CS21004 Formal Languages and Automata Theory, Spring 2011–12

Class test 2

Maximum marks: 20 | Date: 09-April-2012 | Duration: 1 hour

Roll no: ________________ **Name:** ____

[*Write your answers in the question paper itself. Be brief and precise. Answer all questions.*]

1. Using the pumping lemma, prove that the language $L_1 = \{a^n b^{n^2} \mid n \geqslant 0\}$ over $\Sigma = \{a, b\}$ is not context-free. Carefully consider all the cases. **(10)**

Solution Suppose that L_1 is context-free. Let k be a pumping lemma constant for L_1. For the string $a^k b^{k^2} \in L_1$, the pumping lemma gives a decomposition $\beta = \beta_1\beta_2\beta_3\beta_4\beta_5$ with $|\beta_2\beta_4| > 0$ and $|\beta_2\beta_3\beta_4| \leqslant k$. Moreover, for this decomposition, $\beta_1\beta_2^i\beta_3\beta_4^i\beta_5$ is in L_1 for all $i \geqslant 0$. We choose $i = 2$, and show that there is a contradiction in all possible cases.

Case 1: Either β_2 or β_4 contains both a and b. In this case, $\beta_1\beta_2^2\beta_3\beta_4^2\beta_5$ is not of the form a^*b^*.

Case 2: Both β_2 and β_4 are in the block of a's. In this case, $\beta_1\beta_2^2\beta_3\beta_4^2\beta_5$ contains more a's than the square root of the number of b's.

Case 3: Both β_2 and β_4 are in the block of b's. In this case, $\beta_1\beta_2^2\beta_3\beta_4^2\beta_5$ contains more b's than the square of the number of a's.

Case 4: β_2 belongs to the block of a's, and β_4 belongs to the block of b's. In this case, $\beta_1\beta_2^2\beta_3\beta_4^2\beta_5$ is of the form a^ib^j with $i \geqslant k$ and $j \leqslant k^2 + k < (k+1)^2$, where at least one of these inequalities ($\leqslant$ and $\geqslant$) is strict.

2. Design a PDA to accept the language $L_2 = \{\alpha \in \{a,b\}^* \mid \#b(\alpha) \leqslant \#a(\alpha) \leqslant 2\#b(\alpha)\}$. Mention whether your PDA accepts by empty stack or by final state or both. Explain how your PDA accepts the string $aabba$. (**Hint:** Give a *weight* of -1 to each a, and a *weight* of $+1$ or $+2$ (non-deterministic choice) to each b.) **(10)**

Solution We construct a PDA with one state q (well, we need two more states, see below) to accept L_2 by empty stack. Each occurrence of b corresponds to one or two occurrences of a in the input. When b is consumed from the input, the machine non-deterministically pushes one or two $+$'s to its stack. When an a appears, a single $+$ is popped out of the stack. The only troublesome case is the occurrence of an a when the stack contains only the bottom marker $\perp$ (or when the input has not yet supplied enough b's to match the a's seen so far). In this case, a single $-$ is pushed. Later, when a b appears, one or two $-$'s at the top of the stack are to be popped out. If there is only one $-$ at the stack and two $-$'s need to be popped out, the machine should replace the $-$ by a $+$. When the entire input is read, the bottom marker $\perp$ should be exposed, which is then popped out, and the machine accepts with an empty stack. So the transitions of the PDA will be the following:

$a,\ + / \epsilon$	[Each a has weight -1. Neutralize a $+$ already present in the stack.]
$a,\ \perp / - \perp$	[No $+$ is present in the stack.]
$a,\ - / - -$	[No $+$ is present in the stack.]
$b,\ \perp / + \perp$	[This b has weight $+1$.]
$b,\ \perp / + + \perp$	[This b has weight $+2$.]
$b,\ + / + +$	[This b has weight $+1$.]
$b,\ + / + + +$	[This b has weight $+2$.]
$b,\ - / \epsilon$	[This b has weight $+1$.]
$b,\ - - / \epsilon$	[This b has weight $+2$.]
$b,\ - \perp / + \perp$	[This b has weight $+2$.]
$\epsilon,\ \perp / \epsilon$	[The last transition to empty the stack.]

The transition $b, - - / \epsilon$ is not a true transition according to our definition. This problem can be solved by adding a temporary state r. After reading b from the input, a single $-$ is popped from the stack, and the state changes to r. We add an ϵ-transition from state r back to state q which pops the second $-$ from the stack. The other transition $b, - \perp / + \perp$ can be handled analogously using a second temporary state s.

The following table demonstrates two ways of accepting $aabba$. The two computations differ by the non-deterministic choices of giving the weights $+1$ and $+2$ to the two b's.

Input symbol read	Weight	State	Stack
Initially		q	$\perp$
a	-1	q	$- \perp$
a	-1	q	$- - \perp$
b	$+2$	r	$- \perp$
ϵ	0	q	$\perp$
b	$+1$	q	$+ \perp$
a	-1	q	$\perp$
ϵ	0	q	Empty

Input symbol read	Weight	State	Stack
Initially		q	$\perp$
a	-1	q	$- \perp$
a	-1	q	$- - \perp$
b	$+1$	q	$- \perp$
b	$+2$	s	$\perp$
ϵ	0	q	$+ \perp$
a	-1	q	$\perp$
ϵ	0	q	Empty

If we give the weight $+1$ to both the b's, the stack continues to contain one leftover $-$ after the entire input is read. If we give the weight $+2$ to both the b's, the stack contains a leftover $+$ after the entire input is read.