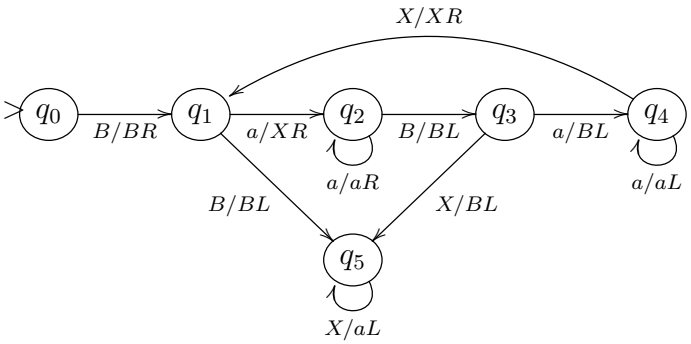
Berekenbaarheid 2017
Uitwerkingen Tentamen
6 november 2017

1. Definieer een standaard Turing-machine M_1 met input alfabet $\Sigma = \{a\}$ die bij input a^n als output $a^{\lfloor n/2 \rfloor}$ heeft. Hierin betekent $\lfloor \ldots \rfloor$ naar beneden afronden. Er moet bijv. gelden dat $M_1(aaaaa) = aa$. Zorg ervoor dat bij terminatie de kop van de machine weer aan het begin van de tape staat.



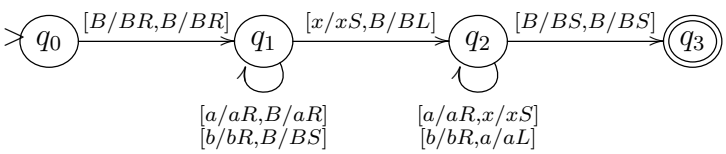
2. Definieer een non-deterministische twee-tape Turing-machine M_2 die de taal

$$L_2 := \{u_1 u_2 \mid u_1, u_2 \in \{a, b\}^* \text{ en } |u_1|_a = |u_2|_b\}$$

herkent door eindtoestand. Hierin is $|u|_x$ het aantal maal dat het symbool x voorkomt in het woord u. Er geldt bijvoorbeeld dat $aabaaabab \in L_2$, want $|aab|_a = 2$ en $|aaabab|_b = 2$.

Zorg ervoor dat een woord $w \in L_2$ altijd wordt herkend in ten hoogste $|w| + 3$ stappen.

M_2:



$x \in \Gamma$

In feite is $L_2 = \Sigma^*$, dus iedere machine die ieder woord accepteert is goed. En het aantal benodigde stappen om $|w|$ te controleren is dus niet $|w| + 3$ maar 0 stappen, in onderstaande machine M_2' die dus ook een correcte uitwerking van deze opgave is.

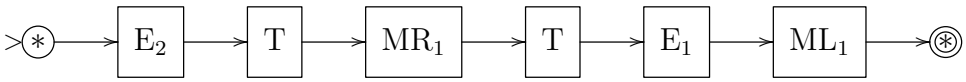
M_2':

Om in te zien dat ieder woord w in deze taal zit, kijk naar het verschil $n = |u_1|_a - |u_2|_b$ voor iedere mogelijke splitsing van w. Als de splitsing helemaal vooraan is ($u_1 = \lambda$, $u_2 = w$), dan is $n \leq 0$, terwijl als de splitsing helemaal achteraan is ($u_1 = w$, $u_2 = \lambda$) dan $n \geq 0$. Bij het één symbool opschuiven van het punt waar gesplitst wordt, verandert het verschil altijd met precies één. Hieruit volgt dat het verschil n ergens tussenin nul moet worden.

Dat deze opgave in zekere zin flauw is was geen vergissing bij het maken van dit tentamen, maar een grap. In het hertentamen zit een variant van deze opgave waarbij de taal niet zo triviaal is.

3. Implementeer de macro $P_3^{(4)}$, uitsluitend gebruik makend van de macro's ML_n, MR_n, E_n en T. Je mag in je machine dus geen andere macro's, en ook geen expliciete toestanden en transities gebruiken.

   Zie pagina 7 voor de betekenis van deze macro's.



4. Waarom is het in het licht van de Church-Turing-hypothese niet verrassend dat Turing-machines met meer dan één tape precies de recursief opsombare talen herkennen?

   Als dit soort machines een andere klasse talen zouden kunnen herkennen, dan zou het berekenbaarheidsmodel van deze machines verschillen van, en krachtiger zijn dan, dat van standaard Turing-machines. En de Church-Turing-hypothese zegt nu juist dat ieder redelijk berekenbaarheidsmodel equivalent is aan dat van standaard Turing-machines.

