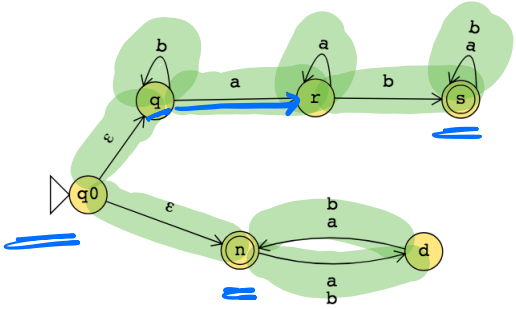
Monday April 11

The state diagram of an NFA over $\{a, b\}$ is below. The formal definition of this NFA is:

5-tuple of parameters



$(\{q_0, q, r, s, n, d\}, \{a, b\}, \delta, q_0, \{s, n\})$

Q, Σ, transition function (arrows), start state, set of accept state

$\delta: Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$

labels

i.e. $\delta: \{q_0, q, r, s, n, d\} \times \{a, b, \varepsilon\} \to \mathcal{P}(\{q_0, q, r, s, n, d\})$

states

	Labels a	b	ε
q_0	$\emptyset$	$\emptyset$	$\{q_0, n\}$
q	$\{r\}$	$\{q\}$	$\emptyset$
r	$\{r\}$	$\{s\}$	$\emptyset$
s	$\{s\}$	$\{s\}$	$\emptyset$
n	$\{d\}$	$\{d\}$	$\emptyset$
d	$\{n\}$	$\{n\}$	$\emptyset$

The language recognized by this NFA is: $\{w \in \{a,b\}^* \mid w$ has substring ab & w has even length$\}$

Suppose A_1, A_2 are languages over an alphabet Σ. **Claim:** if there is a NFA N_1 such that $L(N_1) = A_1$ and NFA N_2 such that $L(N_2) = A_2$, then there is another NFA, let's call it N, such that $L(N) = A_1 \cup A_2$.

Proof idea: Use nondeterminism to choose which of N_1, N_2 to run.

don't want interference

Formal construction: Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ and assume $Q_1 \cap Q_2 = \emptyset$ and that $q_0 \notin Q_1 \cup Q_2$. Construct $N = (Q, \Sigma, \delta, q_0, F_1 \cup F_2)$ where

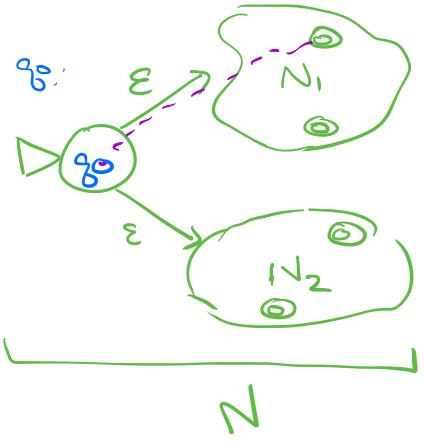
* doesn't include q_0

- $Q = \{q_0\} \cup Q_1 \cup Q_2$
- $\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ is defined by, for $q \in Q$ and $a \in \Sigma_\varepsilon$:

$$\delta((q,a)) = \begin{cases} \{q_1, q_2\} & q = q_0, a = \varepsilon \\ \emptyset & q = q_0, a \in \Sigma \\ \delta_1((q,a)) & q \in Q_1, a \in \Sigma_\varepsilon \\ \delta_2((q,a)) & q \in Q_2, a \in \Sigma_\varepsilon \end{cases}$$

