TENTAMEN COMPUTABILITY
Donderdag 28 maart 2024, 09.00 - 12.00 uur

Dit tentamen bestaat uit zes opgaven, waarbij steeds tussen [en] staat hoeveel punten er ongeveer mee te verdienen zijn. In totaal zijn er 100 punten te verdienen. Wanneer er bij een vraag om uitleg, motivatie of toelichting gevraagd wordt, is het belangrijk om die ook te geven.

---

1. [26 pt] Deze opgave gaat over Turingmachines die werken met natuurlijke getallen (inclusief 0). We gaan hierbij uit van de unaire representatie van de natuurlijke getallen.

   (a) Teken een gewone (deterministische, 1-tape) Turingmachine T_1 die voor twee natuurlijke getallen n_1 en n_2 het product $n_1 \cdot n_2$ berekent en dat achter n_1 en n_2 op de tape zet.

   Om precies te zijn: als q_0 de starttoestand van T_1 is, dan begint T_1 in configuratie $q_0\Delta n_1\Delta n_2$ en eindigt hij in configuratie $h_a\Delta n_1\Delta n_2\Delta n_1{\cdot}n_2$.

   Als je bij dit onderdeel gebruik wilt maken van componenten, zul je die ook moeten uitwerken (tekenen dus).

   Leg ook duidelijk uit hoe T_1 werkt.

   Hint: Wellicht ten overvloede: vermenigvuldigen is herhaald optellen.

   (b) Teken een gewone (deterministische, 1-tape) Turingmachine T_2 die de functie $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ berekent, gedefinieerd door $f(n_1, n_2) = n_1 \cdot n_2$.

   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. Als je T_1 daarbij iets moet aanpassen, beschrijf de benodigde aanpassing dan precies, op het niveau van toestanden en transities. Andere componenten mag je alleen gebruiken als je ze zelf uitwerkt (tekent dus). 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).

---

2. [11 pt] Ook deze opgave gaat over Turingmachines die met natuurlijke getallen in de unaire notatie werken. Laat:

   - de niet-deterministische component G een willekeurig natuurlijk getal n genereren; d.w.z.: G voert een berekening uit van $q_0\Delta$ naar $h_a\Delta n$,
   - de component *Copy* een string x van non-blanks kopieren; d.w.z.: *Copy* voert een berekening uit van $q_0\Delta x$ naar $h_a\Delta x\Delta x$,
   - de component *Equal* controleren of twee strings x_1 en x_2 van non-blanks gelijk zijn; d.w.z.: *Equal* begint in een configuratie $q_0\Delta x_1\Delta x_2$, en accepteert, dan en slechts dan als $x_1 = x_2$,
   - de componenten *NB*, *PB* en T_2 zijn als in opgave 1(b).

   Wat is de taal die geaccepteerd wordt door de volgende, samengestelde Turingmachine T met invoeralfabet $\{1\}$:

   $$NB \to G \to Copy \to T_2 \to PB \to Equal$$

   Motiveer je antwoord door stap voor stap de inhoud van de tape te beschrijven, met de bijbehorende positie van de leeskop.

---

3. [10 pt] De Church-Turing these stelt dat elke algoritmische procedure die überhaupt uitgevoerd kan worden, kan worden uitgevoerd door een Turingmachine.

   Tijdens het college hebben we vier argumenten genoemd die deze these ondersteunen. Geef drie van deze argumenten, kort, maar voldoende duidelijk om als argument te kunnen dienen.

---

4. [23 pt] Laat de taal L als volgt gedefinieerd zijn:

   $$L = \{a^i b^j a^k \mid 0 \le i < j < k\}$$

   (a) Geef de eerste vijf elementen in de canonieke volgorde van L.

   (b) Geef een *unrestricted grammar* G, zó dat $L(G) = L$.
   Als dit je niet lukt, mag je, tegen verlies van 3 punten, een unrestricted grammar G geven voor de volgende variant van L:

   $$\{a^i b^j c^k \mid 0 \le i < j < k\}$$

   (met c in plaats van a aan het eind).
   Leg uit wat de functie is van de diverse variabelen en producties in je grammatica G.

---

5. [12 pt] We noemen een taal L *recursief opsombaar* als er een Turingmachine bestaat die L accepteert.

   (a) 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 Turingmachine T te beschrijven die $L_1 \cup L_2$ accepteert.

   (b) Geldt ook dat een oneindige vereniging van recursief opsombare talen altijd recursief opsombaar is. Ofwel: als $L_0, L_1, L_2, \ldots$ recursief opsombare talen over hetzelfde alfabet Σ zijn, dat dan ook $\cup_{i=0}^{\infty} L_i$ recursief opsombaar is?

   Zo ja, beredeneer waarom. Zo nee, geef een tegenvoorbeeld van een taal L en talen $L_0, L_1, L_2, \ldots$ over hetzelfde alfabet Σ, zó dat $L = \cup_{i=0}^{\infty} L_i$ niet recursief opsombaar is, terwijl $L_0, L_1, L_2, \ldots$ wel recursief opsombaar zijn.

---

6. [18 pt] Beschouw de volgende twee beslissingsproblemen:

   AcceptsAtleastEven:
   Gegeven een Turingmachine T_1, met invoeralfabet Σ,
   accepteert T_1 (in ieder geval) alle strings $x \in \Sigma^*$ van even lengte
   (en eventueel ook andere strings).

   AcceptsEverything:
   Gegeven een Turingmachine T_2, met invoeralfabet Σ,
   is $L(T_2) = \Sigma^*$?

   We willen aantonen dat *AcceptsAtleastEven* $\leq$ *AcceptsEverything*, ofwel dat we *AcceptsAtleastEven* kunnen reduceren naar *AcceptsEverything*.

   (a) Beredeneer, met behulp van een tegenvoorbeeld, waarom we geen geldige reductie krijgen als we simpelweg $T_2 = T_1$ nemen (met T_1 de instantie van *AcceptsAtleastEven* en T_2 de instantie van *AcceptsEverything*).

   (b) Toon aan dat desondanks geldt dat *AcceptsAtleastEven* $\leq$ *AcceptsEverything*. Doe dit door nu een geldige reductie tussen de twee beslissingsproblemen te beschrijven. Laat uiteraard ook zien dat aan alle eisen van een reductie is voldaan.

`einde tentamen`