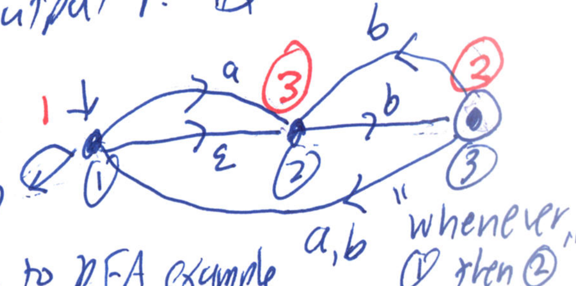
CSE 396 Lecture Tue 4/26 Spr 2019

Top Hat #3523

Theorem: Given any DFA/NFA/GNFA N, we can build a regexp r such that $L(r) = L(N)$.

Text proof: Add a new start state s' and one final state f'. s' has (s', ϵ, s) and for all $q \in F$, add (q, ϵ, f'). Make $F' = \{f'\}$. Then all q in the original Q are nonaccepting and different from s'. So we can eliminate them one by one per end of the previous lecture.

Final result is $s' \xrightarrow{r} f'$. Output r. $\boxtimes$

$N =$ 

Algorithm "Code Style": Maintain a table "T" such that $T(p,q)$ grows knowledge of which strings can be processed from p to q.

Tong

	1	2	3
1	b	$\emptyset$	$\epsilon + a$
2	~~$\emptyset$~~ $a+b$	$\emptyset$	b
3	$\emptyset$	b	$\emptyset$

note: $(b+\epsilon)^* = b^*$, $\emptyset^* = \epsilon$.

(optional) Number the states to be eliminated as 3 ... n. Loop:

Works if $F \subseteq \{1,2\}$, else use text.

NFA to DFA example

$S = \{1, 2\}$

δ	a	b
1	$\{2\}$	$\{1,2\}$
2	$\emptyset$	$\{3\}$
3	$\{1,2\}$	$\{2,1\}$

For all reached states $P \subseteq Q$, and $c \in \Sigma$,

$$\Delta(P, c) = \bigcup_{p \in P} \delta(p, c).$$

```
for (k = n downto 3) { //elim state k
  for (i = 1 to k-1) { // incoming from i<k
    for (j = 1 to k-1) { // outgoing to j
      T(i,j)new = T(i,j)old + T(i,k) T(k,k)* T(k,j).
    }
  }
}
```

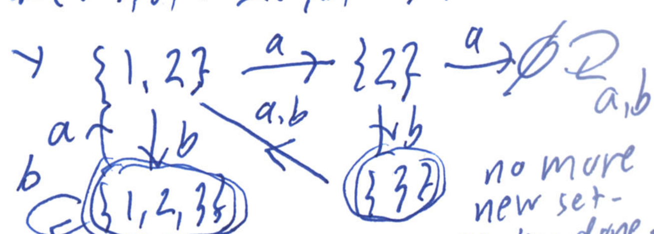
read off from 2-state formula for final $L(N)$.

$\Delta(\{1,2\}, a) = \delta(1,a) \cup \delta(2,a)$
$= \{2\} \cup \emptyset = \{2\}$ new

$\Delta(\{1,2\}, b) = \{1,2\} \cup \{3\} = \{1,2,3\}$ new

$\Delta(\{2\}, a) = \delta(2,a) = \emptyset$ new state.

$\Delta(\{2\}, b) = \delta(2,b) = \{3\}$ also new.

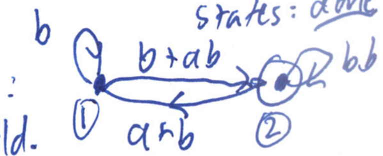


no more new set-states: done.

Exec $T(1,2)_{new} = T(1,2)\,old + T(1,3)\,T(3,3)^*\,T(3,2)$
$= \emptyset + (\epsilon + a) \cdot \epsilon \cdot b = \underline{b + ab}$

$T(2,2)\,new = T(2,2)\,old + T(2,3)\,T(3,3)^*\,T(3,2)$
$= \emptyset + b \cdot \epsilon \cdot b = bb$

$j = 1$ gives $T(i,1)\,new = T(i,1)\,old + T(i,3)\,T(3,3)^*\,T(3,1)$ — $\emptyset$ killer. $= T(i,1)_{old}$.



New Table

	1	2	3
T_{new} = 1	b	$b+ab$	
2	$a+b$	bb	
3			

elim

$L(N) = L_{12}$

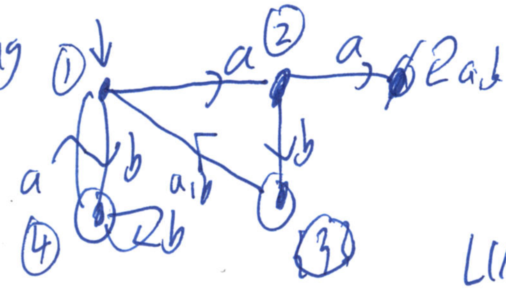
$= [T(1,1) + T(1,2)\, T(2,2)^*\, T(2,1)]^*\, T(1,2)\, T(2,2)^*$

$(\alpha + \beta\ \gamma^*\ \eta)^* \cdot \beta\ \gamma^*$

$= (b + (b+ab)(bb)^*(a+b))^* \cdot (b+ab)(bb)^*$

Redrawing DFA

M:



"Sightreading" the DFA gives first $L_{11} = (bb^*a + ab(a+b))^*$. Then

$L(M) = L_{14} \cup L_{13} = \underline{L_{11} \cdot bb^* \cup L_{11} \cdot ab}$

$= L_{11} \cdot (bb^* + ab)$

Main Utility of Having a DFA:

For any string $x \in \Sigma^*$, define $\delta^*(s,x)$ = the unique state q such that M processes x from s to q.
(Define $\delta^*(p,x)$ for any $p \in Q$ likewise)

Key Insight 1: If $\delta^*(s,x) = \delta^*(s,y)$, then for any $z \in \Sigma^*$, M must give the same accept/reject answer to xz that it gives to yz.

eg for above M, $q = s$, $x = bba$, $y = aba$.

Why? Suppose $q = \delta^*(s,x) = \delta^*(s,y)$

Then $\delta^*(s,xz) = \delta^*(q,z)$ and $\delta^*(s,yz) = \delta^*(q,z)$ too. Whatever state $r = \delta^*(q,z)$ is, r cannot be simultaneously an accept state and a reject state.

∴ Key Insight 2: If x and y are such that for some $z \in \Sigma^*$ xz and yz have different status with regard to a language L, then any DFA M such that $L(M) = L$ must process x and y to different states.

consider $z = \underline{abbb}$ $xz = \underline{bbaabbb}$ $yz = \underline{abaabbb}$ Both in $L(N)$: OK

And OK to process x, y to the same state iff $(\forall z \in \Sigma^*)\ \underline{xz \in L \iff yz \in L}$.

Hence if $\underline{k}$ strings $x, y, ..,$ have z's that give different statuses, M must have k states.