Deterministic Finite Automaton

Textbook references: Within a chapter, each item is numbered consecutively. Figure 1.22 is the twenty-second numbered item in chapter one; it comes right after Example 1.21 and right before Definition 1.23.

In Computer Science, we operationalize "hardest" as "requires most resources", where resources might be memory, time, parallelism, randomness, power, etc. To be able to compare "hardness" of problems, we use a consistent description of problems

Input: String

Output: Yes/ No, where Yes means that the input string matches the pattern or property described by the problem.

So far: we saw that regular expressions are convenient ways of describing patterns in strings. DFA give a model of computation for processing strings and and classifying them into Yes (accepted) or No (rejected). We will see that each set of strings is described by a regular expression if and only if there is a DFA that recognizes it. Another way of thinking about it: properties described by regular expressions require exactly the computational power of DFAs.

Note: Build a DFA that recognizes $\{0,10\}$ means building a DFA that accepts 0 and accepts 10 and rejects all other strings

Build a regular expression that describes $\{0,10\}$ means only 0 and 10 match the pattern, i.e. the language of the regular expression has exactly these two elements.

Monday April 4

Review: Formal definition of DFA: $M = (Q, \Sigma, \delta, q_0, F)$

- Finite set of states Q NOTE: nonempty
- Alphabet Σ
- Transition function δ $\delta : Q \times \Sigma \to Q$
- Start state q_0 $q_0 \in Q$
- Accept (final) states F $F \subseteq Q$ specify as part of machines

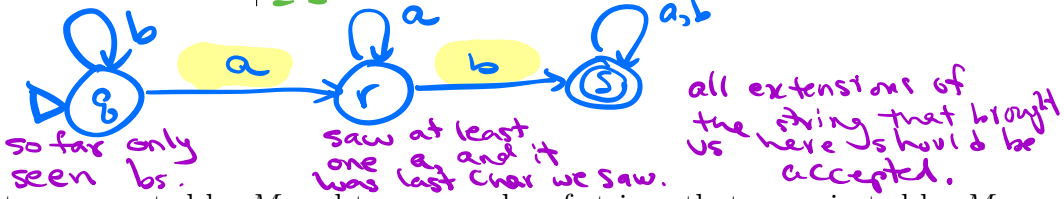
In the state diagram of M, how many outgoing arrows are there from each state? $|\Sigma|$ (including self-loops)

$M = (\{q, r, s\}, \{a, b\}, \delta, q, \{s\})$ where δ is (rows labelled by states and columns labelled by symbols): Σ, $|\Sigma| = 2$

δ	a	b
q	r	q
r	r	s
s	s	s

Q $\delta((q, a)) = r$

The state diagram for M is



Give two examples of strings that are accepted by M and two examples of strings that are rejected by M:

Accepts: q, q, r, s b ab ab Rejects: ε a b

Add "labels" for states in the state diagram, e.g. "have not seen any of desired pattern yet" or "sink state".

We can use the analysis of the roles of the states in the state diagram to describe the language recognized by the DFA.

$L(M) = \{w \in \{a,b\}^* \mid$ w contains at least one a followed by a b$\} = \{w \in \{a,b\}^* \mid$ ab is a substring of w$\}$

A regular expression describing $L(M)$ is

Trace paths that lead from q_0 to s $= \{w_1 abw_2 \mid w_1 \in \{a,b\}^*$ and $w_2 \in \{a,b\}^*\}$

$b^* a a^* b (a \cup b)^*$

Using semantic structure

$\Sigma^* ab \Sigma^*$

remembering Σ is shorthand $(a \cup b)$

example aab

$aab \in L(b^* a a^* b (a \cup b)^*)$

$aab \in L(\Sigma^* ab \Sigma^*)$

$\frac{\varepsilon}{b^*} \frac{a}{a} \frac{a}{a^*} \frac{b}{b} \frac{\varepsilon}{(a \cup b)^*}$

$\frac{a}{\Sigma^*} \frac{ab}{ab} \frac{}{\Sigma^*}$

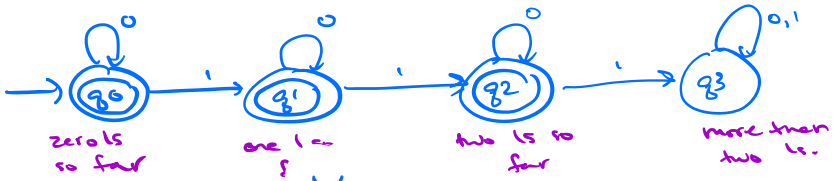
Let the alphabet be $\Sigma_1 = \{0, 1\}$.

