Université le Havre Normandie
Lazaar El Mahdi
Anne universitaire : 2023/2024

Rapport ARN/ABR

Université le Havre Normandie
Lazaar El Mahdi
Anne universitaire : 2023/2024

Contenu

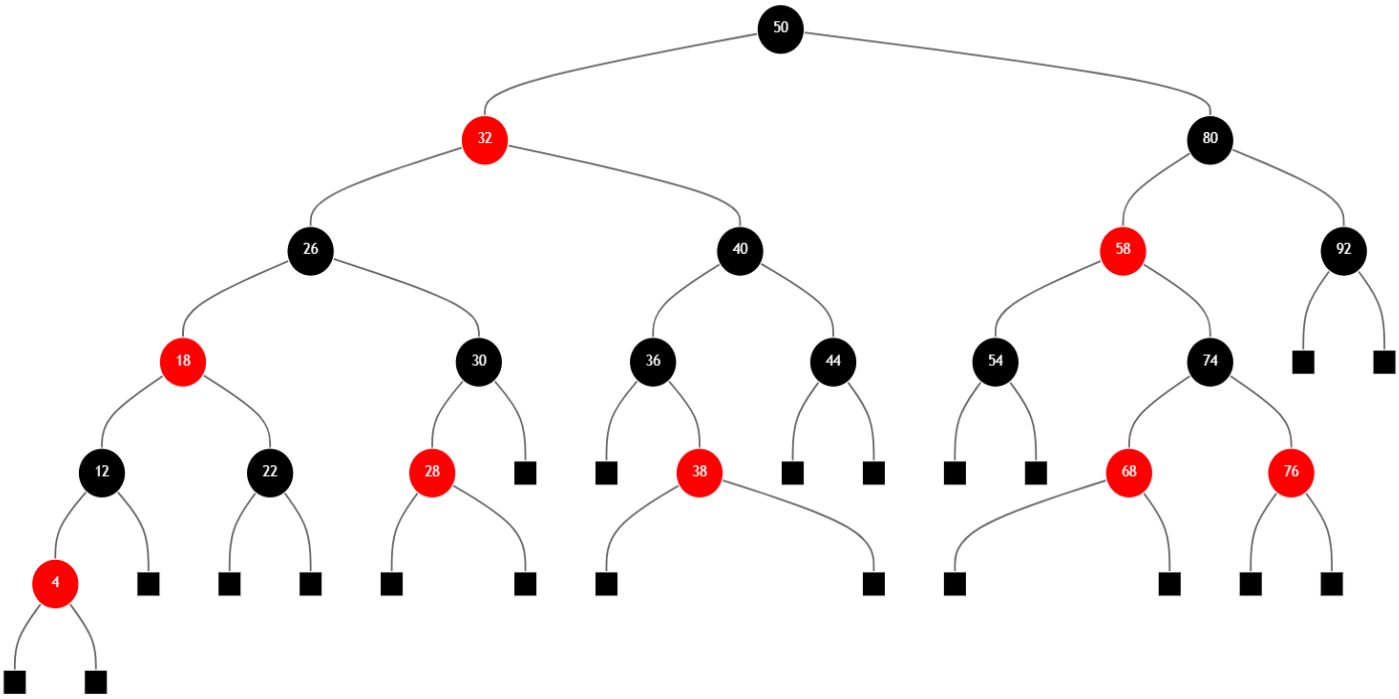
Université le Havre Normandie
Lazaar El Mahdi
Anne universitaire : 2023/2024

1. Les arbres rouges noir (ARN)

1.Définition

Un arbre rouge noir (**ARN**) est un type particulier d'arbre binaire de recherche(**ABR**), ainsi c'est un arbre équilibré où **h=O(log n)**.Un **ARN** est caracterisé par une racine noir et chauque noeud soit rouge soit noir, les feuilles sont noires, si un noeud est rouge alors ses deux fils sont noirs,pour chaque nœud, tous les chemins le reliant à des feuilles contiennent le même nombre de nœuds noirs.ceci permet d'avoir une complexité temporelle plutot efficace lors des opérations de suppression, d'insertion et de recherche.

