F. Drewes 2006-03-30

Datavetenskapens grunder, VT06

– Gruppövning 1 –

1. Tag fram ändliga automater (DFA eller NFA) som accepterar följande språk över $\{a, b, c\}$:

   - mängden av alla strängar som innehåller minst ett c, där inget b förekommer efter det sista c:et
   - mängden av alla icke tomma strängar vars sista symbol förekommer minst två gånger i strängen.

   I vilket/vilka fall är det fördelaktigt att använda sig av en NFA?

2. Låt $n \geq 1$ vara något positivt naturligt tal. Hur stor är NFA:n som accepterar språket

   $$A_n = \{a_k \cdots a_1 \mid a_1, \ldots, a_k \in \{0, 1\} \text{ för något } k \geq n, \text{ där } a_n = 1\}?$$

   (Om t.ex. $n = 10$ ska alltså den tionde symbolen bakifrån vara 1.) Hur stor är motsvarande DFA?

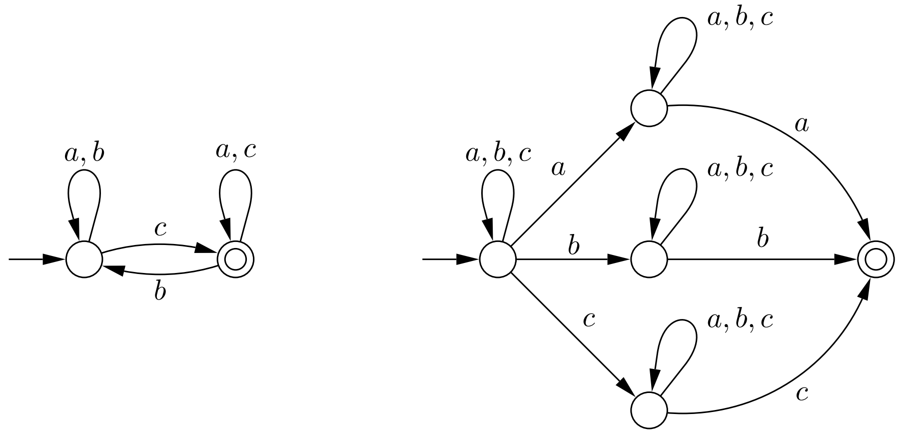
3. Om $A, A' \subseteq \Sigma^*$ är reguljära så är också

   (a) $A_1 = \Sigma^* \setminus A$,
   (b) $A_2 = \{a_n \cdots a_1 \mid a_1 \cdots a_n \in A\}$ och
   (c) $A_3 = A \cap A'$

   reguljära. Försök hitta övertygande argument för det.

Kommentarer...

1. Två lämpliga automater är



   I första fallet hjälper ickedeterminism inte mycket men i det andra vore en DFA något krångligare eftersom man skulle vara tvungen att komma ihåg mängden av symboler som redan har lästs samt vilken symbol var den som lästes sist.

2. NFA:n består av $n+1$ tillstånd $q_0, \ldots, q_n$, där q_0 är starttillståndet och q_n är det unika accepterande tillståndet. Transitionerna är

   - från q_0 till q_0 under både 0 och 1
   - från q_0 till q_1 under 1
   - från q_i till q_{i+1} under både 0 och 1 för $i = 1, \ldots, n-1$.

   En DFA som accepterar samma språk kräver däremot 2^n tillstånd eftersom den måste komma ihåg de senaste n symboler som har lästs.

3. Argumentationsmönstret är alltid att man visar hur en automat för språket i frågan kan konstrueras utifrån automater för A och A'.

   (a) Betrakta en DFA $(Q, \Sigma, \delta, q_0, F)$ som accepterar A. Det borde vara klart att DFA:n $(Q, \Sigma, \delta, q_0, Q \setminus F)$ accepterar A_1, som således också är reguljärt.
   OBS! Det fungerar inte om originalautomaten inte är deterministisk!

   (b) Här kan man resonera på följande sätt. Betrakta en NFA $A = (Q, \Sigma, \delta, q_0, F)$ som accepterar A och se till att den endast innehåller ett accepterande tillstånd, dvs $F = \{q\}$ för något $q \in Q$. (Om F inte uppfyller kravet, lägg till ett nytt tillstånd q dit ϵ-transitioner pekar från alla $q' \in F$. Låt sedan q vara det unika accepterande tillståndet.)
   Nu skapas en ny NFA $A' = (Q, \Sigma, \delta', q, \{q_0\})$ vars transitionsfunktion δ' fås av δ genom att vända på alla transitioner, dvs

$$\delta'(q_1, a) = \{q_2 \in Q \mid q_1 \in \delta(q_2, a)\}.$$

Tydligen accepterar A' språket A_2.

OBS! Igen vore det mycket krångligare att konstruera en DFA utifrån den givna automaten även om man antar att den är deterministisk.

(c) Antag att $(Q, \Sigma, \delta, q_0, F)$ och $(Q', \Sigma, \delta', q_0', F')$ är DFA:er som accepterar A respektive A'. Då accepteras A_3 av

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

där $\delta''((q, q'), a) = (\delta(q, a), \delta'(q', a))$ för alla $q \in Q, q' \in Q', a \in \Sigma$. I den nya automaten jobbar alltså originalautomaterna "parallellt". (Jämför med beviset av teorem 1.25 i kursboken, resp. teorem 1.12 i första upplagan.)