Arbres Binaires de Recherche
H-équilibrage et AVL

Ce document a pour objectif de vous familiariser avec une nouvelle structure de données algorithmique abstraite : les *arbres* (en anglais : *trees*). En particulier, nous y aborderons les *arbres binaires de recherche* ou *ABR* (en anglais : *binary search tree* ou *BST*) qui sont des cas spécifiques d'arbres binaires, ainsi que les arbres *AVL* (*Adelson-Velsky & Landis*) utilisant des rotations pour maintenir leur équilibre.

Pour rappel, un arbre binaire est un arbre d'arité/de degré 2, c'est-à-dire que chaque nœud dispose au maximum de 2 fils : un fils gauche et un fils droit. Il n'existe aucune boucle dans un arbre : chaque nœud n'a qu'un seul père (excepté la racine). La profondeur d'un nœud est la différence de niveau entre le nœud étudié et la racine de l'arbre. La hauteur d'un arbre est la profondeur du nœud le plus éloigné de la racine. La taille d'un arbre est le nombre de nœuds contenus dans l'arbre.

1 Équilibre et déséquilibre

L'insertion en feuille dans un ABR mène parfois à des arbres filiformes où la recherche devient une itération sur une liste chaînée, ce qui perd tout l'intérêt des ABR. Par exemple, si l'on insère en feuille dans cet ordre précis les éléments suivants : $15 - 5 - 10 - 8$, on obtient l'arbre suivant :

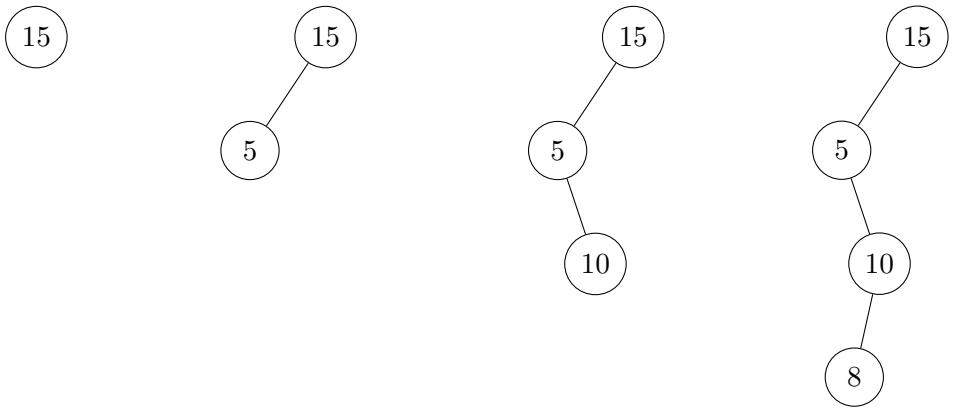


Fig.1 : ABR filiforme

Ce problème où une branche est beaucoup plus profonde que l'autre s'appelle le *déséquilibre* (en anglais *unbalancing*).
Un arbre dont les feuilles sont toutes au même niveau (donc un arbre parfait), est donc dit *équilibré* (en anglais *balanced*). Afin de pouvoir effectuer des recherches rapides, on peut vouloir faire en sorte que l'arbre s'*auto-équilibre* à chaque insertion ou suppression (en anglais *self-balance*).
Néanmoins, plusieurs critères d'équilibrage existent : les arbres *h-équilibrés* (en anglais *height-balanced trees* ou *h-balanced trees*) pour l'équilibrage en *hauteur* de chacun des sous-arbres gauche et droit, et les arbres équilibrés en *poids* (en anglais *weight-balance trees* ou w-balanced trees ou encore *WBT*) pour l'équilibrage basé sur le nombre de feuilles de chacun des sous-arbres gauche et droit.

2 Arbres AVL

Les premiers ABR automatiquement équilibrés ont été conçus par *G. Adelson-Velsky* et *E. Landis* (d'où le nom *AVL*). Ces arbres s'équilibrent automatiquement grâce aux rotations lors de l'insertion et de la suppression afin de garantir une recherche efficace. Les AVL sont équilibrés en hauteur, ils sont donc *h-équilibrés*.

2.1 Facteur d'équilibrage

La structure des AVL est la même que pour les ABR, excepté que chaque nœud dispose d'un champs supplémentaire : le *facteur d'équilibrage* (ou *balance factor* en anglais), c'est-à-dire un entier représentant la différence entre la hauteur du sous-arbre droit et la hauteur du sous-arbre gauche.

```
struct node
{
  int          key;
  struct node *left_child;
  struct node *right_child;
  int          balanced_factor;
};
```

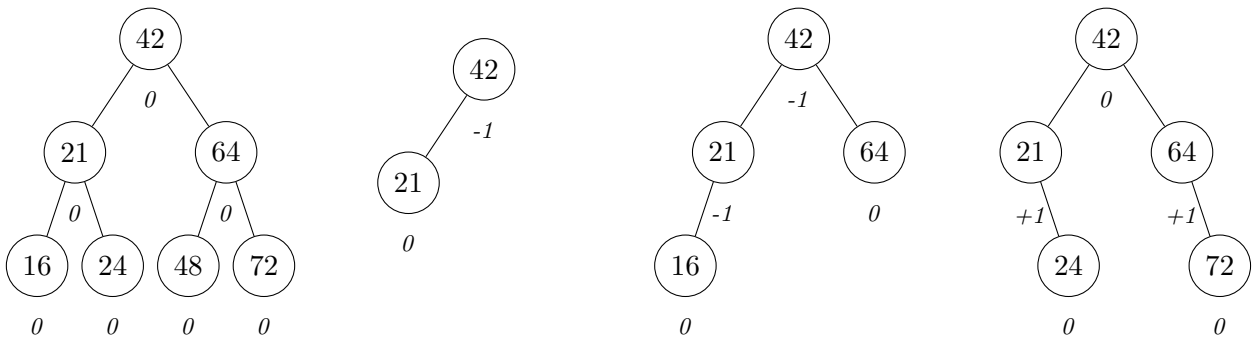
$$BF(N) = \text{hauteur(Sous-Arbre Droit}(N)) - \text{hauteur(Sous-Arbre Gauche}(N))$$

$$BF(N) = \text{hauteur}(N \to \text{fd}) - \text{hauteur}(N \to \text{fg})$$

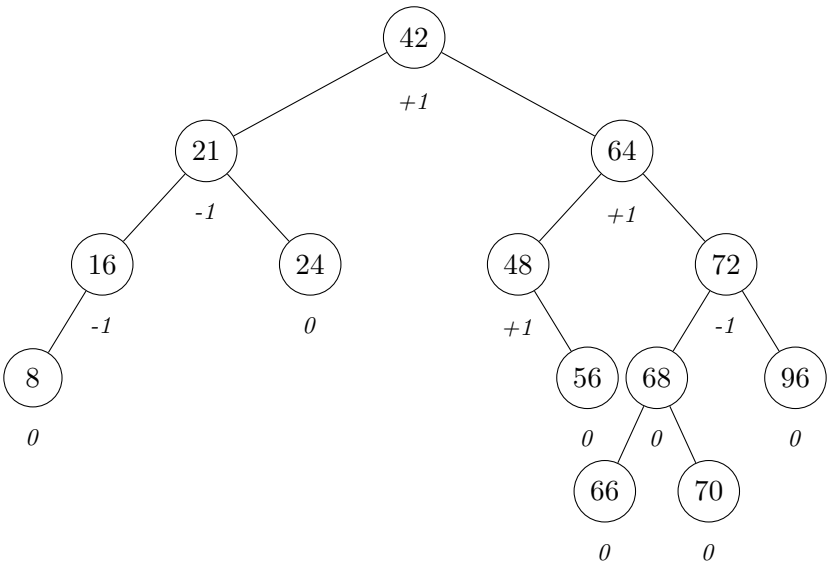
On considère qu'un nœud est équilibré lorsque son facteur d'équilibrage a comme valeurs -1, 0, ou 1 (c'est-à-dire que les sous-arbres gauche et droit du nœud ont une différence de hauteur de 1 au maximum).
À l'inverse, un nœud est déséquilibré lorsque son facteur d'équilibrage est inférieur à -1 ou supérieur à 1 (c'est-à-dire que les sous-arbres gauche et droit du nœudone une différence supérieure à 1).

