Science of Computer Programming 6 (1986) 207-211
North-Holland

SMALL PROGRAMMING EXERCISES 10

M. REM

Department of Mathematics and Computing Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

Our two new exercises are graph problems. In the first one the Strahler number of a given binary tree has to be computed. It is a nice little exercise allowing a solution that is linear in the size of the tree.

The other exercise involves an acyclic directed graph. Such a graph has sources, i.e. vertices without incoming arcs. We have to determine all vertices that are at least a given 'distance' removed from the sources. An unexpected property of this exercise is that although the arcs have weights attached to them, we can still find a solution that is linear in the number of arcs and vertices of the graph.

Exercise 25: Strahler number of a binary tree

A *binary tree* T is either empty or it consists of a vertex, the *root* of T, and two subtrees T_0 and T_1, each of which is again a binary tree. The *Strahler number* $\sigma(T)$ of a binary tree T is defined as follows. $\sigma(T)=0$ if T is empty. If T has subtrees T_0 and T_1 the Strahler number is given by

$$\sigma(T)=\begin{cases}\sigma(T_0)\ \mathbf{max}\ \sigma(T_1) & \text{if } \sigma(T_0)\neq\sigma(T_1),\\ \sigma(T_0)+1 & \text{if } \sigma(T_0)=\sigma(T_1).\end{cases}$$

We have to compute $\sigma(T)$ of a given binary tree T of N vertices, $N\geqslant 1$. The vertices are numbered 0 through $N-1$. Vertex 0 is the root of T. As in Exercise 17, the tree is recorded in an integer array $v(i\colon 1\leqslant i<N)$:

$$(\mathbf{A}i\colon 1\leqslant i<N\colon \text{the tree with vertex } v(i) \text{ as its root has a subtree with vertex } i \text{ as its root})$$

The functional specification is

$|[N\colon int; \{N\geqslant 1\}$

$\quad v(i\colon 1\leqslant i<N)\colon \mathbf{array\ of}\ int;$

$\quad \{v \text{ represents binary tree } T\}$

$\quad |[p\colon int;$

$\qquad S$

$\qquad \{p=\sigma(T)\}$

$\quad]|$

$]|$

Exercise 26: K-sequel of an acyclic digraph

Given is an acyclic directed graph G each arc of which has a positive weight. The weight of a path is the sum of the weights of its arcs. A vertex without predecessors is called a *source*. For each $K \geq 1$ the K-sequel of G is defined as the set of all vertices j for which the weight of each path from a source to j is at least K.

Graph G is represented in arrays b and e in the usual way. The weights are given by an array $w(i: 0 \leq i < M)$: the arc from vertex j to vertex $e(i)$, $b(j) \leq i < b(j+1)$, has weight $w(i)$.

We are requested to find a statement list S such that

$|[N, M, K: int; \{N \geq 1 \wedge M \geq 0 \wedge K \geq 1\}$

  $b(j: 0 \leq j \leq N)$: **array of** *int*;

  $e, w(i: 0 \leq i < M)$: **array of** *int*;

  $\{\textsc{suc}(G, b, e) \wedge G \text{ acyclic} \wedge (\mathbf{A}i: 0 \leq i < M: w(i) \geq 1)\}$

  $|[a(j: 0 \leq j < N)$: **array of** *bool*;

    S

    $\{(\mathbf{A}j: 0 \leq j < N: a(j) \equiv (j \text{ in the } K\text{-sequel of } G))\}$

  $]|$

$]|$

Solution of Exercise 23 (problem of the masks)

A $P(i: 0 \leq i < M)$-mask in $X(j: 0 \leq j < N)$ is an increasing integer sequence $r(i: 0 \leq i < M)$ that satisfies

$$(\mathbf{A}i: 0 \leq i < M: 0 \leq r(i) < N \wedge P(i) = X(r(i))).$$

We have to determine S such that

$|[M, N: int; \{M \geq 1 \wedge N \geq 0\}$

  $P(i: 0 \leq i < M), X(j: 0 \leq j < N)$: **array of** *int*;

  $\{(\mathbf{A}i: 0 \leq i < M: (\mathbf{N}h: 0 \leq h < M: P(h) = i) = 1)\}$

  $|[a: int;$

    S

    $\{a = (\text{number of } P(i: 0 \leq i < M)\text{-masks in } X(j: 0 \leq j < N))\}$

  $]|$

$]|$

If we replace in the postcondition the constant N by a variable n we get the following invariant:

$$a = (\text{number of } P(i\colon 0 \leqslant i < M)\text{-masks in } X(j\colon 0 \leqslant j < n))$$
$$\wedge\, 0 \leqslant n \leqslant N.$$

It can be initialized with $n, a = 0, 0$. In order to determine the number of $P(i\colon 0 \leqslant i < M)$-masks in $X(j\colon 0 \leqslant j < n+1)$ we need to know the number of $P(i\colon 0 \leqslant i < M-1)$-masks in $X(j\colon 0 \leqslant j < n)$, since variable a has to be increased by that number if $P(M-1) = X(n)$. Consequently, rather than a single count of masks we need a whole array $b(h\colon 0 \leqslant h \leqslant M)$ of them. The invariant then becomes

$$P\colon \quad (\mathbf{A}h\colon 0 \leqslant h \leqslant M\colon b(h) = B(h, n))$$
$$\wedge\, 0 \leqslant n \leqslant N$$

in which $B(h, n)$ denotes the number of $P(i\colon 0 \leqslant i < h)$-masks in $X(j\colon 0 \leqslant j < n)$.

