HERTENTAMEN FUNDAMENTELE INFORMATICA 3
Dinsdag 8 juli 2014, 14.00 - 17.00 uur

Dit tentamen bestaat uit 6 opgaven, waarbij steeds tussen [en] staat hoeveel punten er ongeveer mee te verdienen zijn. In totaal zijn er 103 punten te verdienen. Geef de gevraagde Turing machines door middel van hun transitiediagram.
Wanneer er bij een vraag om uitleg of toelichting gevraagd wordt, is het belangrijk om die ook te geven.
Als je het antwoord op een onderdeel niet weet, en je hebt dat antwoord nodig bij een later onderdeel, dan kun je het antwoord 'kopen' bij de docent.

---

1. [23 pt]

   (a) Stel dat f een partiële functie is op Σ^*, met waarden in Γ^*, voor zekere alfabetten Σ en Γ.

   Wat betekent het als we zeggen dat een Turing machine (TM) T, met invoeralfabet Σ, tape alfabet Γ en begintoestand q_0, de functie f berekent?

   Deze opgave gaat verder over binaire representaties van niet-negatieve gehele getallen. Zo'n binaire representatie mag (in deze opgave) **niet** beginnen met een nul. Dat betekent dat voorloopnullen niet zijn toegestaan, en dat de binaire representatie van het getal 0 gelijk is aan de lege string.

   (b) Construeer een TM T_1 die de volgende partiële functie f_1 op $\{0,1\}^*$ berekent:

   $$f_1(x) = \begin{cases} x-1 & \text{als } x \text{ de binaire representatie} \\ & \text{van een positief geheel getal is} \\ \text{niet gedefinieerd} & \text{anders} \end{cases}$$

   Ook de functiewaarde moet weer binair gerepresenteerd zijn. Bijvoorbeeld: $f_1(1010) = 1001$.

   Je mag voor T_1 alleen componenten gebruiken, als je die componenten zelf ook uitwerkt (tekent).

   Leg duidelijk uit hoe T_1 werkt.

   (c) Construeer een TM T_2 die een getal in binaire representatie omzet in unaire repesentatie. Om precies te zijn: T_2 berekent de volgende partiële functie f_2 op $\{0,1\}^*$:

   $$f_2(x) = \begin{cases} 1^x & \text{als } x \text{ de binaire representatie} \\ & \text{van een niet-negatief geheel getal is} \\ \text{niet gedefinieerd} & \text{anders} \end{cases}$$

   Bijvoorbeeld: $f_2(1010) = 1111111111$.

   Je mag voor TM T_2 de TM T_1 als component gebruiken. Daarnaast mag T_2 gebruik maken van de componenten *NB*, *PB*, *Insert*(σ) en *Delete* zoals die in het boek beschreven zijn. Andere componenten mag je alleen gebruiken als je ze zelf uitwerkt. Wellicht ten overvloede:

   - *NB* verplaatst de leeskop naar de eerste Δ rechts van de huidige positie,

- *PB* verplaatst de leeskop (zo mogelijk) naar de eerste Δ links van de huidige positie,
- *Insert*(σ) verandert de tape-inhoud van $y\underline{z}$ in $y\underline{\sigma}z$ (waarbij z geen Δ bevat),
- *Delete* verandert de tape-inhoud van $y\underline{\sigma}z$ in $y\underline{z}$ (waarbij z geen Δ bevat).

Leg ook duidelijk uit hoe T_2 werkt.

---

2. [19 pt] In Stelling 7.26 van het boek wordt bewezen dat voor iedere 2-tapes TM T_1 een gewone 1-tapes TM T_2 bestaat, die dezelfde taal accepteert en (indien van toepassing) dezelfde functie berekent.

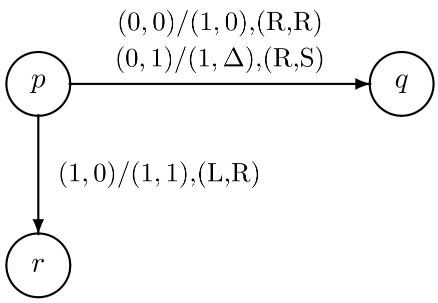
   De TM T_2 die in het bewijs van de stelling geconstrueerd wordt, simuleert T_1. Daartoe moet uiteraard de inhoud van de twee tapes van T_1 gecombineerd worden op de ene tape van T_2.

   (a) Stel dat de invoer van T_1 (en dus ook van T_2) de string $x = a_1a_2 \dots a_n$ is, voor zekere $n \geq 1$. Dan wordt de tape-inhoud van T_2 eerst omgezet van $\Delta a_1a_2 \dots a_n$ in $\$\Delta'\Delta' a_1\Delta a_2\Delta \dots \Delta a_n\#$

   Leg duidelijk uit waarom deze omzetting plaatsvindt. Besteed daarbij aandacht aan

   - de \$ en # (wat is de functie van deze symbolen, en wanneer komen we ze tegen),
   - de extra Δ's tussen de letters
   - en de accenten op enkele symbolen.

