LERNENDE SYSTEME

1. Neuronale Netze
2. Wissensbasierte Systeme

LITERATUR ZU NEURONALEN NETZEN

- R. Rojas, Theorie der neuronalen Netze. Eine systematische Einführung, Springer 1993, mittlerweise ca. 7. Auflage.
- A. Zell, Simulation Neuronaler Netze, Addison-Wesley 1994.
- einzelne Kapitel in vielen KI-Lehrbüchern wie z. B.
    - Russell, Norvig

BIOLOGISCHE NEURONALE NETZE

Bausteine:

- Neuronen (Nervenzellen)
- menschliches Gehirn enthält 10^{10} Neuronen
- Kortex (Großhirnrinde) hat hierarchische Struktur mit verschiedenen Neuronentypen

Neuronen haben folgende Bestandteile:

- *Dendriten* übernehmen Signale aus anderen Neuronen an Kontaktstellen, den sogenannten *Synapsen*.
- *Zellkörper* produziert die für die Arbeit des Neurons notwendigen Stoffe
- *Axon* zur Weitergabe des Ausgabesignals

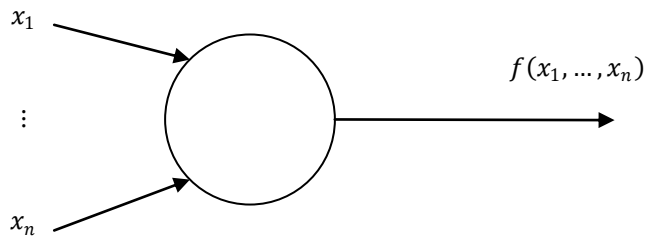
Informationsübertragung zwischen Neuronen

- durch elektrische Signale mit Hilfe von chemischen Transmitterstoffen (geringe Leitfähigkeit)
- Dies geschieht durch Ionendiffusion an semipermeable Membranen
- $NaCl \rightarrow Na^+ + Cl^-$ usw.
- Zellmembran verhält sich wie ein Kondensator
- Durchlassfähigkeit der Membranen ist variabel: Ionenkanäle können geöffnet und geschlossen werden.
- *Signale* sind Potentialschwankungen, die sich entlang des Axons fortbewegen
- *Schaltzeit* etwa 2,4 ms.

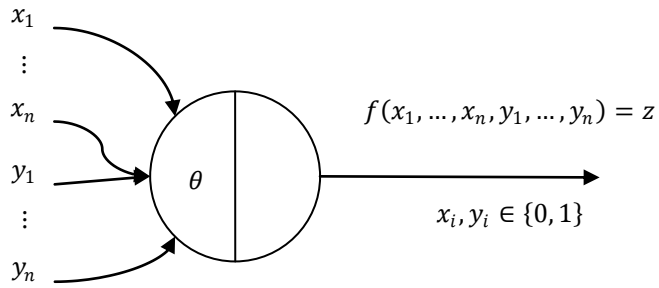
	Gehirn	Computer
Schaltzeit	2,4 ms	im Nanosekundenbereich
	ca. 10^2 Takte pro Sekunde	10^9 – 10^{10} Takte pro Sekunde
	ca. 10^{10} Neuronen	sehr viel weniger als 10^{10}
	Vernetzung	wenig Vernetzung
	Zufall	kein Zufall
	Redundanz	wenig Redundanz

KÜNSTLICHE NEURONALE NETZE

Das Modell von McCullock und Pitts:



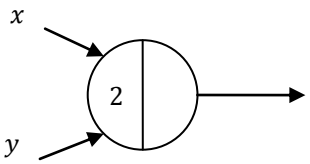
Unterscheidung zwischen hemmenden und aktivierenden Eingängen



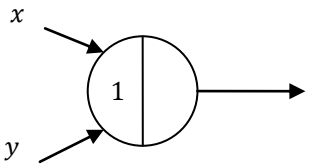
Ein *neuronales Netz* ist ein Netzwerk von Neuronen, d. h. ein gerichteter Graph, bei dem die Kanten die Neuronen sind.

Was leistet ein neuronales Netz?