The proper initialization for $n = 0$ and the way in which array b should be changed when increasing the value of n follow directly from the recurrence relation for $B(h, n)$:

$$B(0, n) = 1,$$
$$B(h, 0) = 0 \quad \text{for } h \geqslant 1$$

and for $h \geqslant 0$ and $n \geqslant 0$,

$$B(h+1, n+1) = \begin{cases} B(h+1, n) + B(h, n) & \text{if } X(n) = P(h), \\ B(h+1, n) & \text{if } X(n) \neq P(h). \end{cases}$$

If $0 \leqslant X(n) < M$ there exists one h such that $X(n) = P(h)$. In order to determine that h we need the inverse of P. To that end, we introduce an integer array $q(h\colon 0 \leqslant h < M)$ and establish

$$(\mathbf{A}h\colon 0 \leqslant h < M\colon q(P(h)) = h)$$

The solution is now straightforward.

S: $\quad |[n\colon \mathit{int};$

$\quad q(h\colon 0 \leqslant i < M), b(h\colon 0 \leqslant h \leqslant M)$: **array of** int;

$\quad b\colon (0) = 1$

$; |[h\colon \mathit{int};\ h := 0$

$\quad ;\ $ **do** $h \neq M \rightarrow q\colon (P(h)) = h\ ;\ h := h+1\ ;\ b\colon (h) = 0$ **od**

$\quad]|$

$; n := 0$

$$
\begin{array}{l}
\quad\quad \mathbf{do}\ n \neq N \\
\quad\quad \rightarrow \mathbf{if}\ 0 \leqslant X(n) \wedge X(n) < M \\
\quad\quad\quad \rightarrow |[h\colon int;\ h := q(X(n))\ ;\ b\colon (h+1) = b(h+1) + b(h)]| \\
\quad\quad\quad \square\ 0 > X(n) \vee X(n) \geqslant M \\
\quad\quad\quad \rightarrow \mathbf{skip} \\
\quad\quad\quad \mathbf{fi} \\
\quad\quad ;\ \ n := n+1 \\
\quad\quad \mathbf{od} \\
\quad ;\ \ a := b(M) \\
\quad]|
\end{array}
$$

Our solution has a computation time that is linear in M and N.

Solution of Exercise 24 (recognizing h-sequences)

An h-sequence is a sequence of zeros and ones generated by the grammar

$$\langle h\text{-}seq\rangle ::= 0 \mid 1\langle h\text{-}seq\rangle\langle h\text{-}seq\rangle.$$

We have to solve S in

$$
\begin{array}{l}
|[N\colon int;\ \{N \geqslant 0\} \\
\quad H(i\colon 0 \leqslant i < N)\colon \mathbf{array\ of}\ int; \\
\quad \{(\mathbf{A}i\colon 0 \leqslant i < N\colon H(i) = 0 \vee H(i) = 1)\} \\
\quad |[b\colon bool; \\
\quad\quad S \\
\quad\quad \{b \equiv (H(i\colon 0 \leqslant i < N)\ \text{is an}\ h\text{-sequence})\} \\
\quad]| \\
]|
\end{array}
$$

In the second part of the grammar a concatenation of two h-sequences occurs, so we should be looking at the problem of recognizing concatentations of h-sequences. Since a proper prefix of an h-sequence is not an h-sequence, a sequence of zeros and ones can in at most one way be partitioned into a concatenation of h-sequences.

A 0 by itself is an h-sequence. Consequently, a sequence of zeros and ones that starts with a 0 is a concatenation of m, $m \geqslant 1$, h-sequences if and only if the rest of the sequence is a concatenation of $m-1$ h-sequences. A 1 followed by two

h-sequences is an h-sequence. Consequently, a sequence of zeros and ones that starts with a 1 is a concatenation of m, $m \geq 1$, h-sequences if and only if the rest of the sequence is a concatenation of $m+1$ h-sequences.

The program follows immediately from these two observations. Its invariant is

$H(i: 0 \leq i < n)$ followed by m h-sequences is an h-sequence

$\wedge\, 0 \leq n \leq N \wedge m \geq 0.$

S: $|[m, n\colon int;\ m, n := 1, 0$

; **do** $m \neq 0 \wedge n \neq N$

→ **if** $H(n) = 0 \rightarrow m := m - 1$

□ $H(n) = 1 \rightarrow m := m + 1$

fi

; $n := n + 1$

od

; $b := (m = 0 \wedge n = N)$

$]|$

The program has an execution time that is proportional to N.