Lorsque le facteur d'équilibrage est de $+1$, alors le nœud est plus lourd à droite (il manque un ou des nœuds à gauche), à l'inverse, si le facteur d'équilibrage de -1, alors le nœud est plus lourd à gauche (il manque un ou des nœuds à droite).

Voici plusieurs exemples d'arbres équilibrés avec les facteurs d'équilibres de chaque nœud :



1 2 3 4



5

Fig.2 : Exemples d'arbres binaires équilibrés avec les facteurs d'équilibre

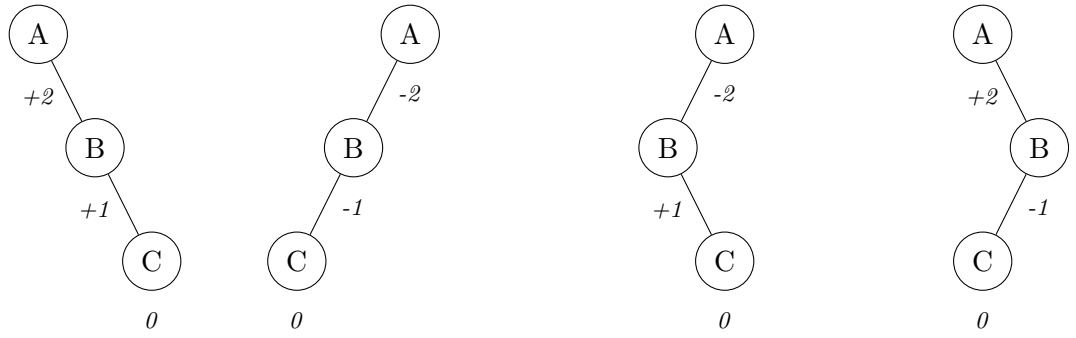
Parmi ces exemples, on constate que seule la hauteur des sous-arbres compte : on ne remonte pas les différences hauteurs telles quelles en les ajoutant ! Le but de ce facteur est de mesurer l'écart de hauteur entre les sous-arbres à partir de chaque nœud.

On remarque également que dans l'exemple 5, les nœuds 24, 56, et 66 sont sur 3 niveaux distincts, mais que la racine ne constate une différence de hauteur de 1. Ce phénomène est normal. Lors de l'insertion en feuille d'un prochain nœud, soit l'arbre rapprochera de 0 certains facteurs, soit il créera des déséquilibres de -2 ou $+2$ (par exemple en insérant en fils de 8, 56, 66, ou 70). De même, si l'on supprime certains nœuds, soit les facteurs se rapprocheront de 0, soit des déséquilibres seront créés (par exemple en supprimant 24, le nœud 21 aura un facteur d'équilibrage de -2.

La gestion de ce facteur se fait lors de l'insertion et de la suppression : on insère/supprime récursivement un nœud en feuille, puis à la remontée, on met à jour les valeurs. Les traitements pour rééquilibrer l'arbre se font lorsque la mise à jour génère un -2 ou un $+2$.

2.2 Rotations

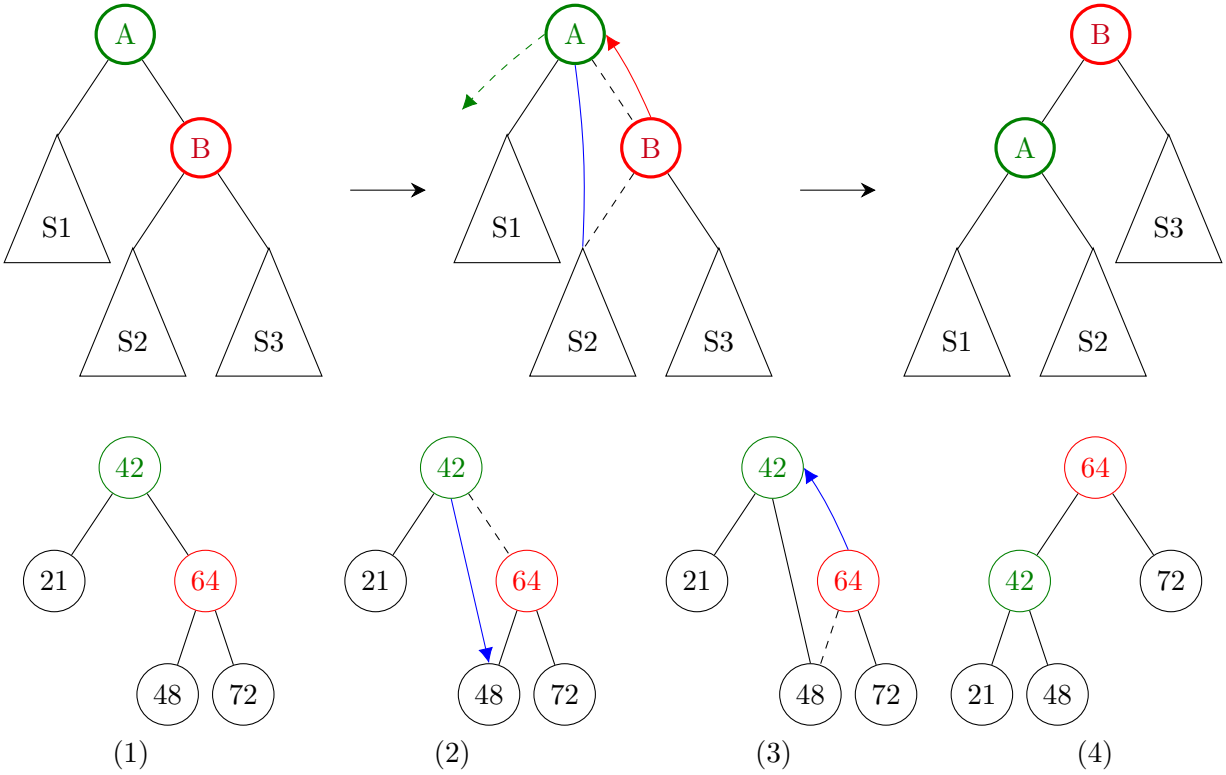
Afin de réduire le déséquilibre engendré lors des insertions et suppressions, on utilise les *rotations*. Vous avez précédemment vu les rotations *gauche* et *droite* permettant de remonter d'un niveau un nœud, mais il peut arriver que l'on utilise une combinaison de ces rotations dans certains cas, en particulier les rotations *gauche droite* et *droite gauche*. Lorsque le facteur d'équilibrage atteint 2 ou -2, on va regarder les facteurs de ses fils, et éventuellement de leurs fils, pour choisir la rotation adaptée qui rééquilibrera l'arbre.



