Uitwerking Tentamen Talen en Automaten, 1 juli 2011

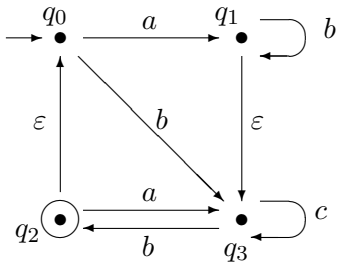
Tijdsduur 3 uur. Gesloten boek tentamen.

Je mag stellingen uit het diktaat gebruiken als je aantoont dat ze toepasbaar zijn.

Opgave 1 (12 %). Beschouw een taal L over het alphabet Σ. Vul voor de puntjes (...) één van de volgende types van talen in:
A: L is recursief
B: L is contextvrij
C: L is regulier
D: L is recursief opsombaar.
E: L is contextgevoelig
(a) L wordt geaccepteerd door een stapelautomaat $\equiv$ ...
(b) L wordt geaccepteerd door een deterministische eindige automaat $\equiv$ ...
(c) L wordt geaccepteerd door een lineair begrensde automaat $\equiv$ ...
(d) L wordt geaccepteerd door een Turingmachine $\equiv$ ...
(e) L wordt geaccepteerd door een Turingmachine die altijd eindigt $\equiv$ ...
(f) L wordt geaccepteerd door een NFAε $\equiv$ ...

Uitwerking. B, C, E, D, A, C.

Opgave 2 (12 %). Beschouw de NFAε M over het alfabet $\Sigma = \{a, b, c\}$, gegeven door



Construeer volgens het standaardalgoritme de overgangstabel van de deterministische eindige automaat M_d equivalent met M. Hoeveel toestanden heeft M_d?

Uitwerking. We noteren i voor q_i.

delta	a	b	c
-> {0}	{1,3}	{3}	{}
{1,3}	{}	{0,1,2,3}	{3}
{3}	{}	{0,2}	{3}
* {0,1,2,3}	{1,3}	{0,1,2,3}	{3}
* {0,2}	{1,3}	{3}	{}
{}	{}	{}	{}

Er zijn dus zes toestanden, waarvan twee accepterend.

Opgave 3 (16 %). (a) Formuleer het Pomplemma voor *reguliere* talen.

De taal L_3 over het alfabet $\{a, b\}$ wordt gegeven door de grammatica:

$$S \;\rightarrow\; a \mid b \mid bS \mid bSb \;.$$

(b) Bepaal de taal L_3 in verzamelingsnotatie.
(c) Bewijs dat de taal L_3 niet regulier is.

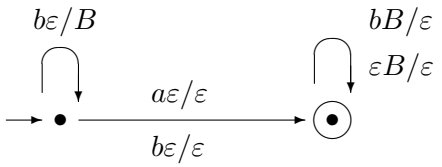
Uitwerking. (a) Voor elke reguliere taal L is er een getal k zo dat elke string $z \in L$ met $|z| \geq k$ een ontbinding $z = uvw$ heeft met $|uv| \leq k$ en $v \neq \varepsilon$ en $uv^iw \in L$ voor elke $i \geq 0$.

(b) $L_3 = \{b^i x b^j \mid (x = a \vee x = b) \;\wedge\; i \geq j\}$, want bij elke herschrijving van S waarbij S behouden blijft, komt er één b voor S bij en mogelijk één b achter S (vandaar $i \geq j$) en aan het eind wordt S herschreven tot $x = a \vee x = b$.

(c) Stel dat L_3 regulier is. Dan is er een pompconstante k als in onderdeel (a). Beschouw de string $z = b^k a b^k$. Hiervoor geldt $z \in L$ wegens onderdeel (b), en $|z| = 2k + 1 \geq k$. Volgens de pompeigenschap heeft z dus een ontbinding $z = uvw$ met $|uv| \leq k$ en $v \neq \varepsilon$ en $uv^iw \in L$ voor elke $i \geq 0$. Wegens $|uv| \leq k$ bestaan de strings u en v alleen uit b's. Dus $v = b^m$ voor zekere getal m. Er geldt $m > 0$ omdat $v \neq \varepsilon$. We nemen nu $i = 0$ en zien dan dat $uw = uv^0w \in L$. Er geldt ook $uw = b^{k-m}ab^k$, dus $uw \notin L$ wegens $m > 0$ en onderdeel (b). Dit is een tegenspraak. Dus L is niet regulier.

