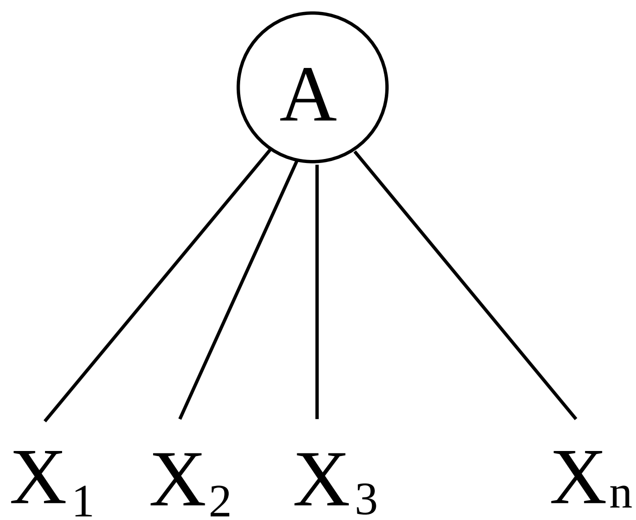
Parse Trees

Data structure of choice to represent programs.

Given $G = (V, T, P, S)$, the **parse tree** for (G, w) is a tree with S at the root, the symbols of w are the leaves (left to right), and each interior node is of the form:



whenever we have a rule $A \longrightarrow X_1 X_2 X_3 \ldots X_n$

Derivation: head $\longrightarrow$ body
Recursive Inference: body $\longrightarrow$ head

The following five are all equivalent:

1. Recursive Inference
2. Derivation
3. Left-most derivation
4. Right-most derivation
5. Yield of a parse tree.

Ambiguity of Grammars

$E \Rightarrow E + E \Rightarrow E + E * E$

$E \Rightarrow E * E \Rightarrow E + E * E$

Two different parse trees! Different meaning.

A grammar is ambiguous if there exists a string w with two different parse trees.

Pushdown Automata (PDA)

ε-NFAs with a stack.

Two (equivalent) versions: (i) accept by final state, (ii) accept by empty stack.

PDAs describe CFLs.

The PDA pushes and pops symbols on the stack; the stack is assumed to be as big as necessary.

Ex. What is a simple PDA for $\{ww^R | w \in \{0,1\}^*\}$?

Formal definition of a PDA:

$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

Q finite set of states

Σ finite input alphabet

Γ finite stack alphabet, $\Sigma \subseteq \Gamma$

$\delta(q, a, X) = \{(p_1, \gamma_1), \ldots, (p_n, \gamma_n)\}$

if $\gamma = \varepsilon$, then the stack is popped, if $\gamma = X$, then the stack is unchanged, if $\gamma = YZ$ then X is replaced Z, and Y is pushed onto the stack

q_0 initial state

Z_0 start symbol

F accepting states