TD n°18 – ABR et Arbres rouge-noir

1 ABR

On va implémenter les opérations des ABR en Ocaml. On définit les arbres binaires de la manière habituelle, `type 'a arbre = Vide | N of 'a arbre * 'a * 'a arbre`.

1. Rappeler la fonction `recherche:'a arbre-> 'a->bool` qui renvoie **`true`** si un ABR contient une certaine étiquette e et **`false`** sinon. On devra garantir une complexité en $O(h(a))$ pour cette fonction.
2. Écrire une fonction `supprime:'a arbre-> 'a->'a arbre` qui supprime une étiquette e d'un ABR (on suppose que e apparait au plus une fois). Le résultat doit être un ABR et la fonction doit avoir une complexité en $O(h(a))$.
3. Écrire une fonction `est_abr:'a arbre->bool` qui vérifie si un arbre est un ABR.

2 Arbres rouges-noirs

On rappelle qu'un arbre rouge-noir est un arbre binaire de recherche, dont les noeuds sont colorés, selon les propriétés suivantes :

- La racine est noire.
- Un nœud rouge ne peut pas avoir d'enfant rouge.
- Un chemin de la racine à n'importe quelle feuille contient toujours le même nombre de noeuds noirs.

On peut remarquer que la dernière propriété est équivalente à "pour tout noeud, tout chemin de ce noeud à une feuille a le même nombre de noeuds noirs".

On appelle hauteur noire le nombre de noeuds noirs séparant la racine de chaque feuille. La hauteur noire d'un arbre vide est 0.

1. Construction

4. Comment rechercher une clé dans un arbre rouge-noir ?
5. Pour l'insertion, comment pourrait-on faire ? Pourquoi ajouter un noeud noir est une mauvaise idée ?

L'idée est donc d'ajouter une feuille rouge au bon endroit.

6. Trouver un exemple d'arbre rouge-noir tel qu'un ajout avec cette méthode fait qu'il ne vérifie plus la définition.
7. Quelles propriétés des arbres rouges-noirs peuvent être violées par la méthode d'ajout proposée ?

Votre réponse à la question précédente devrait être la première et la deuxième. Si ce n'est pas le cas, réfléchissez encore à des exemples.

8. Si notre ajout viole la première règle, comment corriger le problème de manière triviale ?

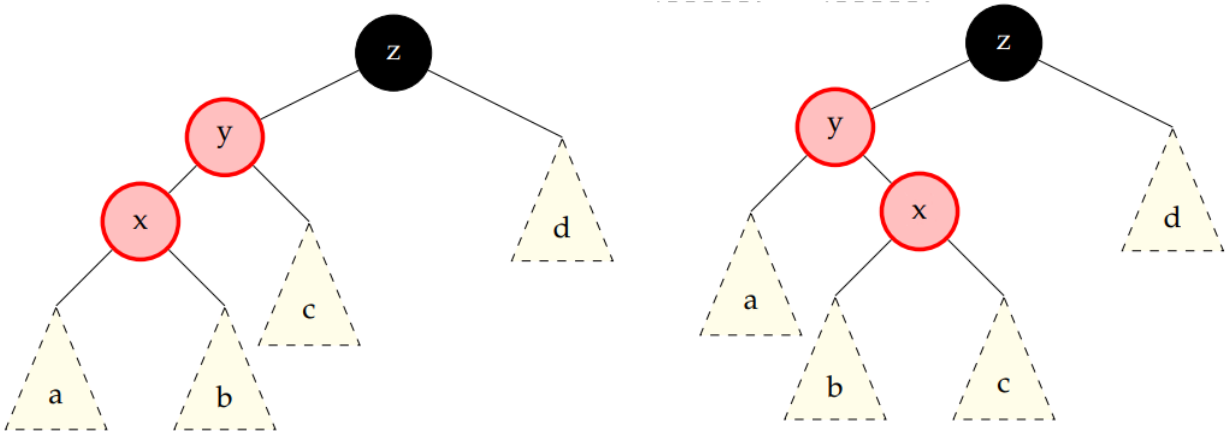
On suppose maintenant que la règle violée est la deuxième (et pas la première) : il existe un et un seul noeud rouge y ayant un fils rouge x, et y n'est pas la racine de l'arbre.

