Nondeterministic Automata Looking for Substrings

We can build an NFA to recognize a string that ends in any given substring $a_1 a_2 \cdots a_n$ if we:

1. Have a start state $s_0$ that goes to itself on any input.
   - I.e., you can always "guess" that the substring has not yet begun, even if the input is $a_1$.
2. For $i = 1, 2, \ldots, n$, $s_{i-1}$ goes to $s_i$ on input $a_i$.
3. $s_n$ is the accepting state.

**Example:** Strings that end in `*`.:

- Careful how you use this automaton: when it accepts, the job is done and you do not continue searching for a later occurrence of `*`.
- We can convert to a DFA as follows:

**Class Problem**

Describe a NFA that accepts those strings of 0's
and 1's such that the 10th position from the end is 1.

- Note this automaton's input has no "end-marker." At all times it accepts if 10 inputs ago it received a 1.

Now, describe a DFA that recognizes the same language. How many states do your automata have?

**Regular Expressions**

- An algebraic notation for describing the *regular sets* (= sets of strings accepted by a FA).
  - Note that the subset construction tells us that NFA's and DFA's accept the same sets of strings.
  - A set of strings is a *language*.

- The RE's use three operators: union, concatenation, and "closure."

- $L(R) = \text{the language represented by RE } R$.

**Why Regular Expressions?**

An important notation for expressing character-string patterns. Used in many UNIX commands, e.g., grep, lex, editors, and (in somewhat different form) the shell.

**Operands**

- Constants, which are symbols $a$ standing for the language $\{a\}$ consisting of one string; that string is of length 1 and has the symbol $a$ in its lone position.

- Variables, standing for unknown languages.

- The special symbols $\emptyset$ standing for the empty language and $\epsilon$ standing for $\{\epsilon\}$ (the set containing only the empty string).
  - Note that $\emptyset \neq \{\epsilon\}$. 
Concatenation

If $R$ and $S$ are RE's, then $RS (=\text{concatenation of } R \text{ and } S)$ denotes the language $L(RS) = \{rs \mid r \text{ is in } R \text{ and } s \text{ is in } S\}$.

- In general, the language of $RS$ is formed by concatenating a string from $R$ and a string from $S$ in all possible combinations.
- Special case: $a_1a_2\cdots a_n$ (concatenation of $n$ RE's, each a single symbol) denotes one-string language \{a_1a_2\cdots a_n\}.

Union

If $R$ and $S$ are RE's then $L(R \mid S) = L(R) \cup L(S)$.

Example: Let $R = (a \mid b)(ab \mid ba)$. What is $L(R)$?

- $L(a \mid b) = \{a, b\}$.
- $L(ab \mid ba) = \{ab, ba\}$.
- $L(R) = \{a, b\}\{ab, ba\} = \{aab, aba, bab, bba\}$.

Closure

If $R$ is an RE, then $L(R^*)$ denotes $\{\epsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup \cdots$.

- That is, the union of zero or more strings chosen arbitrarily from $R$.

Example: $L((a \mid b)^*) = \text{set of all strings of } a\text{'s and } b\text{'s}$.

Example: $L(a^*b^*) = \text{set of all strings of } a\text{'s and } b\text{'s where the } a\text{'s precede the } b\text{'s}$.

Class Problem

Write a regular expression denoting the set of strings of 0's and 1's such that the 10th position from the right end is 1.