

How to prove a CFG is correct for a L.

Eg $G: S \rightarrow aSb \mid \varepsilon \quad A^n B^n = \{ a^n b^n \mid n \geq 0 \}$

Claim: $L(G) = A^n B^n$.

Proof: I. $L(G) \subseteq A^n B^n$

i.e. We must show that any w generated in G is of the form $a^i b^i$ for some i .

Let w be any string generated in G .

Then either

1. $w = \varepsilon$, i.e. $w = a^0 b^0$, or

2. $w = \underline{a} w' \underline{b}$ where w' is a smaller string generated from S

Then by the Ind. Hyp., w' is of the form $a^i b^i$ for some i

Hence $w = a \cdot a^i b^i \cdot b = a^{i+1} b^{i+1}$, and $w \in A^n B^n$.

"Smaller" is a proxy for "has a shorter derivation"

Claim: $\forall w \in L(G), w = a^i b^i$ for some $i \geq 0$.

Proof: By induction on the number of steps in a shortest leftmost derivation of w .

Basis: Number of steps is 1

i.e. $w = \varepsilon$. i.e. $w = a^0 b^0$ \square

Induction: Let w be any string derivable in k steps, $k \geq 1$

Ind Hyp: Any string derivable in $< k$ steps is of form $a^i b^i$ for some value of i .

$\therefore w = a^{i+1} b^{i+1}$, i.e. w is of form $a^i b^i$ for some i \square

II. $A^n B^n \subseteq L(G)$

$$S \rightarrow aSb \mid \varepsilon$$

① ②

Let $w \in A^n B^n$. Then $w = \underline{a^i b^i}$ for some i .

Consider the following derivation of w in G :

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow \dots \Rightarrow a^i S b^i \Rightarrow a^i b^i$$

number of applications of rule ① is i . □

Eg. $L_1 = \{ w \in \{a, b\}^* \mid \#_a(w) = \#_b(w) \}$

A grammar G_1 for L_1 : $S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

② ③ ④ ①

Claim: $L(G_1) = L_1$.

Proof: Part 1: $L(G_1) \subseteq L_1$ ie $\forall w \in L(G_1)$, w has " $\#_a = \#_b$ "

1. ε has 0 a's, 0 b's ie has " $\#_a = \#_b$ "

