TENTAMEN FUNDAMENTELE INFORMATICA 3
Maandag 15 april 2019, 14.00 - 17.00 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.
Geef de gevraagde Turingmachines door middel van hun transitiediagram.
Wanneer er bij een vraag om uitleg of motivatie 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. [32 pt] Laat
$$L = \{a^i b^{i+k} a^k \quad | \quad 0 \leq i < k\}$$

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

   (b) Construeer een gewone (deterministische, 1-tape) Turingmachine T_1, zó dat $L(T_1) = L$. Wanneer je voor T_1 gebruik wilt maken van componenten, zul je die ook moeten uitwerken (tekenen dus).
   Leg ook duidelijk uit hoe T_1 werkt.

   (c) Construeer een gewone (deterministische, 1-tape) Turingmachine T_2 die de functie $f_2 : \{a, b\}^* \to \mathbb{N}$, gedefinieerd door

$$f_2(x) = \begin{cases} i + k & \text{als } x = a^i b^{i+k} a^k \text{ voor zekere } i \text{ en } k \text{ met } 0 \leq i < k \\ \text{niet gedefinieerd} & \text{als } x \notin L \end{cases}$$

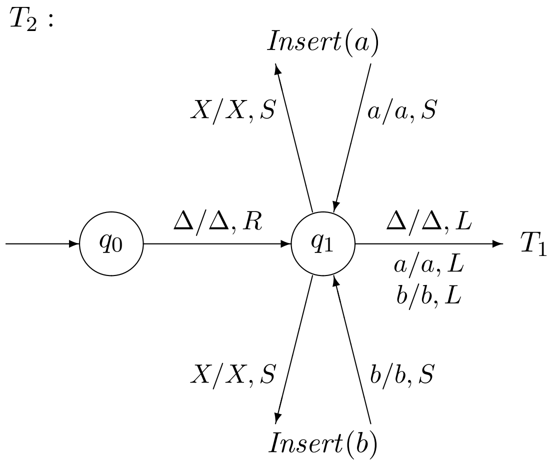
   berekent. De functiewaarde $f_2(x)$ moet unair gerepresenteerd worden, bijvoorbeeld: $f_2(abbbaa) = 111$.
   Leg ook duidelijk uit hoe T_2 werkt.
   Je mag voor T_2 gebruik maken van T_1 als component, alsmede de componenten *NB*, *PB*, *Insert*(σ) en *Delete* zoals die in het boek beschreven zijn. Andere componenten mag je alleen gebruiken als je ze zelf uitwerkt (dus tekent). 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).

   Als je T_1 iets zou moeten aanpassen voor gebruik binnen T_2, beschrijf de benodigde aanpassing dan precies.

---

2. [8 pt] Laat $L_1 \subseteq \{a,b\}^*$ een taal zijn, en laat T_1 een Turingmachine zijn, zó dat $L(T_1) = L_1$. Beschouw nu de volgende niet-deterministische Turingmachine T_2:



waarbij X elk symbool in $\{a, b, \Delta\}$ kan zijn, en waarbij de component *Insert* in opgave (1c) wordt beschreven.

Wat is $L(T_2)$? Motiveer je antwoord.

---

3. [19 pt]

   (a) Geef een *unrestricted grammar* G, zó dat

   $$L(G) = \{a^i b^{i+k} a^k \quad | \quad 0 \leq i < k\}$$

   Leg uit wat de functie is van de diverse variabelen en producties in G.

   (b) Geef een afleiding in je grammatica G uit onderdeel (a), voor het woord *abbbaa*. Wanneer je in de afleiding een aantal 'gelijksoortige' producties achter elkaar toepast, mag je die stappen samenvatten met behulp van $\Rightarrow^*$.

---

4. [13 pt]

   (a) Wanneer noemen we een taal L (volgens de definitie) recursief opsombaar?

   Wanneer noemen we een taal L (volgens de definitie) recursief?

   (b) Laat $L, L_1, L_2 \subseteq \Sigma^*$ talen zijn. Welke van de onderstaande beweringen is/zijn waar **(je hoeft je antwoorden niet toe te lichten!)**:

   i. als L recursief is, dan is ook zijn complement L' recursief;

   ii. als L_1 en L_2 recursief opsombaar zijn, dan is ook $L_1 \cup L_2$ recursief opsombaar;

   iii. als $L_1 \subseteq L_2$ en L_2 is recursief opsombaar, dan is ook L_1 recursief opsombaar.

---

5. [20 pt]

   (a) Geef een voorbeeld van twee beslissingsproblemen P_1 en P_2, zó dat $P_1 \leq P_2$, terwijl niet geldt dat $P_2 \leq P_1$.

   (b) Toon aan, voor de beslissingsproblemen P_1 en P_2 uit onderdeel (a), dat daadwerkelijk $P_1 \leq P_2$, en dat niet geldt dat $P_2 \leq P_1$.

---

6. [8 pt] De stelling van Rice luidt als volgt:

   > Als R een niet-triviale taaleigenschap (*language property*) van Turingmachines is, dan is het beslissingsprobleem
   >
   > P_R:
   > Gegeven een Turingmachine T, heeft T eigenschap R ?
   >
   > niet beslisbaar.

   Beschouw het volgende beslissingsprobleem:

   > *AcceptsInfinite*: Gegeven een Turingmachine T, accepteert T oneindig veel strings?

   Toon met behulp van de stelling van Rice aan dat *AcceptsInfinite* niet beslisbaar is. Dat wil zeggen: laat duidelijk zien dat aan alle voorwaarden van de stelling van Rice is voldaan. Gebruik (en teken) voor het aantonen van niet-trivialiteit concrete Turingmachines.

`einde tentamen`