Tri par tas et arbres AVL

Algorithmique et structures de données, 2022-2023

P. Albuquerque (B410), P. Künzli et O. Malaspinas (A401), ISC, HEPIA

2023-03-24

En partie inspirés des supports de cours de P. Albuquerque

Questions sur les notions du dernier cours (1/2)

- Comment représenter un tableau sous forme d'arbre binaire?

Questions sur les notions du dernier cours (1/2)

- Comment représenter un tableau sous forme d'arbre binaire?
- Qu'est-ce qu'un tas?

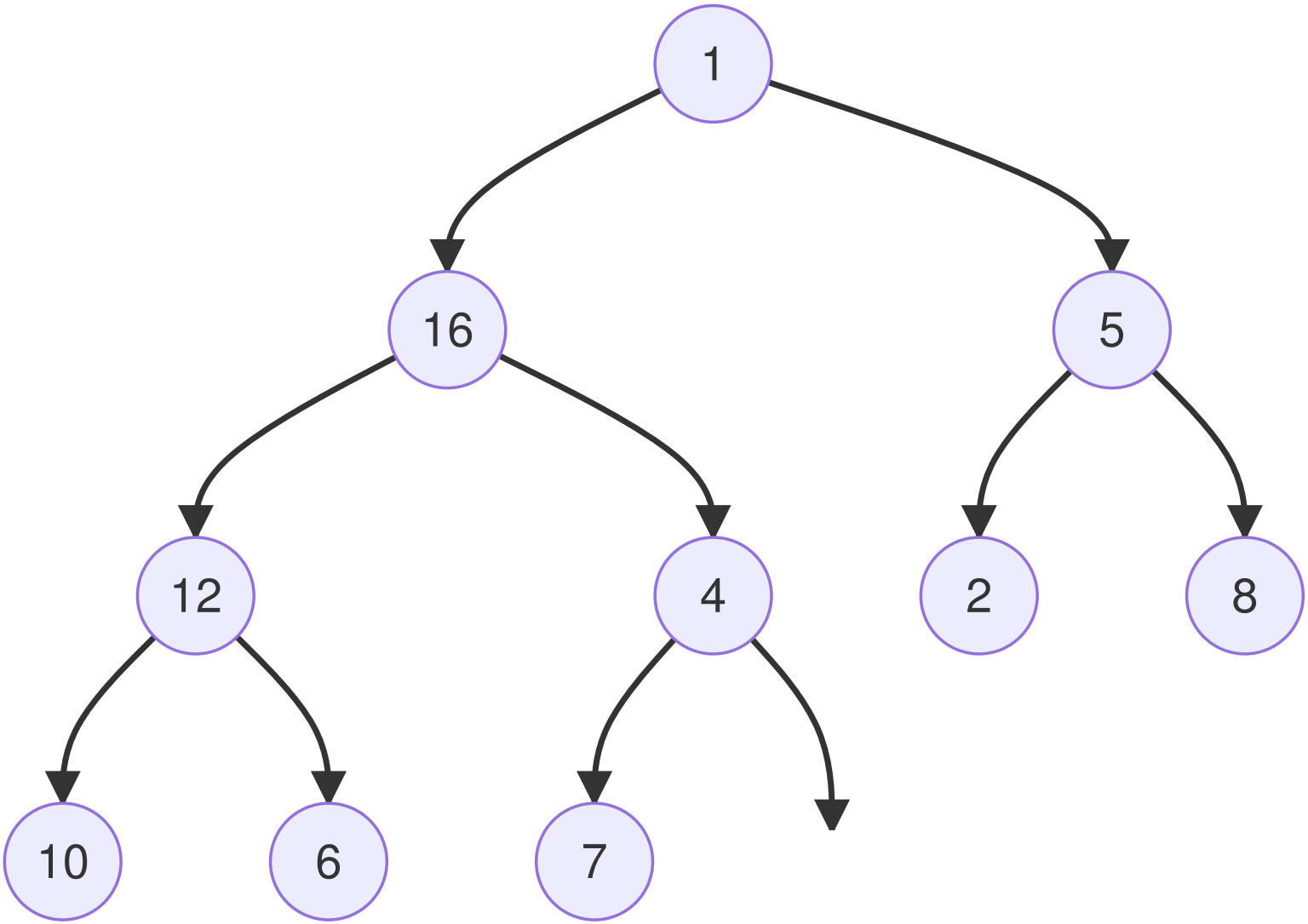
Questions sur les notions du dernier cours (1/2)

- Comment représenter un tableau sous forme d'arbre binaire?
- Qu'est-ce qu'un tas?

| 1 | 16 | 5 | 12 | 4 | 2 | 8 | 10 | 6 | 7 |

- Quel est l'arbre que cela représente?

Questions sur les notions du dernier cours (2/2)



Exercice

- Trier par tas le tableau

  | 1 | 2 | 4 | 5 | 6 | 8 | 10 | 12 | 16

- Mettez autant de détails que possible.
- Que constatez-vous?
- Postez le résultat sur matrix.

L'algorithme du tri par tas (1/2)

Deux étapes

1. Entassement: transformer l'arbre en tas.
2. Échanger la racine avec le dernier élément et entasser la racine.

Pseudo-code d'entassement de l'arbre (15 min, matrix)

L'algorithme du tri par tas (1/2)

Deux étapes

1. Entassement: transformer l'arbre en tas.
2. Échanger la racine avec le dernier élément et entasser la racine.

Pseudo-code d'entassement de l'arbre (15 min, matrix)

```
rien tri_par_tas(tab)
    entassement(tab)
    échanger(tab[0], tab[size(tab)-1])
    pour i de size(tab)-1 à 2
        tamisage(tab, 0)
        échanger(tab[0], tab[i-1])

rien entassement(tab)
    pour i de size(tab)/2-1 à 0
        tamisage(tab, i)

rien tamisage(tab, i)
    ind_max = ind_max(tab, i, gauche(i), droite(i))
    si i != ind_max
        échanger(tab[i], tab[ind_max])
        tamisage(tab, ind_max)
```

L'algorithme du tri par tas (2/2)

- Fonctions utilitaires

```
entier ind_max(tab, i, g, d)
    ind_max = i
    si tab[ind_max] < tab[g] && size(tab) > g
        ind_max = g
    si tab[ind_mx] < tab[d] && size(tab) > d
        ind_max = d
    retourne ind_max

entier gauche(i)
    retourne 2 * i + 1

entier droite(i)
    retourne 2 * i + 2
```

Complexités

Complexité de la recherche

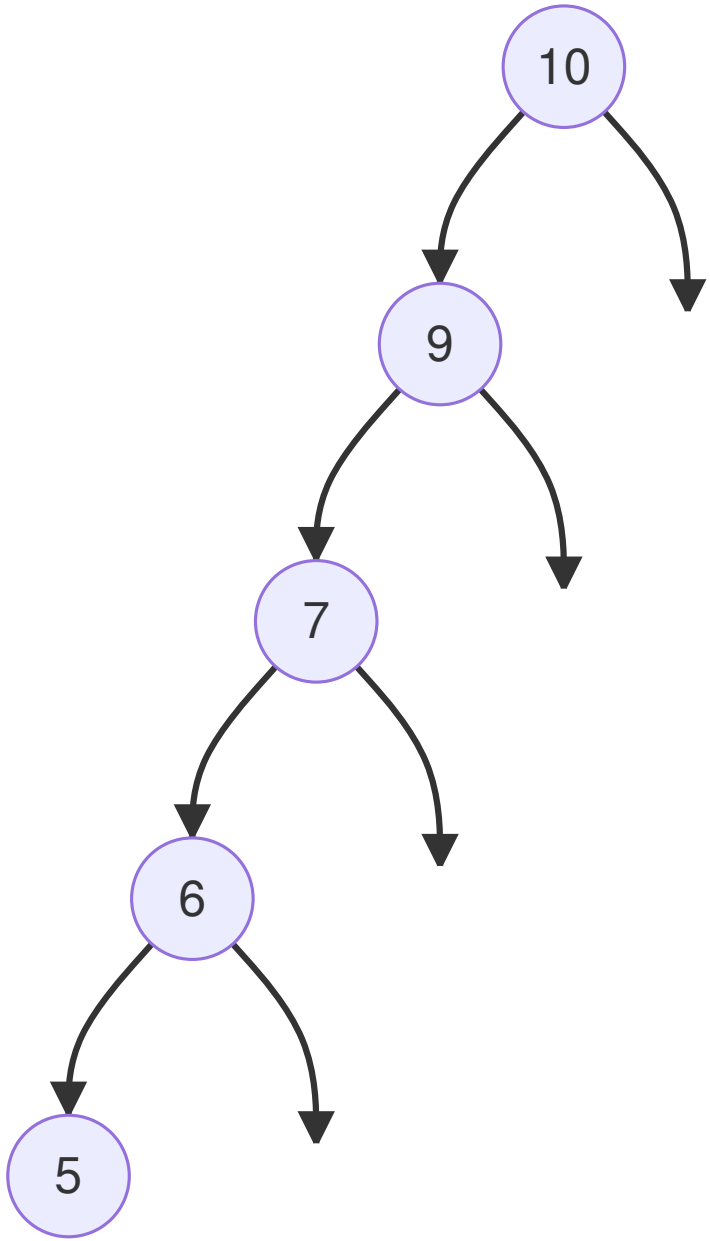
1. En moyenne?
2. Dans le pire des cas?

Complexités

Complexité de la recherche

1. En moyenne?
2. Dans le pire des cas?

1. $O(\log_2(N))$
2. $O(N)$



Un meilleur arbre

- Quelle propriété doit satisfaire un arbre pour être $O(\log_2(N))$?

Un meilleur arbre

- Quelle propriété doit satisfaire un arbre pour être $O(\log_2(N))$?
- Si on a environ le même nombre de nœuds dans les sous-arbres.

Définition

Un arbre binaire est parfaitement équilibré si, pour chaque nœud, la différence entre les nombres de nœuds des sous- arbres gauche et droit vaut au plus 1.

- Si l'arbre est **parfaitement équilibré**, alors tout ira bien.
- Quelle est la hauteur (profondeur) d'un arbre parfaitement équilibré?

