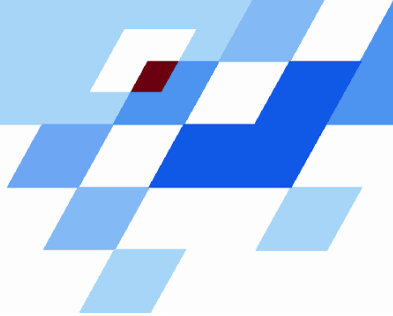


Wintersemester 2006/07

Fundamente der Computational Intelligence

(Vorlesung)

Prof. Dr. Günter Rudolph

Fachbereich Informatik

Lehrstuhl für Algorithm Engineering

Inhalt

- EA in ganzzahligen Suchräumen
- EDA, PBIL & Co.
- Differentialevolution
- Genetic Programming

Ziele:

1. Entwicklung eines EA für ganzzahlige Probleme: $\min\{ f(x) : x \in S \subseteq \mathbb{Z}^n\}$
2. Entwurf einer Mutationsverteilung mit Träger $\mathbb{Z}^n$
3. Mutationsverteilung sollte durch einen Parameter steuerbar sein
4. Mutationsverteilung sollte nur Annahmen bzgl. *starker Kausalität* reflektieren
   - Unimodalität
   - Symmetrie bzgl. einer Norm
5. Leichte Erzeugung von Mutationen (Pseudozufallszahlen)

Idee, um Ziele zu erreichen:

Konstruktion einer Mutationsverteilung mit maximaler Entropie
unter Berücksichtigung der Ziele 2 bis 4

Sei Z Zufallsvariable mit Träger $\mathbb{Z}$ und $p_k = P\{ Z = k \}$ für $k \in \mathbb{Z}$.

Dann ist $H(p) = -\sum_{k=-\infty}^{\infty} p_k \log p_k$ die Entropie von Z.

Ansatz:

Analytische Lösung eines ∞-dimensionalen, nichtlinearen Optimierungsproblems mit Nebenbedingungen!

$$H(p) = -\sum_{k=-\infty}^{\infty} p_k \log p_k \longrightarrow \text{max!}$$

unter

$$p_k = p_{-k} \quad \forall k \in \mathbb{Z}\,, \qquad \text{(Symmetrie bzgl. 0)}$$

$$\sum_{k=-\infty}^{\infty} p_k = 1\,, \qquad \text{(Normalisierung)}$$

$$\sum_{k=-\infty}^{\infty} |k|\, p_k = \theta \qquad \text{(„Streuung“ steuerbar)}$$

$$p_k \geq 0 \quad \forall k \in \mathbb{Z}\,. \qquad \text{(Nichtnegativität)}$$

Resultat:

Eine Zufallsvariable Z mit Träger $\mathbb{Z}$ und Wahrscheinkeitsverteilung

$$p_k := P\{Z = k\} = \frac{q}{2-q}(1-q)^{|k|} \quad , \; k \in \mathbb{Z}, \; q \in (0,1)$$

ist symmetrisch bzgl. 0, unimodal, kontrollierbar durch *q* und von max. Entropie. ■

