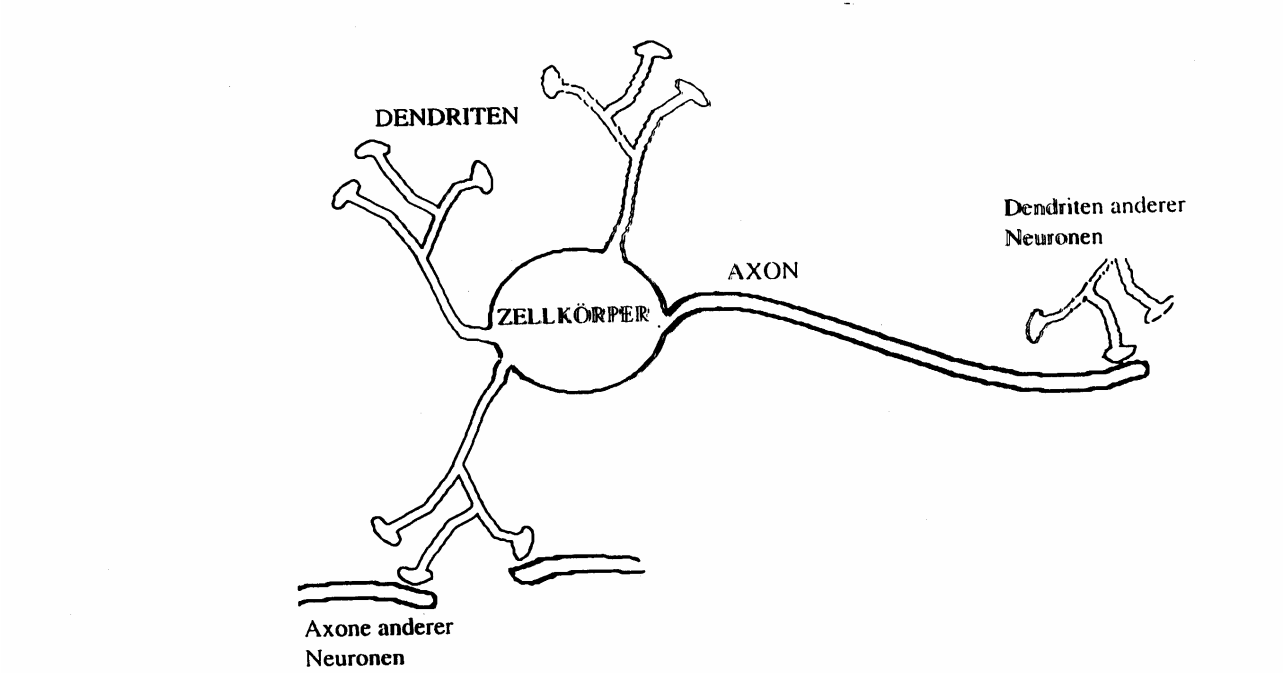
Exkurs Modelle und Algorithmen

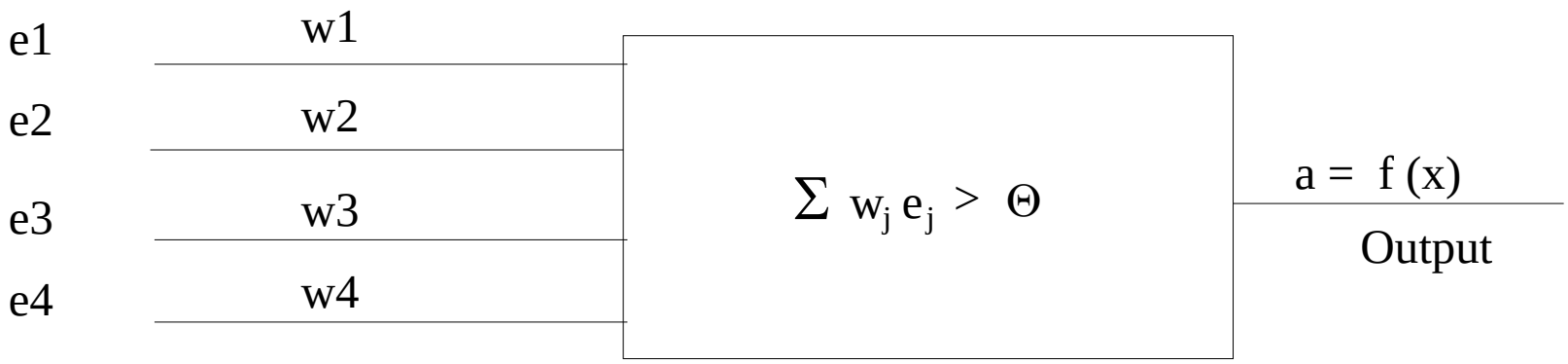
Ansatz künstlich neuronaler Netze (KNN)