Un meilleur arbre

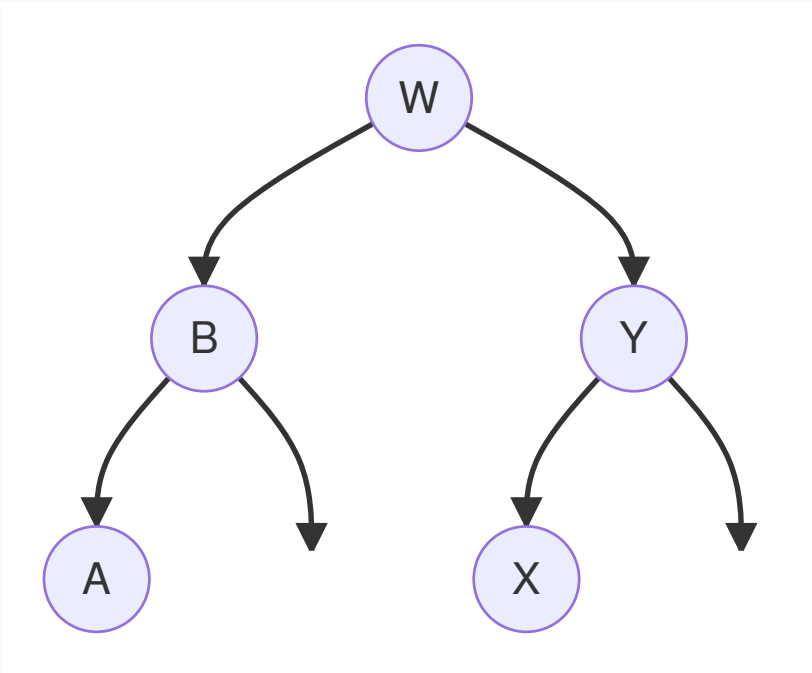
- Quelle propriété doit satisfaire un arbre pour être $O(\log_2(N))$?
- Si on a environ le même nombre de nœuds dans les sous-arbres.

Définition

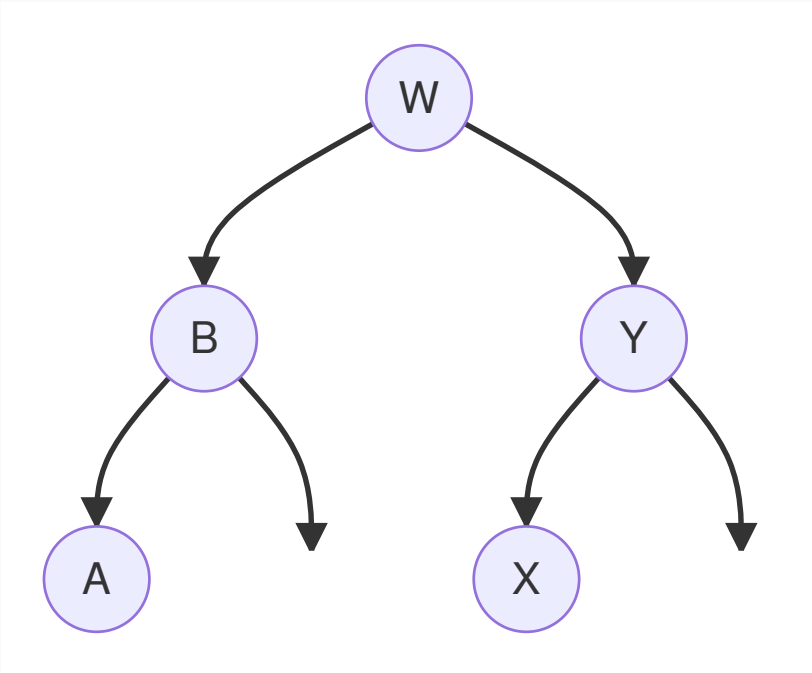
Un arbre binaire est parfaitement équilibré si, pour chaque nœud, la différence entre les nombres de nœuds des sous- arbres gauche et droit vaut au plus 1.

- Si l'arbre est **parfaitement équilibré**, alors tout ira bien.
- Quelle est la hauteur (profondeur) d'un arbre parfaitement équilibré?
- $O(\log_2(N))$.

Équilibre parfait ou pas?

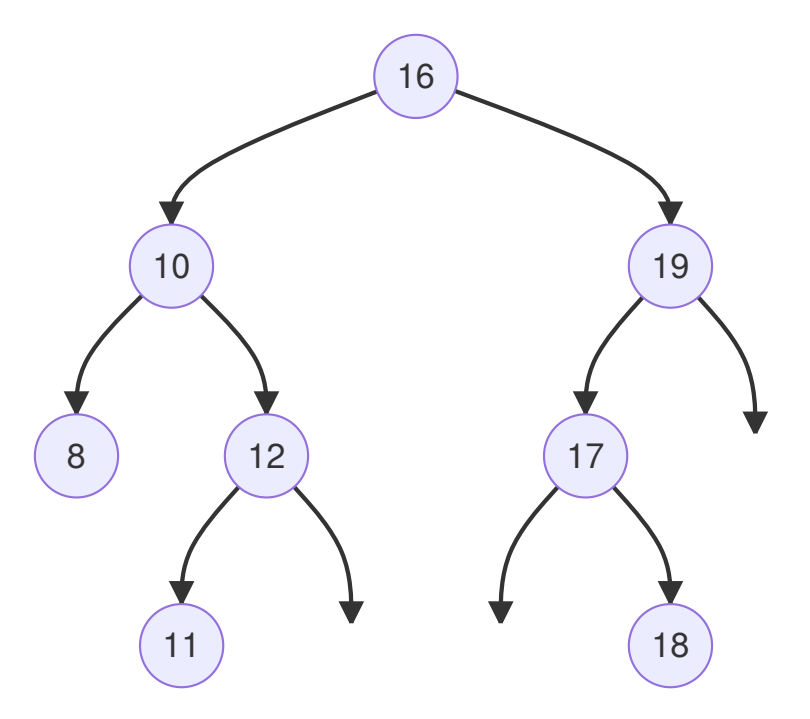


Équilibre parfait ou pas?

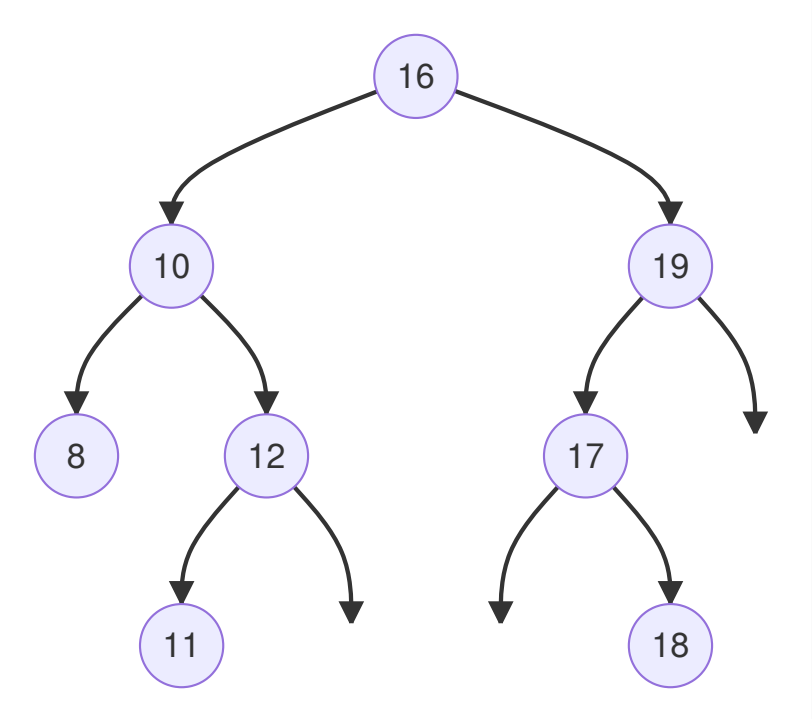


ÉQUILIBRÉ

Équilibre parfait ou pas?



Équilibre parfait ou pas?



PAS ÉQUILIBRÉ

Arbres AVL

- Quand est-ce qu'on équilibre un arbre?

Arbres AVL

- Quand est-ce qu'on équilibre un arbre?

- A l'insertion/suppression.
- Maintenir un arbre en état d'équilibre parfait: cher (insertion, suppression).
- On peut "relaxer" la condition d'équilibre: profondeur (hauteur) au lieu du nombre de nœuds:
  - La hauteur $\sim \log_2(N)$.

Définition

Un arbre AVL (Adelson-Velskii et Landis) est un arbre binaire de recherche dans lequel, pour chaque nœud, la différence de hauteur entre le sous-arbre gauche et droit vaut au plus 1.

- Relation entre nœuds et hauteur:

$$\log_2(1+N) \leq 1+H \leq 1.44 \cdot \log_2(2+N), \quad N = 10^5,\ 17 \leq H \leq 25.$$

- Permet l'équilibrage en temps constant: insertion/suppression $O(\log_2(N))$.

