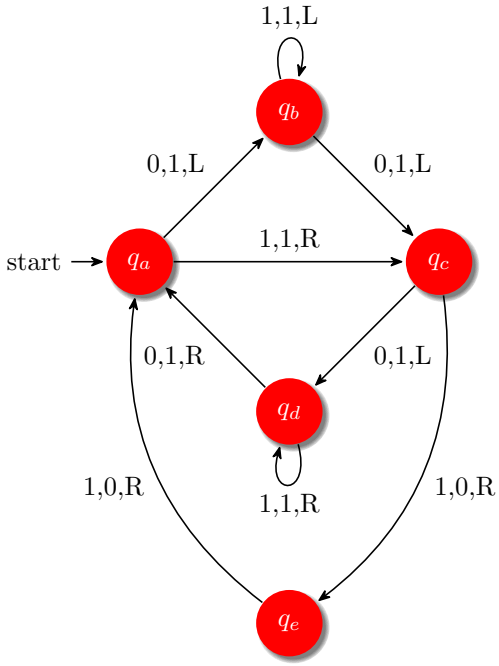
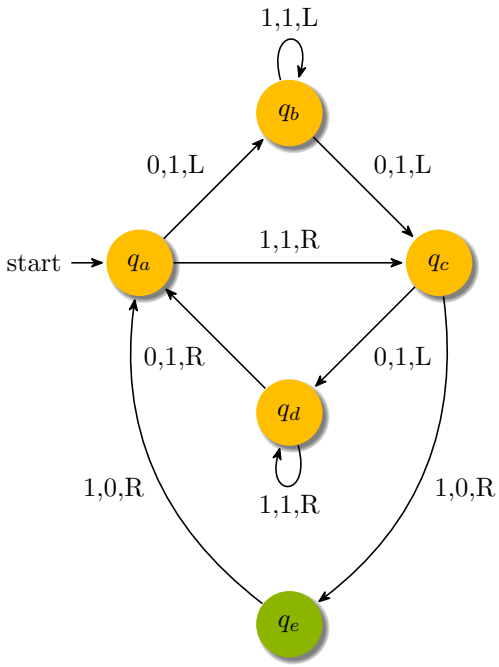
1 Turing machine

1.1 Original busy beaver



The current candidate for the busy beaver for five states. It is presumed that this Turing machine writes a maximum number of 1's before halting among all Turing machines with five states and the tape alphabet $\{0, 1\}$. Proving this conjecture is an open research problem.

1.2 Coloured busy beaver



2 Preliminary

2.1 Mathematische Symbole

https://oeis.org/wiki/List_of_LaTeX_mathematical_symbols

2.2 Menge

https://de.wikipedia.org/wiki/Menge_(Mathematik)

Eine Menge wird über Aufzählung - Liste - oder Konstuktor - Formel - definiert.

2.2.1 Aufzählung - Enumeration

https://de.wikipedia.org/wiki/Sinn_(Wahrnehmung)

S oder Sinne: Hören, Riechen, Schmecken, Sehen, Tasten, Temperatur, Schmerz, Gleichgewicht, Hunger, Durst, Harndrang, ...
$S = \{H, R, \dots\}$

oder

x: ist eine Variable von S.
|: ist eine Operation und bedeutet: x kann ein Element oder Wert aus S zugewiesen werden.
$S = \{x \mid x = H \vee x = R, \dots\}$

2.2.2 Konstruktor - Set builder

https://en.wikipedia.org/wiki/Set-builder_notation

Menge aller natürlichen Zahlen: $\mathbb{N} = \{x \in \mathbb{N} \mid x\}$

oder

$\mathbb{N} = \{n \mid n \in \mathbb{N}\}$

Menge der ungeraden Integer: $\{n \in \mathbb{Z} \mid (\exists k\ [k \in \mathbb{Z} \wedge n = 2k + 1])\}$

oder

$\{2n + 1 \mid n \in \mathbb{Z}\}$

und der inverse Konstruktor: $\{2i + 1 \mid i \in \mathbb{Z}\} = \{j \mid (j - 1)/2 \in \mathbb{Z}\}$

