Karlsruher Institut für Technologie

Theoretische Grundlagen der Informatik Übung

3. Übungstermin · 20. November 2018

Jonas Sauer

INSTITUT FÜR THEORETISCHE INFORMATIK · LEHRSTUHL ALGORITHMIK

Gliederung

Inhalt

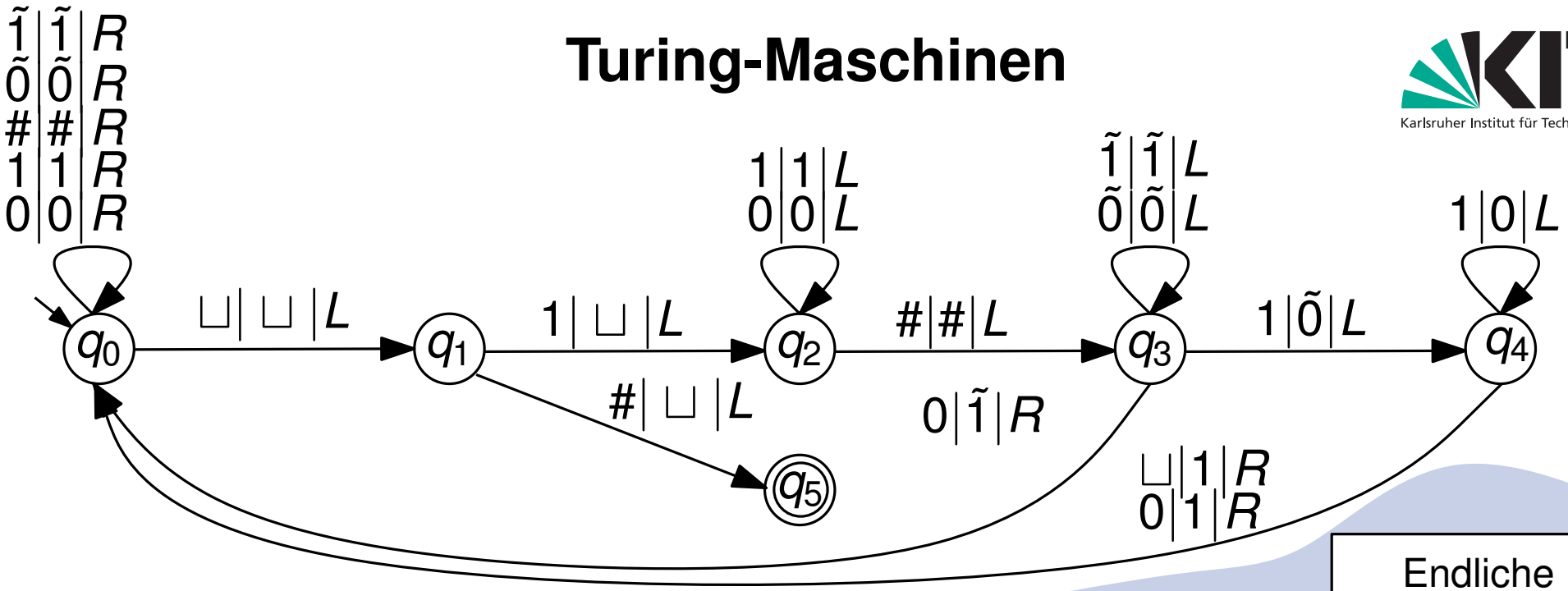
- **Turing-Maschinen**
  - Erweiterungen
- **Entscheidbarkeit**
  - Beispiele
  - Werkzeugkasten

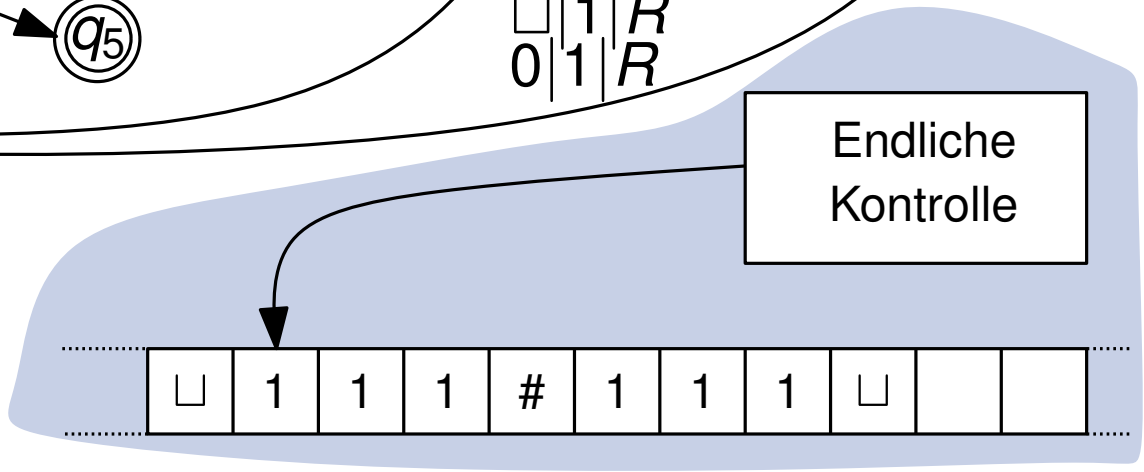
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

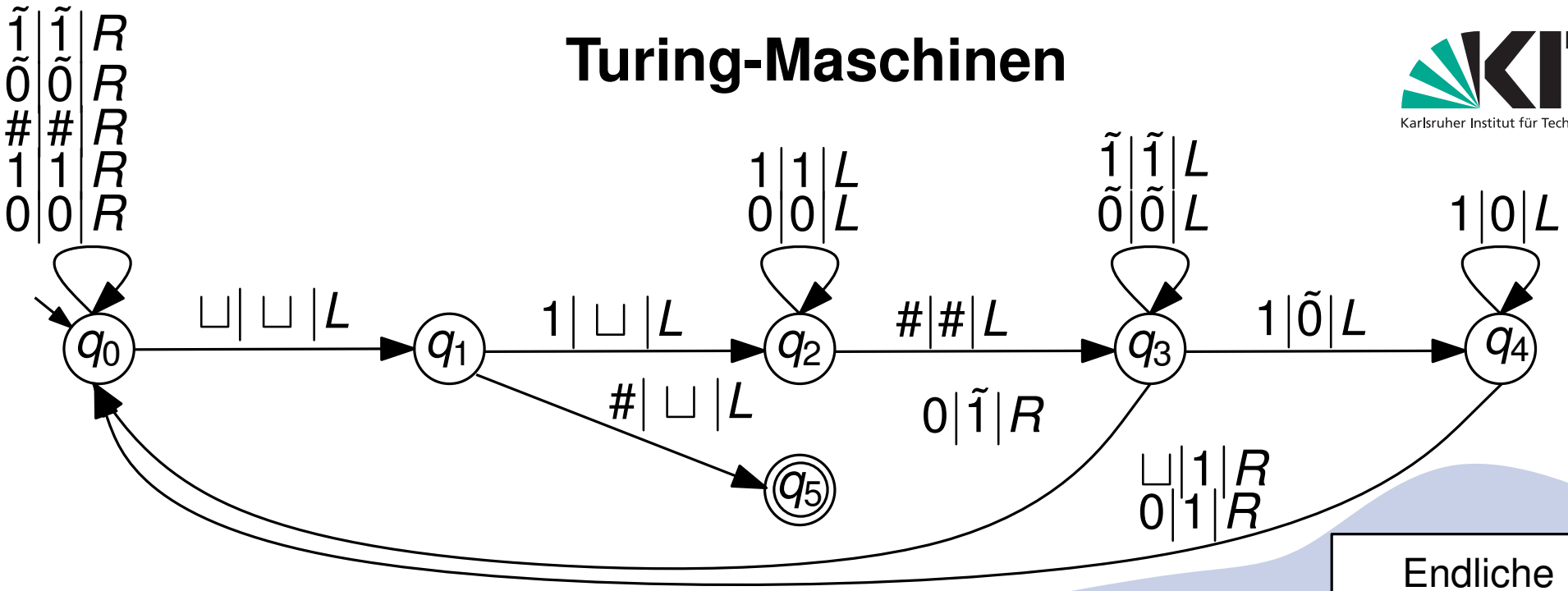


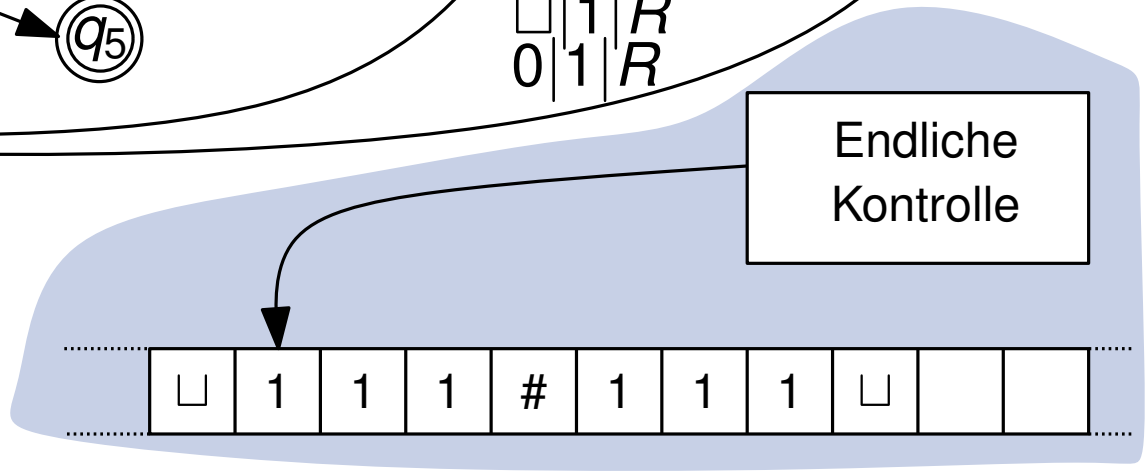


- Q, endliche Zustandsmenge,
- Σ, endliches Eingabealphabet,
- $\sqcup$, Blanksymbol mit $\sqcup \notin \Sigma$,
- Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$,
- $s \in Q$, Startzustand,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, Übergangsfunktion,
- $F \subseteq Q$, Menge von Endzuständen

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

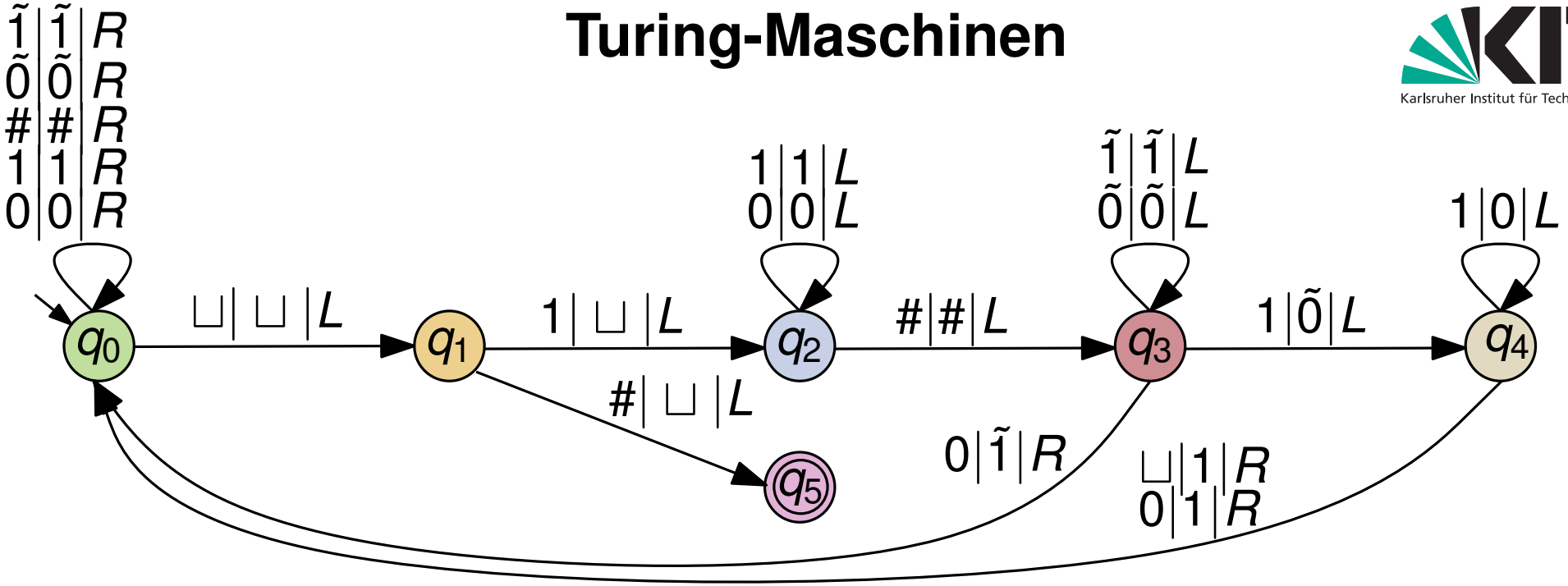




- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \Sigma \cup \{\sqcup, \tilde{0}, \tilde{1}\}$,
- Startzustand q_0,
- Übergangsfunktion $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$
- Endzustand $q_5 \in Q$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

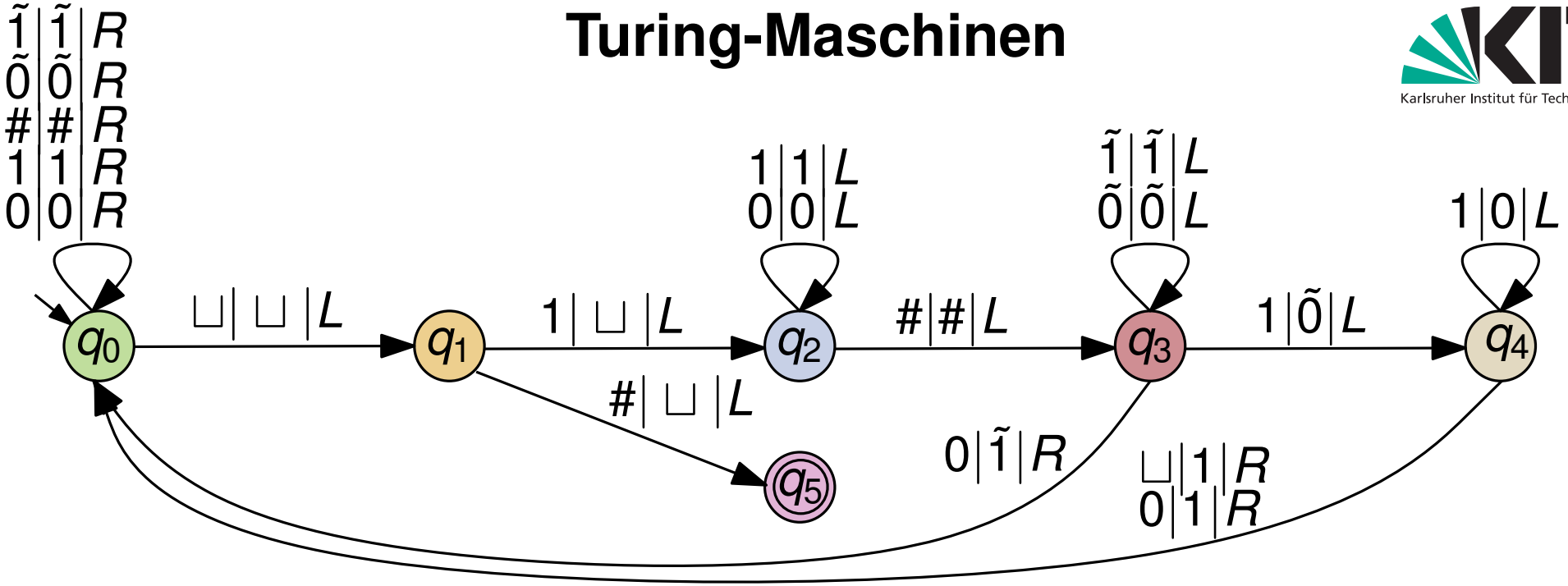
Turing-Maschinen



␣(q0)111#111␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

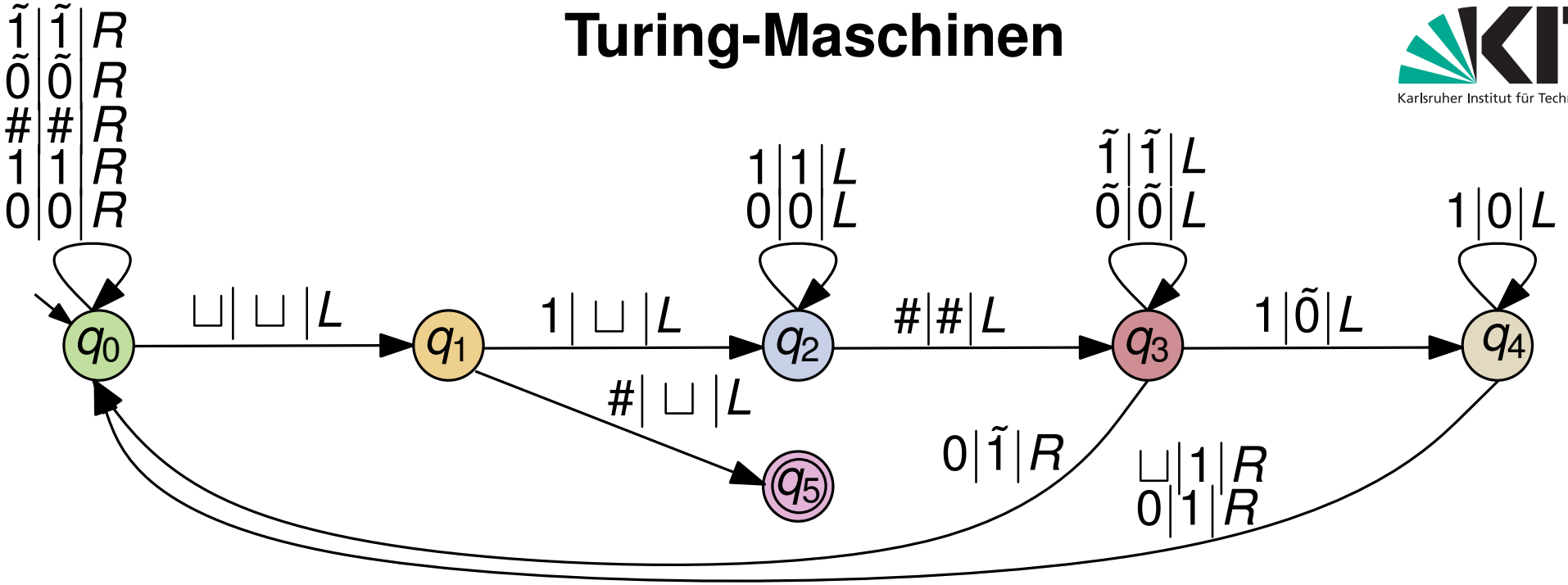


␣(q_0)111#111␣

␣1(q_0)11#111␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



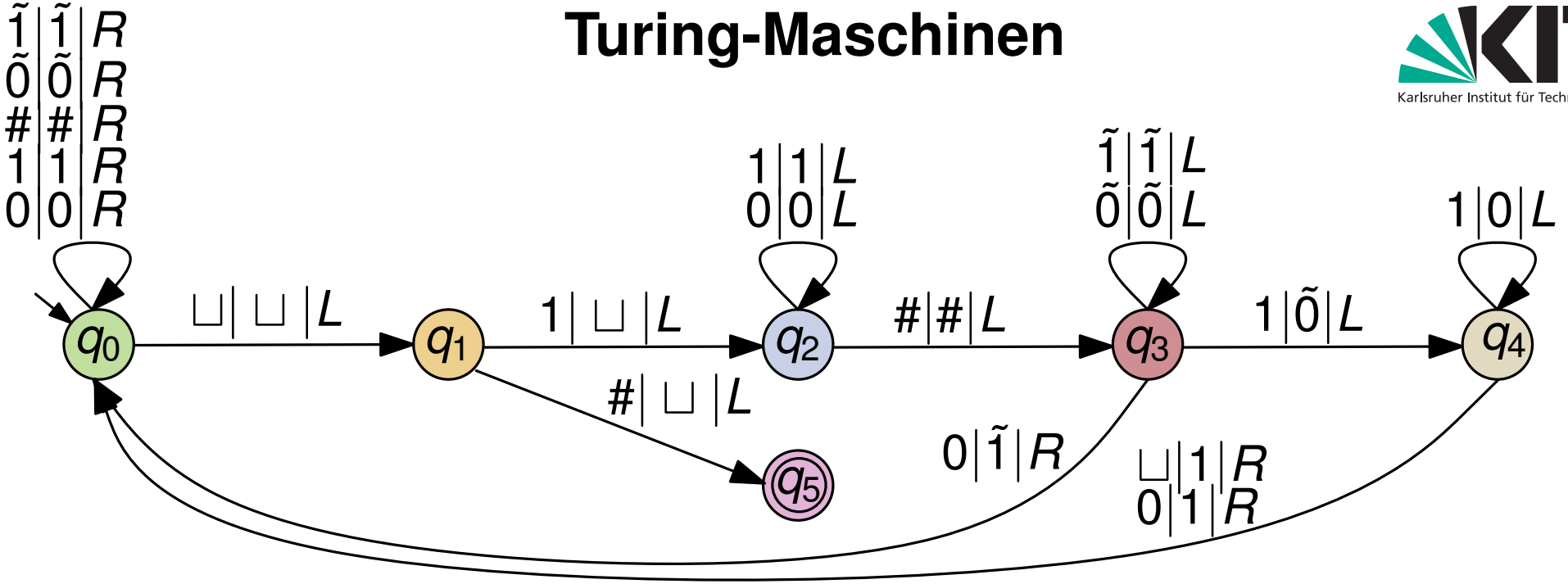
␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

