

Evolutionäre Algorithmen

Variation und genetische Operatoren

Prof. Dr. Rudolf Kruse **Christian Moewes**

{kruse,cmoewes}@iws.cs.uni-magdeburg.de

Otto-von-Guericke-Universität Magdeburg

Fakultät für Informatik

Institut für Wissens- und Sprachverarbeitung

Übersicht

1. Motivation

2. Ein-Elter-Operatoren

3. Zwei- oder Mehr-Elter-Operatoren

4. Interpolierende und extrapolierende Rekombination

5. Selbstanpassende Algorithmen

6. Zusammenfassung

Variation durch Mutation [Weicker, 2007]

- Variationen (Mutationen): kleine Veränderungen in der Biologie
- ⇒ Mutationsoperator: ändert möglichst wenig am Lösungskandidaten bzgl. Fitnessfunktion
- im Folgenden: Untersuchung im Zusammenspiel mit Selektion
 - hier: Verhalten eines einfachen Optimierungsalgorithmus auf sehr einfachem Optimierungsproblem (Abgleich mit einem vorgegebenen Bitmuster)

Bedeutung der Mutation

Exploration oder Erforschung

- stichprobenartiges Erkunden
- auch: weiter entfernte Regionen des Suchraums

Exploitation oder Feinabstimmung

- lokale Verbesserung eines Lösungskandidaten
- wichtig: Einbettung der phänotypischen Nachbarschaft

Binäre Mutation

Algorithm 1 Binäre Mutation

Input: Individuum A mit $A.G \in \{0, 1\}^l$

Output: Individuum B

$B \leftarrow A$

for $i \in \{1, \dots, l\}$ **{**

$u \leftarrow$ wähle zufällig gemäß $U([0, 1])$

if $u \leq p_m$ **{** /* Mutationswahrscheinlichkeit p_m */

$B.G_i \leftarrow 1 - A.G_i$

}

}

return B

Gauß-Mutation

alternative reelwertige Mutation

- direkt auf den reellwertigen Zahlen
- Addition einer normalverteilten Zufallszahl auf jede Komponente

Algorithm 2 Gauß-Mutation

Input: Individuum A mit $A.G \in \mathbb{R}^l$

Output: Individuum B

for $i \in \{1, \dots, l\}$ {

$u_i \leftarrow$ wähle zufällig gemäß $N(0, \sigma)$ /* Standardabweichung σ */

$B_i \leftarrow A_i + u_i$

$B_i \leftarrow \max\{B_i, ug_i\}$ /* untere Wertebereichsgrenze ug_i */

$B_i \leftarrow \min\{B_i, og_i\}$ /* obere Wertebereichsgrenze og_i */

}

return B

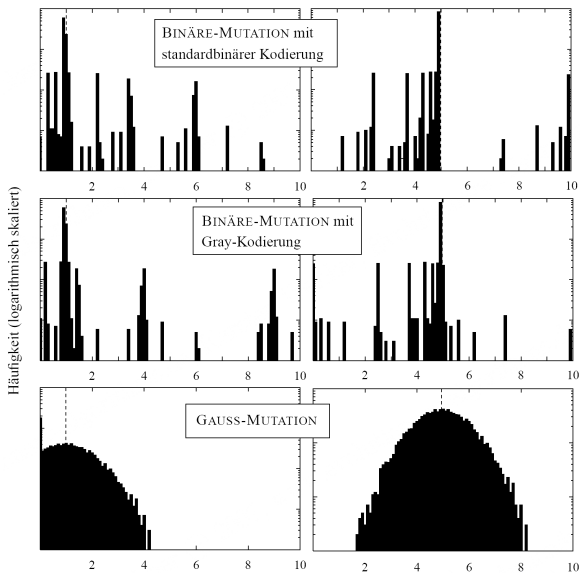
Vergleich der Mutationsverfahren

Ansatz

- Optimierung der einfachen Funktion

$$f_2(x) = \begin{cases} x & \text{falls } x \in [0, 10] \subset \mathbb{R}, \\ \text{undef.} & \text{sonst} \end{cases}$$

- zwei Elternindividuen (1.0 und 4.99)
- Ermittlung der Nachkommensverteilung mit jeweils 10000 Mutationen



Vergleich der Mutationsverfahren

- Gauß-Mutation mit kleinem σ sehr gut für Exploitation
 - mit großem σ sehr breite Erforschung

 - binäre Mutation eines GA hat mehrerer verteilte Schwerpunkte
 - Hamming-Klippen = Brüche in Häufigkeitsverteilung

 - Gray-Kodierung schafft es, phänotypische Nachbarn einzubinden
 - tendiert dennoch zu einer Seite des Suchraums
- ⇒ Gauß-Mutation orientiert sich an phänotypischer Nachbarschaft
- ⇒ binäre Mutation detektiert schneller interessante Regionen in Ω