2. Exemple



3. L'ajout d'un élément

On commence le processus d'ajout d'un élément dans un arbre rouge noir par l'ajout de la clé de l'élément dans l'arbre comme dans un arbre binaire de recherche standard,puis colorer le nouveau nœud en rouge.

Une fois que le nouveau élément est inséré, on doit vérifier que les règles d'un arbre rouge noir (**ARN**) sont respectés, sinon il faut effectuer des opérations de rotations, de changement de couleur.Le type d'ajustement à effectuer sur un arbre rouge-noir pour restaurer ses propriétés dépend du nœud inséré.

1er cas

Si le nouveau nœud inséré est sans parent (il est a la racine de l'arbre), alors il doit être coloré en noir.

2eme cas

Si le parent du nœud inséré est noir et le nouveau nœud est en rouge, alors l'arbre respecte toujours les propriétés des arbres rouge-noir (**ARN**).

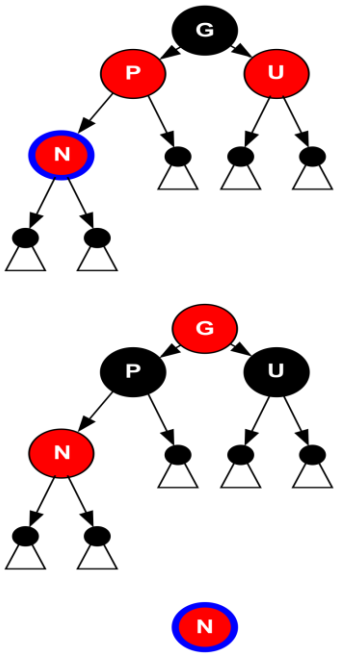
3eme cas

Si le parent du nœud inséré est en rouge :

Donc le changement qu'on doit effectuer dépend du frère du parent du nœud insérer, il existe deux cas :

Si le frère du parent du nœud insérer est rouge :

Donc le grand parent (le parent du parent du nœud insérer) est coloré en rouge, alors le parent et son frère son coloré en noir.



U : C'est l oncle

P : Parent

N : Le nouveau nœud.

Si le frère du parent du nœud insérer est noir :

Alors des rotations spécifiques doivent être effectuées. Ces rotations dépendent de la disposition du nœud par rapport à son parent et son grand-parent. Ces modifications ont pour but de rétablir l'équilibre de l'arbre.

La procédure de correction ne se termine pas toujours immédiatement. Dans le cas 3, le changement de couleur du grand-parent de noir à rouge nécessite une vérification supplémentaire, débutant à partir du grand-parent.

4. La suppression d'un élément

Pour supprimer un élément dans un arbre rouge-noir, on commence par supprimer l'élément comme dans un arbre binaire de recherche. Si le nœud supprimé était rouge, aucune action n'est nécessaire. Si le nœud supprimé était noir, il est possible que l'arbre ne soit plus équilibré. Dans ce cas, on effectue des ajustements pour rétablir l'équilibre.

1er cas

Dans le premier cas simple, si le nœud supprimé était rouge, il n'y a pas de problème car les propriétés de l'arbre sont toujours respectées. Par contre, si le nœud supprimé était noir, cela peut créer un déséquilibre dans l'arbre.

2eme cas

Dans le deuxième cas simple, si le nœud supprimé avait un fils rouge, on peut simplement colorier ce fils en noir pour rétablir la hauteur de l'arbre.

3eme cas

Dans le troisième cas simple, si le nœud supprimé est la racine et qu'il n'a pas de fils rouges, alors la hauteur noire de l'arbre a diminué de 1. Cela ne pose pas de problème car toutes les propriétés des arbres rouge-noirs sont toujours respectées.