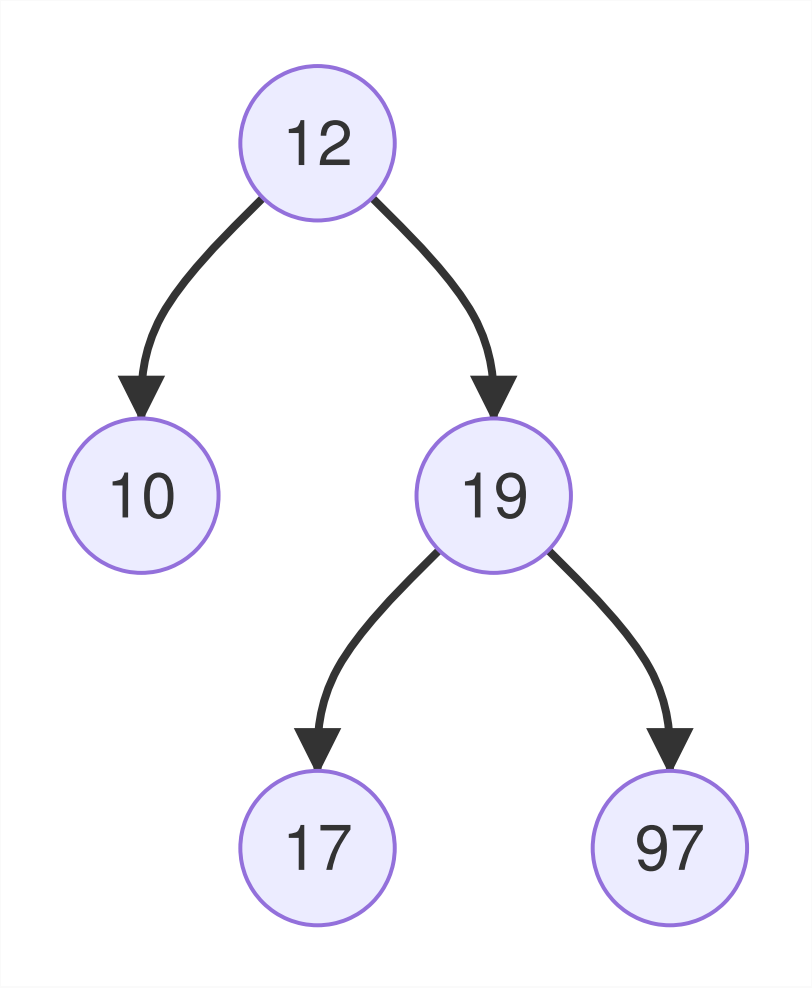
Notation

- `hg`: hauteur du sous-arbre gauche.
- `hd`: hauteur du sous-arbre droit.
- `facteur d'équilibre = fe = hd - hg`
- Que vaut `fe` si l'arbre est AVL?

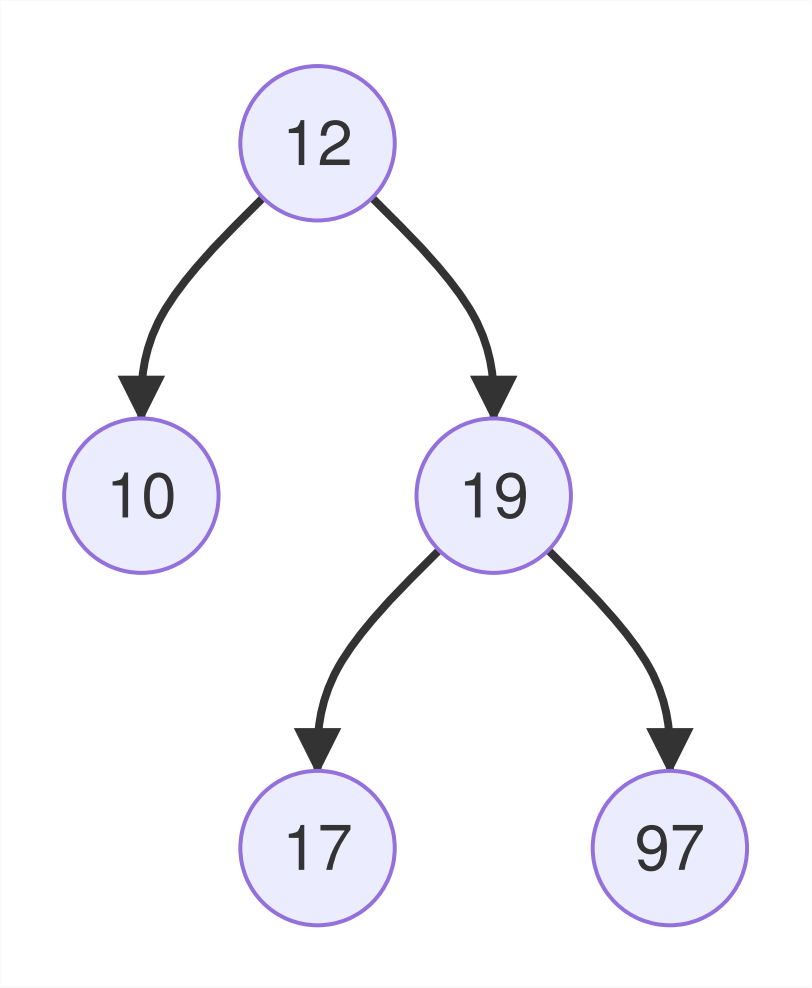
Notation

- `hg`: hauteur du sous-arbre gauche.
- `hd`: hauteur du sous-arbre droit.
- `facteur d'équilibre = fe = hd - hg`
- Que vaut `fe` si l'arbre est AVL?
- `fe = {-1, 0, 1}`

AVL ou pas?



AVL ou pas?

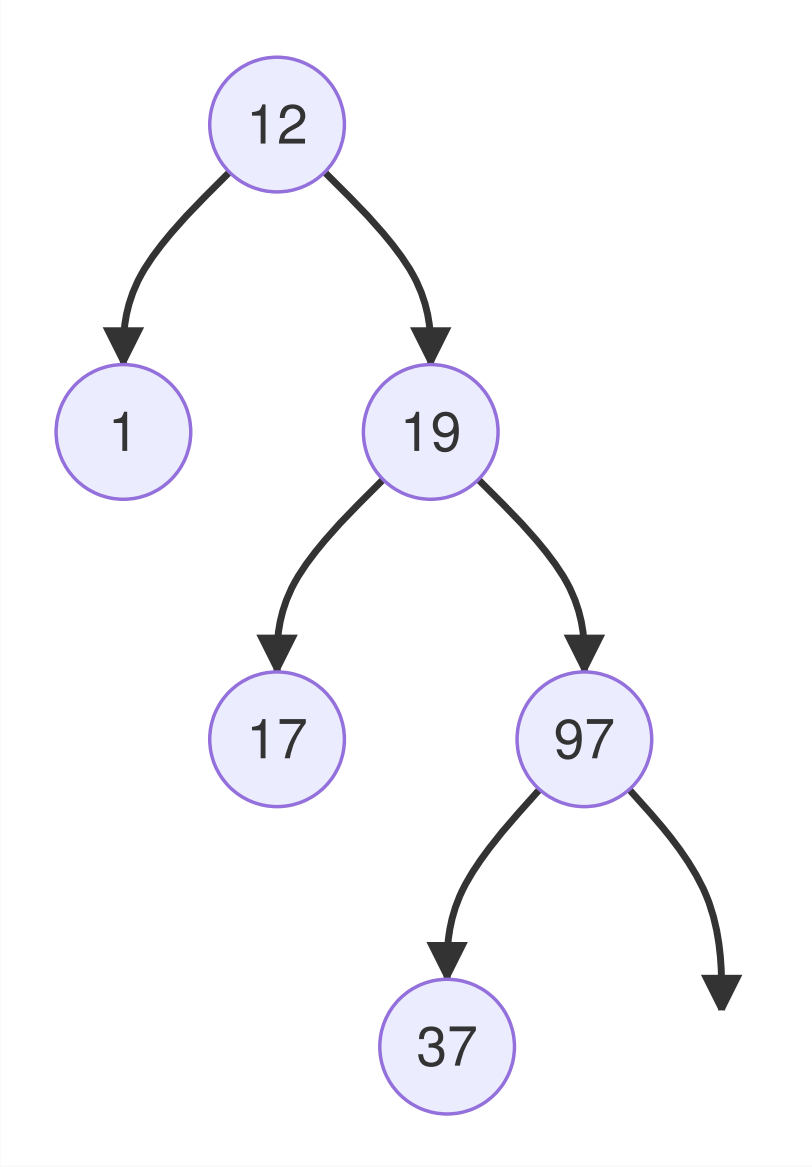


A

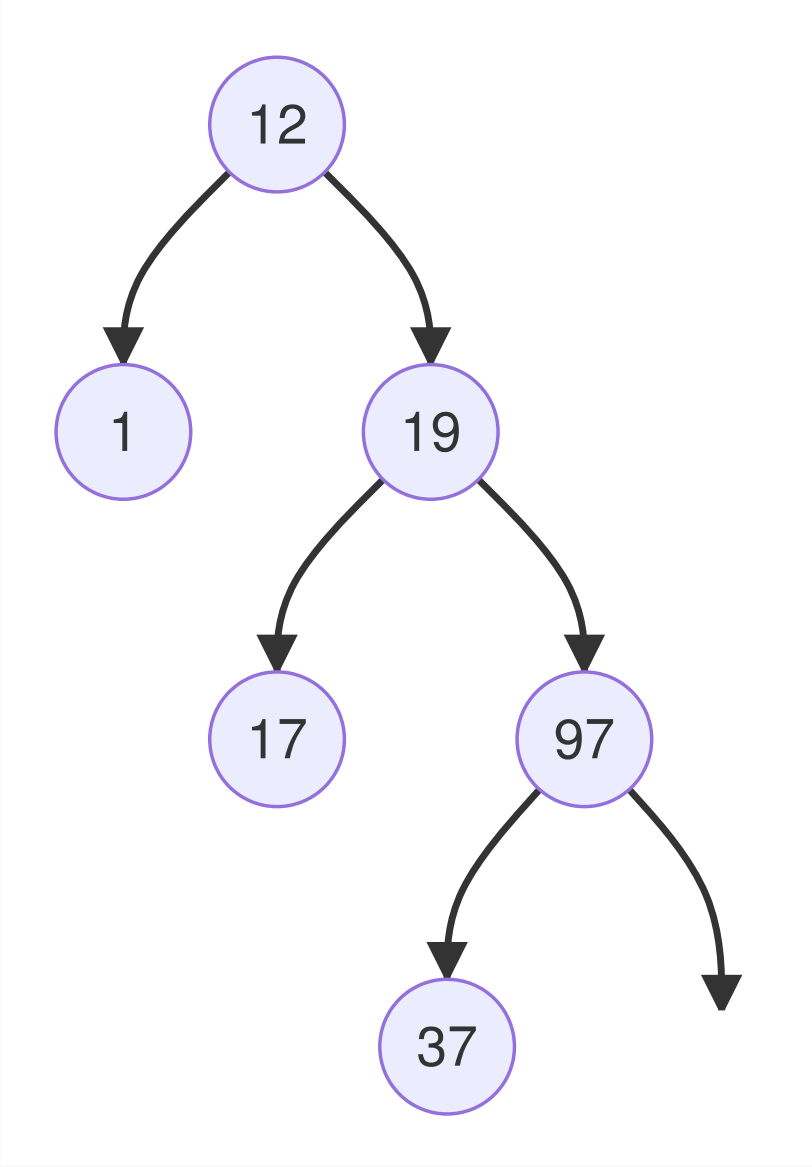
V

L

AVL ou pas?