Genetische Operatoren

- werden auf best. Teil ausgewählter Individuen (Zwischenpopulation) angewandt
- Erzeugung von Varianten und Rekombinationen bestehender Lösungskandidaten
- all. Einteilung genetischer Operatoren nach Zahl der Eltern:
 - Ein-Elter-Operatoren („Mutation“)
 - Zwei-Elter-Operatoren („Crossover“)
 - Mehr-Elter-Operatoren
- genetischen Operatoren haben best. Eigenschaften (abh. v. Kodierung)
 - falls Lösungskandidaten = Permutationen, so permutationserhaltende genetischen Operatoren
 - allg.: falls best. Allelkombinationen unsinnig, sollten genetischen Operatoren sie möglichst nicht erzeugen

Übersicht

1. Motivation

2. Ein-Elter-Operatoren

Standardmutation und Zweiertausch
Operationen auf Teilstücke

3. Zwei- oder Mehr-Elter-Operatoren

4. Interpolierende und extrapolierende Rekombination

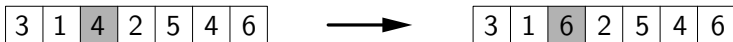
5. Selbstanpassende Algorithmen

6. Zusammenfassung

Standardmutation und Zweiertausch

- **Standardmutation:**

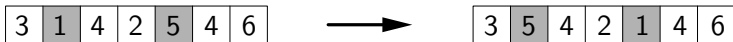
Austausch der Ausprägung eines Gens durch anderes Allel



- ggf. werden mehrere Gene mutiert (vgl. n -Damen-Problem)
- *Parameter:* Mutationswahrscheinlichkeit p_m , $0 < p_m \ll 1$
für Bitstrings der Länge l ist $p_m = 1/l$ annähernd optimal

- **Zweiertausch:**

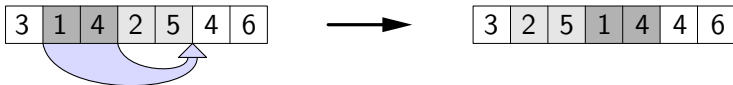
Austausch der Ausprägungen zweier Gene eines Chromosoms



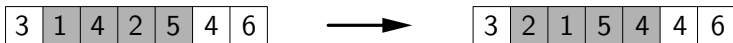
- *Voraussetzung:* gleiche Allelmengen der ausgetauschten Gene
- *Verallgemeinerung:* zyklischer Tausch von 3, 4, ..., k Genen

Operationen auf Teilstücke

- **Verschieben eines Teilstücks:**



- **Mischen/Permutieren eines Teilstücks:**



- **Umdrehen/Invertieren eines Teilstücks:**



- *Voraussetzung:* gleiche Allelmengen im betroffenen Bereich
- *Parameter:* ggf. W'keitsverteilung über Längen (und Verschiebungsweiten für Verschieben eines Teilstücks)

Übersicht

1. Motivation

2. Ein-Elter-Operatoren

3. Zwei- oder Mehr-Elter-Operatoren

Ein-Punkt- und Zwei-Punkt-Crossover

n-Punkt- und uniformes Crossover

Shuffle Crossover

Permutationserhaltende Crossover

Diagonal-Crossover

Charakterisierung

4. Interpolierende und extrapolierende Rekombination

Ein-Punkt- und Zwei-Punkt-Crossover

Ein-Punkt-Crossover

- Bestimmen eines zufälligen Schnittpunktes
- Austausch der Gensequenzen auf einer Seite des Schnittpunktes



Zwei-Punkt-Crossover

- Bestimmen zweier zufälliger Schnittpunkte
- Austausch der Gensequenzen zwischen den beiden Schnittpunkten



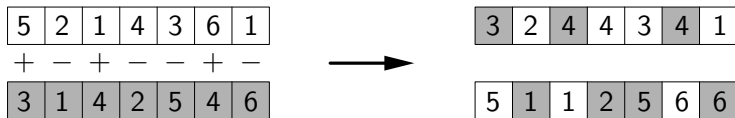
n-Punkt- und uniformes Crossover

n-Punkt-Crossover

- Verallgemeinerung des Ein- und Zwei-Punkt-Crossover
- Bestimmen von n zufälligen Schnittpunkten
- Abwechselndes Austauschen / Nicht-Austauschen der Gensequenzen zwischen zwei aufeinanderfolgenden Schnittpunkten

Uniformes Crossover

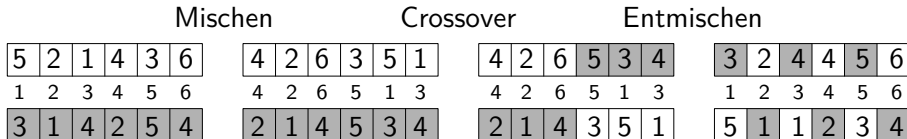
- für jedes Gen: bestimme ob es getauscht wird oder nicht (+: ja, -: nein, *Parameter*: W'keit p_x für Austausch)



- **Beachte:** uniformes Crossover entspricht **nicht** dem $(l - 1)$ -Punkt-Crossover! Zahl der Crossoverpunkte ist zufällig

Shuffle Crossover

- vor Ein-Punkt-Crossover: zufälliges Mischen der Gene
- danach: Entmischen der Gene