Erzeugung von Pseudozufallszahlen: $Z = G_1 - G_2$

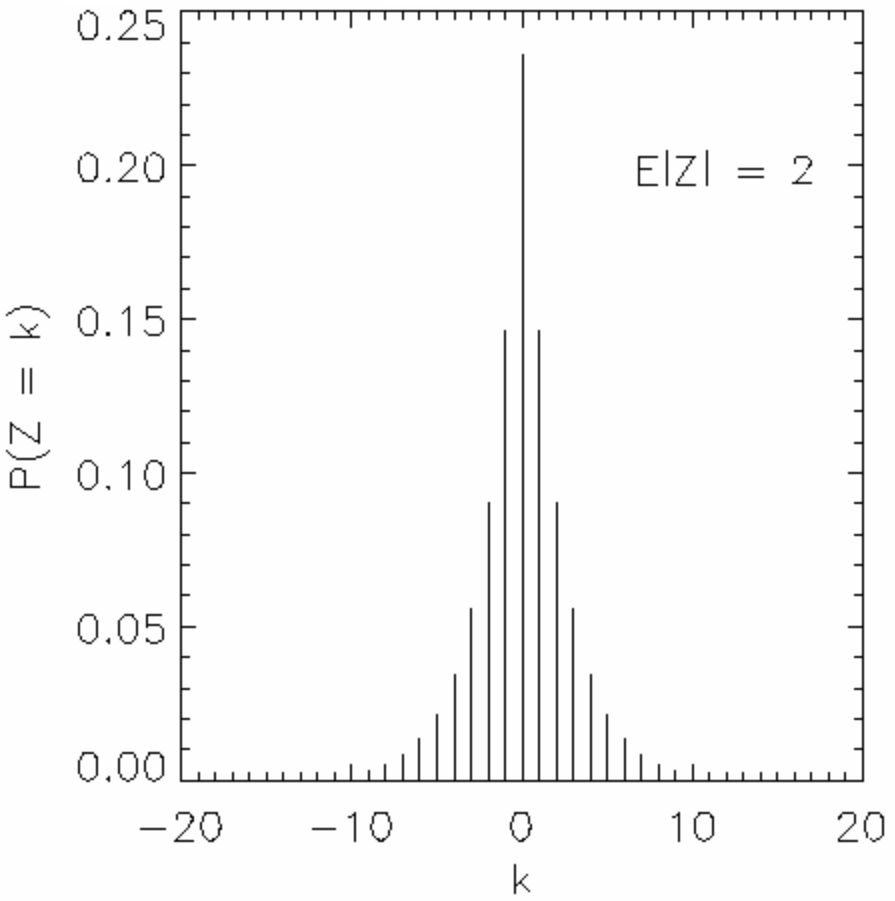
wobei

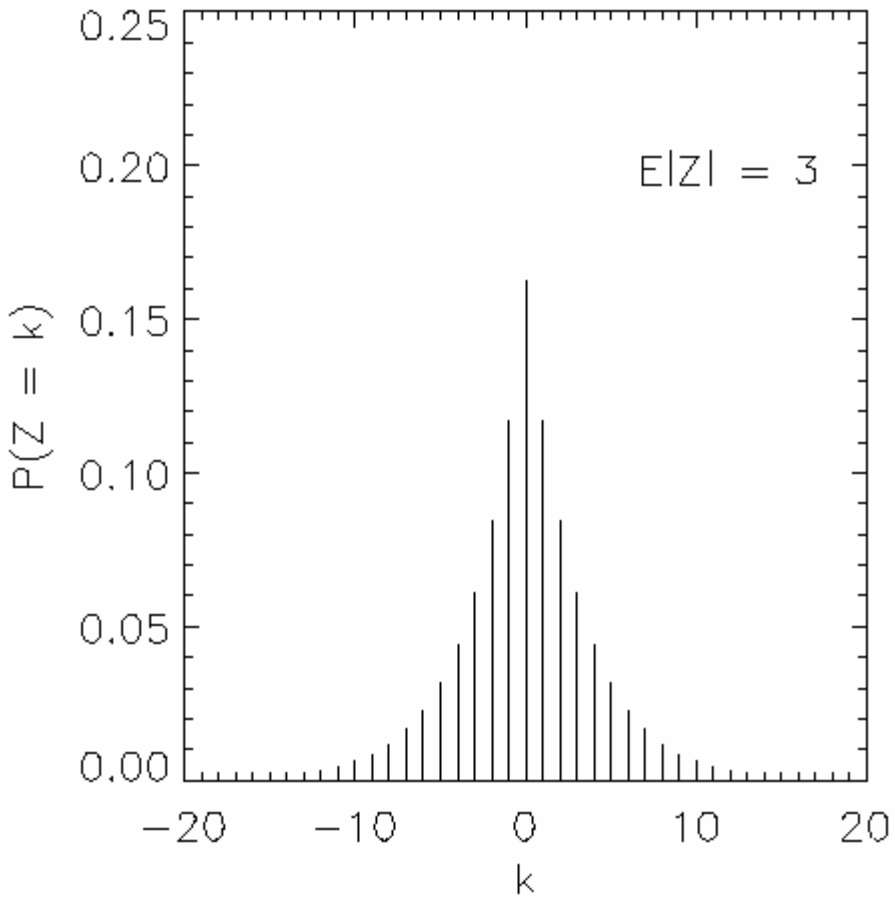
$$U_i \sim U(0,1) \quad \Rightarrow \quad G_i = \left\lfloor \frac{\log(1-U_i)}{\log(1-q)} \right\rfloor \quad , \; i = 1, \; 2.$$

stochastisch unabhängig!

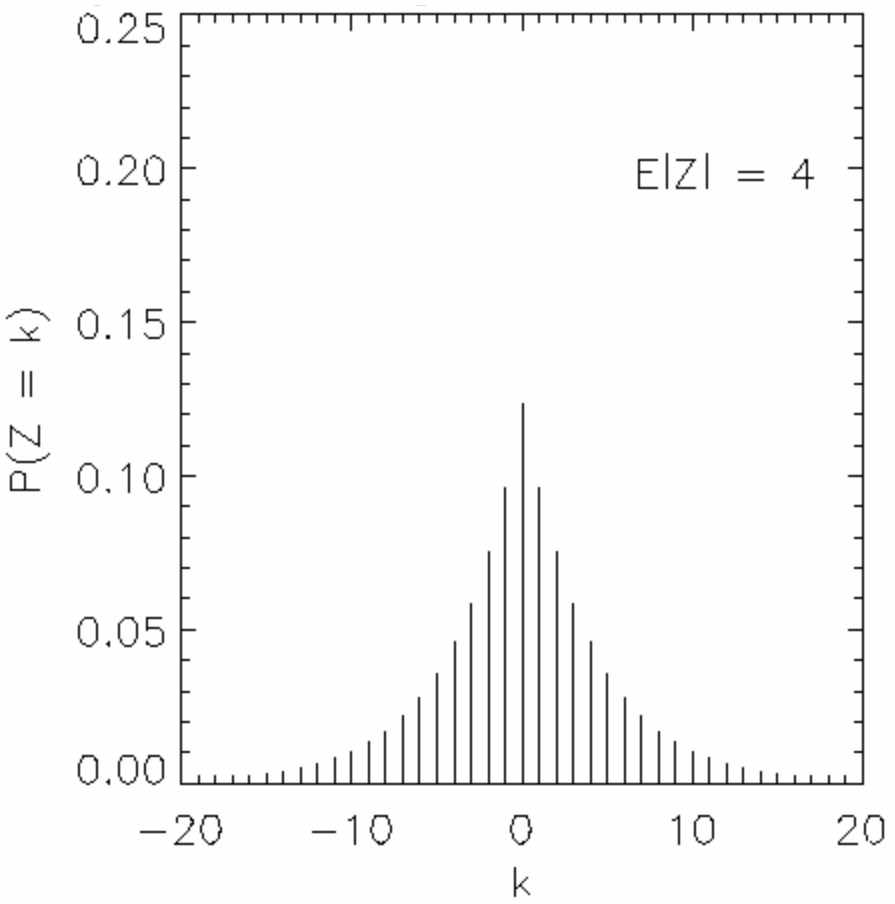
bzgl. Ziel 5: Vertretbarer Aufwand!

