Postcode en Woonplaats:

Jaar van eerste inschrijving:

Datum: 28-06-2004

Naam docent:

5 i ~~$N(P) = \{ w \in \Sigma^* \mid (q, \alpha, \varepsilon) \in \hat{\delta}(q_0, w, Z_0),\ q \in Q, \alpha \in \Sigma^* \}$~~

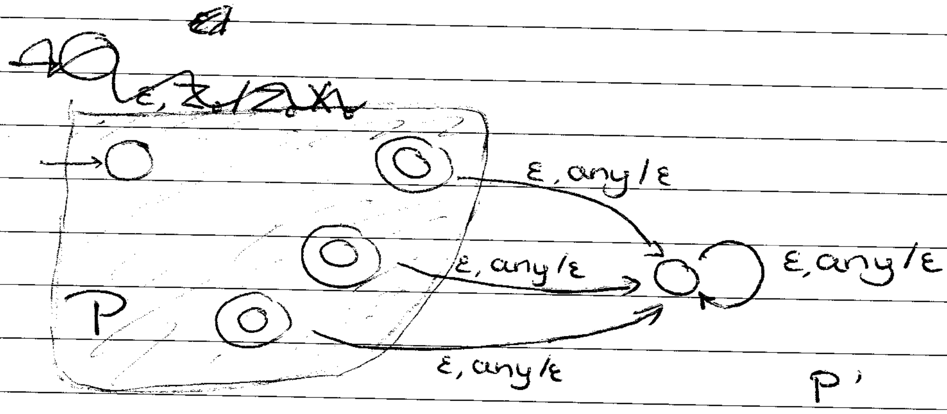
5

~~$L(P) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w, Z_0) \ni (q, \varepsilon, \beta),\ q \in F, \beta \in \Gamma^* \}$~~

$$N(P) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon),\ q \in Q \}$$

$$L(P) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \beta),\ q \in F,\ \beta \in \Gamma^* \}$$

ii



√ nieuw element onderaan

3 Vanuit alle eindtoestanden, gaan we naar een nieuwe toestand en we halen dan alle stack-symbolen weg, terwijl we de overgebleven inputsymbolen onaangetast laten.
Wanneer er geen inputsymbolen meer over waren, zal P' dus accepteren, want de inputstring zat dan in N(P).

3 i $L(A) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F \}$

5 $\Sigma^* \setminus L(A) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F \}$

$= \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q \setminus F \} = L(A')$

ii ~~$L(A) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset \}$~~

~~$\Sigma^* \setminus L(A) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F = \emptyset \}$~~

~~$L(A') = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap (Q \setminus F) \neq \emptyset \}$~~

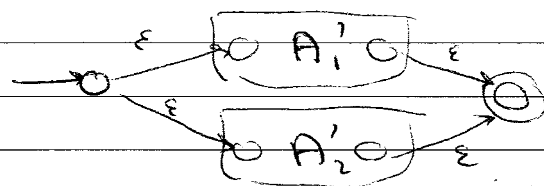
3 iii $L(A_1) \cap L(A_2) = \overline{\overline{L(A_1) \cap L(A_2)}}$

$= \overline{\overline{L(A_1)} \cup \overline{L(A_2)}}$

$\Rightarrow \Big(L(A_1) \cap L(A_2) = \emptyset$

$\Leftrightarrow \overline{\overline{L(A_1)} \cup \overline{L(A_2)}} = \emptyset \Big)$

~~met~~ van de DFA's voor $L(A_1)$ en $L(A_2)$ kunnen ~~wen~~ eenvoudig DFA's maken voor $\overline{L(A_1)}$ en $\overline{L(A_2)}$ (zie onderdeel i).
~~Dan~~ Met twee DFA's kunnen we ~~ook~~ ook een ~~DFA's~~ ~~ε-NFA~~ DFA maken ~~voor~~ die ~~act~~ strings accepteert als ze in één van beide talen zitten.
We kunnen nl. een ε-NFA maken:



dus ook een DFA.

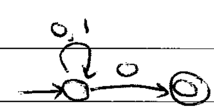
We kunnen dan testen of een string in $\overline{L(A_1)} \cup \overline{L(A_2)}$ zit met een DFA.

5

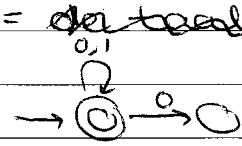
Maar dan kunnen we ook een DFA maken voor $\overline{\overline{L(A_1)} \cup \overline{L(A_2)}} = L(A_1) \cap L(A_2)$.
En er bestaat een algoritme om alle toestanden te bepalen die vanuit de begintoestand bereikbaar zijn.
Als de accepterende toestand er niet bij zit, dan is ~~taal~~ de taal leeg, anders niet.