- Shuffle Crossover ist **nicht** äquivalent zum uniformen Crossover!
- jede Anzahl von Vertauschungen von Genen zwischen Chromosomen ist gleichwahrscheinlich
- uniformen Crossover: Anzahl ist binomialverteilt mit Parameter p_x
- Shuffle Crossover: eines der empfehlenswertesten Verfahren

Uniformes ordnungsbasiertes Crossover

- ähnlich wie uniformes Crossover: entscheide für jedes Gen, ob es erhalten bleibt oder nicht
(+: ja, -: nein, *Parameter*: W'keit p_k für Erhalt)
- fülle Lücken durch fehlende Allele auf (in Reihenfolge der Vorkommen im anderen Chromosom)



- erhält **Reihenfolgeinformation**
- *alternativ*: Erhalten der „+“ bzw. „-“ markierten Gene im einen bzw. anderen Chromosom

Kantenrekombination (speziell für TSP)

- Chromosom wird als Graph (Kette oder Ring) aufgefasst jedes Gen besitzt Kanten zu seinen Nachbarn im Chromosom
- Kanten der Graphen zweier Chromosomen werden gemischt, daher Name
- erhält **Nachbarschaftsinformation**

Vorgehen: 1. *Aufbau einer Kantentabelle*

- liste zu jedem Allel seine Nachbarn (in beiden Eltern) (ggf. erstes und letztes Gen des Chromosoms benachbart)
- falls ein Allel in beiden Eltern gleichen Nachbarn (Seite irrelevant), dann liste diesen Nachbar nur 1x auf (aber markiert)

Kantenrekombination

Vorgehen: 2. Aufbau eines Nachkommen

- wähle erstes Allel zufällig aus einem der beiden Eltern
- lösche ausgewähltes Allel aus Kantentabelle (aus Listen der Nachbarn der Allele)
- wähle jeweils nächstes Allel aus den noch nicht gelöschten Nachbarn des vorangehenden mit folgender Priorität:
 1. markierte (d.h. doppelt auftretende) Nachbarn
 2. Nachbarn mit kürzester Nachbarschaftsliste (wobei markierte Nachbarn einfach zählen)
 3. zufällige Auswahl eines Nachbarn

Erzeugung des zweiten Nachkommen analog aus erstem Allel des anderen Elter (meist jedoch nicht gemacht)

Kantenrekombination

Beispiel:

A:

6	3	1	5	2	7	4
---	---	---	---	---	---	---

B:

3	7	2	5	6	1	4
---	---	---	---	---	---	---

Aufbau der Kantentabelle

Allel	Nachbarn		zusammengefasst
	in A	in B	
1	3, 5	6, 4	3, 4, 5, 6
2	5, 7	7, 5	5*, 7*
3	6, 1	4, 7	1, 4, 6, 7
4	7, 6	1, 3	1, 3, 6, 7
5	1, 2	2, 6	1, 2*, 6
6	4, 3	5, 1	1, 3, 4, 5
7	2, 4	3, 2	2*, 3, 4

- beide Chromosomen = Ring (erstes und letztes Gen benachbart): in **A** ist 4 linker Nachbar der 6, 6 ist rechter Nachbar der 4; **B** analog
- in beiden: 5, 2 und 7 stehen nebeneinander – sollte erhalten werden (siehe Markierungen)

Kantenrekombination

Aufbau eines Nachkommen

6	5	2	7	4	3	1
---	---	---	---	---	---	---

Allel	Nachbarn	Wahl: 6	5	2	7	4	3	1
1	3, 4, 5, 6	3, 4, 5	3, 4	3, 4	3, 4	3		
2	5*, 7*	5*, 7*	7*	7*	—	—	—	—
3	1, 4, 6, 7	1, 4, 7	1, 4, 7	1, 4, 7	1, 4	1	1	—
4	1, 3, 6, 7	1, 3, 7	1, 3, 7	1, 3, 7	1, 3	1, 3	—	—
5	1, 2*, 6	1, 2*	1, 2*	—	—	—	—	—
6	1, 3, 4, 5	1, 3, 4, 5	—	—	—	—	—	—
7	2*, 3, 4	2*, 3, 4	2*, 3, 4	3, 4	3, 4	—	—	—

- starte mit erstem Allel des Chromosoms **A** (also 6) und streiche 6 aus allen Nachbarschaftslisten (dritte Spalte)
- da unter Nachbarn der 6 (1, 3, 4, 5) die 5 kürzeste Liste hat, wird 5 für zweites Gen gewählt
- dann folgt die 2, die 7 usw.

