Karlsruher Institut für Technologie

Theoretische Grundlagen der Informatik Übung

4. Übungstermin · 21. November 2019

Jonas Sauer

INSTITUT FÜR THEORETISCHE INFORMATIK · LEHRSTUHL ALGORITHMIK

Gliederung

Turing-Maschinen und Berechenbarkeit

- Universelle Turing-Maschinen
- Entscheidbarkeit und Semi-Entscheidbarkeit
- Satz von Rice
- Post'sches Korrespondenzproblem

Komplexitätstheorie

- Sprachen, Probleme und Zeitkomplexität
- Klasse NP
- Über die Klassen P und NP hinaus

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Karlsruher Institut für Technologie

Wiederholung

(Semi-)Entscheidbarkeit

Definition: Eine Sprache $L \subseteq \Sigma^*$ heißt **rekursiv** oder **entscheidbar**, wenn es eine Turing-Maschine gibt, die auf allen Eingaben stoppt und eine Eingabe w genau dann akzeptiert, wenn $w \in L$ gilt.

$\Rightarrow \mathcal{M}$ entscheidet L.

Definition: Eine Sprache $L \subseteq \Sigma^*$ heißt **rekursiv-aufzählbar** oder **semi-entscheidbar**, wenn es eine Turing-Maschine gibt, die genau die Eingaben w akzeptiert, für die $w \in L$.

Das Verhalten der Turing-Maschine für Eingaben $w \notin L$ ist damit nicht genau definiert. D.h., die Turing-Maschine stoppt entweder nicht in einem Endzustand oder aber stoppt gar nicht.

$\Rightarrow \mathcal{M}$ akzeptiert L.

Satz: Eine Sprache L ist genau dann entscheidbar, wenn L und deren Komplement L^c semi-entscheidbar sind.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Bisher:

- Bislang beschriebene DTM sind für spezielle Aufgaben

Intuitiver Wunsch:

- Eine Art programmierbarer Rechner, der als Eingabe ein Programm und die Eingabe für dieses Programm bekommt

Beschreibung einer TM

- $\mathcal{M} := (Q, \Sigma, \Gamma, \delta, s, F)$
- Gödelnummer $\langle \mathcal{M} \rangle$ von $\mathcal{M}$ ist definiert durch folgende Kodierungsvorschrift:
  1. Kodiere $\delta(q_i, a_j) = (q_r, a_s, d_t)$ durch $0^i 1 0^j 1 0^r 1 0^s 1 0^t$, mit $d_t \in \{d_1, d_2, d_3\}$, d_1 für L, d_2 für R und d_3 für N
  2. Turing-Maschine wird kodiert durch: $111\text{code}_1 11\text{code}_2 11 \ldots 11\text{code}_z 111$, mit code_i für $i = 1, \ldots, z$ entspricht allen Funktionswerten von δ

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

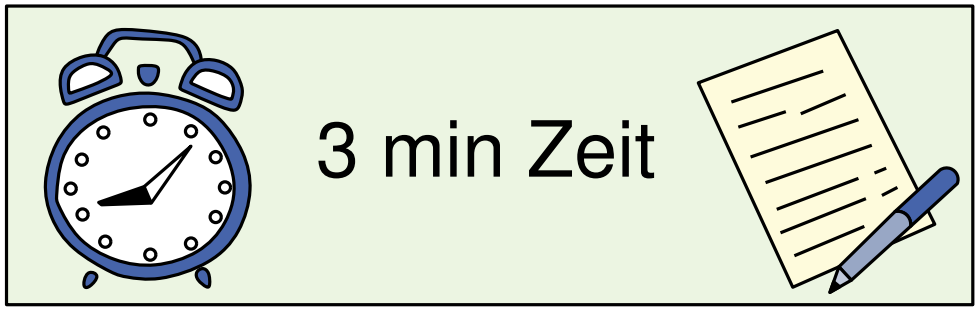
Universelle Turing-Maschine

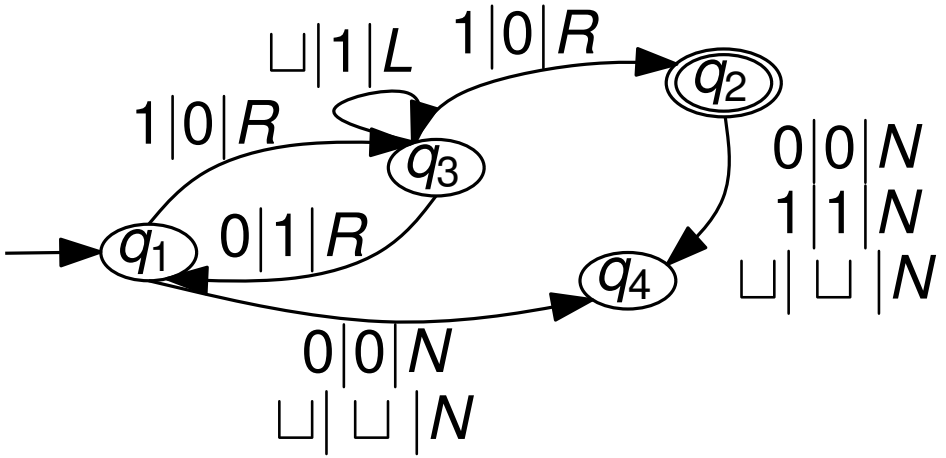
Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ:

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ:





Gödelnummer $\langle \mathcal{M} \rangle$ von $\mathcal{M}$ ist definiert durch folgende Kodierungsvorschrift:

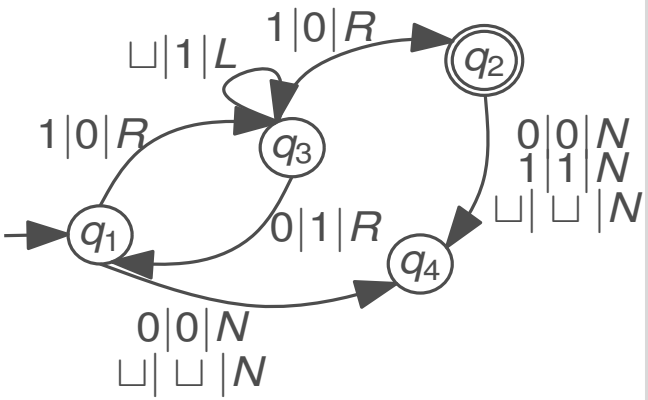
1. Kodiere $\delta(q_i, a_j) = (q_r, a_s, d_t)$ durch $0^i 1 0^j 1 0^r 1 0^s 1 0^t$,
   mit $d_t \in \{d_1, d_2, d_3\}$, d_1 für L, d_2 für R und d_3 für N
2. Turing-Maschine wird kodiert durch:
   $111\text{code}_1 11\text{code}_2 11 \ldots 11\text{code}_z 111$,
   mit code_i für $i = 1, \ldots, z$ entspricht allen Funktionswerten von δ

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

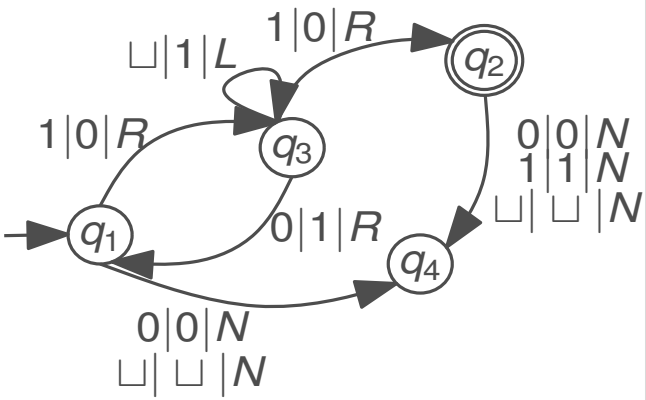
Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

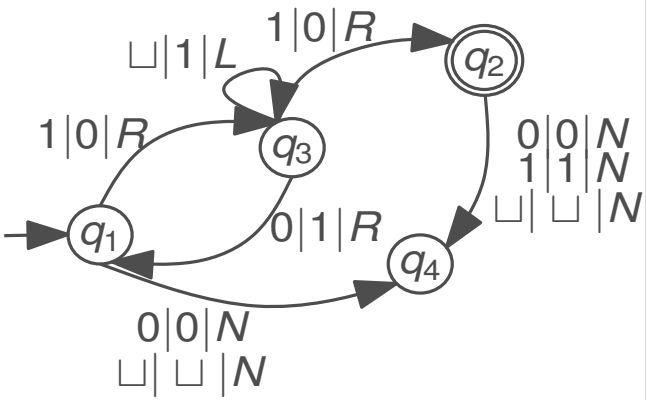
Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$$\left.\begin{array}{l} a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup \\ d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N \end{array}\right\} \quad \delta(q_i, a_j) = (q_k, a_\ell, d_m)$$

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

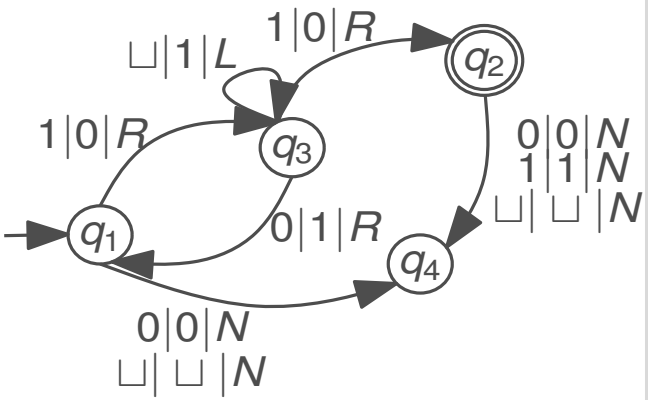
Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$$\left.\begin{array}{l} a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup \\ d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N \end{array}\right\} \quad \delta(q_i, a_j) = (q_k, a_\ell, d_m) \quad \Big\} \quad 0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m$$

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

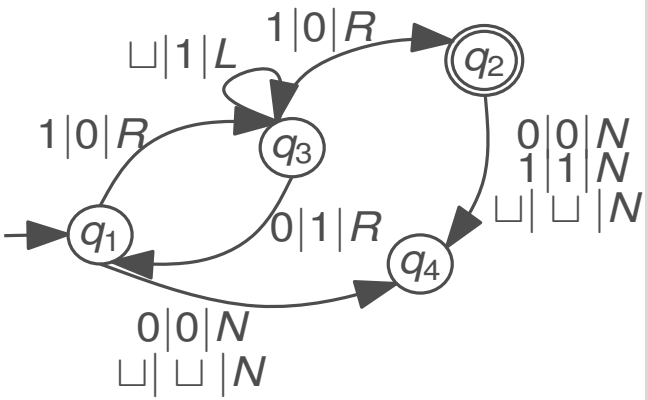
Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$$\left.\begin{array}{l} a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup \\ d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N \end{array}\right\} \quad \delta(q_i, a_j) = (q_k, a_\ell, d_m) \quad \left\} \quad 0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m \right.$$

#	Eintrag	Kodierung

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

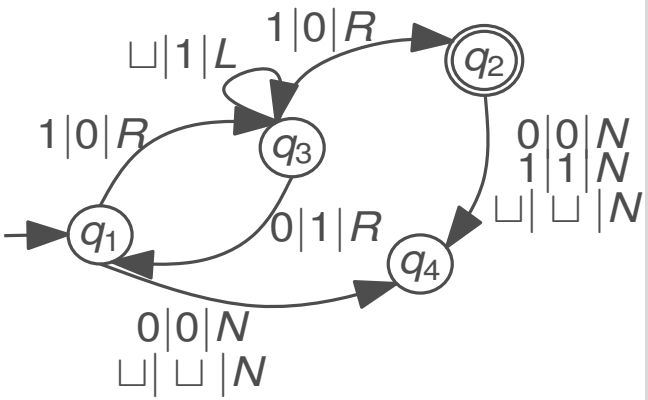
Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$$\left.\begin{array}{l} a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup \\ d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N \end{array}\right\} \quad \delta(q_i, a_j) = (q_k, a_\ell, d_m) \quad \left\} \quad 0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m \right.$$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

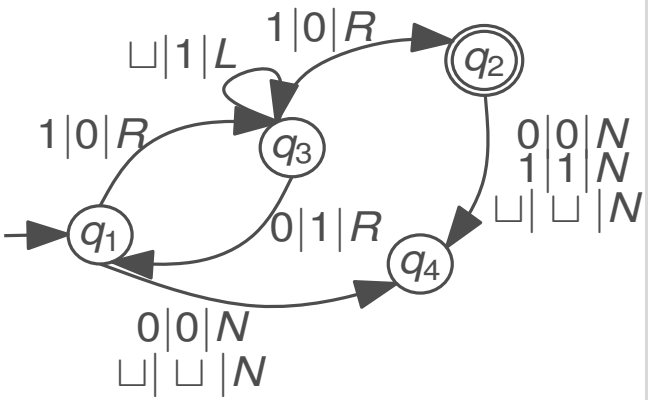
Kodierung:

$$a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup$$
$$d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N$$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	01010000101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

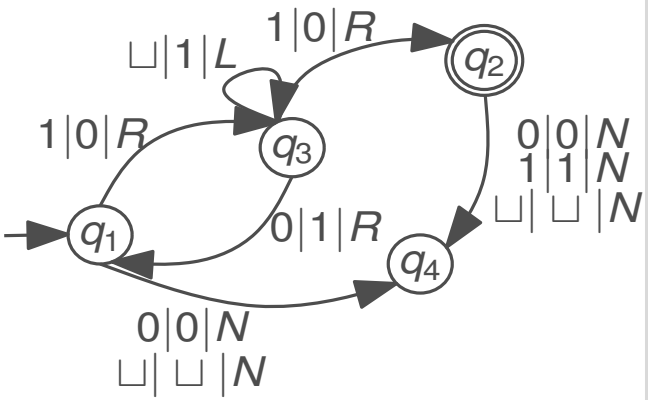
Kodierung:

$$a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup$$
$$d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N$$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\, 1 0^j\, 1\, 0^k\, 1 0^\ell 1 0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	01010000101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

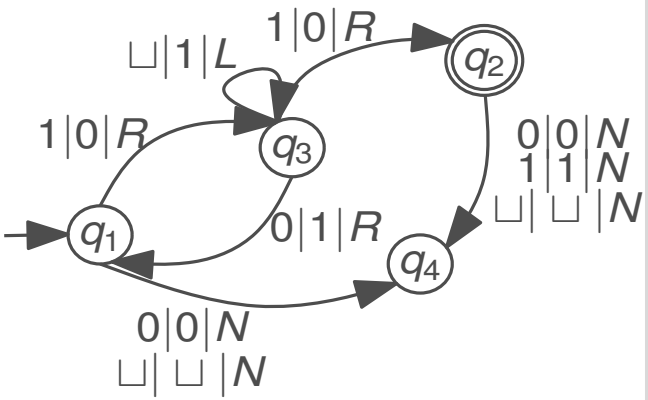
Kodierung:

$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$
$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\}$ $0^i\ 1\ 0^j\ 1\ 0^k\ 1\ 0^\ell\ 1\ 0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

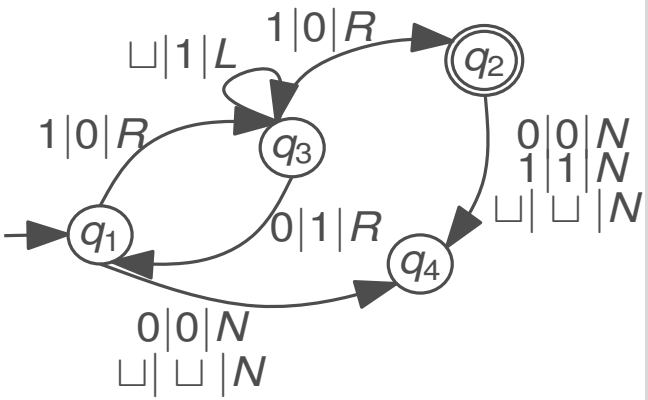
Kodierung:

$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$
$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010100 0
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

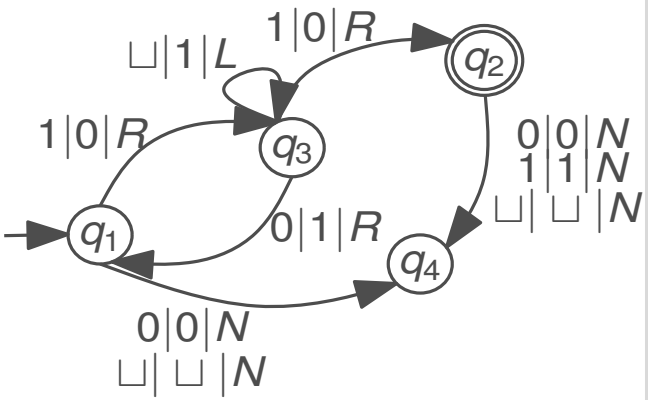
Kodierung:

$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$
$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\}$ $0^i 1 0^j 1 0^k 1 0^\ell 1 0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010 1 01000
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

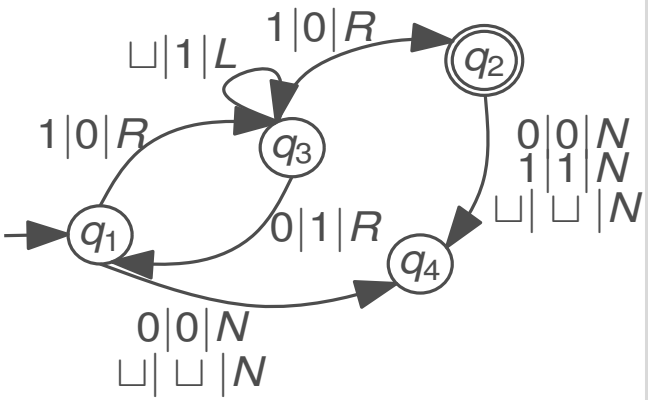
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

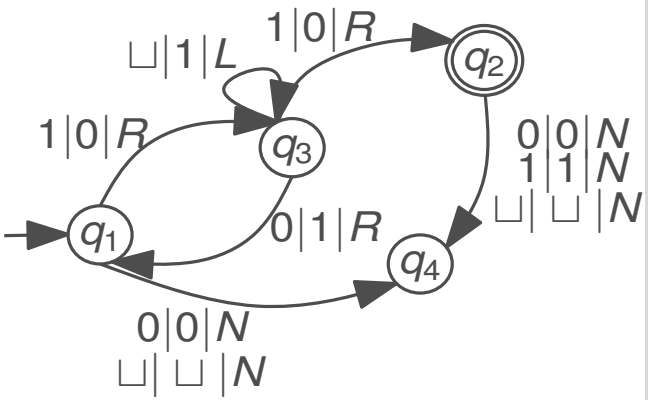
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