Initialement x est l'étiquette qu'on veut rajouter, **mais il est important de noter que la correction qu'on va apporter se fera récursivement en déplaçant le problème, donc on ne suppose pas que x est une feuille, ni l'étiquette qu'on voulait initialement rajouter.**

9. Justifier que y a un père z et que celui-ci est noir.

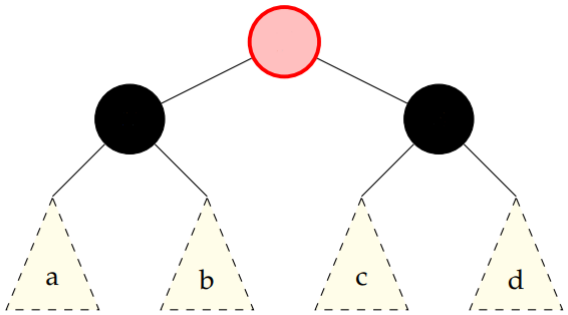
Il existe alors 4 configurations possibles selon si x est fils gauche ou droit de y et y est fils droit ou gauche de z.

Voici les configurations gg et gd.



10. Dessiner les configurations dg et dd (les deux autres possibilités).

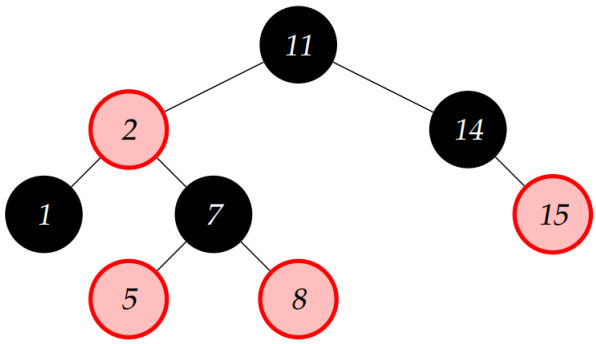
Pour rétablir les propriétés de rouge-noir dans l'arbre, on propose de transformer localement la partie posant problème en la configuration suivante (x, y et z étant les étiquettes des trois noeuds, dans un ordre qui dépend de la configuration initiale) :



11. Trouver comment on ramène les situations gg, gd, dg et dd à cette situation en effectuant uniquement des rotations gauches ou droites des arbres et des changement de couleurs sur x, y et z.
12. Justifier que cette correction permet de conserver la dernière règle, et que la deuxième propriété est vérifiée **dans le sous-arbre obtenu après rotation**.
13. La première règle est-elle préservée ? Si non, comment corriger le problème ?

On peut potentiellement crée un problème vis-à-vis de la deuxième règle, entre y et son nouveau père. On va donc itérer la correction.

14. Justifier que ce processus termine.
15. En utilisant la méthode proposée, ajouter l'étiquette 3 dans l'arbre suivant en obtenant un arbre rouge-noir correct.



Une autre opération intéressante sur les arbres rouge-noir est la suppression, dans laquelle on n'a pas de manière simple de s'assurer que la troisième propriété reste vraie La méthode nécessite à nouveau une étude de tous les cas possibles et plusieurs méthodes de correction différentes.

2. À propos de la hauteur

16. Soit un chemin entre un sommet s et une feuille f. Montrer qu'au moins la moitié des noeuds de ce chemin sont noirs.
17. Soient deux chemins $s, u_1, u_2, \ldots, u_n$ et $s, c_1, c_2, \ldots, c_m$ d'un sommet s à deux feuilles u_n et c_m. Montrer que le plus long de ces deux chemins a une longueur au plus égale au double de celle de l'autre chemin.
18. Montrer qu'un arbre rouge-noir à n noeuds a une hauteur au plus égale à $2\lfloor \log(n+1) \rfloor$.