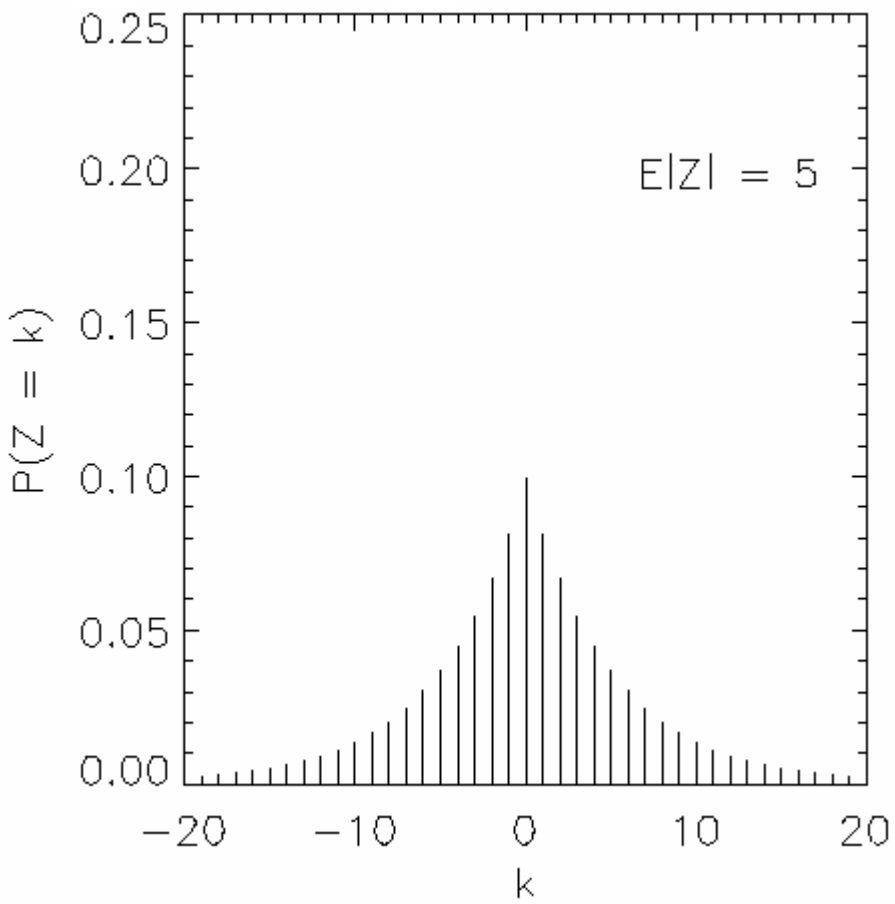
Wahrscheinlichkeitsverteilung für verschiedene mittlere Schrittweiten E|Z| = θ





Wahrscheinlichkeitsverteilung für verschiedene mittlere Schrittweiten E|Z| = θ





Zur Kontrollierbarkeit:

Wir müssen q $\in$ (0,1) anpassen können, um Z mit variablem E|Z| = θ zu erzeugen!

Selbstanpassung von q im Bereich (0,1) ?

⟶ mittlere Schrittweite $E[\,|Z|\,]$ einstellbar machen!

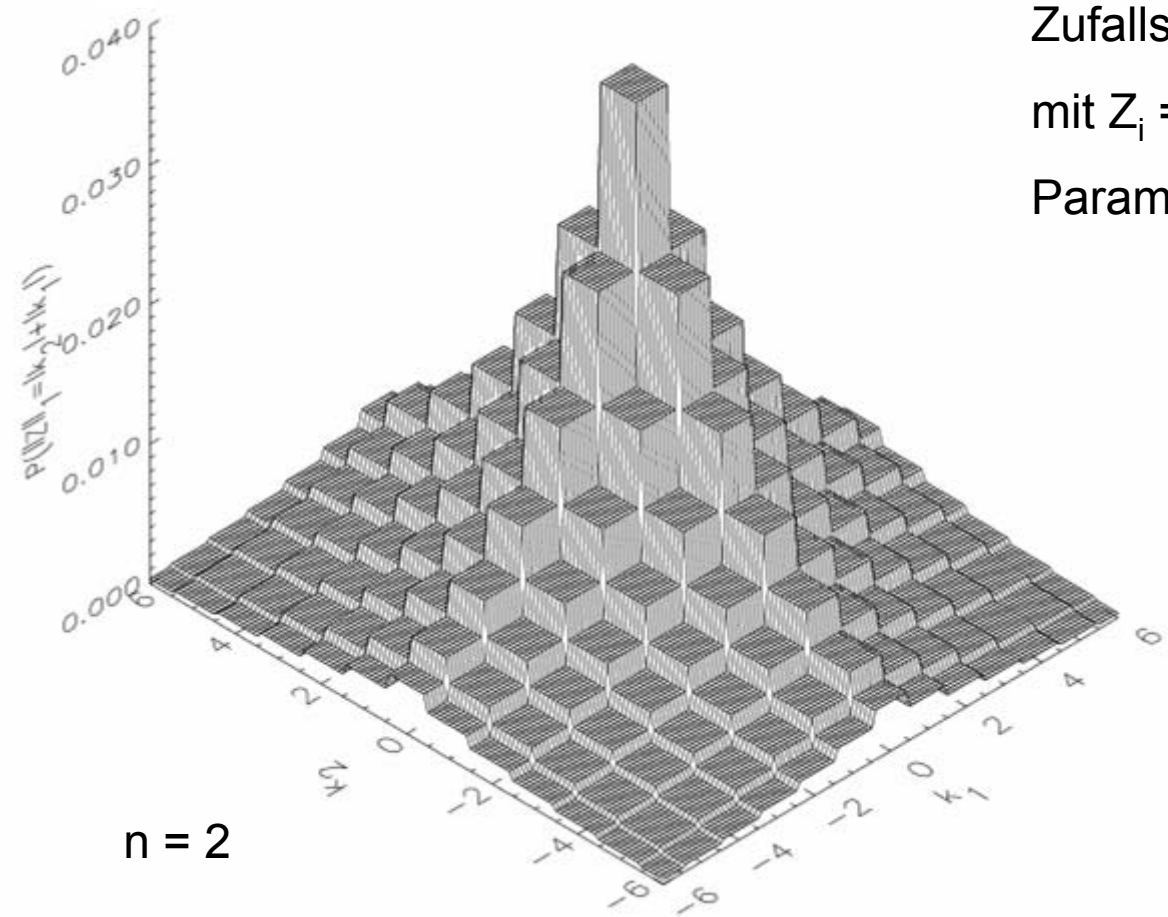
$$E[\,|Z|\,] = \sum_{k=-\infty}^{\infty} |k|\, p_k = \theta = \frac{2\,(1-q)}{q\,(2-q)} \quad \Leftrightarrow \quad q = 1 - \frac{\theta}{(1+\theta^2)^{1/2}+1}$$