rotation gauche rotation droite rotation gauche droite rotation droite gauche

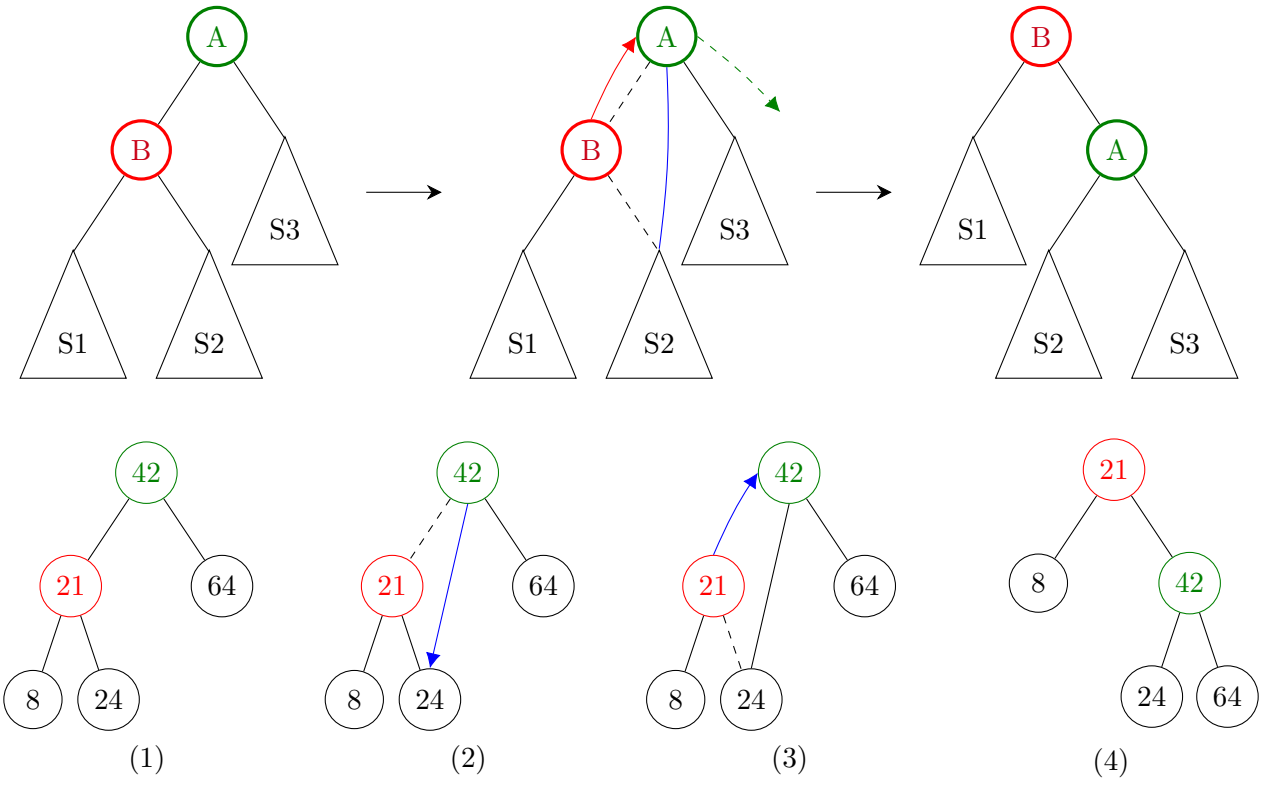
2.2.1 Rotation Gauche

La rotation gauche vise à remonter le fils droit d'un nœud (il passe donc à gauche vers la place de son père).



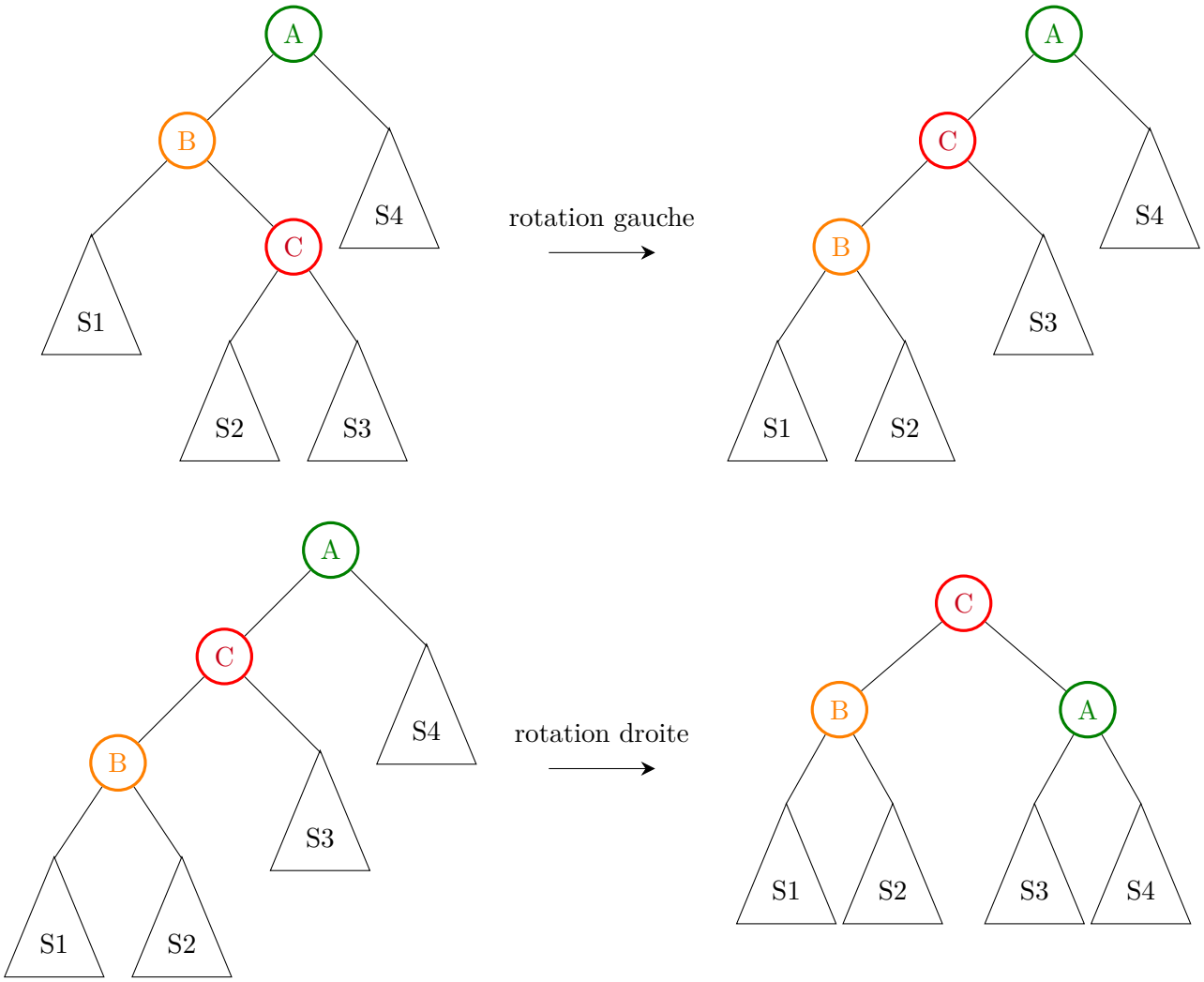
2.2.2 Rotation Droite

La rotation droite vise à remonter le fils gauche d'un nœud (il passe donc à droite vers la place de son père).

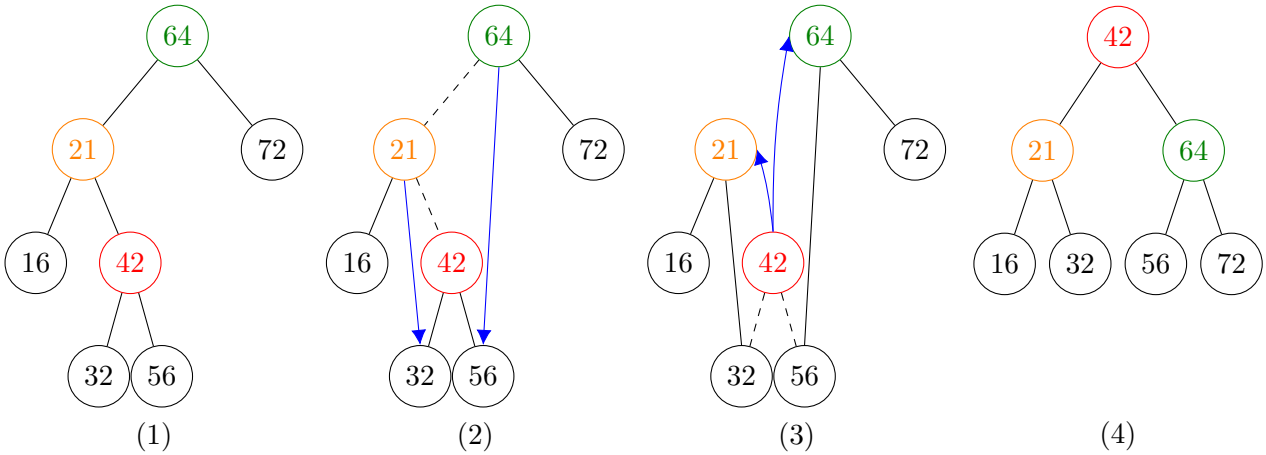


2.2.3 Rotation Gauche Droite

La rotation gauche droite vise à effectuer deux rotations successives : une rotation gauche, puis une rotation droite. À l'issue des deux rotations, on remarque que les sous-arbres ont été répartis entre les deux nouveaux fils de la nouvelle racine.

