Turing-Vollständigkeit von Zwei-Zähler-Systemen

Proseminar Automatentheorie

Albert-Ludwigs-Universität Freiburg

Robin Krahl

Institut für Informatik

Wintersemester 2015/16

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Überblick

Motivation

Grundlagen
- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem
- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Motivation

- Systeme mit nur zwei (unendlich großen) Integer-Variablen und den Operationen Inkrementieren und Dekrementieren
  - Registermaschinen
  - ...

Motivation

- Systeme mit nur zwei (unendlich großen) Integer-Variablen und den Operationen Inkrementieren und Dekrementieren
  - Registermaschinen
  - ...
- ... sind Turing-vollständig!

Motivation

Motivation

- Systeme mit nur zwei (unendlich großen) Integer-Variablen und den Operationen Inkrementieren und Dekrementieren
  - Registermaschinen
  - ...
- ... sind Turing-vollständig!
  (können jede Berechnung durchführen, die man auch mit einer Turingmaschine durchführen kann)

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Pushdown-Systeme I

$\mathcal{P} = (P, \Gamma, \Delta)$

- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times P \times \Gamma^*)$

Pushdown-Systeme I

$\mathcal{P} = (P, \Gamma, \Delta)$

- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times P \times \Gamma^*)$
- bei jedem Übergang: oberstes Zeichen vom Stack lesen, beliebig viele Zeichen schreiben

Pushdown-Systeme I

$\mathcal{P} = (P, \Gamma, \Delta)$

- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times P \times \Gamma^*)$
- bei jedem Übergang: oberstes Zeichen vom Stack lesen, beliebig viele Zeichen schreiben

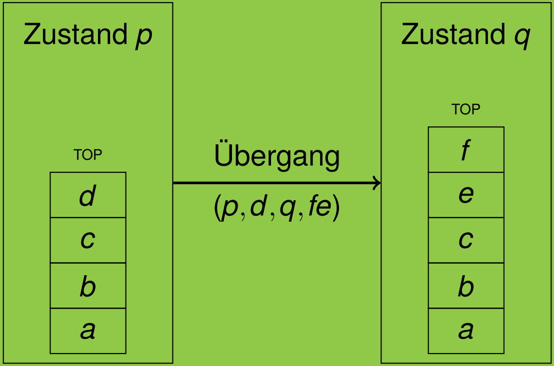
Pushdown-Systeme I

$\mathcal{P} = (P, \Gamma, \Delta)$

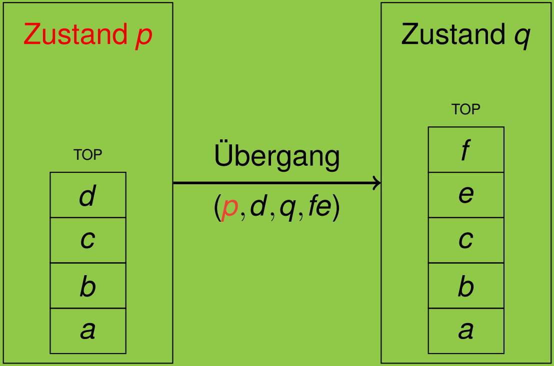
- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times P \times \Gamma^*)$
- bei jedem Übergang: oberstes Zeichen vom Stack lesen, beliebig viele Zeichen schreiben

