J. Functional Programming **2** (2): 237–244, April 1992 © 1992 Cambridge University Press

FUNCTIONAL PEARLS
Two greedy algorithms

R.S. BIRD
Programming Research Group, Oxford University, 11 Keble Rd, Oxford OX1 3QD, UK

1 Introduction

At the recent TC2 working conference on constructing programs from specifications (Moeller, 1991), I presented the derivation of a functional program for solving a problem posed by Knuth (1990). Slightly simplified, the problem was to construct a shortest decimal fraction representing a given integer multiple of $1/2^{16}$. Later in the conference – and in a different context – Robert Dewar described a second problem that he had recently set as an examination question. In brief, the problem was to replace sequences of blanks in a file by tab characters wherever possible. Although Knuth's and Dewar's problems appear to have little in common, I suspected that both had the same 'deep structure' and were instances of a single general result about greedy algorithms. The purpose of this note is to bring the general result to light and to unify the treatment of the two problems. We begin by describing the problems more precisely.

2 Knuth's problem

TEX uses integer arithmetic, with all fractions expressed as integer multiples of $1/2^{16}$. Since the input language of *TEX* documents is decimal, there is the problem of converting between decimal fractions and their nearest internal representations.

Let D denote the set of digits d, where $0 \leqslant d < 10$, and let R denote the set of real numbers r in the range $0 \leqslant r < 1$. The function $\mathbf{dec} \in [D] \to R$ converts a decimal fraction to the number it represents, and is defined by a right-reduction $\oplus \not\leftarrow 0$ (i.e. *foldr* $(\oplus)\,0$ in functional programming), where $d \oplus r = (d + r)/10$. The problem in one direction is to compute the natural number

$$n = \lfloor 2^{16} \mathbf{dec}\, x + 1/2 \rfloor.$$

In other words, introducing $\mathbf{in}$ defined by $\mathbf{in}\, r = \lfloor 2^{16} r + 1/2 \rfloor$, the problem is to compute $\mathbf{in} \cdot \mathbf{dec}$. The restriction is that only integer arithmetic is allowed in the computation.

The more challenging problem is the converse: given n in the range $0 \leqslant n < 2^{16}$, find a decimal fraction $x = \mathbf{out}\, n$ such that $\mathbf{in}\,(\mathbf{dec}\, x) = n$ and x is as short as possible; in symbols,

$$\mathbf{out} \quad = \quad \mathrm{Min}(\#) \cdot \mathrm{Inv}(\mathbf{in} \cdot \mathbf{dec}).$$

Here, $\mathrm{Inv}(f)$ denotes the inverse image of f, defined for $f \in A \to B$ by

$$\mathrm{Inv}(f)\, b = \{a \in A \mid f\, a = b\}.$$

The function $\mathrm{Min}(\#)$ returns the shortest sequence in a set of sequences. More precisely, Min(#) returns the minimum element under some total ordering $\leqslant_\#$ that respects length, i.e. $x \leqslant_\# y$ implies $\#x \leqslant \#y$. There is a finite number of shortest sequences in Inv($\mathit{in} \cdot \mathit{dec}$), so ***out*** is well-defined for any choice of $\leqslant_\#$.

3 Dewar's problem

As originally formulated, Dewar's problem was:

"Write a program TAB that does the exact opposite of what DETAB does, i.e. it replaces blanks with tabs where possible."

In presenting the problem, Dewar added the following simplifications: assume that the given file of characters does not contain tab or backspace characters, and that no line of the file ends with blanks; assume also that tab stops occur in every eighth column.

To formalise the problem we have to interpret the phrase 'replaces blanks with tabs where possible'. One interpretation is that the output should contain as many tab characters as possible. Another interpretation, and the one we shall adopt, is that the output should be as short as possible. Thus, we shall specify **tab** by the equation

$$\mathbf{\mathit{tab}} \quad = \quad \mathrm{Min}(\#) \cdot \mathrm{Inv}(\mathbf{\mathit{detab}}),$$

where ***detab*** is the function which removes tabs. The function ***tab*** is well-defined for any choice of $\leqslant_\#$ since Inv(***detab***) returns a finite set of sequences.