$\theta \in \mathbb{R}_+$ $\qquad$ $q \in (0,1)$

→ θ durch mutative Selbstanpassung einstellen! $\qquad$ → q aus θ berechnen!

wie mutative Schrittweitensteuerung
der σ bei EA in Suchraum $\mathbb{R}^n$!

n - dimensionale Verallgemeinerung



n = 2

Zufallsvektor $Z = (Z_1, Z_2, \ldots Z_n)$

mit $Z_i = G_{1,i} - G_{2,i}$ (stoch. unabh.);

Parameter q für alle G_{1i}, G_{2i} gleich

n - dimensionale Verallgemeinerung

$$P\{Z_i = k\} = \frac{q}{2-q}(1-q)^{|k|}$$

$$P\{Z_1 = k_1, Z_2 = k_2, \ldots, Z_n = k_n\} = \prod_{i=1}^{n} P\{Z_i = k_i\} =$$

$$\left(\frac{q}{2-q}\right)^n \prod_{i=1}^{n} (1-q)^{|k_i|} = \left(\frac{q}{2-q}\right)^n (1-q)^{\sum_{i=1}^n |k_i|}$$

$$= \left(\frac{q}{2-q}\right)^n (1-q)^{\|k\|_1} .$$

$\Rightarrow$ n-dimensionale Verteilung ist symmetrisch zur ℓ_1-Norm!

$\Rightarrow$ alle Zufallsvektoren mit gleicher Schrittlänge haben gleiche W‘keit!

Zur Kontrollierbarkeit von E[|| Z ||$_1$]

$$E[\,\|Z\|_1\,] = E\left[\sum_{i=1}^{n} |Z_i|\right] = \sum_{i=1}^{n} E[|Z_i|] = n \cdot E[|Z_1|]$$

aus Def. — Linearität von E[·] — gleiche Verteilung der Z_i

$$\underbrace{n \cdot E[\,|Z_1|\,]}_{=\theta} = n \cdot \frac{2\,(1-q)}{q\,(2-q)} \quad \Leftrightarrow \quad q = 1 - \frac{\theta/n}{(1+(\theta/n)^2)^{1/2}+1}$$

$=\theta$ ← Selbstanpassung

aus θ berechnen!

Algorithmus:

Individuum	:	$(x, \theta) \in \mathbb{Z}^n \times \mathbb{R}_+$
Mutation	:	$\theta^{(t+1)} = \theta^{(t)} \cdot \exp(N), \quad N \sim N(0, 1/n).$
		if $\theta^{(t+1)} < 1$ then $\theta_{t+1} = 1$
		berechne neues q für G_i aus θ_{t+1}
		$\forall j = 1, \ldots, n : X_j^{(t+1)} = X_j^{(t)} + (G_{1,j} - G_{2,j})$
Rekombination	:	diskret
Selektion	:	(μ, λ)-Selektion

(Rudolph, PPSN 1994)

EDA = Estimation of Distribution Algorithm

Idee:

Repräsentiere Population nicht durch Individuen sondern durch ihre Verteilung!

Prinzip:

Starte mit Anfangsverteilung P(0) mit Träger S, t = 0 (S: Suchraum)

Erzeuge λ Nachkommen aus P(t)

Selektiere μ Eltern (irgendwie)

Aktualisiere P(t) unter Verwendung der selektierten Eltern zu P(t+1)

t = t + 1

Abbruch?

nein

PBIL: Population-Based Incremental Learning

$\in$ EDA

- Baluha & Caruana (1995), Conference on Machine Learning
- Suchraum $\mathbb{B}^n$, Minimierung pseudo-boolescher Funktionen
- Population X mit μ Individuen

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{\mu 1} & x_{\mu 2} & \cdots & x_{\mu n} \end{pmatrix}$$

Individuum 1
Individuum 2
⋮
Individuum μ

Genpool 1 Genpool 2 ... Genpool n

$$p_j = \frac{1}{\mu} \sum_{i=1}^{\mu} x_{ij}$$

Genpool-Rekombination:

Ziehe Gen j für Nachkomme Y gleichverteilt von Komponente j aller Individuen

$\Updownarrow$

Setze Gen j für Nachkomme Y auf 1 mit W'keit $p_j = \frac{1}{\mu} \sum_{i=1}^{\mu} x_{ij}$, sonst auf 0

$\Rightarrow$ Population charakterisiert durch Verteilungen p = (p_1, p_2, …, p_n)

