Karlsruher Institut für Technologie

7. Übung – TGI

Lorenz Hübschle-Schneider, Tobias Maier

INSTITUT FÜR THEORETISCHE INFORMATIK, PROF. SANDERS

Organisatorisches

Da es wohl noch nicht klar genug war:

Verspätete Abgaben geben keine Punkte!

(Wenn euer Tutor ganz besonders nett ist,
korrigiert er es vielleicht aber trotzdem)

Universelle Turingmaschinen

Turingmaschinen, die Turingmaschinen simulieren, die Turingmaschinen simulieren, ...

Reale Beispiele:

Universelle Turingmaschinen

Turingmaschinen, die Turingmaschinen simulieren, die Turingmaschinen simulieren, ...

Reale Beispiele:

- Interpreter für Scriptsprachen
- Emulatoren

Universelle Turingmaschinen

Turingmaschinen, die Turingmaschinen simulieren, die Turingmaschinen simulieren, ...

Reale Beispiele:

- Interpreter für Scriptsprachen
- Emulatoren

Andere Beispiele:

- Turingmaschinensimulatoren
- Game of Life in Game of Life `https://youtu.be/xP5-iIeKXE8`

Quines

Können wir ein selbstreplizierendes Programm schreiben, also eines, das seinen eigenen Quellcode ausgibt?

Quines

Können wir ein selbstreplizierendes Programm schreiben, also eines, das seinen eigenen Quellcode ausgibt?

Ja! Diese Programme heißen Quines. In Python:

```
s = 's = %r\nprint(s%%s)'
print(s%s)
```

Quines

Können wir ein selbstreplizierendes Programm schreiben, also eines, das seinen eigenen Quellcode ausgibt?

Ja! Diese Programme heißen Quines. In Python:

```
s = 's = %r\nprint(s%%s)'
print(s%s)
```

Rekursionssatz: Es gibt Turingmaschinen, die ihre eigene Gödelnummer aufs Band schreiben

$\Rightarrow$ Quines gibt es in jedem turingmächtigen System

Quines

Können wir ein selbstreplizierendes Programm schreiben, also eines, das seinen eigenen Quellcode ausgibt?

Ja! Diese Programme heißen Quines. In Python:

```
s = 's = %r\nprint(s%%s)'
print(s%s)
```

Rekursionssatz: Es gibt Turingmaschinen, die ihre eigene Gödelnummer aufs Band schreiben, und jede TM lässt sich in eine äquivalente TM überführen, die diese Eigenschaft erfüllt

$\Rightarrow$ Quines gibt es in jedem turingmächtigen System

Quines

Können wir ein selbstreplizierendes Programm schreiben, also eines, das seinen eigenen Quellcode ausgibt?

Ja! Diese Programme heißen Quines. In Python:

```
s = 's = %r\nprint(s%%s)'
print(s%s)
```

Rekursionssatz: Es gibt Turingmaschinen, die ihre eigene Gödelnummer aufs Band schreiben, und jede TM lässt sich in eine äquivalente TM überführen, die diese Eigenschaft erfüllt

$\Rightarrow$ Quines gibt es in jedem turingmächtigen System
$\Rightarrow$ TMs dürfen ihre eigene Beschreibung verwenden (vgl. Reflections)

Unentscheidbarkeit

Gegeben zwei Konfigurationen in Game of Life. Können diese ineinander überführt werden?

Unentscheidbar!

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine (oder eines Algorithmus in einem anderen Berechenbarkeitsmodell) algorithmisch zu entscheiden.”

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine (oder eines Algorithmus in einem anderen Berechenbarkeitsmodell) algorithmisch zu entscheiden.”

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine algorithmisch zu entscheiden.”

- Klasse unentscheidbarer Sprachen

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine algorithmisch zu entscheiden.”

- Klasse unentscheidbarer Sprachen
- Sprachen von Turingmaschinen abhängig von der berechneten Funktion

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine algorithmisch zu entscheiden.”

- Klasse unentscheidbarer Sprachen
- Sprachen von Turingmaschinen abhängig von der berechneten Funktion
- Vage gefasst $\Rightarrow$ **nicht direkt einsetzen**

Der Satz von Rice

Ungefähr: ”Es ist unmöglich, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine algorithmisch zu entscheiden.”

- Klasse unentscheidbarer Sprachen
- Sprachen von Turingmaschinen abhängig von der berechneten Funktion
- Vage gefasst $\Rightarrow$ **nicht direkt einsetzen**
- Stattdessen merkt euch die folgenden Beweisschemata

Es gibt keine perfekten Virenscanner

Virus $\Leftrightarrow$ Turingmaschine die `VIRUS` auf das Band schreibt.

Es gibt keine perfekten Virenscanner

Virus $\Leftrightarrow$ Turingmaschine die `VIRUS` auf das Band schreibt.

Satz von Rice $\Rightarrow$ $L_{\texttt{VIRUS}} = \{\langle M \rangle \mid M \text{ ist ein Virus}\}$ ist unentscheidbar

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

- T_{IN} schreibt $\langle M_{\text{IN}} \rangle \# w_{\text{IN}}$ auf das Band

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

- T_{IN} schreibt $\langle M_{\text{IN}} \rangle \# w_{\text{IN}}$ auf das Band
- T_{IN} simuliert M_{IN} auf der Eingabe w_{IN}

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

- T_{IN} schreibt $\langle M_{\text{IN}} \rangle \# w_{\text{IN}}$ auf das Band
- T_{IN} simuliert M_{IN} auf der Eingabe w_{IN}
- T_{IN} löscht den Bandinhalt und schreibt `VIRUS`

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

- T_{IN} schreibt $\langle M_{\text{IN}} \rangle \# w_{\text{IN}}$ auf das Band
- T_{IN} simuliert M_{IN} auf der Eingabe w_{IN}
- T_{IN} löscht den Bandinhalt und schreibt `VIRUS`

Beobachtung: T_{IN} ist genau dann ein Virus, wenn $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ hält.

