Announcements:

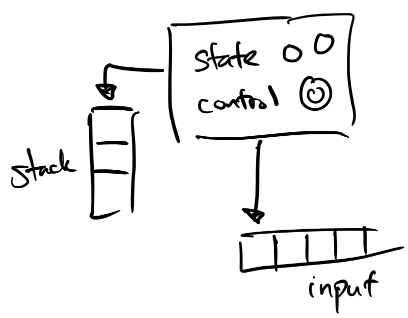
- HW2 back soon (tomorrow?)
  - first 45 mins: past HW (5:15-6)
  - second 45 mins: current stuff
- HW3 due tonight
- HW4 up today

Today:

0. Review
1. The PL for context-free languages
   // why? proof & logic practice
2. Turing Machines (!)

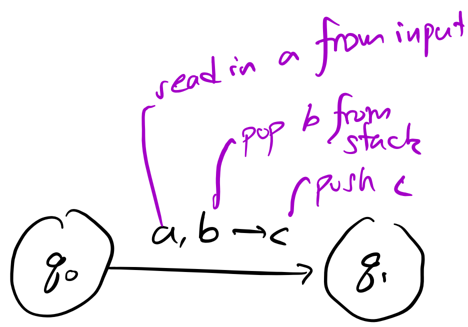
0. Review

Pushdown Automaton (PDA): automaton w/ a stack

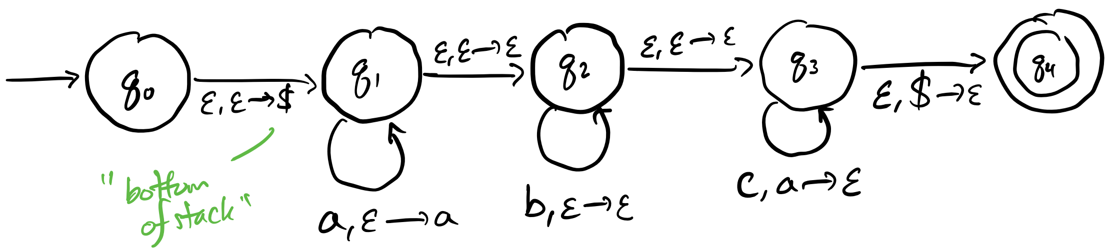


Each step:

- read in an input character
- pop a stack character
- move states
- push a stack character



Accept if, after input is read, at least one live branch in ◎.



test:

aaabbccc

some branch:

- push \$
- loop in q_1 and push a, a, a
- move $q_1 \to q_2$, read in b's
- move $q_2 \to q_3$, pop a's/reading c's.
- pop \$ and move to q_4 at end of input.

other branches?

what about ε?

ε does accept ✓
($q_0 \to q_4$ w/ input on tape rejects)

aaabbcc ?

aaabbcccc ? ✗ (get to q_3 w/ \$ on top)
(move to q_4, popping \$)
(BUT c remains)

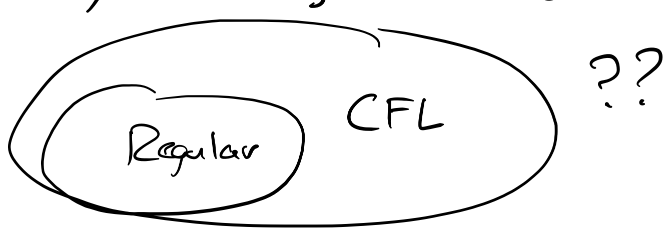
$A = \{a^i b^j c^i \mid i, j \geq 0\}$

Last time:

PDAs $\to$ CFGs.

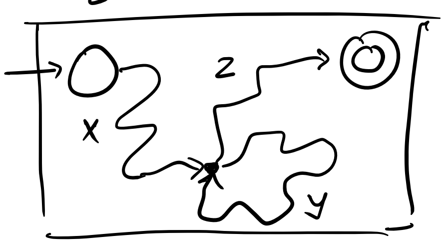
CFGs $\to$ PDAs.

Known: PDAs, as well as CFGs, both recognize the class of context-free languages.



