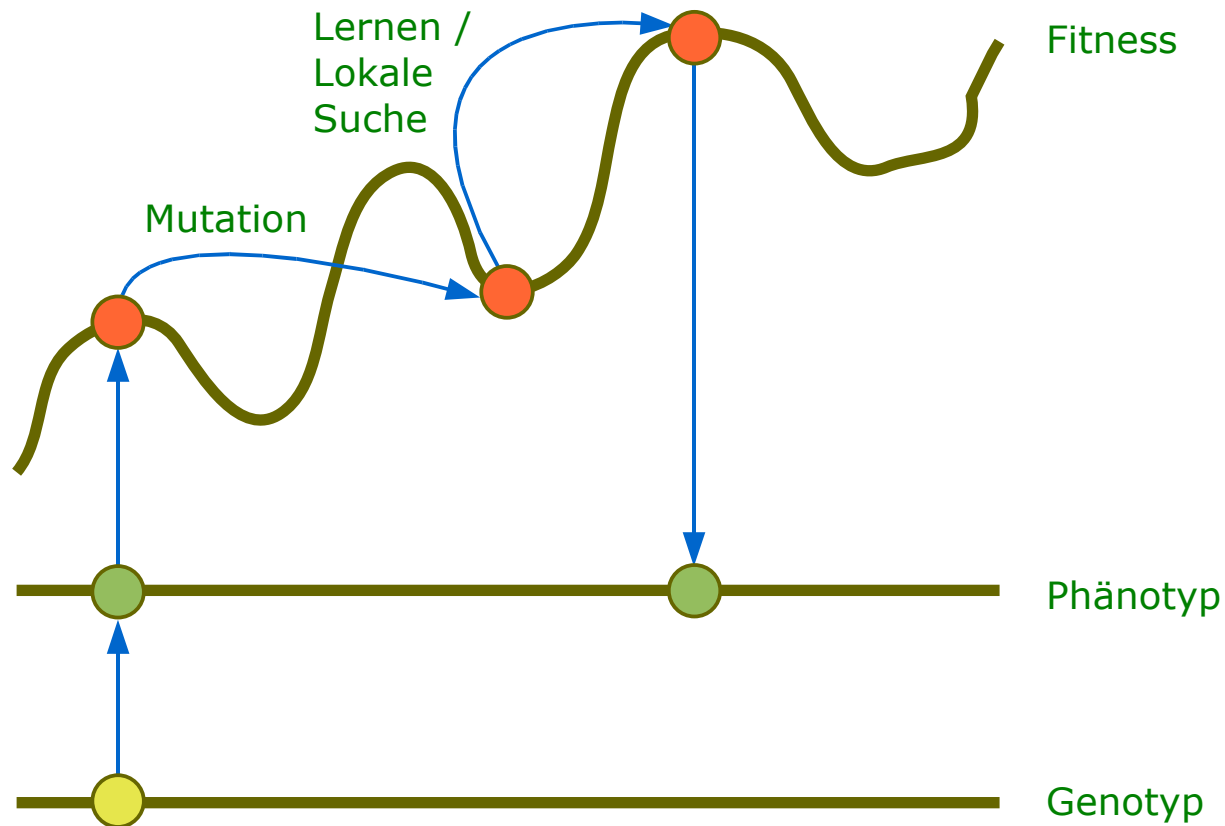
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:

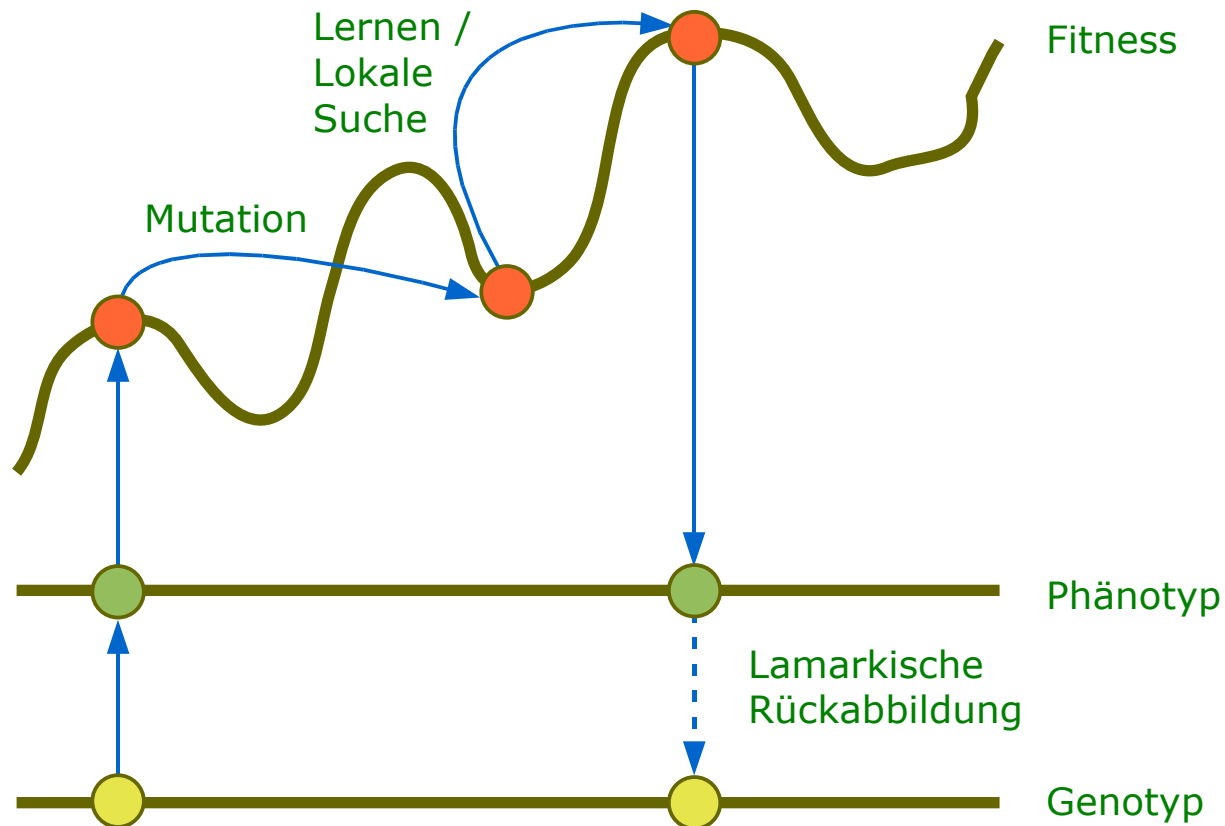




# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

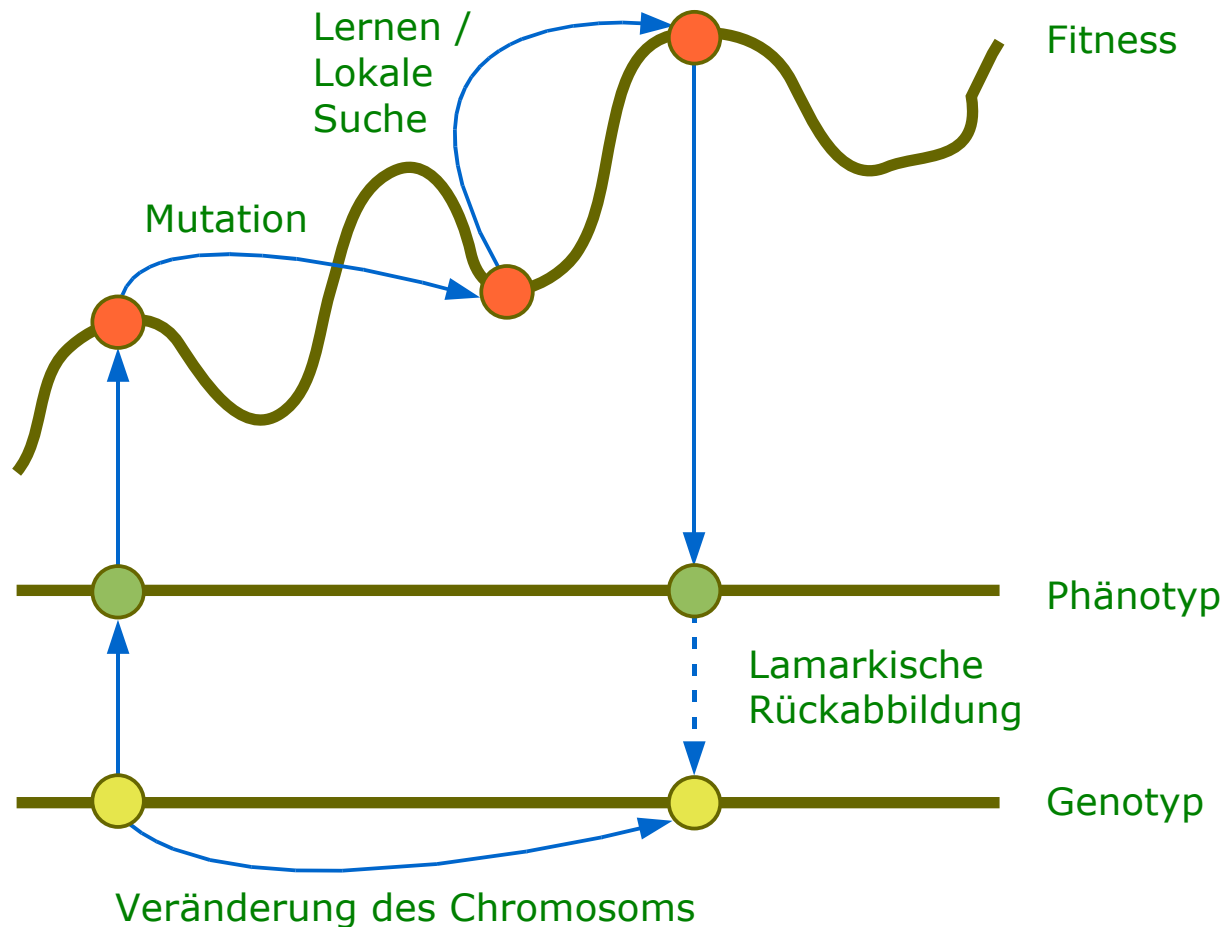
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Lamarckischer memetischer Algorithmus (Lamarckian memetic algorithm)

