18.404/6.840 Lecture 5

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

Today:
- Proving languages not Context Free
- Turing machines
- T-recognizable and T-decidable languages

Posted:
- Solutions to PSet 1
- PSet 2

Equivalence of CFGs and PDAs

Recall Theorem: is a CFL iff some PDA recognizes

Done. $\rightarrow$

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

Corollaries:

1) Every regular language is a CFL.
2) If is a CFL and is regular then is a CFL.

Proof sketch of (2):

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

Note 1: If and are CFLs then may not be a CFL (will show today).

Therefore the class of CFLs is not closed under .

Note 2: The class of CFLs is closed under (see Pset 2).

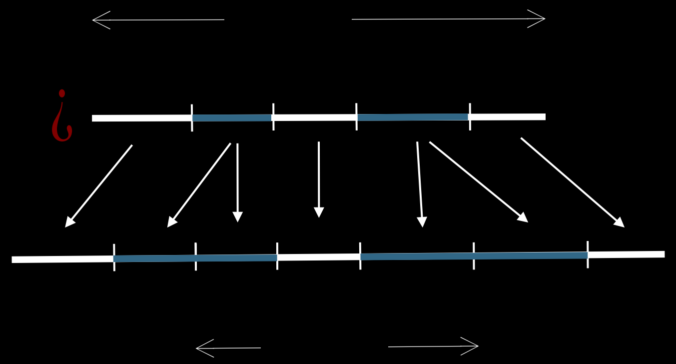
Proving languages not Context Free

Let . We will show that isn't a CFL.

Pumping Lemma for CFLs: For every CFL , there is a such that if and then where

1) for all
2)
3)

