Problem Session 3

Half(L)

Things More Powerful Than a Turing Machine

Some Concerns About Proofs

Half(L)

- If L is any language, Half(L) is the set of strings w such that for some string x, where |x| = |w|, wx is in L.
- If L is regular, so is Half(L).
- Construction: given a DFA A for L, we construct an ϵ-NFA B for Half(L).

Construction of NFA B

- States = pairs of states [p,q] of A, plus additional start state s_0.
- Intuition: If B reads input w, then p = $\delta_A(q_0, w)$.
  - q_0 = start state of A.
- q is any state such that there is some string x, with |x| = |w| such that $\delta_A(q, x)$ is an accepting state.

Accepting States of B

- Those pairs of the form [q, q].
- Notice: If B is in a state [q, q], then it has read some input w, such that $\delta_A(q_0, w) = q$ and there is some input x with $|x| = |w|$, such that $\delta_A(q, x)$ is an accepting state.
- That means wx is in L(A), and w is the first half of wx.

Transitions of B

- $\delta_B(s_0, \epsilon) = \{[q_0, f] \mid f$ is an accepting state of A$\}$.
- B never returns to s_0.
- First move guarantees that B is in the correct state after having read no input.
  - Notice: $[q_0, q_0]$ is an accepting state of B if and only if ϵ is in L(A).

Transitions of B – (2)

- $\delta_B([p,q], a) = \{[r, s]$ | such that:

1. $\delta_A(p, a) = r$.
2. There is some input symbol b such that $\delta_A(s, b) = q\}$.

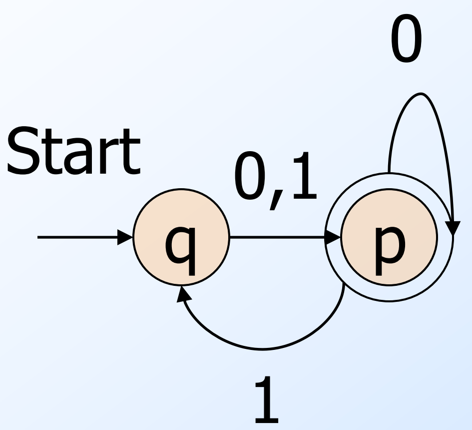
- (1) guarantees the first component continues to track the state of A.
- (2) guarantees the second component is any state that leads to acceptance via some string of length equal to input so far.

Inductive Proof That This Works

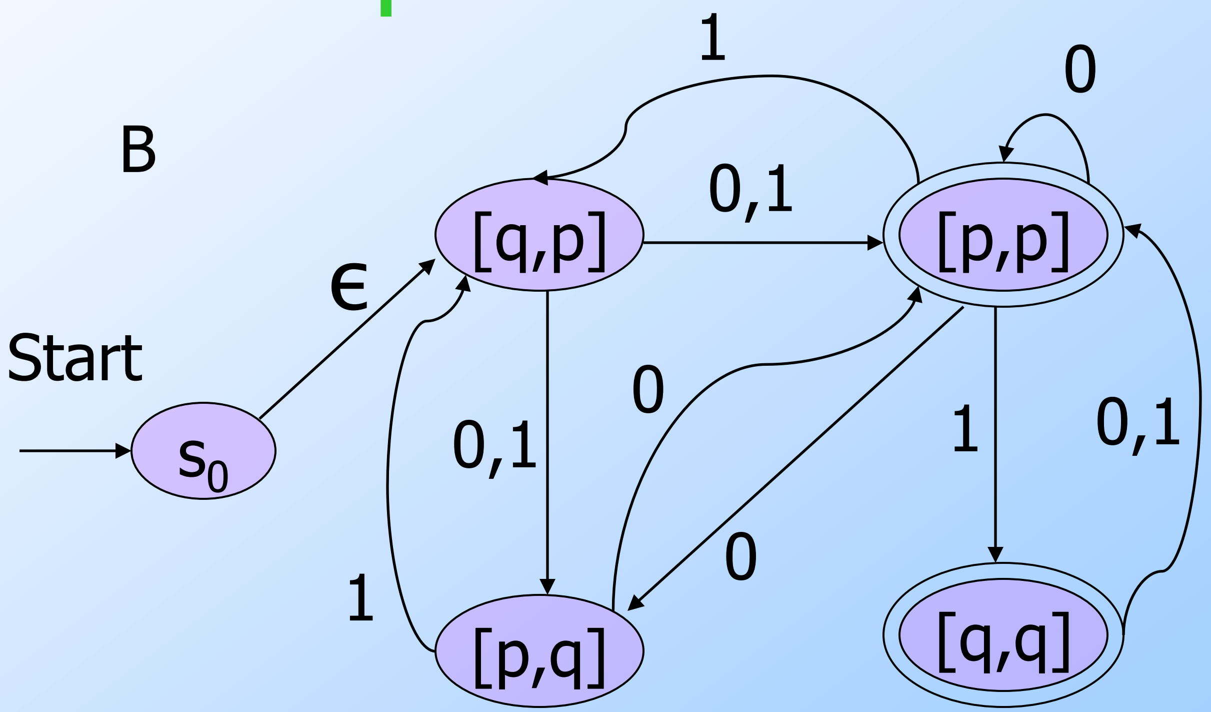
- By induction on $|w|$: $\delta_B([q_0, f], w) = \{[p, q] \mid$ such that:
  1. $\delta_A(q_0, w) = p$, and
  2. For some x, with $|x| = |w|$, $\delta_A(q, x) = f\}$.
- Complete the proof by observing the initial transitions out of s_0 to $[q_0, f]$, for accepting states f, and the definition of the accepting states of B.