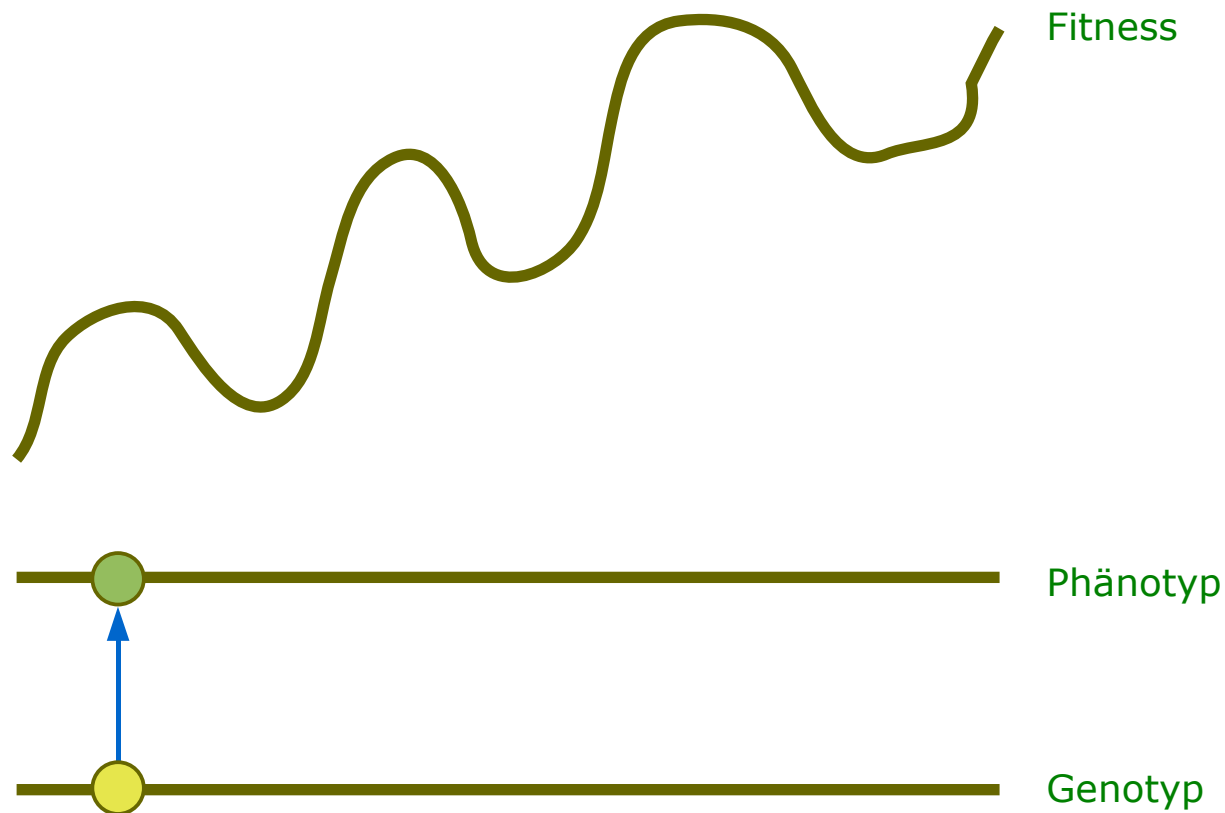
Bei einem lamarckischen memetischen Algorithmus wird das Erbgut des Kandidaten gemäß den Verbesserungen verändert:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