Perhaps the simplest way of defining ***detab*** is to give the program: ***detab*** = ***untab*** 0, where ***untab*** n [] = [] and

$$\mathbf{\mathit{untab}}\, n\, ([a] +\!\!+\, x) \quad = \quad \begin{cases} [\mathbf{\mathit{nl}}] +\!\!+\, \mathbf{\mathit{untab}}\, 0\, x & \text{if } a = \mathbf{\mathit{nl}} \\ [\mathit{bl}]^{8-n} +\!\!+\, \mathbf{\mathit{untab}}\, 0\, x & \text{if } a = \mathbf{\mathit{tb}} \\ [a] +\!\!+\, \mathbf{\mathit{untab}}\, (n+1)\, x & \text{otherwise.} \end{cases}$$

Here, ***nl*** is the newline character, ***tb*** is the tab character, and $[\mathit{bl}]^m$ is a string of m blank characters. We use + to denote addition modulo 8, so the term $n + 1$ in the last line means $(n+1)$ mod 8. The first argument of ***untab*** is a counter in the range $0 \leqslant n < 8$ so that $8 - n$ is the position of the next tab stop. The following property of ***untab*** is needed below and can be proved by induction:

$$\begin{aligned} &\mathbf{\mathit{untab}}\, n\, ([a] +\!\!+\, y) = x \ \wedge\ [\mathit{bl}]^{8-n} \in \mathbf{\mathit{inits}}\, x \qquad (1) \\ \Rightarrow\ &(\exists z \in \mathbf{\mathit{tails}}\, y \ :\ \mathbf{\mathit{untab}}\, n\, ([\mathbf{\mathit{tb}}] +\!\!+\, z) = x\,), \end{aligned}$$

where ***inits*** and ***tails*** return the set of initial and tail segments of a list, respectively.

There is an instructive alternative definition of ***detab*** based on the fact that ***detab*** works line by line. The idea is to break the file into lines, detab each line, and then reassemble the lines. We have

$$\begin{aligned} \mathbf{\mathit{detab}} &= \mathbf{\mathit{unlines}} \cdot \mathbf{\mathit{detabl}}{*} \cdot \mathbf{\mathit{lines}} \\ \mathbf{\mathit{detabl}} &= \odot/ \cdot \mathbf{\mathit{fields}} \\ x \odot y &= x +\!\!+\, [\mathit{bl}]^n +\!\!+\, y \quad \text{where } n = 8 - \#x \bmod 8. \end{aligned}$$

The function ***lines*** is defined as the inverse of the injective function ***unlines***, where

$$\textit{\textbf{unlines}} = \oplus/ \quad \text{where } x \oplus y = x +\!\!\!+ [\textit{\textbf{nl}}] +\!\!\!+ y,$$

and the function ***fields*** is defined as the inverse of the injective function ***unfields***, where

$$\textit{\textbf{unfields}} = \oplus/ \quad \text{where } x \oplus y = x +\!\!\!+ [\textit{\textbf{tb}}] + +y.$$

The functions ***lines*** and ***unlines*** were considered as an exercise in Bird and Wadler, (1988), where essentially the following constructive definition of ***lines*** was synthesised: ***lines*** $= \otimes/ \cdot f*$, where $f\ a = (a = \mathbf{nl} \rightarrow [[\,],[\,]],\ [[a]])$ and

$$(xs +\!\!\!+ [x]) \otimes ([y] +\!\!\!+ ys) = xs +\!\!\!+ [x +\!\!\!+ y] +\!\!\!+ ys.$$

Exactly the same definition works for ***fields*** with ***tb*** replacing ***nl***.

Note that both Knuth's and Dewar's problems have been expressed in terms of computing some shortest sequence in a set of sequences defined as the inverse image of a function returning sequences.

