Turing Machines--
Machines entering into infinite loop

Supplement Slides

Recursively enumerable

- If $w \in L(M)$, then M accepts/recognizes the string w.
- If, $w \notin L(M)$, then M may or may not halt.
  - No transition means M halts and rejects.
- We say $L(M)$ for a given TM M is recursively enumerable (RE).

Recursive languages

- Recursive languages (R) is a subset of RE.
- We say $L(M)$ for a TM M is recursive, if for any given input string w, M halts.
- That is, if $w \in L(M)$, M halts in an accepting (final) state.
- Else, M halts in a non-final state.
  – i.e., It gets stuck in a non-final state.
- M never goes in to an infinite loop.

RE Vs. R

RE

- A TM M recognizes.

R

- A TM M decides.

RE Vs. R

RE

- A TM M recognizes.
- If $L \in RE - R$, then complement of L, that is $\overline{L} \notin RE$.

R

- A TM M decides.
- $L \in R \Leftrightarrow \overline{L} \in R$

RE Vs. R

RE

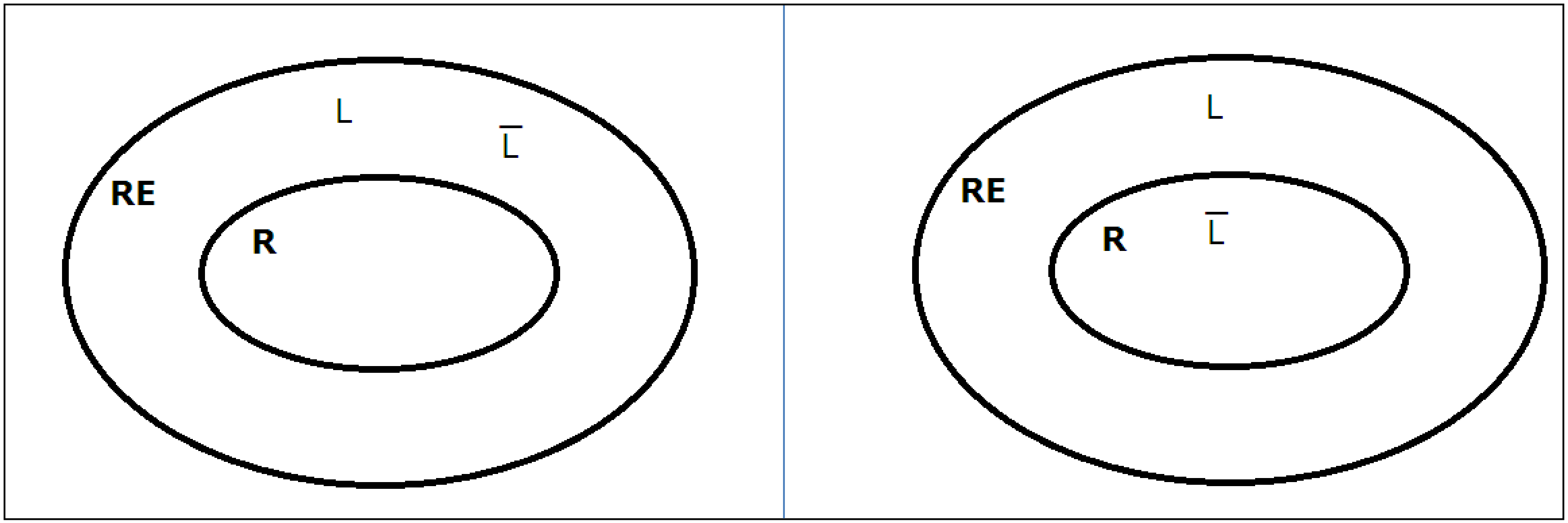
- A TM M recognizes.
- If $L \in RE - R$, then complement of L, that is $\overline{L} \notin RE$.

