Lecture 29

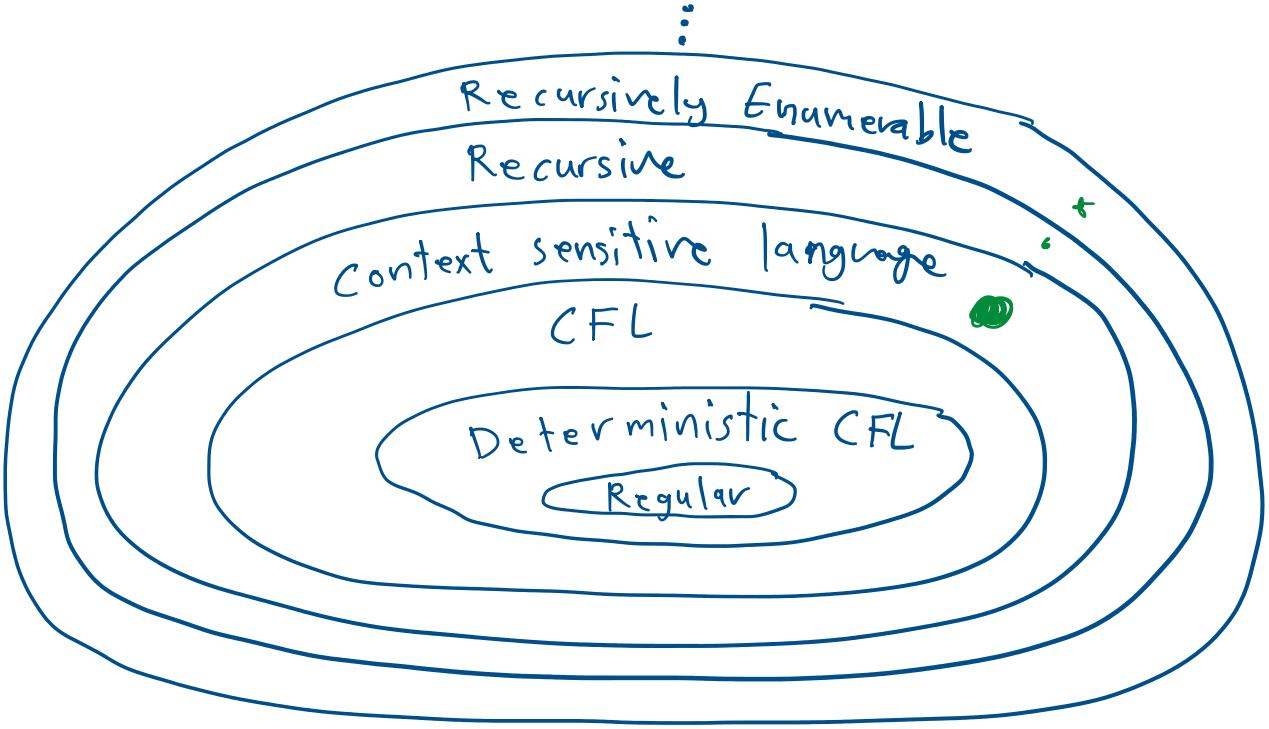
Tuesday, April 2, 2024 11:02 AM

$A = \{ ww : w \in \{a,b\}^* \}$

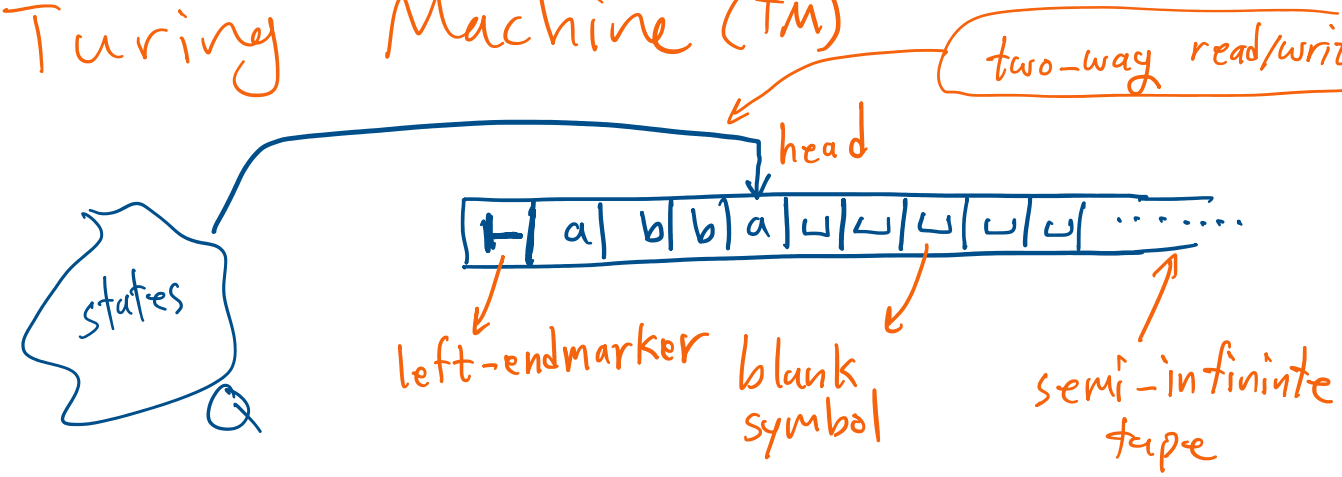
$B = \{a,b\}^* - A = \neg A$

B is CFL, but not A

$\Longrightarrow$ neither is Deterministic CFL.



Turing Machine (TM)



Informal description of deterministic TM:

* The input string is initially written on the beginning of the tape. Also, the start state is S and the head is positioned on the left-endmarker.

* Depending on current state and the symbol written below the head, the machine "act":
  (i) it writes a new symbol under the head
  (ii) it moves the head one step either to left or right.
  (iii) it updates the state.

* There are two special states, q_r and q_t. As soon as we go to these states, the machine halts and accepts (for q_t) or rejects (for q_r) the input string.

* Sometimes the machine loops forever.

But what languages can be described by a TM? It turns out that even [illegible] definition of a

by a TM. It [illegible]

if we change the defenition of a TM somewhat drastically, it will still have the same power.

* TMs with two tape
* TM with infinite tape from left and right.
* C++ language, python, ...
* λ- calculus.
* Post systems
* μ- recursive functions

All of these are powerful enough to simulate each other, and they captare the intuitive notion of computation.

