# University of Central Florida
# School of Computer Science
# COT 4210 Fall 2004

## Prof. Rene Peralta
## Homework 5 Solutions (by TA Robert Lee)

**1.** (2.23 from textbook) Give an example of a language that is not context-free, but that does satisfy the three conditions of the pumping lemma. Prove that your example works. (See the analogous fact for regular languages in Problem 1.37.)

**Solution** Let

$$L = \{a^h b^i c^j d^k \mid h, i, j, k \geq 0 \text{ and if } h = 1, \text{ then } i = j = k.$$

Note that the language $L$ is not context-free because, for strings in $L$ of the form $ab^n c^n d^n, n \geq 0$, a PDA can match the counts of at most two of the three characters $b, c, d$. In fact, if a PDA could recognize these strings, then it could be easily modified (by removing the state that recognizes the initial $a$) to recognize the strings in $\{b^n c^n d^n \mid n \geq 0\}$, which we know is not context-free. (Keep in mind that this is not a formal proof that $L$ is not context-free.)

However, $L$ satisfies the three conditions of the pumping lemma for context-free languages. Split the strings in $L$ into two groups and consider each group separately.

**Case 1:** strings in $L$ of the form $s = ab^n c^n d^n, n \geq 0$. We choose to divide $s$ into five pieces $s = uvxyz$ as follows: $u = \epsilon, v = a, x = \epsilon, y = \epsilon, z = b^n c^n d^n$. Notice that $|vy| = 1 > 0$ and $|vxy| = 1$, and 1 is the minimum possible pumping length, so the condition $|vxy| \leq p$ is satisfied no matter what the pumping length might be. Now the string $uv^i xy^i z = a^i b^n c^n d^n$, which is in $L$ for all $i \geq 0$ because, if $i = 1$, then the number of $b$'s, $c$'s, and $d$'s is the same; if $i \neq 1$, then the numbers of each character are arbitrary. Thus all the conditions of the pumping lemma are satisfied in this case.

**Case 2:** strings in $L$ of the form $s = a^e b^f c^g d^h, e \neq 1, f, g, h \geq 0$. We choose to divide $s$ into five pieces $s = uvxyz$ as follows: $u = a^e, v = b, x = \epsilon, y = \epsilon, z = b^{f-1} c^g d^h$. Notice that $|vy| = 1 > 0$ and $|vxy| = 1$, and 1 is the minimum possible pumping length, so the condition $|vxy| \leq p$ is satisfied no matter what the pumping length might be. Now the string $uv^i xy^i z = a^e b^{f+i-1} c^g d^h$, which is in $L$ for all $i \geq 0$ because $e \neq 1$. Thus all the conditions of the pumping lemma are satisfied in this case.

Therefore language $L$ satisfies the conditions of the pumping lemma for context-free languages even though it is not context-free. Note that this does not contradict the pumping lemma, because the pumping lemma does not guarantee that a language satisfying its three conditions must be context-free.

**2.** (3.1c from textbook) This exercise concerns TM $M_2$, whose description and state diagram appear in Example 3.4 (on page 131 of the textbook). Give the sequence of configurations that $M_2$ enters when started on 000.

**Solution**

$q_1000$
$\sqcup q_200$
$\sqcup xq_30$
$\sqcup x0q_4\sqcup$
$\sqcup x0 \sqcup q_{reject}$

(Note that the blank symbol $\sqcup$ that appears at the end of the fourth line marks the end of the input.)

**3.** (3.19 from textbook) Let $A$ be the language containing only the single string $s$, where

$$s = \begin{cases} 0 & \text{if God does not exist} \\ 1 & \text{if God does exist.} \end{cases}$$

Is $A$ decidable? Why or why not? (Note that the answer doesn't depend on your religious convictions.)

**Solution** Yes, $A$ is decidable. Proof by cases.

**Case 1:** $s = 0$. We can design a Turing Machine that says "yes" if the input string is 0, and says "no" otherwise.

**Case 2:** $s = 1$. We can design a Turing Machine that says "yes" if the input string is 1, and says "no" otherwise.

**4.** (3.5 from textbook) Examine the formal definition of a Turing Machine to answer the following questions, and explain your reasoning.

**a.** Can a Turing machine ever write the blank symbol $\sqcup$ on its tape?

**Solution** Yes. The blank symbol $\sqcup$ is part of the tape alphabet.

**b.** Can the tape alphabet $\Gamma$ be the same as the input alphabet $\Sigma$?

**Solution** No. The input alphabet does not contain the blank symbol $\sqcup$.

**c.** Can a Turing Machine's head *ever* be in the same location in two successive steps?

**Solution** Yes. If the head is at the left end of the tape and the Turing Machine tries to move left, the head will remain in the same location.

**d.** Can a Turing Machine contain just a single state?

**Solution** No. A Turing Machine must have at least two states: an accept state and a (different) reject state.

**5.** Construct a TM to decide the language $L = \{a^i b^i a^i \mid i \geq 0\}$.

**Solution** See the state diagram on the next page.

1. $q_0 \to q_8$ transition: if the first character is $\sqcup$, accept. (This accepts the empty string.)

2. $q_0 \to q_1$ transition: mark the first $a$ (in the first section of $a$'s) with an X.

3. $q_1$ self-loop: if $a$ or Y is read, move right; repeat until a $b$ is read. (This skips the remaining unmarked $a$'s and any Y's.)

4. $q_1 \to q_2$ transition: mark the first unmarked $b$ with an Y.

5. $q_2$ self-loop: if $b$ or Z is read, move right; repeat until an $a$ is read. (This skips the remaining unmarked $b$'s and any Z's.)

6. $q_2 \to q_3$ transition: mark the first $a$ (in the second section of $a$'s) with a Z.

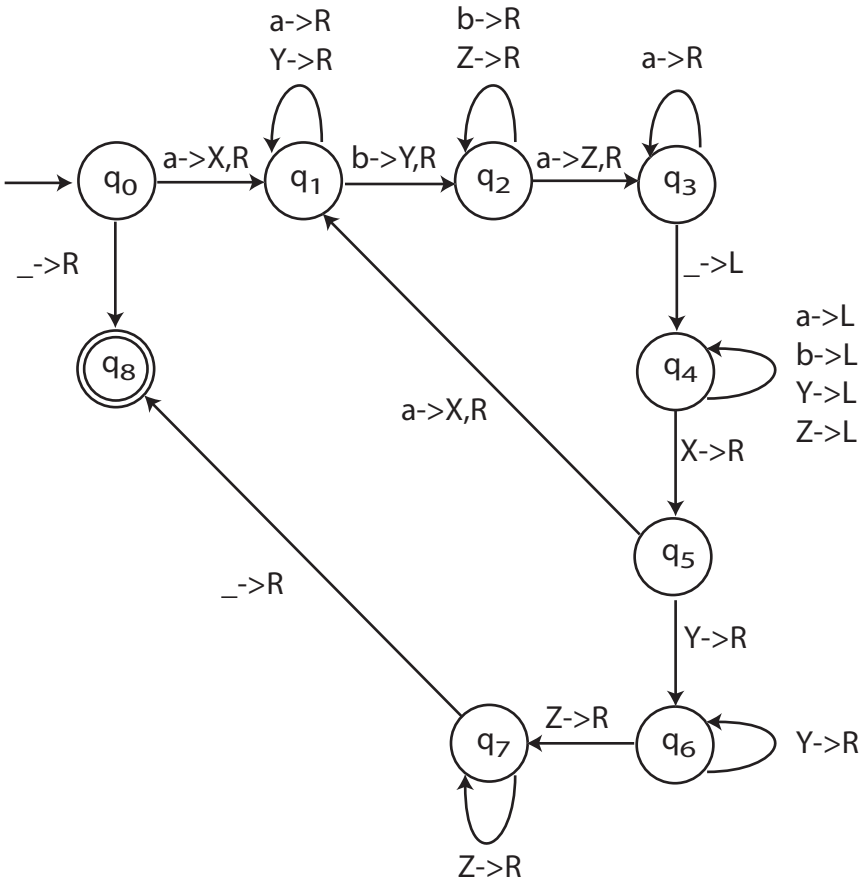
7. $q_3$ self-loop: if $a$ is read, move right; repeat until a $\sqcup$ is read. (This skips the remaining unmarked $a$'s.)

8. $q_3 \to q_4$ transition and $q_4$ self-loop: move the head left until an X is read; note that this is the right-most X.

9. $q_5 \to q_1$ transition: there is still an unmarked $a$ in the first section. Mark that $a$ with an X, and go to step 3.

10. Transitions from $q_5$ to $q_8$: there are no unmarked $a$'s left in the first section. Thus there cannot remain any unmarked characters in the rest of the string; if $Y$'s are now followed by $Z$'s and then a $\sqcup$, accept the string.

5. Turing Machine that decides $\{a^i b^i a^i \mid i >= 0\}$. (See explanation on previous page.)
Note that $q_{reject}$ is not shown, and $q_{accept} = q_8$.



**6.** Does there exist a decidable language that is not recognizable by a pushdown automaton with two stacks? Justify your answer.

**Solution** No. By Definition 3.3, a language is decidable if a Turing Machine decides it. We will show that every Turing Machine has an equivalent two-stack PDA.

Given TM $M = (Q_M, \Sigma_M, \Gamma_M, \delta_M, q_0, q_{accept}, q_{reject})$. Let the corresponding two-stack PDA $P = (Q_P, \Sigma_P, \Gamma_1, \Gamma_2, \delta_P, q_0, F)$, where

$Q_P = Q_M$ (and the states of $P$ are related to the states of $M$ by the identity mapping, i.e., they have the same names);

$\Sigma_P = \Sigma_M$ (the input alphabet of $P$ is the same as the input alphabet of $M$);

$\Gamma_1 = \Gamma_2 = \Gamma_M$ (the stack alphabets for both stacks of $P$ are the same as the tape alphabet of $M$);

the start states of $M$ and $P$ are the same;

and $F = \{q_{accept}\}$ (the only accept state in $P$ is $q_{accept}$).

Stack 1 will represent the TM tape to the left of the read/write head; stack 2 will represent the TM tape to the right of the read/write head. The following is a simple diagram of our strategy to make $P$ behave like $M$:

| 1 | k | c | a | t | S | (Simulated) Read/Write Head → S | t | a | c | k | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|

Note that stack 1 grows to the left, while stack 2 grows to the right. $P$ does not have a real read/write head, so we simulate the read/write head as being positioned on the first character of stack 2.

We will use the notation push1($x$) to mean pushing symbol $x$ onto stack 1; the notations pop1($x$), push2($x$), and pop2($x$) have corresponding meanings.

First load the input by pushing the input string onto stack 1. The order of the input string will be reversed due to the first-in-last-out nature of a stack, so we next pop it off stack 1 and push it onto stack 2. Now the initial condition of $M$ is simulated: the read/write head is at the left end of the input.

To determine the transition function of $P$, we will consider each type of rule in $M$ separately.

For the $x \to R$ rules, $P$ will pop2($x$) and push1($x$); if stack 2 is empty, $M$ would be at the right end of the input, so $P$ will push1(␣).

For the $x \to y, R$ rules, $P$ will pop2($x$) and push1($y$); if stack 2 is empty, $M$ would be at the right end of the input, so $P$ will push1($y$).

For the $x \to L$ rules, $P$ will pop1($x$) and push2($x$); if stack 1 is empty, then $M$ would be at the left end of the input, so $P$ will not do anything.

For the $x \rightarrow y, L$ rules, $P$ will pop2($x$), push2($y$), pop1($z$), and push2($z$), where $z$ is the first character of stack 1; if stack 1 is empty, then $M$ would be at the left end of the input, so $P$ will pop2($x$) and push2($y$).

The given language is decidable, so $M$ must halt in either $q_{accept}$ or $q_{reject}$; therefore $P$ must also halt in the corresponding state. If $P$ halts in $q_{accept}$, it accepts the string, because $q_{accept} \in F$; otherwise $P$ will reject the string.

Therefore $P$ simulates the behavior of $M$, and it follows that any decidable language can be recognized by a two-stack PDA.