CS 321, Assignment 7

Cody Malick
malickc@oregonstate.edu

November 28, 2016

1

a

1. Adversary picks a number $p \geq 0$
2. I pick a string $s \in A$, $bin(2^p)\ c\ bin(2^p) + 1$
3. Adversary breaks s into $s = uvwxy$, such that $|vwx| \leq p$ and $|vx| > 0$
4. I pick a number $i \geq 0$. If $uv^iwx^iy \notin A$, then I win.

There are a couple cases here:
uwx exists only in x
uwx exists only in c, $|uwx| = 1$
uwx exists only in y
uwx exists in all parts, xcy
uwx exists in xc
uwx exists in cy

All of these cases can be resolved by picking $i = 0$. If $uwy \in x$, then y would no longer be equal to $bin(x) + 1$. If $uwy \in xc$, then if c was part of v or x, then there is no longer a c in the string. This case also applies to any of the cases containing c. Finally, if $uwx \in cy$ or $uwx \in y$, pumpin down breaks the equivilence of $bin(x) + 1 = bin(y)$.

b

This problem has several cases, each case will be followed by it's solution:

case 1

uwx exist only in a
uwx exist in 'a' and b
uwx exist only in b

1. Adversary picks a number $p \geq 0$
2. I pick a string $s \in A$, $a^{p-2}b^{p-1}c^p$
3. Adversary breaks s into $s = uvwxy$, such that $|vwx| \leq p$ and $|vx| > 0$
4. I pick a number $i \geq 0$. If $uv^iwx^iy \notin A$, then I win

I select $i = 2$, as v^iwx^i are either all in a, are in a and b, or are only in b, then the number of a's will be equal to the number of b's, or the number of b's will be equal to the number of c's. In those cases, the string is not in the language.

case 2

uwx exist in 'b' and 'c'
uwx exist in only 'c'

1. Adversary picks a number $p \geq 0$
2. I pick a string $s \in A$, $a^{p-2}b^{p-1}c^p$
3. Adversary breaks s into $s = uvwxy$, such that $|vwx| \leq p$ and $|vx| > 0$
4. I pick a number $i \geq 0$. If $uv^iwx^iy \notin A$, then I win

I select $i = 0$, as v^iwx^i are in both b's and c's, or only c's, then because I pumped down, the number of b's are equal to the number of a's, or the number of c's are equal to the number of b's, and therefore, not in the language.

2

My approach to this turing machine is to break the multiplication of the exponents $a^nb^mc^{nm}$ down into an addition problem. For example, $n = 2, m = 3$, $2 * 3 = 2 + 2 + 2$. With this idea in mind, we can count the number of a's onto the number of c's, and repeat for every b.
First, count the number of a's onto c's. This is done by replacing c's with X's equal to the number of a's. This is done from steps $0 \to 2$. The head will start on the left, scan right for an a. Once all a's have been eliminated, we right a b to 0, and check for accepting by seeing if any c's remain. Then, for every a written to the c's, marked by X, we repeat the elimination of c's $|b|$ times.
X's and Y's are eliminated in alternating order until all c's have been replaced, and we read blank memory, $\Box$, we accept.

Turing Machine for $\{a^nb^mc^{nm}\}$			
State #	If in state:	reading:	do:
0	Start State	a b,c,X,Y,0 $\Box$	Write 0, move right, go to state #1 reject accept
1	Find c	c a,b,0,X,Y $\Box$	Write X, move left, go to state #2 move right reject
2	Find a	a b,0 c,X,Y,$\Box$ tape start	Write 0, move right, go to state #1 move left reject move right, go to state #3
3	Find b	b 0 a,c,X,Y $\Box$	Write 0, move right, go to state #4 move right reject
4	Find X	X b,0 a,c,Y $\Box$	Write 0, move right, go to state #5 move right reject accept
5	replace c with Y	c X,0,Y a,b $\Box$	Write Y, move left, go to state #6 move right reject accept
6	eliminate X	X Y,0 b a,c,$\Box$	Write 0, move right, go to state #5 move left write 0, move right , go to state #7 reject
7	eliminate Y	Y 0 $\Box$	Write 0, move right, go to state #8 move right accept
8	replace c with X	c X,Y,0 b $\Box$	Write X, move left, go to state #9 move right reject accept
9	look for a Y	Y 0,X b	Write 0, move right, go to state #8 move left write 0,move right, go to state #4