Example

A



B



Is … More Powerful Than a Turing Machine?

- From an early post: "Can aspect systems do anything a Turing machine can't?"
- I don't know what an "aspect system" is.
- But if it is something that runs on a computer, then no.
- Why? because a Turing machine can simulate a real computer, and hence anything that runs on one.

What About Quantum Computers?

- People have imagined that there will be quantum computers that behave something like nondeterministic computers.
- There has been some progress by physicists on communication via quantum effects.

Quantum Computers – (2)

- The physics of quantum computers is suspect.
  - These would have to be enormous to isolate different bits of storage.
- But even if you had a quantum computer, it could still be simulated by a nondeterministic TM, and thus by a deterministic TM.

Can One PDA Stack Simulate Two?

- I claimed one could not, but I never proved it.
- If you try, you can't but that's no proof.
- Precise definition needed: A construction whereby one PDA P is constructed from two others, P_1 and P_2, so P accepts the intersection of the languages of P_1 and P_2.

Proof

- Assume such a construction exists.
- Let P_1 be a PDA that accepts $\{0^i1^j2^k \mid i=j\geq 1, k\geq 1\}$ and let P_2 be a PDA that accepts $\{0^i1^j2^k \mid j=k\geq 1, i\geq 1\}$.
- Then P would accept $L = \{0^i1^i2^i \mid i \geq 1\}$.
- But we know L is not a CFL, therefore has no PDA accepting it.

Proof – Continued

- We assumed only one thing: that we could construct P from P_1 and P_2.
- Since the conclusion, that L is a CFL, is known to be false, the assumption must be false.
- That is, no such construction exists.

Behind the Curtains of the Proof

- First, we assumed that if a statement S implies something false, then S is false.
- That seems to make sense, but it has to be an axiom of logic.
- Why? "proof" would be "by contradiction," thus using itself in its proof.
  - Aside: similarly, a "proof" that induction works requires an inductive proof.

Behind the Curtains – (2)

- We also made another assertion: the assumption “you can simulate two stacks with one” was the only unproved part of the proof, and therefore at fault.
  - Argument used many times in the course.
- But there were many other steps, some glossed over or left for your imagination.

Proofs as a Social Process

- If there were another unproved point, then my proof of "one stack can't simulate two" would not be valid.
- But proofs are subject to discussion and argument.
- If someone has a point they doubt, they can bring it up and it will be resolved one way or the other.

Aside: Social Processes – (2)

- Many years ago, Alan Perlis, Rich DeMillo and Dick Lipton published a paper arguing:
  - Proofs can only be believed because smart mathematicians will examine them and find flaws if they exist.
  - Proofs of program correctness are boring, and no one will bother to examine them.