Uitwerking tentamen Fundamentele Informatica 1 (I&E) — Dinsdag 23 december 2014

11:48

1(a) De eerste zes elementen in de canonieke volgorde van

$L_a(L_a^* \cup L_b) = \{\Lambda, aa\}\cdot(\{\Lambda, aa, aaaa, aaaaaa, \ldots\} \cup \{b\}) =$
$\{\Lambda, aa\}\cdot(\{\Lambda, b, aa, aaaa, aaaaaa, \ldots\})$

zijn Λ, b, aa, aab, $aaaa$, $aaaaaa$

11:52

(b)

$L_1(L_1^* \cup L_2) = L_1^+ \cup L_1L_2$

Het verschil met $L_1^* \cup L_1L_2$ zit hem in $L_1^0 = \{\Lambda\}$. Die zit automatisch in $L_1^* \cup L_1L_2$ en niet automatisch in $L_1^+ \cup L_1L_2$.

Kies dus een taal L_1 waar Λ niet inzit: $L_1 = \{a\}$

Neem $L_2 = \{b\}$.

Dan zit $x = \Lambda$ wel in $L_1^* \cup L_1L_2 = \{a\}^* \cup \{a\}\{b\} = \{a\}^* \cup \{ab\}$

maar niet in $L_1(L_1^* \cup L_2) = \{a\}(\{a\}^* \cup \{b\})$.

Er geldt dus niet algemeen dat $L_1^* \cup L_1L_2 = L_1(L_1^* \cup L_2)$.

11:58

2) Stel dat L_0 wél door een eindige automaat geaccepteerd kan worden.
Laat dan n het aantal toestanden van deze eindige automaat zijn.

We kiezen het woord $x = a^{n+2}b^{n+1}$

Inderdaad is x te schrijven als $(ab)^i a^j b^k$ met $i,k \geq 0$, $j > i+k$,
namelijk met $i=0$, $j = n+2$, $k = n+1$.

Neem nu een willekeurige opsplitsing van x als $x = uvw$, met $|uv| \leq n$ en $|v| > 0$.

Omdat $|uv| \leq n$, bestaan u en v allebei alleen maar uit a's.

Bovendien begint w met minstens twee a's.

Omdat $|v| > 0$, bevat v minstens een a

Dit betekent dat het woord uv^0w ("we pompen v weg")
van de vorm $a^\ell b^{n+1}$ is, met

$\ell \geq 2$ (want de a's in w zijn sowieso behouden)

en $\ell \leq n+1$ (want we zijn minstens één a kwijtgeraakt, door v weg te pompen).

De enige manier om $uv^0w = a^\ell b^{n+1}$ dan te schrijven als $(ab)^i a^j b^k$,
is met $i = 0$ (we beginnen immers met minstens twee a's),
$j = \ell \leq n+1$
$k = n+1$

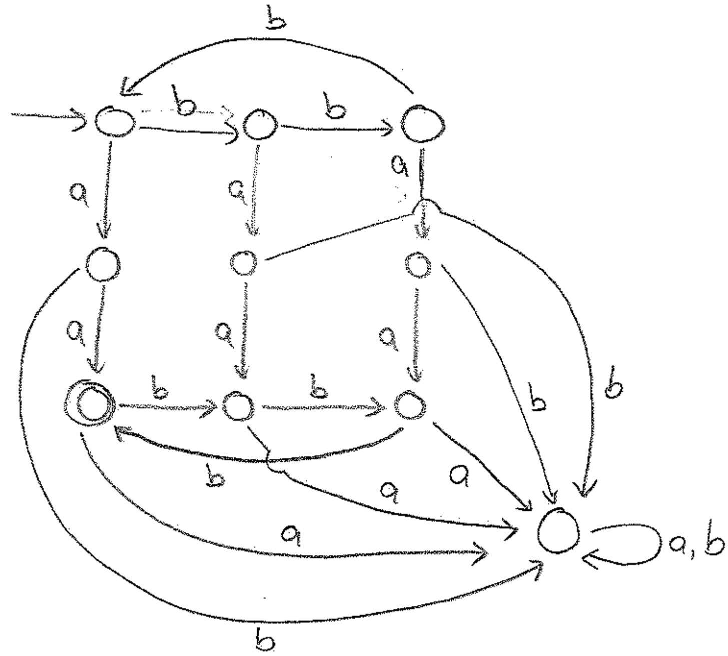
Maar voor deze combinatie van i, j en k geldt niet dat $j > i+k$

Dus $uv^0w = a^\ell b^{n+1} \notin L_0$.

