— BERECHENBARKEITSTHEORIE —

Einführung

Berechenbarkeit

Welche Funktionen sind berechenbar?

$$
\begin{aligned}
g(n) &= n+1 \\
g(n) &= \sinh(n) \\
g(k) &= \text{die } k\text{-te Primzahl} \\
\mathbb{N} &= \{0,1,2,\ldots\} \\
\mathbb{N}^{\mathbb{N}} &= \{f \mid f : \mathbb{N} \mapsto \mathbb{N}\} \\
&= \text{Menge aller Funktionen, die } \textit{von} \text{ den natürlichen Zahlen total} \\
&\quad \textit{in} \text{ die natürlichen Zahlen abbilden} \\
\widetilde{\mathbb{F}}_1 &= \mathbb{N}^{\mathbb{N}} \\
\widetilde{\mathbb{F}}_2 &= \mathbb{N}^{\mathbb{N}\times\mathbb{N}} \\
\widetilde{\mathbb{F}}_i &= \mathbb{N}^{\underbrace{\mathbb{N}\times\mathbb{N}\times\cdots\times\mathbb{N}}_{i\text{-fach}}} \\
\widetilde{\mathbb{F}} &= \bigcup_{\substack{i\in\mathbb{N}\\ i\geq 1}} \widetilde{\mathbb{F}}_i \qquad \text{Menge aller totalen Funktionen von } \bigcup_{\substack{i\in\mathbb{N}\\ i\geq 1}} \underbrace{\mathbb{N}\times\cdots\times\mathbb{N}}_{i\text{-fach}} \text{ in } \mathbb{N} \\
\mathbb{F}_1 &= \{f \mid f : \mathbb{N} \to \mathbb{N}\} \quad \text{(kann auch nichttotal sein)} \\
\mathbb{F}_i &\quad \text{analog} \\
\mathbb{F} &= \bigcup_{\substack{i\in\mathbb{N}\\ i\geq 1}} \mathbb{F}_i \qquad \text{Menge aller Funktionen aus } \bigcup_{\substack{i\in\mathbb{N}\\ i\geq 1}} \underbrace{\mathbb{N}\times\cdots\times\mathbb{N}}_{i\text{-fach}} \text{ in } \mathbb{N}
\end{aligned}
$$

primitive Rekursion

Definition: Wir definieren folgende Grundfunktionen:

1. Nachfolgerfunktion:
$$\text{succ} : \mathbb{N} \to \mathbb{N} \text{ mit } \text{succ}(m) = m+1 \text{ für alle } m \in \mathbb{N}.$$
2. einstellige Funktion:
$$C_0^1 : \mathbb{N} \to \mathbb{N} \text{ mit } C_0^1(m) = 0 \text{ für alle } m \in \mathbb{N}.$$
3. Identitätsfunktionen (Projektionsfunktionen):
$$\text{id}_\ell^m : \mathbb{N} \to \mathbb{N} \text{ mit } \text{id}_\ell^m(n_1,\ldots,n_m) = n_\ell \text{ für alle } m \in \mathbb{N},\ 0 < \ell \leq m.$$

Definition: Wir definieren folgende Operationen:
1. Substitutionen:

$$
\begin{aligned}
\text{SUB}_i^m : \quad & \mathbb{F}^i \times (\mathbb{F}^m)^i \to \mathbb{F}^m \\
& \mathbb{F}^i \times \underbrace{\mathbb{F}^m \times \cdots \times \mathbb{F}^m}_{i\text{-fach}} \to \mathbb{F}^m
\end{aligned}
$$

Mache aus i Stück m-stelligen Funktionen und einer i-stelligen Funktion eine neue m-stellige Funktion. Dabei ist für $g \in \mathbb{F}^i$ und $h_1,\ldots,h_i \in \mathbb{F}^m$ die Funktion $\text{SUB}_i^m(g,h_1,\ldots,h_i)$ wie folgt definiert:

$$\text{SUB}_i^m(g,h_1,\ldots,h_i)(n_1,\ldots,n_m) = g\Big(h_1(n_1,\ldots,n_m),\ldots,h_i(n_1,\ldots,n_m)\Big)$$

