Sur les arbres AVL : partie II

Quel est le problème ?

1-La **hauteur** d'un arbre est une métrique très importante: elle est un indice de **performance** du traitement

-La plupart des algorithmes ont une **complexité** qui dépend de la **hauteur** de l'arbre

3-Plus l'arbre aura une **hauteur élevée**, plus l'algorithme mettra de **temps à s'exécuter**.

Comment faire qu'après des insertions successives, par exemple, la hauteur **H** de l'arbre **n'augmente pas rapidement** ?

Une solution plus **systématique** consiste à passer par des arbres:

- H-équilibrés ou arbres AVL(**A**delson-**V**elsky et **L**andis):

$$\text{hauteur} \leq \log_2 (n)$$

- ou les arbres rouge-noir :

$$\text{hauteur} \leq 2 \times \log_2 (n + 1).$$

Hauteur d'un arbre binaire

1- Calcul de la hauteur **H** d'un arbre:

$$H(\text{arbreVide}) = -1$$

$$H(A) = 1 + \text{Max}\ [\ H(\text{gauche}(A)), H(\text{droit}(A))]$$

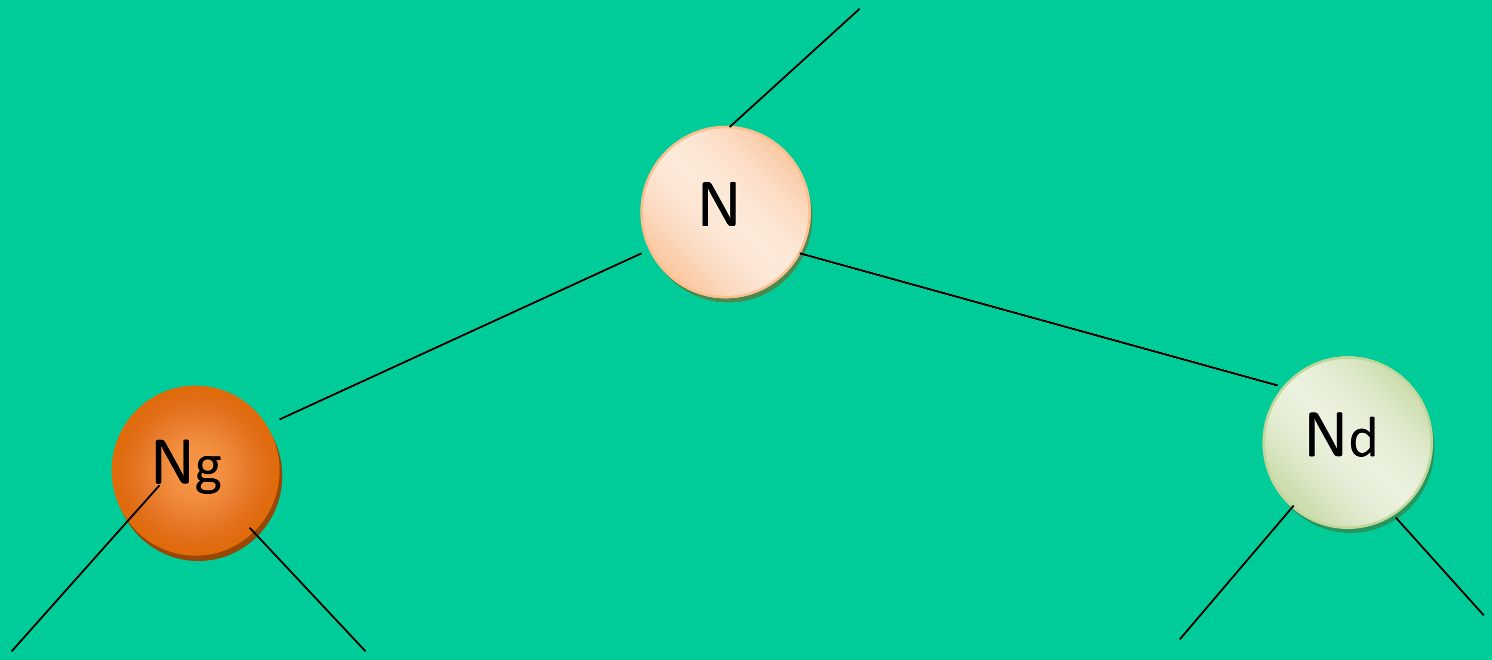
Par abus de langage:

$$H(A) = H(\textbf{racine}(A))$$

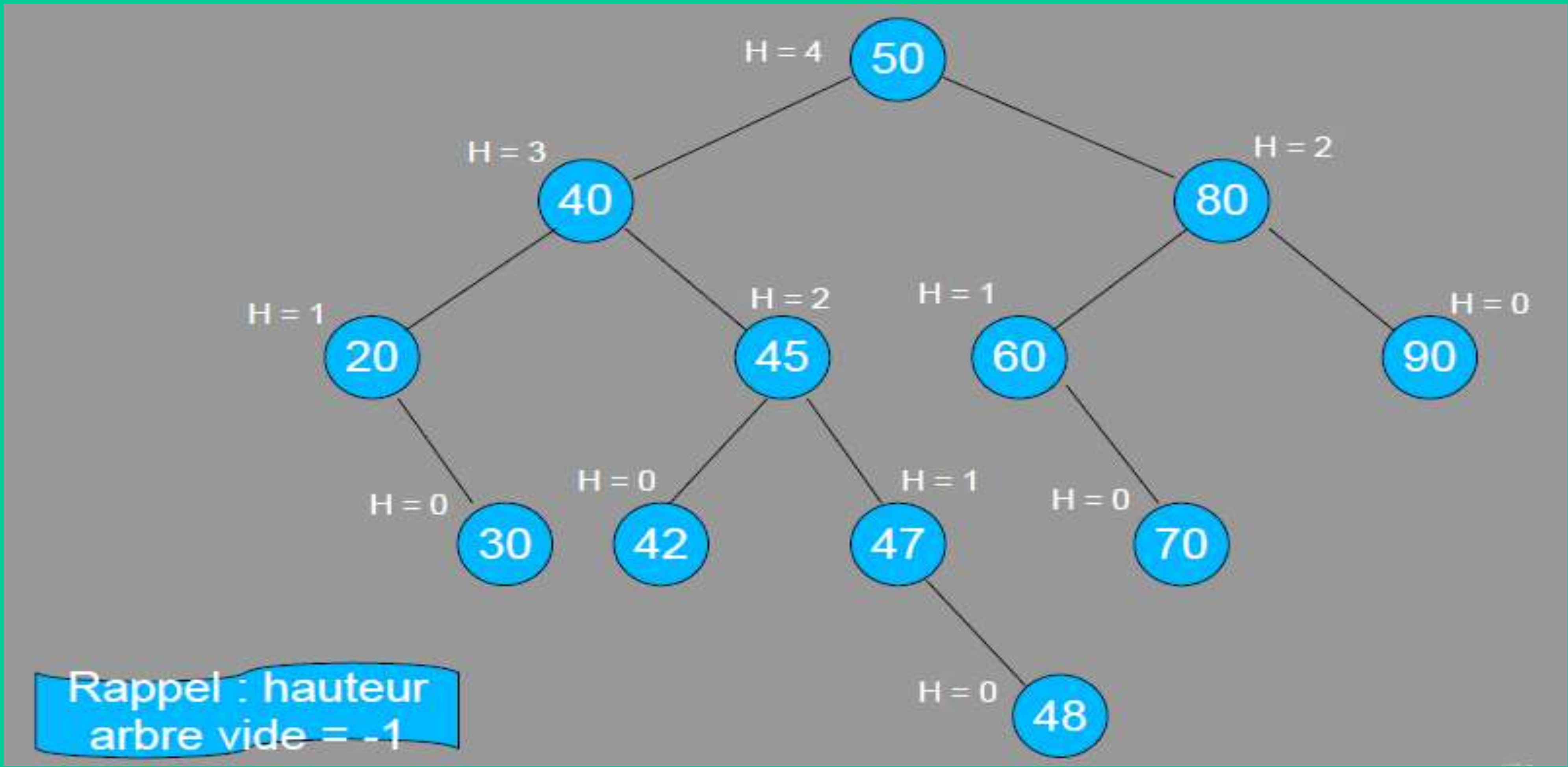
Les arbres H-équilibrés ou AVL

Les arbres AVL sont tels que pour tout nœud N, on a :

$$\forall N \in S \quad \bullet \quad | H(N_g) - H(N_d) | \leq 1$$



Exemple d'arbre AVL



Problème:

Comment maintenir un arbre relativement équilibré au fur et à mesure des **insertions** (et suppression) ?

Une solution pourrait être de passer par un **équilibrage** de l'arbre à chaque insertion, à l'aide des **rotations**.

Ces rotations sont des **transformations de base** centrées sur un nœud.

