CS 4820, Spring 2018 **Homework 9, Problem 2**
Name: Rongguang Wang
NetID: rw564
Collaborators: Siyuao Liu (sl2928); Yihao Chen (yc2288)

(2) *(10 points)* Let us call a Turing machine M "speedy" if it has the following property: for every input string x, the computation of M on input x terminates, and it does so after at most $|x|+100$ steps.

Design an algorithm to decide whether a Turing machine is speedy, given a description of the Turing machine. In more detail, the input to your algorithm consists *only* of the description of a machine M (in particular the input to your algorithm does not specify any particular input string for M) and its output should be "yes" if M is speedy and "no" if M is not speedy.

In your solution you should describe the algorithm — an ordinary algorithm description is fine, no need to explain how to implement the algorithm as a Turing machine. Prove that your algorithm always terminates, and prove that it always outputs the correct answer. However, *you do not need to analyze your algorithm's running time, other than proving that it always terminates.*

Solution:

The problem is aiming to decide whether a Turing Machine M is "speedy" with the property that the computation of M on input x, which a string, terminates within $|x| + 100$ steps. The description of the Turing Machine is given including the alphabet Σ, the state set Q and transition rule $\delta(q, \sigma)$. The input to the machine M is not given. The machine outputs "yes" if M is speedy and "no" if M is not speedy. Since the machine should terminate within $|x| + 100$ steps, we have an observation on this problem. A speedy Turing Machine can never visit a location more than 100 steps to the left of the right most location it has been. This means that if a Turing Machine visited the location which is 101 steps to the left of the right most location it has been, this machine is not speedy. Since the input size of the given Turing Machine is not known for this problem, the possibility of the tape of the machine can be infinity. So, we need to limit the situations into finite which can be solved. The tape of the machine should be refined into 101 slots which including the rightmost location the machine has ever been and the 100 locations on the left of it. We construct abstract of node for the input machine as $(q, z, g) \in K \times \Sigma^{101} \times \{0, 1, \ldots, 101\}$. q denote the state right after the pioneering transition. Pioneering transition refers to transiting to the position that is has not visited before. z denote the tape content at the location of the pioneering transition and the 100 proceeding locations. If there are not enough symbols at the left of the location of the pioneering transition, we fill in with symbol $\sqcup$. For example, the starting node is represent as $(s, \sqcup\ \sqcup\ \sqcup\ \ldots\ \triangleright, 0)$. g denote the difference between the steps it taken from the start and the location of the pioneering transition. The numbers of state K and the alphabet Σ are given by the Turing Machine M and they are finite. The number of the difference g is range from 0 to 101. Therefore, the nodes of the machine is the arrangement of the state set K, 101 digits of alphabet set Σ and the number of difference g. Since all numbers are finite, the arrangement of them is finite. Therefore, there are finite nodes for the machine M.

Next, we need to connect the nodes to construct a graph G. We input the tape content z of all nodes into the Turing Machine M and the output should be the nodes that we constructed since we enumerated all possibilities of the arrangement of the 101 positions of symbols from alphabet Σ. Since there is a transition from the input nodes to the output nodes, the input nodes should be connected to the output nodes with direction from input to output. Then, we already constructed the graph G. Now, we can perform BFS to search the graph G from the starting node $(s, \sqcup\ \sqcup\ \sqcup\ \ldots\ \triangleright, 0)$. If the a path found that connected to the node $(q, z, 101)$, which consists of any state in K, any arrangement of the alphabet Σ and difference $g = 101$, the machine M is not speedy. Otherwise, if the searching cannot find a path described above, the machine M is speedy.

Proof of Correctness:

Lemma 1. *A speedy Turing Machine can never visit a location more than* 100 *steps to the left of the right most location it has been.*

Proof. Since the input size of the Turing Machine is not given, we dont know the number of $|x| + 100$. Suppose we have a input string x with size $|x|$. The machine traverse the string from the $\triangleright$ to location t where $t < x$ and the machine always go right without going left or halt. Then the machine go left 100 steps to location $(t - 100)$. If the machine move left one step to location $(t - 101)$, the machine is no longer speedy. Therefore, a speedy machine should not visit a position more than 100 steps to the position of the pioneering transition. $\square$

Lemma 2. *The algorithm will eventually terminate.*

Proof. Since there are finite nodes in the graph G which has been proved above, the edges between the nodes are finite. The graph G can be cyclic or non-cyclic. When running BFS on the graph with cycle, the algorithm will terminate when reach the node with $g = 101$. When running BFS on the graph with no cycle, the algorithm will either terminate when reach the node with $g = 101$ or reach the end-node with $g < 101$. $\square$

Lemma 3. *Each path in the graph G corresponding to an input string of the Turing Machine M.*

Proof. We proof by induction. Since the edges of the graph G are connected by simulating the tape content z of the input nodes which means that they are connected by the transition rule σ in the machine M, every connection between the nodes is a legal transition between different states. So, the basic case is that the edges connected between the starting node $(s, \sqcup\ \sqcup\ \sqcup\ \ldots\ \triangleright, 0)$ and other nodes that can be reached. Then, the induction step is that starting from the second layer which is the layer of nodes that connecting to the starting node $(s, \sqcup\ \sqcup\ \sqcup\ \ldots\ \triangleright, 0)$. So, all edges connected from the starting node to the second nodes layer, from the second nodes layer to next nodes layer are all legal. Therefore, each path in the graph G corresponding to an input string of the Turing Machine M. $\square$

Lemma 4. *If there is a path lead to the node that has $g = 101$, then the Turing Machine M is not speedy.*

Proof. Since each path is corresponding to an input string of the Turing Machine M, the BFS searching on the graph G is checking whether the Turing Machine is speedy. According to Lemma 1, if a machine visit the location l which is the 100 steps to the left of the location of the pioneering transition twice, the machine is not speedy. Therefore, if there is a path lead to the node with $g = 101$ which means this node visited the location l twice, the machine is not speedy. $\square$