UPPSALA UNIVERSITET
Matematiska institutionen
Salling (070-6527523)

PROV I MATEMATIK

Automatateori och formella språk DV1 4p
19 mars 2004

SKRIVTID: 15–20. POÄNGGRÄNSER: 18–27 G, 28–40 VG. **MOTIVERA** ALLA LÖSNINGAR NOGGRANT.

1. a) (3p) Konstruera ett reguljärt uttryck för språket över $\{a, b\}$ vars strängar innehåller ett jämnt antal förekomster av a *eller* ett jämnt antal förekomster av b. Tex gäller det att ε, b, a, bba, aba tillhör språket men inte ab.

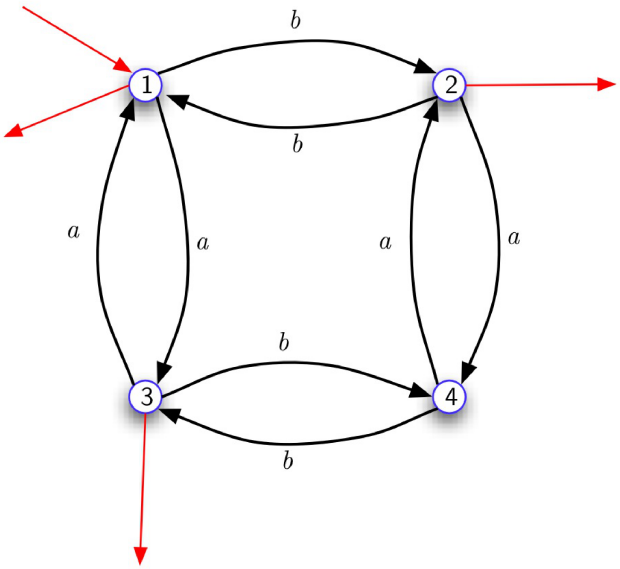
b) (3p) Konstruera en DFA för samma språk.

LÖSNING

a) $b^*(ab^*ab^*)^* \cup a^*(ba^*ba^*)^*$

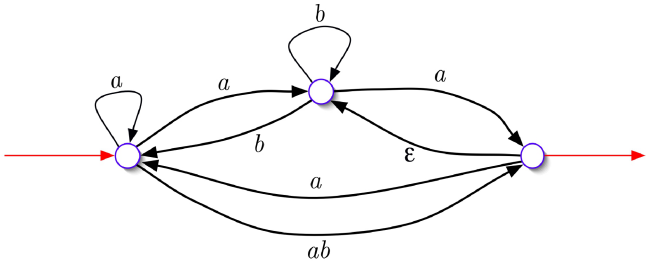
KOMMENTAR: $(aa)^*$ beskriver strängar över $\{a\}$ med ett jämnt antal a-förekomster. Genom att ″fylla i″ med b^* på strategiska platser får man ett uttryck $b^*(ab^*ab^*)^*$ som beskriver strängar över det större alfabetet $\{a, b\}$ med ett jämnt antal a-förekomster. På motsvarande sätt beskriver uttrycket $a^*(ba^*ba^*)^*$ strängar över $\{a, b\}$ med ett jämnt antal b-förekomster.

b)



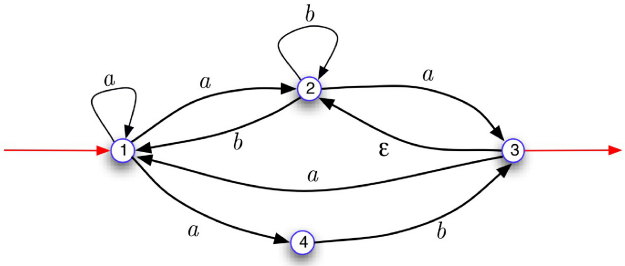
KOMMENTAR: Tillståndet 1 representerar ″jämnt antal a-förekomster och jämnt antal b-förekomster″, tillståndet 2 representerar ″jämnt antal a-förekomster och udda antal b-förekomster″, 3 representerar ″udda antal a-förekomster och jämnt antal b-förekomster″ medan 4 representerar ″udda antal a-förekomster och udda antal b-förekomster″.

2. a) (4p) Transformera nedanstående NFA till en DFA med hjälp av delmängdskonstruktionen.
   b) (3p) Minimera den resulterande DFA:n med hjälp av stegvisa särskiljandealgoritmen.