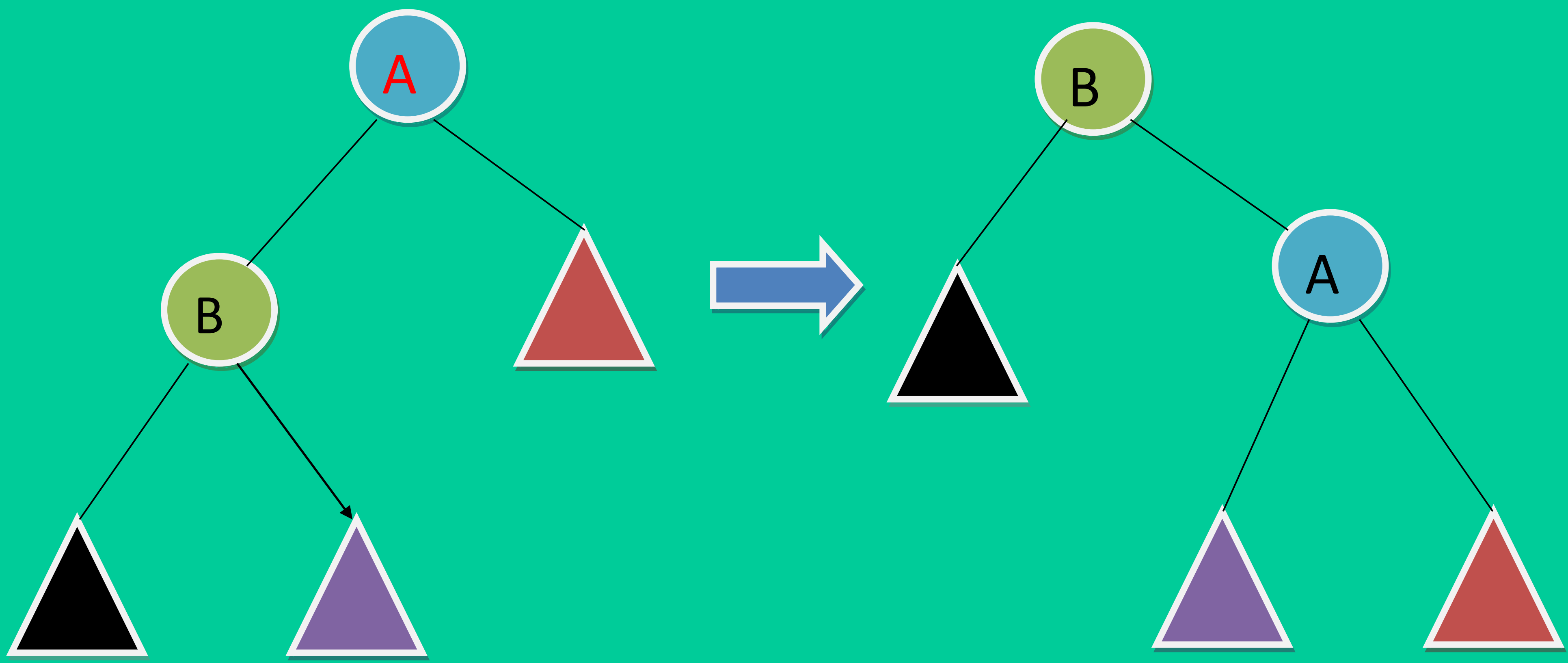
Il existe deux sortes de rotations :

-rotation gauche, notée **rg( )**

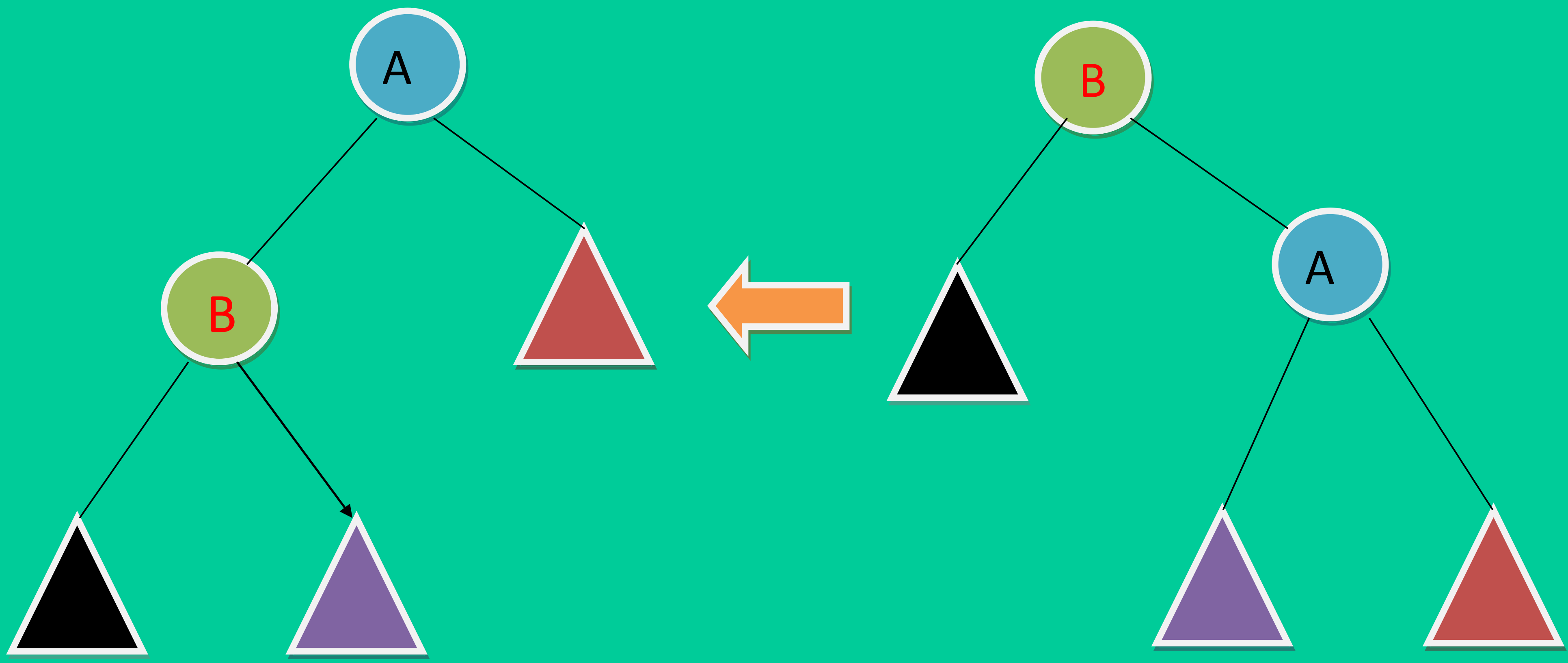
-rotation droite, notée **rd( )**

Ces deux rotations sont symétriques.

Rotation à droite autour de A notée rd(A)



Rotation à gauche autour de B notée rg(B)



