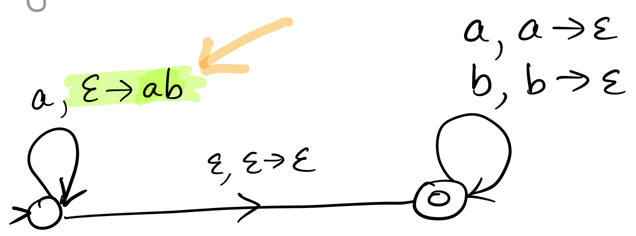
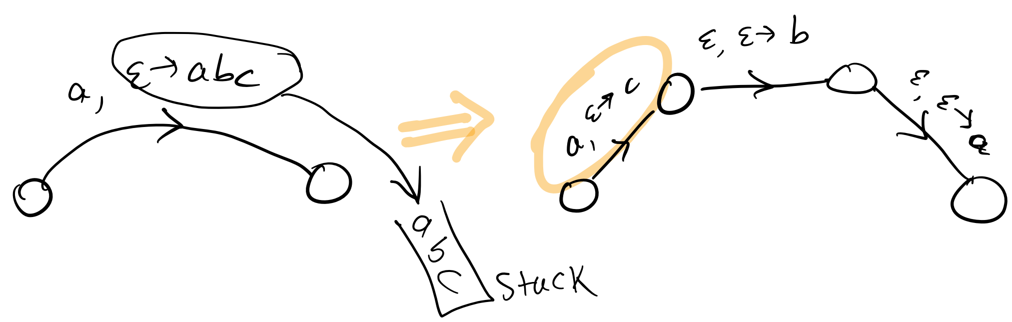
PDA's, continued.

A note about PDAs, and some warm-ups.

For nice, compact PDAS, we will allow ourselves the following notation:



What does ε→ab mean, and how do we get away with it?

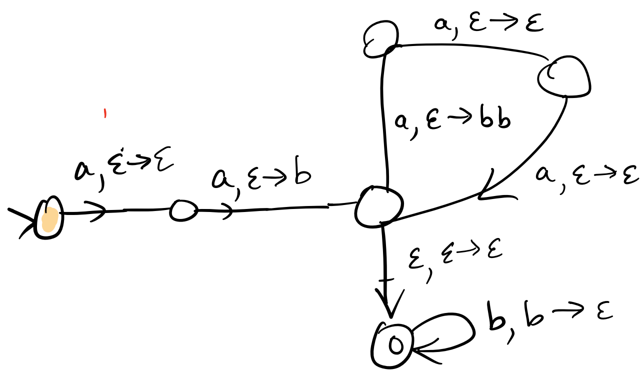


We "get away with it" because it does not change the underlying model of PDA: for any PDA with strings on the stack part of the transition, we can easily convert it to one that just uses single symbols for the stack transitions.

Note: the string slides onto the stack "back end first".

Warm up: a PDA for $\{a^n b^m \mid 2n = 3m+1\}$

Option 1: a natural PDA



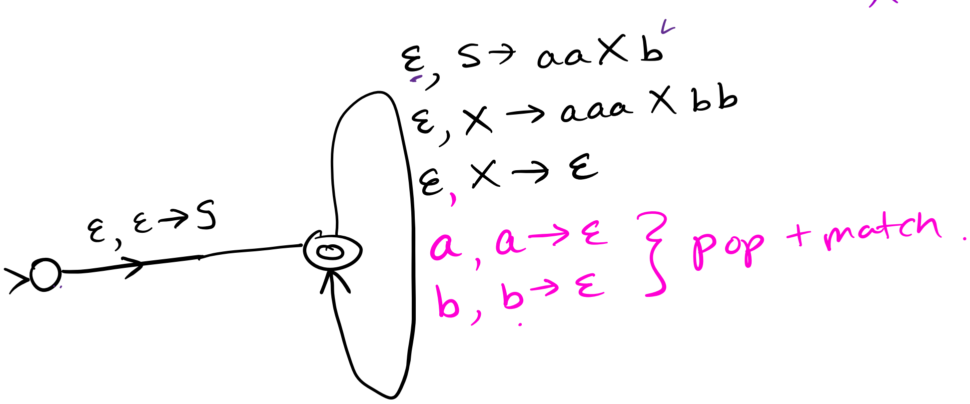
$S \to aaXb$

$X \to aaaXbb \mid \varepsilon$

Option 2: a top-down parser PDA

$S \to aaXb$

$X \to aaaXbb \mid \varepsilon$



Let's continue our proof that

$$\exists \text{ CFG for } L \overset{\longrightarrow}{\underset{\longleftarrow}{\Longleftrightarrow}} \exists \text{ a PDA for } L.$$

We showed that $\forall$ CFG, $\exists$ a PDA for the same language. That is a useful construction.

Claim: $\forall$ PDA M, $\exists$ a CFG G_M such that $L(M) = L(G_M)$.

