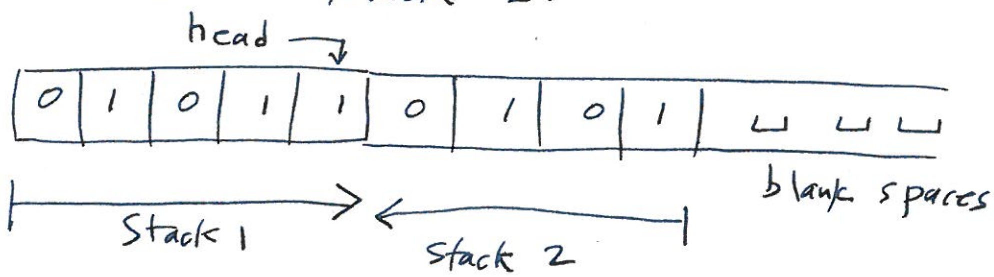
## Problem 1

(a) The language $\{0^n 1^n 2^n \mid n \geq 0\}$ cannot be accepted by a 1-PDA. But we can easily construct a 2PDA for it.

We begin by reading the 0's, for each input 0, push a 0 on both stacks. When we reach the 1's, begin popping one element of the first stack for each 1. When we reach the 2's, we pop one element of the second stack for each 2. If we end with both stacks empty, we accept.

(b) We can simulate a TM with a 2-PDA. The tape to the left on the head, including the head, will be stored in stack 1, while the tape to the right will be stored in stack 2.



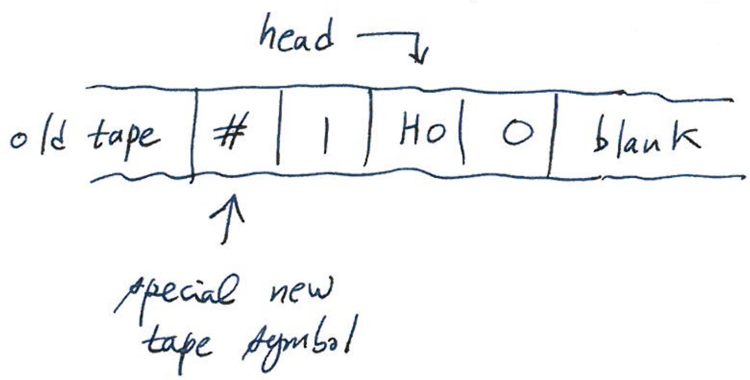

1) read at head : look at top of stack 1
2) write at head : pop stack 1 and push symbol to be written
3) move right : pop $x$ from stack 2, push $x$ onto stack 1
4) move left : pop $x$ from stack 1, push $x$ onto stack 2

To create initial configuration, read $w_1 \dots w_n$ and push on stack 1, then move $w_2 \dots w_n$ to stack 2.

Problem 2.

Using a TM that writes each tape square at most 2 times, we can simulate an ordinary Turing machine. Every time we want to write or move, we recopy the entire tape. The new copy is put at the end of the current tape. To copy the tape we have to write twice due to marking strategy.

We use special extended alphabet to keep track of the head position. e.g.




To convert a write-twice machine to a write-once machine, we represent each square with a pair of squares. The left side will store the first write, and the write side will store the second write. When reading a square, we read the write side if it's not ⊔, otherwise we read the left side.

Given the initial configuration, we can convert it into the two-cell format by copying it to a new tape.

## Problem 3

(a) Given $\delta(q,a) = (r, b, S)$, there are 3 cases:

1) Either the TM will stay put indefinitely: this can be determined since there are at most $|Q| \times |\Gamma|$ possible combinations of states and tape symbols. In this case, we replace the transition by $\delta(q,a) = (q_{reject}, a, R)$

2) the TM will eventually move right and change the symbol to some $c \in \Gamma$. In this case, we change the transition $\delta(q,a) = (q', a, R)$ where $q'$ is the state entered upon moving right.

3) Eventually reach $q' = q_{accept}/q_{reject}$, make $\delta(q,a) = (q', a, R)$.

The modified TM accept the same language.

(b) Since the TM moves only to the right, it can be simulated by an NFA. We change all $\delta(q, \sqcup) = (r, a, R)$ to $\varepsilon$ transitions from state $q$ to state $r$.

Problem 4.

(a) If the language is finite.

All finite languages are decidable and can be enumerated lexicographically.

(b) If the language is infinite.

We need to show both directions. Let A be infinite. Suppose A is decidable. Then we can generate every possible string in lexicographic order. For each string we decide whether it's in A or not. If the string is in A, we output it. This enumerats the language in lexicographic order.

Suppose A is enumerated in lexicographic order. For any string w, we run the enumerator until it generats a string beyond w (this will happen since A is infinite). We can then check if w was enumerated or not and accept/reject accordingly. Thus A is decidable.

## Problem 5

$A_{\varepsilon_{CFG}} = \{ \langle G \rangle \mid G \text{ is a CFG that generats } \varepsilon \}$.

$A_{\varepsilon_{CFG}}$ is decidable. We can convert $G$ into a CNF and check if the rule $S \to \varepsilon$ is there.

## Problem 6

$INFINITE_{DFA} = \{ \langle A \rangle \mid A \text{ is a DFA and } L(A) \text{ is infinite} \}$

This language is decidable.

If a DFA accepts an infinite language it must accept a string $w$ such that $|w| \geq n$ where $n$ is the number of states in the DFA.

Conversely, if a DFA accepts $w$ such that $|w| \geq n$ it's language must be infinite since $w$ must correspond to a sequence of states of length $\geq n+1$ starting from the start state. This means there is a repeating state and, therefore, one can form from $w$ infinitely many strings in the language.

Let $D = \{ w \mid |w| \geq n \}$

observe that $D$ is regular

$$D = \underbrace{\Sigma \Sigma \Sigma \ldots \Sigma}_{n} \Sigma^*$$

we construct a DFA corresponding to

$L(A) \cap D$ and decide whether

the DFA has an empty language or not.

If the language is empty, we reject; otherwise

we accept.