simulate N_1

simulate N_2



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

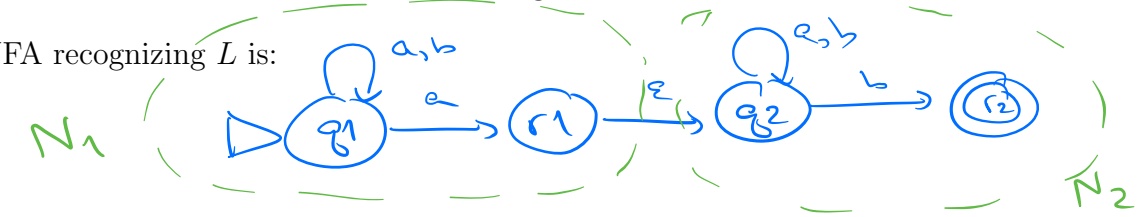
Proof of correctness would prove that $L(N) = A_1 \cup A_2$ by considering an arbitrary string accepted by N, tracing an accepting computation of N on it, and using that trace to prove the string is in at least one of A_1, A_2; then, taking an arbitrary string in $A_1 \cup A_2$ and proving that it is accepted by N. Details left for extra practice.

Extra practice: apply to first example.

Over the alphabet $\{a, b\}$, the language L described by the regular expression $\Sigma^* a \Sigma^* b$

includes the strings ab and aab and excludes the strings ε and a and b

The state diagram of a NFA recognizing L is:



$A_1 \circ A_2 = \{ xy \mid x \in A_1 \text{ and } y \in A_2 \}$. $w =$

Suppose A_1, A_2 are languages over an alphabet Σ. **Claim:** if there is a NFA N_1 such that $L(N_1) = A_1$ and NFA N_2 such that $L(N_2) = A_2$, then there is another NFA, let's call it N, such that $L(N) = A_1 \circ A_2$.

Proof idea: Allow computation to move between N_1 and N_2 "spontaneously" when reach an accepting state of N_1, guessing that we've reached the point where the two parts of the string in the set-wise concatenation are glued together.

in A_1 $w_1 w_2$ in A_2

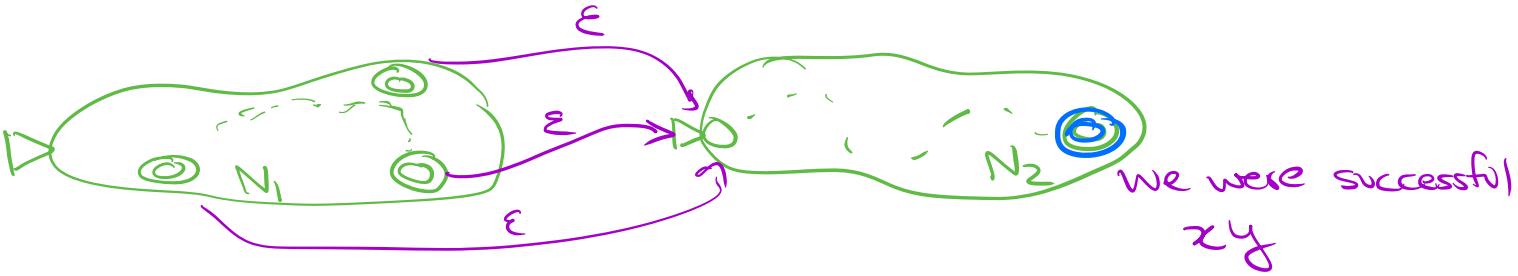
Formal construction: Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ and assume $Q_1 \cap Q_2 = \emptyset$. Construct $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q =$ $Q_1 \cup Q_2$
- $q_0 =$ q_1 start by simulating N_1
- $F =$ F_2 need both pieces to work, and N_2 is second so only use its accept states.
- $\delta : Q \times \Sigma_\varepsilon \to$ ~~Q~~ $\mathcal{P}(Q)$ is defined by, for $q \in Q$ and $a \in \Sigma_\varepsilon$:

$$\delta((q,a)) = \begin{cases} \delta_1((q,a)) & \text{if } q \in Q_1 \text{ and } q \notin F_1 \\ \delta_1((q,a)) & \text{if } q \in F_1 \text{ and } a \in \Sigma \\ \delta_1((q,a)) \cup \{q_2\} & \text{if } q \in F_1 \text{ and } a = \varepsilon \\ \delta_2((q,a)) & \text{if } q \in Q_2 \end{cases}$$

Simulate N_1

jump to N_2.



