Salling, 070 – 6527523
TID : 9 – 14 HJÄLPMEDEL : Inga
BETYGSGRÄNSER : G –18p, VG–28p
SKRIV **TYDLIGT** OCH **MOTIVERA** NOGA !

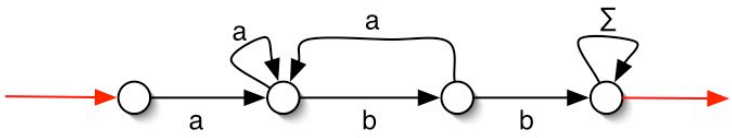
PROV I MATEMATIK
AUTOMATEORI & FORMELLA SPRÅK DV1, 4 p
20 MARS 2002

1. Språket *L* över alfabetet {*a*, *b*} definieras av att dess strängar börjar på *a* och innehåller minst en förekomst av *abb*.

a) Konstruera en DFA för *L*. (3p)
b) Konstruera en reguljär grammatik för *L*. (3p)

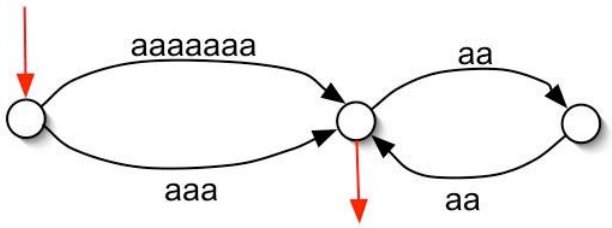
**LÖSNING**

a)




b) $S \to aA,\ A \to aA \mid bB,\ B \to aA \mid bC,\ C \to aC \mid bC \mid \varepsilon$

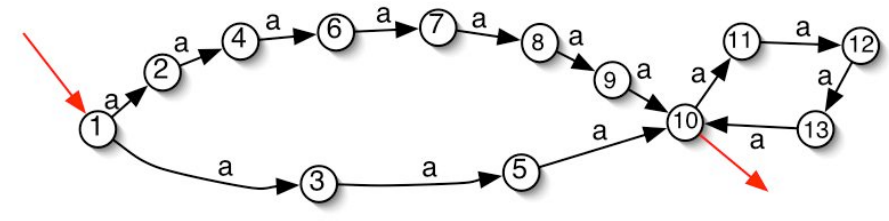
2. Låt *L* vara språket över {*a*} som nedanstående NFA accepterar.

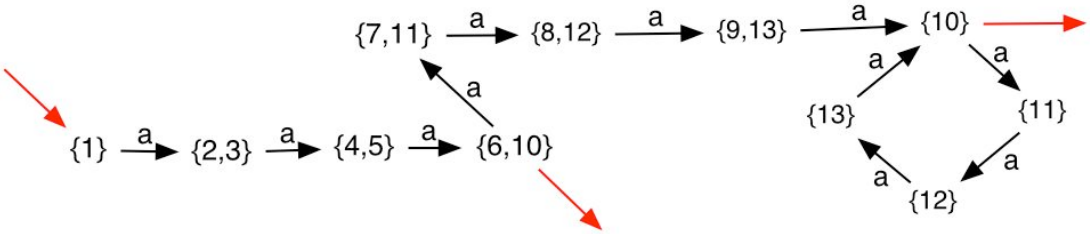


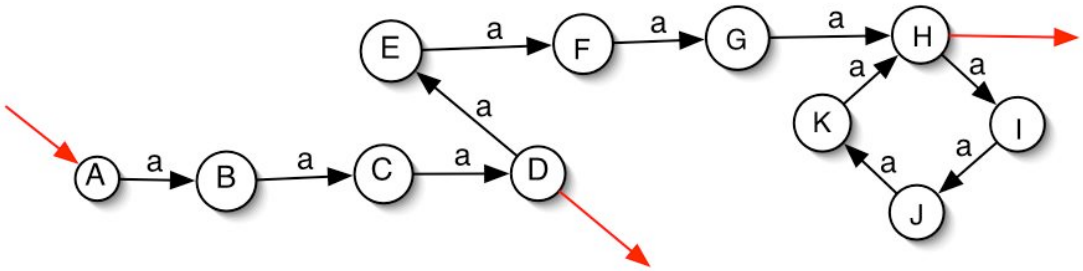

a) Konstruera med hjälp av delmängdskonstruktionen en DFA för *L.* (3p)
b) Konstruera en *minimal* DFA för *L*. (Minimaliteten måste motiveras.) (3p)