4eme cas

Le dernier cas, plus complexe, se produit lorsque le nœud supprimé est noir, n'est pas la racine et n'a pas de fils rouges. Dans ce cas, il y a un déséquilibre dans l'arbre, car la hauteur noire des feuilles en dessous du fils est inférieure de 1 à celle des autres nœuds.

5. Recherche d'un élément :

La recherche dans un arbre rouge-noir est importante car elle permet de trouver un élément tout en conservant les propriétés et l'équilibre de l'arbre. Voici comment fonctionne la recherche dans la classe **ARN** :

Étape d'initialisation :

On commence par initialiser un nœud e à la racine de l'arbre. C'est à partir de ce nœud que la boucle de recherche démarre.

Boucle de recherche :

La boucle continue tant que le nœud e n'est pas vide. Si e devient vide, cela signifie que l'élément recherché n'est pas présent dans l'arbre, et la méthode retourne vide.

Comparaison des clés :

À chaque itération, la procédure compare la clé de l'élément recherché avec la clé du nœud actuel. Si les clés sont égales, la méthode retourne le nœud actuel. Sinon, la méthode descend dans la branche appropriée.

Si la boucle se termine sans trouver l'élément recherché, la méthode retourne le nœud vide, ce qui signifie que l'élément n'est pas présent dans l'arbre.

2. Modelisation d'ARN

Pour modéliser un arbre rouge noir, nous avons utilisé une classe Nœud qui possède quatre attributs : la couleur, le parent, le fils gauche et le fils droit.Les feuilles de l'arbre ne pointent pas vers null mais vers un nœud vide (la sentinelle) de couleur noir, Cela facilite la mise en œuvre de l'algorithme de suppression.

1. Classe generique ARN

Les attributs de la classe **ARN** :

Noeud racine : Représente le nœud racine de l'arbre.
• Int taille : Indique le nombre d'éléments dans l'arbre.
• Comparator< ? super T > cmp : Comparateur utilisé pour définir l'ordre des éléments.
• Noeud noeudVide : Noeud vide avec une couleur noire.

La classe interne Noeud représente les nœuds de l'arbre et possède les attributs suivants :
• T clé : Contenu du nœud.
• Noeud gauche : Fils gauche du nœud.
• Noeud droit : Fils droit du nœud.
• Noeud père : Père du nœud.
• Color couleur : Couleur du nœud

2. Constructeurs

La classe ARN possède trois constructeurs :

- Un constructeur par défaut qui crée un arbre vide avec le comparateur par défaut.

- Un constructeur qui prend un comparateur en paramètre pour définir l'ordre des éléments.
- Un constructeur par copie qui crée un arbre à partir d'une collection existante.

En ce qui concerne les méthodes implémentées, la classe **ARN** dispose de méthodes pour :

- Ajouter, rechercher et supprimer des éléments dans l'arbre tout en respectant les propriétés des arbres Rouge-Noir.
  Obtenir une représentation textuelle de l'arbre pour faciliter l'affichage (toString)

3. Complexité des méthodes

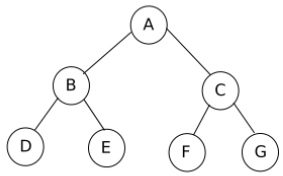
rechercher() , minimum() ,ajouterCorrection et enleverCorec exécutent en temps logarithmique ,leur complexité est de O(log n)

4. Arbre binaire de recherche

Un arbre binaire de recherche est une structure de données arborescente où chaque nœud a au plus deux enfants, appelés fils gauche et droit.
Ils sont de degre inferieur ou egal a 2.