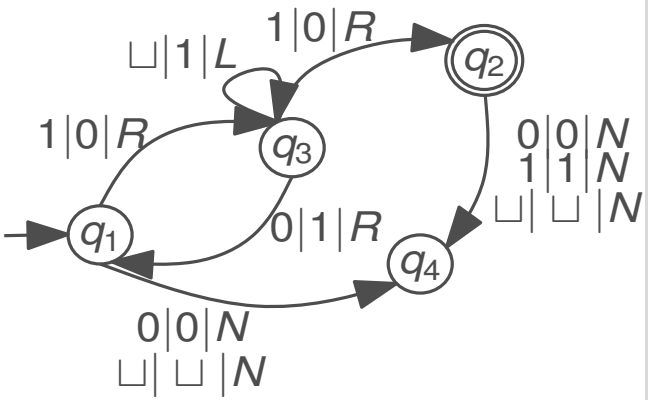
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	0100010000100010000
4	$\delta(q_3, 0) = (q_1, 1, R)$	
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

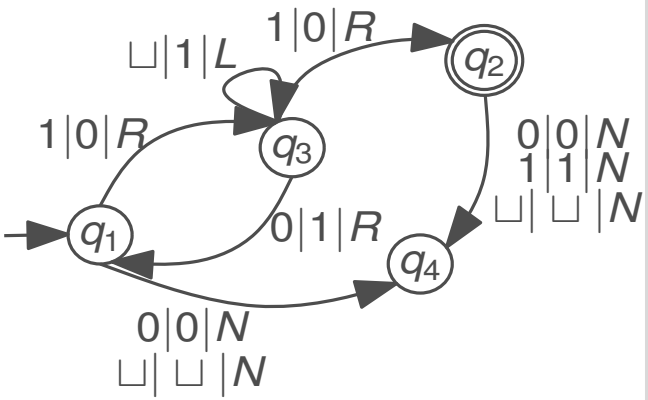
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	0100010000100010000
4	$\delta(q_3, 0) = (q_1, 1, R)$	0001010000101000
5	$\delta(q_3, 1) = (q_2, 0, R)$	
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

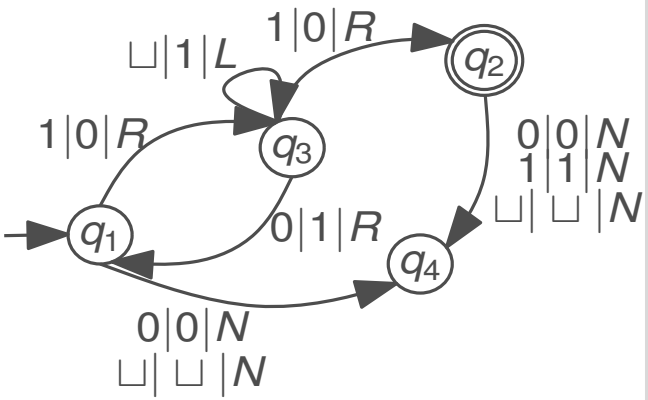
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	0100010000100010000
4	$\delta(q_3, 0) = (q_1, 1, R)$	0001010000101000
5	$\delta(q_3, 1) = (q_2, 0, R)$	0001001001010100
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

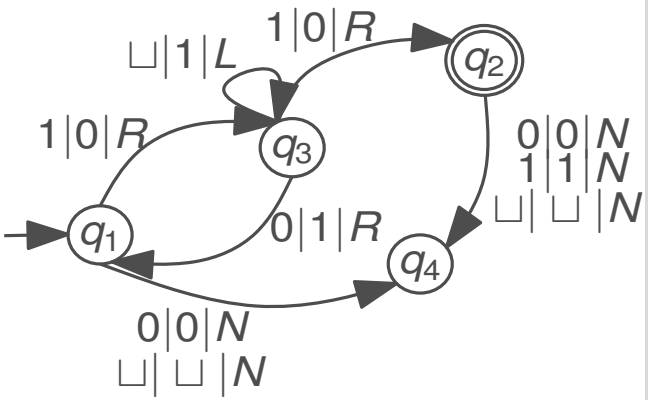
$a_1 \mathrel{\hat{=}} 0, a_2 \mathrel{\hat{=}} 1, a_3 \mathrel{\hat{=}} \sqcup$

$d_1 \mathrel{\hat{=}} L, d_2 \mathrel{\hat{=}} R, d_3 \mathrel{\hat{=}} N$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\,1\,0^j\,1\,0^k\,1\,0^\ell\,1\,0^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	0101000010101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	0100010000100010000
4	$\delta(q_3, 0) = (q_1, 1, R)$	0001010000101000
5	$\delta(q_3, 1) = (q_2, 0, R)$	0001001001010100
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	0001000100010010

δ	0	1	$\sqcup$
q_1	$(q_4, 0, N)$	$(q_3, 0, R)$	$(q_4, \sqcup, N)$
q_3	$(q_1, 1, R)$	$(q_2, 0, R)$	$(q_3, 1, L)$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

Kodierung:

$$\left.\begin{array}{l} a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup \\ d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N \end{array}\right\} \quad \delta(q_i, a_j) = (q_k, a_\ell, d_m) \quad \left\} \quad 0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m \right.$$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	01010000101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	0100010000100010000
4	$\delta(q_3, 0) = (q_1, 1, R)$	0001010000101000
5	$\delta(q_3, 1) = (q_2, 0, R)$	0001001001010 0
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	0001000100010010

$\langle \mathcal{M} \rangle$ = 111010100001010001101001000101001101000100001000100001
00010100001010001100010010010100110001000100010010111

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Gegeben ist folgende TM $\mathcal{M}$ mit $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$, $s = q_1$, $F = \{q_2\}$ und δ. Was ist die Gödelnummer $\langle \mathcal{M} \rangle$ der TM?

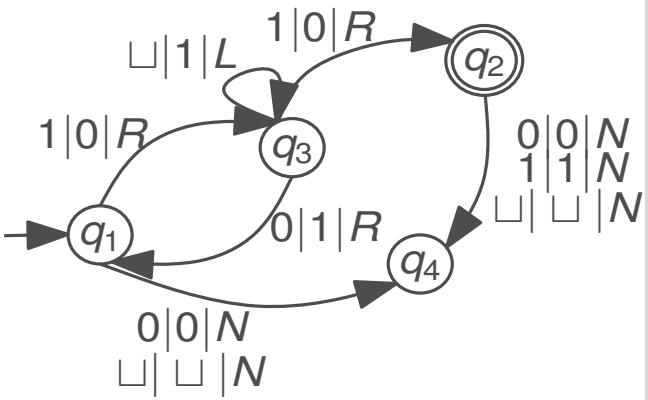
Kodierung:

$$a_1 \mathrel{\hat{=}} 0,\ a_2 \mathrel{\hat{=}} 1,\ a_3 \mathrel{\hat{=}} \sqcup$$
$$d_1 \mathrel{\hat{=}} L,\ d_2 \mathrel{\hat{=}} R,\ d_3 \mathrel{\hat{=}} N$$

$\Big\}$ $\delta(q_i, a_j) = (q_k, a_\ell, d_m)$ $\Big\}$ $0^i\ 10^j\ 1\ 0^k\ 10^\ell 10^m$

#	Eintrag	Kodierung
1	$\delta(q_1, 0) = (q_4, 0, N)$	01010000101000
2	$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
3	$\delta(q_1, \sqcup) = (q_4, \sqcup, N)$	010001000010001000
4	$\delta(q_3, 0) = (q_1, 1, R)$	0001010000101000
5	$\delta(q_3, 1) = (q_2, 0, R)$	00010010010100
6	$\delta(q_3, \sqcup) = (q_3, 1, L)$	0001000100010010

$\langle \mathcal{M} \rangle$ = 1483656541123892524722854796023 27

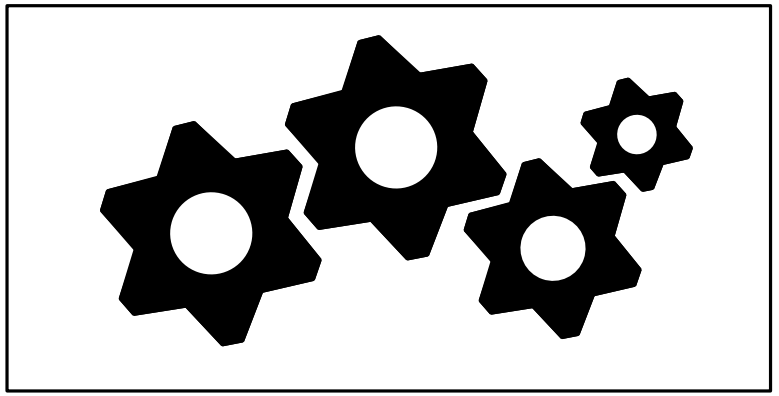


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.



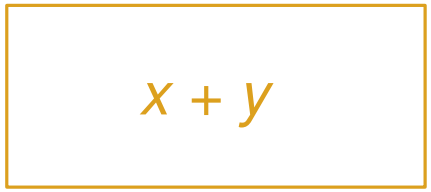
Universelle Turing-Maschine simuliert $\mathcal{M}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

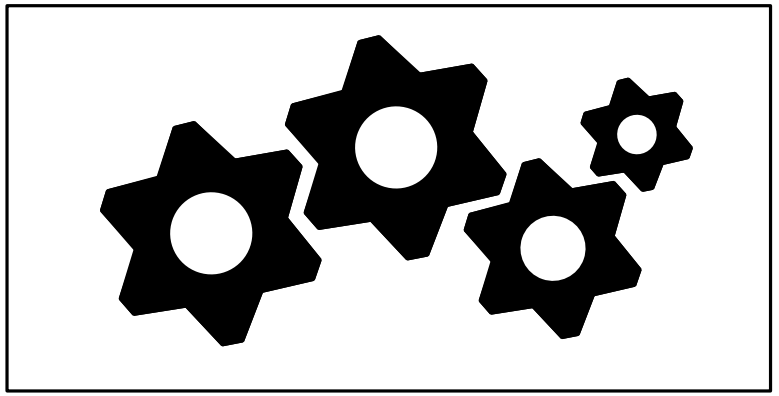
Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.



Spezielle Turing-Maschine $\mathcal{M}$
z.B.: Addition zweier Zahlen



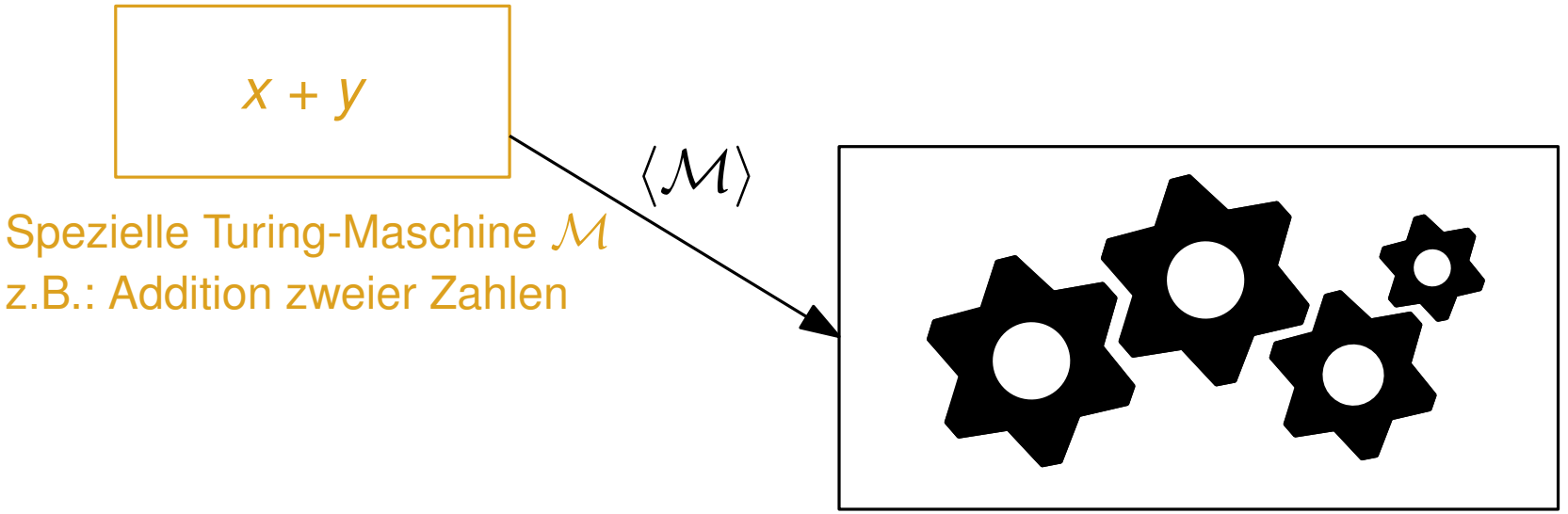
Universelle Turing-Maschine simuliert $\mathcal{M}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.

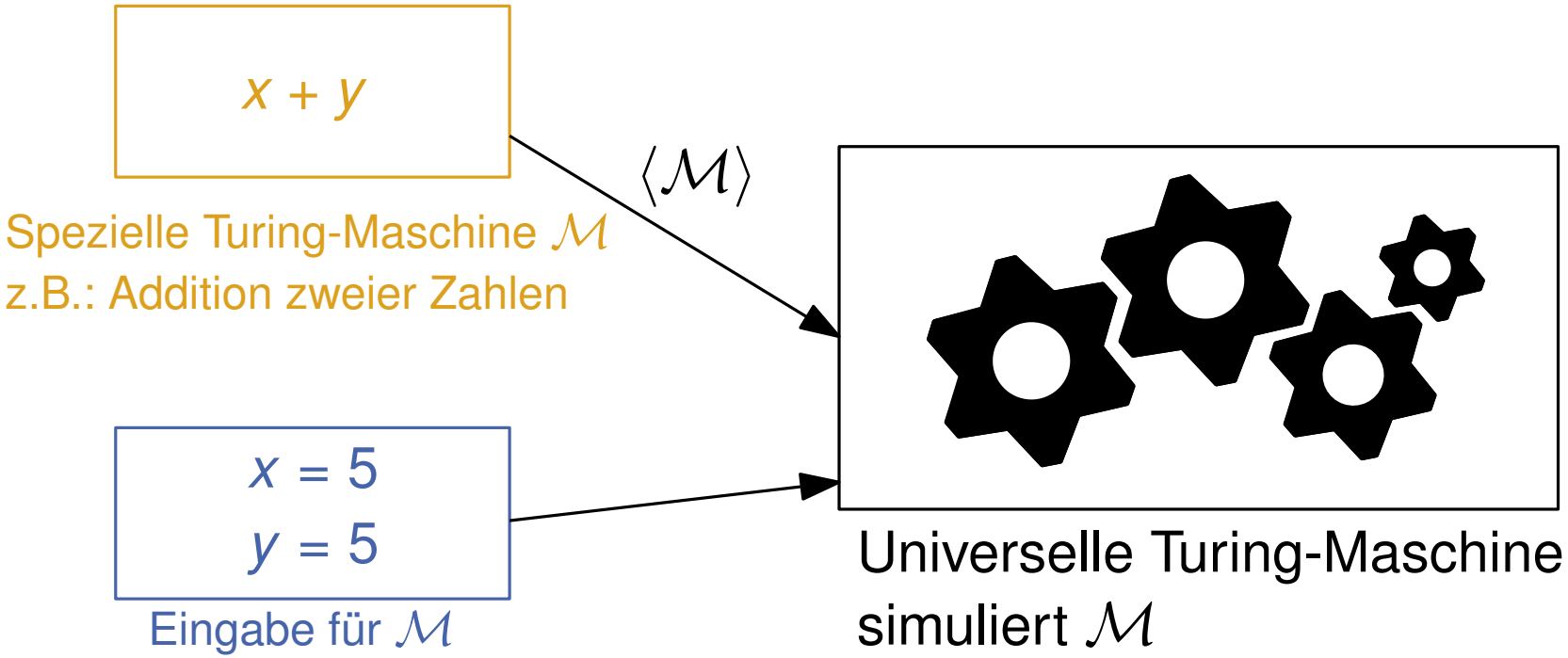


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.

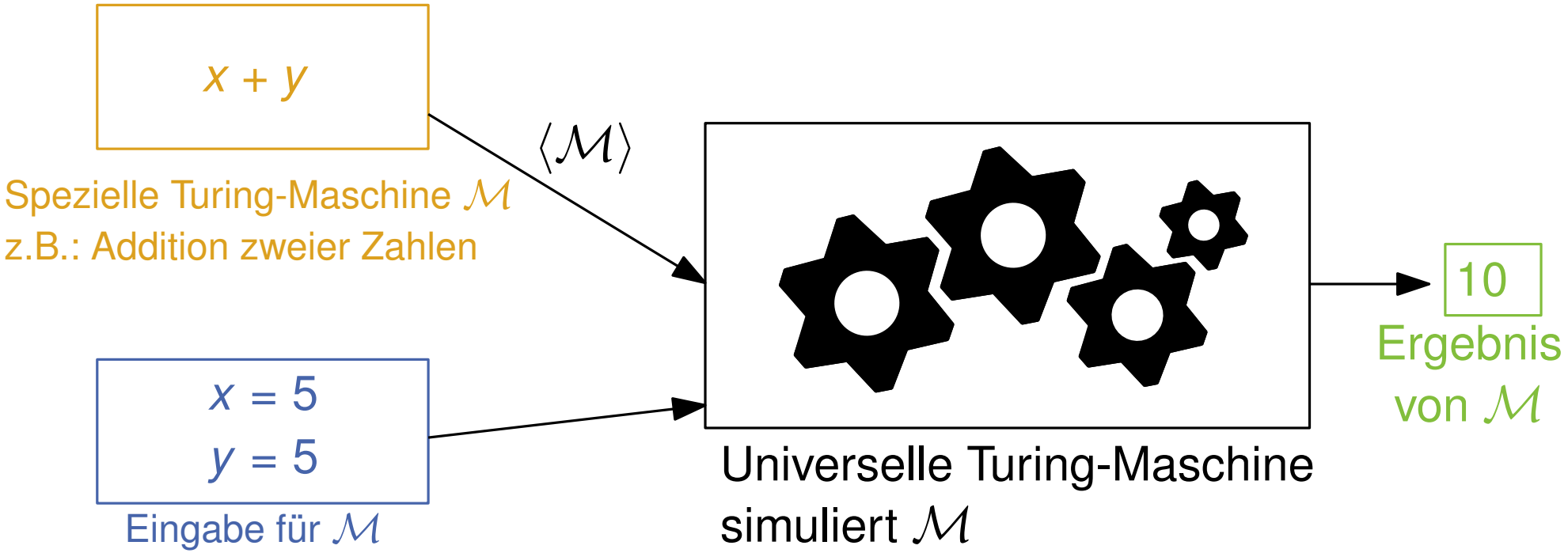


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.

