Mathematical and **Logic**al Aspects of Computing (CS304/CS310)

Copyright Niall Madden, http://www.maths.nuigalway.ie/CS304

Lecture 17: Parse Trees

Friday, 1st of November 2012

In Lecture 14 we started studying some quantifier equivalences.

We'll return to that topic presently. But before then we have to formalise our notion of a variable being *free* or *Bound*. We do this with the idea of a *parse tree* (or "parsing tree"), which dates back to the work of Frege. It gives us a new way of representing mathematical and logical expressions, and is particularly useful in denoting free and bound variables.
Parse trees are also important in modelling natural languages, and in processing computer programming languages.

(2/10) Starting with some examples

How should we interpret the following?

- "3 plus 2 times 5".
- "not a and b".
- "for every positive ϵ there is a δ such that if $|x - c| < \delta$, then $|f(x) - f(c)| < \epsilon$.

Using parentheses can clarify this, or we can use a *Parse Tree*.

Examples:

(3/10) Starting with some examples

In this course, we'll follow the example of Chiswell and Hodges (Mathematical Logic 511.3 CHI) by writing the tree with labels to the right of the corresponding vertex. This makes it easy to reconstruct an expression for the Parse Tree.

Example: Write down the parse trees for

- $a \wedge b$;
- $\neg a \vee b$;

(4/10) Trees

Recall... a *graph* $G = (V, E)$ consists of a set of vertices $V = \{v_1, v_2, \ldots, v_n\}$, and edges $E \subset \{(v_i, v_j) | v_i, v_j \in V\}$. **Example:**

A graph is *undirected* if $(v_i, v_j) \in E \rightarrow (v_j, v_i) \in E$.
A *path of length n from vertex p to vertex q* is a sequence of n edges $(p, v_1), (v_1, v_2), \ldots (v_{n-1}, q)$.
If $p = q$, we call this path a circuit (or cycle).

A graph is **connected** if there is a path between each pair of vertices.
Counterexample:

(5/10) Trees

A *tree* is a connected graph, with no circuits.
A *rooted tree* is a *directed* graph that has one vertex designated as the root; all edges are directed away from the root. **Examples:**

Because of this, we usually designate which vertex is the root, and then omit the arrows from a rooted tree.

(6/10) Trees

If there is an edge from vertex p to vertex q, we call p the *parent* and q the *child*. The number of children that a vertex has is called its *arity* (same as "out-degree").

A vertex with no children is called a *leaf*.
Examples:

Now that we have the notion of a tree, we'll present an algorithm for constructing a (parse) tree for expressions in predicate logic .

(7/10) Parse trees for propositional statements

- First recall how we develop a tree for the unary and binary Boolean operators.
- To parse a compound propositional expression, examine the expression to see if it contains
  - a unary operator with an atomic proposition/symbol that can be replaced with a symbol. OR,
  - a binary operator applied to two atomic propositions/symbols that can be replaced with a symbol.
- If so, make these substitutions. Repeat until there is a single operator left.
- Produce the tree for this operator.
- Replace each symbol in turn with its original meaning. Expand these as trees.

(Note: this algorithm is to automate the procedure, and avoid error. You don't have to follow it to obtain the parse tree).

Examples:

(8/10) Parse trees for predicate statements

We can extend the above ideas to parsing formulae in predicate calculus. Functions name are used to label parents, with their arguments labelling the children.

Most importantly: we label parents with qualifiers (similar to how we treated unary operators earlier).

Functions and predicate are treated as symbols.

Examples: Give the parse tree for

1. $\forall x \exists y \big(xy = 1\big)$
2. $\neg \exists x \forall y \big(xy = 1\big)$
3. $\forall x \big(P(x, y) \rightarrow \neg F(x)\big)$
4. $(x = y) \rightarrow \forall y \left(\exists z \big(x = F(z, c, w)\big) \wedge P(y, z) \right)$

(Note – this last example is taken from Chiswell and Hodges, *Mathematical Logic*, Example 7.2.2).

(9/10) Parse trees for predicate statements

(10/10) Parse trees: free and bound variables

If a vertex label with a particular variable has an ancestor (parent, grandparent, etc) labeled with a quantifier applied to that variable, then that occurrence is **bound**. Otherwise it is **Free**.

Let's return again to the example:

$$(x = y) \rightarrow \forall y \left(\exists z \big(x = F(z, c, w) \big) \wedge P(y, z) \right)$$