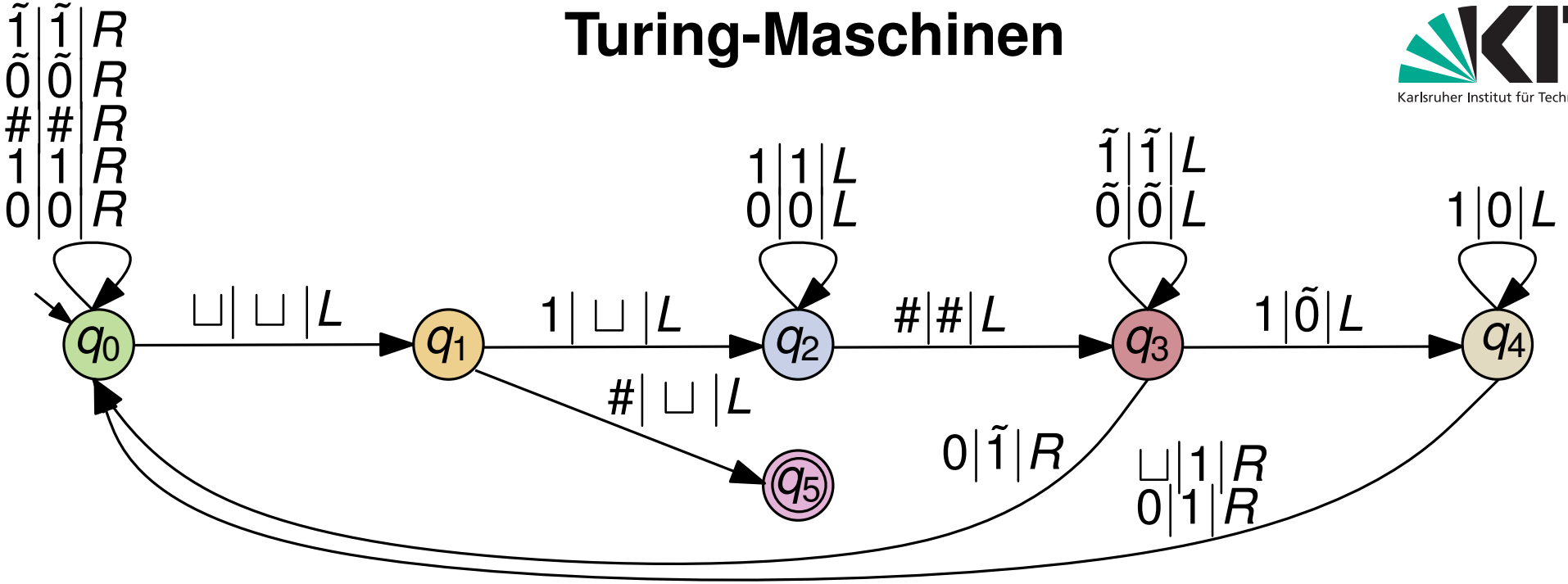
␣11(q_0)1#111␣

. . .

␣111#111(q_0)␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



$\sqcup(q_0)111\#111\sqcup$

$\sqcup1(q_0)11\#111\sqcup$

$\sqcup11(q_0)1\#111\sqcup$

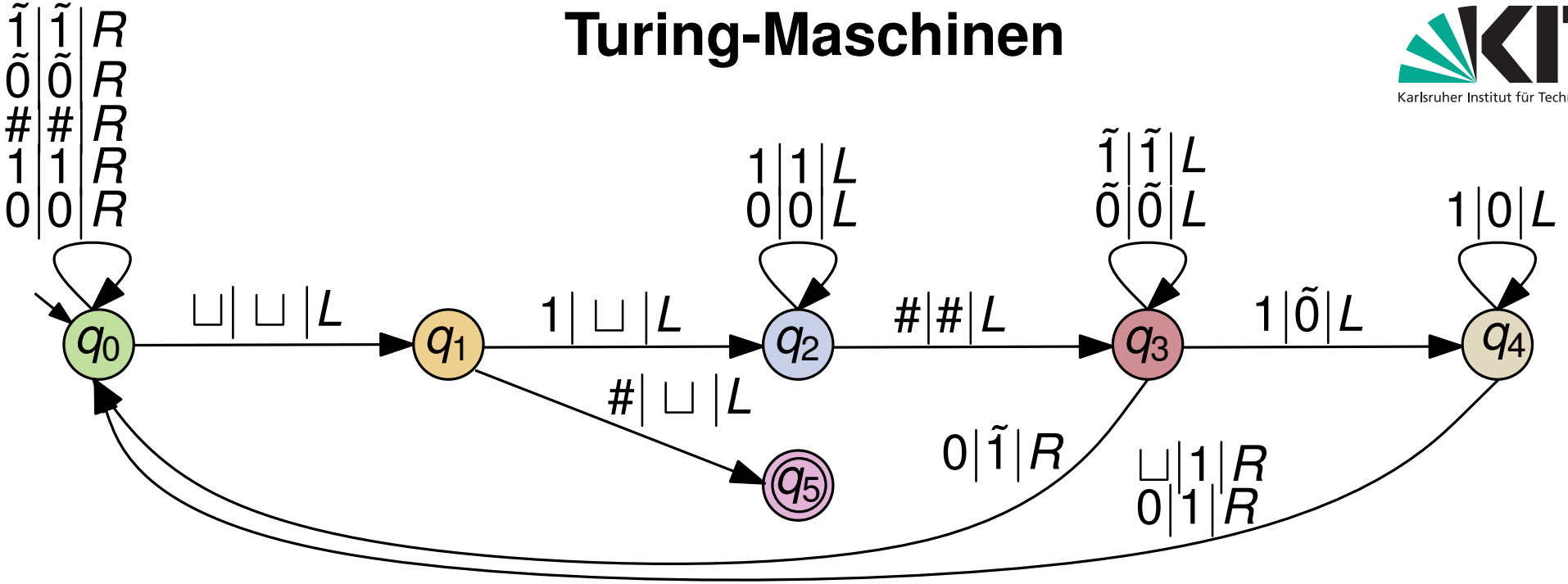
...

$\sqcup111\#111(q_0)\sqcup$

$\sqcup111\#11(q_1)1\sqcup$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



$\sqcup(q_0)111\#111\sqcup$

$\sqcup1(q_0)11\#111\sqcup$

$\sqcup11(q_0)1\#111\sqcup$

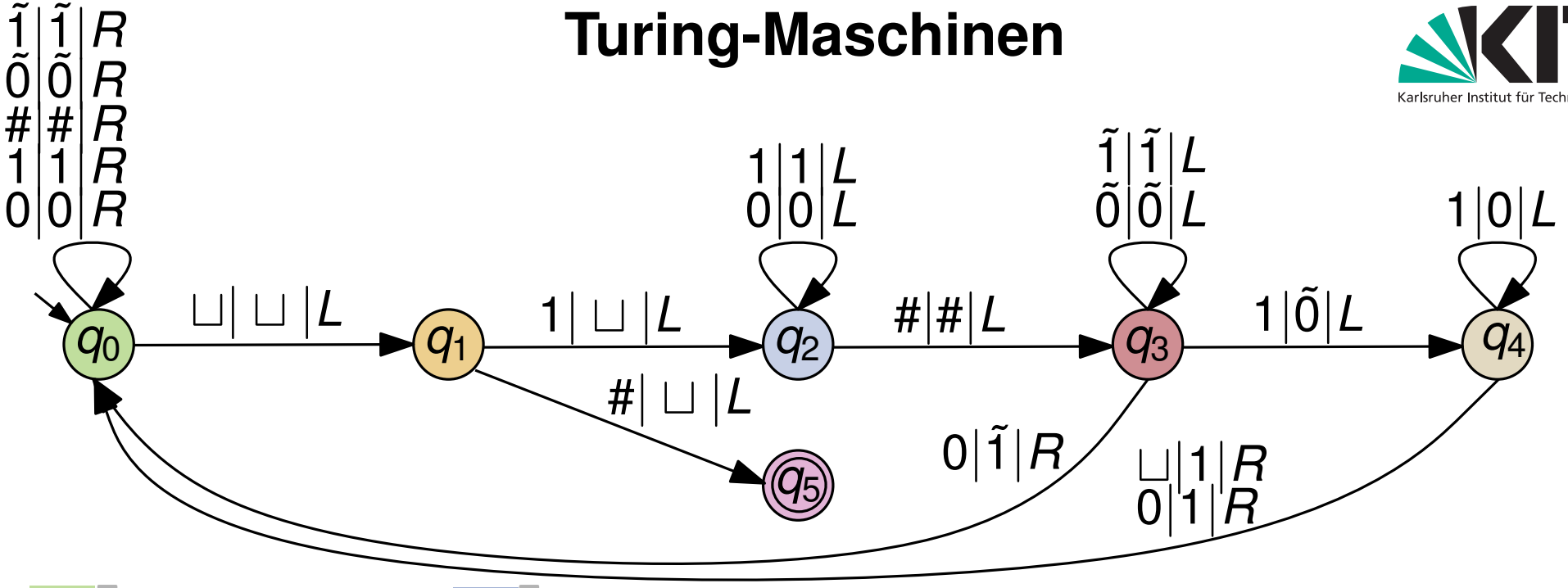
...

$\sqcup111\#111(q_0)\sqcup$

$\sqcup111\#11(q_1)1\sqcup$

$\sqcup111\#1(q_2)1\sqcup$

Turing-Maschinen



$\llcorner(q_0)111\#111\llcorner$ $\llcorner111\#(q_2)11\llcorner$

$\llcorner1(q_0)11\#111\llcorner$

$\llcorner11(q_0)1\#111\llcorner$

...

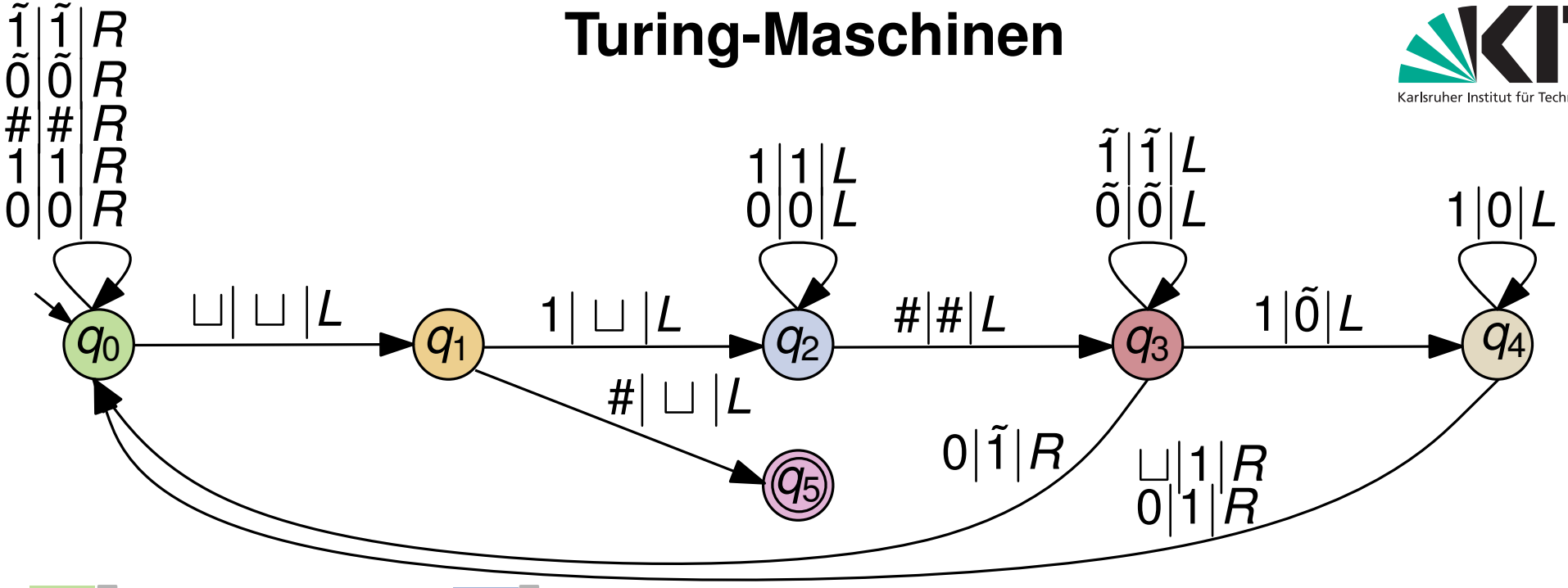
$\llcorner111\#111(q_0)\llcorner$

$\llcorner111\#11(q_1)1\llcorner$

$\llcorner111\#1(q_2)1\llcorner$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



$\sqcup(q_0)111\#111\sqcup$ $\quad$ $\sqcup111\#(q_2)11\sqcup$

$\sqcup1(q_0)11\#111\sqcup$ $\quad$ $\sqcup111(q_2)\#11\sqcup$

$\sqcup11(q_0)1\#111\sqcup$

...

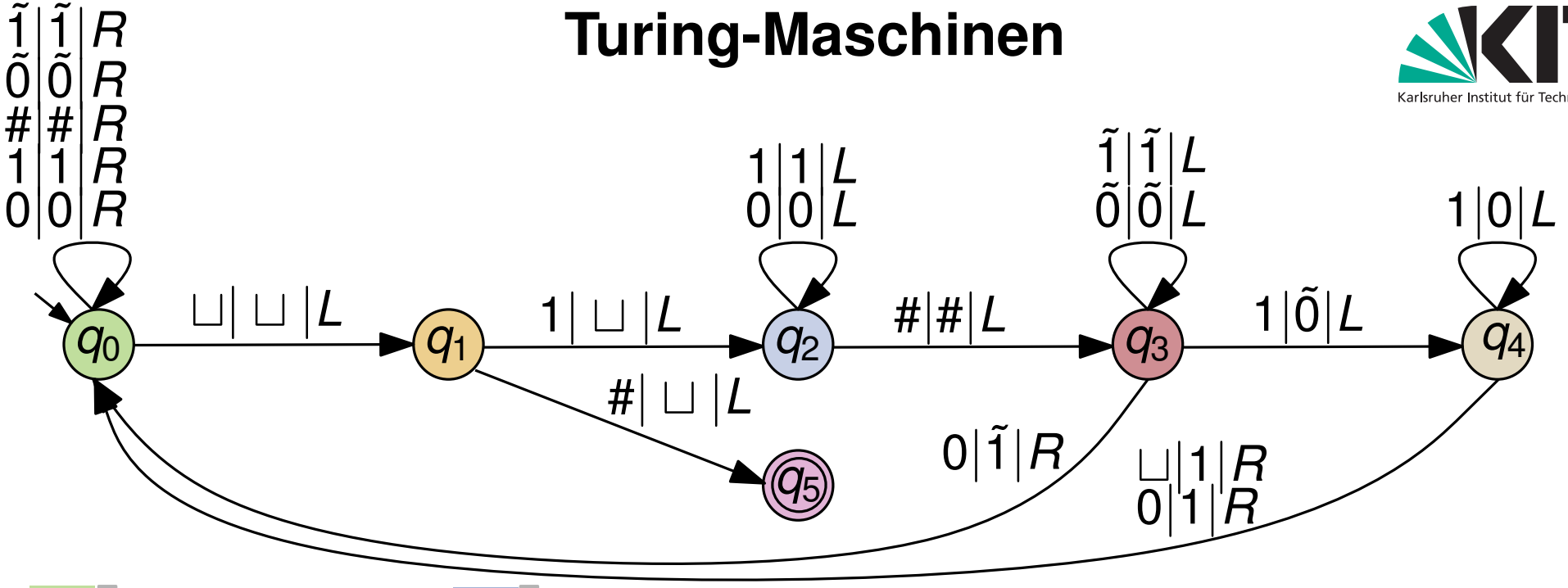
$\sqcup111\#111(q_0)\sqcup$

$\sqcup111\#11(q_1)1\sqcup$

$\sqcup111\#1(q_2)1\sqcup$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



␣(q0)111#111␣ ␣111#(q2)11␣

␣1(q0)11#111␣ ␣111(q2)#11␣

␣11(q0)1#111␣ ␣11(q3)1#11␣

. . .

