CS 273: Intro to Theory of Computation, Fall 2007
Quiz 2 Solutions

1. (1 points) A PDA is basically an NFA with one extra feature. What's the extra feature?

   Solution: a stack.

2. (2 points) Is the language $\{a^n b^m \mid n \in \mathbb{N}, m \in \mathbb{N}\}$ regular? Why or why not?

   Solution: Yes, because the number of a's doesn't have to match the number of b's, so this language is just a^+b^+.

   It's a mistake to argue that it must be regular because it satisfies the pumping lemma. The pumping lemma only works one direction: if it's regular, it satisfies the pumping lemma. There are some non-regular languages that also satisfy the pumping lemma.

   Some people thought it was a^*b^*, because they forget that we are defining $\mathbb{N}$ not to contain zero. (I didn't take off points for this.)

3. (4 points) Give a context-free grammar that generates each of the following subsets of $\{a, b, c\}^*$. Just give the rule(s) for each grammar. Assume that the start symbol is always S and variables are uppercase letters.

   (a) $\{a^n b^n \mid n \geq 0\}$

   Solution: $S \rightarrow aSb \mid \epsilon$

   (b) $c^* \cup b^*$

   Solution:

   $$S \rightarrow B \mid C$$
   $$B \rightarrow bB \mid \epsilon$$
   $$C \rightarrow cC \mid \epsilon$$

   Many people had grammars that generated closely-related languages. An especially common mistake was to generate $(c \cup b)^*$ in part (b).

4. (2 points) What language is generated by the following context-free grammar, with start symbol S?

   $$S \rightarrow AA$$
   $$A \rightarrow a \mid b$$

   Solution: $\{aa, ab, ba, bb\}$

5. (4 points) Give an example of a context-free grammar G which is ambiguous. Explain why it is ambiguous by showing two parse trees yielding the same string.

Solution: There are quite a variety of ambiguous grammars you could have chosen from. A particularly short one is:

$$S \to S - S \mid a$$

Here are two parse trees for the string a+a+a

```
   S
  /| \
 / |  \
S  -   S
|     /|\
a    / | \
    S  -  S
    |     |
    a     a
```

```
      S
     /| \
    / |  \
   S  -   S
  /|\     |
 / | \    a
S  -  S
|     |
a     a
```

Notice that the question asked for parse trees, not derivations. They are closely related, but not the same thing.

One of the shortest solutions was a variant of this idea:

$$S \to a \mid A$$
$$A \to a$$

Many people also got good mileage about variables that could optionally expand into the empty string.

6. (4 points) Define what it means for a function mapping strings to strings to be a homomorphism.

   Solution: This turns out to have been a really hard question.

   A function h on strings is a homomorphism if $h(x)h(y) = h(xy)$ for all strings x and y.

   Or, equivalently, a function h is a homomorphism if its output on a string is the concatenation of its outputs on individual characters. That is, $h(c_1c_2 \dots c_n) = h(c_1)h(c_2)\dots h(c_n)$. for any string $c_1c_2 \dots c_n$).

   Saying that a homomorphism maps each character to a string isn't a technically complete answer, but it captures much of the idea.

7. (4 points) We showed in class that the language $B = \{0^n1^n \mid n \geq 0\}$ is not regular. Let $L = \{2^i0^n1^n \mid n \geq 0, i \geq 0\}$. Prove that L is not regular using closure properties and the fact that B isn't regular.

   Solution: Suppose that L were regular. Then $L \cap O^*1^*$ would be regular, because the intersection of two regular languages is regular. But $L \cap O^*1^*$ is just B, which we know not to be regular. This is a contradiction. So we must have been wrong in our assumption that L was regular.

   Or, suppose that L were regular. Let h be a homomorphism which maps 2 to ϵ, and 0 and 1 to themselves. Then $h(L)$ must also be regular, because regular languages are closed under homomorphism. But $h(L)$ is B, which we know not to be regular. This is a contradiction. So we must have been wrong in our assumption that L was regular.

   Many people wrote proofs which relied on the idea that applying some operation (e.g. intersection) to a regular language and a non-regular language yields a non-regular language. This isn't true. A correct answer to this kind of question normally requires a proof by contradiction.

   Another common wrong approach involved claiming that $L \cap B$ must be regular because regular languages are closed under intersection. However, B isn't regular and the inputs to the closure properties must both be regular.

8. (4 points) Finish the following statement of the pumping lemma.

   If A is a regular language, then there is a number p such that, if s is any string in A of length at least p, then ...

   Solution: ...there are strings x, y and z such that

   - $s = xyz$
   - $|xy| \leq p$
   - $|y| \geq 1$, and
   - $xy^iz \in A$ for every $i \in \mathbb{N}$.

   Each of the four conditions was worth 1/4 of the points.