2. Primitive Rekursion:

$$\text{PR}^{m+1} : \quad \mathbb{F}^m \times \mathbb{F}^{m+2} \to \mathbb{F}^{m+1}$$

Dabei ist die Funktion $f = \mathrm{PR}^{m+1}(g, h)$ für beliebige $g \in \mathbb{F}^m$ und $h \in \mathbb{F}^{m+2}$ und $n_1, \ldots, n_m, n \in \mathbb{N}$ so definiert:

$$f(n_1, \ldots, n_m, 0) = g(n_1, \ldots, n_m)$$
$$f(n_1, \ldots, n_m, n+1) = h(n_1, \ldots, n_m, n, f(n_1, \ldots, n_m, n))$$

Beispiele für Funktionen aus $\mathbb{P}\mathrm{r}$

a) Addieren

b) Multiplizieren

c) Potenzieren

d) modifizierter Vorgänger

e) modifizierte Differenz

f) Vorzeichen

g) absolute Differenz

h) Summen- und Produktbildung

i) Fallunterscheidung

j) beschränkter μ-Operator

k) ganzzahlige Division div mit Rest mod

l) Teilerprädikat

m) Anzahl der Teiler

n) Primzahlen

o) Anzahl der Primzahlen $\leq n$

p) n-te Primzahl

q) jede natürliche Zahl lässt sich eindeutig als Produkt von Primzahlpotenzen schreiben

Ackermann

zu Ackermann: Für jede primitiv rekursive Funktion $f : \mathbb{N}^m \to \mathbb{N}$ gibt es ein $r \in \mathbb{N}$, sodass $f(n_1, \ldots, n_m) \leq A(r, \max\{n_1, \ldots, n_m\})$ ist.
Folge $f_i : \mathbb{N} \to \mathbb{N}$ mit $f_0(n) = n + 1$, $f_{n+1}(0) = f_n(1)$ und $f_{n+1}(m + 1) = f_n(f_{n+1}(m))$ stellt eine primitve Rekursion dar: $f_{n+1} = PR(f_n(1), f_n(f_{n+1}))$

partielle Rekursion

.....

Automaten

Umgebung + Regelwerke
Input $\xrightarrow{\text{Regel}}$ Output

Berechenbarkeit

Was können Automaten berechnen?
Was leisten bestimmte Modelle?
Insbesondere: Welche Modelle können „dasselbe“?

Sprachen

Seien A, B Mengen von Buchstaben bzw. Wörtern. $A \cdot B := \{uv : u \in A, v \in B\}$, wobei uv das Wort ist, was entsteht, wenn man die Buchstaben von u und v (in dieser Reihenfolge) hintereinander schreibt.
Beispiel:

1. $u = \text{Schwes}, v = \text{ter} \rightarrow uv = \text{Schwester}$
2. $\{\text{apfel}, \text{baum}\} \cdot \{\text{kuchen}, \text{duda}\}$
   $= \{\text{apfelkuchen}, \text{baumkuchen}, \text{apfelduda}, \text{baumduda}\}$
3. $\{a, ab\} \cdot \{c, bc\} = \{ac, abc, abc, abbc\} = \{ac, abc, abbc\}$

$A^0 := \{\varepsilon\}$ (Einermenge „das leere Wort“)
$A^1 := A$
$A^n := A^{n-1} \cdot A$
$A^* := \bigcup_{n \in \mathbb{N}_0} A^n$