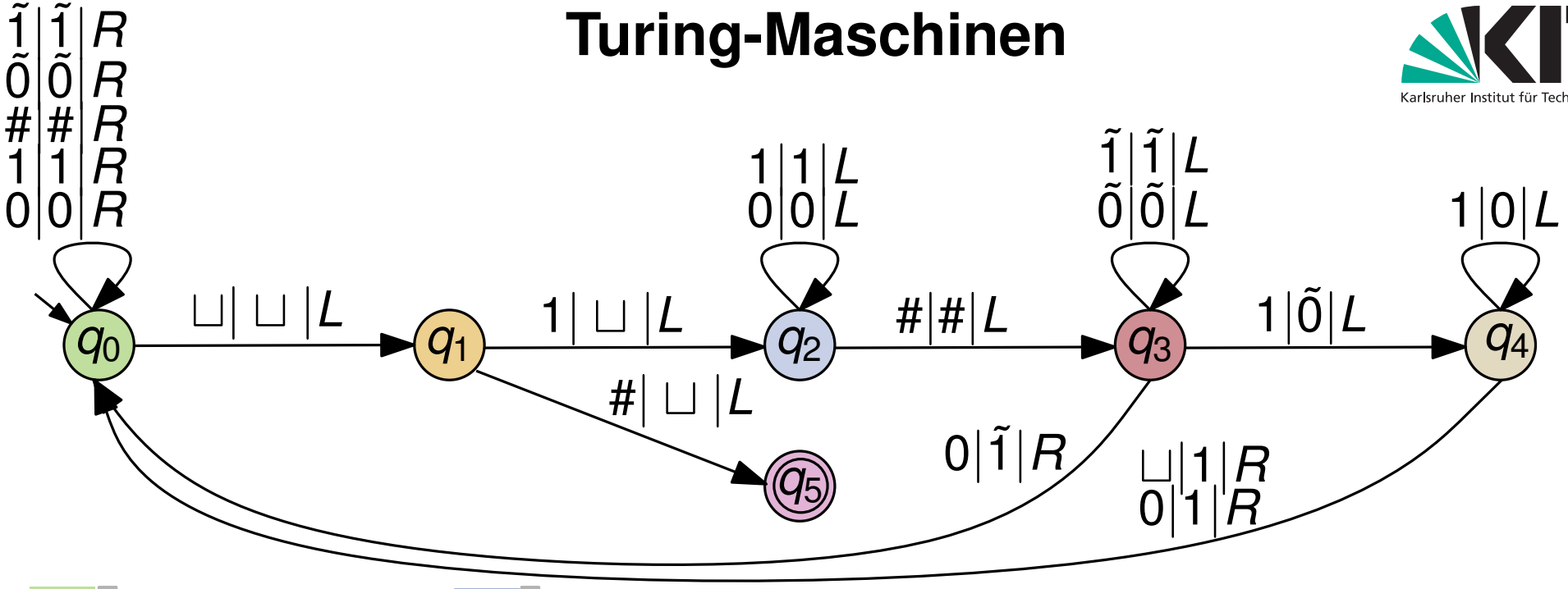
␣111#111(q0)␣

␣111#11(q1)1␣

␣111#1(q2)1␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

. . .

␣111#111(q_0)␣

␣111#11(q_1)1␣

␣111#1(q_2)1␣

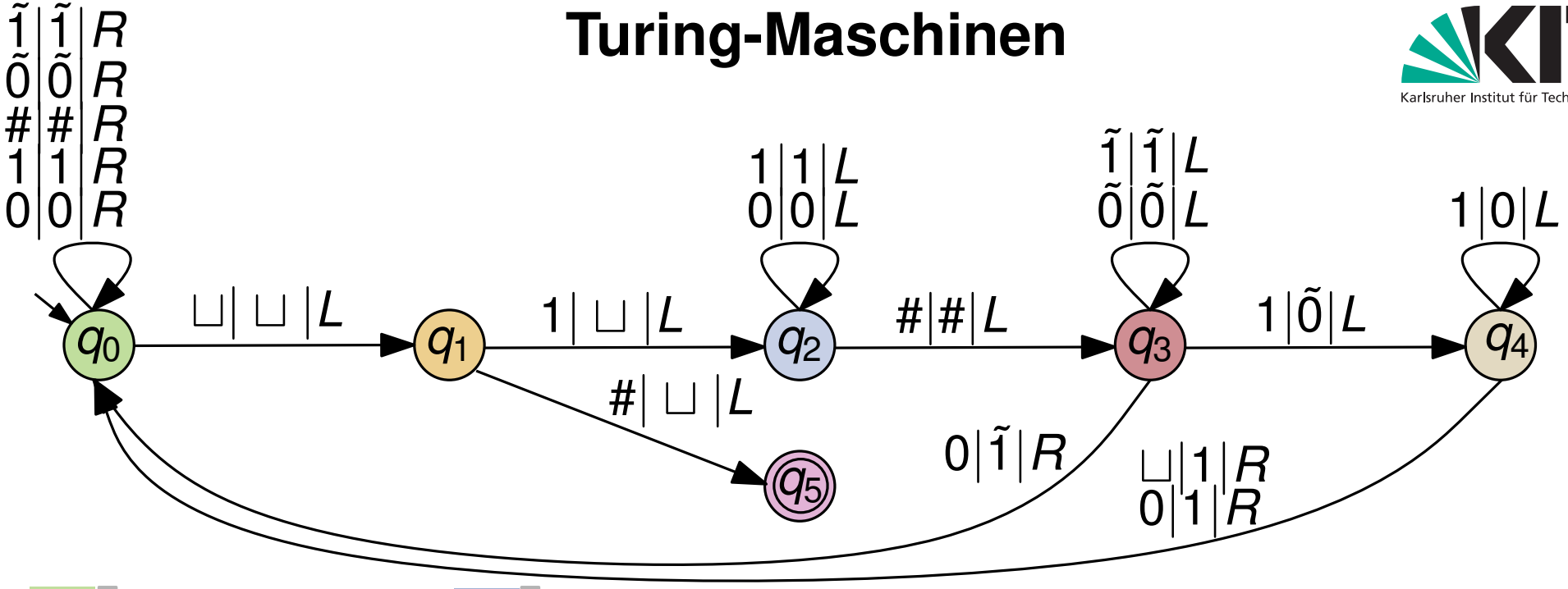
␣111#(q_2)11␣

␣111(q_2)#11␣

␣11(q_3)1#11␣

␣1(q_4)1̃0#11␣

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

. . .

␣111#111(q_0)␣

␣111#11(q_1)1␣

␣111#1(q_2)1␣

␣111#(q_2)11␣

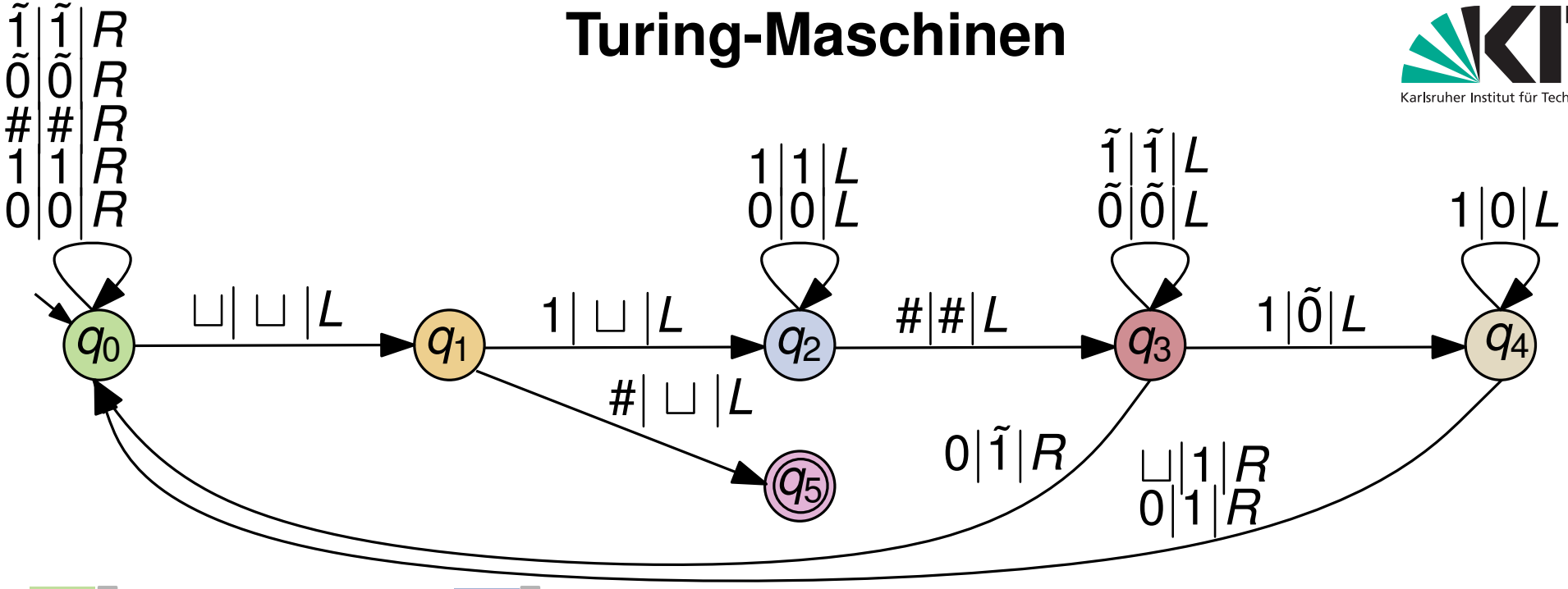
␣111(q_2)#11␣

␣11(q_3)1#11␣

␣1(q_4)1̃0#11␣

␣(q_4)10̃0#11␣

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

...

␣111#111(q_0)␣

␣111#11(q_1)1␣

␣111#1(q_2)1␣

␣111#(q_2)11␣

␣111(q_2)#11␣

␣11(q_3)1#11␣

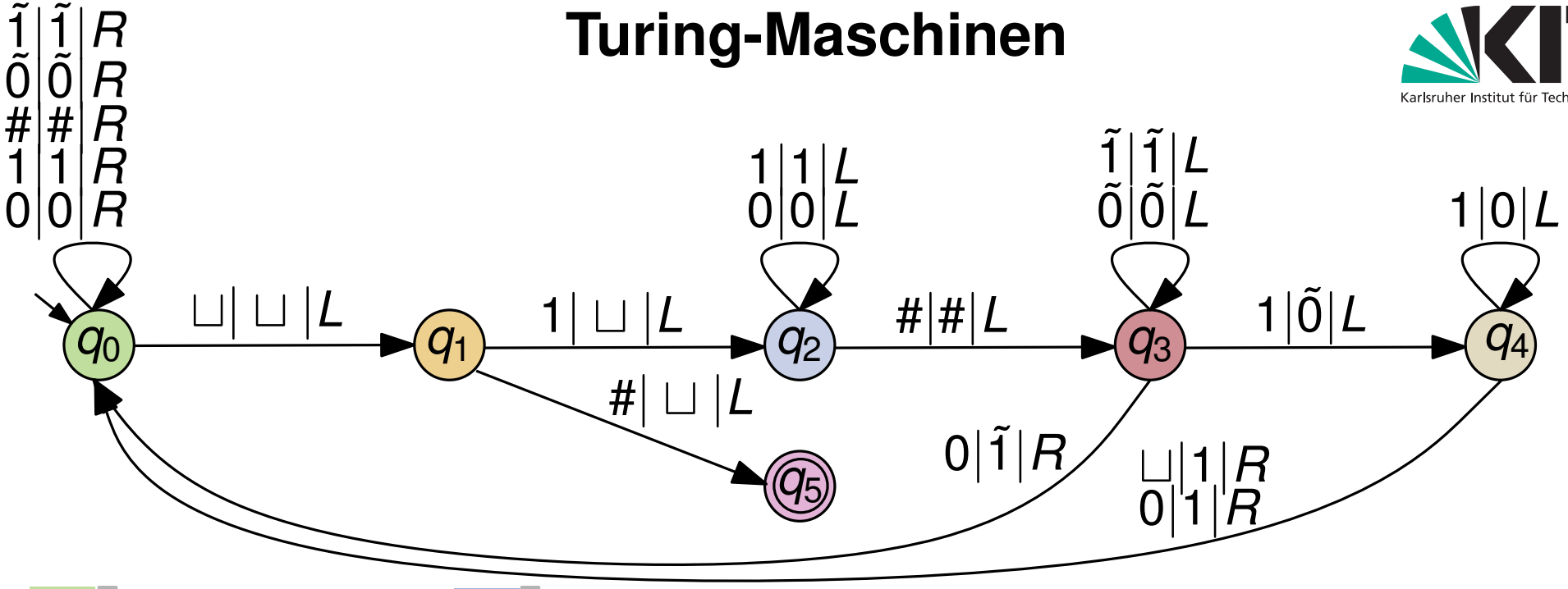
␣1(q_4)10̃#11␣

␣(q_4)100̃#11␣

␣(q_4)␣00̃#11␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

...

␣111#111(q_0)␣

␣111#11(q_1)1␣

␣111#1(q_2)1␣

␣111#(q_2)11␣

␣111(q_2)#11␣

␣11(q_3)1#11␣

␣1(q_4)10̃#11␣

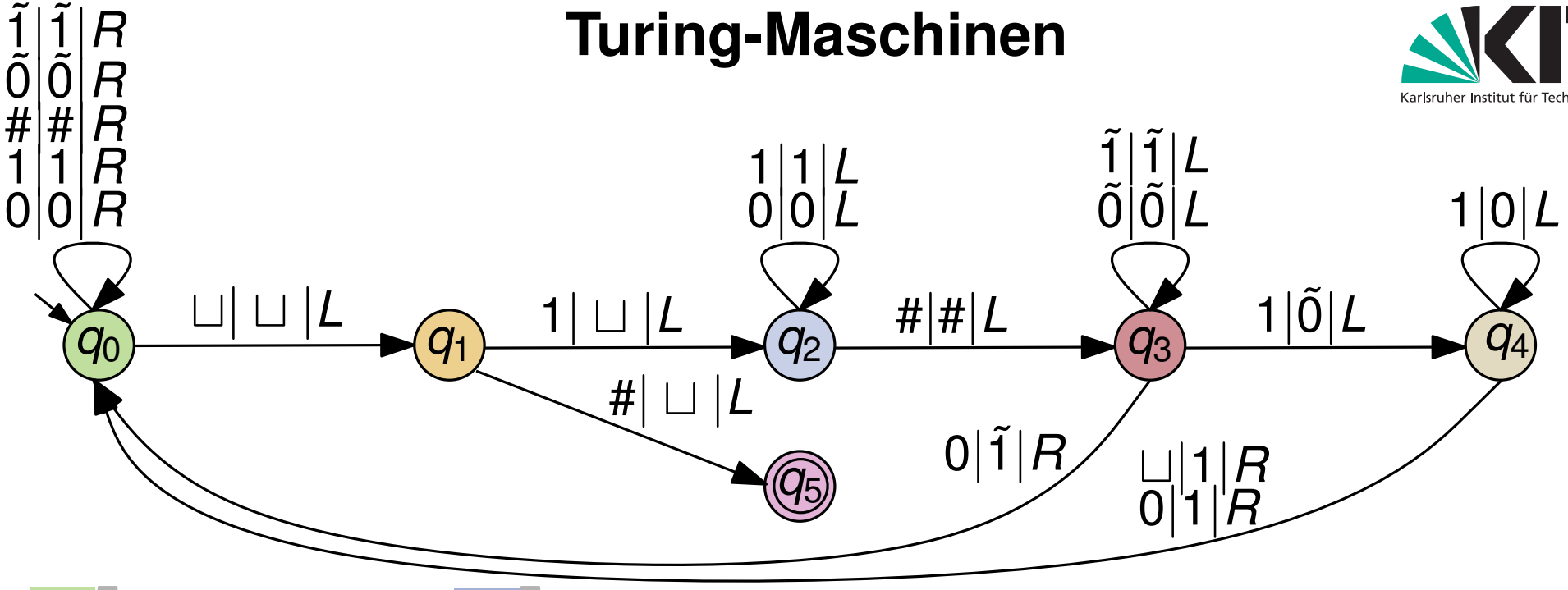
␣(q_4)100̃#11␣

␣(q_4)␣00̃#11␣

␣1(q_0)00̃#11␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



$\sqcup(q_0)111\#111\sqcup$

$\sqcup1(q_0)11\#111\sqcup$

$\sqcup11(q_0)1\#111\sqcup$

...

$\sqcup111\#111(q_0)\sqcup$

$\sqcup111\#11(q_1)1\sqcup$

$\sqcup111\#1(q_2)1\sqcup$

$\sqcup111\#(q_2)11\sqcup$

$\sqcup111(q_2)\#11\sqcup$

$\sqcup11(q_3)1\#11\sqcup$

$\sqcup1(q_4)1\tilde{0}\#11\sqcup$

$\sqcup(q_4)10\tilde{0}\#11\sqcup$

$\sqcup(q_4)\sqcup00\tilde{0}\#11\sqcup$

$\sqcup1(q_0)00\tilde{0}\#11\sqcup$

...

$\sqcup100\tilde{0}\#11(q_0)\sqcup$

$\sqcup100\tilde{0}\#1(q_1)1\sqcup$

$\sqcup100\tilde{0}\#(q_2)1\sqcup$

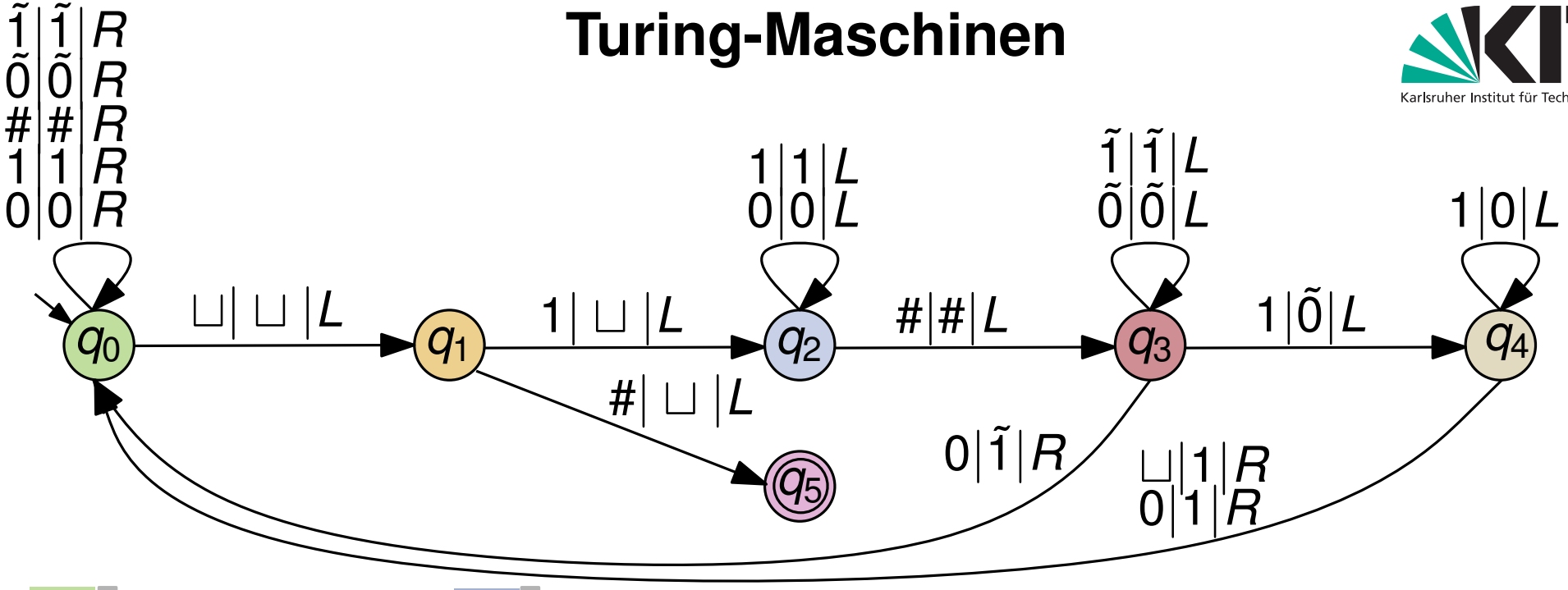
$\sqcup100\tilde{0}(q_2)\#1\sqcup$

$\sqcup100(q_3)\tilde{0}\#1\sqcup$

$\sqcup10(q_3)0\tilde{0}\#1\sqcup$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



␣(q_0)111#111␣

␣1(q_0)11#111␣

␣11(q_0)1#111␣

...

␣111#111(q_0)␣

␣111#11(q_1)1␣

␣111#1(q_2)1␣

␣111#(q_2)11␣

␣111(q_2)#11␣

␣11(q_3)1#11␣

␣1(q_4)1̃0#11␣

␣(q_4)10̃0#11␣

␣(q_4)␣00̃0#11␣

␣1(q_0)00̃0#11␣

...

