<u>CS 692 Capstone Exam Theory Spring 2022</u>

Choose any 2 of the 3 problems.

1. Given regular expression (ab)*(a|b) over $\Sigma = \{a, b\}$:

   (a). (6 pts) Draw the equivalent state diagram of an NFA.

   (b). (8 pts) Draw the equivalent state diagram of a DFA.

   (c). (6 pts) Write a corresponding context-free grammar for it.

2. For decidable languages:

   (a). (4 pts) Give a definition of decidable languages.

   (b). (8 pts) Prove that decidable languages are closed under union.

   (c). (8 pts) Prove that decidable languages are closed under intersection.

3. For Turing machines:

   (a). (8 pts) Give the state diagram of a Turing machine that recognizes the following language over $\Sigma = \{a, b\}$:

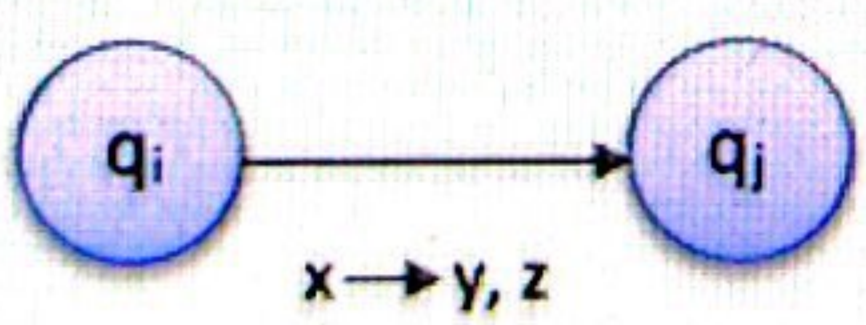
$$L_1 = \{w : w \text{ contains the substring aab}\}$$

   (b). (12 pts) A Turing machine X with Right Tab T is similar to a normal Turing machine except that its transition function is defined as

$$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, T\}$$

   where T is an extra tape directive, in addition to Left and Right, that moves the read/write head to the first blank space to the right. Show that X is Turing-complete.

Note: Please use the following notation to label your Turing machine transitions:

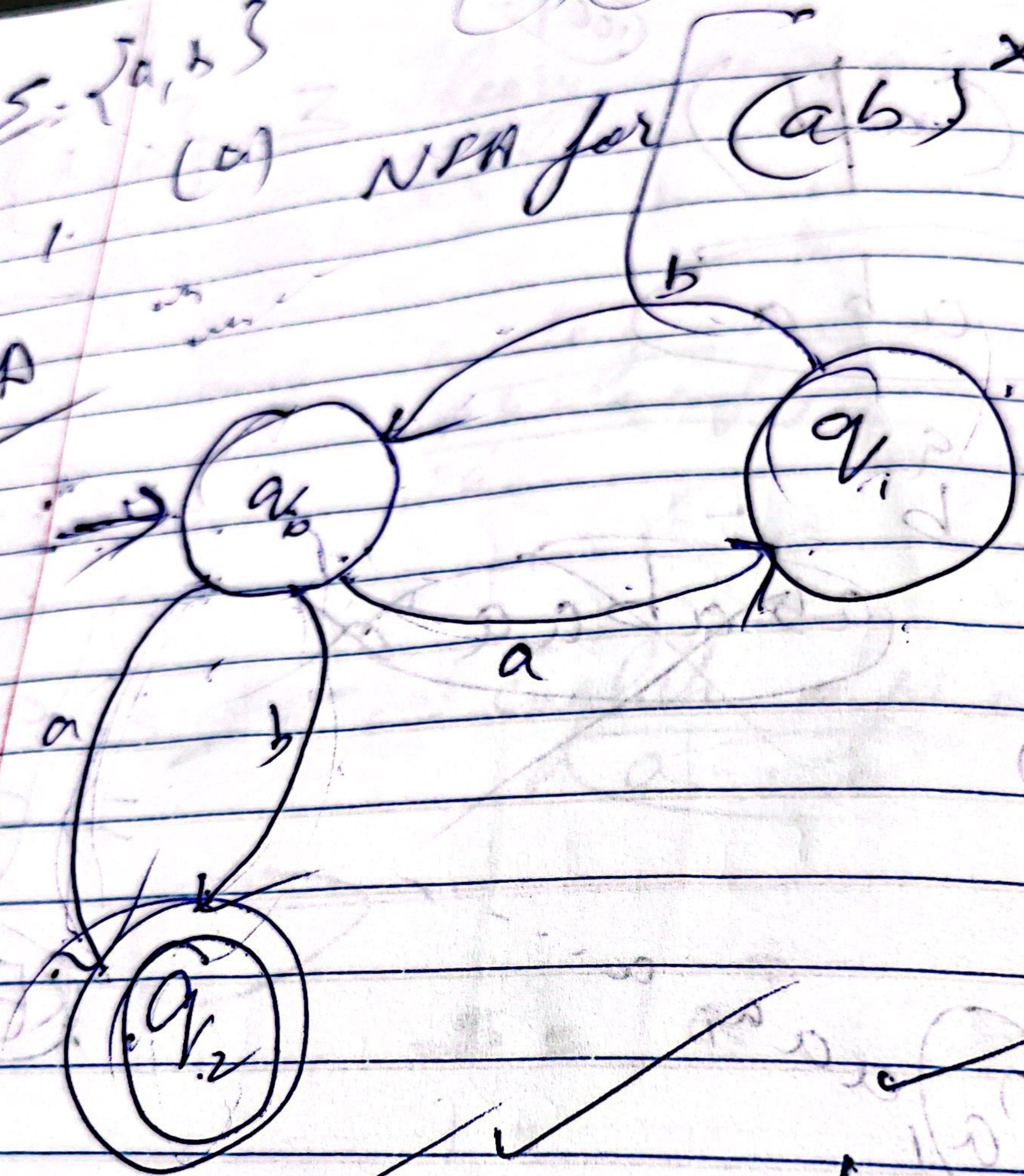


(read symbol x, write symbol y, direction to move is z).