Grundidee der Evolutionsschleife:

1. Generiere λ Nachkommen durch Genpool-Rekombination gemäß p
2. Selektiere μ beste Nachkommen als neue Eltern
3. Aktualisiere p gemäß Genpools der neuen Eltern

Seien $y_{1:\lambda}, y_{2:\lambda}, \ldots, y_{\lambda:\lambda}$ die λ sortierten Nachkommen mit

$$f(y_{1:\lambda}) \leq f(y_{2:\lambda}) \leq \cdots \leq f(y_{\lambda:\lambda})$$

Setze $p^{(0)} = (\frac{1}{2}, \ldots, \frac{1}{2})$ und $k = 0$

repeat

Erzeuge λ Nachkommen y_i gem $p^{(k)}$

Sortiere y_i gemäß ihrer Güte

$$p^{(k+1)} = (1-\alpha)\, p^{(k)} + \alpha \frac{1}{\mu} \sum_{k=1}^{\mu} y_{k:\lambda}^{(k)}$$

until Terminierungsregel erfüllt

$\alpha \in (0,1)$

Achtung!

Aktualisierungsregel $p^{(k+1)} = \frac{1}{\mu} \sum_{i=1}^{\mu} y_{i:\lambda}^{(k)}$ problematisch!
($\alpha = 1$)

$\Rightarrow$ kann zu jedem p $\in \mathbb{B}^n$ führen (durch unglückliches Ziehen)

Analyse:

- Kvasnicka et al. (1995): 1-dimensional, deterministisch
- Rudolph & Höhfeld (1997): Spezialfall $\mu = 1$
- Gonzalez et al. (2001): 2-dimensional, leider falsch!
- ???

Analyse (Höhfeld & Rudolph 1997)

Aktualisierung: $p^{(k+1)} = (1-\alpha)\, p^{(k)} + \alpha\, b^{(k)}$

wobei $b^{(k)} = y_{1:\lambda}^{(k)},\ \alpha \in (0,1),\ p_i^{(0)} = \frac{1}{2}.$

Ziel:

Wir suchen Grenzwert $p^{(\infty)}$ der stochastischen Folge $(p^{(k)} : k \geq 0)$!

Vorüberlegung:

p ist beschränkt, also Konvergenz in W'keit = Konvergenz im Mittel

also: $\lim_{k\to\infty} E[\, p^{(k)}\,] = x \in \{0,1\}^n$ **?**

$$\mathrm{E}[\,p^{(t+1)}\,|\,p^{(t)}\,] = (1-\alpha)\,p^{(t)} + \alpha\,\underbrace{\mathrm{E}[\,b^{(t)}\,|\,p^{(t)}\,]}_{?}$$

$$\mathrm{E}[\,b\,|\,p\,] = \sum_{x\in\{0,1\}^n} x \cdot \underbrace{\mathrm{P}\{\,b = x\,\}}_{?}$$

$$\mathrm{P}\{\,b = x\,\} =$$

$$\mathrm{P}\{\,s = x\,\} \sum_{i=0}^{\lambda-1} \mathrm{P}\{\,f(s) > f(x)\,\}^i \cdot \mathrm{P}\{\,f(s) \geq f(x)\,\}^{\lambda-1-i}$$

$$\mathrm{P}\{ s = x \} = \prod_{i=1}^{n} p_i^{x_i} (1 - p_i)^{1 - x_i}$$

$$\mathrm{P}\{ f(s) = f(x) \} = \sum_{\substack{y \in \{0,1\}^n \\ f(y) = f(x)}} \mathrm{P}\{ s = y \}$$

$$\mathrm{P}\{ f(s) > f(x) \} = \sum_{\substack{y \in \{0,1\}^n \\ f(y) > f(x)}} \mathrm{P}\{ s = y \}$$

Beispiel: n = 2, $\lambda = 2$, $f(x) = \sum_i x_i \rightarrow$ min! (counting ones)

$$\mathrm{P}\left\{ b = \binom{0}{0} \right\} = 1 - (p_1 + p_2 - p_1 p_2)^2$$

$$\mathrm{P}\left\{ b = \binom{0}{1} \right\} = (1 - p_1)\, p_2\, (p_1 + p_2)$$

$$\mathrm{P}\left\{ b = \binom{1}{0} \right\} = p_1\, (1 - p_2)\, (p_1 + p_2)$$

$$\mathrm{P}\left\{ b = \binom{1}{1} \right\} = (p_1 p_2)^2$$

$$\mathrm{E}[\, b \,|\, p \,] := F_2(p) = \begin{pmatrix} p_1\, (1 - p_2)\, (p_1 + p_2) + (p_1 p_2)^2 \\ p_2\, (1 - p_1)\, (p_1 + p_2) + (p_1 p_2)^2 \end{pmatrix}$$

