GZ Theoretische Informatik (WS16/17)

Lösungsvorschlag zu Aufgabenblatt 7

Aufgabe 1

Idee: Der L/S-Kopf von der TM ${}_{cd}T_{bba}$ läuft das Eingabeband von links nach rechts ab. Liest es die Zeichenfolge cd, wird d mit a ersetzt, dann c mit b. ${}_{cd}T_{bba}$ merkt sich, durch einen Zustand, dann das Zeichen links von d und ersetzt es mit b. Darauf hin bewegt sich der L/S-Kopf von rechts nach links und ersetzt das aktuelle Zeichen mit dem jeweiligen Zeichen t, welches es sich im Zustand q_t gemerkt hat. Hat die ${}_{cd}T_{bba}$ das leere Feld erreicht, schreibt es sein letztes Zeichen, wechselt in den Zustand q' und terminiert.
Erreicht ${}_{cd}T_{bba}$ hingegen im Zustand q das Ende des Eingabewortes, wurde cd nie gelesen und die TM wird im Zustand q divergieren.

Spezifikation: ${}_{cd}T_{bba}$ ist gegeben durch das Tupel $\{\Sigma, \Gamma, \#, Q, s, F, \Delta\}$

- $\Sigma := \{a, b, c, d\}$
- $\Gamma := \{a, b, c, d, \#\}$
- $Q := \{q, q_1, q_2, q_3\} \cup \{q_a, q_b, q_c, q_d\} \cup \{q'\}$
- $s = q$
- $F := \{q'\}$
- $\Delta :=$

(derz. Zust., gel. Zeichen)	neuer Zust.	geschr. Zeichen	Kopfbew.	Bemerkung
(q, t) with $t \in \Sigma \setminus c$	q	t	R	
(q, c)	q_1	c	R	erstes Vorkommen von c
(q_1, c)	q_1	c	R	
(q_1, d)	q_2	a	L	auf c folgendes d
(q_1, t) with $t \in \Sigma \setminus \{c, d\}$	q	t	R	auf c folgendes Zeichen ausser c, d
(q_2, c)	q_3	b	L	
(q_3, t) with $t \in \Sigma$	q_t	b	L	
$(q_3, \#)$	q'	b	B	cd war am Wortanfang
(q_t, s) with $t, s \in \Sigma$	q_s	t	L	
$(q_t, \#)$ with $t \in \Sigma$	q'	t	B	Terminierung am linkesten Zeichen
$(q, \#)$	q	$\#$	B	ganzes Wort gelesen, kein cd

Aufgabe 2

Wir wollen eine (k-Band-) Turingmaschine P konstruieren, die die Sprache $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ akzeptiert. Dazu muss die Zahl n auf irgendeine Art gespeichert werden. Es ist naheliegend, dazu ein einziges Arbeitsband zu verwenden, und n darauf unär zu speichern. Das Vorgehen der Turingmaschine soll also in etwa wie folgt aussehen:

1. Zähle die Anzahl der a's in der Eingabe und schreibe für jedes gelesene a ein X auf das Arbeitsband,
2. zähle die Anzahl der b's und laufe dabei rückwärts über das Arbeitsband, um sicherzustellen dass es genausoviele a's gab,
3. zähle die Anzahl der c's und laufe dabei entsprechend vorwärts über das Arbeitsband.

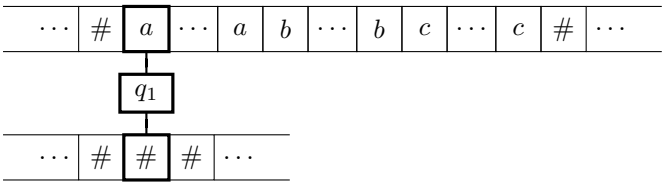
Der Fall $n = 0$ muss dabei gesondert betrachtet werden. Definiere also

- $\Sigma = \{a, b, c\}$,
- $\Gamma = \{\#, X\}$, wobei # das Leerzeichen sei,
- $Q = \{s, q_1, q_2, q_3, f\}$, wobei s der Startzustand und f der einzige Endzustand sei.

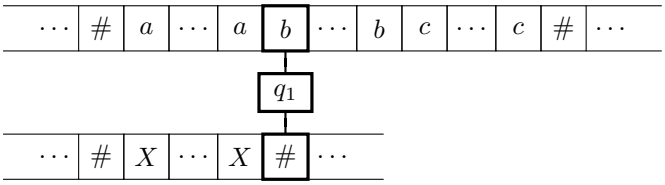
q_1, q_2 und q_3 sollen dabei dem 1., 2. und 3. Schritt obiger Beschreibung entsprechen. Insbesondere werden die Übergangsregeln so gewählt, dass die Reihenfolge (erst a's, dann b's, dann c's) eingehalten werden muss.