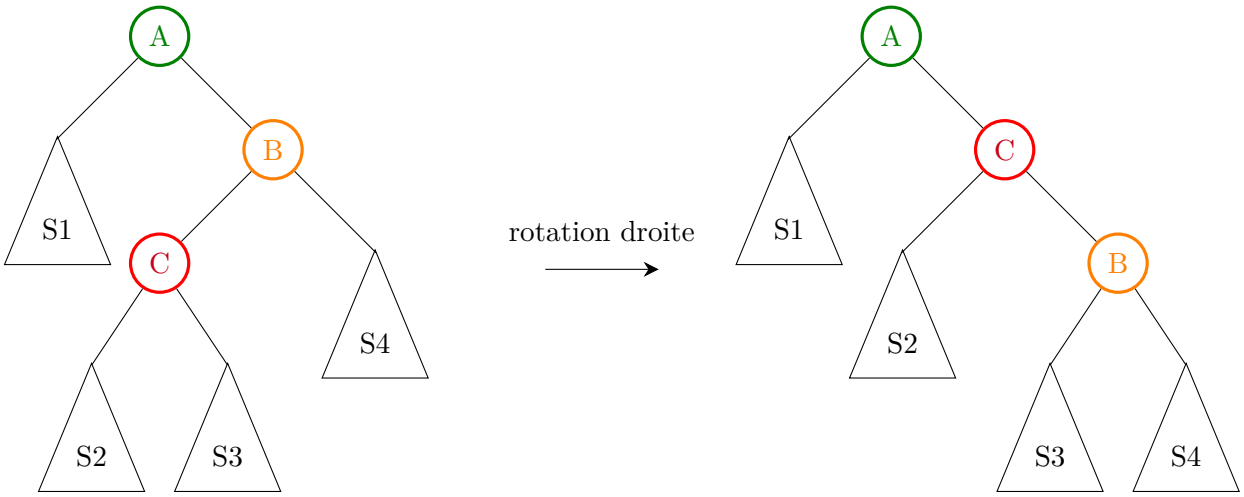


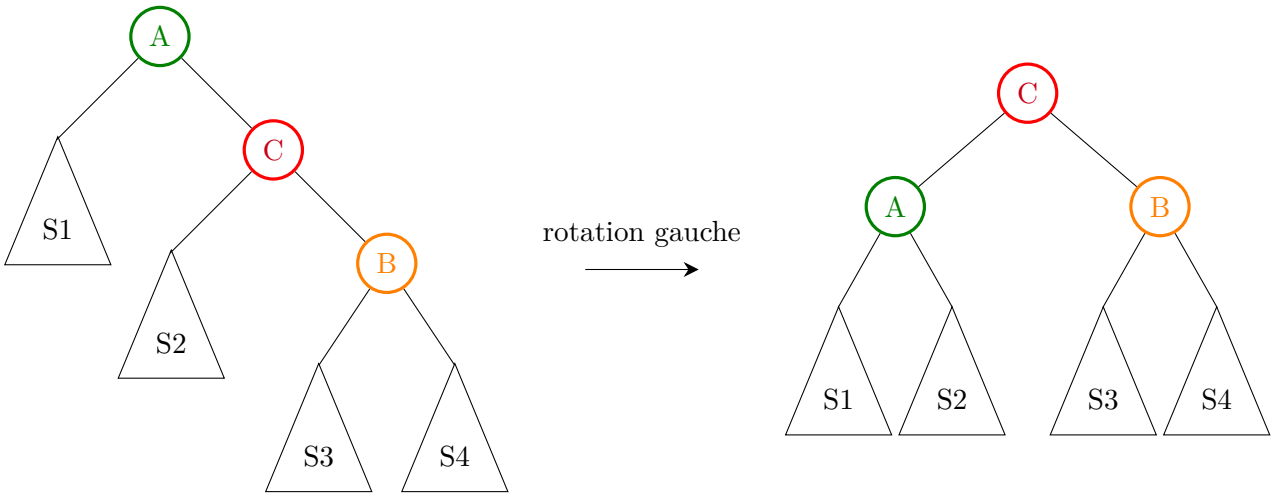
L'implémentation classique effectue l'appel successif à chacune des rotations dans le bon ordre, mais il est possible de directement réorganiser tous les nœuds (attention à ne pas oublier de mettre à jour l'éventuel parent de la racine lors de l'implémentation concrète : les rotations doubles ne se font pas toujours à la racine de l'arbre).



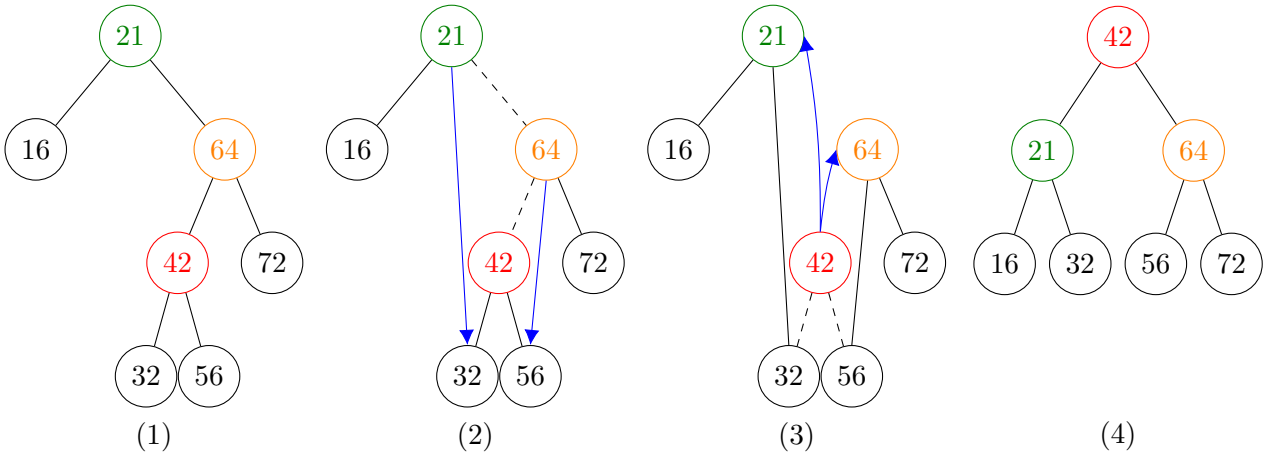
2.2.4 Rotation Droite Gauche

La rotation droite gauche vise à effectuer deux rotations successives : une rotation droite, puis une rotation gauche. À l'issue des deux rotations, on remarque que les sous-arbres ont été répartis entre les deux nouveaux fils de la nouvelle racine.





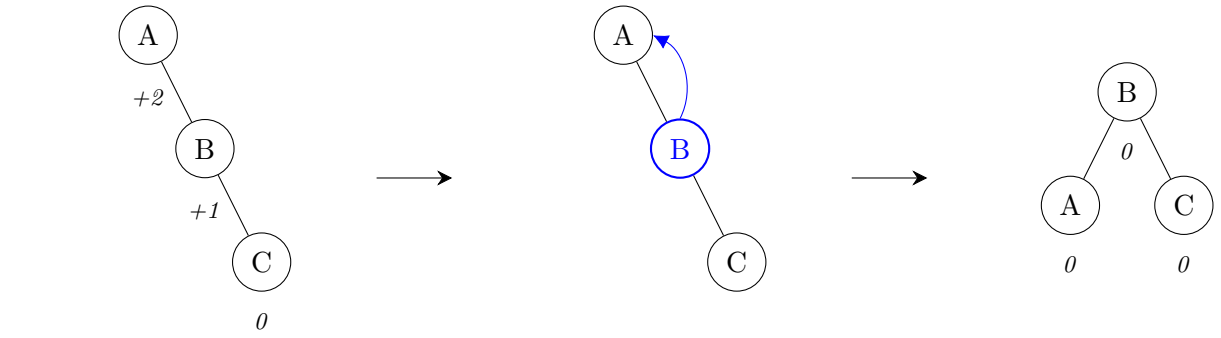
L'implémentation classique effectue l'appel successif à chacune des rotations dans le bon ordre, mais il est possible de directement réorganiser tous les nœuds (attention à ne pas oublier de mettre à jour l'éventuel parent de la racine lors de l'implémentation concrète : les rotations doubles ne se font pas toujours à la racine de l'arbre).