R

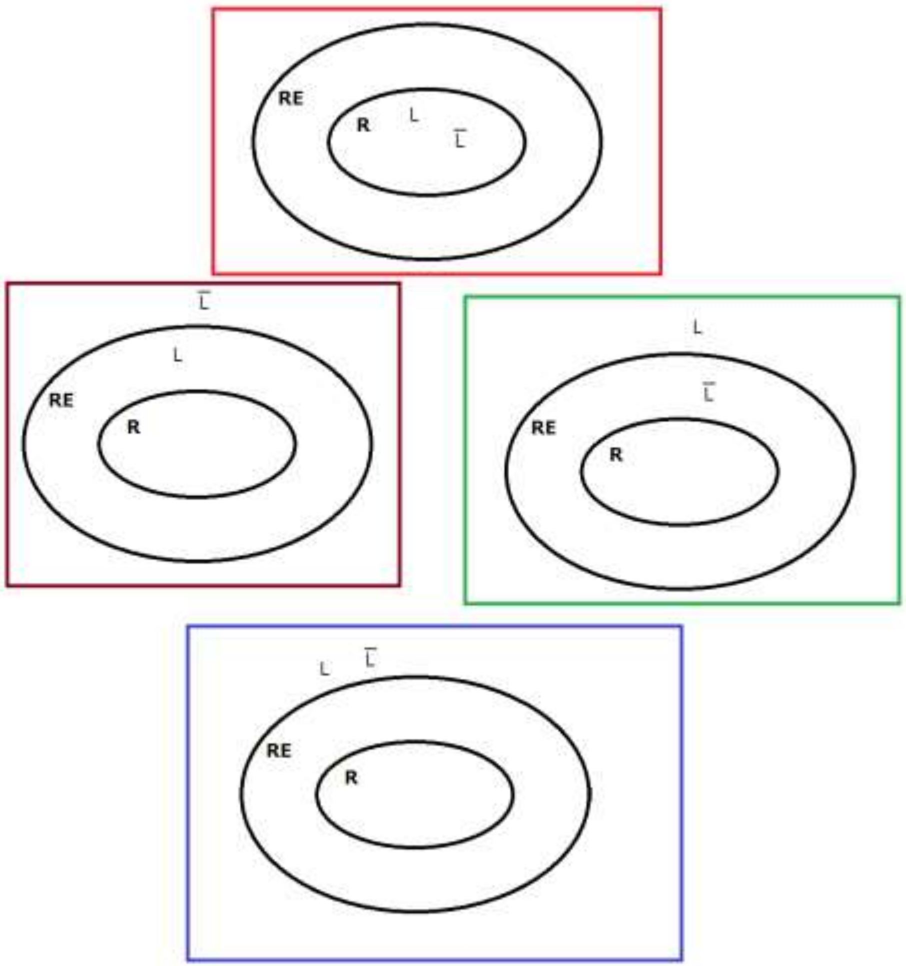
- A TM M decides.
- $L \in R \Leftrightarrow \overline{L} \in R$

$L \in RE \ and \ \overline{L} \in RE \quad \Rightarrow \quad L \in R$

Not Possible



Possible


RE
R
L
$\overline{L}$
$\overline{L}$
L
RE
R
L
$\overline{L}$
RE
R
L
$\overline{L}$
RE
R

Recognizing or accepting a language by a machine (automaton)

- Let us define ID for a DFA/NFA as
  - (q, x) where q is the current state and x is the remaining input.
- We say L is recognized/accepted by a machine M (may be a DFA/NFA/PDA/TM)

$$L = \{w \in \Sigma^* | id_0 \vdash^* id_f \text{ where } id_0 \text{ and } id_f \text{ are initial and accepting ids, respectively}\}$$

DFA/NFA

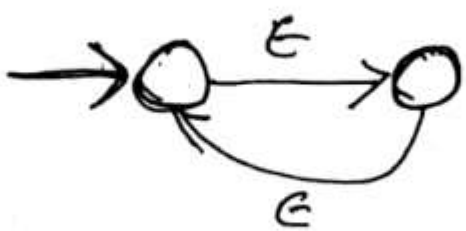
- $L = \{w \in \Sigma^* | \left(q_0, w \right) \vdash^* \left(q_f, \epsilon\right) \text{ where } q_f \text{ is a final state}\}$
- Input has to be exhausted. { The remaining string should become ϵ }
- This is the same criterion even for NFAs.