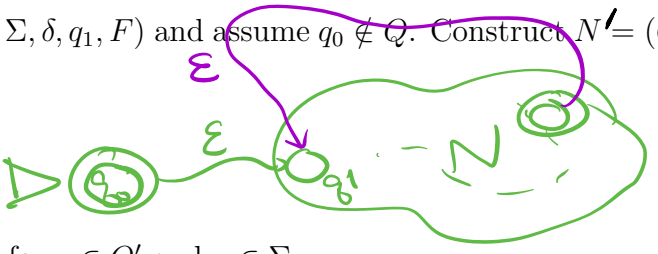
Proof of correctness would prove that $L(N) = A_1 \circ A_2$ by considering an arbitrary string accepted by N, tracing an accepting computation of N on it, and using that trace to prove the string can be written as the result of concatenating two strings, the first in A_1 and the second in A_2; then, taking an arbitrary string in $A_1 \circ A_2$ and proving that it is accepted by N. Details left for extra practice.

Suppose A is a language over an alphabet Σ. **Claim:** if there is a NFA N such that $L(N) = A$, then there is another NFA, let's call it N', such that $L(N') = A^*$.

$A^* = \{w_1 \cdots w_k \mid k \geq 0, k \in \mathbb{Z}, w_i \in A\}$

Proof idea: Add a fresh start state, which is an accept state. Add spontaneous moves from each (old) accept state to the old start state.

Formal construction: Let $N = (Q, \Sigma, \delta, q_1, F)$ and assume $q_0 \notin Q$. Construct $N' = (Q', \Sigma, \delta', q_0, F')$ where



- $Q' = Q \cup \{q_0\}$
- $F' = F \cup \{q_0\}$
- $\delta' : Q' \times \Sigma_\varepsilon \to$ [redacted] is defined by, for $q \in Q'$ and $a \in \Sigma_\varepsilon$:

$\mathcal{P}(Q')$

$$\delta'((q,a)) = \begin{cases} \delta((q,a)) & \text{if } q \in Q \text{ and } q \notin F \\ \delta((q,a)) & \text{if } q \in F \text{ and } a \in \Sigma \\ \delta((q,a)) \cup \{q_1\} & \text{if } q \in F \text{ and } a = \varepsilon \\ \{q_1\} & \text{if } q = q_0 \text{ and } a = \varepsilon \\ \emptyset & \text{if } q = q_0 \text{ and } a \in \Sigma \end{cases}$$

simulate N

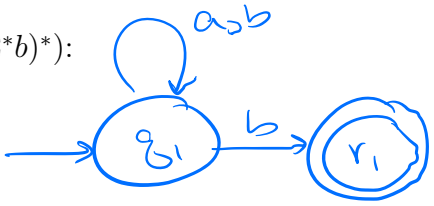
jump to N to start computation

jump to start of N

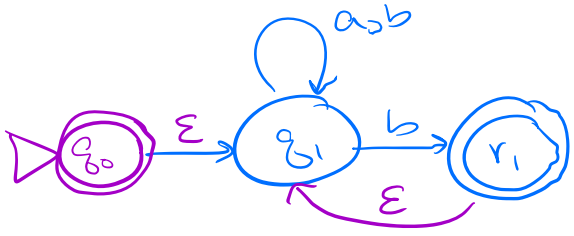
Proof of correctness would prove that $L(N') = A^*$ by considering an arbitrary string accepted by N', tracing an accepting computation of N' on it, and using that trace to prove the string can be written as the result of concatenating some number of strings, each of which is in A; then, taking an arbitrary string in A^* and proving that it is accepted by N'. Details left for extra practice.

Application: A state diagram for a NFA over $\Sigma = \{a, b\}$ that recognizes $L((\Sigma^* b)^*)$:

an NFA recognizing $L(\Sigma^* b)$ is over $\{a,b\}$



Applying construction to get N' gives



True or **False**: The state diagram of any DFA is also the state diagram of a NFA.

True or **False**: The state diagram of any NFA is also the state diagram of a DFA.

True or **False**: The formal definition $(Q, \Sigma, \delta, q_0, F)$ of any DFA is also the formal definition of a NFA.

