Parse trees
Pumping Lemma for CFLs

Parse Trees

$G = (N, \Sigma, P, S)$. A **parse tree** will be a tree with the following properties:

- Every vertex is labelled by a symbol from $N \cup \Sigma \cup \{\epsilon\}$

Parse Trees

$G = (N, \Sigma, P, S)$. A **parse tree** will be a tree with the following properties:

- Every vertex is labelled by a symbol from $N \cup \Sigma \cup \{\epsilon\}$
- The root is labelled S

Parse Trees

$G = (N, \Sigma, P, S)$. A **parse tree** will be a tree with the following properties:

- Every vertex is labelled by a symbol from $N \cup \Sigma \cup \{\epsilon\}$
- The root is labelled S
- An interior vertex is labelled from N

Parse Trees

$G = (N, \Sigma, P, S)$. A **parse tree** will be a tree with the following properties:

- Every vertex is labelled by a symbol from $N \cup \Sigma \cup \{\epsilon\}$
- The root is labelled S
- An interior vertex is labelled from N
- If vertex v is labelled with A and its children $v_1, v_2, \ldots, v_n$ are labelled $X_1, X_2, \ldots, X_n$ respectively, then $A \to X_1 X_2 \ldots X_n \in P$.

Parse Trees

$G = (N, \Sigma, P, S)$. A **parse tree** will be a tree with the following properties:

- Every vertex is labelled by a symbol from $N \cup \Sigma \cup \{\epsilon\}$
- The root is labelled S
- An interior vertex is labelled from N
- If vertex v is labelled with A and its children $v_1, v_2, \ldots, v_n$ are labelled $X_1, X_2, \ldots, X_n$ respectively, then $A \to X_1 X_2 \ldots X_n \in P$.
- If a vertex is labelled with ϵ then it is a leaf and it is the only child of its parent.

Parse Trees

$G = (N, \Sigma, P, S)$. Take a parse tree T with root labelled S.

- Order the leaves of the tree from left to right (order according to inorder/preorder): gives a sentential form derived from S (Can be proven by induction).
  This is the sentential form represented by T.

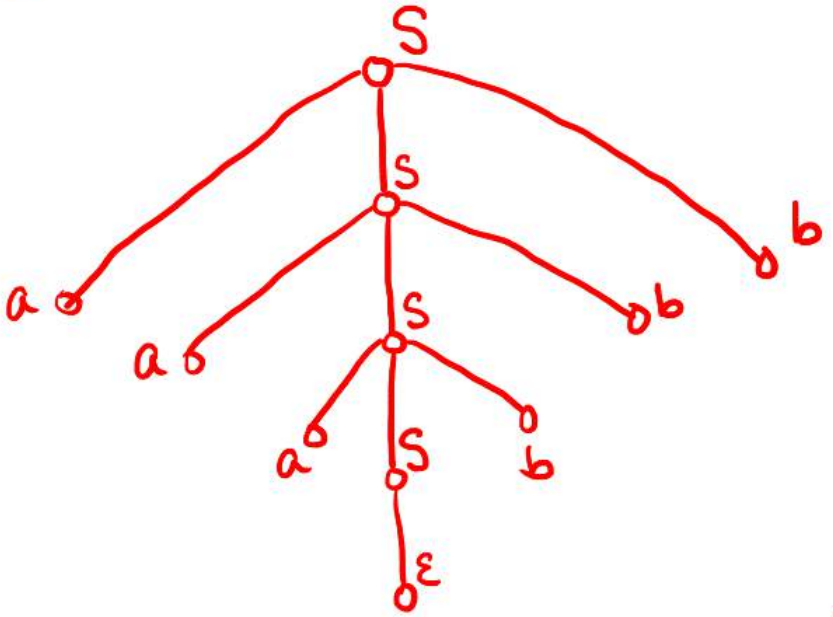
Parse Trees

$G = (N, \Sigma, P, S)$. Take a parse tree T with root labelled S.

- Order the leaves of the tree from left to right (order according to inorder/preorder): gives a sentential form derived from S (Can be proven by induction).
  This is the sentential form represented by T.
- If all leaves are labelled from $\Sigma \cup \epsilon$, then this tree corresponds to a sentence of $L(G)$.