DFA

- A DFA **never** enters in to an infinite loop.
  - Since there are no ϵ transitions, and
  - There is exactly one choice at every stage of the computation.
  - The input has to exhaust progressively (one character at each step) and should become ϵ
    - At this stage if the state is one of final, the input string is accepted, else rejected.

NFA

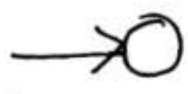
- A NFA can enter in to an infinite loop.



- For any given input this NFA enters in to an infinite loop.
- The language recognized by this machine is ϕ
- The language ϕ is regular, because there is a NFA to recognize this.

For this language we can find NFA and DFA that always halts.

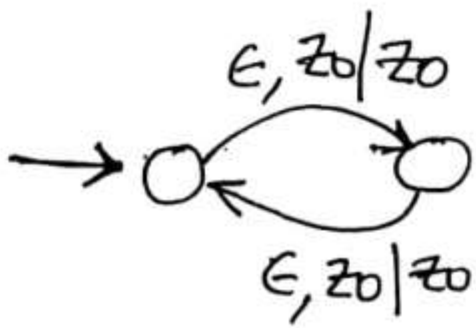
- NFA

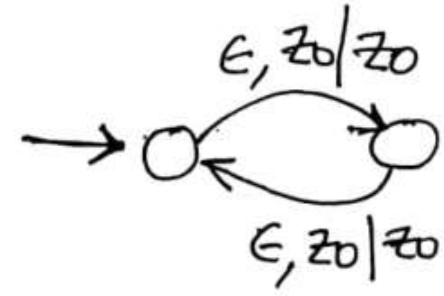


- There is a DFA (existence of either NFA or DFA is enough) also to accept ϕ. {DFA always halts}.

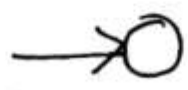
PDA by final state and empty stack

- For a PDA to recognize a language by final state, $L = \{w \in \Sigma^* | \ (q_0, w, Z_0) \ \vdash^* \left(q_f, \epsilon, \alpha\right)$ where q_f is a final state and $\alpha \in \Gamma^*\}$
- A PDA can also enter in to an infinite loop





- The language accepted by this PDA is also ϕ.
- we can find a PDA which recognizes the language ϕ without entering in to an infinite loop.



- Have you noted, both these PDAs are indeed DPDAs.

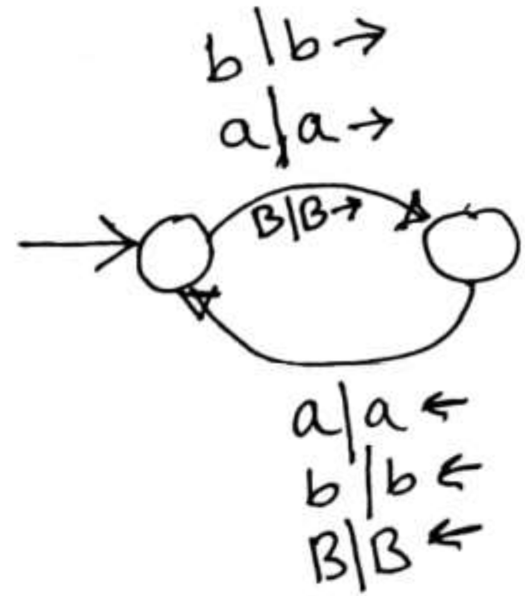
- For PDA by empty stack also similar arguments can be given.

TM

- For the given DTM

also the language

recognized is ϕ.



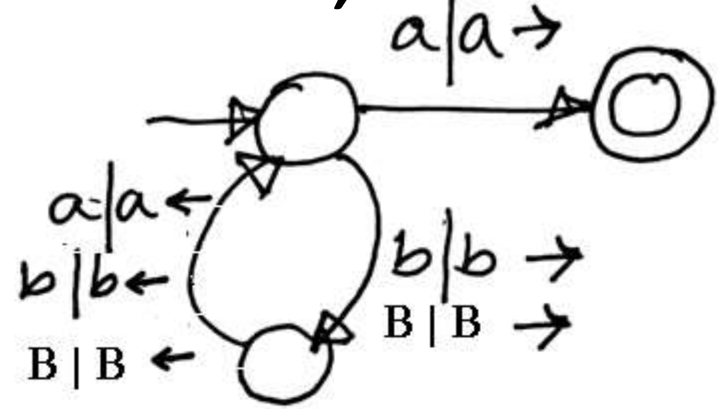
- Again there is a DTM that

Recognizes this language without entering in to infinite loop, which is

- So, $\phi \in RE$

- In fact $\phi \in R$ also.

