Automatateori 2021-10-22

Lösningsförslag

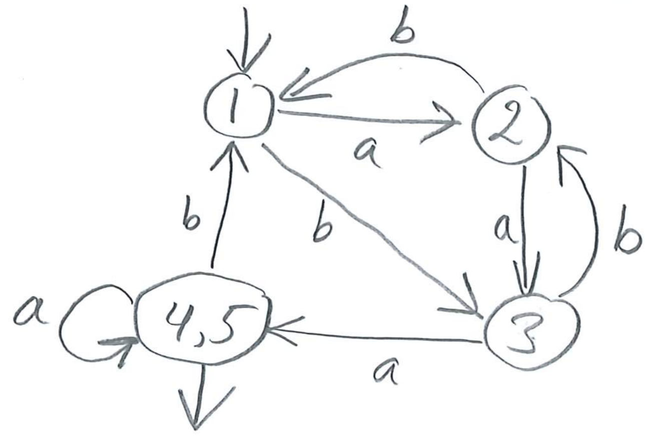
1. Om tillstånden numreras medurs med början i starttillståndet så fås följande tabell över tillståndsövergångarna:

	1	2	3	4	5
a	2	3	4	5	4
b	3	1	2	1	1

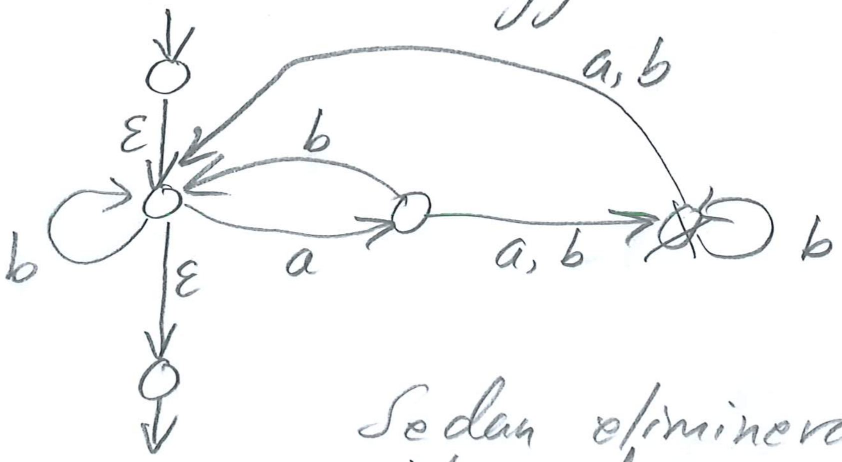
Särskiljandekonstruktionen ger sedan följande uppdelningar av tillståndsmängden:

nivå	uppdelning
1	{1, 2, 3} {4, 5}
2	{1, 2} {3} {4, 5}
3	{1} {2} {3} {4, 5}
4	{1} {2} {3} {4, 5}

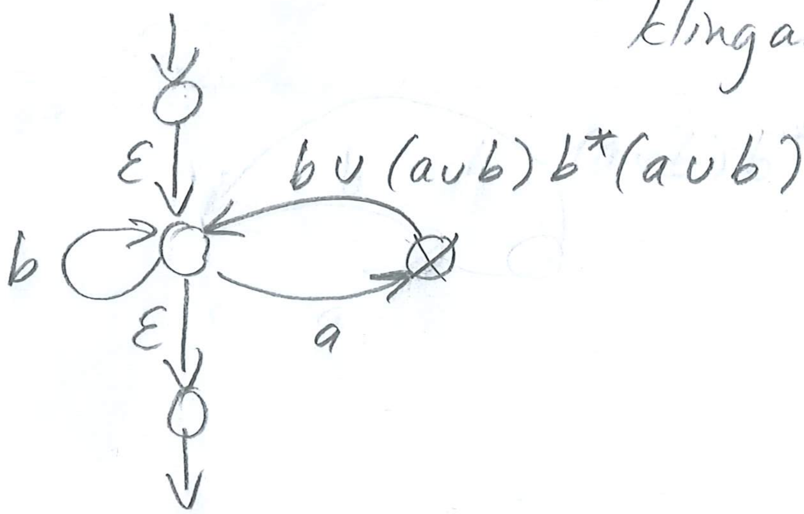
När två nivåer är likadana är vi klara med uppdelningen och kan konstruera en minimal DFA så här:

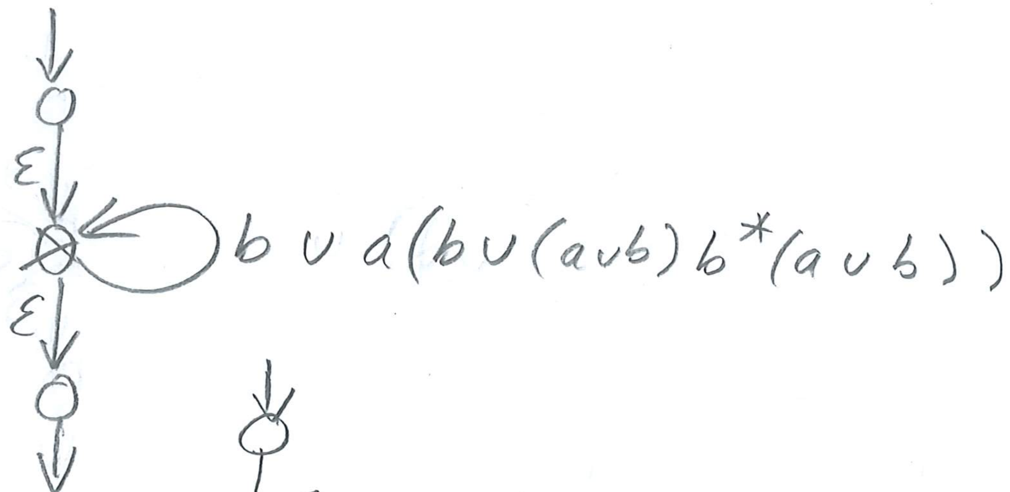


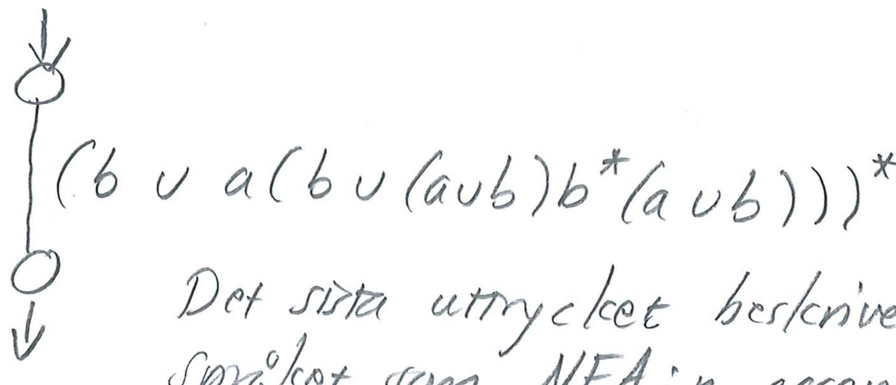
2. Nytt start- och accepterande tillstånd läggs till:



Sedan elimineras de gamla tillstånden, ett i taget, och förenklingar görs direkt:

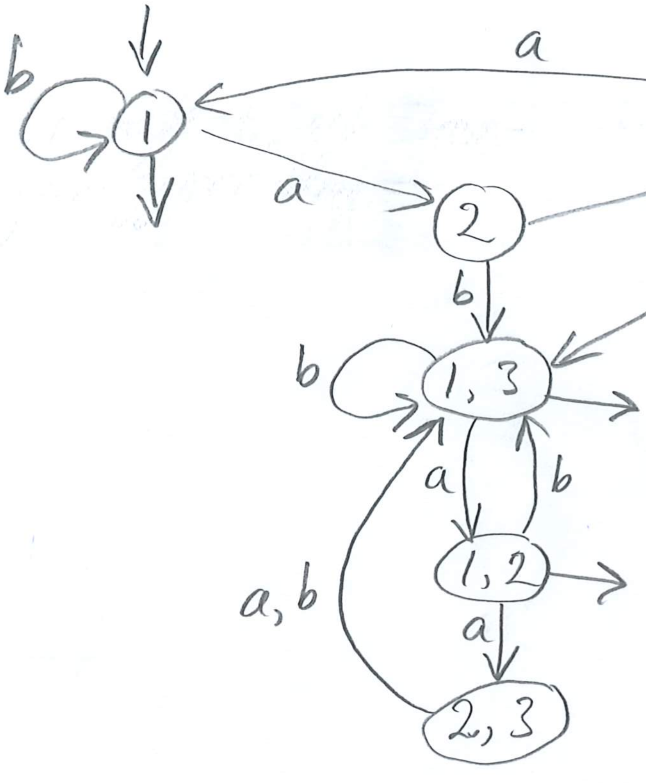




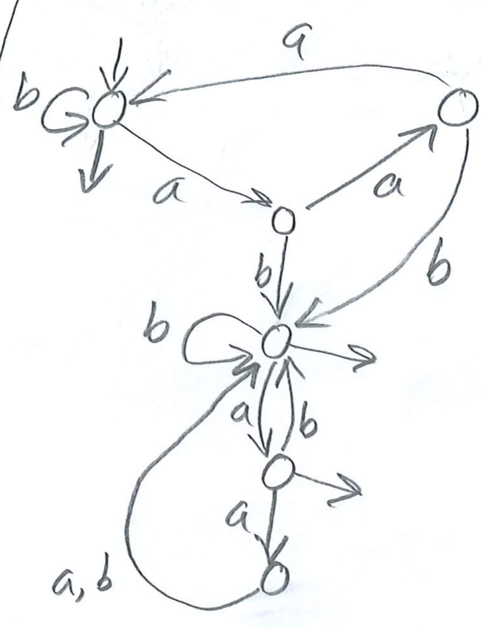


Det sista uttrycket beskriver språket som NFA:n accepterar.

3. Om tillstånden numreras 1, 2, 3 från vänster till höger så ger delmängdsalgoritmen följande DFA med samma språk som NFA:n. (NFA:n är redan icke-glupsk så delmängdsalgoritmen kan användas direkt.)