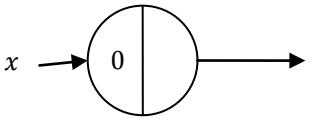
- Realisierung von $AND\,(x, y)$:



- Realisierung von $OR(x, y)$:



- Realisierung von $NOT\,(x)$:

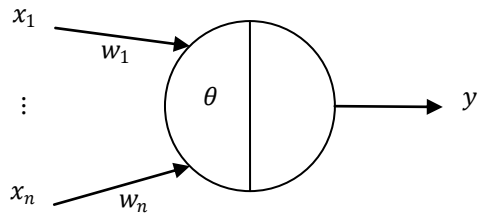


Neuronale Netze ohne Rückkopplung realisieren genau die Booleschen Funktionen, d. h. verhalten sich wie Schaltkreise.

Bei rekursiven Netzen (d. h. Ausgabe kann im folgenden Takt als Eingabe verwendet werden) erhält man genau die durch endliche Automaten berechenbaren Funktionen.

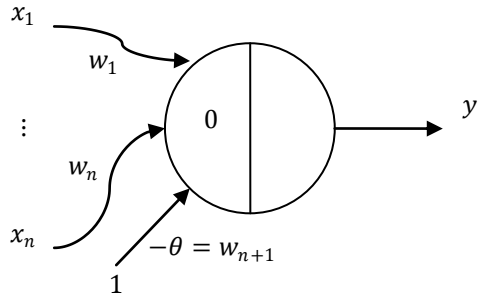
Bei neuronalen Netzen in der KI ist jedoch nicht so sehr die Berechnungsfähigkeit, sondern die Lernfähigkeit solcher Netze gefragt.

Lernvorgänge bei biologischen neuronalen Netzen laufen sehr stark über Verstärkung bzw. Abschwächung von Verbindungen. Dies führt zur Einführung von Kantengewichten.



$$y - 1 \Leftrightarrow \sum_{i=1}^{n} x_i w_i \geq \theta \Leftrightarrow \sum_{i=1}^{n} x_i w_i - \theta \geq 0$$

$\boxed{w_i \in \mathbb{R}}$

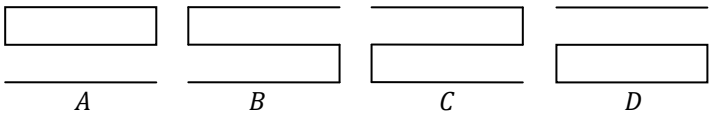


DAS PERZEPTRON VON ROSENBLATT 1958

- Minsky, Papert: Durchmesser-begrenztes Perzeptron kann nicht erkennen, ob die gegebene Abbildung zusammenhängend ist.

SATZ: Ein durchmesserbegrenztes Perzeptron kann nicht erkennen, ob die gegebene Abbildung zusammenhängend ist.

BEWEIS: Es seien A, B, C, D folgende Figuren:



Sind die Durchmesser der Prädikate durch eine Konstante d beschränkt, so kann d' so groß gewählt werden, dass kein Prädikat Punkte des linken und rechten Randes gleichzeitig abtastet (also $d' > d$).

Es sei $A(P_i)$ die Abtastregion des Prädikates P_i.

Einteilung der Prädikate in drei Gruppen:

$$L = \{P_i : A(P_i) \cap Li(X) \neq \emptyset\} \quad \text{für Eingabe } X$$
$$R = \{P_i : A(P_i) \cap Re(X) \neq \emptyset\} \quad \text{für Eingabe } X$$
$$M = \{P_i : A(P_i) \cap \big(Li(X) \cup Re(X)\big) = \emptyset\}$$

wobei $Li(X)$ dem linken Rand von X, $Re(X)$ dem rechten Rand von X entspricht und $Mi(X) = X \setminus \big(Li(X) \cup Re(X)\big)$.

Offenbar gilt:

$$Li(A) = Li(B), \quad Li(C) = Li(D),$$
$$Re(A) = Re(C), \quad Re(B) = Re(D)$$

und $Mi(X)$ ist gleich für alle $X \in \{A, B, C, D\}$.

