10-10-17

$\boxed{\text{LAST CLASS}}$

CFG Chomsky normal form (show → prove)

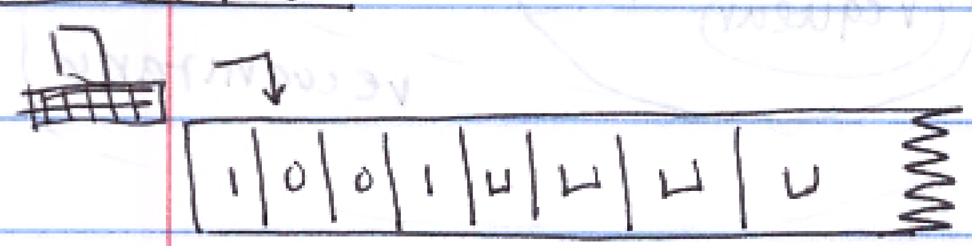
$A \to B\ C$

$A \to a$

x has length n

2n-1 steps

TURING MACHINES



Formal definition: Turing machine (TM)

TM is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ where:

① Q is a finite set of states
$\sqcup \notin \Sigma$ ② Σ is a finite set of input alphabet
③ Γ is a finite set of is tape alphabet $\sqcup \in \Gamma$, $\Sigma \subseteq \Gamma$
⑤ $q_0 \in Q$ is the start state
⑥ $q_{acc} \in Q$ is accepting state.
⑦ $q_{rej} \in Q$ is rejecting state

④ Transition fcn $\delta: Q \times \Gamma \longmapsto \underline{Q} \times \underline{\Gamma} \times \underline{\{L, R\}}$

A TM M computes as follows:

① initially: leftmost n squares of tape hold input $w = w_1 w_2 \dots w_n \in \Sigma^*$
rest of tape is $\sqcup$ symbols

② once M starts, follow transition fcn δ

③ if M tries to move it's head off left end of tape, head stays put over cell 1

④ computation continues until M enters q_{accept} or q_{reject}, at which point it halts.
otherwise, it goes on forever

configuration of TM: consists of

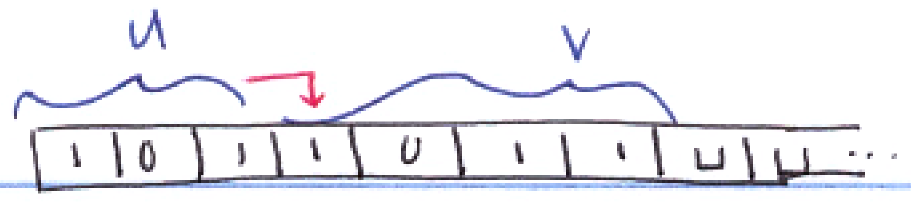
① head location
↳ represented uqv — head is on the 1st symbol of v.
$u, v \in \Sigma^*$ — uv are tape contents
q — current state $q \in Q$

② tape contents

③ state

yields: configuration c_1 yields configuration c_2 if M can go from c_1 to c_2 in 1 step
(1 read, 1 write, 1 move head L or R)

example: $\overbrace{101}^{u}\ q_7\ \overbrace{1011}^{v}$



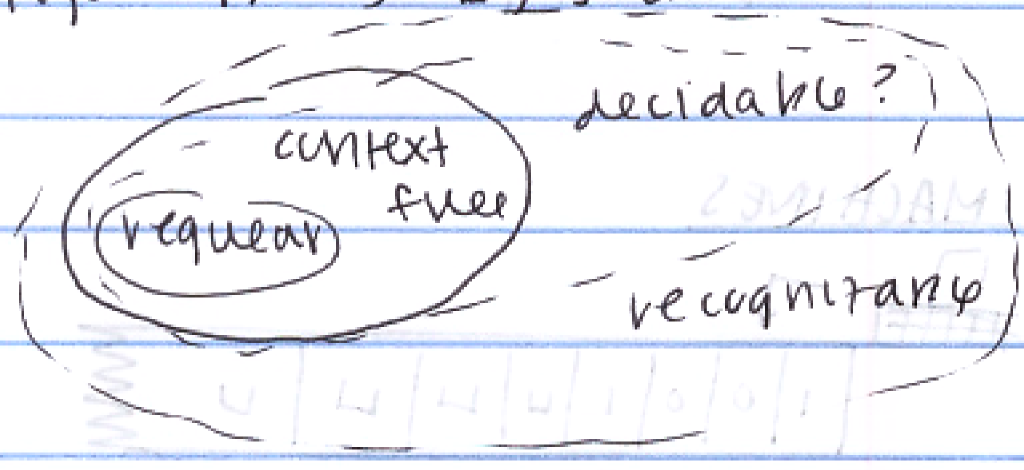
current state: q_7

starting config: $q_0 w$, $w \in \Sigma^*$

- A config is accepting / rejecting if it's state is q_{acc} / q_{rej}; resp.

Formal def of acceptance

A TM accepts input w if $\exists$ a sequence of configs $c_1, \dots, c_k$, s.t.
1. c_1 is start configuration
2. each c_i yields c_{i+1}
3. c_k is accept config



L(M): set of strings accepted by M.

Turing - recognizable (or recursively enumerable)

A language is recognizable if some TM recognizes it
↳ given input x:
1. if $x \in L$: TM halts & accepts
2. if $x \notin L$: TM halts & reject OR loops forever.

Turing decidable

A language is decidable if some TM decides it.
↳ given input x:
if $x \in L$: TM halts & accepts
if $x \notin L$: TM halts & rejects
NO LOOPING FOR EVER!

Notes: if L is decidable $\Rightarrow$ L is recognizable.
if L is recognizable $\not\Rightarrow$ L is decidable.

Example: of Decidable languages

$L = \{ 0^{2^n} \mid n \geq 0 \}$. Let's design a TM M to decide L.

e.g. 0, 00, 0000, ...

PROBLEM: Given input $w \in \mathbb{N}$ (in unary), is w a power of 2?

idea: keep dividing by 2. 2 cases: (1) eventually get the number 1: w is a power of 2.
(2) get odd number $> 1 \Rightarrow$ w not a power of 2!

prime factorization: can write any natural number uniquely as product of prime.

e.g. $10 = \cancel{2} \cdot 5$

power of 2: $2^n = 2 \cdot 2 \cdot \ldots \cdot 2$

$M =$ " on input $w \in \{0\}^*$

1. If tape empty, i.e. first cell contains ⊔, reject
2. Sweep head left to right across tape:
   ① if see single 0 ⇒ accept
   ② else if we see odd number >1 of zeroes, reject
3. Sweep right to left and "cross off" every second zero. Return to step 2.

↳ always halts; ③ reduces length of input each time ⇒ decides L.