18.404/6.840 Lecture 5

Last time:
- Context free grammars (CFGs)
- Context free languages (CFLs)
- Pushdown automata (PDA)
- Converting CFGs to PDAs

Today: (Sipser §2.3, §3.1)
- Proving languages not Context Free
- Turing machines
- T-recognizable and T-decidable languages

Equivalence of CFGs and PDAs

Recall Theorem: A is a CFL iff some PDA recognizes A

$\longrightarrow$ Done.

$\longleftarrow$ Need to know the fact, not the proof

Corollaries:

1) Every regular language is a CFL.
2) If A is a CFL and B is regular then $A \cap B$ is a CFL.

Proof sketch of (2):

While reading the input, the finite control of the PDA for A simulates the DFA for B.

Note 1: If A and B are CFLs then $A \cap B$ may not be a CFL (will show today).
Therefore the class of CFLs is not closed under $\cap$.

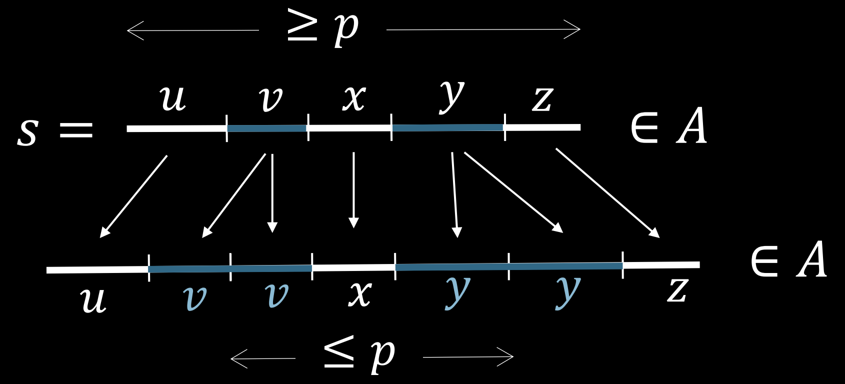
Note 2: The class of CFLs is closed under $\cup, \circ, *$ (see Pset 2).

Proving languages not Context Free

Let $B = \{0^k 1^k 2^k \mid k \geq 0\}$. We will show that B isn't a CFL.

Pumping Lemma for CFLs: For every CFL A, there is a p such that if $s \in A$ and $|s| \geq p$ then $s = uvxyz$ where

1) $uv^i xy^i z \in A$ for all $i \geq 0$ x can be epsilon; y can be epsilon, but both can't be epsilon
2) $vy \neq \varepsilon$
3) $|vxy| \leq p$



