CSCE 355
4/20/2022

① Recall:

$ALL_{CFG} := \{\langle G \rangle : G \text{ is a CFG and } L(G) = \Sigma^*, \text{ where } \Sigma \text{ is the terminal alphabet of } G\}$

Theorem: ALL_{CFG} is undecidable.

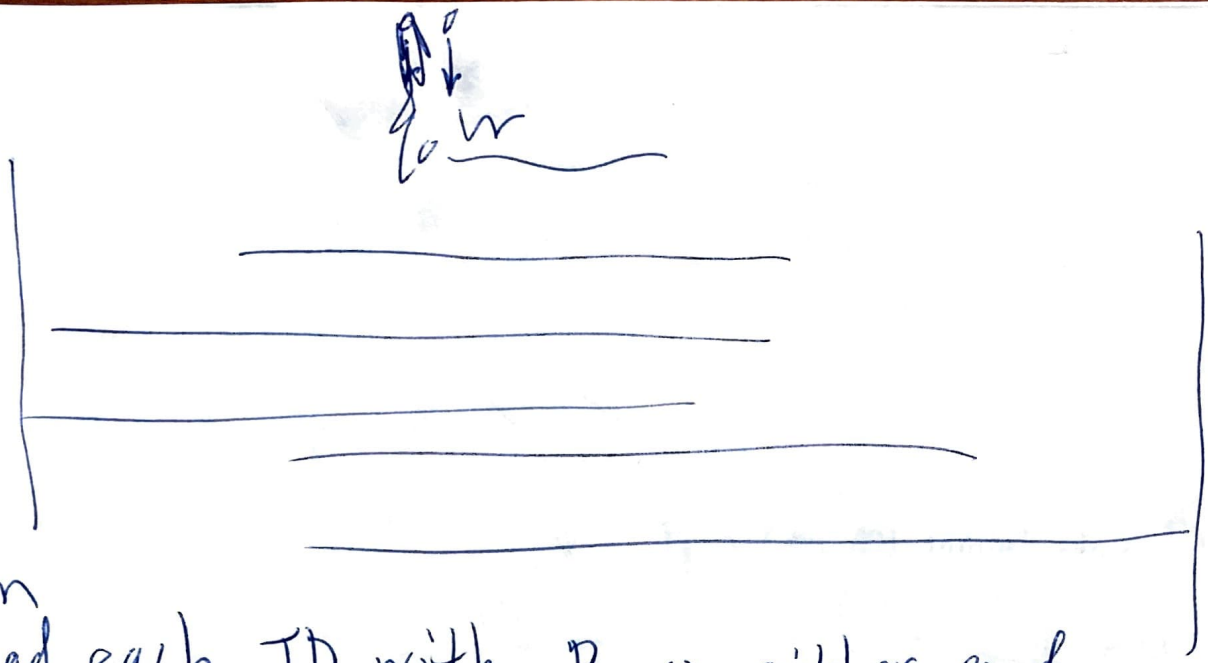
Proof: Given any TM $M = \langle Q, \Sigma, \Gamma, \delta, q_0, B, F \rangle$ and string $w \in \Sigma^*$. Construct a PDA P that accepts all strings (over its alphabet) iff M does not halt on input w.
Then convert P to an equivalent CFG G.

Given M, w as above, if M halts on w there is a sequence of IDs of M $ID_0, ID_1, \ldots, ID_k$ (some k) such that $ID_0 = q_0 w$, $ID_0 \vdash ID_1 \vdash \cdots \vdash ID_k$, ID_k is a halting ID (δ is undefined in ID_k, so no successor).
The rigid computational trace of M on w is a string:

$\$^n \# \underline{ID_0} \# \underline{ID_1} \# \cdots \# \underline{ID_k} \$$

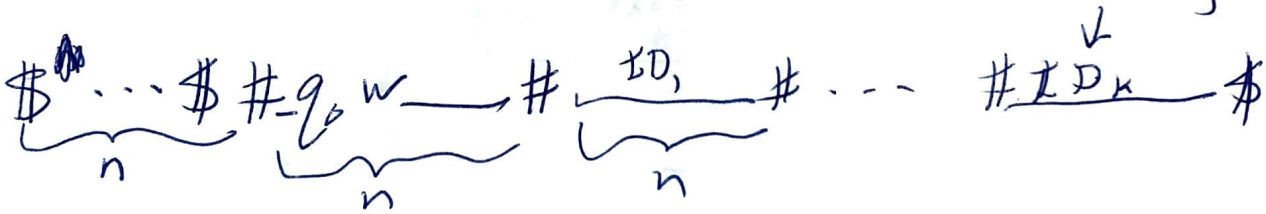
and each ID_j is a string of length n, giving the tape contents in the same location for each j
n is least such that this string exists.

(2)



can pad each ID with B on either end to produce a rigid comp. trace if M halts on w. Such a trace exists, then iff M halts on input w.

Our PDA P will accept all strings except for rigid computational traces.

