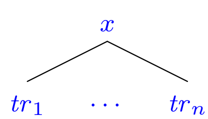
Section 1.3: Trees and Inductive Definitions

In this section, we will introduce and study ordered trees of arbitrary (finite) arity whose nodes are labeled by elements of some set. The definition of the set of such trees will be our first example of an inductive definition. In later chapters, we will define regular expressions (in Chapter 3) and parse trees (in Chapter 4) as restrictions of the trees we consider here.

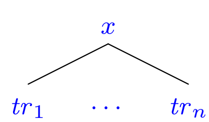
(1.3) The Set $\mathbf{Tree}_X$ of X-Trees

Suppose X is a set. The set $\mathbf{Tree}_X$ of X-*trees* is the least set such that, (†) for all $x \in X$, $n \in \mathbb{N}$ and $tr_1, \ldots, tr_n \in \mathbf{Tree}_X$,



$\in \mathbf{Tree}_X$.

The *root label* of the tree



is x, and tr_1 is the tree's first *child*, etc.

(1.3) On the Definition of $\mathbf{Tree}_X$

When we say that $\mathbf{Tree}_X$ is the "least" set satisfying property (†), we mean least with respect to $\subseteq$. I.e., we are saying that $\mathbf{Tree}_X$ is the unique set such that:

- $\mathbf{Tree}_X$ satisfies property (†); and
- if A is a set satisfying property (†), then $\mathbf{Tree}_X \subseteq A$.

In other words, $\mathbf{Tree}_X$ satisfies (†) and doesn't contain any extraneous elements. $\mathbf{Tree}_X$ consists of precisely those values that can be constructed in some number of steps using (†).

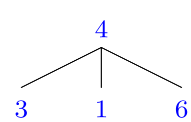
The definition of $\mathbf{Tree}_X$ is our first example of an *inductive definition*, a definition in which we collect together all of the values that can be constructed using some set of rules.

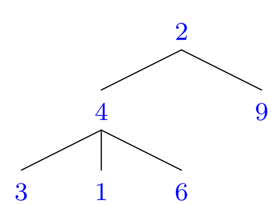
(1.3) Example $\mathbb{N}$-Trees

Here are some example elements of $\mathbf{Tree}_{\mathbb{N}}$:

- (remember that n can be 0)

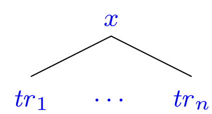
$$3$$

- 

- 

(1.3) Linear Notation for Trees

We sometimes use linear notation for trees, writing an X-tree



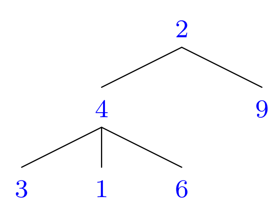
as

$$x(tr_1, \ldots, tr_n).$$

We often abbreviate $x()$ (the childless tree whose root label is x) to x.

(1.3) Linear Notation for Trees (Cont.)

For example, we can write the $\mathbb{N}$-tree



as $2(4(3, 1, 6), 9)$.

(1.3) Induction Principle for $\mathbf{Tree}_X$

Every inductive definition gives rise to an induction principle, and the definition of $\mathbf{Tree}_X$ is no exception.

The *induction principle for* $\mathbf{Tree}_X$ says that

for all $tr \in \mathbf{Tree}_X$, $P(tr)$

follows from showing

for all $x \in X$, $n \in \mathbb{N}$ and $tr_1, \ldots, tr_n \in \mathbf{Tree}_X$,
if $P(tr_1), \ldots, P(tr_n)$,
then $P(x(tr_1, \ldots, tr_n))$.

(1.3) Paths in Trees

When we draw a tree, we can point at a position in the drawing and call it a *node*. The formal analogue of this graphical notion is called a path.

The set $\mathbf{Path}$ of *paths* is the least set such that

- $\mathbf{nil} \in \mathbf{Path}$;
- For all $n \in \mathbb{N}$ and pat in $\mathbf{Path}$, $n \to pat \in \mathbf{Path}$.

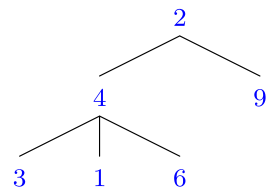
A path

$$n_1 \to \cdots \to n_l \to \mathbf{nil},$$

consists of directions to a node in the drawing of a tree: one starts at the *root node* of a tree, goes from there to the n_1'th child, ..., goes from there to the n_l'th child, and then stops.

(1.3) Example Paths and Corresponding Nodes

Some examples of paths and corresponding nodes for the $\mathbb{N}$-tree



are:

- $\mathbf{nil}$ corresponds to the node labeled 2;
- $1 \to \mathbf{nil}$ corresponds to the node labeled 4;
- $1 \to 2 \to \mathbf{nil}$ corresponds to the node labeled 1.

(1.3) More on Paths and Trees

We consider a path pat to be *valid* for a tree tr iff following the directions of pat never causes us to try to select a nonexistent child. E.g., the path $1 \to 2 \to nil$ isn't valid for the tree $6(7(8))$, since the tree $7(8)$ lacks a second child.

As usual, if the sub-tree at position pat in tr has no children, then we call the sub-tree's root node a *leaf* or *external node*; otherwise, the sub-tree's root node is called an *internal node*.

Note that we can form a tree tr' from a tree tr by replacing the sub-tree at position pat in tr by a tree tr''.

We define the *size* of an X-tree tr to be the number of elements of

$$\{\, pat \mid pat \text{ is a valid path for } tr \,\}.$$

(1.3) Measuring Paths and Trees

The *length* of a path pat ($|pat|$) is defined recursively by:

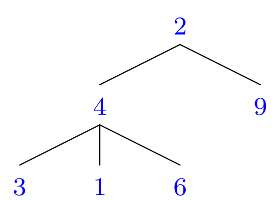
$$|\mathbf{nil}| = 0,$$
$$|n \rightarrow pat| = 1 + |pat| \text{ for all } n \in \mathbb{N} \text{ and } pat \in \mathbf{Path}.$$

Given this definition, we can define the *height* of an X-tree tr to be the largest element of

$$\{\, |pat| \mid pat \text{ is a valid path for } tr \,\}.$$

(1.3) Measuring Paths and Trees (Cont.)

For example, the tree



has:

- size 6, since exactly six paths are valid for this tree; and
- height 2, since the path $1 \rightarrow 1 \rightarrow \mathbf{nil}$ is valid for this tree and has length 2, and there are no paths of greater length that are valid for this tree.