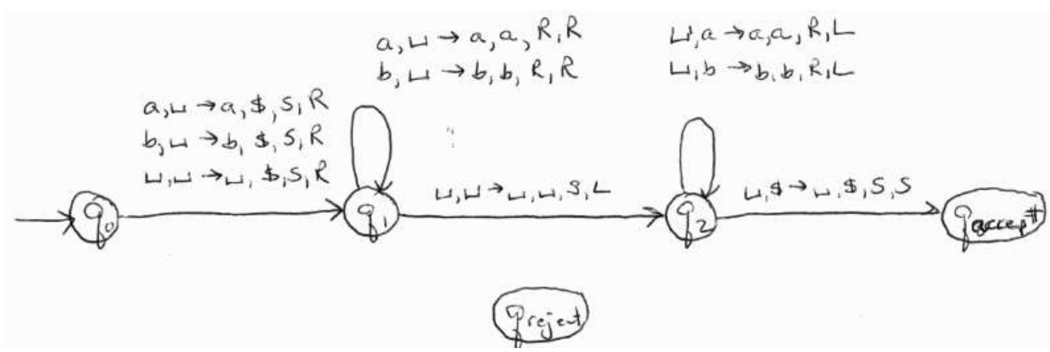
Theory of Computation, CSCI 438 spring 2022
Multitape Turing Machines, pg. 176-178, March 27th

1. Use a 2-tape TM with stays to palindromize a string. That is, given the input w in {a,b}* return (i.e. leave on the tape) ww^R.

   High level plan:
   Mark beginning of tape 2.
   Moving right on both tapes, copy the non-blank contents of the input/output tape onto tape 2.
   Moving left on tape 2, and continuing right on the input/output tape, copy the contents of tape 2 back onto the input/output tape, in reverse order. Accept when reach the beginning of tape 2.

   Detailed plan:
   Place $ at the beginning of tape 2.
   Moving right across a's and b's, until a blank is encountered, copy a's and b's onto tape 2.
   Continue right on the input/output tape, but moving left on tape 2, copy a's and b's onto the input/output tape.
   Accept when $ is reached.



2. What is the relation between Turing recognizable languages and languages recognizable by a 2-tape Turing machine? Prove your answer.

Turing recognizable languages and the set of languages recognizable by 2-tape Turing machines are the same.

In other words, the following Theorem holds:
A language is Turing recognizable iff it can be recognized by a 2-tape Turing machine.

Proof:
$\Rightarrow$ (only if)
A language is Turing-recognizable only if it is recognized by a 2-tape TM.
Other ways to say this:

If a language is Turing-recognizable, then it is recognized by a 2-tape MT

or

$\mathcal{L}(TM) \subseteq \mathcal{L}(TM_{2\text{-tape}})$

If a language is Turing-recognizable then some Turing Machine recognizes it. Notice that also some $TM_{2\text{-tape}}$ recognizes it since the $TM_{2\text{-tape}}$ can be very similar to the regular TM, it simply ignores the second tape.

$\Leftarrow$ (if)
If a language is recognized by a 2-tape TM then it is Turing-recognizable
In other words $\mathcal{L}(TM_{2\text{-tape}}) \subseteq \mathcal{L}(TM)$

Plan

Using a regular TM, simulate $TM_{2\text{-tape}}$ by placing the significant contents of the second tape, after the input of the first tape, separated by some symbol not in the tape alphabet of either tape, say #. Simulate the position of each read/write head on the tape by introducing new symbols x dot for every $x \in \Gamma_1 \cup \Gamma_1$ where x dot indicates that the read/write head is at that location.

Detailed plan

Set up: The input is on the tape as usual, except the first symbol is marked to indicate the position of the read/write head on tape 1. Following the input is a blank, #, and a marked blank, indicating the read/write head of the 2^{nd} tape.

Operation: For each move on the $TM_{2\text{-tape}}$ do the following:

Beginning at the front of the tape, travel right, remembering the two marked symbols indicating the symbols under the first and second read/write head. While this will require many states, first remembering the symbol under the first simulated read/write head, then moving to the second marked symbol and remembering the second simulated read/write head, it will be a finite number of states, a linear multiple of $|\Gamma_1| x |\Gamma_2|$.

Once the two characters are known, the move to be simulated is known. Simulation for each possible transition of the k-tape machine will have been programed when the regular TM was created. Simulate the move by starting at the beginning, traveling right to the first simulated read/write head. Do what is required, writing a symbol if needed, unmarking the symbol which was marked and either marking the left or right symbol. Continue right to the simulated read/write head on the second simulated tape. Do what is required for that move. Move back to the front of the tape, to simulate the next move.

