Nondeterministic Finite Automata
CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN

Chinese University of Hong Kong

Fall 2017
Example from last lecture with a simpler solution

Construct a DFA over alphabet \(\{0, 1\}\) that accepts all strings ending in 01

Three weeks later: DFA minimization
Another example from last lecture

Construct a DFA over alphabet \( \{0, 1\} \) that accepts all strings ending in 101

or

\[
\begin{array}{c}
\rightarrow \quad 0 \\
q_0 \\
\quad 1 \\
q_1 \\
\quad 0 \\
q_2 \\
\quad \ \ \ 0 \\
q_2
\end{array}
\]
String matching DFAs

Ending in 01

Ending in 101

Fast string matching algorithms to turn a pattern into a string matching DFA and execute the DFA:
Boyer–Moore (BM) and Knuth–Morris–Pratt (KMP)
(won’t cover in class)
Nondeterminism
In a few lectures

What problems can finite state machines solve?

We’ll answer this question in the next few lectures. Useful to consider hypothetical machines that are nondeterministic.
Even easier with guesses

Suppose we could guess when the input string has only 3 symbols left

Accept strings ending in 101:

This is not a DFA!
A machine that allows us to make guesses

Each state can have zero, one, or more outgoing transitions labeled by the same symbol.
Choosing where to go

State $q_0$ has two transitions labeled 1
Upon reading 1, we have the **choice** of staying at $q_0$ or moving to $q_1$
Ability to choose

State $q_1$ has no transition labeled 1
Upon reading 1 at $q_1$, die; upon reading 0, continue to $q_2$
Ability to choose

State $q_1$ has no transition going out
Upon reading 0 or 1 at $q_3$, die
Meaning of NFA

Guess if we are 3 symbols away from end of input

If so, guess we will see the pattern 101

Check that we are at the end of input
How to run an NFA

The NFA can have several active states at the same time. NFA accepts if at the end, one of its active states is accepting.
Example

Construct an NFA over alphabet \( \{0, 1\} \) that accepts all strings containing the pattern 001 somewhere

- 11001010, 001001, 111001 should be accepted
- \( \varepsilon, 000, 010101 \) should not
Example

Construct an NFA over alphabet \{0, 1\} that accepts all strings containing the pattern 001 somewhere
Definition

A nondeterministic finite automaton (NFA) is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\)
where

- \(Q\) is a finite set of states
- \(\Sigma\) is an alphabet
- \(\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \text{subsets of } Q\) is a transition function
- \(q_0 \in Q\) is the initial state
- \(F \subseteq Q\) is a set of accepting states

Differences from DFA:

- transition function \(\delta\) can go into several states
- allows \(\varepsilon\)-transitions
Language of an NFA

The NFA accepts string $x$ if there is some path that, starting from $q_0$, ends at an accepting state as $x$ is read from left to right.

The language of an NFA is the set of all strings accepted by the NFA.
\( \varepsilon \)-transitions can be taken for free:

- Accepts: \( a, b, aab, bab, aabab, \ldots \)
- Rejects: \( \varepsilon, aa, ba, bb, \ldots \)
Example

alphabet $\Sigma = \{0, 1\}$
states $Q = \{q_0, q_1, q_2\}$
initial state $q_0$
accepting states $F = \{q_2\}$

<table>
<thead>
<tr>
<th>states</th>
<th>0</th>
<th>1</th>
<th>$\varepsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$\emptyset$</td>
<td>${q_1}$</td>
<td>${q_1}$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>${q_0, q_1}$</td>
<td>$\emptyset$</td>
<td>${q_2}$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
</tbody>
</table>
Running NFA
Running NFA

001

101

11
What is the language of this NFA?
Example of $\varepsilon$-transitions

Construct an NFA that accepts all strings with an even number of 0s or an odd number of 1s.
Example of $\varepsilon$-transitions

Construct an NFA that accepts all strings with an even number of 0s or an odd number of 1s