Algorithmique TD3
L3 Informatique – ENS Cachan

Guillaume Bury

26 septembre 2016

Exercice 1

Une file est une structure de données permettant de stocker une suite d'éléments, où les insertions se font toutes d'un même côté et les suppressions toutes de l'autre côté (à l'inverse des piles dans lesquelles insertions et suppressions sont faites du même côté).

1. Implémenter l'ajout et la suppression dans une file à l'aide de deux piles.
2. Quel est le coût d'une opération dans le pire des cas ?
3. Calculer le coût amorti d'une opération.

Exercice 2

On souhaite implémenter une pile à l'aide d'un tableau. Si on fixe la taille du tableau a priori, on limite la taille de la pile ; de plus, beaucoup de place est perdue lorsqu'il y a trop peu de données. Pour résoudre ces problémes, on utilise des tableaux dynamiques dont la taille peut changer de telle sorte qu'au moins la moitié du tableau est toujours remplie. L'opération qu'on veut implémenter est l'empilement : elle consiste à insérer l'élément à la premiére place libre lorsque le tableau n'est pas plein ; sinon, on crée un nouveau tableau de taille double du précédent, on copie dedans tous les éléments du premier tableau puis on empile finalement l'élément dedans.

On néglige le coût de l'allocation d'un tableau. Le coùt à mesurer est donc le nombre de copies effectuées.

1. Implémenter la fonction `empiler`.
2. Quelle est la complexité de la fonction `empiler` dans le pire des cas ?
3. Calculer la complexité amortie de la fonction `empiler` lorsqu'on commence par une pile vide.
4. On considère une procédure différente `empiler'` qui consiste à augmenter la taille du tableau d'une constante k plutôt que de la doubler, lorsque le tableau est plein. Calculer la complexité (avec potentiel) de cette procédure.
5. Même question dans le cas où l'on élève au carré la taille du tableau lorsqu'il est plein.
6. Supposons qu'on veuille également une opération de dépilement `depiler`, qui consiste à effacer le dernier élément du tableau. Si la suppression du dernier élément résulte en un tableau à moitié plein, on crée un tableau de taille divisée par deux et on copie les éléments du tableau dedans. Quelle est la complexité (avec potentiel) des opérations `empiler` et `depiler`.
7. On propose de diviser par deux la taille du tableau quand le nombre d'éléments descend en dessous d'un quart. Calculer la complexité de cette nouvelle version de `depiler`.

Exercice 3

On considére l'algorithme suivant qui détermine si un mot donné est un palindrome.

Entrée : T : tableau[1..n]
pour i de 1 à $\lfloor \frac{n}{2} \rfloor$ faire
si $T[i] \neq T[n+1-i]$ retourner faux
retourner vrai

1. Quelle est la complexité de cet algorithme dans le pire des cas ?
2. On choisit de manière aléatoire et uniforme un mot de taille n sur un alphabet de k lettres. Quelle est la probabilité que le mot soit un palindrome ? Calculer l'espérance du nombre de passages dans la boucle pour notre algorithme ?

Exercice 4

1. Donner l'algorithme de tri rapide
2. Calculer sa complexité en moyenne, en supposant une répartition uniforme des permutations du tableau.

Exercice 5

On voudrait rechercher un élément x donné dans un tableau A non trié. On considère les deux algorithmes suivants :

— On choisit une permuation du tableau selon une loi de probabilité uniforme, puis on teste tous les éléments de gauche à droite.
— On choisit un indice i selon une loi de probabilité uniforme. Si $A[i] = x$, alors on retourne oui, sinon on recommence.

1. Proposer une modification de l'algorithme aléatoire afin qu'il termine si tous les indices ont été testés.
2. Calculer l'espérance du nombre d'itérations de l'algorithme aléatoire dans les cas suivants :
   (a) On suppose que x n'est pas dans A.
   (b) On suppose qu'il y a un seul indice i tel que $A[i] = x$.
   (c) On suppose qu'il y a $m > 0$ occurences de x dans A.
3. On suppose qu'il y a $m > 0$ occurences de x dans A, soit $k \geq 1$, pour chaque algorithme, caluler la probabilité que le nombre d'itérations de l'algorithme soit supérieur à k.
4. Comparer les deux algorithmes.