

1 Turing Machines

See posted JFLAP file. The basic algorithm is as follows (if at any point the string is of the wrong form, it is rejected):

1. If the string before the < is all 0's then:
2. If the string after the < is not all 0's then accept.
3. If the string after the < is all 0's then reject.
4. If the string after the < is all 0's then reject.
5. Subtract 1 from the number on the right of the <.
6. Subtract 1 from the number on the left of the <.
7. Goto step 1.

2 Equivalence between computational models

We wish to simulate a 2dTM $(Q, \Sigma, \Gamma, \delta, s, a, r)$ with a normal TM. We will assume that the 2dTM has an infinite tape to the right and down, but not up or to the left.

Let $\Gamma' = \{0, 1, ,, , \$, \hat{0}, \hat{1}, \hat{,}\} \cup \Gamma$. The contents of the tape will be of the form $\$current - x, current - y \$x - pos_1, y - pos_1, \gamma_1; x - pos_2, y - pos_2, \gamma_2; \dots; x - pos_n, y - pos_n, \gamma_n$, where $current - x, current - y, x - pos_i$, and $y - pos_i \in \{0, 1\}^*$ are binary encodings of positions on the 2d tape and $\gamma_i \in \Gamma$ is the symbol at position $x - pos_i, y - pos_i$ on the 2d tape. The upper left corner of the tape will be 0, 0, and coordinates will increase down and to the right. The current state of the 2dTM will be stored in the states of the regular TM simulating it.

“On input w :

1. Reformat $w = w_1w_2 \dots w_k$ on the tape as $\$0, 0\$0, 0, w_1; 1, 0, w_2; \dots; k - 1, 0, w_k$. ($k - 1$ denoting the binary representation of the number $k - 1$.)
2. While we have not yet reached an accepting or rejecting state in the TM being simulated, do:
 - (a) Scan left to the beginning of the tape. Scan right looking for a pair $x - pos_i, y - pos_i$ that matches $current - x, current - y$. The symbols with hats over them are used to cross off one symbol at a time until the correct location is found (ignoring leading 0's).
 - (b) If the pair $current - x, current - y$ is not found, append $; current - x, current - y, _$ at the end of the tape.
 - (c) Read the symbol γ_i immediately following the coordinate pair matched (or added). Replace it using the transition function δ of the 2dTM based on the current state. Save the new state of the simulated 2dTM by transitioning accordingly in the states of the simulating TM.
 - (d) Scan left to the beginning of the tape, replacing all symbols with hats with the corresponding version without. Increment/decrement $current - x$ and $current - y$ based on δ from the previous step. If at any point, decrementing would give a negative number, leave that coordinate at 0.
3. If we reached an accepting state then accept. Otherwise, reject.”

3 Reduction

$U = \{\langle M, q \rangle \mid M \text{ is a TM and } q \text{ is a state in } M \text{ and for any input to } M \text{ it enters state } q \text{ at most once}\}$

We will prove that U is undecidable by reduction from $HALT$. We will construct the following TM R with oracle access to U that decides $HALT$:

“On input $\langle M, w \rangle$:

1. Construct the following TM M' with states q and q' which are never entered unless explicitly stated:
“On input x :
 - (a) Run M on input w .
 - (b) Loop between states q and q' forever.”
2. Run the oracle for U on input $\langle M', q \rangle$.
3. If the oracle accepts then reject.
4. If the oracle rejects then accept.”

To argue the correctness of our reduction, we consider 2 cases:

- M halts on w . In this case, regardless of input to M' , it will first run M on w . This will finish, and then it will loop in between states q and q' . Thus, it will enter state q more than once. Therefore the oracle for U will reject and R will then accept.
- M does not halt on w . In this case, regardless of input to M' , it will loop forever running M on w . Thus, it will never enter state q . The oracle for U will accept and R will reject.

In both cases, R does as expected for $HALT$ and always halts. Thus R^U decides $HALT$ so U must be undecidable.