Foundations of Computing

Lecture 2

Arkady Yerukhimovich

January 19, 2023

Outline

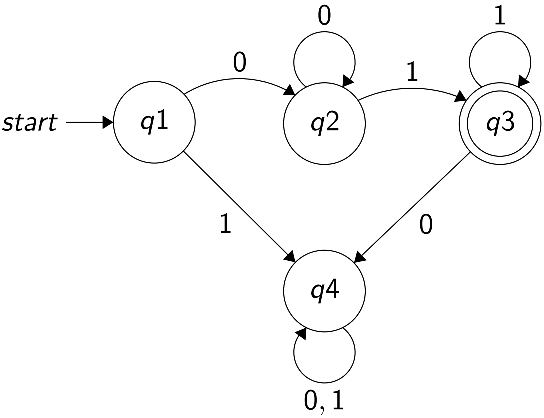
1. Lecture 1 Review
2. Quiz Solutions
3. Building DFAs
4. Proving Correctness of a DFA

Lecture 1 Review

- Syllabus review and course details
- Strings and languages
- Finite automata

Outline

Quiz Solutions



- Does M accept 00011?: Yes
- Does M accept 01100? No
- Describe the language $L(M)$: all strings with one or more 0s followed by one or more 1s

Outline

Important Rules of Deterministic Finite Automata

Deterministic Finite Automata

- Transition function must be fully defined:
  - For every state in Q, for every symbol in Σ, δ must specify a next state
- Transition function must be a function
  - For every state in Q, for every symbol in Σ, δ must specify exactly one next state

Important: Deterministic means that the execution of M on any input must be fully specified.

DFA as an Algorithm

DFA Execution

1. Read next input symbol and use transition function to determine next step until run out of input symbols
2. If stop in accept state, then output 1

Memory in a DFA:

- Each state stores a summary of the input seen so far
- Next state depends on this history and the next symbol
- Think of this as an "if" statement

Important

Since $|Q|$ is finite, input string may be longer than number of states

- Cannot just store the entire string

Example 1

Problem

Build a DFA that accepts

$$L = \{w | w \in \{0,1\}^* \text{ and } w \text{ contains the substring } 101\}$$

Building the DFA:

- Idea: State should store the part of 101 seen so far
- Transition function should change state depending on whether next symbol fits pattern

Observations:

- If see a 0:
  - this cannot be the first symbol of 101
  - but can be second character if previous symbol was a 1
- If see a 1:
  - this can be the first character of 101
  - or, it can be the last character if we previously saw 10 – in this case, we should accept

Example 1 – The Algorithm

Problem

Build a DFA that accepts

$$L = \{w | w \in \{0, 1\}^* \text{ and } w \text{ contains the substring } 101\}$$

Algorithm:

1. Start:
   - If read a 0, stay in step 1 – first symbol cannot be a 0
   - If read a 1, goto step 2 – record that we saw a 1
2. Step 2:
   - If read a 0, goto step 3 – record that we saw 10
   - If read a 1, stay in step 2 – may be first 1 of 101
3. Step 3:
   - If read a 0, goto step 1 – this is not 101, time to start over
   - If read a 1, goto step 4 – we have seen 101
4. Step 4:
   - On any input, stay in step 4 and accept

Build the DFA

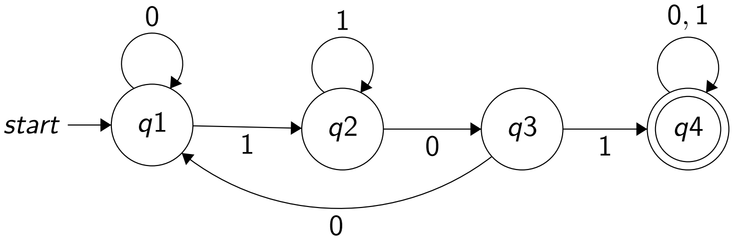
1. Start:
   - If read a 0, stay in step 1 – first symbol cannot be a 0
   - If read a 1, goto step 2 – record that we saw a 1
2. Step 2:
   - If read a 0, goto step 3 – record that we saw 10
   - If read a 1, stay in step 2 – may be first 1 of 101
3. Step 3:
   - If read a 0, goto step 1 – this is not 101, time to start over
   - If read a 1, goto step 4 – we have seen 101
4. Step 4:
   - On any input, stay in step 4 and accept