AVL ou pas?



P

A

S

A

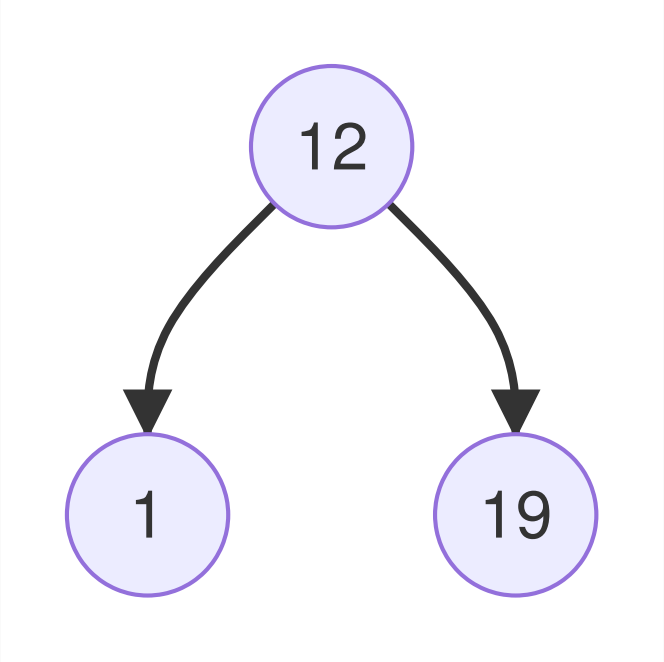
V

L

Insertion dans un arbre AVL (1/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

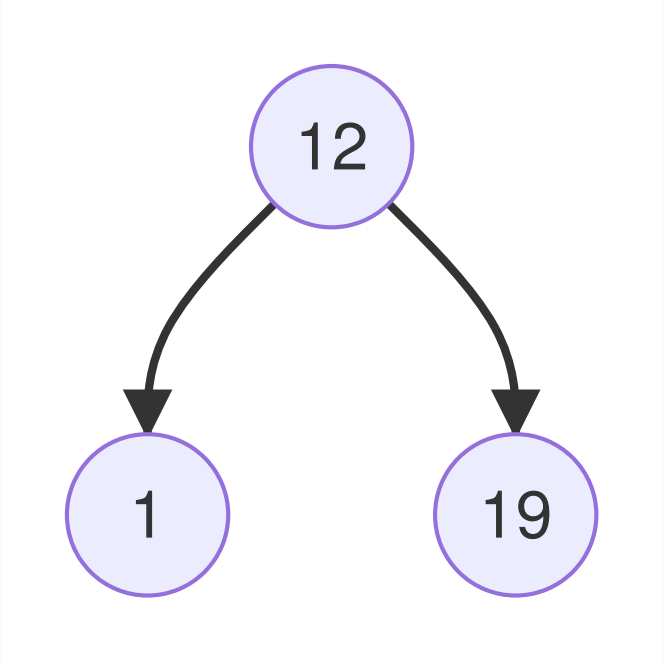
- `hd ? hg`.
- Insertion de 4?



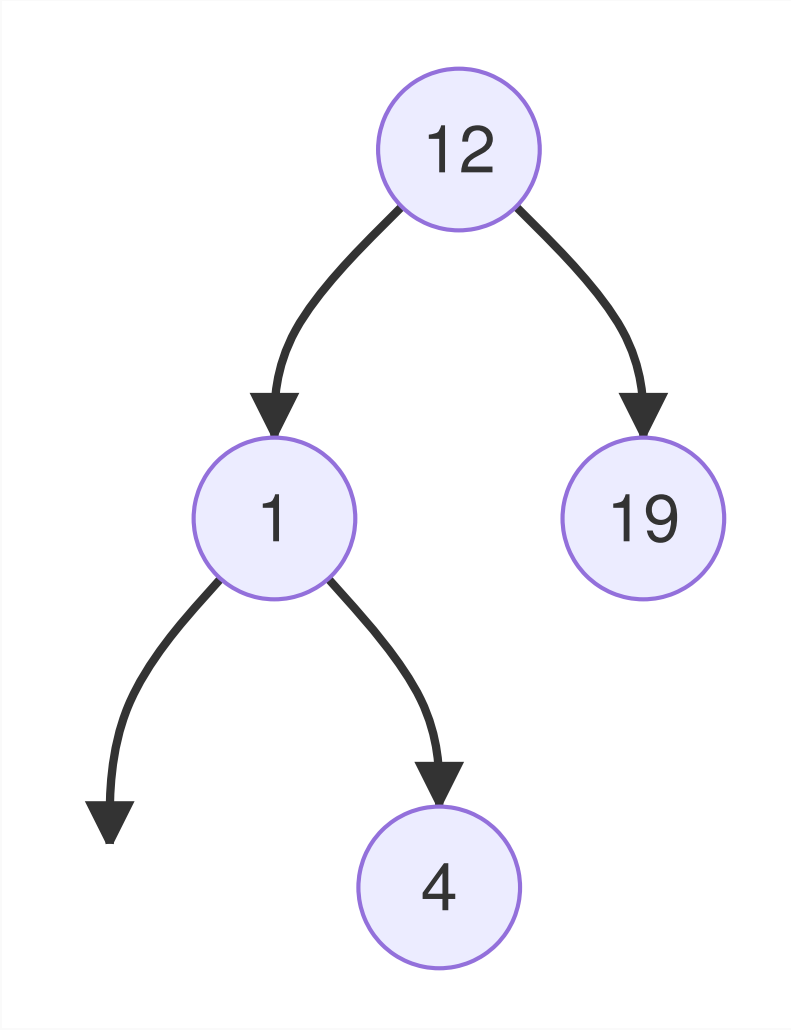
Insertion dans un arbre AVL (1/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

- `hd ? hg`.
- Insertion de 4?



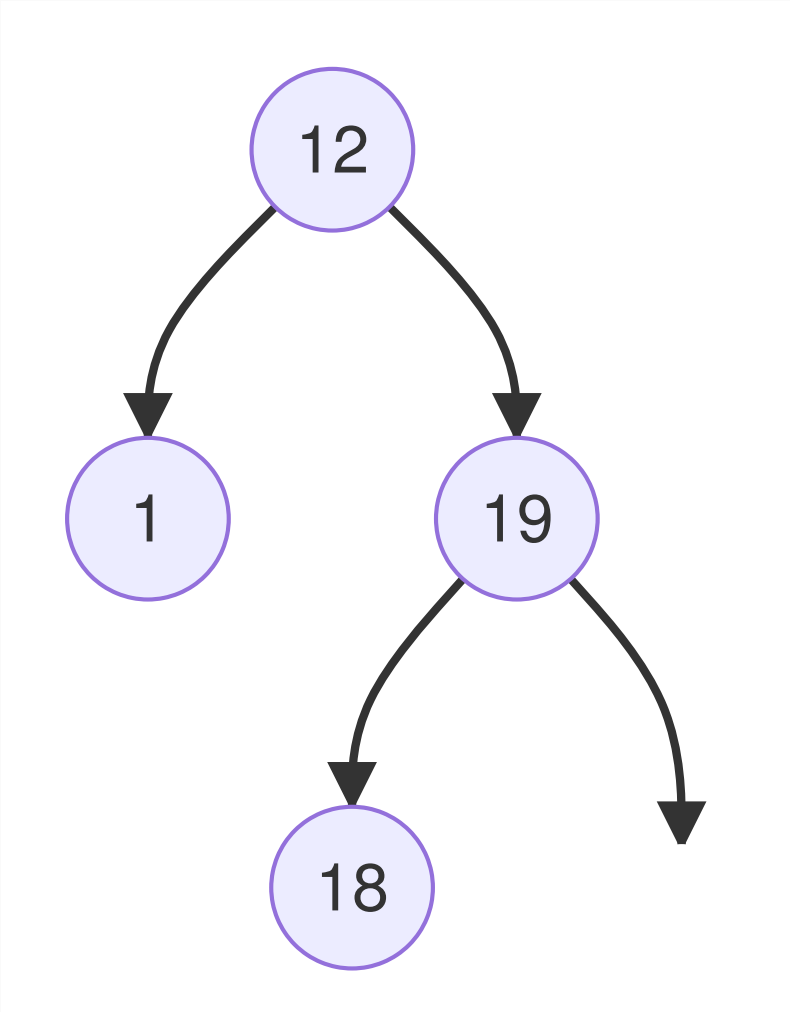
- `hd = hg`



Insertion dans un arbre AVL (2/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

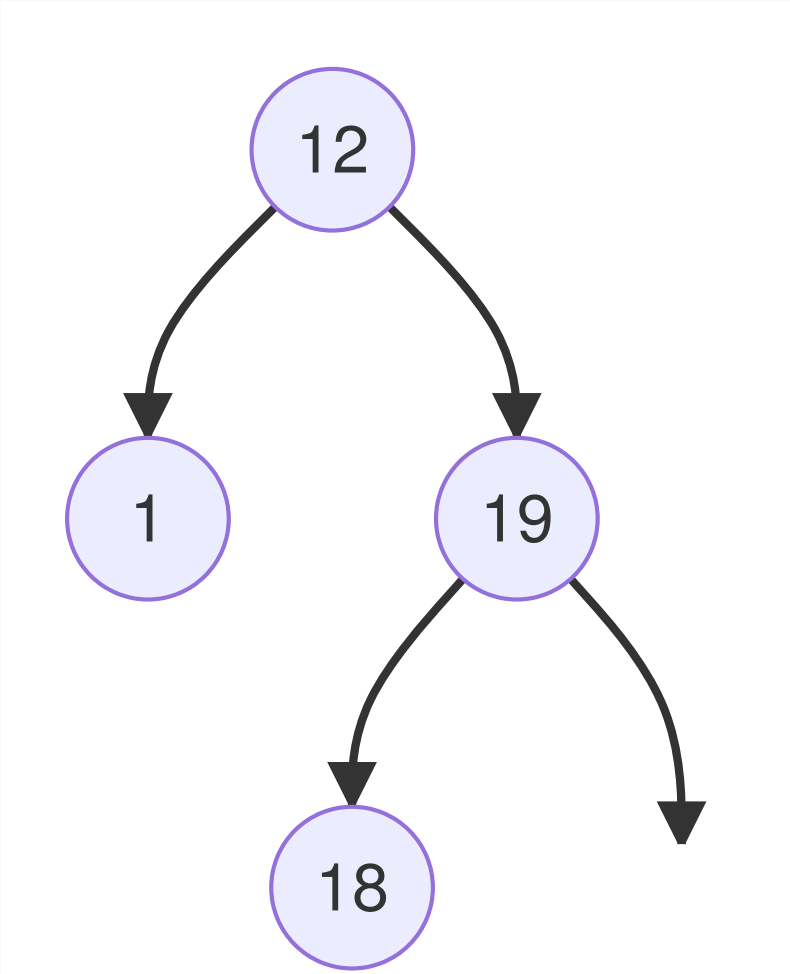
- `hd ? hg`.
- Insertion de 4?