4 Greedy algorithms

Let us now put the two problems in a more general context. Suppose F is a function with type

$$F \in A \rightarrow \{[B]\}^{+}.$$

Thus, for $a \in A$, the value $F\ a$ is a nonempty, and possibly infinite, set of finite sequences over B. We are interested in computing

$$f = \mathrm{Min}(C) \cdot F$$

efficiently, where $C \in [B] \rightarrow \mathbf{N}$ and $\mathbf{N}$ is the type of natural numbers. We assume that F returns a finite set of sequences with minimal value under C, so f is well-defined for any ordering $\leqslant_C$ that respects C.

We cannot say much about the computation of f unless we impose some conditions on C and F. We shall assume that C satisfies the following two conditions:

$$\begin{aligned} C\ x = 0 \quad &\equiv \quad x = [\,] \\ C\ x \leqslant C\ y \quad &\equiv \quad C\ (u +\!\!\!+ x) \leqslant C\ (u +\!\!\!+ y). \end{aligned}$$

A function C satisfying these conditions will be called a *cost* function. Clearly, $\#$ is a cost function, as is any function of the form $+/ \cdot w*$, provided w returns nonnegative values. One immediate consequence of the conditions is the fact that $C\ x \leqslant C\ (x +\!\!\!+ y)$ for all x and y. Another is the fact that

$$C\ x = C\ y \quad \equiv \quad C\ (u +\!\!\!+ x) = C\ (u +\!\!\!+ y).$$

Turning now to F, we shall assume that F satisfies the following conditions for some p, H, and $\ominus$. For all $a \in A$,

$$\begin{aligned} [\,] \in F\ a \quad &\equiv \quad p\ a \\ [b] +\!\!\!+ x \in F\ a \quad &\equiv \quad b \in H\ a \wedge x \in F\ (a \ominus b). \end{aligned}$$

These conditions are equivalent to the assertion that F satisfies the equation

$$F\ a = \{[\,] \mid p\ a\} \cup \{[b] +\!\!+\ x \mid b \in H\ a \wedge x \in F\ (a \ominus b)\}.$$

One can think of the sequences in $F\ a$ in terms of the paths in a directed graph. Nodes of the graph are labelled with elements $a \in A$, and edges are labelled with elements $b \in B$. Node a has outgoing edges labelled with elements in $H\ a$. An edge labelled b from node a leads to the node $a \ominus b$. Nodes with labels a satisfying $p\ a$ are called terminal nodes. The elements of $F\ a$ are then the sequences of labels along paths from a to some terminal node.

Finally, we impose an important third condition on F and C, one that enables us to compute f by a greedy algorithm. The condition is that there is some ordering $\leqslant_B$ on B with the property that for all a for which $p\ a$ is false, if

$$[b] +\!\!+\ x \in F\ a\ \wedge\ b \neq \mathrm{Min}(\leqslant_B)(H\ a),$$

then there exists a c and y such that

$$[c] +\!\!+\ y \in F\ a\ \wedge\ c <_B b\ \wedge\ C\ ([c] +\!\!+\ y) \leqslant C\ ([b] +\!\!+\ x).$$

This condition will be referred to as the *greedy* condition. Its effect is that we can compute $f\ a$ by choosing the smallest element under $\leqslant_B$ at each stage. More precisely,

Theorem 1 *Suppose* $f = \mathrm{Min}(C) \cdot F$, *where* C *and* F *satisfy the conditions enumerated above. Then we can find an ordering* $\leqslant_C$ *such that*

$$f\ a = \begin{cases} [\,] & \textit{if } p\ a \\ [b] +\!\!+ f\ (a \ominus b) & \textit{otherwise} \\ \textit{where } b = \mathrm{Min}(\leqslant_B)(H\ a). & \end{cases}$$

Proof
To define $\leqslant_C$, we first use the given ordering $\leqslant_B$ to construct a lexicographic ordering $\leqslant_L$ on $[B]$. We take $[\,] \leqslant_L x$ for all $x \in [B]$, and define

$$[b] +\!\!+\ x \leqslant_L [c] +\!\!+\ y\ \equiv\ b <_B c\ \vee\ (b = c \wedge x \leqslant_L y).$$

Now we define $\leqslant_C$ by

$$x \leqslant_C y\ \equiv\ C\ x < C\ y\ \vee\ (C\ x = C\ y \wedge x \leqslant_L y).$$

By construction, $\leqslant_C$ respects C.