To elucidate further,



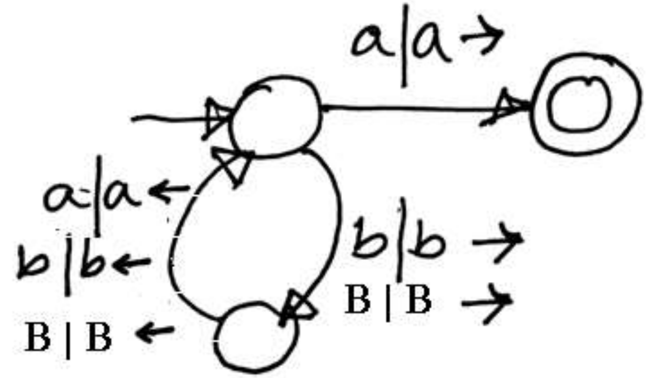
- The given DTM

recognizes the language

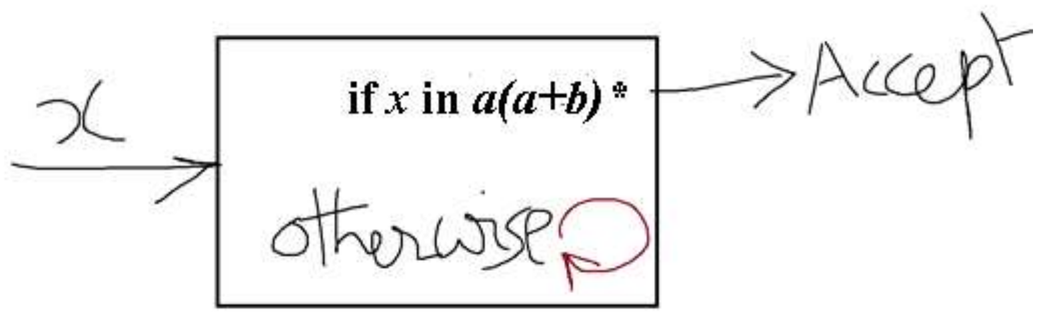
$a(a+b)^*$, but enters in to an

infinite loop for all other inputs.

- So, the language recognized by this DTM is $a(a+b)^*$
- This is in RE.



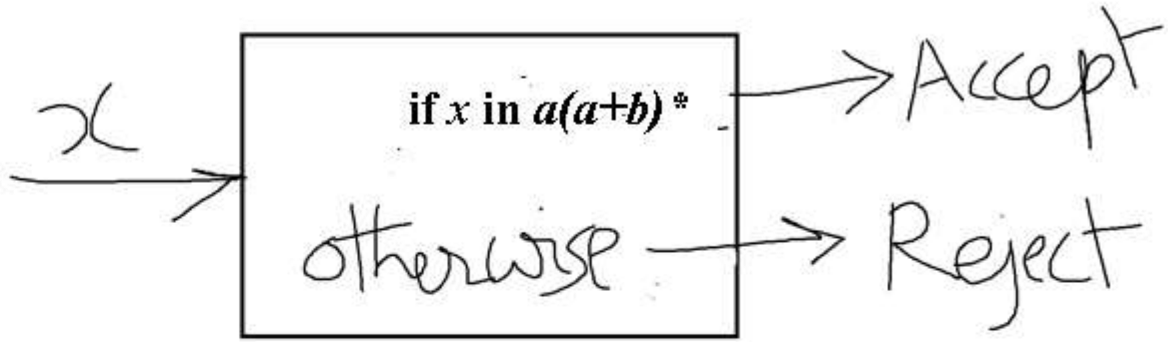
- Behaviour of this machine is



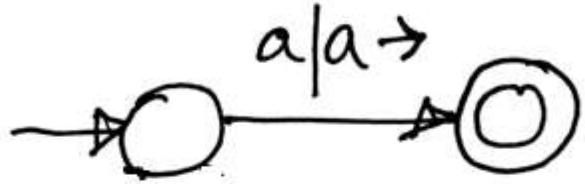
- so, the language $a(a+b)^* \in RE$

Can we say $a(a+b)^*$ is in R ?

