Derivation of Algorithms

Partition Problems using Invariant Diagrams

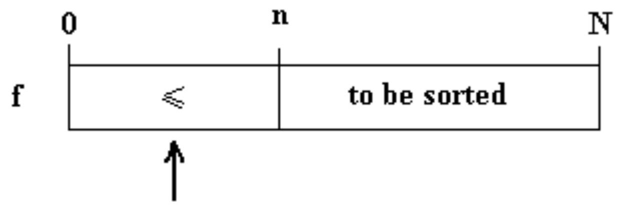
COMP H 4018
Lecturer: Stephen Sheridan

Invariant Diagrams & Partition Problems

Invariant Diagram

Diagram used to describe the state of computation during the 'middle' of processing. Invariant diagrams are very useful for array partitioning problems.

For example: to describe sorting one might draw the following diagram.



$\forall j : 0 \le j < n\text{-}1 : f.j \le f.j + 1$

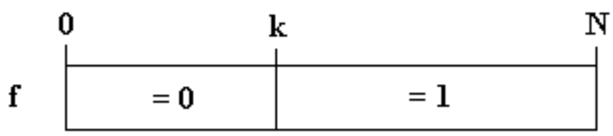
Example 1

Using an invariant diagram specify and, hence, derive an O(N) solution to the following: Given f[0..N) containing only 0's and 1's, sort it so that all 0's proceed all 1's.

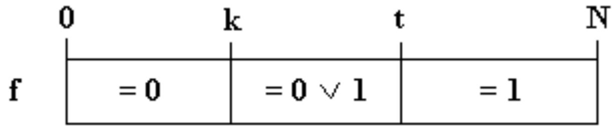
Specification

```
|[ Con N: int {N ≥ 0}

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

        k: int;

        S

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

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



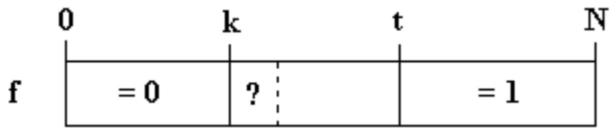
Step 2: Construct an invariant diagram by weakening the post-condition, we do this by introducing another segment representing the unsorted data.



Mathematical equivalent of above diagram

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

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



<u>**At the start of processing**</u>

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

<u>**At the end of processing**</u>

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

<u>**In the middle of processing**</u>

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…

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

i.e. $k := k + 1$

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

i.e. $f.k, f.t - 1 := f.t\text{-}1, f.k;\ t := t\ \text{-}1;$

Termination

Decrease $t - k$

At start:

$(t - k \geq 0)\ (t,k := N,0)$
$\equiv$ {substitution}
$N - 0 \geq 0$
$\Leftarrow$
$N \geq 0$

Body for each case

$(t - k)\ (k := k + 1)$
$\equiv$ {substitution}
$t - (k + 1)$
$\equiv$ {arithmetic}
$t - k - 1$
$<$
$t - k$

$(t - k)\ (t := t - 1)$
$\equiv$ {substitution}
$t - 1 - k$
$\equiv$ {arithmetic}
$t - k - 1$
$<$
$t - k$

Code for S

```
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;
```

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