2.3 Mengenprodukt

https://de.wikipedia.org/wiki/Kartesisches_Produkt

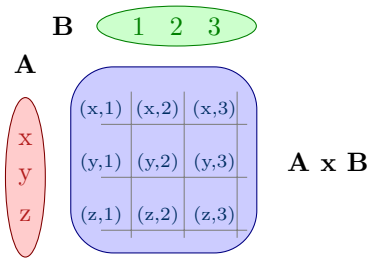
Das kartesiche Produkt $A \times B$ - ,,A kreuz B" - zweier Mengen A und B ist die Menge aller geordneten Paare - 2-Tupel oder Liste - (a, b), in denen a ein Element aus A und b ein Element aus B ist:

$A \times B := \{(a, b) \mid a \in A, b \in B\}$

2.3.1 Beispiel

Bild

https://tex.stackexchange.com/questions/232536/drawing-rectangle-with-tikzpicture
https://mirror.dogado.de/tex-archive/graphics/pgf/base/doc/pgfmanual.pdf,
p. 35



Satz

$A = \{x, y, z\}$
$B = \{1, 2, 3\}$
$A \times B = \{(x, 1), (x, 2), (x, 3), (y, 1), (y, 2), (y, 3), (z, 1), (z, 2), (z, 3)\}$

2.4 Natürliche Zahlen - $\mathbb{N}$

2.4.1 Sinn

https://de.wikipedia.org/wiki/Proposition_(Linguistik)
https://de.wikipedia.org/wiki/Valenz_(Linguistik)
https://de.wikipedia.org/wiki/%C3%9Cber_Sinn_und_Bedeutung
https://tex.stackexchange.com/questions/98052/umlauts-in-math-mode
https://tex.stackexchange.com/questions/161246/quotation-marks-around-operator-in-math-mod
https://de.wikipedia.org/wiki/Ellipse_(Sprache)
https://de.wikipedia.org/wiki/Kontext_(Sprachwissenschaft)
https://de.wikipedia.org/wiki/Sprachspiel
https://de.wikipedia.org/wiki/Abstraktion
https://golatex.de/viewtopic.php?t=1926

https://www.philosophie.hu-berlin.de/de/lehrbereiche/natur/forschung/dfg-wittgenstein-rams
https://plato.stanford.edu/entries/ramsey/

https://tex.stackexchange.com/questions/155181/coordinate-system-in-latex-with-tikz
https://ctan.net/macros/latex/contrib/tkz/tkz-euclide/doc/tkz-euclide.pdf
https://de.wikipedia.org/wiki/Euklidischer_Raum
https://de.wikipedia.org/wiki/Affine_Ebene

Proposition ist der Sinn eines Satzes - Frege:
Die Valenz oder Leerstellen von *zählen* ist 2.

Proposition:	$zählen(A, B)$
Positiver Satz:	A zählt B.
Negativer Satz:	A zählt nicht B.
Frage:	Zählt A B?
Befehl:	A, zähle B!
Ellipse:	Zähle! - Kontext, Sprachspiel, ...
Konditionalsatz:	Wenn A B zählt, ...

...

2.4.2 Funktion und Begriff

https://de.wikipedia.org/wiki/Funktion_und_Begriff
https://ia802700.us.archive.org/25/items/functionundbegr00freggoog/functionundbegr00freggo

Die ungeraden natürlichen Zahlen $2 * x + 1$ kann man als Funktion darstellen mit Definitionsmenge und Zielmenge:

Zuordnungsvorschrift mit D und Z: $f : \mathbb{N} \to \mathbb{N},\ x \mapsto 2 * x + 1$
oder

$$f : \begin{cases} \mathbb{N} \to \mathbb{N} \\ x \mapsto 2 * x + 1 \end{cases}$$

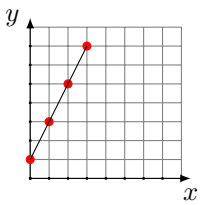
Das Argument x oder () ist eine unbestimmte Zahl der Funktion $2 * x + 1$ oder $2 * () + 1$.