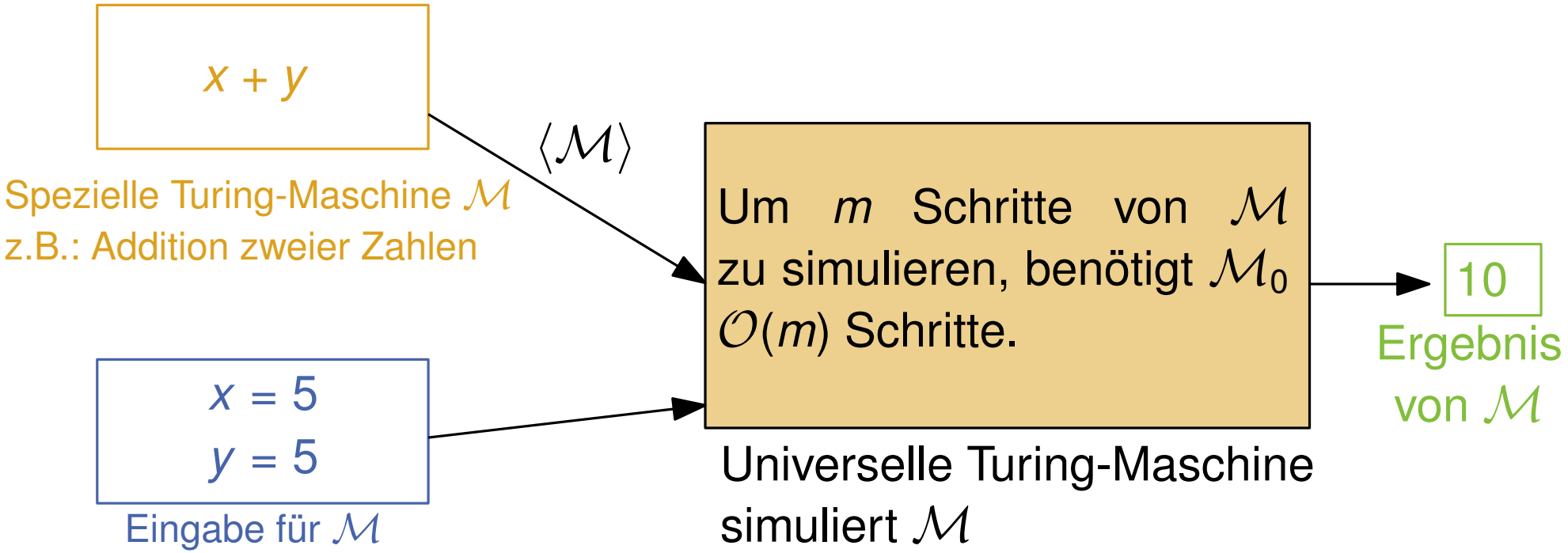


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle Turing-Maschine

Definition: Eine Turing-Maschine $\mathcal{M}_0$ heißt universell, falls für jede 1-Band-DTM $\mathcal{M}$ und jedes $x \in \{0,1\}^*$ gilt:

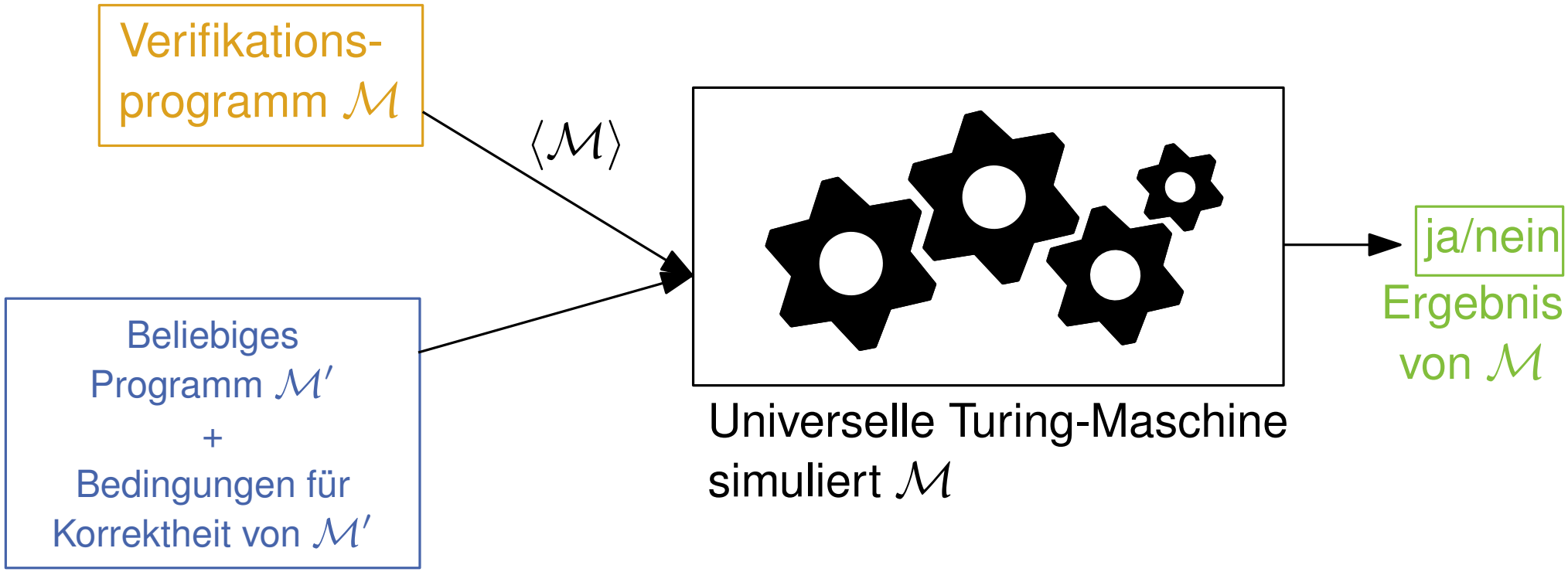
- $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ hält genau dann, wenn $\mathcal{M}$ gestartet mit x hält.
- Falls $\mathcal{M}$ gestartet mit x hält, berechnet $\mathcal{M}_0$ gestartet mit $\langle \mathcal{M} \rangle x$ die gleiche Ausgabe wie $\mathcal{M}$ gestartet mit x. Insbesondere akzeptiert $\mathcal{M}_0$ die Eingabe $\langle \mathcal{M} \rangle x$ genau dann, wenn $\mathcal{M}$ die Eingabe x akzeptiert.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle TM – Beispiel

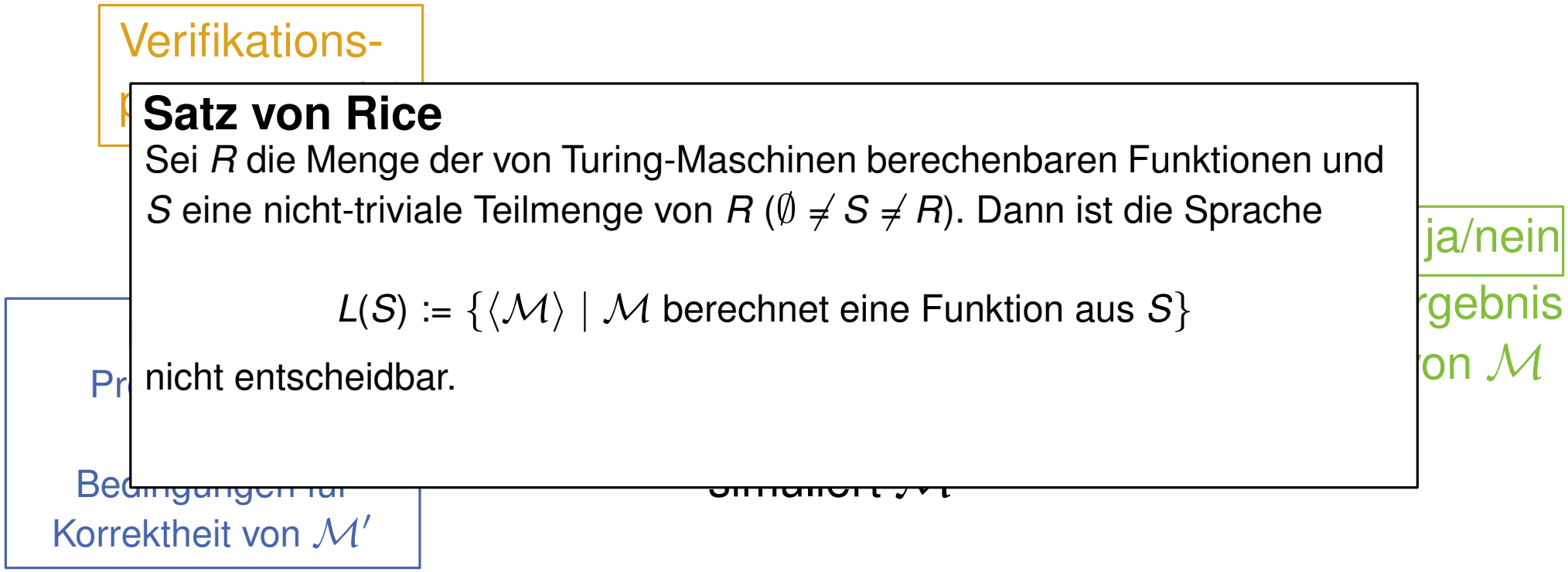
Gibt es eine Programm $\mathcal{M}$, das für jedes beliebige Programm $\mathcal{M}'$ dessen Korrektheit beweist?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Universelle TM – Beispiel

Gibt es eine Programm $\mathcal{M}$, das für jedes beliebige Programm $\mathcal{M}'$ dessen Korrektheit beweist?



Satz von Rice

Sei R die Menge der von Turing-Maschinen berechenbaren Funktionen und S eine nicht-triviale Teilmenge von R ($\emptyset \neq S \neq R$). Dann ist die Sprache

$$L(S) := \{\langle \mathcal{M} \rangle \mid \mathcal{M} \text{ berechnet eine Funktion aus } S\}$$

nicht entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?
$w_1 =$ 0 0 0 0 0 ...		
$w_2 =$ 1 0 0 0 0 ...		
$w_3 =$ 0 1 0 0 0 ...		
$w_4 =$ 1 1 0 0 0 ...		
$w_5 =$ 0 0 1 0 0 ...		

$L_d = \{$ $\}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	→ T_{w_1} entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$			
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$			
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$			
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$			

$L_d = \{\ w_1 \qquad \}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$			
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$			
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$			

$L_d = \{$ w_1 w_2 $\}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_3}$ entscheidet nicht L_d
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$			
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$			

$L_d = \{ w_1 \quad w_2 \quad \}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_3}$ entscheidet nicht L_d
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_4}$ entscheidet nicht L_d
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$			

$L_d = \{ w_1 \quad w_2 \quad w_4 \quad \}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_3}$ entscheidet nicht L_d
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_4}$ entscheidet nicht L_d
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_5}$ entscheidet nicht L_d

$L_d = \{ w_1 \quad w_2 \quad w_4 \quad \}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_3}$ entscheidet nicht L_d
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_4}$ entscheidet nicht L_d
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_5}$ entscheidet nicht L_d
$\ldots$	$\ldots$	$\ldots$	$\rightarrow T_{w_i}$ entscheidet nicht L_d

$L_d = \{ w_1 \quad w_2 \quad w_4 \quad \ldots \}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Unentscheidbarkeit – Diagonalargument

Zeige mit Diagonalargument, dass es unentscheidbare Sprachen gibt:

Gödelnummer in kan. Reihenf.	$w_i \in L(T_{w_i})$?	$w_i \in L_d$?	
$w_1 = 0\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_1}$ entscheidet nicht L_d
$w_2 = 1\ 0\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_2}$ entscheidet nicht L_d
$w_3 = 0\ 1\ 0\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_3}$ entscheidet nicht L_d
$w_4 = 1\ 1\ 0\ 0\ 0\ \ldots$	nein	ja	$\rightarrow T_{w_4}$ entscheidet nicht L_d
$w_5 = 0\ 0\ 1\ 0\ 0\ \ldots$	ja	nein	$\rightarrow T_{w_5}$ entscheidet nicht L_d
$\ldots$	$\ldots$	$\ldots$	$\rightarrow T_{w_i}$ entscheidet nicht L_d

$L_d = \{ w_1 \quad w_2 \quad w_4 \quad \ldots \}$

- jede TM hat Gödelnummer $\Rightarrow$ jede TM taucht in Tabelle auf
- keine TM entscheidet die Diagonalsprache L_d

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgaben zu Entscheidbarkeit

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

- Wähle nichtdeterministisch ein nicht-leeres Präfix π von x.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

- Wähle nichtdeterministisch ein nicht-leeres Präfix π von x.
- Überprüfe mithilfe von $\mathcal{M}$, ob π in L liegt.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

- Wähle nichtdeterministisch ein nicht-leeres Präfix π von x.
- Überprüfe mithilfe von $\mathcal{M}$, ob π in L liegt.

  Fall: π liegt nicht in $L \rightarrow M$ akzeptiert x nicht.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

- Wähle nichtdeterministisch ein nicht-leeres Präfix π von x.
- Überprüfe mithilfe von $\mathcal{M}$, ob π in L liegt.

  Fall: π liegt nicht in $L \rightarrow M$ akzeptiert x nicht.

  Fall: π liegt in L: M löscht π vom Band. Falls das Band nun leer ist, *akzeptiert* M die Eingabe x. Sonst wiederhole Verfahren.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(a) Die Menge der entscheidbaren Sprachen ist unter dem Kleene'schen Abschluss abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass L^* auch entscheidbar ist.

Idee:

- Sei L eine entscheidbare Sprache.
- Es gibt also eine Turing-Maschine $\mathcal{M}$, die L entscheidet: $L(\mathcal{M}) = L$.
- Konstruiere eine NTM $\mathcal{M}'$.

Verfahren: Sei x die Eingabe.

- Wähle nichtdeterministisch ein nicht-leeres Präfix π von x.
- Überprüfe mithilfe von $\mathcal{M}$, ob π in L liegt.

  Fall: π liegt nicht in $L \rightarrow M$ akzeptiert x nicht.

  Fall: π liegt in L: M löscht π vom Band. Falls das Band nun leer ist, *akzeptiert* M die Eingabe x. Sonst wiederhole Verfahren.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Die Operation min ist für eine entscheidbare Sprache L definiert als

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Hinweis: Ein Präfix von x heißt *echt*, wenn es nicht x ist.

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen, d.h. für jede entscheidbare Sprache L gilt, dass $\min(L)$ auch entscheidbar ist.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.
4. Es gibt kein weiteres Präfix mehr: T akzeptiert.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.
4. Es gibt kein weiteres Präfix mehr: T akzeptiert.
5. T_L entscheidet p.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.
4. Es gibt kein weiteres Präfix mehr: T akzeptiert.
5. T_L entscheidet p.
6. Wenn $p \in L$, dann hält T und akzeptiert nicht.

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.
4. Es gibt kein weiteres Präfix mehr: T akzeptiert.
5. T_L entscheidet p.
6. Wenn $p \in L$, dann hält T und akzeptiert nicht.

sonst

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(b) Die Menge der entscheidbaren Sprachen ist bzgl. der Operation min abgeschlossen mit

$$\min(L) := \{x \in L \mid \text{kein echtes Präfix von } x \text{ ist in L}\}.$$

Idee:

- Die TM T_L entscheidet die Sprache $L \subseteq \Sigma^*$.
- T' generiert alle echten Präfixe ihrer Eingabe ohne Wiederholung.

Arbeitsweise der TM T, die $\min(L)$ entscheidet, mit der Eingabe x:

1. T_L entscheidet x.
2. Wenn T_L nicht akzeptiert, hält T und akzeptiert nicht.
3. T' generiert das nächste echte Präfix p von x.
4. Es gibt kein weiteres Präfix mehr: T akzeptiert.
5. T_L entscheidet p.
6. Wenn $p \in L$, dann hält T und akzeptiert nicht.

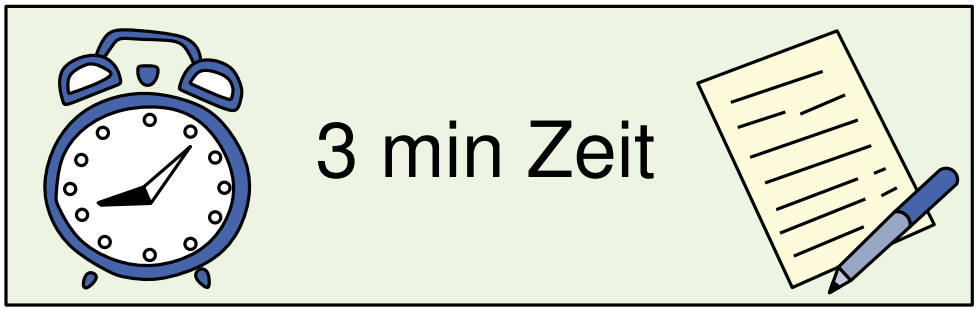
sonst

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Aus der Vorlesung:

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Aus der Vorlesung:

(a) L_u ist semi-entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Aus der Vorlesung:

(a) L_u ist semi-entscheidbar.

(b) L_u ist nicht entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Aus der Vorlesung:

(a) L_u ist semi-entscheidbar.

(b) L_u ist nicht entscheidbar.

(c) Für jede Sprache L: L, L^c semi-entscheidbar $\Leftrightarrow$ L entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Abgeschlossenheit von entscheidbaren Sprachen

Zeigen Sie:

(c) Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

Idee:

- Verwende universelle Sprache $L_u := \{wv \mid v \in L(T_w)\}$.

Aus der Vorlesung:

(a) L_u ist semi-entscheidbar.

(b) L_u ist nicht entscheidbar.

(c) Für jede Sprache L: L, L^c semi-entscheidbar $\Leftrightarrow$ L entscheidbar.

Annahme: L_u^c ist semi-entscheidbar.
$\Rightarrow$ Da L_u semi-entscheidbar ist, wäre damit L_u entscheidbar.
Widerspruch zu: L_u ist nicht entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

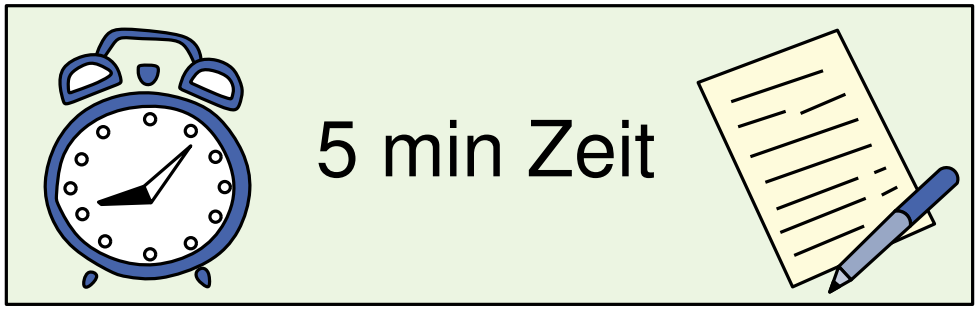
Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Hinweis: Es ist nicht bekannt, für welche n die Dezimaldarstellung von π das Teilwort 1^n enthält, aber das ist hier auch nicht wichtig!



Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Beweis: Unterscheide zwei Fälle:

- Die Dezimaldarstellung von π enthält 1^n für jedes $n \in \mathbb{N}$.

Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Beweis: Unterscheide zwei Fälle:

- Die Dezimaldarstellung von π enthält 1^n für jedes $n \in \mathbb{N}$.
  - → Dann akzeptiert die TM, die jedes Wort, das nur das Zeichen 1 enthält, gerade die Sprache L.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Beweis: Unterscheide zwei Fälle:

- Die Dezimaldarstellung von π enthält 1^n für jedes $n \in \mathbb{N}$.
  - → Dann akzeptiert die TM, die jedes Wort, das nur das Zeichen 1 enthält, gerade die Sprache L.
- Es gibt ein maximales $\hat{n}$, sodass die Dezimaldarstellung von π das Wort $1^{\hat{n}}$ enthält, das Wort $1^{\hat{n}+1}$ aber nicht.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Beweis: Unterscheide zwei Fälle:

- Die Dezimaldarstellung von π enthält 1^n für jedes $n \in \mathbb{N}$.
  - → Dann akzeptiert die TM, die jedes Wort, das nur das Zeichen 1 enthält, gerade die Sprache L.
- Es gibt ein maximales $\hat{n}$, sodass die Dezimaldarstellung von π das Wort $1^{\hat{n}}$ enthält, das Wort $1^{\hat{n}+1}$ aber nicht.
  - → Dann akzeptiert die TM, die jedes Wort, das nur das Zeichen 1 enthält und Länge max. $\hat{n}$ hat, gerade die Sprache L.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Sei

$$L = \{1^n \mid 1^n \text{ ist Teilwort der Dezimaldarstellung von } \pi\}.$$

Zeigen Sie, dass L entscheidbar ist.

Beweis: Unterscheide zwei Fälle:

- Die Dezimaldarstellung von π enthält 1^n für jedes $n \in \mathbb{N}$.
- Es gibt ein maximales $\hat{n}$, sodass die Dezimaldarstellung von π das Wort $1^{\hat{n}}$ enthält, das Wort $1^{\hat{n}+1}$ aber nicht.

- Es ist für den Beweis egal, dass wir nicht wissen, welcher der beiden Fälle zutrifft.
- Die Sprache L ist sogar regulär!

Aufgabe – Entscheidbarkeit

Das Halteproblem definiert folgende Sprache:

$$\mathcal{H} = \{\langle w, v\rangle \mid T_w \text{ hält auf der Eingabe } v\}$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

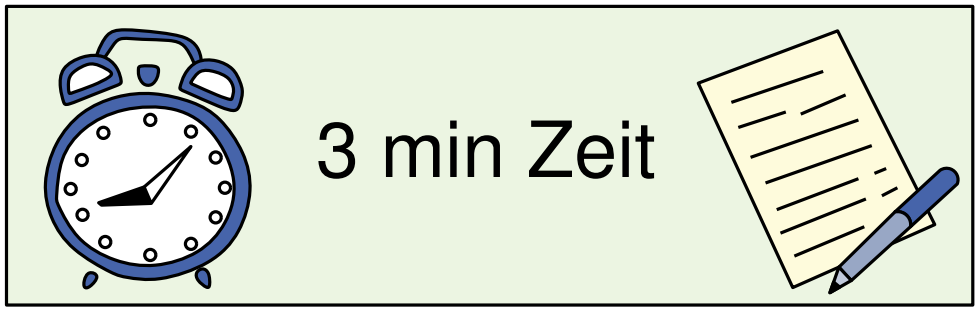
Das Halteproblem definiert folgende Sprache:

$$\mathcal{H} = \{\langle w, v\rangle \mid T_w \text{ hält auf der Eingabe } v\}$$

Fehlerhafter Beweisversuch, dass $\mathcal{H}$ entscheidbar ist:

Unterscheide zwei Fälle, wie bei der letzten Aufgabe:

- T_w hält auf der Eingabe v.
  - → Dann liefert die TM, die alles akzeptiert, die richtige Antwort.
- T_w stoppt bei Eingabe v niemals.
  - → Dann liefert die TM, die alles ablehnt, die richtige Antwort.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Das Halteproblem definiert folgende Sprache:

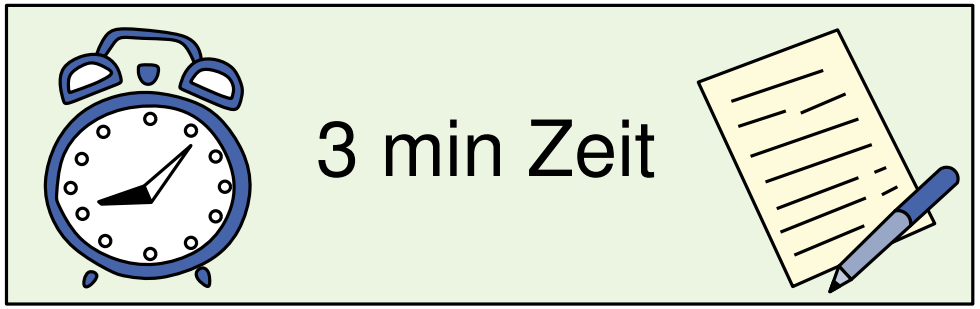
$$\mathcal{H} = \{\langle w, v\rangle \mid T_w \text{ hält auf der Eingabe } v\}$$

Fehlerhafter Beweisversuch, dass $\mathcal{H}$ entscheidbar ist:

Unterscheide zwei Fälle, wie bei der letzten Aufgabe:

- T_w hält auf der Eingabe v.
  - → Dann liefert die TM, die alles akzeptiert, die richtige Antwort.
- T_w stoppt bei Eingabe v niemals.
  - → Dann liefert die TM, die alles ablehnt, die richtige Antwort.

Wieso ist dieser Beweis nicht korrekt?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Aufgabe – Entscheidbarkeit

Das Halteproblem definiert folgende Sprache:

$$\mathcal{H} = \{\langle w, v\rangle \mid T_w \text{ hält auf der Eingabe } v\}$$

Fehlerhafter Beweisversuch, dass $\mathcal{H}$ entscheidbar ist:

Unterscheide zwei Fälle, wie bei der letzten Aufgabe:

- T_w hält auf der Eingabe v.
  - → Dann liefert die TM, die alles akzeptiert, die richtige Antwort.
- T_w stoppt bei Eingabe v niemals.
  - → Dann liefert die TM, die alles ablehnt, die richtige Antwort.

Wieso ist dieser Beweis nicht korrekt?

T_w hängt von Eingabe w ab!

Es kann nicht **eine** TM **für jedes** T_w entscheiden, welcher Fall zutrifft.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

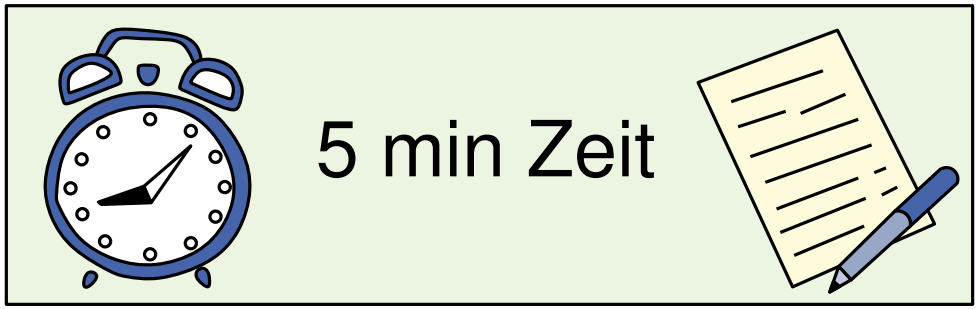
Klausuraufgabe – Entscheidbarkeit

Zeigen Sie, dass die Sprache

$$L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$$

nicht entscheidbar ist.

Verwenden Sie nicht den Satz von Rice.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.
  - $L(\mathcal{M}_{wv}) = \begin{cases} \Sigma^* & \text{falls } \mathcal{M}_{wv} \text{ auf } v \text{ hält} \\ \emptyset & \text{sonst} \end{cases}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.
  - $L(\mathcal{M}_{wv}) = \begin{cases} \Sigma^* & \text{falls } \mathcal{M}_{wv} \text{ auf } v \text{ hält} \\ \emptyset & \text{sonst} \end{cases}$
- Konstruiere TM $\mathcal{M}_*$, die alle Eingaben akzeptiert.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.
  - $L(\mathcal{M}_{wv}) = \begin{cases} \Sigma^* & \text{falls } \mathcal{M}_{wv} \text{ auf } v \text{ hält} \\ \emptyset & \text{sonst} \end{cases}$
- Konstruiere TM $\mathcal{M}_*$, die alle Eingaben akzeptiert.
- Dann gilt: $(\langle M_{wv} \rangle, \langle M_* \rangle) \in L \Leftrightarrow \langle w, v \rangle \in \mathcal{H}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.
  - $L(\mathcal{M}_{wv}) = \begin{cases} \Sigma^* & \text{falls } \mathcal{M}_{wv} \text{ auf } v \text{ hält} \\ \emptyset & \text{sonst} \end{cases}$
- Konstruiere TM $\mathcal{M}_*$, die alle Eingaben akzeptiert.
- Dann gilt: $(\langle M_{wv} \rangle, \langle M_* \rangle) \in L \Leftrightarrow \langle w, v \rangle \in \mathcal{H}$
- Simuliere $\mathcal{M}_L$ auf der Eingabe $(\langle M_{wv} \rangle, \langle M_* \rangle)$. Akzeptiere genau dann, wenn $\mathcal{M}_L$ akzeptiert.

Klausuraufgabe – Entscheidbarkeit

Zeige: $L = \{(u, v) \mid w \in L(T_u) \Leftrightarrow w^R \in L(T_v)\}$ nicht entscheidbar.

Annahme: L ist entscheidbar von TM $\mathcal{M}_L$.

Konstruiere daraus TM $\mathcal{M}_{\mathcal{H}}$, die das Halteproblem $\mathcal{H}$ entscheidet:

- Eingabe: $\mathcal{H}$-Instanz $\langle w, v \rangle$
- Konstruiere TM $\mathcal{M}_{wv}$:
  - Verwerfe eigene Eingabe und simuliere $\mathcal{M}_w$ mit Eingabe v.
  - Akzeptiere genau dann, wenn $\mathcal{M}_w$ stoppt.
  - $L(\mathcal{M}_{wv}) = \begin{cases} \Sigma^* & \text{falls } \mathcal{M}_{wv} \text{ auf } v \text{ hält} \\ \emptyset & \text{sonst} \end{cases}$
- Konstruiere TM $\mathcal{M}_*$, die alle Eingaben akzeptiert.
- Dann gilt: $(\langle M_{wv} \rangle, \langle M_* \rangle) \in L \Leftrightarrow \langle w, v \rangle \in \mathcal{H}$
- Simuliere $\mathcal{M}_L$ auf der Eingabe $(\langle M_{wv} \rangle, \langle M_* \rangle)$. Akzeptiere genau dann, wenn $\mathcal{M}_L$ akzeptiert.

$\Rightarrow$ Halteproblem entschieden. Widerspruch.

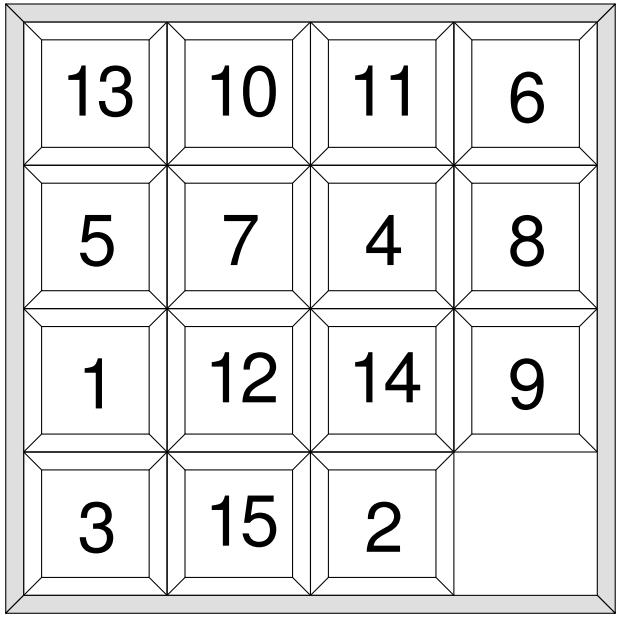
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

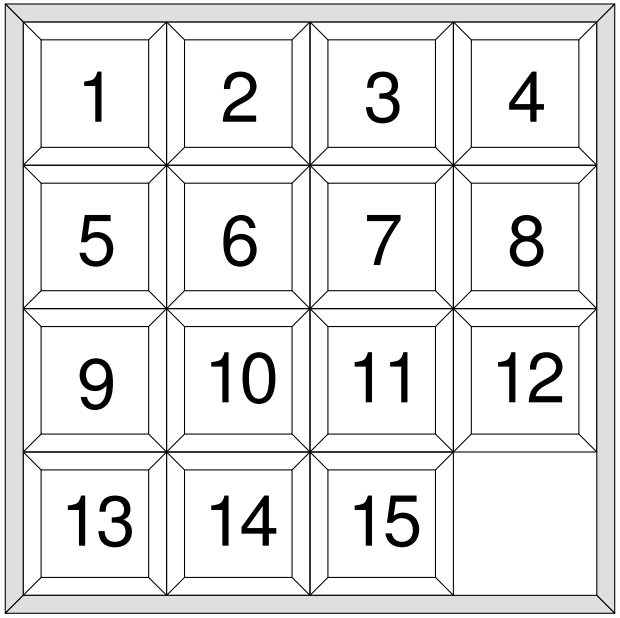
Komplexitätstheorie

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Problemstellung und -kodierung

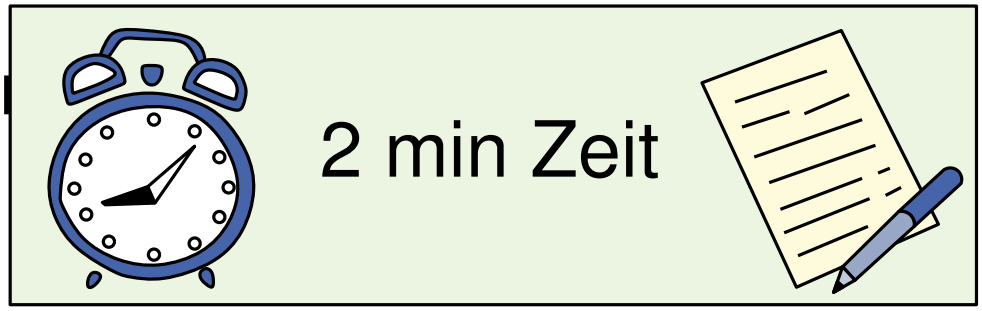
15-Puzzle





- Formulieren Sie das Problem 15-Puzzle als Entscheidungs-, Optimalwert- und Optimierungsproblem.
- Geben Sie ein Kodierungsschema an und bestimmen Sie die Kodierungslänge der Instanzen.

Problemstellung und -kodieru



15-Puzzle

13	10	11	6
5	7	4	8
1	12	14	9
3	15	2	

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

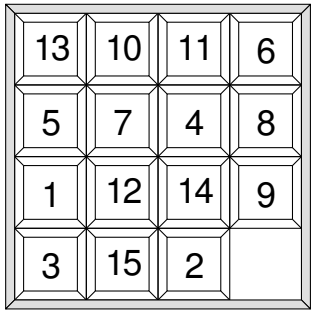
- Formulieren Sie das Problem 15-Puzzle als Entscheidungs-, Optimalwert- und Optimierungsproblem.
- Geben Sie ein Kodierungsschema an und bestimmen Sie die Kodierungslänge der Instanzen.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

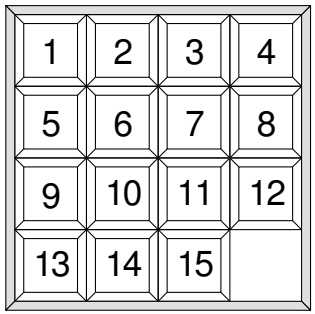
Problemstellung und -kodierung

Entscheidungsproblem:

Optimalwertproblem:



Optimierungsproblem:



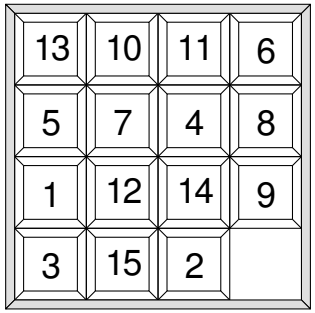
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Problemstellung und -kodierung

Entscheidungsproblem:

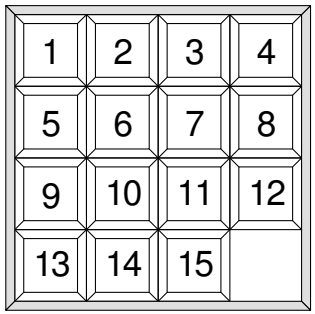
Gegeben: Anfangskonfiguration von 15-Puzzle und ein Parameter k

Gesucht: Gibt es eine Folge von Zügen der Länge $\leq k$, die die Anfangskonfiguration in die Zielkonfiguration überführt?



Optimalwertproblem:

Optimierungsproblem:



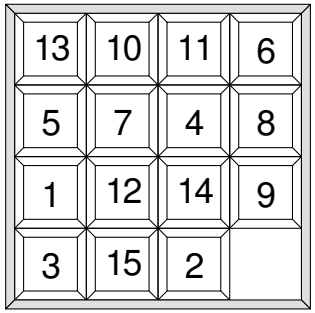
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Problemstellung und -kodierung

Entscheidungsproblem:

Gegeben: Anfangskonfiguration von 15-Puzzle und ein Parameter k

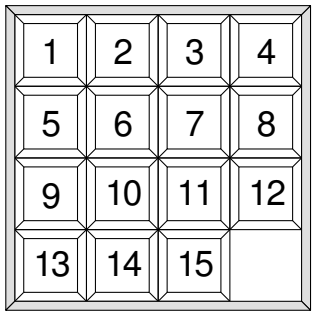
Gesucht: Gibt es eine Folge von Zügen der Länge $\leq k$, die die Anfangskonfiguration in die Zielkonfiguration überführt?



Optimalwertproblem:

Gegeben: Anfangskonfiguration von 15-Puzzle

Gesucht: Die Länge einer kürzesten Folge von Zügen, die die Anfangskonfiguration in die Zielkonfiguration überführt.



Optimierungsproblem:

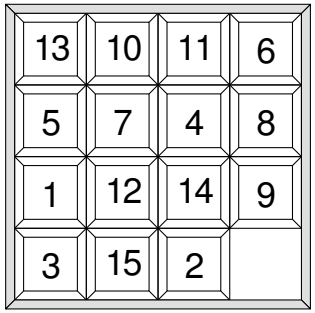
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Problemstellung und -kodierung

Entscheidungsproblem:

Gegeben: Anfangskonfiguration von 15-Puzzle und ein Parameter k

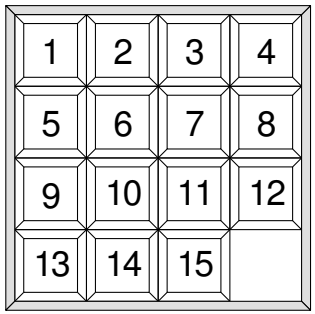
Gesucht: Gibt es eine Folge von Zügen der Länge $\leq k$, die die Anfangskonfiguration in die Zielkonfiguration überführt?