Example 1: Parse Tree for a^3b^3

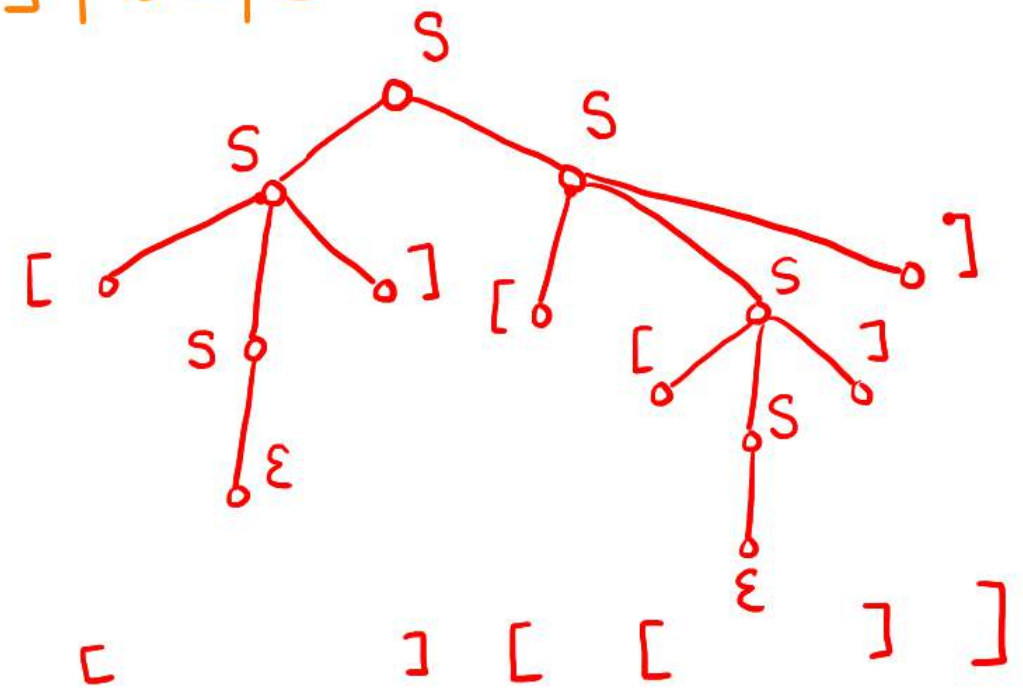
$S \to aSb \mid \varepsilon$



a a a b b b

Example 2: Parse Tree for [][[]]

$S \rightarrow [S] \mid SS \mid \varepsilon$



Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.

$A \rightarrow \omega_1 \underline{B} \omega_2 C$

Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.
- **Rightmost derivation**: At each step of derivation, a production is applied to the rightmost nonterminal.

$A \rightarrow \omega_1 B \omega_2 C$

Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.
- **Rightmost derivation**: At each step of derivation, a production is applied to the rightmost nonterminal.
- Note: pre-order traversal of a parse tree corresponds to a leftmost derivation of the sentential form.

Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.
- **Rightmost derivation**: At each step of derivation, a production is applied to the rightmost nonterminal.
- Note: pre-order traversal of a parse tree corresponds to a leftmost derivation of the sentential form.
- This leftmost derivation of the sentential form is unique to the parse tree. But the sentential form may have many non-isomorphic parse trees.

Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.
- **Rightmost derivation**: At each step of derivation, a production is applied to the rightmost nonterminal.
- Note: pre-order traversal of a parse tree corresponds to a leftmost derivation of the sentential form.
- This leftmost derivation of the sentential form is unique to the parse tree. But the sentential form may have many non-isomorphic parse trees.
- **Ambiguous CFG**: when a sentence has two leftmost or two rightmost derivations in the grammar.

Leftmost and Rightmost derivations

- **Leftmost derivation**: At each step of derivation, a production is applied to the leftmost nonterminal.
- **Rightmost derivation**: At each step of derivation, a production is applied to the rightmost nonterminal.
- Note: pre-order traversal of a parse tree corresponds to a leftmost derivation of the sentential form.
- This leftmost derivation of the sentential form is unique to the parse tree. But the sentential form may have many non-isomorphic parse trees.
- **Ambiguous CFG**: when a sentence has two leftmost or two rightmost derivations in the grammar.
- **Inherently ambiguous CFL**: When every CFG is ambiguous. Such CFLs exist.

