A generalization of the functions head and tail.

by Edsger W. Dijkstra and C. S. Scholten

We consider sequences defined as structures on the natural coordinate x. Let S be such a sequence. The functions h (= head) and t (= tail) are defined by

$$h.S = S_0^x \qquad \text{and} \qquad t.S = S_{1+x}^x \quad .$$

Note that $h.S$ is an "element" – viz. the "leading" one – and $t.S$ is again a sequence – viz. "the rest" – . (With the $:$ for concatenation – as, for instance in SASL – we have the identity $S = h.S : t.S$.)

There are several ways of expressing S' sub S, i.e. that S' is a postfix of S :

$$S' \ \underline{\text{sub}}\ S \ \equiv \ (\underline{E}n{:}\ n\geq 0{:}\ \ S' = t^n.S) \qquad \text{or}$$

$$S' \ \underline{\text{sub}}\ S \ \equiv \ (\underline{E}n{:}\ n\geq 0{:}\ \ S' = S_{n+x}^x) \quad .$$

We prefer the latter one. Representing the natural number n by a string of n zeros, and hence addition by juxtaposition, we would get

$$S' \ \underline{\text{sub}}\ S \ \equiv \ (\underline{E}n{:}\ n\in\{0\}^*{:}\ \ S' = S_{nx}^x) \quad .$$

The above is extended to tuples of sequences. Illustrating it for two we thus define

$$(S',T') \ \underline{\text{sub}}\ (S,T) \equiv (\underline{E}n{:}\ n\in\{0\}^*{:}\ \ (S',T') = (S,T)_{nx}^x) \ .$$

Substitution being defined to distribute over pair forming.

the quantified expression may be rewritten as

$$(S', T') = (S^x_{nx}, T^x_{nx}) \qquad ;$$

with element-wise application of equality this yields

$$S' = S^x_{nx} \;\wedge\; T' = T^x_{nx} \qquad .$$

To complete the understanding of the above we define for sequences S and T equality by

$$S = T \;\equiv\; h.S = h.T \;\wedge\; t.S = t.T \qquad .$$

We mention without proof

$$S = T \;\equiv (\underline{A}\ (S', T') : (S', T')\ \underline{\mathrm{sub}}\ (S, T) : \ h.S' = h.T') \quad .$$

(The proof is left as an exercise for the authors.)

* * *

A sequence is a special instance of a rooted tree with constant fan-out, viz. with fan-out = 1, in exactly the same way as {0} is a special case of a finite alphabet. In the following, C stands for an alphabet of m characters, our tuples will be m-tuples and our trees trees with constant fan-out = m .

We now consider a tree as a structure defined on a coordinate x ranging over C^* . Let S be such a tree. The function head has its obvious analogue: it is known under the name root , and we shall denote it by r and define it by

$$r.S = S^x_{<>}$$

in which <> denotes the empty string.

The corresponding notion <u>sub</u>, however, poses a problem. Do we define

$$S' \; \underline{sub} \; S \equiv (\underline{E} n : n \in C^* : S' = S^x_{nx})$$

or

$$S' \; \underline{sub} \; S \equiv (\underline{E} n : n \in C^* : S' = S^x_{xn}) \qquad ?$$

<u>Note</u> In either case we have the theorem – mentioned without proof – that for trees S and T

$$S = T \equiv (\underline{A} S', T' : (S', T') \; \underline{sub} \; (S, T) : r.S' = r.T') \quad .$$

(End of Note.)

In this stage we have no grounds for preferring the one <u>sub</u> over the other. (For a single character alphabet, the two definitions coincide.)

For $m \geq 2$, we have two different ways of defining under control of a parameter c , $c \in C^*$, a new tree in terms of a given one:

$$c \; \underline{ex} \; S = S^x_{cx}$$

$$S \; \underline{ex} \; c = S^x_{xc} \quad .$$

Here we have used the same operator <u>ex</u> as asymmetric infix operator between a tree and an element of C^* .

