HERTENTAMEN FUNDAMENTELE INFORMATICA 3
Maandag 17 augustus 2020, 11.15 - 14.15 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 of motivatie gevraagd wordt, is het belangrijk om die ook te geven.

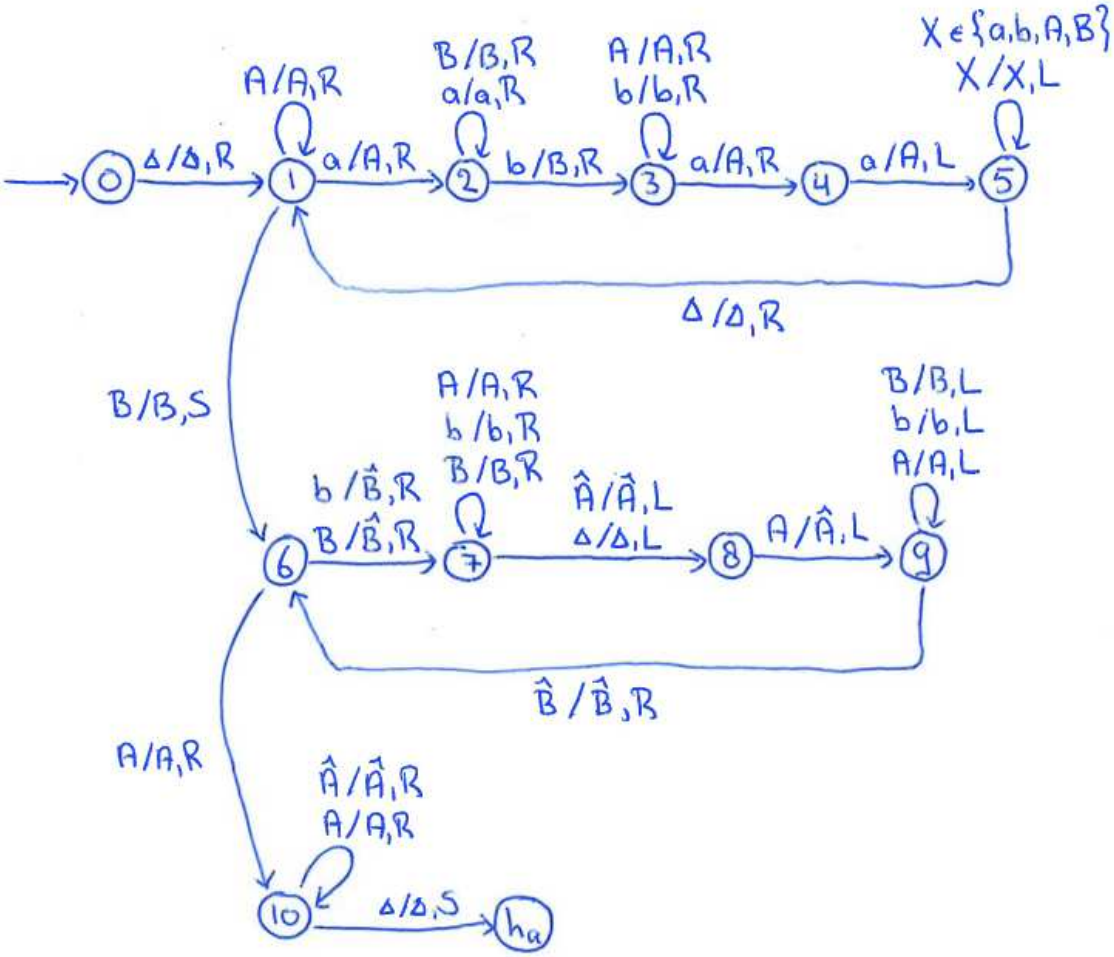
---

1. [20 pt] Als we een Turingmachine tekenen die een taal L moet accepteren, tekenen we doorgaans niet de transities naar de toestand h_r (halt-reject). Bij elke combinatie van toestand en tapesymbool waarvoor geen transitie is getekend, nemen we impliciet aan dat er een transitie naar h_r is (of dat de Turingmachine crasht).

   Toch kan het nuttig zijn om je af te vragen waar en wanneer dat crashen zou kunnen gebeuren. Bijvoorbeeld om er zeker van te zijn dat strings die niet in L zitten ook echt niet geaccepteerd worden. Daar gaat deze opgave over.

   Onderstaande Turingmachine T accepteert de taal

   $$L = \{a^i b^j a^{2i} \quad | \quad i, j \geq 0 \text{ en } i \leq j < 2i\}$$



   Een (te) korte toelichting bij deze Turingmachine: In toestanden 1 t/m 5 worden steeds van links naar rechts een a van a^i, een b van b^j en twee a's van

a^{2i} gemarkeerd als hoofdletter. Hiermee controleren we dat $i \leq j$ en dat er rechts voldoende a's staan. Als alle a's van a^i gemarkeerd zijn, gaan we naar de tweede loop, met toestanden 6 t/m 9. Daar markeren we van buiten naar binnen steeds aan de ene kant een B of b van b^j en aan de andere kant een A van a^{2i} met een dakje. Daarmee controleren we onder andere dat $j < 2i$.