2. $a w' b$ if w' has " $\#_a = \#_b$ ", then so does $a w' b$

3. $b w' a$ " w' " " " " $b w' a$

4. $w_1 w_2$ if w_1 and w_2 have " $\#_a = \#_b$ " then so does $w_1 w_2$



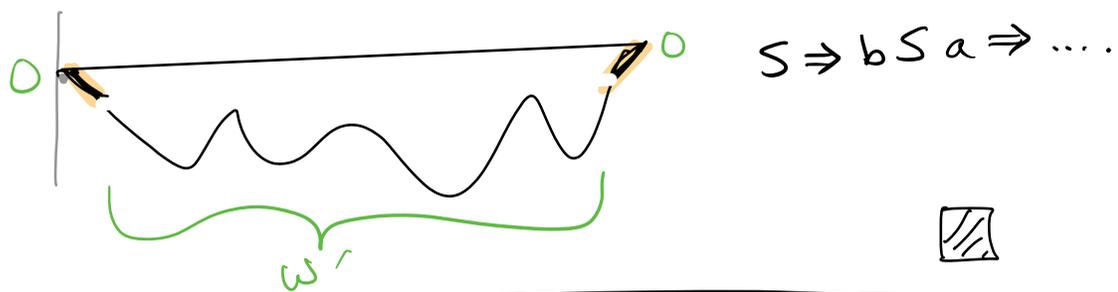
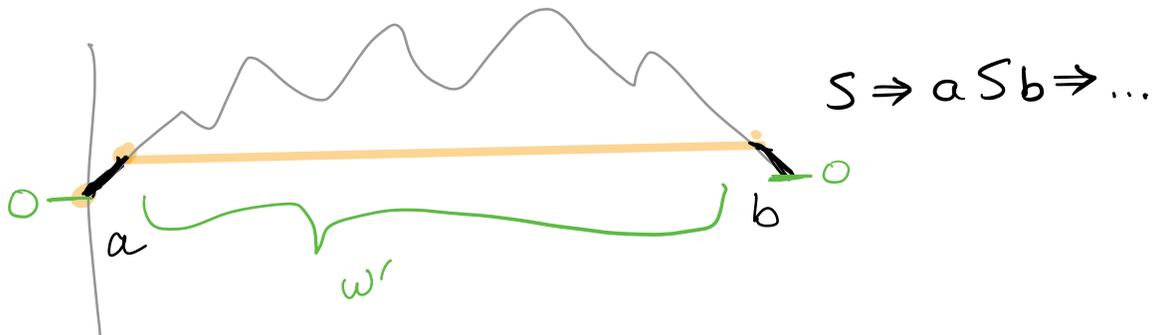
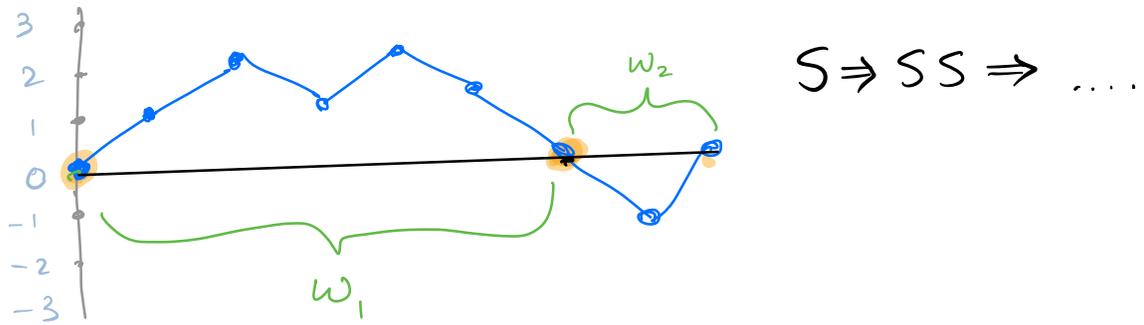
$$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$$

Part 2: $L_1 \subseteq L(G_1)$.

Let w be any string with $\#_a = \#_b$.

Let's graph the "excess a's" in the prefixes of w .

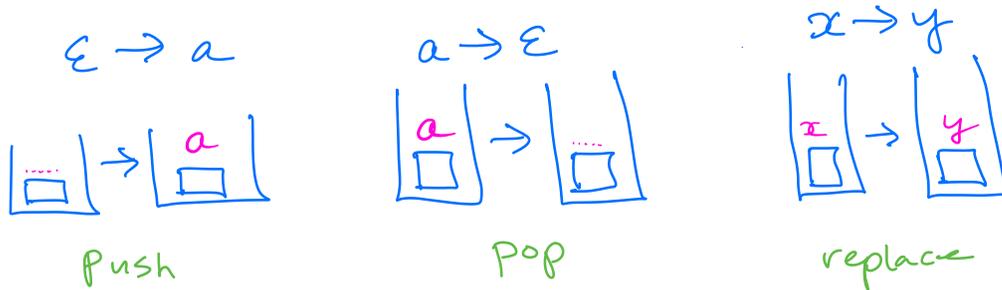
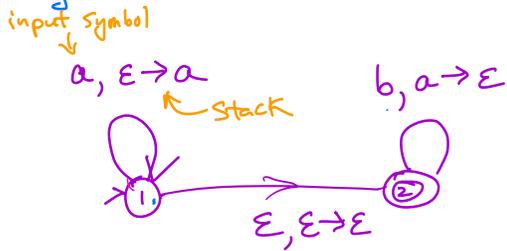
a a b a b b b a



Section 3.5 Push-Down Automata (PDAs)

- Like an NFA with a stack
- transition: in a state, can read - an input symbol
can read & pop what's on top of stack.
then decide - new state to go to
- push a new symbol on stack.
- A stack is LIFO. store of symbols
- can be arbitrarily large.

