AP1 - Travaux dirigés, séance 2

Complexité

Le tri par insertion et le tri par sélection

*O*n rappelle le principe de ces deux algorithmes de tri, et on précise quelle variante on va utiliser.
Donnée : un tableau de n éléments, indicé de 1 à n. Le type des éléments est quelconque, muni d'un ordre total.
Résultat : les mêmes éléments dans le même tableau, mais dans l'ordre croissant.
1. Tri par insertion : une itération ($i = 2$ à n) à chaque pas de laquelle on insère à sa place l'élément d'indice i dans la séquence triée formée des $i - 1$ premiers éléments.
Initialement : l'élément 1 forme une séquence triée.
Finalement : les n éléments sont triés.
On effectue l'insertion par une recherche séquentielle de l'emplacement k de l'élément i, et un décalage vers la droite des éléments de k à $i-1$. L'algorithme classique effectue ces deux opérations ensemble, c'est-à-dire décale l'élément i vers la gauche (par un échange) jusqu'à ce qu'il atteigne sa "bonne" place.
2. Tri par sélection (du minimum) : une itération ($i = 1$ à $n - 1$) à chaque pas de laquelle on trouve et met à sa place i l'élément correspondant.
Finalement : les $n - 1$ premiers éléments sont à leur place, donc le dernier est bien placé aussi.
On effectue la sélection par un parcours des éléments de i à n en sélectionnant le minimum, et en mémorisant son indice k, puis en effectuant l'échange entre les éléments i et k.

Le choix d'une mesure du coût

On choisit de mesurer le **nombre de comparaisons entre éléments**.
Discussion : et le nombre d'échanges ? en exercice à la maison !

La taille des données

Pourquoi pas le **nombre d'éléments** ? De toute façon, pour les tris, on n'a pas le choix : tout le monde utilise le nombre d'éléments comme taille, donc il faut faire de même.

Le cas n = 2

Modélisation : quel que soit le type des éléments, et leurs valeurs, seul intervient dans ces algorithmes la relation d'ordre entre ces éléments. On peut donc, sans perte de généralité, choisir un modèle de données simple : les entiers de 1 à n...
Ici, les entiers 1 et 2.
Oui, mais si les deux éléments sont égaux ?
On en fera la "discussion" plus tard (si on a le temps), mais adoptons tout de suite l'hypothèse : "tous les éléments sont différents".
Il y a deux données possibles : A = (1, 2) et B = (2, 1).
1. Tri par insertion.
A. On compare 2 à 1. Il est plus grand, donc à sa place. C'est fini. **1 comparaison**.
B. On compare 1 à 2. Il est plus petit, on échange. C'est fini. **1 comparaison**.
Le coût est 1 dans tous les cas. La complexité est donc 1. **2. Tri par sélection**.
A. On cherche le minimum, **1 comparaison**, c'est 1, il est à sa place, c'est fini.

B. On cherche le minimum, **1 comparaison**, c'est 1, on l'échange avec 2, c'est fini.
Le coût est 1 dans tous les cas. La complexité est donc 1.

Le cas n = 3

Les données sont : A = (1,2,3), B = (1,3,2), C = (2,1,3), D = (2,3,1), E = (3,1,2), F = (3,2,1).
1. Tri par insertion.
A. On compare 2 à 1 : 1 comparaison. Puis 3 à 2 : 1 comparaison. **2 comparaisons**.
B. On compare 3 à 1 : 1 comparaison. Puis 2 à 3, puis 2 à 1 : 2 comparaisons. **3 comparaisons**.
C. On compare 1 à 2 : 1 comparaison. Puis 3 à 2 : 1 comparaison. **2 comparaisons**.
D. On compare 3 à 2 : 1 comparaison. Puis 1 à 3, puis 1 à 2 : 2 comparaisons. **3 comparaisons**.
E. On compare 1 à 3 : 1 comparaison. Puis 2 à 3, puis 2 à 1 : 2 comparaisons. **3 comparaisons**.
F. On compare 2 à 3 : 1 comparaison. Puis 1 à 3, puis 1 à 2 : 2 comparaisons. **3 comparaisons**.
Le coût est soit 2, soit 3.
La complexité est 3. La complexité "au mieux" (cas favorables) est 2.
Si l'on fait l'hypothèse que les 6 données possibles ont la même probabilité d'occurrence (hypothèse classique), la complexité "en moyenne" est $2*1/6+3*1/6+2*1/6+3*1/6+3*1/6+3*1/6$, soit $(4+12)/6 = 8/3$, ou 2,66... ou $2+2/3$.
Comme on l'avait vu au TD1, la complexité "en moyenne" peut correspondre à un coût jamais observé : ici, c'est particulièrement évident, puisque le coût est un entier.
2. Tri par sélection.
Pas besoin de regarder les données possibles une par une : l'algorithme effectue toujours les mêmes comparaisons.
Il y a 2 comparaisons pour trouver le minimum des 3 éléments, qui devient l'élément d'indice 1. Puis 1 comparaison pour trouver le minimum des 2 éléments restants. Au total, **toujours 3 comparaisons**.
La complexité est donc égale à 3 (et aussi la complexité "au mieux" et la complexité "en moyenne").