␣100̃0#11(q_0)␣

␣100̃0#1(q_1)1␣

␣100̃0#(q_2)1␣

␣100̃0(q_2)#1␣

␣100(q_3)0̃#1␣

␣10(q_3)00̃0#1␣

␣10 1̃ (q_0)0̃#1␣

...

␣10 1̃ 0̃#1(q_0)␣

␣10 1̃ 0̃#(q_1)1␣

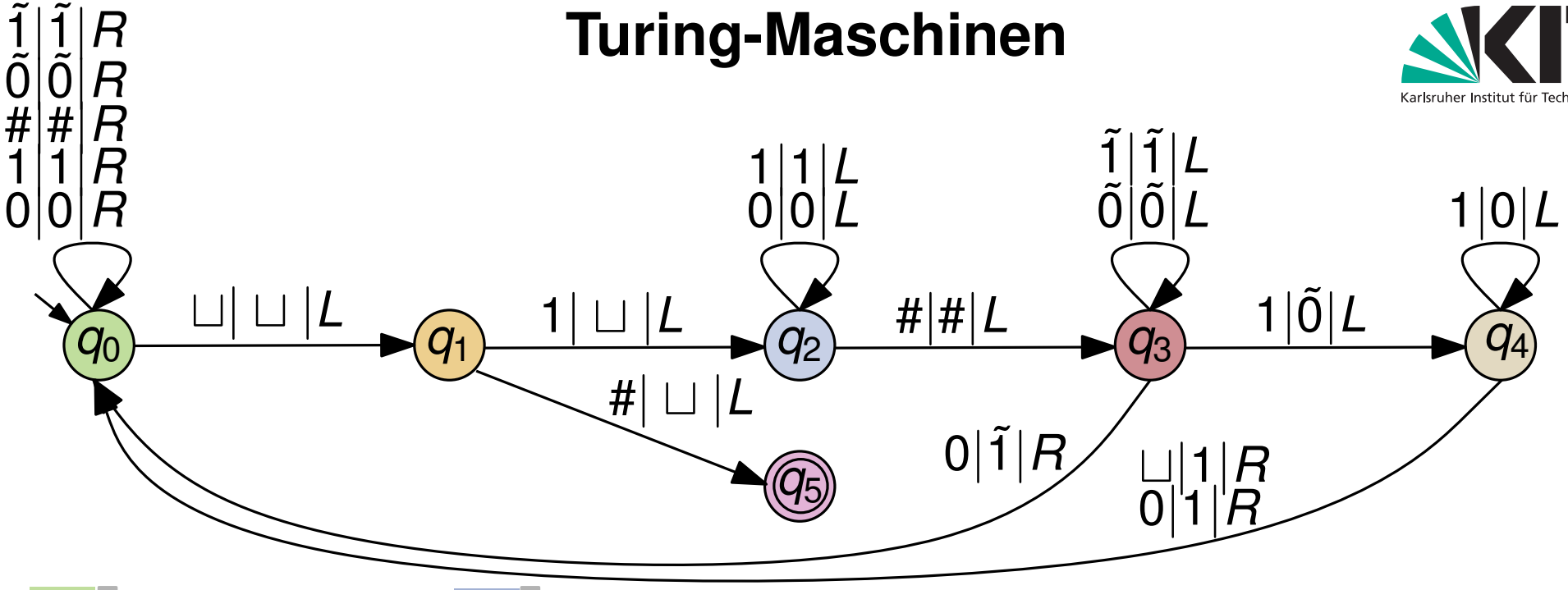
␣10 1̃ 0̃(q_2)#␣

␣10 1̃ (q_3)0̃#␣

␣10(q_3)1̃0̃#␣

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen



$\sqcup(q_0)111\#111\sqcup$

$\sqcup1(q_0)11\#111\sqcup$

$\sqcup11(q_0)1\#111\sqcup$

$\ldots$

$\sqcup111\#111(q_0)\sqcup$

$\sqcup111\#11(q_1)1\sqcup$

$\sqcup111\#1(q_2)1\sqcup$

$\sqcup111\#(q_2)11\sqcup$

$\sqcup111(q_2)\#11\sqcup$

$\sqcup11(q_3)1\#11\sqcup$

$\sqcup1(q_4)1\tilde{0}\#11\sqcup$

$\sqcup(q_4)10\tilde{0}\#11\sqcup$

$\sqcup(q_4)\sqcup00\tilde{0}\#11\sqcup$

$\sqcup1(q_0)00\tilde{0}\#11\sqcup$

$\ldots$

$\sqcup100\tilde{0}\#11(q_0)\sqcup$

$\sqcup100\tilde{0}\#1(q_1)1\sqcup$

$\sqcup100\tilde{0}\#(q_2)1\sqcup$

$\sqcup100\tilde{0}(q_2)\#1\sqcup$

$\sqcup100(q_3)\tilde{0}\#1\sqcup$

$\sqcup10(q_3)0\tilde{0}\#1\sqcup$

$\sqcup10\tilde{1}(q_0)\tilde{0}\#1\sqcup$

$\ldots$

$\sqcup10\tilde{1}\tilde{0}\#1(q_0)\sqcup$

$\sqcup10\tilde{1}\tilde{0}\#(q_1)1\sqcup$

$\sqcup10\tilde{1}\tilde{0}(q_2)\#\sqcup$

$\sqcup10\tilde{1}(q_3)\tilde{0}\#\sqcup$

$\sqcup10(q_3)\tilde{1}\tilde{0}\#\sqcup$

$\sqcup1(q_3)0\tilde{1}\tilde{0}\#\sqcup$

$\sqcup1\tilde{1}(q_0)\tilde{1}\tilde{0}\#\sqcup$

$\ldots$

$\sqcup1\tilde{1}\tilde{1}\tilde{0}\#(q_0)\sqcup$

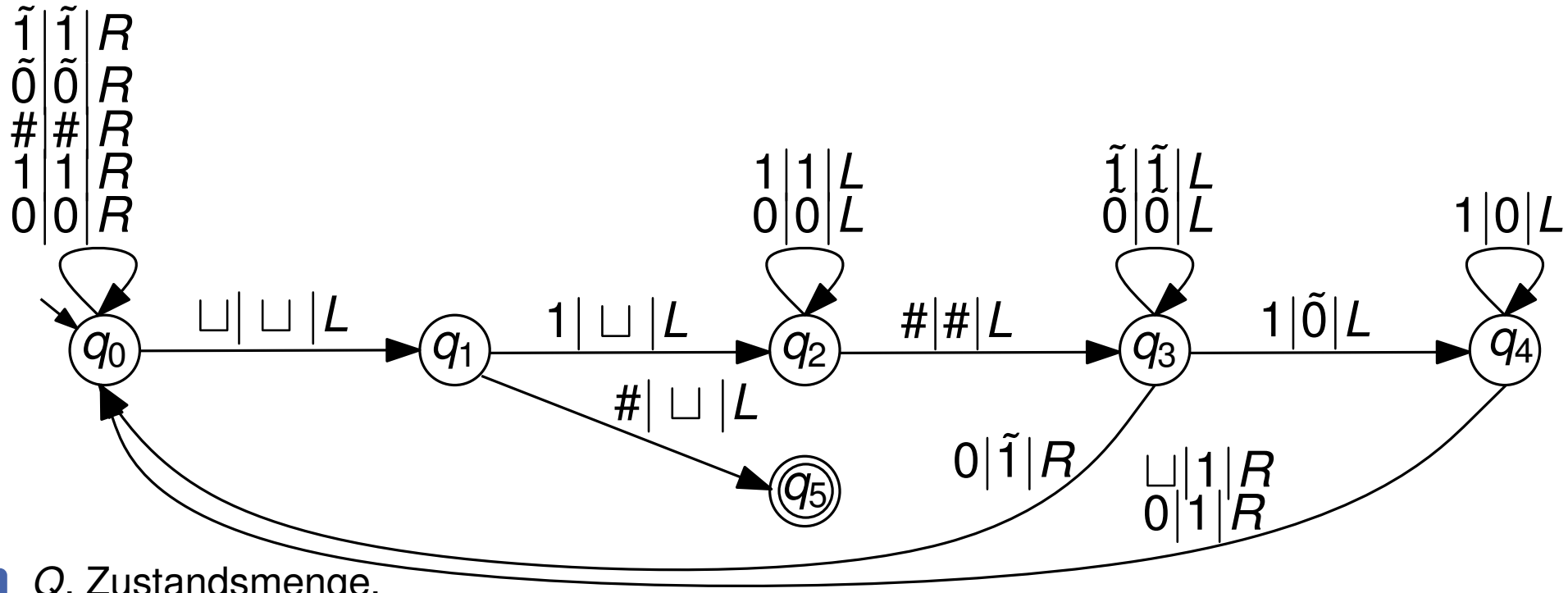
$\sqcup1\tilde{1}\tilde{1}\tilde{0}(q_1)\#\sqcup$

$\sqcup1\tilde{1}\tilde{1}\tilde{0}(q_5)\sqcup$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Ist die Turing-Maschine vollständig beschrieben?

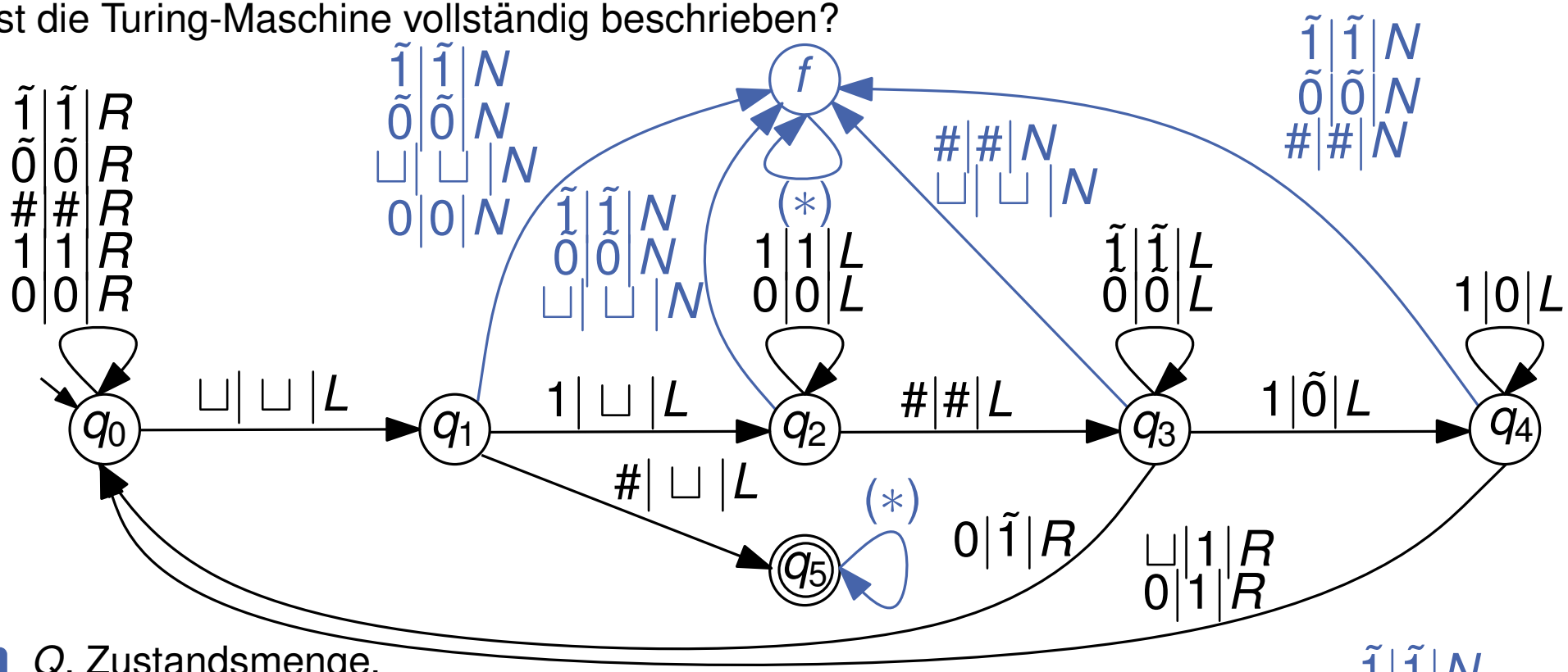


- Q, Zustandsmenge,
- Σ, einem endlichen Eingabealphabet,
- $\sqcup$, einem Blanksymbol mit $\sqcup \notin \Sigma$,
- Γ, einem endlichen Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$,
- $s \in Q$, einem Startzustand,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, einer Übergangsfunktion.
- $F \subseteq Q$, einer Menge von Endzuständen.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Ist die Turing-Maschine vollständig beschrieben?



- Q, Zustandsmenge,
- Σ, einem endlichen Eingabealphabet,
- $\sqcup$, einem Blanksymbol mit $\sqcup \notin \Sigma$,
- Γ, einem endlichen Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$,
- $s \in Q$, einem Startzustand,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, einer Übergangsfunktion.
- $F \subseteq Q$, einer Menge von Endzuständen.

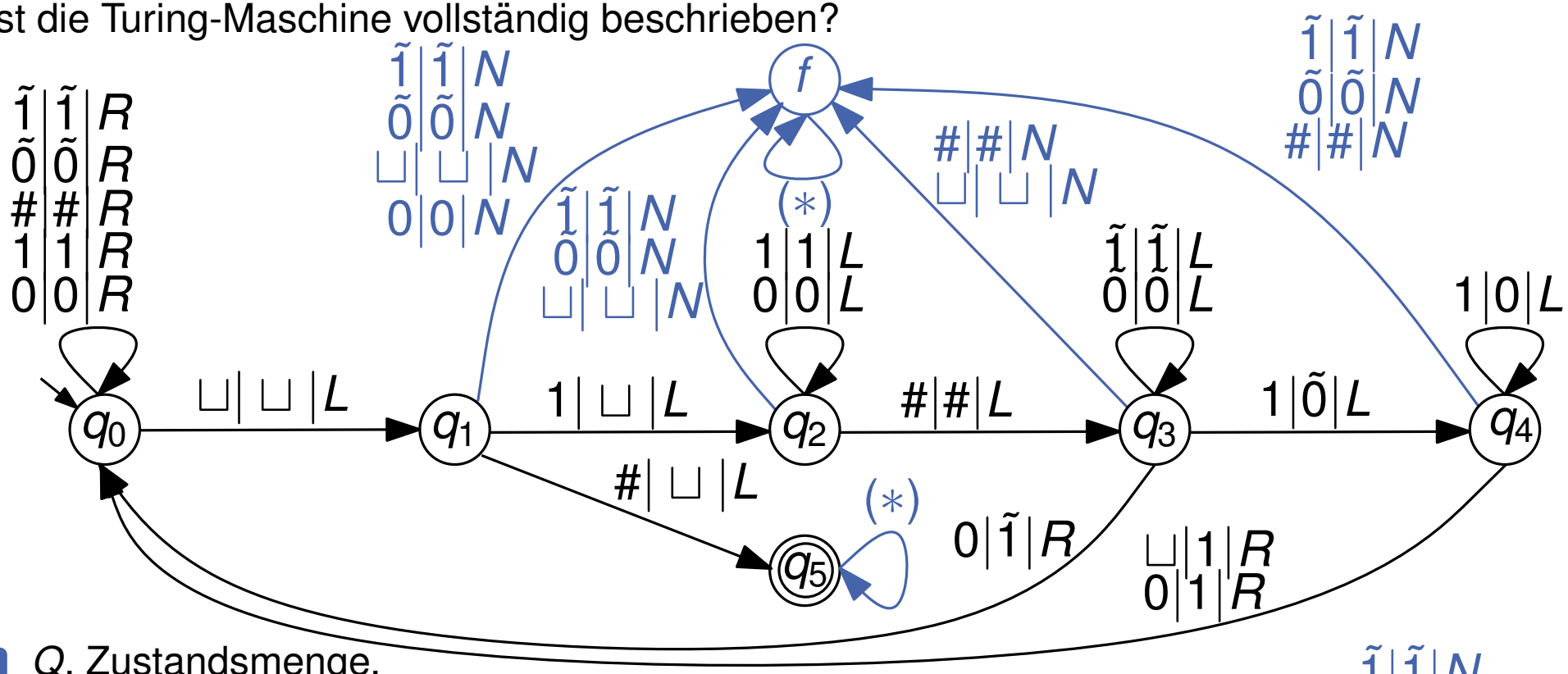
$(*)$

$\tilde{1}$	$\tilde{1}$	N
$\tilde{0}$	$\tilde{0}$	N
$\#$	$\#$	N
1	1	N
0	0	N
$\sqcup$	$\sqcup$	N

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Ist die Turing-Maschine vollständig beschrieben?



- Q, Zustandsmenge,
- Σ, einem enc
- $\sqcup$, einem Bla
- Γ, einem end
- $s \in Q$, einem
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, einer Übergangsfunktion.
- $F \subseteq Q$, einer Menge von Endzuständen.

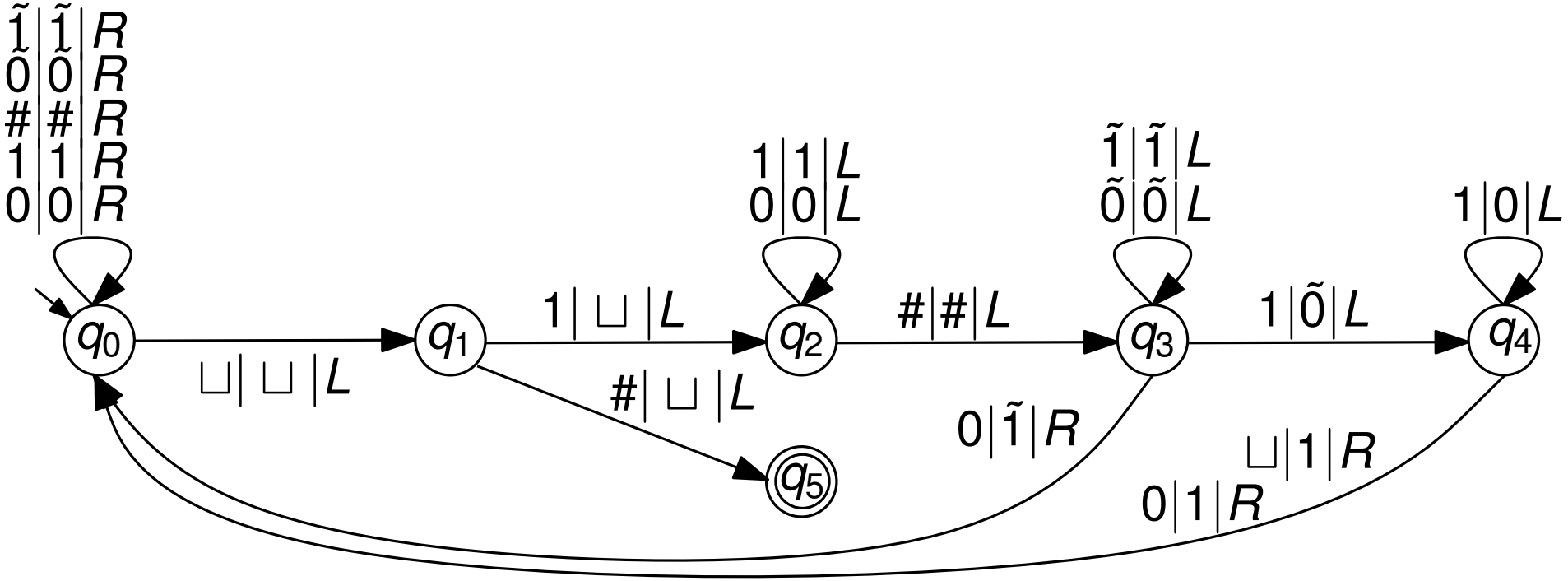
Was berechnet die Turing-Maschine?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Wie ändern, damit Addition von allgemeinen Eingaben funktioniert?