Pushdown-Systeme II

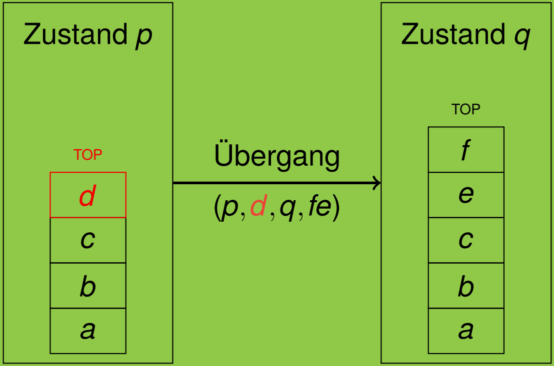
Beispiel Pushdown-Systeme



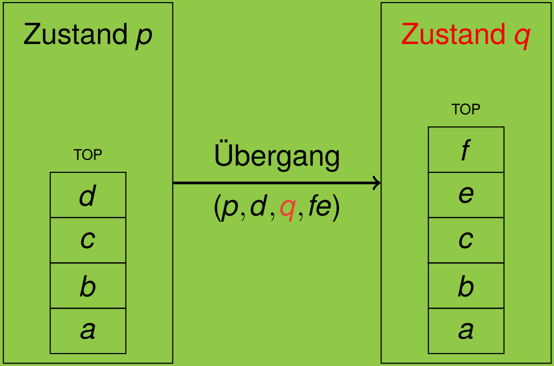
Beispiel Pushdown-Systeme



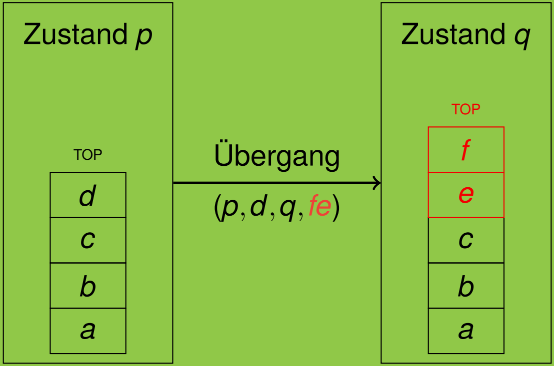
Beispiel Pushdown-Systeme



Beispiel Pushdown-Systeme



Beispiel Pushdown-Systeme



n-Pushdown-Systeme I

UNI FREIBURG

$\mathcal{P}_n = (P, \Gamma, \Delta)$

- nicht einer, sondern n verschiedene Stacks

n-Pushdown-Systeme I

$\mathcal{P}_n = (P, \Gamma, \Delta)$

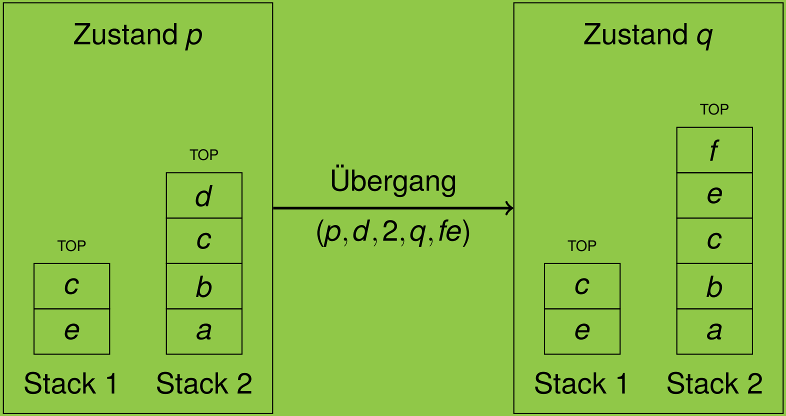
- nicht einer, sondern n verschiedene Stacks
- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times \{1, ..., n\} \times P \times \Gamma^*)$

n-Pushdown-Systeme I

$\mathcal{P}_n = (P, \Gamma, \Delta)$

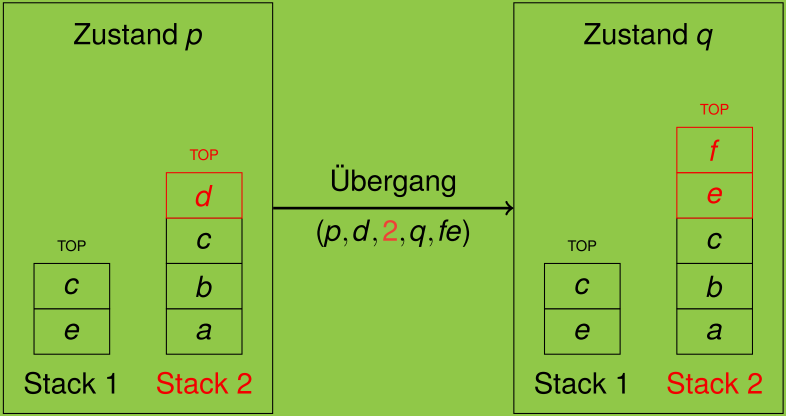
- nicht einer, sondern *n* verschiedene Stacks
- Zustände P
- Stackalphabet Γ
- Übergangsrelation $\Delta \subseteq (P \times \Gamma \times \{1, ..., n\} \times P \times \Gamma^*)$
- bei jedem Übergang: oberstes Zeichen vom Stack i lesen, beliebig viele Zeichen schreiben