Falls das Perzeptron richtig entscheidet, gilt:

$$\sum w_i P_i(A) < \theta, \quad \sum w_i P_i(D) < \theta,$$
$$\sum w_i P_i(B) \geq \theta, \quad \sum w_i P_i(C) \geq \theta.$$

Einteilung in Gruppen L, R, M:

$$\sum_L w_i P_i(A) + \sum_M w_i P_i(A) + \sum_R w_i P_i(A) < \theta$$
$$\sum_L w_i P_i(B) + \sum_M w_i P_i(B) + \sum_R w_i P_i(B) \geq \theta$$

Wegen $Li(A) = Li(B), Mi(A) = Mi(B)$ gilt

$$\sum_L w_i P_i(A) = \sum_L w_i P_i(B), \quad \sum_M w_i P_i(A) = \sum_M w_i P_i(B)$$

und somit $\sum_R w_i P_i(A) < \sum_R w_i P_i(B)$.

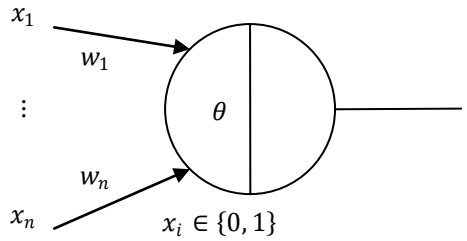
Also ist wegen $Re(A) = Re(C)$ und $Re(B) = Re(D)$ auch $\sum_R w_i P_i(C) < \sum_R w_i P_i(D)$ und wegen $Li(C) = Li(D), Mi(C) = Mi(D)$ auch $\sum w_i P_i(C) < \sum w_i P_i(D)$.

Widerspruch, da C zusammenhängend ist und D nicht.

■

REALISIERUNG LOGISCHER FUNKTIONEN MIT PERZEPTRONEN

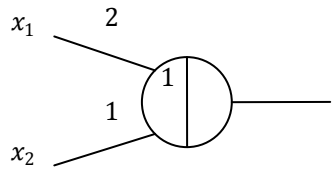
DEFINITION: Ein *einfaches Perzeptron* ist eine McCullock-Pitts-Zelle mit gewichteter Eingabe



Ausgabe: $\begin{cases} 1, & \sum_{i=1}^n x_i w_i \geq \theta \\ 0, & \sum_{i=1}^n x_i w_i < \theta \end{cases}$

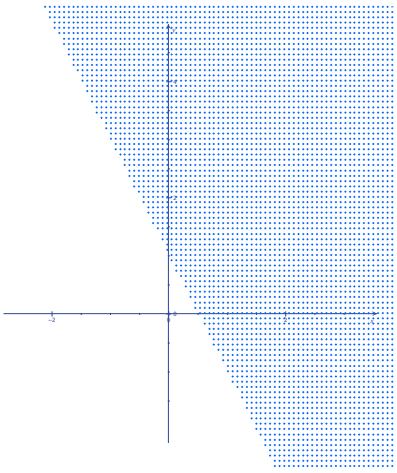
Die Ungleichung liefert eine Hyperebene, die den Raum in zwei Regionen $0/1$ linear teilt.

BEISPIEL:



Neuron gibt 1 aus, falls $2x_1 + 1x_2 \geq 1$, ansonsten 0.

Die Trenngerade ist $2x_1 + x_2 = 1$.



XOR-PROBLEM

Welche Booleschen Funktionen lassen sich mit einem einzigen Perzeptron realisieren?

BEISPIEL:

- OR:

x_1	x_2	$OR(x_1, x_2)$
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

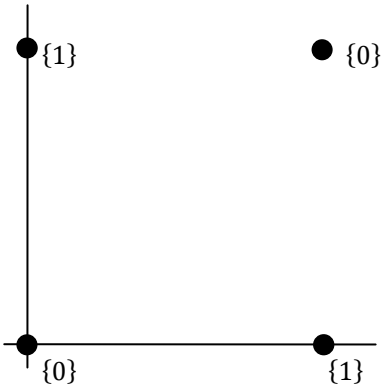
- AND:

x_1	x_2	$AND(x_1, x_2)$
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

- XOR:

x_1	x_2	$XOR(x_1, x_2)$
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

graphisch:



SATZ: XOR lässt sich nicht mit einem einzelnen Perzeptron realisieren.

