# Context-Free and Noncontext-Free Languages

Chapter 13

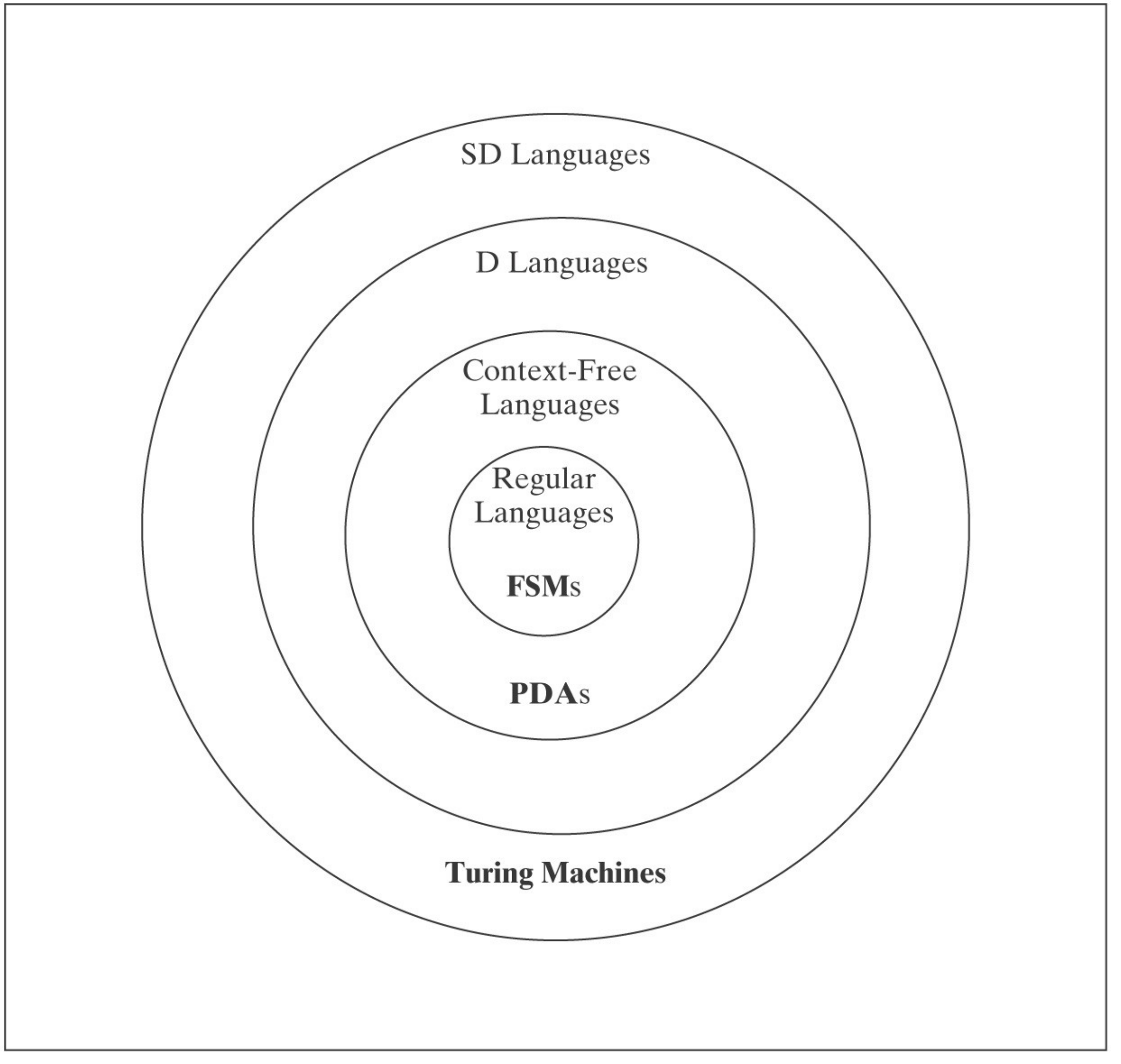
# Languages That Are and Are Not Context-Free

$\mathtt{a}^*\mathtt{b}^*$ is regular.

$A^nB^n = \{\mathtt{a}^n\mathtt{b}^n : n \geq 0\}$ is context-free but not regular.

$A^nB^nC^n = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n : n \geq 0\}$ is not context-free.

# Languages and Machines



# The Regular and the CF Languages

***Theorem:*** The regular languages are a proper subset of the context-free languages.

***Proof:*** In two parts:

- Every regular language is CF.
  - Every regular grammar is context-free.

  or
  - Every FSM is a PDA (that is, ignoring its stack).
- There exists at least one language that is CF but not regular.
  - $A^nB^n$

# How Many Context-Free Languages Are There?

***Theorem:*** There is a countably infinite number of CFLs.

***Proof:***

- Upper bound: we can lexicographically enumerate all the CFGs.

- Lower bound: {a}, {aa}, {aaa}, … are all CFLs.

So there must exist some languages that are not context-free:

$$\{a^n b^n c^n : n \geq 0\}$$

# Showing that L is Context-Free

Techniques for showing that a language *L* is context-free:

1. Exhibit a context-free grammar for *L*.

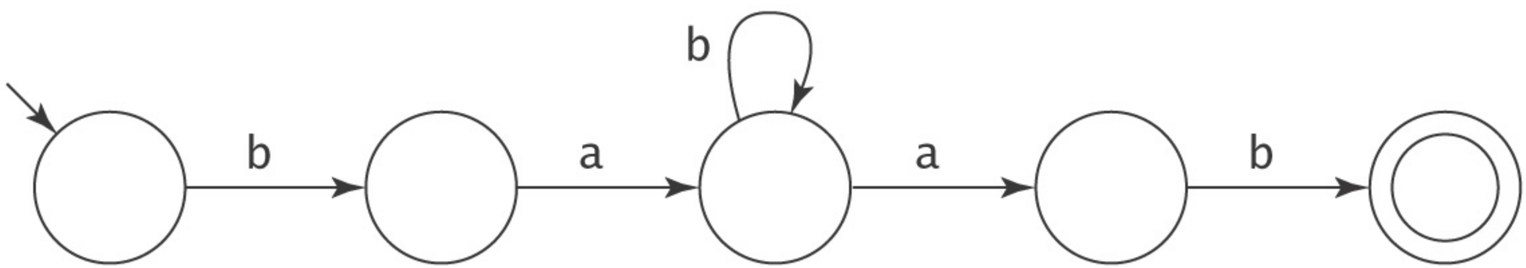
2. Exhibit a PDA for *L*.