The DFA

Problem

Build a DFA that accepts

$$L = \{w|w \in \{0,1\}^* \text{ and } w \text{ contains the substring } 101\}$$

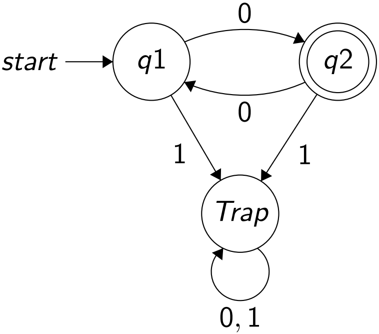


1. $q1$ – not yet read first 1 in 101
2. $q2$ – last input was a 1, could be start of 101
3. $q3$ – have read 10
4. $q4$ – have read 101

Trap States

A useful tool for designing DFAs:

- Trap states allow you to “reject” as soon as you know that $w \notin L$
- Trap states have no out edges – no way to get to accept



For convenience

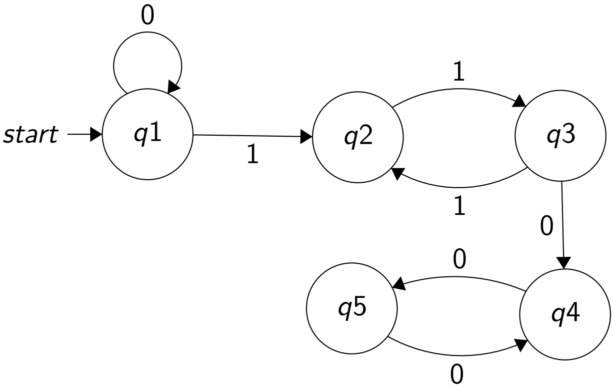
You can omit edges from transition diagram that point to the trap state

Example 2

Problem

Build a DFA that accepts:

$L = \{w | w \in \{0, 1\}^*$ and has even number (≥ 2) 1's followed by odd number of 0s$\}$

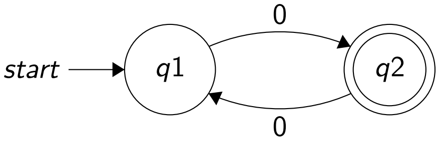


Outline

1. Lecture 1 Review
2. Quiz Solutions
3. Building DFAs
4. Proving Correctness of a DFA

Another Example

Consider the following DFA M



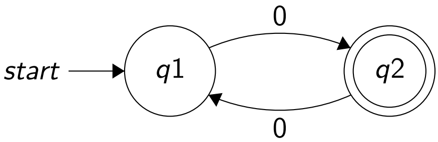
Theorem: This DFA recognizes

$$L = \{w \in \{0,1\}^* | w \text{ has odd number of 0s and no 1s}\}$$

Proof:

- Need to prove that $L = L(M)$
- Instead we prove the $L \subseteq L(M)$ and $L(M) \subseteq L$

$L \subseteq L(M)$



$$L = \{w \in \{0,1\}^* | w \text{ has odd number of 0s and no 1s}\}$$

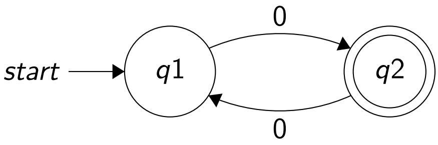
Definition: For string w, define $\delta(q1, w) =$ state M stops in on input w

Inductive Hypothesis

- For w of length k,
  if $\delta(q1, w) = q2$, then w has an odd number of 0s and no 1s
- For w of length k,
  if $\delta(q1, w) = q1$, then w has an even number of 0s and no 1s

Base Case: $|w| = 1$, If $\delta(q1, w) = q2$ then $w = 0$, so this is true

$L \subseteq L(M)$



$$L = \{w \in \{0,1\}^* | w \text{ has odd number of 0s and no 1s}\}$$

Inductive Hypothesis

- For w of length k,
  if $\delta(q1, w) = q2$, then w has an odd number of 0s and no 1s
- For w of length $k-1$,
  if $\delta(q1, w) = q1$, then w has an even number of 0s and no 1s

Induction:
For $|w'| = k+1$, $\delta(q1, w') = q2$ implies that $\delta(q2, w'_{1,\ldots,k}) = q1$.
Thus, by hypothesis, $w'_{1,\ldots,k}$ has an even number of 0s and no 1's and the last character of w' is a 0.

$L(M) \subseteq L$

Proof by contradiction:
Assume there exists a string w accepted by M that is not in L

- i.e., has an even number of 0's or a 1

Proof:

1. w cannot have a 1, as any such input will not stop in $q2$
2. By previous proof, any w that stops in $q2$ must have an odd number of 0s
3. Contradiction!