5. Voor iedere Turing machine M is het probleem H_M (het halting probleem voor de machine M) gedefinieerd als:

   input: w
   vraag: stopt M met input w?

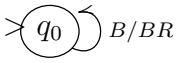
   Geef een code $R(M_5)$ van een Turing-machine M_5 waarvoor het probleem H_{M_5} niet-triviaal en beslisbaar is. Verklaar je antwoord.

   Zie bladzijde 8 voor een relevant citaat uit het boek van Sudkamp.

   Dit probleem vraagt dus of $w \in L(M)$.

   We zoeken dus een M_5 waarvoor dit beslisbaar is, wat erop neerkomt dat $L(M_5)$ recursief moet zijn. En we moeten hebben dat $L(M_5) \neq \emptyset$ en $L(M_5) \neq \Sigma^*$, want anders is het probleem triviaal.

Een M_5 met deze eigenschappen is:



Hiervoor geldt $L(M_5) = \Sigma^* \setminus \{\lambda\}$, en die taal is recursief, want wordt geaccepteerd door eindtoestand door de machine M_5' die voor iedere input stopt:

De gevraagde code is dus:

$$R(M_5) = 000101111010111011000$$

6. Zowel in deze opgave als in de volgende opgave is U de universele Turingmachine uit het boek.

   (a) Het probleem H_U is gedefinieerd als:

   input: w
   vraag: stopt U met input w?

   Laat zien dat H_U onbeslisbaar is.

   Dit volgt uit het feit dat het blank tape probleem B reduceert naar het probleem H_U.

   Voor deze reductie moeten we bij iedere input $R(M)$ van B een input w voor H_U construeren zodat geldt:

   $$M(\lambda)\downarrow \iff w \text{ voldoet aan } H_U \iff U(w)\downarrow$$

   Een constructie die dit doet is:

   $$w := R(M)$$

   Want dan:

   $$U(w)\downarrow \iff U(R(M))\downarrow \iff U(R(M)\lambda)\downarrow \iff M(\lambda)\downarrow$$

   Bij de laatste equivalentie gebruiken we dat U de universele Turingmachine is.

   (b) We definiëren het probleem P_6 als:

   input: $R(M)$
   vraag: stopt U met input $R(M)$?

   Is P_6 onbeslisbaar? Zo ja, laat zien dat dit zo is. Zo nee, verklaar waarom niet.

   Ja, P_6 is ook onbeslisbaar, en het bewijs hiervan is identiek aan het onbeslisbaarheidsbewijs voor H_U.

7. We definiëren het probleem P_7 als:

   input: $R(M)$
   vraag: stopt M met input $R(U)$?

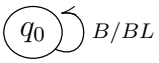
   Laat zien dat P_7 onbeslisbaar is.

   Dit probleem vraagt of $R(U) \in L(M)$. Dit is een eigenschap van de taal $L(M)$, en we kunnen dus de stelling van Rice toepassen om te bewijzen dat dit probleem onbeslisbaar is.

   Hiervoor moeten we alleen nog laten zien dat het probleem niet-triviaal is. Hiertoe moeten we inputs $R(M_{\text{ja}})$ en $R(M_{\text{nee}})$ geven waarvoor het probleem respectievelijk een positief en een negatief antwoord geeft.

   We kunnen hiervoor bijvoorbeeld nemen

   M_{ja} : 

   M_{nee} : 

   want M_{ja} stopt met iedere input dus ook met input $R(U)$, en M_{nee} stopt voor geen enkele input dus ook niet voor input $R(U)$.

8. We definiëren de partiële numerieke functie k als:

$$k(n) := \begin{cases} \uparrow & \text{als } n = 0 \\ 0 & \text{als } n > 0 \end{cases}$$

   Geef functies f_1 en f_2 zodat:

$$f_1 \circ f_2 = e$$
$$f_2 \circ f_1 = k$$

   Hierin is e de lege functie die voor iedere input ongedefinieerd is.

   Verklaar je antwoord.

   Een mogelijke oplossing is:

$$f_1 = k$$
$$f_2 = z = c_0^{(1)}$$

9. We definiëren de numerieke functie f_9 door:

$$f_9 := \textsf{primrec}(c_{2017}^{(0)}, \textsf{exp})$$

   Bereken $f_9(4)$ en verklaar hoe je aan dat antwoord bent gekomen.

