Complexité

INFORMATIQUE COMMUNE - TP n° 2.2 - Olivier Reynet

À la fin de ce chapitre, je sais :

- ☞ calculer la complexité d'un algorithme simple
- ☞ donner les complexités dans le pire des cas des tris comparatifs génériques
- ☞ expliquer la différence entre le tri rapide et le tri fusion

A Complexité d'algorithmes simples

A1. Calculer la complexité de l'algorithme 1. Y-a-il un pire et un meilleur des cas ?

Algorithme 1 Produit scalaire

```
1 : Fonction PRODUIT_SCALAIRE(x, y)          ▷ x et y sont des vecteurs à n éléments
2 :     s ← 0
3 :     pour i = 0 à n − 1 répéter
4 :         s ← s + x_i y_i                  ▷ coût ?
5 :     renvoyer s
```

A2. Calculer la complexité de l'algorithme 2 dans le meilleur et dans le pire des cas.

Algorithme 2 Palindrome

```
1 : Fonction PALINDROME(w)
2 :     n ← la taille de la chaîne de caractères w
3 :     i ← 0
4 :     j ← n − 1
5 :     tant que i < j répéter
6 :         si w[i] = w[j] alors
7 :             i ← i + 1
8 :             j ← j − 1
9 :         sinon
10 :            renvoyer Faux
11 :    renvoyer Vrai
```

A3. Calculer la complexité de l'algorithme 3. Y-a-t-il un pire et un meilleur des cas ? On fait l'hypothèse que la fonction PUISSANCE(a,b) est de complexité constante $O(1)$. Cette hypothèse vous semble-t-elle raisonnable ?

Algorithme 3 Évaluation simple d'un polynôme

```
1 : Fonction EVAL_POLYNÔME(p, v)
2 :     d ← degré de p
3 :     r ← p[0]
4 :     pour i = 0 à d répéter
5 :         r ← r + p[i] ×PUISSANCE(v, i)
6 :     renvoyer r
```

A4. Utiliser la méthode de Horner (cf. algorithme 4) pour écrire un autre algorithme pour évaluer un polynôme. Quelle complexité pouvez-vous obtenir ? Est-ce plus rapide ?

Algorithme 4 Évaluation d'un polynôme par la méthode d'Horner

```
1 : Fonction HORNER(p, d, v)
2 :     r ← p[d]
3 :     pour i = d − 1 à 0 répéter
4 :         r ← r × v
5 :         r ← r + p[i]
6 :     renvoyer r
```

B Tri fusion

B1. Programmer le tri fusion en Python.

B2. Quelle est la complexité de cet algorithme ? Y-a-t-il un pire et un meilleur cas ?

B3. Vérifier, en mesurant le temps d'exécution, la justesse de votre calcul précédent.

C Tri rapide

C1. Programmer le tri rapide en Python.

C2. Quelle est la complexité de cet algorithme ? Y-a-t-il un pire et un meilleur cas ?

C3. Vérifier, en mesurant le temps d'exécution, la justesse de vos calculs précédents.