Beispiel n-Pushdown-Systeme



n-Pushdown-Systeme II

Beispiel n-Pushdown-Systeme



n-Pushdown-Systeme III

Konfiguration

- Konfiguration $c = (p, w_1, ..., w_n)$ mit $p \in P$, $w_i \in \Gamma^*$ $(1 \leq i \leq n)$
- Status zu einem bestimmten Zeitpunkt
  - aktueller Zustand p
  - Inhalt w_i des Stacks i

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

n-Zähler-Systeme I

$\mathcal{Z} = (P, \Delta)$

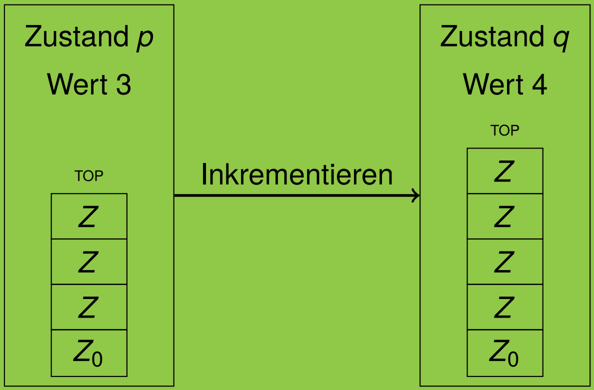
- entspricht n-Pushdown-System $\mathcal{P} = (P, \Gamma, \Delta)$
- nur zwei Stacksymbole: $\Gamma = \{Z, Z_0\}$

n-Zähler-Systeme I

$\mathcal{Z} = (P, \Delta)$

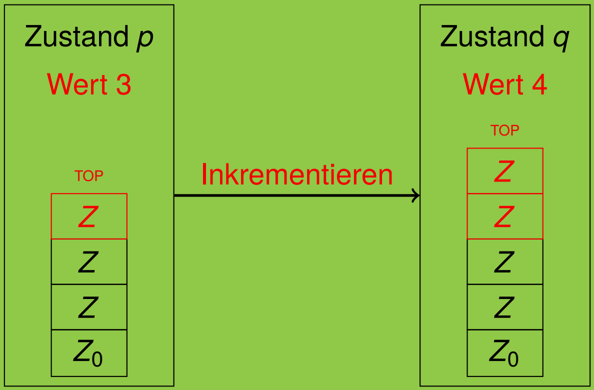
- entspricht n-Pushdown-System $\mathcal{P} = (P, \Gamma, \Delta)$
- nur zwei Stacksymbole: $\Gamma = \{Z, Z_0\}$
- Stackinhalt $Z^m Z_0$ codiert Zahl m
  - $Z_0 = 0$
  - $ZZ_0 = 1$
  - $ZZZZ_0 = 3$
  - ...
  - ZZ_0ZZ verboten

Beispiel n-Zähler-Systeme



n-Zähler-Systeme II

Beispiel n-Zähler-Systeme



Überblick

Turingmaschinen

$M = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup)$

- Zustandsmenge Q
- Anfangszustand q_0
- Bandalphabet Γ
- Eingabealphabet $\Sigma \subset \Gamma$
- Leerzeichen $\sqcup \in \Gamma \setminus \Sigma$
- Übergangsmege $\delta \subseteq (Q \times \Gamma \times Q \times \Gamma \times \{L, R, N\})$

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Erreichbarkeitsproblem I

Erreichbarkeitsproblem

- 2-Zähler-System $\mathcal{Z} = (P, \Delta)$
- Konfigurationen c_1, c_2 von $\mathcal{Z}$

Erreichbarkeitsproblem I

Erreichbarkeitsproblem

- 2-Zähler-System $\mathcal{Z} = (P, \Delta)$
- Konfigurationen c_1, c_2 von $\mathcal{Z}$
- Kann ich c_2 erreichen, wenn ich das System mit c_1 starte?

