L3 MIAGE
X32I090

Université de Nantes
2020/2021

Feuille de Travaux Pratiques

Implémentation et Manipulation d'ABR de Tas Binaires

Mars 2021

Ce projet est à réaliser en **binômes** (un seul monôme autorisé si le groupe possède un nombre impair d'étudiants).**Trois séances de TP d'1h20** y seront consacrées.

La programmation se fera dans le langage de votre choix. *Assurez-vous que vos programmes compilent et fonctionnent sous Linux sur les machines du CIE.*

Vous rendrez **l'ensemble des sources de votre projet**, sous la forme d'une archive NOM1-NOM2.zip (où NOM1 et NOM2 sont les noms de famille des étudiants formant le binôme). La décompression de l'archive doit produire un répertoire NOM1-NOM2 contenant tous les éléments de votre travail (compte-rendu, sources, exécutable, jeux d'essai) et *un fichier texte contenant les instructions détaillées de compilation et d'exécution*.

Cette archive est à **déposer sous Madoc**, au plus tard :
— Groupes 688 et 689 : le **Lundi 12 Avril 2021 à 23h**
— Groupes 690 et 691 : le **Lundi 31 mai 2021 à 23h**

Le non respect des consignes entraîne automatiquement des points en moins. En particulier, les retards seront sanctionnés de la façon suivante : moins d'1 heure de retard : -2 points ; entre 1 et 2 heures : -4 points ; entre 2 et 12 heures de retard : -8 points ; au-delà de 12h de retard : -16 points.

1 Arbre Binaire de Recherche de Tas Binaires (ABRTB)

Nous considérons dans ce projet une structure de données appelée *arbre binaire de recherche de tas binaires* (abrégée ABRTB). Il s'agit d'un ABR A dont chaque nœud contient 3 informations :
— deux entiers m et M vérifiant $m \leq M$
— un Tas Binaire et plus précisément un max-tas binaire (TB) T, dans lequel chaque position contient un entier k compris entre m et M ($m \leq k \leq M$).

L'arbre A doit être un ABR "classique" lorsqu'on prend pour élément de comparaison les valeurs m des nœuds de A. Par ailleurs, les intervalles $[m; M]$ des nœuds de A doivent tous être *disjoints*, c'est-à-dire que deux intervalles différents ne se chevauchent pas. Un exemple d'ABRTB est donné dans la Figure 1, page suivante.

2 Travail demandé

1. Proposer un programme convivial qui, par l'intermédiaire d'un menu, permet à l'utilisateur de choisir de réaliser une ou plusieurs des fonctions détaillées en Sections 2.1 et 2.2.
2. Pour chacune des fonctions proposées ci-après, indiquer sa complexité en temps au pire, en détaillant vos arguments **dans le rapport**. On attend ici une complexité sous la forme $O()$ ou $\Theta()$, en étant bien entendu le plus précis possible.