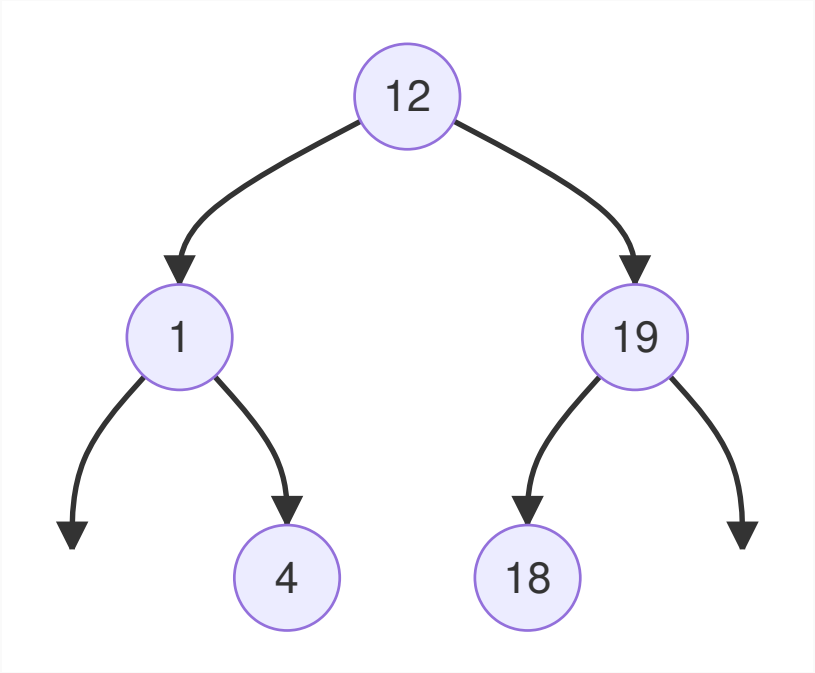
Insertion dans un arbre AVL (2/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

- `hd ? hg`.
- Insertion de 4?



- `hd < hg`

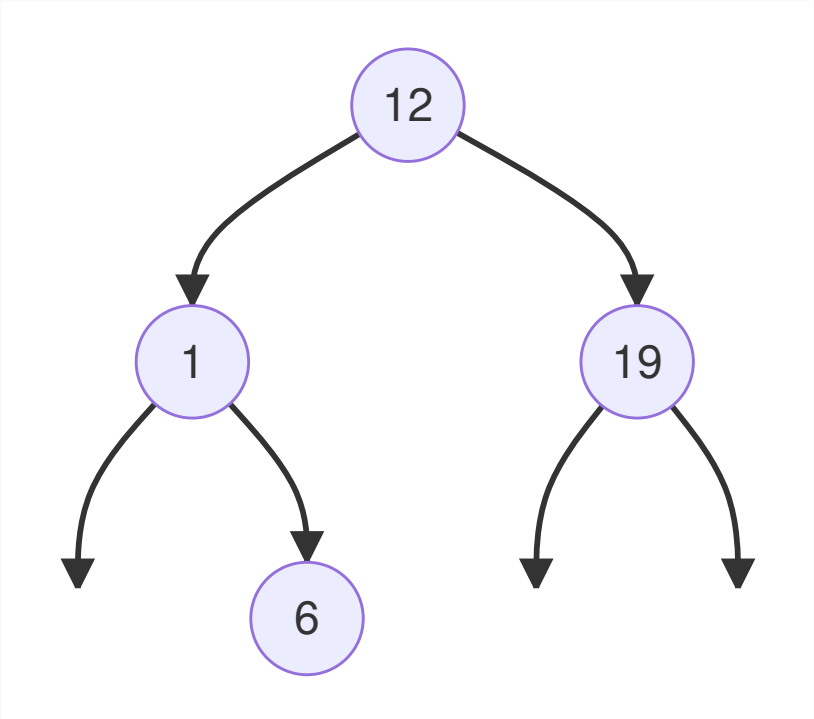


- `fe = 0`

Insertion dans un arbre AVL (3/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

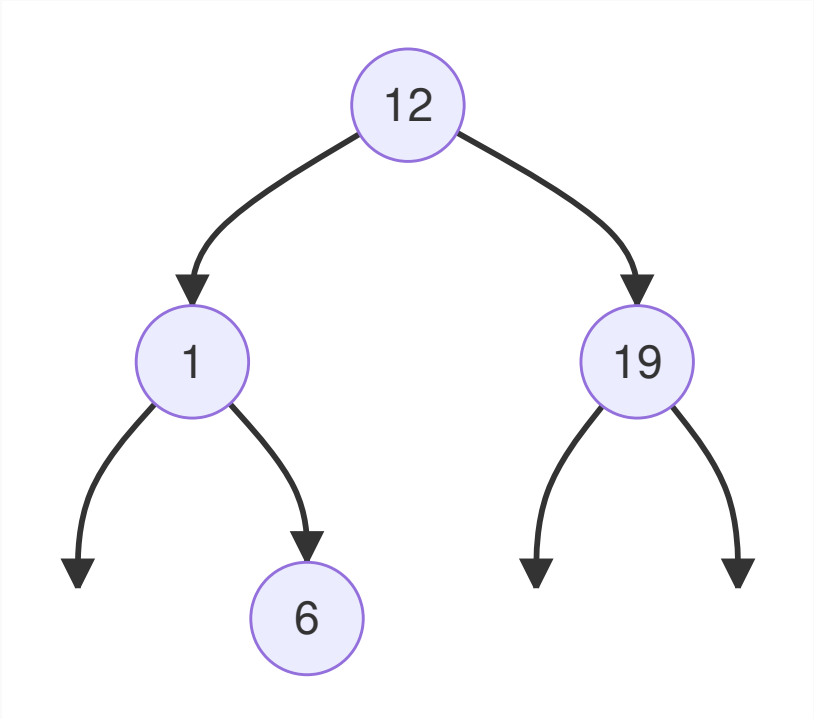
- `hd ? hg`.
- Insertion de `4`?



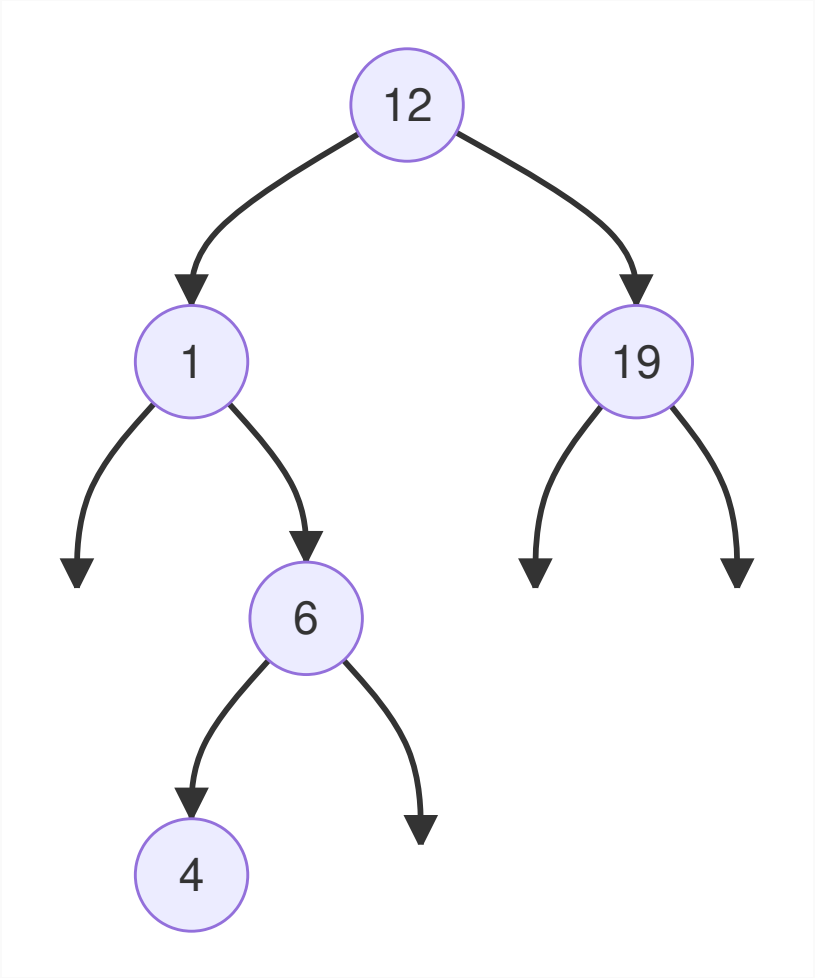
Insertion dans un arbre AVL (3/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

- `hd ? hg`.
- Insertion de 4?