**LÖSNING**

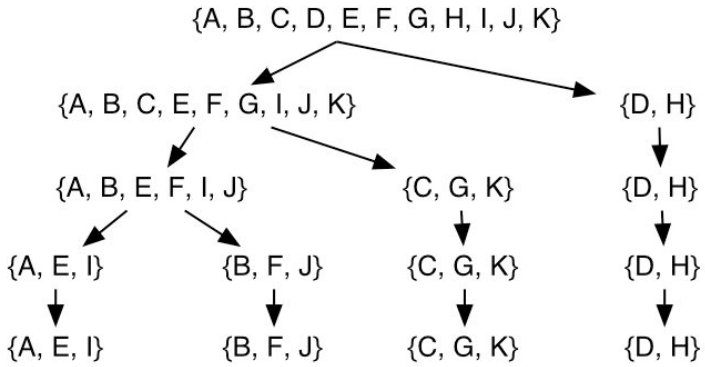
a)

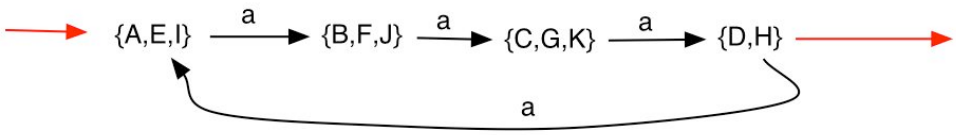





b) Man har en stark känsla av att ovanstående DFA *inte* är minimal. Följande stegvisa särskiljande visar mycket riktigt att så är fallet …



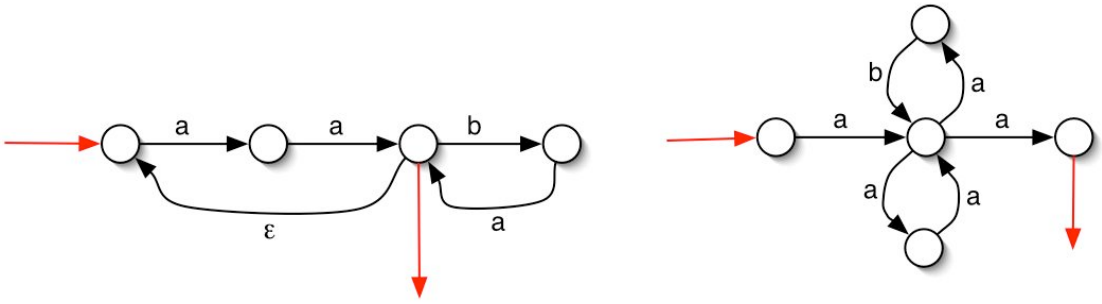




3. Beskriver nedanstående reguljära uttryck samma språk över {*a*, *b*}? (6p)

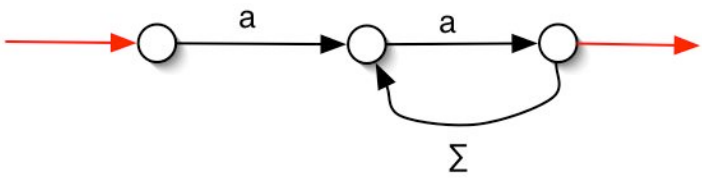
$(a\,a\,(b\,a)^*)^+$ $\qquad\qquad$ $a\,(a\,a \cup a\,b)^*\,a$

Motivera noga och beskriv dessutom en generell metod för jämförelse av reguljära uttryck.