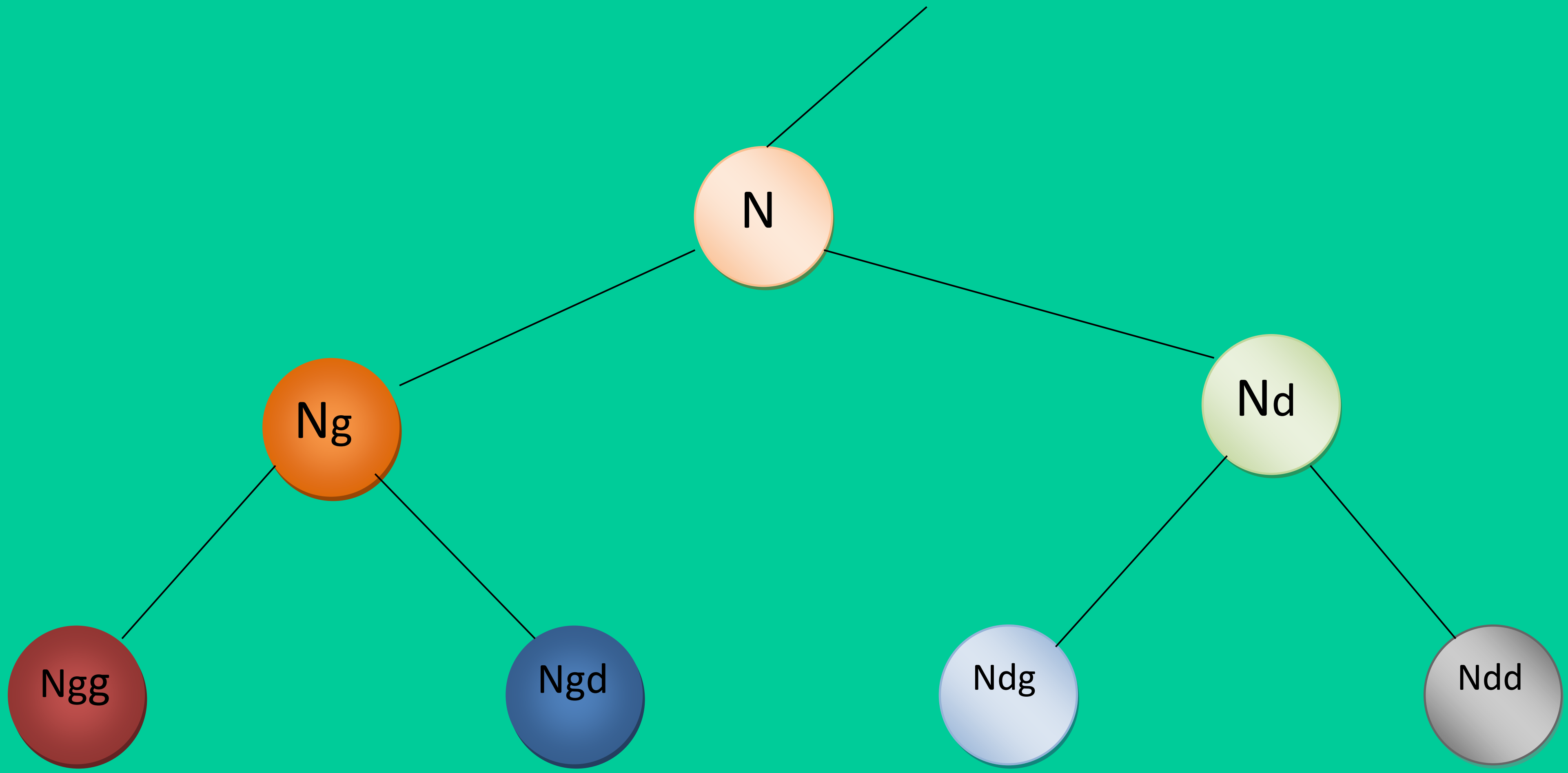
Insertion dans un arbre AVL

Le principe de l'insertion est le suivant:

- **insérer** le nœud candidat,

- au fur et à mesure de la remontée dans l'arbre depuis le nœud père à la racine, **rééquilibrer** en effectuant les **rotations** adéquates.

Quelles rotations appliquer ?



Trois cas peuvent se présenter
(N étant le nœud courant)

Cas 1:

$$| H(Ng) - H(Nd) | \leq 1$$

alors «ne rien faire»

Cas 2:

$H(Ng) - H(Nd) = 2$

si $H(Ngg)$ **>** $H(Ngd)$ alors **rd**(N)

sinon **rg**(Ng) puis **rd**(N)

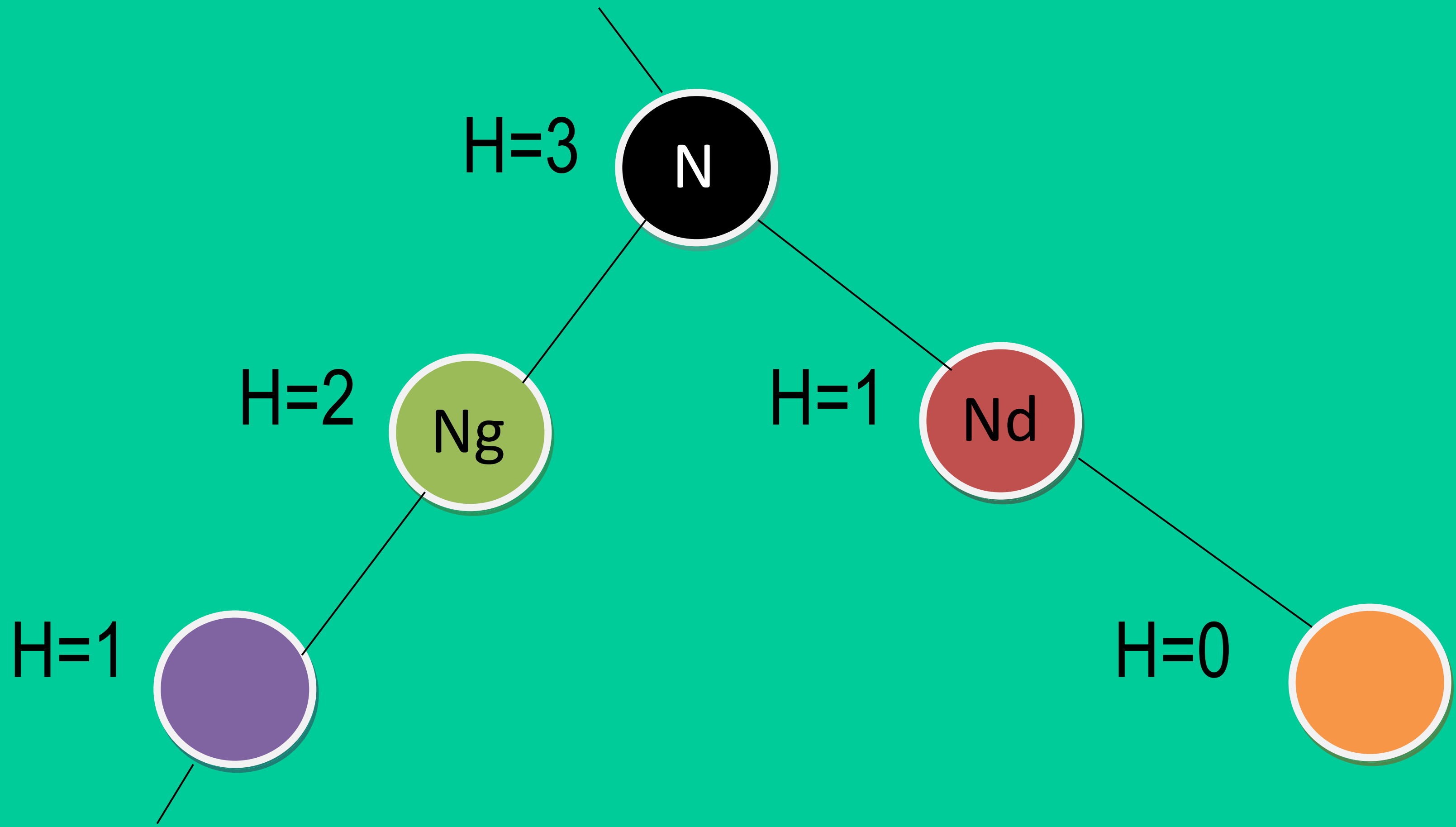
Cas 3:

$$H(Ng) - H(Nd) = -2$$

si H(Ndd) **>** H(Ndg) alors **rg**(N)

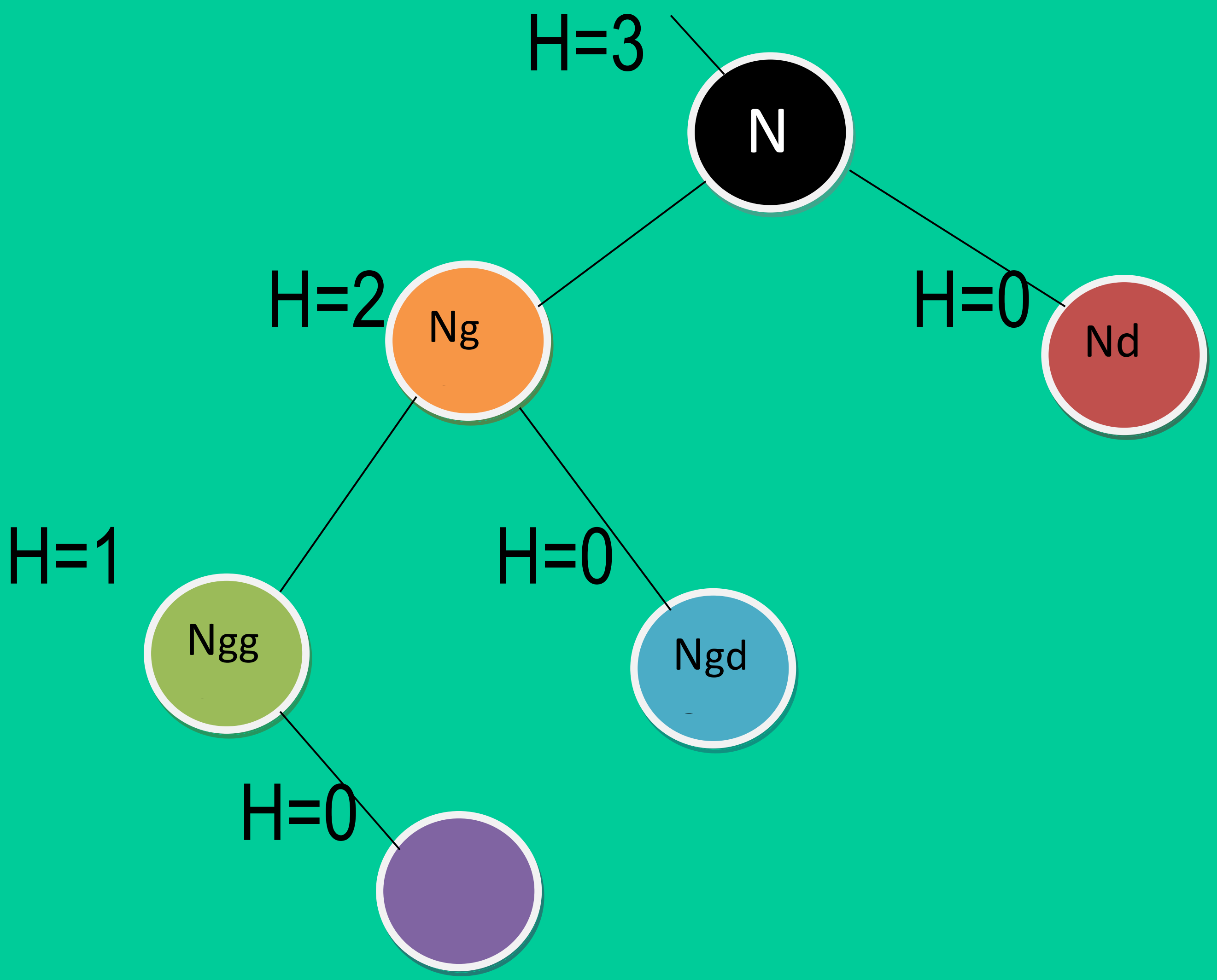
sinon **rd**(Nd) puis **rg**(N)

