Linguistics 201 – Introduction to Linguistic Theory November 8, 2007
Fall 2007 – Section D

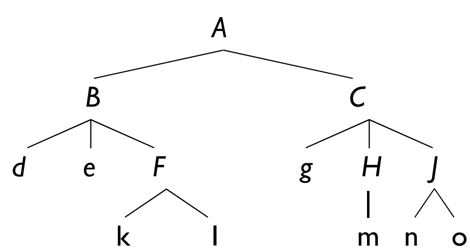
Tree Structures

Tree structures have 2 main ingredients: *nodes* and a *"parent-of"* relation between the nodes. Except for the *root node*, every node has exactly one *parent node*. Nodes that have no *children* are called *terminal nodes*, or *leaves*, ones that do are called *non-terminal nodes*.

Parts of a tree that form itself a tree-structure are called *sub-trees*.

Trees in which every node has at most *n* children are called *n-ary* branching – a tree in which every node has at most two children is called *binary* branching, one with at most three *ternary*, etc.

For this class, we'll sometimes *label* the nodes, that is, we will give them names. We will also fix the order between the terminal nodes.



A simple example:

root note: *A*
terminal nodes: *d, e, k, l, g, m, n, o*
non-terminal nodes: *A, B, C, F, H, J*

A is parent-of *B* and *C*, *B* is parent-of *d*, e, and *F*, *F* is parent of *k* and *l*, etc.

We generalized the parent-of relation, called dominance:

immediate dominance:
A node X *immediately dominates* a node Y if and only if X is the parent of Y.

dominance:
A node X *dominates* a node Y if and only if either
(a) X immediately dominates Y or
(b) there is a node Z such that X dominates Z and Z dominates Y.

Two side notes about the dominance relation:

1) The dominance relation is an example of a *transitive* relation. A relation R is transitive if, whenever x stands in R to y, and y stands in R to z, then x also stands in R to z.

2) The definition of the dominance relation is *recursive*. A recursive definition is one that makes reference to itself. You see that the definition of *dominance* here make reference to itself in clause (b).

Since we fixed the order of the terminal nodes, we can also talk about another relation, called *precedence*:

precedence:
A node X precedes a node Y if and only if X and all of the nodes X dominates appear to the left of Y and all of the nodes Y dominates.

immediate precedence:
A node X immediately precedes Y if there is no node that is preceded by X and that precedes Y.

Some examples from the tree above

- *C* immediately dominates *J*.
- *B* dominates *k*.
- *k* precedes *g*.
- *B* immediately precedes *C*.

- *A* dominates every other node in the tree.

- Two nodes are either in a dominance relation or in a precedence relation, but never both.

Trees and bracket representations

Another way of representing tree structures is by a bracket notation. The tree above would look like this:

$[_A\ [_B\ d\ e\ [_F\ k\ l\ ]]\ [_C\ g\ [_H\ m\ ]\ [_J\ n\ o\ ]]]$

Here are two short recipes to convert back and forth between trees and brackets:

From trees to brackets.

Start at the root node and trace the outline of the tree counter-clockwise.
- If you encounter the left-hand side of a non-terminal node, draw an opening bracket labeled with the name of the node.
- If you encounter the right-hand side of e non-terminal node, draw a closing bracket.
- If you encounter a terminal node, write its name.

Quick checks:
- The number of opening brackets should match the number of closing brackets.
- If you feel unsure, you can label the closing brackets with the node name as well: each opening bracket should correspond to exactly one closing bracket, and what's between the brackets should correspond to the sub-tree starting with the node named like your bracket label.

From brackets to trees.

Go through the brackets from left to right.

- For an opening bracket, draw a new node as a child of your current “working node” with the same label as on the bracket. Let this node be your new working node.
  (“Draw a new node and go there”)
- For a closing bracket, leave your current working node, and let its parent node be your new working node.
  (“Go one up”)
- For any “non-bracket string” draw a new terminal node as a child of your current working node.
  (“Draw a terminal node, but stay where you are.”)