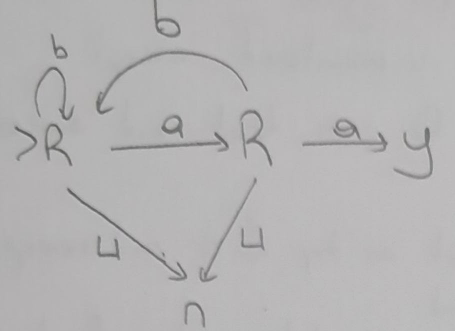
# Homework 4

Q1)




Q2) Move tape head right
If symbol is nonblank save it into variable a
Move tape head right
If symbol is nonblank save it into variable b
Move tape head right until a blank is seen
Write a to the first blank
Move tape head right until blank is found
Write b to the first blank

Q3) No, stay put does not make any difference. We can simulate a TM with stay put (It has transition function from $(K-H)\times\Sigma$ to $K\times(\Sigma\cup\{\leftarrow,\rightarrow,S\})$ with a standard TM as follows:

For every $\delta(q_i,a)=(q_j,S)$ where $q_i\in(K-H)$, $q_j\in(K-H)$ $a\in\Sigma$ replace it with $\delta(q_i,a)=(q_j,\rightarrow)$ and $\delta(q_j,b)=(q_j,\leftarrow)$ where $b\in\Sigma$ and b succeeds a in the tape.

Q4)

a) 1. Move tape head right until current symbol is not crossed. If blank is found accept.

2.1 If symbol is 0 cross that and sweep right until 1 is found

2.1.1 If 1 is found cross that and sweep left until finding a crossed off symbol

2.1.2. If 1 is not found and blank is encountered, reject

2.2. If symbol is 1 cross that and sweep right until 0 is found

2.2.1 If 0 is found cross that and sweep left until finding a crossed off symbol

2.2.2. If 0 is not found and blank is encountered, reject.

3. Repeat Step 1.

Q4)

b)

1 Move tape head right until current symbol is neither crossed nor a dot is put on top of it. If ⊔ is encountered accept.

2.1 If the current symbol is 0 cross that and sweep right until 1 is found.

2.1.1 If 1 is found put a dot on top of it and sweep right until another 1 is found.

2.1.1.1 If 1 is found put a dot on top of it and sweep left until the first crossed off 0 is encountered.

2.1.1.2 If ⊔ is encountered, reject.

2.1.2 If ⊔ is encountered, reject.

2.2 If the current symbol is 1 put a dot on top of it and sweep right until a 0 or 1 is found.

2.2.1 If the current symbol is 0 cross that and sweep right until a 1 is found.

2.2.1.1 If 1 is found put a dot on top of it and sweep left until first 1 with a dot is found.

2.2.1.2 If ⊔ is encountered, reject.

2.2.2 If the current symbol is 1 put a dot on top of it and sweep right until a 0 is found.

2.2.2.1 If a 0 is found cross it and sweep left until finding second 1 with a dot on top.

2.2.2.2 If ⊔ is encountered, reject.

2.2.3 If a ⊔ is encountered, reject.

3 Repeat Step 1.

Q4)

c) Simulate TM in part b, accept if it rejects, reject if it accepts.

Q5) a) Firstly, we need to show that 2-PDAs can simulate 1-PDAs to show they are at least as much powerful as 1-PDAs. On top of it we need to show they can recognize more languages than 1-PDAs.

* Showing the simulation of a 1-PDA with a 2-PDA is trivial. If we never use the 2nd stack, they are equivalent and recognize same languages.

* In second part, we know that 1-PDA does not recognize $L = \{ww \mid w \in \{a,b\}^*\}$ (if you give an unknown example you need to prove that language is not recognized by 1-PDAs before showing 2-PDA's recognition) but 2-PDA recognizes it.

High level description for a 2-PDA that recognizes L.

