Tavlenoter
Forelæsning om Endelige automater

Rasmus Gadensgaard

5. februar 2013

Indhold

1 Om kurset

Eksamen:

- Skriftlig
- Opgaver til hver kursusgang ligner eksamensopgaver
- Ingen hjælpemidler til eksamen pånær egne svar på tekstspørsmål "portfolio"
- I kurset skal man bedømme andres tekstspørgsmål for at få ens egne svar i sin portfolio.
- 80–85 % bestod sidste år

Kurset omhandler matematiske teorier for programmers form og adfærd. Målet med disse teorier er at kunne give beskrivelser, der er uafhængige af en implementation. Det indeholder to dele:

Syntaks Der svarer på spørgsmålet: Hvad kan man skrive?

Semantik Der svarer på spørgsmålet: Hvad sker der når et program bliver udført?

Hver forelæsning introducerer nyt stof, så man skal læse *efter* hver forelæsning (men man må selvfølgelig også gerne læse stoffet inden).

2 Indledning

Denne forelæsning handler om

- Eksempel {strenge, sprog osv.}
- Definition af DFA'er
- Eksempler
- Definition af accept, regulære sprog
- Regulære operationer
- Regulære sprog lukket under $\cup$
- Gyser

3 Grundlæggende begreber om sprog

Definition	Eksempel
Et *alfabet* er en endelig mængde af tegn	$B = \{0, 1\}$ $A = \{a, b, c, d, e\}$
En *streng* over alfabet Σ er en endelig følge af tegn fra Σ	$001 \in B^*$, $aab \in A^*$, $00a \notin B*$
Mængden af alle strenge kalder vi Σ^*	
Hvis $u, v \in \Sigma^*$ betegner uv *kontatenationen* af u og v	$aab \in A^*$, $fed \in A^*$, $aabfed \in A^*$
Et sprog over Σ er en delmængde af Σ^*, dvs. en mængde af strenge Σ	$\emptyset$ er et sprog over B, $\{0, 1\}$ er et sprog over B, $\{0, 00, 000, 000, \ldots\}$ er et sprog over B, $\{aab, fete, acd\}$
Den tomme streng kaldes ε	

4 Endelige automater

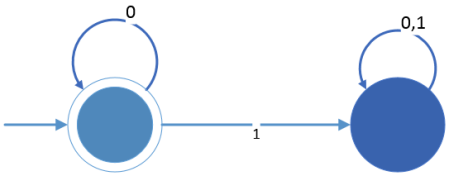
En endelig automat er en lille algoritme som afgør om en streng er en del af et sprog.

Eksempel 1. I Figur 1 ses et eksempel på en endelig automat.

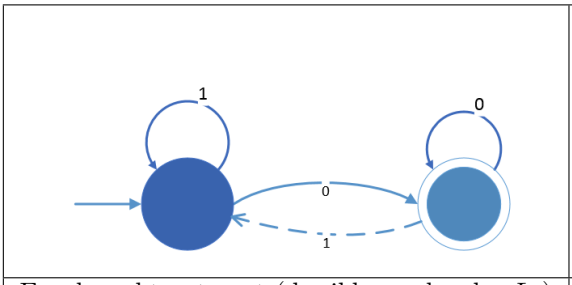
Eksempel 2. Betragt sproget

$$L_1 = \{w \in \{0, 1\}^* \mid w \text{ starter med 1, slutter med 0 }\}$$

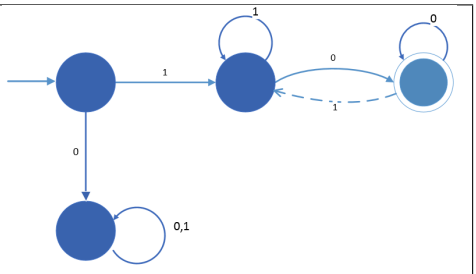
Figur 2 viser dels en automat, der ikke genkender L_1, dels en automat, der genkender L_1.



Figur 1: En endelig automat M_1. Strengen 000 accepteres af M_1. Derimod bliver strengen 001 ikke accepteret.