Ameisen

Ameise anschaulich einführen
Eine Ameise ist ein Paar $A = (F, \delta)$, wobei F eine endliche Menge ist (Farben) und $\delta : F \rightarrow F \times \{R, L\}$ die Überführungsfunktion.
Eine Ameise A kann aufgefasst werden als Wort über $\{R, L\}$: $A \in \{R, L\}^*$.
$\{(F, \delta)\} \leftrightarrow \{R, L\}^*$
$(\{f_1, f_2, \ldots, f_n\}, \{f_1 \rightarrow (f_2, r_1), f_2 \rightarrow (f_3, r_2), \ldots, f_n \rightarrow (f_1, r_n)\}) \leftrightarrow r_1 r_2 \ldots r_n$
Was kann man damit berechnen? (Invertieren einer 1-D Eingabe) Nicht viel :-(

Meady-Automaten

Meady-Automaten anschaulich einführen
Ein Meady-Automat M ist ein 5-Tupel $M = (\Sigma, \Gamma, Z, z_0, \delta)$ mit folgenden Komponenten:

1. Σ ist eine endliche Menge (Eingabealphabet)
2. Γ ist eine endliche Menge, $\Sigma \subseteq \Gamma$ (Arbeitsalphabet, Bandalphabet)
3. Z ist eine endliche Menge (Zustandsmenge)
4. $z_0 \in Z$ (Startzustand)
5. $\delta : \Gamma \times Z \rightarrow \Gamma \times Z$ (Überführungsfunktion)

Einfache Beispiele + Substitutionschiffre
Aber: Zwei Eingaben, die gleich Anfangen führen zu zwei Ausgaben, die ebenfalls gleich anfangen. Also z. B. keine Addition.

Turing-Maschinen

Einführung anschaulich in Turingmaschinen
Beispiel:

	z_1	z_2	z_3
M	(z_2, M, R)		(z_2, G, R)
I		(z_2, O, R)	
S		(z_2, L, R)	
T		(z_3, E, L)	
L		(z_2, L, R)	(z_3, L, L)
O		(z_2, O, R)	(z_3, O, L)
E		$(z_2, D, 0)$	

Definition: Eine (deterministische) Turingmaschine M ist ein 7-Tupel

$$M = (\Sigma, \Gamma, Z, z_0, \delta, Z_E, \square)$$

mit folgenden Komponenten:

1. Σ ist eine endliche Menge (Eingabealphabet)
2. Γ ist eine endliche Menge, $\Sigma \subseteq \Gamma$ (Arbeitsalphabet, Bandalphabet)
3. Z ist eine endliche Menge (Zustandsmenge)
4. $z_0 \in Z$ (Startzustand)
5. $\delta : \Gamma \times Z \to \Gamma \times Z \times \{R, L, 0\}$ (Überführungsfunktion)
6. $Z_E \subseteq Z$ (Endzustandsmenge)
7. $\square \in \Gamma$, $\square \notin \Sigma$ (leere Kiste)

(Achtung: Unterschied zu Meady klein!)
Bemerkung: nichtdeterministische Turingmaschine:
$\delta : \Gamma \times Z \to \mathcal{P}(\Gamma \times Z \times \{R, L, 0\})$ (Überführungsfunktion)
Satz dazu: bezüglich „Berechnungskraft“ sind nichtdeterministische und deterministische TMen äquivalent, d. h., zu jeder nichtdeterministischen TM N gibt es eine deterministische TM M mit $L(N) = L(M)$.

Definition: $K \in (\Gamma \setminus \{\square\})^* \cdot Z \cdot (\Gamma \setminus \{\square\})^*$ heißt Konfiguration einer Turingmaschine.
Startkonfiguration für Eingabe w: $z_0 w$
$K_1 \vdash_M K_2$ heißt, es gibt $a, b, c, b' \in \Gamma \setminus \{\square\}$, $\alpha, \beta \in \Gamma^*$ und $z, z' \in Z$ mit $K_1 = \alpha a z b c \beta$ und $K_2 \in \{\alpha z' a b' c \beta, \alpha a b' z' c \beta\}$.
Endkonfiguration: $\alpha z \beta$ mit $z \in Z_E$.
$\vdash_M^*$ definiert man als reflexive und transitive Hülle von $\vdash_M$.
Die von M akzeptierte Sprache $L(M)$ ist definiert als

$$L(M) := \{w \in \Sigma^* : \text{es gibt ein } z \in Z_E \text{ und } \alpha, \beta \in \Gamma^* \text{ mit } z_0 w \vdash_M^* \alpha z \beta\}$$