- `hd < hg`

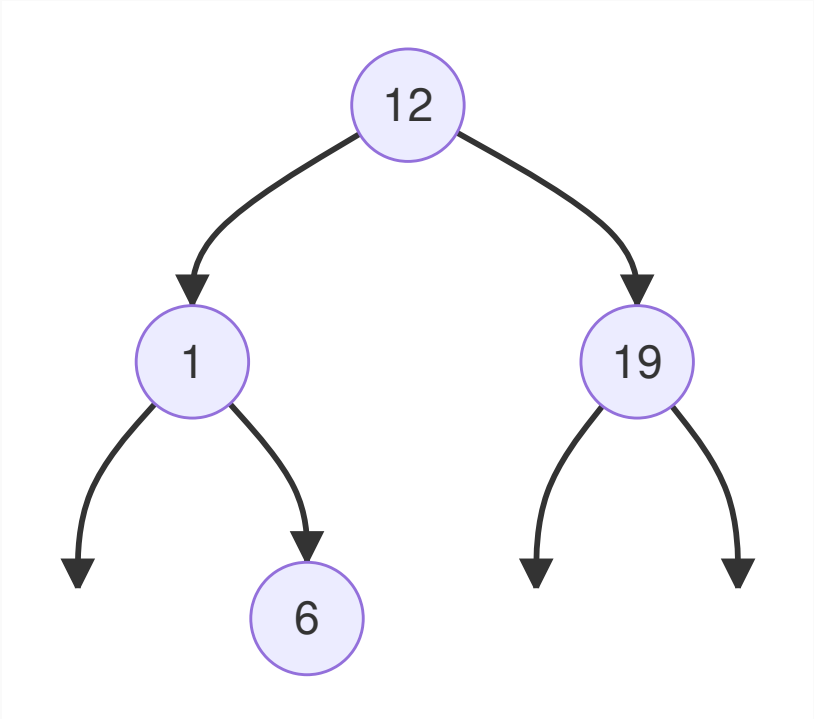


Déséquilibre! Que vaut `fe`?

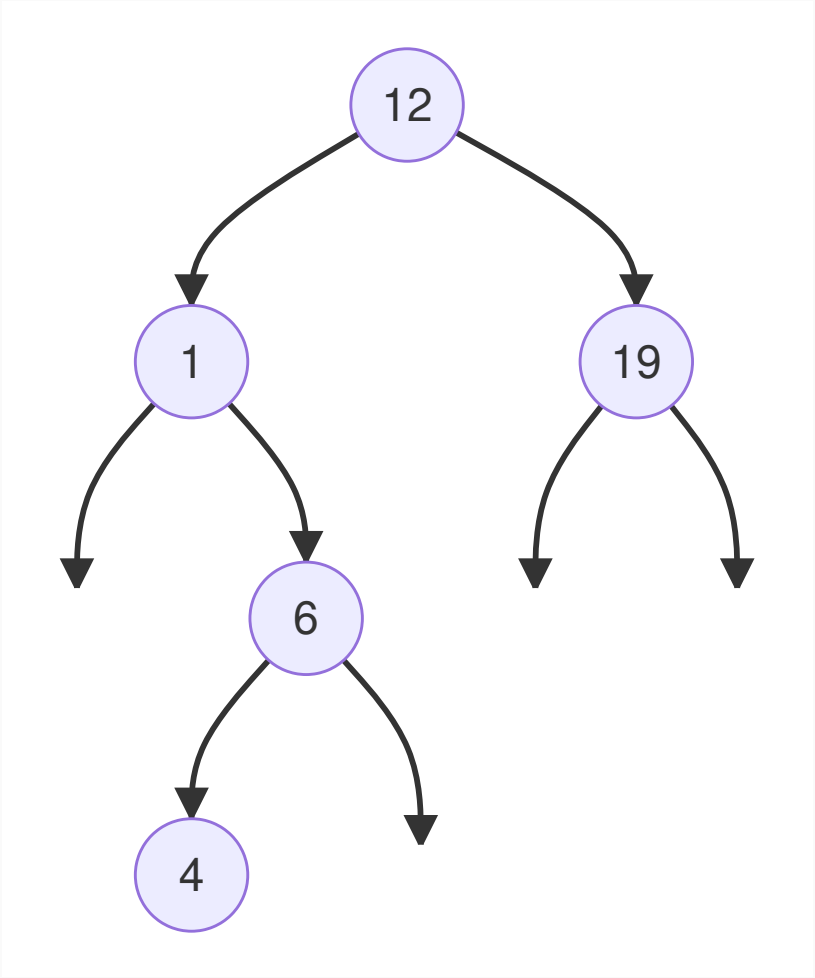
Insertion dans un arbre AVL (3/N)

1. On part d'un arbre AVL.
2. On insère un nouvel élément.

- `hd ? hg`.
- Insertion de 4?



- `hd < hg`



Déséquilibre! Que vaut `fe`?

Les cas de déséquilibre

Cas 1a

- `u`, `v`, `w` même hauteur.
- déséquilibre en `B` après insertion dans `u`

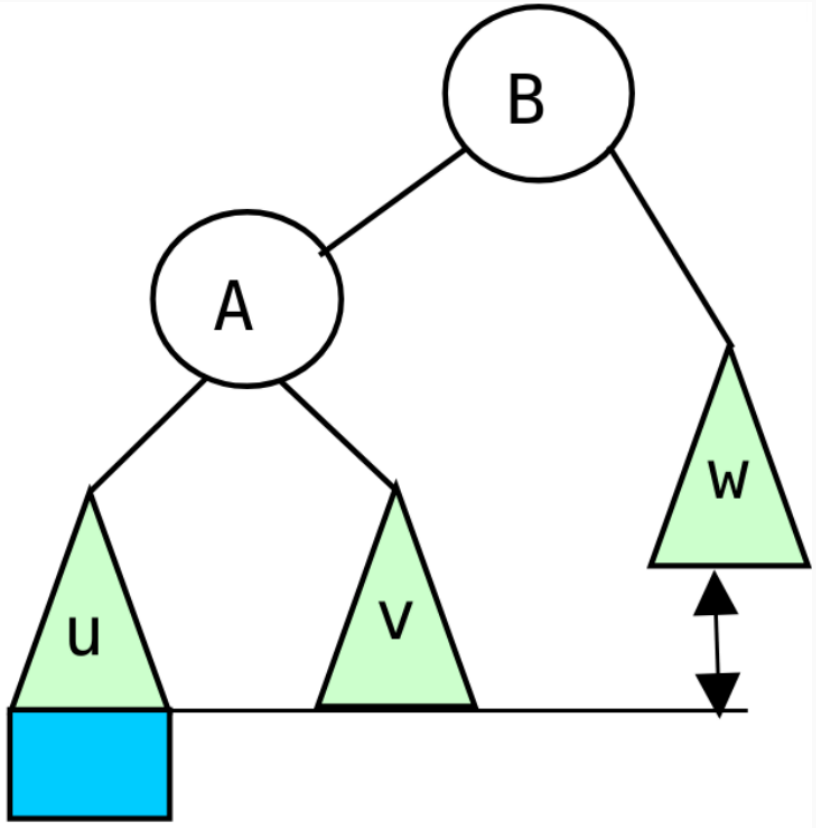


Figure 1: Après insertion

Cas 1a

- Comment rééquilibrer?

Les cas de déséquilibre

Cas 1a

- `u`, `v`, `w` même hauteur.
- déséquilibre en `B` après insertion dans `u`

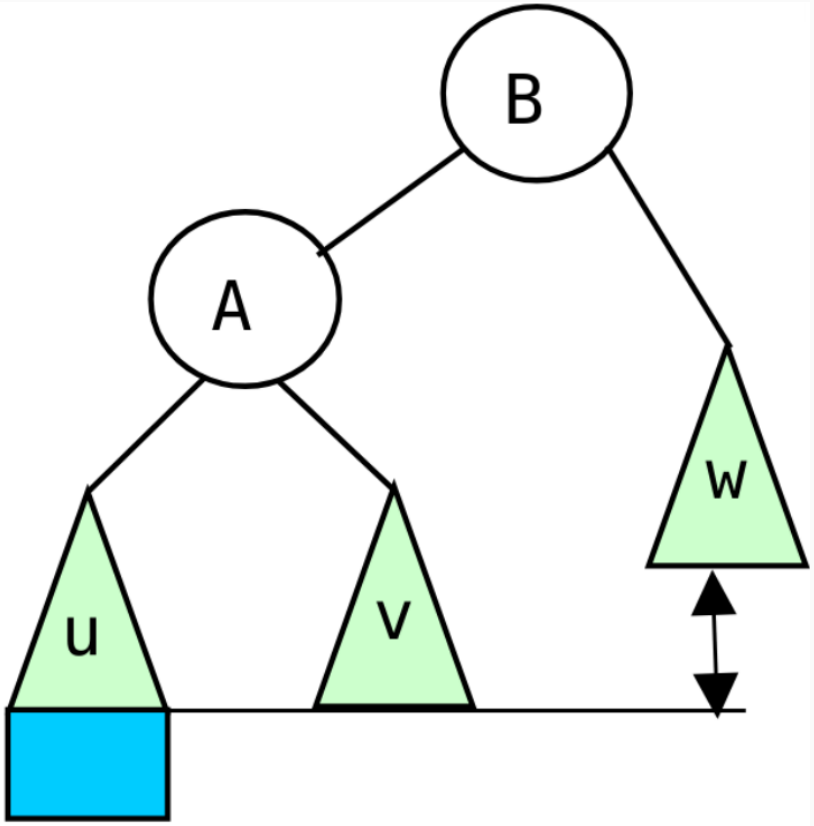


Figure 1: Après insertion

Cas 1a

- Comment rééquilibrer?
- ramène `u`, `v` `w` à la même hauteur.
- `v` à droite de `A` (gauche de `B`)

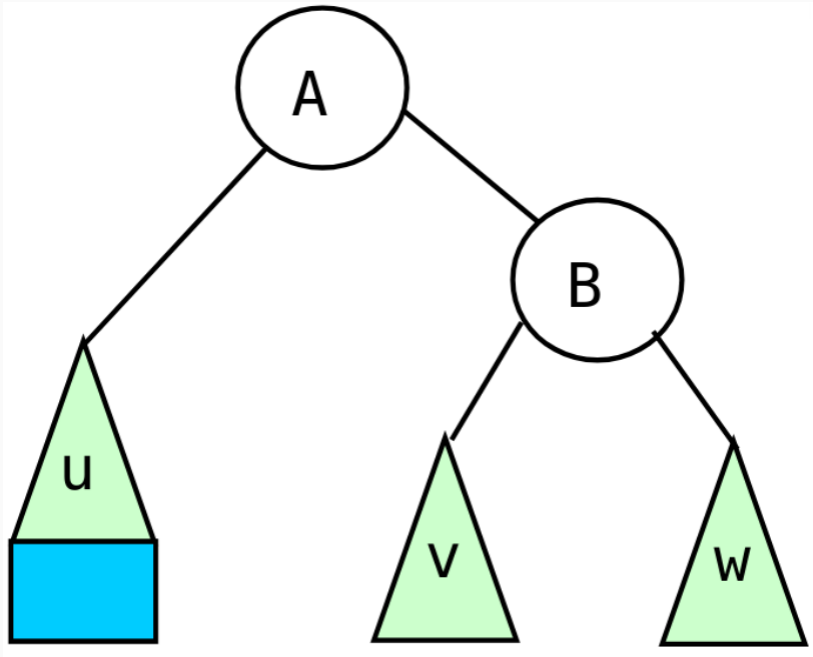


Figure 2: Après équilibrage

Les cas de déséquilibre

Cas 1b (symétrique 1a)

