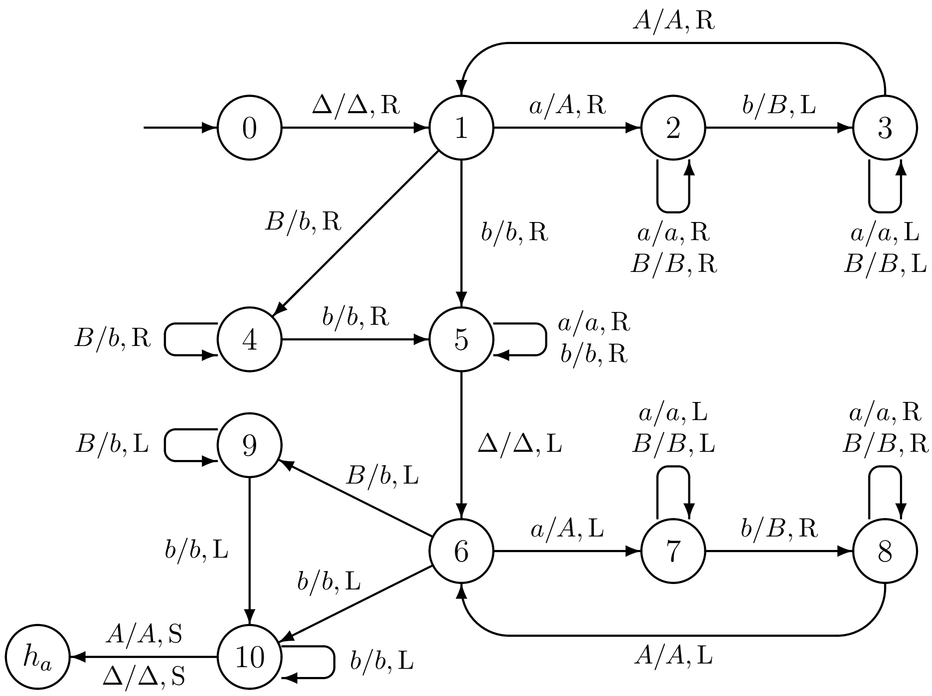
HERTENTAMEN COMPUTABILITY
Donderdag 27 mei 2021, 09.00 - 12.00 uur

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.
Wanneer er bij een vraag om uitleg, motivatie of toelichting gevraagd wordt, is het belangrijk om die ook te geven.

---

1. [25 pt] Laat
$$L = \{a^i b^j a^k \mid 0 \leq i, j, k \text{ en } j > \max\{i, k\}\}$$
De eerste zes elementen van L in de canonieke volgorde zijn b, bb, abb, bba, bbb, en $abba$.

   Beschouw nu de volgende Turingmachine T, waarvoor geldt dat $L(T) = L$:



   Een (te) korte toelichting bij deze Turingmachine: In toestanden 1, 2 en 3 worden steeds van links naar rechts een a van a^i en een b van b^j gemarkeerd als hoofdletter. Dat is bedoeld als onderdeel van de controle dat $i < j$. In toestanden 6, 7 en 8 worden steeds van rechts naar links een a van a^k en een b van b^j gemarkeerd als hoofdletter. Dat is bedoeld als onderdeel van de controle dat $k < j$.

   (a) Dit onderdeel gaat over het nut van een specifieke transitie van T.
   Voor welke invoerstrings $x \in L$ (allemaal) wordt de transitie van toestand 6 naar toestand 10 (met label $b/b, \text{L}$) gebruikt? Geef ook, voor één zo'n invoer x, de configuratie van T in de berekening voor x, direct vóórdat de transitie gebruikt wordt (en vermeld ook om welke invoer x het gaat).

   (b) Wanneer we een Turingmachine tekenen die een taal L moet accepteren, tekenen we doorgaans niet de transities naar de toestand h_r (halt-reject). Ook bij bovenstaande Turingmachine T is dat niet gebeurd. 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 dit onderdeel over.

Geef alle combinaties van een toestand en een tapesymbool (ook Δ tellen we hier mee) waar de Turingmachine T 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 $x = aaba$ omdat hij op enig moment in die toestand het symbool ... op de tape ziet staan. (Natuurlijk hoef je niet iedere keer deze hele zin op te schrijven.)