- Versuch, die Wirkungsweise menschlicher Gehirnzellen nachzubilden
- dabei wird auf formale mathematische Beschreibungen und Algorithmen zurückgegriffen

Funktionsweise von Neuronen



mathematische Modellierung

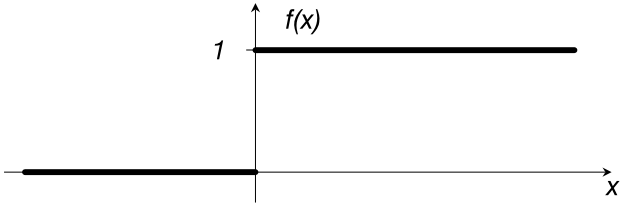


Die Eingangswerte e1, e2, e3 und e4 werden mit den Synapsenwerten w1, w2, w3 und w4 multipliziert. Falls die Summe den Schwellwert übersteigt, "feuert" das Neuron (0<a1); ansonsten bleibt a=0

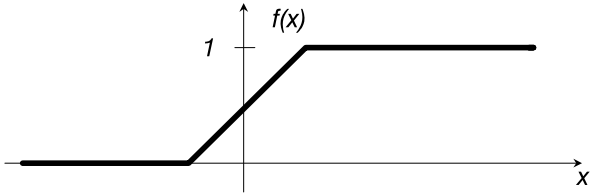
f(x) ist die sogenannte Transferfunktion $x = \sum w_j e_j - \Theta$

Transferfunktionen

binäre Funktion

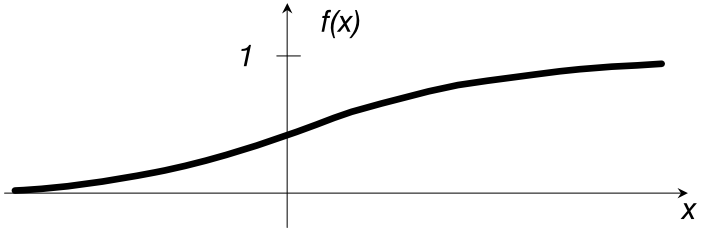


$$f(x) = \begin{cases} 1 & x > 0 \\ 0 & \text{sonst} \end{cases}$$



$$f(x) = \begin{cases} 0 & x \leq -\frac{c}{2} \\ \frac{x}{c} + \frac{1}{2} & -\frac{c}{2} < x \leq \frac{c}{2} \\ 1 & x > \frac{c}{2} \end{cases}$$

sigmoide Funktion



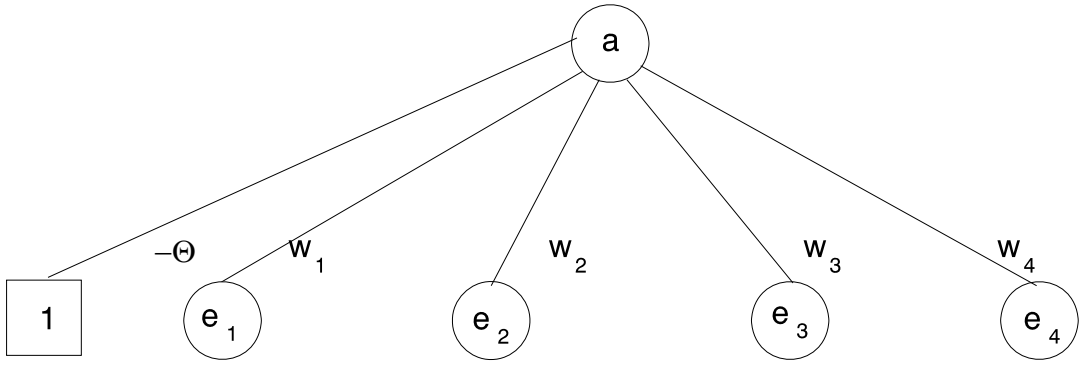
$$f(x) = \frac{1}{1 + e^{-cx}}$$

mathematische Modellierung und graphische Darstellung

Output a ergibt sich mit:

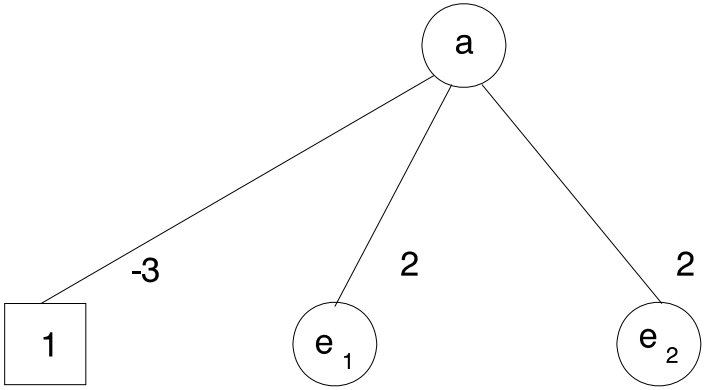
$$a = \begin{cases} 1 & \text{falls} \quad \sum w_j e_j > \Theta \\ 0 & \text{sonst} \end{cases} \qquad \text{bzw.} \quad a = \begin{cases} 1 & \text{falls} \quad (\sum w_j e_j - \Theta) > 0 \\ 0 & \text{sonst} \end{cases}$$

damit kann man den Schwellwert als mit (-1) gewichtete "Zusatzeingabe" auffassen: $a = f\left(\sum w_j e_j + (-1)\Theta\right)$

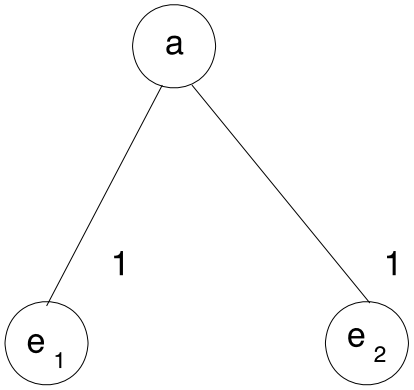


Beispiele

UND - Funktion

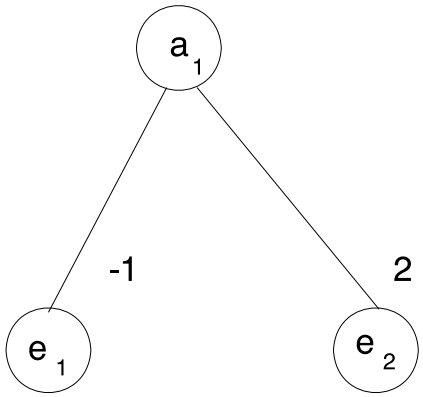


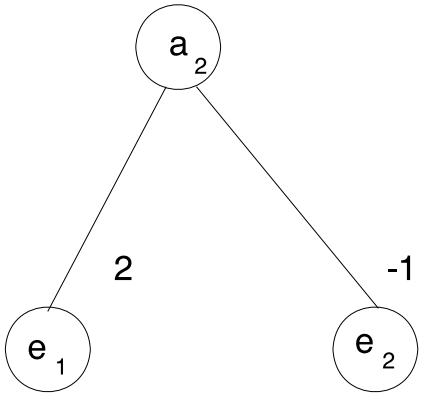
ODER - Funktion



$(\Theta = 0)$

Verknüpfung von Neuronen





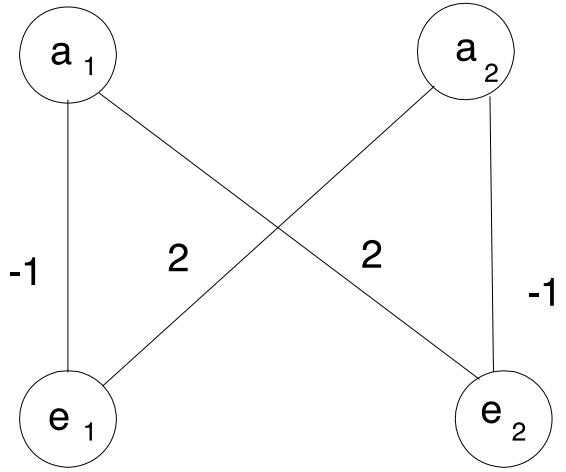
$a_1 = f(net_1 - 0)$

$net_1 = -1 \cdot e_1 + 2 \cdot e_2$

$a_2 = f(net_2 - 0)$ (f binäre Funktion, $\Theta = 0$)