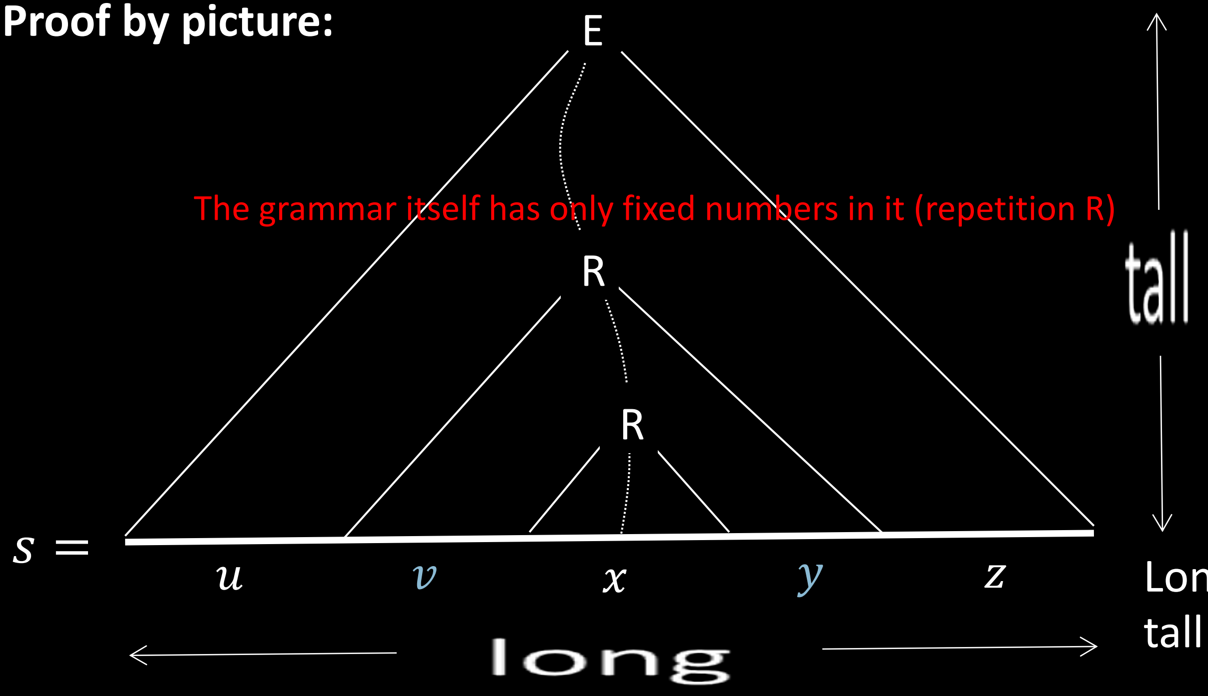
Informally: All long strings in A are pumpable and stay in A.

Pumping Lemma – Proof

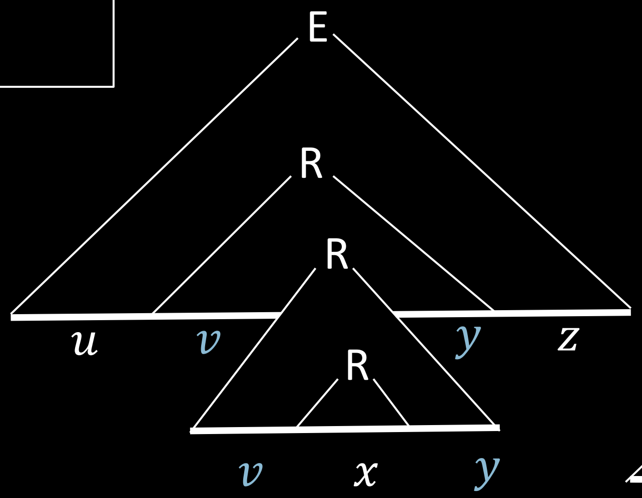
Pumping Lemma for CFLs: For every CFL A, there is a p such that if $s \in A$ and $|s| \geq p$ then $s = uvxyz$ where

1) $uv^ixy^iz \in A$ for all $i \geq 0$
2) $vy \neq \varepsilon$
3) $|vxy| \leq p$

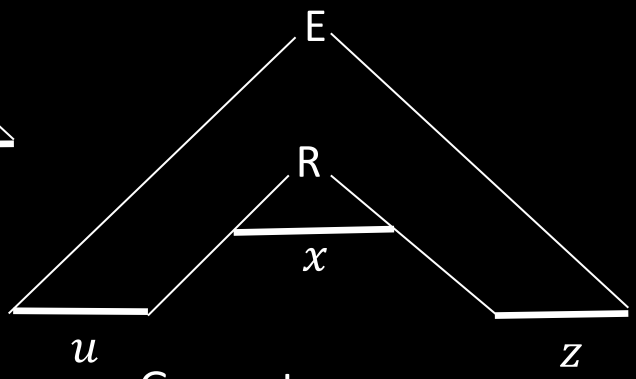
Proof by picture:



Long $s \rightarrow$ tall parse tree



Generates $uvvxyyz$
$= uv^2xy^2z$



Generates uxz
$= uv^0xy^0z$

"cutting and pasting" argument

Pumping Lemma – Proof details

Read the book to get a better understanding

For $s \in A$ where $|s| \geq p$, we have $s = uvxyz$ where:

1) $uv^i xy^i z \in A$ for all $i \geq 0$ …cutting and pasting
2) $vy \neq \varepsilon$ …start with the smallest parse tree for s
3) $|vxy| \leq p$ …pick the lowest repetition of a variable

Cannot be an inefficient parse tree which can be shorted and still generate s
Do not have R -> R

The |vxy| won't be very long
Because that would force another repetition to occur

Let b = the length of the longest right hand side of a rule (E → E+T)
= the max branching of the parse tree

E
E + T

Let h = the height of the parse tree for s.

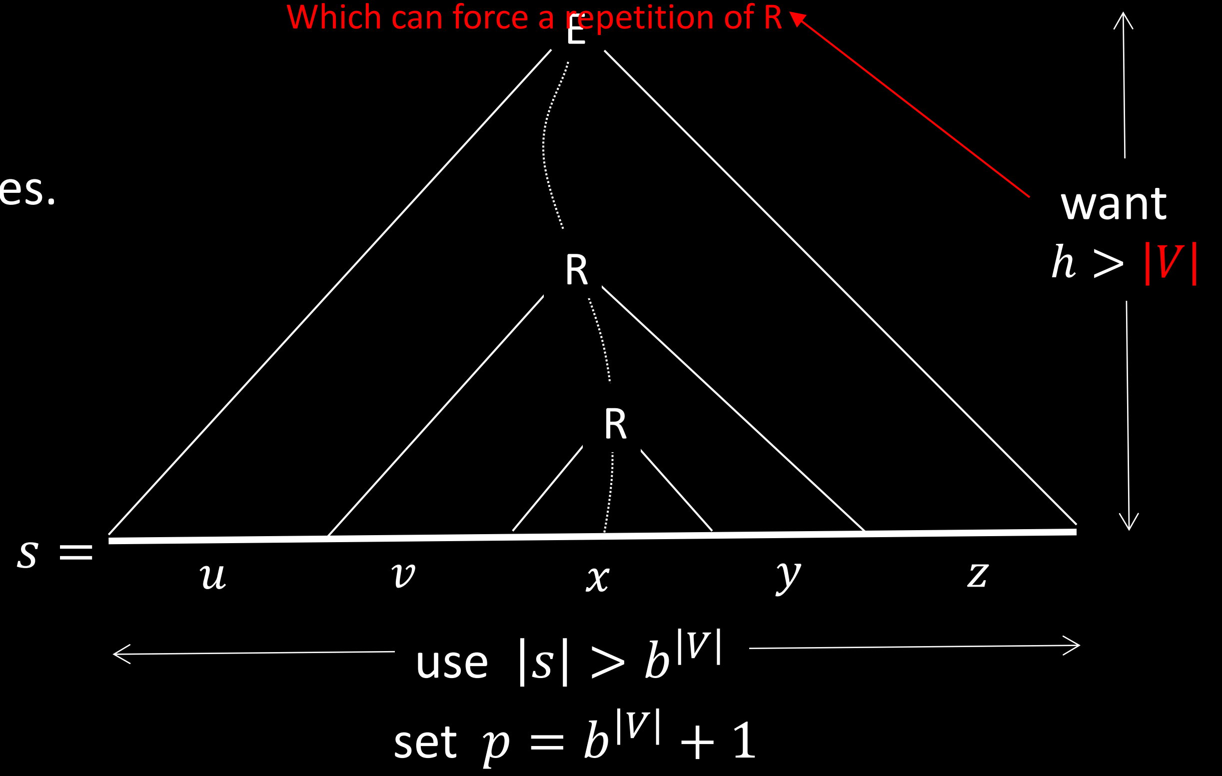
A tree of height h and max branching b has at most b^h leaves.
So $|s| \leq b^h$.