Dit is in strijd met het pomplemma, en dus kan L_0 niet door een eindige automaat geaccepteerd worden

12:15
12:28

3 a)



De linker drie toestanden corresponderen met:
"het aantal b's tot nu toe is 0 mod 3."
De middelste drie met "1 mod 3"
De rechter drie met "2 mod 3"
Rechtsonder hebben we het afvoerputje

12:27

b) Een woord $x \in L$ bevat de substring aa, met daarvoor en daarachter een aantal ≥ 0 b's, zodat het totaal aantal b's een veelvoud van 3 is

$\Rightarrow$ of 0 mod 3 b's ervoor en 0 mod 3 b's erachter
of 1 mod 3 b's ervoor en 2 mod 3 b's erachter
of 2 mod 3 b's ervoor en 1 mod 3 b's erachter.

Dit levert de volgende reguliere expressie op:

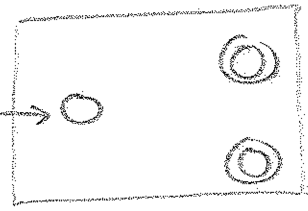
$$\underbrace{(bbb)^*}_{0 \bmod 3} aa \underbrace{(bbb)^*}_{0 \bmod 3} + \underbrace{b(bbb)^*}_{1 \bmod 3} aa \underbrace{bb(bbb)^*}_{2 \bmod 3} + \underbrace{bb(bbb)^*}_{2 \bmod 3} aa \underbrace{b(bbb)^*}_{1 \bmod 3.}$$

12:33

Uitwerking tentamen Fundamentele Informatica 1 (I&E)
Dinsdag 23 december 2014

4 a) We moeten L^* accepteren.
Dat betekent dat we 0 of meer keer M moeten doorlopen.
Laat M er als volgt uit zien

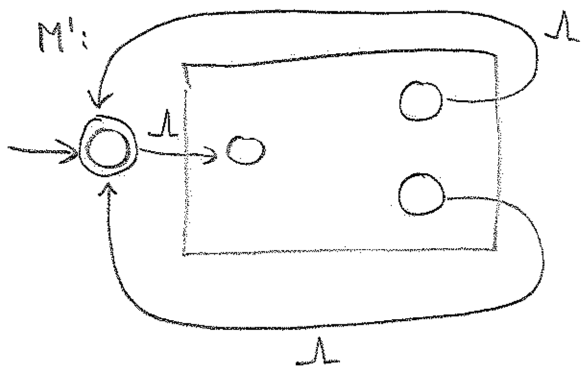
M



Met een begintoestand en 0 of meer accepterende toestanden

Dan construeren we M' als volgt

M':