Cas 1 en **N**: $|H(Ng) - H(Nd)| \leq 1$

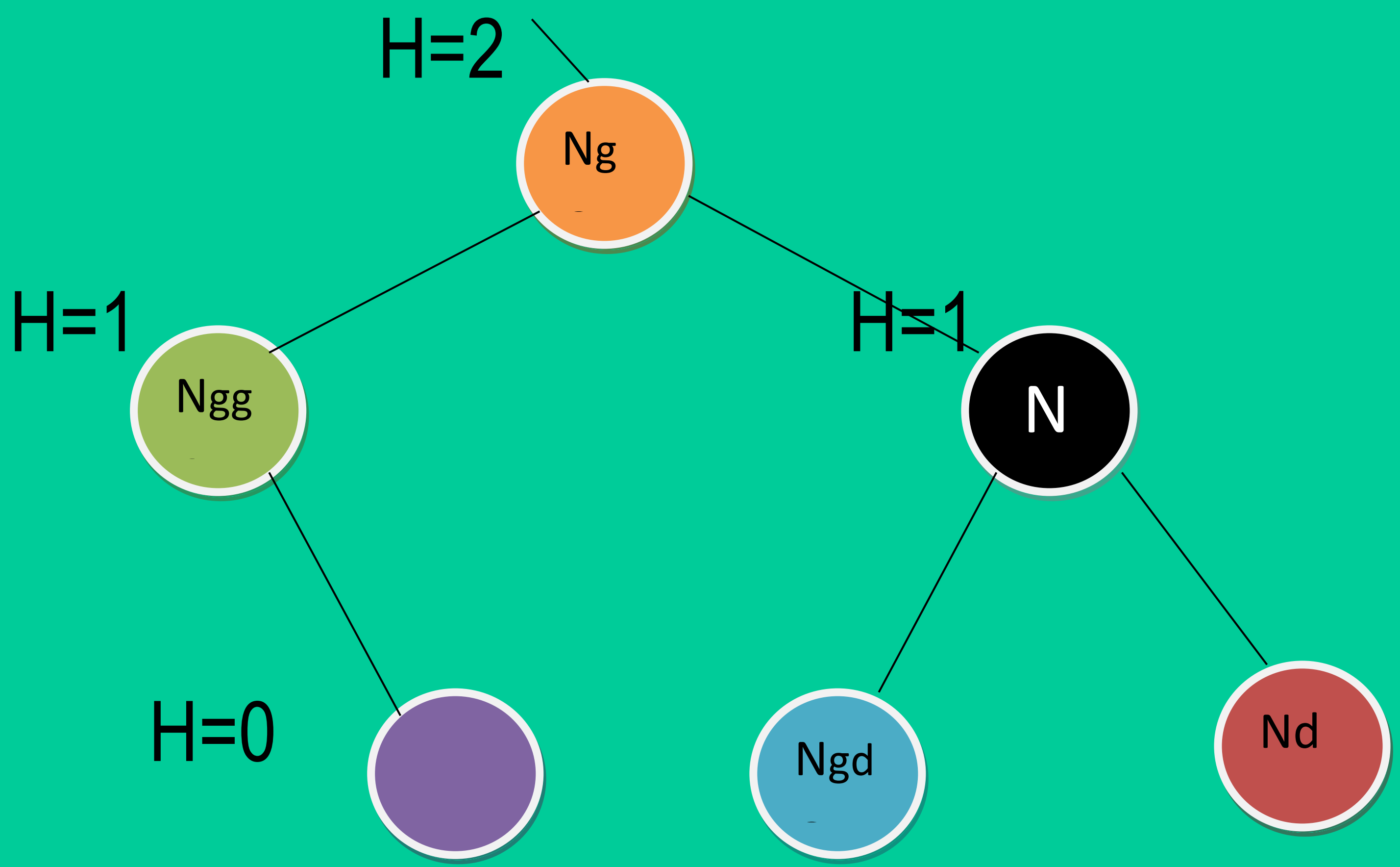


Ne rien faire !

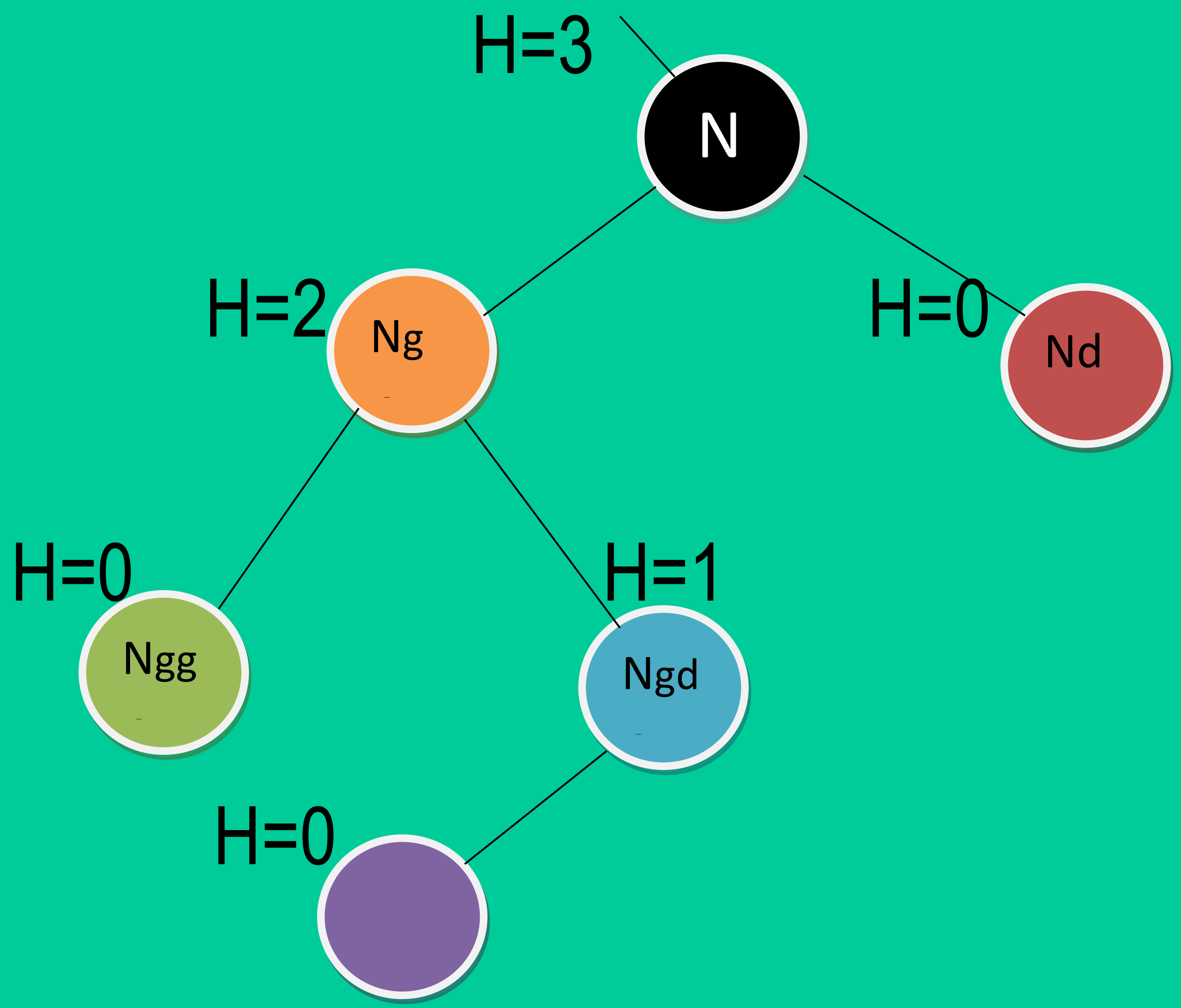
Cas 2 en **N**: H(Ng) - H(Nd) = 2 et H(Ngg) **>** H(Ngd)