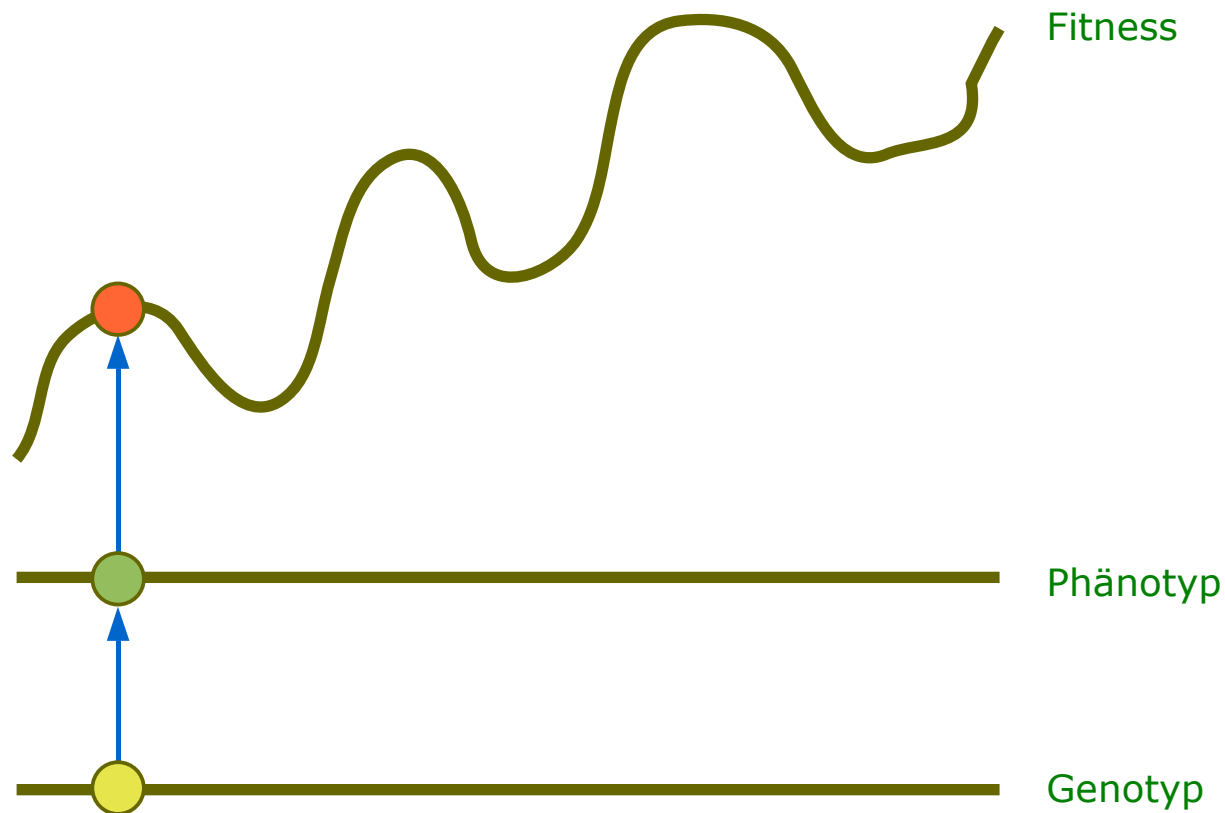
Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