Eine Funktion mit dem Ausdruck $2 * x + 1$ oder mit einem anderen Ausdruck $2 * () + 1$ für das Argument 1 hat den Wert 3.

In der affinen Ebene werden zwei Punkte mit eine Geraden verbunden und es gibt parallele Geraden.

Der zweidimensionale euklidischen Raum ist die euklidische Ebene oder affine Ebene.

Für beide Ausdrücke ergibt sich derselbe Werteverlauf, den wir in der euklidischen Ebene darstellen:



2.4.3 Tractatus

https://en.wikipedia.org/wiki/Nota_bene
https://people.umass.edu/klement/tlp/tlp.pdf
https:https://oeis.org/wiki/List_of_LaTeX_mathematical_symbols
https://tex.stackexchange.com/questions/50804/explicit-space-character
https://en.wikipedia.org/wiki/Logical_NOR
https://de.wikipedia.org/wiki/%C3%9Cberstrich
https://en.wikipedia.org/wiki/Tractatus_Logico-Philosophicus
https://tex.stackexchange.com/questions/177000/math-mode-accents

https://de.wikipedia.org/wiki/Paradoxon
https://de.wikipedia.org/wiki/Satz_vom_Widerspruch

https://de.wikipedia.org/wiki/Neun-Punkte-Problem

NOR

Das Ergebnis der NOR-Operation - $p\ NOR\ q$ oder $p \downarrow q \Leftrightarrow \neg(p \vee q)$ - ist *wahr*,

wenn beide Operanden $falsch$ sind.

Ein Satz $p \downarrow q$ ist $wahr$, wenn: $Weder\ p\ noch\ q\ ist\ wahr$.

W bedeutet $wahr$ und F bedeutet $falsch$.

p	**q**	**(WWWF)(p,q)**
W	W	F
W	F	F
F	W	F
F	F	W

Den OR- oder $\vee$-Operation definieren wir mit NOR:
$P = p \vee q \Leftrightarrow (p \downarrow q) \downarrow (p \downarrow q)$

p	**q**	**P**
W	W	W
W	F	W
F	W	W
F	F	F

Wahrheitsfunktion

TLP 2.0211, TLP 2.0212:
Die Welt hat eine Substanz und das sind die Gegenstände. Deswegen können wir ein Bild von der Welt erzeugen:

Ein Bild ist $wahr$ oder $falsch$.

TLP 2.21, 2.223:
Wenn ein Bild wahr ist, gibt es eine Übereinstimmung mit der Wirklichkeit.
Das finden wir heraus über einen Vergleich und das ist eine Abbildung.

TLP 3.01:
Ein Bild der Welt sind Gedanken:
Ein Gedanke ist wahr, wenn er Bild der Welt ist.

TLP 3.313:
Eine Satzvariable wird ersetzt durch Sätze mit einem *konstanten* und *variablen* Ausdruck.
N.b:
e := expression
P := complex proposition
$P = e^1 e^2 e^3$
$P = e^1 \_\ wohnt\ in \_ e^3$

TLP 3.313, 3.333:
Ein Satz ist eine Funktion und ein Argument ist ein Ausdruck.
Ein Satz oder eine Funktion kann kein Argument sein, um Paradoxien aufzulösen.

Eine Paradoxon ist eine widersprüchliche Aussage: ein Barbier rasiert alle Männer, die sich nicht selbst rasieren.

TLP 4.022:
Ein Gedanke ist immer sinnlich wahrnehmbar, auch wenn wir nichts tun oder

schlafen: wie wir daliegen, wie sich unser Gesichtsausdruck verändert, ...

Beim Sprechen, Schreiben und was wir sonst noch so tun wie Kochen oder Aufräumen ist ein Gedanke deutlicher sinnlich wahrnehmbar und wir zeigen uns gegenseitig, wie wir das herausfinden, was wahr ist.

TLP 5.5:
Wenn wir die NOR-Operation $(FFFW)(\xi, \ldots)$ mit Elementarsätzen ausführen, ist das Ergebnis eine Wahrheitsfunktion.