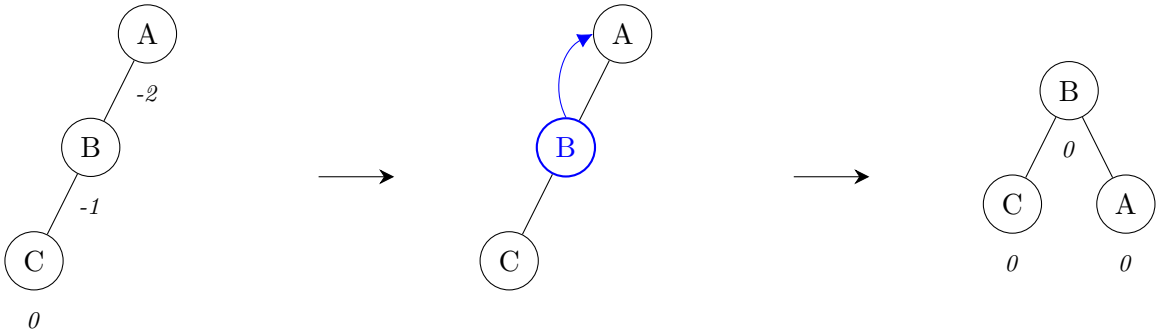
2.3 Insertion dans un AVL

L'insertion d'un nœud dans un AVL est littéralement l'insertion récursive en feuille, suivie d'une mise à jour des facteurs d'équilibrage à la remontée, avec éventuellement des rotations adaptées lorsqu'un facteur atteint 2 ou -2. Le choix de la rotation à effectuer dépend donc du cas rencontré.

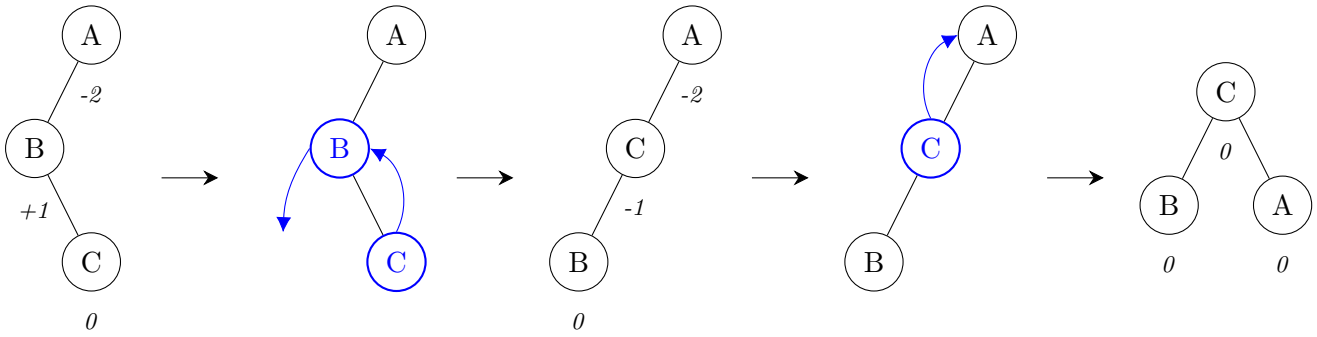
Rotation gauche



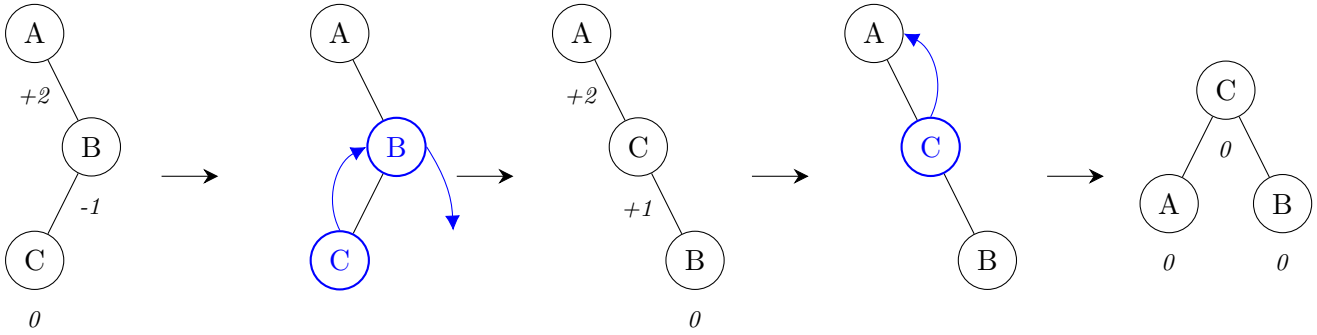
Rotation droite



Rotation gauche droite



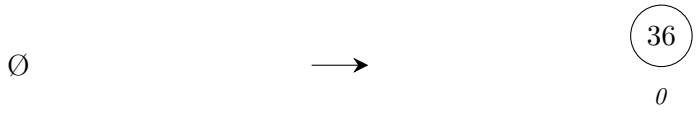
Rotation droite gauche



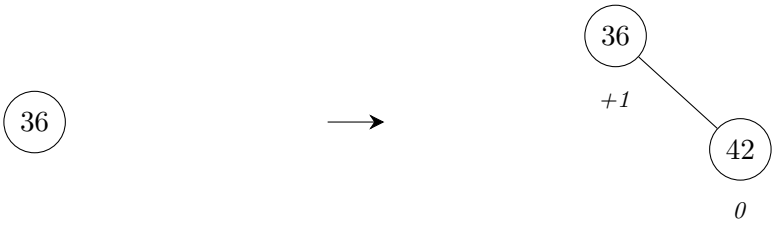
Exemple complet d'insertion dans un AVL

L'exemple suivant effectue dans cet ordre précis les insertions successives dans un AVL des éléments 36, 42, 56, 32, 28, 72, 64, 16, 21, 8.

