

Algorithmique – écriture de programmes efficaces et sûrs

Romain Gille

27/01/2016

Exemple

$T[0:n]$ est un tableau d'entiers à n valeurs.
Calculer dans s , la somme des éléments de T .

```
int somme (int[] T){
    int n = T.length;
    int k = 0, s = 0; // initialisation des variables
    while(k != n){    // arrêt quand k = n
        s = s + T[k];
        k++;          // incrémentation de k
    }
    return s;         // retourne s
}
```

Modélisation

s = somme des valeurs du k -préfixe de T

$$s = \sum_{x \in [0:k]} T[x] \equiv I(s, k)$$

```
int somme (int[] T){
    int n = T.length;
    int k = 0, s = 0; // I(s, k)
    while(k != n){    // I(s, k) et k != n => I(s + T[k], k + 1)
        s = s + T[k]; // I(s, k + 1)
        k++;          // I(s, k)
    }
    return s;
}
```

Modélisation n°2 (variation sur thème)

s = somme des valeurs du k-suffixe de T

k-suffixe de $T[k : n] = [T[k], T[k + 1], \dots, T[n - 1]]$

Initialisation : s = 0, k = n

Condition d'arrêt : k = 0

Progression : $I(s, k)$ et $(k \neq 0) \Rightarrow I(s + T[k - 1], k - 1)$

```
int somme (int[] T){
    int n = T.length;
    int k = 0, s = 0; // I(s, k)
    while(k != n){    // I(s, k) et k != 0 => I(s + T[k - 1], k - 1)
        s = s + T[k]; // I(s, k - 1)
        k++;          // I(s, k)
    }
    return s;
}
```

Modélisation n°3

s = somme de $T[i:j]$

On connaît :

- $i \geq j : s = 0$
- $j = i + 1 : s = T[i]$

s = sg + sd

```
int somme(int[] T, int i, int j){
    if(j - i <= 0) return 0;
    if(j - i == 1) return T[i];
    else{
        int k = (i + j) / 2;
        int sg = somme(T, i, k);
        int sd = somme(T, k, j);
        return sg + sd;
    }
}
```