COMP-170: Homework #2

Ben Tanen - February 5, 2017

Problem 5

A Turing machine is a *Throw Back* machine if instead of moving left or right in its transition function, the machine can either go back to the beginning tape cell or move right. Prove that this model is equivalent to the standard Turing machine.

* * *

In order to show that a *Throw Back* machine is equivalent to a standard Turing machine, we will show that every computation / movement for a Turing machine can be modeled in a *Throw Back* machine and vice versa.

Turing Machine $\rightarrow$ Throw Back Machine: First, we will show that every movement in a *Throw Back* machine can be done with a traditional Turing machine.

Let w be any standard Turing machine that we are given and the machine we would like to model as a *Throw Back* machine.

It is obvious that the right tape movement(s) of a *Throw Back* machine can be exactly replicated with the right tape movement(s) of a standard Turing machine. Because both machines have R movements, we do not need to make any changes to maintain this behavior.

In order to model a *Throw Back* machine's ability to jump back to the beginning, we will need to make a few small changes to our machine. To start, we will need to add a new state q_{start} to Q. Let q_0 be the state that w originally starts in. We will replace q_0 with q_{start} so that w initalizes in q_{start}. We will also add $\langle q_{start}, c\rangle \rightarrow \langle q_0, cS, L\rangle$, for all $c \in \Sigma$, to δ. We will additionally have to add $\langle q_i, cS\rangle \rightarrow \langle q_j, dS, m \in \{L, R\}\rangle$ for each transition $\langle q_i, c\rangle \rightarrow \langle q_j, d, m \in \{L, R\}\rangle \in \delta$. By doing this, we made it so that the first transition of w will add a marking to the first cell of our input and then we will then be in the state w was originally intended to start in.

Next, in order to actually do a *Throw Back* movement, we will need to add more states to w. Specifically, for every $q_i \in Q$ (excluding the q_{start} we just added), we will add a q_{i-TB} to Q. Next, any time we would like to perform a throw back step $\langle q_i, c\rangle \rightarrow \langle q_j, d, TB\rangle$ for any $q_i, q_j \in Q$ (excluding the states we added) and any $c, d \in \Gamma$, we can replace that step with $\langle q_i, c\rangle \rightarrow \langle q_{j-TB}, d, L\rangle$ in δ. Additionally, we will need to add $\langle q_{i-TB}, c\rangle \rightarrow \langle q_{i-TB}, c, L\rangle$ to δ for all of the states we added to Q and for all $c \in \Gamma$ such that c is not marked with the S that we added above. Finally, we will add $\langle q_{i-TB}, cS\rangle \rightarrow \langle q_i, cS, L\rangle$ for all of the states we added of the form q_{i-TB} and all characters of the form $cS \in \Gamma$. All of these additions will make it so that we can model a *throw back* movement by repeatedly moving to the left until we hit the marked start of our tape.

Thus, through these additions of states, characters, and transitions, we have made it so that the right tape movement(s) and the jump back tape movement(s) of a *Throw Back* machine can be modeled in our traditional Turing machine w. This now makes it so that the two machine types can do the same behaviors and are thus equivalent.

Throw Back Machine $\rightarrow$ Turing Machine: Next, we will show that every movement in a traditional Turing machine can be done with a *Throw Back* machine.

Let w be any standard Turing machine that we want to simulate with a *Throw Back* machine. Let w' be this *Throw Back* machine that we will construct to do this.

To start, add all of the tape and input characters, states, and transitions of w to w'.

Like the reverse direction above, because both *Throw Back* machines and traditional Turing machines have R tape movements, we can see that there are no changes necessary to model the right tape movements of w in w'. Thus, that is already set up.

Now, a more difficult implementation comes from modeling a left tape movement using jump backs. To do this, consider the following high-level algorithm to model $\langle q_i, c\rangle \rightarrow \langle q_j, d, L\rangle$:

1. Mark current position with x and jump back
2. Check if on x marked position:
   (a) If so, replace cell with d and jump back again, entering state q_j - **FINISHED**
   (b) Else, mark current cell with a p and move right
3. Check if on x marked position:
   (a) If so, replace cell with d, jump back again, remove p mark on beginning of tape, and jump back again, entering state q_j - **FINISHED**
   (b) Else, mark current cell with p, and jump back
4. Replace p marking with o marking and move right
5. Repeat the following loop:
   (a) Walk right one cell
   (b) Check if on x marked cell
      i. If so, replace cell with d, jump back again, and go to step 6
      ii. Else, mark current cell with p, and jump back
   (c) Walk forward until we find o marked cell
   (d) Remove o marking and walk right one cell
   (e) Replace p marking with o marking and walk right one cell
6. Walk right until we find p marked cell
7. Remove p marking and jump back
8. Walk right until we find o marked cell

9. Remove o marking and walk right one cell, entering state q_j - **FINISHED**

While significantly more involved than a simple left tape movement in a standard Turing machine, the above algorithm will achieve the same behavior as a $\langle q_i, c\rangle \rightarrow \langle q_j, d, L\rangle$ transition, using only right tape movements and jump backs.

To do this algorithm, we would need to add 9 new states for each existing $q_j \in Q$. Additionally, we would need to add a *co* and a *cp* marking to Γ for every $c \in \Gamma$. Finally, in order to update δ for this algorithm, we would need to add 19 new transitions for each $q_j \in Q$ and each $d \in \Gamma$. By doing this, we give w' the ability to model the left tape movements of w using only jump backs and right tape movements.

Ultimately, while the above algorithm does significantly expand w' over w, we can see that with correct usage and these additional transitions, states, and tape markings, w' (a *Throw Back* machine) can decide any language w can decide because all of the same tape movements and computations are available in both w and w'.

Thus, because we can model every computation / movement for a standard Turing machine on a *Throw Back* machine and vice versa, the two machine types are equivalent. $\boxtimes$