D.A. Turner's reply.

In reply to EWD759, D.A. Turner sent me the following proof by returning mail.

"Thank you for your "somewhat open letter", which arrived yesterday. You pose several tasks - the order in which I have decided to tackle them is to establish first the precise formal relationship between \( f \) and \( g \).

First some notation. I shall call the elements of a list \( f \), \( f_0 f_1 f_2 \) etc. and I shall use the notation
\[
[h(i)]^B_{i=A}
\]
for the list \([h(A), h(A+1), ..., h(B)]\).

Now I define a function "upto"
\[
\text{upto } i \cdot f = \text{least } j \geq 0 \text{ such that } f_j > i \quad (\text{upto.0})
\]
The claim to be established is that
\[
g = [\text{upto } i \cdot f]_{i=0}^\infty \quad (\text{Theorem 1})
\]

From upto.0 we deduce the following two propositions, which could be considered a SASL definition of upto

\[
\text{upto } i \cdot f
\]
\[ a > i \vdash \text{upto } i \ (a; f) = 0 \quad \text{(upto.1)} \]
\[ a \leq i \vdash \text{upto } i \ (a; f) = 1 + \text{upto } i \ f \quad \text{(upto.2)} \]

We have also your definition of the function “\( k \)"

\[ p \leq y \vdash k \times y \ (p; q) = k \ (x + 1) \ y \ q \quad \text{(k.1)} \]
\[ p > y \vdash k \times y \ (p; q) = x : k \times (y + 1) \ (p; q) \quad \text{(k.2)} \]

From these four propositions we shall deduce the following generalization of Theorem 1.

**Theorem 0.** \[ k \times y \ f = \ [x + \text{upto } i \ f] \ ^\infty \]

Proof by structural induction on \( f \), which is an infinite list of integers.

**Case** \( \Omega_L \) (Note: we need to distinguish between \( \Omega_L \), the undefined element in the space to which \( f \) belongs, and \( \Omega_I \), the undefined integer. The relationship between them is \( \Omega_L = [\Omega_I]^\infty \)).

\[ k \times y \ \Omega_L = \Omega_L \quad \text{from } \text{k.1, k.2 by case exhaustion} \]

whereas
\[ [x + \text{upto } i \ \Omega_L] \ ^\infty \]
\[ = [x + \Omega_I] \ ^\infty \quad \text{from upto.1, upto.2} \]
\[ = [\Omega_I] \ ^\infty \quad \text{properties of } \Omega \]
\[ = \Omega_L \quad \text{as required} \]
\[
\text{case } p : P \\
= [x]^{P : y} + k \times p (p : P) \text{ by repeated appl of } k.2 \\
= [x]^{P : y} + k (x+i) p P \text{ by } k.1 \\
= [x]^{P : y} + [(x+i) + \text{upto } i \ f]_i^p \text{ ex hyp.} \\
= [x]^{P : y} + [x + (1 + \text{upto } i \ f)]_i^p \text{ properties of +} \\
= [x]^{P : y} + [x + \text{upto } i (p : P)]_i^p \text{ by upto.2} \\
= [x + \text{upto } i (p : P)]_i^y \text{ by upto.1 and rearranging} \\
\]

QED Theorem 0.

Whence, since \( g = k \ 0 \ 0 \ f \), we have immediately

\text{Theorem 1 } \quad g = [\text{upto } i \ f]_i^0 .

Also you asked me to establish that \( g \) is
A) ascending and B) unbounded, given appropriate assumptions about \( f \). This now follows easily from the above. (Relaxing the level of formality somewhat) we have:
A) From upto.0 it follows (by transitivity of "\( \rightarrow \)") that "upto \( i \ f \)" is an ascending function of \( i \).
Therefore, whatever the nature of \( f \), \( g \) is ascending.
B) Let us define "\( f \) is unbounded" to mean:
"for any \( N \geq 0 \), there is a \( j \geq 0 \) such that \( f_j > N \)."
Assume \( f \) is unbounded (if ascending not relevant). Then, from \( \text{uplo} \ 0 \),

\[ \text{uplo} \ i \ f \ \text{is defined for all } i \geq 0 \]

Given any \( N > 0 \), define \( j = \max \{ f_0, \ldots, f_N \} \)

then \( \text{uplo} \ j \ f \ = g \) exists, and by construction \( > N \)

So \( g \) too is unbounded.

* * *

So far Turner's reply. I like Turner's proof, and in view of the fact that Turner answered me by re-

turning mail it would be misplaced to complain too much about the fact that in "case \( p; f \)" the def-

initely less interesting case \( p < y \) hasn't been dealt with explicitly.

I am slightly uneasy in "case \( Q_k \)" — particularly after the parenthetical remark explaining the

difference between \( Q_k \) and \( Q_i \) — about the

justification "from k.1, k.2 by case exhaustion". My uneasiness is certainly caused by lack of

familiarity how to deal with \( Q_k \). Take

\[
\text{funny } (p; q) = \begin{cases} 
\text{if } p \geq 10 \rightarrow 1 : \text{funny } q \\
\text{if } p < 10 \rightarrow 1 : \text{funny } q \\
\end{cases}
\]

is \( \text{funny } Q_k = Q_k \) ? Or is \( \text{funny } Q_k = \text{ones} \)

(with \( \text{ones} = 1; \text{ones} \))? I expect the first answer,
though I would prefer the second one, if I am giving full weight to the remark (in [1], pg 57)
"The first point to be made is that in reasoning about SASL programs, ω can be treated just like any other value as regards being substitutable in equations."
Perhaps I have failed to fathom the complete depth of the constraint "as regards being substitutable in equations".

The correspondence was triggered by remarks in [1] such as the recommendation of applicative programming (pg. 14):
"The proofs (like the programs themselves) are very much shorter than the proofs of the corresponding imperative programs."
I had my doubts, which have not been dispelled by the comparison of Turner's proof with the one given in EWD758.


Plataanstraat 5
5671 AL NUENEN
The Netherlands

27 December 1980
prof. dr. Edsger W. Dijkstra
Burroughs Research Fellow