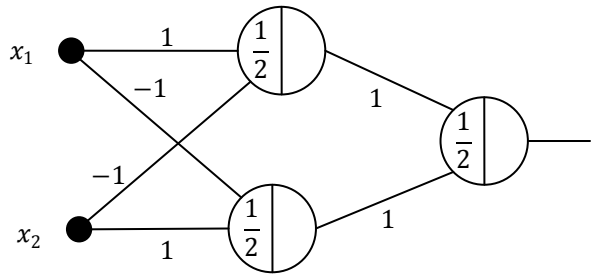
BEWEIS: Angenommen, es geht doch:

$$\begin{array}{lllr} (x_1, x_2) = & (0,0) & 0w_1 + 0w_2 < \theta & 0 < \theta \\ & (0,1) & 0w_1 + 1w_2 \geq \theta & w_2 \geq \theta \\ & (1,0) & 1w_1 + 0w_2 \geq \theta & w_1 \geq \theta \\ & (1,1) & 1w_1 + 1w_2 < \theta & w_1 + w_2 < \theta \end{array}$$

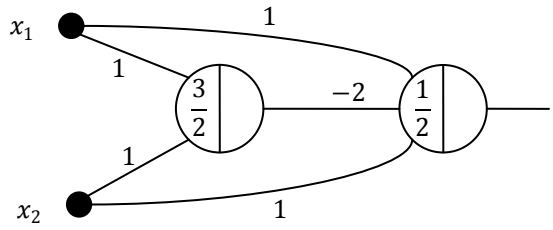
Widerspruch!

■

Eine Lösung für XOR mit mehrschichtigen Netzen:



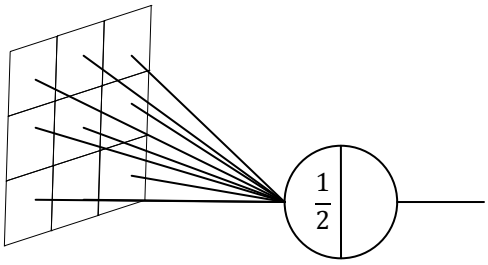
andere Variante:



KANTENERKENNUNG MIT PERZEPTRONEN

Erkennung von Helligkeitsunterschieden.

Folgendes Modell:



Gewichtung:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Faltungsoperator

Pixel weiß: 0, Pixel schwarz: 1.

- Fall 1: alle Pixel weiß: $y = 0$, weil $\sum x_{ij} w_{ij} = 0$.
- Fall 2: mittleres Pixel ist schwarz:
  - Fall 2.1: alle übrigen Pixel sind auch schwarz: $y = 0$, weil dito.
  - Fall 2.2: mindestens eins der übrigen Pixel ist nicht schwarz: $\boxed{y = 1}$.

„VERSCHALTUNG“ DER NETZHAUT

Das visuelle System ist der am besten verstandene Teil des menschlichen Gehirns.

- Enge Verbindung zwischen Augen und Nervensystem
- Netzhaut besteht aus Nervenzellen, die in Schichten „verschaltet“ sind und in denen bereits ein Teil der für das Sehen notwendigen Informationsverarbeitung erfolgt.

Auf der Netzhaut gibt es lichtempfindliche Fotorezeptoren. Jede Nervenleitung, die von den Augen ins Gehirn verläuft, fasst die Ergebnisse mehrerer Fotorezeptoren zusammen. Ähnlich dem Faltungsoperator übertragen die Nervenleitungen einen Impuls in Abhängigkeit von der Helligkeit eines Punktes relativ zu einer unmittelbaren Umgebung.

Horizontalzellen „berechnen“ die mittlere Helligkeit einer Region der Netzhaut.

DIE *SILICON RETINA* VON CARVER MEAD

Mead u. a. haben die ersten drei Schichten der Netzhautzellen modelliert:

- 1. Schicht: Fotorezeptoren
- 2. Schicht: Horizontalzellen
- 3. Schicht: Bipolarzellen

Horizontalzellen werden durch Widerstände simuliert. Jeder Fotorezeptor ist an seine sechs Nachbarrezeptoren angeschlossen (Bienenwabe) und produziert Potential zum Logarithmus der gemessenen Helligkeit. Neuronen in der *Silicon retina* feuern nur, wenn der Unterschied zwischen eigenem Potential und mittlerem Potential der Umgebung signifikant ist.