TLP 5.501:
ξ ist eine Satzvariable, der man z.B. drei Sätze P, Q, R zuweissen kann.
Der Überstrich $\bar{}$ von $\bar{\xi} = P, Q, R$ bedeutet, dass die Werte ungeordnet sind.

Die allgemeine Form eines Satzes - Proposition - ist eine Wahrheitsfunktion: $[\bar{p}, \bar{\xi}, N(\bar{\xi})]$

- $\bar{p}$ steht für alle Elemenatarsätze wie $p, q, \ldots$.
- $\bar{\xi}$ ist eine Satzvariable wie $P, Q, R, \ldots$, in der Elementarsätze mit logischen Operatoren verbunden werden.
- $N(\bar{\xi})$ ist die NOR-Operation mit Elementarsätzen.

TLP 6:
Die Wahrheitsfunktion für $N(P)$ ist $(FFFW)(p, q)$ und wird berechnet mit

p	**q**	**P**
W	W	F
W	F	F
F	W	F
F	F	W

Operation

TLP 6.01:
Ein Satzübergang ist eine Operation Ω ' $(\bar{\eta})$: $[\bar{\xi}, N(\bar{\xi})]$ ' $(\bar{\eta})$ $(= [\bar{\eta}, \bar{\xi}, N(\bar{\xi})])$

N.b:
$[\bar{\xi}, N(\bar{\xi})]$ ist ein 2-er Tupel oder Liste.
Mit der Operation Ω ' $(\bar{\eta})$ tauschen wir die atomaren Sätze der Umgangssprache $\bar{p}$ durch die der Mathematik $\bar{\eta}$ aus, um die natürlichen Zahlen zu definieren.

Koordinatensystem

`https://de.wikipedia.org/wiki/Koordinatensystem`

Gleichung

`https://de.wikipedia.org/wiki/Gleichung`

Eine Gleichung ist $T(l) = (m)T(r)$.
In einer Gleichung gibt es mindestens vier Aspekte: Koordinatensystem, Term, Gleichheitszeichen und Wahrheitsfunktion.
Sie wird aufgebaut aus einem Term $T(l)$ auf der linken Seite, dem Gleichheitszeichen $= (m)$ in der Mitte und dem Term $T(r)$ auf der rechten Seite.
Das Gleichheitszeichen bedeutet, dass $T(l)$ mit $T(r)$ verglichen wird.
Wenn $T(l)$ mit $T(r)$ übereinstimmt wie $p = p$, ist die Gleichung wahr, ansonsten

falsch wie $p = q$

Natürlichen Zahlen

TLP 6.02, 6.021:
Die Zahl ist der Exponent einer Operation.

Regeln: $x = \Omega^{0'}x$
$\Omega'\Omega^{\nu'}x = \Omega^{\nu+1'}x$
Reihe: $x, \Omega'x, \Omega'\Omega'x, \Omega'\Omega'\Omega'x, \ldots = \Omega^{0'}x, \Omega^{0+1'}x, \Omega^{0+1+1'}x, \Omega^{0+1+1+1'}x, \ldots$
Operation: $[x, \xi, \Omega'\xi] = [\Omega^{0'}x, \Omega^{\nu'}x, \Omega^{\nu+1'}x]$
Definition: $0 + 1 = 1$
$0 + 1 + 1 = 2$
$0 + 1 + +1 = 3$
$\vdots$

Gedanke und Satz oder Plan
`https://de.wikipedia.org/wiki/Reflexive_Relation`
`ttps://de.wikipedia.org/wiki/Selbstreferenzialit%C3%A4t`

2.4.4 Von Neumann

`https://de.wikipedia.org/wiki/Nat%C3%BCrliche_Zahl`

2.4.5 Ordinialzahl

`https://de.wikipedia.org/wiki/Ordinalzahl`

2.4.6 Folge - Tupel

`https://de.wikipedia.org/wiki/Folge_(Mathematik)`

2.5 Funktion

`https://en.wikipedia.org/wiki/Function_(mathematics)`
`https://de.wikipedia.org/wiki/Funktion_(Mathematik)`

Eine Funktion oder Abbildung ist eine Beziehung (Relation) zwischen zwei Mengen, die jedem Element der einen Menge (Funktionsargument: x-Wert) genau ein Element der anderen Menge (Funktionswert: y-Wert) zuordnet.