With $b \in C^*$ and $c \in C^*$ we then have

$$c \ \underline{ex}\ (b \ \underline{ex}\ S) = (bc) \ \underline{ex}\ S$$

$$(S \ \underline{ex}\ b) \ \underline{ex}\ c = S \ \underline{ex}\ (cb) \quad ;$$

note that on account of the types of b, c, and S, the parentheses in the left-hand sides of the above could have been omitted.

Furthermore, we have

$$b \ \underline{ex}\ (S \ \underline{ex}\ c) = (b \ \underline{ex}\ S) \ \underline{ex}\ c \quad ,$$

both sides being equal to $S^*_{b \times c}$. Consequently, also here the parentheses may be omitted. We conclude that the "continued" ex of which 1 operand is a tree while the others are from C^* needs no parentheses.

Finally, note that $<> \underline{ex}$ and $\underline{ex} <>$ are identity operators. Note also

$$r.(c \ \underline{ex}\ S) = r.(S \ \underline{ex}\ c) \quad .$$

So much for the general ex .

* * *

Of special interest is the use of ex with the string operand of length 1 . Let e be a parameter ranging over the strings of length 1 in C^* or — if we don't distinguish between one-element strings and elements — ranging over C ; e has m distinct possible values. For such e,

e ex S is called "son tree nr. e of S" and
S ex e is called "daughter tree nr. e of S".

These are the closest analogue of the function tail: firstly it has an additional parameter ranging over C, secondly there is the distinction between sons and daughters. The latter distinction gives us two alternative recursive definitions for the equality of two trees S and T:

$$S = T \equiv r.S = r.T \wedge (\underline{A}e : e \in C : e \ \underline{ex}\ S = e \ \underline{ex}\ T)$$
$$S = T \equiv r.S = r.T \wedge (\underline{A}e : e \in C : S \ \underline{ex}\ e = T \ \underline{ex}\ e) \quad .$$

After ex we turn our attention to a number of unary operators, to begin with some that form a new tree from a given one.

Consider the function rev on strings, with $b \in C^*$, $c \in C^*$, and $e \in C$ given by

$$\text{rev}.\langle\rangle = \langle\rangle$$
$$\text{rev}.e = e$$
$$\text{rev}.(bc) = (\text{rev}.c)(\text{rev}.b) \quad .$$

In terms of rev we now define the "transpose"

$$S^T = S^x_{\text{rev}.x} \quad .$$

Since $\text{rev}.(\text{rev}.x) = x$, $(S^T)^T = S$. The connection between the transpose and ex is given by

$(b \ \underline{ex} \ S)^T = S^T \ \underline{ex} \ (\text{rev}.b)$, and in particular

$(e \ \underline{ex} \ S)^T = S^T \ \underline{ex} \ e$.

For our purposes, the transpose is not a very important operator; it has been mentioned because it illustrates an underlying duality so nicely.

For the sake of completeness we also mention ROT defined by

$$\text{ROT}.S = S^x_{\text{rot}.x}$$

where $\text{rot}.\langle\rangle = \langle\rangle$
$\text{rot}.(e\,b) = b e$.

This is a function in which we are even less interested than in the transpose. This is because our interest in such infinite trees, i.e. functions on C^*, stems from considerations about recursion, which relate elements of C^* with, say, a common prefix, a relation which is completely destroyed by rot. (So we hardly take the trouble to observe

$$e \ \underline{ex} \ (\text{ROT}.S) = S \ \underline{ex} \ e \quad .)$$

Now we return to $e \ \underline{ex} \ S$; it is again a tree, comprising, so to speak, 1/m-th of the elements of S minus its root. Let now e range over C; the combined elements of the resulting m trees comprise all the elements of S exept the root: it can be viewed, therefore, as a function on C^+, i.e. all non-empty finite strings of C^*. We can denote

the aggregate of the son trees of S by s.S
and