Recall: the PL for regular languages.

D_L



D_L has $|Q|$ states — strings longer than $|Q|$ must have loops!

"ALL CFLs have some property — this property applies to sufficiently long strings."

(Proof. p. 125 - 127.)

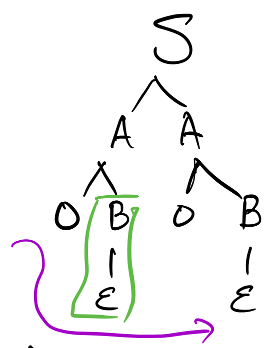
Example (infinite) grammar:

(R1) $S \longrightarrow AA$

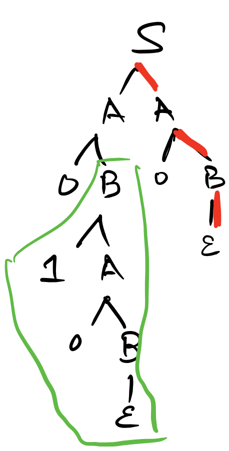
(R2) $A \longrightarrow 0B$

(R3) $B \longrightarrow 1A$

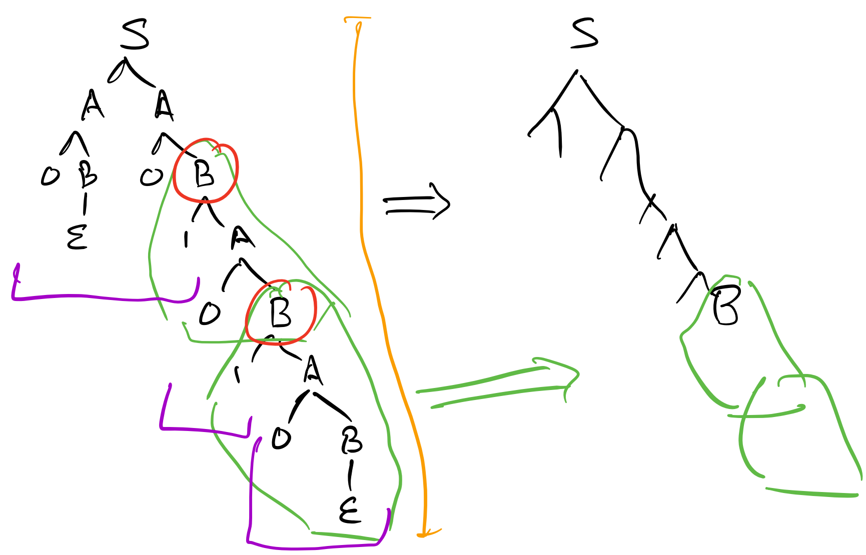
(R4) $B \longrightarrow \varepsilon$



read: 00



0100



What is "loopiness" in grammars?

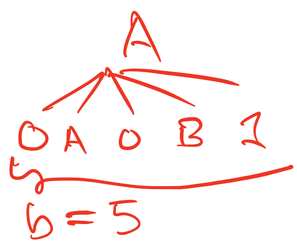
repeated variables.

How long is the longest path in a tree w/ no loops?

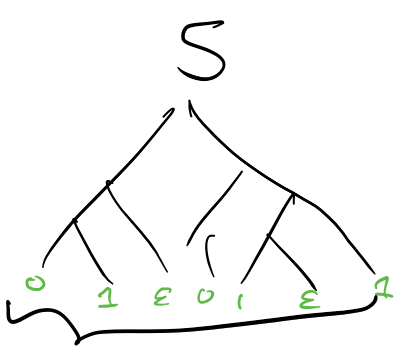
If I have $|V|$ variables,

a path of length $|V|+1$ edges must hit some variable twice.

- Say that b is the "branching factor" of my grammar: the greatest number of symbols produced by any rule.



Say I have a parse tree for a long string s, generated by some grammar G with $|V|$ variables, branching factor b.



length of $s \leq$ # of leaves.

