TENTAMEN FUNDAMENTELE INFORMATICA 3
Woensdag 5 juni 2013, 14.00 - 17.00 uur

Dit tentamen bestaat uit 5 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. (a) Construeer een Turing machine T_1 die als invoer een woord $x \in \Sigma^*$ van lengte $|x|$ krijgt (voor algemeen invoeralfabet Σ), en die een stuk tape van lengte $\lfloor \frac{3}{2}|x| \rfloor$ afbakent (dus $\frac{3}{2}|x|$, naar beneden afgerond).

   Om precies te zijn: als $q_0\Delta x$ de initiële configuratie van T_1 is, dan eindigt T_1 in configuratie $h_a\Delta x\Delta^i\#$, waarbij i zo gekozen is dat het symbool $\#$ (dat niet in Σ zit) op positie $\lfloor \frac{3}{2}|x| \rfloor + 1$ op de tape staat.

   Indien T_1 gebruik maakt van componenten, zul je ook die componenten moeten geven. Leg ook duidelijk uit hoe T_1 werkt.

   (b) Construeer een niet-deterministische Turing machine T_2 die als invoer een unair, niet-negatief getal x krijgt en een random unair getal y produceert met $x \leq y \leq \frac{3}{2}x$.

   Om precies te zijn: als $q_0\Delta x$ de initiële configuratie van T_2 is, dan eindigt T_2 in een configuratie $h_a\Delta y$, waarbij y een random getal is dat voldoet aan $x \leq y \leq \frac{3}{2}x$.

   Indien T_2 gebruik maakt van componenten, zul je ook die componenten moeten geven. Leg ook duidelijk uit hoe T_2 werkt.

---

2. Laat de taal L gedefinieerd zijn door

$$L = \{1^n \mid \ n \in \mathbb{N}, n = n_1 \times n_2 \text{ voor } n_1, n_2 \geq 2\}$$

   Ofwel: L bevat alle (unair gerepresenteerde, positieve) samengestelde getallen.

   (a) Geef een unrestricted grammatica G, zó dat $L(G) = L$. Leg uit wat de functie is van de diverse variabelen in G.

   Hint: genereer als het ware eerst n_1 en n_2.

   (b) Geef een afleiding in G van het woord 111111. Geef bij iedere stap in de afleiding aan wat de linkerkant is van de productie die je gebruikt.

---

3. In het boek worden onder andere de volgende twee (vergelijkbare) constructies beschreven:

   - een constructie van een unrestricted grammatica G_1 bij een gegeven Turing machine T_1, zó dat $L(G_1) = L(T_1)$;
   - een constructie van een context-gevoelige grammatica G_2 bij een gegeven lineair begrensde automaat $M_2 = (Q, \Sigma, \Gamma, q_0, \delta)$, zó dat $L(G_2) = L(M_2)$.

Wanneer de Turing machine T_1 een transitie van de vorm

$$p \xrightarrow{a/b, R} q$$

heeft, dan levert dat in G_1 producties op van de vorm $p(\sigma_1 a) \to (\sigma_1 b)q$ voor bepaalde σ_1.

(a) Leg uit waarom er tussen de haken in de productie steeds twee symbolen staan: bijvoorbeeld σ_1 en a. Wat is de functie van elk van de symbolen in zo'n paar met symbolen?

Wanneer de lineair begrensde automaat M_2 een transitie van de hierboven getekende vorm heeft, levert dat in G_2 onder andere producties op van de vorm $p(\sigma_1 a)(\sigma_2\sigma_3) \to (\sigma_1 b)q(\sigma_2\sigma_3)$ voor bepaalde $\sigma_1, \sigma_2, \sigma_3$.

(b) Leg duidelijk uit waarom in producties van dit type in G_2 het tweetal $(\sigma_2\sigma_3)$ voorkomt, terwijl dat in G_1 niet zo is.