**LÖSNING** Det torde vara uppenbart att de reguljära uttrycken beskriver samma sak som nedanstående automater

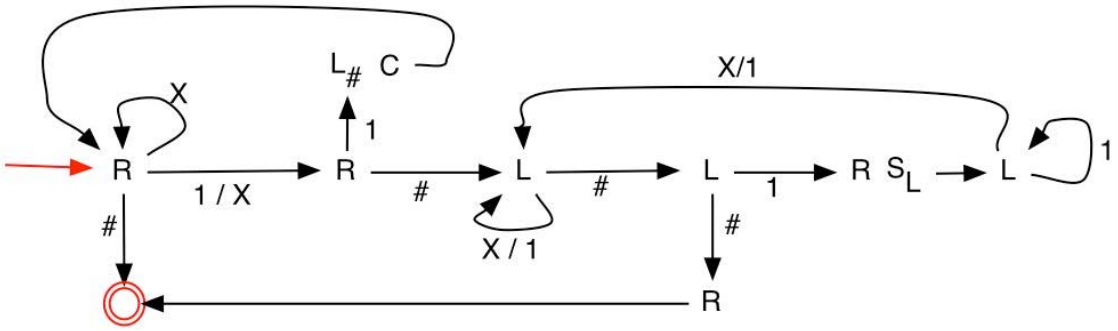


När man konstruerar de minimala DFA:erna hörande till de två automaterna (förslagsvis genom att först köra delmängdskonstruktionen på var och en av NFA:erna och sedan minimera de resulterande DFA:erna) får man en och samma DFA i bägge fallen, nämligen




*En* generell metod är den som illustreras ovanför, dvs att konstruera minimala DFA:er för vardera uttrycket (detta går ju att göra med kända algoritmer).Och sedan jämföra dessa DFA:er (vilket går att göra med någon form av ändlig fallundersökning även om DFA:erna är stora).
En *bättre* generell metod är dock att utnyttja att två mängder $L$ och $L'$ är lika omm symmetriska mängddifferensen $(L - L') \cup (L' - L)$, dvs $(\overline{L} \cap L') \cup (\overline{L'} \cap L)$ är tom.
Givet två reguljära uttryck för $L$ resp. $L'$, så behöver man således bara bilda den minimala DFA:n för $(\overline{L} \cap L') \cup (\overline{L'} \cap L)$, och sedan jämföra denna DFA med den minimala DFA:n för tomt språk. (Den senare har ett enda tillstånd som är ickeaccepterande.)

4. Turingmaskinen i figuren är en funktionsberäknande typ med inputalfabet {1} och tapealfabet {1, *X*, #}. Vilken funktion beräknar den? (6p)




ANM. Hjälpmaskinerna *R*, *L*, $L_{\#}$, $S_L$ och *C* är standardmaskinerna från kursboken. Tex söker $L_{\#}$ upp första blanktecknet till vänster, kopieringsmaskinen *C* förvandlar $\underset{\uparrow}{\sigma}\, w\,\#\,\ldots$ till $\sigma\underset{\uparrow}{w}\,\#\,w\,\#\,\ldots$ och vänstershiftaren $S_L$ gör om $\underset{\uparrow}{\sigma}\, w\,\#\,\ldots$ till $\underset{\uparrow}{w}\,\#\,\ldots$

**LÖSNING**

Maskinen beräknar $n^2$, för godtyckliga naturliga tal *n* representerade unärt.
MOTIVERING: Efter *ett* varv i (den stora) slingan från den inledande *R* -maskinen och tillbaka till densamma förändras tapekonfigurationen

$$\ldots \#\#\underset{\uparrow}{\#}\,1^n\,\#\#\#\,\ldots \qquad \text{till} \qquad \ldots \#\#\#X\,1^{n-1}\,\underset{\uparrow}{\#}X\,1^{n-1}\,\#\#\#\,\ldots.$$

Dvs efter ett varv har tapen delats in i två block som vart och ett består av *n* ickeblanka tecken. Efter två varv har tapen delats in i tre block:

$$\ldots \#\#\#X\,1^{n-1}\,\#X^2\,1^{n-2}\,\underset{\uparrow}{\#}X^2\,1^{n-2}\,\#\#\#\,\ldots.$$