Mittleres Potential S von $S_i, i = 1, \dots, n$:

$$S = \frac{1}{n}\sum_{i=1}^{n} S_i = \frac{1}{n}\sum_{i=1}^{n} \log H_i = \frac{1}{n}\log \prod_{i=1}^{n} H_i = \log \sqrt[n]{\prod\nolimits_{i=1}^{n} H_i}$$

(H_i bezeichnet hier die Helligkeit).

Ein Neuron feuert, wenn $\log H_i - \log \sqrt[n]{\prod_{i=1}^{n} H_i} \geq y$ ist, d. h.

$$\boxed{\log \frac{H_i}{\sqrt[n]{H_1 \cdot H_2 \cdot \dots \cdot H_n}} \geq y}$$

DER PERZEPTRON-LERNALGORITHMUS

- entscheidende Eigenschaft neuronaler Netze: Lernfähigkeit (durch Gewichtsanpassung)
- einfachster Fall: Lernvorgänge bei einem einzelnen Perzeptron

Gegeben seien Punktmengen $A, B \subseteq \mathbb{R}^n$.

A und B heißen *linear trennbar,* wenn es eine Hyperebene (d. h. es gibt einen Gewichtsvektor $w = (w_1, \dots, w_n)^T$ und einen Schwellenwert θ) gibt, die A und B räumlich trennt:

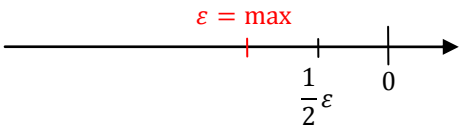
- für alle $x \in A$ gilt $x \cdot w \geq \theta$
- für alle $y \in B$ gilt $y \cdot w < \theta$

A und B heißen *absolut linear trennbar,* wenn die erste Ungleichung mit $>$ und dir zweite mit $<$ erfüllbar ist.

LEMMA: Linear trennbare *endliche* Mengen A und B sind sogar absolut linear trennbar.

BEWEIS: Weil A und B linear trennbar sind, existieren Gewichte $w_1, \dots, w_{n+1}$ mit $\sum_{i=1}^{n} w_i x_i \geq w_{n+1}$ für alle $x = (x_1, \dots, x_n)^T \in A$ und $\sum_{i=1}^{n} w_i y_i < w_{n+1}$ für alle $y = (y_1, \dots, y_n)^T \in B$.

Sei $\varepsilon := \max\{\underbrace{\sum_{i=1}^{n} w_i y_i - w_{n+1}}_{\text{negativ}} \mid y = (y_1, \dots, y_n)^T \in B\}$



Offenbar ist $\varepsilon < \frac{1}{2}\varepsilon < 0$.

Setzen $w' := w_{n+1} + \frac{1}{2}\varepsilon < w_{n+1}$

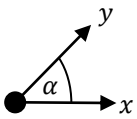
Nun gilt:

$$\sum_{i=1}^{n} w_i x_i > w' \quad \text{für alle} \quad x = (x_1, \dots, x_n)^T \in A \quad \text{und}$$

$$\sum_{i=1}^{n} w_i y_i < w' \quad \text{für alle} \quad y = (y_1, \dots, y_n)^T \in B.$$

■

Geometrische Deutung des Skalarproduktes $x \cdot y$ für Vektoren x, y mit $\|x\| = \|y\| = 1$:

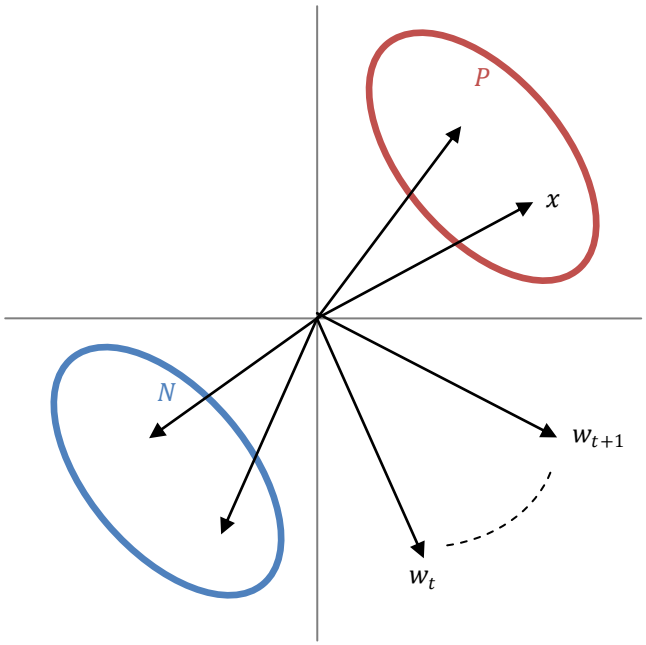


$$x \cdot y = \cos \alpha$$

DER PERZEPTRON-LERNALGORITHMUS:

Eingabe: Endliche Punktmengen P und N in $\mathbb{R}^n$. $P, N \subseteq \mathbb{R}^n$. $(P \cap N = \emptyset)$
Gesucht: Gewichtsvektor w, der P und N absolut linear trennt.

START:	Wähle zufällig einen Anfangsvektor w_0. Setze $t := 0$.
TESTEN:	Ein Punkt $x \in P \cup N$ wird „zufällig“ gewählt.
	Falls $x \in P$ und $w_t \cdot x > 0$, so gehe zu TESTEN.
	Falls $x \in P$ und $w_t \cdot x \leq 0$, so gehe zu ADDIEREN.
	Falls $x \in N$ und $w_t \cdot x < 0$, so gehe zu TESTEN.
	Falls $x \in N$ und $w_t \cdot x \leq 0$, so gehe zu SUBTRAHIEREN.
ADDIEREN:	Setze $w_{t+1} := w_t + x$; $t := t + 1$; gehe zu TESTEN.
SUBTRAHIEREN:	Setze $w_{t+1} := w_t - x$; $t := t + 1$; gehe zu TESTEN.



Im Erfolgsfall tritt „Stabilität“ ein.

SATZ: Falls P und N endlich und linear trennbar sind, so verändert der Algorithmus den Anfangsvektor w_0 nur in einer endlichen Zahl von Iterationen.

Wenn alle Eingabevektoren nacheinander in der Testphase verwendet werden, so wird nach endlicher Schrittzahl t ein Gewichtsvektor w_t gefunden, der P und N linear trennt.

BEWEIS: Folgende Vereinfachungen sind möglich:

1) P und N werden zu $P' = P \cup (-N)$
2) Normieren Vektoren in P'. Sind P und N linear trennbar, so existiert w zu P' mit Winkeln zwischen $90°$ und $-90°$.
3) Lösungsvektor w, der P und N linear trennt, kann auch zu $w^\star$ normiert werden.

Es soll w_{t+1} berechnet werden, weil w_t einen Vektor $p_i \in P'$ falsch klassifiziert hatte. Also ist $w_{t+1} = w_t + p_i$.

Für den Winkel τ zwischen w_{t+1} und $w^\star$ gilt:

$$\cos\tau = \frac{w^\star \cdot w_{t+1}}{\|w_{t+1}\|}.$$

Es gilt

$$w^\star \cdot w_{t+1} = w^\star(w_t + p_i) = w^\star \cdot w_t + \underbrace{w^\star \cdot p_i}_{\geq \delta ?} \geq w^\star \cdot w_t + \delta$$

$$\delta := \min\{w^\star \cdot p | p \in P'\}$$

Da $w^\star$ P und N absolut linear trennt, ist $\delta > 0$.

Wiederholte Anwendung liefert

$$\boxed{w^\star \cdot w_{t+1} \geq w^\star \cdot w_0 + (t+1) \cdot \delta}$$

Andererseits gilt:

$$\|w_{t+1}\|^2 = (w_t + p_i)(w_t + P_i) = \|w_t\|^2 + \underbrace{2w_t \cdot p_i}_{\leq 0} + \|p_i\|^2,$$

d. h.

$$\|w_{t+1}\|^2 \leq \|w_t\|^2 + \underbrace{\|p_i\|^2}_{1} \leq \|w_t\|^2 + 1$$

Einsetzen ergibt:

$$\cos\tau \geq \frac{w^\star \cdot w_0 + (t+1)\cdot\delta}{\sqrt{\|w_0\|^2 + (t+1)}} \quad \underset{t\to\infty}{\longrightarrow} \quad \infty$$

daher ist die Rundenzahl begrenzt.