Erreichbarkeitsproblem I

Erreichbarkeitsproblem

- 2-Zähler-System $\mathcal{Z} = (P, \Delta)$
- Konfigurationen c_1, c_2 von $\mathcal{Z}$
- Kann ich c_2 erreichen, wenn ich das System mit c_1 starte?

Behauptung

Das Erreichbarkeitsproblem für 2-Zähler-Systeme ist nicht entscheidbar.

Erreichbarkeitsproblem II

Beweisansatz

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Turingmaschine $\longrightarrow$ 2-Pushdown-System I

Ausgangslage

- gegeben: Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup)$
- gesucht: 2-Pushdown-System $\mathcal{P}_M = (P, \Gamma', \Delta)$ und Konfigurationen c_1, c_2

Turingmaschine $\longrightarrow$ 2-Pushdown-System I

Ausgangslage

- gegeben: Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup)$
- gesucht: 2-Pushdown-System $\mathcal{P}_M = (P, \Gamma', \Delta)$ und Konfigurationen c_1, c_2
- c_2 soll von c_1 aus genau dann erreichbar sein, wenn M hält

Turingmaschine $\longrightarrow$ 2-Pushdown-System I

UNI FREIBURG

Ausgangslage

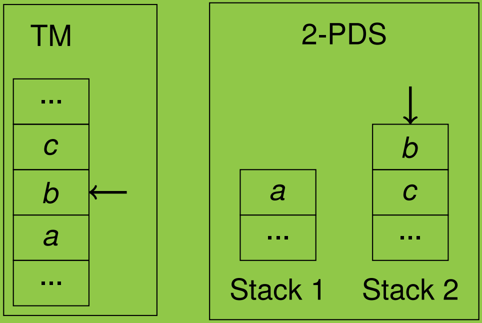
- gegeben: Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup)$
- gesucht: 2-Pushdown-System $\mathcal{P}_M = (P, \Gamma', \Delta)$ und Konfigurationen c_1, c_2
- c_2 soll von c_1 aus genau dann erreichbar sein, wenn M hält

Hinweis

Halteproblem für Turingmaschinen nicht entscheidbar!

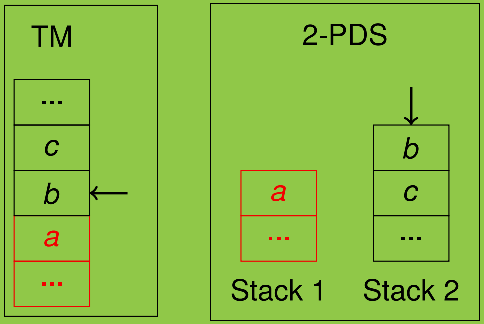
Turingmaschine $\longrightarrow$ 2-Pushdown-System II

Veranschaulichung



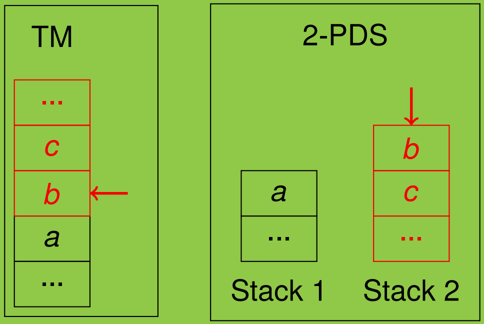
Turingmaschine ⟶ 2-Pushdown-System II

Veranschaulichung



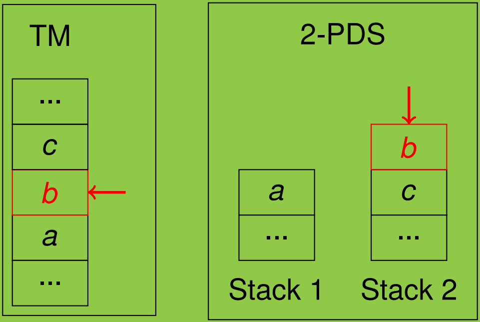
Turingmaschine ⟶ 2-Pushdown-System II

Veranschaulichung



Turingmaschine ⟶ 2-Pushdown-System II

Veranschaulichung



Turingmaschine $\longrightarrow$ 2-Pushdown-System III

Vorgehensweise

- Übergänge der Turingmaschine in 2-PDS abbilden
- Stack 1: Band links des Lesekopfes
- Stack 2: Band am Lesekopf und rechts davon

