Université de Mons
Faculté Des Sciences

Structure de données II

Windowing

Professeur:
Véronique Bruyère
Assistants:
Pierre Hauweele

Auteurs:
Cyril Moreau
Arnaud Moreau

Année académique 2022-2023

Contents

1 Questions préliminaires

1.1 Question 1

Si on ne considère que les coordonées en x, voyez-vous que celles-ci sont organisées selon une file à priorité ? Où est la coordonées minimum (maximum) ? Celle file à priorité est-elle un tas ? Justifiez.

Tout d'abord, une fille de priorité est un arbre binaire tel que pour tout noeud, la donnée qui s'y trouve est supérieure ou égale (resp. inférieure ou égale) à celles de ses fils.

En ne considérant que les coordonnées en x, on remarque qu'elles sont arrangées tel que la valeur de chaque noeud est toujours inférieure ou égale à celle de ses fils ce qui correspond bien à la définition d'une file de priorité.

La coordonnée minimum est la racine tandis que la coordonnée maximum est une feuille de l'arbre.

Cependant, ce n'est pas un tas car elle ne respecte pas l'équilibre du tas. En effet, en plus d'avoir la caractéristique d'une file de priorité, un tas doit être équilibré tel que ses niveaux sont complètement remplis de noeuds excepté éventuellement le dernier niveau qui doit commencer à être rempli *par la gauche*.

1.2 Question 2

Si on ne considère que les coordonées en y, voyez-vous que celles-ci sont organisées selon un arbre binaire de recherche ? Où est la coordonée minimum (maximum) ? Justifiez.

Si on ne considère que la coordonnée y, on pourrait penser que ce n'est pas un ABR cepandant la valeur qui est stockée dans chaque noeud est y_{mid} et non simplement y donc il faut regarder par rapport à cela.

La définition d'un ABR est :
Un arbre binaire de recherche est un arbre binaire tel que pour tout noeud x, la donnée qui s'y trouve est

- $<$ aux données du sous-arbre gauche de x
- $>$ aux données du sous-arbre droit de x

Or, dans la définition de la structure de donnée employée, il est dit que l'on note P un ensemble de points et y_{mid} la médiane des coordonnées de tous les points sauf la racine. Aussi, on note

$$P_{below} := \{p \in P \backslash \{p_{min}\} \mid p_y < y_{mid}\}$$
$$P_{above} := \{p \in P \backslash \{p_{min}\} \mid p_y > y_{mid}\}$$

La structure de données est construite telle que la racine possède deux sous-arbres : P_{below} comme sous-arbre gauche et P_{above} comme sous-arbre droit. Il est donc clair que nous avons affaire à un ABR, puisque pour toutes les données d'un noeud x, y_{mid} est supérieur aux données contenues dans son sous-arbre gauche et inférieure aux données contenues dans son sous-arbre droit.

La y_{mid} minimum est la feuille la plus à gauche de l'arbre tandis que le y_{mid} maximum est la feuille la plus à droite de l'arbre.

1.3 Question 3

De quelle façon est équilibré un arbre de recherche à priorité ?

Lorsqu'on insère une nouvelle donnée, il faut la placer correctement selon sa coordonnée x et y. Par simplicité, notons p le noeud portant la donnée que l'on traite. Pour la coordonnée x, il faut que tous les descendants p soient des noeuds dont la coordonnée x est supérieure à celle de p. Pour la coordonnée y, il faut que tous les noeuds appartenant au sous-arbre gauche (resp. droit) de p aient une coordonnée y inférieure (resp. supérieure) à celle de p. On pourrait placer un nouveau noeud (que l'on note a) en parcourant tous les fils gauche jusqu'en bas de l'arbre. Arrivé en bas, on place a en tant que fils gauche. Ensuite, on le remonte en l'inversant avec son père tant que sa coordonnée x est inférieure à celle de son père. Chaque fois qu'on le remonte, on le considère comme père de son précédent père et fils gauche/droite selon la place de son ancien père. Une fois que l'on s'est arrêtés, on va vérifier la coordonnée y. Si le noeud est fils gauche de son père et que sa coordonnée y est inférieure à celle de son père, on s'arrête. Si elle est supérieure, le noeud devient fils droit. Inversément, si le noeud est fils droit de son père et que sa coordonnée y est supérieure à celle de son père, on s'arrête. Si elle est inférieure, le noeud devient fils gauche. Si l'on a déplacé le noeud (fils droit -> fils gauche ou fils gauche -> fils droit, on va le redescendre tant que sa coordonnée x est supérieure à celle d'un de ses fils. De nouveau, lorsqu'on s'arrête, on vérifie s'il ne faut pas le déplacer de fils droit -> fils gauche ou de fils gauche -> fils droit, et on répète la descente. Tout cela, jusqu'à ce qu'il ne faille ni le descendre, ni le changer de position (car c'est lorsqu'il ne faut pas le changer que l'on s'arrête, comme écrit plus haut).