Osv. …
Efter $n-1$ varv fås *n* block:

$$\ldots \#\#\#X\,1^{n-1}\,\#X^2\,1^{n-2}\,\#X^3\,1^{n-3}\,\#\,\ldots\,\#X^{n-1}\,1\,\underset{\uparrow}{\#}X^{n-1}\,1\,\#\#\#\,\ldots.$$

När i detta läge den inledande *R* -maskinen körs igång igen drivs den givna Turingmaskinen *inte* runt i nämnda slinga längre. Nej, nu drivs den istället "rakt fram" mot *L*-maskinerna. Strax innan första *L* startas ser tapen ut så här:

$$\ldots \#\#\#X\,1^{n-1}\,\#X^2\,1^{n-2}\,\#X^3\,1^{n-3}\,\#\,\ldots\,\#X^{n-1}\,1\,\#\,\underset{\uparrow}{X^n}\,\#\#\#\,\ldots$$

Denna *L*-maskin kommer efter $n+1$ varv i sin ögla att förändra tapen till

$$\ldots \#\#\#X\,1^{n-1}\,\#X^2\,1^{n-2}\,\#X^3\,1^{n-3}\,\#\,\ldots\,\#X^{n-1}\,1\,\underset{\uparrow}{\#}\,1^n\,\#\#\#\,\ldots.$$

Därefter står den andra *L*-maskinen i begrepp att dras igång och därmed kommer en ny (stor) slinga att gås igenom, närmare betstämt från den senast nämnda *L*-maskinen och tillbaka igen. För varje varv i slingan vänstershiftas blocket längst till höger ihop med blocket till vänster om detsamma, samtidigt som samtliga *X* i det senare blocket skrivs om till 1:or. Under tex *första* varvet förändras tapen till

$$\ldots \#\#\#X\,1^{n-1}\,\#X^2\,1^{n-2}\,\#X^3\,1^{n-3}\,\#\,\ldots\,\underset{\uparrow}{\#}\,1^n\,1^n\,\#\#\#\,\ldots$$

Efter $n-1$ varv har samtliga *n* block shiftats ihop till ett enda block som innehåller 1:or som enda ickeblanka tecken, närmare bestämt $n\cdot n$ stycken. Pekaren står nu till vänster om detta block när *L*-et står i begrepp att dras igång för sista gången.

$$\ldots \#\#\underset{\uparrow}{\#}\,1^n\;\ldots\;1^n\,\#\#\#\,\ldots$$

Nu bär det iväg till stopptillståndet, och resultatet av beräkningen är att *n* stycken 1:or blev till $n\cdot n$ stycken, dvs den funktion som beräknas är $n \mapsto n^2$.

5. Konstruera en alternativ men enklare ‡ grammatik än nedanstående dito. (De stora bokstäverna är icketerminaler, och de små är terminaler.) (6p)

$$S \to A\ B\ C$$
$$A \to a\,A \mid b\,A \mid \varepsilon$$
$$C \to C\,a \mid C\,b$$
$$B\,C \to a\,b\,b\,a$$

‡. Inom hierarkin reguljär ≺ sammanhangsfri ≺ restriktionsfri finns de enklaste grammatikerna till vänster. Din grammatik skall beskriva samma språk som den givna grammatiken.
Full poäng erhålls enbart om en enklaste grammatik presenteras.

**LÖSNING**

Med *A*-reglerna produceras *samtliga strängar* över {*a*, *b*}, och *C*-reglerna producerar på motsvarande sätt varje sträng av typen $C\,w$ där $w$ är en sträng av längd ≥1 över {*a*, *b*}.
Så *S*-regeln tillsammans med *A*- och *C*- reglerna beskriver strängar av typ

$$u\,B\,C\,w,$$

där $u$ är godtycklig över {*a*, *b*} och $w$ är godtycklig av längd≥1 över {*a*, *b*} .
Tillsammans med $B\,C \to a\,b\,b\,a$ - regeln beskrivs därvid varje sträng med minst en *abba*-förekomst som har en icketom sträng efter nämnda förekomst.