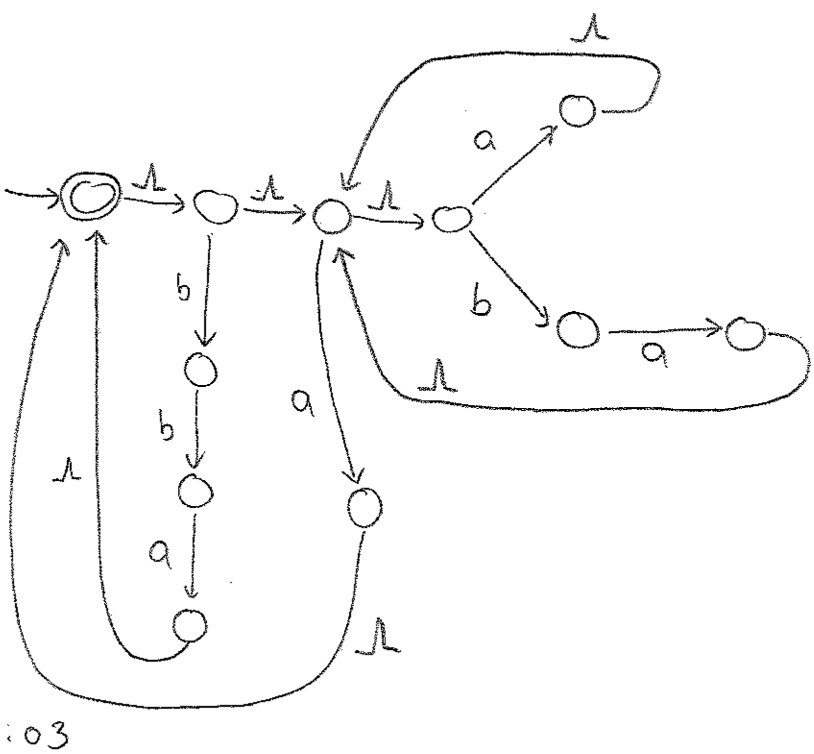


Met
* een nieuwe toestand, die begintoestand van M' is, en tevens de enige accepterende toestand is
* een Λ-transitie van de nieuwe naar de oude begintoestand
* Λ-transities van de oude accepterende toestanden naar de nieuwe begintoestand.

12:40

b) Voor de ster operatie in de reguliere expressie gebruiken we de constructie van onderdeel (a). Voor de rest gaan we ad hoc te werk.

12:47
12:59



13:03

Uitwerking tentamen Fundamentele Informatica 1 (I&E)
Dinsdag 23 december 2014

4

5 (a)

Er moet gelden

$i \leq j \Rightarrow$ elke voorkomen van ab links moet samengaan met een voorkomen van a in het midden,

$j < i+k \Rightarrow$ elke extra a in het midden (ten opzichte van de ab's links) moet samengaan met een b rechts, en dan nog minstens een b extra

We nemen

$S_1 \rightarrow XYZ$

$X \rightarrow abXa \mid \Lambda$ Hiermee zorgen we dat $0 \leq i \leq j$ $\Rightarrow$ elke ab links gaat samen met a in het midden

$Y \rightarrow aYb \mid \Lambda$ Hiermee zorgen we dat elke extra a in het midden samen gaat met een b rechts

$Z \rightarrow bZ \mid b$ Hiermee zorgen we dat we nog minstens een b rechts extra hebben.

13:13

(b)

$S_1 \Rightarrow XYZ \Rightarrow abXaYZ \Rightarrow abaYZ \Rightarrow abaaYbZ \Rightarrow abaabZ \Rightarrow abaabb$

13:15

(c)

De eerste zevenelementen in de canonieke volgorde van L_2 zijn

	i	j	k	
b	0	0	1	
ab	1	0	0	
bb	0	0	2	
abb	1	0	1	b.v.
bbb	0	0	3	
abab	2	0	0	b.v.
abbb	1	0	2	b.v.

En vervolgens

bbbb	0	0	4	
aabbb	0	2	3	
ababa	2	1	0	
ababb	2	0	1	b.v.

13:21

(d) Als $j < i+k$, zijn er twee mogelijkheden:

* $i \leq j < i+k$
* of $i > j$, en dan is automatisch $j < i+k$

13:24

± 14:45

Er geldt dus:

$L_2 = \{(ab)^i a^j b^k \mid i,j,k \geq 0$ en $i \leq j < i+k\} \cup$

$\{(ab)^i a^j b^k \mid i,j,k \geq 0$ en $i > j\} =$

$L_1 \cup \{(ab)^i a^j b^k \mid i,j,k \geq 0$ en $i > j\}$

Voor L_1 kunnen we de variabelen en producties uit onderdeel (a) gebruiken.