Since, by definition of a cost function, $[\,]$ is the least element under $\leqslant_C$, we have $\mathrm{Min}(C)(F\ a) = [\,]$ if $p\ a$ holds. The proof of the theorem is completed by showing that if $p\ a$ does not hold, then

$$[b] +\!\!+\ x = \mathrm{Min}(C)(F\ a)\ \Rightarrow\ b = \mathrm{Min}(\leqslant_B)(H\ a)\ \wedge\ x = \mathrm{Min}(C)(F\ (a \ominus b)).$$

To establish the first conjunct of the consequent, we argue:

$$\begin{array}{ll} & [b] +\!\!+\ x \in F\ a\ \wedge\ b \neq \mathrm{Min}(\leqslant_B)(H\ a) \\ \Rightarrow & \{\text{greedy condition}\} \\ & (\exists c, y : [c] +\!\!+\ y \in F\ a\ \wedge\ c <_B b\ \wedge\ C([c] +\!\!+\ y) \leqslant C([b] + +x)) \\ \Rightarrow & \{\text{definition of } \leqslant_C\} \end{array}$$

$$\begin{array}{ll} & (\exists c, y : [c] \mathbin{+\!\!+} y \in F\ a\ \wedge\ [c] \mathbin{+\!\!+} y <_C [b] \mathbin{+\!\!+} x) \\ \equiv & \quad \{\text{definition of Min}(C)\} \\ & [b] \mathbin{+\!\!+} x \neq \text{Min}(C)(F\ a). \end{array}$$

For the second conjunct, we argue, for $x, y \in F\ (a \ominus b)$, that:

$$\begin{array}{ll} & [b] \mathbin{+\!\!+} x \leqslant_C [b] \mathbin{+\!\!+} y \\ \equiv & \quad \{\text{definition of } \leqslant_C\} \\ & (C\ ([b] \mathbin{+\!\!+} x) < C\ ([b] \mathbin{+\!\!+} y)) \vee \\ & (C\ ([b] \mathbin{+\!\!+} x) = C\ ([b] \mathbin{+\!\!+} y) \wedge [b] \mathbin{+\!\!+} x \leqslant_L [b] \mathbin{+\!\!+} y) \\ \Rightarrow & \quad \{C \text{ is a cost function}\} \\ & C\ x < C\ y \vee (C\ x = C\ y \wedge [b] \mathbin{+\!\!+} x \leqslant_L [b] \mathbin{+\!\!+} y) \\ \equiv & \quad \{\text{definition of } \leqslant_L\} \\ & C\ x < C\ y \vee (C\ x = C\ y \wedge x \leqslant_L y) \\ \equiv & \quad \{\text{definition of } \leqslant_C\} \\ & x \leqslant_C y. \end{array}$$

completing the proof of the theorem. □

5 Dewar's problem solved

Setting $F(n,x) = \text{Inv}(\mathbf{untab}\ n)\,x$, Dewar's problem is to compute $f(0,x)$, where $f(n,x) = \text{Min}(\#)F(n,x)$. We know $\#$ is a cost function, so we have to find a decompostion $(p, H, \ominus)$ for F and show that a suitable greedy condition holds.

