A programming exercise communicated by Oege de Moor

On March 24 and 25, 2003, a symposium titled "The Fun of Programming" was held in Oxford UK, as a tribute to Richard S. Bird on the occasion of his 60th birthday. During that symposium Oege de Moor dealt with the following programming problem:

Given array $f[0..N)$, $1 \le N$, of integers, compute the maximum sum of an "atiguous" subsequence of f, where "atiguous" stands for "containing no pair of neighbouring elements of f".

In Oege's case the array was a list, and he dealt with the exercise through his theory of fusion. Here we tackle the problem in terms of our "oldfashioned" method of (imperative) programming. The exercise itself is worth of being recorded.

* * *

The problem as stated contains a little smoke screen, viz. by its mentioning of the notion "subsequence". Since in computing the sum of a subsequence the order of the elements in the subsequence is totally irrelevant, we had better focus on the set of indices in f making up for that subsequence. Therefore we define, for $0 \le n \le N$:

$$G.n \;=\; \langle \uparrow S \colon S \subseteq [0..n) \wedge nn.S : \Sigma.S \rangle$$

in which $\qquad \Sigma.S \;=\; \langle \Sigma i \colon i \in S \colon f.i \rangle$

$$nn.S \;\equiv\; \langle \forall i \colon i \in S \colon i+1 \notin S \rangle ,$$

in terms of which the required answer is G.N. (Predicate nn captures the desired "atiguity".)

* * *

Before doing anything else, we first try to derive a recurrence relation for G. We venture a simple one:

$$
\begin{array}{ll}
 & G.(n+1) \\
= & \quad \{\text{definition of } G\} \\
 & \langle \uparrow S \colon S \subseteq [0..n+1) \wedge nn.S : \Sigma.S \rangle \\
= & \quad \{ n \notin S \vee n \in S, \text{ range splitting} \} \\
 & \langle \uparrow S \colon S \subseteq [0..n+1) \wedge n \notin S \wedge nn.S : \Sigma.S \rangle \\
 & \uparrow \\
 & \langle \uparrow S \colon S \subseteq [0..n+1) \wedge n \in S \wedge nn.S : \Sigma.S \rangle \\
= & \quad \{ \text{simplification of the first term,} \\
 & \quad \ \text{dummy transformation } S := T + \{n\} \\
 & \quad \ \text{with } T \subseteq [0..n) \text{ on the second term} \}
\end{array}
$$

$$\langle \uparrow S : S \subseteq [0..n) \wedge nn.S : \Sigma.S \rangle$$
$$\uparrow$$
$$\langle \uparrow T : T \subseteq [0..n) \wedge nn\,(T+\{n\}) : \Sigma.(T+\{n\}) \rangle$$
$=$ {definition of G}
$$G.n$$
(*) $$\uparrow \langle \uparrow T : T \subseteq [0..n) \wedge nn.(T+\{n\}) : \Sigma.(T+\{n\}) \rangle \,.$$

Here we interrupt our calculation for the purpose of investigating the newly emerging expressions $nn.(T+\{n\})$ and $\Sigma.(T+\{n\})$. For the latter we straightforwardly have

$$\Sigma.(T+\{n\}) = \Sigma.T + f.n$$

For the former we observe

|[$T \subseteq [0..n)$ % the context for our
% calculation

$\triangleright$ $nn.(T+\{n\})$
$\equiv$ {definition of nn}
$\langle \forall i : i \in T+\{n\} : i+1 \notin T+\{n\} \rangle$
$\equiv$ {range splitting}
$\langle \forall i : i \in T : i+1 \notin T+\{n\} \rangle$
$\wedge$
$\langle \forall i : i=n : i+1 \notin T+\{n\} \rangle$
$\equiv$ {spelling out term of the first conjunct, one-point-rule on the second conjunct}
$\langle \forall i : i \in T : i+1 \notin T \wedge i+1 \neq n \rangle$
$\wedge$ $n+1 \notin T+\{n\}$
$\equiv$

$\equiv$ { term split on the first conjunct, second conjunct equivales true since $T \subseteq [0..n)$ - context. }

$\langle \forall i : i \in T : i+1 \notin T \rangle \wedge \langle \forall i : i \in T : i+1 \neq n \rangle$

$\equiv$ { definition of nn on first conjunct, shunting and one-point-rule on second conjunct } { • $0 \leq n-1$ }

$nn.T \;\wedge\; n-1 \notin T$.

]|.

And now we can resume our main calculation from (*) :

$G.n \uparrow \langle \uparrow T : T \subseteq [0..n) \wedge nn.(T + \{n\}) : \Sigma.(T+\{n\}) \rangle$

$=$ { by the above, provided • $0 \leq n-1$ }

$G.n \uparrow \langle \uparrow T : T \subseteq [0..n) \wedge nn.T \wedge n-1 \notin T : \Sigma.T + f.n \rangle$

$=$ { + over $\uparrow$, simplification }

$G.n \uparrow (\langle \uparrow T : T \subseteq [0..n-1) \wedge nn.T : \Sigma.T \rangle + f.n)$

$=$ { definition of G, • $0 \leq n-1$ }

$G.n \uparrow (G.(n-1) + f.n)$

Thus we have derived

$$G.(n+1) = G.n \uparrow (G.(n-1) + f.n) \quad , \quad 1 \leq n$$

With the observation that $G.0 = 0$ and $G.1 = f.0$, the corresponding program is straightforward:

$n := 0$; $x := 0$; $y := f.0$
; {inv. $x = G.n \land y = G.(n+1)$}{vf. $N - n$}
<u>do</u> $n \neq N - 1 \rightarrow$

$x, y := y, y \uparrow (x + f.(n+1))$
; {$x = G.(n+1)$}{$y = G.(n+2)$}
$n := n+1$

<u>od</u>
; {$y = G.N$}
print(y) .

* * *

I am pretty sure that the above program is not within easy reach for the operationally inclined — no matter how simple and tiny the final code is. It is in this kind of exercise where formal methods pay off, and as such Oege's example is a valuable contribution to our profession.

W.H.J. Feijen,
28 March 2003