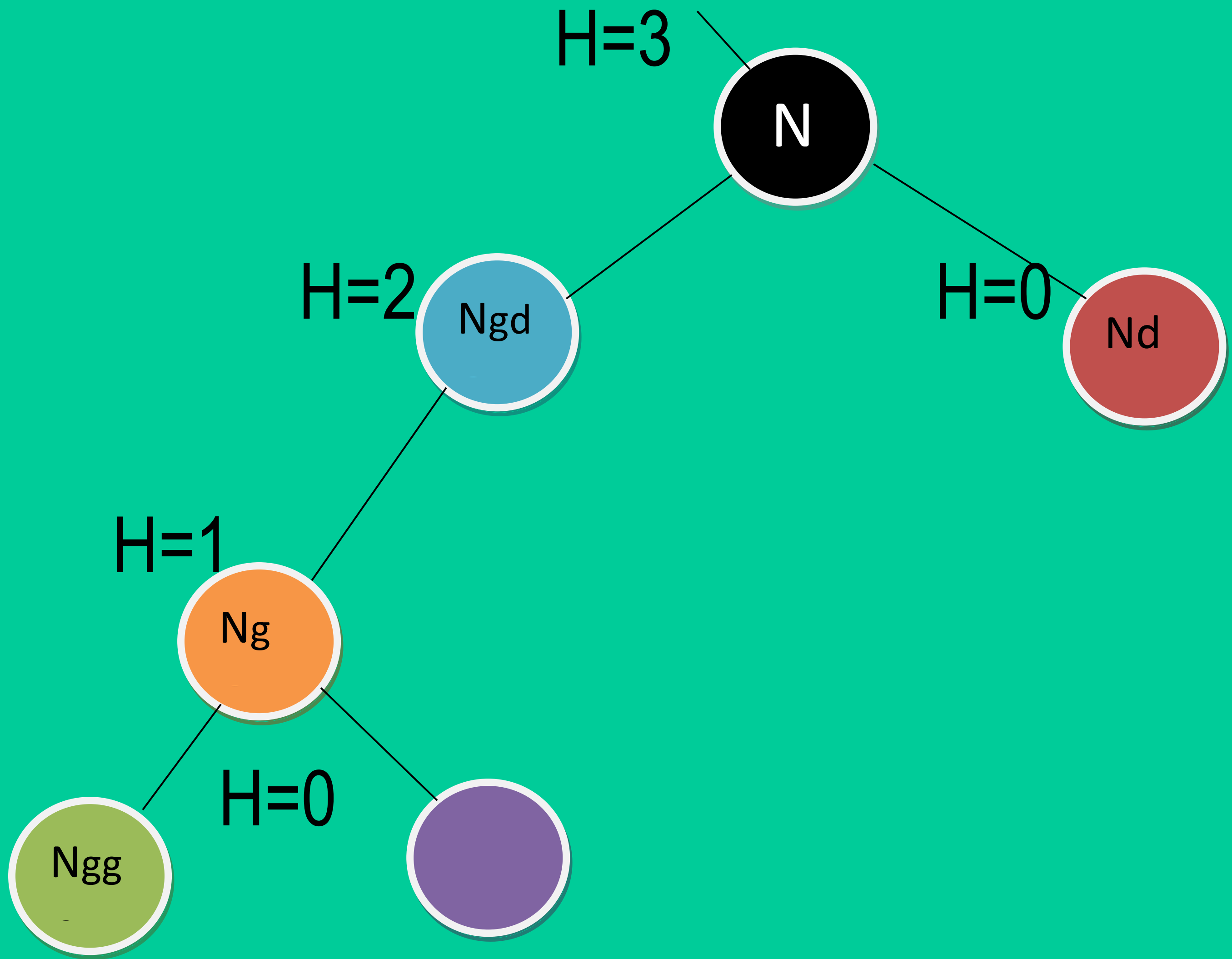
Appliquer la rotation **rd(N)**



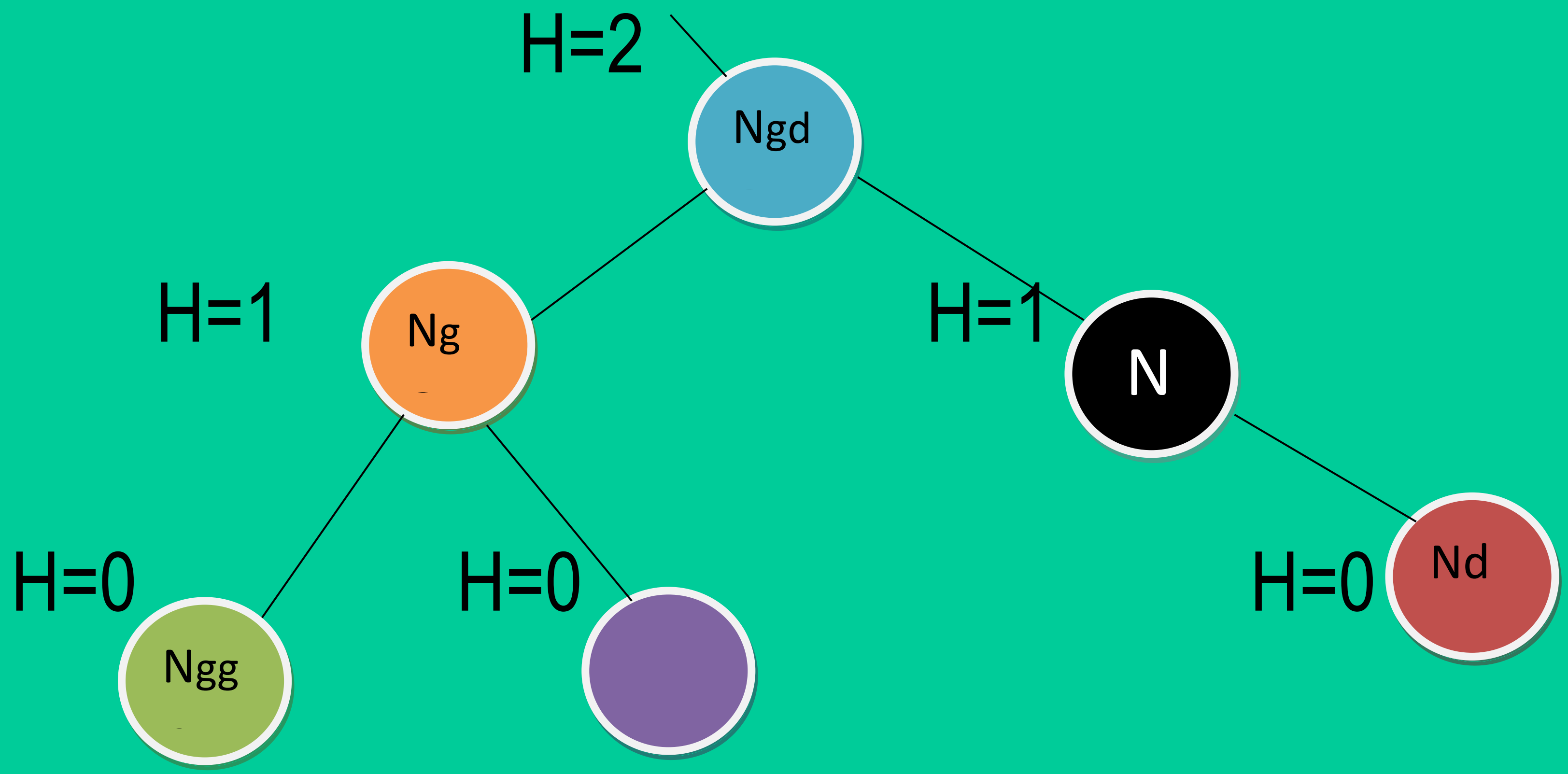
Cas 2 en N: H(Ng) - H(Nd) = 2 et H(Ngg) **<** H(Ngd)