Vektorfeld der Abbildung $F_2(p)$

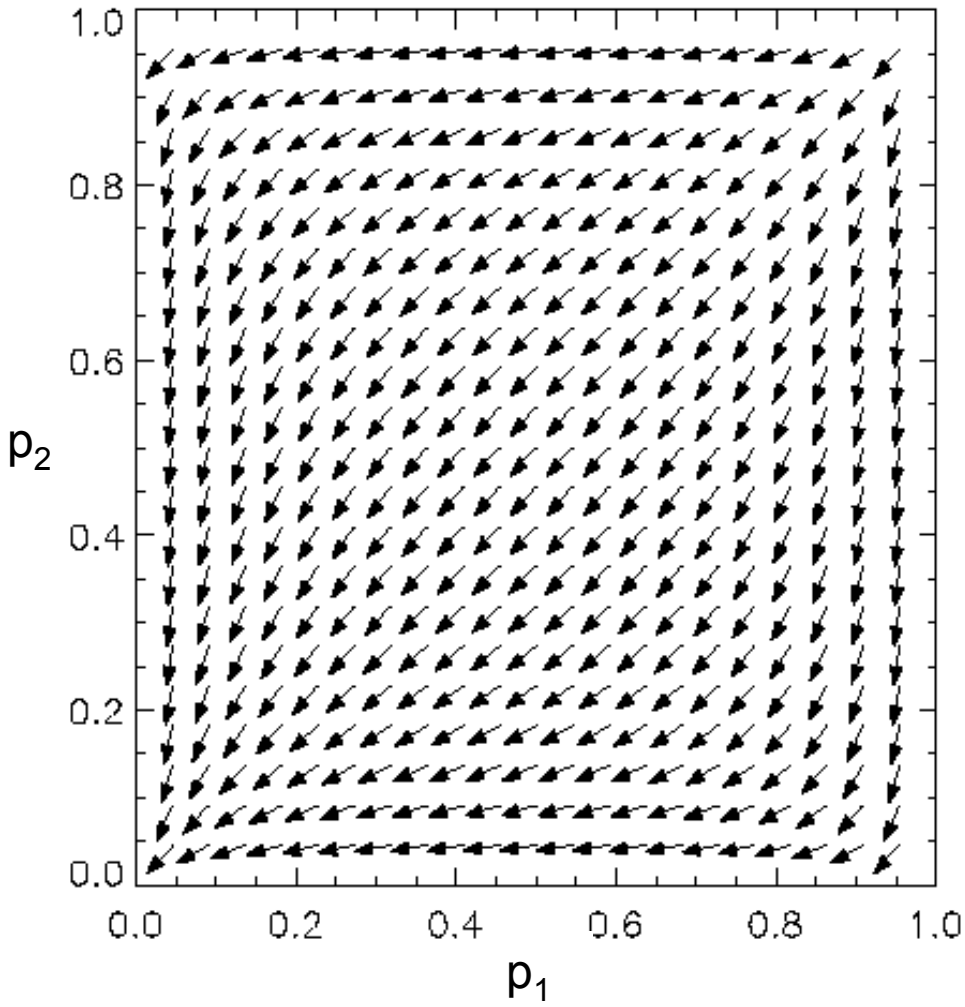
augenscheinlich: $F_2(p) < p$

<u>Interpretation:</u>

Unabhängig vom Startwert $p^{(0)}$ und unabhängig von aktueller Position $p^{(k)}$ hat $E[p^{(k+1)}]$ eine Tendenz in Richtung Optimum.

bzw.

$$E[p^{(k)}] \xrightarrow{\text{i.M.}} x^* \text{ für } k \to \infty$$



Satz:

Sei f(x) = c‘ x eine lineare pseudo-boolesche Funktion, die zu minimieren ist.

Für $c_i < 0$ gilt $E[b_i \mid p] > p_i$, für $c_i > 0$ gilt $E[b_i \mid p] < p_i$, wobei b das selektierte beste Individuum ist.

Daraus folgt: PBIL konvergiert im Mittel zum Optimum linearer Funktionen.

Beweis: (Rudolph & Höhfeld 1997, S. 3f) ■

→ Lineare Funktionen sind einfach! (1+1)-EA braucht i.M. O(n log n) Schritte.

→ Ähnliches Resultat für nichtlineare Funktionen?

→ Wenn Beweise nicht gelingen wollen, dann Gegenbeispiel finden!

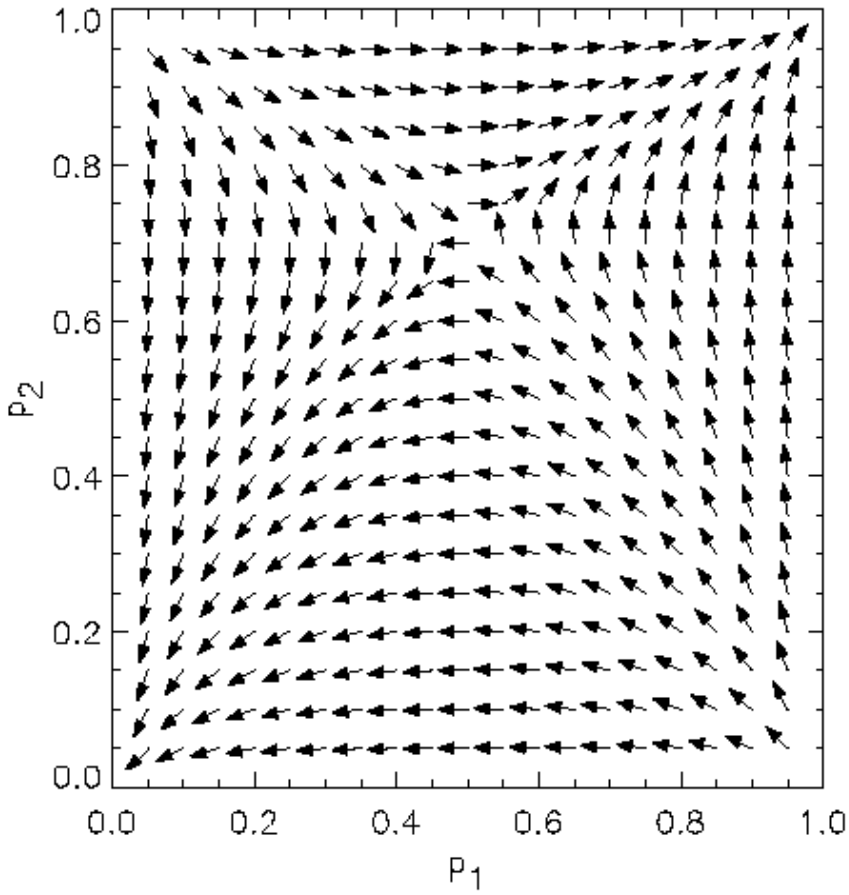
$f(00) < f(11) < f(01) < f(10)$

z. B. $f(x) = 3x_1 + 2x_2 - 4x_1x_2$

<u>Interpretation:</u>

Abhängig vom Startwert $p^{(0)}$
Tendenz zum lokalen oder globalen Optimum!

