

Section 3.2 Recursively Defined Functions and Procedures

A function f is recursively defined if at least one value $f(x)$ is defined in terms of another value $f(y)$, where $x \neq y$. Similarly, a procedure P is recursively defined if the action of $P(x)$ is defined in terms of another action $P(y)$, where $x \neq y$.

Technique for recursive definitions when the argument domain is inductively defined.

1. Specify a value $f(x)$, or action $P(x)$, for each basis element x of S .
2. Specify rules that, for each inductively defined element x in S , define the value $f(x)$, or action $P(x)$, in terms of previously defined values of f , or actions of P .

Example 1. Find a recursive definition for the function $f : \mathbf{N} \rightarrow \mathbf{N}$ defined by

$$f(n) = 0 + 3 + 6 + \dots + 3n.$$

Solution: Notice that \mathbf{N} is an inductively defined set: $0 \in \mathbf{N}$; $n \in \mathbf{N}$ implies $n + 1 \in \mathbf{N}$. So we need to give $f(0)$ a value in \mathbf{N} and we need to define $f(n + 1)$ in terms of $f(n)$. The given definition of f tells us to set $f(0) = 0$. To discover a definition for $f(n + 1)$ we can write

$$\begin{aligned} f(n + 1) &= (0 + 3 + 6 + \dots + 3n) + 3(n + 1) \\ &= f(n) + 3(n + 1). \end{aligned}$$

So we have a recursive definition for f :

$$f(0) = 0$$

$$f(n + 1) = f(n) + 3(n + 1).$$

Two alternative definitions:

- $f(0) = 0$
 $f(n) = f(n - 1) + 3n$ ($n > 0$).
- (*if-then-else* form): $f(n) = \text{if } n = 0 \text{ then } 0 \text{ else } f(n - 1) + 3n.$

Example 2: Find a recursive definition for $\text{cat} : A^* \times A^* \rightarrow A^*$ defined by $\text{cat}(s, t) = st$.

Solution: Notice that A^* is inductively defined: $\Lambda \in A^*$; $a \in A$ and $x \in A^*$ imply $ax \in A^*$, where ax denotes the string version of cons. We can define cat recursively using the first argument. The definition of cat gives $\text{cat}(\Lambda, t) = \Lambda t = t$. For the recursive part we can write $\mathbf{cat}(ax, t) = axt = a(xt) = a\mathbf{cat}(x, t)$. So we have a definition:

$$\mathbf{cat}(\Lambda, t) = t$$

$$\mathbf{cat}(ax, t) = a\mathbf{cat}(x, t).$$

If-then-else form using head and tail for strings:

$$\mathbf{cat}(s, t) = \text{if } s = \Lambda \text{ then } t \text{ else } \mathbf{head}(s)\mathbf{cat}(\mathbf{tail}(s), t).$$

Example 3: Find a definition for $f : \text{lists}(\mathbf{Q}) \rightarrow \mathbf{Q}$ defined by $f(\langle x_1, \dots, x_n \rangle) = x_1 + \dots + x_n$.

Solution: The set $\text{lists}(\mathbf{Q})$ is inductively defined:

$$\langle \rangle \in \text{lists}(\mathbf{Q}); \quad h \in \mathbf{Q} \text{ and } t \in \text{lists}(\mathbf{Q}) \text{ imply } h :: t \in \text{lists}(\mathbf{Q}).$$

To discover a recursive definition, we can use the definition of f as follows:

$$\begin{aligned} f(\langle x_1, \dots, x_n \rangle) &= x_1 + \dots + x_n \\ &= x_1 + (x_2 + \dots + x_n) \\ &= x_1 + f(\langle x_2, \dots, x_n \rangle) \\ &= \mathbf{head}(\langle x_1, \dots, x_n \rangle) + f(\mathbf{tail}(\langle x_1, \dots, x_n \rangle)). \end{aligned}$$