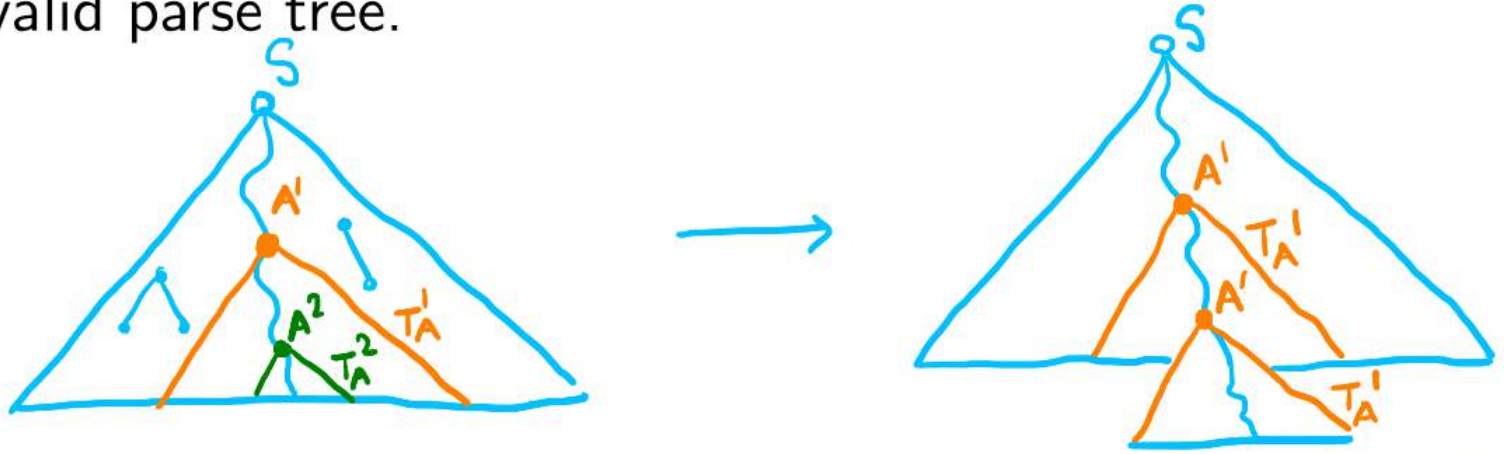
Pumping Lemma for CFLs

For every CFL A, there exists $k \geq 0$ such that every $z \in A$ of length at least k can be broken up into five substrings $z = uvwxy$ such that $vx \neq \epsilon, |vwx| \leq k$, and for all $i \geq 0, uv^i wx^i y \in A$.

Note the difference from Pumping Lemma for regular sets. Here we are simultaneously pumping two substrings v, x, separated by a substring w.

Proof of Pumping Lemma for CFLs

- Intuitive proof: Consider the CNF for $A - \{\epsilon\}$. A parse tree for any sufficiently long string z must have a "very long" path, as each node can have at most 2 children.
- Any "very long" path must have at least two occurrences of some nonterminal symbol A, call them A^1, A^2.
- Let T_A^1 and T_A^2 be the subtrees rooted at A^1, A^2 respectively. We can throw out T_A^2, in its place put in T_A^1 and this will still be a valid parse tree.

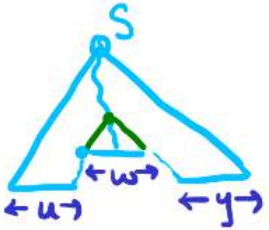


Proof of Pumping Lemma for CFLs

- Let $|N| = n$. Take $k = 2^{n+1}$.
- Suppose $z \in A$ and $|z| \geq k$.
- Draw a parse tree for the derivation of z from the CNF grammar of A. Each node has at most 2 children (subtree of a binary tree).
- Depth of the parse tree is at least $n + 1$. Consider the longest path in the parse tree.
- Longest path has at least $n + 1$ nonterminals - by PHP there must be some nonterminal that appears twice on the path. Reading from the bottom, let A be the first non-terminal appearing twice on the path.

Proof of Pumping Lemma for CFLs

- Take the two occurrences of A that are farthest from the root.
- Break z into $uvwxy$: w is generated by the lower occurrence of A (A^2) and vwx is generated by the upper occurrence of A (A^1).
- By CNF, each node will have at least 1 child, so $vx \neq \epsilon$.
- Tree rooted at A^i is T_A^i, $i \in \{1, 2\}$.
- If we replace T_A^1 by T_A^2, then we get the string $uwy = uv^0wx^0y$.



