Finite Automata Homework 8

Due: Thursday, Nov 5
Alexander Powell

1. Let $Y = \{w | w = t_1\#t_2\#\ldots\#t_k$ for $k \geq 0,$ each $t_i \in 1^*,$ and $t_i \neq t_j,$ whenever $i \neq j\}$. Here, $\Sigma = \{1, \#\}$. Prove Y is not context free.

   Solution:

   Proof. We will use a proof by contradiction. Begin by assuming Y is context free. Then the pumping lemma for CFLs applies. If we define s to be

   $$s = 1^{P+1}\#1^{P+2}\#\ldots\#1^{3P},$$

   then we can clearly see that $|s| > P$. Also, via the pumping lemma we have that $s = uvxyz$, $|vxy| \leq P$ and $|vy| > 0$. Because $|vxy| \leq P$, then we have three cases:

   (a) Either v or y contains the # symbol.
   If v contains the # symbol then v will look something like $v = 1^m\#1^n$. By fixing $i = 3$ (or really any positive integer greater than 1) then v will look like

   $$v = 1^m\#1^n1^m\#1^n1^m\#1^n = 1^m\#1^{n+m}\#1^{n+m}\#1^n$$

   which is clearly not an element of the langage Y. The case for if y contains the # is practically identical.

   (b) In the second case both v and y are contained within the same string of 1s of length j. In the case where $j > 2P$ then we set $i = 1$. In the case where $j \leq 2P$ then we set $i = 0$. In both of these cases we arrive at a contradiction.

   (c) If the # symbol is contained within x. This case needs to be split into three subcases:

      i. v is contained in 1^j in such a way that $j < 2P$. In this case, if $|v| = 1$, then we can set $i = 2$ and if $|v| > 1$ we can set $i = 1$, which will result in a case where $t_i = t_j$, which is a contradiction to our assumption.

      ii. In this case, y is contained in 1^j in such a way that $j \geq 2P$. In this case, if $|y| = 1$ then we can set $i = 2$ and if $|y| > 1$ we can set $i = 1$, which will result in a contradiction.

      iii. In the third subcase, if $|v| = 0$ or $|y| = 0$ then we set $i = 1$ (or really any positive integer) which will result in a contradiction because there will exist an interval in s between one of the # symbols that doesn't have the correct number of 1s.

   Therefore, because we reach a contradiction in all possible cases, we have proven that Y is not a context free language. $\square$

2. Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet $\{0,1\}$.

   (b) $\{w|w$ contains twice as many $0s$ as $1s\}$

   Solution: We can describe the Turing machine as a set of sequential steps to decide a language.

   (1) Go through the tape and mark the first 1 we see that is not marked. If there are no unmarked 1s to be found then we go to step 5. If we do encounter a 1 that is unmarked, we move the pointer back to the beginning of the tape.

   (2) Go through the tape until the first unmarked 0 is found and mark that 0. If none are found, then reject.

   (3) Go through the tape until the first unmarked 0 is found and mark that 0. If none are found, then reject. (Repetition of step 2)

   (4) Go back to Step 1 and follow the sequence of instructions.

   (5) We go through the tape to see if there are any unmarked 0s remaining. We reject if we find any. Otherwise, accept.

3. Show that the collection of decidable languages is closed under the operation of:

   (b) Concatenation

   Solution:

   Proof. Let M, N be decidable languages. Then the concatenation of M and N is the language MN, where

   $$MN = \{mn|m \in M,\ n \in N\}$$

   Since M and N are decidable then there exist Turing machines T_M and T_N that decide the languages M and N. We need to construct a turing machine T_{MN} that decides MN. The machine will use both T_M and T_N. It will take in the input string w. To determine if w can be written in the form mn, where $m \in M$ and $n \in N$ the machine will iterate through all possible paritions of the string w (because w has a finite length and thus there are a finite number of ways to partition w). For each of these partitions we will pass the m component into T_M and the n component into T_N. If for any of these iterations we find that T_M and T_N both accepted their respective inputs, we say T_{MN} accepts the string w. If none of the iterations give us a partition whose components are accepted, then T_{MN} rejects the string. Therefore, because a Turing machine, T_{MN} could be created to decide MN, we can conclude that the collection of decidable languages is closed under concatenation. □

   (c) Star

   Solution:

   Proof. The star of a language Y can be defined as $Y^* = \{x \in Y \cup YY \cup YYY \cup \ldots\}$. This can be thought of as a string being concatenated with itself any number of times. Let's take a Turing machine M_Y that decides Y and M_{Y^*} that decides Y^*. Then, for any input x into that machine M_{Y^*}, we partition x into all possible partitions $x_1x_2\ldots x_n$. Now, we can run each x_i through the machine M_L for $i = 1,\ldots,n$. If M_L accepts each x_i, then the machine M_{Y^*} accepts. If M_L rejects at least one x_i, then M_{Y^*} rejects. This can be drawn as a repeating loop back to the M_L machine inside the larger M_{Y^*} machine. Therefore, we have proven that decidable languages are closed under the star operation. □

(e) Intersection

Solution:

Proof. We begin by taking two Turing decidable languages, L_1 and L_2 as well as Turing machines M_1 and M_2 that decide L_1 and L_2 respectively. Then we can construct a new Turing machine M that decides $L_1 \cap L_2$. For any input string x:

(a) We run x through M_1. If M_1 rejects then we say M rejects and we're done. If M_1 accepts we continue to step 2.

(b) Next we run x through M_2. If M_2 rejects then M rejects and we're done. If M_2 accepts (and M_1 has already accepted) then we say that M accepts.

Therefore, it is clear that $L(M) = L_1 \cap L_2$ and therefore we can conclude that the collection of decidable languages is closed under intersection. □