1. Arbre binaire complet :

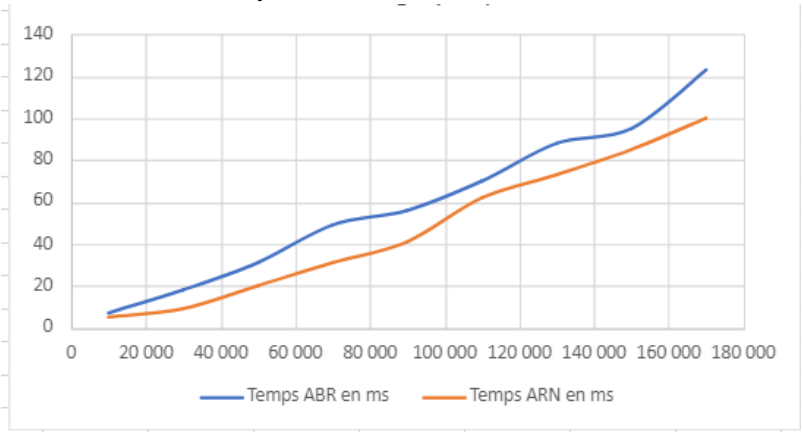
Un arbre binaire *complet* est un arbre dans lequel tous les nœuds sont de degré 2 sauf ceux du dernier niveau.



Arbre 2

3. Comparaison des performances de l'ABR et de l'ARN

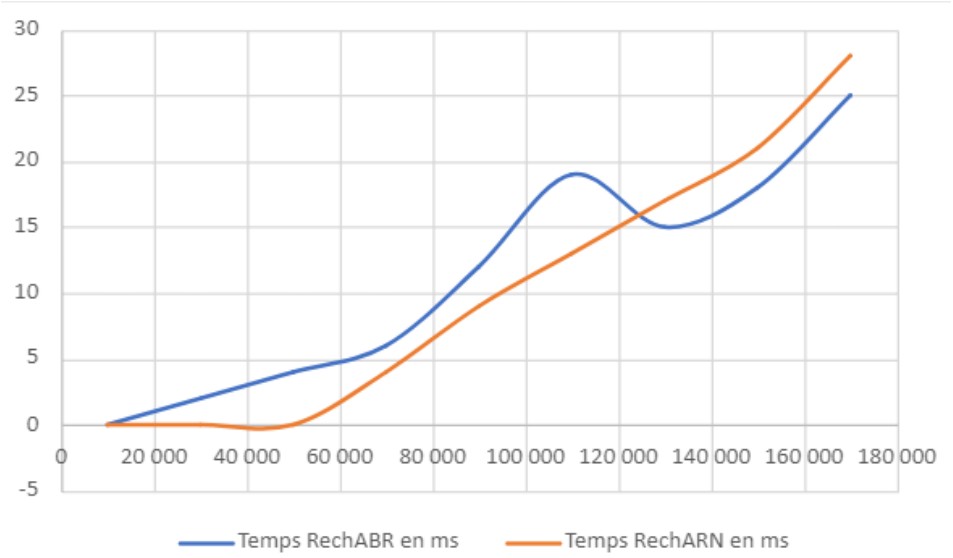
Les tests sur des arbres de taille comprise entre 30 000 et 170 000 montrent que les arbres Rouge-Noir sont plus performants que les arbres binaires de recherche pour la construction. La différence de performance entre les deux types d'arbres n'est significative qu'à partir de valeurs très élevées.



Légende : Mesure de temps de construction des arbres rouges-noirs et arbre binaire de recherche.

1. Mesure du temps de recherche

En comparant les deux courbes on remarque que : à partir de très grandes valeurs les deux méthodes de recherche pour **l'ABR** et **l'ARN** sont un peu près les mêmes mais pour des petits valeurs la recherche pour un nœud avec une clé spécifique est plus efficace pour **l'ARN**.



Légende : Mesure de temps de recherche pour une clé pour des arbres rouges-noirs et arbre binaire de recherche.

2. Le cas défavorable

Le cas le plus défavorable pour un **ABR** c'est lorsqu'on ajoute les clés dans l'arbre dans un ordre croisant du plus petit au plus grand.D'après le graphe et le tableau ci-dessous que **l'ABR** prend plus de temps pour construire l'arbre avec des clés ordonnées qu'avec des clés aléatoires.Par conséquent on aura une chaine linéaire des nœuds.

Taille	Temps en
100	0
150	1
200	3
250	16
300	15
350	16
400	16
450	18
500	18