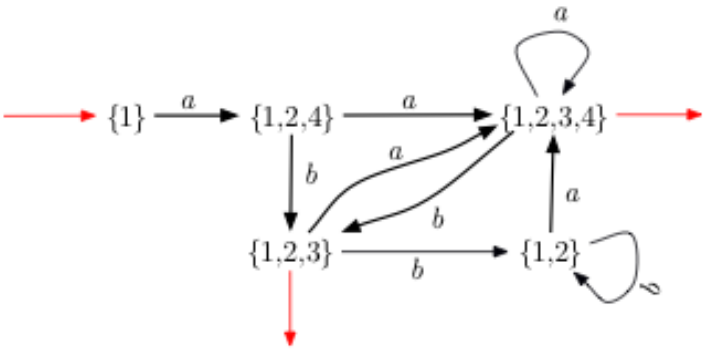


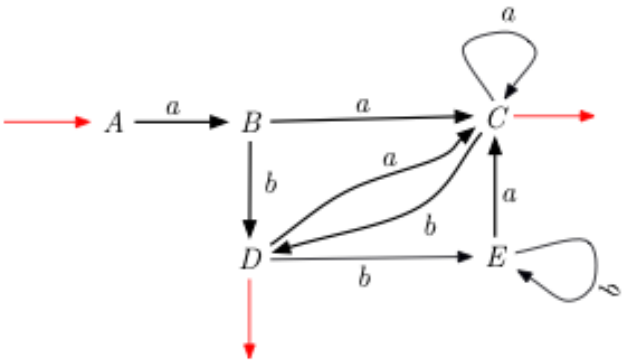
LÖSNING

a) Delmängdskonstruktionens första steg är att eliminera glupskheten . . .

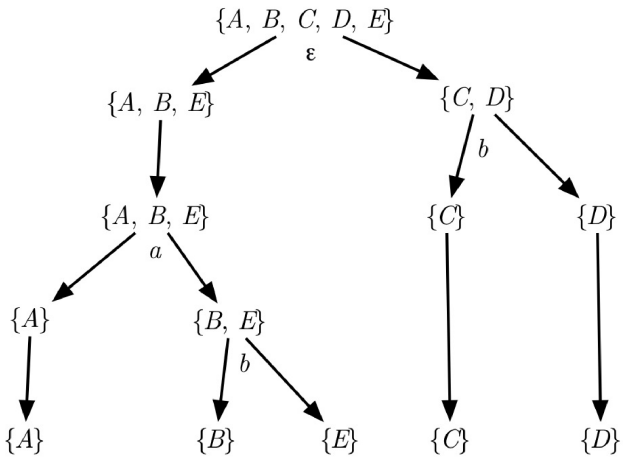


Andra steget är att bilda delmängder av tillstånd . . .





b) DFA:n från delmängdskonstruktionen ovanför är redan minimal.
Beviset av detta är att minimeringsalgoritmen särskiljer samtliga fem tillstånd. Se nedanför.



3. (6p) Betrakta språket L av strängar xyz över $\{a, b\}$ sådana att $|x| = |y| = |z|$, och så att det sista tecknet i y såväl som i z är a. T.ex. gäller det att baa och $abbababba$ tillhör L men inte $aaabaaaab$. Konstruera

a) en CFG för L,

b) en PDA för L som är bottom-up-parser för din grammatik i a).

c) en PDA för L som **inte** är en top-down- eller bottom-up-parser.