Eine Funktion f ordnet jedem Element x einer Definitionsmenge D genau ein Element y einer Zielmenge Z zu.

`https://mo.mathematik.uni-stuttgart.de/kurse/kurs44/seite25.html`

$$\mathbf{f} : \mathbf{D} \to \mathbf{Z},\ \mathbf{x} \mapsto \mathbf{y} \quad \text{oder} \quad \mathbf{f} : \begin{cases} \mathbf{D} \to \mathbf{Z} \\ \mathbf{x} \mapsto \mathbf{y} \end{cases}$$

2.5.1 Notation

`https://tex.stackexchange.com/questions/87867/do-i-need-a-specific-package-for-n`

Funktionsgleichung: $f(x) = x^2,\ x \in \mathbb{N}$
Zuordnungsvorschrift mit D: $x \mapsto x^2,\ x \in \mathbb{N}$
Zuordnungsvorschrift mit D und Z: $f : \mathbb{N} \to \mathbb{N},\ x \mapsto x^2$ oder $f : \begin{cases} \mathbb{N} \to \mathbb{N} \\ x \mapsto x^2 \end{cases}$

Wertetabelle:

x	1	2	3	...
y	1	4	9	...

Relation oder Aufzählung: $f = \{(1,1),(2,4),(3,9),\ldots\}$

2.5.2 Ausdruck

`https://de.wikipedia.org/wiki/Funktion_(Mathematik)`

- x wird abgebildet auf f von x
- f von x wird x eindeutig zugeordnet
- y gleich f von x
- y ist das Bild von x unter der Abbildung f

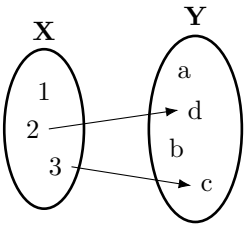
2.5.3 Partielle Funktion

`https://mirror.dogado.de/tex-archive/graphics/pgf/base/doc/pgfmanual.pdf`
`https://de.wikipedia.org/wiki/Partielle_Funktion`
`https://en.wikipedia.org/wiki/Partial_function`
`https://latexdraw.com/exploring-tikz-arrows/`

Eine partielle Funktion ordnet einem Element aus der Menge X ein Element aus der Menge Y zu.

Aufzählung

$X = \{1,2,3,\}$
$Y = \{a,b,c,d\}$
$f = \{(2,d),(3,c)\}$



Konstruktion

Ungerade natürliche Zahlen:
$f(x) = 2x+1,\ x \in \mathbb{N}$ oder
$f : \mathbb{N} \to \mathbb{N},\ x \mapsto 2x+1$

2.6 Algorithmus

`https://de.wikipedia.org/wiki/Algorithmus`

Ein Algorithmus ist eine Turingmaschine.

2.6.1 Beweis

Transitives Denken

Reflexives Denken

`https://de.wikipedia.org/wiki/Neun-Punkte-Problem`
`https://de.wikipedia.org/wiki/0,999%E2%80%A6`

Verbinde 9 Punkte mit 4 Linien.

N.b.: Verbinde x Punkte mit y Linien: optimale Weg.

oder $0,999\ldots = 1$.

2.6.2 Deterministischer Algorithmus

`https://de.wikipedia.org/wiki/Determinismus_(Algorithmus)`

Ein Algorithmus ist deterministisch, wenn er für die gleiche Eingabe immer die gleiche Ausgabe erzeugt und die gleichen Zustände durchlaufen werden.

2.7 Deterministische Turingmaschine

`https://de.wikipedia.org/wiki/Turingmaschine`
`https://en.wikipedia.org/wiki/Turing_machine`

Die Zielmenge der partiellen Funktion ersetze ich Γ durch Σ, um eine Endlosschleife, heutiges Verständnis, zu vermeiden.