Om $B\,C \to a\,b\,b\,a$ - regeln tillämpas direkt i *S*-regeln fås $A\,a\,b\,b\,a$, som därmed (tack vare *A*-reglerna) kommer att beskriva samtliga strängar som slutar på *abba.*

Sammantaget beskriver den givna grammatiken således språket (över {*a*, *b*}) av strängar som har minst en *abba*-förekomst. Detta är ett reguljärt språk.
Här är en reguljär grammatik för språket: $S \to a\,S \mid b\,S \mid a\,b\,b\,a\,A$, $A \to \varepsilon \mid a\,A \mid b\,A$

6. Betrakta följande kategorier av algoritmiska språk:
$K_1$: reguljära
$K_2$: inte reguljära, men sammanhangsfria
$K_3$: inte sammanhangsfria, men Turingavgörbara
$K_4$: inte Turingavgörbara, men Turingaccepterbara

Din uppgift är att placera nedanstående språk (över alfabetet {0, 1, @} i rätt kategori. (10p)
a) $L_1 = \{ u\,@\,u \mid u \in \{0, 1\}^* \}$
b) $L_2 = \{ u\,@\,u \mid u \in \{0, 1\}^*$ och $|u| = 1000\}$
c) $L_3 = \{ u\,@\,w \mid u, w \in \{0, 1\}^*$ och $u$ förekommer som delsträng i $w\}$
d) $L_4 = \{ u\,@\,w \mid u, w \in \{0, 1\}^*$ och $u \neq w\}$
e) $L_5 = \{ u\,@\,w \mid u, w \in \{0, 1\}^*$, där $w$ är ett palindrom
och $u$ är en (binärkodning av en) TM som accepterar $w\}$

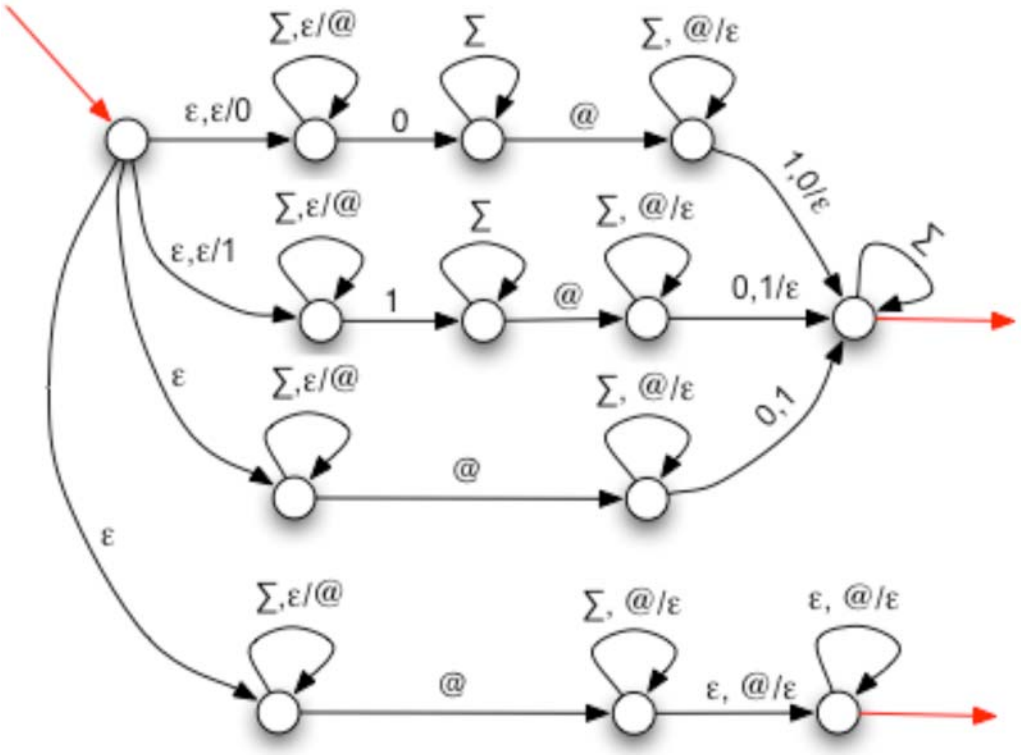
Om du menar att ett språk ligger i $K_1$ räcker det att presentera ett reguljärt uttryck eller en finit automat för språket ifråga. Om du menar att det ligger i $K_2$ måste du motivera bristen på reguljäritet och dessutom presentera en CFG eller en PDA för språket. Om du menar att det ligger i $K_3$ måste du motivera bristen på sammanhangsfrihet, och skissera en algoritm som kan avgöra språket. Om du menar att det ligger i $K_4$ måste du motivera bristen på Turingavgörbarhet, och skissera en algoritm (eller hänvisa till en känd dito) som accepterar språket.