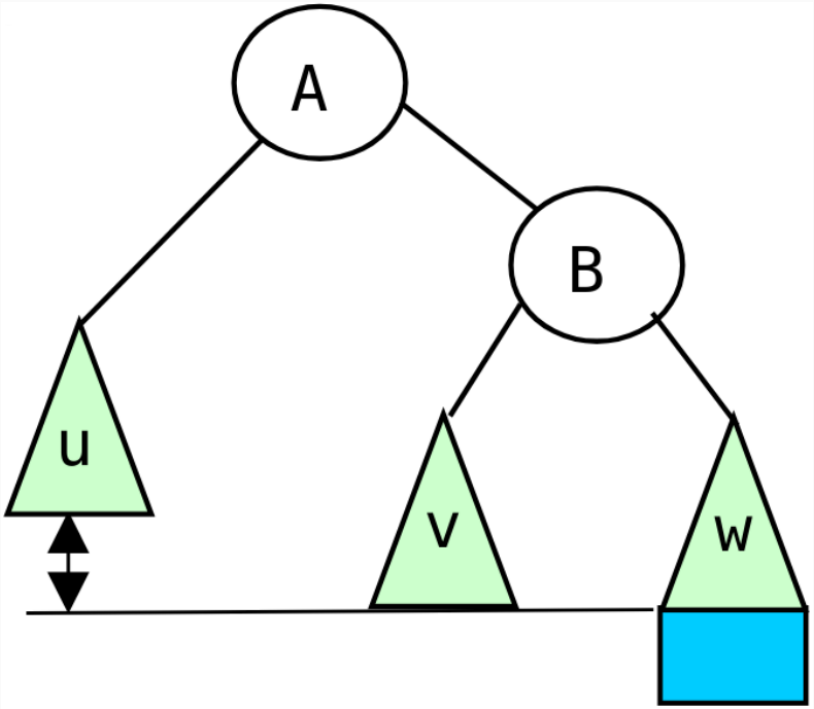


Figure 3: Après insertion

Cas 1b (symétrique 1a)

- Comment rééquilibrer?

Les cas de déséquilibre

Cas 1b (symétrique 1a)

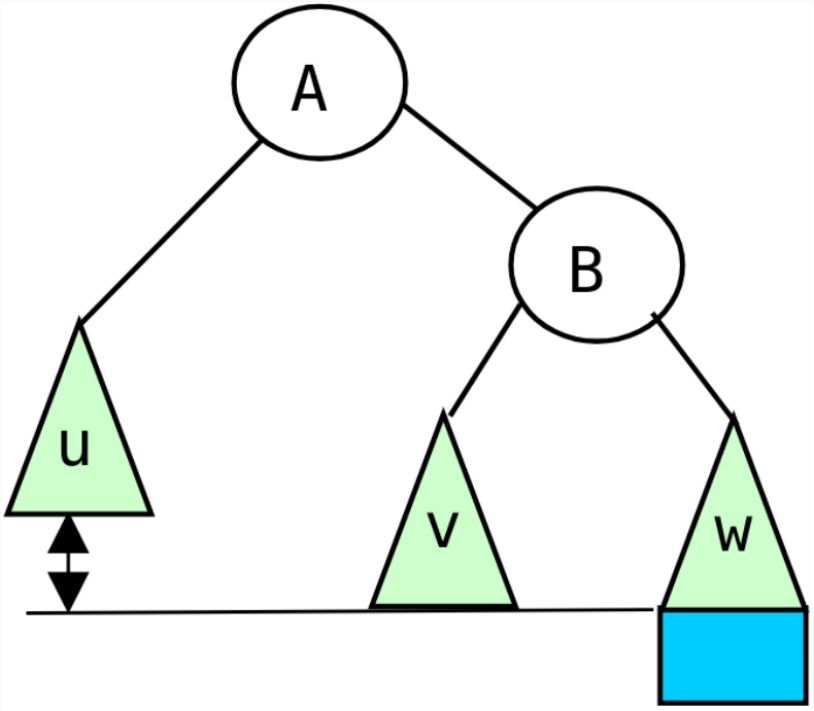


Figure 3: Après insertion

Cas 1b (symétrique 1a)

- Comment rééquilibrer?

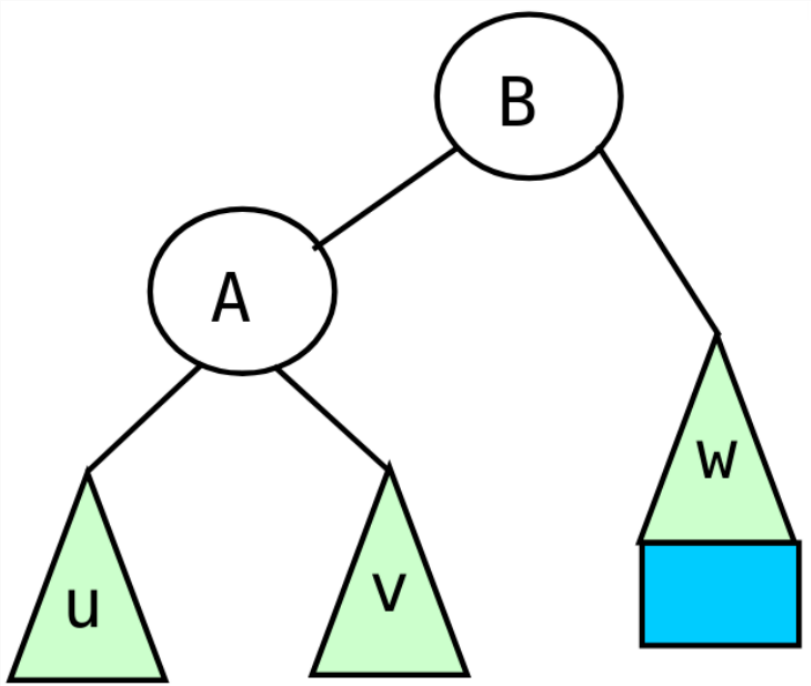
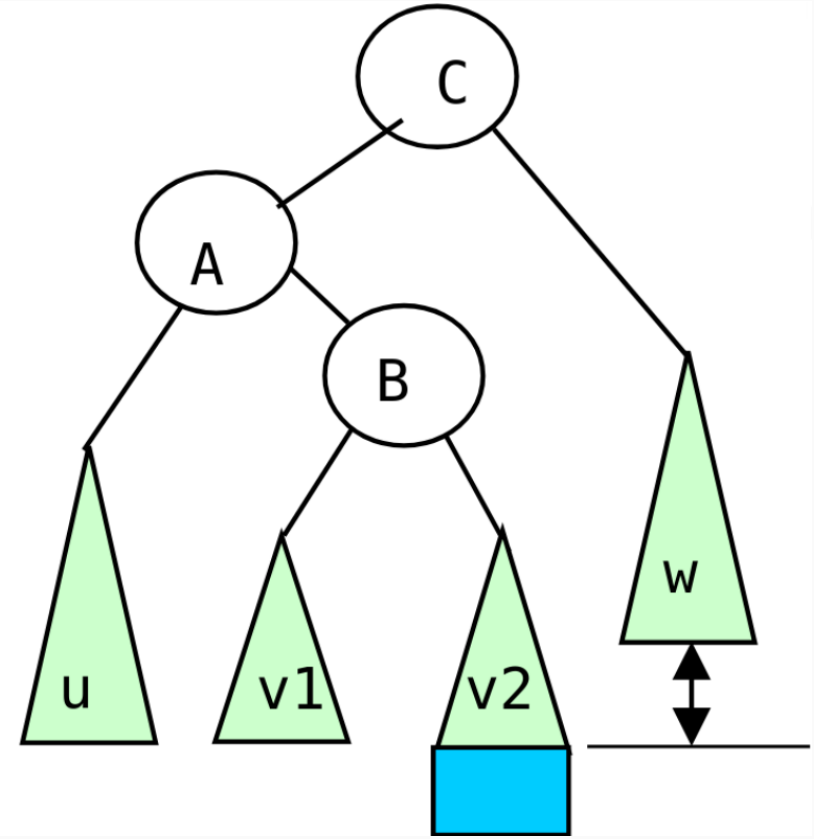


Figure 4: Après équilibrage

Les cas de déséquilibre

Cas 2a

- `h(v1)=h(v2)`, `h(u)=h(w)`.
- déséquilibre en `C` après insertion dans `v2`



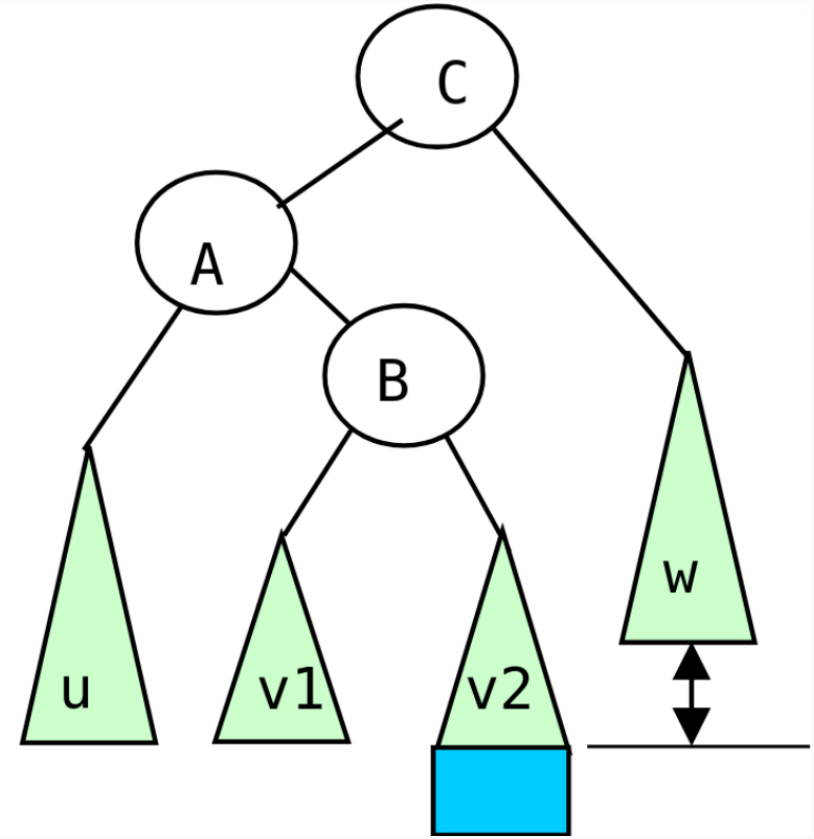
Cas 2a

- Comment rééquilibrer?