Beispiel: 100#0110

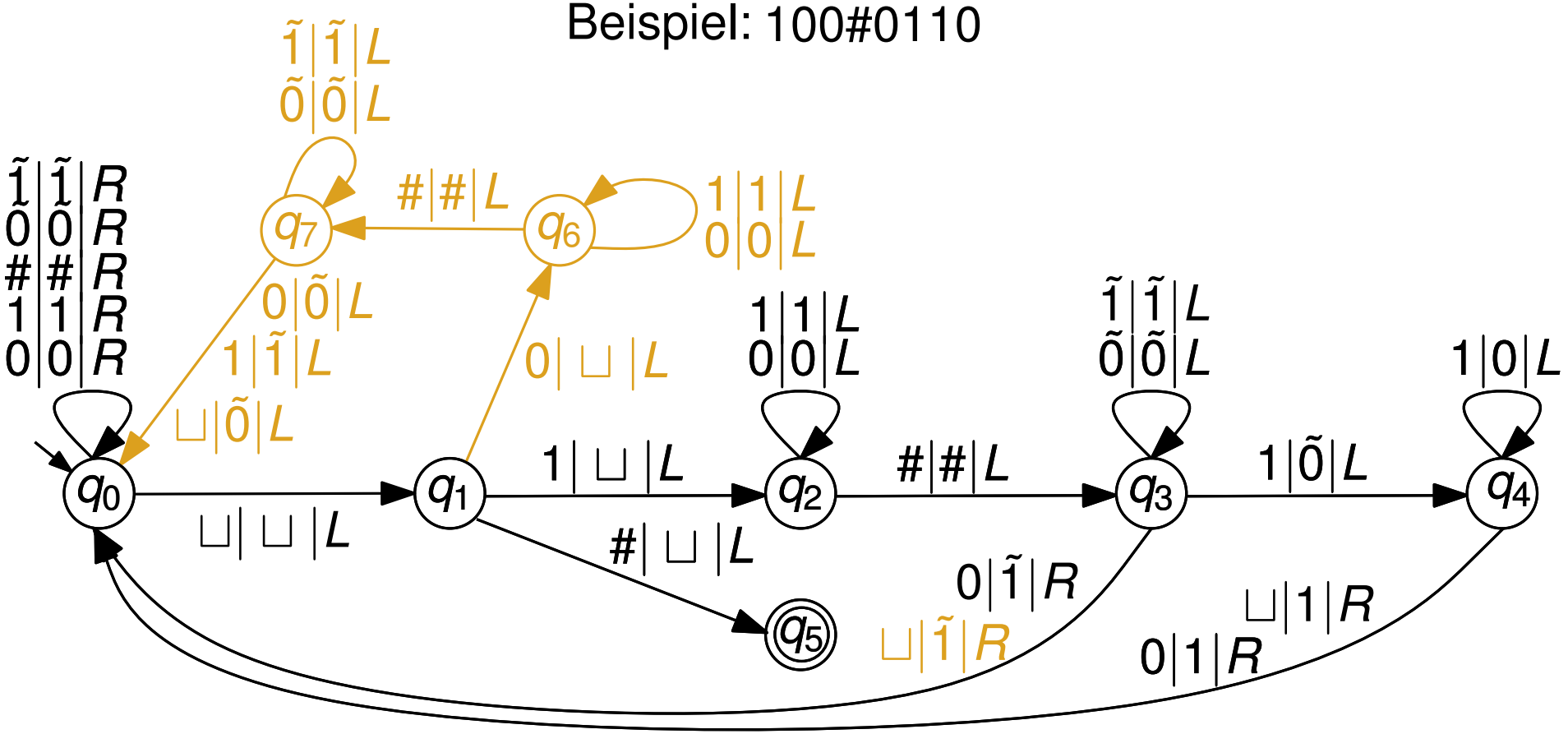


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Turing-Maschinen

Wie ändern, damit Addition von allgemeinen Eingaben funktioniert?

Beispiel: 100#0110



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterungen von Turing-Maschinen

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Mehrere Spuren

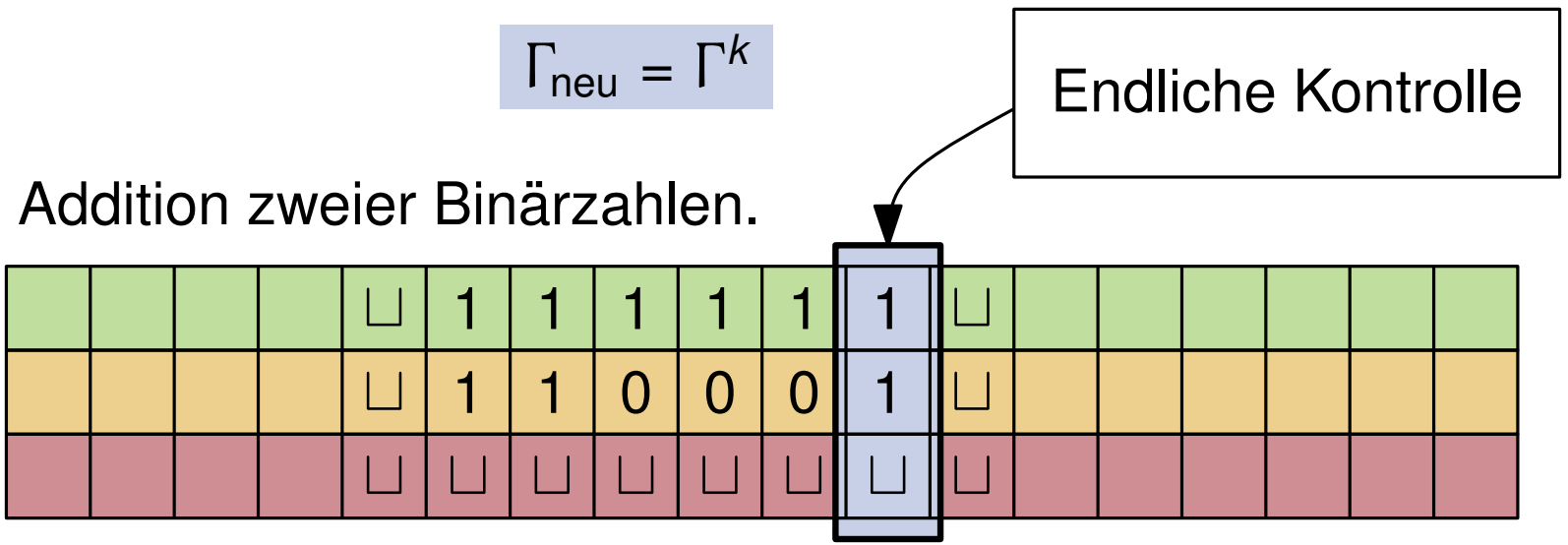
Turing-Maschine mit k Spuren:

Idee: Teile das Eingabeband in k Spuren ein.

Formal: Erweitere das Bandalphabet um k-dimensionale Vektoren:

$$\Gamma_{neu} = \Gamma^k$$

Beispiel: Addition zweier Binärzahlen.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

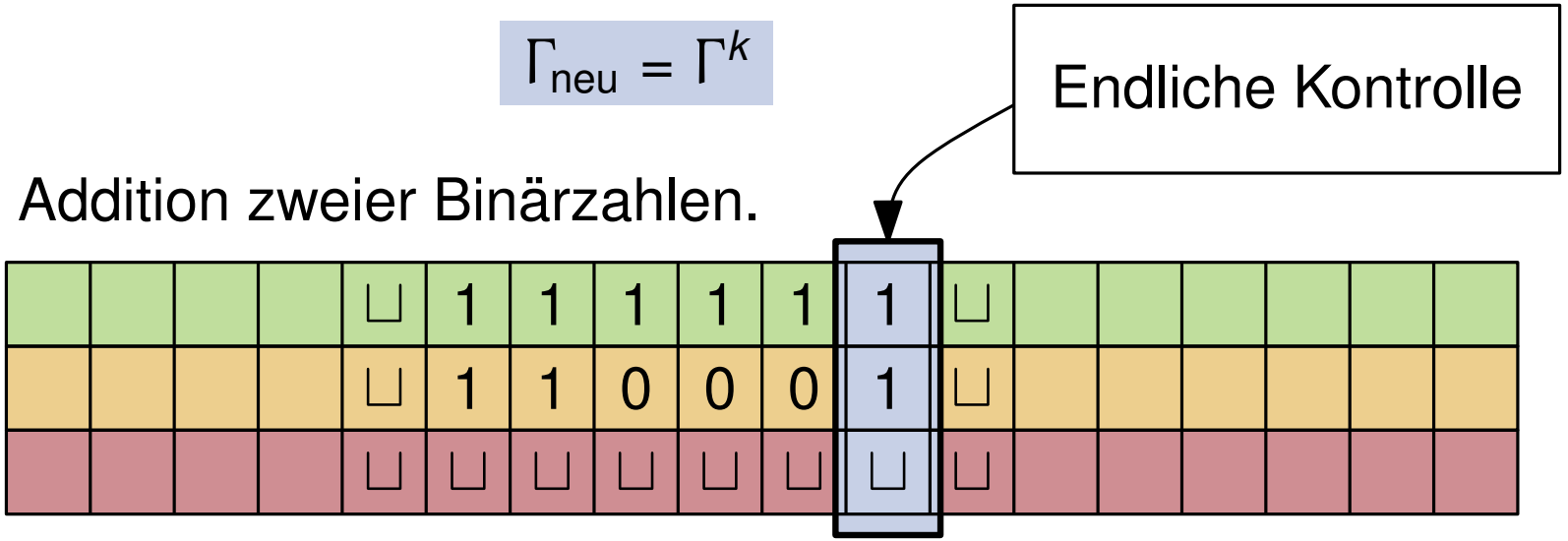
Mehrere Spuren

Turing-Maschine mit k Spuren:

Idee: Teile das Eingabeband in k Spuren ein.

Formal: Erweitere das Bandalphabet um k-dimensionale Vektoren:

$$\Gamma_{\text{neu}} = \Gamma^k$$



Beispiel: Addition zweier Binärzahlen.

- Verwende erste Spur für ersten Summand, zweite Spur für zweiten Summand und dritte Spur für Ergebnis.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

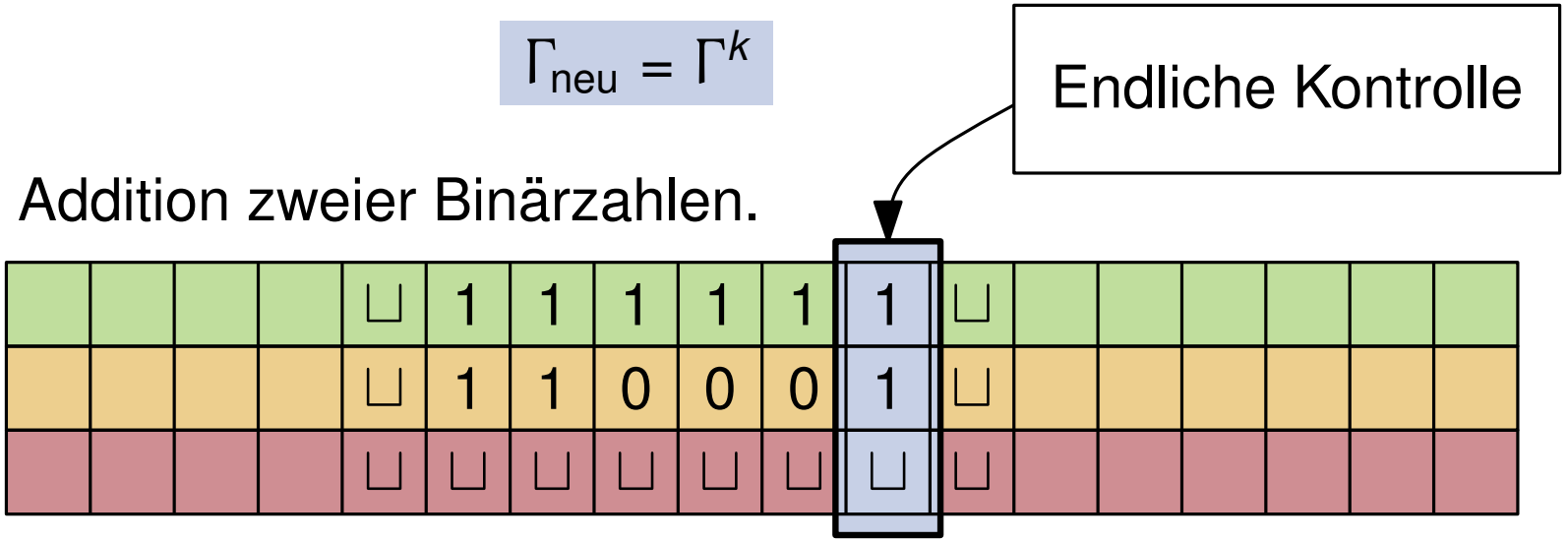
Mehrere Spuren

Turing-Maschine mit k Spuren:

Idee: Teile das Eingabeband in k Spuren ein.

Formal: Erweitere das Bandalphabet um k-dimensionale Vektoren:

$$\Gamma_{neu} = \Gamma^k$$



Beispiel: Addition zweier Binärzahlen.

- Verwende erste Spur für ersten Summand, zweite Spur für zweiten Summand und dritte Spur für Ergebnis.
- Verwende *Schulmethode* für Addition: Kopf läuft von links nach rechts.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Mehrere Spuren

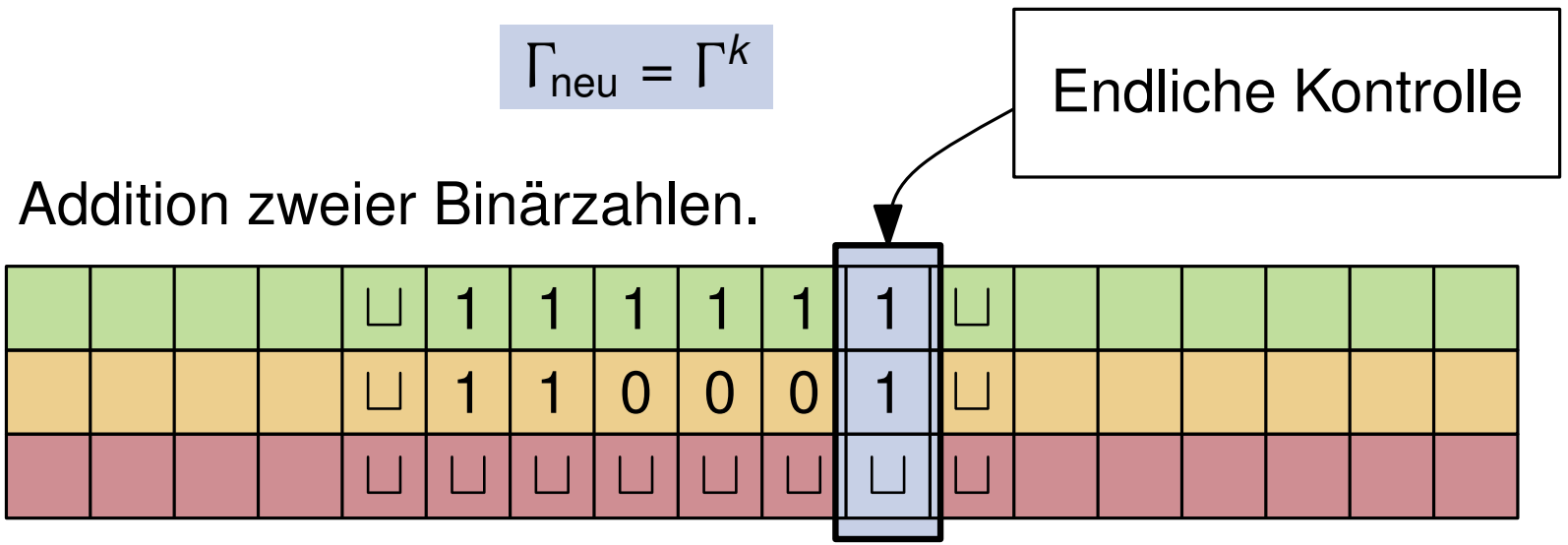
Turing-Maschine mit k Spuren:

Idee: Teile das Eingabeband in k Spuren ein.

Formal: Erweitere das Bandalphabet um k-dimensionale Vektoren:

$$\Gamma_{neu} = \Gamma^k$$

Beispiel: Addition zweier Binärzahlen.



- Verwende erste Spur für ersten Summand, zweite Spur für zweiten Summand und dritte Spur für Ergebnis.
- Verwende *Schulmethode* für Addition: Kopf läuft von links nach rechts.
- Speichere Übertrag in Zustand.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

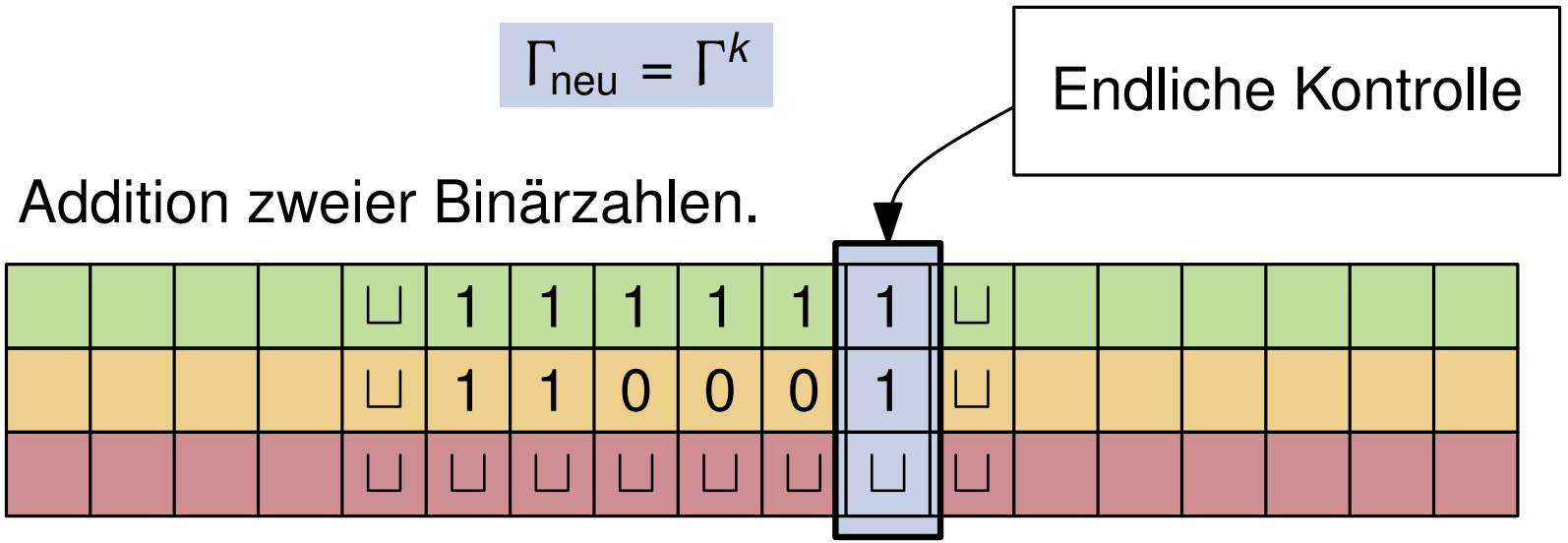
Mehrere Spuren

Turing-Maschine mit k Spuren:

Idee: Teile das Eingabeband in k Spuren ein.