8

Hiervoor levert de theorie in feite direct een algoritme (zie p. 155 in boek)

ii neem A: 

$L(A) =$ ~~de taal van~~ $\{0^n \mid n \geq 1\}$

A': 

$\Sigma = \{0, 1\}$

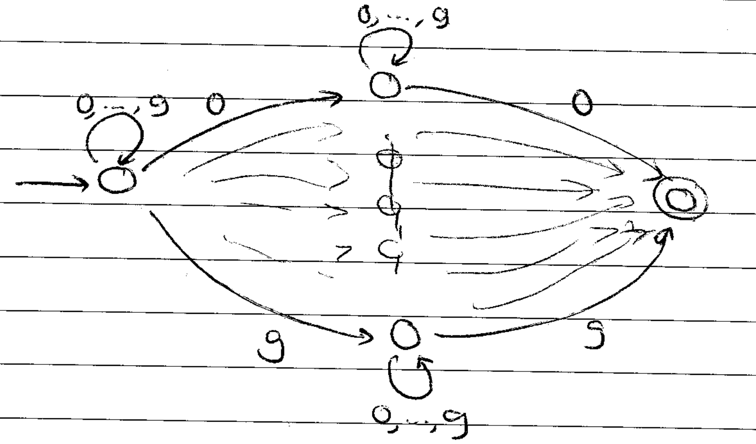
5

dan geldt dat $00 \in L(A)$ en $00 \in L(A')$
$\Rightarrow L(A) \cap L(A') \neq \emptyset$
dus $L(A')$ is niet het complement van $L(A)$

8

Postcode en Woonplaats: | Jaar van eerste inschrijving: | Datum: Naam docent:

1 5 i



5 ii $(0+\dots+9)^* 0 (0+\dots+9)^* 0$

$+ \quad \dots$

$+ (0+\dots+9)^* 9 (0+\dots+9)^* 9$

5 iii $L(\emptyset) = \emptyset$ is eindig

$L(\varepsilon) = \{\varepsilon\}$ } eindig

$L(a) = \{a\}$

stel R_1 en R_2 reguliere ~~als~~ expressies met $L(R_1)$ en $L(R_2)$ eindig

$\Rightarrow$

- $L(R_1+R_2) = L(R_1) \cup L(R_2)$ eindig (eindige vereniging van eindige talen)
- $L(R_1R_2) = L(R_1)L(R_2)$ eindig (eindige concatenatie van eindige talen)
- $L((R_1)) = L(R_1)$ is eindig

dus als R een reguliere expressie zonder * is, dan is L(R) eindig

2 2 i a ECLOSE(q) is de verzameling ~~van~~ die q bevat ~~en alle~~ van alle toestanden die bereikbaar zijn vanuit q met een pad met een lengte groter dan of gelijk aan nul, waarbij ~~alle~~ het pad alleen pijlen met een ε bevat

3 b $q \in ECLOSE(q)$ en

(2 i b) $\{ p \in ECLOSE(q) \wedge r \in \delta(p, \varepsilon) \Rightarrow r \in ECLOSE(q)$

2   3 ii a $\begin{cases} \hat{\delta}(q, \varepsilon) = q & q \in Q \\ \hat{\delta}(q, xa) = \delta(\hat{\delta}(q,x), a) & x \in \Sigma^* \; a \in \Sigma \end{cases}$

4   b b_1 $n = 0 \Rightarrow \hat{\delta}(q, \varepsilon) = q$ (zie definitie $\hat{\delta}$)

stel $\hat{\delta}(q, a^n) = q$

$\Rightarrow \hat{\delta}(q, a^{n+1}) \underset{\text{def. } \hat{\delta}}{=} \delta(\hat{\delta}(q, a^n), a) \underset{\text{gegeven}}{=} \hat{\delta}(q, a^n) \underset{\text{hypothese}}{=} q$

dus $\hat{\delta}(q, a^n) = q \quad \forall n \in \mathbb{N}$

0   b_2 $L(A) = L(a^*) L(a_1) L(a^*) \cancel{L(a^*)} L(a_2) L(a^*) \ldots L(a_k) L(a^*)$ voor een bepaalde $k \in \mathbb{N}$

?

4   3 i

$$\left.\begin{array}{l} S \to AC \\ A \to \varepsilon \mid aAb \\ C \to \varepsilon \mid cCd \end{array}\right\} P \qquad G = (\{S, A, C\}, \{a, b, c, d\}, P, S)$$