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

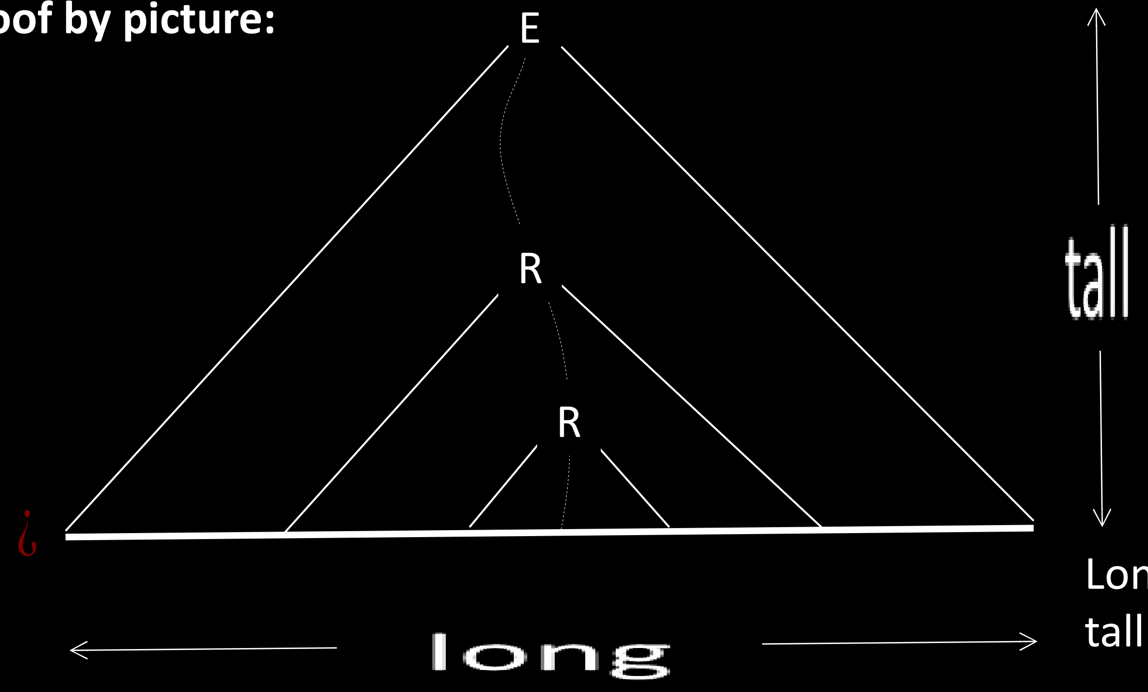


Pumping Lemma – Proof

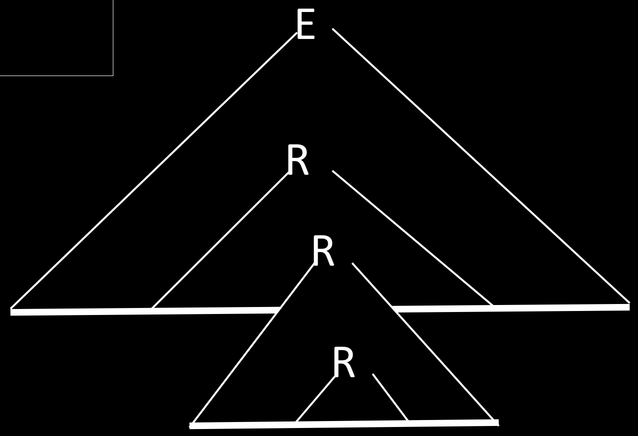
Pumping Lemma for CFLs: For every CFL , there is a
such that if and then where

1) for all
2)
3)

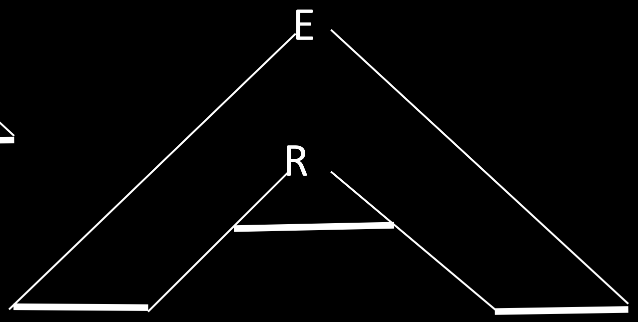
Proof by picture:



Long
tall parse tree



Generates



Generates

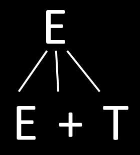
“cutting and pasting” argument

Pumping Lemma – Proof details

For where , we have where:

1) for all ...cutting and pas
2) ...start
3) ...pick the

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



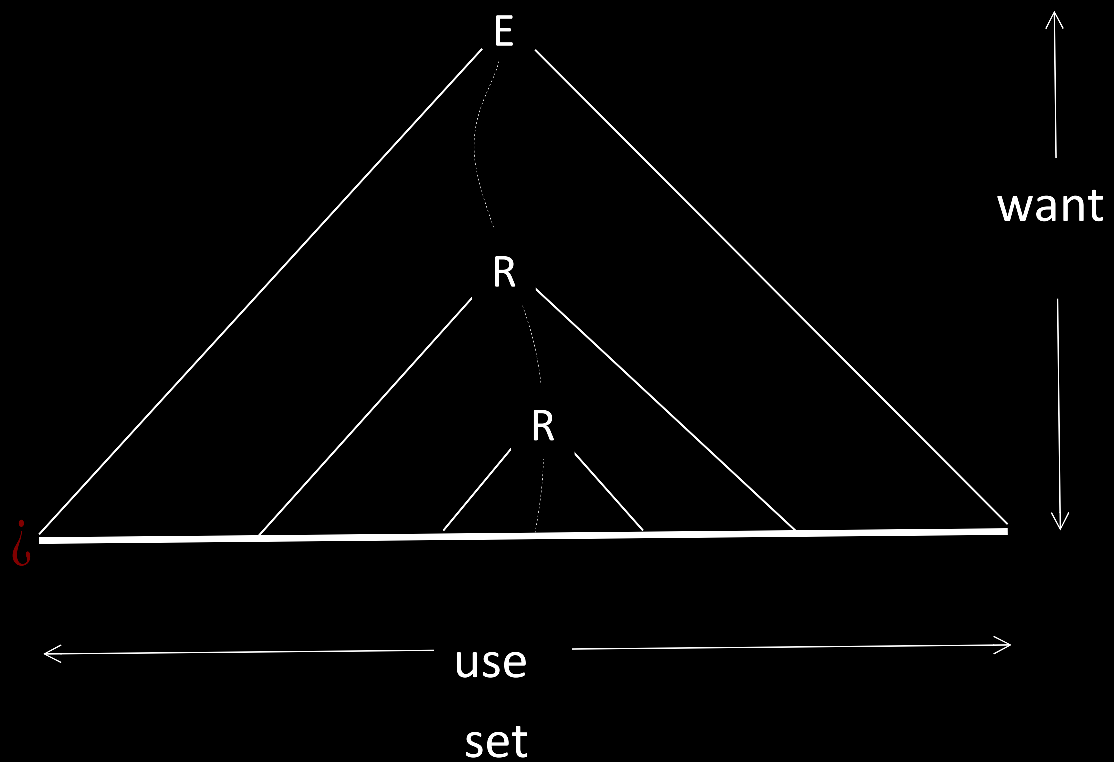
Let the height of the parse tree for .