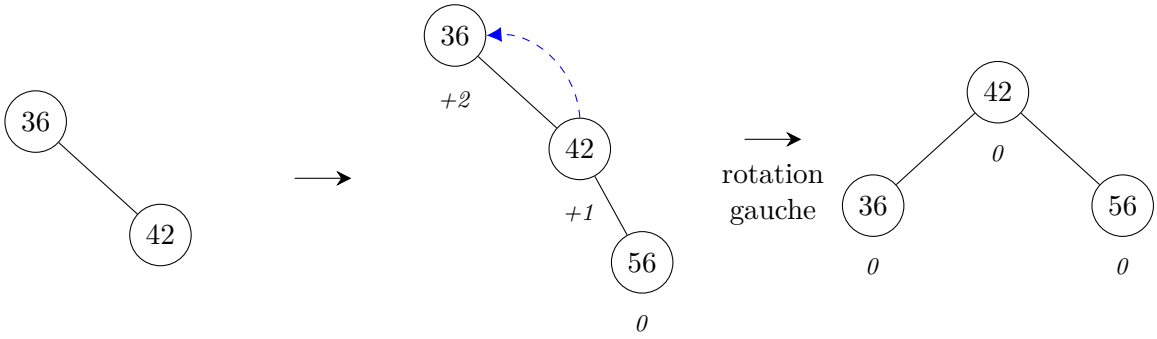
(1) Insertion de 36



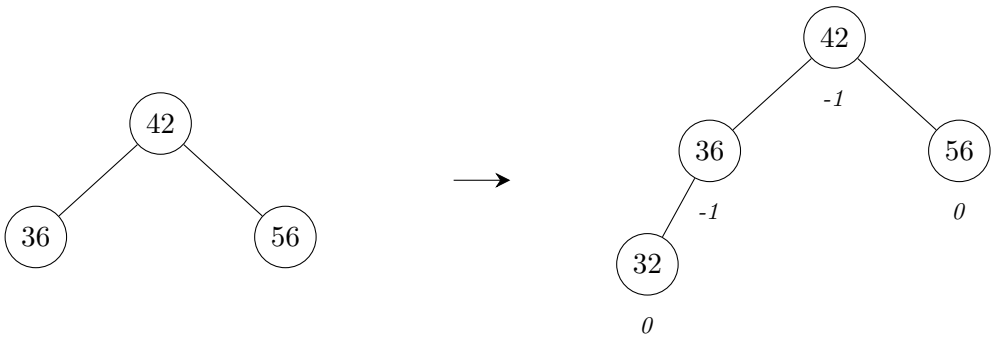
(2) Insertion de 42



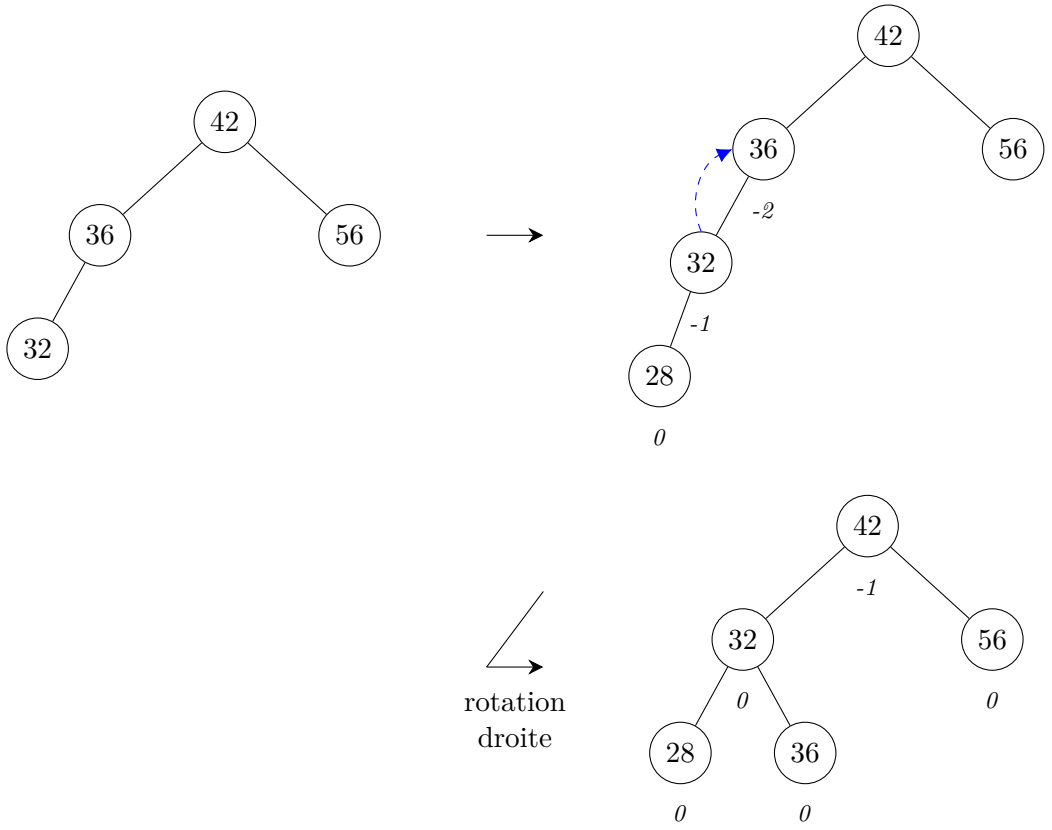
(3) Insertion de 56



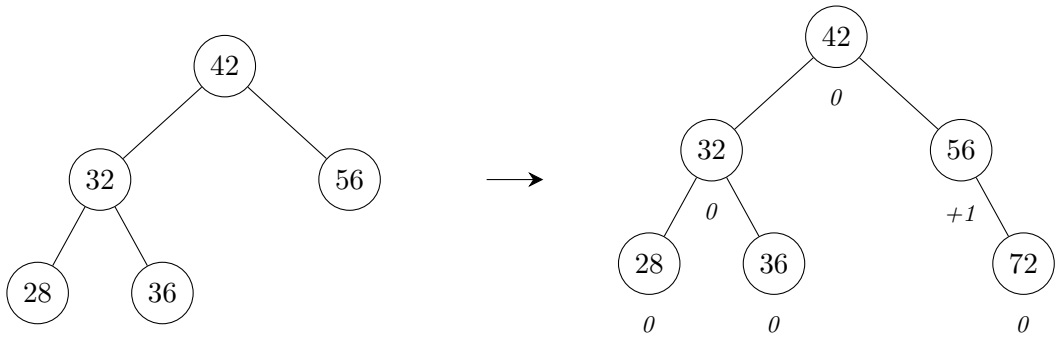
(4) Insertion de 32



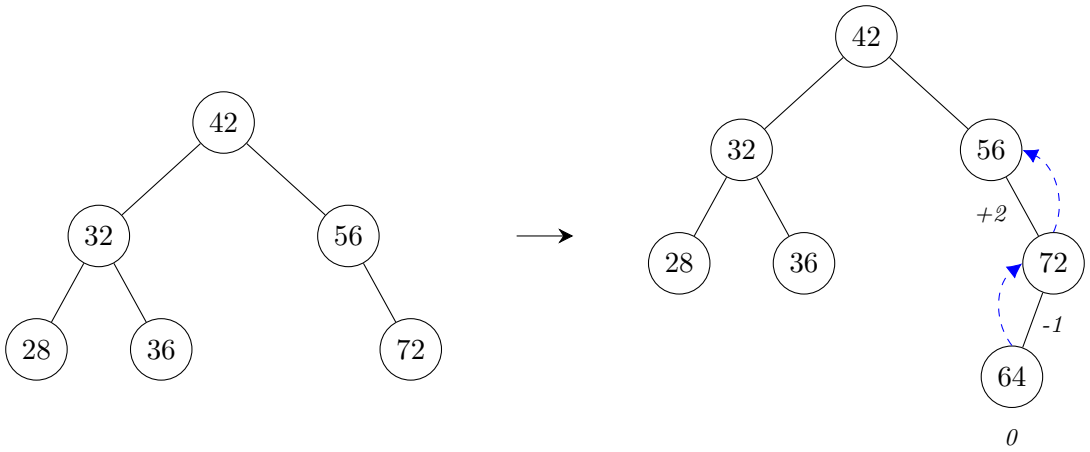
(5) Insertion de 28

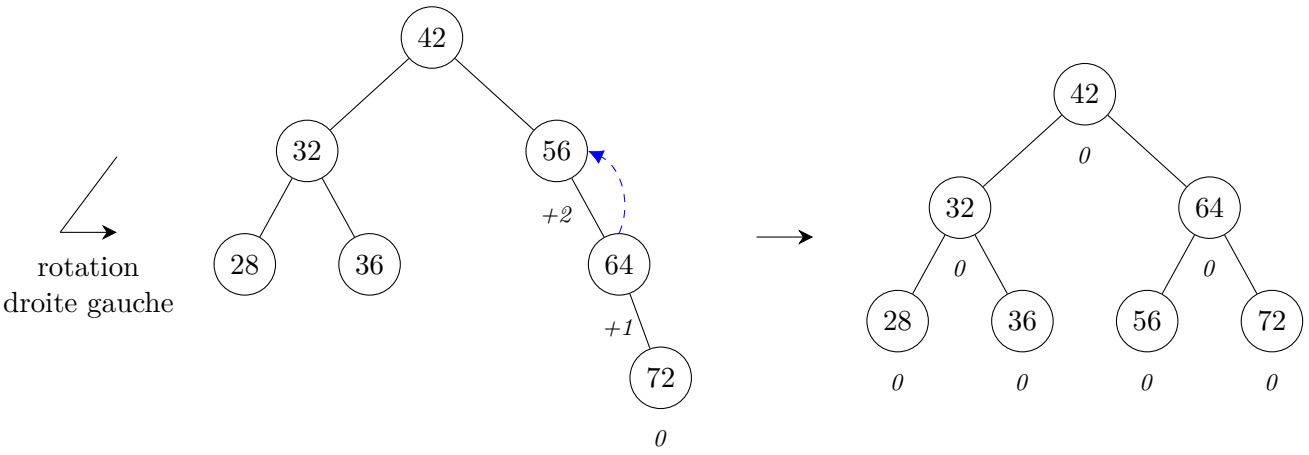


(6) Insertion de 72

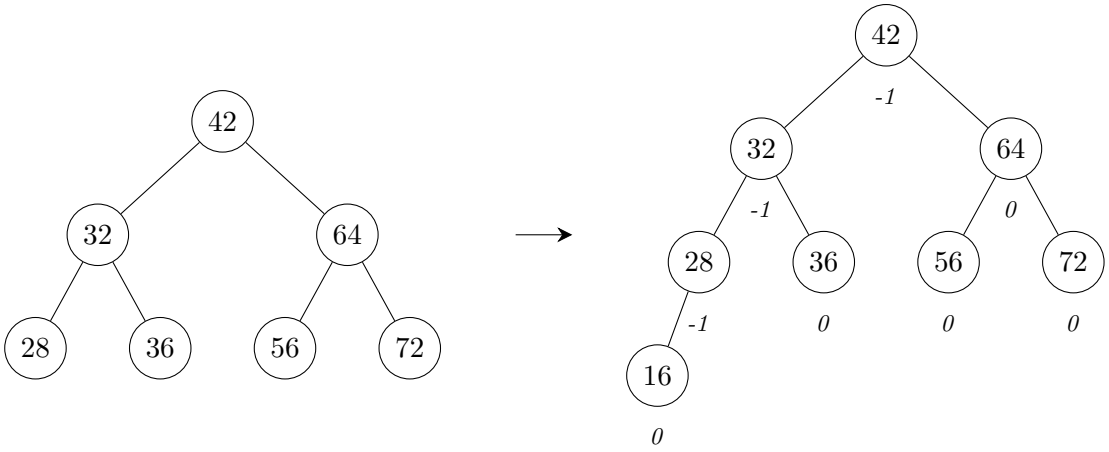


(7) Insertion de 64

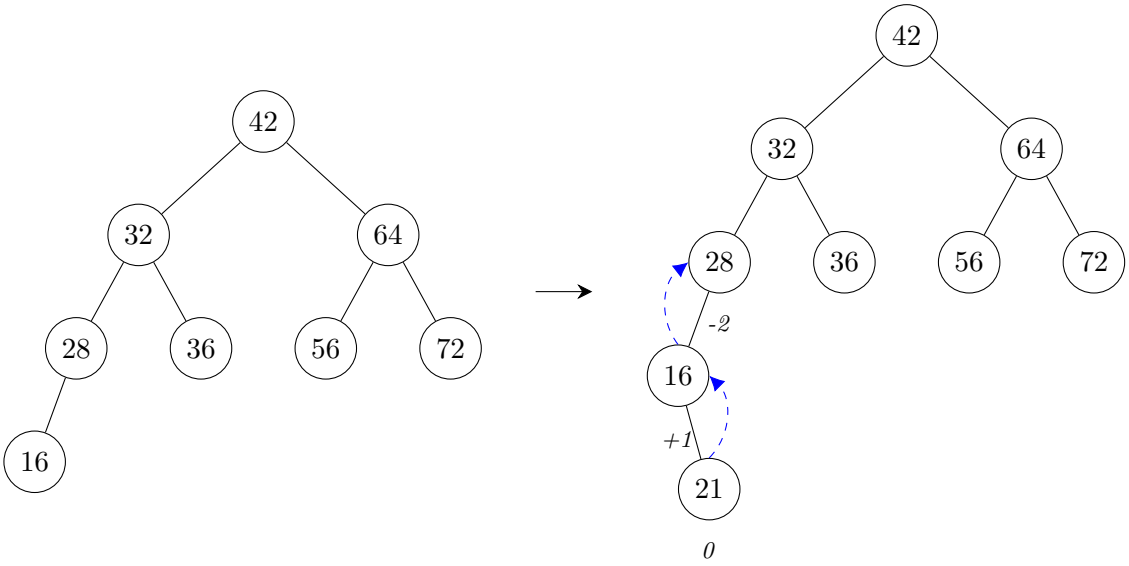


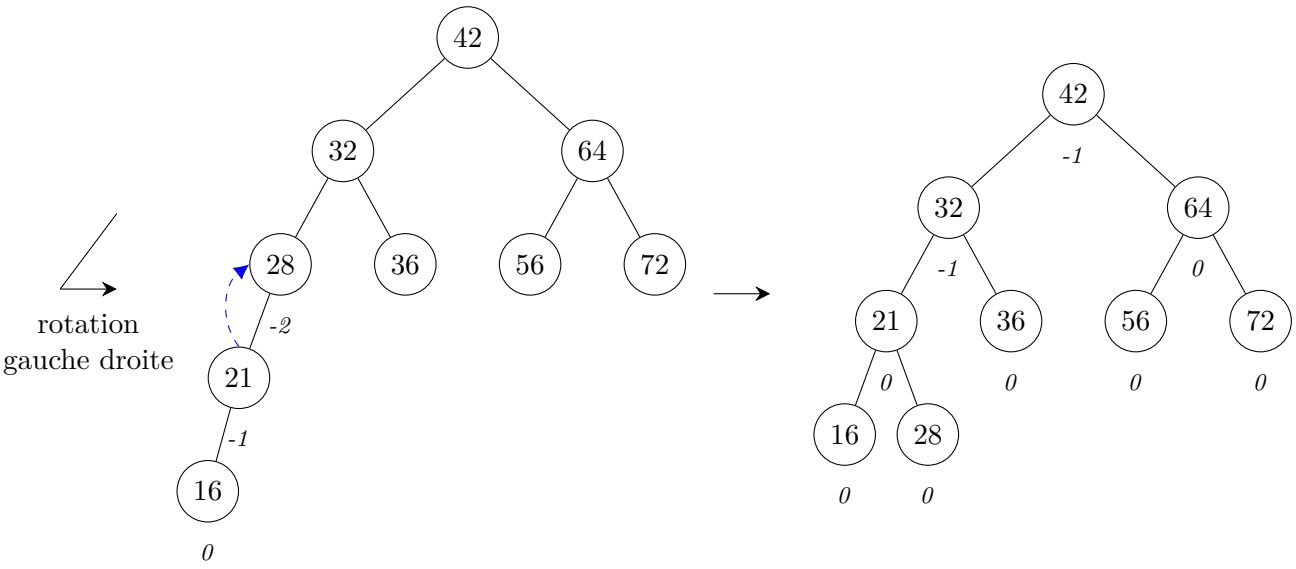


(8) Insertion de 16

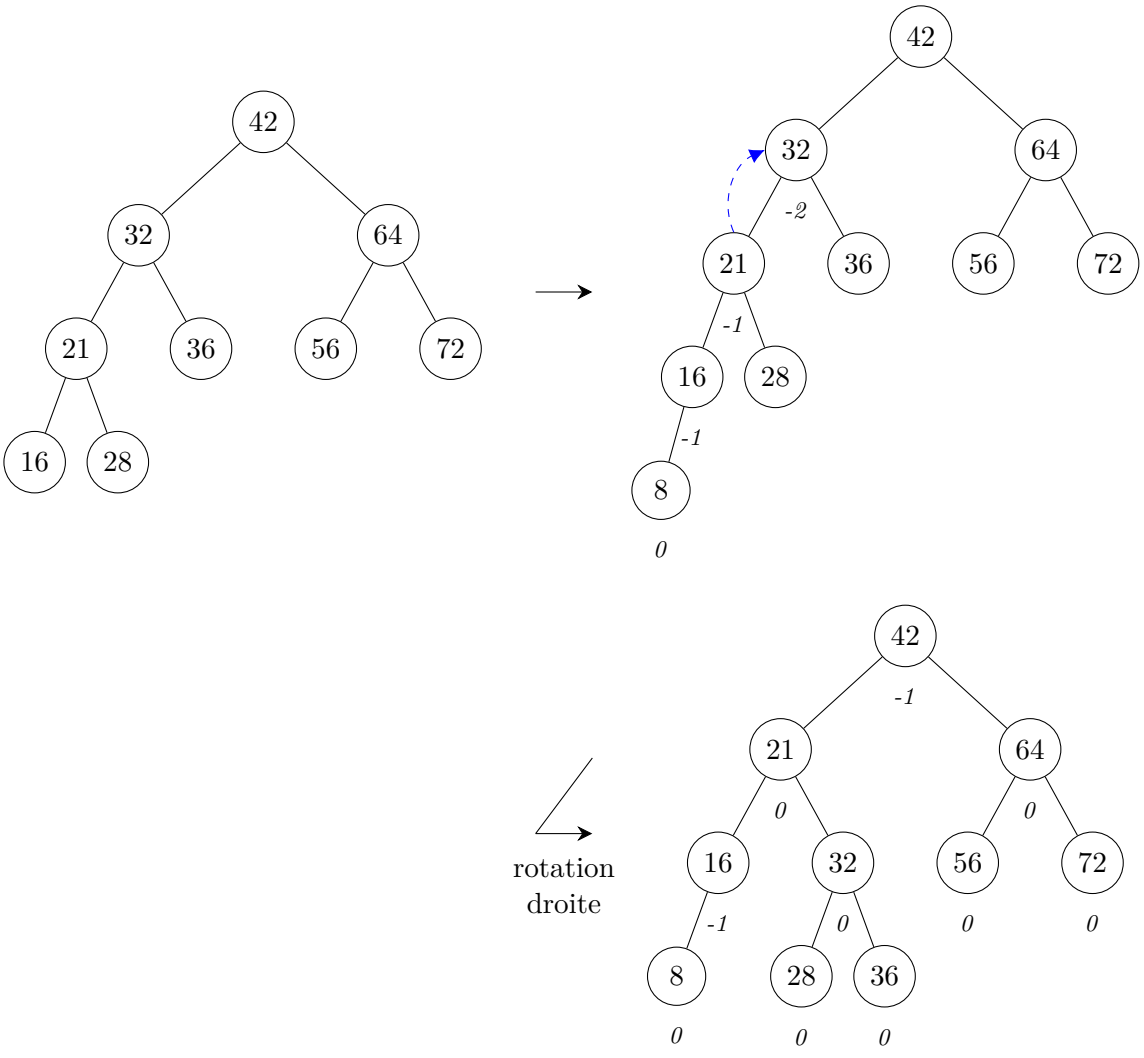


(9) Insertion de 21





