Another example of a CFG

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

What's a grammar for this?

$S \rightarrow aSb \mid A \mid B$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

or

$S \rightarrow A \mid B$

$A \rightarrow aAb \mid aA \mid a$

$B \rightarrow aBb \mid Bb \mid b$

<u>Moral</u>: When you have 2 disjoint possibilities for your strings, you can branch to 2 variables.

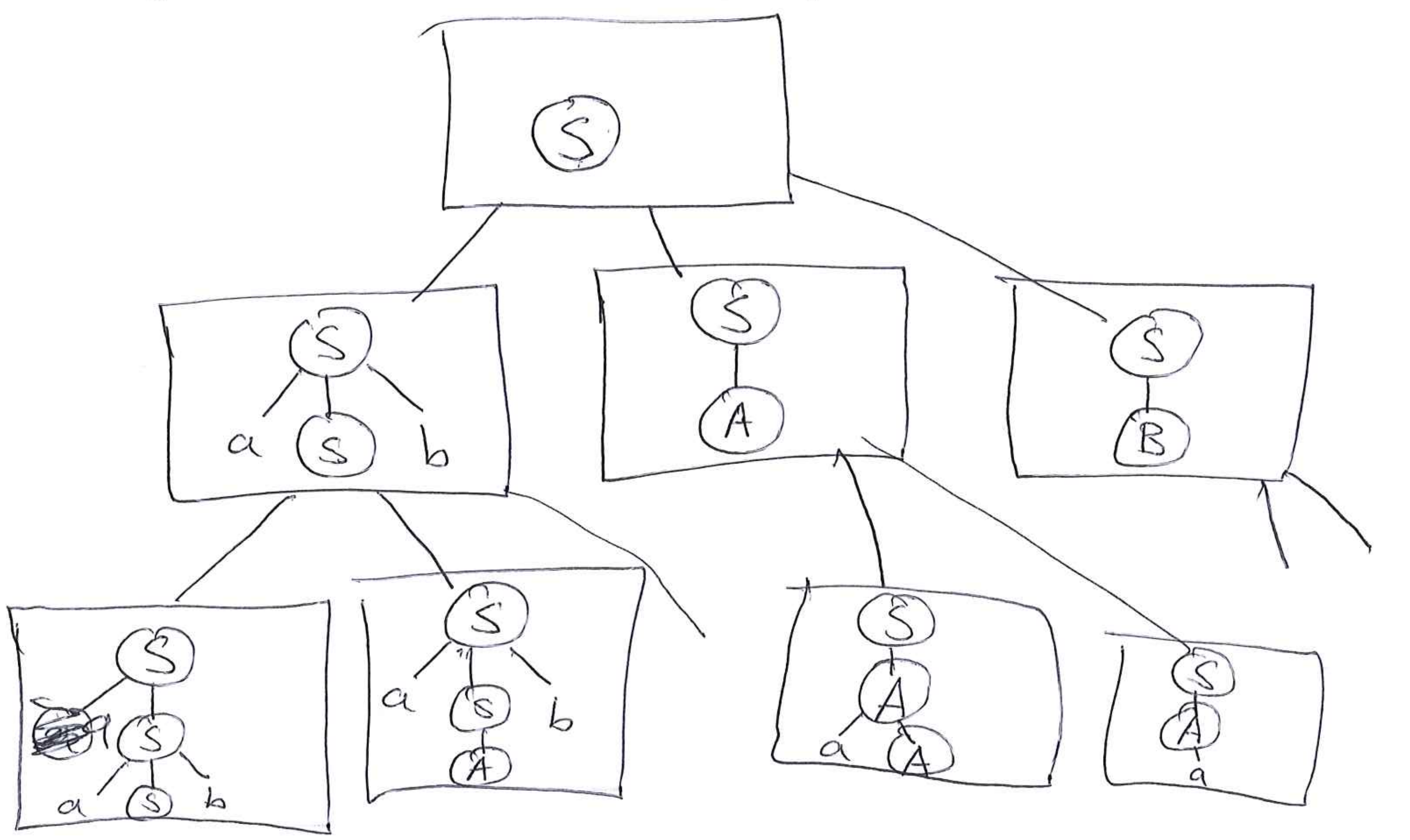
Parsing

i.e. finding a derivation tree for a string given a grammar.

The brute-force method (exhaustive parsing):

We have a tree of possible derivation trees:

e.g. $S \to aSb \mid A \mid B$, $A \to aA \mid a$, $B \to bB \mid b$



Nodes are possible partial derivation trees - one node is a child of another if we get it by expanding (i.e. doing a production rule with) the leftmost variable.

A leaf of this tree of deriv. trees is complete derivation tree for some string.

Exhaustive parsing is ~~depth~~ breadth-first search through this tree of trees. (stop when I match the string I am looking for).

Do <u>not</u> do depth-first search - this is an infinite tree and you might go down one infinite path.

If your string can be generated by the grammar you will eventually find it. by this breadth-first search.

Problem: How do we know when to stop if our string is not generated by the grammar?

Easy sol'n: Ensure that our grammar has no productions that look like

$A \to \lambda$ ← not having these means your string can't get shorter.

or

$A \to B$. ← not having these means you can't have a loop of productions getting nowhere.

The only productions that don't make your string longer are those of the form $A \to a$.

Note - every grammar can be transformed into ~~an equivalent one~~ another grammar generating the same language that does not have these kinds of productions.
(unless λ is in the language)

In this case, the largest possible number of steps to produce a string of length k is $2k-1$.

So - you know you are done when you have searched throw $2k-1$ levels of the tree (where your string has length k).

There are better methods - if we have time I might spend an hour on one ~~(for~~ (after converting the grammar to a special form).

Special kinds of grammars that have better parsing methods.

① Right-linear grammars $\longleftrightarrow$ ~~DFAs~~ NFAs $\longleftrightarrow$ DFAs.

② S-grammars: A grammar is an S-grammar if

a) Every prod looks like $A \to a\,\underbrace{\text{stuff}}_{}$ other

first thing is a letter

b) Given any variable A and any letter a, there is at most one prod. of the form

$$A \rightarrow a\ \underline{\text{stuff}}$$

that particular var

that particular letter

E.g. $S \rightarrow aSb \mid bSa \mid c$

is an S-grammar

but $S \rightarrow aSb \mid bSa \mid abS \mid b$

is not an S-grammar

With an S-grammar, parsing is deterministic - you can always know the production to do by looking at the target string.

☹ - Many context-free languages have no S-grammars.