A tree of height and max branching has at most leaves.
So .

Let where # variables in the grammar.

So if then and so .

Thus at least variables occur in the longest path.
So some variable must repeat on a path.



Example 1 of Proving Non-CF

Pumping Lemma for CFLs: For every CFL , there is a such that if and then where

1) for all
2)
3)

Let

Show: is not a CFL

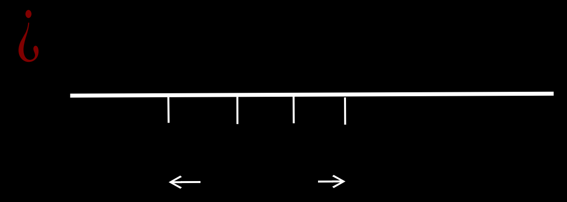
Check-in 5.1

Let (equal #s of 0s and 1s)

Let (equal #s of 1s and 2s)

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

a) The class of CFLs is not closed under intersection.
b) The Pumping Lemma shows that is not a CFL .
c) The class of CFLs is closed under complement.



Example 2 of Proving Non-CF

Pumping Lemma for CFLs: For every CFL , there is a
such that if and then where

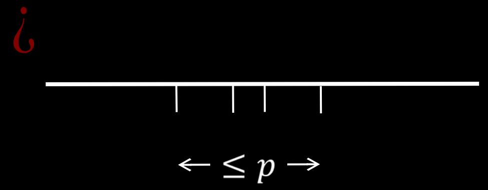
1) for all
2)
3)

Let .

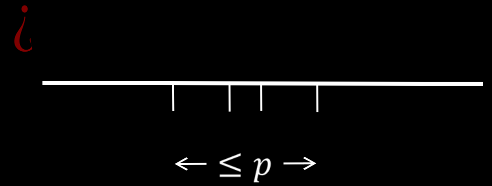
Show: is not a CFL.

Assume (for contradiction) that is a CFL.
The CFL pumping lemma gives as above. Need to choose . Which ?

Try . But can be pumpe

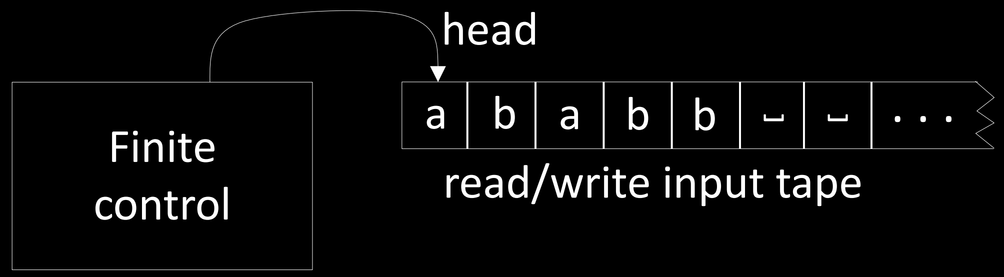


Try .
Show cannot be pumped satisfying the 3 conditions.
Condition 3 implies that does not overlap two runs of 0s or two runs of 1s.
Therefore, in , two runs of 0s or two runs of 1s have unequal length.
So violating Condition 1. Contradiction! Thus is not a CFL.