It is immediate from the definition of $\mathbf{untab}\ n$ that $[\,] \in F(n,x)$ if and only if $x = [\,]$, so $p\ x \equiv (x = [\,])$. For $x \neq [\,]$ we have

$$\begin{array}{ll} & [b] \mathbin{+\!\!+} y \in F(n,x) \\ \equiv & \quad \{\text{definition of } F\} \\ & \mathbf{untab}\ n\ ([b] \mathbin{+\!\!+} y) = x \\ \equiv & \quad \{\text{definition of } \mathbf{untab}\} \\ & (b = \mathbf{nl}\ \wedge\ [\mathbf{nl}] \mathbin{+\!\!+} \mathbf{untab}\ 0\ y = x) \vee \\ & (b = \mathbf{tb}\ \wedge\ [\mathbf{bl}]^{8-n} \mathbin{+\!\!+} \mathbf{untab}\ 0\ y = x) \vee \\ & (b \notin \{\mathbf{nl}, \mathbf{tb}\}\ \wedge\ [b] \mathbin{+\!\!+} \mathbf{untab}\ (n+1)\ y = x) \\ \equiv & \quad \{\text{list calculus; definition of } F\} \\ & (b = \mathbf{nl}\ \wedge\ \mathbf{nl} = \mathbf{hd}\ x\ \wedge\ y \in F(0, \mathbf{tail}\ x)) \vee \\ & (b = \mathbf{tb}\ \wedge\ [\mathbf{bl}]^{8-n} \in \mathbf{inits}\ x\ \wedge\ y \in F(0, x \rightharpoondown [\mathbf{bl}]^{8-n})) \vee \\ & (b \notin \{\mathbf{nl}, \mathbf{tb}\}\ \wedge\ b = \mathbf{hd}\ x\ \wedge\ y \in F(n+1, \mathbf{tail}\ x)), \end{array}$$

where $\mathbf{inits}\ x$ is the set of initial segments of x, and $x \rightharpoondown z$ is what remains when initial segment z of x is removed from x. Thus, for $x \neq [\,]$, the set $H\,(n,x)$ of first elements of sequences in $F(n,x)$ is given by

$$H\,(n,x) = ([\mathbf{bl}]^{8-n} \in \mathbf{inits}\ x \rightarrow \{\mathbf{tb}, \mathbf{bl}\}, \{\mathbf{hd}\ x\}).$$

Furthermore, we can define $\ominus$ by

$$(n,x) \ominus b \;=\; \begin{cases} (0, \mathit{tail}\, x) & \text{if } b = \mathit{nl} \\ (0, x \rightharpoondown [\mathit{bl}]^{8-n}) & \text{if } b = \mathit{tb} \\ (n+1, \mathit{tail}\, x) & \text{otherwise.} \end{cases}$$

For the greedy condition, choose any ordering $\leqslant$ on characters for which tb is the minimum. We have

$$\begin{array}{ll} & ([b] \mathbin{+\!\!+} y) \in F(n,x) \;\wedge\; b \neq \mathrm{Min}(H(n,x)) \\ \Rightarrow & \{\text{since } b \neq \mathrm{Min}(H(n,x)) \text{ implies } b = \mathit{bl} \text{ and } [\mathit{bl}]^{8-n} \in \mathit{inits}\, x\} \\ & ([\mathit{bl}] \mathbin{+\!\!+} y) \in F(n,x) \;\wedge\; [\mathit{bl}]^{8-n} \in \mathit{inits}\, x \\ \equiv & \{\text{definition of } F\} \\ & \mathit{untab}\, n\,([\mathit{bl}] \mathbin{+\!\!+} y) = x \;\wedge\; [\mathit{bl}]^{8-n} \in \mathit{inits}\, x \\ \Rightarrow & \{\text{condition (1)}\} \\ & (\exists z \in \mathit{tails}\, y \;:\; \mathit{untab}\, n\,([\mathit{tb}] \mathbin{+\!\!+} z) = x) \\ \Rightarrow & \{\text{since } z \in \mathit{tails}\, y \text{ implies } \#z \leqslant \#y\} \\ & (\exists z \;:\; \#z \leqslant \#y \;\wedge\; ([\mathit{tb}] \mathbin{+\!\!+} z) \in F(n,x)). \end{array}$$