True or **False**: The formal definition $(Q, \Sigma, \delta, q_0, F)$ of any NFA is also the formal definition of a DFA.

Review Quiz.

Review: Week 3 Monday

Recall: Review quizzes based on class material are assigned each day. These quizzes will help you track and confirm your understanding of the concepts and examples we work in class. Quizzes can be submitted on Gradescope as many times (with no penalty) as you like until the quiz deadline: the three quizzes each week are all due on Friday (with no penalty late submission open until Sunday).

Please complete the review quiz questions on Gradescope about constructions using NFAs.

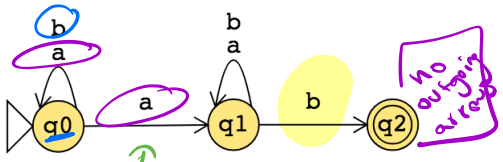
Pre class reading for next time: Theorem 1.39 "Proof Idea", Example 1.41, Example 1.56, Example 1.58.

Wednesday April 13

Consider the state diagram of an NFA over $\{a, b\}$:

$\Sigma = \{a, b\}$

(no spontaneous moves)



3 states

$L(\Sigma^* a \Sigma^* b)$

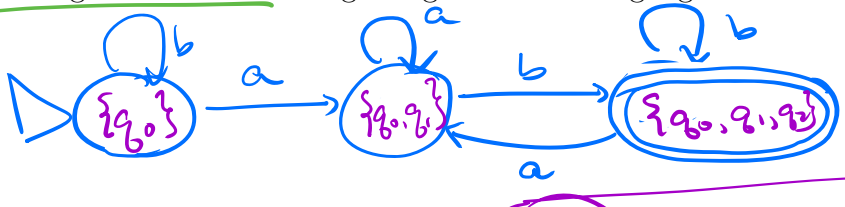
last character is b

at least one a

$\{w \in \{a,b\}^* \mid w \text{ has } ab \text{ as a substring and ends in } b\}$

The language recognized by this NFA is $\{w \in \{a,b\}^* \mid w \text{ has at least one } a \text{ and ends in } b\}$

The state diagram of a DFA recognizing this same language is:



Justifying ---

Labels coming from general construction in

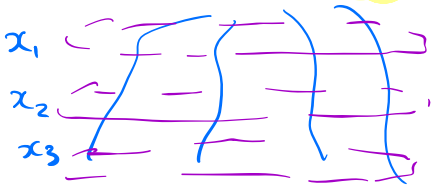
Suppose A is a language over an alphabet Σ. **Claim:** if there is a NFA N such that $L(N) = A$ then there is a DFA M such that $L(M) = A$.

Proof idea: States in M are "macro-states" – collections of states from N – that represent the set of possible states a computation of N might be in.