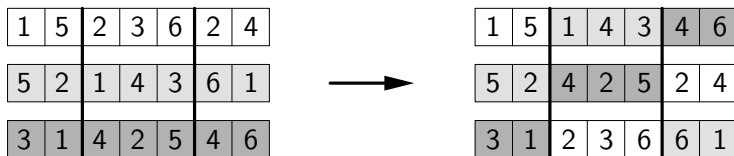
Kantenrekombination

- Nachkomme hat meist neue Kante (vom letzten zum ersten Gen)
- kann auch angewendet werden, wenn erstes und letztes Gen nicht als benachbart gelten: Kanten werden dann nicht in Kantentabelle aufgenommen
- sind erstes und letztes Gen benachbart, dann Startallel beliebig falls nicht, dann ein am Anfang stehendes Allel
- Aufbau eines Nachkommen: es ist möglich, dass Nachbarschaftsliste des gerade ausgewählten Allels leer (Prioritäten sollen W 'keit dafür gering halten; sind aber nicht perfekt)
in diesem Fall: zufällige Auswahl aus den noch übrigen Allelen

Drei- und Mehr-Elter-Operatoren

Diagonal-Crossover

- ähnlich wie 1-, 2- und n -Punkt-Crossover, aber für mehr Eltern
- bei drei Eltern: zwei Crossover-Punkte
- verschiebt Gensequenzen an Schnittstellen über Chromosomen diagonal und zyklische



- Verallgemeinerung auf > 3 Eltern:
wähle für k Eltern $k - 1$ Crossover-Punkte
- führt zu sehr guter Durchforstung des Suchraums,
besonders bei großer Elternzahl (10–15 Eltern)

Charakterisierung von Crossover-Operatoren

Ortsabhängige Verzerrung (engl. positional bias):

- falls W' keit, dass 2 Gene zusammen vererbt werden (im gleichen Chromosom bleiben, zusammen ins andere Chromosom wandern) von ihrer relativen Lage im Chromosom abhängt
 - unerwünscht, weil Anordnung der Gene im Chromosom entscheidenden Einfluss auf Erfolg/Misserfolg des EA haben (bestimmte Anordnungen lassen sich schwerer erreichen)
 - **Beispiel: Ein-Punkt-Crossover**
 - 2 Gene werden voneinander getrennt (gelangen in verschiedene Nachkommen), falls Crossover-Punkt zwischen sie fällt
 - je näher 2 Gene im Chromosom beieinander, desto weniger mögliche Crossover-Punkte gibt es zwischen ihnen
- ⇒ nebeneinanderliegende Gene werden mit höherer W' keit als entferntliegende in gleichen Nachkommen gelangen

Charakterisierung von Crossover-Operatoren

Verteilungsverzerrung (engl. distributional bias):

- falls Wahrscheinlichkeit, dass best. Anzahl von Genen ausgetauscht wird, nicht für alle Anzahlen gleich
- unerwünscht, weil Teillösungen unterschiedl. Größe unterschiedl. gute Chancen haben, in nächste Generation zu gelangen
- Verteilungsverzerrung meist weniger kritisch (d.h. eher tolerierbar) als ortabhängige Verzerrung
- **Beispiel: uniformes Crossover**
 - da jedes Gen unabhängig von allen anderen mit W'keit p_x ausgetauscht, Anzahl k der ausgetauschten Gene ist binomialverteilt mit Parameter p_x :

$$P(K = k) = \binom{n}{k} p_x^k (1-p_x)^{n-k} \quad \text{mit } n \hat{=} \text{ Gesamtzahl der Gene}$$

⇒ sehr kleine und sehr große Anzahlen sind unwahrscheinlicher

Übersicht

1. Motivation

2. Ein-Elter-Operatoren

3. Zwei- oder Mehr-Elter-Operatoren

4. Interpolierende und extrapolierende Rekombination

Interpolierende Operatoren

Extrapolierende Operatoren

5. Selbstanpassende Algorithmen

6. Zusammenfassung

Motivation [Weicker, 2007]

- bisher: nur *kombinierende Operatoren* für Verknüpfung mehrerer Individuen
 - Ein-Punkt-, Zwei-Punkt- und n -Punkt-Crossover
 - Uniformes (ordnungsbasiertes) Crossover
 - Shuffle Crossover
 - Kantenrekombination
 - Diagonal-Crossover
- alle stark abhängig von Diversität der Population
- erschaffen keine neuen Genbelegungen und können somit nur Teilbereiche von Ω erreichen, die in Individuen der Population enthalten
- falls Diversität einer Population groß, dann erforschen kombinierende Operatoren Ω sehr gut

Interpolierende Operatoren

- vermischen Eigenschaften der Eltern, sodass neues Individuum mit neuen Eigenschaften entsteht
 - Eigenschaften bewegen sich zwischen denen der Eltern
- ⇒ Ω somit weniger durchforstet
- interpol. Rekombination konzentriert Population auf 1 Schwerpunkt
 - fördert damit Feinabstimmung von sehr guten Individuen
 - um Ω anfangs genügend zu erforschen: Verwenden einer stark zufallsbasierte, diversitätserhaltende Mutation

Arithmetischer Crossover

- ist Beispiel für interpolierende Rekombination
- arbeitet auf reellwertigen Genotypen
- geometrisch: kann alle Punkte auf Strecke zwischen beiden Eltern erzeugen