Let $p = b^{|V|} + 1$ where $|V|$ = # variables in the grammar.

So if $|s| \geq p > b^{|V|}$ then $|s| > b^{|V|}$ and so $h > |V|$.

Thus at least $|V| + 1$ variables occur in the longest path.
So some variable R must repeat on a path.

We want height bigger than the number of variables |V|
Which can force a repetition of R



Example 1 of Proving Non-CF

Pumping Lemma for CFLs: For every CFL A, there is a p such that if $s \in A$ and $|s| \geq p$ then $s = uvxyz$ where

1) $uv^ixy^iz \in A$ for all $i \geq 0$
2) $vy \neq \varepsilon$
3) $|vxy| \leq p$ Cannot be the hole string (a piece of it)

Key: how to divide ?

From Book: Condition2 says that either v or y is nonempty:

1. If both v and y contain only one type of alphabet symbol, let's say both a and b or both b and c. in this case, the string uv^2xy^2z cannot contain equal number of a's b's and c's
2. If either v or y contains more than one type of symbol, then the uv^2xy^2z may contain equal numbers of the three alphabets symbol but not in the correct order

Let $B = \{0^k1^k2^k \mid k \geq 0\}$

Show: B is not a CFL

Proof by Contradiction:

Assume (to get a contradiction) that B is a CFL .
The CFL pumping lemma gives p as above. Let $s = 0^p1^p2^p \in B$.
Pumping lemma says that can divide $s = uvxyz$ satisfying the 3 conditions.
Condition 3 $(|vxy| \leq p)$ implies that vxy cannot contain both 0s and 2s.
So uv^2xy^2z has unequal numbers of 0s, 1s, and 2s.
Thus $uv^2xy^2z \notin B$, violating Condition 1. Contradiction!
Therefore our assumption (B is a CFL) is false. We conclude that B is not a CFL .

$$s = 00\cdots0011\cdots1122\cdots22$$

$u \mid v \mid x \mid y \mid z$

$\leftarrow \leq p \rightarrow$

Check-in 5.1

Check-in 5.1

Let $A_1 = \{0^k 1^k 2^l \mid k, l \geq 0\}$ (equal #s of 0s and 1s)

Let $A_2 = \{0^l 1^k 2^k \mid k, l \geq 0\}$ (equal #s of 1s and 2s)

Observe that PDAs can recognize A_1 and A_2. What can we now conclude?

a) The class of CFLs is not closed under intersection. The intersection is just B = {$0^k1^k2^k$)

b) The Pumping Lemma shows that $A_1 \cup A_2$ is not a CFL .

Context-free language is closed under union

c) The class of CFLs is closed under complement.

Example 2 of Proving Non-CF

Pumping Lemma for CFLs: For every CFL A, there is a p such that if $s \in A$ and $|s| \geq p$ then $s = uvxyz$ where

1) $uv^ixy^iz \in A$ for all $i \geq 0$
2) $vy \neq \varepsilon$
3) $|vxy| \leq p$

Not a very clear description (many cases)

F is the two copies of same string

Let $F = \{ww \mid w \in \Sigma^*\}$. $\Sigma = \{0,1\}$.

Show: F is not a CFL.

Assume (for contradiction) that F is a CFL.
The CFL pumping lemma gives p as above. Need to choose $s \in F$. Which s?

Try $s_1 = 0^p10^p1 \in F$.

$$s_1 = 000\cdots001000\cdots001$$

$$u \quad v \quad x \quad y \quad z$$

$$\leftarrow \leq p \rightarrow$$

Try $s_2 = 0^p1^p0^p1^p \in F$.