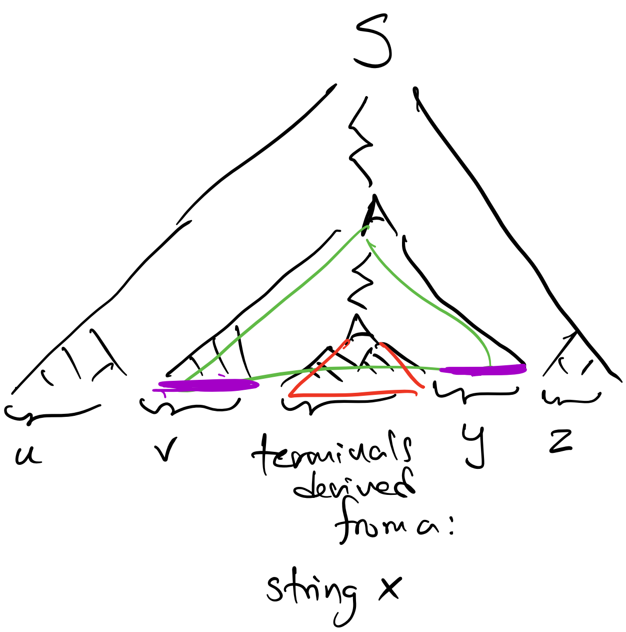
Say that $|s| \geq b^{|V|+1}$.

level 0: 1 node
level 1: $\leq b$ nodes
level 2: $\leq b^2$ nodes.
⋮
level $|V|+1$ $\leq b^{|V|+1}$ nodes.

---

conclusion: if s is long enough, I have some path of length $|V|+1 \longrightarrow$ I have a repeated variable.

Say A is our variable repeated on some path, and s is the string derived by our parse tree.

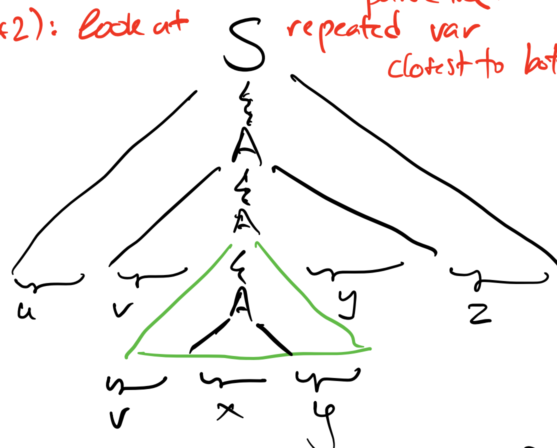


$s = uvxyz \in L(G)$

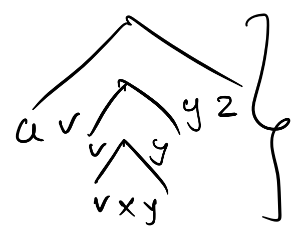
we can make other strings in $L(G)$ by copy-pasting!

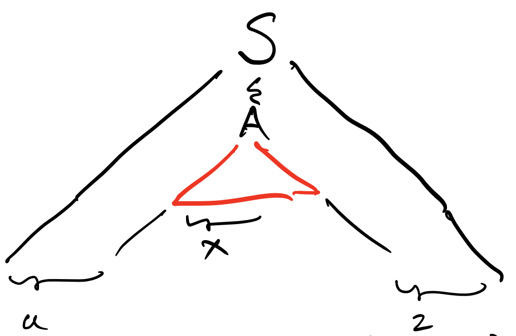
(*1): specify minimum height parse tree.

(*2): look at repeated var closest to bottom.



$uvvxyyz \in L(G)$.





$uxz \in L(G)$

$uxz = uv^0xy^0z$

Conclusion: If we have a CFG G with $|V|$ variables and branching factor b, then any string s sufficiently long can be divided into

$s = uvxyz$ such that

$uv^ixy^iz \in L(G)$ for all $i \geq 0$.

<u>Theorem (Pumping Lemma for CFLs)</u>. For any context-free language L, there is some number p such that all $s \in L$ with $|s| \geq p$, s can be divided into five parts $s = uvxyz$ such that

(1) $uv^ixy^iz \in L$ for all $i \geq 0$,

(*1) (2) $|vy| > 0$

(*2) (3) $|vxy| \leq p$.