Formal: Erweitere das Bandalphabet um k-dimensionale Vektoren:

$$\Gamma_{neu} = \Gamma^k$$



Beispiel: Addition zweier Binärzahlen.

- Verwende erste Spur für ersten Summand, zweite Spur für zweiten
- Ist eine Turing-Maschine mit $k \geq 2$ mächtiger als eine Turing-Maschine mit einer Spur?
- ts.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

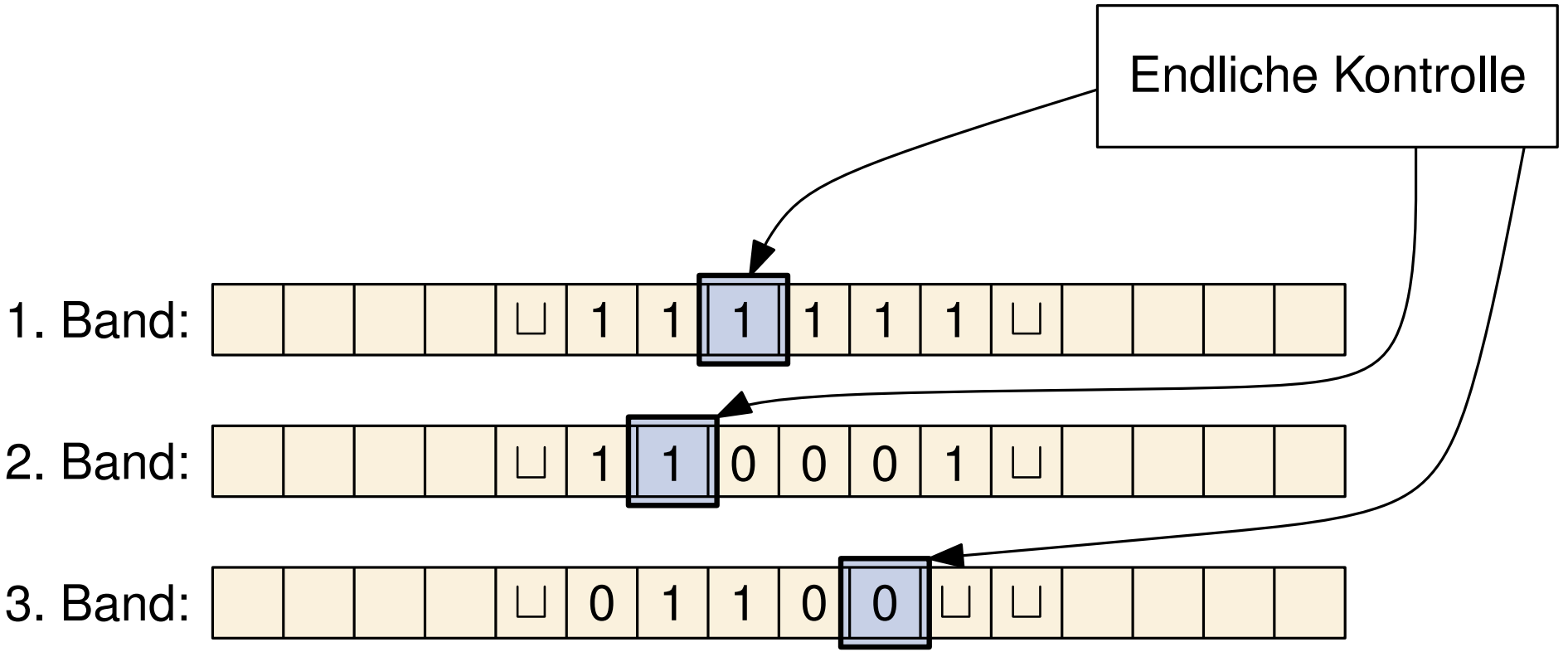
Mehrere Bänder

Turing-Maschine mit k Bändern:

Idee: Es gibt k Bänder und auf jedem Band arbeitet eigener Kopf.

Formal: Passe Übergangsfunktion δ an.

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{R, L, N\}^k$$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

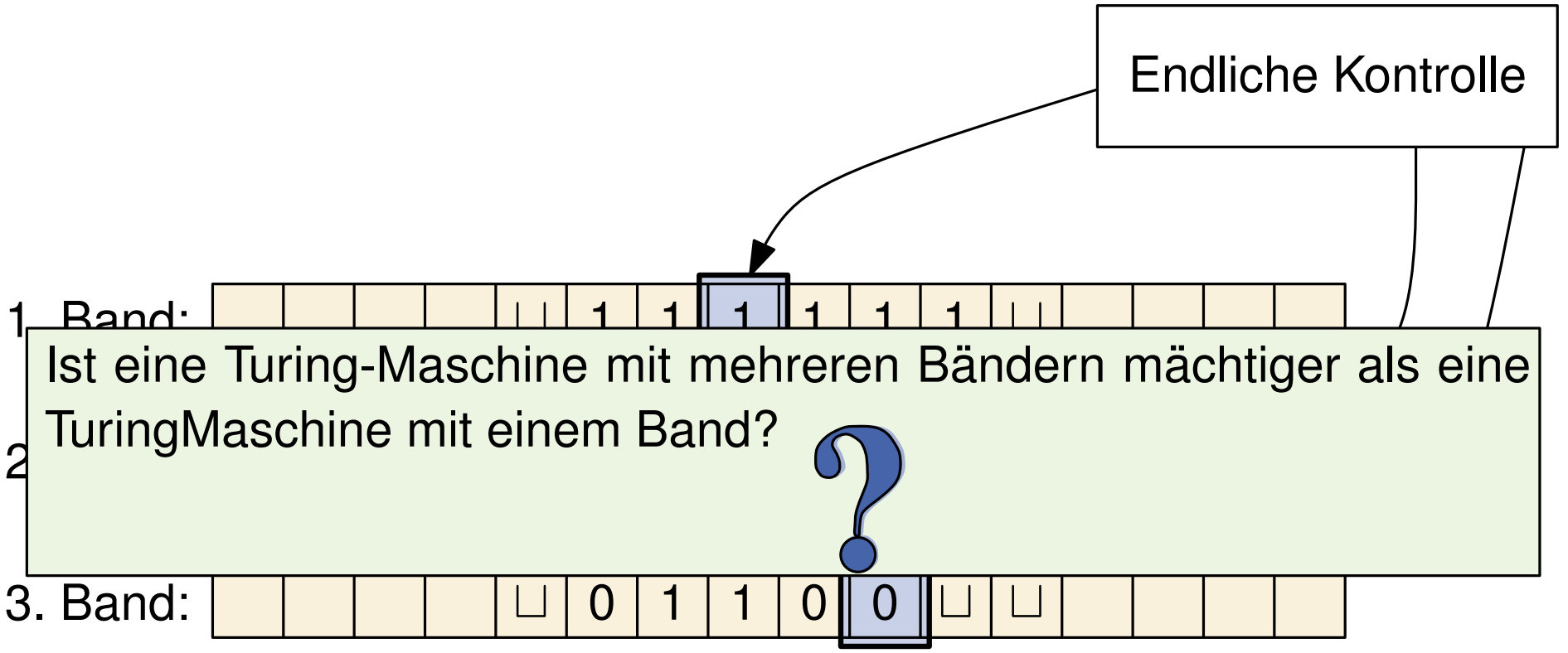
Mehrere Bänder

Turing-Maschine mit k Bändern:

Idee: Es gibt k Bänder und auf jedem Band arbeitet eigener Kopf.

Formal: Passe Übergangsfunktion δ an.

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{R, L, N\}^k$$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

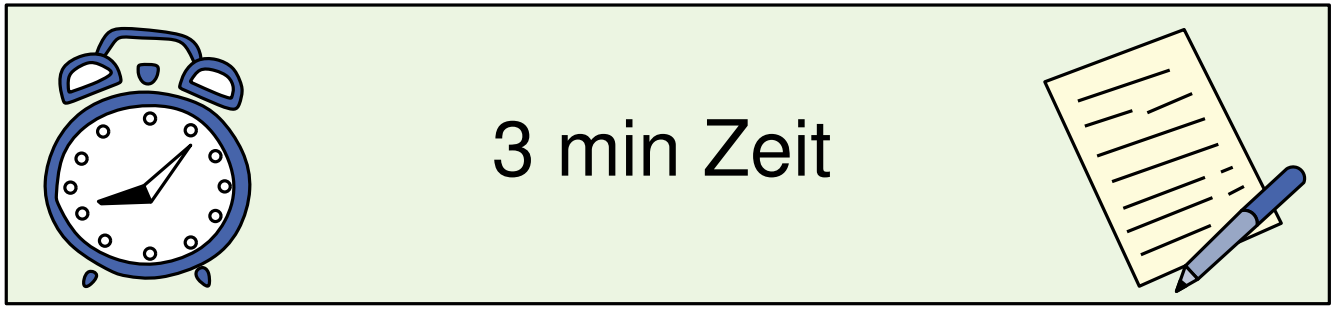
Beweis:

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis:



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Die Turing-Maschine $\mathcal{M}'$ verwendet $2k$ Spuren.

Nach Simulation des t-ten Schritts für $1 \leq t \leq t(n)$ gilt:

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

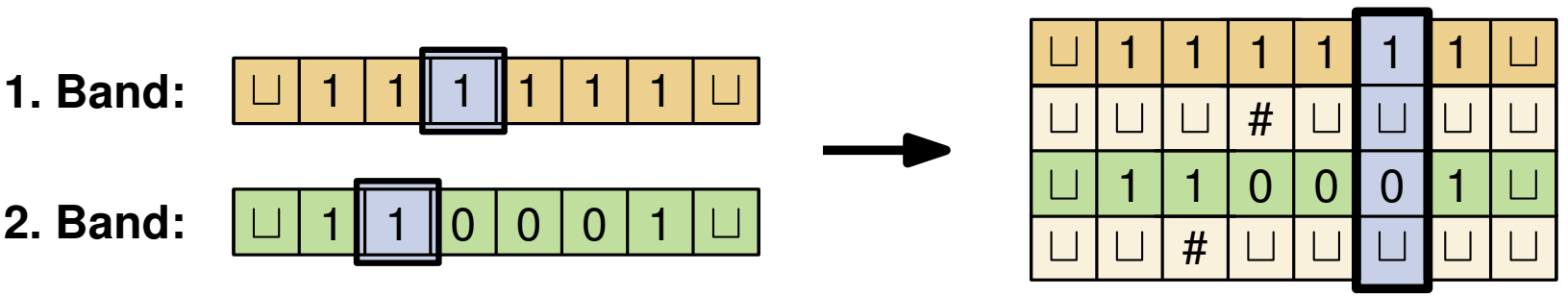
Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Die Turing-Maschine $\mathcal{M}'$ verwendet $2k$ Spuren.

Nach Simulation des t-ten Schritts für $1 \leq t \leq t(n)$ gilt:

- Die ungeraden Spuren $1, 3, 5, \ldots, 2k-1$ enthalten Inhalt der Bänder $1 \ldots, k$ von M.
- Auf den geraden Spuren $2, 4, \ldots, 2k$ ist die Kopfposition auf diesen Bändern mit dem Zeichen # markiert.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Der Kopf von $\mathcal{M}'$ steht auf dem linkesten #.

Jeder Rechenschritt von $\mathcal{M}$ wird durch $\mathcal{M}'$ wie folgt simuliert:

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Der Kopf von $\mathcal{M}'$ steht auf dem linkesten #.

Jeder Rechenschritt von $\mathcal{M}$ wird durch $\mathcal{M}'$ wie folgt simuliert:

1. Schritt: Der Kopf von $\mathcal{M}'$ läuft nach rechts bis zum rechtesten # und merkt sich in einem *endlichen Speicher* ($\rightarrow$ eigener Zustand), was die k Köpfe von $\mathcal{M}$ lesen würden.

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Der Kopf von $\mathcal{M}'$ steht auf dem linkesten #.

Jeder Rechenschritt von $\mathcal{M}$ wird durch $\mathcal{M}'$ wie folgt simuliert:

1. Schritt: Der Kopf von $\mathcal{M}'$ läuft nach rechts bis zum rechtesten # und merkt sich in einem *endlichen Speicher* ($\rightarrow$ eigener Zustand), was die k Köpfe von $\mathcal{M}$ lesen würden.

2. Schritt: Der Kopf von $\mathcal{M}'$ läuft nach links und an den entsprechenden Markierungen werden die Bandbeschriftungen so verändert, wie es $\mathcal{M}$ tun würde.

Die Positionsmarkierungen werden ggf. nach links bzw. rechts verschoben.

Merke: Zustand, in dem sich nun $\mathcal{M}$ befindet.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Beweis: Der Kopf von $\mathcal{M}'$ steht auf dem linkesten #.

Jeder Rechenschritt von $\mathcal{M}$ wird durch $\mathcal{M}'$ wie folgt simuliert:

1. Schritt: Der Kopf von $\mathcal{M}'$ läuft nach rechts bis zum rechtesten # und merkt sich in einem *endlichen Speicher* ($\rightarrow$ eigener Zustand), was die k Köpfe von $\mathcal{M}$ lesen würden.

2. Schritt: Der Kopf von $\mathcal{M}'$ läuft nach links und an den entsprechenden Markierungen werden die Bandbeschriftungen so verändert, wie es $\mathcal{M}$ tun würde.

Die Positionsmarkierungen werden ggf. nach links bzw. rechts verschoben.

Merke: Zustand, in dem sich nun $\mathcal{M}$ befindet.

3. Schritt: Der Kopf von $\mathcal{M}'$ läuft zum linkesten #.

$\Rightarrow$ Ausgangssituation für nächsten Berechnungsschritt $t + 1$.

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Laufzeit:

- $t(n)$ beschränkt die Breite des betrachteten Bandes von $\mathcal{M}'$.
- Jede Rechenschritt von $t(n)$ wird in $\mathcal{O}(t(n))$ Zeit simuliert.

$\Rightarrow$ $\mathcal{M}'$ simuliert $\mathcal{M}$ in $\mathcal{O}(t^2(n))$ Zeit.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Erweiterte Turing-Maschinen

Satz: Eine k-Band-Turing-Maschine $\mathcal{M}$, die mit Rechenzeit $t(n)$ und Platz $s(n)$ auskommt, kann von einer Turing-Maschine $\mathcal{M}'$ mit Zeitbedarf $\mathcal{O}(t^2(n))$ und Platzbedarf $\mathcal{O}(s(n))$ simuliert werden.

Laufzeit:

- $t(n)$ beschränkt die Breite des betrachteten Bandes von $\mathcal{M}'$.
- Jede Rechenschritt von $t(n)$ wird in $\mathcal{O}(t(n))$ Zeit simuliert.

$\Rightarrow$ $\mathcal{M}'$ simuliert $\mathcal{M}$ in $\mathcal{O}(t^2(n))$ Zeit.

Speicherverbrauch:

$\mathcal{M}'$ braucht nur um einen konstanten Faktor mehr Speicher:
Anzahl Spuren von $\mathcal{M}'$ = 2· Anzahl Bänder von $\mathcal{M}$.

Entscheidbarkeit

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

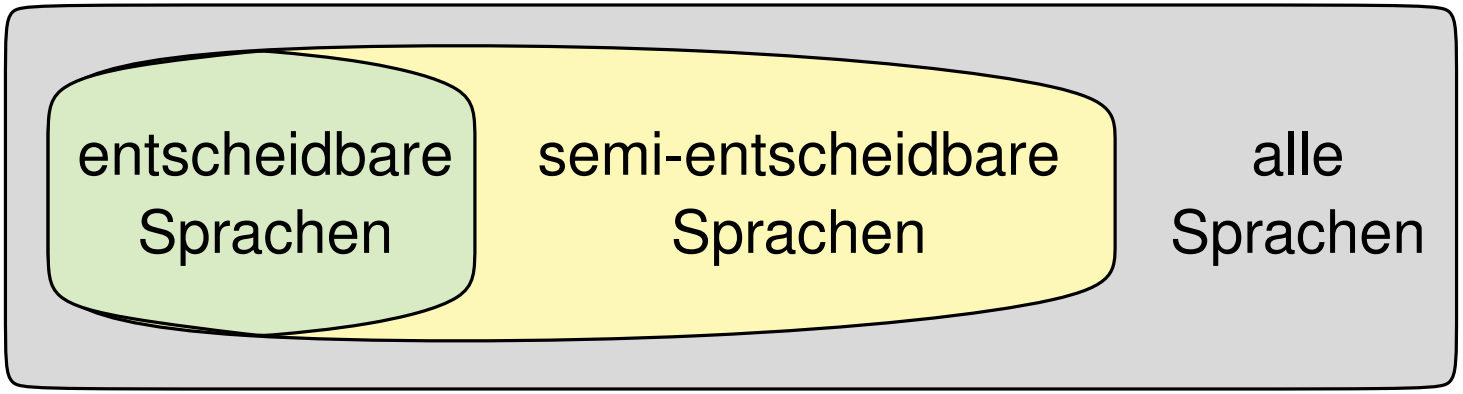
Entscheidbarkeit – Definitionen

Eine Sprache $L \subseteq \Sigma^\star$ heißt **semi-entscheidbar**, wenn
es eine Turing-Maschine gibt, die
eine Eingabe w genau dann akzeptiert, wenn $w \in L$ gilt.

L wird "erkannt"

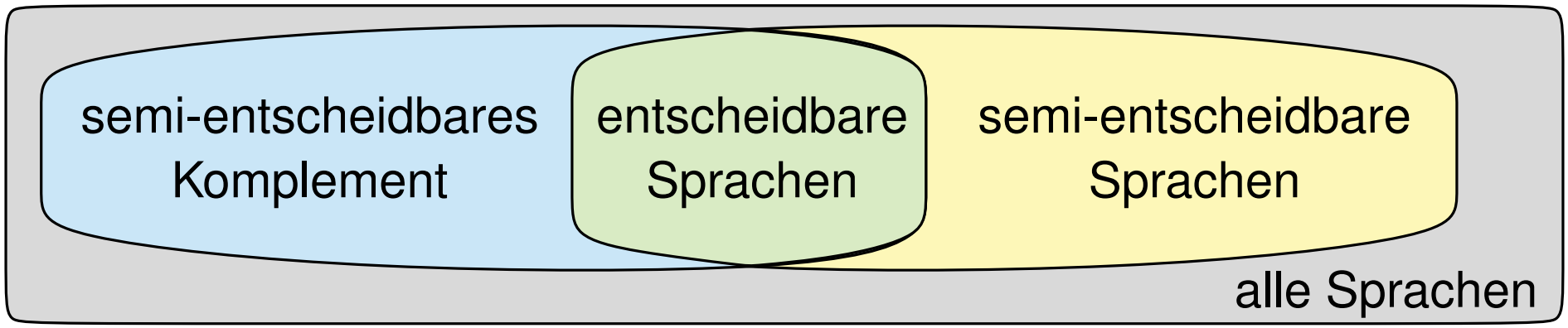
Eine Sprache $L \subseteq \Sigma^\star$ heißt **entscheidbar**, wenn
es eine Turing-Maschine gibt, die
auf allen Eingaben stoppt und
eine Eingabe w genau dann akzeptiert, wenn $w \in L$ gilt.