---

2. [17 pt] Een 1-tape Turingmachine om de taal

$$L = \{a^i b^j a^k \mid 0 \leq i, j, k \text{ en } j > \max\{i, k\}\}$$

te accepteren, vereist voor een invoer x al gauw een aantal stappen dat kwadratisch is in de lengte van x. Teken een **2-tapes Turingmachine** T die in een **lineair** aantal stappen bepaalt of een invoer $x \in \{a, b\}^*$ wel of niet in L zit. In het eerste geval moet T (uiteraard) accepteren, in het tweede geval verwerpen of crashen.

Als je voor T gebruik wilt maken van componenten, moeten dat wel 2-tapes componenten zijn, en zul je ze ook moeten uitwerken (tekenen dus).

Leg ook duidelijk uit hoe T werkt.

---

3. [18 pt] Laat

$$L = \{a^i b^j a^k \mid 0 \leq i, j, k \text{ en } j > \max\{i, k\}\}$$

Geef een unrestricted grammar G, zó dat $L(G) = L$.

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

Als het niet lukt om een grammatica voor L te verzinnen, mag je, tegen aftrek van 2 punten, een grammatica (met uitleg) geven voor de taal L', gedefinieerd door

$$L' = \{a^i b^j c^k \mid 0 \leq i, j, k \text{ en } j > \max\{i, k\}\}$$

Zeg het expliciet als je voor L' kiest.

---

4. [19 pt]

(a) Laat T_1 een gewone (deterministische, 1-tape) Turingmachine zijn die een taal L accepteert.

Beschrijf de werking van een Turingmachine T_2 die eveneens de taal L accepteert, maar die gegarandeerd nooit links van de tape afloopt. T_2 mag wel op een andere manier crashen, namelijk doordat er geen transitie is voor een combinatie van toestand en tapesymbool die hij in de loop van zijn berekening voor een invoer x kan tegenkomen.

Omdat we van L niets anders weten dan dat $L = L(T_1)$, zal T_2 gebruik moeten maken van component T_1, of een variant van T_1. Beschrijf T_2 en eventueel benodigde aanpassingen van T_1 duidelijk, op het niveau van de functionaliteit (je hoeft geen beschrijving op het niveau van transities te geven).

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

Als je dit antwoord niet weet, kun je het 'kopen' van de docent, zodat je het volgende onderdeel wellicht wel kunt maken.

(c) Toon aan dat als een taal L geaccepteerd kan worden door een Turingmachine T_1 die oneindig loopt voor precies één invoer x_1 (en dus voor alle andere invoeren halt), dat L dan recursief is.

Doe dit door de werking van een Turing machine T_2 te beschrijven die voldoet aan de definitie bij onderdeel (b). Als je daarbij gebruik maakt van T_1, beschrijf eventueel benodigde aanpassingen van T_1 dan duidelijk, op het niveau van de functionaliteit. Vergeet niet om je 'bewijs' netjes af te ronden met een conclusie.

---

5. [21 pt] Laat T_1 een vaste Turingmachine met invoeralfabet $\{a, b\}$ zijn. Beschouw dan het volgende beslissingsprobleem:

Accepts-T_1-Plus: Gegeven een Turingmachine T_2 met invoeralfabet $\{a, b\}$, bestaat er een string $x_1 \in \{a, b\}^*$, zó dat $L(T_2) = L(T_1) \cup \{x_1\}$?

Beschouw ook het volgende beslissingsprobleem:

Accepts-Plus: Gegeven twee Turingmachine T_3 en T_4 met invoeralfabet $\{a, b\}$, bestaat er een string $x_3 \in \{a, b\}^*$, zó dat $L(T_4) = L(T_3) \cup \{x_3\}$?

(a) 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.

Voor een van de twee beslissingsproblemen *Accepts-T_1-Plus* en *Accepts-Plus* volgt de niet-beslisbaarheid rechtstreeks uit de stelling van Rice. Welk probleem is dit?

Motiveer je antwoord door voor dit ene probleem aan te tonen dat het aan alle voorwaarden van de stelling van Rice voldoet. Gebruik voor het aantonen van niet-trivialiteit concrete Turingmachines.

(b) Voor een van de twee beslissingsproblemen *Accepts-T_1-Plus* en *Accepts-Plus* volgde de niet-beslisbaarheid dus rechtstreeks uit de stelling van Rice.

Toon aan dat ook het andere beslissingsprobleem niet beslisbaar is, met behulp van een reductie tussen de twee problemen. Laat uiteraard ook zien dat aan alle eisen van een reductie is voldaan, en vergeet niet om de conclusie te trekken.

`einde tentamen`