Break: back at 2:14

<u>Goal</u>: Show $B = \{a^nb^nc^n \mid n \geq 0\}$ is not context-free.

1) Assume for contradiction B is context-free

∴ B satisfies the CFPL.

∴ There exists a number p such that $s \in B$ with $|s| \geq p$ can be divided into five parts

$s = uvxyz$ satisfying

(1) $uv^ixy^iz \in B$ for $i \geq 0$,

(2) $|vy| > 0$

(3) $|vxy| \leq p$.

2) Choose a string for contradiction.

$s = a^pb^pc^p$, $s \in B$, $|s| \geq p$.

// heuristic for choosing cont. strings: simple, but capturing "essence" of the language!

We'll show $a^pb^pc^p$ can't be split into $uvxyz$ satisfying (1)-(3) by carefully considering all cases.

$$\overbrace{aaa\cdots aaa}^{p\text{ times}}\,bb\cdots bbbcc\cdots ccc$$

v $\quad$ y

Case (1): suppose v and y each contain only one type of symbol.

This means when I repeat v and y (for instance, to make the string $uvvxyyz = uv^2xy^2z$) I'll increase the number of at least one character by (2), but will also leave another character's count unchanged.

The result is substrings of uneven length. $uv^2xy^2z \notin B$, contradiction.

Case (2): suppose at least one of v or y contains two types of symbol.

$$aa \cdots \underbrace{aa\,bb}_{v} \cdots bbcc \cdots cc$$

Now $uv^2xy^2z = uvvxyyz$ has symbols out of order and is not in B!

Conclusion: any split of $s = a^p b^p c^p$ fails one of conditions (1)-(3), so B fails the CFPL and B is not context-free. □

another argument that we can't pump $s = a^p b^p c^p$:

By (3), $|vxy| \leq p$

so: vxy contains at most 2 types of character.

By (2), $|vy| > 0$

so uvvxyyz has more of some character type than another.

Dividing into cases - how?

aa ⋯ aa bb ⋯ bb cc ⋯ cc

cases for v: all a's
a's and b's
all b's
b's and c's
all c's

~~a's, b's, and c's.~~
~~a's and c's~~

$\boxed{|vxy| \le p}$

$|vy| > 0$

case 1: v, y both either all a's all b's all c's.

Let $D = \{ww \mid w \in \{0,1\}^*\}$

Goal: Show D not context-free.

1) Assume D context-free.
∴ D satisfies CFPL
∴ there exists p such that for all $s \in D$, $|s| \ge p$,
$s = uvxyz$ for substrings satisfying
(1) $uv^ixy^iz \in D$ for all $i \ge 0$
(2) $|vy| > 0$,
(3) $|vxy| \le p$.

2) Choose a contradiction string.

Try $s = 0^p 1 0^p 1$. $s \in D$, $|s| \ge p$.

$$\underbrace{000\cdots000}_{u}\ \underbrace{0}_{v}\ \underbrace{1}_{x}\ \underbrace{0}_{y}\ \underbrace{0\cdots000\,1}_{z}$$

Try $s = 0^p 1^p 0^p 1^p$.

$$\underbrace{0}_{v}\ \underbrace{1^p}_{x}\ \underbrace{0}_{y} \qquad |vxy| = p+2 \quad \times$$

Case 1: vxy is a substring of the first half $(0^p 1^p)$.

$$00\cdots\underbrace{0011\cdots}_{}110\cdots01\cdots1$$

vxy in here somewhere.

Now: pumping v and y to make $uv^0xy^0z = uxz$.

In uxz, the first 0^k1^j substring has between p and $2p-1$ characters because we've removed vy.

$|vy| > 0 \quad |vxy| \le p$.

result:

$$\underbrace{0000}_{i \text{ times}}\underbrace{11\cdots1}_{j \text{ times}}\ 0^p 1^p$$

midpoint is now in the second string of 0's.

This no longer has the form ww. ✗ contradiction

Case 2: vxy is a substring of the second half - similar.