Definiere die Übergangsregeln:

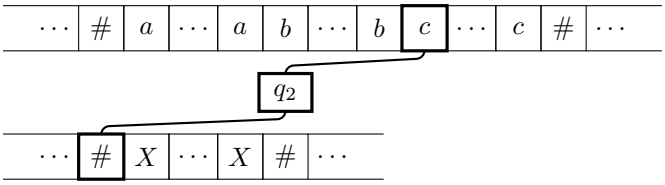
- Für den Randfall $n = 0$ führen wir die Regel $(s, \#, \#) \to (f, B, (\#, B))$ ein. Dadurch wird bei leerer Eingabe sofort akzeptiert.
- Ansonsten gehe in den Zustand q_1 über, ohne etwas zu verändern: $(s, a, \#) \to (q_1, B, (\#, B))$
  Danach sieht die Konfiguration wie folgt aus:



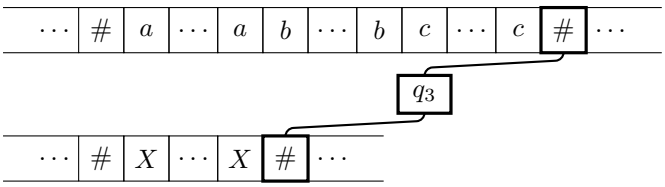
- Für jedes gelesene a schreibe ein X aufs Arbeitsband und rücke um eins vor: $(q_1, a, \#) \to (q_1, R, (X, R))$
  Schließlich erreicht P folgende Konfiguration:



- Rücke nur den Arbeitsbandkopf um eins zurück und gehe in Zustand q_2 über: $(q_1, b, \#) \to (q_2, B, (\#, L))$
- Für jedes gelesene b lies ein X vom Arbeitsband und rücke auf dem Arbeitsband nach links:
  $(q_2, b, X) \to (q_2, R, (X, L))$
  Schließlich erreicht P folgende Konfiguration:



- Rücke nur den Arbeitsbandkopf um eins vor und gehe in Zustand q_3 über: $(q_2, c, \#) \to (q_3, B, (\#, R))$
- Für jedes gelesene C lies wieder ein X vom Arbeitsband und rücke auf dem Arbeitsband nach rechts:
  $(q_3, c, X) \to (q_3, R, (X, R))$
  Schließlich erreicht P folgende Konfiguration:



- Akzeptiere die Eingabe: $(q_3, \#, \#) \to (f, B, (\#, B))$

Aufgabe 3

(1) Das Programm tauscht als erstes die Werte der beiden Register, so das in x_2 a steht und in x_1 0.

(2) Dann wird geschaut, ob $x_2 = 0$ ist, wenn ja gehe zu HALTJA ansonsten zähle x_2 5 mal runter. Sollte hierbei x_2 vor dem runterzählen 0 werden gehe zu HALTNEIN. Nun wiederhole (2), bis das Programm HALTJA oder HALTNEIN erreicht.

Start:	if $x_1 =? 0$ then (tunix) goto MOD1 else $x_1 = x_1 \dot{-} 1$ goto L1
L1:	$x_2 = x_2 + 1$ goto Start
MOD1:	if $x_2 =? 0$ then (tunix) goto HALTJA else $x_2 = x_2 \dot{-} 1$ goto MOD2
MOD2:	$x_1 = x_1 + 1$
MOD3:	if $x_2 =? 0$ then (tunix) goto HALTNEIN else $x_2 = x_2 \dot{-} 1$ goto MOD4
MOD4:	$x_1 = x_1 + 1$
MOD5:	if $x_2 =? 0$ then (tunix) goto HALTNEIN else $x_2 = x_2 \dot{-} 1$ goto MOD6
MOD6:	$x_1 = x_1 + 1$
MOD7:	if $x_2 =? 0$ then (tunix) goto HALTNEIN else $x_2 = x_2 \dot{-} 1$ goto MOD8
MOD8:	$x_1 = x_1 + 1$
MOD9:	if $x_2 =? 0$ then (tunix) goto HALTNEIN else $x_2 = x_2 \dot{-} 1$ goto MOD10
MOD10:	$x_1 = x_1 + 1$ goto MOD1

Da als erstes die Werte aus x_1 und x_2 getauscht werden, und dann immer zusammen die Werte aus x_2 dekrementiert und x_1 inkrementiert werden, bis $x_2 = 0$ ist, stehen am ende wieder die ursprünglichen Werte in den Registern.
Da der Loop (MOD1-MOD10) x_2 genau 5-mal dekrementiert. Ist der Wert genau dann danach durch 5 teilbar, wenn er es davor auch war. Da am Anfang des Loops geschaut wird ob der $x_2 = 0$, geht das Programm genau dann zu HALTJA, wenn a durch 5 teilbar ist.