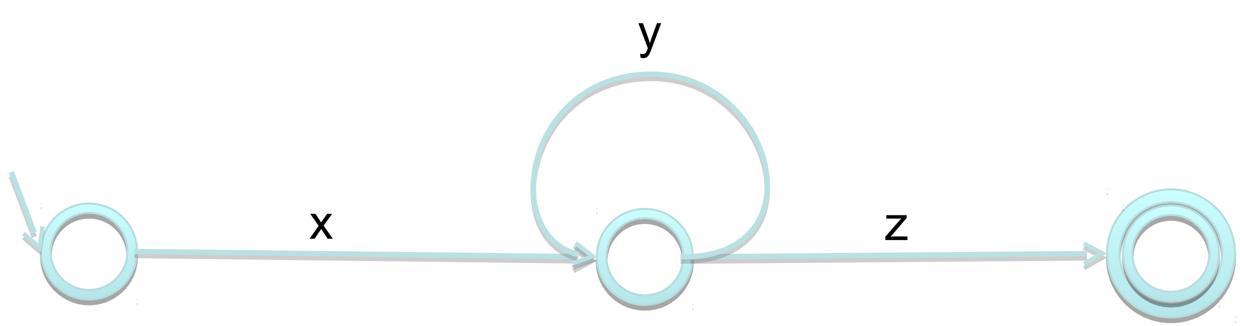
3. Use the closure properties of context-free languages.

   Unfortunately, these are weaker than they are for regular languages.

# Showing that *L* is Not Context-Free

Remember the pumping argument for regular languages:



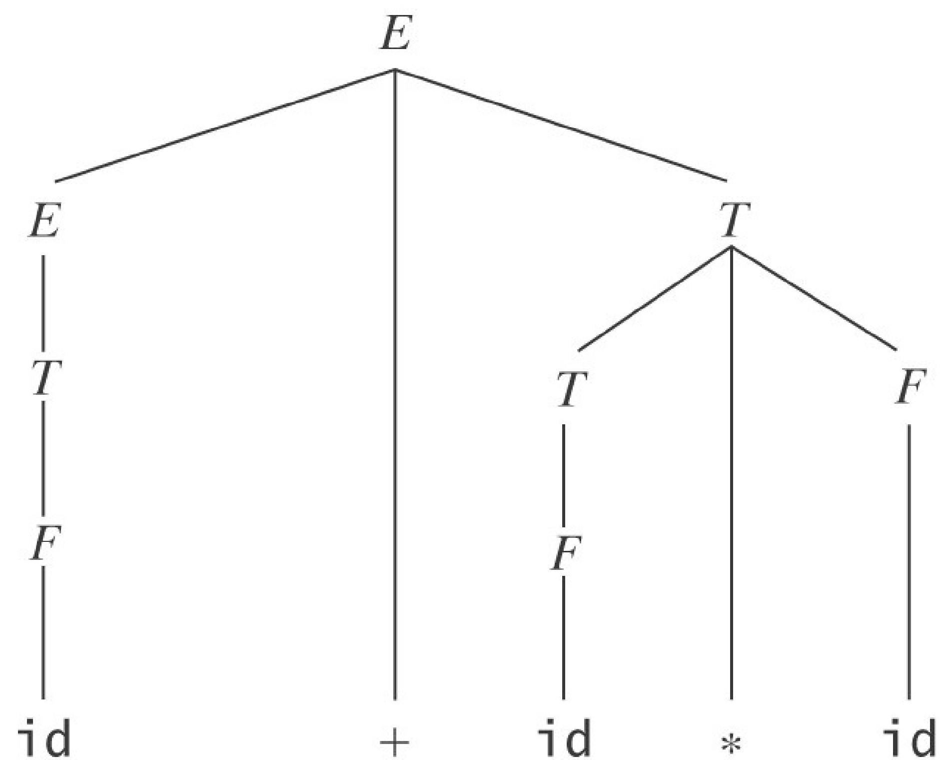
# A Review of Parse Trees

A parse tree, derived by a grammar $G = (V, \Sigma, R, S)$, is a rooted, ordered tree in which:

- Every leaf node is labeled with an element of $\Sigma \cup \{\varepsilon\}$,
- The root node is labeled $S$,
- Every other node is labeled with some element of $V - \Sigma$,
- If $m$ is a nonleaf node labeled $X$ and the children of $m$ are labeled $x_1, x_2, \ldots, x_n$, then the rule $X \rightarrow x_1, x_2, \ldots, x_n$ is in $R$.

# Example

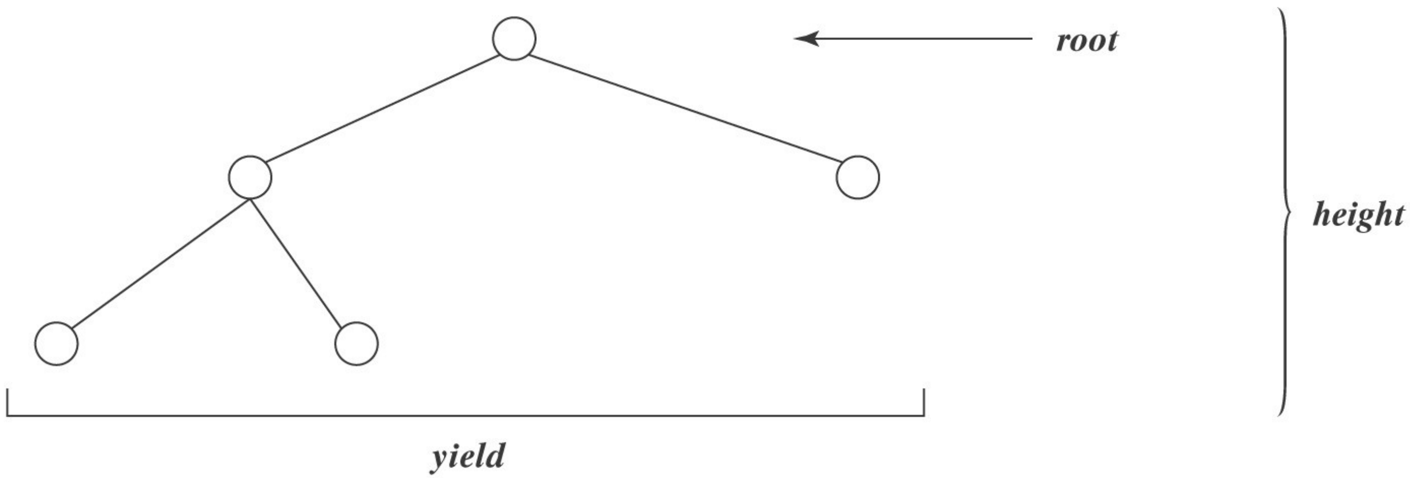
$E \to E + T$
$E \to T$
$T \to T * F$
$T \to F$
$F \to (E)$
$F \to \text{id}$



# Some Tree Basics

The ***height*** of a tree is the length of the longest path from the root to any leaf.