Dan moeten we alleen nog variabelen en producties voor

$\{(ab)^i a^j b^k \mid i,j,k \geq 0$ en $i > j\}$

verzinnen, maar dat is niet zo moeilijk:

$S_2 \rightarrow S_1 \mid TUW$

$S_1 \rightarrow XYZ$

$X \rightarrow abXa \mid \Lambda$

$Y \rightarrow aYb \mid \Lambda$

$Z \rightarrow bZ \mid b$

$T \rightarrow abT \mid ab$

$U \rightarrow abUa \mid \Lambda$

$W \rightarrow bW \mid \Lambda$

14:54

14:57

6) Een string x die door M geaccepteerd wordt:

* begint met een a (om van q_0 in q_1 te komen)
* bevat vervolgens een substring y met evenveel a's als b's, waarbij de a's 'nooit voorlopen' op de b's
  (voor elke prefix z van y geldt $n_b(z) \geq n_a(z)$, en dus $n_a(y) = n_b(y) \geq 0$)
  Dat 'niet voorlopen' wordt afgedwongen doordat we alleen b's op de stapel kunnen zetten in het lusje op q_1, en dat doen we als we b's lezen.
* bevat vervolgens een substring u met $n_a(u) = n_b(u) \geq 1$, die begint met een a (om van q_1 in q_2 te komen), en waarin de a's steeds voorlopen op de b's, tot het eind van u, waar ze gelijk worden.
  (Voor elke prefix v van u met $v \neq \Lambda$ en $v \neq u$ geldt $n_a(v) > n_b(v)$)
  Dat voorlopen wordt afgedwongen doordat we alleen in het lusje op q_2 kunnen blijven als er nog een a op de stapel staat.

En die a's komen op de stapel als we een a lezen.

* bevat na u niets meer: zodra $n_a(u) = n_b(u) \geq 1$ geworden, zien we Z_0 op de stapel en gaan we met een Λ-transitie van q_2 naar de accepterende toestand q_3.

15:13

7 (a)

Dat betekent dat als we in toestand p zitten, een invoerletter σ lezen en het symbool X bovenop de stapel zien staan, we twee mogelijkheden hebben:

* naar toestand q_1 gaan en het symbool X op de stapel vervangen door de string van stapelsymbolen α_1
* naar toestand q_2 gaan en het symbool X op de stapel vervangen door de string van stapelsymbolen α_2

15:18

(b)

Er moet gelden:

(1) voor elke toestand $p \in Q$, elk stapelsymbool $X \in \Gamma$ en elke $\sigma \in \Sigma \cup \{\Lambda\}$ (invoerletter of lege string), bevat $\delta(p, \sigma, X)$ hoogstens één element.

(2) voor elke toestand $p \in Q$, elk stapelsymbool $X \in \Gamma$ en elke invoerletter $\sigma \in \Sigma$ geldt: als $\delta(p, \sigma, X)$ niet leeg is, is $\delta(p, \Lambda, X)$ wel leeg

15:23

Ofwel

(1) voor elke toestand $p \in Q$, elk stapelsymbool $X \in \Gamma$ en elke $\sigma \in \Sigma \cup \{\Lambda\}$ (invoerletter of lege string) is er hoogstens één transitie

(2) voor elke toestand $p \in Q$, elk stapelsymbool $X \in \Gamma$ en elke invoerletter $\sigma \in \Sigma$ geldt: als er een transitie is vanuit p met invoer σ en X op de stapel, dan is er geen Λ-transitie vanuit p met X op de stapel.