Since $\mathit{tb} < \mathit{bl}$, the greedy condition is established. We obtain $\mathit{tab}\, x = f(0,x)$, where $f(n,[\,]) = [\,]$ and, for $x \neq [\,]$,

$$f(n,x) = \begin{cases} [\mathit{nl}] \mathbin{+\!\!+} f(0, \mathit{tail}\, x) & \text{if } \mathit{hd}\, x = \mathit{nl} \\ [\mathit{tb}] \mathbin{+\!\!+} f(0, x \rightharpoondown [\mathit{bl}]^{8-n}) & \text{if } [\mathit{bl}]^{8-n} \in \mathit{inits}\, x \\ [\mathit{hd}\, x] \mathbin{+\!\!+} f(n+1, \mathit{tail}\, x) & \text{otherwise.} \end{cases}$$

The program can be made more efficient by introducing g, where

$$g(n,m,x) = f(n, [\mathit{bl}]^m \mathbin{+\!\!+} x),$$

and where $n+m \leqslant 7$. Using the assumption that no line of the text ends with blanks, we obtain after a short calculation that $\mathit{tab}\, x = g(0,0,x)$, where $g(n,m,[\,]) = [\,]$ and

$$g(n,m,[a] \mathbin{+\!\!+} x) \;=\; \begin{cases} [\mathit{nl}] \mathbin{+\!\!+} g(0,0,x) & \text{if } a = \mathit{nl} \\ [\mathit{tb}] \mathbin{+\!\!+} g(0,0,x) & \text{if } a = \mathit{bl} \wedge n+m = 7 \\ g(n,m+1,x) & \text{if } a = \mathit{bl} \wedge n+m < 7 \\ [\mathit{bl}]^m \mathbin{+\!\!+} [a] \mathbin{+\!\!+} g(n',0,x) & \text{otherwise where} \\ & n' = (n+m+1) \bmod 8. \end{cases}$$

6 Knuth's problem solved

Recall that Knuth's problem is to compute $\mathit{out}\, n$, where $0 \leqslant n < 2^{16}$ and $\mathit{out} = \mathrm{Min}(\#) \cdot \mathrm{Inv}(\mathit{in} \cdot \oplus \not\leftarrow 0)$. We have $\mathit{in}\, r = \lfloor 2^{16} r + 1/2 \rfloor$ and $d \oplus r = (d + r)/10$.

Let $F = \mathrm{Inv}(\mathit{in} \cdot \oplus \not\leftarrow 0)$. We know that $\#$ is a cost function, so it remains to find a decomposition $(p, H, \ominus)$ for F and show that a suitable greedy condition holds.

As a first step we use the following rule for the inverse image of the composition of two functions:

$$\mathrm{Inv}(f \cdot g) = \cup/ \cdot \mathrm{Inv}(g)* \cdot \mathrm{Inv}(f).$$

We then obtain $F = \cup/ \cdot \mathrm{Inv}(\oplus \not\leftarrow 0)* \cdot \mathrm{Inv}(in)$, where

$$\mathrm{Inv}(in)\,n = \{r \in R \mid 2n - 1 \leqslant 2^{17}r < 2n + 1\}.$$

It is not possible to obtain a decomposition $(p, H, \ominus)$ for F. However, a decomposition is possible if we generalise F to a function G, defined by $G = \cup/ \cdot \mathrm{Inv}(\oplus \not\leftarrow 0)* \cdot I$, where

$$I(a,b) \quad = \quad \{r \in R \mid a \leqslant 2^{17}r < b\}.$$

We restrict the definition of I to values b satisfying $0 < b < 2^{17}$. Note that $\mathrm{Inv}(in)\,n = I(2n-1, 2n+1)$ and $0 < 2n+1 < 2^{17}$ by the precondition on n, so that $F\,n = G\,(2n-1, 2n+1)$.