Optimalwertproblem:

Gegeben: Anfangskonfiguration von 15-Puzzle

Gesucht: Die Länge einer kürzesten Folge von Zügen, die die Anfangskonfiguration in die Zielkonfiguration überführt.



Optimierungsproblem:

Gegeben: Anfangskonfiguration von 15-Puzzle

Gesucht: Folge von Zügen mit minimaler Länge, die die Anfangskonfiguration in die Zielkonfiguration überführt.

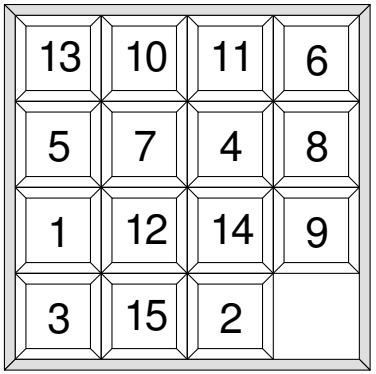
Problemstellung und -kodierung

Mögliche Kodierung:

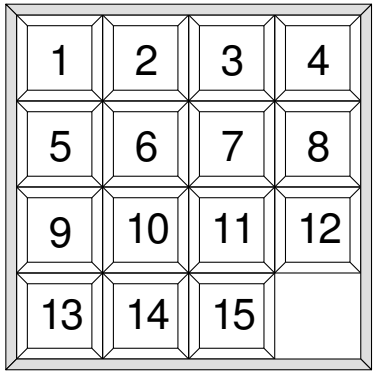
- Eine Konfiguration als Folge $v_1, v_2, \ldots, v_{16}$ der 16 Kacheln (inklusive der leeren Kachel)
- Die Kachel mit Nummer i als Hexadezimalzahl i, und
- das leere Feld mit der Hexadezimalzahl 0.

Die Länge der Kodierung ist somit

$$\sum_{i=1}^{16} \langle v_i \rangle = \sum_{i=1}^{16} 1 = 16$$



D A B 6 5 7 4 8 1 C E 9 3 F 2 0



1 2 3 4 5 6 7 8 9 A B C D E F 0

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Das Entscheidungsproblem Π, ob eine gegebene Zahl eine Zweierpotenz ist, ist durch die Probleminstanzen $D_\Pi := \mathbb{N}$ und die Ja-Instanzen $J_\Pi := \{2^i | i \in \mathbb{N}\}$ gegeben.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Das Entscheidungsproblem Π, ob eine gegebene Zahl eine Zweierpotenz ist, ist durch die Probleminstanzen $D_\Pi := \mathbb{N}$ und die Ja-Instanzen $J_\Pi := \{2^i | i \in \mathbb{N}\}$ gegeben.

Seien s_b die Kodierungsschemata, die natürliche Zahlen auf ihre b-äre Repräsentation abbilden.

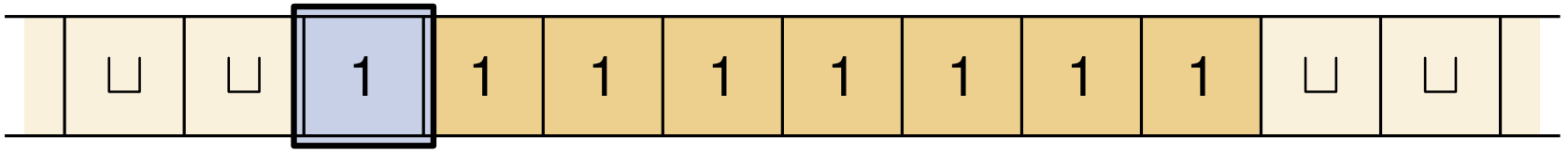
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Das Entscheidungsproblem Π, ob eine gegebene Zahl eine Zweierpotenz ist, ist durch die Probleminstanzen $D_\Pi := \mathbb{N}$ und die Ja-Instanzen $J_\Pi := \{2^i | i \in \mathbb{N}\}$ gegeben.

Seien s_b die Kodierungsschemata, die natürliche Zahlen auf ihre b-äre Repräsentation abbilden.

- Ja-Instanz 8_{10} für s_1 auf TM-Band kodiert:



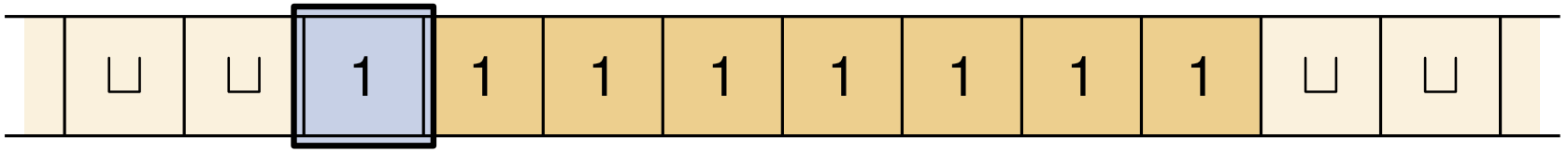
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

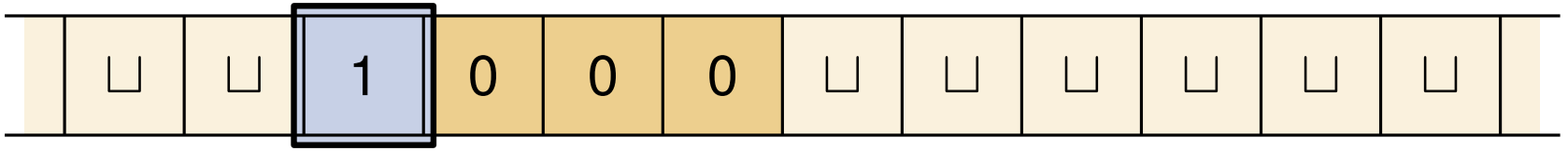
Das Entscheidungsproblem Π, ob eine gegebene Zahl eine Zweierpotenz ist, ist durch die Probleminstanzen $D_\Pi := \mathbb{N}$ und die Ja-Instanzen $J_\Pi := \{2^i | i \in \mathbb{N}\}$ gegeben.

Seien s_b die Kodierungsschemata, die natürliche Zahlen auf ihre b-äre Repräsentation abbilden.

- Ja-Instanz 8_{10} für s_1 auf TM-Band kodiert:



- Ja-Instanz 8_{10} für s_2 auf TM-Band kodiert:



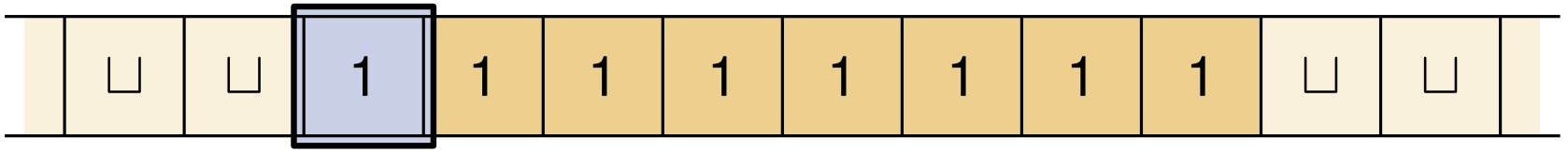
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

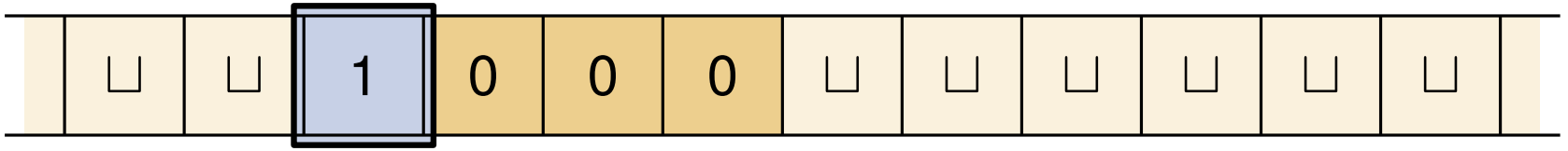
Kann die Wahl des Kodierungsschemas die Laufzeitkomplexität eines Problems beeinflussen?

Seien s_b die Kodierungsschemata, die natürliche Zahlen auf ihre b-äre Repräsentation abbilden.

- Ja-Instanz 8_{10} für s_1 auf TM-Band kodiert:



- Ja-Instanz 8_{10} für s_2 auf TM-Band kodiert:



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.
- Überprüfe dabei, ob nach der führenden 1 noch eine 1 vorkommt.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.
- Überprüfe dabei, ob nach der führenden 1 noch eine 1 vorkommt.
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.
- Überprüfe dabei, ob nach der führenden 1 noch eine 1 vorkommt.
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst akzeptiere die Eingabe.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.
- Überprüfe dabei, ob nach der führenden 1 noch eine 1 vorkommt.
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst akzeptiere die Eingabe.

$\Rightarrow$ Zeitkomplexität der TM ist linear in der Eingabegröße.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_2]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid i \in \mathbb{N}\})$.

Konvention: Es gibt keine führenden Nullen (außer die Eingabe ist 0).

Beobachtung: Die Zweierpotenzen haben in Binärdarstellung genau die Form $10 \ldots 0$.

- Überprüfe, ob die Eingabe 0 ist (also ob an erster Stelle keine 1 steht).
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst gehe schrittweise nach rechts bis zum Ende der Eingabe.
- Überprüfe dabei, ob nach der führenden 1 noch eine 1 vorkommt.
- Falls ja, stoppe die Berechnung und lehne die Eingabe ab.
- Sonst akzeptiere die Eingabe.

$\Rightarrow$ Zeitkomplexität der TM ist linear in der Eingabegröße.

$\Rightarrow$ $L[\Pi, s_2]$ liegt in P.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.
- Wenn bei einem Durchlauf ungerade viele Einsen erkannt wurden, stoppe und lehne die Eingabe ab.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.
- Wenn bei einem Durchlauf ungerade viele Einsen erkannt wurden, stoppe und lehne die Eingabe ab.
- Ansonsten: akzeptiere die Eingabe, falls am Ende eine 1 stehen bleibt.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.
- Wenn bei einem Durchlauf ungerade viele Einsen erkannt wurden, stoppe und lehne die Eingabe ab.
- Ansonsten: akzeptiere die Eingabe, falls am Ende eine 1 stehen bleibt.

$\Rightarrow$ Zeitkomplexität der TM ist *quadratisch* in der Eingabegröße.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Wir betrachten $L[\Pi, s_1]$ zu dem Problem $\Pi = (D_\Pi := \mathbb{N}, J_\Pi := \{2^i \mid \in \mathbb{N}\})$.

Eine TM, die $L[\Pi, s_1]$ entscheidet, kann wie folgt konstruiert werden:

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.
- Wenn bei einem Durchlauf ungerade viele Einsen erkannt wurden, stoppe und lehne die Eingabe ab.
- Ansonsten: akzeptiere die Eingabe, falls am Ende eine 1 stehen bleibt.

$\Rightarrow$ Zeitkomplexität der TM ist *quadratisch* in der Eingabegröße.

$\Rightarrow$ $L[\Pi, s_1]$ liegt in P.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kodierungsschemata

Fazit (ohne Beweis): Die Wahl des Kodierungsschemas kann die Laufzeitkomplexität eines Problems beeinflussen.

Konvention: Falls nicht anderweitig spezifiert, wird die Eingabe binär kodiert.

- Durchlaufe immer wieder den ursprünglichen Eingabebereich.
- Merke bei jedem Durchlauf, ob gerade oder ungerade viele Einsen auf dem Band stehen.
- Ersetze bei jedem Durchlauf jede zweite Eins durch eine Null.
- Wenn bei einem Durchlauf ungerade viele Einsen erkannt wurden, stoppe und lehne die Eingabe ab.
- Ansonsten: akzeptiere die Eingabe, falls am Ende eine 1 stehen bleibt.

$\Rightarrow$ Zeitkomplexität der TM ist *quadratisch* in der Eingabegröße.

$\Rightarrow$ $L[\Pi, s_1]$ liegt in P.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – 1

P

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Zeit** gelöst werden.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – 1

P

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Zeit** gelöst werden.

NP

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Zeit** gelöst werden.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – 1

Karlsruher Institut für Technologie

P

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Zeit** gelöst werden.

NP

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Zeit** gelöst werden.

PSPACE

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Platz** gelöst werden.

Komplexitätsklassen – 1

Karlsruher Institut für Technologie

P

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Zeit** gelöst werden.

NP

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Zeit** gelöst werden.

PSPACE

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Platz** gelöst werden.

NPSPACE

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Platz** gelöst werden.

Komplexitätsklassen – 2

L

Klasse von Entscheidungsproblemen, die von einer **DTM** in **logarithmisch** viel **Platz** gelöst werden.

NL

Klasse von Entscheidungsproblemen, die von einer **NTM** in **logarithmisch** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

EXPSPACE

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Platz** gelöst werden.

Komplexitätsklassen – 3

P

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Zeit** gelöst werden.

NP

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Zeit** gelöst werden.

- $\mathrm{P} \overset{?}{=} \mathrm{NP}$ ist eine der großen offenen Fragen der Informatik.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – 3

PSPACE

Klasse von Entscheidungsproblemen, die von einer **DTM** in **polynomial** viel **Platz** gelöst werden.

NPSPACE

Klasse von Entscheidungsproblemen, die von einer **NTM** in **polynomial** viel **Platz** gelöst werden.

- Satz von Savitch: PSPACE = NPSPACE.

Komplexitätsklassen – 4

Karlsruher Institut für Technologie

PSPACE

Klasse von Entscheidungsproblemen, die von einer TM in **polynomial** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

Komplexitätsklassen – 4

Karlsruher Institut für Technologie

PSPACE

Klasse von Entscheidungsproblemen, die von einer TM in **polynomial** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

Zeige PSPACE $\subseteq$ EXP:

Komplexitätsklassen – 4

PSPACE

Klasse von Entscheidungsproblemen, die von einer TM in **polynomial** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

Zeige PSPACE $\subseteq$ EXP:

- Sei $\mathcal{M}$ eine TM, die ein Entscheidungsproblem Π in polynomial viel Platz entscheidet.

Komplexitätsklassen – 4

PSPACE

Klasse von Entscheidungsproblemen, die von einer TM in **polynomial** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

Zeige PSPACE $\subseteq$ EXP:

- Sei $\mathcal{M}$ eine TM, die ein Entscheidungsproblem Π in polynomial viel Platz entscheidet.
- $\mathcal{M}$ durchläuft höchstens $c(n) = \left((\Sigma \cup \Gamma) \times (Q \dot{\cup} \{\text{„kein Kopf“}\})\right)^{p(n)}$, also exponentiell viele Konfigurationen.
- $\mathcal{M}$ terminiert innerhalb von $c(n)$ Schritten (oder es tritt eine Endlosschleife auf).

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – 4

PSPACE

Klasse von Entscheidungsproblemen, die von einer TM in **polynomial** viel **Platz** gelöst werden.

EXP

Klasse von Entscheidungsproblemen, die von einer **DTM** in **exponentiell** viel **Zeit** gelöst werden.

Zeige PSPACE $\subseteq$ EXP:

- Sei $\mathcal{M}$ eine TM, die ein Entscheidungsproblem Π in polynomial viel Platz entscheidet.
- $\mathcal{M}$ durchläuft höchstens $c(n) = \left((\Sigma \cup \Gamma) \times (Q \dot{\cup} \{\text{„kein Kopf“}\})\right)^{p(n)}$, also exponentiell viele Konfigurationen.
- $\mathcal{M}$ terminiert innerhalb von $c(n)$ Schritten ~~(oder es tritt eine Endlosschleife auf)~~. **$\mathcal{M}$ entscheidet Π!**

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

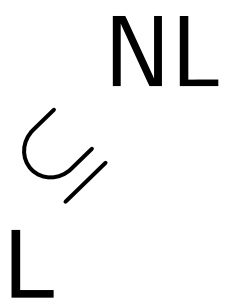
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

L

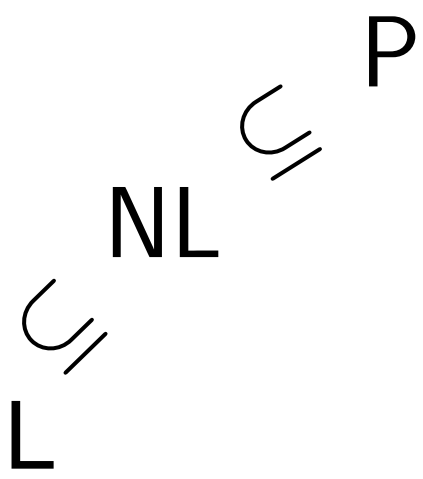
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie



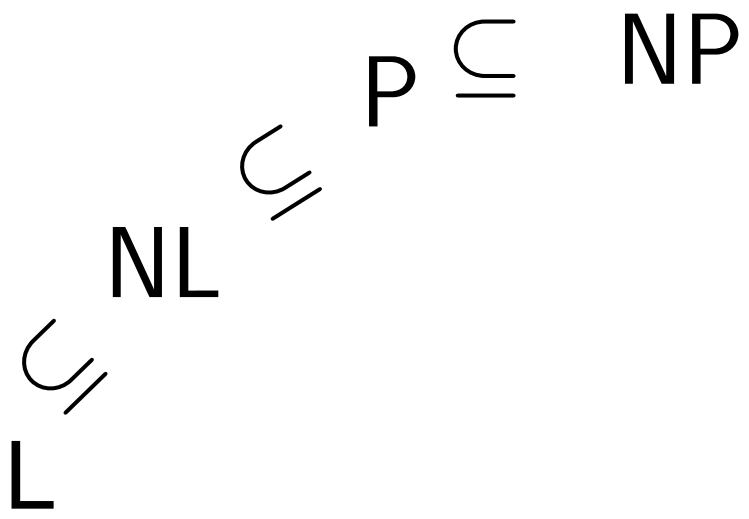
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