$\Sigma = \{a, b\}$

1. (a) NFA for $(ab)^* (a|b)$

NFA

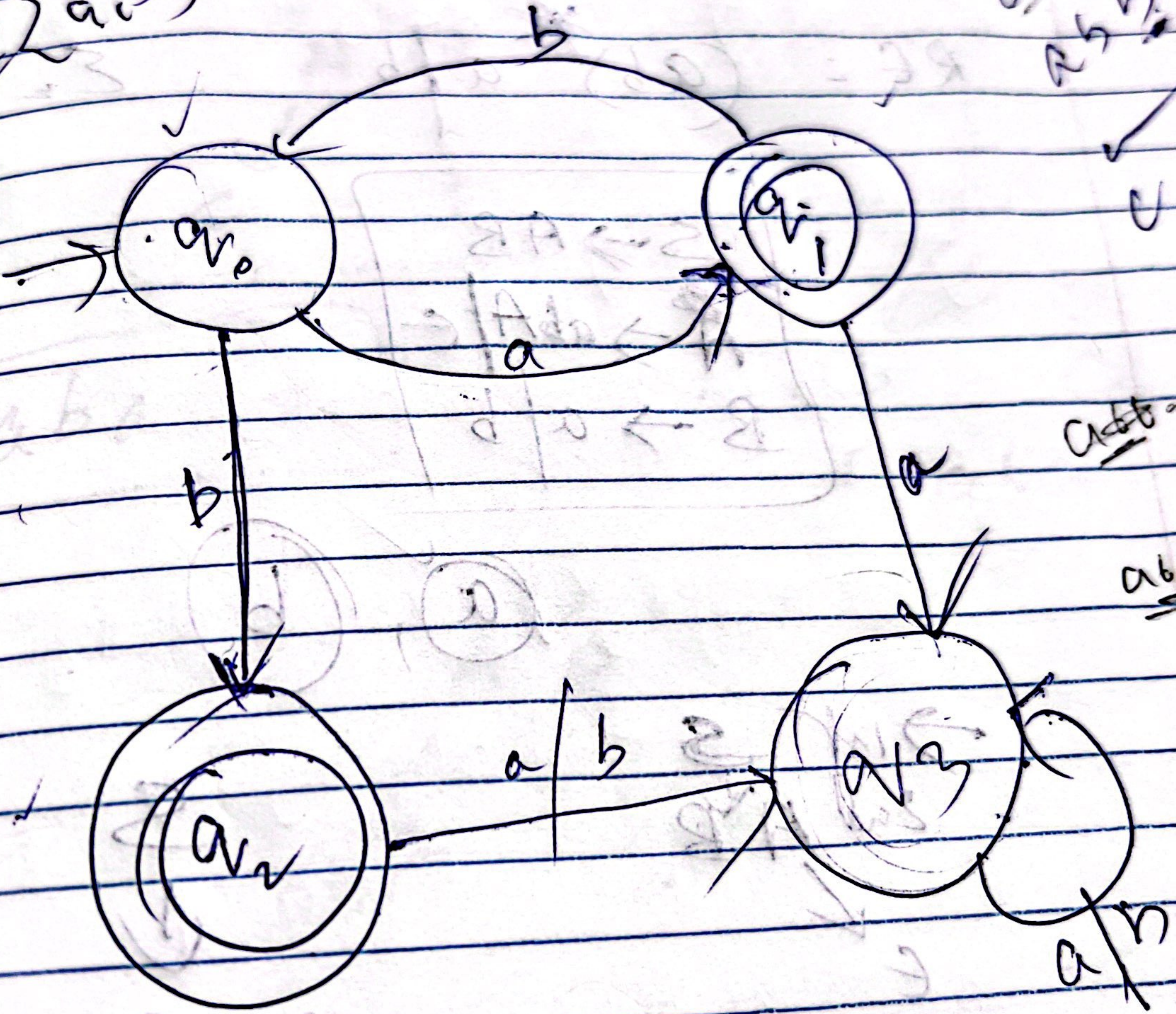


a, b, aba, abb

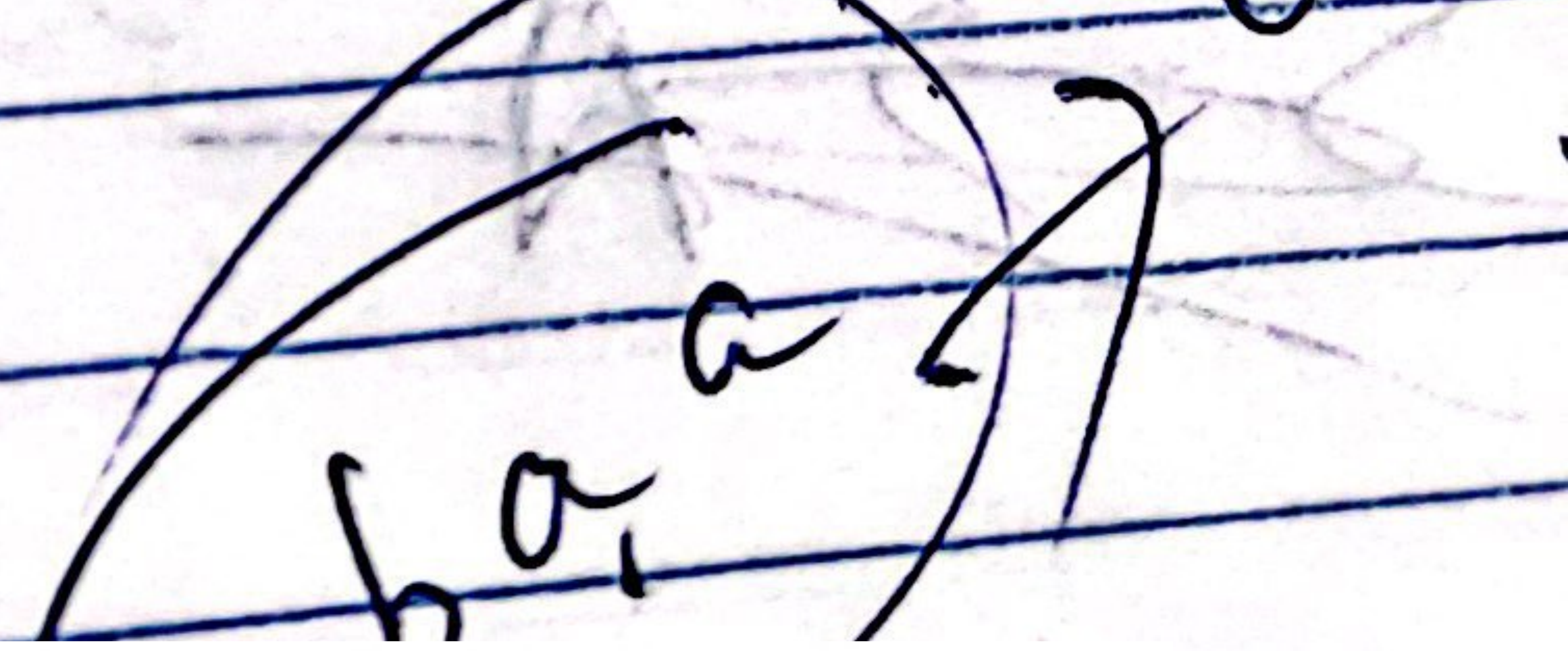
DFA

$\Sigma = \{a, b\}$

b
a
ab
abb



abba
ababa



Step 4/4

C)

CONTEXT FREE GRAMMAR:

Let S be the starting Symbol

S-> aA | bB | aB

A -> bS

B -> e

Here e means null string or epsilon.

