(Deterministic) Finite Automata

A finite automaton has a finite set of states with which it accepts or rejects strings.

A Deterministic Finite Automaton

A DFA has three components:

1. ***input tape*** contains single string;
2. ***head*** reads input string one symbol at a time; and
3. ***Memory*** is in one of a finite number of states.

Operating a DFA

Operating a DFA.

1) Set the machine to start state.

2) If End-of-String then halt.

3) Read a symbol.

4) Update state according to current state and symbol read.

5) Goto Step 2.

A DFA Accepts Strings

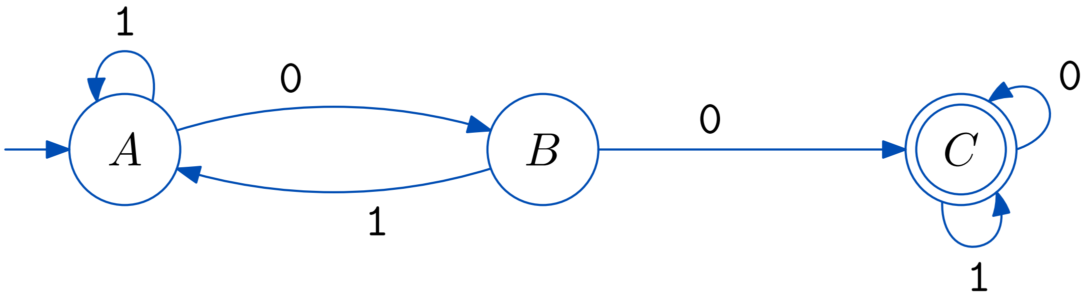
“Program” prescribes how symbols read affect current state.

Final state is state DFA is in when finished reading the input string.

There are ***accept*** states (double circle) and ***reject*** states.

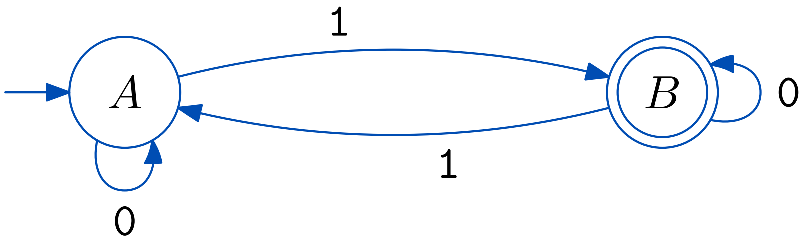
A DFA ***accepts*** input string if final state is accept state; otherwise it rejects.

An Example DFA



Final state for 101001 is C, final state for 11101 is A.

Example DFA



Accepts all strings of 0's and 1's with odd number of 1's.

Terminology

alphabet is a set of symbols (often denoted Σ)

language is a set of strings (***unary*** language means $|\Sigma| = 1$)

language of DFA is the set of strings it accepts

length of a string is the number of symbols

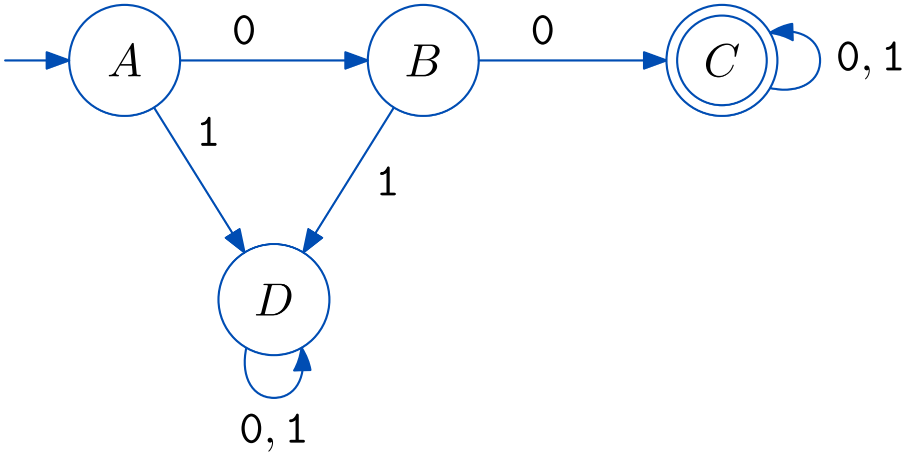
empty string is denoted ε.