1. Put all symbols into the 1st stack until the middle of the input string.
2. Non deterministically determine the middle of the input and then pop from 1st stack and push into 2nd.
3. Read rest of the input and match top of the 2nd stack with the current symbol. Reject if there is any mismatch.
4. Accept.

Q5) b) Firstly, we simulate TM with 2-PDA to show equivalence.

1. Push stack bottom symbol \$, ⊔ together with input of TM and push one more \$ to the 1st stack
2. Then push everything until the blank to the 2nd stack.
3. If TM makes a transition to the left as follows:

$$\delta(q_i, w_i) = (q_j, \leftarrow)$$

   3.1 If $w_i$ is not \$ pop $w_i$ from 1st stack and move into the second stack.
   3.2 If $w_i$ is \$ do not change contents.
   3.3 If $q_j$ is a halting state halt and accept/reject.
4. If TM makes a transition to the right as follows:

$$\delta(q_i, w_i) = (q_j, \rightarrow)$$

   4.1 If $w_i \neq \$$ pop from 2nd stack and push into first stack
   4.2 If \$ is at the top of the stack push a ⊔ into the first stack.
   4.3 Same as 3.3.
5. If TM changes the current symbol, pop from 1st stack and push that symbol into the 1st stack.

Then we should simulate 3-PDA with a TM to show their equivalence.

We can use a 4-tape Nondeterministic TM to simulate the 3-PDA
First tape keeps track of the PDA's input.
Other 3 tapes each are used to keep track of stacks of PDA
If a push is made to any of the stacks, the corresponding tape moves its head to right and writes that symbol
If a pop is made to any of the stacks, the corresponding tape writes ⊔ and moves its head to the left.
If stacks do not change TM rewrites the current symbols.
If a symbol is being read from tape of PDA TM moves head of first tape to the right.
If a symbol is not read from tape of the PDA, TM rewrites the current symbol.
Since we can do these simulations 2-PDA and 3-PDA are equivalent in terms of languages they recognize.

Q6) a) Suppose $L_1$ and $L_2$ be two Turing-recognizable languages that have TMs $M_1$ and $M_2$ respectively.
Their union $L_3 = L_1 \cup L_2$ is recognized by following TM $M_3$

$M_3$ = "on input w":
1. Run $M_1$ and $M_2$ on w step by step. If either accepts accept, if both halt and reject, reject.
2. Otherwise loop (does not halt)

b) Suppose $L_1$ and $L_2$ be two Turing-recognizable languages that have TMs $M_1$ and $M_2$ respectively.
Their concatenation $L_3 = L_1 L_2$ is recognized by following TM

$M_3$ = "on input w":
1. Non deterministically divide w into $s_1$ and $s_2$
2. Run $s_1$ on $M_1$, if $M_1$ rejects, reject
3. Run $s_2$ on $M_2$, if $M_2$ accepts, accept. If $M_2$ halts and rejects, reject.

c) Suppose L be a Turing-recognizable language and M recognizes it. $L^*$ is the star of L. TM $M^*$ recognizes $L^*$ as follows:

$M^*$ = "On input w":
1. Divide w into $s_1, s_2, \dots s_n$ non-deterministically
2. For each part, run M, accept if M accepts all. Reject if any part is rejected by M.

d) Suppose $L_1$ and $L_2$ be two Turing-recognizable languages that have TMs $M_1$ and $M_2$. Their intersection $L_3 = L_1 \cap L_2$ have TM $M_3$ as follows:
$M_3$ = "On input w":
1. Run $M_1$ on w. If it accepts, Run $M_2$ on w, else reject.
2. If $M_2$ accepts, accept. If $M_2$ rejects, reject.

Q7) L is decidable. The following TM M decides it.

M = "On input <R, w> where R is a RegExp and w is a string":

1. Convert R to an equivalent NFA A by using the algorithm we learned in Regular Languages
2. Run A on input w.
3. If A ends its simulation in an accept state, accept. If it ends in a non accepting state, reject.