When simulating a right move on the first tape, if '#' is encounter, replace the '#' with a blank and shift the '#' and significant contents of the second tape one position to the right.

When simulating a left move on the second tape, if the # is encountered, simulate the bouncing behavior that would occur on the second tape when a left more is made at the front of the tape.

With careful thinking about it, you will see that this new TM will recognize the same language that the $TM_{2\text{-tape}}$ recognized. Thus when a language is Turing-recognizable, it is recognized by a $TM_{2\text{-tape}}$.

3. Use a multitape machine to recognize the language:

   $L=\{w \mid w \in \{a,b\}^* \text{ and } n_a(w)=n_b(w)\}$.

   Begin with a high level plan. Following this with a detailed plan. Finally, define $TM_{multitape}$.

High level plan

Use a 3-tape machine – the input tape, a tape for the a's and a tape for the b's. Run through the input tape copying the a's onto the a tape and the b's onto the b tape. Travel back through the a's and the b tape, making sure that they have the same number of elements.

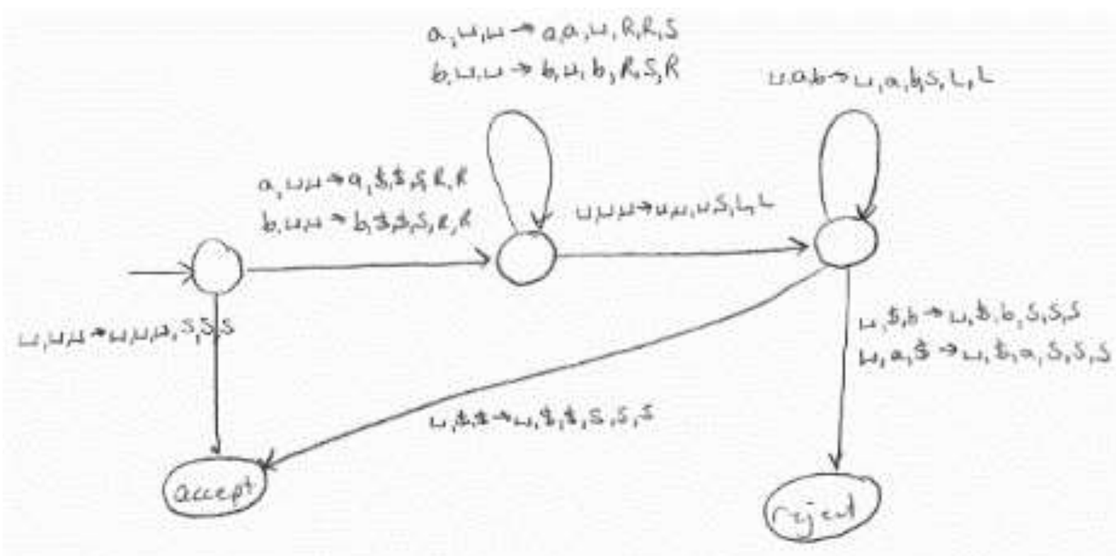
Detail plan

Mark the front of the a and b tape with $.

Travel right across the input tape, copying the a's to the a tape and the b's to the b tape. Stop when encounter a blank on the input tape.

Travel left across the a and b tape until both reach a $, and accept. If one reaches a $ first, reject.

Machine



4. What is the relation between Turing recognizable languages and languages recognizable by a multitape Turing machine? Prove your answer.

Turing recognizable languages and the set of languages recognizable by multitape Turing machines are the same.

In other words, the following Theorem holds:
A language is Turing recognizable iff it can be recognized by a multitape Turing machine.

Pf:
Easy direction ($\Leftarrow$, if) Given a language that is Turing-recognizable then some Turing machine recognizes it. That Turing Machine can be considered a multitape Turing Machine which ignores the other tapes. (Note that each transition in the regular TM would need to match the format of the k-tape transitions, but it would not be difficult to make these changes.)

Hard direction ($\Rightarrow$, only if) Given a language that is recognizable by some multitape Turing machine, show that it is recognizable by a normal Turing Machine (This is Theorem 3.13 in the text)

Plan

Using a regular TM, simulate $TM_{multitape}$ on the single tape by placing the significant contents of each tape, separated by a new tape symbol, say #, onto the single tape. (This is the same as what was done to simulate a 2-tape machine on a regular Turing machine, only in this case, there will be k-1 # symbols, separating the non-blank tape symbols.)

Simulate the position of each read/write head on the tape by introducing new symbols x dot for every $x \in \Gamma_1 \cup \Gamma_1 \cup \ldots \cup \Gamma_k$ where x dot indicates that the read/write head is at that location.

The remainder of the proof is similar to what was done for the 2-tape machine. This is Theorem 3.13 in the text.