UNIVERSITÉ DE RENNES 1

Bases d'algorithmique

Interrogation n°1

Durée : 1h
Les notes de cours et de TD sont autorisées.

Exercice 1

On considère un tableau de n objets, que l'on peut comparer entre eux selon plusieurs critères différents (par exemple un tableau de t-uples d'entiers, que l'on peut comparer selon le premier entier de chaque t-uple, ou selon le deuxième, etc). Deux objets peuvent être égaux selon un critère mais distincts selon un autre. On dit qu'un algorithme de tri est *stable* s'il ne change pas l'ordre respectif d'objets égaux pour la comparaison utilisée. Par exemple un tri stable, selon la troisième composante, appliqué à la liste de quadruplets suivante :

$$(4,3,1,7),(1,1,4,0),(0,2,1,5),(3,0,1,4)$$

donnerait :

$$(4,3,1,7),(0,2,1,5),(3,0,1,4),(1,1,4,0).$$

En effet, les quadruplets sont maintenant bien triés par ordre croissant de leur troisième coordonnée (1, 1, 1, puis 4) et, de surcroît, l'ordre des trois quadruplets qui partageaient la même troisième coordonnée n'a pas été modifié par rapport à la liste d'origine.

1. Citer un algorithme de tri stable et un algorithme de tri non stable parmi les algorithmes vus en cours.
2. Étant donné un tableau de n entiers d'au plus k chiffres en base 2 (représentés en base 2, de sorte que le i-ième chiffre de chaque entier puisse être lu en temps constant), montrer que l'on peut trier le tableau par ordre croissant en appliquant d'abord un tri stable selon le chiffre le moins significatif (i.e. le chiffre des unités), puis en appliquant un tri stable selon le chiffre des 2-aines, et ainsi de suite jusqu'à un tri stable selon le chiffre le plus significatif (celui des 2^{k-1}-aines).
3. Décrire, sous forme de pseudocode, un algorithme de tri stable en temps $O(n)$ pour un tableau de n entiers d'au plus k chiffres en base 2, en comparant selon la i-ème chiffres. (On pourra utiliser un ou plusieurs tableaux auxilaires de taille n, et y ranger temporairement les entiers dont le i-ème chiffre a une valeur donnée).
4. Quelle est la complexité de l'algorithme de tri d'entiers de la question 2, en fonction de n (en considérant que k est fixé), si l'on utilise le tri stable précédent ?
5. Comment se fait-il que l'on trouve une meilleure complexité que $O(n\log(n))$, étant donné qu'un tri par comparaisons ne peut pas avoir une meilleure complexité que $O(n\log(n))$ (cf. TD 1, exercice 1) ?