X=

A state diagram for a DFA that recognizes $\{w \mid w$ contains at most two 1's$\}$ is

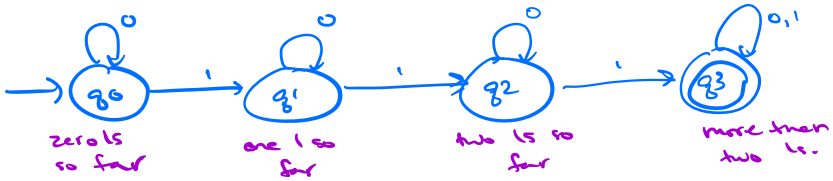
$Q = \{q_0, q_1, q_2, q_3\}$

$F_X = \{q_0, q_1, q_2\}$



Y=

A state diagram for a DFA that recognizes $\{w \mid w$ contains more than two 1's$\}$ is



$F_Y = \{q_3\}$.

Notice similarities of these two state diagrams

$Y = \overline{X} = \{w \in \Sigma^* \mid w \notin X\}$.

Extra example: A state diagram for DFA recognizing

$$\{w \mid w \text{ is a string over } \{0,1\} \text{ whose length is not a multiple of } 3\}$$

Let n be an arbitrary positive integer. What is a formal definition for a DFA recognizing

$$\{w \mid w \text{ is a string over } \{0,1\} \text{ whose length is not a multiple of } n\}?$$

Review: Week 2 Monday

Please complete the review quiz questions on Gradescope about the languages recognized by DFAs.

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).

Pre class reading for next time: Pages 45-47.

Wednesday April 6

Suppose A is a language over an alphabet Σ. By definition, this means A is a subset of Σ^*. **Claim:** if there is a DFA M such that $L(M) = A$ then there is another DFA, let's call it M', such that $L(M') = \overline{A}$, the complement of A, defined as $\{w \in \Sigma^* \mid w \notin A\}$.

Proof idea: Start with M, flip roles of accept & reject states.

Proof: Let A be an arbitrary language over Σ.
Assume there is DFA $M=(Q,\Sigma,\delta,q_0,F)$ for which $L(M)=A$.
We want to build M' so that $L(M')=\overline{A}$
Define witness $M' = (Q, \Sigma, \delta, q_0, \{q \in Q \mid q \notin F\})$
FORMAL DEF
Same set of states
Same alphabet
Same transition function
Same start state

Prove that the witness works

WTS "want to show"

WTS $L(M') = \overline{A}$

Goal ① WTS $L(M') \subseteq \overline{A}$. Let w be arbitrary string accepted by M'. By definition, this means that the computation of M' on w ends in a state (call it r) in the set of accept states of M', $\{q \in Q \mid q \notin F\}$. Consider the computation of M on w. Since the state diagrams of M and M' agree, this computation also ends in r. This computation rejects w because $r \in \{q \in Q \mid q \notin F\}$ so $r \notin F$. Thus $w \notin L(M)$ and since $L(M) = A$, $w \in \overline{A}$, as required for the subset inclusion.

Goal ② WTS $L(M') \supseteq \overline{A}$. Let w be arbitrary string in $\overline{A}$ and we want to show it will be accepted by M'. Since $A = L(M)$, $w \in \overline{A}$ means M rejects w. In other words the computation of M on w has as its last state a state that's not in F. Since the state diagrams of M and M' are identical, the computation of M' on w ends at this same state, which is in $\{q \in Q \mid q \notin F\}$, the set of accept states of M'. Thus, $w \in L(M')$, as required for this direction of subset inclusion. ∎

A useful (optional) bit of terminology: the **iterated transition function** of a DFA $M = (Q, \Sigma, \delta, q_0, F)$ is defined recursively by

$$\delta^*(\ (q,w)\ ) = \begin{cases} q & \text{if } q \in Q, w = \varepsilon \\ \delta(\ (q,a)\ ) & \text{if } q \in Q,\ w = a \in \Sigma \\ \delta(\ (\delta^*(q,u),a)\ ) & \text{if } q \in Q,\ w = ua \text{ where } u \in \Sigma^* \text{ and } a \in \Sigma \end{cases}$$

Using this terminology, M accepts a string w over Σ if and only if $\delta^*(\ (q_0,w)\ ) \in F$.

