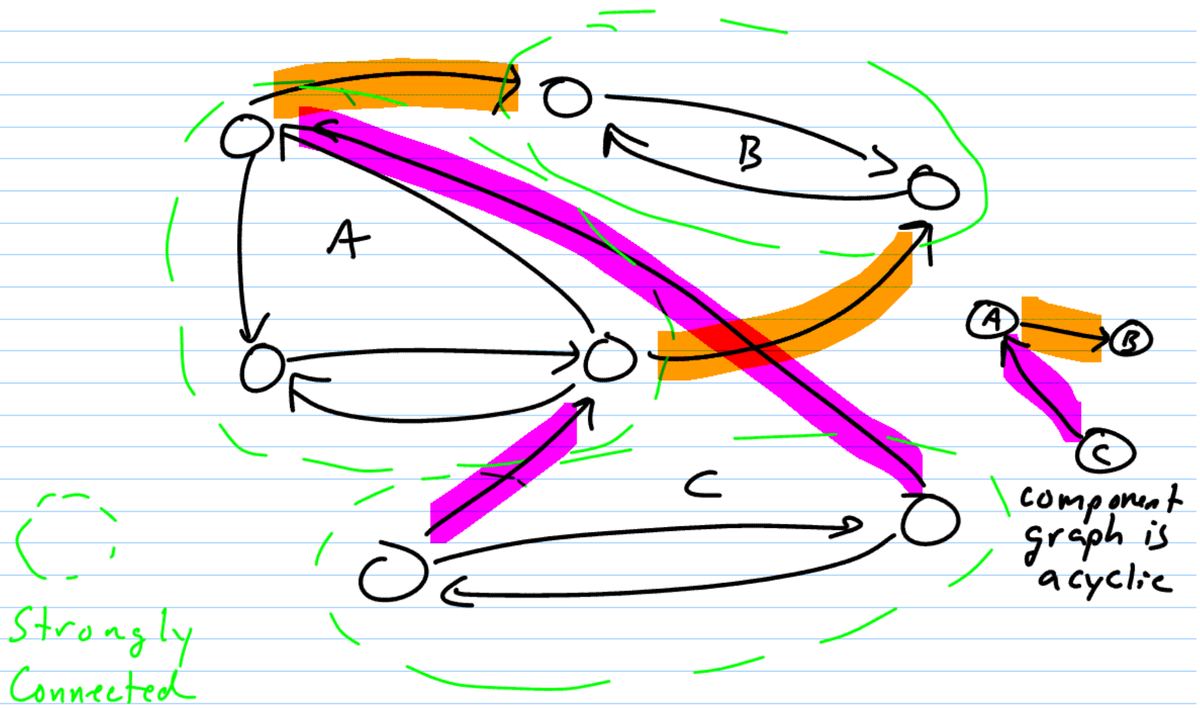
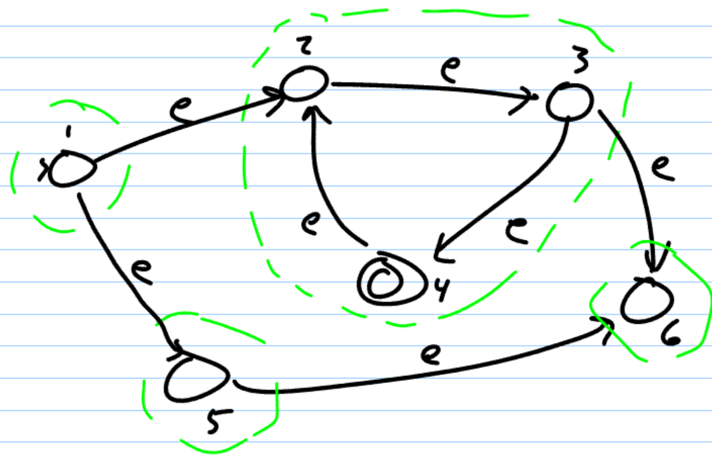
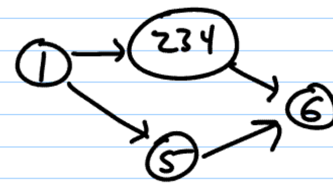


Strongly Connected Components – maximal set of vertices s.t. there is a path between each pair of vertices in the set





e-closure (2) = {2, 3, 4, 6}
e- closure (3) = {2, 3, 4, 6}
e- closure (4) = {2, 3, 4, 6}