Algorithm 3 Arithmetischer Crossover

Input: Individuen A, B mit $A.G, B.G \in \mathbb{R}^l$

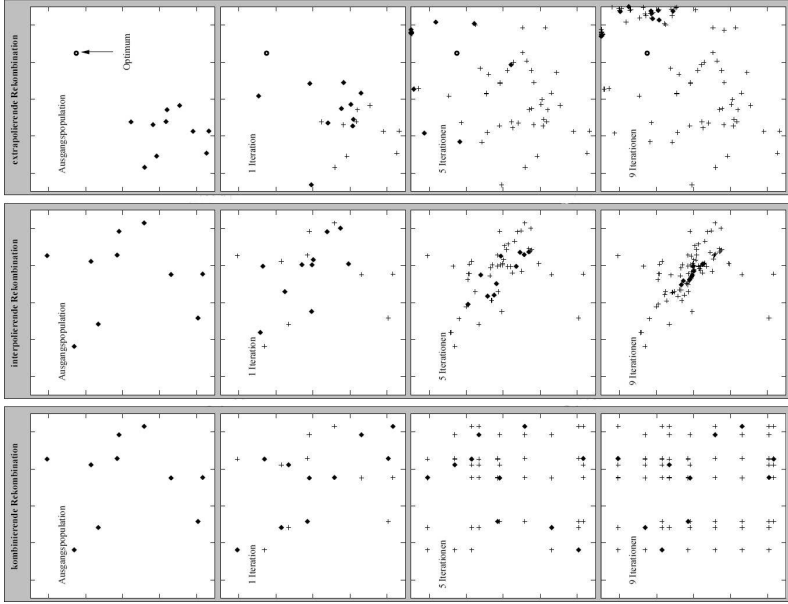
Output: neues Individuum C

- 1: $u \leftarrow$ wähle zufällig aus $U([0, 1])$
 - 2: **for** $i \in \{1, \dots, l\}$ {
 - 3: $C.G_i \leftarrow u \cdot A.G_i + (1 - u) \cdot B.G_i$
 - 4: }
 - 5: **return** C
-

Extrapolierende Operatoren

- versuchen gezielt Information aus mehreren Individuen abzuleiten
- ⇒ Erstellen eine Prognose, wo Güteverbesserungen zu erwarten sind
- extrapolierende Rekombination kann bisherigen Ω verlassen
 - ist einzige Art der Rekombination, die Gütewerte benutzt
 - Einfluss der Diversität hier schwer nachzuvollziehen
 - Algorithmus ist z.B. Arithmetisches Crossover mit $u \in U([1, 2])$

Vergleich



Übersicht

1. Motivation

2. Ein-Elter-Operatoren

3. Zwei- oder Mehr-Elter-Operatoren

4. Interpolierende und extrapolierende Rekombination

5. Selbstanpassende Algorithmen

Experiment anhand des TSP

Lokalität des Mutationsoperators

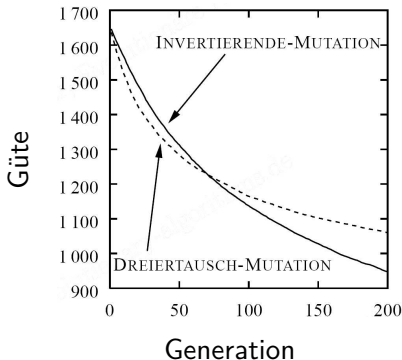
Anpassungsstrategien

6. Zusammenfassung

Selbstanpassende Algorithmen [Weicker, 2007]

- bisher: Mutation soll kleine Veränderung bzgl. Phänotyps vornehmen
 - jetzt: hinterfragen, ob dies zu jedem Zeitpunkt der Optimierung gilt
 - dafür Kontrollexperiment
 - TSP (hier 51 Städte) durch Hillclimbing lösen
- ⇒ keine Rekombination
- unterschiedlich lokale Mutationsoperatoren seien
 - Invertieren eines Teilstücks,
 - zyklischer Tausch dreier zufälliger Städte

Einfluss des Stands der Suche



- vermeintlich ungeeigneter Dreiertausch: in ersten 50 Generationen besser als favorisiertes Invertieren eines Teilstücks
- darum: Definieren der relativen erwarteten Verbesserung als Maß dafür, welche Verbesserung Operator bringt

Relative erwartete Verbesserung

Definition

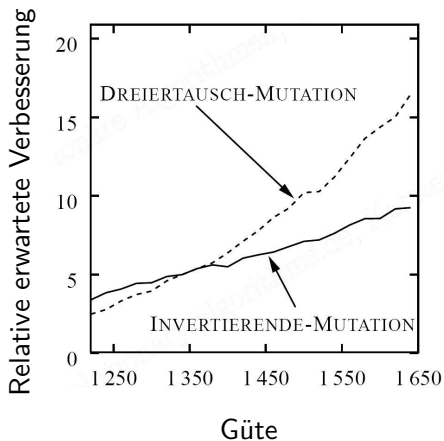
Die *Güteverbesserung* von einem Individuum $A \in \mathcal{G}$ zu einem Individuum $B \in \mathcal{G}$ wird definiert als

$$\text{Verbesserung}(A, B) = \begin{cases} |B.F - A.F| & \text{falls } B.F \succ A.F, \\ 0 & \text{sonst.} \end{cases}$$

