The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual

Chapter 2, Problem 2.4

Our function is

$$\begin{cases} f(1) = 1 \\ f(2n) = f(N) + 1 \\ f(2n+1) = 1/f(2n) \end{cases}$$

We claim that it maps even natural numbers to positive rational numbers with a value greater than 1 and odd natural numbers to positive rational numbers with a value less than 1—with 1 being mapped to 1. First note that any values produced must be positive: we either add 1 or take the (multiplicative) inverse, but never affect the sign of the result nor have any way to reduce the result to 0 if started with a nonzero value. That being the case, $f(2n)$ must be larger than 1 for any $n > 1$, since it is defined as $f(n) + 1$; and thus also $f(2n+1)$ must be less than 1 for any $n > 1$, since it is the inverse of $f(2n)$.

To verify that it is a bijection, we need to verify injectivity and surjectivity. Let us start with injectivity. Let i and j be two positive integers; we claim that $f(i)$ and $f(j)$ can only be equal if i and j are. From our remarks above, $f(i)$ and $f(j)$ can only be equal if i and j are both odd or both even. Suppose that i and j are both even, say $i = 2m$ and $j = 2n$; then we have $f(i) = f(2m) = f(m) + 1$ and $f(j) = f(2n) = f(n) + 1$, so that $f(i)$ and $f(j)$ can only be equal if $f(n)$ and $f(m)$ are. Suppose that i and j are both odd, say $i = 2m + 1$ and $j = 2n + 1$; then we have $f(i) = f(2m+1) = 1/f(2m)$ and $f(j) = f(2n+1) = 1/f(2n)$, so that $f(i)$ and $f(j)$ can only be equal if $f(2n)$ and $f(2m)$ are and, in turn, these two can only be equal if $f(n)$ and $f(m)$ are. Thus, in both cases, we have reduced the arguments by a factor of 2 and still have the same implication; in other words, we have an inductive step! The basis is clearly true (for instance, we can observe $f(1) \neq f(2)$).

To verify surjectivity, we need to show that, for any rational number a/b (a reduced fraction), there is a positive integer n with $f(n) = a/b$. The easiest way to prove such a statement is to solve the second question in the problem:

how is f reversed? Then, given a/b, we can reverse f on it and obtain x. So, given some reduced fraction a/b, we begin by testing whether it is larger than or less than 1: if larger, it was obtained by mapping an even integer, otherwise it was obtained by mapping an odd integer. Each step in the mapping can be reversed directly: if a/b is larger than 1, the step is reversed by subtracting 1, otherwise it is reversed by taking the multiplicative inverse. The result is the following simple recursive procedure:

```
function invert{fraction}
  case fraction of
  = 1: return(1)
  > 1: return(2*invert(reduce(fraction-1)))
  < 1: return(1+invert(1/fraction))
```

(where the function `reduce` returns the fraction in reduced terms). The remaining question is whether or not this procedure always converges; if it does, then it is clear that what it returns is indeed the pre-image of the given fraction, so that it does indeed implement f^{-1}, so that also f is surjective.

To see that it converges, note first that the current value of the fraction is always strictly positive: we start with a strictly positive value and each step keeps it strictly positive. Next, note that the inversion step (for fractions less than 1) must be preceded and followed by one or more subtraction steps: inverting a fraction less than 1 yields a fraction larger than 1. (On the other hand, we can have many consecutive subtraction steps.) Thus at least one out of two steps is a subtraction step and each such step reduces the numerator of the fraction. Since numerators and denominators are integers, the number of times they can be decremented before reaching 1 is bounded, so that the procedure must converge.