Geef alle combinaties van een toestand en een tapesymbool (ook Δ tellen we hier mee) waar de Turingmachine zal crashen voor een bepaalde invoer $x \in \{a, b\}^*$ die niet in L zit. Geef bij elke combinatie van toestand en tapesymbool die je noemt, ook een invoer x waarvoor T bij die combinatie zal crashen.

Bijvoorbeeld: In toestand ... zal T crashen voor invoer $x = abba$ omdat hij op enig moment in die toestand een symbool ... op de tape ziet staan. (Natuurlijk hoef je niet iedere keer deze hele zin op te schrijven.)

---

2. [19 pt] Teken een gewone (deterministische, 1-tape) Turingmachine T die de functie $f : \{a, b\}^* \to \mathbb{N}$ berekend, die gedefinieerd wordt door $f(x) = \max\{n_a(x), n_b(x)\}$. Ofwel: T controleert welke letter (a of b) het vaakst voorkomt in zijn invoer x, en bepaalt het aantal voorkomens van die letter in x. De functiewaarde is dus een natuurlijk getal, en daarvoor gebruiken we de unaire representatie. Bijvoorbeeld: $f(aababbbab) = 11111$.

   Je mag voor T 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 (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).

   Leg ook duidelijk uit hoe T werkt.

---

3. [20 pt] Laat de taal L als volgt gedefinieerd zijn:

$$L = \{a^k b^m \mid k \geq 1 \text{ en } 2^k < m < 2^{k+1}\}$$

   Ofwel: m is geen twee-macht en k is de exponent van de grootste twee-macht die kleiner is dan m. Bijvoorbeeld, $a^4 b^{23} \in L$.

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

   (b) Geef een *unrestricted grammar* G, zó dat $L(G) = L$.
   Hint: Je kunt bijvoorbeeld $m = 23$ verkrijgen door eerst de twee-macht 16 te genereren, en die vervolgens te vergroten tot 23.
   Leg uit wat de functie is van de diverse variabelen en producties in G.

---

4. [17 pt] In het bewijs van Stelling 8.13 in het boek wordt beschreven hoe je bij een willekeurige *unrestricted grammar* G een niet-deterministische Turingmachine (NTM) T kunt construeren, zó dat $L(T) = L(G)$. De NTM moet dus precies die strings accepteren, die de grammatica genereert. De werking van T bestaat uit drie fases. De tweede fase bestaat eruit dat T op zijn tape een willekeurige afleiding in G simuleert.

   (a) Wat gebeurt er in de andere twee fases van T?

   (b) Laat $G = (V, \Sigma, S, P)$ de unrestricted grammar zijn met $V = \{S, A, T\}$, $\Sigma = \{a, b\}$ en de volgende producties:

   $$S \to AAS \mid AbT$$
   $$Ab \to bA \qquad AT \to AbT \qquad A \to a \qquad T \to \Lambda$$

   Teken een NTM T_2 die op zijn tape een willekeurige afleiding in deze grammatica G simuleert (vanuit S) en het resultaat van de afleiding op de tape achterlaat. T_2 kan dus dienen als component voor de tweede fase van T.

   Om precies te zijn: laat de string y het resultaat van de afleiding zijn, en laat q_0 de begintoestand van T_2 zijn. Dan moet T_2 beginnen in configuratie $q_0\Delta$ en na het succesvol simuleren van de afleiding eindigen in configuratie $h_a\Delta y$.

   Leg ook duidelijk uit hoe T_2 werkt.

---

5. [8 pt]

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

   (b) Laat $L_1, L_2 \subseteq \Sigma^*$ twee talen zijn. Is de volgende bewering waar:

   > Als $L_1 \subseteq L_2$ en L_2 is recursief opsombaar, dan is ook L_1 recursief opsombaar.

   Zo ja, beschrijf de werking van een Turingmachine voor L_1. Zo nee, geef een tegenvoorbeeld van twee talen L_1 en L_2 met $L_1 \subseteq L_2$, waarvoor L_2 wel recursief opsombaar is en L_1 niet.

---

6. [16 pt] Beschouw de volgende twee beslissingsproblemen:

   > *Accepts-Prefix*: Gegeven een Turingmachine T_1 met invoeralfabet $\{a, b\}$ en een string $x_1 \in \{a, b\}^*$, accepteert T_1 minstens één prefix van x_1 ?
   >
   > *Accepts-ab*: Gegeven een Turingmachine T_2 met invoeralfabet $\{a, b\}$, accepteert T_2 de string ab ?

   Toon aan dat *Accepts-Prefix* $\leq$ *Accepts-ab*. Laat uiteraard ook zien dat aan alle eisen van een reductie voldaan is.

   Hint: Simuleer een Turingmachine voor meerdere strings tegelijk.

`einde tentamen`