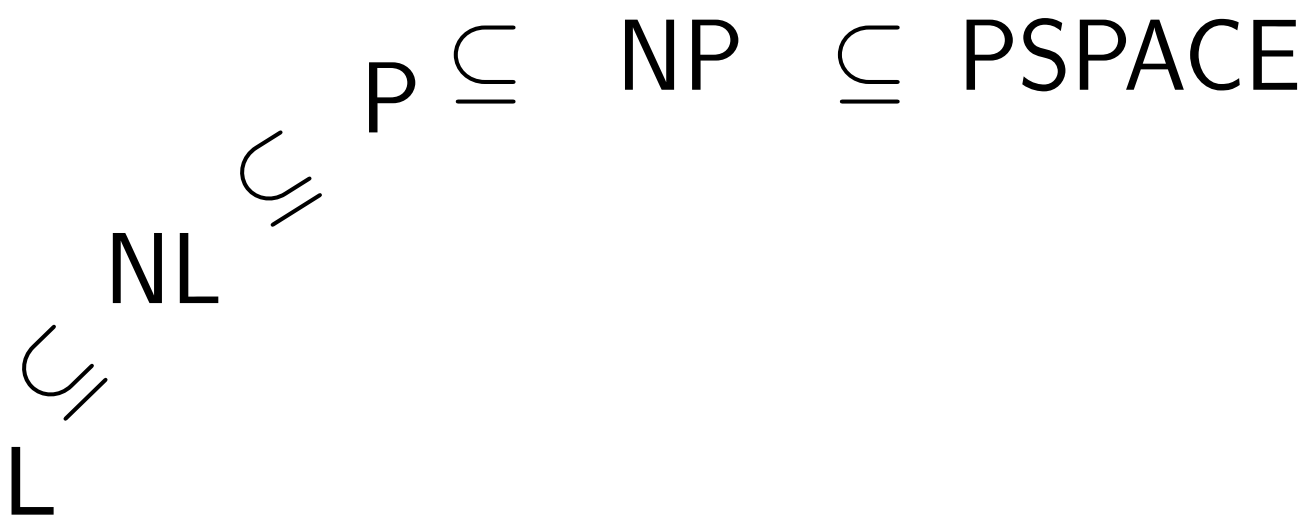


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

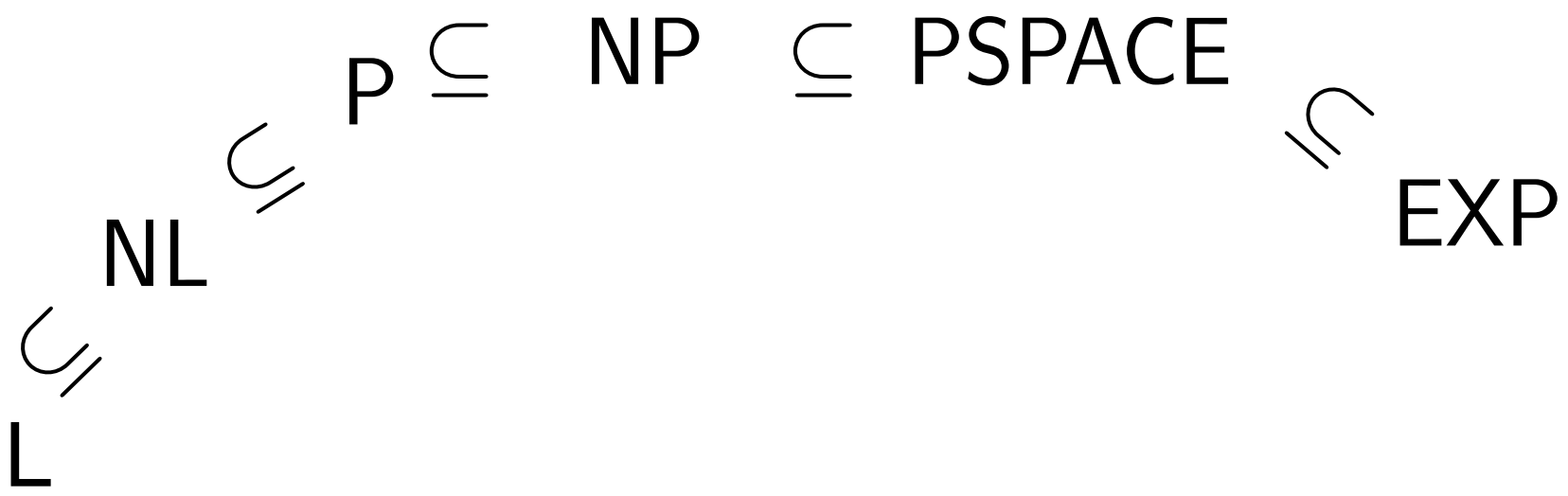


Komplexitätsklassen – Hierarchie



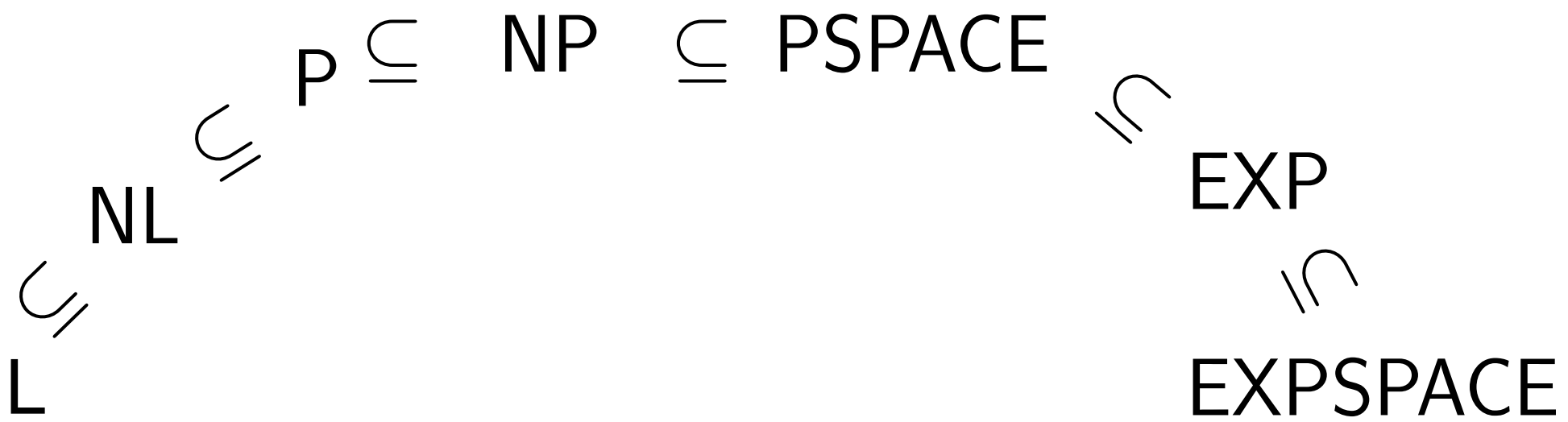
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie



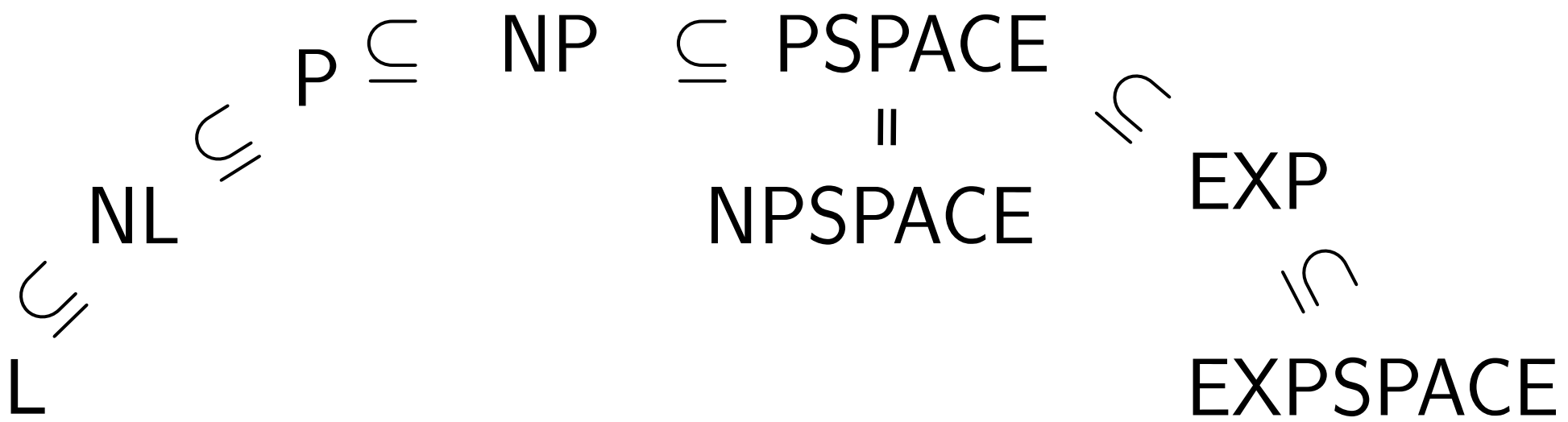
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

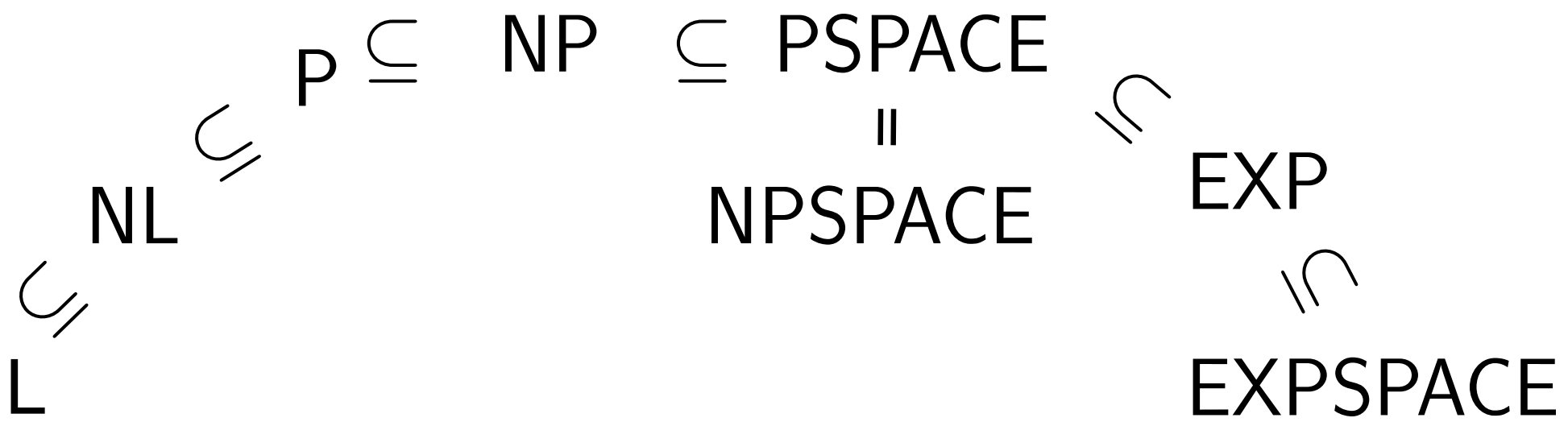
Komplexitätsklassen – Hierarchie



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

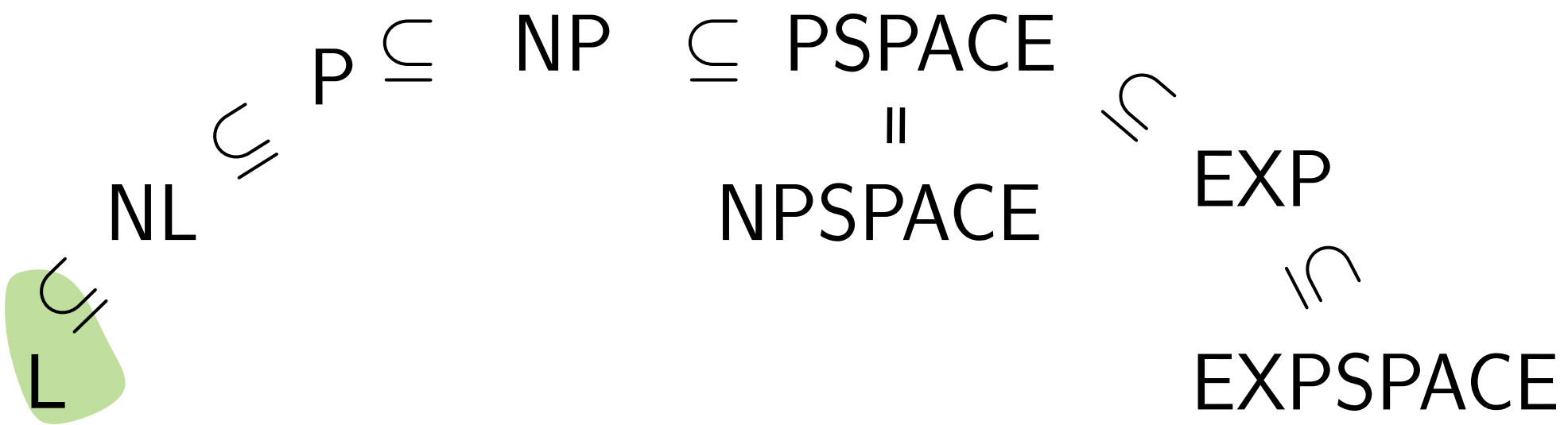
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

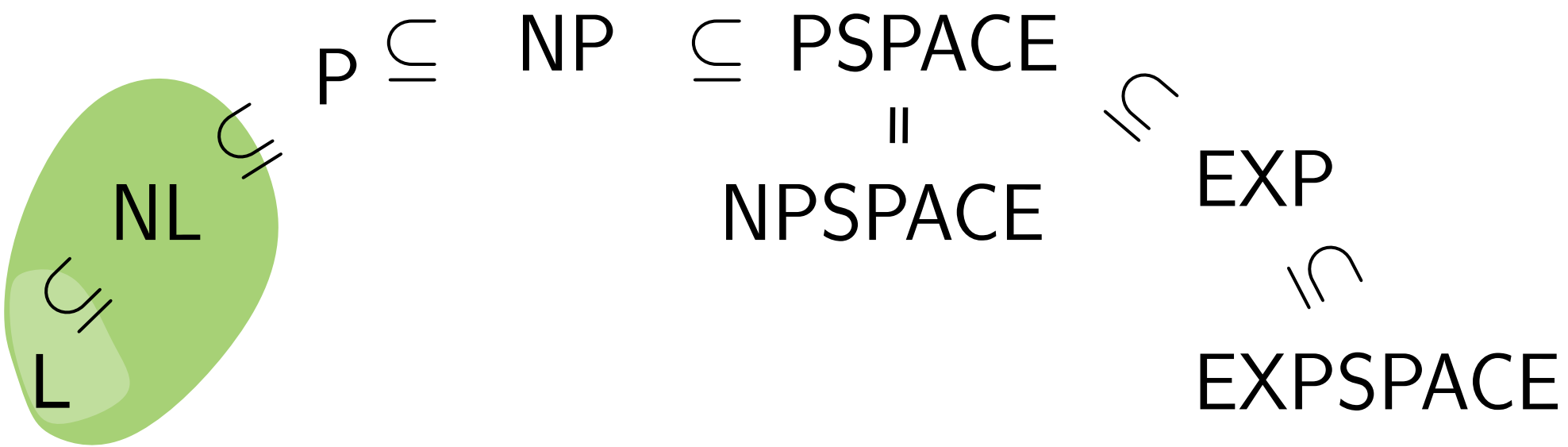
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

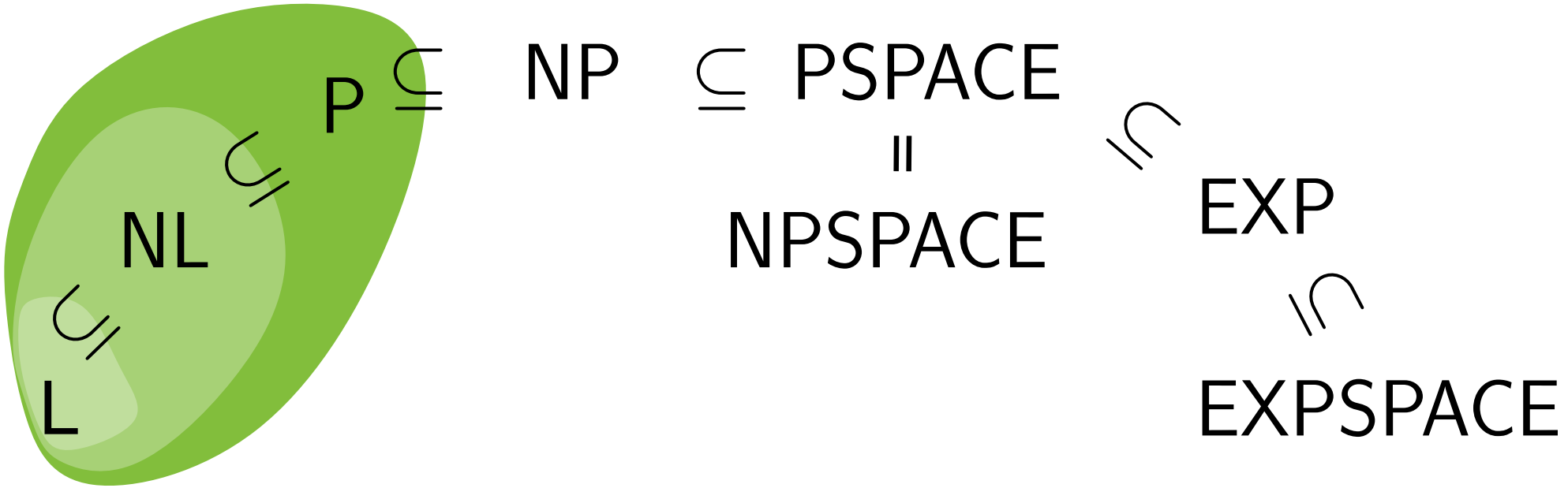
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