Dann lässt sich die *relative erwartete Verbesserung* eines Operators Mut bzgl. Individuum A definieren als

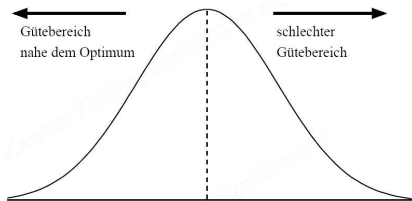
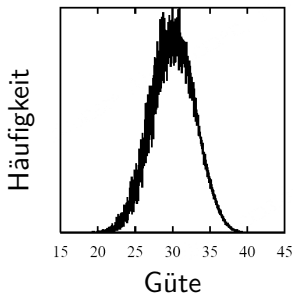
$$\text{relEV}_{\text{Mut}, A} = E \left(\text{Verbesserung}(A, \text{Mut}^{\xi}(A)) \right).$$

Einfluss des Stands der Suche



- Ermitteln der relativen erwarteten Verbesserung in unterschiedlichen Gütebereichen durch Stichproben aus Ω
 - verantwortlich für dargestellten Effekt
- ⇒ wie häufig sind einzelne Fitnesswerte in Ω ?

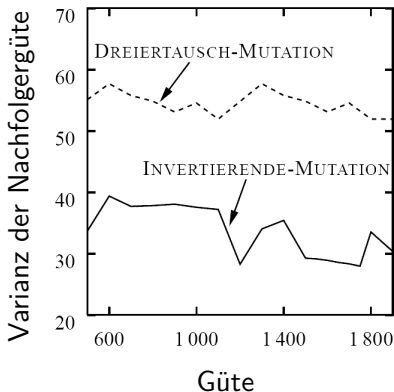
Gesamter Suchraum



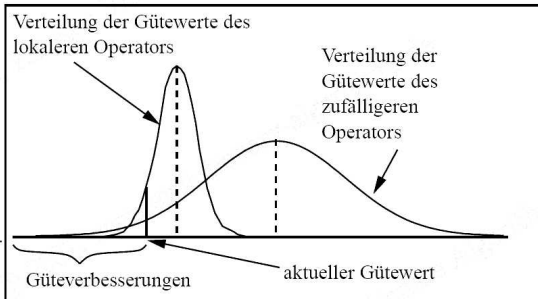
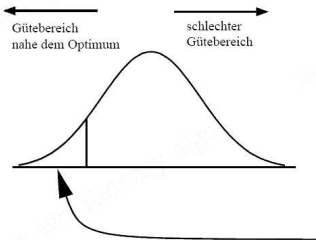
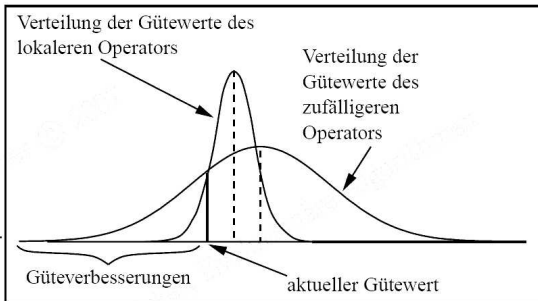
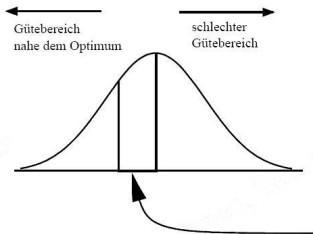
- links: Dichteverteilung eines TSP mit 11 Städten
- rechts: idealisierte Dichteverteilung eines Minimierungsproblems
- ähnliche Verteilung bei Kindindividuen (nach Mutation entstanden)

Varianz der erzeugten Güte

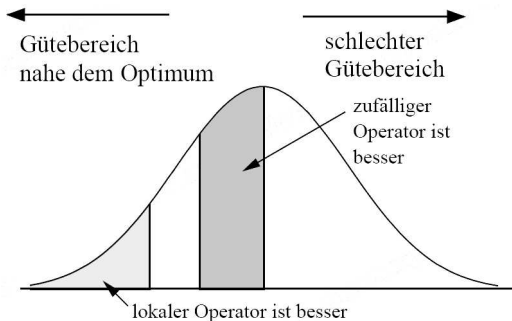
- wichtig ist, wie *lokal* Mutationsoperator ist
- sehr lokal \Rightarrow Gütewerte nahe der Güte des Elternindividuum
- wenig lokal \Rightarrow größerer Bereich an Gütewerten wird abgedeckt



- invert. Mut. ist über gesamten Gütebereich lokaler als Dreiertausch



Ergebnis der Überlegungen



- Qualität eines Mutationsoperators kann nicht unabhängig vom aktuellen Güteniveau beurteilt werden
- Operator ist niemals optimal über gesamten Verlauf der Optimierung
- bei zunehmender Annäherung an Optimum: lokalere Operatoren!

