Algorithmique III. L2 Informatique I41.

TD 6. Tris empiriques et complexité des tris comparatifs [1]

EXERCICE 1. Si L est une liste de n éléments d'un ensemble $(X, \leqslant)$ totalement ordonné.

(1) Combien y-a-t-il de comparaisons du type $L[i] \leqslant L[j]$ possibles si les indices $(i, j) \in [\![1, n]\!]^2$?

(2) Même question pour les indices $(i, j) \in [\![1, n]\!]^2$ tels que $i < j$?

EXERCICE 2. Soit n un entier naturel. On considère l'algorithme de génération d'une permutation aléatoire de S_n suivant : on construit la permutation identité, i.e. la liste $L := [1, 2, \ldots, n]$ et pour toute valeur de i allant de 1 à $n-1$, on tire un nombre j au hasard dans l'intervalle $[\![i, n]\!]$ et on échange les termes d'indices i et j de la liste, autrement dit $L \leftarrow (L[i], L[j]) \circ L$ où $(L[i], L[j])$ désigne une transposition.

(1) Écrivez un algorithme `GenPerm(n)` qui renvoie une permutation "aléatoire" de S_n sur la base de la description précédente. On utilisera sans l'écrire l'algorithme auxiliaire `Alea(a,b)` qui renvoie une valeur de l'intervalle $[\![a, b]\!]$ avec la probabilité uniforme $\frac{1}{b-a+1}$.

(2) Démontrez que la liste obtenue après ces $n-1$ transpositions est une permutation aléatoire de $\mathfrak{S}_n$, au sens où la distribution de probabilités sur les permutations est la distribution uniforme : chaque évènement élémentaire a une probabilité de $1/(n!)$.

Indication : remarquez que l'entier à la position i a été tiré au hasard parmi les $n - i + 1$ valeurs à partir de sa position (lui compris).

EXERCICE 3. **(1)** Écrivez un algorithme Propager(L, g, d) qui balaie la liste L de l'indice g à l'indice $d-1$, compare les termes adjacents $L[k]$ et $L[k+1]$ en les échangeant si $L[k] > L[k+1]$. L'algorithme devra renvoyer un booléen indiquant s'il y a eu ou non des échanges.

(2) Démontrez que votre algorithme s'arrête.

1. Version du 22 mars 2024, 09 : 25

(3) À l'aide d'un invariant de boucle adéquat, démontrez qu'après cette propagation, on a :

$$L[d] = \max\{L[i] \mid i \in [\![g, d]\!]\}. \tag{1}$$

(4) Démontrez qu'après l'exécution de l'algorithme du tri par propagation (le tri à bulles), la liste L est triée.

EXERCICE 4. Le *tri cocktail* est une variation autour du tri par propagation. Au lieu de balayer la liste uniquement dans le sens gauche-droite, on alterne un balayage gauche-droite avec un balayage droite-gauche.

(1) Modifiez l'algorithme Propager(L, a, b) pour que la propagation se fasse de gauche à droite si $a < b$ et de droite à gauche sinon. On supposera que $a \neq b$.

(2) Écrivez l'algorithme du tri cocktail.

(3) Soit $L = [2, 3, 4, 5, 1]$. Comparez le nombre d'échanges et le nombre de tests effectués par le tri à bulles et le tri cocktail pour trier L ?

EXERCICE 5. Le *tri à peigne* est une autre déclinaison du tri par propagation, qui s'avère plus efficace que le tri cocktail. L'objectif est de faire remonter les bulles plus rapidement en avançant avec un pas plus grand que 1. Ce pas décroît d'un facteur donné après chaque balayage de la liste pour atteindre la valeur 1 et revenir au comportement usuel du tri par propagation. Ainsi, au lieu de comparer les termes contigus $L[i]$ et $L[i+1]$, on compare les termes $L[i]$ et $L[i+p]$ où la pas p est initialisé à $\lfloor n/\rho \rfloor$ avec $1 < \rho < 2$ et mis à jour après chaque passe sur la liste par $p \leftarrow \max\{1, \lfloor s/\rho \rfloor\}$.

Une fois que le pas p a atteint la valeur 1, le tri se comporte comme le tri par propagation optimisé, mais la condition d'entrée dans la boucle se résume à savoir s'il y a eu un échange lors du précédent passage, sans limiter progressivement la propagation en fin de liste. Adaptez l'algorithme `Propager` puis écrivez l'algorithme du tri à peigne avec deux paramètres, la liste à trier et le coefficient de réduction.

EXERCICE 6. **(1)** Soit $(X, \leq)$ un ensemble totalement ordonné et fini de cardinal n. Montrez qu'il existe une unique bijection croissante entre l'ensemble X et l'ensemble $[\![1, n]\!]$ muni de l'ordre naturel $\leqslant$.

(2) Démontrez qu'une liste L d'éléments d'un ensemble $(X, \leq)$ totalement ordonné est triée si et seulement si L est une application croissante.

EXERCICE 7. Construisez les arbres de décision des trois tris empiriques, i.e., le tri par sélection dans sa version où le min est sélectionné à chaque étape, le tri par propagation (optimisé, cf. algorithme Propager plus haut pour le test) et le tri par insertion (on insère en partant de la fin) pour les listes qui décrivent le groupe $\mathfrak{S}_3$.