$net_2 = 2 \cdot e_1 - 1 \cdot e_2$

➔



Netz liefert:

e_1	e_2	a_1	a_2
0	0	0	0
1	0	0	1
0	1	1	0
1	1	1	1

Sei:

$$\underline{e} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}; \underline{a} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}; \underline{\underline{M}} = \begin{pmatrix} -1 & 2 \\ 2 & -1 \end{pmatrix}$$

mit:

$$net =_{Df} \underline{\underline{M}} \cdot \underline{e}$$

ergibt sich folgender Output:

$$\underline{a} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = f\left(net - \underline{0}\right) = f\left(\underline{\underline{M}} \cdot \underline{e} - \underline{0}\right) = \begin{pmatrix} f(net_1 - 0) \\ f(net_2 - 0) \end{pmatrix} = \begin{pmatrix} f((-e_1 + 2e_2) - 0) \\ f((2e_1 - e_2) - 0) \end{pmatrix}$$

f binäre Transferfunktion

praktische Vorgehensweise

(1) Gegeben ist eine darzustellende (zu lernende) logische Funktion. Konkret liegen Eingabevektoren e_1, e_2, e_3, ... , e_n vor, denen Ausgabevektoren a_1, a_2, a_3, ... , a_n zugeordnet sind. Diese Funktion soll durch ein neuronales Netz dargestellt werden.

(2) Für das Netz ist eine Topologie zu wählen.

(3) Die Gewichte w_{ij} des Netzes sind so zu bestimmen, dass das Netz in der gewählten Topologie die vorgegebene Funktion darstellt. Zur Bestimmung der Gewichte sind *Lernverfahren* einzusetzen

(4) Nachdem die Gewichte gelernt wurden und damit das Netz zur Verfügung steht, ist es beliebig oft einsetzbar

Beispiele für heuristische Prinzipien/Verfahren

- Simulated Annealing
- genetische Algorithmen

grundsätzliche Vorgehensweise:

- ausgehend von einer zulässigen Lösung $\underline{x}$ werden zulässige Lösungen der „Nachbarschaft“ NB($\underline{x}$) daraufhin untersucht, ob sie eine Verbesserung darstellen
- eine „Nachbarschaftsrelation“ wird z.B. durch Abänderung oder Vertauschung von Elementen des Lösungsvektors x erreicht
- die Art der Nachbarschaftsbeziehung ist vom Problem und gewähltem Algorithmus abhängig. Sie ist ggf. stochastisch

Simulated Annealing:

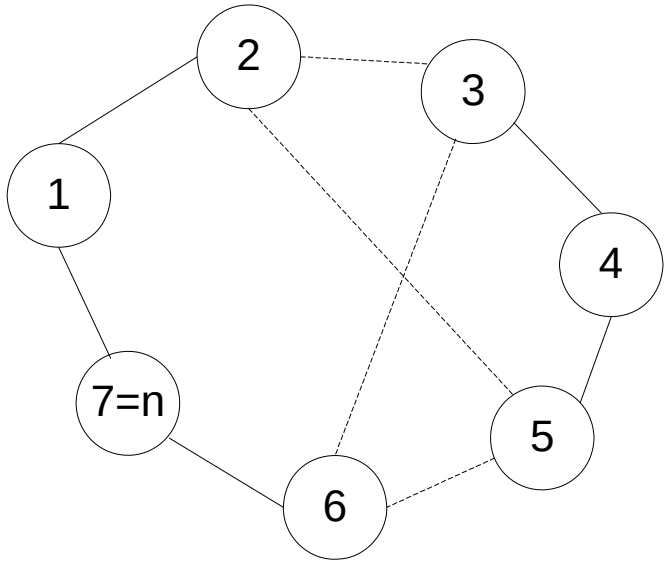
- Eine Nachbarlösung $\underline{x}' \in NB(\underline{x})$ wird nach einer bestimmten Vorschrift (ggf. auch zufällig) gewählt.
- Führt zu $\underline{x}'$ einer Verbesserung des Zielfunktionswertes, so wird $\underline{x}$ durch $\underline{x}'$ ersetzt.
- Führt $\underline{x}'$ einer Verschlechterung des Zielfunktionswertes, so wird $\underline{x}$ durch $\underline{x}'$ nur mit einer bestimmten Wahrscheinlichkeit ersetzt. Diese ist abhängig vom Ausmaß der Verschlechterung.
- Darüber hinaus wird diese Wahrscheinlichkeit über einen sogenannten „Temperatur“-Parameter so kontrolliert, dass sie mit fortschreitendem Lösungsprozess gegen Null geht.
- Eine vereinfachte Variante des Simulated Annealing ist Threshold Accepting. Hierbei wird jede Lösung akzeptiert, die den bisherigen Zielfunktionswert höchstens um einen vorzugebenden Wert Δ verschlechtert. Im Laufe des Verfahrens wird Δ dann sukzessive auf 0 reduziert.