So if we let $f(\langle \rangle) = 0$, we have a recursive definition:

$$f(\langle \rangle) = 0$$

$$f(h :: t) = h + f(t).$$

If-then-else form: $f(L) = \text{if } L = \langle \rangle \text{ then } 0 \text{ else } \mathbf{head}(L) + f(\mathbf{tail}(L)).$

Example 4: Given $f : \mathbf{N} \rightarrow \mathbf{N}$ defined recursively by

$$f(0) = 0$$

$$f(1) = 0$$

$$f(x + 2) = 1 + f(x).$$

The *if-then-else* form for f can be written as follows:

$$f(x) = \text{if } x = 0 \text{ or } x = 1 \text{ then } 0 \text{ else } 1 + f(x - 2).$$

What does f do?

Answer: List a few values to get the idea. For example,

$$\text{map}(f, \langle 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \rangle) = \langle 0, 0, 1, 1, 2, 2, 3, 3, 4, 4 \rangle .$$

So $f(x)$ returns the floor of $x/2$. i.e., $f(x) = \lfloor x/2 \rfloor$.

Example 5: Find a recursive definition for $f : \text{lists}(\mathbf{Q}) \rightarrow \mathbf{Q}$ defined by

$$f(\langle x_1, \dots, x_n \rangle) = x_1x_2 + x_2x_3 + \dots + x_{n-1}x_n.$$

Solution: Let $f(\langle \rangle) = 0$ and $f(\langle x \rangle) = 0$. Then for $n \geq 2$ we can write

$$f(\langle x_1, \dots, x_n \rangle) = x_1x_2 + (x_2x_3 + \dots + x_{n-1}x_n) = x_1x_2 + f(\langle x_2, \dots, x_n \rangle).$$

So we have the following recursive definition.

$$f(\langle \rangle) = 0$$

$$f(\langle x \rangle) = 0$$

$$f(h :: t) = h \cdot \text{head}(t) + f(t).$$

If-then-else form:

$$f(L) = \text{if } L = \langle \rangle \text{ or } \text{tail}(L) = \langle \rangle \text{ then } 0 \text{ else } \text{head}(L) \cdot \text{head}(\text{tail}(L)) + f(\text{tail}(L)).$$

Example 6: Find a recursive definition for $\text{isin} : A \times \text{lists}(A) \rightarrow \{\text{true}, \text{false}\}$ where $\text{isin}(x, L)$ means that x is in the list L .

Solution:

$$\text{isin}(x, \langle \rangle) = \text{false}$$
$$\text{isin}(x, x :: t) = \text{true}$$
$$\text{isin}(x, h :: t) = \text{isin}(x, t).$$

If-then-else form:

$$\text{isin}(x, L) = \text{if } L = \langle \rangle \text{ then false else if } x = \text{head}(L) \text{ then true else } \text{isin}(x, \text{tail}(L)).$$

Example 7: Find a recursive definition for $\text{sub} : \text{lists}(A) \times \text{lists}(A) \rightarrow \{\text{true}, \text{false}\}$ where $\text{sub}(L, M)$ means the elements of L are elements of M .

Solution:

$$\text{sub}(\langle \rangle, M) = \text{true}$$
$$\text{sub}(h :: t, M) = \text{if } \text{isin}(h, M) \text{ then } \text{sub}(t, M) \text{ else false.}$$

If-then-else form:

$$\text{sub}(L, M) = \text{if } L = \langle \rangle \text{ then true else if } \text{isin}(\text{head}(L), M) \text{ then } \text{sub}(\text{tail}(L), M) \text{ else false.}$$

Example 8: Find a recursive definition for $\text{intree} : \mathbf{Q} \times \text{binSearchTrees}(\mathbf{Q}) \rightarrow \{\text{true}, \text{false}\}$ where $\text{intree}(x, T)$ means x is in the binary search tree T .