Anpassungsstrategien: 3 Techniken

Vorbestimmte Anpassung:

- lege Veränderung vorab fest

Adaptive Anpassung:

- erhebe Maßzahlen für Angepasstheit
- leite Anpassung von Regeln ab

Selbstadaptive Anpassung:

- nutze Zusatzinformation im Individuum
- zufallsbasiert sollen sich Parameter individuell einstellen

Vordefinierte Anpassung

Betrachtete Parameter:

- reellwertige Gauß-Mutation
- σ bestimmt durchschnittliche Schrittweite
- Modifikationsfaktor $0 < \alpha < 1$ lässt σ exponentiell fallen

Umsetzung:

Algorithm 4 Vordefinierte Anpassung

Input: Standardabweichung σ , Modifikationsfaktor α

Output: angepasste Standardabweichung σ

- 1: $\sigma' \leftarrow \alpha \cdot \sigma$
 - 2: **return** σ'
-

Adaptive Anpassung

- Maß: Anteil der verbessernden Mutationen der letzten k Generationen
- falls dieser Anteil „hoch“ ist, soll σ vergrößert werden

Algorithm 5 Adaptive Anpassung

Input: Standardabweichung σ , Erfolgsrate p_s , Schwellwert θ , Modifikationsfaktor $\alpha > 1$

Output: angepasste Standardabweichung σ

```
1: if  $p_s > \theta$  {  
2:   return  $\alpha \cdot \sigma$   
3: }  
4: if  $p_s < \theta$  {  
5:   return  $\sigma/\alpha$   
6: }  
7: return  $\sigma$ 
```

Selbstadaption

Umsetzung:

- Speichern der Standardabweichung σ bei Erzeugung des Individuums als Zusatzinformation
- ⇒ Verwenden eines *Strategieparameters*
(wird beim Mutieren leicht zufällig variiert)
- „gute“ Werte für σ setzen sich durch bessere Güte der Kinder durch

Experimenteller Vergleich

Testumgebung

- 10-dimensionale Sphäre
- Hillclimber
- **aber** pro Generation werden $\lambda = 10$ Kindindividuen erzeugt
- reellwertige Gauß-Mutation mit $\sigma = 1$
- Umweltselektion der Besten von Eltern und Kindern
- $\theta = \frac{1}{5}$ und $\alpha = 1.224$

Selbstadaptive Gauß-Mutation

Algorithm 6 Selbstadaptive Gauß-Mutation

Input: Individuum A mit $A.G \in \mathbb{R}^l$

Output: variiertes Individuum B mit $B.G \in \mathbb{R}^l$

1: $u \leftarrow$ wähle zufällig gemäß $\mathcal{N}(0, 1)$

2: $B.S_1 \leftarrow A.S_1 \cdot \exp(\frac{1}{\sqrt{l}} u)$

3: **for each** $i \in \{1, \dots, l\}$ {

4: $u \leftarrow$ wähle zufällig gemäß $\mathcal{N}(0, B.S_1)$

5: $B.G_i \leftarrow A.G_i + u_i$

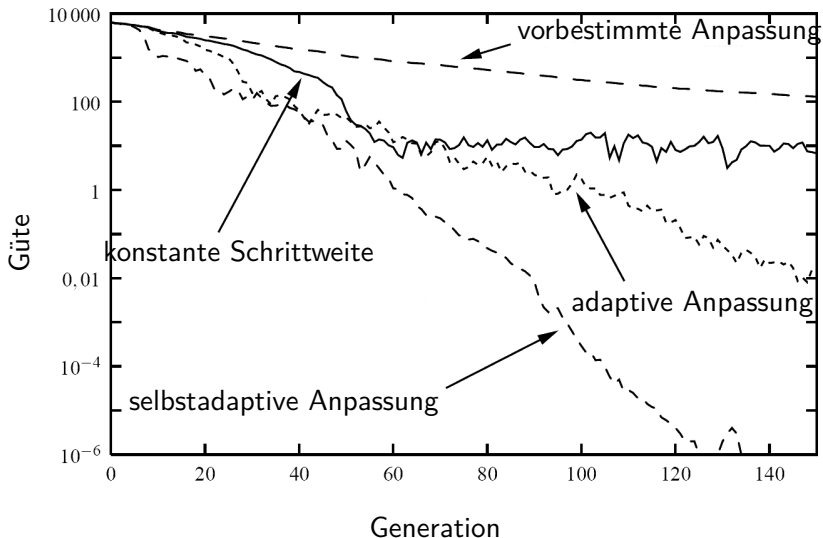
6: $B.G_i \leftarrow \max\{B.G_i, ug_i\}$ /* untere Wertebereichsgrenze ug_i */