The ***branching factor*** of a tree is the largest number of children associated with any node in the tree.



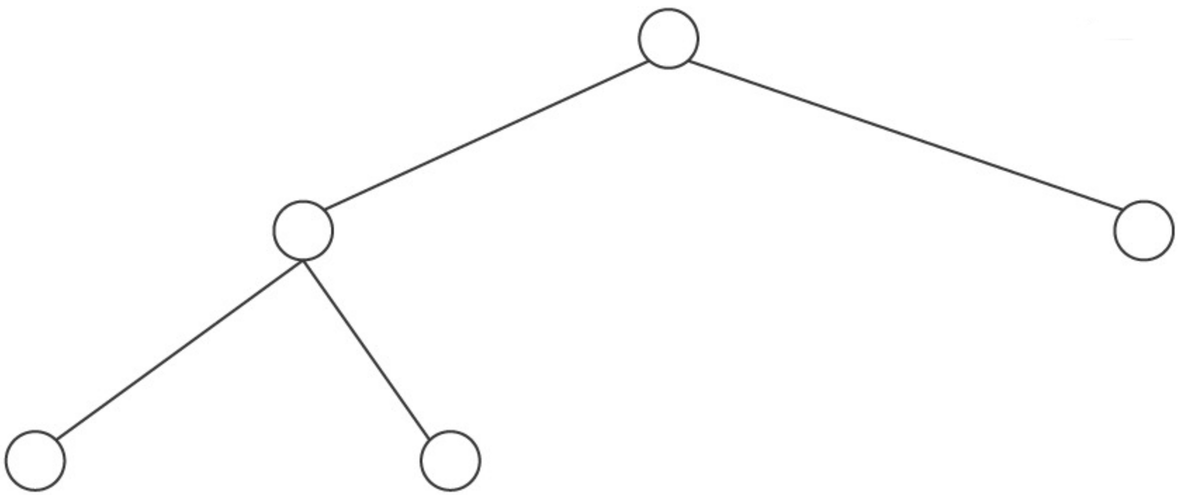

***Theorem:*** The length of the yield of any tree $T$ with height $h$ and branching factor $b$ is $\leq b^h$.

# From Grammars to Trees

Given a context-free grammar *G*:

- Let *n* be the number of nonterminal symbols in *G*.
- Let *b* be the branching factor of *G*, i.e., $\max\{|\alpha| : A \rightarrow \alpha \text{ in } G\}$

Suppose that *T* is generated by *G* and no nonterminal appears more than once on any path:



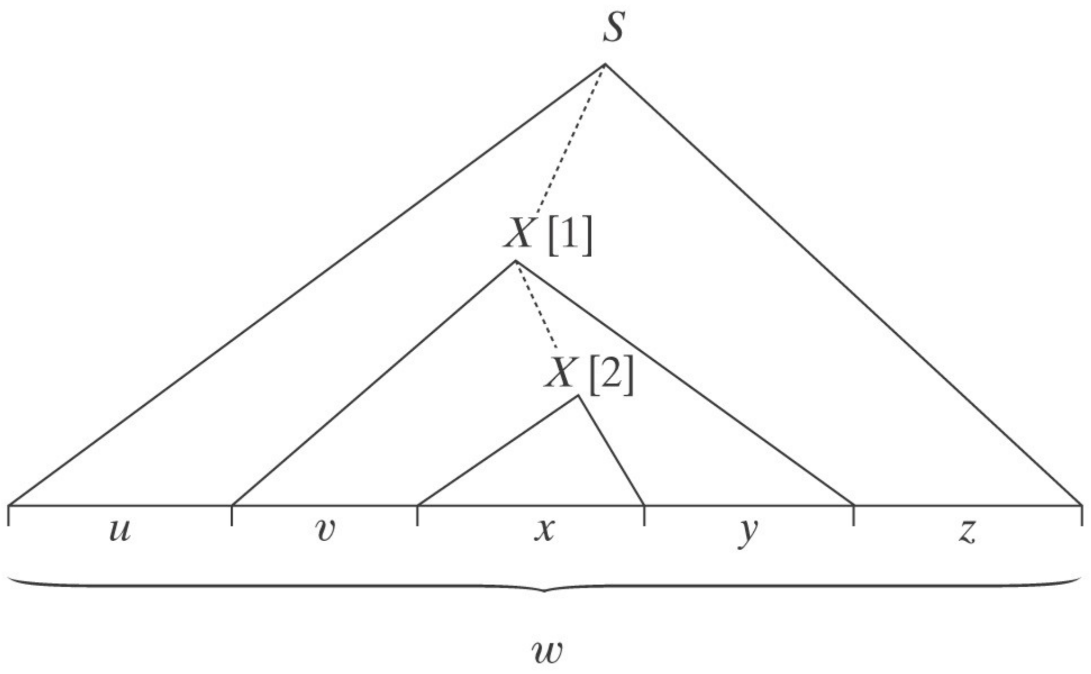
The maximum height of *T* is: $n$

The maximum length of *T*'s yield is: $b^n$

# The Context-Free Pumping Theorem

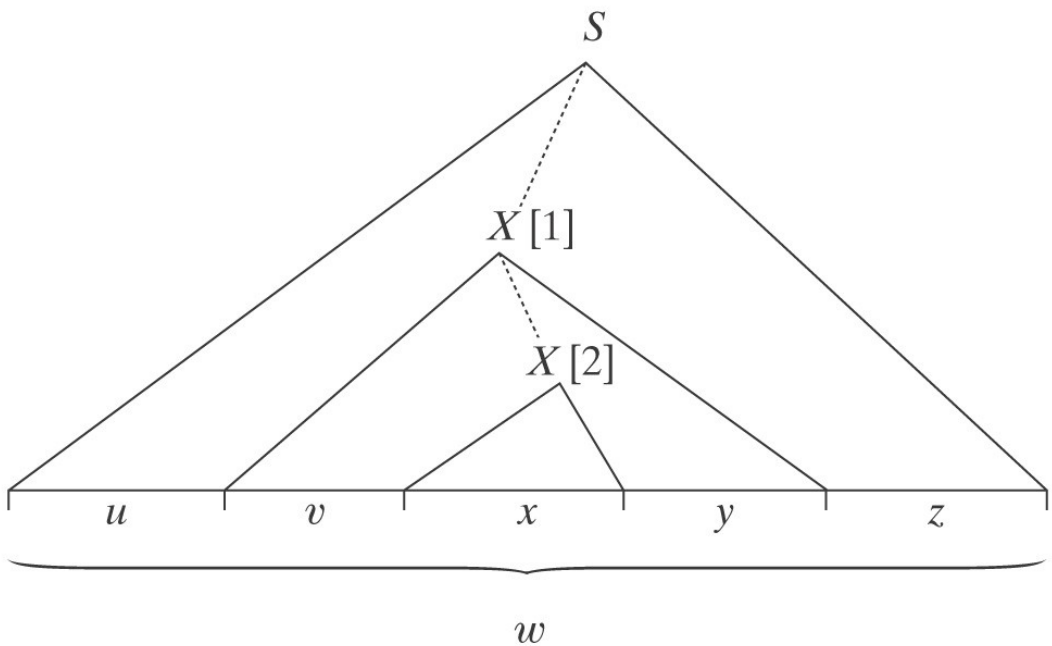
This time we use parse trees, not automata as the basis for our argument.

If $w$ is "long" ($|w| > b^n$), then its parse trees must look like:




Choose one such tree such that there's no other with fewer nodes.
Choose the X's the bottommost instances of a repeating nonterminal.

# The Context-Free Pumping Theorem




We have the derivations:

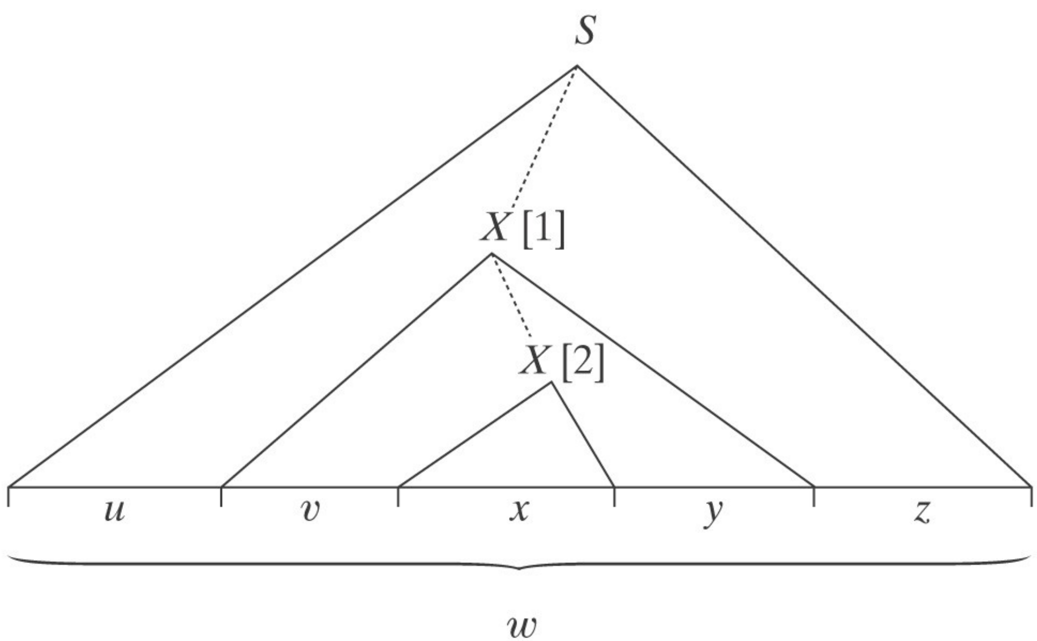
$S \Rightarrow^* uXz \Rightarrow^* uxz \in L(G)$

$S \Rightarrow^* uXz \Rightarrow^* uvXyz \Rightarrow^* uvvXyyz \Rightarrow^* uvvxyyz \in L(G)$