- Review Quiz -

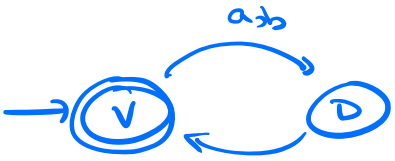
example string not in this language aba a ε
example strings in this language: ab aaab

Fix $\Sigma = \{a, b\}$. A state diagram for a DFA that recognizes $\{w \mid w \text{ has } ab \text{ as a substring and is of even length}\}$:

Notice this language is $\{w \mid w \text{ has } ab \text{ as a substring}\} \cap \{w \mid w \text{ has even length}\}$

M_1

$\{w \mid w \text{ has } ab \text{ as a substring}\}$



M_2

$L(M_2) = \{w \mid w \text{ has even length}\}$

Suppose A_1, A_2 are languages over an alphabet Σ. **Claim:** if there is a DFA M_1 such that $L(M_1) = A_1$ and DFA M_2 such that $L(M_2) = A_2$, then there is another DFA, let's call it M, such that $L(M) = A_1 \cap A_2$.

Proof idea: Run computations of M_1, M_2 in parallel, recording states in ordered pairs.

Formal construction:

Let A_1, A_2 be languages over Σ.

Suppose $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ is such that $L(M_1) = A_1$
and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ is such that $L(M_2) = A_2$

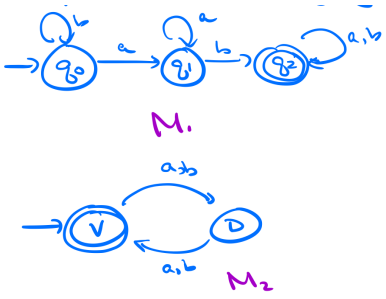
FORMAL DEFINITION

Define $M = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_2), F_1 \times F_2)$ where $\delta : (Q_1 \times Q_2) \times \Sigma \to Q_1 \times Q_2$

and $\delta((x, y), \sigma) = (\delta_1(x, \sigma), \delta_2(y, \sigma))$ for $(x, y) \in Q_1 \times Q_2$, $\sigma \in \Sigma$

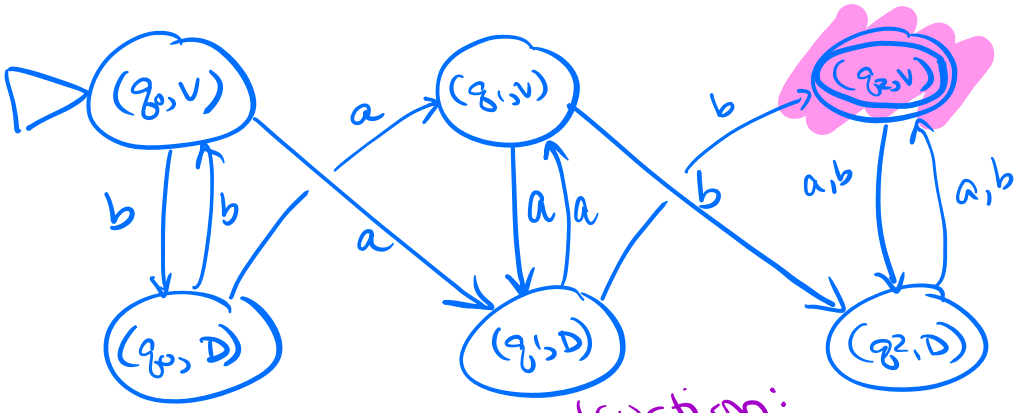
Application: When $A_1 = \{w \mid w \text{ has } ab \text{ as a substring}\}$ and $A_2 = \{w \mid w \text{ is of even length}\}$.

lower case sigma σ
upper case sigma Σ



M_1

M_2



Proof of correctness of formal construction:

WTS $L(M) = A_1 \cap A_2$.

Goal ① WTS every string accepted by M is in $A_1 \cap A_2$.

Consider an arbitrary string accepted by M, w.