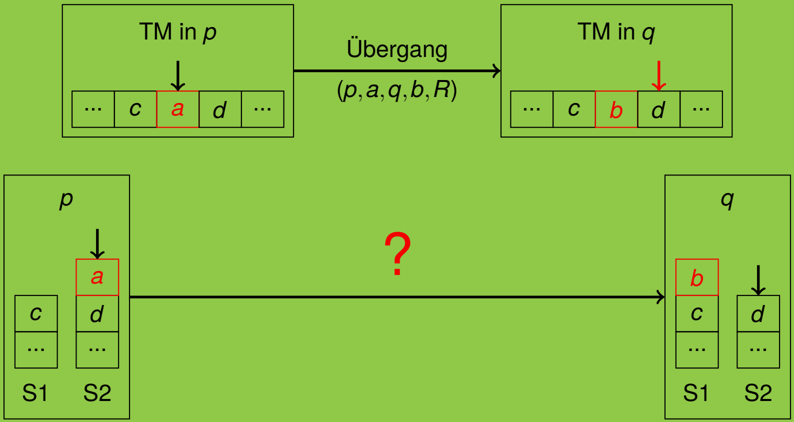
Turingmaschine $\longrightarrow$ 2-Pushdown-System III

Vorgehensweise

- Übergänge der Turingmaschine in 2-PDS abbilden
- Stack 1: Band links des Lesekopfes
- Stack 2: Band am Lesekopf und rechts davon
- Problem: 2-PDS kann immer nur auf einem Stack arbeiten

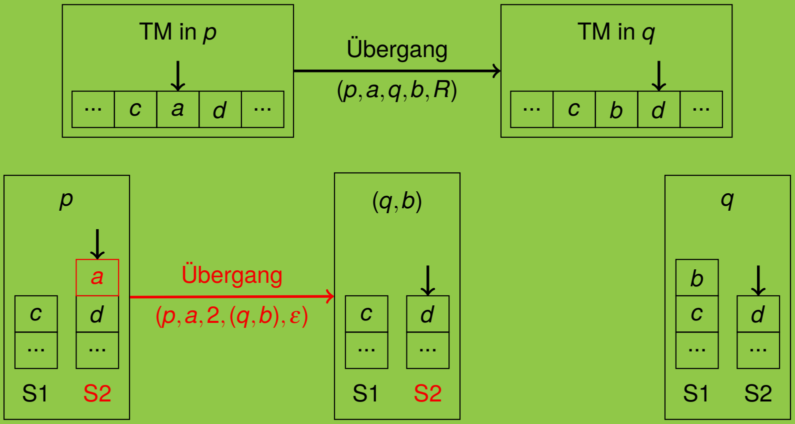
Turingmaschine $\longrightarrow$ 2-Pushdown-System IV