We can similarly derive all the strings: $uv^2xy^2z$, $uv^3xy^3z$, … $\in L(G)$

Thus: $\forall q \geq 0,\ uv^qxy^qz \in L(G)$

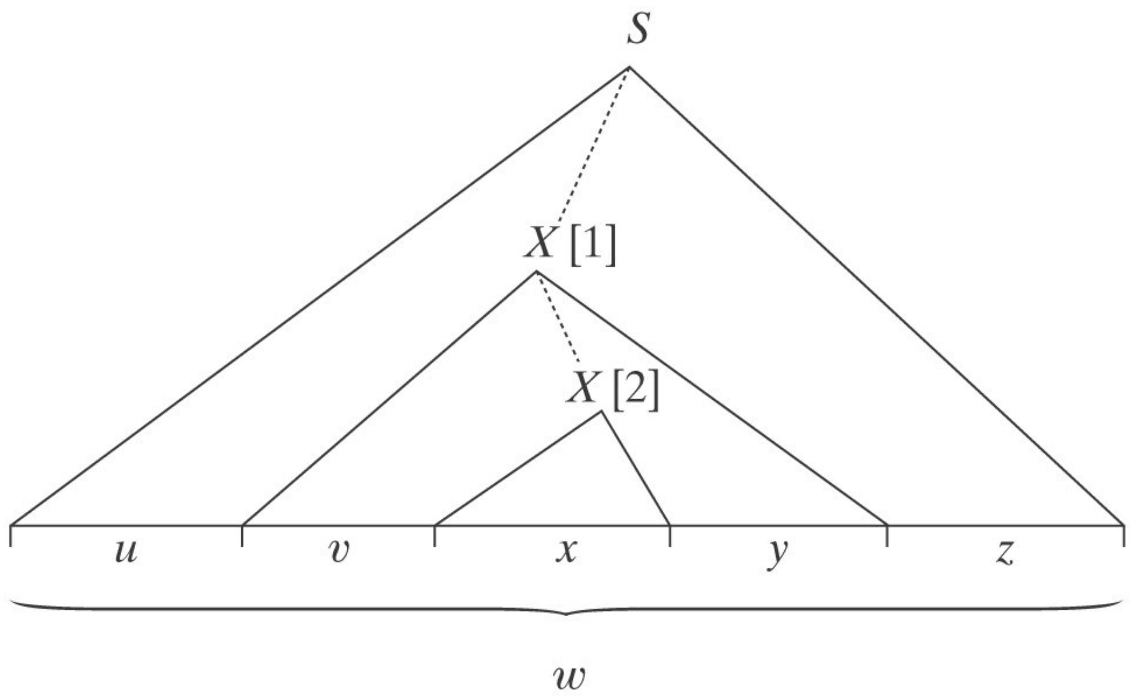
# The Context-Free Pumping Theorem




**$vy \neq \varepsilon$**

Proof: If $vy = \varepsilon$, then the derivation $S \Rightarrow^* uXz \Rightarrow^* uxz$ would also yield $w$ and it would create a parse tree with fewer nodes. But that contradicts the assumption that we started with a tree with the smallest possible number of nodes.

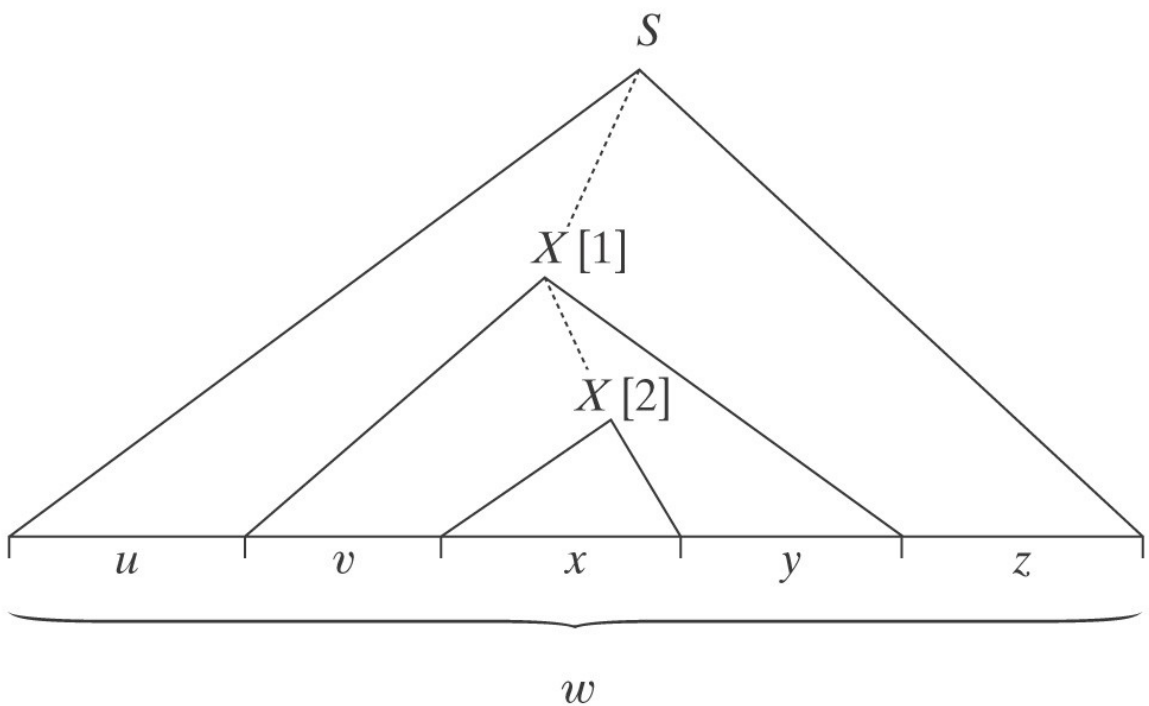
# The Context-Free Pumping Theorem




The height of the subtree rooted at $X[1]$ is at most: $n + 1$ (because the X's are the bottommost repeated nonterminals)

So $|vxy| \leq b^{n+1}$.

# The Context-Free Pumping Theorem



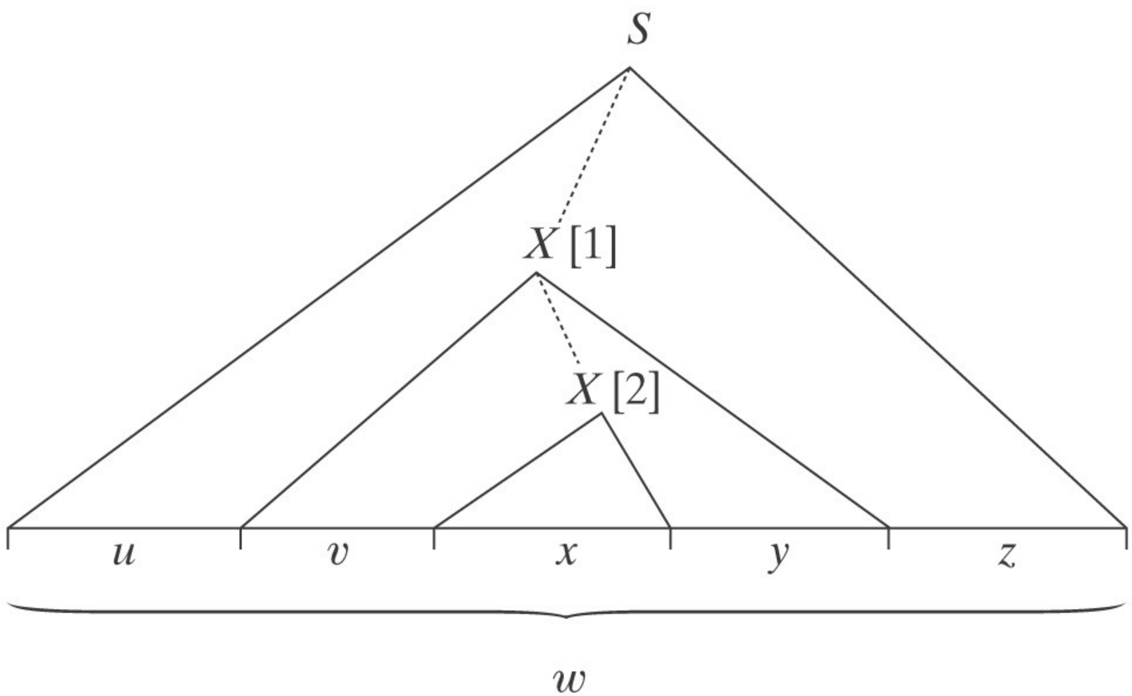

If $L$ is a context-free language, then $\exists k \geq 1$ such that any $w \in L$ with $|w| \geq k$ can be written as $w = uvxyz$, for some $u, v, x, y, z \in \Sigma^*$, such that