Formal construction: Let $N = (Q, \Sigma, \delta, q_0, F)$. Define

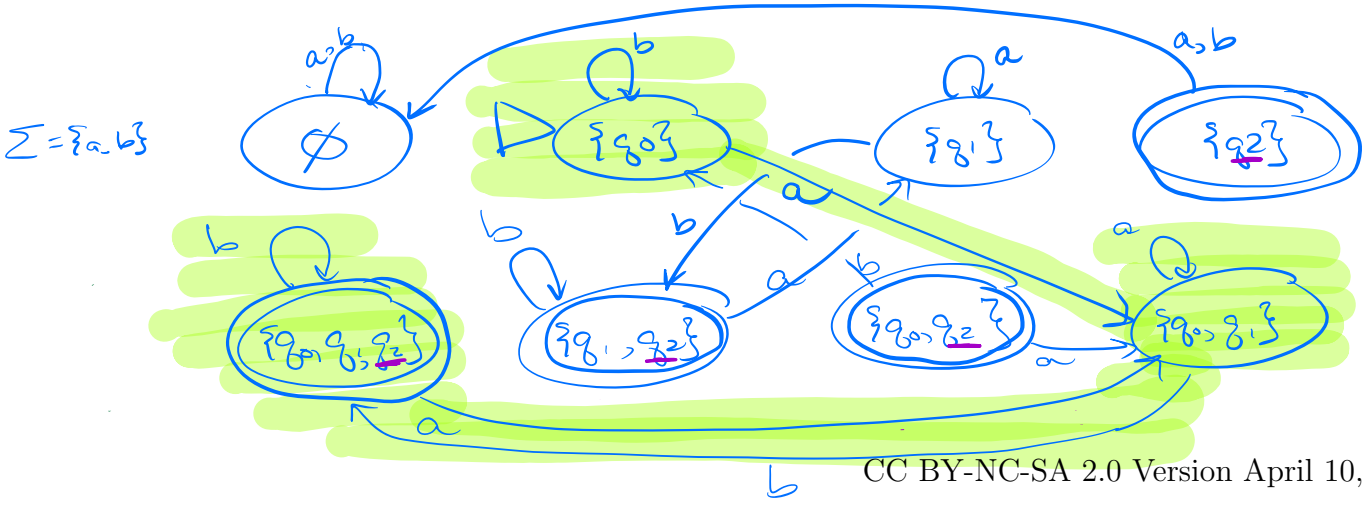
at least one computation path succeeds

$$M = (\mathcal{P}(Q), \Sigma, \delta', q', \{X \subseteq Q \mid X \cap F \neq \emptyset\})$$

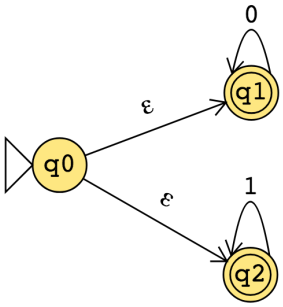


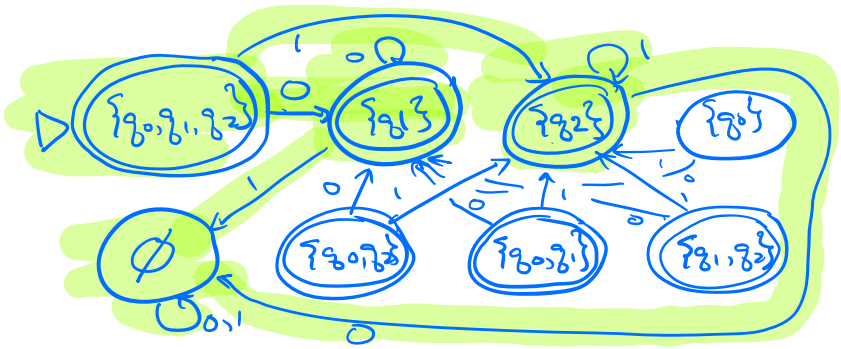
where $q' = \{q \in Q \mid q = q_0$ or is accessible from q_0 by spontaneous moves in $N\}$ and

$\delta'((X, x)) = \{q \in Q \mid q \in \delta((r, x)) \text{ for some } r \in X$ or is accessible from such an r by spontaneous moves in $N\}$



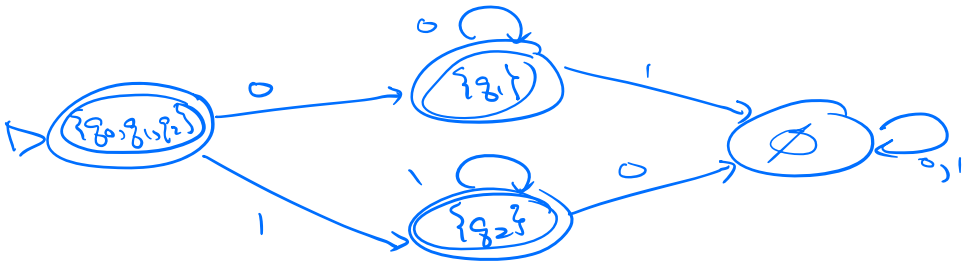
Consider the state diagram of an NFA over $\{0, 1\}$. Use the "macro-state" construction to find an equivalent DFA.





8 states

Prune this diagram to get an equivalent DFA with only the "macro-states" reachable from the start state.