You can have a universal Turing Machine that simulates other TMs (or even other notions of computation).

Deterministic TM:

A deterministic TM is a 9-tuple

$M = (Q, \Sigma, \Gamma, \vdash, \sqcup, \delta, s, q_t, q_r)$ where:

Q: finite set of states

Σ: " " " input symbols

Γ: " " " tape symbols, $\Sigma \subset \Gamma$

$\sqcup \in \Gamma - \Sigma$: blank symbol

$\vdash \in \Gamma - \Sigma$: left-endmarker

$s \in Q$: start state

$q_r \in Q$: reject "

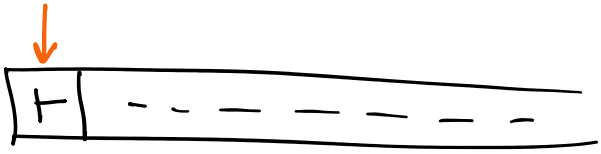
$q_t \in Q$: accept ".

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

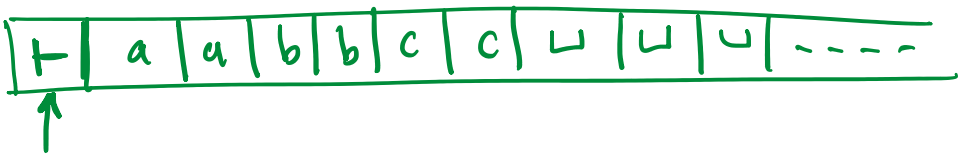
We always assume δ satisfies

$\forall q \in Q, \quad \delta(q, \vdash) = (q', \vdash, R)$

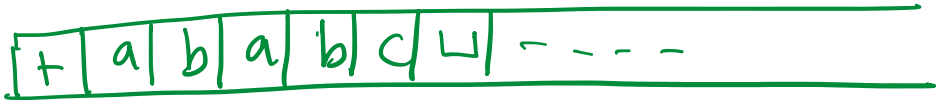
for some $q' \in Q$



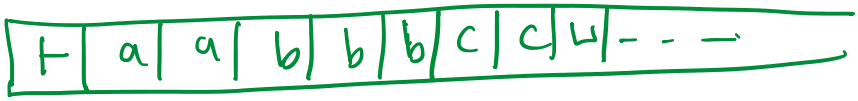
Example: $A = \{a^n b^n c^n : n \geq 0\}$



$x \in A$



$x \notin A$



$x \notin A$

High-level idea:

* First go one pass over the input and check if $x \in L(a^*b^*c^*)$ otherwise reject x.
* Do multiple passes over the input. In each pass, replace one "a", one "b", one "c" with "#".
* Accept x if "a", "b", and "c" run out on the same pass. (otherwise reject)

$\Gamma = \{a, b, c, \sqcup, \vdash, \#\}$

	⊢	a	b	c	␣	#
S	$(s, \vdash, R)$	(s, a, R)	(q_1, b, R)	(q_2, c, R)	$(q_3, \sqcup, L)$	—
q_1						
q_2						
q_3						
q_4						
q_5						
q_6						