Les cas de déséquilibre

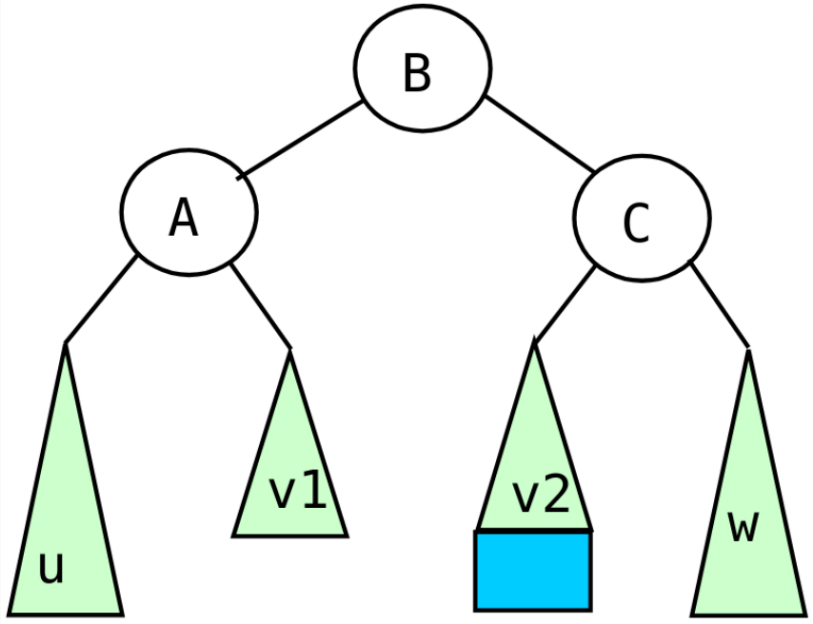
Cas 2a

- `h(v1)=h(v2)`, `h(u)=h(w)`.
- déséquilibre en `C` après insertion dans `v2`



Cas 2a

- Comment rééquilibrer?
- ramène `u`, `v2`, `w` à la même hauteur (`v1` pas tout à fait).
- `v2` à droite de `B` (gauche de `C`)
- `B` à droite de `A` (gauche de `C`)
- `v1` à droite de `A` (gauche de `B`)



Les cas de déséquilibre

Cas 2b (symétrique 2a)

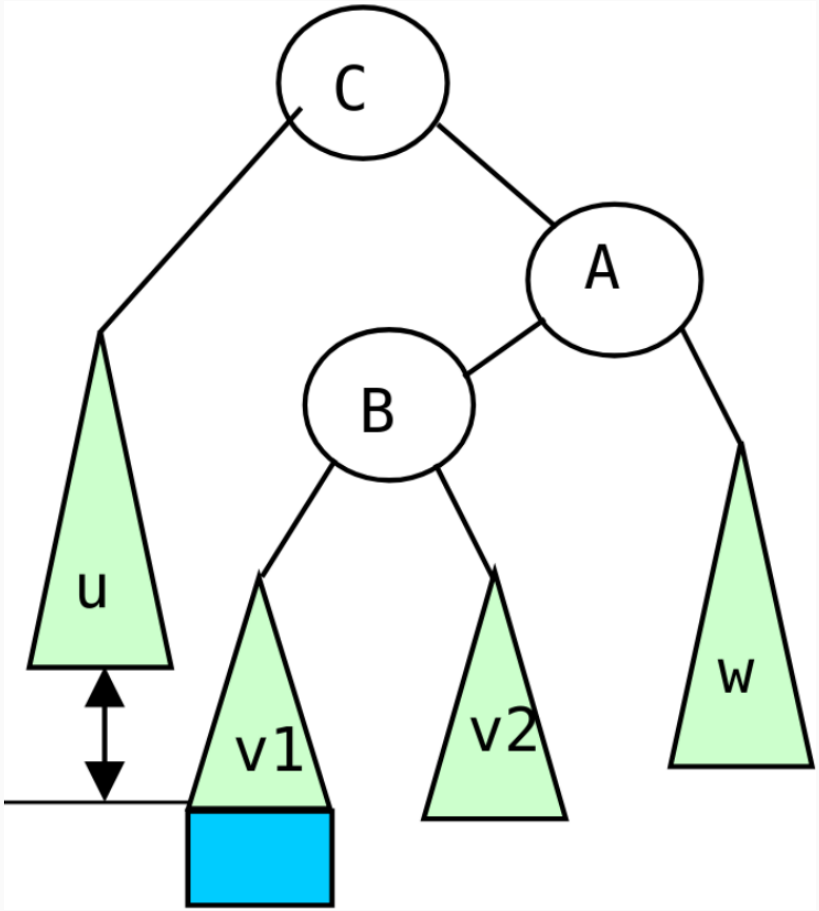


Figure 7: Après insertion

Cas 2b (symétrique 2a)

- Comment rééquilibrer?

Les cas de déséquilibre

Cas 2b (symétrique 2a)

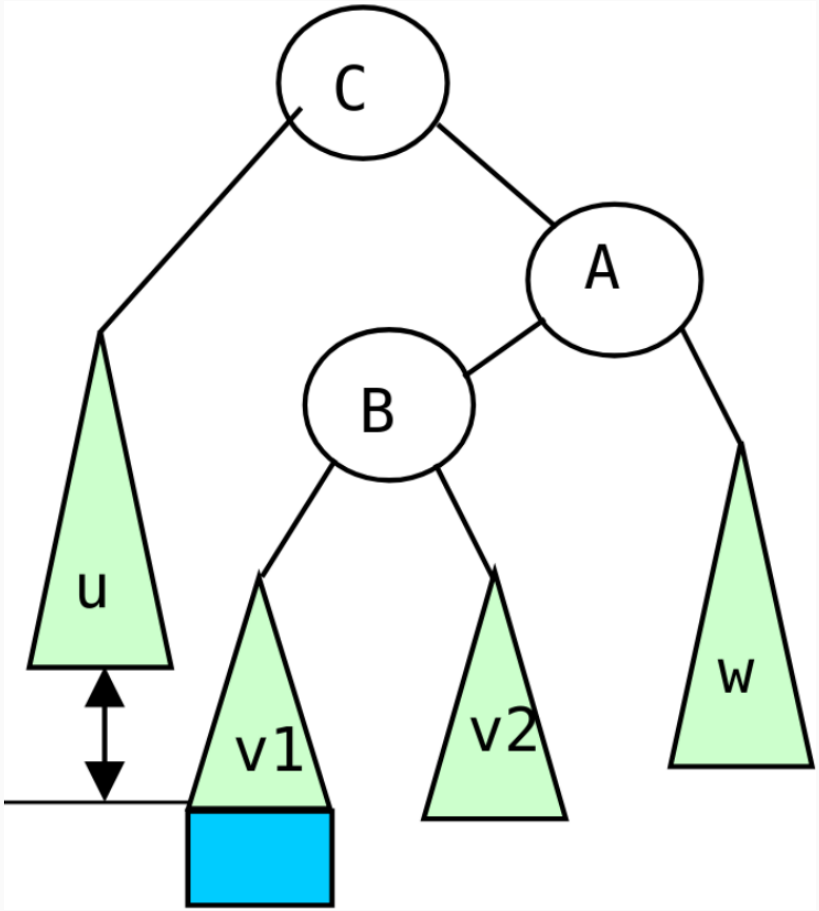


Figure 7: Après insertion

Cas 2b (symétrique 2a)

- Comment rééquilibrer?

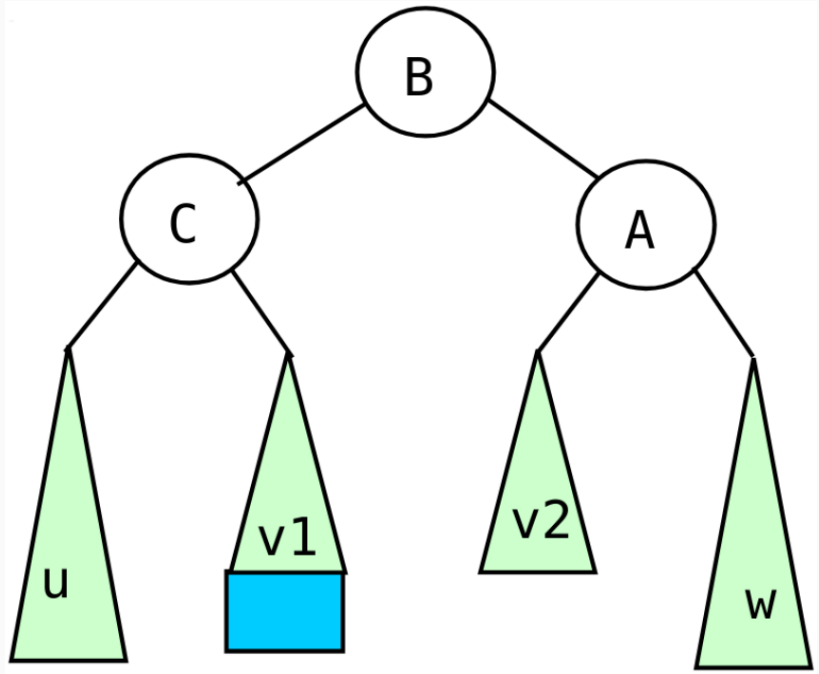


Figure 8: Après équilibrage