Solution:

$$\text{intree}(x, \langle \rangle) = \text{false}$$
$$\text{intree}(x, \text{tree}(L, x, R)) = \text{true}$$
$$\text{intree}(x, \text{tree}(L, y, R)) = \text{if } x < y \text{ then } \text{intree}(x, L) \text{ else } \text{intree}(x, R).$$

If-then-else form:

$$\text{intree}(x, T) = \text{if } T = \langle \rangle \text{ then false}$$
$$\text{else if } x = \text{root}(T) \text{ then true}$$
$$\text{else if } x < \text{root}(T) \text{ then } \text{intree}(x, \text{left}(T))$$
$$\text{else } \text{intree}(x, \text{right}(T)).$$

Traversing Binary Trees

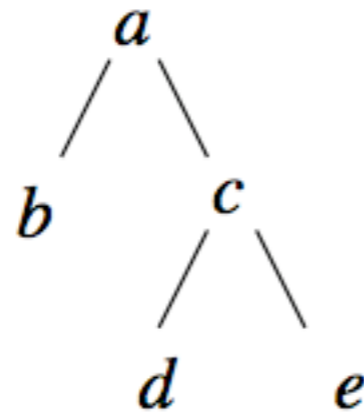
The three standard procedures to traverse a binary tree are defined recursively as follows:

preorder(T): *if* T \neq $\langle \rangle$ then visit root(T); preorder(left(T)); preorder(right(T)) *fi*.

inorder(T): *if* T \neq $\langle \rangle$ then inorder(left(T)); visit root(T); inorder(right(T)) *fi*

postorder(T): *if* T \neq $\langle \rangle$ then postorder(left(T)); postorder(right(T)); visit root(T) *fi*.

Example 9: Traverse the following tree in each of the three orders.



Solution:

Preorder: a b c d e

Inorder: b a d c e

Postorder: b d e c a

Example 10: Find a recursive definition for $\text{post} : \text{binaryTrees}(A) \rightarrow \text{lists}(A)$ where $\text{post}(T)$ is the list of nodes from a postorder traversal of T.

Solution: $\text{post}(\langle \rangle) = \langle \rangle$

$$\text{post}(\text{tree}(L, x, R)) = \text{cat}(\text{post}(L), \text{cat}(\text{post}(R), \langle x \rangle))$$

where cat concatenates two lists and can be defined by,

$$\text{cat}(\langle \rangle, L) = L$$
$$\text{cat}(h :: t, L) = h :: \text{cat}(t, L).$$

Example 11: Find a recursive definition for $f : \text{binaryTrees}(\mathbf{Q}) \rightarrow \mathbf{Q}$ where $f(T)$ is the sum of the nodes in T .

Solution: $f(\langle \rangle) = 0$

$$f(\text{tree}(L, x, R)) = x + f(L) + f(R).$$

Infinite Sequences

We can construct recursive definitions for infinite sequences by defining a value $f(x)$ in terms of x and $f(y)$ for some value y in the sequence.

Example 12. Suppose we want to represent the infinite sequence $f(x) = \langle x, x^2, x^4, x^8, \dots \rangle$

Solution: Use the definition to discover a solution as follows:

$$f(x) = \langle x, x^2, x^4, x^8, \dots \rangle = x :: \langle x^2, x^4, x^8, \dots \rangle = x :: f(x^2).$$

So define $f(x) = x :: f(x^2)$.

Example 13: What sequence is defined by $g(x, k) = x^k :: g(x, k + 1)$?

Answer: $g(x, k) = x^k :: g(x, k + 1) = x^k :: x^{k+1} :: g(x, k + 2) = \dots = \langle x^k, x^{k+1}, x^{k+2}, \dots \rangle$.

Example 14: How do we obtain the sequence $\langle x, x^3, x^5, x^7, \dots \rangle$?

A Solution: Define $f(x) = h(x, 1)$, where $h(x, k) = x^k :: h(x, k + 2)$.

Example 15: How do we obtain the sequence $\langle 1, x^2, x^4, x^6, x^8, \dots \rangle$?

A Solution: Use $h(x, 0)$ from Example 14.