Beispiel: Kopfbewegung nach rechts

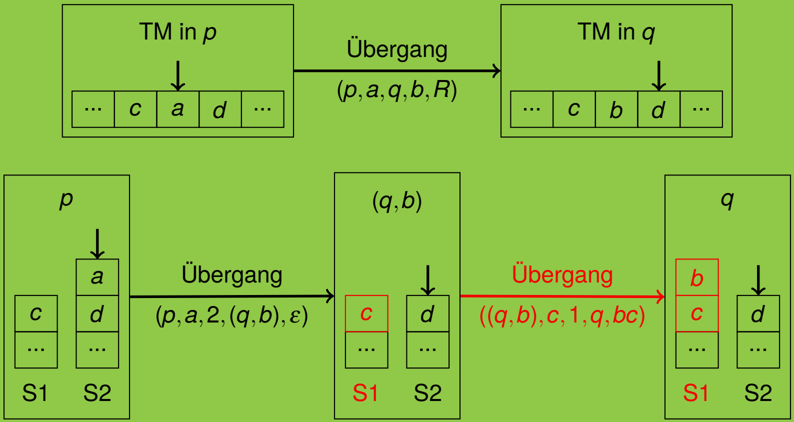


Turingmaschine $\longrightarrow$ 2-Pushdown-System IV

Beispiel: Kopfbewegung nach rechts



Beispiel: Kopfbewegung nach rechts



Turingmaschine $\longrightarrow$ 2-Pushdown-System III

Vorgehensweise

- Übergänge der Turingmaschine in 2-PDS abbilden
- Stack 1: Band links des Lesekopfes
- Stack 2: Band am Lesekopf und rechts davon
- Problem: 2-PDS kann immer nur auf einem Stack arbeiten
- Lösung: zusätzliche Zwischenzustände $(q,b) \in (Q \times \Gamma)$, die sich einen Buchstaben merken

Turingmaschine $\longrightarrow$ 2-Pushdown-System III

Vorgehensweise

- Übergänge der Turingmaschine in 2-PDS abbilden
- Stack 1: Band links des Lesekopfes
- Stack 2: Band am Lesekopf und rechts davon
- Problem: 2-PDS kann immer nur auf einem Stack arbeiten
- Lösung: zusätzliche Zwischenzustände $(q, b) \in (Q \times \Gamma)$, die sich einen Buchstaben merken
- $\mathcal{P}_M = (Q \cup (Q \times \Gamma), \Gamma, \Delta)$

Turingmaschine $\longrightarrow$ 2-Pushdown-System V

Beispiel: Kopfbewegung nach rechts (Fortsetzung)

Übergang der Turingmaschine $(p, a, q, b, R) \in \delta$

- Zeichen an der alten Kopfposition lesen und löschen: neuer Übergang $(p, a, 2, (q, b), \varepsilon)$

Turingmaschine $\longrightarrow$ 2-Pushdown-System V

Beispiel: Kopfbewegung nach rechts (Fortsetzung)

Übergang der Turingmaschine $(p, a, q, b, R) \in \delta$

- Zeichen an der alten Kopfposition lesen und löschen: neuer Übergang $(p, a, 2, (q, b), \varepsilon)$
- Neues Zeichen hinter der neuen Kopfposition einfügen: neuer Übergang $((q, b), c, 1, q, bc)$

Turingmaschine $\longrightarrow$ 2-Pushdown-System V

Beispiel: Kopfbewegung nach rechts (Fortsetzung)

Übergang der Turingmaschine $(p,a,q,b,R) \in \delta$

- Zeichen an der alten Kopfposition lesen und löschen: neuer Übergang $(p,a,2,(q,b),\varepsilon)$
- Neues Zeichen hinter der neuen Kopfposition einfügen: neuer Übergang $((q,b),c,1,q,bc)$
- ... für alle $c \in \Gamma$

UNI FREIBURG

Beispiel: Kopfbewegung nach rechts (Fortsetzung)

Übergang der Turingmaschine $(p,a,q,b,R) \in \delta$

- Zeichen an der alten Kopfposition lesen und löschen: neuer Übergang $(p,a,2,(q,b),\varepsilon)$
- Neues Zeichen hinter der neuen Kopfposition einfügen: neuer Übergang $((q,b),c,1,q,bc)$
- ... für alle $c \in \Gamma$
- ... analog für keine Kopfbewegung (N) oder Kopfbewegung nach links (L)

Überblick

2-Pushdown-System $\longrightarrow$ 4-Zähler-System I

Ausgangslage

- gegeben: 2-Pushdown-System $\mathcal{P} = (P, \Gamma', \Delta)$ und Konfigurationen c_1, c_2

2-Pushdown-System $\longrightarrow$ 4-Zähler-System I

Ausgangslage

- gegeben: 2-Pushdown-System $\mathcal{P} = (P, \Gamma', \Delta)$ und Konfigurationen c_1, c_2
- gesucht: 4-Zähler-System $\mathcal{Z}_4 = (Q, \Delta')$ und Konfigurationen c'_1, c'_2
- c'_2 soll von c'_1 aus genau dann erreichbar sein, wenn c_2 von c_1 aus erreichbar ist

2-Pushdown-System $\longrightarrow$ 4-Zähler-System II

Vorgehensweise

- Simuliere je einen Stack des 2-PDS durch ein 2-Zähler-System $\mathcal{Z}_2 = (Q, \Delta')$
- Problem: Stack des 2-PDS darf beliebige Zeichen enthalten

2-Pushdown-System $\longrightarrow$ 4-Zähler-System II

UNI FREIBURG

Vorgehensweise

- Simuliere je einen Stack des 2-PDS durch ein 2-Zähler-System $\mathcal{Z}_2 = (Q, \Delta')$
- Problem: Stack des 2-PDS darf beliebige Zeichen enthalten
- Lösung: Codierung!