3 ii $L(G) \subset L(G')$, want het nieuwe startsymbool is A en we hebben productie $A \to S$ (waarbij S het oude startsymbool)

de andere nieuwe productie is $A \to cAd$

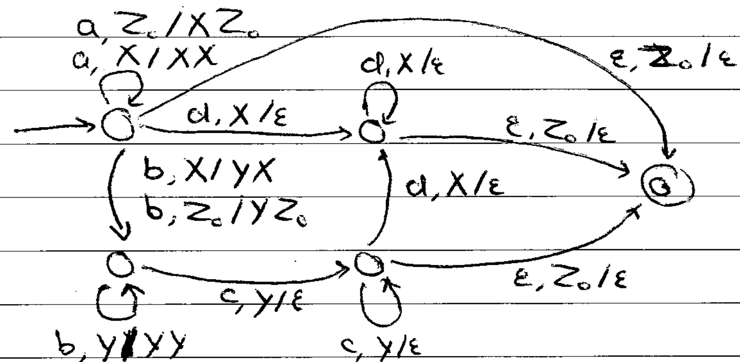
$\Rightarrow L(G') \cancel{= L(c^*) L(G) L(d^*)}$

$= \{c^n w d^n \mid w \in L(G), n \in \mathbb{N}\}$

3 iii

$$\left.\begin{array}{l} S \to \varepsilon \mid aSd \mid B \\ B \to bc \mid bBc \end{array}\right\} P \qquad G = (\{S, B\}, \Sigma, P, S)$$

3 iv voor elke CFG is er een PDA, en L_2 heeft een CFG, dus ook een PDA:



Postcode en Woonplaats: | Jaar van eerste inschrijving: | Datum: Naam docent:

4 9 v Stel L_1 is regulier, dan bestaat er een $n \in \mathbb{N}$ zodanig dat als $w \in L_1$ en $|w| \geq n$, dan kunnen we w schrijven als $w = xyz$ waarbij $y \neq \varepsilon$, $|xy| \leq n$ en $xy^kz \in L_1$ voor $k \geq 0$.
Neem nu $w = a^n b^n$, dan bestaat ~~xy uit~~ is ~~uit~~ $xy = a^m$ voor $1 \leq m \leq n$ en $y = a^l$ met $1 \leq l \leq n$ en $xy^0z \in L_1$, dus $xz \in L_1$
$\Rightarrow xz = a^{n-l} b^n \notin L_1$ (want $n - l \neq n$)
we hebben een tegenspraak, want een string kan niet tegelijk wel en niet in L_1 zitten
$\Rightarrow L_1$ niet regulier

2 vi a $L = L_1 \cap L_2 = \{a^n b^n c^n d^n \mid n \in \mathbb{N}\}$

4 b Als L een contextvrije taal is, dan bestaat er een $n \in \mathbb{N}$ zodanig dat als $z \in L$ en $|z| \geq n$, dan is $z = uvwxy$ en $vx \neq \varepsilon$, $|vwx| \leq n$ en $uv^iwx^iy \in L$ $\forall i \geq 0$.

6 i neem $M = (Q, \Sigma, \Sigma \cup \{B\}, \delta', q_0, B, \cancel{F}\ \{q_e\})$
3 waarbij $\delta'(q,a) = (\delta(q,a), a, R)$
Er is nodig: een nieuwe accepterende toestand q_e.
Neem verder (enkel) nog: $\delta'(q, B) = (q_e, B, R)$ voor $q \in F$.
ii Als een taal L regulier is, dan bestaat er een DFA voor L. Een DFA voor een taal L is om te zetten naar een ~~DFA~~ terminerende TM voor L (zie onderdeel i).
2 Dus als L regulier is, bestaat er een (terminerende) TM voor L. En als er een TM voor L bestaat, dan is L recursief.
Dus een reguliere taal L is recursief.

iii a L_1 en L_2 recursief $\Rightarrow$ er bestaan algoritmes voor L_1 en L_2

willen we testen of een string w in $L_1 L_2$ zit, dan ~~kunnen~~ kunnen we de string opdelen in twee delen: $w = xy$

L is recursief ⟺ er is een algoritme voor L.

(6 iii) voor elke w zijn er een eindig aantal verschillende opdelingen mogelijk.
We kunnen dus voor elke opdeling nagaan of ~~..~~ x in L_1 zit en y in L_2.
Wanneer dit voor een bepaalde opdeling geldt, dan zit w in L_1L_2.

4

Wanneer het voor geen ~~..~~ enkele opdeling geldt, dan zit w niet in L_1L_2.
We kunnen dus testen of w in L_1L_2 zit, ~~..~~ dus L_1L_2 is recursief.

b als L_1 en L_2 recursief opsombaar, dan zijn er TMs ~~voor L_1 en L_2~~ M_1 en M_2 met $L(M_1) = L_1$ en $L(M_2) = L_2$.

Nu delen we de inputstring w weer op in twee delen: $w = xy$.

4

Voor alle mogelijke opdelingen (dit zijn er eindig veel voor elke w) stoppen we x in M_1 en y in M_2.
Als ~~x en y ac~~ M_1 en M_2 beide accepteren ~~voor~~ bij een bepaalde opdeling, dan weten we dat $w \in L_1L_2$.

Geef iets meer details

dit moet tegelijk, want ~~..~~ op sommige strings stoppen de TMs mogelijk niet