Essence of the proof:

$\forall$ pairs of states $p, q \in Q_M$,

create a variable A_{pq}

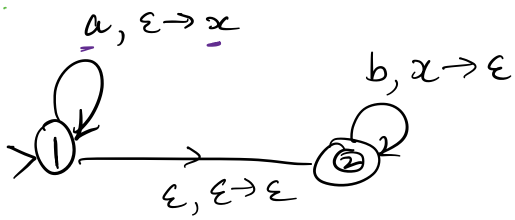
Develop rules, based on the transitions in M, that are designed so that

$$A_{pq} \Rightarrow \quad \Rightarrow \quad \Rightarrow w$$

if and only if w is a string that can "drive" M from state p to state q. with net-zero effect to stack.

Then the start symbol is ...

Eg.



$A_{12} \to \varepsilon$

$A_{12} \to a A_{12} b$

$A^n B^n = \{ a^n b^n \mid n \geq 0 \}$

That's the idea, but there are a lot of moving parts in the general case. $\square$

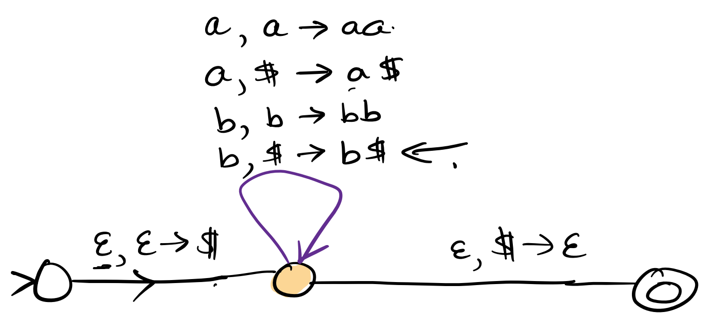
You will not be responsible for the <u>construction</u> in the PDA ⇒ CFG direction, but you might be given a language (that has a CFG) and asked to give a PDA for it — "grok + blurt".

PDA for $\{ w \in \{a,b\}^* \mid \#_a(w) = \#_b(w) \}$

input   stack   replacement on stack.

$b, \boxed{a \to \varepsilon}$

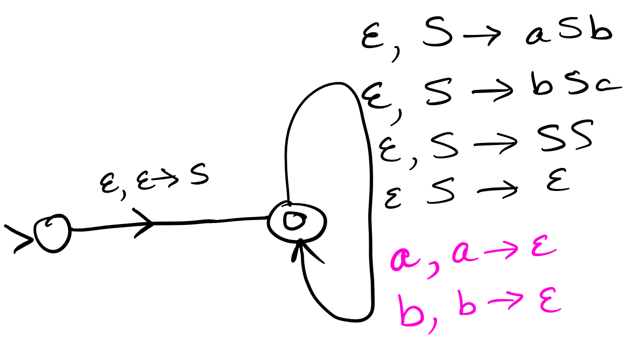
$a, b \to \varepsilon$



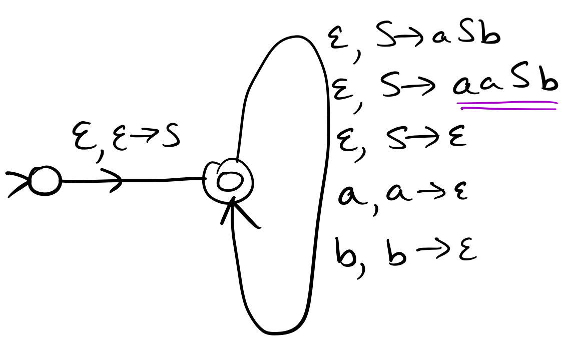
Grammar for $\{ w \in \{a,b\}^* \mid \#_a(w) = \#_b(w) \}$

$S \to aSb \mid bSa \mid SS \mid \varepsilon$.

Top-Down Parser for $\{ w \in \{a,b\}^* \mid \#_a(w) = \#_b(w) \}$

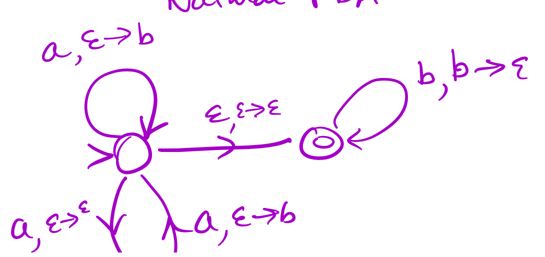


$S \to aSb \mid aaSb \mid \varepsilon$



$\{ a^n b^m \mid m \le n \le 2m \}$

Natural PDA.



Top down parsing answers the question, "is string w derivable in G?" by starting with S and applying rules of G ("lucky guessing" what rule to apply next) until it has a string of terminals on top of stack, which it "pop+ matches".

Bottom-up Parsing

- start with w
- apply rules of G "backwards" on w.
- see whether can get to S.
- also called a "Shift-Reduce" parser.

$S \to aaSb \mid \varepsilon$

$aaaabb$