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)
- o. B. d. A. sei $k = 9$

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)
- o. B. d. A. sei $k = 9$

Beispiel

- $\Gamma = \{a, b, c, d, e, f, g, h, i\}$
- *iad* entspricht 419

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)
- o. B. d. A. sei $k = 9$

Beispiel

- $\Gamma = \{a,b,c,d,e,f,g,h,i\}$
- iad entspricht 419

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)
- o. B. d. A. sei $k = 9$

Beispiel

- $\Gamma = \{a, b, c, d, e, f, g, h, i\}$
- iad entspricht 419

2-Pushdown-System $\longrightarrow$ 4-Zähler-System III

Codierung

- betrachte $k = |\Gamma|$
- nummeriere Elemente von Γ von 1 bis k
- jedes $w \in \Gamma^*$ entspricht dann einer Zahl im $(k+1)$-er System (oberstes Zeichen ist niedrigste Ziffer)
- o. B. d. A. sei $k = 9$

Beispiel

- $\Gamma = \{a, b, c, d, e, f, g, h, i\}$
- iad entspricht 419

2-Pushdown-System $\longrightarrow$ 4-Zähler-System IV

Operationen

Stackinhalt 419

- oberstes Zeichen ermitteln — 9
- oberstes Zeichen löschen — 41
- neues Zeichen x schreiben — 419x

2-Pushdown-System $\longrightarrow$ 4-Zähler-System IV

Operationen

Stackinhalt 419

- oberstes Zeichen ermitteln — 419 mod 10 = 9
  Rest der Division durch 10
- oberstes Zeichen löschen — 41
- neues Zeichen x schreiben — $419x$

2-Pushdown-System $\longrightarrow$ 4-Zähler-System IV

Operationen

Stackinhalt 419

- oberstes Zeichen ermitteln — 419 mod 10 = 9
  Rest der Division durch 10
- oberstes Zeichen löschen — 419 div 10 = 41
  Division durch 10
- neues Zeichen x schreiben — 419x

2-Pushdown-System $\longrightarrow$ 4-Zähler-System IV

UNI FREIBURG

Operationen

Stackinhalt 419

- oberstes Zeichen ermitteln $419 \bmod 10 = 9$
  Rest der Division durch 10
- oberstes Zeichen löschen $419 \text{ div } 10 = 41$
  Division durch 10
- neues Zeichen x schreiben $419 \cdot 10 + x = 419x$
  Multiplikation mit 10, Addition des Zeichens

2-Pushdown-System $\longrightarrow$ 4-Zähler-System V

Umsetzung

- Herausforderung: Implementierung der Rechenoperationen

2-Pushdown-System $\longrightarrow$ 4-Zähler-System V

Umsetzung

- Herausforderung: Implementierung der Rechenoperationen
- Lösung: Nutze zweiten Zähler als Hilfsvariable

2-Pushdown-System $\longrightarrow$ 4-Zähler-System V

Umsetzung

- Herausforderung: Implementierung der Rechenoperationen
- Lösung: Nutze zweiten Zähler als Hilfsvariable
- also je ein Zähler für den Inhalt eines Stackes, außerdem ein Hilfszähler

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

4-Zähler-Systeme $\longrightarrow$ 2-Zähler-System I

Überlegung

- Ziel: vier Zähler durch zwei ersetzen

UNI FREIBURG

Überlegung

- Ziel: vier Zähler durch zwei ersetzen
- Codierung?