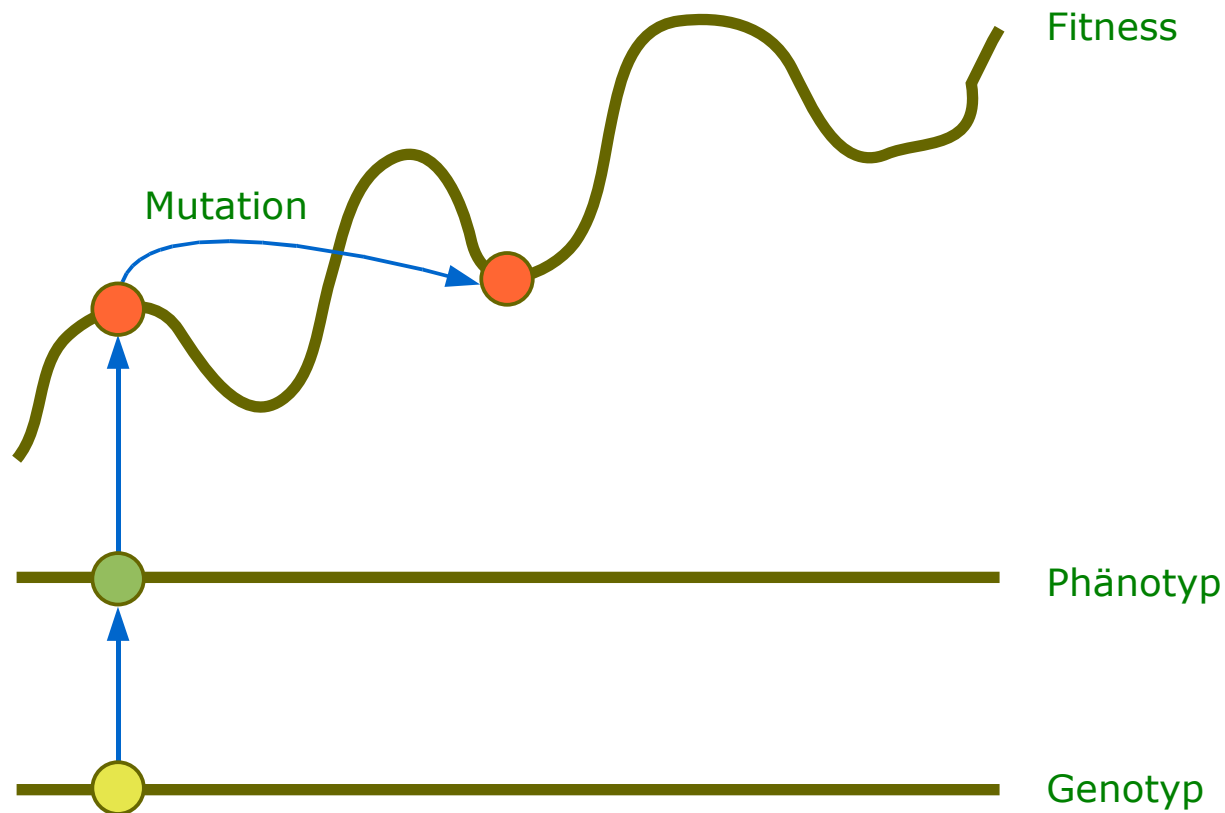
Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

