Gavin Grob
CS 510 Automata Theory
Homework 7

1. Let's define a *useless state* in a Turing Machine as a state that is never entered on any input string. Consider the problem of determining whether a Turing Machine has any useless states. Formulate this problem as a language and show that it is undecidable by using a reduction.

$TM_{useless} = \{< M > | M \text{ is a TM } \}$

$TM_{accept} = \{< M, w > | M \text{ is a TM and w is a string. } \}$

We will assume $TM_{useless}$ exists, and use it to create $TM_{accept}$.
We run $TM_{accept}$ on input M, we then create $TM_{transform}$ which hard codes M and w so that w is on the input tape and $TM_{transform}$ states are equal to M states. If M accepts then $TM_{transform}$ manually goes threw all of its states. If M does not accept(either spins forever or not in the language) then $TM_{transform}$ doesn't go into an accepting state so it has useless states.
So when $TM_{transform}$ is feed to $TM_{useless}$ it will accept when M did not accept, so reject. $TM_{useless}$ will reject when M did accept, so accept.

-$TM_{accept}$ = on input (M, w)
- 1.Construct $TM_{transform}$ by modifying M on w
- a. copy w into the input tape (this is what is read when running)
- b. copy all the state of M into $TM_{transform}$
- c. add a case when M goes into the accepting state, $TM_{transform}$ enters every state.
- 2. run $TM_{useless}$ on $TM_{transform}$
- 3. if $TM_{useless}$ rejects then accept, otherwise reject.

However we know that $TM_{accept}$ is undecidable, so then $TM_{useless}$ cannot exist, making it undecidable.

2. Consider the problem of determining whether a two-tape TM $M$ ever writes a blank symbol over a non-blank symbol on its second tape during the course of its computation on any input string. Formulate this problem as a language and show that it is undecidable by using a reduction.

$TM_{ttbs} = \{< M > | M \text{ is a TM}\}$

$TM_{accept} = \{< M, w > | M \text{ is a TM and w is a string}\}$

We will assume $TM_{ttbs}$ exists, and use it to create a $TM_{accept}$. We run $TM_{accept}$ on input M, w we then feed M,w to a $TM_{transform}$.
We then create $TM_{transform}$ which hard codes M and w so that w is on the input tape and $TM_{transform}$ states are equal to M states, besides when M was going to write a blank on tape 2, instead it writes a character not in the language of M, ill call it $\hat{o}$. Whenever $TM_{transform}$ reads the character $\hat{o}$ off the tape, it treats it as a blank symbol. Then when it goes into a accepting state it writes a non-blank symbol on the second tape, then write a blank over that same space (incase tape 2 is empty).
So when $TM_{transform}$, has a non-blank written over with a blank in the second tape, w was accepted by M.
So when $TM_{transform}$ is feed to $TM_{ttbs}$ it will accept when M accepted, so accept. $TM_{ttbs}$ will reject when M rejected, so reject.

-$TM_{accept}$ = on input (M, w)
- 1. Construct $TM_{transform}$ by modifying M on w
- a. copy w into the input tape (this is what is read when running)
- b. copy all the state of M into $TM_{transform}$
- c. when M was going to write a blank on tape 2, $TM_{transform}$ writes a $\hat{o}$ instead
- d. when $\hat{o}$ is read off the tape, $TM_{transform}$ treats it as a blank symbol
- e. when $TM_{transform}$ goes into a accepting state, $TM_{transform}$ writes a non-blank
- symbol on the second tape, then write a blank over that same space
- 2. run $TM_{ttbs}$ on $TM_{transform}$
- 3. if $TM_{ttbs}$ accept then accept, otherwise reject.

However we know that $TM_{accept}$ is undecidable, so then $TM_{useless}$ cannot exist, making it undecidable.