By definition of M, the computation of M on w

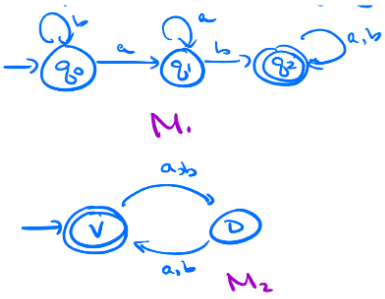
Continued on page 8.

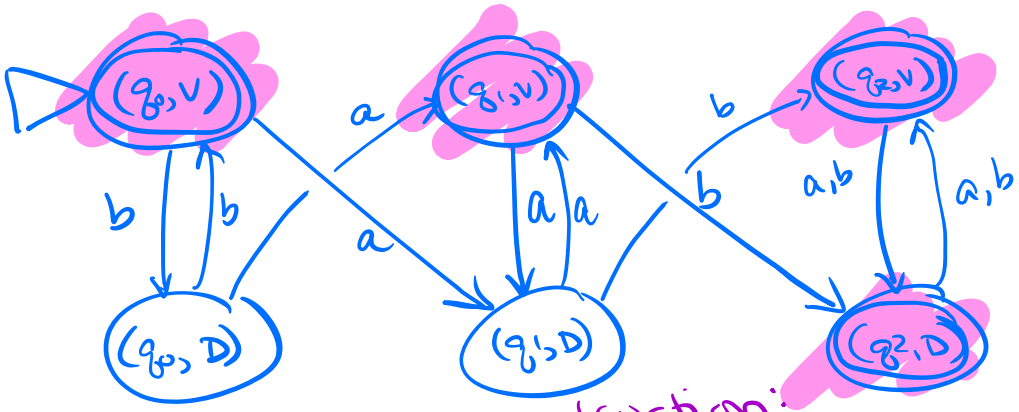
Suppose A_1, A_2 are languages over an alphabet Σ. **Claim:** if there is a DFA M_1 such that $L(M_1) = A_1$ and DFA M_2 such that $L(M_2) = A_2$, then there is another DFA, let's call it M, such that $L(M) = A_1 \cup A_2$.
Sipser Theorem 1.25, page 45

Proof idea: Just like intersection construction, run machines in parallel, but have different set of accept state.

Formal construction: Let A_1, A_2 be languages over Σ and suppose $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ is such that $L(M_1) = A_1$ and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ is such that $L(M_2) = A_2$. We define $M = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_2), (F_1 \times Q_2) \cup (Q_1 \times F_2))$ where $\delta : (Q_1 \times Q_2) \times \Sigma \to Q_1 \times Q_2$ is given by $\delta((x, y), \sigma) = (\delta_1(x, \sigma), \delta_2(y, \sigma))$ for $x \in Q_1, y \in Q_2, \sigma \in \Sigma$

Application: A state diagram for a DFA that recognizes $\{w \mid w$ has ab as a substring or is of even length$\}$:





Proof of correctness of formal construction:
(Left as exercise)

Review: Week 2 Wednesday

Please complete the review quiz questions on Gradescope about the languages recognized by DFAs.

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).

Pre class reading for next time: Introduction to Section 1.2

Continuation of proof of correctness of formal construction from page 6.

is a sequence of states of M determined by the transition function, and the last state is of the form (r_1, r_2) is in the set of accept states of M, so by definition $r_1 \in F_1$ and $r_2 \in F_2$.

The computation of M_1 on w is the sequence of states in the first component of the states in the computation of M on w. The last state in this sequence is r_2, which is in the set of accept states of M_1 so M_1 accepts w.

Similarly, the computation of M_2 on w is the sequence of states in the second component of states in the computation of M, and ends with r_2, which is in the set of accept states of M_2. Thus, M_2 also accepts w. Hence, we have that $w \in L(M_1) \cap L(M_2)$, and since $L(M_1) = A_1$ and $L(M_2) = A_2$, we have proved $w \in A_1 \cap A_2$, as required.

Goal ② WTS every string rejected by M is not in $A_1 \cap A_2$.

Consider an arbitrary string w rejected by M. As before, the computation of M on w can be decomposed into its components to give sequences of states that, by definition of δ, equal the computation of M_1 on w and the computation of M_2 on w. Since M reject w, at least one of the last states in these computations is not in the corresponding set of accept states of M_1 or M_2, so w is rejected by at least one of M_1 or M_2.