Anwendung Simulated Annealing zur Lösung TSP

2-optimales-Verfahren

systematische Überprüfung aller Vertauschungsmöglichkeiten von jeweils zwei Kanten einer gegebenen Rundreise gegen zwei andere.

Beispiel:



algorithmische Darstellung(2-opt):

Voraussetzungen:
Kostenmatrix $\underline{C} = [c_{ij}]$ *; i, j = 1, ..., n*
zulässige Rundreise $[t_1, ..., t_n, t_{n+1}=t_1]$

Iteration μ ***(=1, 2, ...)***
for *i := 1* ***to*** *n-2* ***do***
begin
for *j := i + 2* ***to*** *n* ***do***
begin *berechne* $\Delta := c_{t_i t_j} + c_{t_{i+1} t_{j+1}} - c_{t_i t_{i+1}} - c_{t_j t_{j+1}}$
falls $\Delta < 0$*, bilde neue Rundreise*
$[t_1, ..., t_n, t_{n+1}=t_1] :=$
$[t_1, ..., t_i, t_j, t_{j-1}, \ldots, t_{i+1}, t_{j+1}, \ldots, t_n, t_{n+1}=t_1]$
und gehe zu Iteration $\mu+1$*.*
end
end

Ergebnis: *eine 2-optimale Rundreise*

2-opt mit Simulated Annealing (stochastisches Verfahren):

Voraussetzungen:
Kostenmatrix $\underline{C} = [c_{ij}]$ *; i, j = 1, ..., n*
zulässige Rundreise $[t_1, ..., t_n, t_{n+1}=t_1]$
vorzugebende reellwertige Parameter $\alpha > 0$; $0 < \beta < 1$;
it := Anzahl der bei unverändertem α durchzuführenden Iterationen

Iteration $\mu = 1, 2, ..., it$**:**

for i := 1 **to** n-2 **do**

begin

for j := i + 2 **to** n **do**

begin berechne $\Delta := c_{t_i t_j} + c_{t_{i+1} t_{j+1}} - c_{t_i t_{i+1}} - c_{t_j t_{j+1}}$

if $\Delta > 0$ **then**

begin berechne $P(\Delta,\alpha) := e^{-\frac{\Delta}{\alpha}}$ und eine in (0,1) gleichverteilte Zufallszahl γ;

if $\gamma > P(\Delta,\alpha)$ then goto M1

end

bilde neue Rundreise $r = [t_1, ..., t_n, t_{n+1}=t_1] :=$

$[t_1, ..., t_i, t_j, t_{j-1}, \ldots, t_{i+1}, t_{j+1}, \ldots, t_n, t_{n+1}=t_1]$;

falls c(r)<c (r*), setze r*:=r u. gehe zu Iteration μ+1.

M1: **end**

end

Setze $\alpha := \alpha \cdot \beta$ und beginne erneut mit Iteration $\mu = 1$.

Abbruchkriterium: Beende das Verfahren, wenn in den letzten it Iterationen keine neue Rundreise gebildet wurde.