En ukorrekt automat (der ikke genkender L_1)



En korrekt automat for L_1

Figur 2: To endelige automater

4.1 Definition af endelig automat

Definition 1. En endelig automat M er en 5-tupel

$$(Q, \Sigma, q_0, \delta, F)$$

hvor

$$\begin{aligned} Q &\text{ er en endelig mængde af tilstande} \\ \Sigma &\text{ er et endeligt alfabet} \\ q_0 &: \text{Startilstand, } q_0 \in Q \\ \delta &: \text{Overføringsfunktion, } \delta : Q \times \Sigma \to Q \\ F &: \text{Mængde af accepttilstande, } F \subseteq Q \end{aligned}$$

Eksempel 3. Betragt igen automaten fra Eksempel 1. Vi har

$$\begin{aligned} Q &= \{A, B\} \\ \Sigma &= \{0, 1\} \\ q_0 &= A \\ F &= \{A\} \end{aligned}$$

og overføringsfunktionen kan skrives op i tabelform som

	A	B
0	A	B
1	B	B

Definition 2 (Accept af streng)**.** Lad

$$A = (Q, \Sigma, \delta, q_0, F)$$

være en DFA og lad

$$w \in \Sigma^*, w = a_1 \dots a_k$$

A accepterer w hvis A ved at læse w kan besøge tilstande $r_0, ..., r_k$ så $r_k \in F$, dvs. der skal findes en sådan følge af tilstande som opfylder at

- $r_0 = q_0$
- for alle $0 \leq i \leq k$ har vi at $\delta(r_i, a_i) = r_i + 1$
- $r_k \in F$

Definition 3 (Sproget genkendt af en automat)**.** Sproget *genkendt* af en DFA A er givet ved

$$L(A) = \{w \in \Sigma^* \mid A \text{ accepterer } w\}$$

Definition 4 (Regulært sprog)**.** Et sprog kaldes regulært hvis det genkendes af en DFA

5 De regulære operationer

De regulære operationer er 3 operationer, der kan anvendes til at bygge sprog med.

Lad L_1, L_2 være sprog. Så er:

$$L_1 \cup L_2 = \{x \mid x \in L_1 \text{ eller } x \in L_2\} \qquad \text{(Foreningsmængde)}$$
$$L_1 \circ L_2 = \{x \mid \text{Der findes } u, v \text{ så } x = uv\} \qquad \text{(Konkatenation)}$$

Lad L være et sprog. Så er

$$L^* = \{x_1 \dots x_k \mid k \geq 0, x_i \in L \text{ for alle } 0 \leq i \leq k\} \qquad \text{(Kleene-stjerne)}$$

Bemærk at vi ud fra definitionen får at ε altid er element i L^*.

Foreningsmængde og konkatenation kan imidlertid give et tomt sprog i visse tilfælde. Vi har at

$$L_1 \cup L_2 = \emptyset \iff L_1 = L_2 = \emptyset$$

$$L_1 \circ L_2 = \emptyset \iff L_1 = \emptyset \text{ eller } L_2 = \emptyset$$

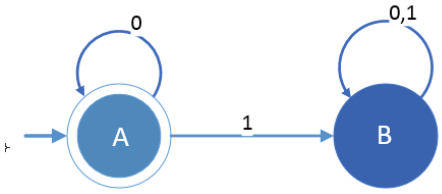
Vi har også at

$$\emptyset^* = \epsilon$$

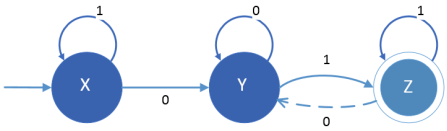
da ε altid kommer med, når vi anvender stjerne-operationen.

6 De regulære sprog er lukket under $\cup$

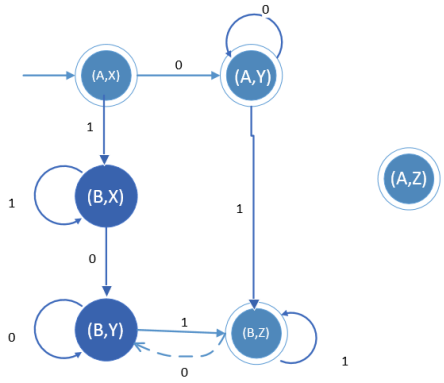
Sætning 1. *Hvis L_1 og L_2 er regulære sprog, så er $L_1 \cup L_2$ også et regulært sprog.*



Figur 3: Endelig automat M_1



Figur 4: Endelig automat M_2



Figur 5: Endelig automat, der kører M_1 og M_2 parallelt og genkender $L(M_1) \cup L(M_2)$

Ideen i beviset er at lave en automat, der genkender $L_1 \cup L_2$ ved at køre automater for L_1 og L_2 i parallel.

Figurerne 3 og 4 viser et eksempel på to automater. Figur 5 viser et

Bevis: Vi skal lave en DFS der genkender $L_1 \cup L_2$. Vi ved at L_1 regulært, dvs. der findes en DFA M_1 så $L(M_1) = L_1$ og at L_2 regulært dvs. der findes en DFA M_2 så $L(M_2) = L_2$.