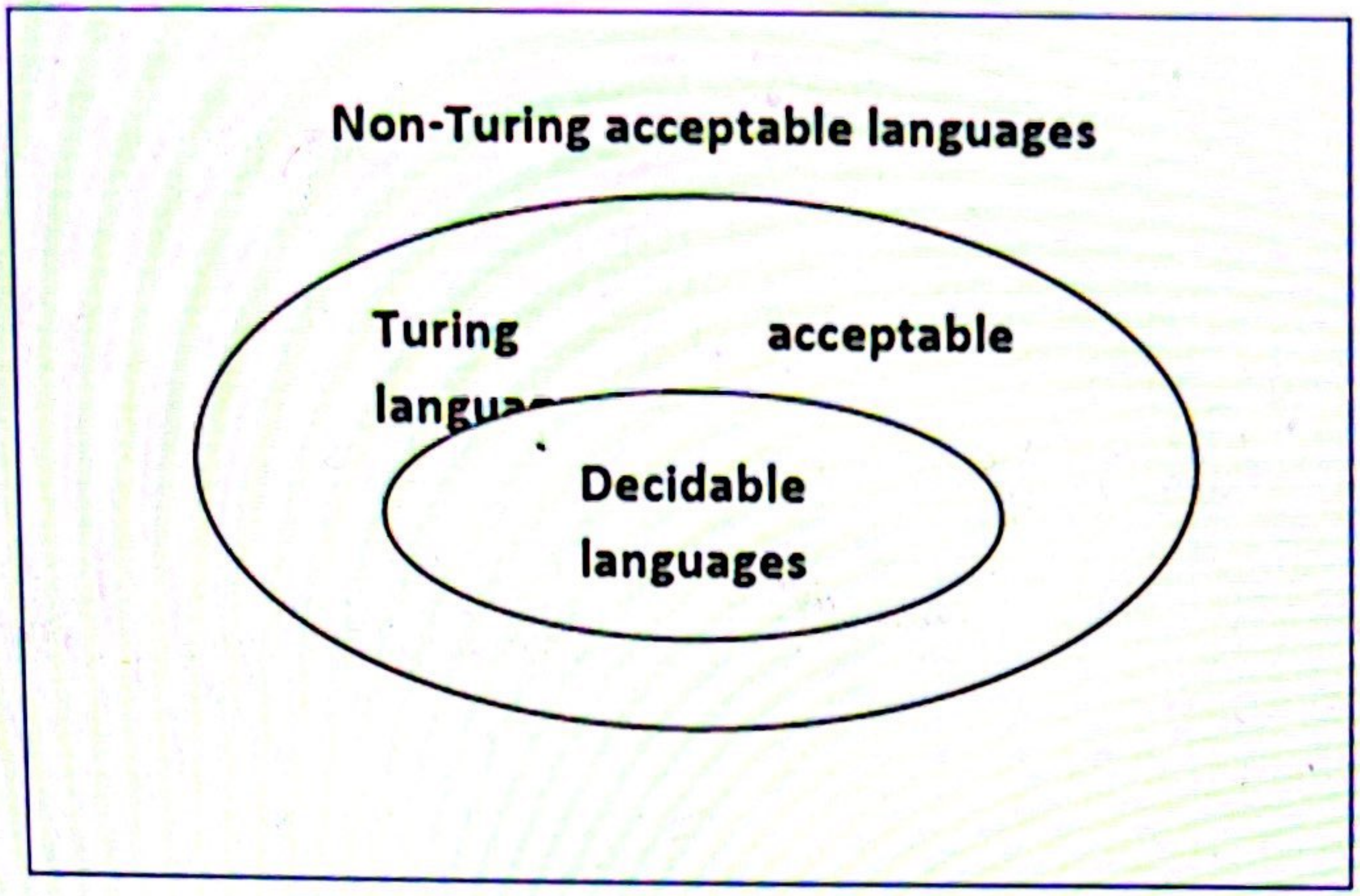
Final answer

Was this answer helpful? 0 0

Questions viewed by other students

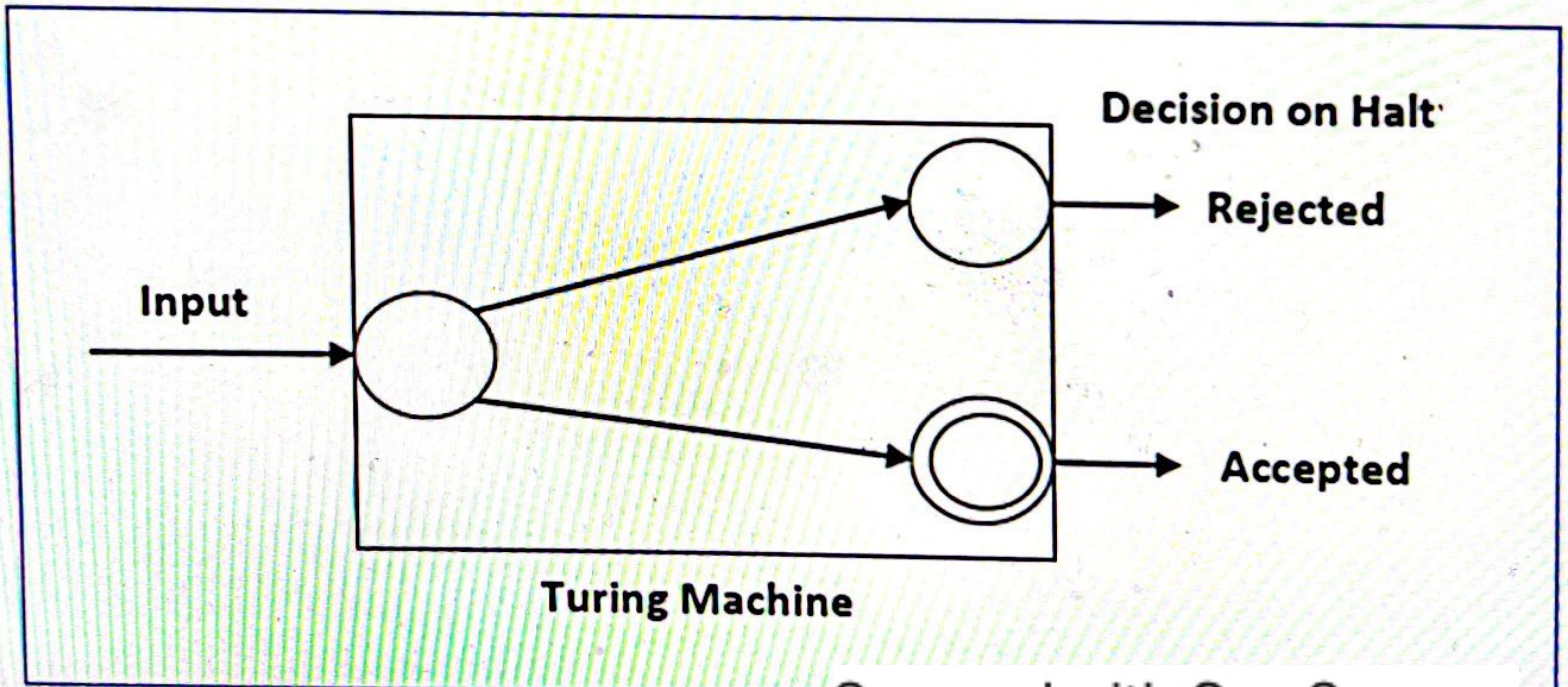
ammars
cation
ar Grammar
troduction
G

A language is called **Decidable** or **Recursive** if there is a Turing machine which accepts and halts on every input string **w**. Every decidable language is Turing-Acceptable.



A decision problem **P** is decidable if the language **L** of all yes instances to **P** is decidable.

For a decidable language, for each input string, the TM halts either at the accept or the reject state as depicted in the following diagram –



(e) **Intersection :** This is again fairly simple. Let K and L be two turing decidable languages, and let M_K and M_L denote the turing machines deciding K and L respectively. Let $M_{K\cap L}$ denote the turing machine deciding $K \cap L$. $M_{K\cap L}$ works as follows.

i. On input w to $M_{K\cap L}$,
ii. Input w to M_K.
iii. If M_K rejects, *reject.*

iv. Else Input w to M_L.
v. If M_L accepts, *accept.* Else *reject.*

(d) **Complementation :** This is fairly straightforward, but the point to note is that *turing recognizable languages* are **NOT** closed under complementation, while *turing decidable* languages are. For a language L, let M_L denote the turing machine deciding L. Then the turing machine for the complement is $M_{L'}$ which on input w, accepts if M_L rejects, and accepts otherwise.

(c) **Star:** For a language L, $L^* = \{x \in L \cup LL \cup LLL \cup \cdots\}$. i.e. all strings obtained by concatenating L with itself, and so on. To show that L^* is decidable, the idea is similar to the previous solution. We want to find cuts of the input string w, such that each of them is accepted by the TM M_L that decides L. Let M_{L^*} be the machine that that decides L^*.

i. On input w : For each way to cut w into parts $w_1 w_2 \cdots w_n$

ii. Run M_L on w_i for $i = 1, \cdots, n$.

iii. If M_L accepts each of the strings w_i *accept.*

iv. If all cuts have been tried without su