Thus $w \notin L(M_1) \cap L(M_2)$, and since $L(M_1) = A_1, L(M_2) = A_2$ we have proved $w \notin A_1 \cap A_2$, as required ∎

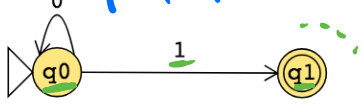
Note: an even more rigorous version of these proofs of correctness use induction on the number of steps in accepting computations.

Friday April 8

Nondeterministic finite automaton $M = (Q, \Sigma, \delta, q_0, F)$	
Finite set of states Q	Can be labelled by any collection of distinct names. Default: $q0, q1, \ldots$
Alphabet Σ	Each input to the automaton is a string over Σ.
Arrow labels Σ_ε	$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$. Arrows in the state diagram are labelled either by symbols from Σ or by ε
Transition function δ	$\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ gives the **set of possible next states** for a transition from the current state upon reading a symbol or spontaneously moving.
Start state q_0	Element of Q. Each computation of the machine starts at the start state.
Accept (final) states F	$F \subseteq Q$.
M accepts the input string	if and only if **there is** a computation of M on the input string that processes the whole string and ends in an accept state.

The formal definition of the NFA over $\{0,1\}$ given by this state diagram is: $(\{q_0, q_1\}, \{0,1\}, \delta, q_0, \{q_1\})$

where $\delta : \{q_0, q_1\} \times \{0, 1, \varepsilon\} \to \mathcal{P}(\{q_0, q_1\})$

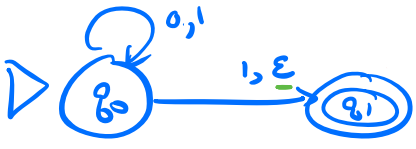


	0	1	ε
q_0	$\{q_0\}$	$\{q_0, q_1\}$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\emptyset$

The language over $\{0,1\}$ recognized by this NFA is:

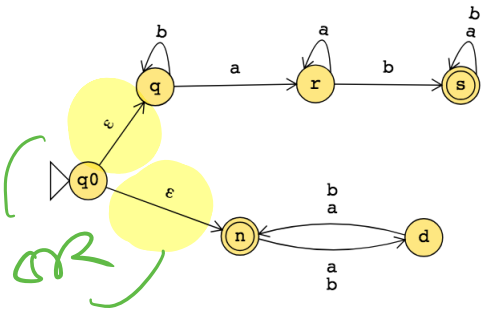
$\{w \in \{0,1\}^* \mid w \text{ ends in } 1\} = \{u1 \mid u \in \{0,1\}^*\} = L(\{0,1\}^*1)$

Change the transition function to get a different NFA which accepts the empty string.



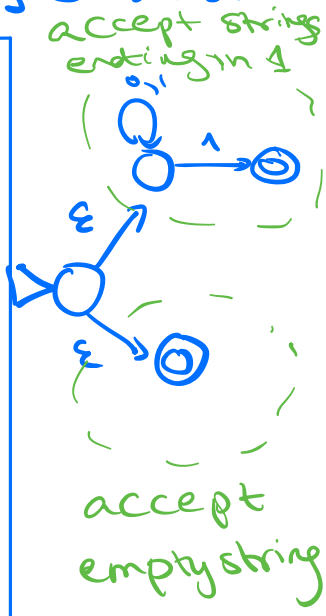
What about NFA recognizing $\{u1 \mid u \in \{0,1\}^*\} \cup \{\varepsilon\}$?

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

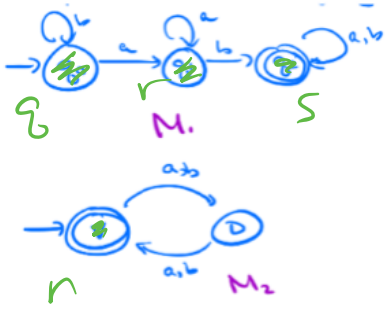


Simulate M_1

Simulate M_2



The language recognized by this NFA is:



$\{w \in \{a,b\}^* \mid w$ has ab as a substring or w has even length $\}$

Review: Week 2 Friday

Please complete the review quiz questions on Gradescope about NFA.

Pre class reading for next time: Theorem 1.47, Theorem 1.49