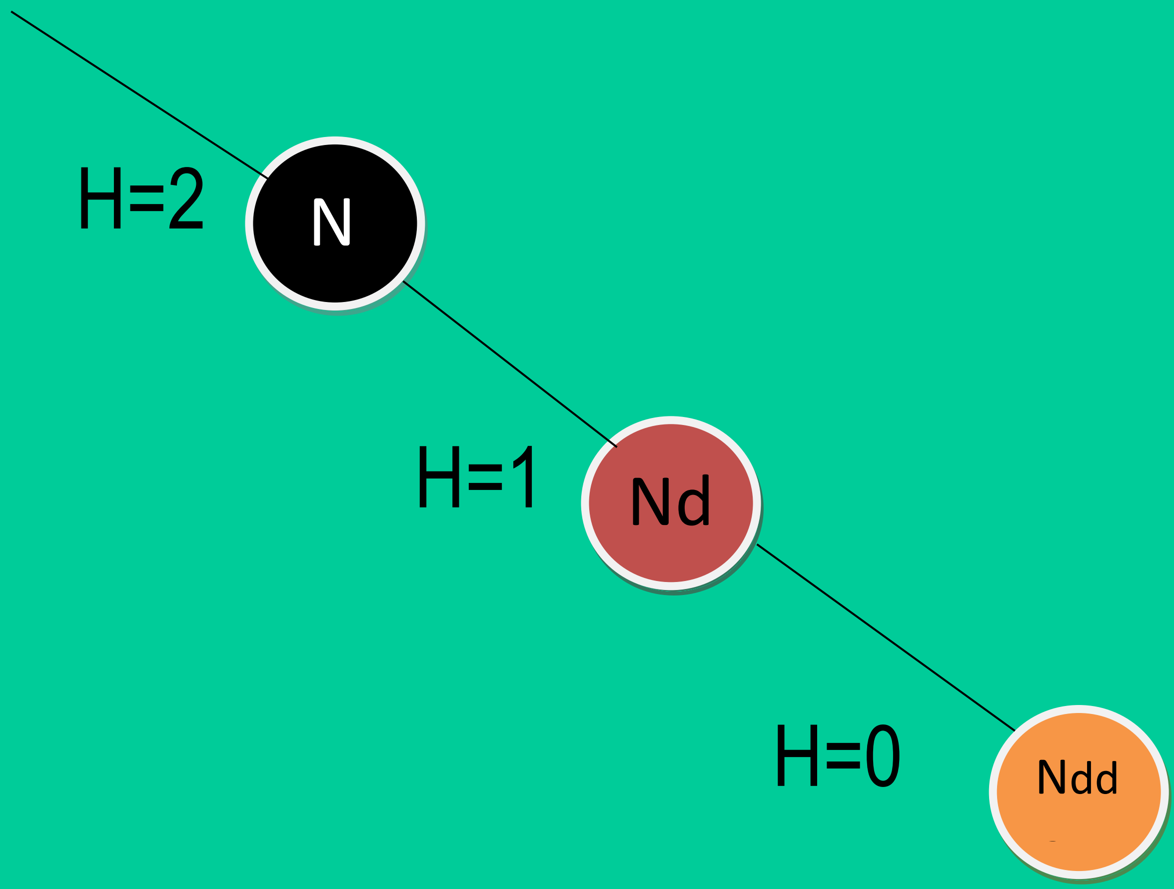
1)Rotation rg(Ng)



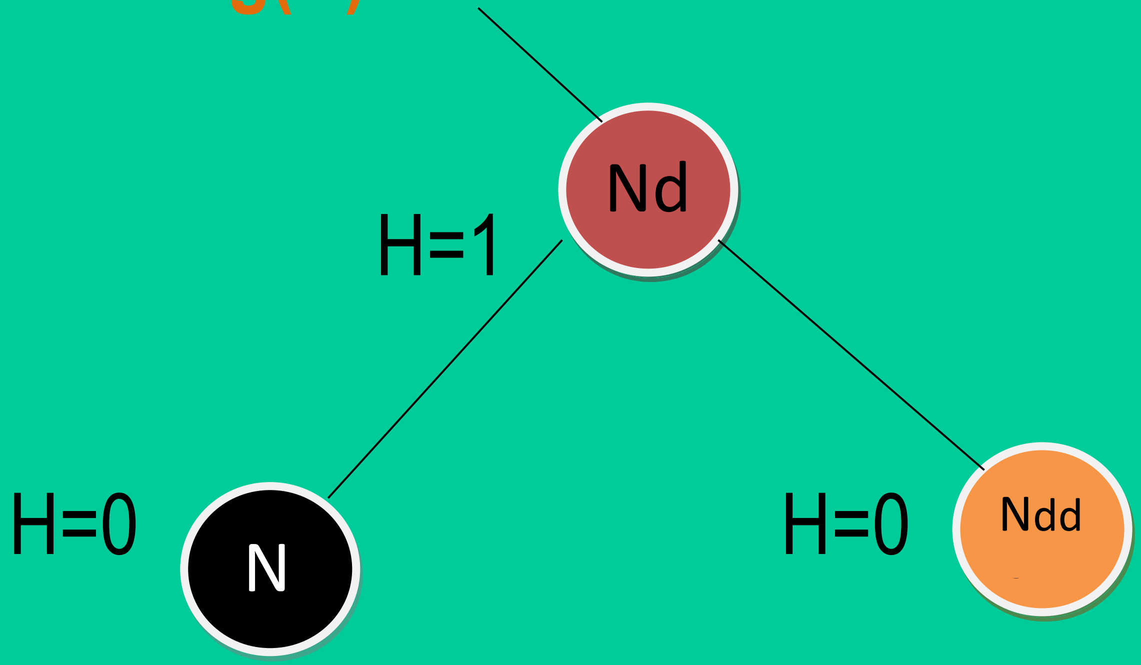
2)Rotation rd(N)



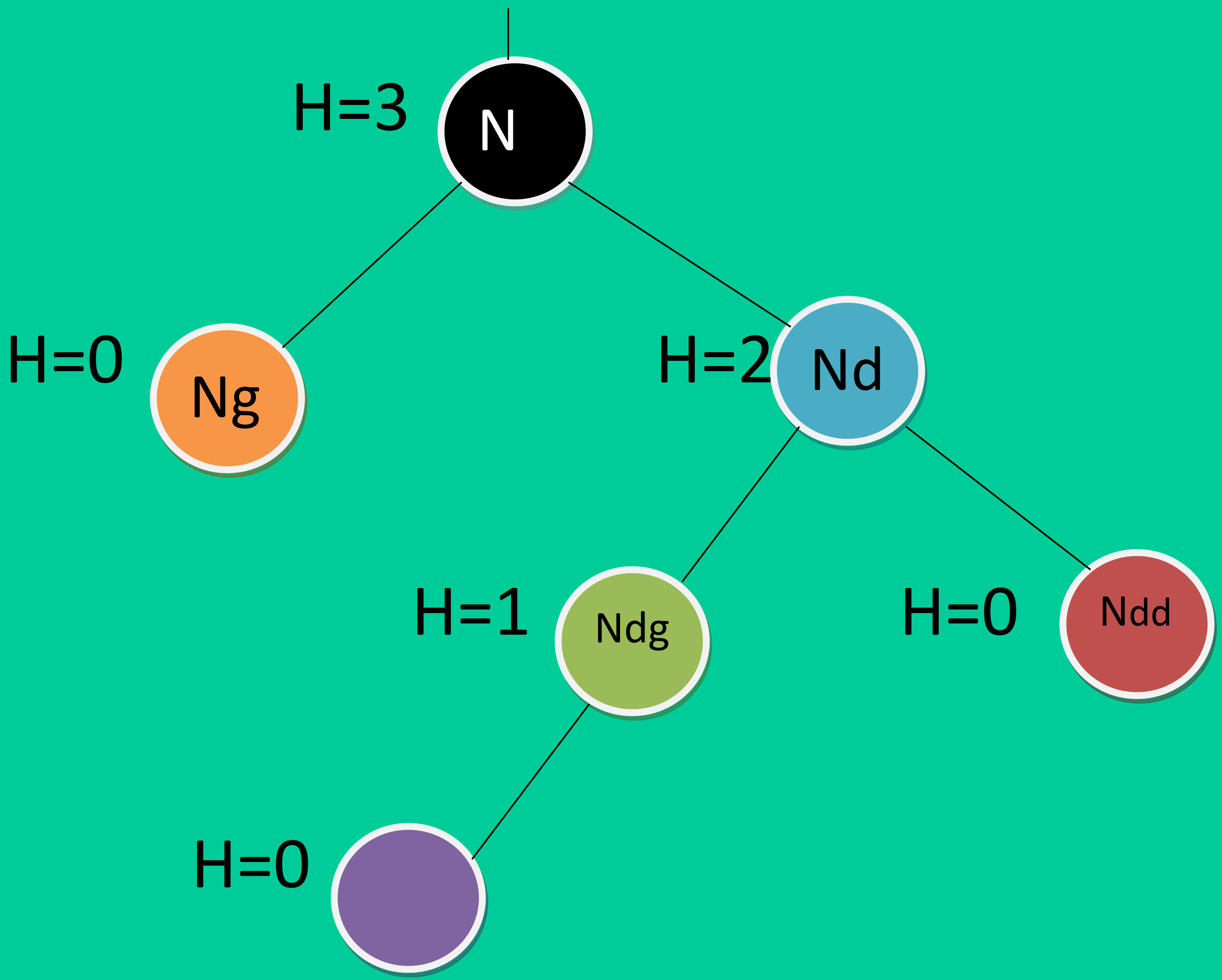
Cas 3 en N: H(Ng)- H(Nd) = -2 et H(Ndd)> H(Ndg)