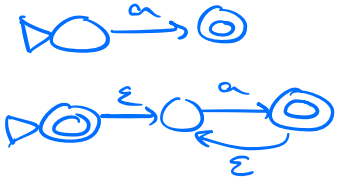
Suppose A is a language over an alphabet Σ. **Claim:** if there is a regular expression R such that $L(R) = A$, then there is a NFA, let's call it N, such that $L(N) = A$.

Structural induction: Regular expression is built from basis regular expressions using inductive steps (union, concatenation, Kleene star symbols). Use constructions to mirror these in NFAs.

Application: A state diagram for a NFA over $\{a, b\}$ that recognizes $L(a^*(ab)^*)$:

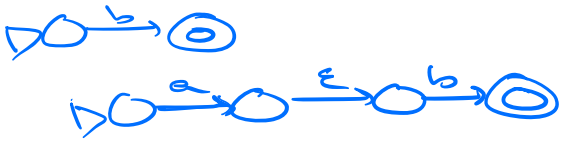
NFA for L(a)

NFA for L(a*)



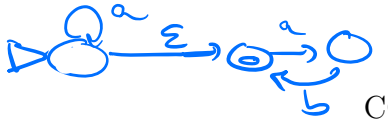
NFA for L(b)

NFA for L(ab)



NFA for L((ab)*)

NFA for L(a*(ab)*)



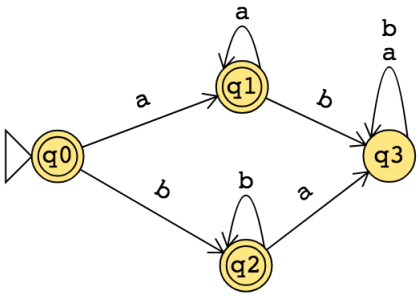
Suppose A is a language over an alphabet Σ. **Claim:** if there is a DFA M such that $L(M) = A$, then there is a regular expression, let's call it R, such that $L(R) = A$.

Proof idea: Trace all possible paths from start state to accept state. Express labels of these paths as regular expressions, and union them all.

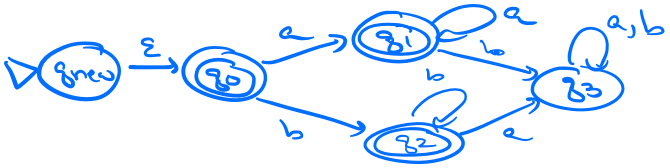
GNFA

1. Add new start state with ε arrow to old start state.
2. Add new accept state with ε arrow from old accept states. Make old accept states non-accept.
3. Remove one (of the old) states at a time: modify regular expressions on arrows that went through removed state to restore language recognized by machine.

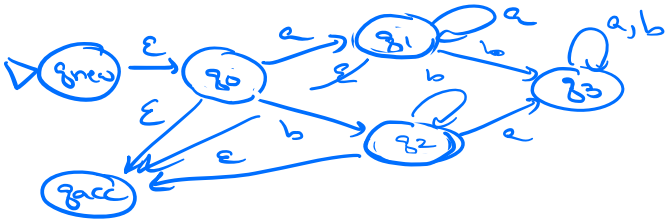
Application: Find a regular expression describing the language recognized by the DFA with state diagram



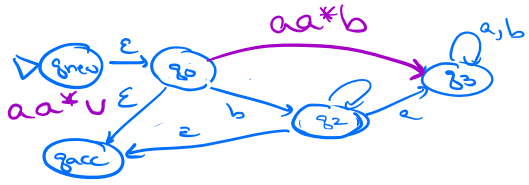
1.



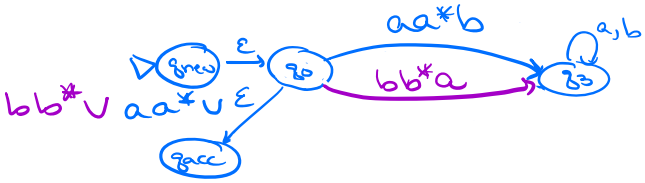
2.