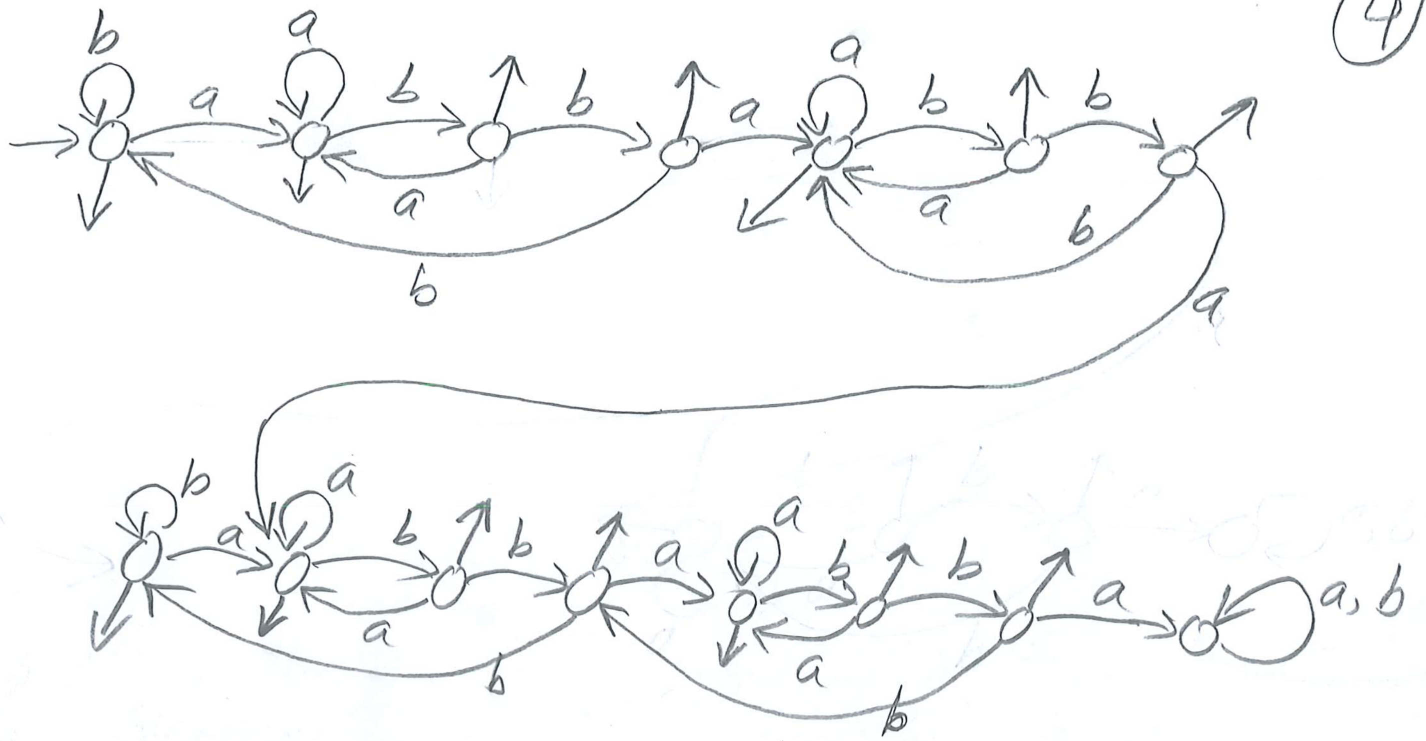
Tillsnyggat:



4. L_2 är reguljärt och accepteras av följande DFA:n på nästa sida.

Notera att tex. abbabbabba innehåller tre förekomster av abba eftersom olika förekomster kan överlappa varandra.

Bara om man kommit till det sista tillståndet så har 4 förekomster av abba registrerats så bara det sista tillståndet är icke-accepterande.



L_1 är <u>inte</u> reguljärt.

Bevis med pumpsatsen för reguljära språk.

1. L_1 är oändligt för tex.
   $(abba)^{n+1} a^{n+1} \in L_1$ för alla $n = 0, 1, 2, \ldots$
2. Antag att L_1 är reguljärt.
3. Låt N vara givet för L_1 av pumpsatsen.
4. Välj $u = (abba)^{N+1}$, $w = a^{N+1}$ och $v = \varepsilon$.
   Då har $uwv = (abba)^{N+1} a^{N+1}$ exakt $N+1$ förekomster av abba och exakt N förekomster av aaa (för aa^{N+1} har exakt N förekomster av aaa) så $uwv \in L_1$ och $|w| \geq N$.

5. Antag att $w = xyz$ och $y \neq \varepsilon$. Eftersom w består av endast a:n så $xy^2z = a^m$ där $m > N+1$ så aa^m innehåller minst $N+1$ förekomster av aaa och det följer att $uxy^2zv = (abba)^{N+1}a^m \notin L_1$.

6. Punkt 5 motsäger pumpsatsen för reg. språk så L_1 är inte regul-järt.

5. (a) Strängen aabbaa kan produceras:

$S \Rightarrow TP \Rightarrow aATbP \Rightarrow aAaATbbP \Rightarrow$
$\Rightarrow aAaAbbP \Rightarrow aAabAbP \Rightarrow$
$aAabbAP \Rightarrow aAabbPa \Rightarrow aaAbbPa$
$\Rightarrow aabAbPa \Rightarrow aabbAPa \Rightarrow$
$aabbPaa \Rightarrow aabbaa$.

Strängen aababa kan inte produceras och motivering ges i (b)-delen.