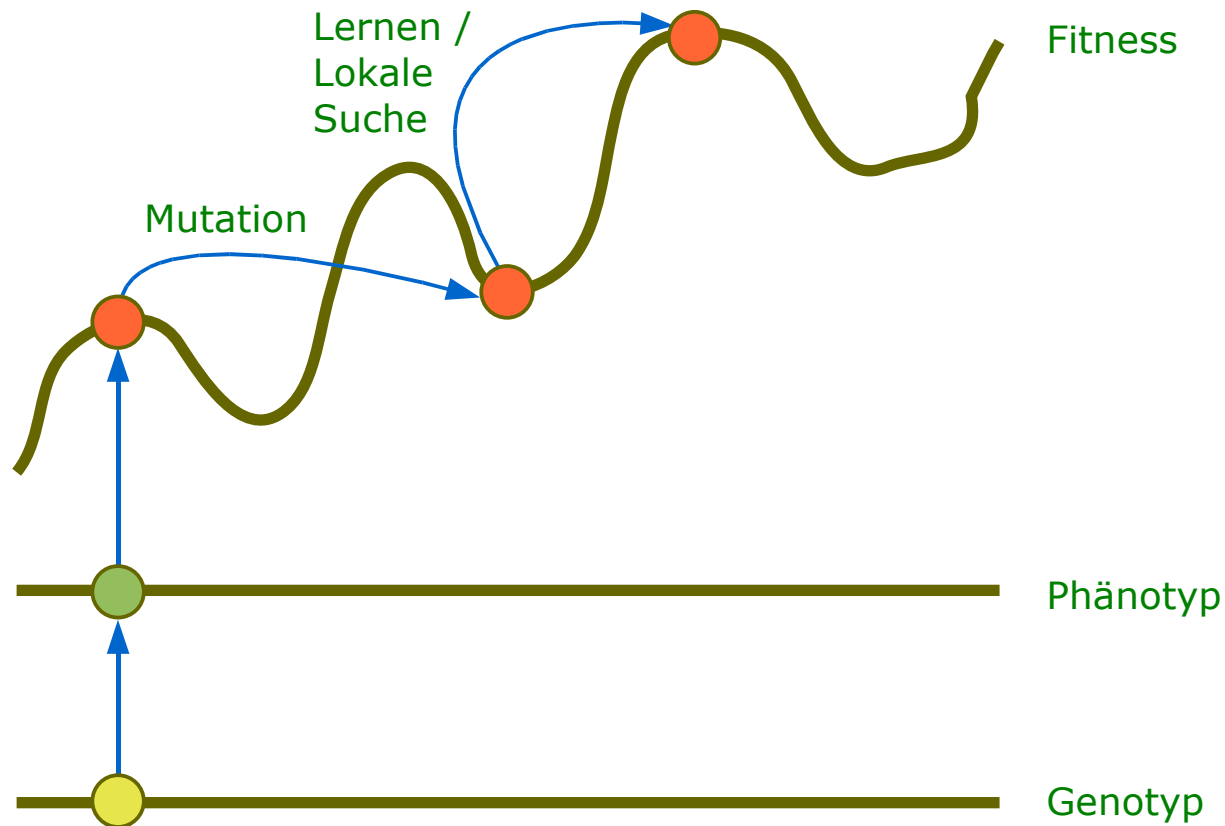
Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