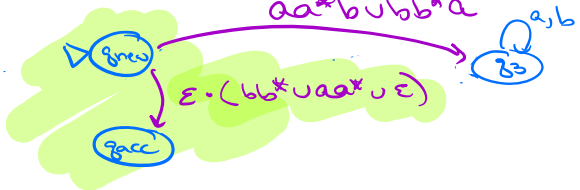
3. Removing q_1: note paths $q_0 \to q_1 \to q_3$ and $q_0 \to q_1 \to q_{acc}$



Removing q_2: note paths $q_0 \to q_2 \to q_3$ and $q_0 \to q_2 \to q_{acc}$



Removing q_0: note paths $q_{new} \to q_0 \to q_3$ and $q_{new} \to q_0 \to q_{acc}$



R is $\varepsilon \circ (bb^* \cup aa^* \cup \varepsilon)$

Conclusion: For each language L,

There is a DFA that recognizes L $\quad \exists M$ (M **is a DFA and** $L(M) = A$)
if and only if
There is a NFA that recognizes L $\quad \exists N$ (N **is a NFA and** $L(N) = A$)
if and only if
There is a regular expression that describes L $\exists R$ (R **is a regular expression and** $L(R) = A$)

A language is called **regular** when any (hence all) of the above three conditions are met.

Review: Week 3 Wednesday

Please complete the review quiz questions on Gradescope about translating between DFA, NFA, and regular expressions.

Pre class reading for next time: Introduction to Section 1.4 (page 77)

Friday April 15

1. Recorded
2. HW1 grading
3. Project 1 released

Theorem: For an alphabet Σ, For each language L over Σ,

L is recognized by some DFA
iff
L is recognized by some NFA
iff
L is described by some regular expression

If (any, hence all) these conditions apply, L is called **regular**.

Prove or Disprove: There is some alphabet Σ for which there is some language recognized by an NFA but not by any DFA.

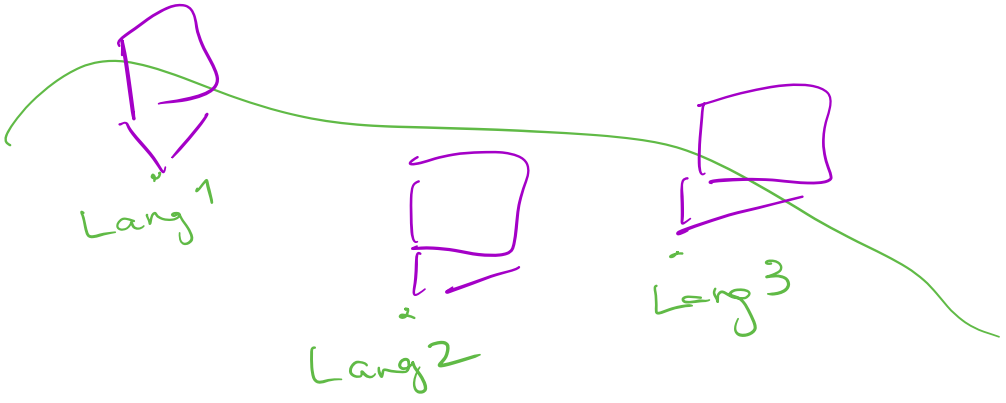
Let Σ be arb. alphabet. Let L be language over Σ recognized by an NFA, call it N so $L(N) = L$. By Wednesday's thm, there is a DFA M with $L(M) = L(N) = L$, so there's a DFA recognizing L.

Prove or Disprove: There is some alphabet Σ for which there is some finite language not described by any regular expression over Σ.

induction on size of set.

Prove or Disprove: If a language is recognized by an NFA then the complement of this language is not recognized by any DFA.

Let L be an arbitrary lang recognized by an NFA. Earlier, we saw this means L is also recognized by some DFA. Using thm from last week, can find another DFA that recognizes $\overline{L}$.