(b) De tre första reglerna kan producera strängar på formen $(aA)^n b^n P$
Med de två följande reglerna kan alla 'A' flyttas bakåt tills de möter 'P' och endast då kan 'A' omvandlas till en terminerande symbol 'a'. Det följer att alla 'A' måste flyttas till höger om alla 'b' för att kunna omvandlas till 'a'. Det följer att endast strängar på formen $a^n b^n a^n$ kan produceras, och alla sådana strängar kan produceras. Så grammatikens språk är

$$\{a^n b^n a^n : n \in \mathbb{N}\}.$$

6. (a) TM:en accepterar baababa.

#baababa
baababa (R_a kördes)
bababa# (S_L kördes)
#bababa ($L_\#$ kördes
bababa (R kördes)
ababa# (S_L kördes)

#ababa
ababa
baba#
#baba
baba
aba#

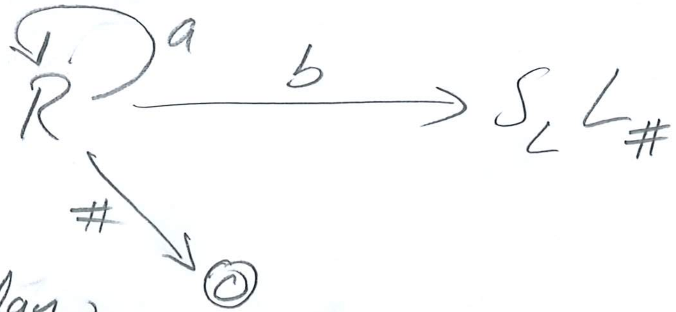
#aba
aba
ba#
#ba
ba
a#
#a

a
#
#
#
stannar

(b) TM accepterar språket

$\{w \in \{a,b\}^* : w \text{ har fler a:n än b:n}\}$.

Motivering: Delen $R_a S_L L_\#$ letar efter ett 'a' och raderar det med hjälp av S_L och läshuvudet placeras sedan till vänster om de icke-blanka kvarvarande tecknen. Delen



söker (sedan) efter ett tecken som inte är 'a'. Om 'b' påträffas så raderas det och läshuvudet placeras i början och den första delen körs igen, så minst ett till 'a' måste finnas om strängen ska accepteras. I annat fall stannar TM:en.

<u>Kortfattat</u>: w accepteras om w innehåller minst ett 'a' och för varje nytt 'b' måste ett till 'a' finnas (utöver de som redan har påträffats och raderats).

(c) En restriktionsfri (icke sammanhangsfri) grammatik som producerar språket som TM:en accepterar:

$S \to ASB \mid A$
$A \to AA$

Producerar strängar på formen $A^n B^m$ där $n > m$.

$AB \to BA$
$BA \to AB$

används för att flytta om symbolerna i vilken ordning som helst

$A \to a$
$B \to b$

skapar de terminerande symbolerna

Språket är faktiskt sammanhangsfritt (inses enklast med en PDA) men jag tycker att det är svårare att motivera varför en CFG för språket "gör rätt".

7. L_4 är reguljärt (och därmed en CFL) för $L_4 = \{a^i b^j : i, j \in \mathbb{N}\}$ som beskrivs av a^*b^*. Motivering: Det är tydligt att alla $w \in YX$ har formen $a^i b^j$. Antag nu att $w = a^i b^j$. Då gäller att $w = a^i a^0 b^j$ där

$a^i \in Y$ och $a^0 b^j \in X$ (eftersom vi kan låta $n=0$ och $m=j$), så $w \in YX$.

L_3 är en CFL men inte reguljärt.