- $vy \neq \varepsilon$,
- $|vxy| \leq k$ and
- $\forall q \geq 0$, $uv^qxy^qz \in L$

# What Is *k*?

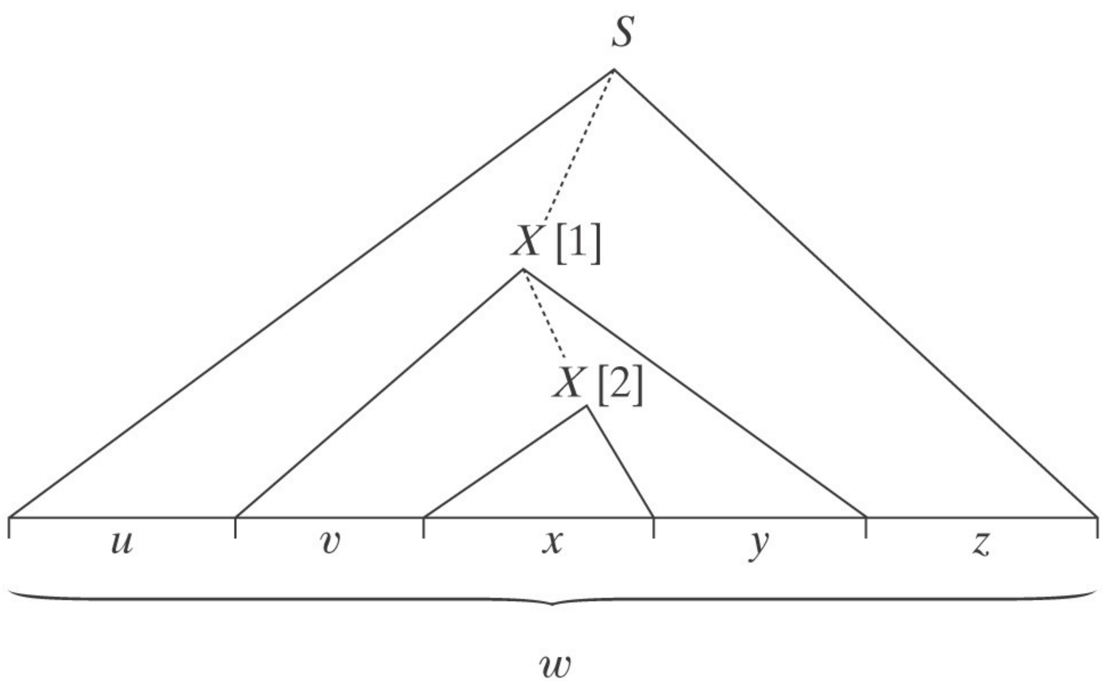
*k* serves two roles:

- How long must *w* be to guarantee it is pumpable?
- What's the bound on $|vxy|$?




Let $n$ be the number of nonterminals in $G$.
Let $b$ be the branching factor of $G$.

# How Long Must *w* be?

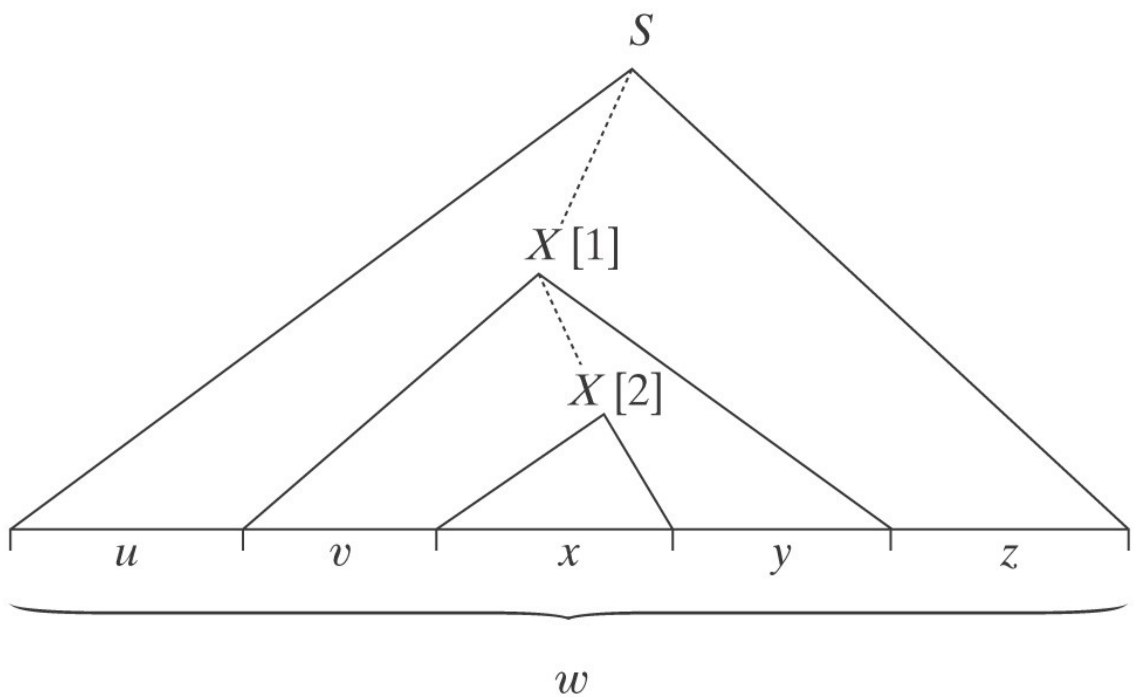



If *height*(*T*) > *n*, then some nonterminal occurs more than once on some path. So *T* is pumpable.

If *height*(*T*) $\leq$ *n*, then |*uvxyz*| $\leq$ $b^n$.

