Derivation of Algorithms

Partition Problems using Invariant Diagrams

COMP H 4018
Lecturer: Stephen Sheridan

Example Solution: Two Segment Problem

Question
Using an invariant diagram *specify* and, hence, *derive* an O(N) solution to the following: Given f[0..N) {N>0} containing only 0's and 1's, sort it so that all 0's precede all 1's. Include a complete solution as part of your answer.

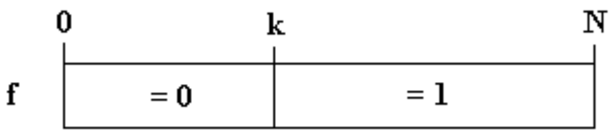
Specification

|[Con N: int {N > 0}

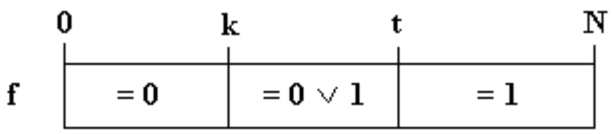
var
    f: array [0..N) of int;
    $\{\forall j: 0 \leq j < N: f.j = 0 \vee f.j = 1\}$
    k: int;
    S
    $\{0 \leq k \leq N: \forall j: 0 \leq j < k: f.j = 0 \wedge \forall j: k \leq j < N: f.j = 1\}$
]|

Derivation

Step 1: Draw a diagram to describe the post condition.



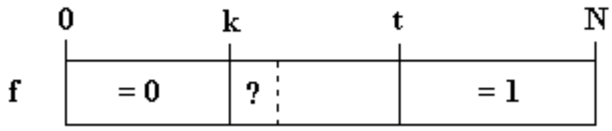
Step 2: Construct an invariant diagram by weakening the post-condition, we do this by introducing another segment representing the unsorted data.



Mathematical equivalent of above diagram

$$\{ 0 \leq k \leq N: \forall j: 0 \leq j < k: f.j = 0 \wedge \forall j: k \leq j < t: f.j = 0 \vee f.j = 1 \wedge \forall j: t \leq j < N: f.j = 1 \}$$

Step 3: Use the invariant diagram to derive $O(N)$ solution.



At the start of processing

f[k..t) represents the whole array. f[0..k) and f[t..N) are both empty. In other words

k,t : = 0,N.

At the end of processing

f[k..t) will be empty, that is, $k = t$. Therefore, the guard on the loop is $k < t$.

In the middle of processing

In the body of the loop we focus on f.k. There are two possible cases:

$f.k = 0 \vee f.k = 1$

We consider each one…

Case 1:

f.k = 0 => Simply increase k by 1

i.e. k := k + 1

Case 2:

f.k = 1 => Swap f.k with f.t – 1 and decrement t.

i.e. f.k,f.t – 1 := f.t-1, f.k; t := t -1;

```
{if..fi}
if f.k = 0 →
        k := k + 1
[] f.k = 1 →
        f.k, f.t – 1 := f.t – 1, f.k;
        t := t – 1;
fi
```

Step 4: Prove termination.

<u>Termination</u>

Decrease t – k

At start:

(t – k > 0) (t,k := N,0)
≡ {substitution}
N – 0 > 0
⇐
N > 0

<u>Proof for each case</u>

(t - k) (k := k + 1)
≡ {substitution}
t – (k + 1)
≡ {arithmetic}
t – k – 1
<
t – k

(t - k) (t:= t - 1)
≡ {substitution}
t – 1 – k
≡ {arithmetic}
t – k – 1
<
t - k

Step 5: Write out a complete solution.

Complete Solution

```
|[ Con N: int {N > 0}

  var
        f: array [0..N) of int;
        {∀j: 0 ≤ j < N: f.j = 0 ∨ f.j = 1}

        k: int;

        k, t := 0, N;
        do k < t →
                if f.k = 0 →
                        k := k + 1
                [] f.k = 1 →
                        f.k, f.t – 1 := f.t – 1, f.k;
                        t := t – 1;
                fi
        od;

        {0 ≤ k ≤ N: ∀j: 0 ≤ j < k: f.j = 0 ∧ ∀j: k ≤ j < N: f.j = 1}
]|
```

This solution if $O(N)$ because only a single iteration of the data is required!