- If T_A^2 is replaced by T_A^1, the string generated is uv^2wx^2y! We can repeated replace the lowest T_A^2 by T_A^1 $k \geq 1$ times to get $uv^{k+1}wx^{k+1}y$.
- $|vwx| \leq k$: By choice of A, T_A^1 can have height at most $n + 1$, and therefore have at most $2^{n+1} = k$ leaves.

Use of PL for CFL

Sufficient property that a language is not a CFL:
For all $k \geq 0$, there exists $z \in A$ of length at least k such that for all ways of breaking z up into substrings $z = uvwxy$ with $vx \neq \epsilon$ and $|vwx| \leq k$, there exists an $i \geq 0$ such that $uv^i wx^i y \notin A$.

Use of PL for CFL: Adversarial Game

- The adversary picks $k \geq 0$.

Use of PL for CFL: Adversarial Game

- The adversary picks $k \geq 0$.
- You pick $z \in A$ of length at least k.

Use of PL for CFL: Adversarial Game

- The adversary picks $k \geq 0$.
- You pick $z \in A$ of length at least k.
- The adversary picks u, v, w, x, y such that $z = uvwxy, |vx| \neq \epsilon, |vwx| \leq k$.

Use of PL for CFL: Adversarial Game

- The adversary picks $k \geq 0$.
- You pick $z \in A$ of length at least k.
- The adversary picks u, v, w, x, y such that $z = uvwxy, |vx| \neq \epsilon, |vwx| \leq k$.
- You pick $i \geq 0$.

Use of PL for CFL: Adversarial Game

- The adversary picks $k \geq 0$.
- You pick $z \in A$ of length at least k.
- The adversary picks u, v, w, x, y such that $z = uvwxy, |vx| \neq \epsilon, |vwx| \leq k$.
- You pick $i \geq 0$.
- If $uv^i wx^i y \notin A$, then you win.

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$
- The adversary picks u, v, w, x, y with $vx \neq \epsilon$ and $|vwx| \leq k$.

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$
- The adversary picks u, v, w, x, y with $vx \neq \epsilon$ and $|vwx| \leq k$.
- You pick $i = 2$ to win; $z' = uv^2wx^2y \notin A$:

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$
- The adversary picks u, v, w, x, y with $vx \neq \epsilon$ and $|vwx| \leq k$.
- You pick $i = 2$ to win; $z' = uv^2wx^2y \notin A$:
- *Case 1*: v or x has at least one a and at least one b: z' is not of the form $a^*b^*a^*$.

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$
- The adversary picks u, v, w, x, y with $vx \neq \epsilon$ and $|vwx| \leq k$.
- You pick $i = 2$ to win; $z' = uv^2wx^2y \notin A$:
- *Case 1*: v or x has at least one a and at least one b: z' is not of the form $a^*b^*a^*$.
- *Case 2*: v and x have only a's. Then number of a's is more than twice the number of b's in z'. Similarly, if v, x have only b's.

Example 1

Set $\{a^n b^n a^n | n \geq 0\}$ is not a CFL.

- Given a k, pick $z = a^k b^k a^k$; $|z| = 3k$
- The adversary picks u, v, w, x, y with $vx \neq \epsilon$ and $|vwx| \leq k$.
- You pick $i = 2$ to win; $z' = uv^2wx^2y \notin A$:
- *Case 1*: v or x has at least one a and at least one b: z' is not of the form $a^* b^* a^*$.
- *Case 2*: v and x have only a's. Then number of a's is more than twice the number of b's in z'. Similarly, if v, x have only b's.
- *Case 3*: One of v, x has only a's and the other has only b's. Then z' cannot be of the form $a^m b^m a^m$.

Example 2

Set $A = \{ww | w \in \{a, b\}^*\}$ is not a CFL.

- It suffices to show that set
  $A' = \{a^n b^m a^n b^m | m, n \geq 0\} = A \cap L(a^* b^* a^* b^*)$ is not a CFL:
  $L(a^* b^* a^* b^*)$ is a regular set.
  Fact: Intersection of regular sets and CFLs are CFLs. [Try this out when you learn about an equivalent machine model for CFLs]
  Thus if we show that A' is not a CFL then A cannot be a CFL.

Example 2

Set $A = \{ww | w \in \{a, b\}^*\}$ is not a CFL.

