Chapter 43 : The Longest Common Subsequence.

Given a list of values called S, a subsequence of S is got by deleting some of the values of S. For example

S = 3,1,5,7,4,6,2,4,1

contains subsequences such as

X = 3,5,7,6,1

X'= 6,4,1

etc…..

What is clear is that the values in the subsequences remain in the same order as they occurred in the original list.

Given X[0..M) and Y[0..N) of int. determine the length of the longest common subsequence of X and Y.

We begin by defining the longest common subsequence.

* (0) lcs.m.0	=	0		
* (1) lcs.0,n	=	0		
* (2) lcs.(m+1).(n+1)	=	1 + lcs.m.n	<=	X.m = Y.n
* (3) lcs.(m+1).(n+1)	=	lcs.(m+1).n $\uparrow$ lcs.m.(n+1)	<=	X.m $\neq$ Y.n

The postcondition which we want to establish is

Post : r = lcs.M.N

Invariants.

Our first attempt at choosing invariants is to choose

P0 : r = lcs.m.n
P1 : $0 \leq m \leq M \wedge 0 \leq n \leq N$

This could be easily established by

r, m, n := 0, 0, 0

or

r, m, n := 0, M, 0

or

r, m, n := 0, 0, N

In the first case this will result in the guard on the loop being

$$m \neq M \lor n \neq N$$

This will mean that we would have to be careful anytime we inspect X.m, Y.n so as not to try accessing outside the array bounds. Similar concerns would arise from both of the other two choices.

Another option would be to break the symmetry and to consider something of the form

P0 : r = lcs.m.N

This is easily established by

r, m := 0, 0

Consider what would happen as we increased m by one. We would be faced with

lcs.(m+1).N

Definitions (2) and (3) above would indicate that we would now need lcs.m.(N-1), lcs.m.N and lcs.(m+1).(N-1)

and lcs.(m+1).(N-1) would require lcs.m.(N-2) etc …

Clearly, recomputing these values of lcs many times is inefficient. So, rather than recompute we introduce an auxiliary array h[0..N] and invariants

P0 : $\langle \forall\ i : 0 \leq i \leq N\ : h.i = lcs.m.i \rangle$
P1 : $0 \leq m \leq M$

Establish Invariants.

We observe

(m := 0).P0
= {text substitution }
$\langle \forall\ i : 0 \leq i \leq N\ : h.i = lcs.0.i \rangle$
= { definition (1) }
$\langle \forall\ i : 0 \leq i \leq N\ : h.i = 0 \rangle$

This suggests the following to establish the invariants.

```
m, k := 0, 0;
Do k ≠ N+1 —>

        k, h.k := k+1, 0

Od
{ P0 ∧ P1 }
```

Achieving postcondition.

We observe.

$$
\begin{array}{ll}
 & P0 \wedge P1 \wedge m = M \\
\Rightarrow & \qquad \{\text{Leibniz}\} \\
 & \langle \forall\, i : 0 \le i \le N \;:\; h.i = lcs.M.i \,\rangle \\
\Rightarrow & \qquad \{\text{Instantiate } i = N\,\} \\
 & h.N = lcs.M.N
\end{array}
$$

Guard.

$$m \neq M$$

Variant.

$$M\text{-}m$$

Loop body.

We observe.

$$
\begin{array}{ll}
 & (m := m+1).P0 \\
= & \qquad \{\text{text substitution }\} \\
 & \langle \forall\, i : 0 \le i \le N \;:\; h.i = lcs.(m+1).i \,\rangle
\end{array}
$$

Achieving this will require a loop. We propose the following invariants.

$$
\begin{array}{ll}
Q0 : & \langle \forall\, i : 0 \le i \le n \;:\; h.i = lcs.(m+1).i \,\rangle \;\wedge \\
 & \langle \forall\, i : n < i \le N \;:\; h.i = lcs.m.i \,\rangle
\end{array}
$$

$$Q1 : \quad 0 \le n \le N$$

We observe

$$P0 \Rightarrow (n := 0).Q0$$

Also

$$Q0 \wedge n = N \;\Rightarrow (m := m+1).P0$$

Guard.

$$n \neq N$$

Variant

$$N\text{-}n$$

Loop body.

(n := n+1).Q0

= {text substitution}

$\langle \forall\ i : 0 \le i \le n+1\ : h.i = lcs.(m+1).i \rangle\ \wedge$
$\langle \forall\ i : n+1 < i \le N\ : h.i = lcs.m.i \rangle$

= {split off i = n+1 term in first conjunct }

$\langle \forall\ i : 0 \le i \le n\ : h.i = lcs.(m+1).i \rangle\ \wedge$
$\langle \forall\ i : n+1 < i \le N\ : h.i = lcs.m.i \rangle\ \wedge\ h.(n+1) = lcs.(m+1).(n+1)$

= {Q0 , so first two conjuncts are true }

h.(n+1) = lcs.(m+1).(n+1)

= {case analysis, $X.m \ne Y.n$, definition (3) }

$h.(n+1) = lcs.(m+1).n \uparrow lcs.m.(n+1)$

= { Q0 }

$h.(n+1) = h.n \uparrow h.(n+1)$

So this suggests

If $X.m \ne Y.n$ —> n, h.(n+1) := n+1, $h.n \uparrow h.(n+1)$

Now we consider the other case

h.(n+1) = lcs.(m+1).(n+1)

= {case analysis, X.m = Y.n, definition (2) }

h.(n+1) = 1 + lcs.m.n

= { Q0 doesn't give us a value for lcs.m.n so propose Q2: a = lcs.m.n}

h.(n+1) = 1 + a

And this suggests

If X.m = Y.n —> n, h.(n+1) := n+1, 1 + a

Now we consider this new invariant.

Q2 : a = lcs.m.n

Establish invariant

n, a := 0,0

Maintain invariant

```
        (n, a :=  n+1, E ). Q2
=               {text substitution }
        E = lcs.m.(n+1)
=               {Q0}
        E = h.(n+1)
```

so

n, a := n+1, h.(n+1)

maintains Q2.

Now put it all together..

```
        m, k := 0, 0;
        Do k ≠ N+1 —>

                k, h.k := k+1, 0

        Od;
        { P0 ∧ P1 }
        Do m ≠ M —> { P0 ∧ P1 ∧ m ≠ M}

                        n, a := 0, 0;
                        {Q0 ∧ Q1 ∧  Q2 }
                        Do n ≠ N —> { Q0 ∧ Q1 ∧  Q2  ∧ n ≠ N}

                                If X.m ≠ Y.n —> n, a, h.(n+1) := n+1, h.(n+1), h.n ↑ h.(n+1)
                                [] X.m = Y.n —> n, a, h.(n+1) := n+1, h.(n+1), 1 + a
                                Fi
                        Od;
                        m := m+1
        Od;
        r := h.N
        {r = lcs.M.N}
```

This algorithm has complexity O(M*N)