COMS 381, Summer 2005
Supplementary Handout 1
Nondeterministic Finite Automata and Computation Trees

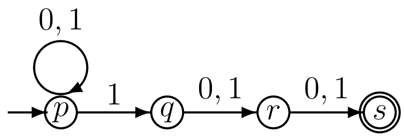
Nondeterministic computation is often thought of as a *guess and verify* process. For instance, suppose we have a NFA N that is given an input string x. We can think of the computation of N as a two-step process:

- 'guess' a sequence of transitions that will allow the processing of x by N to end in an accept state of N,
- 'verify' that guess by actually carrying out the transitions and checking that we end up in an accept state.

Nondeterminism can also be understood as a situation where there are multiple possible computation paths, which can be carried out in parallel if desired. A nondeterministic computation is accepting if at least one of these possible paths leads to an accepting state. If no such path exists, the computation is rejecting.

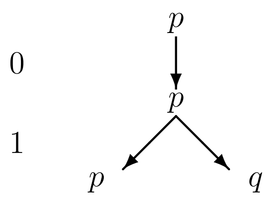
A standard way of visualizing the multiple possible computation paths for a NFA is a *computation tree*.

Consider the following simple NFA which accepts all strings over $\{0,1\}^*$ having a 1 in the third-to-last position:

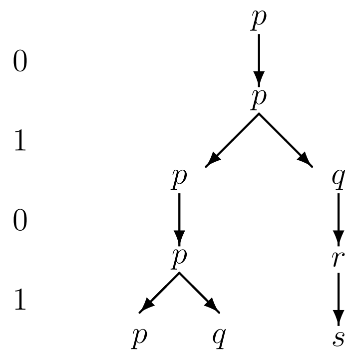


Consider the computation of the automaton on the string 0101 (which should be accepted).

We begin in state p. On the first 0, we remain in state p. On the next symbol, which is 1, we have a choice - to remain in state p or to move to q. This can be represented as a branch in the following (partial) computation tree:

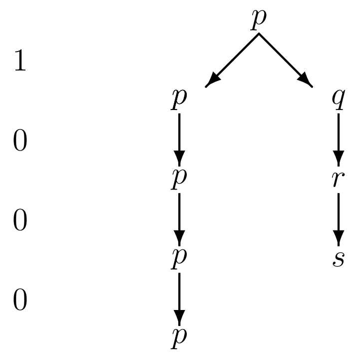


We can continue in this manner, until we finish processing the string. The full computation tree for 0101 is as follows:



Note that one of the paths in the tree leads to s, which is an accepting state.

And here is a computation tree for 1000, which is not accepted by the automaton.



Note two things:

1. At the bottom level of the tree, which is four steps below the root, there are no accepting states. Thus 1000 is rejected by our automaton.
2. One branch of the tree represents a computation that got 'stuck' after the transitions $p \to q \to r \to s$. Although this branch ends in an accept state, it does not represent an accepting computation because the entire input string 1000 has not been processed.

We have seen in class NFAs that have more than one start state. We can still draw computation trees for such automata; we simply need to add a 'dummy' start node which will be the root of our tree. The actual starting states will be the children of that node. Alternatively, this situation can be dealt with using epsilon-transitions, a topic covered in class and in Lecture 6 in our textbook.