(b) **Concatenation:** Let K, L be decidable languages. The concatenation of languages K and L is the language $KL = \{xy | x \in K \text{ and } y \in L\}$. Since K and L are decidable languages, it follows that there exist turing machines M_K and M_L that decide the languages K and L respectively. In order to prove that KL is decidable, we can construct a turing machine that decides KL.

This machine, M_{KL} can use the machines M_K and M_L to decide if a string is in KL or not. The machine can be constructed as follows : Consider an input string w. We need to decide if w is of the form xy for $x \in K$ and $y \in L$. If this is the case, there must be a position at which

This machine, M_{KL} can use the machines M_K and M_L to decide if a string is in KL or not. The machine can be constructed as follows : Consider an input string w. We need to decide if w is of the form xy for $x \in K$ and $y \in L$. If this is the case, there must be a position at which we can partition w into x and y. Since there are only finitely many ways to partition the string, we can try all possibilities and accept if there is such a partition and reject otherwise. We will describe a non-deterministic turing machine since it is easier to describe.

i. On input w, non-deterministically partition w into strings xy.

ii. Input x to M_K and y to y on M_L.

iii. *accept* if both M_K and M_L accept, else *reject*.

If there is an accepting computation path, then we have found a successful split and the string is in KL. If all computation paths reject, then the string is not in KL. In either case, it is easy to see that the machine M_{KL} halts. Thus, KL is decidable.

1 Closure Properties

1.1 Decidable Languages

Boolean Operators

Proposition 1. *Decidable languages are closed under union, intersection, and complementation.*

Proof. Given TMs M_1, M_2 that decide languages L_1, and L_2

- A TM that decides $L_1 \cup L_2$: on input x, run M_1 and M_2 on x, and accept iff either accepts. (Similarly for intersection.)
- A TM that decides $\overline{L_1}$: On input x, run M_1 on x, and accept if M_1 rejects, and reject if M_1 accepts.

$\square$