So if |w| = |*uvxyz*| > $b^n$, *w* = *uvxyz* must be pumpable.

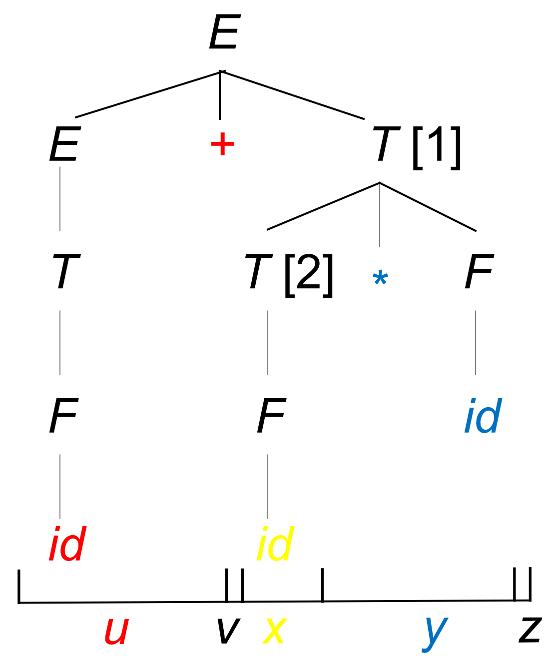
# What's the Bound on $|vxy|$?




Recall that we are considering the bottom-most two instances of a repeated nonterminal. Then the yield of the upper one has length at most $b^{n+1}$. That is, $|vxy| \leq b^{n+1}$.

So, we need $k = \max(b^n+1, b^{n+1})$; let $\boldsymbol{k = b^{n+1}}$.

# Example




grammar:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow \text{id}$

$n = 3$

$b = 3$

$k = 3^4 = 81$

string

$w = \text{id + id * id}$

$w = uvxyz$

$u = \text{id+}$

$v = \varepsilon$

$x = \text{id}$

$y = \text{*id}$

$z = \varepsilon$

$vy \neq \varepsilon$,

$|vxy| \leq k$

$\forall q \geq 0$:

$uv^qxy^qz = id{+}id(*id)^q$ is in the language $L(G)$

# An Example of Pumping: $A^nB^nC^n$

$A^nB^nC^n = \{a^n b^n c^n, n \geq 0\}$

Choose $w = a^k\ b^k\ c^k = uvxyz$

1 | 2 | 3

If *v* or *y* spans regions, then for $q = 2$ the order of the letters changes.

If *v* and *y* each contain only one letter, then for $q = 2$ the numbers of letters are not the same.

# An Example of Pumping: $\{ a^{n^2} : n \geq 0\}$

$L = \{ a^{n^2}, n \geq 0\}$.

For $n = k^2$ we have $w = a^{k^4}$ = uvxyz

$vy = a^p$, for some nonzero $p$.

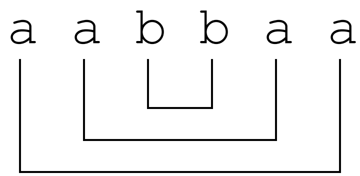
For q=2, $a^{k^4+p}$ must be in $L$ but it is too short.

The next longer string in $L$, for $n = k^2+1$, is $a^{(k^2+1)^2} = a^{k^4+2k^2+1}$

That means, $p = 2k^2+1$ but $p = |vy| \leq k$.

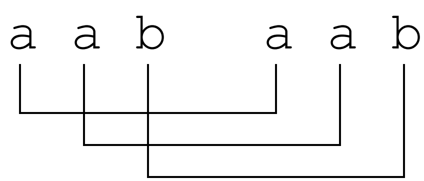
# Nested and Cross-Serial Dependencies

PalEven = $\{ww^R : w \in \{\texttt{a}, \texttt{b}\}^*\}$




The dependencies are nested – is context-free

WW = $\{ww : w \in \{\texttt{a}, \texttt{b}\}^*\}$




Cross-serial dependencies – is not context-free

# WW = {*ww* : *w* ∈ {a, b}*}

Let $a^{k+1}b^{k+1}a^{k+1}b^{k+1} = uvxyz$

For $q=0$, we have that $uxz \in$ WW, that is, $uxz = ww$ for some $w$.

Observe that $uxz$ still has the shape $a^+b^+a^+b^+$ (because $|vxy| \leq k$ so deleting $v$ and $y$ cannot remove more than $k$ symbols). Moreover, only one or two (adjacent) same-letter regions are affected. Also, $uxz = ww$ means that $w$ must have the shape $a^+b^+$.

If one region only is decreased, then *ww* is not in WW.

If two (adjacent) regions are decreased, then *ww* is not in WW.

# Closure Theorems for Context-Free Languages

The context-free languages are closed under:

- Union
- Concatenation
- Kleene star
- Reverse

# Closure Under Union

Let $G_1 = (V_1, \Sigma_1, R_1, S_1)$, and
$G_2 = (V_2, \Sigma_2, R_2, S_2)$.

Assume that $G_1$ and $G_2$ have disjoint sets of nonterminals, not including $S$.

Let $L = L(G_1) \cup L(G_2)$.

We can show that $L$ is CF by exhibiting a CFG for it:

$$G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\}, S)$$

# Closure Under Concatenation

Let $G_1 = (V_1, \Sigma_1, R_1, S_1)$, and
$G_2 = (V_2, \Sigma_2, R_2, S_2)$.

Assume that $G_1$ and $G_2$ have disjoint sets of nonterminals, not including $S$.

Let $L = L(G_1)L(G_2)$.

We can show that $L$ is CF by exhibiting a CFG for it:

$$G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \rightarrow S_1S_2\}, S)$$

# Closure Under Kleene Star

Let $G = (V, \Sigma, R, S_1)$.

Assume that $G$ does not have the nonterminal $S$.

Let $L = L(G)^*$.

We can show that $L$ is CF by exhibiting a CFG for it:

$$G = (V_1 \cup \{S\}, \Sigma_1,$$
$$R_1 \cup \{S \rightarrow \varepsilon, S \rightarrow SS_1\},$$
$$S)$$

# Closure Under Reverse

$L^R = \{w \in \Sigma^* : w = x^R \text{ for some } x \in L\}$.

Let $G = (V, \Sigma, R, S)$ be a context-free grammar.

Every rule in $G$ is of the form $X \rightarrow \alpha$, for $\alpha \in V^*$.