4-Zähler-Systeme $\longrightarrow$ 2-Zähler-System I

Überlegung

- Ziel: vier Zähler durch zwei ersetzen
- Codierung?
- Primfaktorzerlegung! $k = 2^{k_1} \cdot 3^{k_2} \cdot 5^{k_3} \cdot 7^{k_4}$

4-Zähler-Systeme $\longrightarrow$ 2-Zähler-System I

Überlegung

- Ziel: vier Zähler durch zwei ersetzen
- Codierung?
- Primfaktorzerlegung! $k = 2^{k_1} \cdot 3^{k_2} \cdot 5^{k_3} \cdot 7^{k_4}$
- Inkrementieren und Dekrementieren durch Multiplikation und Division

4-Zähler-Systeme $\longrightarrow$ 2-Zähler-System I

Überlegung

- Ziel: vier Zähler durch zwei ersetzen
- Codierung?
- Primfaktorzerlegung! $k = 2^{k_1} \cdot 3^{k_2} \cdot 5^{k_3} \cdot 7^{k_4}$
- Inkrementieren und Dekrementieren durch Multiplikation und Division
- ein Zähler zur Durchführung der Operationen (wie oben)

Überblick

Motivation

Grundlagen

- Pushdown-Systeme
- Zähler-Systeme
- Turingmaschinen

Erreichbarkeitsproblem

- Turingmaschine $\longrightarrow$ 2-Pushdown-System
- 2-Pushdown-System $\longrightarrow$ 4-Zähler-System
- 4-Zähler-System $\longrightarrow$ 2-Zähler-System
- Zusammenfassung

Zusammenfassung

- Wir können ausgehend von einer Turingmaschine M ein Zwei-Zähler-System $\mathcal{Z}$ mit Konfigurationen c_1, c_2 konstruieren, sodass c_2 genau dann von c_1 erreichbar ist, wenn M hält.

Zusammenfassung

- Wir können ausgehend von einer Turingmaschine M ein Zwei-Zähler-System $\mathcal{Z}$ mit Konfigurationen c_1, c_2 konstruieren, sodass c_2 genau dann von c_1 erreichbar ist, wenn M hält.
- Wenn das Erreichbarkeitsproblem für Zwei-Zähler-Systeme entscheidbar wäre, wäre daher auch das Halteproblem für Turingmaschinen entscheidbar.

Zusammenfassung

- Wir können ausgehend von einer Turingmaschine M ein Zwei-Zähler-System $\mathcal{Z}$ mit Konfigurationen c_1, c_2 konstruieren, sodass c_2 genau dann von c_1 erreichbar ist, wenn M hält.
- Wenn das Erreichbarkeitsproblem für Zwei-Zähler-Systeme entscheidbar wäre, wäre daher auch das Halteproblem für Turingmaschinen entscheidbar.
- Das ist es aber nicht.

Zusammenfassung

- Wir können ausgehend von einer Turingmaschine M ein Zwei-Zähler-System $\mathcal{Z}$ mit Konfigurationen c_1, c_2 konstruieren, sodass c_2 genau dann von c_1 erreichbar ist, wenn M hält.
- Wenn das Erreichbarkeitsproblem für Zwei-Zähler-Systeme entscheidbar wäre, wäre daher auch das Halteproblem für Turingmaschinen entscheidbar.
- Das ist es aber nicht.
- Also ist auch das Erreichbarkeitsproblem für Zwei-Zähler-Systeme unentscheidbar.

Turing-Vollständigkeit

- Turing-Vollständigkeit von Zwei-Zähler-Systemen

Turing-Vollständigkeit

- Turing-Vollständigkeit von Zwei-Zähler-Systemen
- Beweis bisher: Zwei-Zähler-Systeme können Turingmaschinen simulieren

Turing-Vollständigkeit

- Turing-Vollständigkeit von Zwei-Zähler-Systemen
- Beweis bisher: Zwei-Zähler-Systeme können Turingmaschinen simulieren
- Also: Zwei-Zähler-Systeme sind Turing-vollständig!

Quellen

- Wolfgang Thomas: *Applied Automata Theory*. http://drona.csa.iisc.ernet.in/%7Edeepakd/atc-common/wolfgang-aat.pdf
- V. Claus, E.-R. Olderog: *Informatik III: Theoretische Informatik.* http://proglang.informatik.uni-freiburg.de/teaching/info3/2015ws/material/Kap1-6.pdf