$$s_2 = 0\cdots01\cdots10\cdots01\cdots1$$

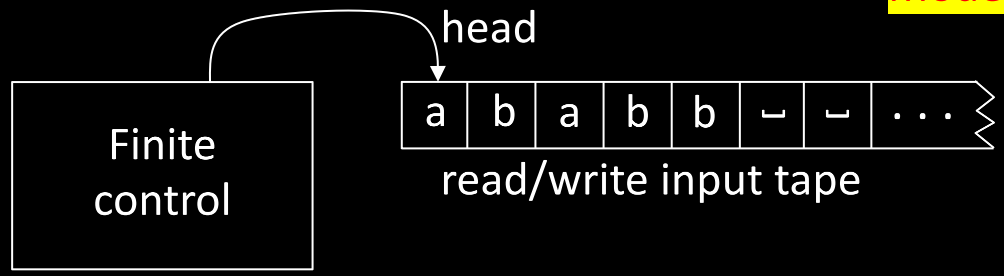
$$u \quad v \quad x \quad y \quad z$$

$$\leftarrow \leq p \rightarrow$$

Show s_2 cannot be pumped $s_2 = uvxyz$ satisfying the 3 conditions.
Condition 3 implies that vxy does not overlap two runs of 0s or two runs of 1s.
Therefore, in uv^2xy^2z, two runs of 0s or two runs of 1s have unequal length.
So $uv^2xy^2z \notin F$ violating Condition 1. Contradiction! Thus F is not a CFL.

Turing Machines (TMs)

Model of general-purpose computer

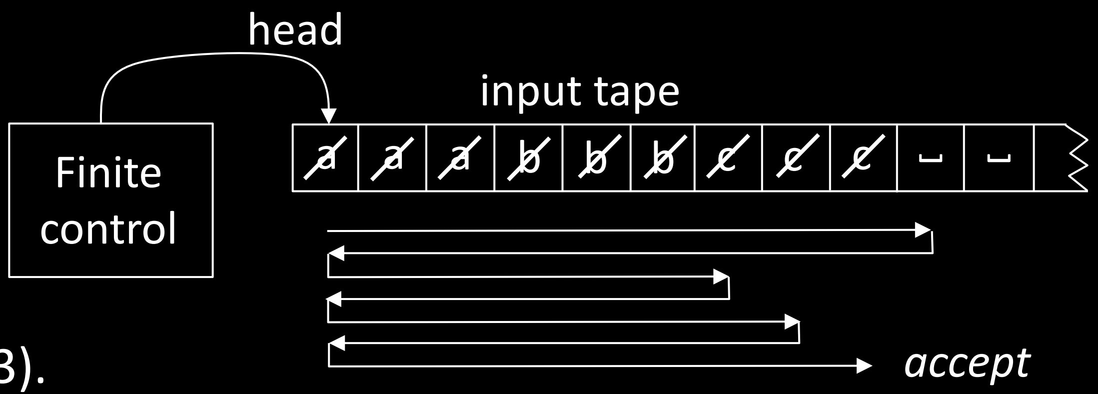


1) Head can read and write
2) Head is two way (can move left or right)
3) Tape is infinite (to the right)
4) Infinitely many blanks "␣" follow input
5) Can accept or reject any time (not only at end of input)

TM – example

TM recognizing $B = \{\mathrm{a}^k\mathrm{b}^k\mathrm{c}^k \mid k \geq 0\}$

1) Scan right until $\sqcup$ while checking if input is in $\mathrm{a}^*\mathrm{b}^*\mathrm{c}^*$, *reject* if not.
2) Return head to left end.
3) Scan right, crossing off single a, b, and c.
4) If the last one of each symbol, *accept*.
5) If the last one of some symbol but not others, *reject*.
6) If all symbols remain, return to left end and repeat from (3).



Check-in 5.2

How do we get the effect of "crossing off" with a Turing machine?

a) We add that feature to the model.

b) We use a tape alphabet $\Gamma = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \cancel{\mathrm{a}}, \cancel{\mathrm{b}}, \cancel{\mathrm{c}}, \sqcup\}$.

c) All Turing machines come with an eraser.