   (b) Stel dat een deel van T_1 er als volgt uitziet:



   Stel verder dat we in toestand p zijn, en dat de tape van T_2 er als volgt uitziet:

$$\$\Delta 01'\Delta 10'\Delta\#$$

   Hoe ziet de tape van T_2 eruit na simulatie van de volgende stap in T_1? Motiveer je antwoord.

---

3. [15 pt] Laat $G = (V, \Sigma, S, P)$ de context-gevoelige grammatica zijn met $V = \{S, A, B, C\}$, $\Sigma = \{a, b, c\}$ en de volgende producties:
   $S \to SABC \mid ABC$
   $BA \to AB \qquad CA \to AC \qquad CB \to BC$
   $A \to a \qquad B \to b \qquad C \to c$

   (a) Wat zijn de eerste vier elementen van $L(G)$ in de canonieke volgorde?

   (b) Wat is $L(G)$? Geef een precies antwoord in woorden of formules. Motiveer je antwoord, door uit te leggen wat de functie is van de diverse variabelen en producties in G.

---

4. [6 pt] Laat e de coderingsfunctie uit het boek zijn, die TMs afbeeldt op bitstrings. Toon met behulp van een diagonaalargument aan dat de verzameling

   $$NSA = \{e(T) \mid T \text{ is een TM, en } e(T) \notin L(T)\}$$

   niet recursief opsombaar is.

---

5. [19 pt] Post's Correspondence Problem (PCP) luidt als volgt:

   Gegeven een verzameling paren

   $$\{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)\},$$

   waarbij de α_i's en de β_i's niet-lege strings over een alfabet Σ zijn.
   Bestaat er een rijtje indices $i_1, i_2, \ldots, i_k$ (met $k \geq 1$) zó dat

   $$\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_k} = \beta_{i_1}\beta_{i_2}\ldots\beta_{i_k}$$

   De verzameling paren (α_i, β_i) wordt een correspondentiesysteem genoemd, en kan grafisch worden weergegeven als een verzameling 'dominostenen'.

   (a) Beschouw het volgende correspondentiesysteem I:

0110	110	0	1
0	00	011	11

   Is I een ja-instantie van PCP? Zo ja, geef een match voor het systeem. Zo nee, beargumenteer waarom niet.

   (b) Beschouw nu het volgende beslissingsprobleem:

   CFGNonEmptyIntersection: Gegeven twee context-vrije grammatica's G_1 en G_2, is $L(G_1) \cap L(G_2)$ niet-leeg?

   Gegeven is dat PCP niet beslisbaar is.

   Toon aan dat ook *CFGNonEmptyIntersection* niet beslisbaar is, met behulp van een reductie tussen PCP en *CFGNonEmptyIntersection*.

   Laat hierbij uiteraard zien dat aan alle eisen van een reductie voldaan is, en vergeet niet om de conclusie te trekken.

   Als je geen geschikte reductie tussen CFGNonEmptyIntersection *en* PCP *weet, kun je een deel van de punten daarvan verdienen door uit te leggen hoe je voor algemene beslissingsproblemen* P_1 *en* P_2 *aantoont dat* $P_1 \leq P_2$.

---

6. [21 pt]

   (a) Laat $n \geq 0$ en laat $g : \mathbb{N}^n \to \mathbb{N}$ en $h : \mathbb{N}^{n+2} \to \mathbb{N}$ twee functies zijn.
   Wat betekent het als we zeggen dat een functie f is ontstaan uit g en h met de operatie van primitieve recursie? Hoeveel argumenten heeft de functie f in dit geval?

   (b) Beschouw de volgende primitieve recursieve afleiding:
   - $f_1 = s$ (de opvolger functie),
   - $f_2 = p_3^3$,
   - f_3 wordt verkregen uit f_1 en f_2 door compositie,
   - f_4 wordt verkregen uit f_1 en f_3 door compositie,
   - f_5 wordt verkregen uit f_1 en f_4 door primitieve recursie.

   Geef eenvoudige formules voor f_3, f_4 en f_5 (waarbij je ook duidelijk kunt zien hoeveel argumenten ze hebben). Leg bij f_5 uit hoe je aan je antwoord komt.

   (c) Laat de functie $f : \mathbb{N}^2 \to N$ gedefinieerd zijn door

$$f(n,k) = \begin{cases} \binom{n}{k} = \dfrac{n!}{k! \cdot (n-k)!} & \text{als } n \geq k \\ 0 & \text{anders} \end{cases}$$

   Toon aan dat f met de operatie van primitieve recursie verkregen kan worden uit twee functies g en h, die beide primitief recursief zijn. Geef hierbij ook een beschrijving van g en h voor algemene parameters $x_1, x_2, \ldots$. Je hoeft hierbij echter niet zover te gaan dat je projecties gebruikt.

   Je mag gebruiken dat de functies *Add*, *Sub*, *Mult*, *Div* en *Mod*, zoals die in het boek gedefinieerd zijn, primitief recursief zijn,

---

`einde tentamen`