L3-INFO Algorithmique

Université Joseph Fourier GRENOBLE UFR IM²AG

ALGO5 : Travaux dirigés, séance 2
Analyse de coût, coût moyen

Le coût d'une séquence d'instructions dépend en général des valeurs de certaines variables du programme. Le coût maximal (respectivement minimal) est la plus grande (respectivement plus petite) valeur du coût qui puisse **être observée** lors d'une exécution de cette séquence. Le coût moyen est la moyenne des valeurs du coût, chacune de ces valeurs étant **pondérée par sa probabilité d'être observée dans les conditions de l'expérience**. Le calcul du coût moyen (sauf dans le cas trivial où le coût maximal est égal au coût minimal) nécessite donc **toujours** de faire des hypothèses (probabilistes) sur les valeurs de certaines variables du programme.

1 Analyse de programmes simples

Exercice 1 Evaluer le nombre d'additions exécutées par l'algorithme ci-dessous en considérant d'abord les cas favorables (coût minimal) et défavorables (coût maximal). Puis faire l'analyse en moyenne, en prenant pour hypothèse que la probabilité pour que le test $T[i] > a$ soit *vrai* est $1/2$.

```
(1) pour i de de 1 a n faire
(2)         si T[i] > a alors
(3)             s := s + T[i]
```

Exercice 2 Mêmes questions en prenant pour hypothèse que la probabilité pour que le test $a > b$ soit *vrai* est $1/2$. Peut-on observer le résultat obtenu, lors d'une exécution particulière de l'algorithme ?

```
(1) si a > b alors
(2)     pour i = 1 a n faire
(3)             x := x+a
(4) sinon x := b
```

Exercice 3 Mêmes questions en prenant pour hypothèse que la probabilité pour que le test $T[i] > a$ soit *vrai* est $1/2$.

```
(1) i := 1
(2) tant que i < n etpuis T[i] > a faire
(3)             x := x+a
(4)             i := i+1
```

Calcul de coût

Exercice 4 Même question pour les algorithmes ci-dessous. La séquence des appels successifs à la fonction `random()` est modélisée par une suite $U_1, \cdots, U_n, \cdots$ de variables aléatoires réelles, indépendantes, de même loi uniforme sur l'intervalle $[0, 1[$ (pour l'implantation de cette fonction voir par exemple en C la fonction `drand()`).

(1) tant que random() $>$ 1/2 faire
(2) x := x+a

(1) n := 1
(1) tant que random() $<=$ 1/n faire
(2) n := n+1

2 Algorithme de tri par segmentation (Quicksort)

Le problème est de trier les éléments d'un ensemble de taille n. Les éléments de l'ensemble sont tous comparables deux à deux (unicité des clés) et on suppose qu'il n'y a pas de doublons (ordre total). L'algorithme consiste à choisir un élément pivot, segmenter l'ensemble par rapport à la valeur du pivot puis à trier récursivement les éléments plus petits, puis plus grand et assembler l'ensemble. La preuve de cet algorithme est laissée en exercice.

TriSegmentation(E);
Données : Un ensemble E de n éléments comparables
Résultat : Les éléments par ordre croissant
if $E \neq \emptyset$
  Pivot = **Extrait** (E) ; `// retire l'élément pivot de` E
  (E_1, E_2) = **Segmentation** (E,pivot) ; `//` E_1 `(resp.` E_2`) éléments plus petits (resp. plus grand) que pivot`
  return Assemble (**TriSegmentation** (E_1),*pivot,* **TriSegmentation** (E_2))

Algorithme 1 : Algorithme du tri par segmentation

L'objectif est d'analyser le coût de cet algorithme en nombre de comparaisons effectuées entre des éléments. Une première étape consiste à se donner des exemples pour comprendre le déroulement de l'algorithme. Ensuite on recherche des exemples présentant des "cas extrêmes" au pire et au mieux. Puis on formalise le problème en écrivant les équations vérifiées par le coût de l'algorithme. On résout ces équations, ou bien, si on ne trouve pas de solution on essaye de majorer/minorer le coût (en ordre de grandeur). Si il y a une grande différence entre le coût au pire et le coût au mieux (par exemple s'il ne sont pas sur la même échelle), on évalue l'ordre de grandeur du coût moyen.

Etape 1 Faire une trace de l'algorithme pour $E = \{C, A, R, I, B, O, U\}$ en considérant l'ordre alphabétique. Essayer avec $E = \{S, T, Y, L, O, G, R, A, P, H, I, E\}$.

Etape 2 Donner des exemples extrémaux.

Etape 3 Formalisation du problème : se donner des notations et donner l'équation de récurrence "de base".