15:26

Uitwerking tentamen Fundamentele Informatica 1 (I&E), Dinsdag 23 december 2014

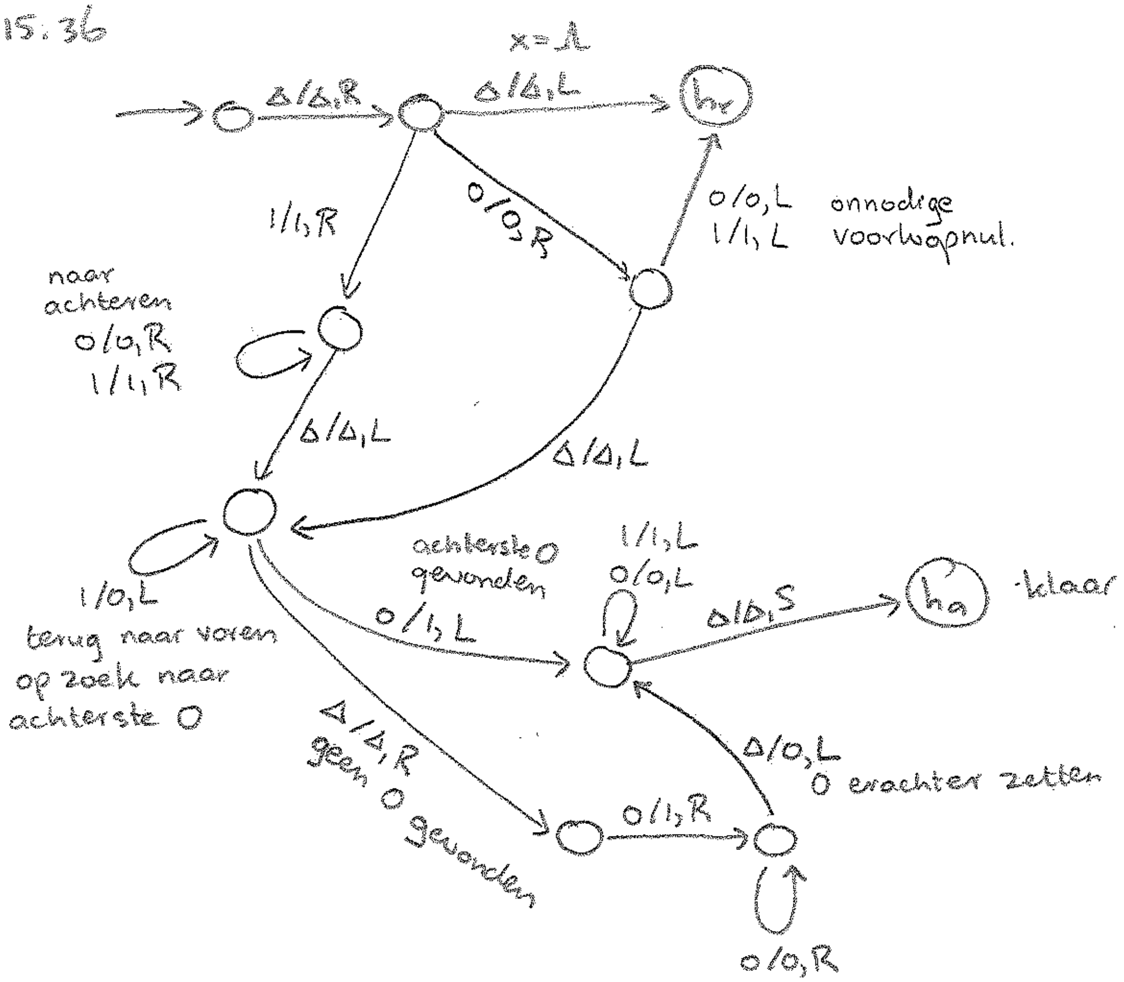
8) T controleert eerst of x wel de binaire representatie van een natuurlijk getal is. Dat wil zeggen: T controleert
- of $x \neq \Lambda$, en
- of x geen onnodige voorloopnullen bevat.

Als x niet 'goed' is, gaat T direct naar h_r (reject).

Anders (x is 'goed') loopt T naar het achtereind van x, en loopt dan terug naar voren op zoek naar de achterste 0 in x. Dat is de eerste 0 die T hierbij tegenkomt. Onderweg naar deze 0 worden de 1'en die T passeert vast vervangen door 0. De gevonden 0 zelf wordt een 1, en T loopt terug naar voren.

Als T geen 0 vindt, heeft T wel elke 1 al vervangen door een 0. De voorste 0 wordt nu teruggezet op 1, en achteraan wordt een 0 toegevoegd. De functiewaarde $f(x) = 10\ldots0$ moet namelijk beginnen op vakje 1 op de tape (en niet op vakje 0).

± 15:36



15:46