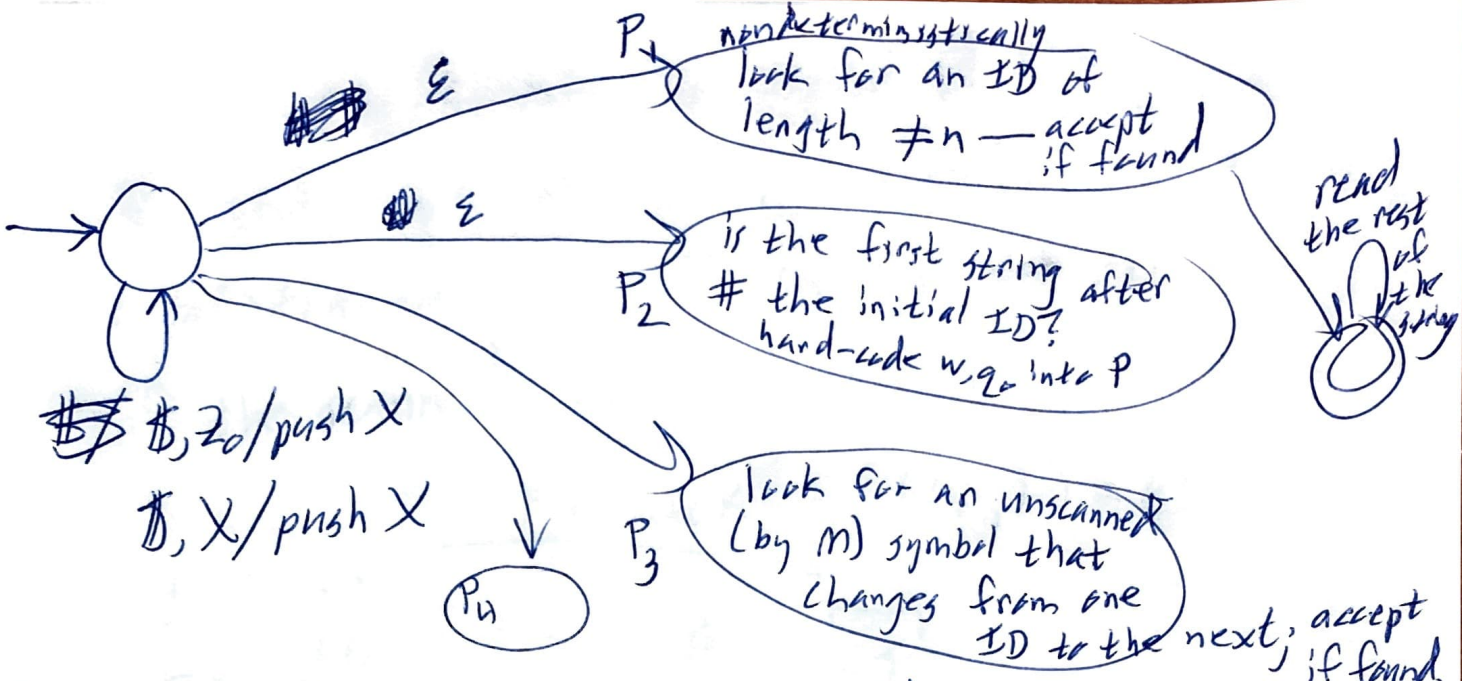


P has 2 stack symbols: Z_0 and X

P has input alphabet $Q \cup \Gamma \cup \{\#, \$\}$ ($\# \notin Q \cup \Gamma$, $\$ \notin Q \cup \Gamma$)

P will split nondet. into a handful of modules, each looking for a different kind of "error" (meaning the string is not a rigid comp. trace of M on w) and will accept if it finds one.

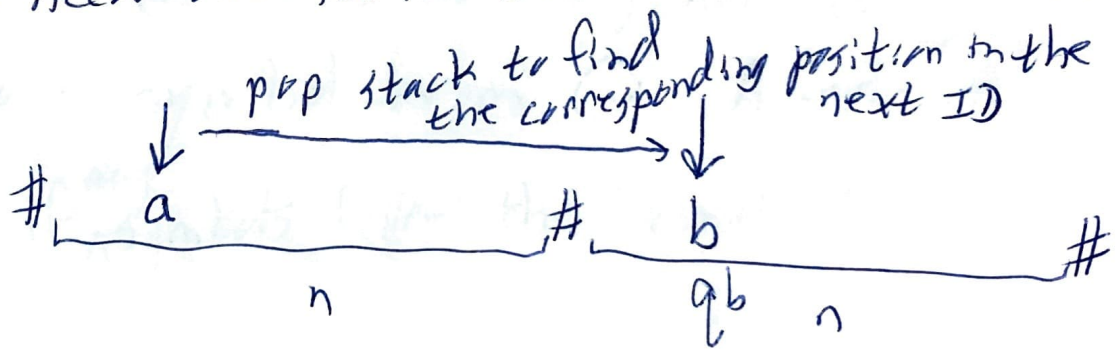
(3)



P_1 nondet. chooses an ID, then reads the ID, popping its stack. If it sees the next # on input before Z_0 is exposed, or if Z_0 is exposed before seeing the next #, then accepts.

P_2 doesn't need the stack (behaves like a DFA).