Opgave 4 (10 %). Beschouw nogmaals de taal L_3 uit de vorige opgave. Construeer een enkelvoudige stapelautomaat die de taal L_3 accepteert. Het is voldoende het toestandsdiagram te geven en duidelijk te maken waarom deze stapelautomaat de taal L_3 accepteert.

Uitwerking. We nemen het stapelalfabet $\Gamma = \{B\}$. We onthouden op de stapel hoeveel b er zijn ingelezen voor de nondeterministische overgang naar de accepterende toestand.



In de accepterende toestand lezen we ten hoogste zoveel b's weg als er B's op de stapel staan. De resterende B's op de stapel kunnen verwijderd worden.

Opgave 5 (10 %). Gegeven is de grammatica G over $\Sigma = \{a, b, c, d\}$ volgens

$$\begin{array}{lcl} S & \rightarrow & AB \mid dAD \\ A & \rightarrow & \varepsilon \mid aA \\ B & \rightarrow & BC \mid bD \\ C & \rightarrow & A \mid Cc \\ D & \rightarrow & CA \mid Sd \;. \end{array}$$

(a) Bepaal volgens het standaardalgoritme de *nullables* van G.
(b) Bepaal volgens het standaardalgoritme een gelijkwaardige *essentially noncontracting* grammatica.

(a) Bepaling van de nullables: A is nullable want tot ε te herschrijven. C is nullable want tot A te herschrijven. D is nullable want tot CA te herschrijven. Hier stopt het. B en S zijn niet nullable. De nullables zijn dus $\{A, C, D\}$.

(b) Het startsymbool is recursief. We maken het niet-recursief door een nieuw startsymbool T in te voeren. Voor elke geproduceerde nullable voegen we een productie toe waarin deze nullable verwijderd is. Dit geeft de grammatica G_1:

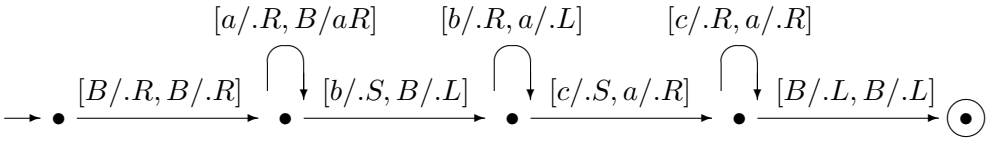
$$\begin{array}{lcl} T & \to & S \\ S & \to & AB \mid dAD \mid B \mid dD \mid dA \mid d \\ A & \to & \varepsilon \mid aA \mid a \\ B & \to & BC \mid bD \mid B \mid b \\ C & \to & A \mid Cc \mid \varepsilon \mid c \\ D & \to & CA \mid Sd \mid C \mid A \mid \varepsilon \ . \end{array}$$

We laten vervolgens de producties van ε weg (behalve indien aanwezig $T \to \varepsilon$, maar die is er niet). Dit geeft het resultaat:

$$\begin{array}{lcl} T & \to & S \\ S & \to & AB \mid dAD \mid B \mid dD \mid dA \mid d \\ A & \to & aA \mid a \\ B & \to & BC \mid bD \mid B \mid b \\ C & \to & A \mid Cc \mid c \\ D & \to & CA \mid Sd \mid C \mid A \ . \end{array}$$

Opgave 6 (10 %). Construeer een tweebands Turingmachine met invoeralfabet $\{a, b, c\}$ die de taal $\{a^i b^k c^k \mid i > k > 0\}$ accepteert, en die op de eerste band de invoerstring ongewijzigd laat en daar alleen naar rechts gaat of stationair blijft.