Building DFAs: Do the Obvious

Starts with `00`:

Building DFAs: Do the Obvious

Starts with 00:

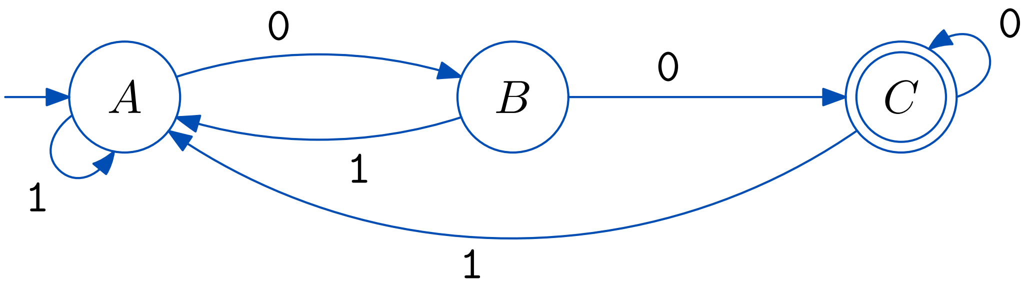


Building DFAs: Recent Memory

Ends with `00`:

Building DFAs: Recent Memory

Ends with `00`:



Building DFAs: Traps

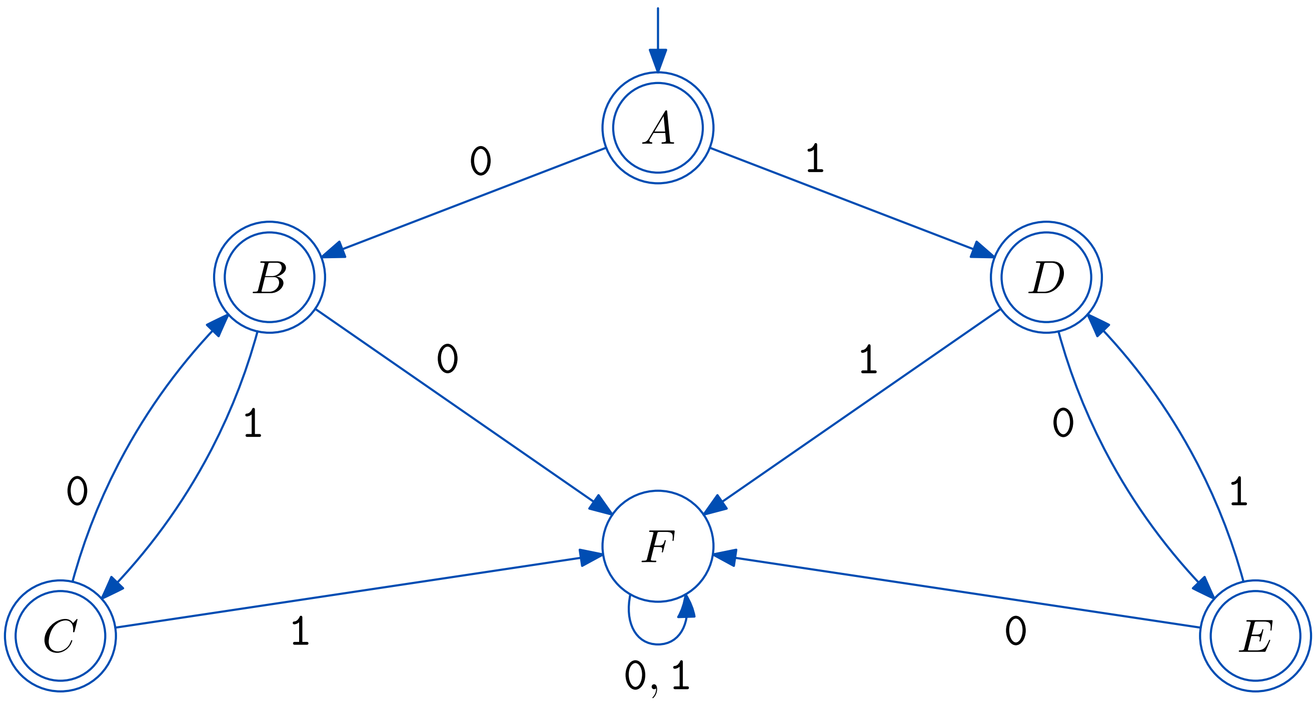
A ***trap*** is state that, once entered, one can never leave. Used to reject partly read strings that will never be accepted, or to accept partly read strings that will definitely be accepted.