Construct, from $G$, a new grammar $G'$, such that $L(G') = L^R$:

$G' = (V_G, \Sigma_G, R', S_G)$, where $R'$ is constructed as follows:

- For every rule $X \rightarrow \alpha$ in $G$, add $X \rightarrow \alpha^R$.

# Closure Under Intersection

The context-free languages are not closed under intersection:

The proof is by counterexample. Let:

$L_1 = \{a^n b^n c^m : n, m \geq 0\}$ /* equal a's and b's.
$L_2 = \{a^m b^n c^n : n, m \geq 0\}$ /* equal b's and c's.

Both $L_1$ and $L_2$ are context-free, since there exist straightforward context-free grammars for them.

But now consider:

$$L = L_1 \cap L_2$$
$$= \{a^n b^n c^n : n \geq 0\}$$

# Closure Under Complement

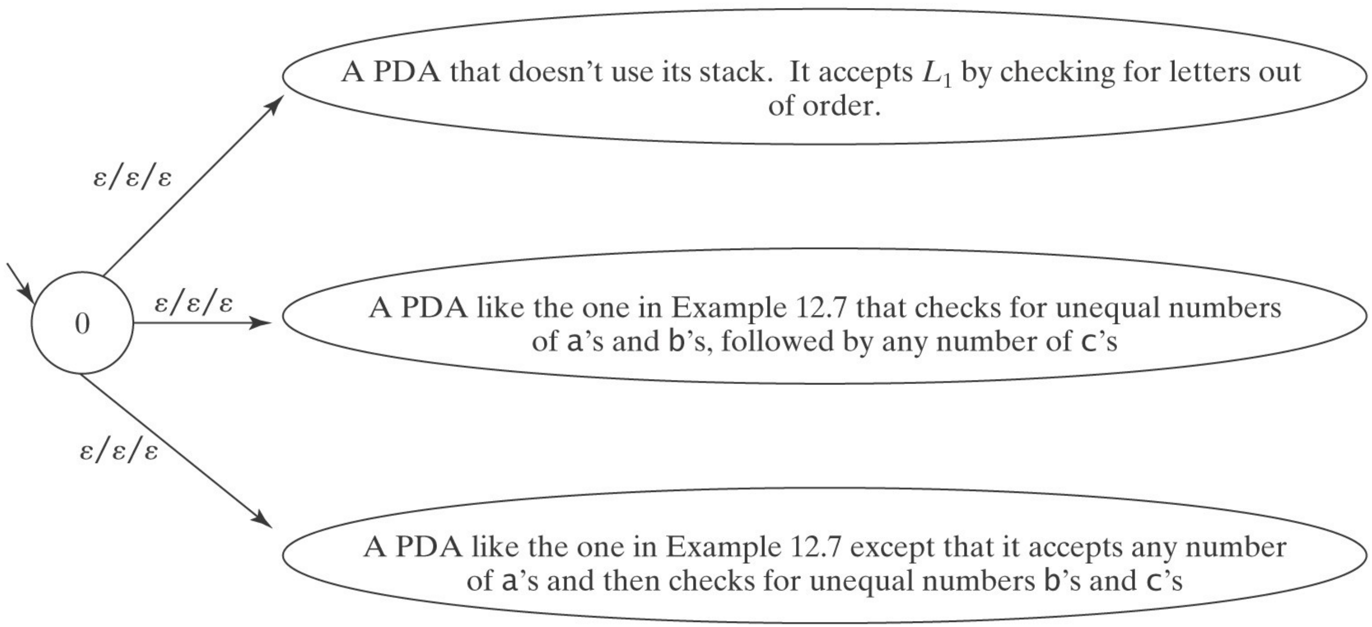
The context-free languages are not closed under complement:

$$L_1 \cap L_2 = \neg(\neg L_1 \cup \neg L_2)$$

The context-free languages are closed under union, so if they were closed under complement, they would be closed under intersection (which they are not).

# Closure Under Complement An Example

$\neg A^n B^n C^n$ is context-free:




But $\neg(\neg A^n B^n C^n) = A^n B^n C^n$ is not context-free.

# Closure Under Difference

The context-free languages are not closed under difference:

$\neg L = \Sigma^* - L.$

$\Sigma^*$ is context-free. So, if the context-free languages were closed under difference, the complement of any context-free language would necessarily be context-free. But we just showed that that is not so.

# The Intersection of a Context-Free Language and a Regular Language is Context-Free

$L = L(M_1)$, a PDA = $(K_1, \Sigma, \Gamma_1, \Delta_1, s_1, A_1)$.
$R = L(M_2)$, a deterministic FSM = $(K_2, \Sigma, \delta, s_2, A_2)$.

We construct a new PDA, $M_3$, that accepts $L \cap R$ by simulating the parallel execution of $M_1$ and $M_2$.

$M = (K_1 \times K_2, \Sigma, \Gamma_1, \Delta, (s_1, s_2), A_1 \times A_2)$.

Insert into $\Delta$:

For each rule $((q_1, a, \beta), (p_1, \gamma))$ in $\Delta_1$,
and each rule $(q_2, a, p_2)$ in $\delta$,

$$(((q_1, q_2), a, \beta), ((p_1, p_2), \gamma)).$$

For each rule $((q_1, \varepsilon, \beta), (p_1, \gamma))$ in $\Delta_1$,
and each state $q_2$ in $K_2$,

$$(((q_1, q_2), \varepsilon, \beta), ((p_1, q_2), \gamma)).$$

This works because: we can get away with only one stack.

# The Difference between a Context-Free Language and a Regular Language is Context-Free

***Theorem:*** The difference ($L_1 - L_2$) between a context-free language $L_1$ and a regular language $L_2$ is context-free.

***Proof:*** $L_1 - L_2 = L_1 \cap \neg L_2$.

If $L_2$ is regular then so is $\neg L_2$.

If $L_1$ is context-free, so is $L_1 \cap \neg L_2$.

# Using intersection with a regular language

$L = \{w \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^* : \#_{\mathtt{a}}(w) = \#_{\mathtt{b}}(w) = \#_{\mathtt{c}}(w)\}$

If $L$ were context-free, then $L' = L \cap \mathtt{a}^*\mathtt{b}^*\mathtt{c}^*$ would also be context-free.

But $L' = \mathtt{A}^n\mathtt{B}^n\mathtt{C}^n$

So neither is $L$.