For each language, give a regular expression that generates the language and a DFA that accepts it.

$L_1 = \{ w \in \{0, 1\}^* : \text{has both 00 and 11 as substrings}\}$

$L_2 = \{ w \in \{a, b\}^* : w \text{ ends with abab}\}$

$L_3 = \{ w \in \{a, b\}^* : w \text{ has 2 or more a's}\}$

$L_4 = \{ w \in \{a,b\}^* : w$ is a string consisting of an even number of a's followed by an odd number of b's. $\}$

$L_5 = \{ w \in \{a,b\}^* :$ the number of a's in w is congruent to 2 or 4 modulo 5 $\}$

Announcements

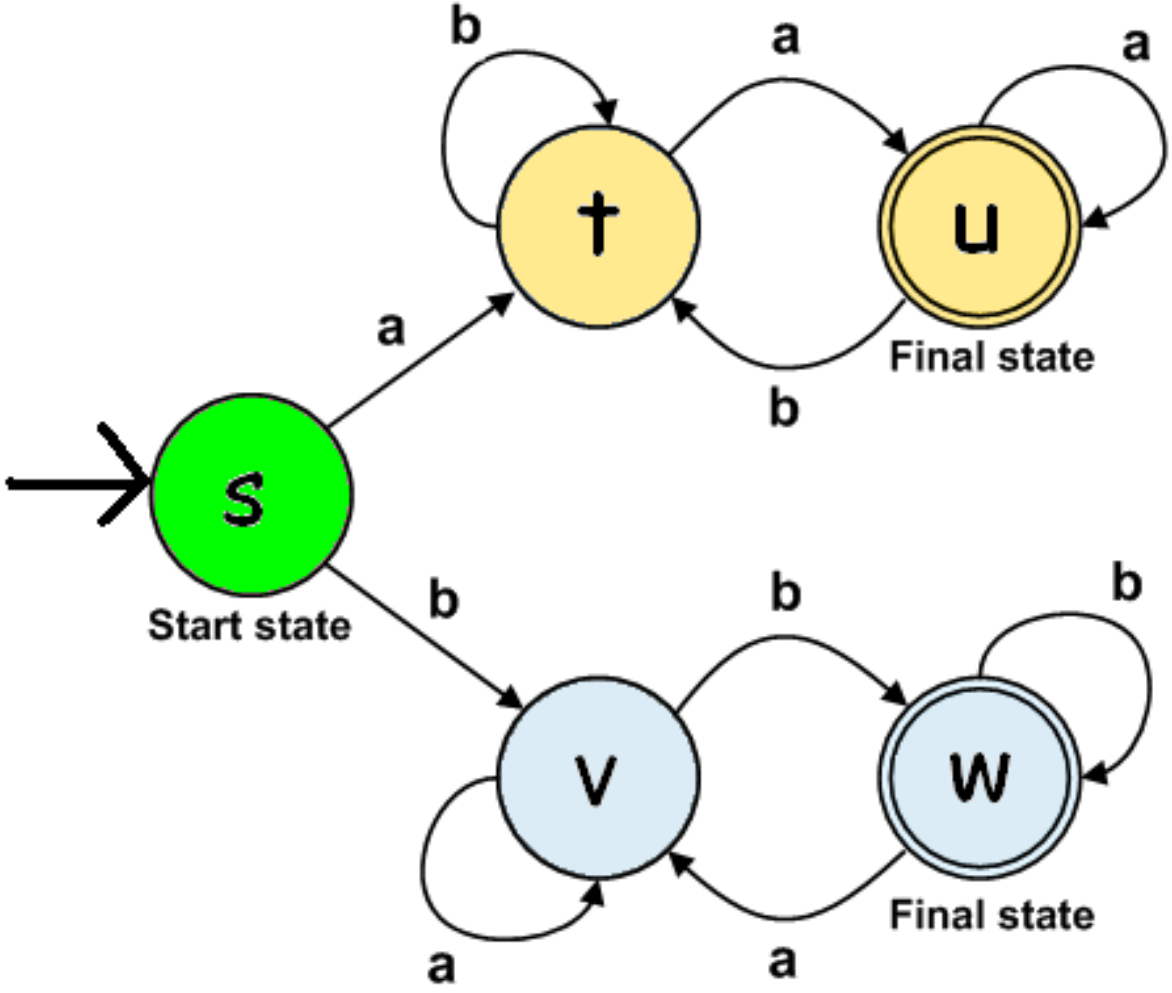
Assignment #2 is posted: Due Fri. Oct. 7.

Tutorial #3: print and take to lab.

Have you tried the regular expression/DFA tutorials?

Did you put your name on your assignment and some boxes for the TA to record your marks in? Put 0 in the box for any questions you omitted.

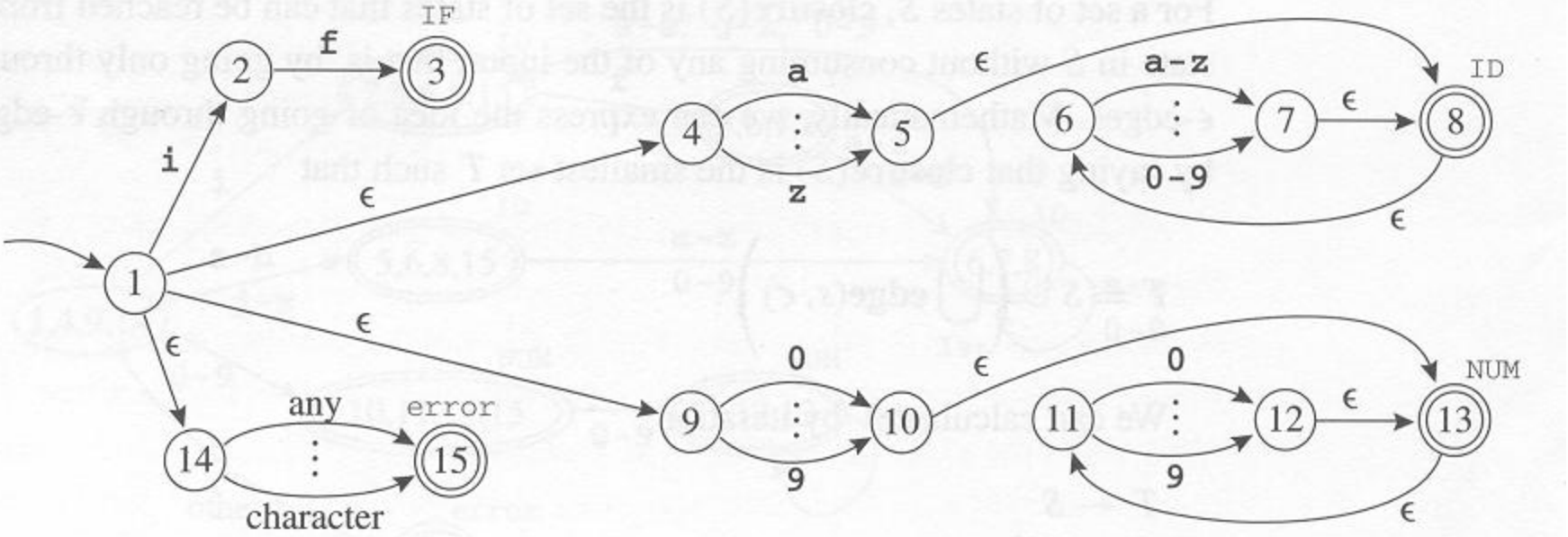
What language does this DFA accept?



CSC 320: Lecture 8: Nondeterministic Finite Automata

if	IF
[a-z][a-z0-9]*	ID
[0-9]+	NUM

Picture from: Raymond Wisman, Notes from compiler design class



Outline: NDFA's are like DFA's but their operation is not deterministic. NDFA's are defined and several examples are given. We prove that for every NDFA there is an equivalent DFA by giving an algorithm which constructs one.

Context: By definition a language is regular if and only if there is a regular expression for it. We will soon prove that a language is regular if and only if there is a DFA which accepts it.

Ultimate Consequences:

You can prove that a language is regular by either

1. finding a regular expression which generates it,
2. finding a DFA which accepts it, or
3. finding a NDFA which accepts it.

A Nondeterministic Finite Automaton (NDFA) M is defined to be a quintuple

$(K, \Sigma, \Delta, s, F)$ where

K is a finite set of states,

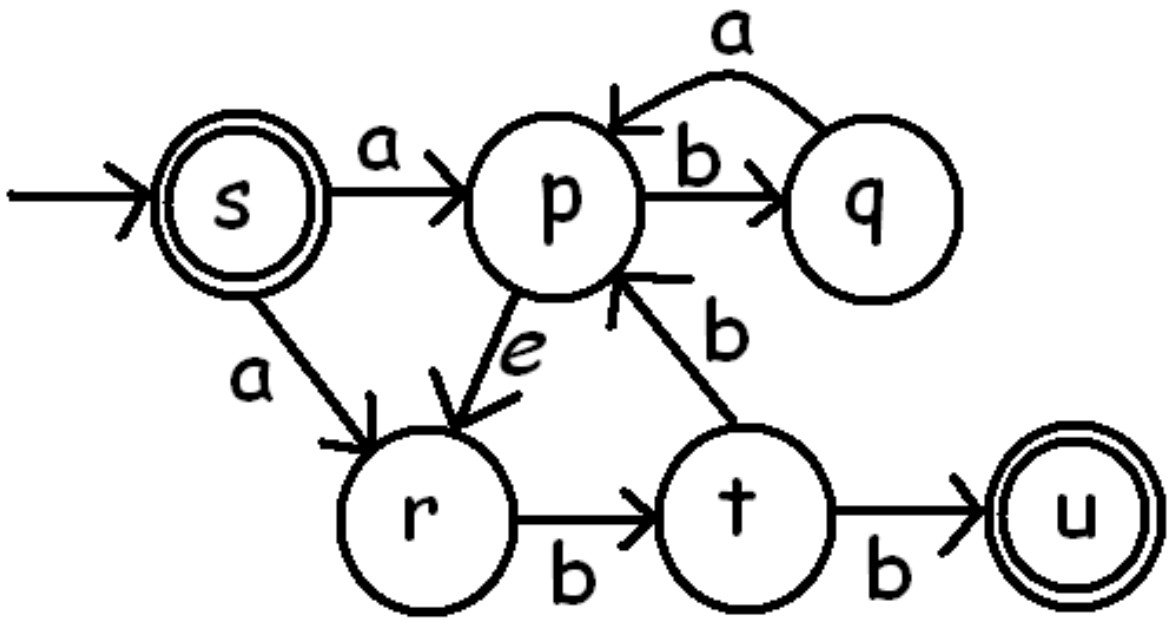
Σ is an alphabet,

Δ, the transition relation, is a subset of $K \times (\Sigma \cup \{\varepsilon\}) \times K$,

$s \in K$ is the start state, and

$F \subseteq K$ is the set of final states.

M



$M = (K, \Sigma, \Delta, s, F)$:

A configuration of a M is an element of $K \times \Sigma^*$.

For $\sigma \in (\Sigma \cup \{\varepsilon\})$, configuration

$(q, \sigma w) \vdash (r, w)$ if (q, σ, r) is in Δ.

The notation $\vdash^*$ means yields in zero or more steps.

$M = (K, \Sigma, \Delta, s, F)$:

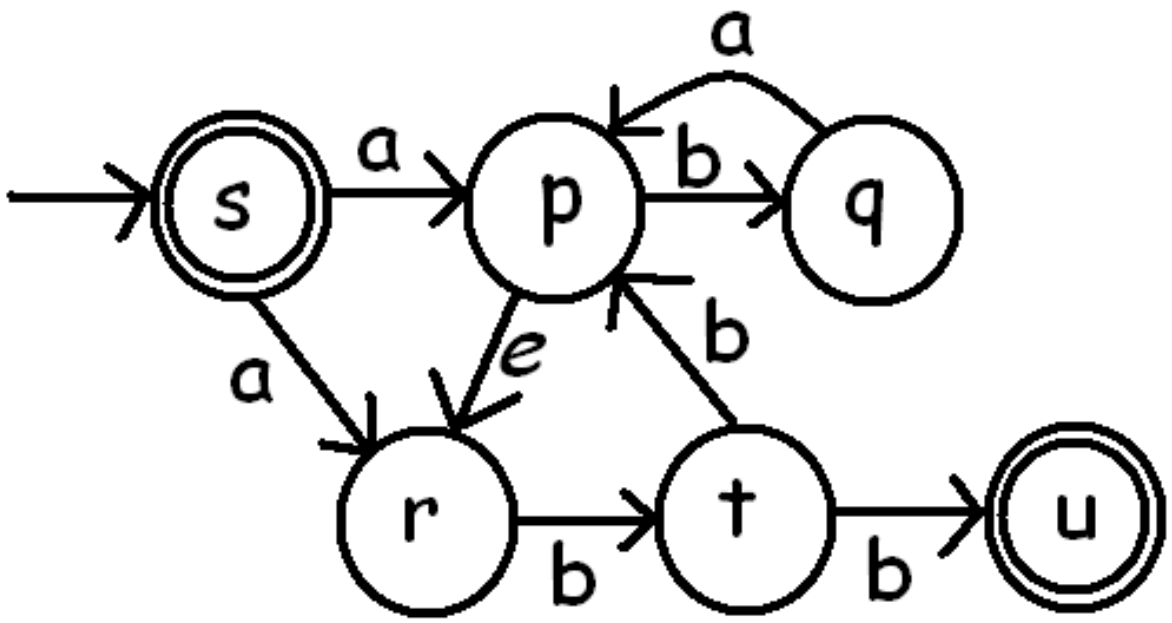
NDFA M accepts input w if and only if

there exists some computation such that

$(s, w) \vdash^* (f, \varepsilon)$ for some $f \in F$.

L(M), the language accepted by M is

$\{ w \in \Sigma^* : (s, w) \vdash^* (f, \varepsilon)$ for some $f \in F\}$.



M

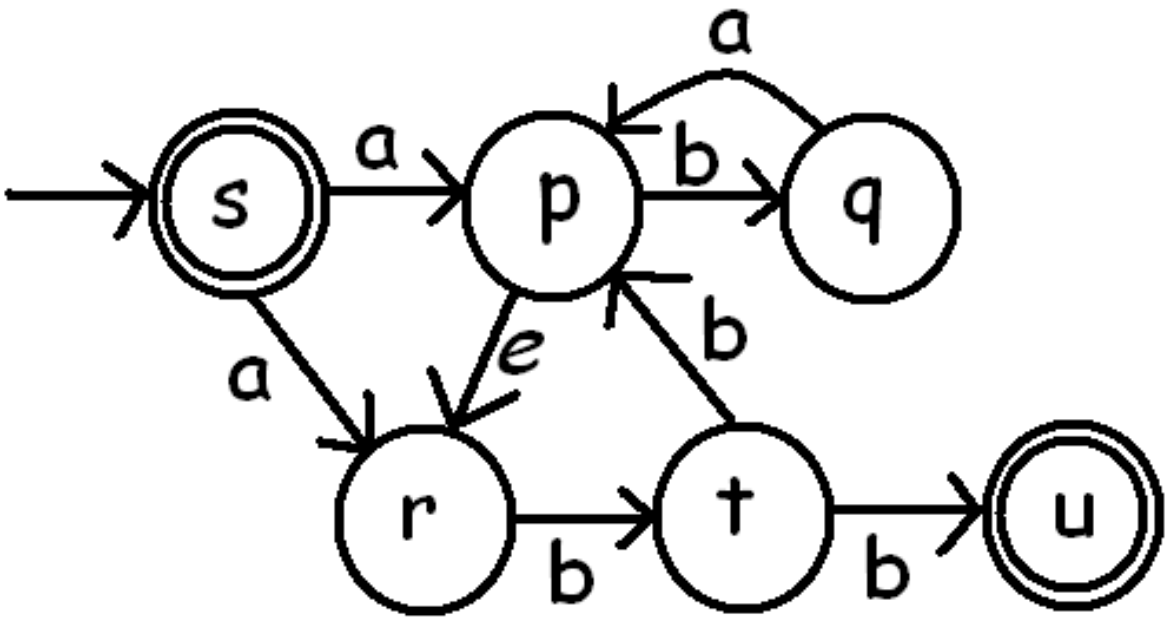
Some non-accepting computations:

(s, abb) ⊢ (p, bb) ⊢ (q, b)

STUCK- no way to finish consuming the input.

(s, abb) ⊢ (r, bb) ⊢ (t, b) ⊢ (p,*e*)

Ends in a non-final state p.



M

Some accepting computations:

(s, abb) $\vdash$ (r, bb) $\vdash$ (t, b) $\vdash$ (u, *e*)

(s, abb) $\vdash$ (p, bb) $\vdash$ (r, bb) $\vdash$ (t, b) $\vdash$ (u, *e*)

abb is in L(M) since it has at least one accepting computation.

Prove the following languages over $\Sigma = \{0, 1\}$ are regular by constructing NDFA's which accept them.

1. $L_1 = \{ w : w$ starts and ends with $0\}$.
2. $L_2 = (000 \cup 11 \cup 01)^*$
3. $L_1 \cup L_2$
4. $L_1 \cdot L_2$