Kein globales Verfahren!



- Price (1996) + Storn (1996)
- (μ + μ)-EA mit speziellem Variationsoperator

initialisiere μ Individuen $P = (X_1, X_2, \ldots, X_\mu)$

```
repeat
   Q = ( )                          // leere Liste
   foreach Individuum x ∈ P:
      wähle 3 Individuen ≠ x aus P ohne Zurücklegen // ergibt (a,b,c)
      y = Variation(x, a, b, c)
      if f(y) < f(x) then Q.add(y) else Q.add(x)
   endfor
   P = Q
until Terminierung
```

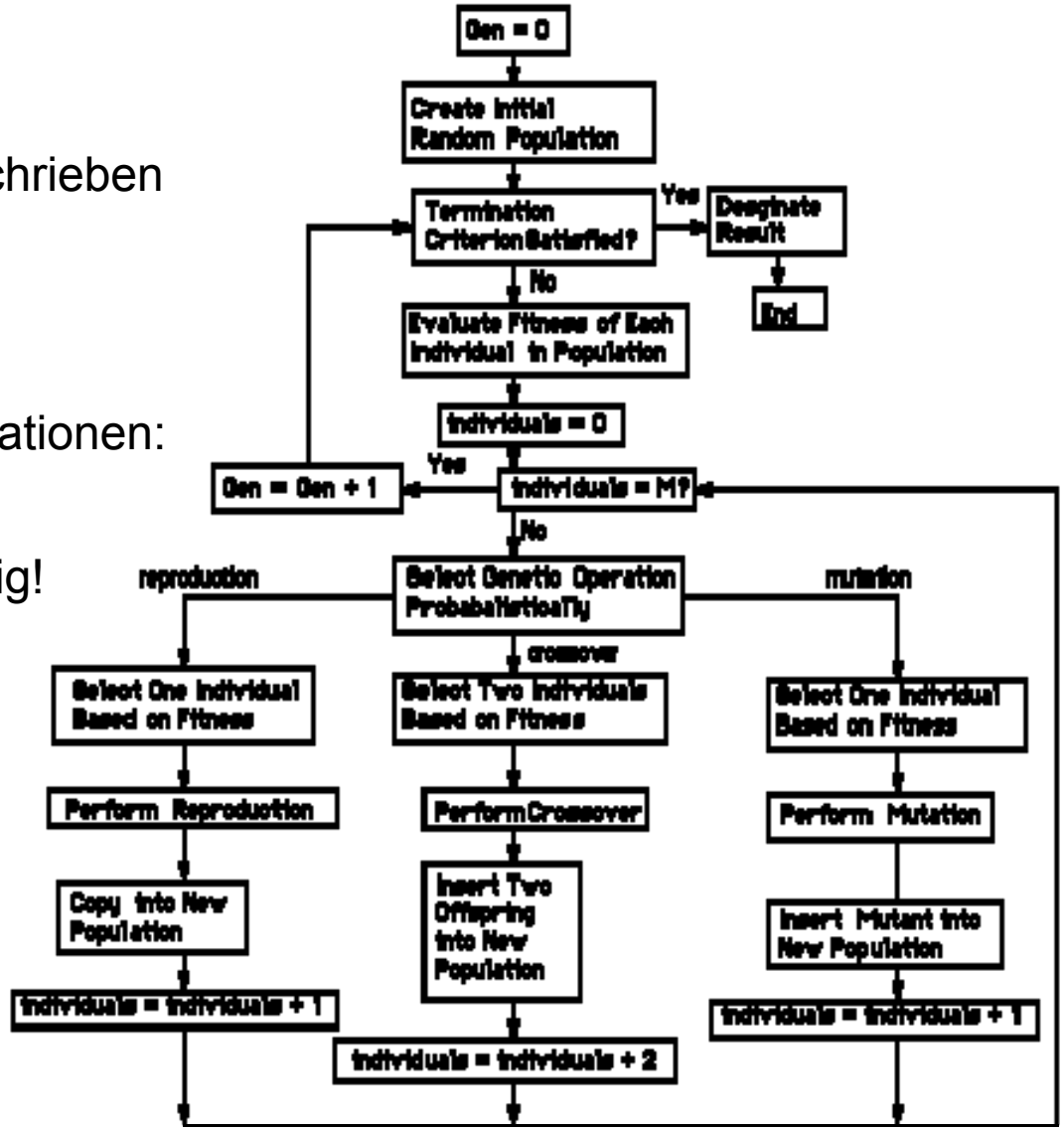
```
Variation(x, a, b, c) =
   wähle gleichverteilt ein j aus {1, 2, …, n}
   for i = 1 to n
     if i = j or mit W'keit p:
       y_i = a_i + α(b_i – c_i)
     else
       y_i = x_i
   endfor
   return y
```

p: Rekombinationsrate
$\alpha \in (0,1)$

Problem:

Der kleinste Hyperquader, der die initiale Population beinhaltet, kann nicht verlassen werden!