LÖSNING

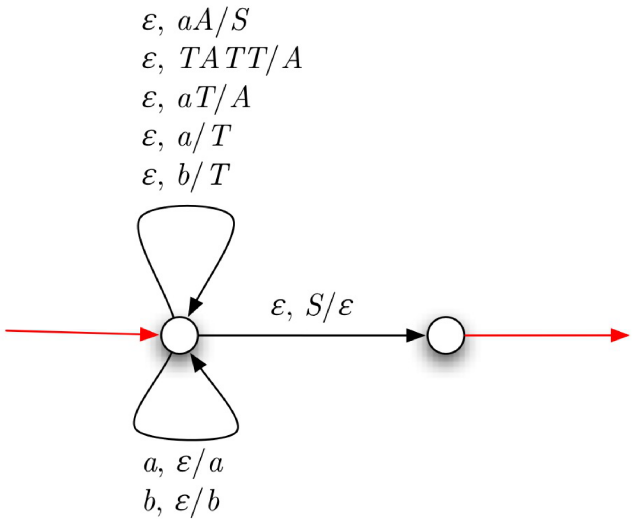
Σ a a,

$\Sigma\ \Sigma\ \ \Sigma$ a $\ \Sigma$ a,

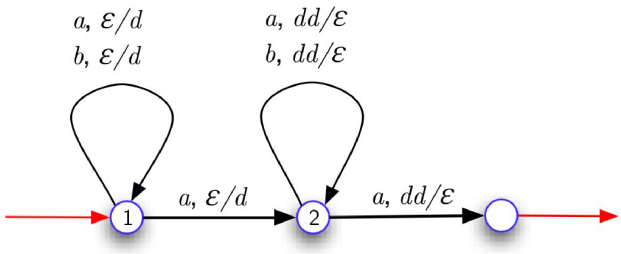
$\Sigma\ \Sigma\ \Sigma\ \ \Sigma\ \Sigma$ a $\ \Sigma\ \Sigma$ a

a) $S \to A\,a, \quad A \to T\,T\,A\,T \mid T\,a, \quad T \to a \mid b$

b)



c)



KOMMENTAR: Stacken blir tom i det accepterande tillståndet omm man har gjort dubbelt så många pushningar i 1:s ögla och på väg mot 2 som man har gjort poppningar i 2:s ögla och på väg mot det accepterande tillståndet Detta i kombination med att a konsumeras vid den sista pusningen såväl som vid den sista poppningen bevisar att PDA:ns språk är det som beskrivs i uppgiften.

4. (8p) Sant eller falskt?
   a) $\{0^m\,1^n\,0^m \mid m,\ n \in \mathbb{N}\}$ accepteras av någon DFA
   b) $\{0^m\,0^m\,1^n \mid m,\ n \in \mathbb{N}\}$ accepteras av någon DFA
   c) $\{0^n\,1^n\,1^n \mid n \in \mathbb{N}\}$ accepteras av någon PDA
   d) Komplementspråket till $\{\langle M\rangle \mid M$ är en TM som stannar för tom sträng$\}$ accepteras av någon TM.

ANM. Med $\langle M\rangle$ avses en kodning av M i form av en binär sträng.

LÖSNING

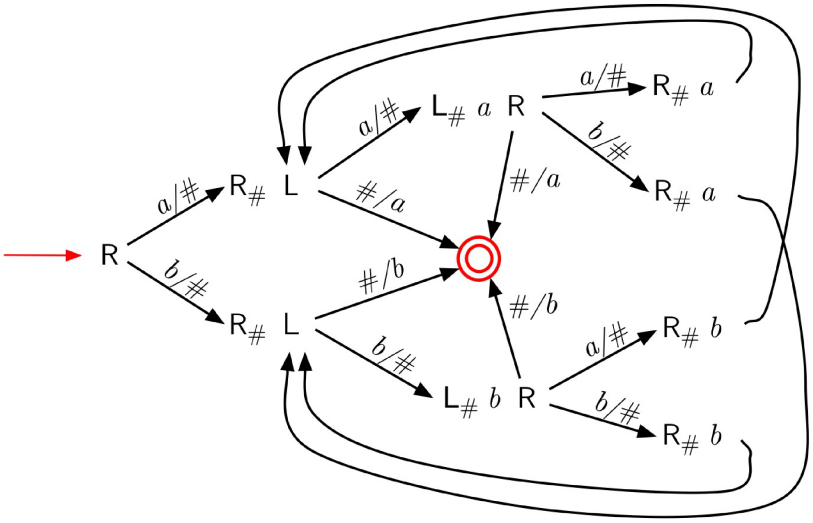
a) Falskt. Ty antag att en DFA med N tillstånd accepterar språket ifråga. Då skulle $0^N\,1\times 0^N$ driva denna DFA att besöka något tillstånd två gånger redan under det inledande N-blocket. Dvs någon (icketom) delsträng, säg 1^n inuti det inledande N-blocket skulle driva DFA:n att gå runt i en slinga. Därmed skulle $0^{N+n}\,1\times 0^N$ driva DFA:n att gå runt en extra gång i slingan för att sedan fortsätta ″som vanligt″ till acceptans. Men $0^{N+n}\,1\times 0^N$ tillhör ju inte det givna språket. Således finns ingen DFA för språket.

b) Sant. Ty språket är reguljärt. Det ges tex av (00)*1*.

c) Sant. Ty språket är sammanhangsfritt. Beskrivs enklast av $S \to \varepsilon \,|\, 0\,S\,11$.