(c) Wat zijn de mogelijke waardes voor $\sigma_1, \sigma_2, \sigma_3$ in de gegeven producties in G_2 ?

---

4. (a) Wanneer noemen we een eigenschap R van Turing machines een *taaleigenschap* (*language property*) ?

(b) Voor Turing machines T met invoeralfabet $\Sigma = \{a, b\}$ kunnen we de volgende eigenschap R definiëren (die een specifieke Turing machine wel of niet kan hebben):

> T accepteert de string *ababab* zonder ooit positie 10 (tien) op de tape te bereiken.

Toon aan dat deze eigenschap R geen taal-eigenschap van Turing machines is. Doe dit met behulp van twee concrete Turing machines T_1 en T_2, en beredeneer dat T_1 en T_2 inderdaad duidelijk maken dat R geen taaleigenschap is.

(c) Beschouw de volgende twee beslissingsproblemen:

> *AcceptsNothing*: gegeven een Turing machine T, is $L(T) = \emptyset$?
>
> *Subset*: gegeven twee Turing machines T_1 en T_2, is $L(T_1) \subseteq L(T_2)$?

Gegeven is dat *AcceptsNothing* niet beslisbaar is. Toon aan dat ook *Subset* niet beslisbaar is, met behulp van een reductie tussen *AcceptsNothing* en *Subset*.

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 Subset *en* AcceptsNothing *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. (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 door de operatie van primitieve recursie? Hoeveel argumenten heeft de functie f in dit geval?

Om aan te tonen dat alle Turing-berekenbare functies μ-recursief zijn, worden in het boek configuraties van een Turing machine gecodeerd als een getal: het configuratiegetal.

Het *one-place* predikaat $\mathit{IsConfig}_T$ ‘controleert’ of een getal $m \in \mathbb{N}$ een geldig configuratiegetal is voor Turing machine T.
De functie $\mathit{IsAccepting}_T : \mathbb{N} \to \mathbb{N}$ is 0 als zijn argument m het configuratiegetal bij een accepterende configuratie van Turing machine T is. Anders is $\mathit{IsAccepting}_T(m) = 1$.
De functie $\mathit{Move}_T : \mathbb{N} \to \mathbb{N}$ voert een getal m (als het een geldig configuratiegetal is) over in het volgende configuratiegetal. Dat wil zeggen: in het configuratiegetal corresponderend met de volgende configuratie in de berekening van Turing machine T. Ofwel: de functie Move_T correspondeert met één stap in T.

Gegeven is dat het predikaat $\mathit{IsConfig}_T$ en de functies $\mathit{IsAccepting}_T$ en Move_T primitief recursief zijn.

(b) De functie $\mathit{Moves}_T(m, k)$ voert het getal m (als het een geldig configuratiegetal is) over in het configuratiegetal na k stappen van Turing machine T.

    Toon aan dat ook de functie Moves_T primitief recursief is. Bij gebruik van de operatie van primitieve recursie hoef je de benodigde functie h niet voor algemene parameters $x, y, z, \ldots$ te beschrijven. Ook is het niet nodig om projecties en dergelijke te gebruiken bij de beschrijving van h.

(c) Ten slotte bekijken we de partiële functie $\mathit{NumberOfMovesToAccept}_T : \mathbb{N} \to \mathbb{N}$. Deze bepaalt voor zijn argument m hoeveel stappen Turing machine T nodig heeft om vanuit (de configuratie corresponderend met) m een accepterende configuratie te bereiken. Dat wil zeggen: *indien* m een geldig configuratiegetal is en T vanuit m inderdaad een accepterende configuratie kan bereiken. Anders is $\mathit{NumberOfMovesToAccept}_T(m)$ niet gedefinieerd.

    Toon aan dat $\mathit{NumberOfMovesToAccept}_T$ μ-recursief is. Maak duidelijk welke stappen je allemaal maakt in dit bewijs.

`einde tentamen`