Lad

$$M_1 = (Q_1, \Sigma, q_1, \epsilon_1, F_1)$$
$$M_2 = (Q_2, \Sigma, q_2, \epsilon_2, F_2)$$

Vores DFA for $L_1 \cup L_2$ er da

$$Q = Q_1 \times Q_2$$
$$q_0 = (q_1, q_2)$$
$$F = \{(p, q) \mid p \in F_1 \text{ eller } q \in F_2\}$$

Overføringsfunktionen δ er givet ved

$$\delta((p, q), a) = (r_1, r_2) \text{ hvor } \delta_1(p, a) = r_1 \text{ og } \delta_2(q, a) = r_2$$

Denne *produktkonstruktionen.* □

Ud fra produktkonstruktionen er det meget nemt at vise at de regulære sprog også er lukket under fællesmængde og under mængdedifferens. Vi konstruerer en automat med samme slags tilstande og samme slags overføringsfunktion; kun accepttilstandene er defineret anderledes.

- Hvis L_1, L_2 regulære, så er $L_1 \cap L_2$ regulært. Her sætter vi

$$F = \{(p, q) \mid p \in F_1 \text{ og } q \in F_2\}$$

- Mængdedifferens er defineret ved

$$L_1 \setminus L_2 = \{x \mid x \in L_1, x \notin L_2\}$$

  Så har vi at hvis L_1, L_2 er regulære, så er $L_1 \setminus L_2$ regulært. Her sætter vi

$$F = \{(p, q) \mid p \in F_1, q \in F_2\}$$

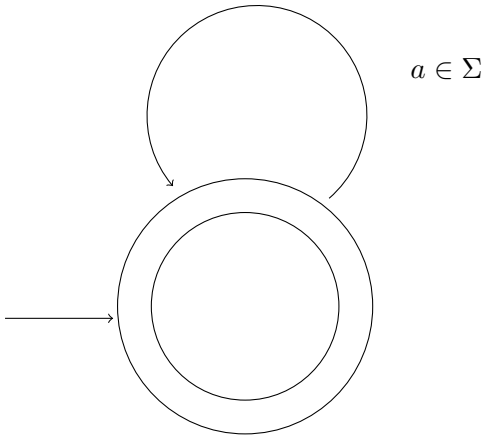
7 Gyser

- Få styr på terminologien! – det er vigtigt at kunne tale fagets sprog. Vi taler ikke om 'sæt', 'stadier', 'transaktioner' osv. Og husk at 'sprog' og 'streng' ikke betyder det samme og ikke må anvendes i flæng!! *Et sprog er en mængde af strenge.*

- En populær misforståelse blandt studerende er at den tomme streng skulle være med i ethvert sprog. Det gælder selvfølgelig ikke. Den tomme streng er ikke element i f.eks. $\{aab, abba\}$. Personer, der holder af denne misforståelse, har ofte fejlhørt et udsagn om Kleene-stjerne. Det er korrekt at vi for ethvert sprog L har at $\varepsilon \in L^*$.

- En anden populær misforståelse blandt studerende er at konkatenation og kartesisk produkt 'er det samme'. Men kontatenation er en operation, der bygger sprog (et sprog er en *mængde af strenge*), mens kartesisk produkt bygger en mængde af ordnede par. Lad

$$A = \{aab, abba\}$$
$$B = \{bb, bbb, ba\}$$



Figur 6: En automat A, som mange fejlagtigt holder af

Vi har

$$A \circ B = \{aabbb, aabbbb, aabba, abbabb, abbabbb, abbaba\}$$

som er en mængde af strenge, mens

$$A \times B = \{(aab, bb), (aab, bbb), (aab, ba), (abba, bb), (abba, bbb), (abba, ba)\}$$

Dette er to forskellige mængder.

Misforståelsen opstår formodentlig ofte, fordi nogle studerende har det med at glemme at sætte parenteser om ordnede par.

- Hvad betyder det at de regulære sprog er lukket under $\cup$? Nogle tror at det betyder at foreningsmængden af alle de regulære sprog er regulært. Men det gør det ikke.

- Mange glemmer, at sprog kan have uendeligt mange elementer. Et eksempel er sproget af alle strenge, der består af 0 eller flere 0er:

$$\{\epsilon, 0, 00, 000, ...\})$$

- En populær misforståelse er at alle sprog er regulære. De, der holder af denne populære misforståelse, siger at det kan indses ved at betragte DFA'en A på Figur 6.

  Det er klart at det for ethvert sprog L gælder at denne automat accepterer alle strenge fra L. Desværre accepterer den også alle andre strenge! Vi har faktisk at $L(A) = \Sigma^*$.

  En automat, der genkender et sprog L skal acceptere de strenge, der er elementer i L *og kun dem*.