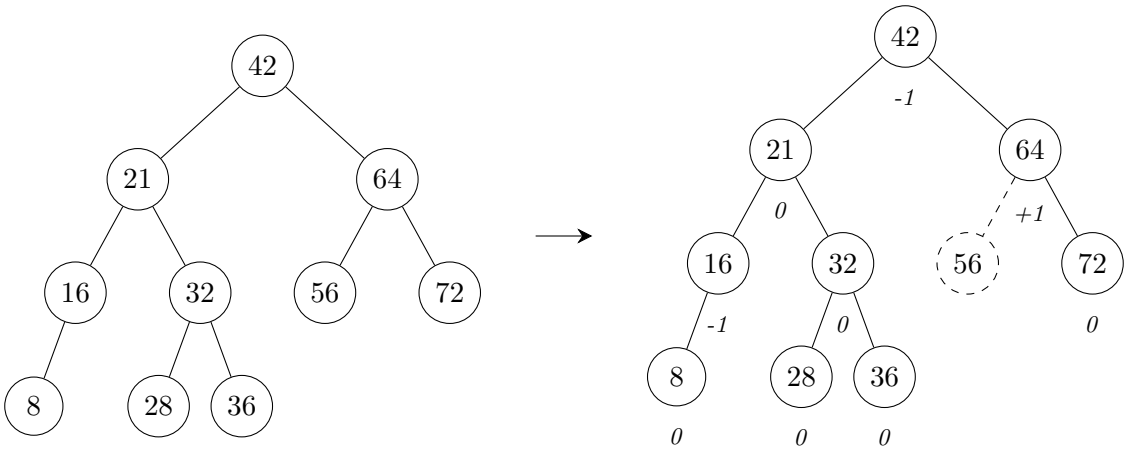
(10) Insertion de 8



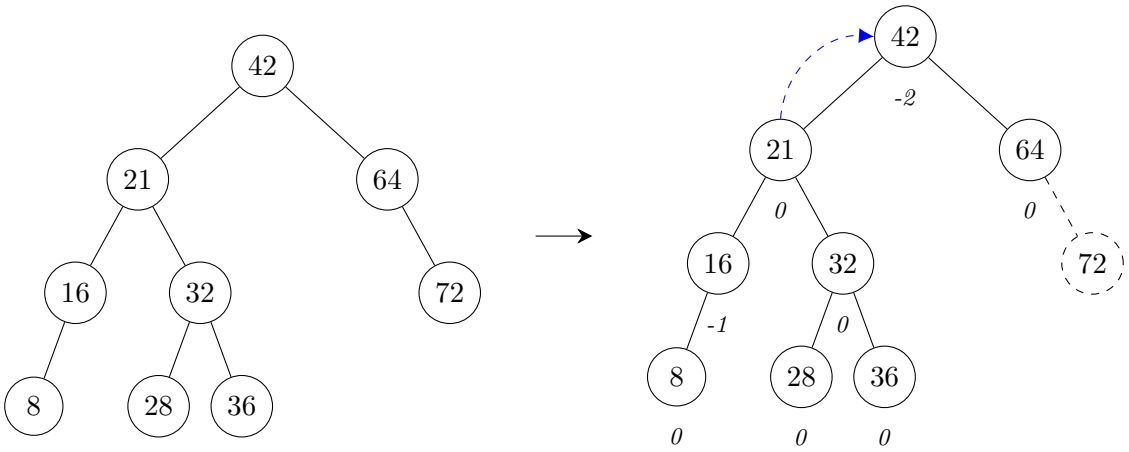
2.4 Suppression dans un AVL

La suppression dans un AVL suit globalement la même démarche : on effectue d'abord une suppression classique dans un ABR (donc en vérifiant les 3 cas où le nœud est une feuille, un point simple, ou possédant deux fils), puis on met à jour les facteurs d'équilibrage afin de déterminer si une rotation est nécessaire (et si oui, laquelle).

(1) Suppression de 56

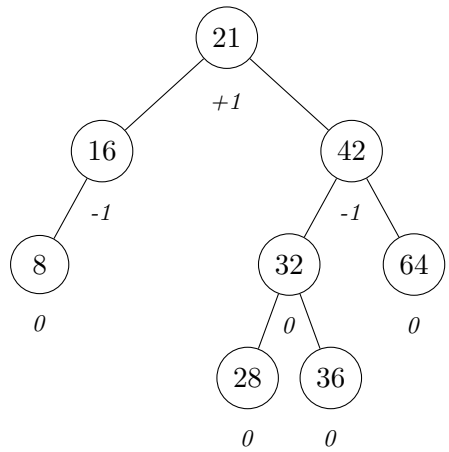


(2) Suppression de 72



De manière générale, l'équilibrage s'effectue selon le facteur d'équilibrage d'un premier nœud, puis, du facteur d'un de ses fils. Si un nœud a un facteur de -2 (respectivement 2), on s'intéresse à son fils gauche (respectivement droit) pour tester si son facteur est inférieur à 1 (respectivement supérieur à -1), si c'est le cas on effectue une rotation droite (respectivement gauche), sinon, on effectue une rotation gauche droite (respectivement droite gauche).

rotation droite



Ce document et ses illustrations ont été réalisés par Fabrice BOISSIER en mars 2023. Certaines illustrations sont inspirées des supports de cours de Nathalie "Junior" BOUQUET, et Christophe "Krisboul" BOULLAY.