HERTENTAMEN FUNDAMENTELE INFORMATICA 3
Dinsdag 7 juli 2015, 14.00 - 17.00 uur

Dit tentamen bestaat uit vijf opgaven.
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. Deze opgave gaat over Turing machines om functies $f_i : \mathbb{N} \to \mathbb{N}$ te berekenen. We gaan hierbij uit van de unaire representatie van de natuurlijke getallen.

   (a) Construeer een Turing machine T_1 die de functie $f_1 : \mathbb{N} \to \mathbb{N}$ berekent, gedefinieerd door
   $$f_1(x) = \lceil x/2 \rceil$$
   ('x gedeeld door 2, naar boven afgerond').
   Als je bij dit onderdeel gebruik wilt maken van componenten, zul je die ook moeten uitwerken.

   (b) Construeer een Turing machine T_2 die de partiële functie $f_2 : \mathbb{N} \to \mathbb{N}$ berekent, gedefinieerd door
   $$f_2(x) = \begin{cases} \text{niet gedefinieerd} & \text{als } x = 0 \\ \lceil {}^2\log x \rceil & \text{als } x \geq 1 \end{cases}$$
   ('de 2-log van x, naar boven afgerond').
   Bij dit onderdeel mag je gebruik maken van de component T_1 uit onderdeel (a) en 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.

   (c) Leg uit hoe je T_2 aan moet passen om een Turing machine T_3 te krijgen die de partiële functie $f_3 : \mathbb{N} \to \mathbb{N}$ berekent, gedefinieerd door
   $$f_3(x) = \begin{cases} \text{niet gedefinieerd} & \text{als } x = 0 \\ \lfloor {}^2\log x \rfloor & \text{als } x \geq 1 \end{cases}$$
   ('de 2-log van x, naar beneden afgerond').

---

2. Beschouw de *unrestricted grammar* G met startsymbool S en producties

$$S \to aAS \mid bBS \mid \Lambda$$
$$\begin{array}{lll} Aa \to aA & Ab \to bA & AS \to Sa \\ Ba \to aB & Bb \to bB & BS \to Sb \end{array}$$

Een mogelijke afleiding in G is

$$S \Rightarrow^* aAaAbBS \Rightarrow^* aabAABS \Rightarrow^* aabSaab \Rightarrow aabaab$$

(a) Laat zien dat $L(G)$ NIET gelijk is aan de taal $XX = \{ss \mid s \in \{a, b\}^*\}$.

(b) Wat is $L(G)$ wél? Geef een duidelijke motivatie voor je antwoord.

---

3. Laat L_1 en L_2 twee recursief opsombare talen zijn.

Toon aan dat ook $L_1 \cup L_2$ recursief opsombaar is, door (duidelijk) de werking van een Turing machine T te beschrijven die $L_1 \cup L_2$ accepteert.

---

4. Een *modified correspondence system* is een verzameling van paren $\{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)\}$, waarbij $n \geq 1$, en de α_i's en β_i's niet-lege strings zijn over een alfabet Σ. Zo'n systeem is overzichtelijk weer te geven als een verzameling dominostenen.

   (a) Formuleer het *modified Post correspondence problem MPCP*. Dat wil zeggen: wat zijn de instanties, en wat is de ja/nee vraag bij dit beslissingsprobleem?

   In het boek wordt een reductie beschreven tussen *MPCP* en het volgende beslissingsprobleem

   > *Accepts*:
   > Gegeven een Turing machine T en een string w, is $w \in L(T)$?

   Bij deze reductie wordt voor een willekeurige instantie (T, w) van *Accepts* een instantie I van *MPCP* geconstrueerd, zó dat (T, w) een ja-instantie van *Accepts* is, dan en slechts dan als I een ja-instantie van *MPCP* is.

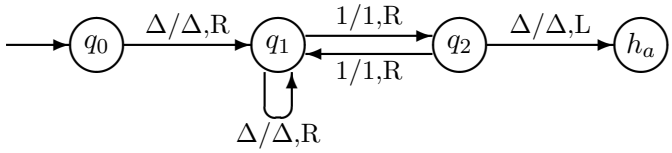
   (b) Is dit een reductie van de vorm *Accepts* $\leq$ *MPCP* of een reductie van de vorm *MPCP* $\leq$ *Accepts* ?

   (c) Wat kunnen we uit deze reductie concluderen over de relatie tussen de (on-)beslisbaarheid van *Accepts* en *MPCP* ?

   De instantie I van *MPCP*, die het resultaat is van de genoemde reductie, simuleert als het ware met 'dominostenen' de berekening van T voor invoer w.

   (d) Geef een succesvolle berekening (dat wil zeggen: een berekening van de initiële configuratie naar een accepterende configuratie) van de volgende Turing machine T voor de invoer $w = 111$:

   T:

   N.B.: dit onderdeel (d) gaat puur over een berekening met de Turing machine. Je hebt hier geen 'dominostenen' nodig.

   (e) Voer de genoemde reductie uit voor de de instantie (T, w) van *Accepts*, waarbij T de Turing machine hierboven is en $w = 111$. Dat wil zeggen: geef de *complete* instantie I van *MPCP* die met (T, w) correspondeert. Desgewenst kun je verschillende 'dominostenen' in één keer beschrijven met behulp van variabelen σ_i. Maak dan wel duidelijk wat de σ_i's precies kunnen zijn.

   Leg globaal uit waar de verschillende soorten 'dominostenen' in je instantie I voor bedoeld zijn.

Z.O.Z.

5. Bij geen enkel onderdeel van deze opgave is het nodig om projecties en dergelijke te gebruiken bij de beschrijving van de gevraagde functies. Ook hoef je bij het beschrijven van de operatie van primitieve recursie de benodigde functies g en h niet voor algemene parameters $x, y, z, \ldots$ te beschrijven.

   (a) Toon aan dat de functie *Pow* gedefinieerd door $Pow(x, y) = x^y$ primitief recursief is. Je mag ervan uitgaan dat operaties als optellen, aftrekken en vermenigvuldigen primitief recursief zijn.

   (b) Geef de definitie van μ-recursieve functies.

   (c) Toon aan dat de partiële functie $f_2 : \mathbb{N} \to \mathbb{N}$, gedefinieerd door

$$f_2(x) = \begin{cases} \text{niet gedefinieerd} & \text{als } x = 0 \\ \lceil {}^2\log x \rceil & \text{als } x \geq 1 \end{cases}$$

   ('de 2-log van x, naar boven afgerond') μ-recursief is.

   Je mag er nu van uitgaan

   - dat operaties als optellen, aftrekken en vermenigvuldigen primitief recursief zijn,
   - dat eenvoudige vergelijkingen tussen twee getallen primitief recursief zijn,
   - dat operaties als $\wedge$, $\vee$ en $\neg$ de primitieve recursiviteit behouden,
   - dat begrensde kwantificatie, begrensde minimalisatie en begrensde maximalisatie de primitieve recursiviteit behouden.

`einde tentamen`