- It suffices to show that set
  $A' = \{a^n b^m a^n b^m | m, n \geq 0\} = A \cap L(a^* b^* a^* b^*)$ is not a CFL:
  $L(a^* b^* a^* b^*)$ is a regular set.
  Fact: Intersection of regular sets and CFLs are CFLs. [Try this out when you learn about an equivalent machine model for CFLs]
  Thus if we show that A' is not a CFL then A cannot be a CFL.
- For a k, pick $z = a^k b^k a^k b^k$.

Example 2

Set $A = \{ww | w \in \{a, b\}^*\}$ is not a CFL.

- It suffices to show that set
  $A' = \{a^n b^m a^n b^m | m, n \geq 0\} = A \cap L(a^* b^* a^* b^*)$ is not a CFL:
  $L(a^* b^* a^* b^*)$ is a regular set.
  Fact: Intersection of regular sets and CFLs are CFLs. [Try this out when you learn about an equivalent machine model for CFLs]
  Thus if we show that A' is not a CFL then A cannot be a CFL.
- For a k, pick $z = a^k b^k a^k b^k$.
- If you pick $i = 2$ then no matter what u, v, w, x, y are chosen with $vx \neq \epsilon$ and $|vwx| \leq k$, you will win.

Non-closure under complement

$\overline{A} = \{a, b\}^* - \{ww | w \in \{a, b\}^*\}$ is a CFL:

- Productions:
  $S \to AB|BA|A|B$
  $A \to CAC|a$
  $B \to CBC|b$
  $C \to a|b$.

Non-closure under complement

$\overline{A} = \{a, b\}^* - \{ww | w \in \{a, b\}^*\}$ is a CFL:

- Productions:
  $S \to AB|BA|A|B$
  $A \to CAC|a$
  $B \to CBC|b$
  $C \to a|b$.
- $S \to A|B$ generate all the odd strings: A generates strings like xay, $|x| = |y|$, B generates strings like ubv, $|u| = |v|$.

Non-closure under complement

$\overline{A} = \{a, b\}^* - \{ww | w \in \{a, b\}^*\}$ is a CFL:

- Productions:
  $S \to AB|BA|A|B$
  $A \to CAC|a$
  $B \to CBC|b$
  $C \to a|b$.
- $S \to A|B$ generate all the odd strings: A generates strings like xay, $|x| = |y|$, B generates strings like ubv, $|u| = |v|$.
- $S \to AB|BA$ generate strings of the form $xayubv$ and $ubvxay$, resp. where $x, y, u, v \in \{a, b\}^*, |x| = |y|, |u| = |v|$.
  Occurrence of a, b at a distance of length$/2$ - cannot be of the form ww.

Non-closure under complement

$\overline{A} = \{a, b\}^* - \{ww | w \in \{a, b\}^*\}$ is a CFL:

- Productions:
  $S \to AB|BA|A|B$
  $A \to CAC|a$
  $B \to CBC|b$
  $C \to a|b$.
- $S \to A|B$ generate all the odd strings: A generates strings like xay, $|x| = |y|$, B generates strings like ubv, $|u| = |v|$.
- $S \to AB|BA$ generate strings of the form $xayubv$ and $ubvxay$, resp. where $x, y, u, v \in \{a, b\}^*, |x| = |y|, |u| = |v|$.
  Occurrence of a, b at a distance of length$/2$ - cannot be of the form ww.
- Any string in $\overline{A}$ must be in one of the above 2 forms.

Other Closure Properties

- Union: $G_1 = (N_1, \Sigma_1, P_1, S_1), G_2 = (N_2, \Sigma_2, P_2, S_2)$.
  $G_1 \cup G_2 = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 | S_2\}, S)$.

Other Closure Properties

- Union: $G_1 = (N_1, \Sigma_1, P_1, S_1), G_2 = (N_2, \Sigma_2, P_2, S_2)$.
  $G_1 \cup G_2 = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 | S_2\}, S)$.
- Concatenation:
  $G_1.G_2 = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\}, S)$.

Other Closure Properties

- Union: $G_1 = (N_1, \Sigma_1, P_1, S_1), G_2 = (N_2, \Sigma_2, P_2, S_2)$.
  $G_1 \cup G_2 = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 | S_2\}, S)$.
- Concatenation:
  $G_1.G_2 = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\}, S)$.
- Intersection: CFLs not closed.
  Eg:
  $\{a^n b^n a^m | n, m \geq 0\} \cap \{a^m b^n a^n | n, m \geq 0\} = \{a^n b^n a^n | n \geq 0\}$.
  (Show that first 2 languages are CFLs and the last is not.)