Writing G as a set comprehension, we have

$$G(a,b) \quad = \quad \{[\,] \mid 0 \in I(a,b)\} \cup \{[d] +\!\!+\, x \mid d \oplus (\oplus \not\leftarrow 0)\, x \in I(a,b)\}.$$

Using the assumption $0 < b$ we have $0 \in I(a,b)$ if and only if $a \leqslant 0$, so $p\,a \equiv (a \leqslant 0)$. Furthermore, for $d \in D$ and $r \in R$,

$$\begin{array}{ll} & d \oplus r \in I(a,b) \\ \equiv & \quad \{\text{definitions of } \oplus \text{ and } I\} \\ & 10a \leqslant 2^{17}(d + r) < 10b \\ \equiv & \quad \{\text{since } 0 \leqslant r < 1\} \\ & (10a/2^{17} - 1 < d < 10b/2^{17}) \;\wedge\; (10a - 2^{17}d \leqslant 2^{17}r < 10b - 2^{17}d) \\ \equiv & \quad \{\text{since } d \text{ is natural and } 10b/2^{17} \text{ is not}\} \\ & (\lfloor 10a/2^{17} \rfloor \leqslant d \leqslant \lfloor 10b/2^{17} \rfloor) \;\wedge\; r \in I(10a - 2^{17}d, 10b - 2^{17}d). \end{array}$$

Hence, introducing

$$\begin{array}{rcl} H(a,b) & = & \{d \in D \mid \lfloor 10a/2^{17} \rfloor \leqslant d \leqslant \lfloor 10b/2^{17} \rfloor\} \\ (a,b) \ominus d & = & (10a - 2^{17}d, 10b - 2^{17}d), \end{array}$$

we have

$$\begin{array}{ll} & [d] +\!\!+\, x \in G(a,b) \\ \equiv & \quad \{\text{definition of } G\} \\ & (\exists r : d \oplus r \in I(a,b) \;\wedge\; (\oplus \not\leftarrow 0)x = r) \\ \equiv & \quad \{\text{above}\} \\ & (\exists r : d \in H(a,b) \;\wedge\; r \in I((a,b) \ominus d) \;\wedge\; (\oplus \not\leftarrow 0)x = r) \\ \equiv & \quad \{\text{definition of } G\} \\ & d \in H(a,b) \;\wedge\; x \in G((a,b) \ominus d). \end{array}$$

We are left now with the verification of the greedy condition. Observe that if $x, y \in G(a,b)$ with $dec\,x \leqslant dec\,y$, then $z \in G(a,b)$ for any z with $dec\,x \leqslant dec\,z \leqslant dec\,y$. Thus, setting $e = \mathrm{Max}(H(a,b))$, we can argue

$$\begin{array}{ll} & [d] +\!\!+\, x \in G(a,b) \;\wedge\; d \neq e \\ \Rightarrow & \quad \{\text{definition of } H\} \end{array}$$

$$
\begin{array}{ll}
 & (\exists y : [d] +\!\!+ x \in G(a,b) \wedge [e] +\!\!+ y \in G(a,b)) \\
\Rightarrow & \quad \{\text{since } \mathit{dec}([d] +\!\!+ x) \leqslant \mathit{dec}[e] \leqslant \mathit{dec}(\ \mathit{seqe} +\!\!+ y)\} \\
 & [e] \in G(a,b),
\end{array}
$$

and so the greedy condition is satisfied by taking the largest digit at each step. By assumption $b < 2^{17}$, so the largest digit in $H(a,b)$ is $\lfloor 10b/2^{17} \rfloor$. With this choice of d we have $0 < 10b - 2^{17}d < 2^{17}$, so the restriction on the second argument of I is maintained. The greedy algorithm is

$$
f(a,b) = \begin{cases} [\,] & \text{if } a \leqslant 0 \\ [d] +\!\!+ f(10a - 2^{17}d, 10b - 2^{17}d) & \text{otherwise} \quad \text{where } d = \lfloor 10b/2^{17} \rfloor. \end{cases}
$$

References

Bird, R. S. and Wadler, P. 1988. *Introduction to Functional Programming*. Prentice Hall.

Moeller, B. (Editor). 1991. *Constructing Programs from Specifications*. North-Holland.

Knuth, D. E. 1990. A simple program whose proof isn't. In *Beauty is our Business*, (W. Feijen, D. Gries, N. van Gasteren, editors). Springer-Verlag.