Example with a Trap

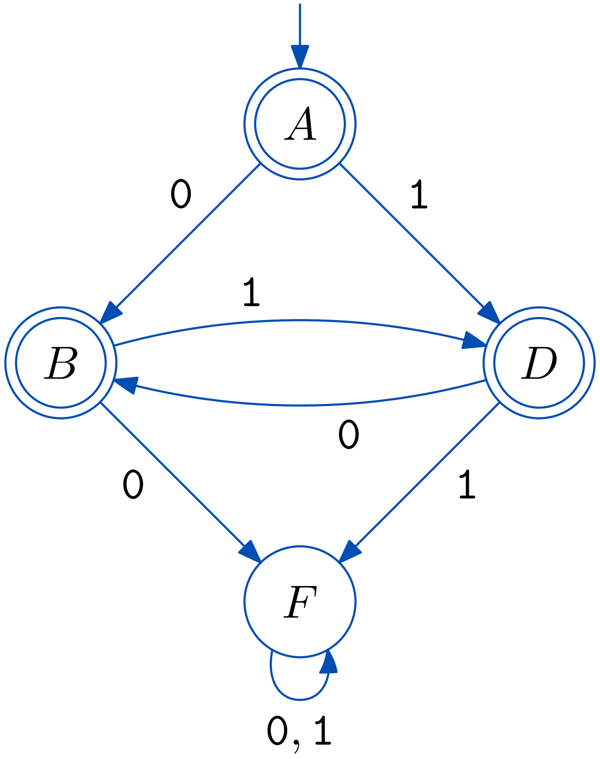
Alternating `0`’s and `1`’s:

Example with a Trap

Alternating 0's and 1's:



Alternating 0's and 1's again

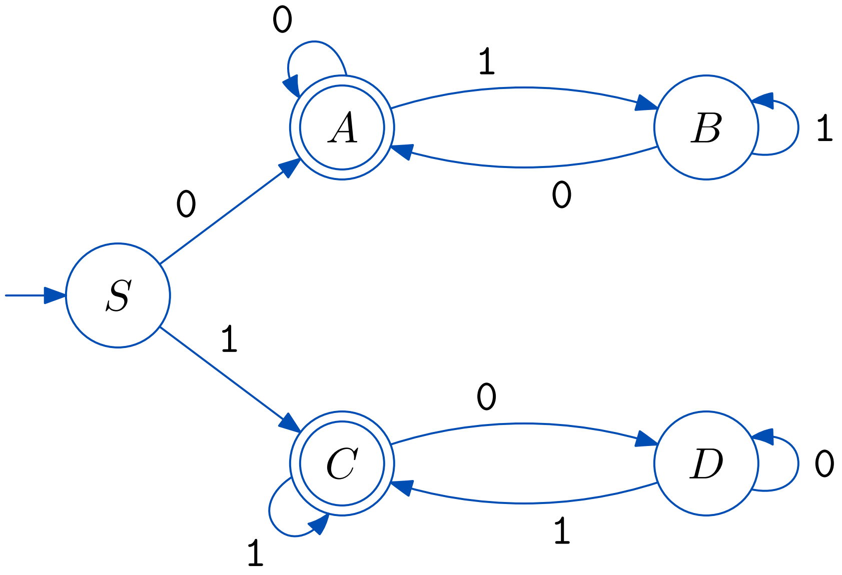


Building DFAs: Permanent Memory

A DFA remembers permanently by splitting into pieces. Here is one for first and last bit the same:

Building DFAs: Permanent Memory

A DFA remembers permanently by splitting into pieces. Here is one for first and last bit the same:



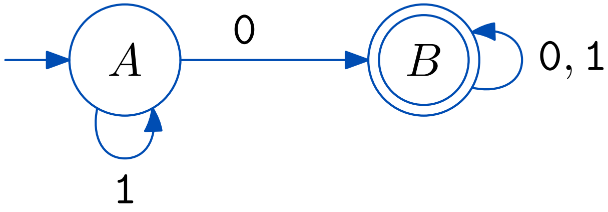
Practice

Give DFAs for each of the following three languages:

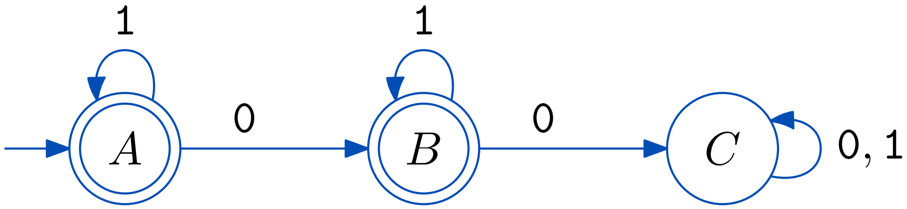
1. All binary strings with at least one 0
2. All binary strings with at most one 0
3. All binary strings starting and ending with 0 (and single-0 string counts)

Solutions to Practice

1)

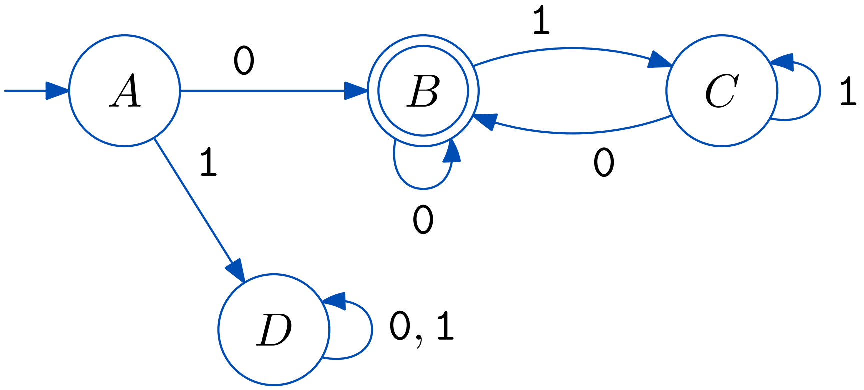


2)



Solutions to Practice

3)



Transition Table

A ***transition table*** lists new state given current state and symbol read. Here's one for DFA for all binary strings that begin and end with same symbol.

		Input	
		0	1
State	S	A	C
	A	A	B
	B	A	B
	C	D	C
	D	D	C

Formal Definition

A DFA is 5-tuple $(Q, \Sigma, q_0, T, \delta)$ where:

- Q is finite set of states;
- Σ is alphabet of input symbols;
- q_0 is start state;
- T is subset of Q giving the accept states; and
- δ is ***transition function*** that maps state and symbol to state. (Mathematically, $\delta\colon Q \times \Sigma \mapsto Q$.)

Summary

A deterministic finite automaton (DFA) is a device that recognizes a language (set of strings). It has finite memory and an input tape; each input symbol that is read causes the machine to update its state based on its current state and the symbol read. The machine accepts the input if it is in an accept state at the end of the string; otherwise, the input is rejected.