Motivering: $L_3 = XX^{rev} =$

$\{a^n b^m b^j a^i : n \leq m$ och $j \geq i\}$ så en CFG för L_3 ges av reglerna

$S \to UV$

$U \to aUb \mid Ub \mid \varepsilon$ (producerar $a^n b^m$, $n \leq m$)

$V \to bVa \mid bV \mid \varepsilon$ (producerar $b^j a^i$, $j \geq i$)

Skiss med pumpsatsen av att L_3 inte är reguljärt. Eftersom man kan välja $i=j=0$ så följer att alla $a^n b^m$ där $n \leq m$ tillhör L_3. Om N är given av pumpsatsen så $a^N b^N \in L_3$. Välj (t.ex.) $u = a^N$, $w = b^N$, $v = \varepsilon$, så $uwv = a^N b^N$. Om $w = xyz$ och $y \neq \varepsilon$ så $uxy^0zv = uxzv = a^N b^k$ där $k < N$ så $uxzv \notin L_3$.

L_5 är <u>inte</u> en CFL och därmed <u>inte</u> reguljärt. (10)

Bevis med pumpsatsen för CFL.

1. $a^n b^n \in L_5$ för alla $n \in \mathbb{N}$ så L_5 är oändligt

2. Antag att L_5 är en CFL.

3. Låt k vara givet för L_5 av pumpsatsen.

4. Välj (t.ex.) $w = (a^k b^k)^2 = a^k b^k a^k b^k$, så $w \in L_5$ och $|w| \geq k$.

5. Antag att $w = uvxyz$, $|vxy| \leq k$ och $vy \neq \varepsilon$. Då är vxy en delsträng av första eller andra förekomsten av $a^k b^k$ eller av $b^k a^k$. Oavsett vilket som gäller så kommer uv^0xy^0z att

ha olika antal a:n i de två a-blocken

eller

ha olika antal b:n i de två b-blocken.

Oavsett fall så $uv^0xy^0z \notin L_5$.

6. Punkt 5 motsäger pumpsatsen för CFL så L_5 är <u>inte</u> en CFL.

8. (a) L_6 är <u>inte</u> avgörbart och jag visar det med Rices sats.

Låt $\Omega = \{L : L$ är TM-accepterbart och det finns $n \in \mathbb{N}$ så att $a^n b^n \in L\}$.

Per definition så innehåller Ω bara TM-accepterbara språk.

Exempelvis $\{\varepsilon\}$ är TM-accepterbart (för det är reguljärt) och innehåller $\varepsilon = a^0 b^0$ så Ω är icke-tom.

Men t.ex. språket $\{aba\}$ är TM-accepterbart (för det är reguljärt) och innehåller <u>inte</u> någon sträng på formen $a^n b^n$, så $\{aba\}$ tillhör <u>inte</u> Ω.

Enligt Rices sats följer att ingen TM existerar som kan avgöra för en godtycklig TM M om $L(M) \in \Omega$, dvs om $K_M \in L_6$.

(b) L_6 är TM-accepterbart. Enligt Church–Turings tes räcker det att beskriva informellt en algoritm som accepterar L_6.

ALGORITM

Input: koden K_M för en TM M.
Låt initialt $n = 0$.

(*) Kör M i $n+1$ steg på varje sträng på formen $a^m b^m$ där $m \leq n$.
Om M accepterar någon sträng $a^m b^m$ där $m \leq n$ inom $n+1$ steg så stannar algoritmen (och accepterar K_M).
Om inte så sätt $n := n+1$ (dvs n ökas med ett) och upprepa (*).

Denna algoritm stannar om M accepterar någon sträng på formen $a^n b^n$ och i annat fall stannar den inte.

9. (a) Falskt. Motexempel: $L_1 = \{a^n b^n : n \in \mathbb{N}\}$ och $L_2 = \overline{L_1}$. Då är inte L_1 reguljärt men $L_1 \cup L_2 = \{a,b\}^*$ är reguljärt och $L_1 \cap L_2 = \emptyset$ är reguljärt.

(b) Falskt. Motexempel: $L_1 = L_6$ (från uppg. 8) och $L_2 = \overline{L_1}$. Då är L_1 inte avgörbart men $L_1 \cup L_2 = \{0,1\}^*$ och $L_1 \cap L_2 = \emptyset$ är reguljära så de av avgörbara.