How to build a Bottom-Up Parser from a CFG.

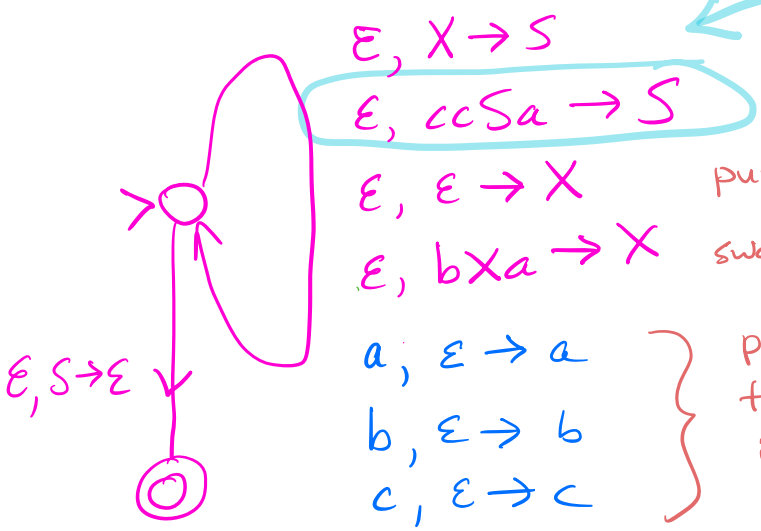
Given a CFG:

$S \to aScc \mid X$

$X \to aXb \mid \varepsilon$

Each rule of the grammar is "reversed" in the PDA, in a rule that consumes no input

Create a PDA that has 2 states.



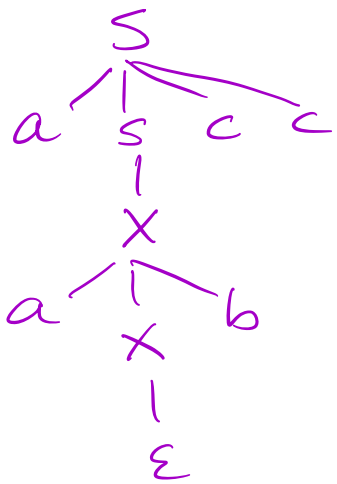
How it works (illustrated using an example)

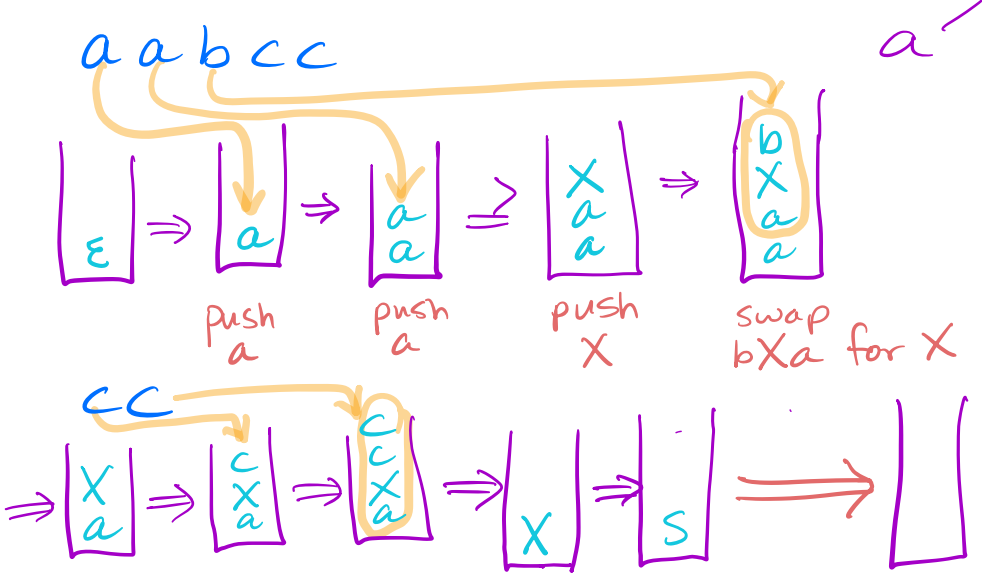
We know that the following is a derivation of aabcc

$S \Rightarrow aScc \Rightarrow aXcc \Rightarrow aaXbcc \Rightarrow aabcc$

So our PDA should accept aabcc, and should do so in a "bottom up" manner.

PDA: almost all the work is done on the stack, where we also occasionally push a symbol from the input:





transition to accept state, popping S.

ACCEPT