7: $B.G_i \leftarrow \min\{B.G_i, og_i\}$ /* obere Wertebereichsgrenze og_i */

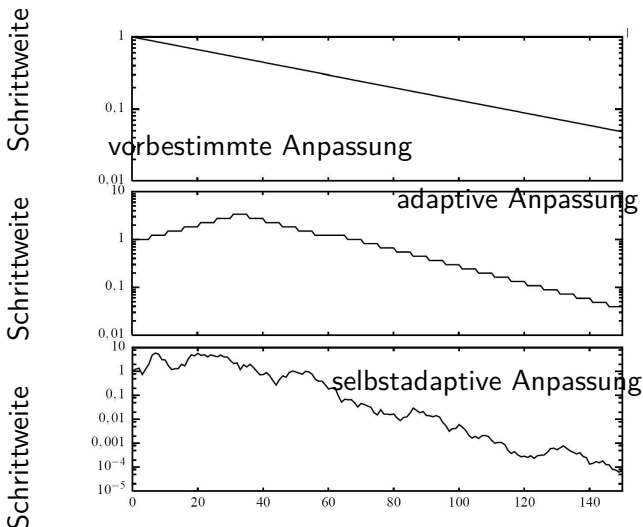
8: }

9: **return** B

Ergebnis des Vergleichs



Ergebnis des Vergleichs



Übersicht

1. Motivation

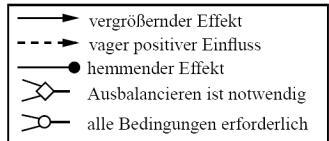
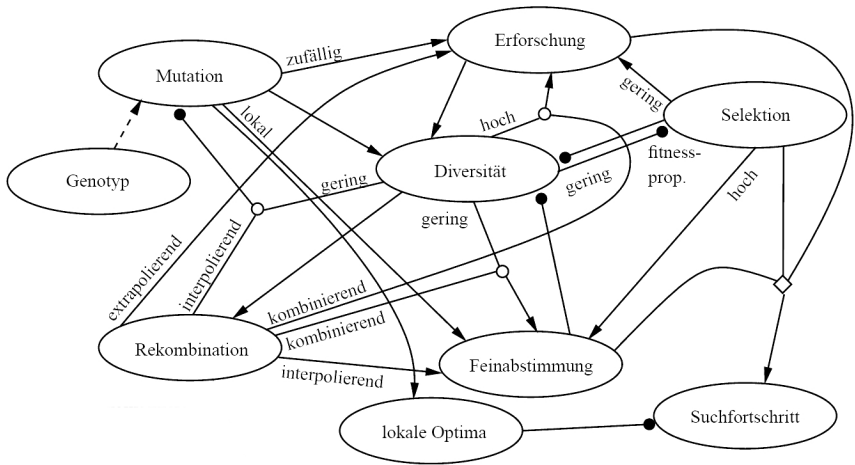
2. Ein-Elter-Operatoren

3. Zwei- oder Mehr-Elter-Operatoren

4. Interpolierende und extrapolierende Rekombination

5. Selbstanpassende Algorithmen

6. Zusammenfassung



Zusammenhänge I

Bedingung	Zielgröße	Erwarteter Effekt
Genotyp	Mutation	Nachbarschaft des Mutationsoperators wird beeinflusst
Mutation	Erforschung	zufällige Mutationen unterstützen Erforschung
Mutation	Feinabst.	gütelokale Mutationen unterstützen Feinabstimmung
Mutation	Diversität	Mutation vergrößert Diversität
Mutation	lokale Optima	gütelokale Mutationen erhalten lokale Optima des Phänotyps (zufällige Mutationen können noch mehr einführen)
Rekombination	Erforschung	extrapolierende Operatoren stärken Erforschung
Rekombination	Feinabst.	interpolierende Operatoren stören Feinabstimmung

Zusammenhänge II

Bedingung	Zielgröße	Erwarteter Effekt
Div./Rekomb.	Mutation	geringe Diversität und interpolierende Rekombination dämpfen Ausreißer der Mutation
Diversität	Rekombination	hohe Diversität unterstützt Funktionsweise der Rekombination
Selektion	Erforschung	geringer Selektionsdruck stärkt Erforschung
Selektion	Feinabst.	hoher Selektionsdruck stärkt Feinabstimmung
Selektion	Diversität	Selektion verringert meist Diversität
Div./Rekomb.	Erforschung	kombinierende Rekombination stärkt Erforschung bei hoher Diversität
Div./Rekomb.	Feinabst.	kombinierende Rekombination stärkt Feinabstimmung bei hoher Diversität

Zusammenhänge III

Bedingung	Zielgröße	Erwarteter Effekt
Erforschung	Diversität	erforschende Operationen erhöhen Diversität
Feinabst.	Diversität	feinabstimmende Operationen verringern Diversität
Diversität	Selektion	geringe Diversität verringert Selektionsdruck der fitnessproportionalen Selektion
lokale Optima	Suchfortschritt	viele lokale Optima hemmen Suchfortschritt
Erf./Fein./Sel.	Suchfortschritt	Ausbalancieren der drei Faktoren ist notwendig

Literatur zur Lehrveranstaltung



Weicker, K. (2007).

Evolutionäre Algorithmen.

Teubner Verlag, Stuttgart, Germany, 2nd edition.