Le cas général

Il y a $n!$ données possibles. Commençons par le plus facile...
1. Tri par sélection.
Ce qu'on a vu pour $n = 3$ se généralise de façon évidente.
Le coût est toujours égal à $n(n-1)/2$, somme des entiers de 1 à $n-1$.
Et la (les) complexité(s) aussi.
2. Tri par insertion.
Cas défavorables : le nombre de comparaisons, à l'étape i (rappel : i de 2 à n), est au pire égal à $i-1$: on compare l'élément à insérer à tous les autres. Ce qui, au total, donnerait un coût égal à $n(n-1)/2$, somme des entiers de 1 à $n-1$. Cette valeur est un **majorant** de la complexité.
MAIS... est-elle égale à la complexité ? C'est-à-dire, existe-t'il une donnée (au moins) pour laquelle le coût atteint cette valeur ?
Oui : c'est la donnée (n, n-1, ..., 2,1).
La complexité est donc égale à $n(n-1)/2$.
Cas favorables : le nombre de comparaisons, à l'étape i est au mieux égal à 1, chaque élément est à sa place. La donnée (1,2, ... n-1, n) correspond à ce cas, bien sûr. La complexité "au mieux" est donc égale à $n-1$.
Complexité "en moyenne" : elle est évidemment encadrée par les deux valeurs précédentes ; donc elle est comprise entre $n-1$ et $n(n-1)/2$, **quelles que soient les hypothèses probabilistes que l'on fait sur la répartition des données**.
Peut-on en dire plus ?
Le minorant est une fonction d'ordre n, le majorant une fonction d'ordre n^2 : la moindre des choses serait de savoir quel est l'ordre de grandeur de la complexité "en moyenne" ! Au fait, cela peut-il être autre chose que n ou n^2 ?

Analyse en moyenne du tri par insertion

Hypothèse probabiliste : les $n!$ permutations sont équiprobables.

Conséquence 1 : le dernier élément est 1,2,3... ou n avec la probabilité $1/n$. **Conséquence 2** : si le dernier élément est i, les (n-1) ! possibilités pour les autres éléments sont équiprobables ; ces (n-1) ! permutations équiprobables sont les données possibles pour le tri par insertion de n-1 éléments (moyennant un re-codage des entiers 1,2,3...i-1, i+1, ...n en 1,2,3,...n-1 dans cet ordre).
Donc la complexité en moyenne du tri par insertion vérifie l'équation de récurrence

C(n) = C(n-1) * f(n) où f(n) est la complexité moyenne de l'insertion du dernier élément, avec l'hypothèse que ce dernier élément a la même probabilité d'être inséré en chacune des n positions possibles. La complexité moyenne de cette insertion est donc 1/n (1 + 2 + ... + n-1 + n-1) = (n+1)/2 - 1/n [n-1 apparaît 2 fois : on fait n-1 comparaisons si i=1, et aussi si i= 2] Donc C(n) = C(n-1) + (n+1)/2 - 1/n et C(1) = 0

On vérifie que C(2) = 0 + 3/2 - 1/2 = 1 et que C(3) = 1 + 2 - 1/3 = 2 + 2/3

La forme compacte de C(n) est donc :

C(n) = n(n+1)/4 + n/2 - Hn ou C(n) = (n+1)(n+2)/4 - Hn - 1/2

où Hn est le nombre harmonique : Hn = 1 + 1/2 + 1/3 + ... + 1/n

C(n) est de l'ordre de n^2 (et voisin de la moitié de la complexité "au pire"). NB. Hn est de l'ordre de log(n)

On a par exemple C(10) = 33 - 1/2 - H10 = 29.57