Eg a PDA for $A^n B^n$:



Definition 2.13 A Push-down Automaton (PDA) is

a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where:

- Q is a finite set of states
- Σ is input alphabet
- Γ is a stack alphabet
- $\delta: Q \times \Sigma_{\epsilon} \times \Gamma_{\epsilon} \rightarrow \mathcal{P}(Q \times \Gamma_{\epsilon})$ is transition function
- $q_0 \in Q$ is start state
- $F \subseteq Q$ is set of accept states.

$$\Sigma_{\epsilon} \text{ is } \Sigma \cup \{\epsilon\} \quad \Gamma_{\epsilon} \equiv \Gamma \cup \{\epsilon\}$$

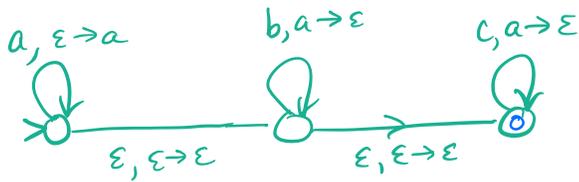
Note: the definition of PDA includes non-determinism.

Defn: A string w is accepted by a PDA M if \exists a sequence of transitions M can make on input w that leads to a final state & empty stack & (no input)

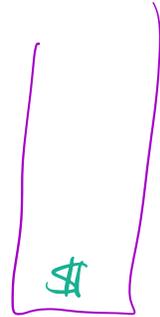
↑
accept

Eg $\{ a^{i+j} b^i c^j \mid i \geq 0, j \geq 0 \}$

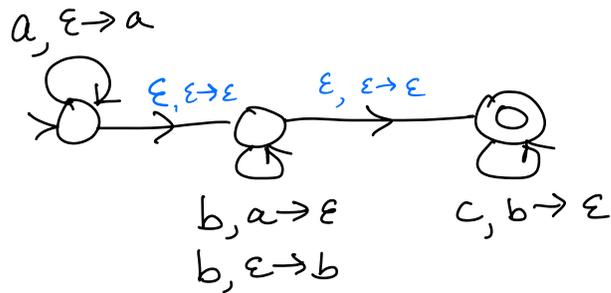
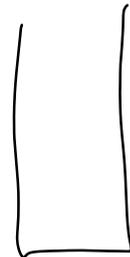
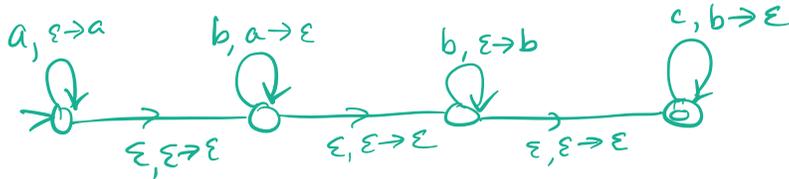
aaabcc



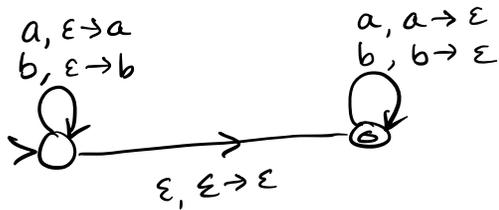
aaabc ↑



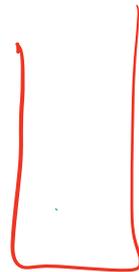
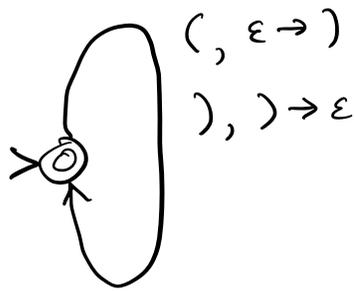
Eg $\{ a^i b^{i+j} c^j \mid i, j \geq 0 \}$



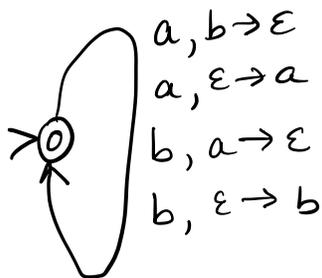
$$\text{Even Pal} = \{ ww^R \mid w \in \{a, b\}^* \}.$$



$$\text{Bal} = \{ w \in \{ (,) \}^* \mid \#_c(w) = \#_s(w) \text{ and } \forall \text{ prefix } w' \text{ of } w, \#_c(w') \geq \#_s(w') \}.$$



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



It would be nice to be able to detect the bottom of the stack...