L wird "entschieden"



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

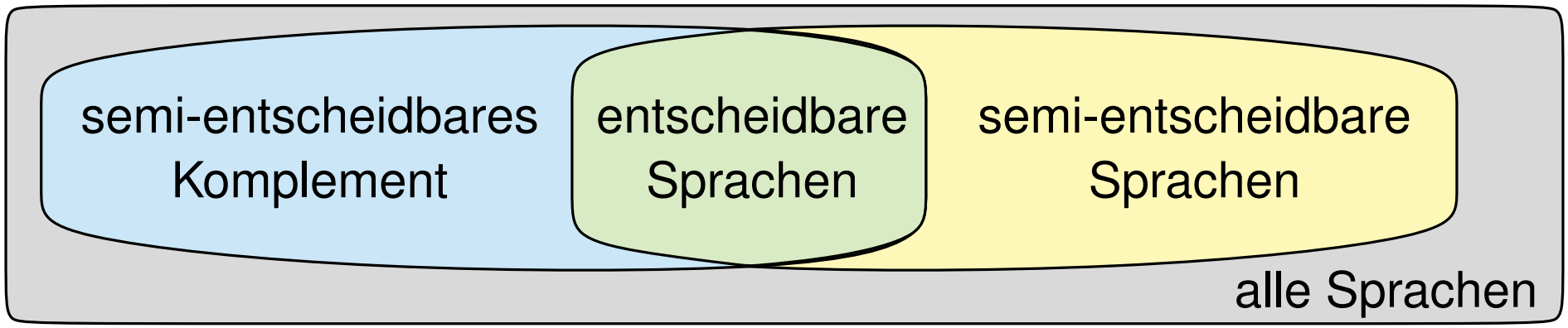
Entscheidbarkeit – Zusammenhänge



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Zusammenhänge



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Beweis, links nach rechts:

L entscheidbar $\Rightarrow$ L^c entscheidbar

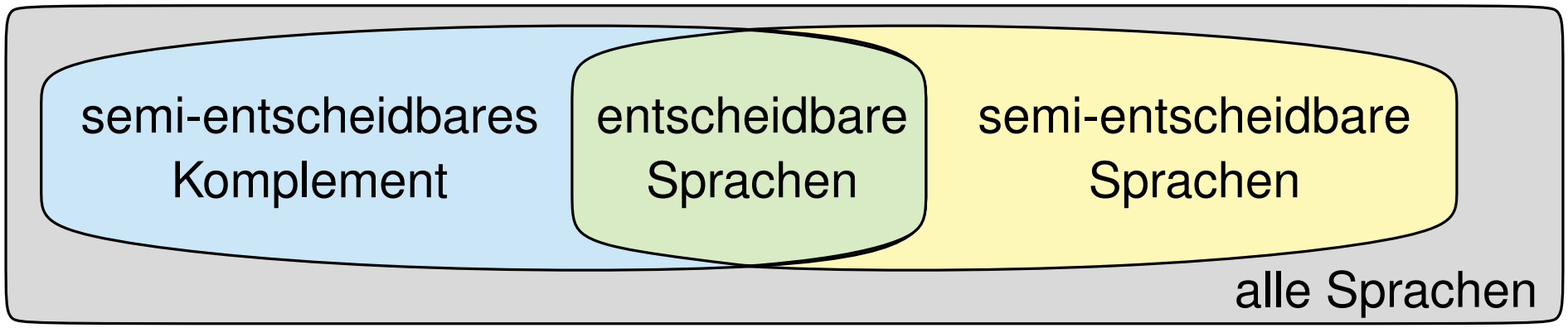
$\Downarrow$ $\Downarrow$

L semi-entscheidbar L^c semi-entscheidbar

einfache Modifikation einer TM $\mathcal{M}$ für L (siehe VL)

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Zusammenhänge



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

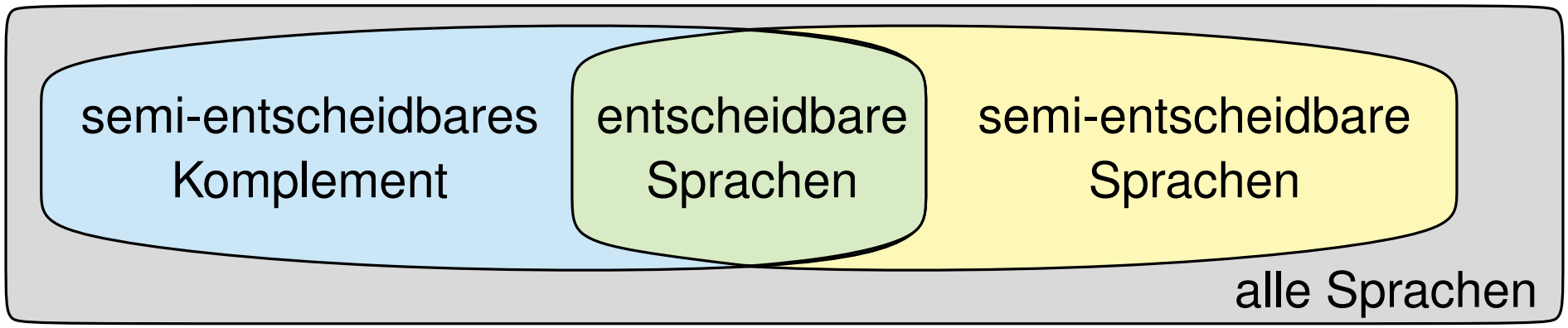
Beweis, rechts nach links:

L semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L$

L^c semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \notin L$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Zusammenhänge



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Beweis, rechts nach links:

L semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L$

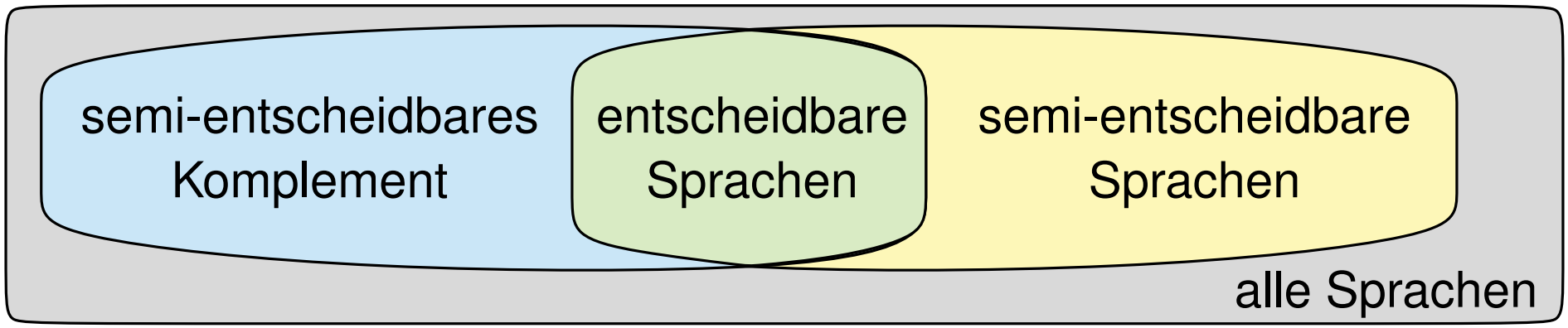
L^c semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \notin L$

Konstruiere 2-Band-Turing-Maschine $\mathcal{M}$:

- Schreibe w auf beide Bänder
- Führe $\mathcal{M}_1$ auf Band 1 und $\mathcal{M}_2$ auf Band 2 aus
- Akzeptiere wenn $\mathcal{M}_1$ akzeptiert, halte wenn $\mathcal{M}_2$ akzeptiert

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Zusammenhänge



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Beweis, rechts nach links:

L semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L$

L^c semi-entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \notin L$

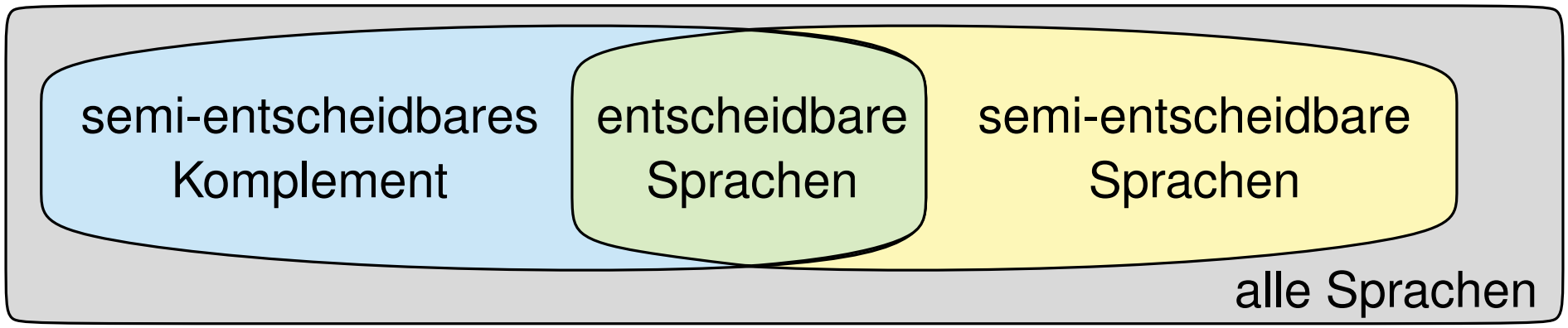
Konstruiere 2-Band-Turing-Maschine $\mathcal{M}$:

- Schreibe w auf beide Bänder
- Führe $\mathcal{M}_1$ auf Band 1 und $\mathcal{M}_2$ auf Band 2 aus
- Akzeptiere wenn $\mathcal{M}_1$ akzeptiert, halte wenn $\mathcal{M}_2$ akzeptiert

$\mathcal{M}$ entscheidet L

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Überblick



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Korollar: L ist entscheidbar $\iff$ L^c ist entscheidbar

Frage: L_1, L_2 entscheidbar $\Rightarrow$ $L_1 \cup L_2$, $L_1 \cap L_2$ entscheidbar?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Komplementbildung

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Komplementbildung

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Beweis:

L_1 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L_1$

L_2 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \in L_2$

Entscheidbarkeit – Komplementbildung

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Beweis:

L_1 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L_1$

L_2 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \in L_2$

Konstruiere 2-Band-Turing-Maschine $\mathcal{M}$:

- Schreibe w auf beide Bänder
- Führe $\mathcal{M}_1$ auf Band 1 und $\mathcal{M}_2$ auf Band 2 aus
- Akzeptiere wenn $\mathcal{M}_1$ akzeptiert und $\mathcal{M}_2$ nicht akzeptiert

Entscheidbarkeit – Komplementbildung

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Beweis:

L_1 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_1$ akzeptiert bei $w \in L_1$

L_2 ist entscheidbar $\rightsquigarrow$ Turing-Maschine $\mathcal{M}_2$ akzeptiert bei $w \in L_2$

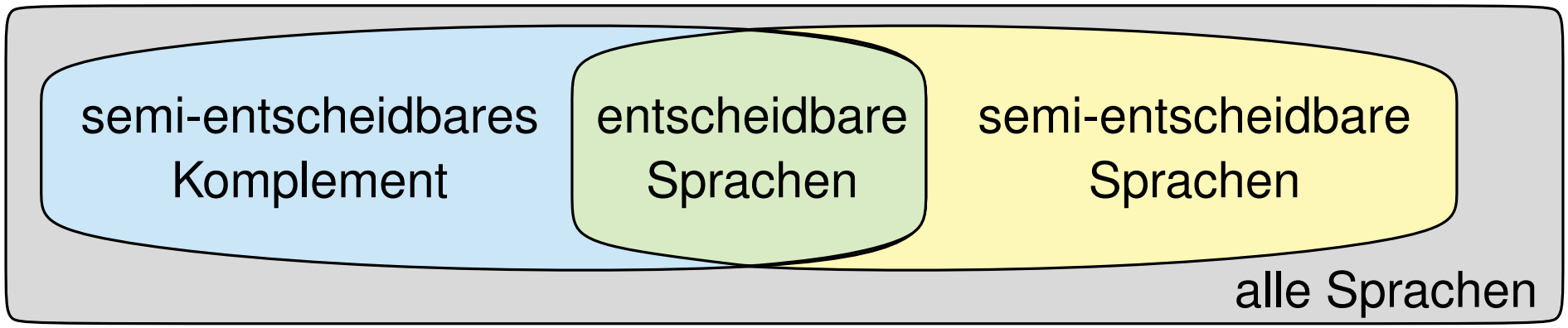
Konstruiere 2-Band-Turing-Maschine $\mathcal{M}$:

- Schreibe w auf beide Bänder
- Führe $\mathcal{M}_1$ auf Band 1 und $\mathcal{M}_2$ auf Band 2 aus
- Akzeptiere wenn $\mathcal{M}_1$ akzeptiert und $\mathcal{M}_2$ nicht akzeptiert

$\mathcal{M}$ entscheidet L

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Überblick



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

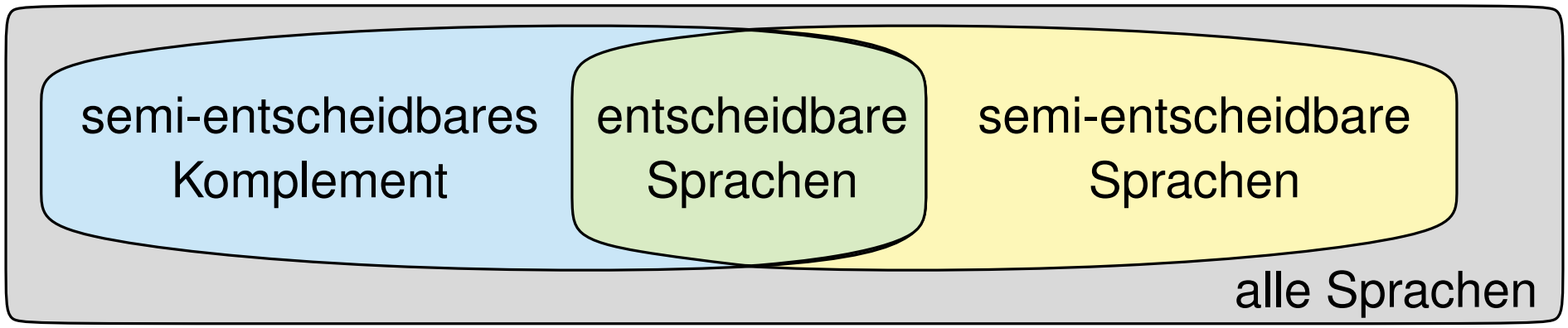
Korollar: L ist entscheidbar $\iff$ L^c ist entscheidbar

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Korollar: L_1, L_2 entscheidbar $\Rightarrow$ $L_1 \cap L_2, L_1 \cup L_2$ entscheidbar

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Überblick



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Korollar: L ist entscheidbar $\iff$ L^c ist entscheidbar

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

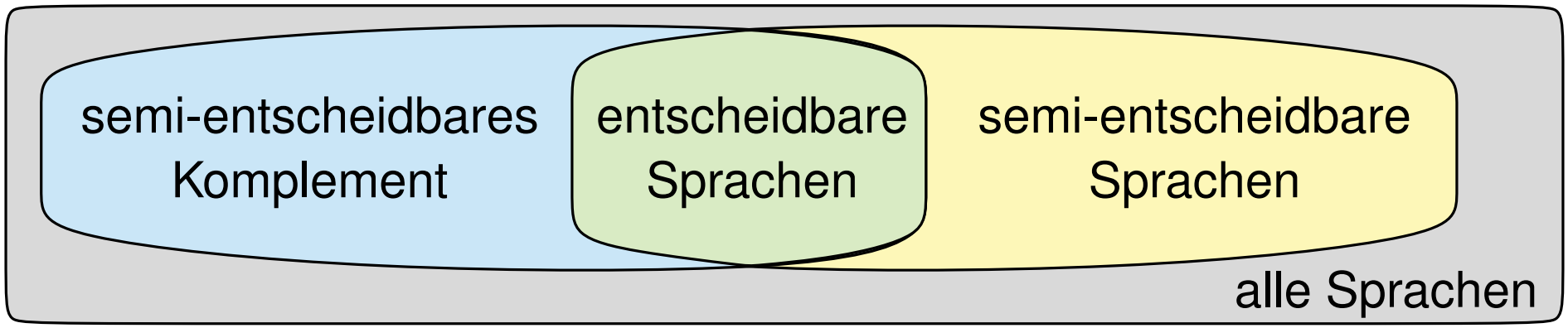
Korollar: L_1, L_2 entscheidbar $\Rightarrow$ $L_1 \cup L_2, L_1 \cap L_2$ entscheidbar

Frage: L_1, L_2 semi-entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ semi-entscheidbar?

Frage: L_1, L_2 semi-entscheidbar $\Rightarrow$ $L_1 \cup L_2$, $L_1 \cap L_2$ semi-entsch.?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Überblick



Satz: L ist entscheidbar $\iff$ L und L^c sind semi-entscheidbar

Korollar: L ist entscheidbar $\iff$ L^c ist entscheidbar

Satz: L_1 und L_2 sind entscheidbar $\Rightarrow$ $L_1 \setminus L_2$ ist entscheidbar

Korollar: L_1, L_2 entscheidbar $\Rightarrow$ $L_1 \cup L_2, L_1 \cap L_2$ entscheidbar

Frage: L_1, L_2 semi-entscheidbar ✗ $L_1 \setminus L_2$ semi-entscheidbar?

