TD Algo2 – session 2 – Files de priorités et ensembles disjoints

Objectifs d'apprentissage :
— étudier les structures de file de priorité et de forêt d'ensembles disjoints ;
— concevoir des algorithmes relatifs aux files de priorités et aux forêts d'ensembles disjoints ;
— mettre en pratique des techniques de preuve pour la validité ou la complexité d'algorithmes.

Exercice 1 : Extraire-max-tas

Quel est le nombre minimum d'échanges lors d'un appel à l'opération `Extraire-Max-Tas` dans un tas de taille n sans clés dupliquées ? Fournir un tas de taille 15 pour lequel on obtient ce minimum. Même question si l'on appelle l'opération 2 fois, puis 3 fois de suite.

Exercice 2 : Diminuer-clé-tas

Écrire le pseudo-code d'une procédure DIMINUER-CLÉ-TAS(A, i, k) dans un tas max. Quelle en est sa complexité en temps ?

Exercice 3 : Tri de listes triées

Donner un algorithme à temps $O(n \log k)$ qui fusionne k listes triées pour produire une liste triée unique, n étant ici le nombre total d'éléments toutes listes confondues. (Conseil : utiliser un tas min pour la fusion multiple.)

Exercice 4 : Pile et file

Montrer comment implémenter une file FIFO (first-in first-out) avec une file de priorité. Montrer comment implémenter une pile avec une file de priorité.

Exercice 5 : Pire cas

Donner une séquence de m opérations UNION et TROUVER-ENSEMBLE, contenant n opérations CRÉER-ENSEMBLE, dont le temps d'exécution est $O(m \log n)$ quand on se sert uniquement de l'union par rang.

Exercice 6 : Compression de chemin

La compression de chemin modifie le parent dans une forêt d'ensembles disjoints pour que le parent pointer directement sur le représentant dès que c'est possible en $\Theta(1)$ opérations.

Il reste un cas où elle n'est pas mise en place dans l'algorithme. Compléter l'algorithme pour réaliser cette compression dans ce cas également.

Exercice 7 : Échange

Chaque échange dans la boucle de AUGMENTER-CLÉ-TAS nécessite 3 affectations. Comment peut-on ajuster l'algorithme pour qu'il n'y ait plus qu'une seule affectation par itération de la boucle ?

Exercice 8 : Augmenter-clé-tas

Pour prouver la validité de AUGMENTER-CLÉ-TAS, on considère l'invariant de boucle suivant : Au début de chaque itération de la boucle, le tableau $A[1 : A.n]$ satisfait la propriété de tas max, à une infraction potentielle près : $A[i]$ risque d'être plus grand que $A[\text{PARENT}(i)]$. Cet invariant n'est pas suffisant pour prouver la validité de l'algorithme. Comment faudrait-il le préciser ?

Exercice 9 : Composantes connexes

Pendant l'exécution de `Composantes-Connexes` sur un graphe non orienté $G = (V, E)$ à k composantes connexes, combien de fois UNION est-elle appelée ? Combien de fois exécute t-on la partie de UNION qui modifie `parent` ? Les réponses seront exprimées en fonction de V, E et k.

Exercice 10 : Afficher ensemble

Considérons l'opération AFFICHER-ENSEMBLE, qui reçoit un élément x et imprime tous les membres de l'ensemble de x, dans n'importe quel ordre. Montrer comment ajouter un seul attribut à chaque nœud dans une forêt d'ensembles disjoints de sorte que l'opération prenne un temps linéaire par rapport au nombre de membres de l'ensemble de x et que les temps d'exécution asymptotiques des autres opérations restent inchangés. On suppose que l'on peut imprimer chaque membre de l'ensemble en temps constant.