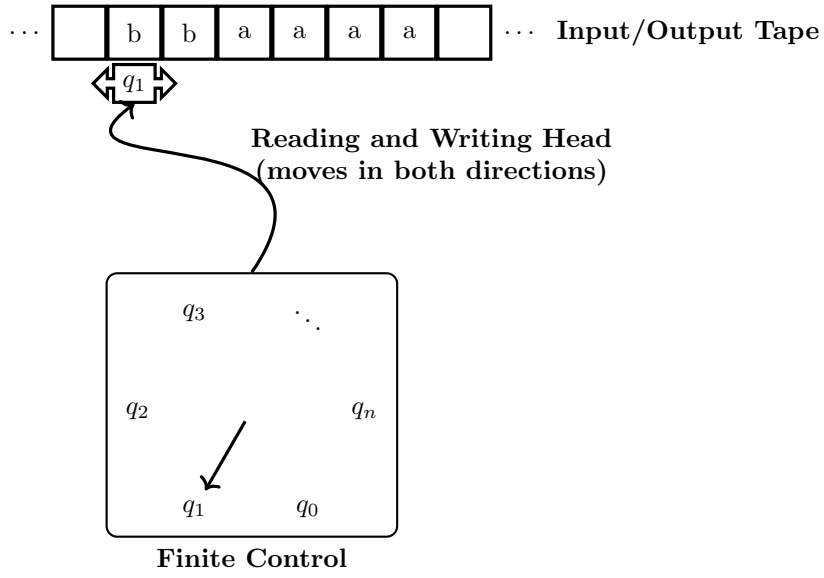
DTM ist eine deterministische Turingmaschine: $< Q, \Gamma, b, \Sigma, \delta, q_0, F >$:

- $\mathbf{Q}$ ist eine Zustandsmenge definiert über Aufzählung oder Generator, wie z.B. $2k + 1$, weil es sonst zu lange dauert, bis man alle aufgesagt hat, wie Zählen.
- $\mathbf{\Gamma}$ ist eine Menge von Speichersymbolen definiert über den I/O-Test.
- $\mathbf{b} \in \mathbf{\Gamma}$ ist ein Blank- oder Format-Symbol, das beim Parsen ignoriert wird.
- $\mathbf{\Sigma} \subseteq \mathbf{\Gamma} \setminus \{\mathbf{b}\}$ ist die Menge der Ausgabesymbole.
- $\delta : (\mathbf{Q} \setminus \mathbf{F}) \times \mathbf{\Gamma} \not\to \mathbf{Q} \times \mathbf{\Sigma} \times \{\mathbf{L}, \mathbf{R}\}$ ist eine partielle Funktion oder Übergangsfunktion:
  $L := Linksbewegung$ (left shift)
  $R := Rechtsbewegung$ (right shift)
  if $\delta \notin Q$ then $stuck(DTM)$
  Die Übergangsfunktion definiert den Folgezustand, der vom aktuellen Zustand abgeleitet wird. Man übersetzt das Eingabesymbol und sucht über eine Tabelle, ob der aktuelle Zustand die Eingabe erlaubt. Wenn das nicht zutrifft, stoppt die DTM und erzeugt eine Kopie von sich selbst - Core-Dump.

  Oder.

  Die Übergangsfunktion sagt mir, welchen Nachfolgezustand ich über den aktuellen Zustand erreichen kann.
- $\mathbf{q_0} \in \mathbf{Q}$ ist der Anfangszustand.
- $\mathbf{F} \subseteq \mathbf{Q}$ ist die Menge der Endzustände.