Die von M berechnete Funktion $F(M) : \Sigma^* \to \Gamma^*$ ist definiert als

$$F(M) := \{(w, \beta) \in \Sigma^* \times \Gamma^* : \text{es gibt ein } z \in Z_E \text{ und } \alpha \in \Gamma^* \text{ mit } z_0 w \vdash_M^* \alpha z \beta\}$$

oder auch

$$F(M)(w) := \beta \text{ mit } z_0 w \vdash_M^* \alpha z \beta \text{ für ein } z \in Z_E \text{ und } \alpha \in \Gamma^*$$

Beispiel: Eine Turing-Maschine für unäre Addition:
$M = (\{1, +\}, \{1, +, \square\}, \{z_0, z_1, \ldots, z_6, z_E\}, z_0, \delta, \{z_E\}, \square)$
mit δ gegeben durch folgende Vorschrift:

	z_0	z_1	z_2	z_3	z_4	z_5	z_6
1	$(1, z_0, R)$	$(1, z_1, R)$	$(1, z_3, L)$	$(1, z_3, L)$	$(1, z_4, L)$	$(\square, z_E, R)$	$(1, z_6, L)$
+	$(+, z_1, R)$		$(\square, z_6)$	$(1, z_4, L)$			
$\square$		$(\square, z_2, L)$			$(\square, z_5, R)$		$(\square, z_E, R)$

Plus mehr Beispiele!

Turing-Berechenbarkeit

Sei M eine Turing-Maschine, $M = (\Sigma, \Gamma, Z, z_0, \delta, Z_E, \square)$. Wir sagen, M hält bei der Eingabe $w \in \Sigma^*$ an genau dann, wenn es ein $z \in Z_E$ und $\alpha, \beta \in \Gamma^*$ gibt, sodass

$$z_0 w \vdash_M^* \alpha z \beta\,.$$

Definition: Eine (zahlentheoretische) Funktion $f \in \mathbb{F}_i$ heißt turingberechenbar genau dann, wenn gilt:
Es gibt eine Turingmaschine $M = (\{0, 1, 2, \ldots, 9, \#\}, \Gamma, Z, z_0, \delta, Z_E, \square)$ sodass für alle $(n_1, \ldots, n_i) \in \underbrace{\mathbb{N} \times \cdots \times \mathbb{N}}_{i\text{-fach}}$
folgendes gilt:

1. $(n_1, \ldots, n_i) \in D_f \Leftrightarrow M$ hält bei Eingabe von $n_1 \# \ldots \# n_i$ an.
2. Für alle $(n_1, \ldots, n_i) \in D_f$ gilt: $f(n_1, \ldots, n_i) = m \Leftrightarrow$ es gibt $z \in Z_E$ und ein $\alpha \in \Gamma^*$ sodass $z_0 w \vdash_M^* \alpha z m$.

Analoges gilt für Wortfunktionen.
Beispiel 1: succ ist Turing-berechenbar.
Beispiel 2: nd ist Turing-berechenbar.

Gödelisierung der TMn/Universelle Turing-Maschine

TM wird als Wort codiert.
Zustände: z_0 z_1 z_2
$\downarrow$ $\downarrow$ $\downarrow$
$|$ $||$ $|||$
Buchstaben: $0, 1, \square$
Kopfbewegungen: $+, -, \pm$
Trennsymbol: $\#$
z. B. $|0||\square + \#|1|||||0 - \# \ldots \widehat{=} (z_0, 0) \to (z_1, \square, R), (z_0, 1) \to (z_4, 0, L)$
Jede TM lässt sich als Wort über

$$\widehat{\Sigma} = \{|, 0, 1, \square, +, -, \pm, \#\}$$

aufschreiben.

