

Design NDFFA's which accept:

$L_1 = \{ w \in \{a, b\}^* : w \text{ contains one or more of } aabb, abab, \text{ or } abba \text{ as substrings} \}$

$L_2 = \{ w \in \{0, 1\}^* : w = uv \text{ for some } u \text{ such that the length of } u \text{ is even and for some } v \text{ which contains } 3k+1 \text{ 0's} \}$

$L_3 = a(ab)^*a \cup ab(a \cup b)^*a$

Announcements

Assignment #2 is posted: Due Fri. Oct. 7.

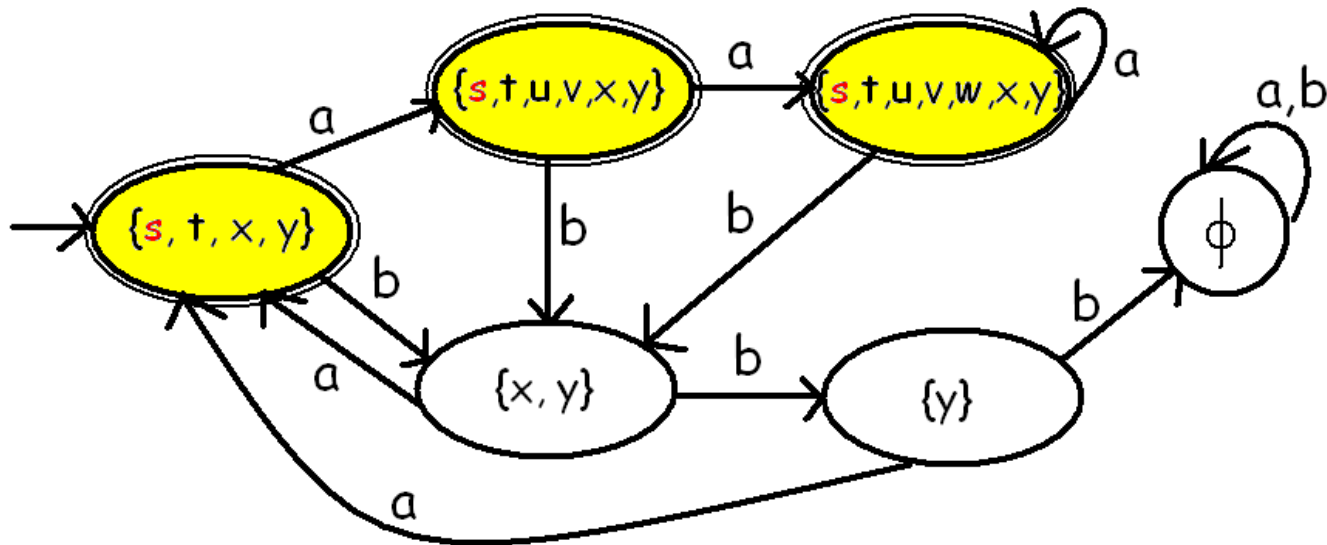
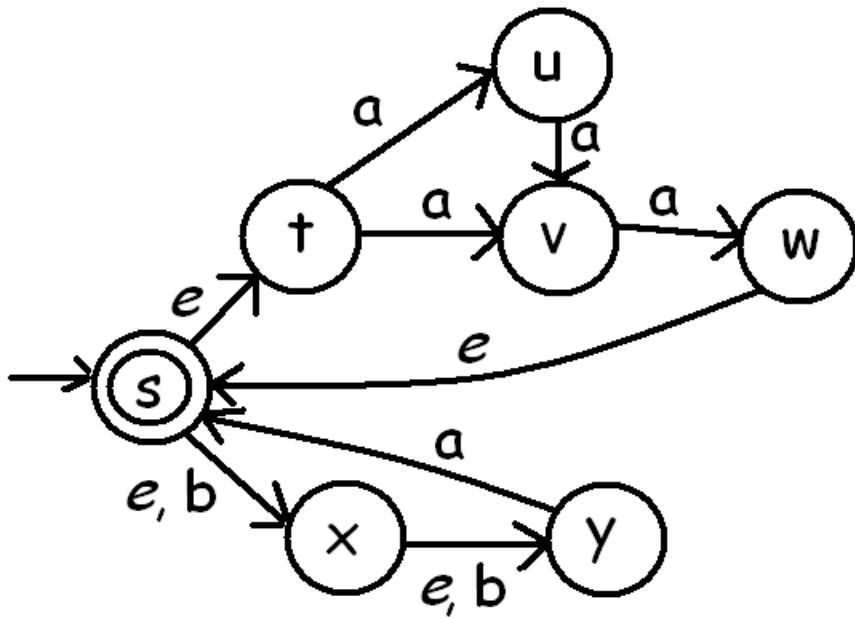
Tutorial #3: print and take to lab.

Have you tried the regular expression/DFA tutorials?

CSC 320

Lecture 9:

Converting NDFA's to DFA's



Two machines M_1 and M_2 are **equivalent** if $L(M_1) = L(M_2)$.

Theorem: For any NDFA, there exists an equivalent DFA.

Proof: By construction.

Proofs by construction are nice because they don't just tell us that an object exists- they also give us an algorithm for constructing the object.

$E(q) = \{ p: p \text{ is reachable from } q \text{ by following only transitions on } \varepsilon \}.$

Note: q is always in $E(q)$.

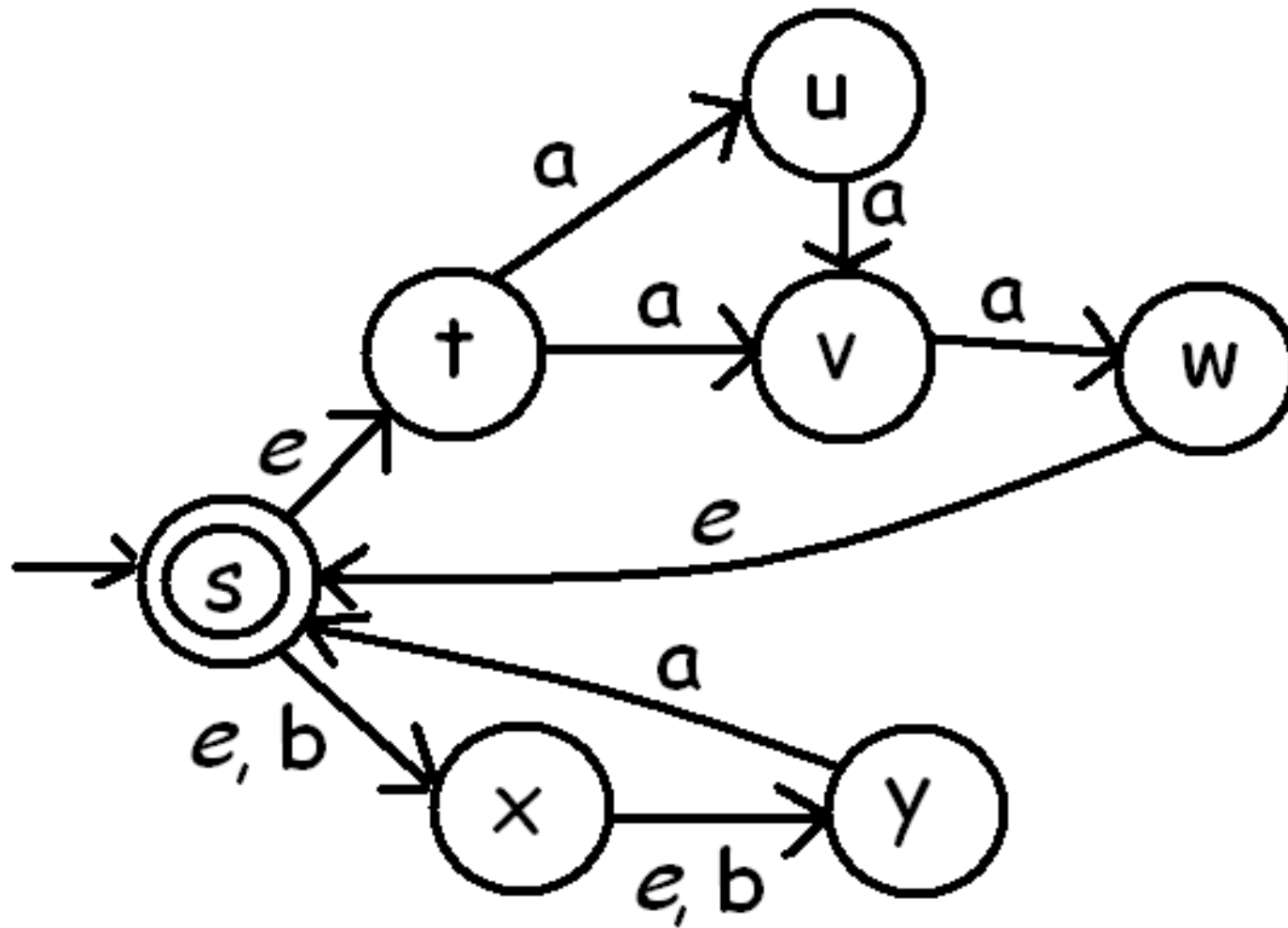
For a set S of states, $E(S) = \bigcup_{q \in S} E(q)$

Transition function for new DFA:

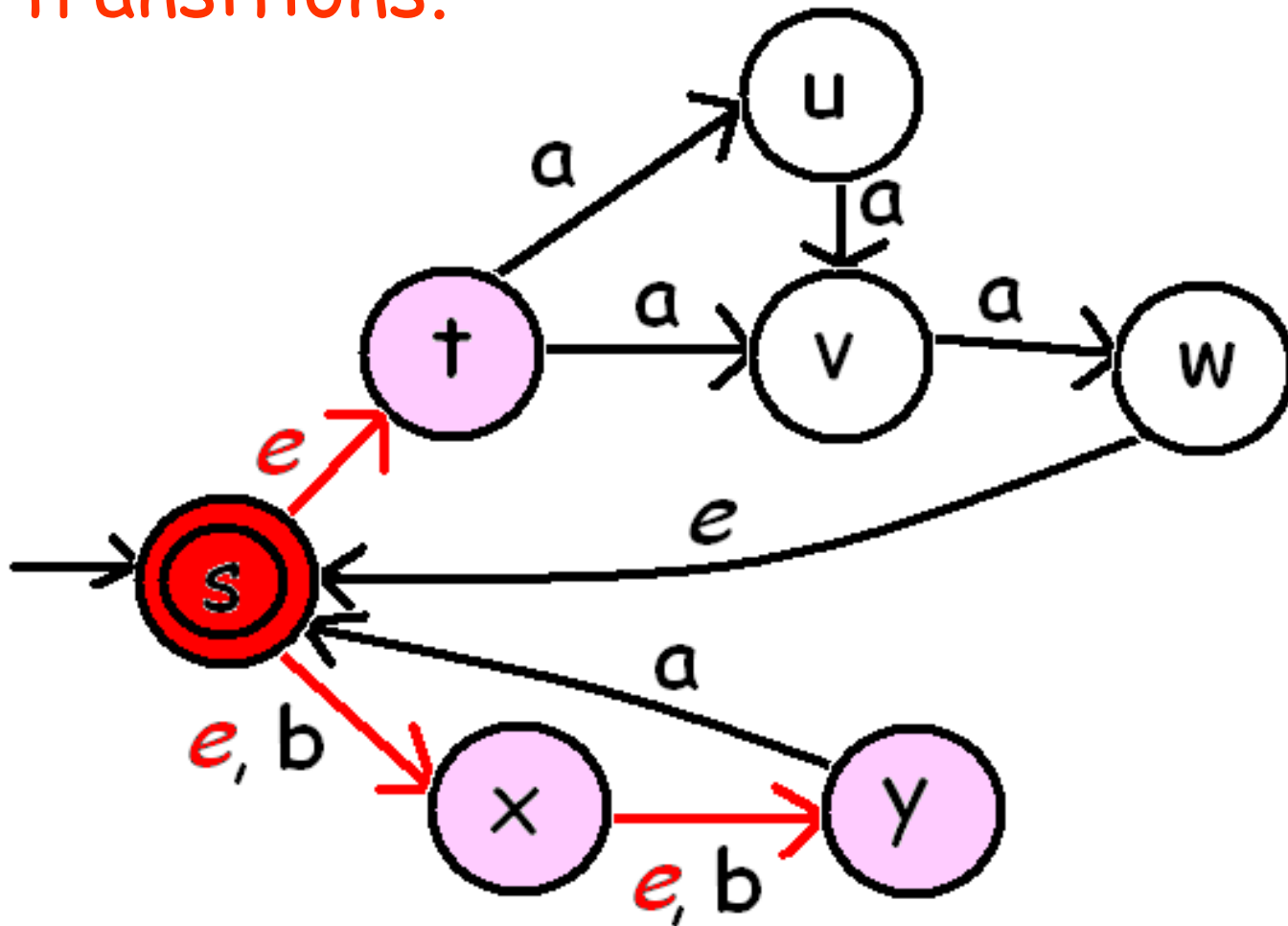
$\delta(P, \sigma) = E(Q)$ where

$Q = \{ q: \text{for some } p \text{ in } P, (p, \sigma, q) \text{ is in } \Delta \}$

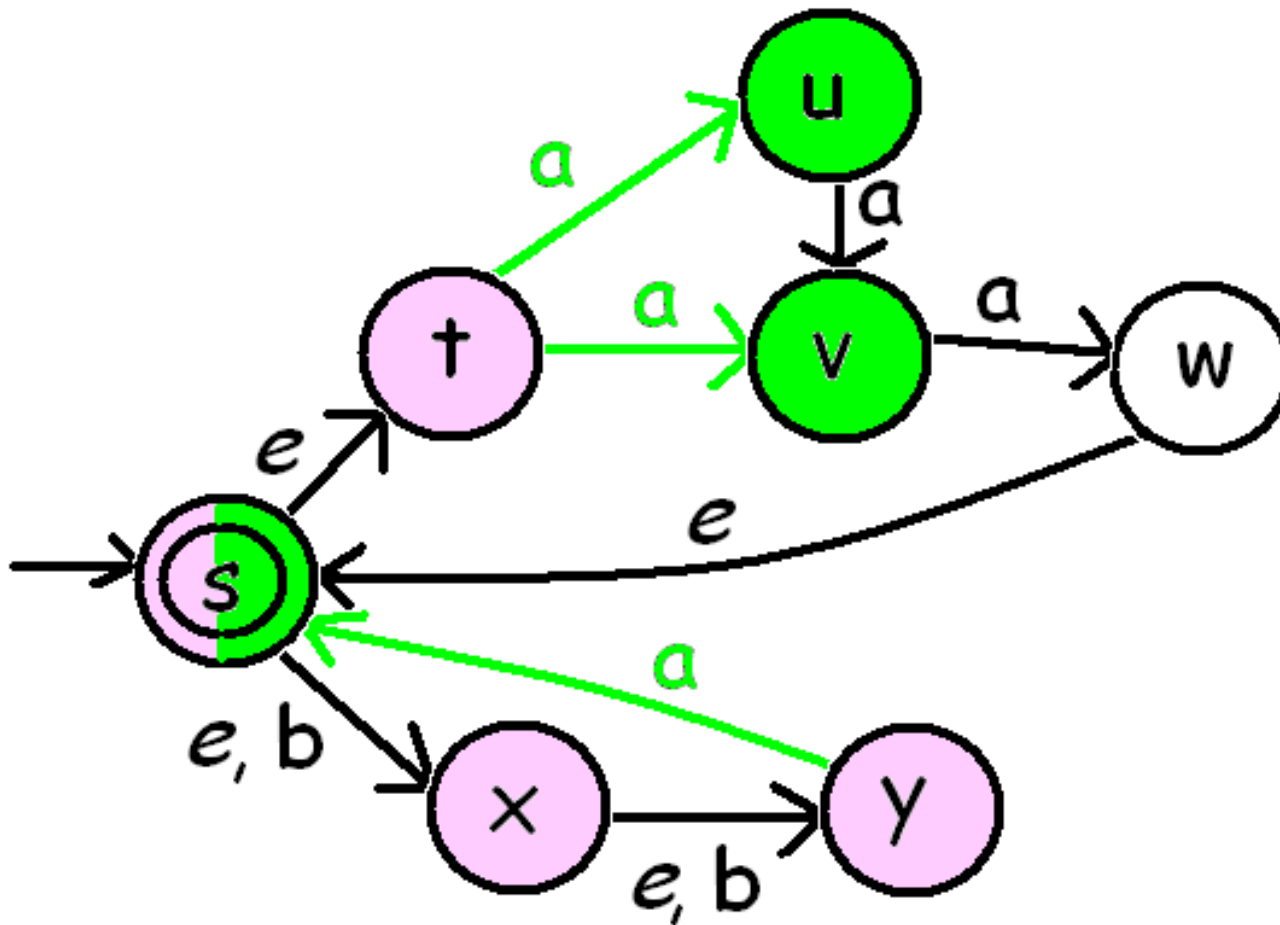
Convert this N DFA to a DFA:



Start state is $\{s, t, x, y\}$ = states reachable from s by traversing 0 or more e -transitions.

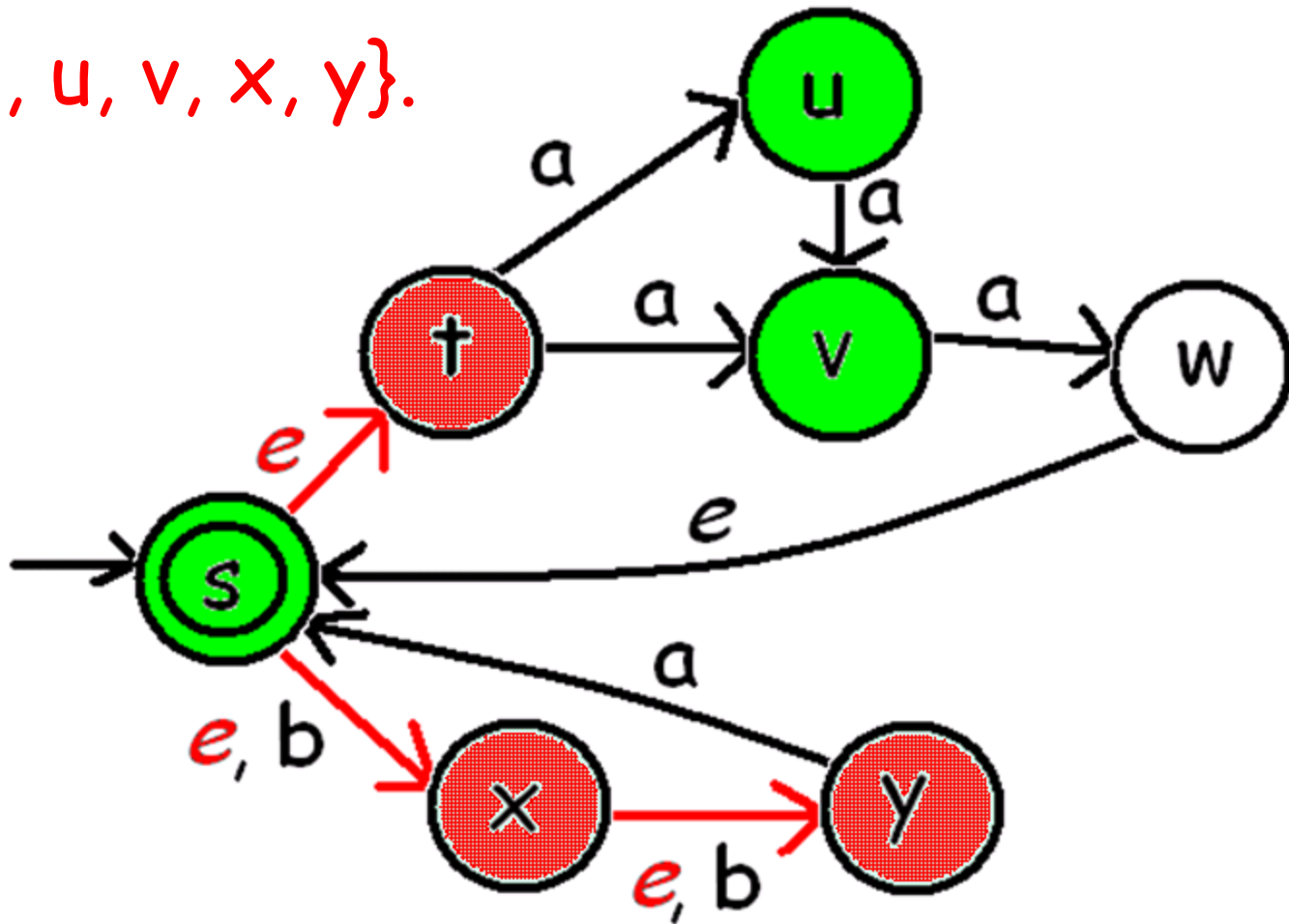


Read a from $\{s, t, x, y\}$ and the next state can be $\{s, u, v\}$.

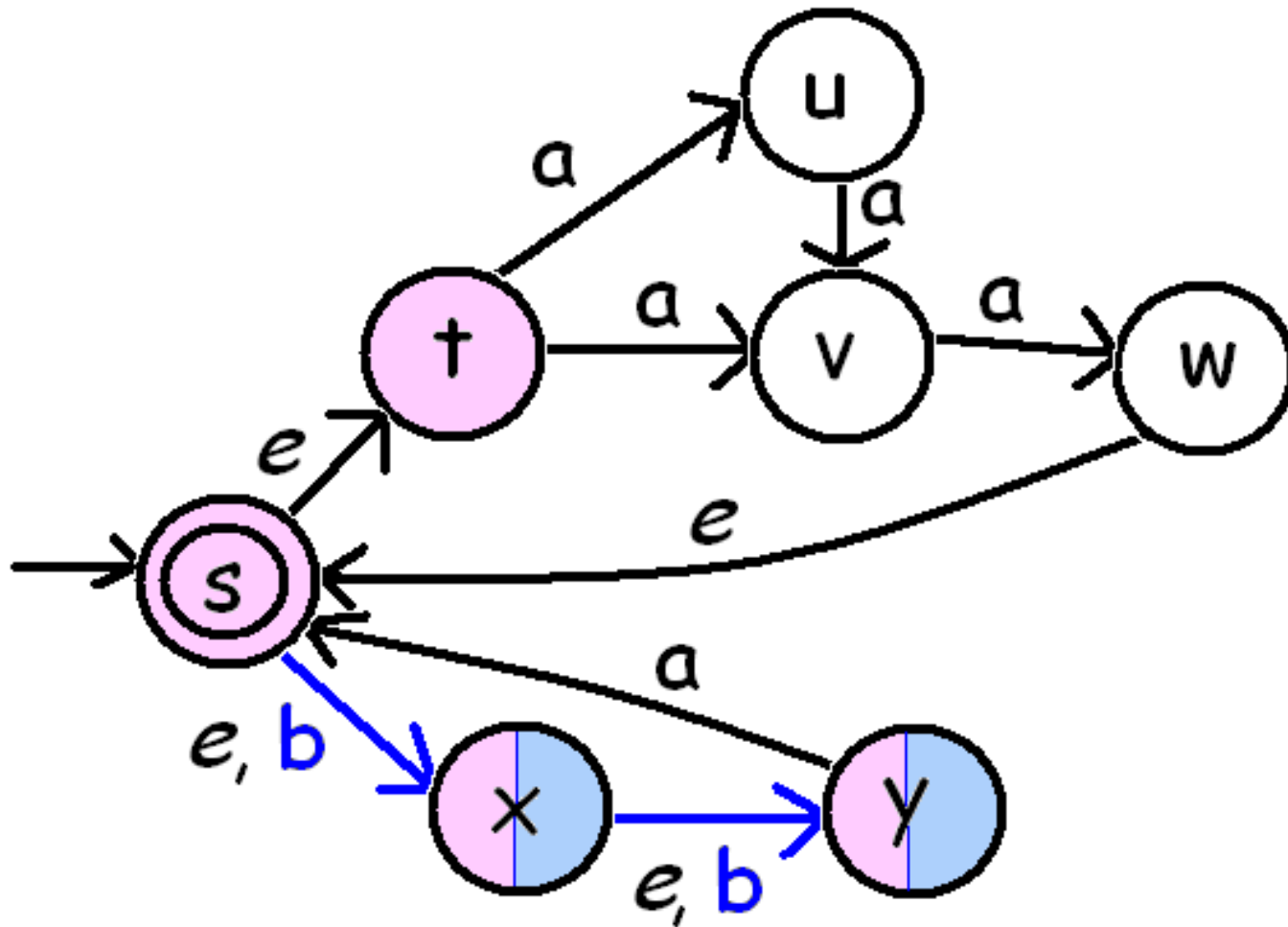


Then follow zero or more ϵ -transitions from $\{s, u, v\}$. The next state is

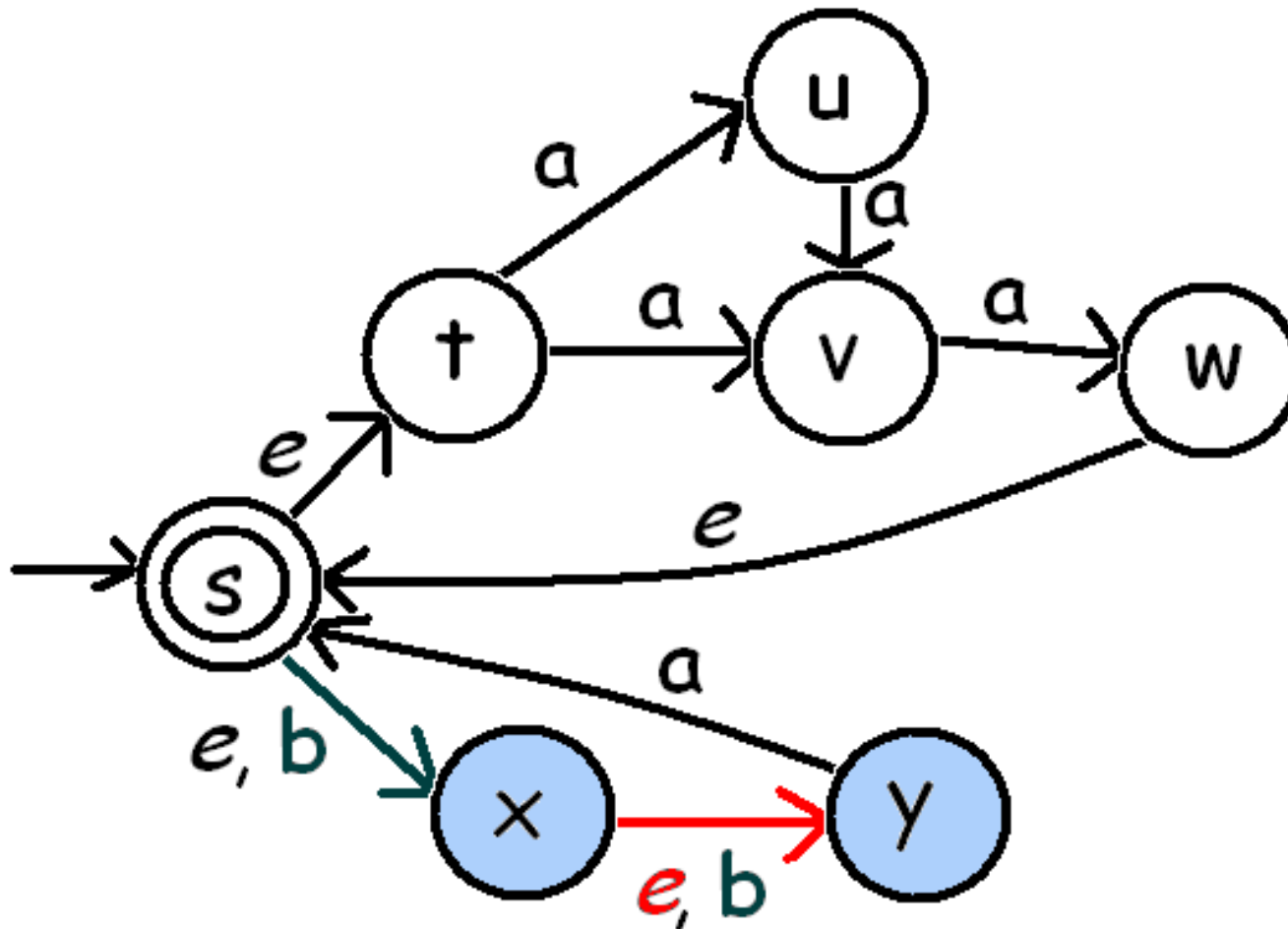
$\{s, t, u, v, x, y\}$.



Read b from $\{s, t, x, y\}$ and the next state can be $\{x, y\}$.



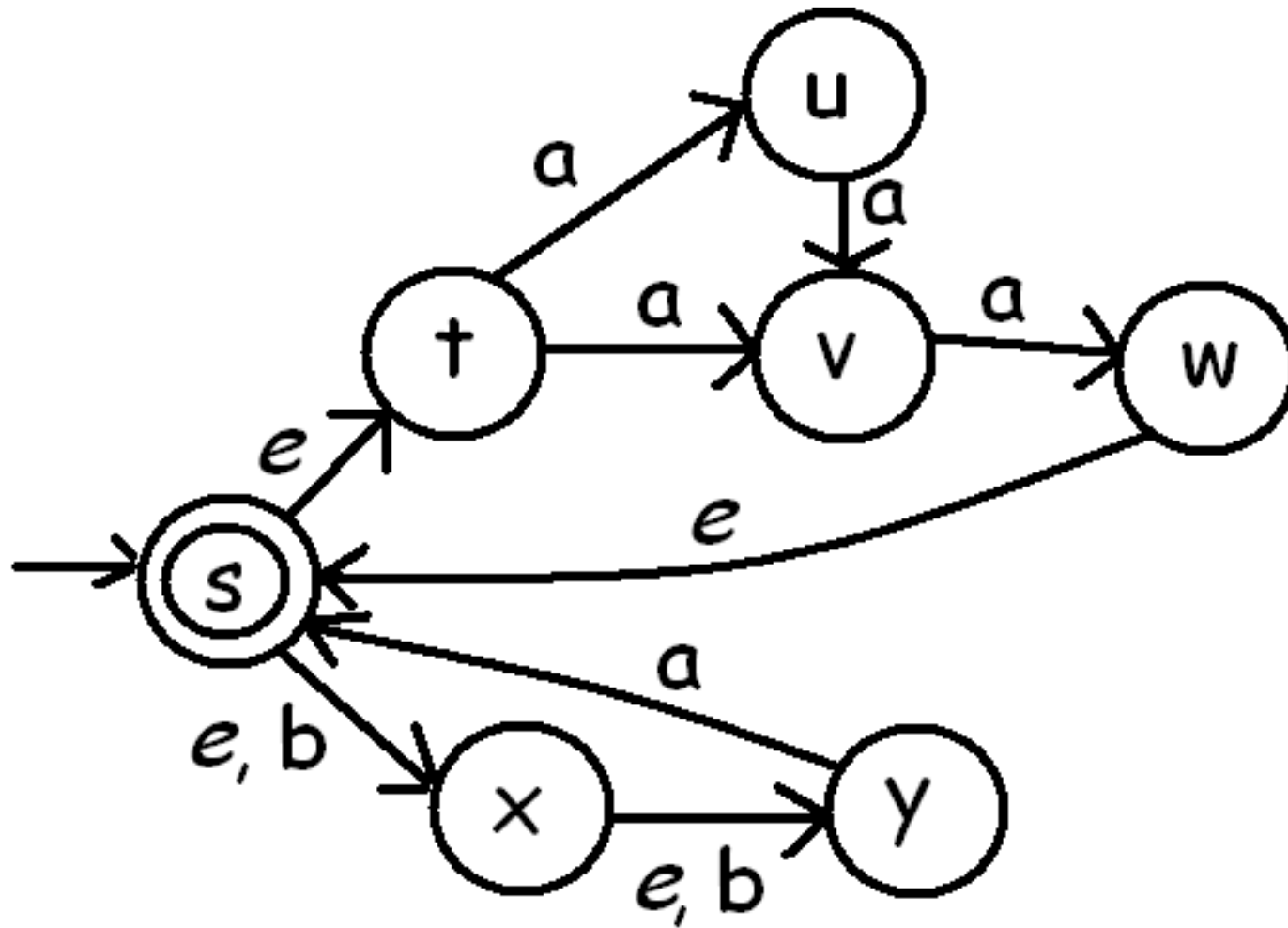
Then follow zero or more e -transitions from $\{x, y\}$. The next state is $\{x, y\}$.



The number of possible states could be exponential. But on assignments and exams, only a small subset of them will ever be pertinent.

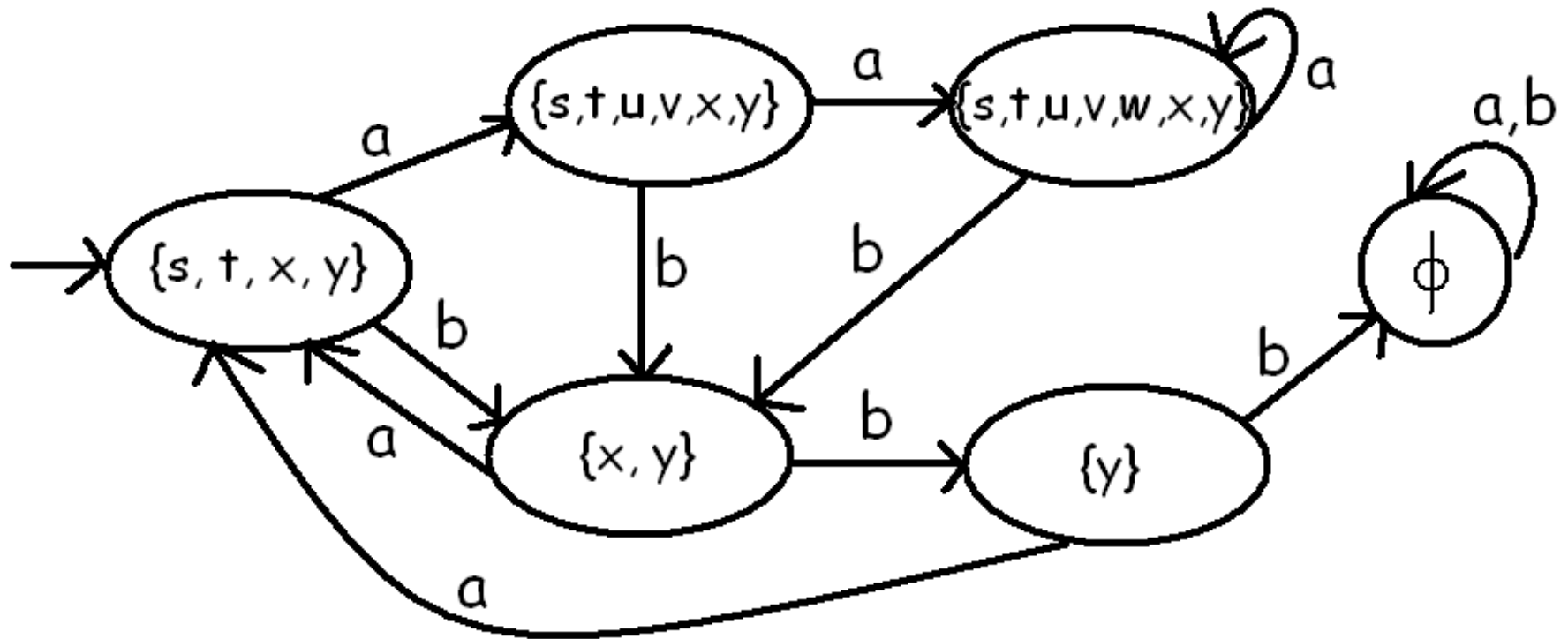
Only work out transitions for pertinent states.

Convert this N DFA to a DFA:



Which states should be final states?

Original machine: only final state was s .



Which states should be final states?

Answer: new states whose subsets contain a state which was a final state originally.