Coffee
Break

Turing Machines (TMs)

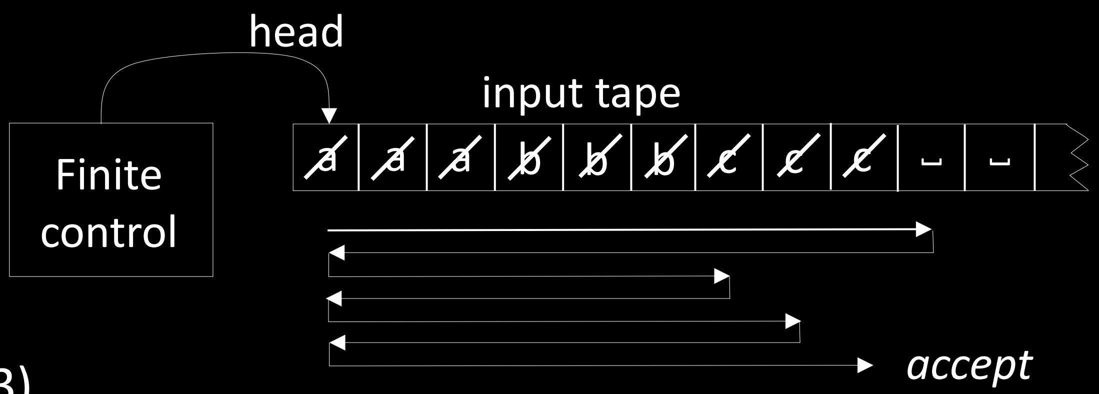


1) Head can read and write
2) Head is two way (can move left or right)
3) Tape is infinite (to the right)
4) Infinitely many blanks “␣“ follow input
5) Can accept or reject any time (not only at end of input)

TM – example

TM recognizing

1) Scan right until ␣ while checking if input is in , *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 , , , , , , ␣ }.

c) All Turing machines come with an eraser.

TM – Formal Definition

Defn: A Turing Machine (TM) is a 7-tuple

input alphabet

tape alphabet ()

{L, R} (L = Left, R = Right)

R

On input a TM may halt (enter or)
or may run forever ("loop").

So has 3 possible outcomes for each input :

1. *Accept* (enter)
2. *Reject* by halting (enter)
3. *Reject* 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 to be {L, R}

c) Change the tape alphabet to be infinite.

TM Recognizers and Deciders

Let be a TM. Then accepts .

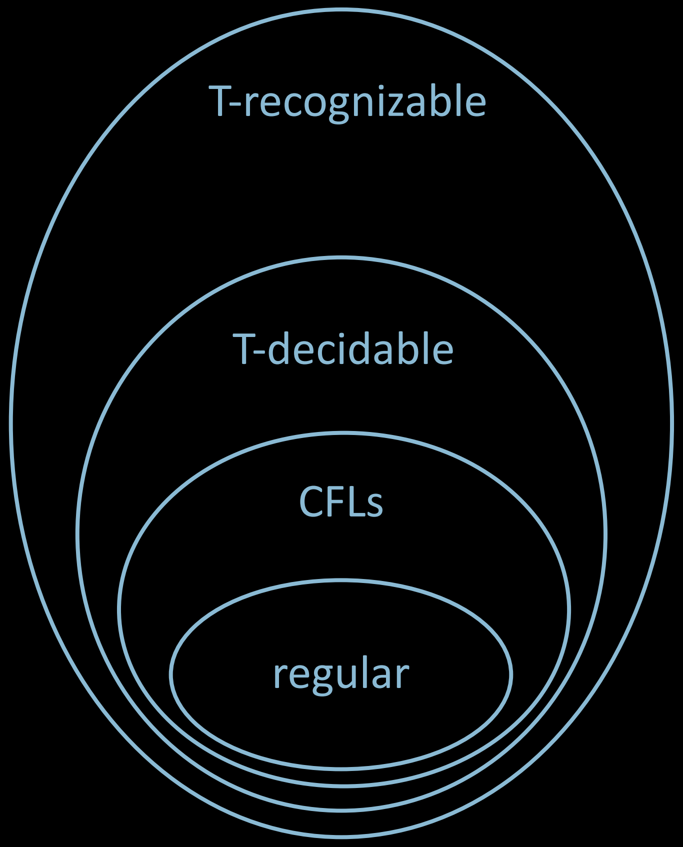
Say that recognizes if .

Defn: is Turing-recognizable if for some TM .

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

Say that decides   if   and is a decider.

Defn: is Turing-decidable if for some TM decider .



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.