Uitwerking. Bedoeld werd een deterministische Turingmachine. Een niet-deterministische is alleen maar lastiger. We kopiëren de a's naar de tweede band tot we een b tegenkomen.



We gaan dan op de tweede band zoveel stappen terug als we b's tegenkomen. We gaan vervolgens op de tweede band weer zoveel stappen naar rechts als we c's tegenkomen. We verifiëren onderwijl dat het aantal c's gelijk is aan het aantal b's en dat $i > k > 0$.

Opgave 7 (4 %). Geef de definities van *recursieve* en *recursief opsombare* talen.

Uitwerking. Een taal L heet recursief als er een Turingmachine is die altijd eindigt en die L accepteert. Een taal L heet recursief opsombaar als er een Turingmachine is die L accepteert.

Opgave 8 (16 %). Zoals bekend is *TM0* de klasse van de deterministische eenbands Turingmachines over $Bit = \{0, 1\}$ met acceptatie door stoppen.
(a) Geef een *specificatie* van de universele Turingmachine *UTM*: beschrijf hoe een Turingmachine $M \in TM0$ in Bit gecodeerd wordt, beschrijf de taal

$L(UTM)$, en leg uit wat "*UTM* simuleert M" betekent (er wordt geen *implementatie* van *UTM* gevraagd).
(b) Is de taal $L(UTM)$ recursief? Geef een goede argumentatie.
(c) Is de taal $L(UTM)$ recursief opsombaar? Geef een goede argumentatie.

Uitwerking. (a) Voor een $M \in TM0$ wordt elke overgang $\delta(q, X) = [r, Y, d]$ gecodeerd als een string van de vorm $T = (1^+0)^5$, waarbij de toestanden q en r, de symbolen X en Y en de richtingen $d = L/R$ onderscheiden worden door het aantal enen, gescheiden door nullen. De Turingmachine wordt vervolgens gecodeerd als $R(M) = 00(T0)^*0$. De *UTM* verwacht invoer van de vorm $R(M)w$ waarbij de string w wordt opgevat als invoer voor M. *UTM* stopt alleen als zijn invoer van deze vorm is en machine M stopt bij invoer w. Er geldt dus

$$L(UTM) = \{R(M)w \mid M \text{ stopt bij invoer } w\} \ .$$

Dit is de Haltingtaal L_H uit het diktaat. "*UTM* simuleert M" betekent dat *UTM* stopt op invoer $R(M)w$ precies als M stopt op invoer w.
(b) $L(UTM)$ is niet recursief, volgens Turing's halting theorem. Het feit dat *UTM* niet hoeft te eindigen is een foutief argument omdat er best een andere altijd-eindigende Turingmachine zou kunnen zijn die dezelfde taal accepteert.
(c) $L(UTM)$ is recursief opsombaar omdat deze taal geaccepteerd wordt door de Turingmachine *UTM*.

Opgave 9 (10 %). Gegeven zijn twee recursieve talen L_a en L_b over een alfabet Σ. Bewijs dat de concatenatie L_aL_b recursief is. Gebruik hiertoe Turingmachines en beschrijf hoe je daarmee het gestelde bewijst.

Uitwerking. Omdat L_a en L_b recursief zijn, bestaan er Turingmachines M_a en M_b over Σ, die altijd eindigen, met de talen $L(M_a) = L_a$ en $L(M_b) = L_b$. We mogen aannemen dat M_a en M_b deterministische eenbands Turingmachines zijn. We maken een nondeterministische tweebands Turingmachine M' voor de taal L_aL_b. Deze machine knipt zijn invoer w nondeterministisch in twee stukken $w = uv$, verplaatst de tweede helft v naar de tweede band, en laat dan M_a los op u op de eerste band, en (bv. daarna) M_b op v op de tweede band. De invoer w wordt geaccepteerd als u geaccepteerd wordt door M_a en v geaccepteerd wordt door M_b. Aldus wordt de taal L_aL_b geaccepteerd door een nondeterministische Turingmachine M'. De berekening eindigt altijd omdat M_a en M_b altijd eindigen. De taal is dus recursief (zie stelling in het diktaat).