CSCE 428/828 HW 4

due in class Monday 4/5/2017
1. (20 pts) Design a standard Turing machine (i.e., semi-infinite (infinite on the right direction) tape, deterministic) to recognize the following language. Draw the low-level state transition diagram of your TM. You do not need to describe your high-level algorithm.

$$\{ x \text{ over } \{a, b\} \mid x \text{ has at most two } b's \}$$

Wrong solution: This machine recognizes strings with exactly two b's.

Correct Solution: This machine recognizes strings with at most two b's.
2. (20 pts) Let a $k$-PDA be a pushdown automaton with $k$ stacks. Thus, a 0-PDA is an NFA and a 1-PDA is a conventional PDA. Show that the model of 2-PDAs has the same power as the model of standard Turing machines. That is, prove that for any 2-PDA we can construct an equivalent TM, and for any TM we can construct an equivalent 2-PDA.

**PROOF:**

(a) First, we prove that a 2-PDA can simulate a TM.

A 2-PDA can record the tape of a TM on two stacks. Stack 1 stores the characters on the left of the head, with the bottom of stack storing the left-most character on the tape. Stack 2 stores the characters on the right of the head, with the bottom of the stack storing the right-most non-blank character on the tape. In other words, if a TM’s configuration is $aq_ib$, the corresponding 2-PDA is in state $q_i$, with stack 1 storing the string $a$ from bottom to top, and stack 2 storing the string $b$ from top to bottom.

For each transition $\delta(q_i, c_i) = (q_j, c_j, L)$ of the TM, the corresponding PDA transition pops $c_i$ off stack 2, pushes $c_j$ onto stack 2, then pops stack 1 and pushes the character onto stack 2, and goes from state $q_i$ to $q_j$. For each transition $\delta(q_i, c_i) = (q_j, c_j, R)$ in the TM, the corresponding PDA transition pops $c_i$ off stack 2, pushes $c_j$ onto stack 1, and goes from state $q_i$ to $q_j$.

There are two special cases. First, if stack 1 is empty (which happens when the tape head is at the left-most cell of the tape) and a transition moves the head to the left, then stack 2 should be updated accordingly and nothing happens to stack 1. Second, if stack 2 is empty (which happens when the tape head is on a blank cell which is on the right-hand side of the string on the tape) and a transition moves the head to the right, then stack 1 should be updated accordingly and nothing happens to stack 2.

(b) Second, we prove that a standard TM can simulate a 2-PDA. A standard TM can simulate a 2-tape non-deterministic TM, which can simulate a 2-PDA. Thus a standard TM can simulate a 2-PDA.
3. (20 pts) Show that the class of Turing-recognizable languages is closed under the operation of star. You only need to give a high-level description (i.e. an algorithm or pseudo-code) of your machine.

**PROOF:**

For a Turing-recognizable language $L$, let $M$ be the TM that recognizes it. We construct a nondeterministic TM $M'$ that recognizes the star of $L$ as follows.

1. Nondeterministically decompose $w$ into parts so that $w = w_1w_2...$. Since $w$ has a finite number of symbols, there is only a finite number of decompositions. Different copies of the machine will check different decompositions.
2. Each copy of the machine runs $M$ on $w_i$ for every $i$. If $M$ accepts all of them, then this copy accepts. (Optional: If $M$ halts and rejects any of them, then this copy rejects.)

If there is a way to decompose $w$ into substrings such that $M$ accepts all the substrings, $w$ belongs to the star of $L$, and $M'$ will accept $w$ after a finite number of steps.
4. (20 pts) The following shows the state transition diagram of a Turing machine for language \( A = \{ a^{2^n} \mid n \geq 0 \} \). For example, strings \( a, aa, aaaa, \) and \( aaaaaaa \) are members of language \( A \). Please modify this Turing machine to recognize language \( B = \{ a^{(2^n)-1} \mid n \geq 0 \} \). For example, strings \( \epsilon, a, aaa, \) and \( aaaaaaa \) are members of language \( B \). You can modify, add, or remove transitions, but you cannot add or remove any states.

Solution 1:
Solution 2:
5. **(EXTRA CREDIT: 20 pts)** Design a Turing machine with two tapes for the palindrome language over alphabet \( \{a, b\} \). Give both the high-level description and the low-level state transition diagram, and analyze how many steps it takes to accept a palindrome. If a palindrome has \( n \) symbols, your TM should accept it with \( O(n) \) steps (one transition is considered as a step).

Assume that initially an input string is on the first tape and the second tape is empty. Use the following transition notation: \((a, b) \rightarrow (c, d), (L, R)\) means that if the current symbol on the first tape is \(a\) and the current symbol on the second tape is \(b\), then for the first tape we replace \(a\) with \(c\) and move the tape head to the left, and for the second tape we replace \(b\) with \(d\) and move the tape head to the right. You may use the stay option, if needed.

**Solution:**

With two tapes, we can first make a reverse copy of an input string on the second tape, and then compare whether two tapes are same. It takes \( n \) steps to move the first tape head to the last symbol of a palindrome, \( n \) steps to make a reverse copy, \( n \) steps to move the second tape heads back to the the left-most cell, and finally \( n \) steps to compare two tapes. Therefore, it takes \( O(4n) = O(n) \) steps to accept a palindrome.

---

![State Transition Diagram](https://via.placeholder.com/150)