2.7.1 Architecture



Computer system
`https://www.tutorialspoint.com/Computer-System-Architecture`

2.8 Busy beaver

`https://en.wikipedia.org/wiki/Turing_machine`

Legend

P := Program counter
S := Start or initial state
C := Current state
I := Input
O := Output
A := ALU
N := Next state
M := Memory state

DTM

- $Q = \{A, B, C, H\}$
- $\Gamma = \{0, 1\}$
- $b = 0$
- $\Sigma = \{1\}$
- $q_0 = S = A$
- $F = \{H\}$

2.8.1 States

`http://www.cs.uni.edu/~barr/CS1000/ppt/Chapter09-ComputerOperation.pdf`
`https://en.wikipedia.org/wiki/Address_generation_unit`

A := Arithmetical logical unit including AGU - address generation unit - with the micro operations: *read*, *calculate*, *write*, *address*
R := Right address
L := Left address
S := Stop or stay on the same address

C	I	O	A	N
S	0	0	S	A
A	0	1	R	B
A	1	1	L	C
B	0	1	L	A
B	1	1	R	B
C	0	1	L	B
C	1	1	R	H
H	1	1	S	H

2.8.2 Trace

Log

P	C	I	O	A	N	M
0	S	0	0	S	A	$\{0, \ldots, 0, 0, 0, 0, \mathbf{0}, 0, 0, 0, \ldots, 0\}$
1	A	0	1	R	B	$\{0, \ldots, 0, 0, 0, 0, 1, \mathbf{0}, 0, 0, \ldots, 0\}$
2	B	0	1	L	A	$\{0, \ldots, 0, 0, 0, 0, \mathbf{1}, 1, 0, 0, \ldots, 0\}$
3	A	1	1	L	C	$\{0, \ldots, 0, 0, 0, \mathbf{0}, 1, 1, 0, 0, \ldots, 0\}$
4	C	0	1	L	B	$\{0, \ldots, 0, 0, \mathbf{0}, 1, 1, 1, 0, 0, \ldots, 0\}$
5	B	0	1	L	A	$\{0, \ldots, 0, \mathbf{0}, 1, 1, 1, 1, 0, 0, \ldots, 0\}$
6	A	0	1	R	B	$\{0, \ldots, 0, 1, \mathbf{1}, 1, 1, 1, 0, 0, \ldots, 0\}$
7	B	1	1	R	B	$\{0, \ldots, 0, 1, 1, \mathbf{1}, 1, 1, 0, 0, \ldots, 0\}$
8	B	1	1	R	B	$\{0, \ldots, 0, 1, 1, 1, \mathbf{1}, 1, 0, 0, \ldots, 0\}$
9	B	1	1	R	B	$\{0, \ldots, 0, 1, 1, 1, 1, \mathbf{1}, 0, 0, \ldots, 0\}$
10	B	1	1	R	B	$\{0, \ldots, 0, 1, 1, 1, 1, 1, \mathbf{0}, 0, \ldots, 0\}$
11	B	0	1	L	A	$\{0, \ldots, 0, 1, 1, 1, 1, \mathbf{1}, 0, 0, \ldots, 0\}$
12	A	1	1	L	C	$\{0, \ldots, 0, 1, 1, 1, \mathbf{1}, 1, 0, 0, \ldots, 0\}$
13	C	1	1	R	H	$\{0, \ldots, 0, 1, 1, \mathbf{1}, 1, 1, 0, 0, \ldots, 0\}$
14	H	1	1	S	H	$\{0, \ldots, 0, 1, 1, \mathbf{1}, 1, 1, 0, 0, \ldots, 0\}$

2.9 TLP

Beobachtung: In der Berufsschule verwendet man bunte Begriffe, die man 2-dimensionalen mit z.T. bidrektionen Pfeilen verbindet und zwar unspezifiert.

Idee: Das ist ein Algorithmus und ein Automat. Als Modell verwende ich deterministische Turing Maschine, damit es nicht zu kompliziert wird.

Spiel: Jeder Mensch, Mathematiker oder Philosoph spielt dauernd mit irgendwelchen Zeichen:

xxx: Um alles zu vereinfachen, beschränke ich mich auf eine deterministische Turing Maschine - DTM - oder einen Computer, der das macht, was - tbd. - er laut Beschreibung kann und zwar sofort - tbd. - ein Ergebnis sofort liefert, das mit der Erwartiung übereinstimmt.

xxx: Ziemlich frei habe ich alles umdefininiert:

DTM = (C, I, R, B, F, T, S)

C := States of the computer

I := I/O characters (abstract)