$$M = (\Sigma, \Gamma, Z, z_0, \delta, Z_E, \square)$$

$$\underbrace{0\#1}_{\Sigma} \#\# \underbrace{0\#1\#\square}_{\Gamma} \#\# \underbrace{|\#||\#|||\# \ldots}_{Z} \#\# \ldots$$

Man kann einem Wort über $\widehat{\Sigma}$ ansehen, ob es eine TM kodiert.
$\to$ Liste aller TM'en
$\to$ Liste aller TM-berechenbaren Funktionen (Verweis auf später)

$\left(\widehat{\Sigma}\right)^*$: literweise Quark rausstreichen $\begin{cases} --- \\ --- \\ --- \\ --- \\ --- \\ --- \\ --- \\ M_0 \\ --- \\ --- \\ --- \\ M_1 \\ --- \\ \ldots \end{cases}$

$\to$ Liste aller TM'en hat schöne Eigenschaften:
$i \to M_i$ Das kann eine TM!
$N \to j$ mit $M_j = N$ Das kann eine TM!
$\rightsquigarrow$ Gödelisierung der TM!
Bemerkung: Umgang mit mehrstelligen Funktionen: Cantor-Nummerierung
$C_2 : \mathbb{N}^2 \to \mathbb{N}, C_2(x, y) = 2^x \cdot (2y + 1) - 1$ ist Bijektion von $\mathbb{N}^2 \to \mathbb{N}$.
$C_k : \mathbb{N}^k \to \mathbb{N}, C_k(n_1, \ldots, n_k) = C_2(n_1, C_2(n_2, \ldots, C_2(n_{k-1}, n_k) \ldots))$ ist Bijektion von $\mathbb{N}^k \to \mathbb{N}$.
Mit geeignetem δ wird $M_U = (\{a, b, c, d\}, \{a, b, c, d, \square\}, \{z_0, \ldots, z_7\}, z_0, \delta, Z_E, \square)$ zu einer universellen Turing-Maschine (wenige Buchstaben und Zustände reichen aus).

Churchsche These

Jede intuitiv berechenbare Funktion ist Turing-berechenbar.
Bemerkung: „vernünftige" Berechnungsmodelle im Sinne der Churchschen These (gleichmächtig zu TM):

- TM
- Mensch + Papier + Bleistift
- PASCAL (WHILE)
- RAM
- DNA

- Quantencomputer
- Billardball-Computing
- Game of Life
- Quax
- Eisenbahnen

„vernünftige“ Berechnungsmodelle im Sinne der Churchschen These (weniger mächtig zu TM):

- DFA
- NFA
- PDA
- NPDA
- LBA

„unvernünftige“ Berechnungsmodelle:

- Maschine, die für den $n+1$-ten Schritt die Hälfte der Zeit braucht, die der n-te Schritt benötigte (kann Halteproblem entscheiden)
- Orakelmaschinen

Quax/Fractran

Zettel austeilen.
Plus-Automat ($Q := [[3/5], [3, 5], 3]$) anschreiben, ohne zu sagen, was das ist und fragen: „Was macht der?“
Erstelle einen Automaten für

- modifizierte Differenz ($Q := [[1/15], [3, 5], 3]$) und
- Multiplikation ($Q := [[14/11, 429/70, 17/14, 5/13, 2/17], [5, 7], 3]$)

Life

Life einführen:

- Entwickelt von John Horton Conway
- Zellulärer Automat mit eine zweidimensionalen Raster und zwei Zuständen
- Übergangsregel:
  tot + 3 lebende Nachbarn → lebend
  lebend + 2 oder 3 lebende Nachbarn → lebend
  sonst → tot (Vereinsamung und Überbevölkerung)

Gleichgewicht zwischen Bevölkerung und Leere
Beispiel: $O \quad O \quad O \quad O$
Einfache Muster durchspielen (auch später Schüler selber experimentieren lassen) Beispiele:

- $O \quad . \quad . \quad O$
  $. \quad O \quad O \quad .$
- $O \quad O \quad O$
  $O \quad . \quad .$
  $O \quad . \quad .$
  $O \quad O \quad O$
- $O \quad . \quad O$
  $O \quad . \quad O$
  $O \quad . \quad O$
  $O \quad O \quad O$

```
  O O O
  O . .
• O O O
  . . O
  O O O
```

Stationäre Muster betrachten: naive Muster mit Periode eins
Beispiele (suche lassen):

- Block:
```
O O
O O
```

```
  . O .
• O . O
  . O .
```

- Wabe:
```
    . O O .
O   .   .   O
    . O O .
```

```
  . O O .
• O . . O
  . O . O
  . . O .
```

```
  . O O .
• O . . O
  O . . O
  . O O .
```

```
  . O O
• O . O
  . O .
```

- Boot:
```
    . O O
O   .   O
    O O .
```

- Schlange:
```
O   .   O O
O O .   O
```

```
  . . O O
  . . O .
• O . O .
  O O . .
```

aber schon mit zwei Blöcken eine unendliche Anzahl.
Insel: Umgeben von toten Zellen
Stabile Muster (still lifes): Periode 1 und nicht in zwei (oder mehr) stabile Inseln zerlegbar

```
. . . . . . O O
. . O . O . . O
. O . O O . O .
. O . . . . O O
O O . O O . . .
. . . O O . O O
O O . . . . O .
. O . O O . O .
O . . O . O . .
O O . . . . . .
```

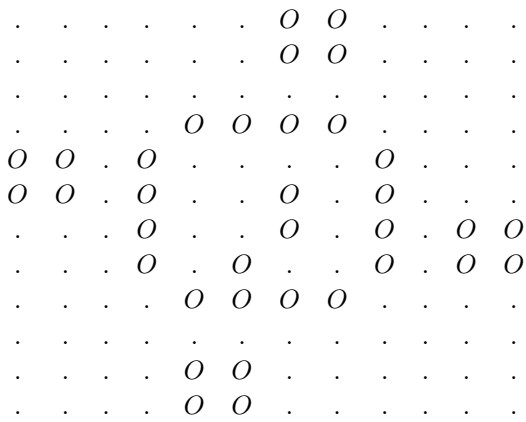
```
. . . . . . . . . O O .
. . . O O . O . . O .
. . . O . O O . O . . .
. . . . . . . . . O O .
. . . O . O O . . . O
. O O O . O O . O O .
O . . . . . . . O . .
. O O O . O O . O . . .
. . . O . O . O . . .
```

Wegen Vier-Farben-Satz kann man jedes Zerlegbare in vier Teile zerlegen

Oszilierende Muster betrachten Beispiele (suche lassen):

- Blinker: $O\ \ O\ \ O$ (p2)
- Unruhe (1): $\begin{matrix} O & O & O & . \\ . & O & O & O \end{matrix}$ (p2)
- Unruhe (2): $\begin{matrix} . & . & O & . \\ O & O & . & . \\ . & . & O & O \\ . & O & . & . \end{matrix}$ (p2)
- $\begin{matrix} O & O & . & . \\ O & . & . & . \\ . & . & . & O \\ . & . & O & O \end{matrix}$ (p2)
- $\begin{matrix} O & O & . & . & . \\ O & . & . & . & . \\ . & O & . & O & . \\ . & . & . & . & O \\ . & . & . & O & O \end{matrix}$ (p2)
- Pentadekathlon: $\begin{matrix} . & . & O & . & . & . & . & O & . & . \\ O & O & . & O & O & O & O & . & O & O \\ . & . & O & . & . & . & . & O & . & . \end{matrix}$ (p15)
- Oktagon: $\begin{matrix} . & . & . & O & O & . & . & . \\ . & . & O & . & . & O & . & . \\ . & O & . & . & . & . & O & . \\ O & . & . & . & . & . & . & O \\ O & . & . & . & . & . & . & O \\ . & O & . & . & . & . & O & . \\ . & . & O & . & . & O & . & . \\ . & . & . & O & O & . & . & . \end{matrix}$ (p5)
- Pulsar: $\begin{matrix} . & . & O & O & O & . & . & . & O & O & O & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . & . \\ O & . & . & . & . & O & . & O & . & . & . & . & O \\ O & . & . & . & . & O & . & O & . & . & . & . & O \\ O & . & . & . & . & O & . & O & . & . & . & . & O \\ . & . & O & O & O & . & . & . & O & O & O & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & O & O & O & . & . & . & O & O & O & . & . \end{matrix}$ (p3)