Set	Cardinality
$\{0,1\}$	Finite (2)
$\{0,1\}^*$	Countably infinite
$\{\emptyset, \{0\}, \{1\}, \{0,1\}\} = \mathcal{P}(\{0,1\})$	Finite (4)
$\mathcal{P}(\{0,1\}^*)$ The set of all languages over $\{0,1\}$	Uncountable set
"nice descriptions" The set of all regular expressions over $\{0,1\}$	countably infinite
The set of all regular languages over $\{0,1\}$	countably infinite $\mathcal{P}(\{0,1\}^*)$

$\Sigma = \{0,1\}$ Binary alphabet

$\Sigma^* = \{0,1\}^*$ The set of all binary strings. finite length

(string order)

$\varepsilon, 0, 1, 00, 01, 10, 11 \ldots$

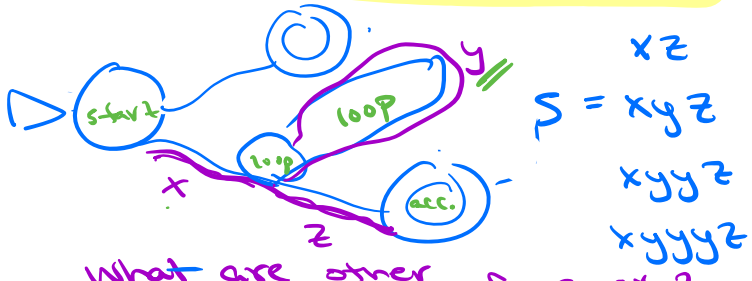
To prove L is regular, build a specific representation.

To prove (L) is nonregular, analyze properties of L.

The Pumping Lemma gives a property shared by all regular languages. It can be used to prove that a language is nonregular by showing that language does not have this property.

Pumping Lemma (Sipser Theorem 1.70): If A is a regular language, then there is a number p (a *pumping length*) where, if s is any string in A of length at least p, then s may be divided into three pieces, $s = xyz$ such that

- $|y| > 0$
- for each $i \geq 0$, $xy^iz \in A$
- $|xy| \leq p$.



True or False: A pumping length for $A = \{0,1\}^*$ is $p = 5$.

What are other pumping lengths for $\{0,1\}^*$?

For s any string in $\{0,1\}^*$ of length at least 5, s can be divided into three pieces $s = xyz$ such that $|y| > 0$, for each $i \geq 0$ $xy^iz \in \{0,1\}^*$, $|xy| \leq 5$

$|xy| = 0+1 = 1 \leq 5$ ✓ choose witness $x = \varepsilon$ $y = b$ $z = w$ write $s = bw$ where $b \in \{0,1\}$, $w \in \{0,1\}^*$

True or False: A pumping length for $A = \{1, 01, 001, 0001, 00001\}$ is $p = 4$.

Counterexample is $s = 00001$, a string in A with length ≥ 4 but for which we can't find x,y,z that work. Why? We show that if x,y,z satisfy $s=xyz$, $|y|>0$, $|xy|\leq p$ then there is some i with $xy^iz \notin A$.

Let x,y,z be arbitrary with $s=xyz$, $|y|>0$, $|xy|\leq p$. Consider $i = 2$. Then $|xy^2z| = |x| + 2|y| + |z| = |s| + |y| > |s|$ but s is longest string in A so $xy^2z \notin A$.

True or False: A pumping length for $A = \{0^j1 \mid j \geq 0\}$ is $p = 3$.

$s = 0^k1$

For s any string in $\{0^j1 \mid j \geq 0\}$ and $|s| \geq 3$ s can be divided into three pieces $s = xyz$ such that $|y| > 0$, for each $i \geq 0$ $xy^iz \in \{0,1\}^*$, ✓ $|xy| \leq 3$.

$x = \varepsilon$ $y = 0$ $z = 0^{k-1}1$

True or False: For any language A, if p is a pumping length for A and $p' > p$, then p' is also a pumping length for A.

extra practice

Review: Week 3 Friday

Please complete the review quiz questions on Gradescope about the class of regular languages.

Pre class reading for next time: Example 1.75, Example 1.77

Week 4 Monday+Wednesday on Zoom