Équilibré = équilibre entre les sous-arbres ? Parce que si oui alors mon raisonnement sert à rien lol

oui

1.4 Question 4

La construction d'un arbre de recherche à priorité peut se faire en $O(n \log_2 n)$ si n est le nombre de points de $\mathbb{R}^2$ contenus dans l'arbre. Expliquez comment on peut y parvenir et comment le prouver. Il faut sans doute utiliser une autre structure de données qui permet de calculer efficacement la médiane.

oui

1.5 Question 5

La structure d'arbre de recherche à priorité permet d'obtenir efficacement l'ensemble des points de T qui sont contenus dans une fenêtre donnée de la forme $(-\infty : x'] \times (y : y']$. Expliquer comment adapter l'algorithme proposé pour traiter une fenêtre de la forme $[x : +\infty] \times [y : y']$, $[x : x'] \times (-\infty : y]$ ou $[x : x'] \times [y : +\infty]$.

Pour une fenêtre de la forme $(-\infty : x'] \times (y : y']$, on a :

- Un minimum à la racine selon la coordonée x
- Un parcours dit "vers le bas" qui incrémente la valeur des coordonnées x (c'est-à-dire tout fils d'un noeud a une coordonnée x inférieure à celle de ce noeud)
- Les files gauche/droits sont ordonnés selon leur coordonnée y par rapport à la valeur y_{mid}.

Pour une fenêtre de la forme $[x : +\infty] \times [y : y']$, il faudrait avoir un maxmim à la racine selon la coordonnée x et un parcours "vers le bas" qui incrémente la coordonnée x.

Pour une fenêtre de la forme $[x : x'] \times (-\infty : y]$, il faut inverser le rôle des coordonnées : on place le minimum à la racine selon la coordonnée y, et un parcours "vers le bas" qui incrémente la coordonnée y. Aussi, les fils gauche/droits sont ordonnés selon leur coordonnée x par rapport à la valeur x_{mid}.

Pour une fenêtre de la forme $[x : x'] \times [y : y']$, on peut choisir sur quelle composante jouera le parcours "vers le bas" (x ou y). La racine aura comme valeur de la composante (x ou y) choisie un minimum ou un maximum, pour lequel on imposera une borne (x(y) si on y met le minimum en x(y), x'(y') si on y met le maximum en x(y)). On imposera que tout noeud (racine comprise) doit se trouver dans l'intervalle désormais bornée à gauche et à droite, alors qu'avant, on pouvait avoir une absence de borne et donc pas de minimum/maximum à respecter dans certains cas. On imposera aussi que les fils gauche/droits seront ordonnées selon la composante qui n'a pas été choisie, et que celle-ci doit se trouver dans l'intervalle bornée à gauche et à droite.

1.6 Question 6

Les auteurs du Chapitre 10 font l'hypothèse que tous les points ont des coordonées bien distinctes en x et en y. Expliquer pourquoi.

On accepte les points ayant une composante identique en x ou en y (ou exclusif) puisqu'on peut les représenter différemment comme suit : (x, y) devient ((x|y), (y|x)). Il est trivial de voir que si deux points ont une composante identique, alors, sous cette nouvelle forme, ils ont deux composantes distinctes. On empêche cependant d'avoir des points avec deux composantes identiques (par exemple (2, 3) et (2, 3)) car alors, sous leur nouvelle forme, ils conserveraient leur identicité ((2, 3) devient ((2|3), (3|2)), et on ne peut plus différencier plusieurs instances de (2, 3), là où c'était le cas lorsqu'une composante variait)).

1.7 Question 7

Une technique est présentée à la page 111. Celle-ci permet de simuler des coordonnées distinctes pour un ensemble de points quelconques et une requête. Expliquez comment.

La technique est de remplacer les coordonnées x et y de chaque point par des paires notées $(x \mid y)$ pour avoir un nouveau point $((x \mid y), (y \mid x))$. L'ordre sur ces nouvelles coordonées est défini tel que pour deux paires $(a \mid b)$ et $(c \mid d)$, on a

$$(a \mid b) < (c \mid d) \iff a < c \vee (a = c \wedge b < d)$$

Avec cet ordre, on a que les coordonées sont distinctes pour un ensemble de points quelconques.

Imaginons maintenant que l'on souhaite récupérer les points d'un ensemble P qui se trouve dans la fenêtre $R = [x : x'] \times [y : y']$ en utilisant la technique vue ci-dessus. On va donc construire $\hat{\text{P}}$ l'ensemble P où chaque point $p = (p_x, p_y)$ est remplacé par $((p_x \mid p_y), (p_y \mid p_x))$.

Nous devons également modifier la fenêtre R pour qu'elle soit compatible avec la nouvelle représentation des points. on a donc :

$$\hat{\text{R}} = [(x \mid -\infty) : (x' : +\infty)] \times [(y \mid -\infty) : (y' : +\infty)]$$

1.8 Question 8

Dans le chapitre, la technique présentée est adaptée à des points. Quelles différences importantes aurait-on avec des segments de droite ? Comment peut-on adapter la technique ?

oui