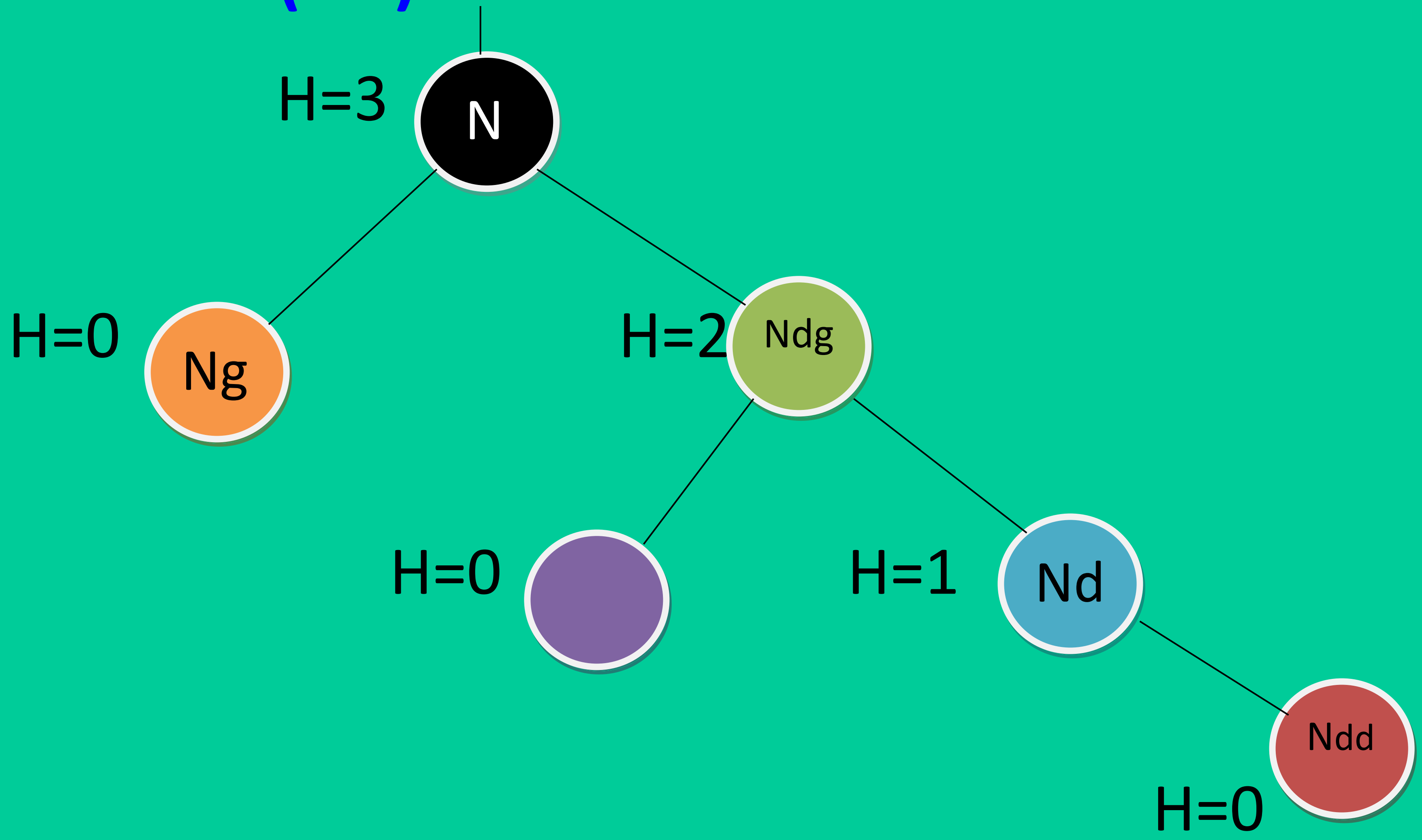
Appliquer la rotation rg(N)



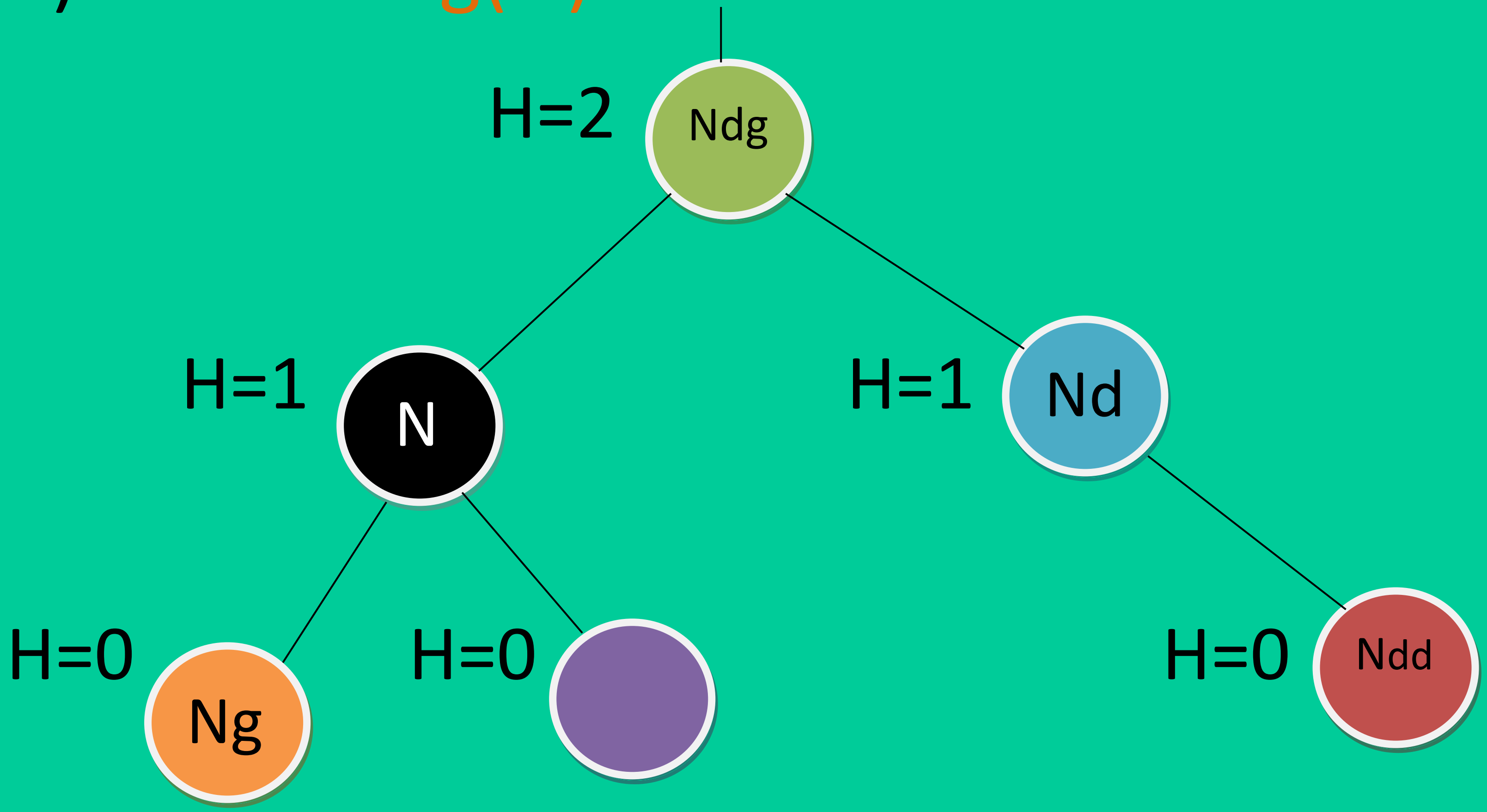
Cas 3 en N: H(Ng)- H(Nd) = -2 et H(Ndd) < H(Ndg)