Ergebnis: r* mit der Länge c(r*) ist die beste gefundene Rundreise

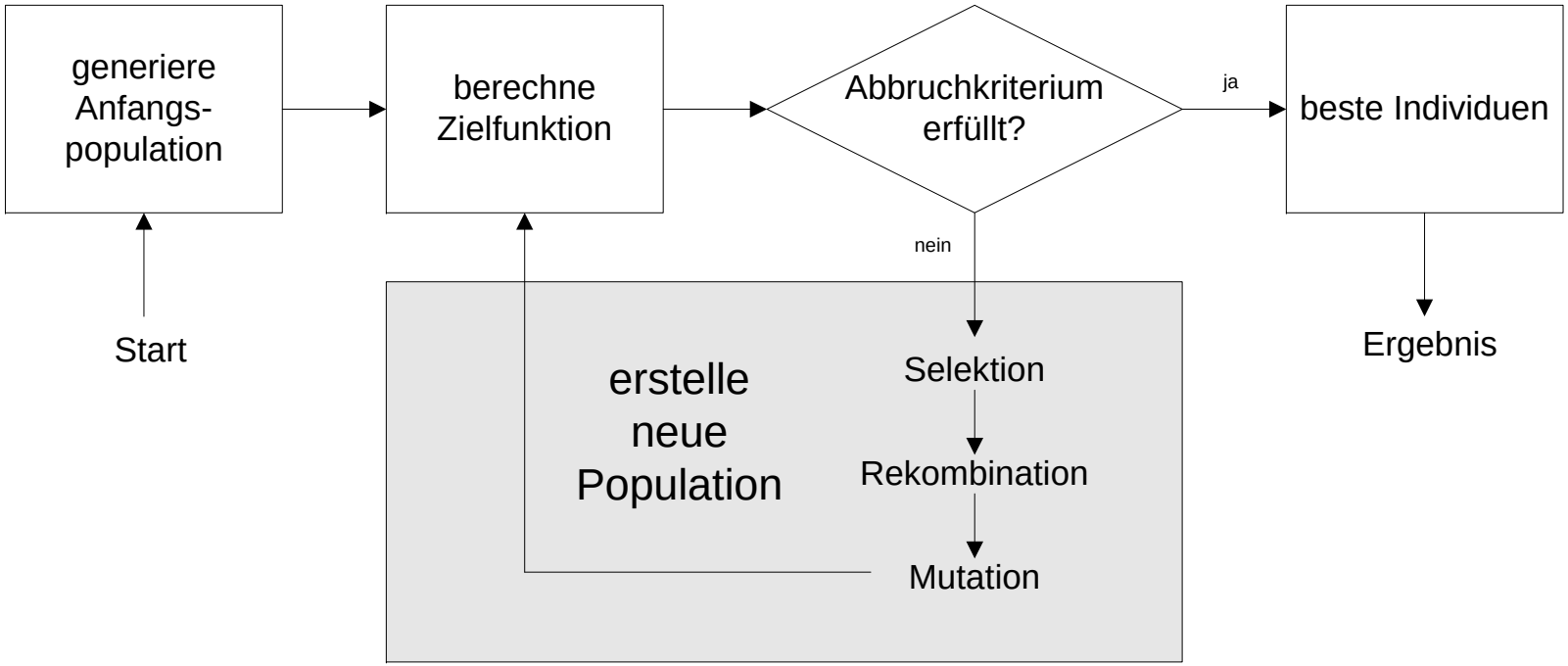
Bemerkungen:

- $P(\Delta,\alpha) = e^{-\frac{\Delta}{\alpha}}$ ist die Wahrscheinlichkeit dafür, dass eine Verschlechterung des Zielfunktionswertes in Kauf genommen wird .
- Es gilt: $\lim_{\Delta \to 0} P(\Delta,\alpha) = 1$ und $\lim_{\Delta \to \infty} P(\Delta,\alpha) = 0$
- Entscheidend für das numerische Verhalten von Simulated Annealing ist die Wahl der Parameter α, β und it .
- Wählt man α, gemessen an den Zielfunktionskoeffizienten groß, so ist anfangs die Wahrscheinlichkeit $P(\Delta,\alpha)$ für die Akzeptanz einer Verschlechterung der Lösung ebenfalls groß. Wählt man α dagegen klein, so werden Lösungsverschlechterungen mit kleiner Wahrscheinlichkeit in Kauf genommen. .
- Je kleiner β gewählt wird, umso schneller reduziert sich die Wahrscheinlichkeit für die Akzeptanz schlechter Lösungen .