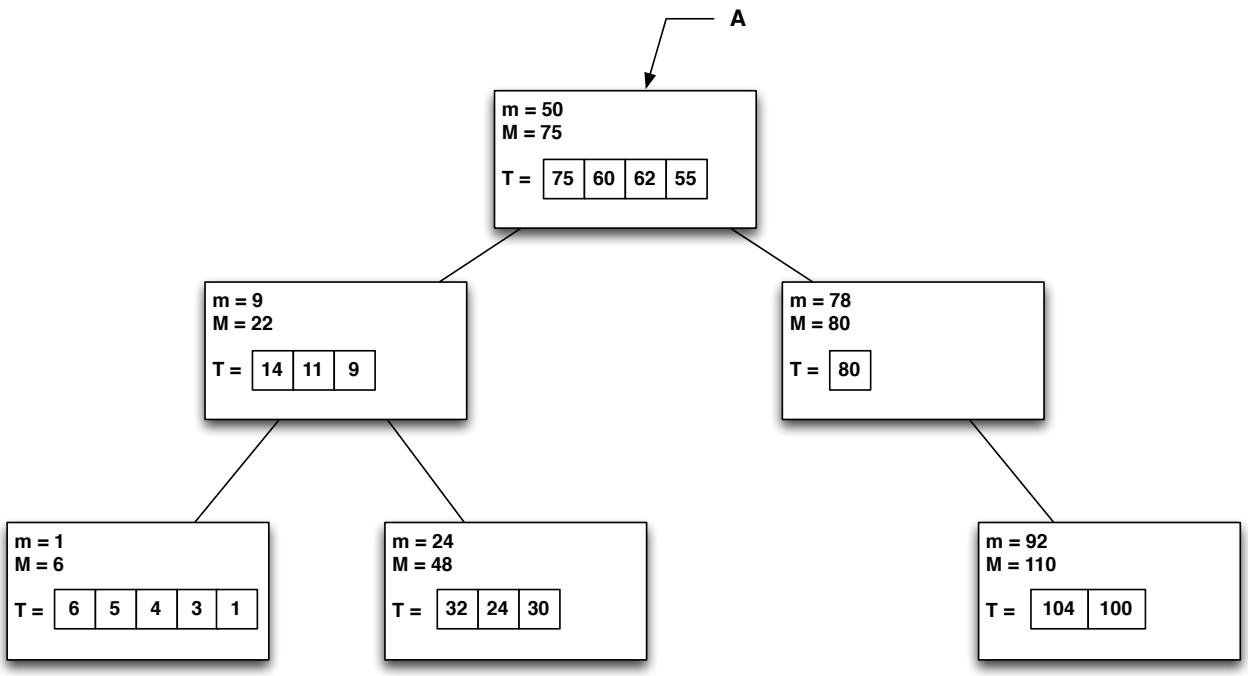

FIGURE 1 – Exemple d'un ABRTB

2.1 Génération, sauvegarde et affichage des ABRTBs

1. (**Fichier vers ABRTB**) Écrire une fonction permettant de créer un ABRTB A à partir d'un fichier F donné. Chaque ligne du fichier est une chaîne $m : M\,;\, x_1 : x_2 : ... : x_k$ telle que $x_1\, x_2\, \ldots\, x_k$ est le résultat du parcours du TB T. D'autre part, la lecture des entiers m des lignes de F, de haut en bas, donne le parcours préfixe de A.

   Par exemple, le contenu du fichier correspondant à l'ABRTB de la Figure ci-dessous est :

   ```
   50:75;75:60:62:55
   9:22;14:11:9
   1:6;6:5:4:3:1
   24:48;32:24:30
   78:80;80
   92:110;100:104
   ```

2. (**ABRTB vers fichier**) Écrire une fonction qui effectue la sauvegarde d'un ABRTB dans un fichier dont le nom et l'emplacement sont fournis par l'utilisateur.
3. (**Affichage à l'écran**) Écrire une fonction permettant d'afficher un ABRTB à l'écran, c'est-à-dire simplement afficher à l'écran ce que contiendrait le fichier correspondant à A.
4. (**ABRTB aléatoire**) Écrire une fonction qui prend en argument deux entiers p et q et qui crée un ABRTB A à p nœuds, remplis avec des données générées aléatoirement telles que $\min(m) = 1$, $\max(M) = q$ et tel que chacun des T possède au moins un élément.
5. (**Vérification**) Écrire une fonction qui vérifie si un ABRTB construit à partir d'un fichier donné en argument est correct, c'est-à-dire que :
   (a) A est un ABR (sur les valeurs de m)
   (b) tous les intervalles $[m; M]$ des nœuds de A sont disjoints
   (c) tous les tableaux T des nœuds de A sont des max-tas binaires contenant des éléments compris entre m et M.

2.2 Manipulation des ABRTBs

1. (**Recherche d'un entier**) Écrire une fonction de recherche d'un entier x dans un ABRTB A. Cela suppose de trouver le nœud v de A dans lequel x se trouve (si x est effectivement dans l'ABRTB). Si x est trouvé, on affiche à l'écran dans quel intervalle $[m; M]$ de l'ABRTB il se situe. Sinon, on affiche à l'écran dans quel cas on se trouve : soit (a) aucun intervalle ne contient x, soit (b) un des intervalles contient x (et on indiquera lequel), mais le TB T correspondant ne contient pas x.
2. (**Suppression d'un entier**) Écrire une fonction de suppression d'un entier x d'un ABRTB A. Cela suppose de trouver le nœud v de A dans lequel x se trouve (si x est effectivement dans l'ABRTB) et de supprimer x du TB T correspondant. Si x existe dans T, on affiche à l'écran le nouvel ABRTB après suppression. Si x n'existe pas dans l'ABRTB, on affiche à l'écran dans quel cas on se trouve : soit (a) aucun intervalle ne contient x, soit (b) un des intervalles contient x (et on indiquera lequel), mais le TB T correspondant ne contient pas x.
3. (**Insertion d'un entier**) Écrire une fonction d'insertion d'un entier x dans un ABRTB A. Cela suppose de trouver le nœud v de A dans lequel x doit être inséré. Si aucun intervalle existant ne contient x, l'insertion ne peut pas se faire, et on l'indique par un affichage à l'écran. Sinon, on insère x dans le TB T correspondant et le nouvel arbre est affiché à l'écran.
4. (**ABRTB vers ABR**) Écrire une fonction qui, étant donné un ABRTB A, construit un ABR $\mathcal{A}$ contenant tous les éléments des TB T contenus dans A (et aucun élément supplémentaire). Après création de $\mathcal{A}$, on affiche à l'écran les suites $S_P(\mathcal{A})$ et $S_I(\mathcal{A})$ correspondant respectivement aux parcours préfixe et infixe de $\mathcal{A}$.
5. (**ABRTB vers ABR des k-ièmes plus grands éléments**) Écrire une fonction qui, étant donné un ABRTB A et un entier k, construit un ABR $\mathcal{A}$ contenant les k-ièmes plus grands éléments des TB T de A. Si un TB T ne possède pas de k-ième plus grand élément, ce TB sera alors ignoré dans la construction de $\mathcal{A}$. Après création de $\mathcal{A}$, on affiche à l'écran les suites $S_P(\mathcal{A})$ et $S_I(\mathcal{A})$ correspondant respectivement aux parcours préfixe et infixe de $\mathcal{A}$.