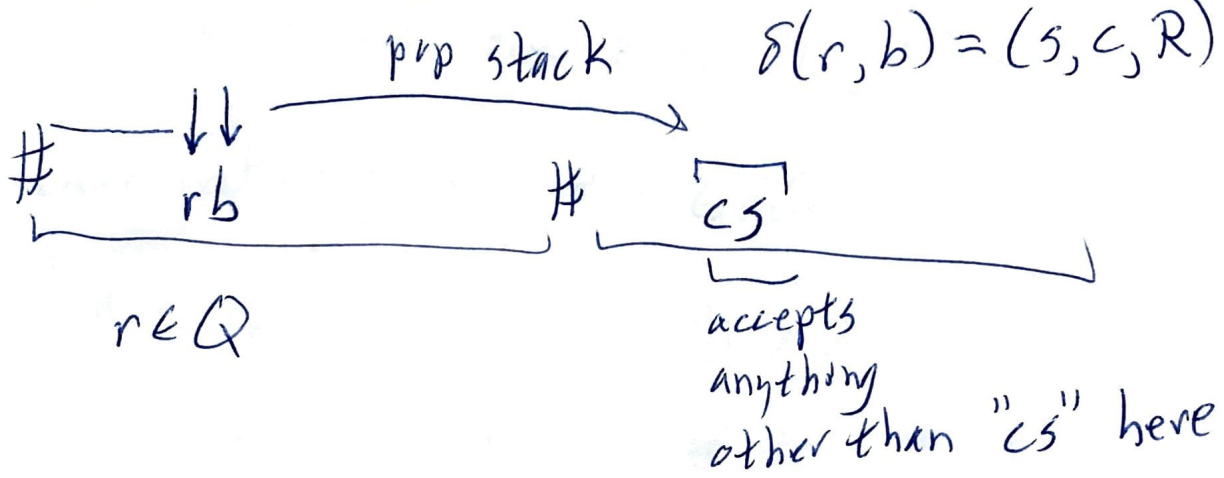
P_3



P_4 checks if the last ID is halting. If not accept; else reject

checks if $\delta(q, a)$ is undefined. If undefined, rejects else accepts. Doesn't need its stack.

(4) P_5 ~~check~~ picks two successive ~~#~~ ID_5, checks if the 2nd ~~#~~ ID is consistent with a legal transition of M in the ~~this~~ vicinity of the scanned symbol:



P_6: checks that each string between successive #'s or (# and $ at end) has exactly ~~any~~ one symbol from Q. Accepts if not.

P_7: Accept if $\overset{\text{any}}{\wedge}$ symbols follow the "final" $.

Summary: P can be constructed by a computer program with M and w as input.

- P only rejects rigid comp. traces of M on w and these exist iff M halts on w.
- Can write a computer prog. to convert P into an equivalent grammar G, so $L(G) = (Q \cup \Gamma \cup \{\#, \$\})^*$ iff M does not halt on w. If you could decide the former, then you can decide the latter, and know that's ~~impossible~~

(5) impossible. So ALL_{CFG} is undecidable ///

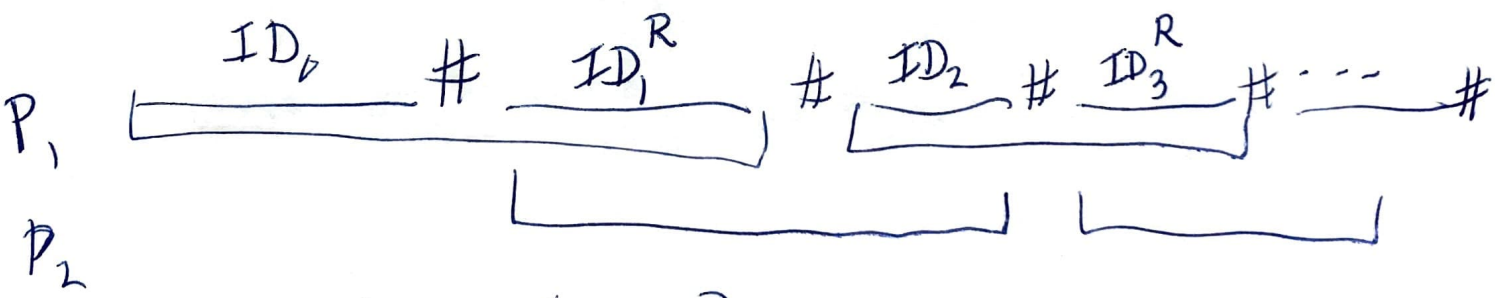
Intersection problem for CFGs:

Given two CFGs G_1 and G_2,
is it the case that $L(G_1) \cap L(G_2) \neq \emptyset$?

Thm: Intertection problem for CFGs is undecidable.

Proof idea: Given any TM M and string w, construct (in a computable way) ~~~~ two CFGs G_1, G_2 such that M halts on w iff $L(G_1) \cap L(G_2) \neq \emptyset$.

Suppose M halts on w. $ID_0 \vdash \cdots \vdash ID_k$



Idea behind P_1 ($\&$ P_2):