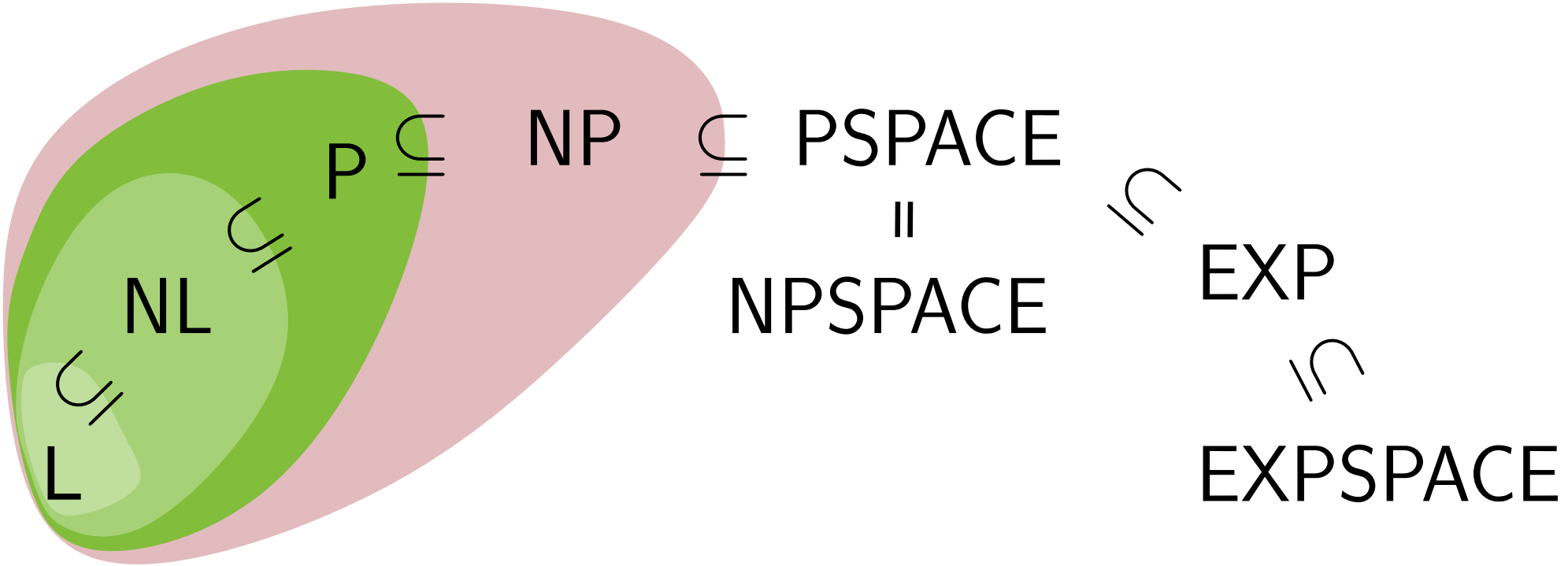
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

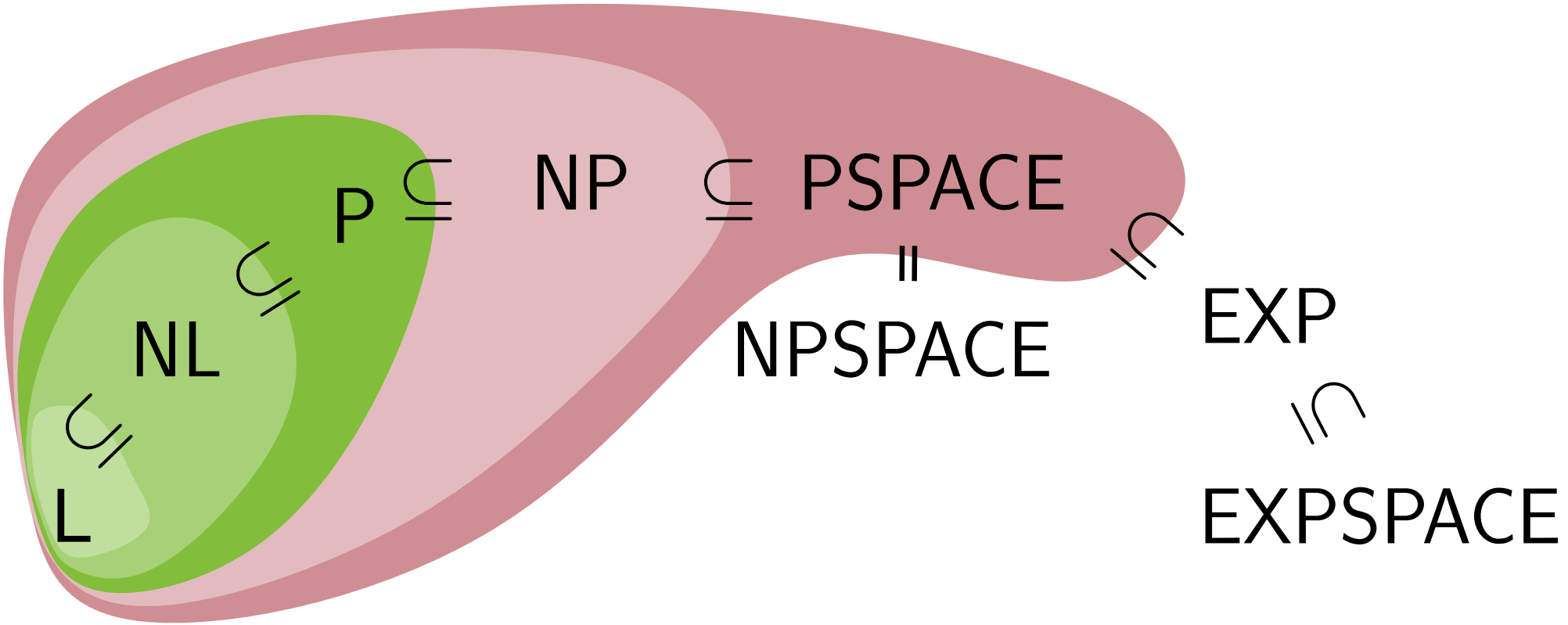
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

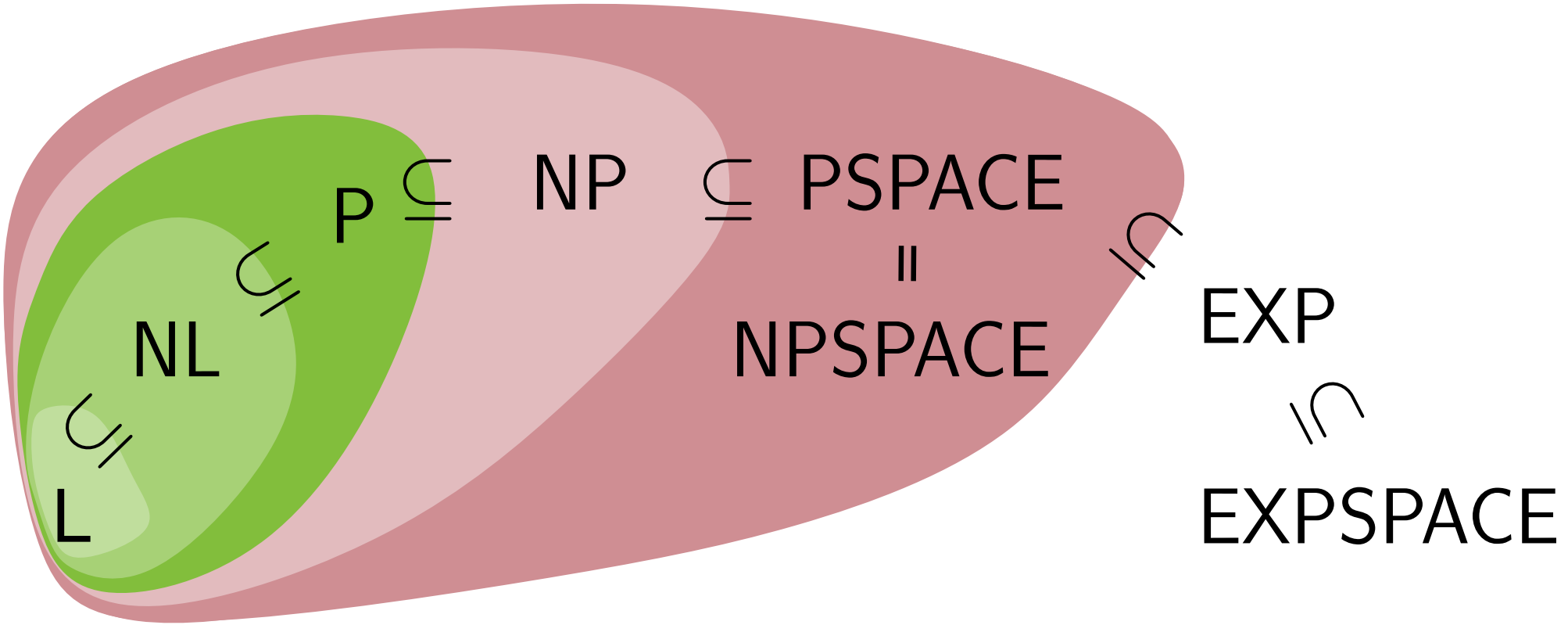
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

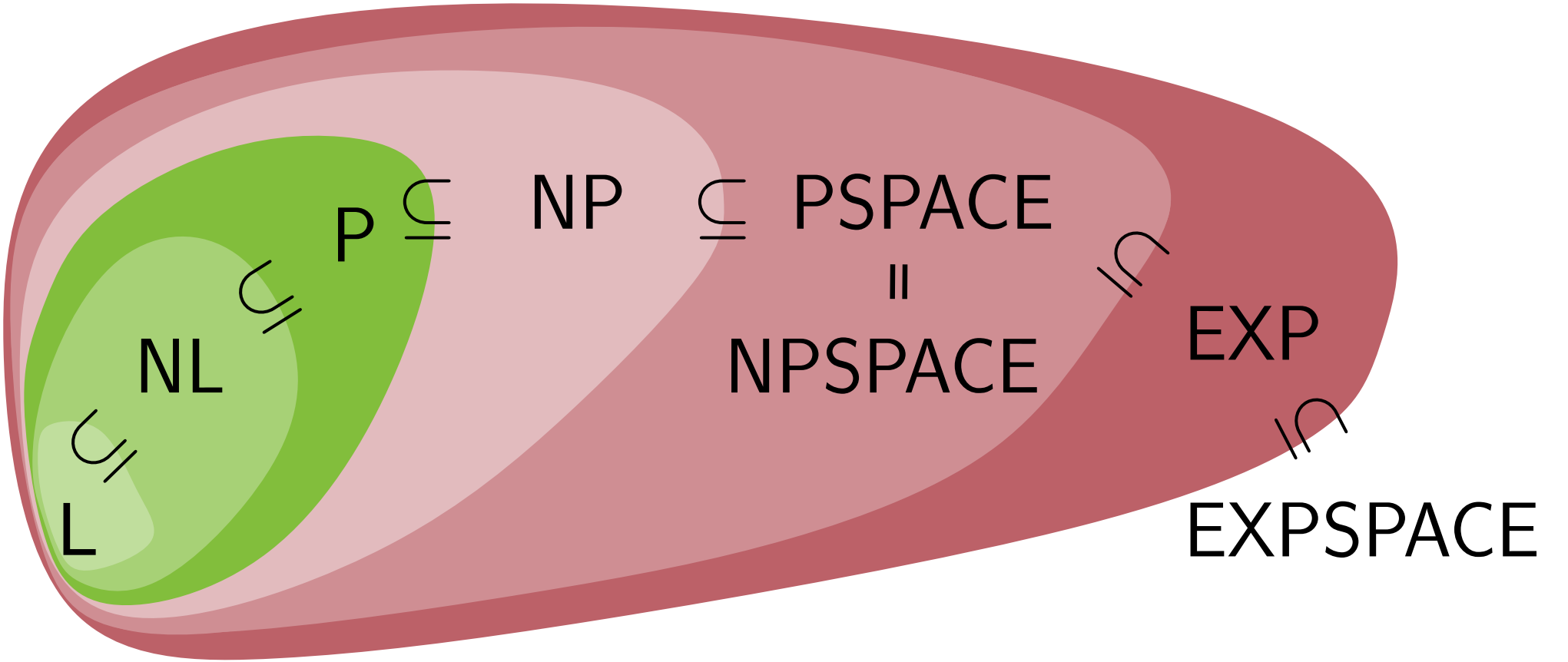
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

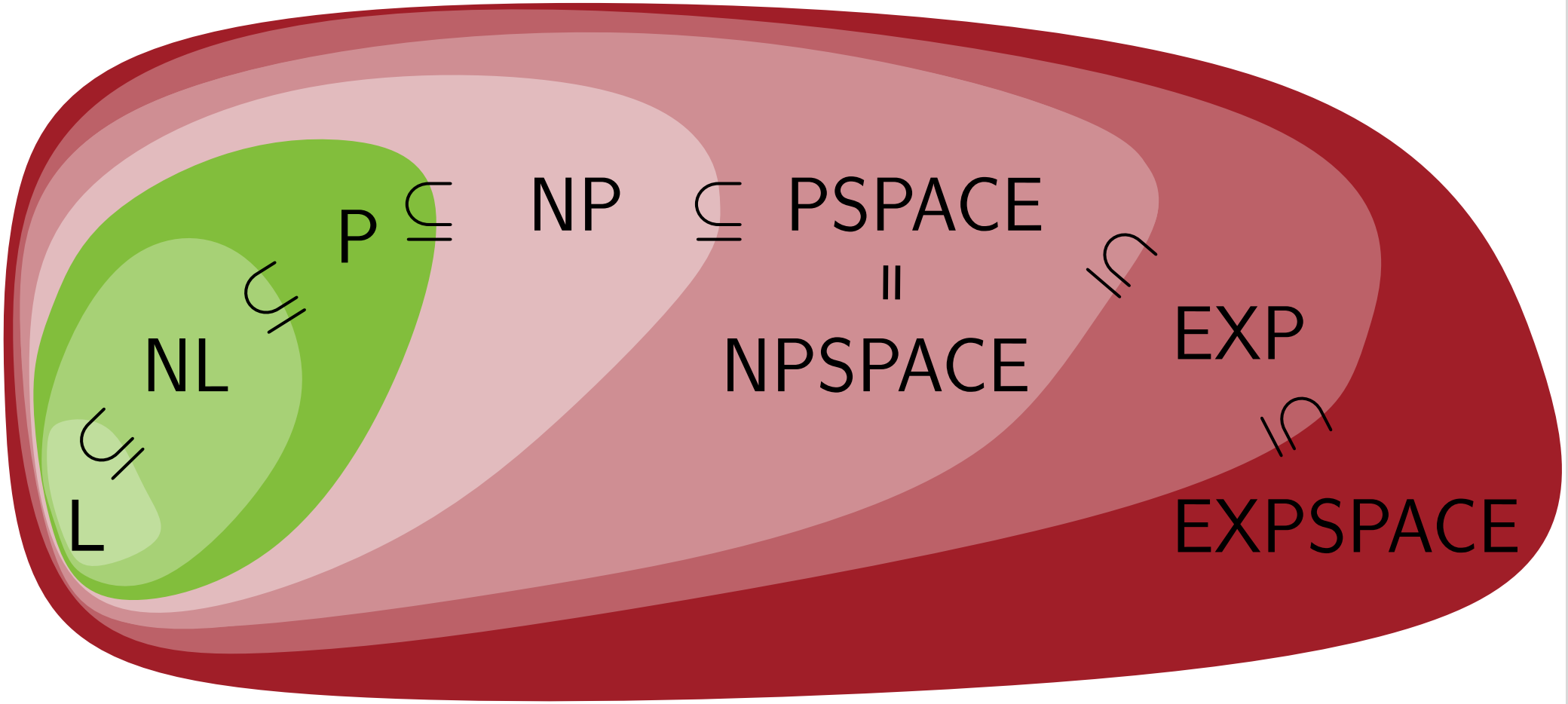
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

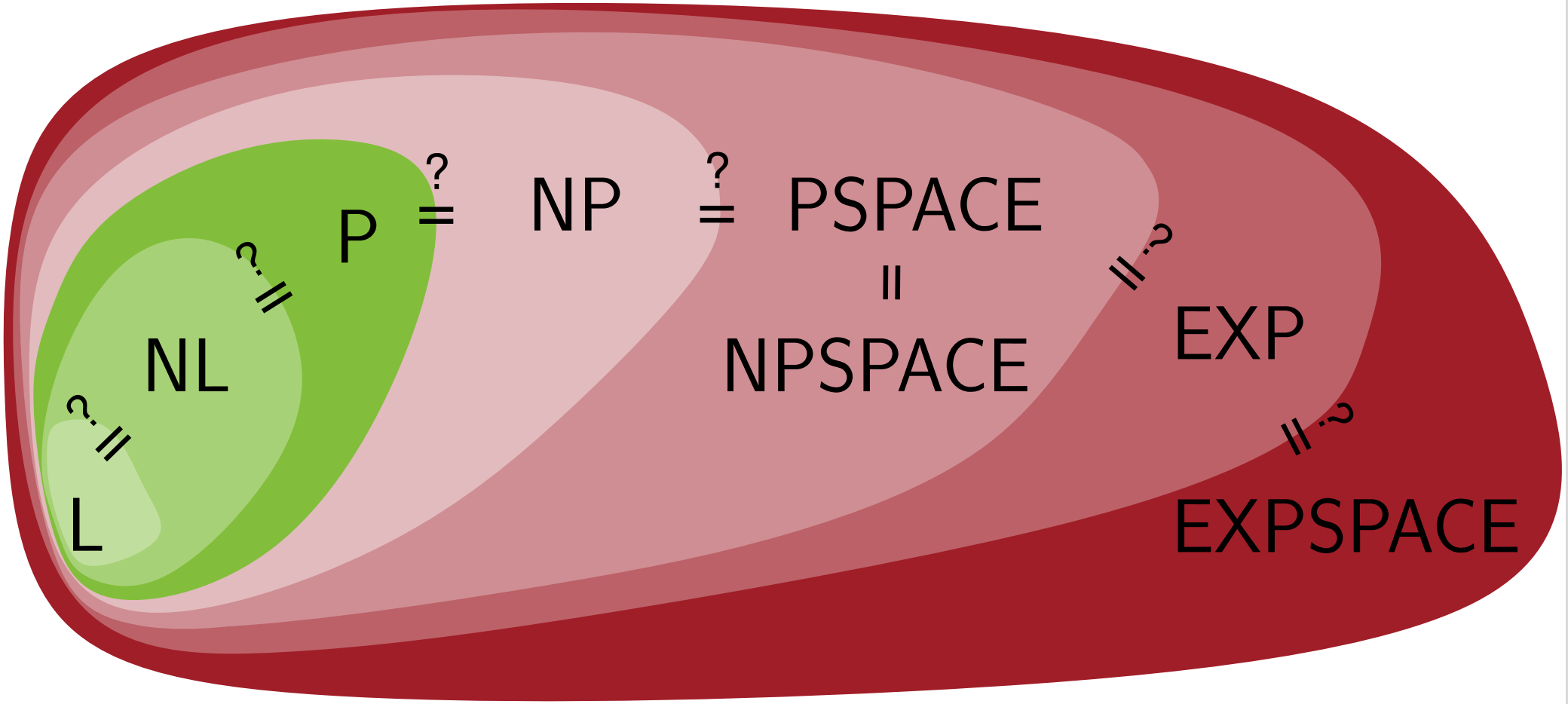
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

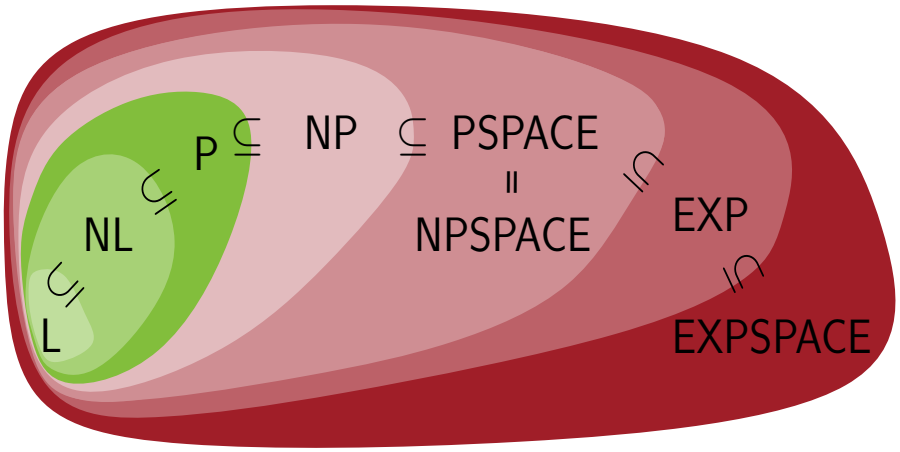
In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