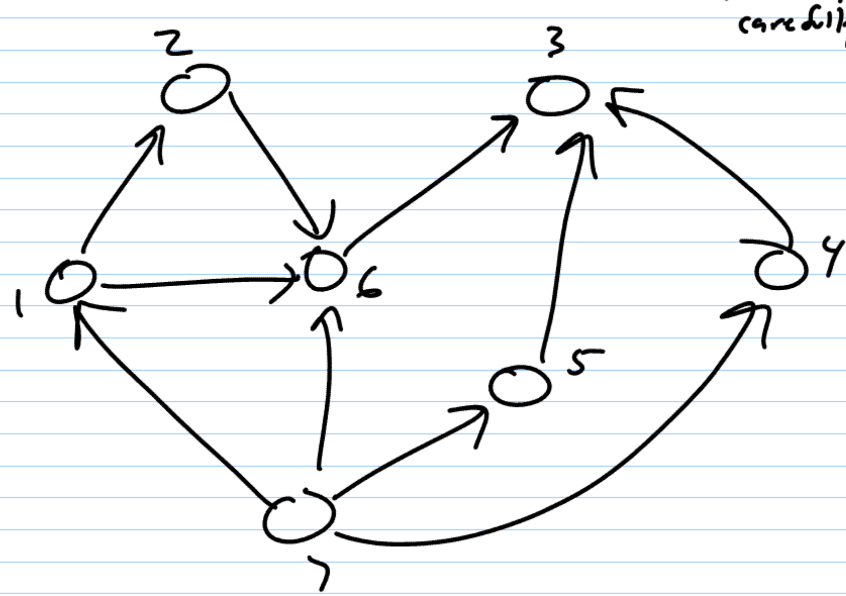
All states in SCC have same e-closure

1) find SCCs

2) work in order of topological sort $O(|K|^3)$
e-closure(V) = V ∪ ⋃ e-closure(u) oof!

$(v,u) \in$ SCC graph

can we do better?
(or analyze more carefully?)



7 1 2 4 5 6 3

topological sort: ordering so all edges go in one direction

WS1S : weak second-order theory of one successor

We can use FA to, given sentence in language of WS1S, determine if it is T.

language allows: natural numbers (and vars)

finite sets of natural numbers (+vars)

$\wedge, \vee, \sim$

$\in, =, +1$

$\forall, \exists$

There is a finite set that is a subset of every other finite set.

$$\exists X \forall Y \forall z \quad z \in X \rightarrow z \in Y \qquad T$$

There is a finite set such that for every element in that set, the next element is in as well.

$$\exists X\ \forall y\ \ y \in X \rightarrow y+1 \in X \qquad T$$

To decide WS1S using FA:

Work inside out constructing FA for subformulas

To build FA that, given y, X, determines if $y \in X$

need to encode y, X

$\hookrightarrow$ use characteristic fxn

if $y = 0$ use e (or $0\cdots$) $\{1, 2, 5\} \rightarrow 01100100..$

otherwise use $\underbrace{0 \ldots 0}_{y-1\ 0s}1$

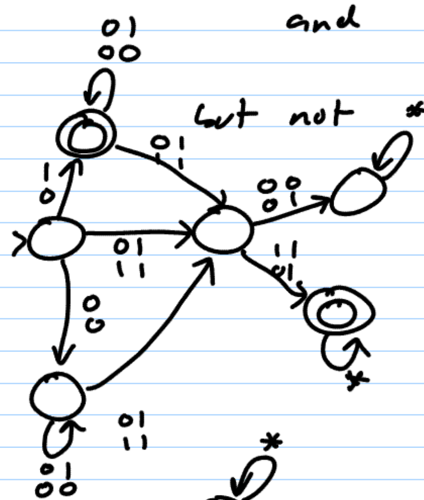
use symbols $\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{matrix}$ $\begin{matrix} \leftarrow X \\ \leftarrow y \end{matrix}$

$1 \in \{1,2,5\}$ so M should accept $\begin{matrix}011001\\100000\end{matrix}$

and $\begin{matrix}011001\\000010\end{matrix}$

but not $\begin{matrix}011001\\001000\end{matrix}$

$y \in X$



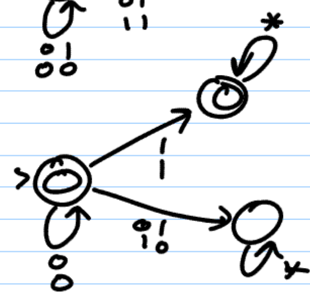
$X = \{0,2,7\}$

$y = 2$

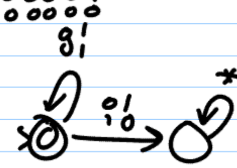
$\begin{matrix}001000001\\010000000\end{matrix}$