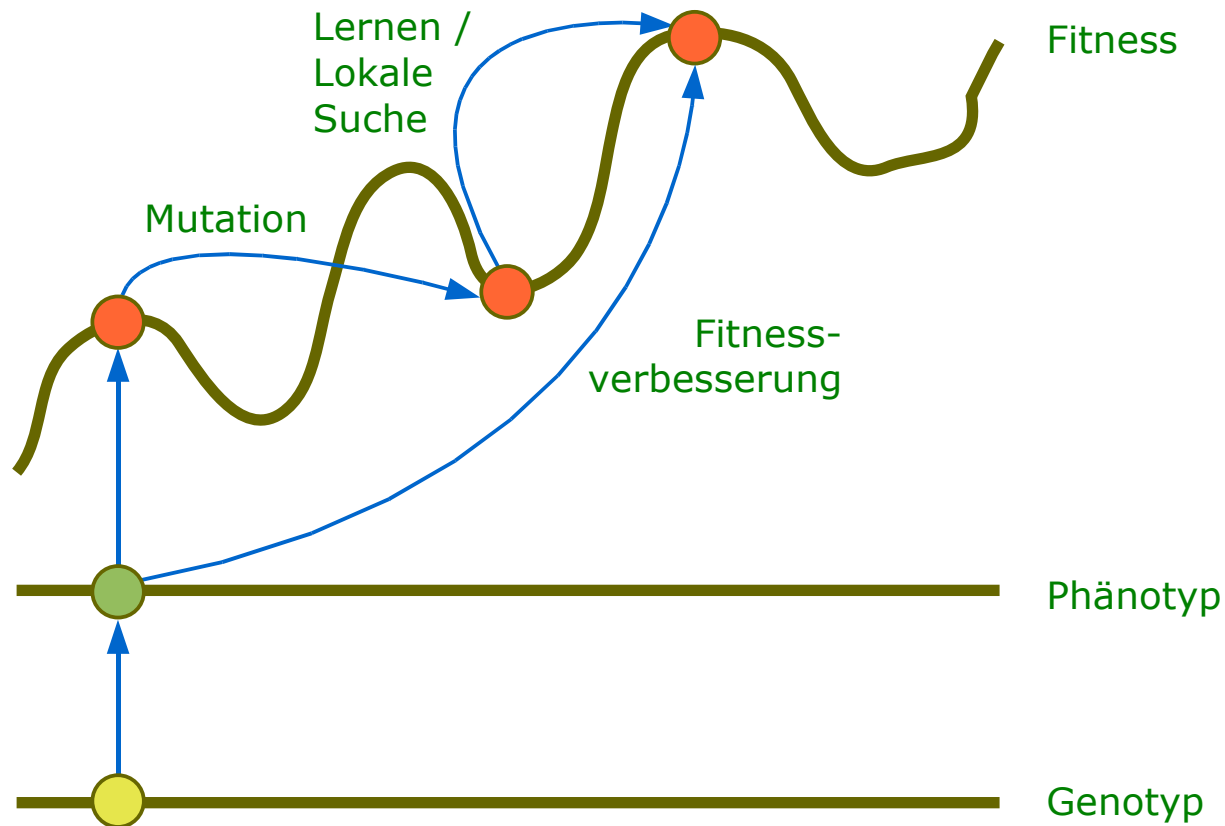
Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