Frage: L_1, L_2 semi-entscheidbar ✓ $L_1 \cup L_2$, $L_1 \cap L_2$ semi-entsch.?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Beispiele

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Diagonalsprache

Gödelnummer:

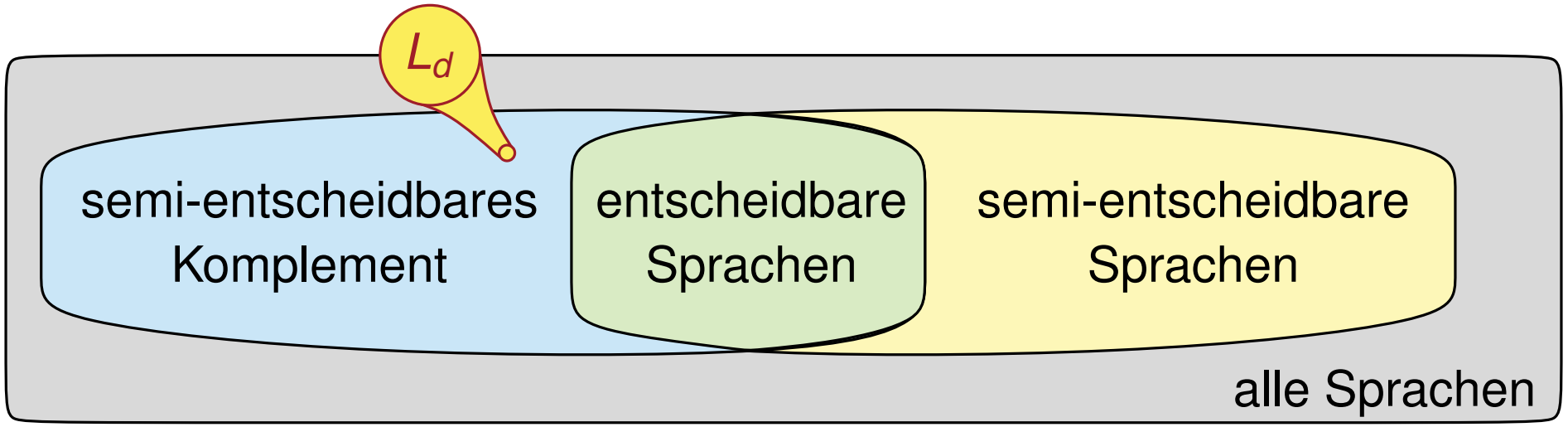
injektive Abbildung von { alle Turing-Maschinen } nach $\{0,1\}^\star$

Notation: $\mathcal{M} \longrightarrow \langle \mathcal{M} \rangle \in \{0,1\}^\star$ $\qquad$ $w \in \{0,1\}^\star \longrightarrow T_w$

Diagonalsprache:

$L_d = \{w \in \{0,1\}^\star \mid T_w \text{ akzeptiert } w \text{ nicht}\}$

VL: L_d ist nicht entscheidbar. $\qquad$ **ÜB**: L_d^c ist semi-entscheidbar.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Halteproblem

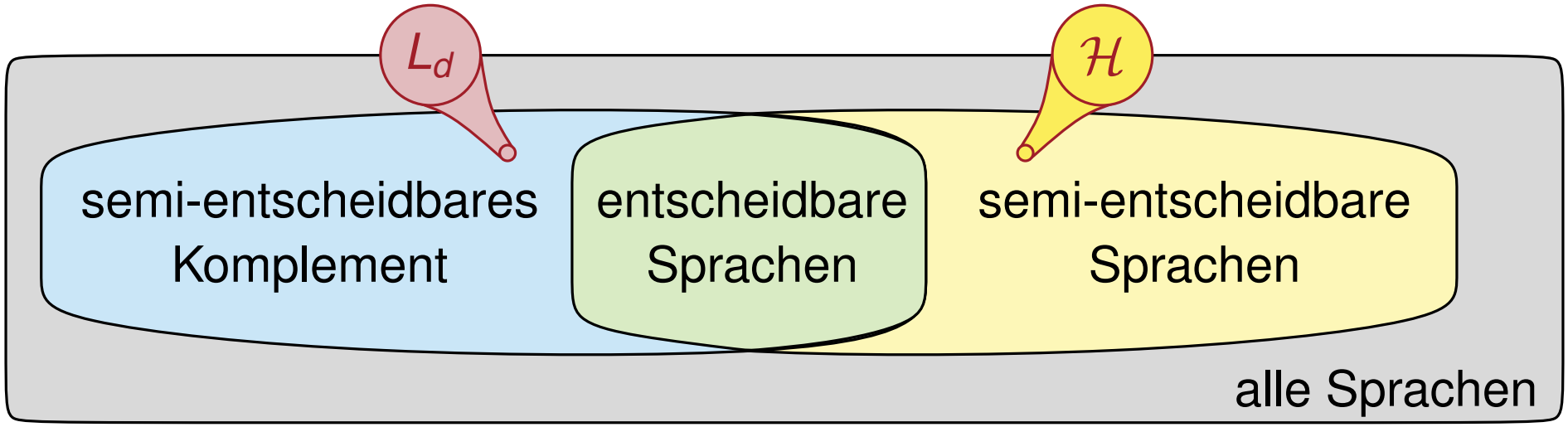
Sprache des Halteproblems:

$\mathcal{H} = \{wv \in \{0,1\}^\star \mid T_w \text{ hält bei Eingabe } v\}$

universelle Turing-Maschine:

bei Eingabe (w, v) wird T_w mit Eingabe v simuliert

VL: $\mathcal{H}$ ist nicht entscheidbar. **ÜB**: $\mathcal{H}^c$ ist nicht semi-entscheidbar.
universelle Turing-Maschine: $\mathcal{H}$ ist semi-entscheidbar.



Universelle Sprache

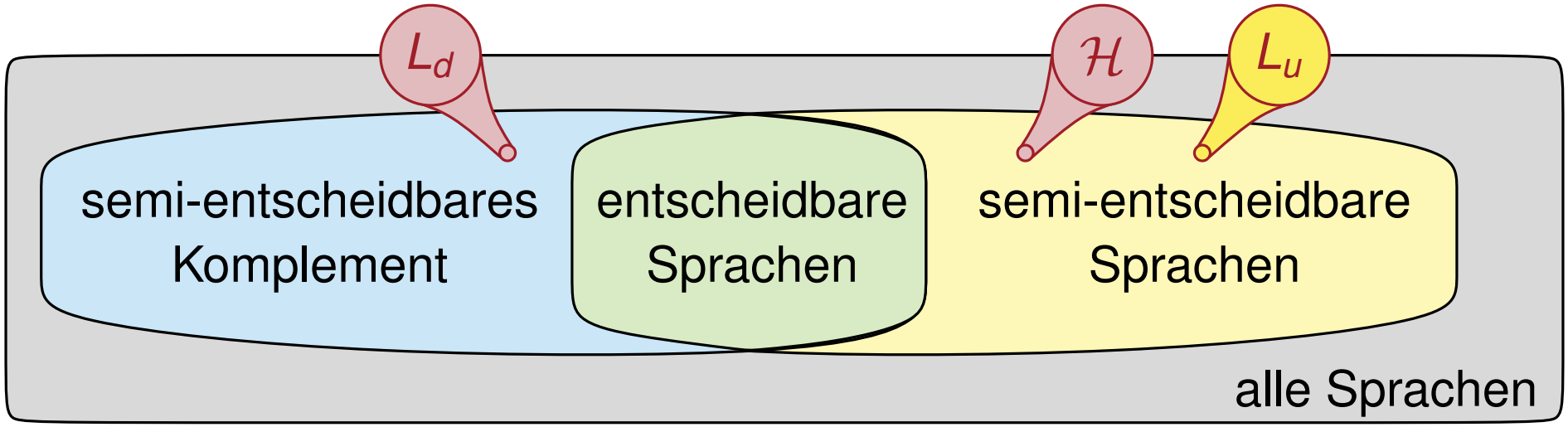
Die universelle Sprache:

$$L_u = \{wv \in \{0,1\}^\star \mid v \in L(T_w)\}$$
$$= \{wv \in \{0,1\}^\star \mid T_w \text{ hält und akzeptiert bei Eingabe } v\}$$

Vergleiche:

$$\mathcal{H} = \{wv \in \{0,1\}^\star \mid T_w \text{ hält bei Eingabe } v\}$$

VL: L_u ist nicht entscheidbar. **VL**: L_u ist semi-entscheidbar.



Post'sches Korrespondenzproblem

Das Problem Π:

1:	2:	3:
1 / 101	10 / 00	011 / 11

Lösungen:

Indexfolge

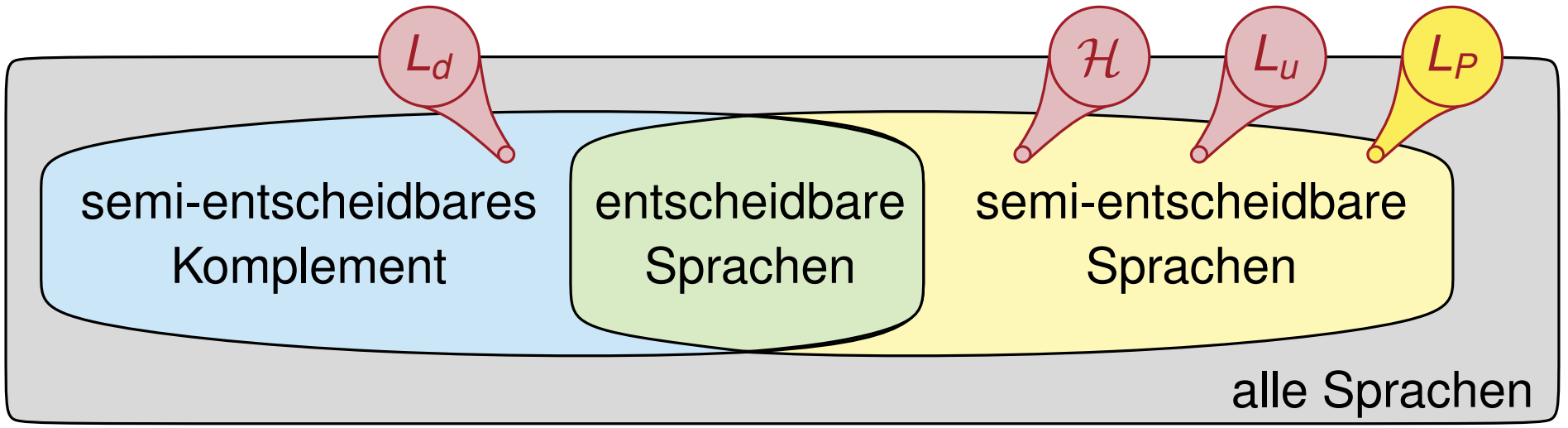
1 3 2 3

1	011	10	011
101	11	00	11

Die zugehörige Sprache:

$L_P := L[\Pi, s] = \{x \in \Sigma^\star \mid x \text{ kodiert Ja-Beispiel von } \Pi \text{ unter } s\}$

VL: L_P ist nicht entscheidbar, aber semi-entscheidbar.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$

$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Idee für N_1: Überprüfe, ob die Folge $0, 1, -1, 2, -2, \ldots$ eine Nullstelle von p enthält.

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Idee für N_1: Überprüfe, ob die Folge $0, 1, -1, 2, -2, \ldots$ eine Nullstelle von p enthält.

- Verfahren terminiert nur, wenn p eine ganzzahlige Nullstelle hat.
- Einschränkung auf ein bestimmtes Interval möglich

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Idee für N_1: Überprüfe, ob die Folge $0, 1, -1, 2, -2, \ldots$ eine Nullstelle von p enthält.

- Verfahren terminiert nur, wenn p eine ganzzahlige Nullstelle hat.
- Einschränkung auf ein bestimmtes Interval möglich

$\rightsquigarrow$ N_1 ist entscheidbar

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Idee für N_1: Überprüfe, ob die Folge $0, 1, -1, 2, -2, \ldots$ eine Nullstelle von p enthält.

- Verfahren terminiert nur, wenn p eine ganzzahlige Nullstelle hat.
- Einschränkung auf ein bestimmtes Interval möglich

$\rightsquigarrow$ N_1 ist entscheidbar

Dieses Intervall existiert für N nicht.

Hilberts zehntes Problem

Frage von Hilbert: Sind die folgenden Sprachen entscheidbar?

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$
$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

Beobachtung: N und N_1 sind semi-entscheidbar.

Idee für N_1: Überprüfe, ob die Folge $0, 1, -1, 2, -2, \ldots$ eine Nullstelle von p enthält.

- Verfahren terminiert nur, wenn p eine ganzzahlige Nullstelle hat.
- Einschränkung auf ein bestimmtes Interval möglich

$\rightsquigarrow$ N_1 ist entscheidbar

Dieses Intervall existiert für N nicht.

Matijasevič: N ist nicht entscheidbar

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Hilberts zehntes Problem

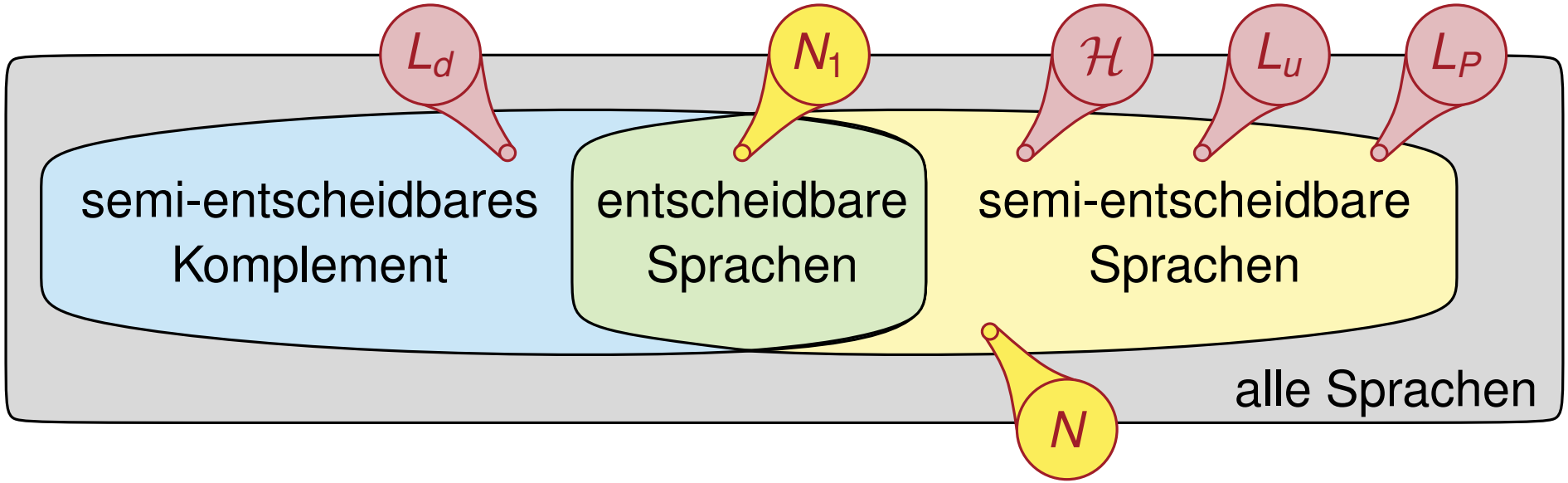
Polynome mit ganzzahligen Nullstellen:

$$N = \{p \mid p \text{ ist ganzzahliges Polynom mit ganzzahliger Nullstelle}\}$$

$$N_1 = \{p \mid p \text{ ist ganzzahliges Polynom über } x \text{ mit ganzzahliger Nullstelle}\}$$

letzte Folie: N_1 ist entscheidbar, N ist semi-entscheidbar.

Matijasevič: N ist nicht entscheidbar.



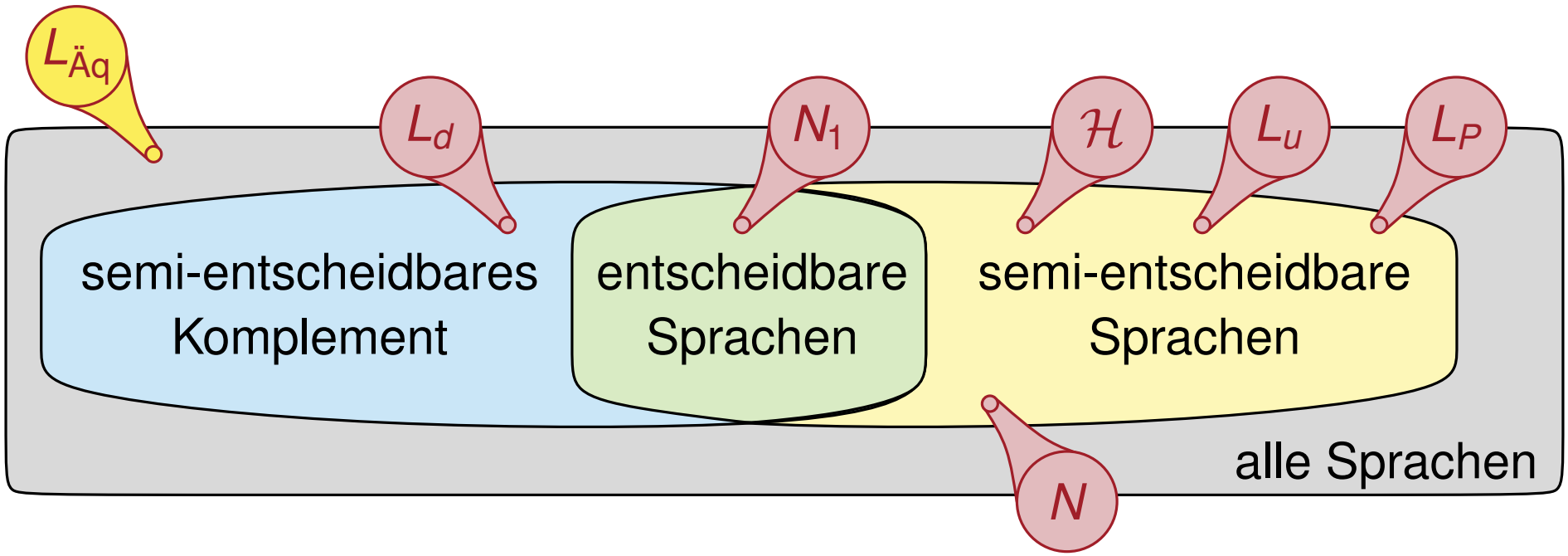
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Äquivalenzproblem

Äquivalenz von Turingmaschinen:

$$L_{\text{Äq}} := \{w\#v \mid w, v \in \{0,1\}^\star, L(T_w) = L(T_v)\}$$

ÜB: Sowohl $L_{\text{Äq}}$ als auch $L^c_{\text{Äq}}$ lassen sich auf L^c_u reduzieren.



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Entscheidbarkeit – Werkzeugkasten

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Satz von Rice

Satz von Rice: Sei S eine nicht-triviale Teilmenge von berechenbaren Funktionen. Dann ist

$$L(S) := \{w \in \{0,1\}^\star \mid T_w \text{ realisiert Funktion aus } S\}$$

nicht entscheidbar.

Beweisidee: Zeige, dass $L(S)$ Erweiterung von $\mathcal{H}_\varepsilon$ ist.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Satz von Rice

Satz von Rice: Sei S eine nicht-triviale Teilmenge von berechenbaren Funktionen. Dann ist

$$L(S) := \{w \in \{0,1\}^\star \mid T_w \text{ realisiert Funktion aus } S\}$$

nicht entscheidbar.

Beweisidee: Zeige, dass $L(S)$ Erweiterung von $\mathcal{H}_\varepsilon$ ist.

Beispiel 1: $L_1 = \{w \in \{0,1\}^\star \mid T_w \text{ berechnet 17 bei Eingabe 42}\}$

- L_1 ist nicht berechenbar nach Satz von Rice.

Satz von Rice

Satz von Rice: Sei S eine nicht-triviale Teilmenge von berechenbaren Funktionen. Dann ist

$$L(S) := \{w \in \{0,1\}^\star \mid T_w \text{ realisiert Funktion aus } S\}$$

nicht entscheidbar.

Beweisidee: Zeige, dass $L(S)$ Erweiterung von $\mathcal{H}_\varepsilon$ ist.

Beispiel 1: $L_1 = \{w \in \{0,1\}^\star \mid T_w \text{ berechnet 17 bei Eingabe 42}\}$

- L_1 ist nicht berechenbar nach Satz von Rice.

Beispiel 2: $L_2 = \{w \in \{0,1\}^\star \mid T_w \text{ macht 17 Schritte bei Eingabe 42}\}$

- Keine Aussage über L_2 durch Satz von Rice.
- Ist L_2 berechenbar?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Zusammenfassung (Werkzeugkasten)

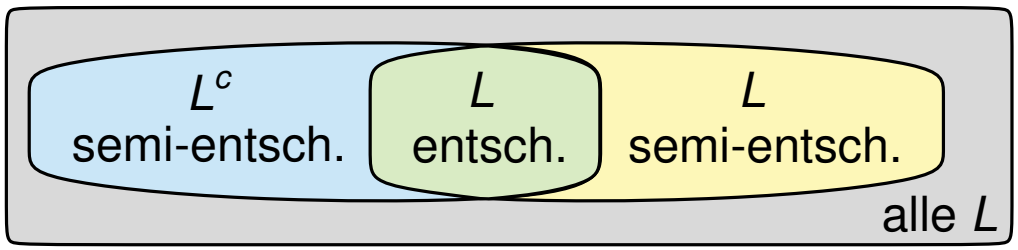
Karlsruher Institut für Technologie

L ist entscheidbar

- Konstruiere TM oder NTM, die L entscheidet.
- Zeige L, L^c semi-entscheidbar.

L ist semi-entscheidbar

- Konstruiere TM oder NTM, die L erkennt.



L ist nicht entscheidbar

- Benutze Satz von Rice.
- Reduziere nicht entscheidbare Sprache N auf L.
  - Konstruiere (N)TM für N aus (N)TM für L.

L ist nicht semi-entscheidbar

- Zeige L nicht entscheidbar und L^c semi-entscheidbar.
- Reduziere nicht semi-entscheidbare Sprache auf L.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik