CSU390 Theory of Computation — Lecture 4 - Closure Properties, Nondeterminism
Spring 2008 — January 24, 2008

Lecture 4 - Closure Properties, Nondeterminism

Formal Definitions
Theorem: The class of regular languages is closed under the union operation.

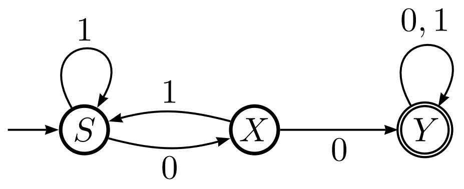
Proof: Let L_1 and L_2 be regular languages, and let
$M1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize L_1.
$M2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize L_2.

Construct $M = (Q, \Sigma, \delta, q_0, F)$ to recognize $L_1 \cup L_2$.

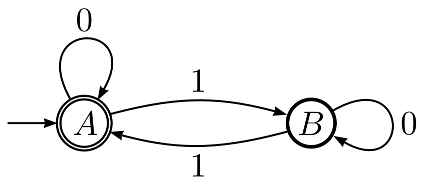
1. $Q = \{(r_1, r_2) \mid r_1 \in Q_1 \text{ and } r_2 \in Q_2\}$
2. Σ is the same or $\Sigma = \Sigma_1 \cup \Sigma 2$ and change proof.
3. $\delta((r_1, r_2), a) = (\delta(r_1, a), \delta(r_2, a))$.
4. $q_0 = (q_1, q_2)$
5. $F = \{(r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$

Example $M1 \cup M6$ from lecture 2 but change the state names first and alphabets.

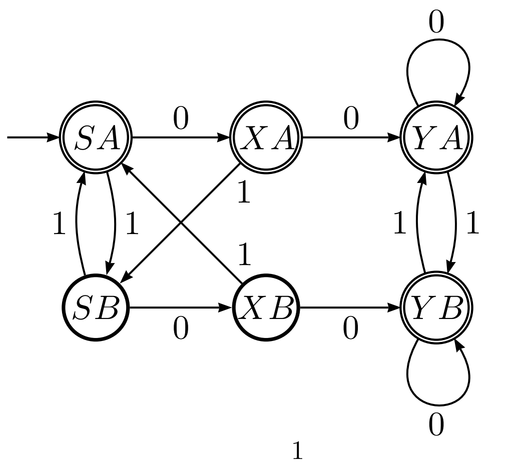
M1 Alphabet $= \{0, 1\}$



M6 Alphabet $= \{0, 1\}$



M to recognize L(M1) $\cup$ L(M6)



Theorem: The class of regular languages is closed under the intersection operation.

Theorem: The class of regular languages is closed under concatenation.

Theorem: The class of regular languages is closed under the Kleene star operation.

We could try to prove the concatenation theorem by linking two automata together

$$M_1 \longrightarrow M_2$$

where M_1 accepts L_1 and M_2 accepts L_2 and we make the M_1 accept states non-accepting but send transition arrows from them to the start state of M_2.

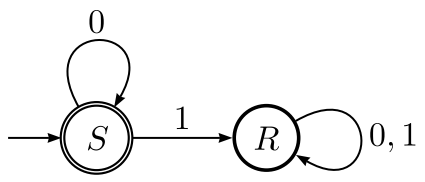
But what symbols do we write on these arrows?

A string in $L_1 \circ L_2$ might have many initial substrings that are in L_1 only some of which have an ending substring in L_2.

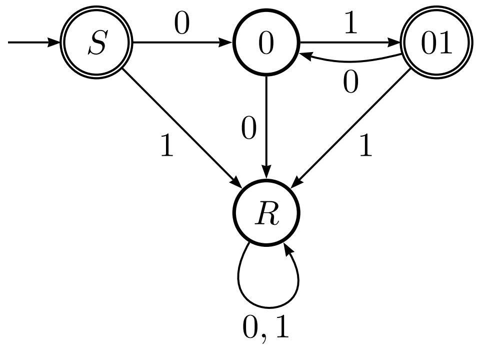
Example
Let $L_1 = 0^*$ and $L_2 = \{01\}^*$.

The string 00001 is in $L_1 \circ L_2$ as $000 \in L_1$ and $01 \in L_2$. But we can also split the string 00001 as 00 followed by 001 or as 0000 followed by 1. In both cases, the first string in in L_1 but the second string is not in L_2.

M1 Alphabet $= \{0, 1\}$

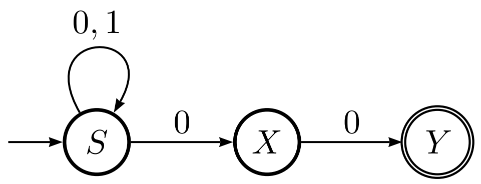


M2 Alphabet $= \{0, 1\}$



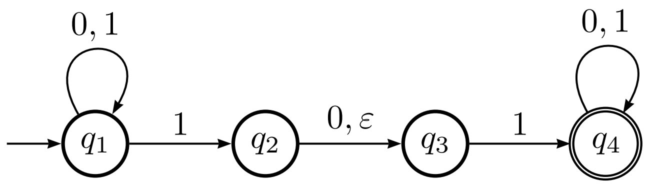
Nondeterminism - NFAs

N1 Alphabet $= \{0, 1\}$



Where can the strings 00, 001000 end up?

N2 Alphabet $= \{0, 1\}$



Where can the strings 000010010, 001000 end up?

What language does this automaton recognize?

Formal Definition of a Nondeterministic FInite Automaton

An ***NFA*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of states,
2. Σ is a finite alphabet,
3. $\delta : Q \times \Sigma_\varepsilon \longrightarrow \mathfrak{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$.

A DFA is always an NFA.

Write out the formal definition of N2. This is in Sipser. Use $\emptyset$ (as in the book) for transitions such as $\delta(\{q2\}, \varepsilon) = \emptyset$. (not $\delta(\{q2\}, \varepsilon) = q2$ as I said in class)

N ***accepts*** $w = y_1 y_2 \cdots y_m$ where $y_i \in \Sigma_\varepsilon$ if there is a sequence of states $r_0 r_2 \cdots r_m \in Q$ such that

1. $r_0 = q_0$,
2. $r_{i+1} \in \delta(r_i, y_{i+1})$, for $i = 1, 2, \ldots, m-1$ and
3. $r_m \in F$.

You can think of a string as a bus ticket that lets you travel around the NFA.
Your ticket can take you to many different places.

From the start state, you can go to any place that you can get to for free (stay put or follow an ε arrow). Then repeat these steps.

1. Rip off the first symbol on the string and use it to get to some place it will let you go.
2. From there go to any place you can get to for free.

until there string (ticket) is used up.

If you can use you string (ticket) to get from the start state to any accepting state, then the string is accepted.

Equivalence of NFAs and DFAs

If $N = (Q, \Sigma, \delta', q_0', F')$ is and NFA, recognizing language A, define DFA $M = (\mathfrak{P}(Q), \Sigma, \delta, q_0, F)$ recognizing A. First, assume no ε's in the transitions.

1. $Q' = \mathfrak{P}(Q)$.
2. For $R \in Q'$ and $a \in \Sigma$ let $\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$.
3. $q_0' = \{q_0\}$.
4. $F' = \{r \in Q' \mid R \text{ R contains an accept state of } N\}$.

For $R \subseteq Q$ let

$$E(R) = \{q \in Q \mid q \text{ can be reached from } R \text{ by traveling 0 or more } \varepsilon \text{ arrows}\}.$$

Now define

$$\delta'(R, a) = \{q \in Q \mid q \in E(\delta(r, a)) \text{ for some } r \in R\}.$$

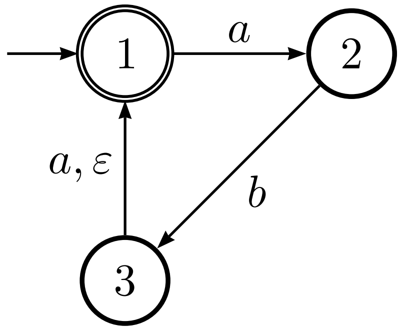
Finally, Use $E(\{q_0\})$ as the start state instead of $\{q_0\}$.

The construction serves as an informal proof.

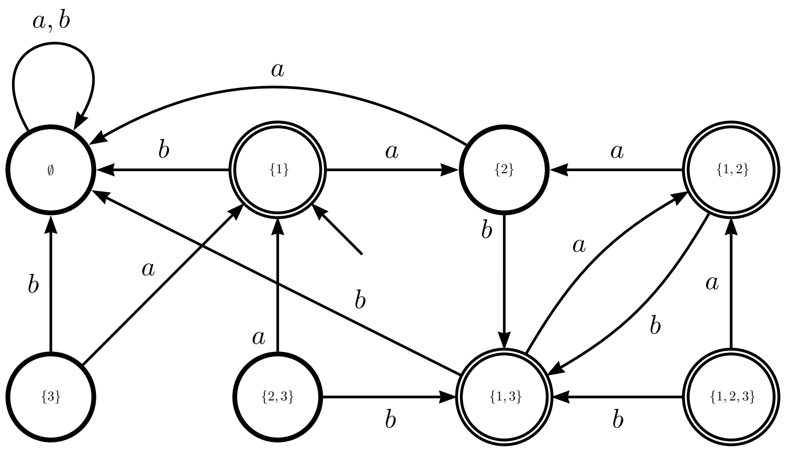
Corollary: A language is regular iff some NFA recognizes it.

Example

N2 Alphabet $= \{a, b\}$



N2DFA Alphabet $= \{a, b\}$



Closure Under Regular Operations

Closure Under Union - again but with NFAs

Just add a new start state and ε arrows to the two original start states.

Closure Under Concatenation

If N1 recognizes L1 and N2 recognizes L2, keep the N1 start state, make N1's accepting states non-accepting but add ε arrows from them to the N2 start state (which will no longer be a start state).

Closure Under Kleene Star

If N1 recognizes L1, add a new start/final state with ε arrow to the old start state and add ε arrows from the final states to the old start state.

There are very good explanations and pictures for these constructions in Sipser.