LÖSNING

a), c) Dessa två språk ligger båda i $K_3$. MOTIVERING: Ej sammanhangsfria, ty $0^K\ 1^K \text{ @ } 0^K\ 1^K$ kan inte pumpas i något enda $K$-block utan att falla ur. (Pumpning till vänster om @ som involverar 0 (1) kan inte involvera pumpning av 0 (1) till höger om @ om pumpblocket inte är längre än $K$, och därmed driver upp-pumpning i sådant litet block ut strängen ur bägge språken.) Däremot är bägge språken Turingavgörbara. En avgörande TM i a) behöver bara kontrollera tecken för tecken ifall det står samma sak till vänster som till höger om @. Detta kan en TM göra genom att ila fram och tillbaka på tapen förbi @.I c) kan på motsvarande sätt en avgörande TM kontrollera tecken för tecken ifall det som står till vänster om @ även finns till höger om @.

b) Detta språk ligger i $K_1$. MOTIVERING: Språket är ändligt, säg att det är $\{w_1,\ w_2,\ \ldots,\ w_N\}$. Då är $w_1 \cup w_2 \cup \ldots \cup w_N$ ett reguljärt uttryck för språket.

d) Detta språk ligger i $K_2$. MOTIVERING: Ej reguljärt, ty komplementet är inte reguljärt. Det senare följer t ex av att $0^N \text{ @ } 0^N$ inte kan pumpas upp i det inledande $N$-blocket utan att falla ur språket. Däremot är språket sammanhangsfritt, ty nedanstående PDA accepterar språket. PDA:n ifråga är en parallellkoppling vars fyra komponenter i tur och ordning är specialiserade på följande typer av strängar:
(i) $u \text{ @ } w$, $u = x\,0\,y$ och $w = x'\,1\,z$, $|x| = |x'|$, (ii) $u \text{ @ } w$, $u = x\,1\,y$ och $w = x'\,0\,z$, $|x| = |x'|$,
(iii) $u \text{ @ } w$, $|u| < |w|$, (iv) $u \text{ @ } w$, $|u| > |w|$.
ANM. PDA:ns inputalfabet är $\{0,\ 1,\ @\}$ och dess tapealfabet är $\{@\}$. Tecknet $\sum$ i figuren avser teckenmängden $\{0,\ 1\}$.




e) Detta språk ligger i $K_4$. MOTIVERING: Ej Turingavgörbart, enligt Rices' sats. Ty om $T$ vore en TM som avgjorde språket, skulle $T$ (eftersom tom sträng är ett palindrom) bl. a. kunna avgöra för varje TM $u$, om $u \text{ @ } \varepsilon \in L_5$ dvs om $u$ accepterar tom sträng. M.a.o.skulle $T$ kunna avgöra om $L(u)$ tillhör mängden $\Omega$ av språk som innehåller tom sträng. Men detta motsägs av Rice's sats eftersom några men inte alla $L(u)$ innehåller tom sträng.
Å andra sidan är språket Turingaccepterbart. Den universella Turingmaskinen kan användas för acceptans.