Homework 4

Feb 19, 2017

(804501476)

Problem 1.0:

a)

$L_1 = \{0^a 1^b \mid a \text{ divides } b\}$ is equivalent to $L_1 = \{0^a 1^{am} \mid m \geq 0\}$.

Proof.
Towards contradiction:
- Assume L_1 is CF.
- By pumping lemma $\exists$ pumping length p.
- Let $s = 0^k 1^{km}$ s.t. $|s| \geq p$.
- By pumping lemma $0^k 1^{km} = xyzuv$

1) $|yzu| \leq p$
2) $|yu| > 0$

Cases:

1) $yzu = 0^\alpha$, $1 \leq k \leq p$. Then $xy^2zu^2v = 0^{k'} 1^{nk'}$
The number of 1's does not change therefore:

$$k' n = km \tag{1.1}$$
$$n = \frac{k}{k'} m \tag{1.2}$$
$$n = \frac{k}{k+\alpha} m \tag{1.3}$$

n has to be an integer but $k' > k$ therefore $n \notin \mathbb{N}$.

Problem 1.0 continued on next page...

2) $yzu = 1^{\alpha}$, $1 \leq k \leq p$. Then $xzv = 0^{\frac{k'}{n}} 1^{k'}$
The number of 0's does not change therefore:

$$\frac{k}{m} = \frac{k'}{n} \tag{1.4}$$
$$kn = k'm \tag{1.5}$$
$$n = \frac{k'}{k}m \tag{1.6}$$
$$n = \frac{k-\alpha}{k}m \tag{1.7}$$

n has to be an integer but $k' < k$ therefore $n \notin \mathbb{N}$

3) $yzu = 0^{\alpha}1^{\beta}$, $1 \leq \alpha + \beta \leq p$ then $xy^2zv^2u = 0^{p+\alpha}1^{pm+\beta}$

$$\frac{a}{b} = m \tag{1.8}$$
$$\frac{pm+\beta}{p+\alpha} = m \tag{1.9}$$
$$pm + \beta = pm + m\alpha \tag{1.10}$$
$$\beta = m\alpha \tag{1.11}$$

□

b)

Let $L = \{a^n b^n \mid n \geq 0\}$ then $\tilde{L} = \{a^n b^{2n} a^n \mid w \in L \text{ or } w^R \in L\}$

Proof. Towards contradiciton assume that $\tilde{L}$ is context-free.

- By pumping lemma $\exists$ pumping length p.
- Let $w \in \tilde{L},\ \ w = a^p b^{2p} a^p,\ \ |w| \geq p$
- By pumping lemma $a^p b^{2p} a^p = xyzuv$
- $|yzu| \leq p$
- $yu > 1$

Cases:

1) $yzu = a^{\alpha}$ on the left, $1 \leq \alpha \leq p$

$$xzv = a^{p-\alpha}b^{2p}a^{p} \notin \tilde{L}$$

2) $yzu = a^{\alpha}$ on the right, $1 \leq \alpha \leq p$

$$xzv = a^{p}b^{2p}a^{p-\alpha} \notin \tilde{L}$$

3) $yzu = b^{\beta}$, $1 \leq \beta \leq p$

$$xzv = a^{p}b^{2p-\beta}a^{p} \notin \tilde{L}$$

□

Problem 2.0:

a)

We have two stacks S_1 and S_2. First the elements of a string must be pushed onto S_1. Once all elements are in S_1 push all but the bottom most element onto S_2. Now the stacks will represent a window into two elements of of the input string, the top of S_1 will be the element on the left and the top of S_2 will be the element on the right. To simulate head movement of Turing machien T we define left and right movement by pushing and popping from stacks of multi-stack machine M.

Move head to the right
$Push_{S_1}(Pop_{S_2})$
Move head to the right and replace element with β
$Pop_{S_1}, Push_{S_1}(\beta), Push_{S_1}(Pop_{S_2}(\alpha))$
Move head to the right and write β
$Push_{S_1}(\beta), Push_{S_1}(Pop_{S_2}(\alpha))$
Move head to the left
$Push_{S_2}(Pop_{S_1}(\alpha))$
Move head to the left and replace element with β
$Pop_{S_1}, Push_{S_2}(\beta)$
Move head to the left and write β
$Push_{S_2}(Pop_{S_1}(\alpha)), Push_{s_2}(\beta)$

The states of machine M are equal to the states of T. Transitioning between states is done using the δ function of machine T. An accepting state of this machine would be conditioned on a specific configuration of the elements in the stacks and we can move left to right to determine if the configuration is valid and then transitioning on to an accept state that is in T.

Problem 2.0 continued on next page...

b)

We simulate three stacks using a turing machine T. In class we showed that a multi-tape turing machine can be simulated using a single tape. For simplicity we use multiple seperate tapes. We have an input tape to store the input string and t_1, t_2, and t_3 each tape for one stack. We are given machine M with three stacks S_1, S_2, and S_3. Let $x = x_1x_2 \ldots x_n$ be the input string.

Starting States of Stacks and Tapes
Each stack has a starting state with an end of stack symbol on it. Machine T has the input x on its input tape and each tape t_i will have the corresponding stack symbol as its first symbol.
Stack Operations
Upon reading x_i machine M can:
1) $Push_{S_i}(x_i)$ - Push x_i onto stack S_i for $i \in \{1, 2, 3\}$
2) Pop_{S_i} - Pop element off of S_i.
3) $Push_{S_i}(Pop_{S_j}),\ i \neq j$ - Pop and element of of one stack and push to another.
Equivalent Tape Operations
Each operation corresponds to the operations defined above respectively. Upon reading x_i machine T:
1) Move head to end of tape t_i, when first $\sqcup$ is under head, and write x_i onto it.
2) Move head to end of tape t_i, then move the the left by one and write $\sqcup$.
3) Move to end of tape t_j and the left by one. Then move to end of t_i and write symbol under head of t_j onto t_i.

The states of machine T are equal to the states of machine M. Transitions between states in T follow the δ function of machine M. T accepts if after processing string x it is in the accept states of machine M.

In part a) we showed that a Turing machine is no more powerful than a 2 stack PDA. In part b) we showed that a 3-stack machine is no more powerful than a TM machine. Therefore a 3-stack PDA is no more powerful than a 2-stack PDA.