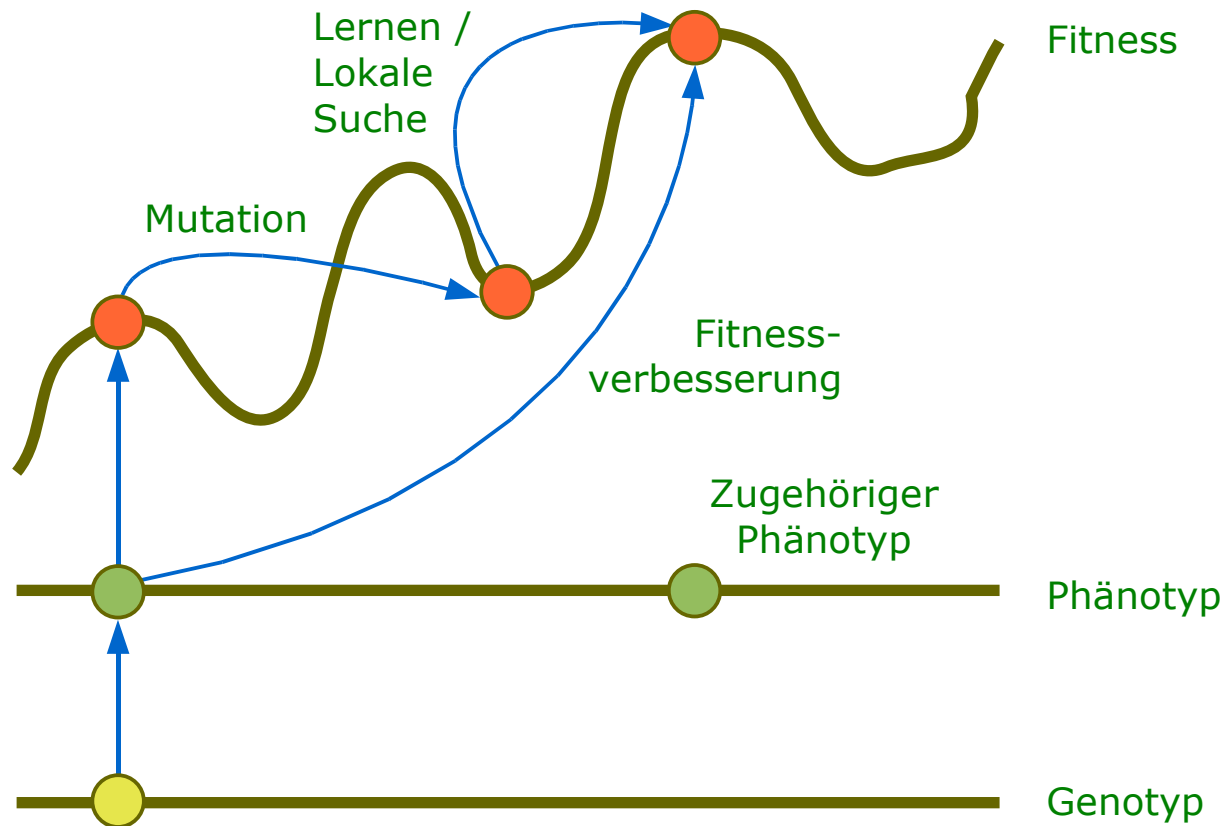
Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:



# Memetische Algorithmen

## Baldwinischer memetischer Algorithmus

Bei dem baldwinischen memetischen Algorithmus können die erlernten Verbesserungen nicht den Nachkommen weitergegeben werden, sondern erhöhen lediglich die Fitness des Kandidaten:





# Memetische Algorithmen

## Hörsaalübung

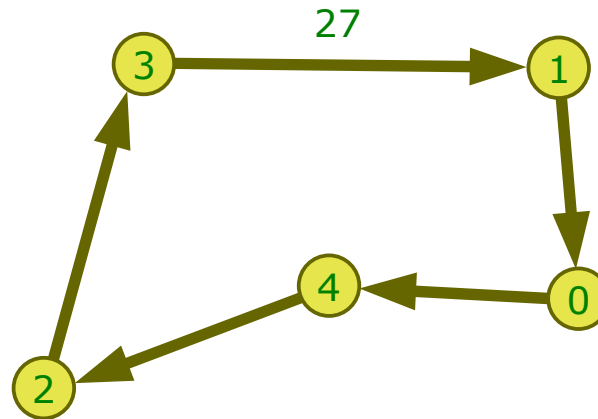
Wandeln Sie einen einfachen genetischen Algorithmus für das Problem des Handlungsreisenden in einen memetischen Algorithmus um.

Überlegen Sie sich hierzu, was der Begriff *lokale Suche* in diesem Zusammenhang bedeutet und geben Sie ein Beispiel an.

Nehmen Sie an, dass der Phänotyp ein Integerfeld  $s$  mit einer Permutation von  $n$  Städten ist, und eine Methode  $d(i,j)$  die Distanz der beiden Städte  $i$  und  $j$  zurückliefert.

Beispiel:

$s[0] = 1$   
 $s[1] = 0$   
 $s[2] = 4$   
 $s[3] = 2$   
 $s[4] = 3$   
  
 $d(3,1) = 27$   
usw.



# Memetische Algorithmen

## Hörsaalübung

Nach der Mutation und vor der Berechnung der Fitness soll folgende lokale Suche stattfinden:

```
for i=0 to n-4
  if (d(s[i],s[i+1]) + d(s[i+1],s[i+2]) + d(s[i+2],s[i+3]) >
      d(s[i],s[i+2]) + d(s[i+2],s[i+1]) + d(s[i+1],s[i+3])) {
    vertausche(s[i+1], s[i+2]);
  }
}
```