$y=0$ $\begin{matrix}101000001\\000000000\end{matrix}$

$x=y$



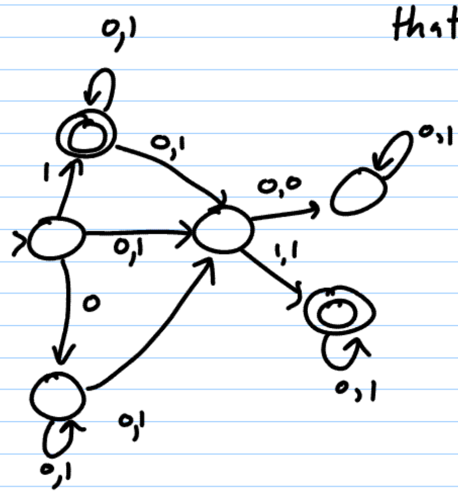
$X=Y$



handle $\wedge, \vee, \sim$ as for normal DFAs

$\exists x$: erase x part from every input symbol (and fix final states by making final all states that can reach a final state on 0 input)

$\exists y \;\; y \in X$



This should (and does!) accept the representation of any set X that makes $\exists y \; y \in X$ true (non-empty sets) — such representations have at least one 1.

$\forall x \qquad \sim \exists x \sim$

In end we get  F

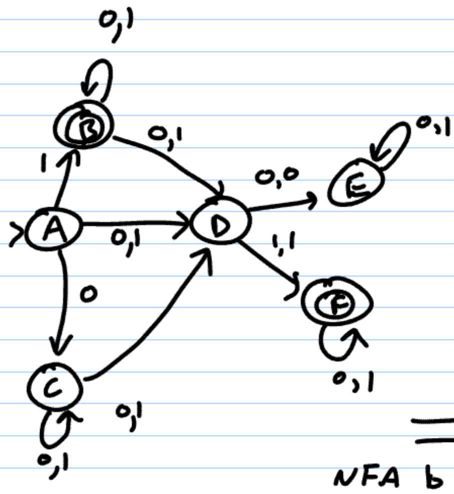
or  T

$\forall X \, \exists y \; y \in X$: treat as

$$\sim \exists X \sim \exists y \, (y \in X)$$

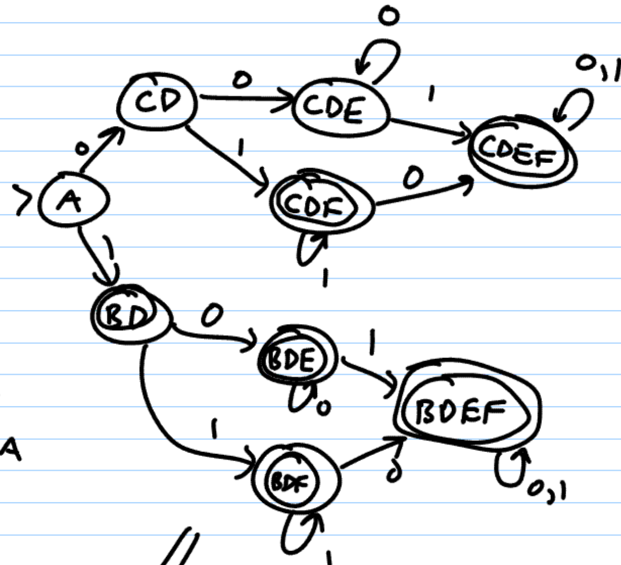
have M from above for this

need to convert to DFA to negate



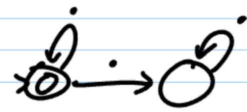
$\exists y \; y \in X$

NFA to DFA



minimize

negate :  this machine corresponds to $\sim \exists y \ y \in X$

erase X : 
(for $\exists X$)

NFA → DFA 

negate  so $\sim \exists X \sim \exists y \ y \in X$ is FALSE

$$\sim \exists X \sim \exists y \ y \in X \equiv \forall X \exists y \ y \in X$$

$$\exists X \forall y , y \in X$$

$$\equiv$$

$$\exists X \sim \exists y \sim y \in X$$