d) Falskt. Följer av den kända satsen att ett språk L är TM-avgörbart omm L och L:s komplement är TM-accepterbara. Ty låt L vara språket $\{\langle M\rangle \mid M$ är en TM som stannar för tom sträng$\}$. Detta L är TM-accepterbart Det accepteras nämligen av en modifierad universell TM som vid körning på $\langle M\rangle$ kör M på tom sträng. Om nu L:s komplement vore TM-accepterbart skulle (enligt den nämnda satsen) L tvingas vara TM-avgörbart, Men det är det **inte**. Det senare följer av Rices sats eftersom ″M stannar för tom sträng″ är liktydigt med ″M:s språk innehåller tom sträng″ och egenskapen att innehålla tom sträng självklart är en icketrivial språkegenskap. Således kan L:s komplement inte vara TM-accepterbart.

5. a) (3p) Beskriv det språk som följande Turingmaskin accepterar. Du måste förklara hur de olika delarna i maskinen medverkar till att språket blir just det som du föreslår.
   b) (2p) Presentera en grammatik för språket.



LÖSNING

a) Språket består av alla icketomma palindromsträngar över $\{a, b\}$.
MOTIVERING: Övre (undre) delen är specialiserad på att kontrollera om första och sista tecknet är ett a (b), samt att undersöka nästkommande tecken (det som kommer efter det första tecknet). Beroende på om nästkommande tecken är ett a eller ett b körs övre eller undre delen igen. Och nu kontrolleras om andra och näst sista tecknet överensstämmer. På detta sätt fortskrider körningen ... Ifall inputsträngen är en palindromsträng av udda längd drivs TM:en till stopptillståndet via en av de främre övergångarna som pekar på nämnda tillstånd, och i fall palindromsträngen har jämn längd används en av de bakre övergångarna. Varje sträng som **inte** är palindromisk driver TM:en till hängning (efter submaskinen L).

6. (8p) Diskutera för vart och ett av följande problem om någon TM kan avgöra problemet ifråga.
a) Givet två godtyckliga Turingmaskiner, accepterar de någon gemensam icketom sträng?
b) Givet två godtyckliga DFA:er, accepterar de någon gemensam icketom sträng?
c) Givet en godtycklig Turingmaskin och en godtycklig sträng, har Turingmaskinen något tillstånd som besöks två gånger vid körning på strängen?

ANM Om Du menar att någon TM kan avgöra ett problem, måste Du motivera denna åsikt genom att skissera åtminstone en informellt beskriven algoritm. (Om Du menar att ingen TM kan avgöra ett problem, måste Du givetvis motivera denna åsikt också.)

LÖSNING

a) Ej TM-avgörbart. Ty om en TM kunde avgöra det givna problemet för två godtyckliga Turingmaskiner skulle den kunna avgöra problemet i det fall att de två är *identiska*, dvs då skulle problemet ″accepterar M någon icketom sträng″ vara TM-avgörbart. Men det är det inte. Det senare följer av Rices sats, med Ω vald som mängden av TM-accepterbara språk som innehåller någon icketom sträng. (Denna Ω är ju icketrivial, eller hur!)

b) TM-avgörbart. Här följer en skiss av en avgörande algoritm:
1. Konstruera en DFA M för snittet mellan de två DFA:ernas språk och språket $\sum^+$.
2. Minimera M.
3. Om M:s språk är tomt (dvs om M har ett enda tillstånd och om detta tillstånd är icke-accepterande) så accepterar de givna två DFA:erna ingen gemensam icketom sträng. Annars så gör de det.

c) TM-avgörbart.
MOTIVERING: För varje Turingmaskin M och varje inputsträng w till M kan problemet avgöras genom att låta en universell TM köra M på w under (högst) *lika många* övergångar (säg N st) som M har tillstånd. Om M under dessa N övergångar **inte** drivs till stopptillståndet kommer M garanterat att tvingas återbesöka något tillstånd (eftersom N tillstånd är otillräckligt om idel nya tillstånd skall besökas under N övergångar). Notera också att om M hänger sig menar vi att M har fastnat i ett skräptillstånd (som i förekommande fall besöks inte bara två gånger utan oändligt många gånger). Om å andra sidan M drivs till stopptillståndet under denna körning, behöver man bara genom en enkel bokföring kontrollera ifall något tillstånd har återbesökts innan stopptillståndet. Sammanfattningsvis: Efter N övergångar vet vi svaret på frågan om något tillstånd besöks två gånger.