Case 3: vxy straddles the midpoint of the string.

$$0^p\, 11\cdots \underbrace{11\ 00}_{vxy}\cdots 00\, 1^p$$

If I pump down again, I get

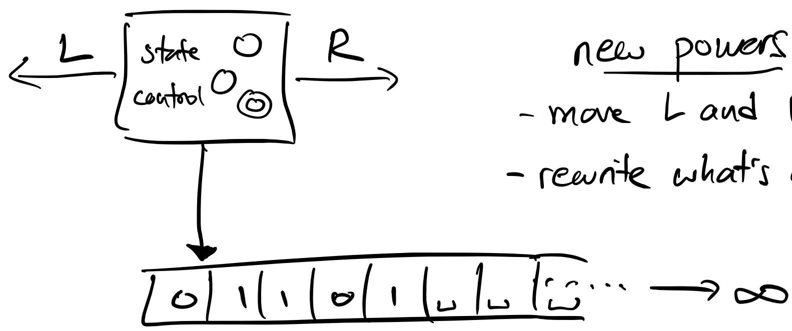
$$uv^0xy^0z = uxz = 0^p 1^i 0^j 1^p$$

one of my middle substrings is shorter, so I no longer have the form ww. (by $|vy| > 0$)

Conclusion: s fails one or more conditions any way it is divided, so D fails the CFPL and is not context-free.

Break: back at 3:03

Turing Machine:



input

new powers:
- move L and R on tape
- rewrite what's on the tape.

Effectively — this makes our tape into random access memory.

Every step, our TM does the following:

(1) read in input symbol off the current tape square.

(2) move to a new internal state

* (3) write something new in our current square

* (4) move one space L or R

Def. A Turing Machine is a 7-tuple

$$(Q, \Sigma, \Gamma, \delta, q_0, \underline{q_{accept}, q_{reject}})$$

Q: state set

Σ: input alphabet

Γ: tape alphabet (contains Σ and "␣", blank)

q_0: start state

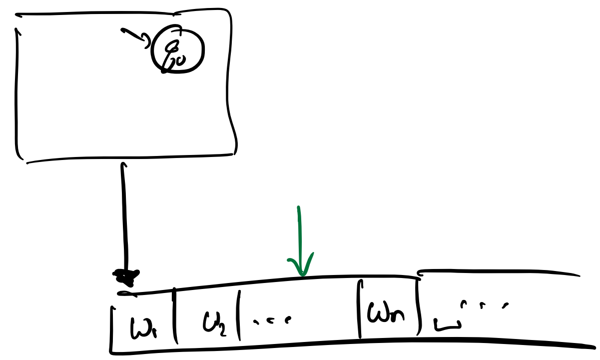
q_{accept}, q_{reject}: (no longer end once we read in all input.)

q_{acc}, q_{rej} are special states — enter them, halt immediately.

$$\delta : \underbrace{Q \times \Gamma}_{\text{current state, tape symbol}} \longrightarrow \underbrace{Q \times \Gamma \times \{L, R\}}_{\text{new state, symbol to write, direction to move 1 space.}}$$

TM computation:

– Start in q_0 with input string $w = w_1 \cdots w_n$, $w_i \in \Sigma$, at the left end of the tape. Go until we enter q_{accept} or q_{reject}.



– Summary of the whole machine: a configuration.

$$q_0 w_1 w_2 \cdots w_n.$$

more generally, a configuration can be written

uqv, where u is the tape to the left of the ↓ (tape head),

q is the current state,

v is the tape to the right of the head, starting with our current symbol.

$$w_1 w_2 \cdots w_i q_j w_{i+1} \cdots w_n$$

Def (TM acceptance) A TM M accepts a string w if there exists a sequence of configurations

$$C_1, C_2, \cdots, C_k \text{ such that}$$

(1) $C_1 = q_0 w$ (the start configuration)

(2) C_k is an accept configuration

$$C = (u q_{accept} v, \; u, v \in \Gamma^*)$$

(3) C_i yields C_{i+1} for all $i < k$,

where $C_i = u_1 u_2 \cdots u_m q_i v_1 \cdots v_n$,

yields $C_{i+1} = u_1 u_2 \cdots u_{m-1} q_j u_m c v_2 \cdots v_n$,

if $\delta(q_i, v_i) = (q_j, c, L)$.

