HERTENTAMEN FUNDAMENTELE INFORMATICA 3
Donderdag 9 maart 2017, 14.00 - 17.00 uur

Dit tentamen bestaat uit vijf opgaven, waarbij steeds tussen [en] staat hoeveel punten er ongeveer mee te verdienen zijn. In totaal zijn er 100 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. [19 pt] Bij deze opgave moet je twee Turing machines construeren die werken met natuurlijke getallen. Ga hierbij uit van de unaire representatie van de natuurlijke getallen.

   Verder mag je voor beide Turing machines 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).

   (a) Construeer een Turing machine T_1 die de functie $f_1 : \mathbb{N}^2 \to \mathbb{N}$ berekent, gedefinieerd door

   $$f_1(x, y) = x \times y$$

   Leg ook duidelijk uit hoe T_1 werkt.

   (b) Construeer een *niet-deterministische* Turing machine T_2 die een willekeurig samengesteld natuurlijk getal genereert. Wellicht ten overvloede: een samengesteld natuurlijk getal is een natuurlijk getal dat te schrijven is als het product van twee natuurlijke getallen die elk minstens 2 zijn.

   Leg ook duidelijk uit hoe T_2 werkt.

---

2. [18 pt]

   (a) Wanneer noemen we een *unrestricted* grammatica $G = (V, \Sigma, S, P)$ context-gevoelig?

   (b) Laat de taal L gedefinieerd zijn door

   $$L = \{xyx^r \quad | \quad x, y \in \{a, b\}^* \text{ en } |x| = |y| \geq 1\}$$

   Een element van L bestaat dus uit een string x, gevolgd door een even lange string y, gevolgd door de omkering van x.
   Geef een context-gevoelige grammatica G, zó dat $L(G) = L$. Leg uit wat de functie is van de diverse variabelen en producties in G.
   Als het niet lukt om een context-gevoelige grammatica voor L te bedenken, kun je het grootste deel van de punten verdienen met een unrestricted grammatica voor de taal, met de bijbehorende uitleg.

   (c) Geef een afleiding in G voor het woord *abbbba*.

---

3. [23 pt] In het bewijs van Stelling 8.14 in het boek wordt beschreven, hoe je bij een willekeurige Turing machine $T = (Q, \Sigma, \Gamma, q_0, \delta)$ een unrestricted grammar $G = (V, \Sigma, S, P)$ kunt construeren, zó dat $L(G) = L(T)$.

   De grammatica G bevat drie soorten producties. De eerste soort is bedoeld om initiële configuraties van de Turing machine na te bouwen. De tweede soort is voor het simuleren van berekeningen van de Turing machine. De derde soort is voor het reconstrueren van de invoer van de Turing machine, aan het eind van de berekening.

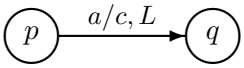
   (a) Wanneer we met G een berekening van T voor de string $a_1 \dots a_n \in \Sigma^*$ willen simuleren, wordt met de producties van de eerste soort de volgende string gegenereerd:

   $$q_0(\Delta\Delta)(a_1a_1)\dots(a_na_n)(\Delta\Delta)\dots(\Delta\Delta)$$

   Leg duidelijk uit

   - waarom de letters a_i dubbel voorkomen in de string; wat is de rol van elk van de twee voorkomens bij het simuleren van de berekening?
   - waarom er $(\Delta\Delta)$ vóór $(a_1a_1)\dots(a_na_n)$ staat;
   - waarom er $(\Delta\Delta)\dots(\Delta\Delta)$ na $(a_1a_1)\dots(a_na_n)$ staat.

   (b) Stel dat T de volgende transitie bevat:

   Geef de producties van de tweede soort in G die met deze transitie corresponderen. Je mag de producties algemeen beschrijven, met behulp van variabele(n) σ_i; zeg dan wel wat de waarde(s) van de σ_i('s) kan/kunnen zijn. Als alternatief mag je ook aannemen dat $\Sigma = \{a, b\}$ en dat $\Gamma = \{a, b, c\}$.

   (c) Stel dat we met producties van de tweede soort in G daadwerkelijk bezig zijn met het simuleren van de berekening in T. Hoe kun je dan aan de huidige string in de afleiding zien dat de leeskop van T in de gesimuleerde berekening op vakje 0 van de tape staat.

   (d) Stel dat T tijdens een berekening links van de tape afloopt. Wat gebeurt er dan in G tijdens het simuleren van de berekening? Motiveer je antwoord.

Z.O.Z.

4. [25 pt] De stelling van Rice luidt als volgt:

   Als R een niet-triviale taaleigenschap (*language property*) van Turing machines is, dan is het beslissingsprobleem

   P_R:
   Gegeven een Turing machine T, heeft T eigenschap R ?

   niet beslisbaar.

   (a) Wanneer noemen we een eigenschap R van Turing machines een *niet-triviale taaleigenschap*?

   (b) Toon met behulp van de stelling van Rice aan dat het volgende beslissingsprobleem niet beslisbaar is:

   AcceptsSomeLength: Gegeven een Turing machine T, is er een $i \geq 0$, zó dat T precies i verschillende strings van lengte i accepteert?

   Dat wil zeggen: laat duidelijk zien dat aan alle voorwaarden van de stelling van Rice is voldaan.

   (c) Beschouw nu het volgende beslissingsprobleem:

   AcceptsSomeSameLength: Gegeven twee Turing machines T_1 en T_2, is er een $i \geq 0$, zó dat T_1 en T_2 evenveel verschillende strings van lengte i accepteren?

   Toon aan dat ook *AcceptsSomeSameLength* niet beslisbaar is, met behulp van een reductie met *AcceptsSomeLength*. Wees precies in je beschrijving van de reductie. Laat uiteraard ook zien dat aan alle eisen van een reductie voldaan is, en vergeet niet om de conclusie te trekken.
   Als je geen geschikte reductie tussen AcceptsSomeSameLength *en* AcceptsSomeLength *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 $P_1 \leq P_2$.*

---

5. [15 pt]

   (a) Wat zijn de drie soorten initiële functies waaruit primitieve recursieve functies worden geconstrueerd? Vermeld hierbij ook hoeveel argumenten de functies hebben en wat hun functiewaarde is.

   (b) Toon aan dat de functie $Mult : \mathbb{N}^2 \to \mathbb{N}$, definieerd door

   $$Mult(x, y) = x \times y$$

   primitief recursief is. Je mag hierbij gebruiken dat de functie $Add : \mathbb{N}^2 \to \mathbb{N}$, gedefinieerd door $Add(x, y) = x + y$ primitief recursief is.
   Indien je gebruik maakt van de operatie primitieve recursie, geef dan de gebruikte functies g en h ook voor algemene parameters $x_1, x_2, x_3, \ldots$, en beschrijf deze functies tot op het detailniveau van de initiële functies en (eventueel) de functie Add.

`einde tentamen`