De recursievergelijkingen voor f_9 zijn:

$$f_9(0) = 2017$$
$$f_9(y+1) = y^{f(y)}$$

Hiermee krijgen we:

$$\begin{aligned}
f_9(0) &= 2017 \\
f_9(1) &= f_9(0+1) = 0^{2017} = 0 \\
f_9(2) &= f_9(1+1) = 1^{f_9(1)} = 1^0 = 1 \\
f_9(3) &= f_9(2+1) = 2^{f_9(2)} = 2^1 = 2 \\
f_9(4) &= f_9(3+1) = 3^{f_9(3)} = 3^2 = 9
\end{aligned}$$

Het antwoord is dus:

$$f_9(4) = 9$$

10. We coderen eindige rijtjes natuurlijke getallen als een natuurlijk getal op de volgende manier. Het rijtje

$$\langle a_0, \ldots, a_n \rangle$$

wordt gecodeerd door het getal

$$2^{a_0+1} \cdot 3^{a_1+1} \cdot 5^{a_2+1} \cdot \ldots \cdot p_n^{a_n+1}$$

Hierin is p_i het i-de priemgetal, tellend vanaf nul. De code van het rijtje $\langle 2, 1 \rangle$ is bijvoorbeeld $2^{2+1} \cdot 3^{1+1} = 8 \cdot 9 = 72$. De code van het lege rijtje $\langle\rangle$ is 1.

De functie f_{10} geeft aan of een getal een code van een rijtje is. Deze is dus gedefinieerd als:

$$f_{10}(x) = \begin{cases} 1 & \text{als } x \text{ de code van een rijtje is} \\ 0 & \text{anders} \end{cases}$$

We hebben bijvoorbeeld $f_{10}(72) = 1$, want 72 is de code van $\langle 2, 1 \rangle$, maar $f_{10}(75) = 0$, want $75 = 2^0 \cdot 3^1 \cdot 5^2$ is niet de code van een rijtje.

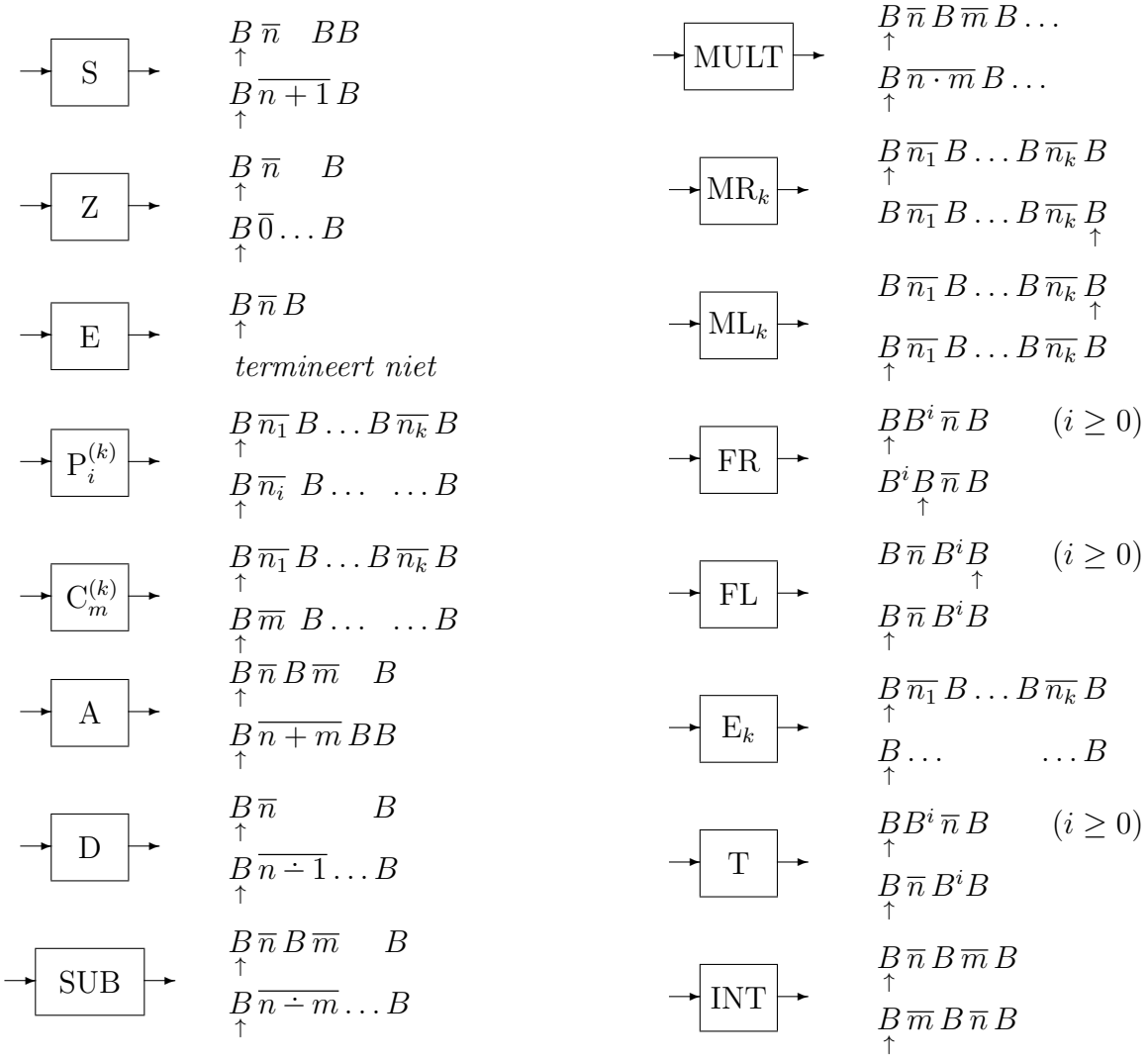
Laat zien dat f_{10} primitief recursief is. Je mag gebruiken dat alle functies in het lijstje op bladzijde 8 primitief recursief zijn.