TM – Formal Definition

Defn: A <u>Turing Machine</u> (TM) is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$

Σ input alphabet

Γ tape alphabet $(\Sigma \subseteq \Gamma)$

δ: $\mathrm{Q} \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$ (L = Left, R = Right)

$\delta(q, \mathrm{a}) = (r, \mathrm{b}, \mathrm{R})$

Example: if we are in state q and the head is looking at an a currently on the tape, then we can move to state r. then change a to b and we move the head right 1.

On input w a TM M may halt (enter q_{acc} or q_{rej}) or M may run forever ("loop").

So M has 3 possible outcomes for each input w:

1. *<u>Accept</u>* w (enter q_{acc})
2. *<u>Reject</u>* w by halting (enter q_{rej})
3. *<u>Reject</u>* w by looping (running forever)

Check-in 5.3

This Turing machine model is deterministic.
How would we change it to be nondeterministic?

a) Add a second transition function.

b) Change δ to be δ: $\mathrm{Q} \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\})$

c) Change the tape alphabet Γ to be infinite.

Check-in 5.3

TM Recognizers and Deciders

The language of the machine is the collection of strings that the machine accepts

Let M be a TM. Then $L(M) = \{w \mid M \text{ accepts } w\}$.

Say that M recognizes A if $A = L(M)$.

Defn: A is Turing-recognizable if $A = L(M)$ for some TM M.

If a Turing machine may sometimes reject by looping, then it's only recognizing its language

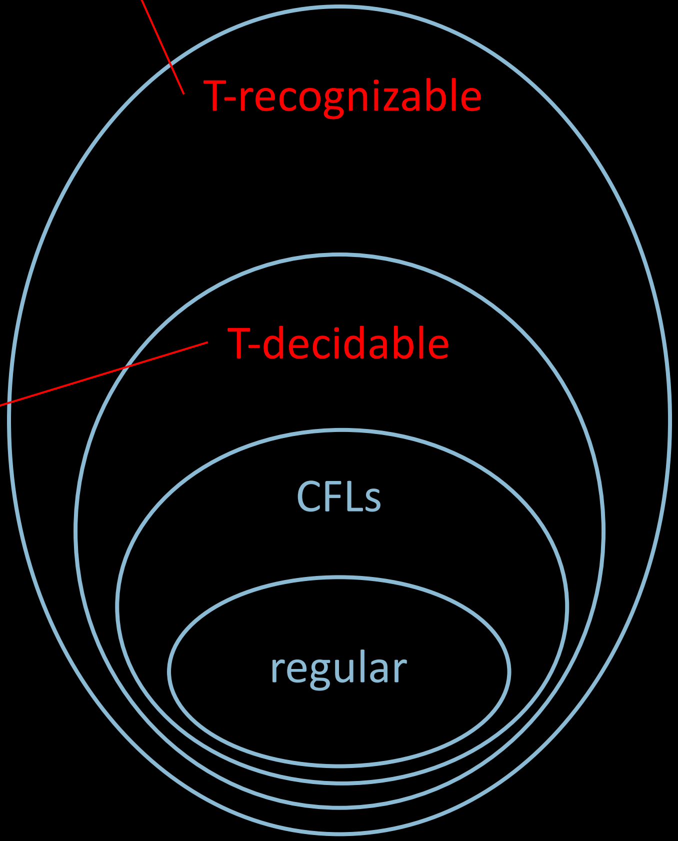
Defn: TM M is a decider if M halts on all inputs.

Machine halts -> made a decision of accepting or rejecting at that point which it had halted

Say that M decides A if $A = L(M)$ and M is a decider.

Defn: A is Turing-decidable if $A = L(M)$ for some TM decider M.

If the Turing machine is always halting, which means always rejecting by explicitly coming to a reject state and halting, then we says it deciding the language



Quick review of today

1. Proved the CFL Pumping Lemma as a tool for showing that languages are not context free.
2. Defined Turing machines (TMs).
3. Defined TM deciders (halt on all inputs).
4. T-recognizable and T-decidable languages.