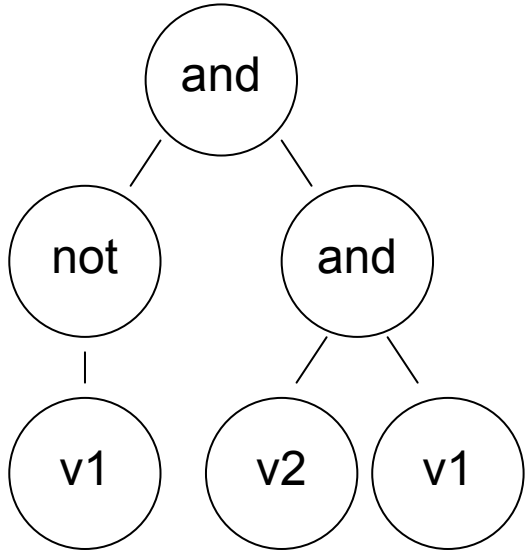
- entstanden im GA-Umfeld
- wird meist Koza (1989) zugeschrieben
- Evolution von Programmen
- Suchraum S: Syntaxbäume
- später auch andere Repräsentationen:
  z.B. Assembler, binary GP, …
- neue Variationsoperatoren nötig!

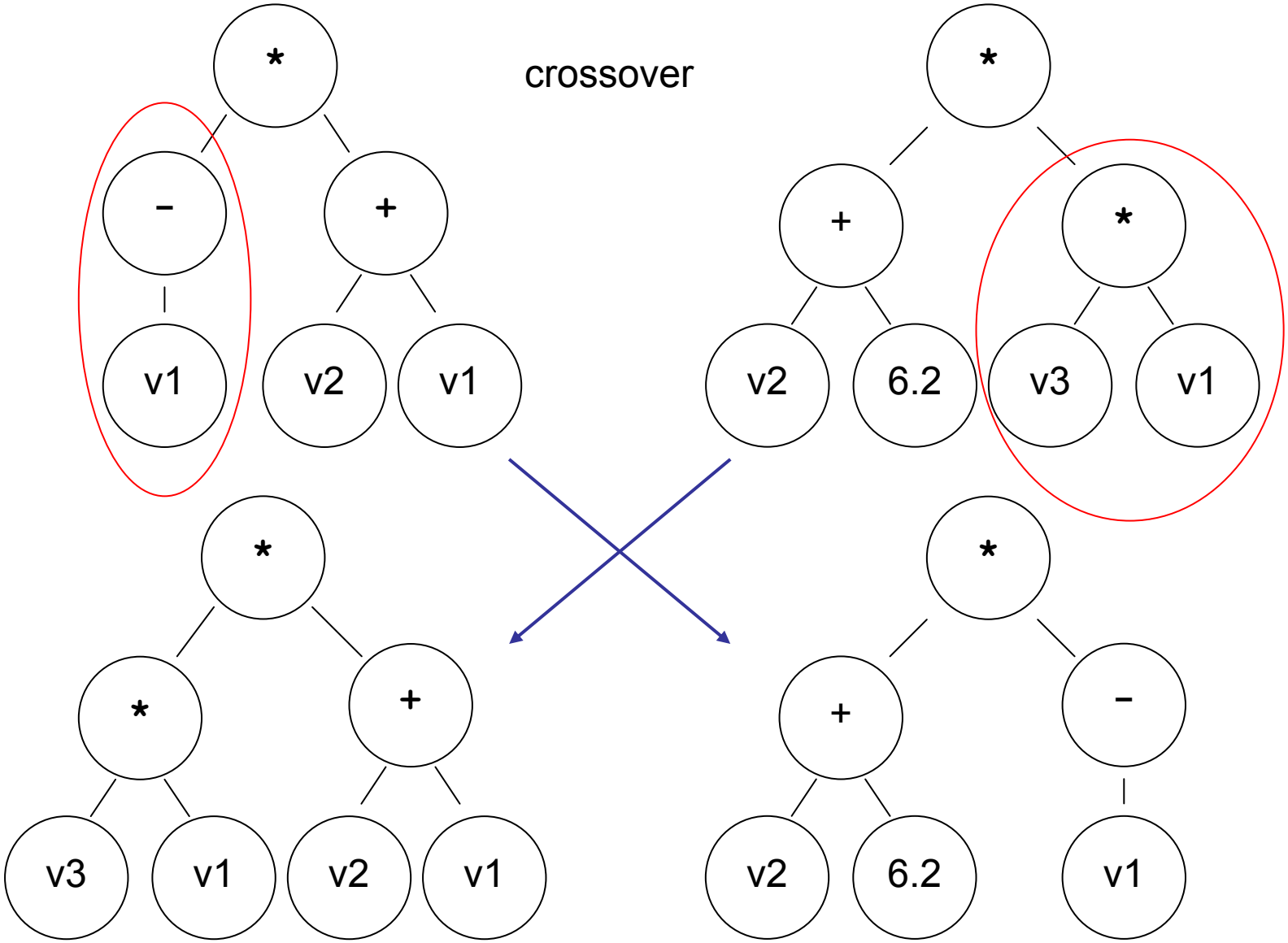


Klasse `tree`

Methoden:

```
createTree()
delete(node)
getSubtree(node)
insertTreeAt(node, tree)
contains(node)
size()
...
```




crossover
*
-
+
v1
v2
v1
*
+
*
v2
6.2
v3
v1
*
*
+
v3
v1
v2
v1
*
+
-
v2
6.2
v1

Mutation:

- Ersetzen eines Teilbaums durch zufälligen Baum
- Nur Konstanten werden verändert (keine Operationen)
- Wird eher selten eingesetzt
  → meistens riesige Populationen nur mit Crossover

Typische Anwendung

Symbolische Regression (Evolvieren einer Formel)

Theorie

sehr junges Feld (> 2000), Poli et al.
Schema-Theorie genannt, ist aber i.W. Markoff-Theorie