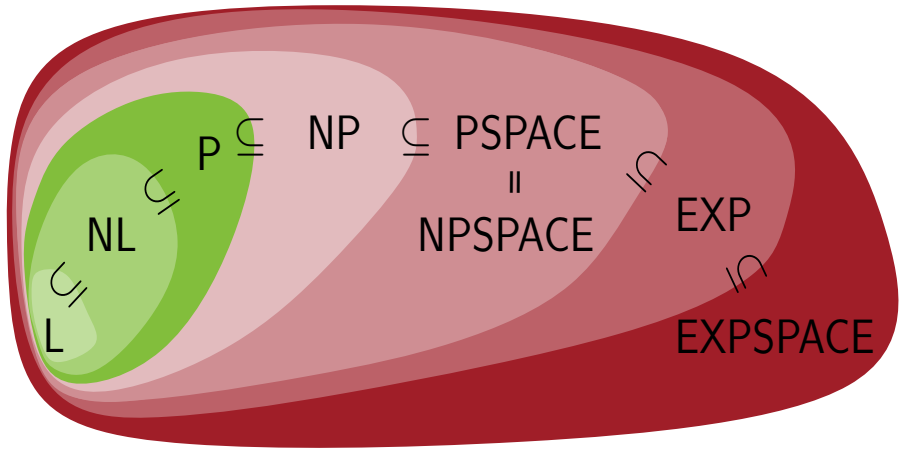


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

Problem des Handlungsreisenden (NP-vollständig)



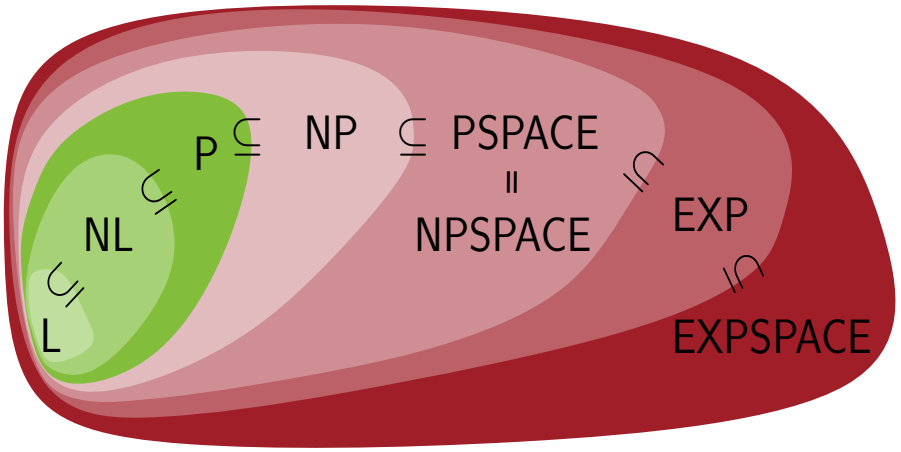
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

Problem des Handlungsreisenden (NP-vollständig)

- Mögliches exaktes Lösungsverfahren: Alle Weglängen aller möglichen Rundreisen berechnen



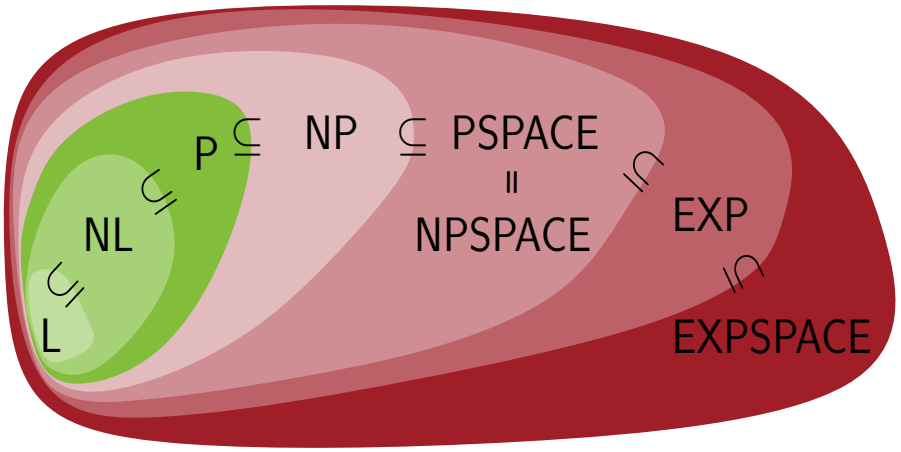
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

Problem des Handlungsreisenden (NP-vollständig)

- Mögliches exaktes Lösungsverfahren: Alle Weglängen aller möglichen Rundreisen berechnen
- Schon bei kleiner Anzahl von Städten unpraktikabel



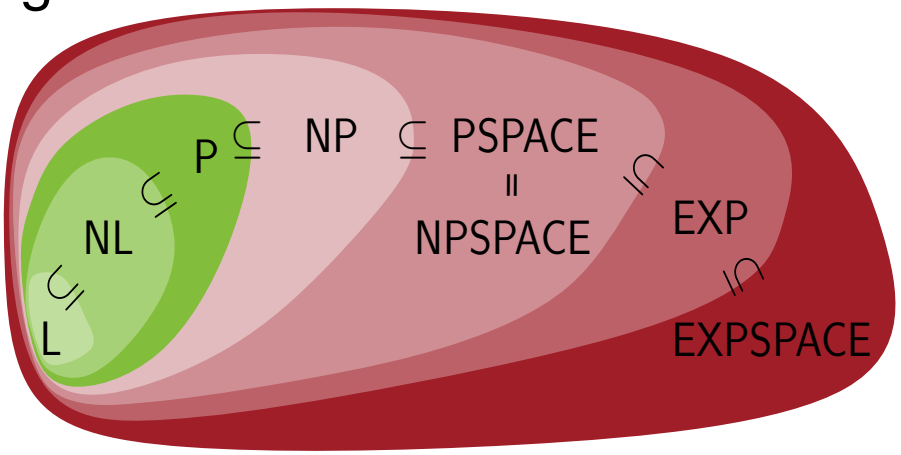
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

Problem des Handlungsreisenden (NP-vollständig)

- Mögliches exaktes Lösungsverfahren: Alle Weglängen aller möglichen Rundreisen berechnen
- Schon bei kleiner Anzahl von Städten unpraktikabel
- Bei n Städten $\rightarrow \frac{(n-1)!}{2}$ verschieden mögliche Rundreisen



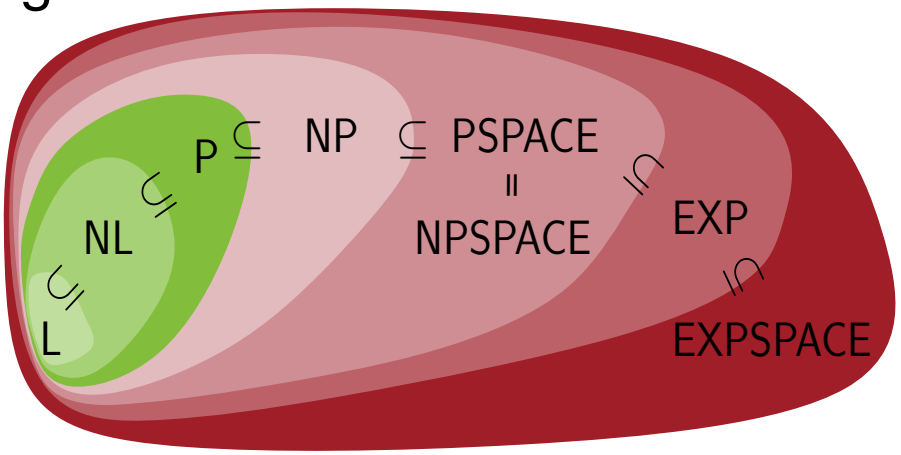
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – Hierarchie

In welchen Komplexitätsklassen können Probleme von heutigen Computern mit realistisch viel Rechenzeit und Speicherplatz gelöst werden?

Problem des Handlungsreisenden (NP-vollständig)

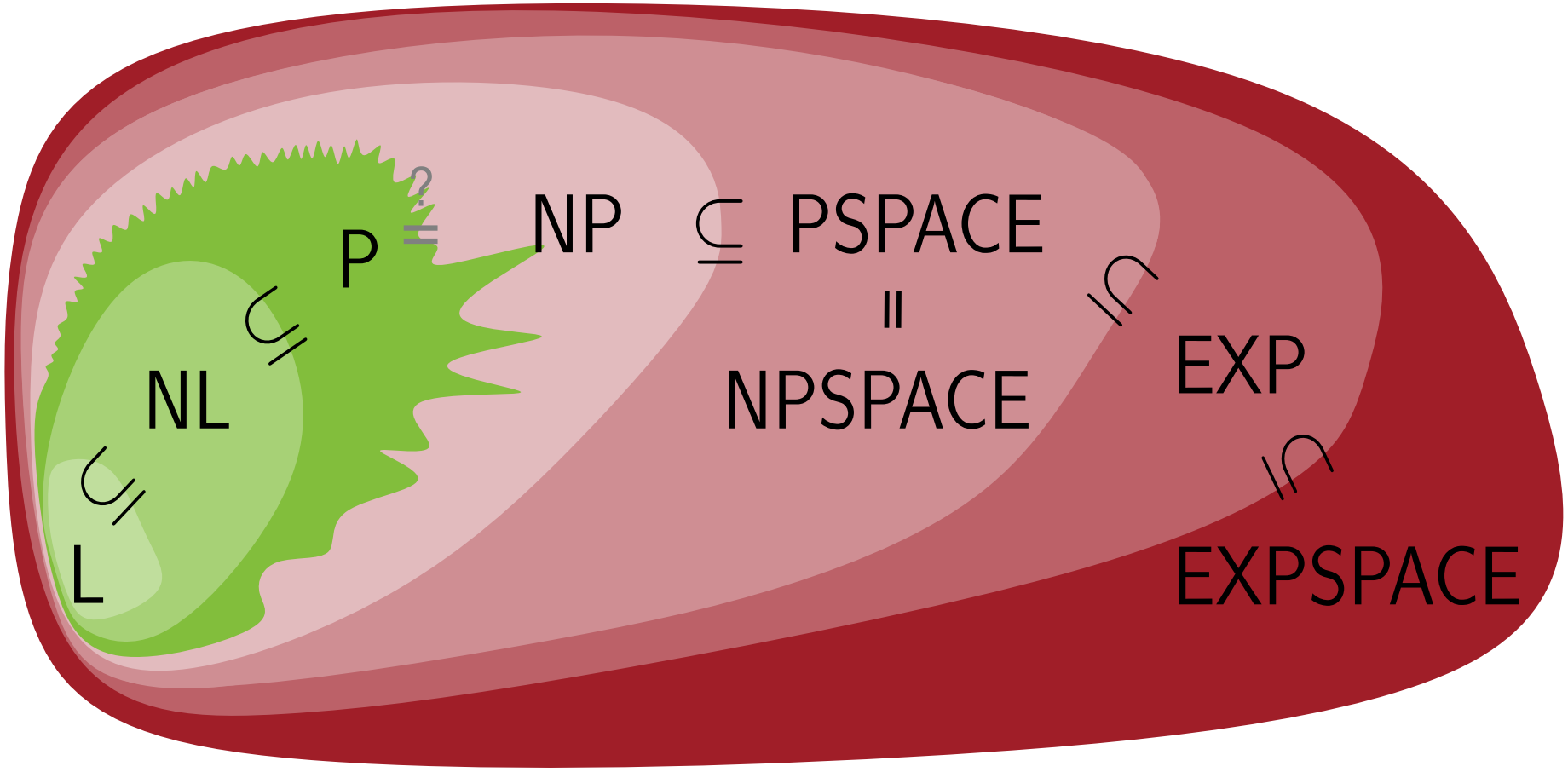
- Mögliches exaktes Lösungsverfahren: Alle Weglängen aller möglichen Rundreisen berechnen
- Schon bei kleiner Anzahl von Städten unpraktikabel
- Bei n Städten $\rightarrow \frac{(n-1)!}{2}$ verschieden mögliche Rundreisen
- Für 16 Städte $\rightarrow$ 653 Mrd. Rundreisen



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme
- Können Probleme in NP genauso effizient gelöst werden wie in P?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme
- Können Probleme in NP genauso effizient gelöst werden wie in P?
- Wenn der Beweis nicht konstruktiv ist, weiß man die Auswirkung nicht

Komplexitätsklassen – P und NP

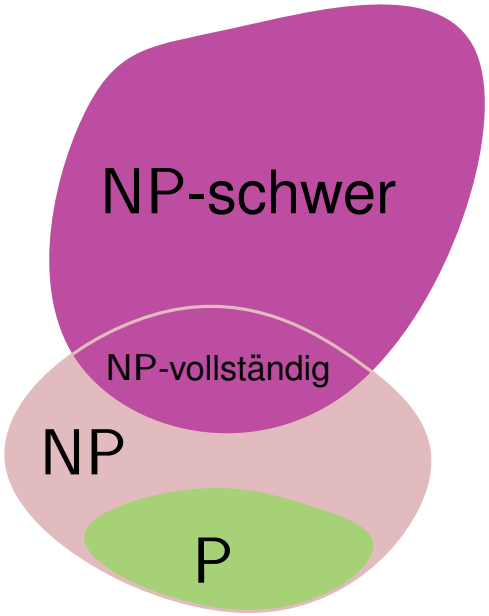
$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme
- Können Probleme in NP genauso effizient gelöst werden wie in P?
- Wenn der Beweis nicht konstruktiv ist, weiß man die Auswirkung nicht
- Viele Probleme sind NP-vollständig

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme
- Können Probleme in NP genauso effizient gelöst werden wie in P?
- Wenn der Beweis nicht konstruktiv ist, weiß man die Auswirkung nicht
- Viele Probleme sind NP-vollständig

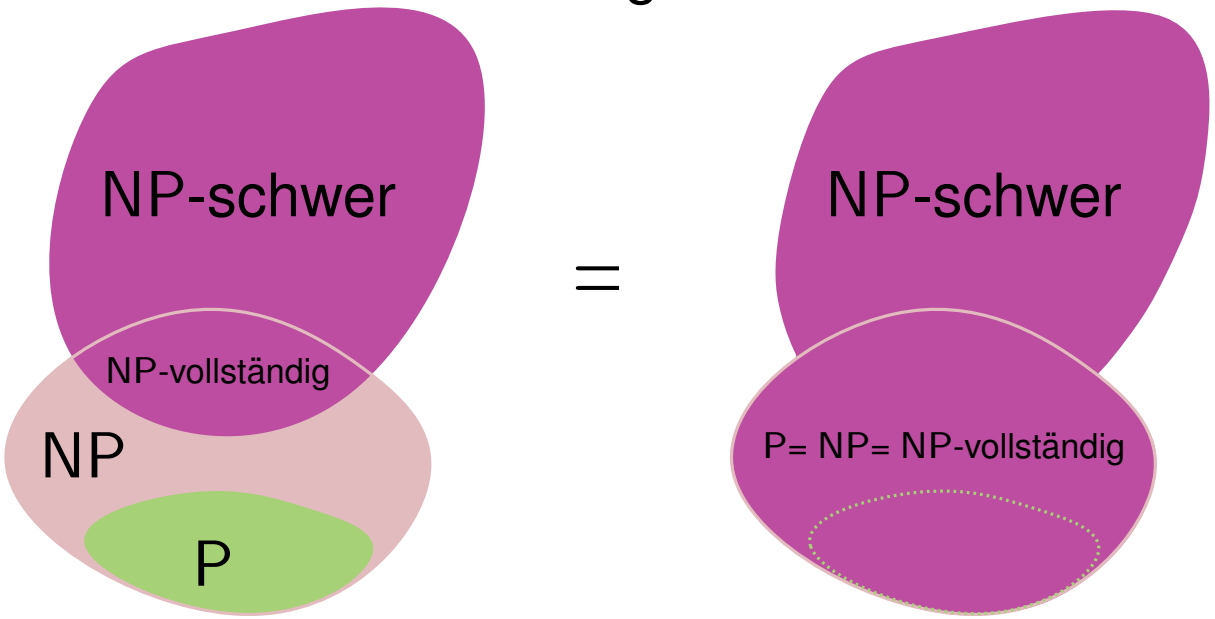


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

- Eines der Millennium-Probleme
- Können Probleme in NP genauso effizient gelöst werden wie in P?
- Wenn der Beweis nicht konstruktiv ist, weiß man die Auswirkung nicht
- Viele Probleme sind NP-vollständig



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Aber Achtung:

- Es gilt nicht: „Kryptographie ist möglich $\iff$ $P \neq NP$“

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Aber Achtung:

- Es gilt nicht: „Kryptographie ist möglich $\iff$ $P \neq NP$“
- Bei P und NP geht es um asymptotisches Laufzeitverhalten!

  → aber z.B. bei AES: Schlüsselgröße 256 Bit, Blockgröße 128, also feste Instanzgröße

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Aber Achtung:

- Es gilt nicht: „Kryptographie ist möglich $\iff$ P $\neq$ NP“
- Bei P und NP geht es um asymptotisches Laufzeitverhalten!
- worst-case-Laufzeit vs. average-case-Laufzeit

  → Damit ein Problem NP-vollständig ist, reicht es, wenn manche Instanzen schwierig zu lösen sind. Es sollten aber alle verschlüsselten Nachrichten schwierig zu entschlüsseln sein!

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Aber Achtung:

- Es gilt nicht: „Kryptographie ist möglich $\iff$ P $\neq$ NP“
- Bei P und NP geht es um asymptotisches Laufzeitverhalten!
- worst-case-Laufzeit vs. average-case-Laufzeit
- Viele als schwierig angenommene kryptographische Probleme sind nicht als NP-vollständig bekannt, z.B. Ganzzahlfaktorisierung.

Komplexitätsklassen – P und NP

$$P \stackrel{?}{=} NP$$

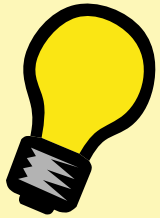
Beispiel Kryptographie:

- Aus P = NP würde folgen, dass einige kryptographische Primitive nicht existieren, z.B. Einwegfunktionen.

Aber Achtung:

- komplizierte und nuancierte Situation
- Fragestellungen weit über $P \stackrel{?}{=} NP$ hinaus
- bei Interesse:

  → Vorlesung „Komplexitätstheorie, mit Anwendungen in der Kryptographie“



Institut für Theoretische Informatik
Lehrstuhl Algorithmik