- Uhr:

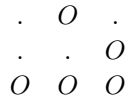


(p4)

Omniperiodizität unbekannt (19, 23, 31, 37, 38, 41, 43 und 53 fehlen; bei 34 und 51 keine Zelle die ständig oszilliert)

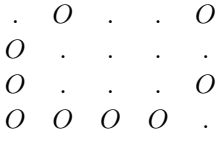
Raumschiffe betrachten: Lichtgeschwindigkeit $c = 1$

- Gleiter:



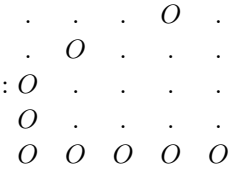
($c/4$ diag., p4)

- Segler (1) (LWSS):



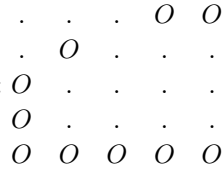
($c/2$ orth., p4)

- Segler (2) (MWSS):



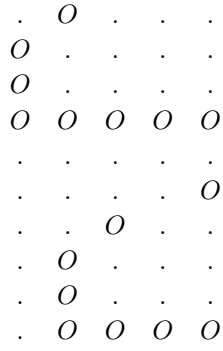
($c/2$ orth., p4)

- Segler (3) (HWSS):



($c/2$ orth., p4)

-



($c/2$ orth., p4)

Überlichtgeschwindikeit: Fast Forward Force Field erlaubt für LWSS 11 Schritte in 6 Generationen

Kanonen betrachten (Gosper glider gun, Nachweis für unbegrenztes Wachstum)

Sonstige nette Elemente

- Reflektoren,
- Absorber,
- breeder,
- Garten Eden,
- agar,

- boat-bit,
- sliding-block-memory,
- methuselah,
- kickback,
- Gleiter-Färbung,
- Dichte,
- fanout,
- Gleiter-Synthese,
- grow-by-one,
- hashlife,
- Herschel,
- inline inverter,
- relay,
- sawtooth,
- Fermat-primes

Turing-Maschine betrachten

Zugabe

Billiard-Berechenbarkeit

Adventure-Gaming

Eisenbahnen

Allgemeine Beispielliste

a) Addieren

b) Multiplizieren

c) Potenzieren

d) modifizierter Vorgänger

e) modifizierte Differenz

f) Vorzeichen

g) absolute Differenz

h) Summen- und Produktbildung

i) Fallunterscheidung

j) beschränkter μ-Operator

k) ganzzahlige Division div mit Rest mod

l) Teilerprädikat

m) Anzahl der Teiler

n) Primzahlen

o) Anzahl der Primzahlen $\leq n$

p) n-te Primzahl

q) jede natürliche Zahl lässt sich eindeutig als Produkt von Primzahlpotenzen schreiben

r) Ackermann

s) Caesar-Chiffre

t) Vigenere-Chiffre

nette Life-Muster

- Cheshire Katze:

$$\begin{array}{cccccc} . & O & . & . & O & . \\ . & O & O & O & O & . \\ O & . & . & . & . & O \\ O & . & O & O & . & O \\ O & . & . & . & . & O \\ . & O & O & O & O & . \end{array}$$

-