(same for R)

TM state diagrams!

$$A = \{ 0^{2^n} \mid n \geq 0 \}$$

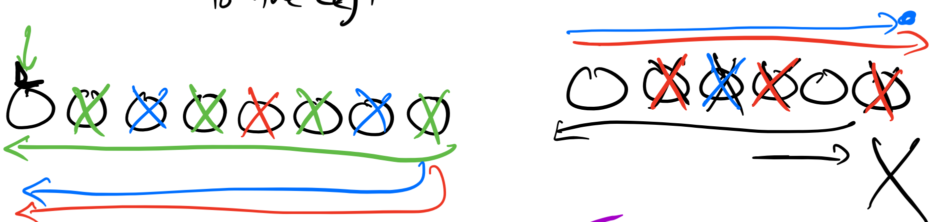
0
00
0000
00000000
⋮

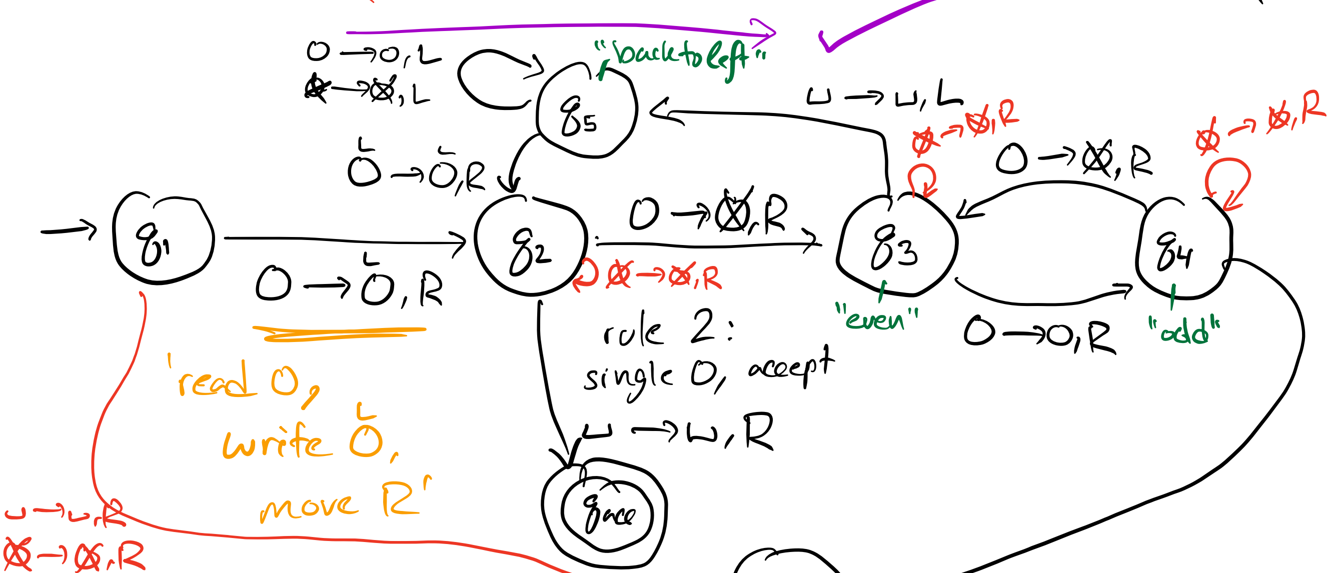
We'll build a TM M_1 to recognize A.

M_1 = "On input w:

1. Read input left to right; cross off every other zero.
2. If I saw one single 0, accept.
3. If we saw an odd number of uncrossed 0's, reject.
4. Go back to the left and start from step 1."

Example:





$\Gamma = \{ 0, \cancel{0}, \check{0}, \sqcup \}$

replace $\check{0}$ with $\sqcup$, or say "go to reject" from all states on δ.