Dit volgt uit de schrijfwijze:

$$f_{10}(x) = \mathsf{sg}(x) \cdot \mathsf{cosg}\Big(\sum_{p_1=2}^{x} \sum_{p_2=p_1+1}^{x} \mathsf{cosg}(\mathsf{divides}(x, p_1)) \cdot \mathsf{divides}(x, p_2) \Big)$$

Om een code van een rijtje te zijn moet namelijk gelden dat als een priemgetal de code deelt, alle kleinere priemgetallen die code ook delen. Of in andere woorden, het mag niet voorkomen dat er priemgetallen $p_1 < p_2$ bestaan zodat p_2 de code deelt, maar p_1 niet.

Macro's voor Turing-machines voor numerieke berekeningen



Codering van transities

Symbol	Encoding
0	1
1	11
B	111
q_0	1
q_1	11
$\vdots$	$\vdots$
q_n	1^{n+1}
L	1
R	11

Let $en(x)$ denote the encoding of a symbol x. A transition $\delta(q_i, x) = [q_j, y, d]$ is encoded by the string

$$en(q_i)0en(x)0en(q_j)0en(y)0en(d).$$

Primitief recursieve functies

$$\begin{aligned} \mathsf{id}(x) &= x \\ z(x) &= 0 \\ s(x) &= x+1 \\ p_i^{(k)}(x_1, \ldots, x_k) &= x_i \\ c_n^{(k)}(x_1, \ldots, x_k) &= n \end{aligned}$$

$$\begin{aligned} \mathsf{pred}(y) &= y \dot{-} 1 \\ \mathsf{add}(x,y) &= x + y \\ \mathsf{mult}(x,y) &= x \cdot y \\ \mathsf{sub}(x,y) &= x \dot{-} y \\ \mathsf{exp}(x,y) &= x^y \\ \mathsf{fact}(x) &= x! \\ \mathsf{sg}(x) &= \text{als } x \neq 0 \text{ dan 1 anders 0} \\ \mathsf{cosg}(x) &= \text{als } x \neq 0 \text{ dan 0 anders 1} \\ \mathsf{lt}(x,y) &= \text{als } x < y \text{ dan 1 anders 0} \\ \mathsf{gt}(x,y) &= \text{als } x > y \text{ dan 1 anders 0} \\ \mathsf{le}(x,y) &= \text{als } x \leq y \text{ dan 1 anders 0} \\ \mathsf{ge}(x,y) &= \text{als } x \geq y \text{ dan 1 anders 0} \end{aligned}$$

$$\begin{aligned} \mathsf{eq}(x,y) &= \text{als } x = y \text{ dan 1 anders 0} \\ \mathsf{ne}(x,y) &= \text{als } x \neq y \text{ dan 1 anders 0} \\ \mathsf{max}(x,y) &= \text{het maximum van } x \text{ en } y \\ \mathsf{min}(x,y) &= \text{het minimum van } x \text{ en } y \\ \mathsf{quo}(x,y) &= \text{als } y \neq 0 \text{ dan } \lfloor x/y \rfloor \text{ anders 0} \\ \mathsf{rem}(x,y) &= \text{als } y \neq 0 \text{ dan } x \bmod y \text{ anders } x \\ \mathsf{divides}(x,y) &= \text{als } y \neq 0 \text{ en } y \mid x \text{ dan 1 anders 0} \\ \mathsf{even}(x) &= \text{als } x \text{ even is dan 1 anders 0} \\ \mathsf{prime}(x) &= \text{als } x \text{ priem is dan 1 anders 0} \\ \mathsf{pn}(x) &= \text{het } x\text{-de priemgetal} \\ & \quad (\text{dus } \mathsf{pn}(0) = 2,\ \mathsf{pn}(1) = 3, \text{ etc.}) \end{aligned}$$