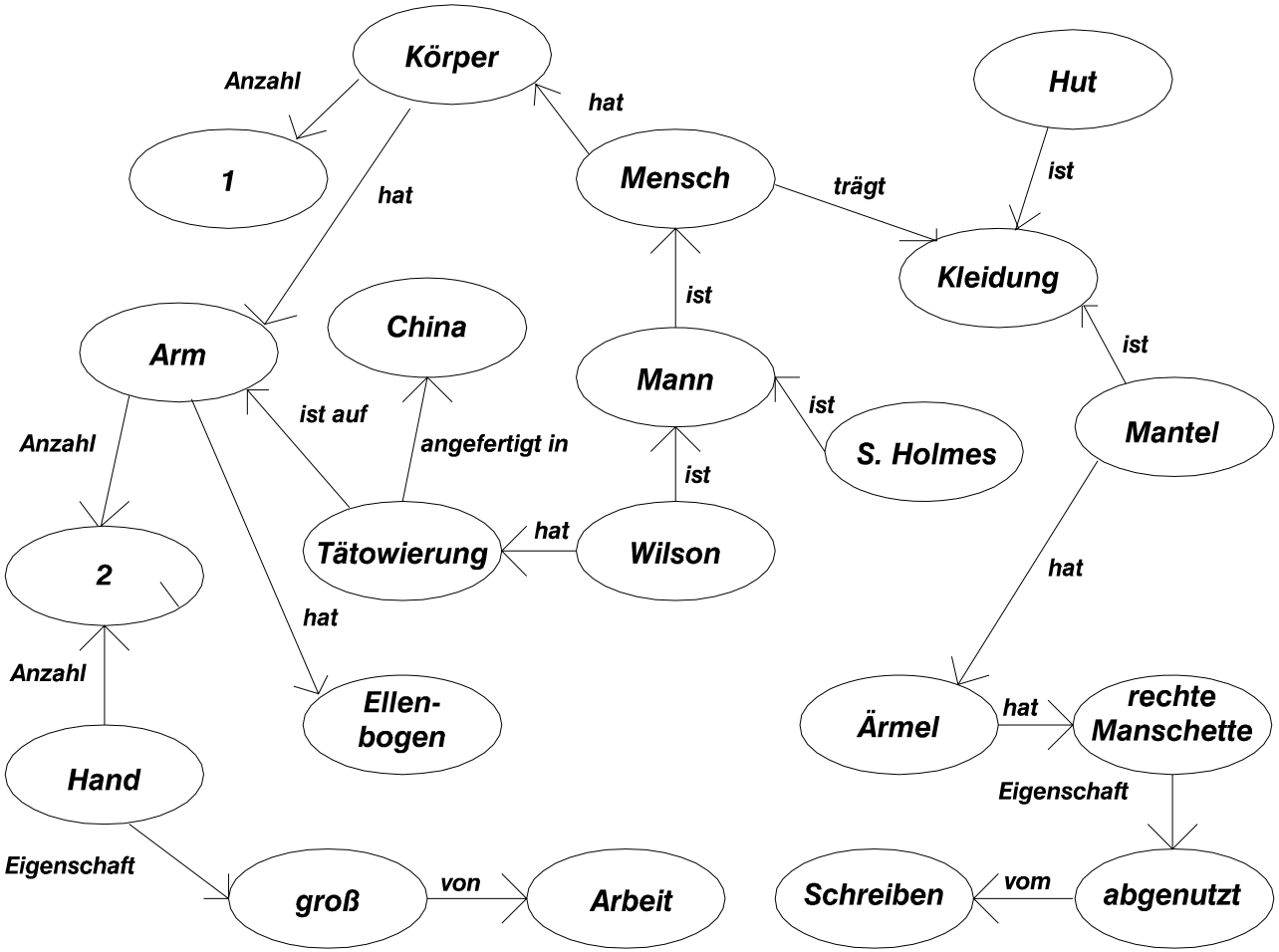
Genetische Algorithmen :

- Ausgangspunkt genetischer Algorithmen sind Mengen *(Populationen)* von Lösungen.
- Auf Elemente dieser Mengen (also Lösungen oder *Individuen*) werden sogenannte *genetische Operationen* an-gewandt, um neue Lösungen zu erzielen. Solche Opera-tionen sind *Kreuzung*, *Mutation* und *Selektion*.
- *Kreuzung* beinhaltet das Dekomponieren von Lösungen der *Elterngeneration* und anschließende wechselseitige Kom-binieren (Konfigurieren, Komponieren) zu neuen Lösungen einer *Nachkommengeneration*.
- Mit einer *Mutation* wird eine bestehende Lösung an einer oder mehreren Positionen abgeändert.
- *Selektion* bedeutet, aus einer Elterngeneration besonders *„fitte“ Individuen*, d.h. besonders gute Lösungen, auszu-wählen und in die nachkommende Generation (*Population*) aufzunehmen .