Beweis über das Halteproblem

Ansatz: $L_{\texttt{VIRUS}}$ entscheidbar $\Rightarrow$ Halteproblem lösbar!

Sei $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ eine Instanz des Halteproblems
mit Hilfe dieser Instanz konstruieren wir die TM T_{IN}

- T_{IN} schreibt $\langle M_{\text{IN}} \rangle \# w_{\text{IN}}$ auf das Band
- T_{IN} simuliert M_{IN} auf der Eingabe w_{IN}
- T_{IN} löscht den Bandinhalt und schreibt `VIRUS`

Beobachtung: T_{IN} ist genau dann ein Virus, wenn $(\langle M_{\text{IN}} \rangle, w_{\text{IN}})$ hält.

$T_{\texttt{SCAN}}$ entscheidet $L_{\texttt{VIRUS}} \Rightarrow T_{\texttt{SCAN}}$ entscheidet das Halteproblem

Einschub: Rekursionssatz

Kurzfassung:

Für jede partiell berechenbare Funktion f existiert eine TM $T_{\text{rec}(f)}$

Einschub: Rekursionssatz

Kurzfassung:

Für jede partiell berechenbare Funktion f existiert eine TM $T_{\text{rec}(f)}$

- $T_{\text{rec}(f)}$ schreibt $\langle T_{\text{rec}(f)} \rangle$ auf das Band

Einschub: Rekursionssatz

Kurzfassung:

Für jede partiell berechenbare Funktion f existiert eine TM $T_{\text{rec}(f)}$

- $T_{\text{rec}(f)}$ schreibt $\langle T_{\text{rec}(f)} \rangle$ auf das Band
- $T_{\text{rec}(f)}$ wendet die Funktion f auf $\langle T_{\text{rec}(f)} \rangle$ an

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

- $T_{\texttt{HIDDEN}}$ schreibt zunächst $\langle T_{\texttt{HIDDEN}} \rangle$ auf das Band

Rekursionssatz

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

- $T_{\texttt{HIDDEN}}$ schreibt zunächst $\langle T_{\texttt{HIDDEN}} \rangle$ auf das Band
- $T_{\texttt{HIDDEN}}$ schreibt nun $\langle T_{\texttt{SCAN}} \rangle \#$ links daneben

bei der Konstruktion ist $T_{\texttt{SCAN}}$ bekannt

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

- $T_{\texttt{HIDDEN}}$ schreibt zunächst $\langle T_{\texttt{HIDDEN}} \rangle$ auf das Band
- $T_{\texttt{HIDDEN}}$ schreibt nun $\langle T_{\texttt{SCAN}} \rangle \#$ links daneben
- $T_{\texttt{HIDDEN}}$ simuliert $T_{\texttt{SCAN}}$ auf der Eingabe $\langle T_{\texttt{HIDDEN}} \rangle$

Universelle Turingmaschinen

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

- $T_{\texttt{HIDDEN}}$ schreibt zunächst $\langle T_{\texttt{HIDDEN}} \rangle$ auf das Band
- $T_{\texttt{HIDDEN}}$ schreibt nun $\langle T_{\texttt{SCAN}} \rangle \#$ links daneben
- $T_{\texttt{HIDDEN}}$ simuliert $T_{\texttt{SCAN}}$ auf der Eingabe $\langle T_{\texttt{HIDDEN}} \rangle$
  - $T_{\texttt{SCAN}}$ akzeptiert $\Rightarrow$ $T_{\texttt{HIDDEN}}$ schreibt `FALSE POSITIVE`
  - $T_{\texttt{SCAN}}$ akzeptiert nicht $\Rightarrow$ $T_{\texttt{HIDDEN}}$ schreibt `VIRUS`

Beweis mit Rekursionssatz

Angenommen es gäbe die TM $T_{\texttt{SCAN}}$, die $L_{\texttt{VIRUS}}$ entscheidet

Ansatz: konstruiere TM $T_{\texttt{HIDDEN}}$, die von $T_{\texttt{SCAN}}$ falsch klassifiziert wird

- $T_{\texttt{HIDDEN}}$ schreibt zunächst $\langle T_{\texttt{HIDDEN}} \rangle$ auf das Band
- $T_{\texttt{HIDDEN}}$ schreibt nun $\langle T_{\texttt{SCAN}} \rangle \#$ links daneben
- $T_{\texttt{HIDDEN}}$ simuliert $T_{\texttt{SCAN}}$ auf der Eingabe $\langle T_{\texttt{HIDDEN}} \rangle$
  - $T_{\texttt{SCAN}}$ akzeptiert $\Rightarrow$ $T_{\texttt{HIDDEN}}$ schreibt `FALSE POSITIVE`
  - $T_{\texttt{SCAN}}$ akzeptiert nicht $\Rightarrow$ $T_{\texttt{HIDDEN}}$ schreibt `VIRUS`

Da $T_{\texttt{SCAN}}$ die Eingabe $T_{\texttt{HIDDEN}}$ falsch klassifiziert, kann $T_{\texttt{SCAN}}$ die Sprache $L_{\texttt{VIRUS}}$ nicht korrekt entscheiden.

Widerspruch