- Yes.



- We need a TM which behaves like above



- Note, existence of one such TM is enough.

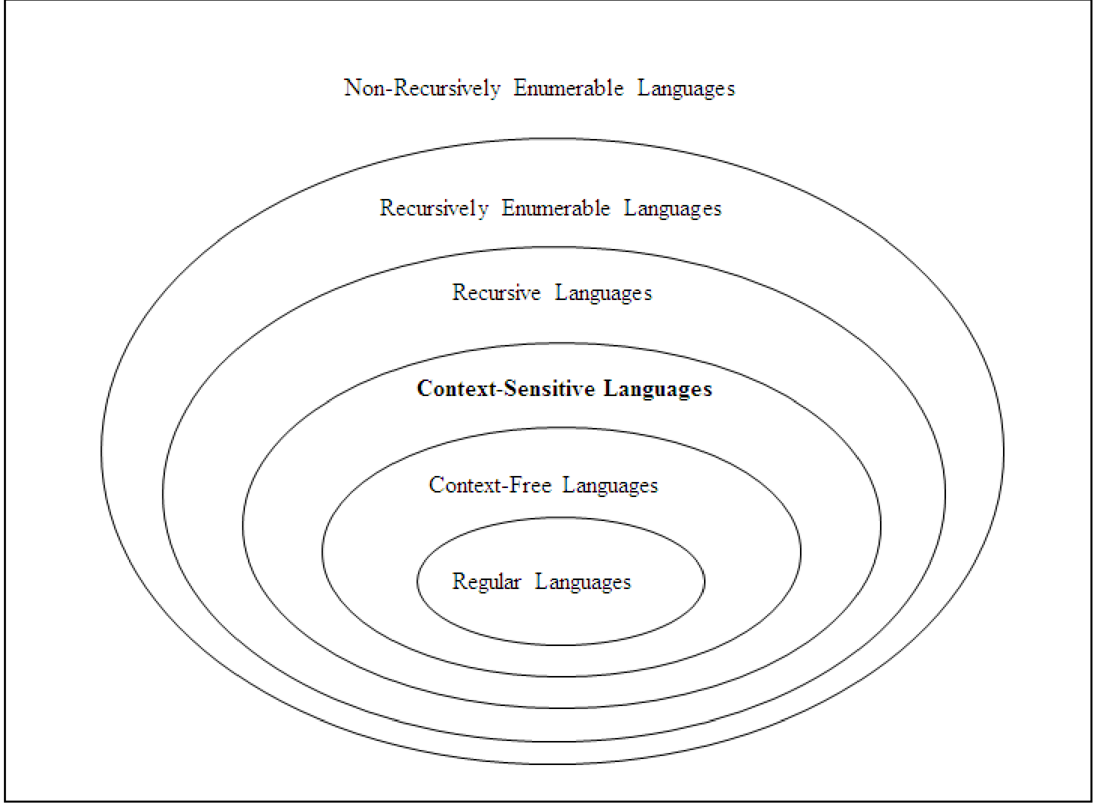
- Are there languages which are in RE but not in R ?

- Yes.
- Finding an $L \in RE - R$ is a challenging task.

Finding an $L \in RE - R$ is a challenging task

- That is, for any string in L, the machine should halt in a final state.
- There is a string which is not in L, for which every possible TM (which accepts L) enters in to an infinite loop.

- There exists such languages !!
- This one important aspect of the theory of computation.
- This, indeed points at limitations of computing machines.



- This diagram is saying that regular, context free and some other languages are recursive.
- That is, all these languages are Turing Decidable.

- We can say, for a regular language there exists a DFA (and also a NFA) that decides the language.
- Similarly for a CFL, a PDA that decides the language always exists.

- So, we do not need powerful TMs to decide regular and CFLs.