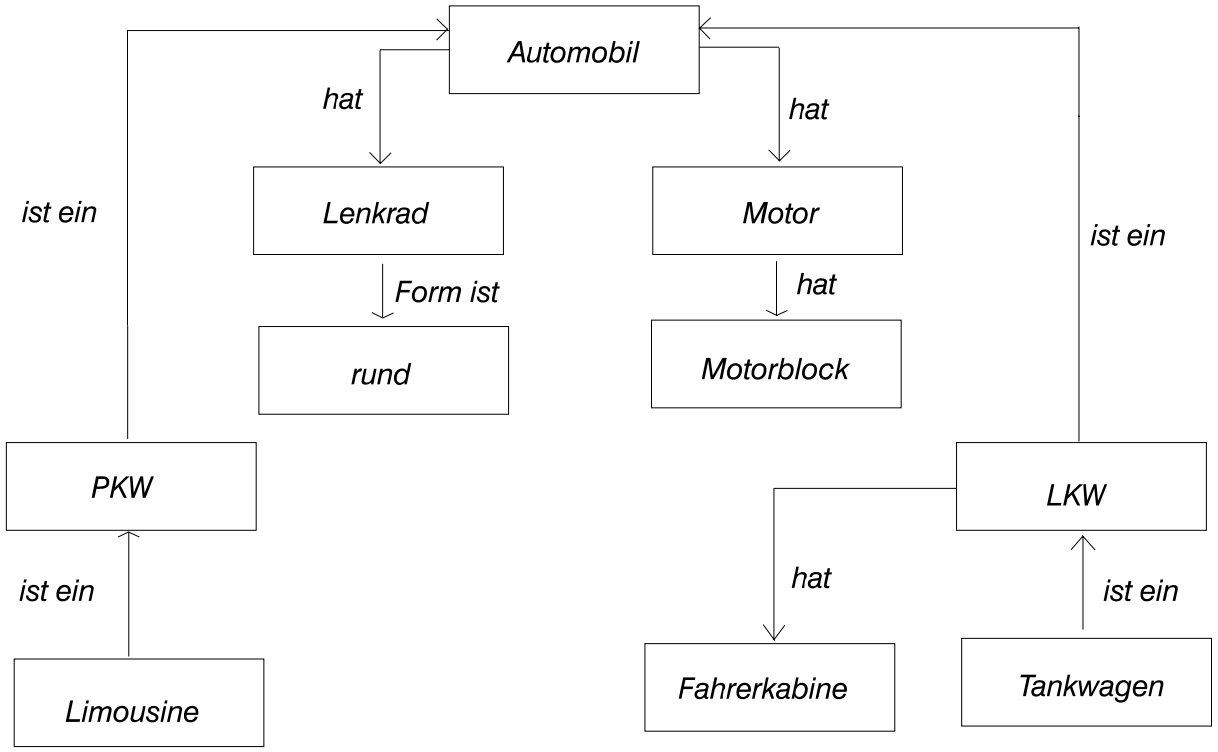
Genetische Algorithmen (schematischer Ablauf):



Semantisches Netz (Beispiel 1)



Semantisches Netz (Beispiel 2)



Frames – schematischer Aufbau

Objektname:

Supers: Liste von Objektnamen

Subs: Liste von Objektnamen

Slotname 1: Slotwert 1

Aspektname 1_1: Aspektwert 1_1

•
•
•

Aspektname 1_{m1}: Aspektwert 1_{m1}

Slotname n: Slotwert n

Aspektname n_1: Aspektwert n_1

•
•
•

Aspektname n_{mn}: Aspektwert n_{mn}

Frames – Beispiele

Zimmer:

Supers:	Haus		
Subs:	Wohnzimmer, Kinderzimmer, Arbeitszimmer		
Wände:	Mauer		
		Zahl:	default 4
		Restriktion:	> 2
		Material:	Beton, Ziegel, Gips
Decke:	1		
Fußboden:	1		
Ausstattung:	Möbel		

Frames – Beispiele (Fortsetzung)

Wohnzimmer:	
Supers:	Zimmer
Möbel:	Stuhl, Tisch, Sessel, Couch, Schrank
Technik:	TV, CD-Player, Video-Recorder

Arbeitszimmer:	
Supers:	Zimmer
Möbel:	Stuhl, Schreibtisch, Bücherschrank
Technik:	Computer, Telefon