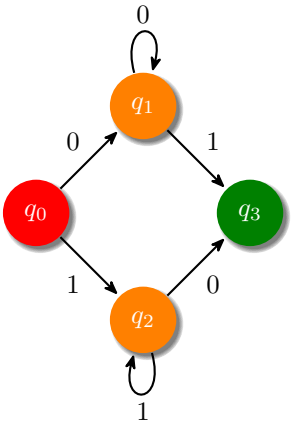
M := Memory

B := Power on or boot phase

F := Functions

T := Timeout actions like powersaveing

S := Power off or shutdown



3 Derivation

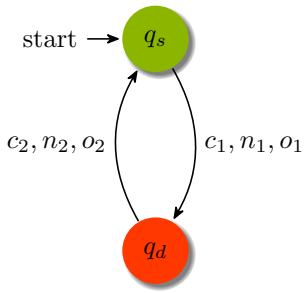
3.1 Legend

Applegreen: initial state
Amber: any state
Coquelicot: final state

3.2 Subject - Deadline

1. **Subject - s:** Die Tractatus Dimensionen oder wie alle in jedem Kosmos denken
2. **Deadline - d:** 23.12.2022

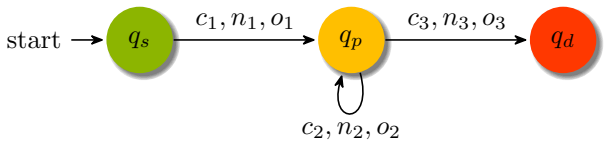
```
Draft:
s <-> d
```



3.3 Proposition

1. **Subject - s:** Die Tractatus Dimensionen oder wie alle in jedem Kosmos denken
2. **Proposition - p:** Literate Programming / LuaTeX
3. **Deadline - p:** 23.12.2022

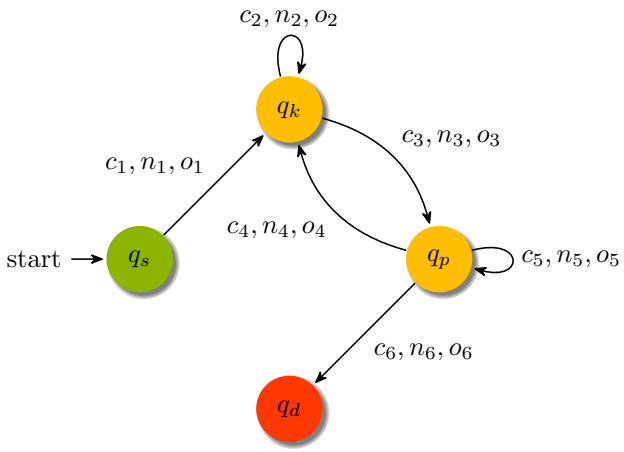
```
Draft:
s -> l(p) -> d
```



3.4 Knowledge

1. **Subject - s:** Die Tractatus Dimensionen oder wie alle in jedem Kosmos denken
2. **Knowledge - k:** Rezitation des Tractatus
3. **Proposition - p:** Literate Programming / LuaTeX
4. **Deadline - p:** 23.12.2022

```
Draft:
     -> l(k) <-
    |          |
    s          v
             l(p)
              |
          d <-
```




```
Contents

1. Subject - s:     Die Tractatus Dimensionen
                    oder wie alle im Weltall denken
2. Knowledge - k:   Rezitation des Tractatus
3. Proposition - p: Literate Programming / LuaTeX
4. Deadline - d:    23.12.2022
5. Hypothesis - h:  i. Begriffe sind Koordinatenachsen

Legend:
Eine Landkarte versteht man über eine
Übersetzung.
https://de.wikipedia.org/wiki/Legende_(Karte)

Colors:
Wenn man die Ampelfarben nicht versteht,
kann man ganz schnell sterben.

htps://de.wikipedia.org/wiki/Ampel

1. s <-> d

2. s -> l(p) -> d

3.   -> l(k) <-
    |          |
    s          v
             l(p)
              |
          d <-

4.   -> l(k) <-
    |          |
    s          v
             l(p)
```

```
    |
d <-
```