Office hours 1-2 today 10/1/18
2-3 tommorrow
(later if asked)

More on context-free grammars

$L = \{ a^n b^m \mid n > m \}$

can think of this as equal # w/ extra a's on the left.

$S \rightarrow aT$

$T \rightarrow aTb \mid \lambda \mid aT$

or

$S \rightarrow ~~a \mid aT \mid aTb~~ \; a \mid aS \mid aSb$

$L = \{$ partial parenthesization,
i.e. # ('s $\geq$ #)'s at any point $\}$
(but we don't need to end with
('s = #)'s)

How about:

$S \to (S) \mid (S \mid SS \mid \lambda$?

---

okay- I thought about the case where the extra ('s are in the middle.

L = same, except I require at least one extra (.

~~How~~ $S \to (S) \mid (S \mid SS \mid ($?

1 2 3 4

~~S(~~ 5

() S

S() 6

$S \to SS \to (S \to ((($

Can we get ()(?

((()())(?

I don't think so.

How to get (()())(?

$S \to SS \to$ ~~SSS~~

This can't work- every S generates one extra (- this will generate 2 extra ((s already- not okay

$S \to (S)$ can't end with (.

$S \to ((A) \mid (A)(\mid (AA \mid A(A$

.... maybe this is too complicated.

$S \to (A \mid A(\mid A(A$

$A \to (A) \mid AA \mid \lambda \mid (A$ (matching ~~exactly~~ extra prens)

covered by last one

()(() might still be a problem

adding $S \to A(A$ fixes it?

Derivation trees

It really doesn't matter what order I do substitutions for variables. I would like a canonical way to either

a) choose the order

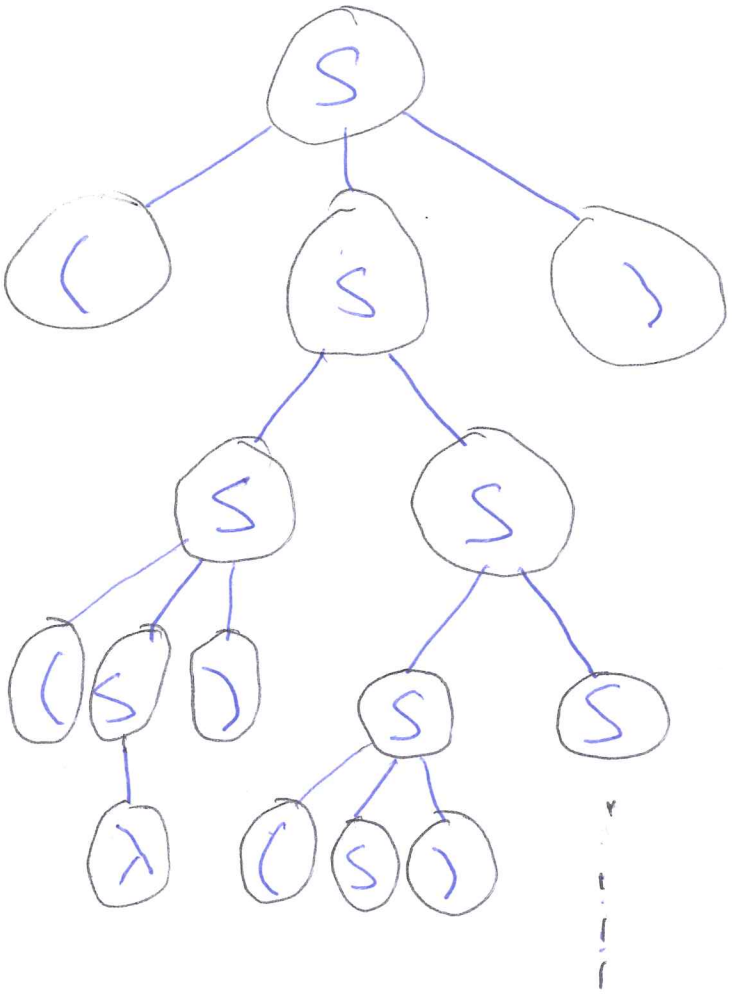
or b) represent my derivations w/o seeing the order.

Possibility for (a) - require that I always substitute for leftmost variable.

A derivation (sequence of substitutions to get a string) that follows this rule is a leftmost derivation.

Possibility for (b) - draw a tree

e.g.



A tree where

- every ~~var~~ internal node is a variable
- every leaf is a letter or λ
- the children of a node is what is substituted for that variable (this means they form a production for that variable)
- you read the final string as leafs from L to R.