1)Rotation **rd(Nd)**

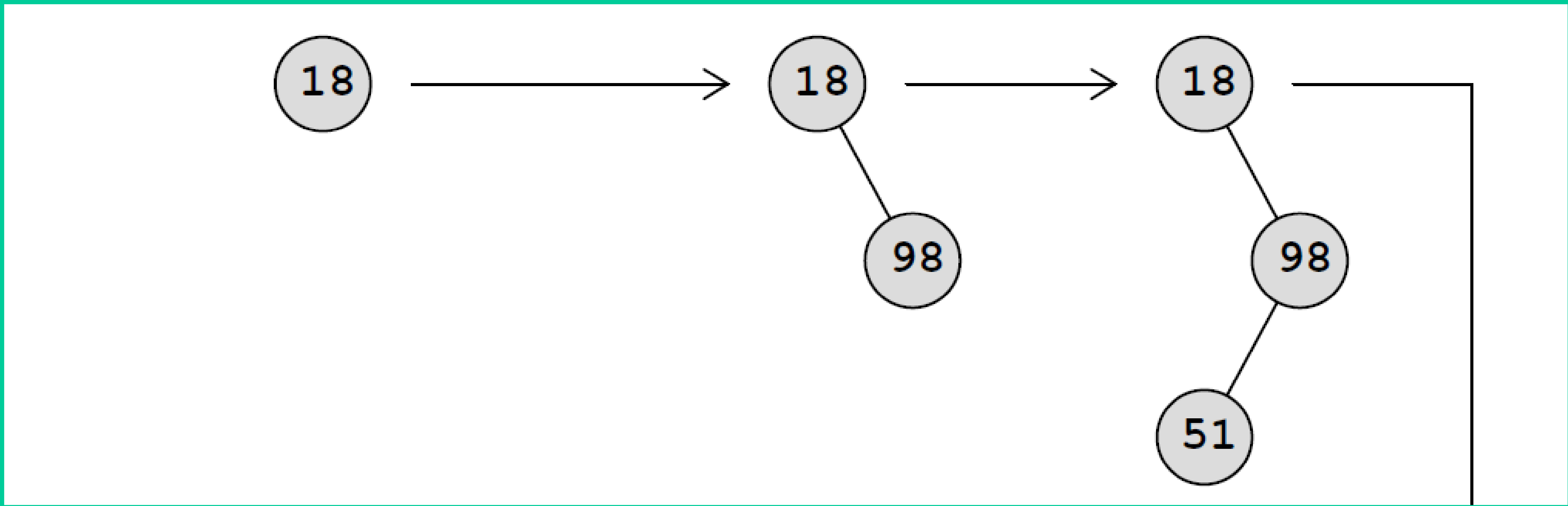


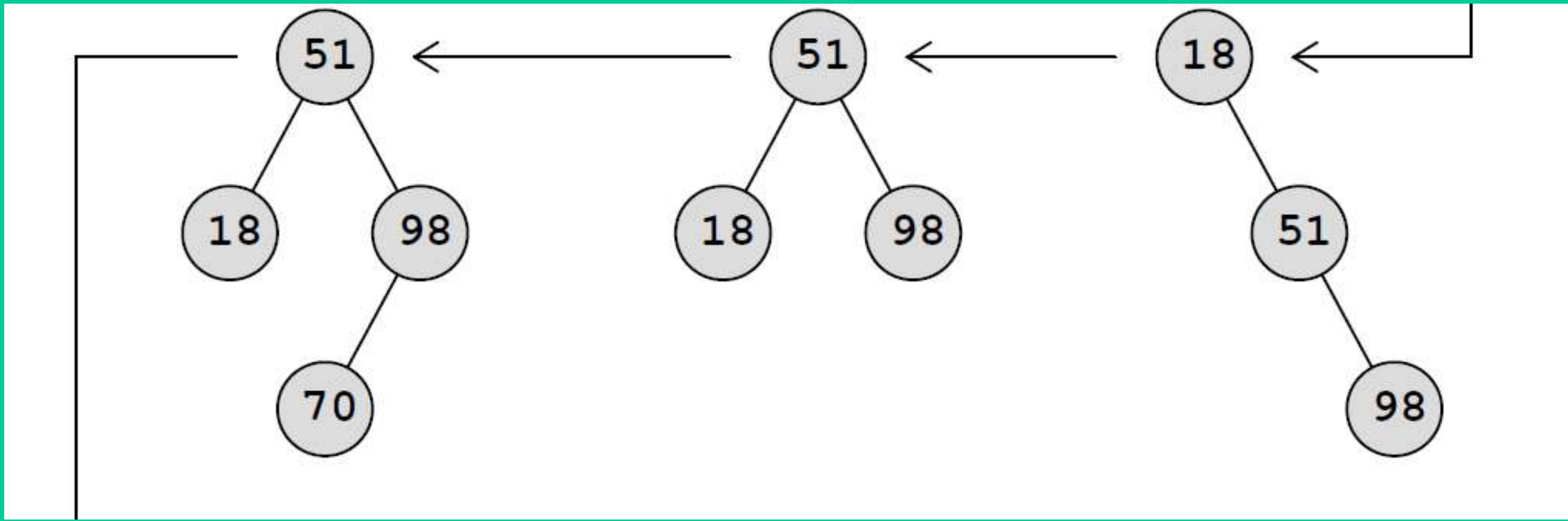
2)Rotation rg(N)

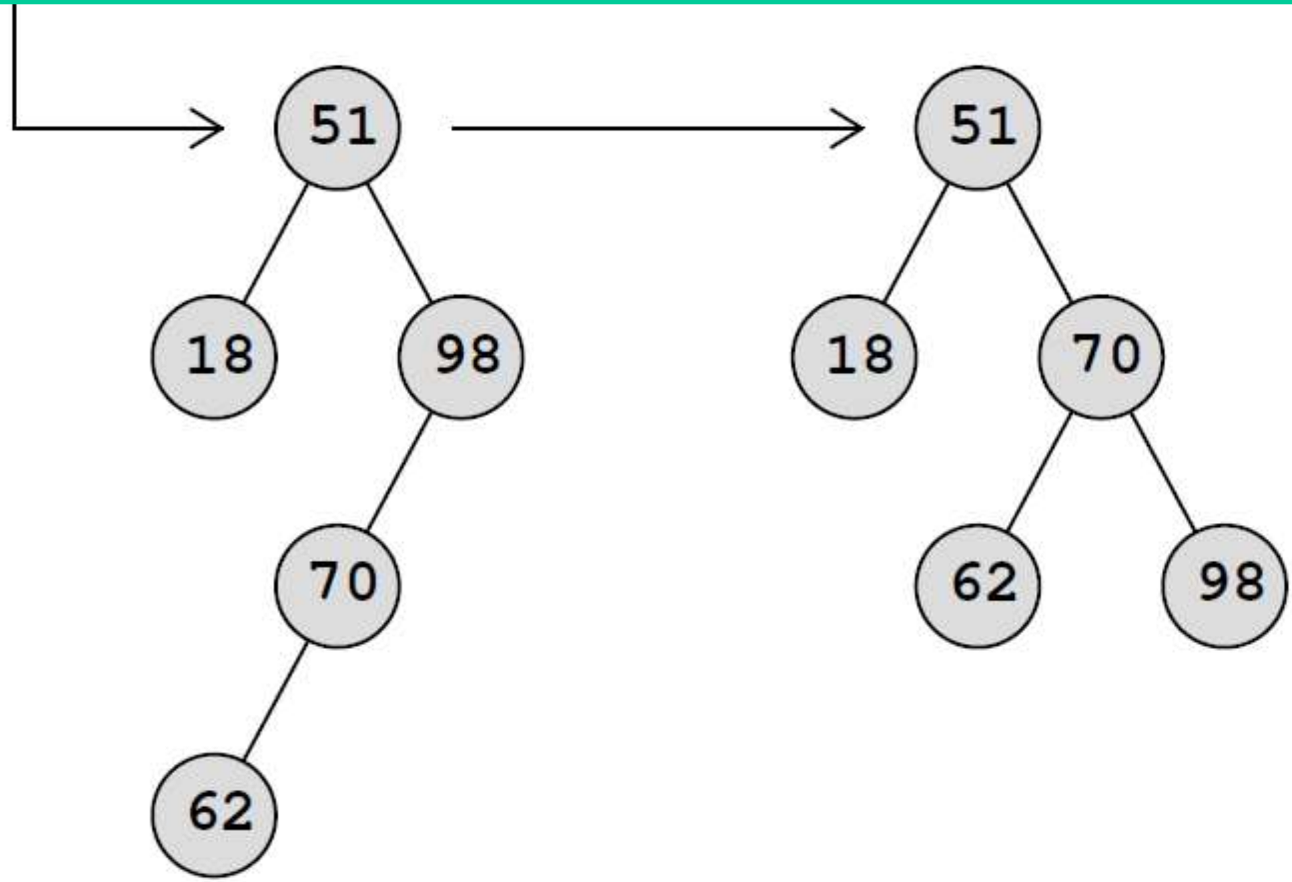


Exemple:

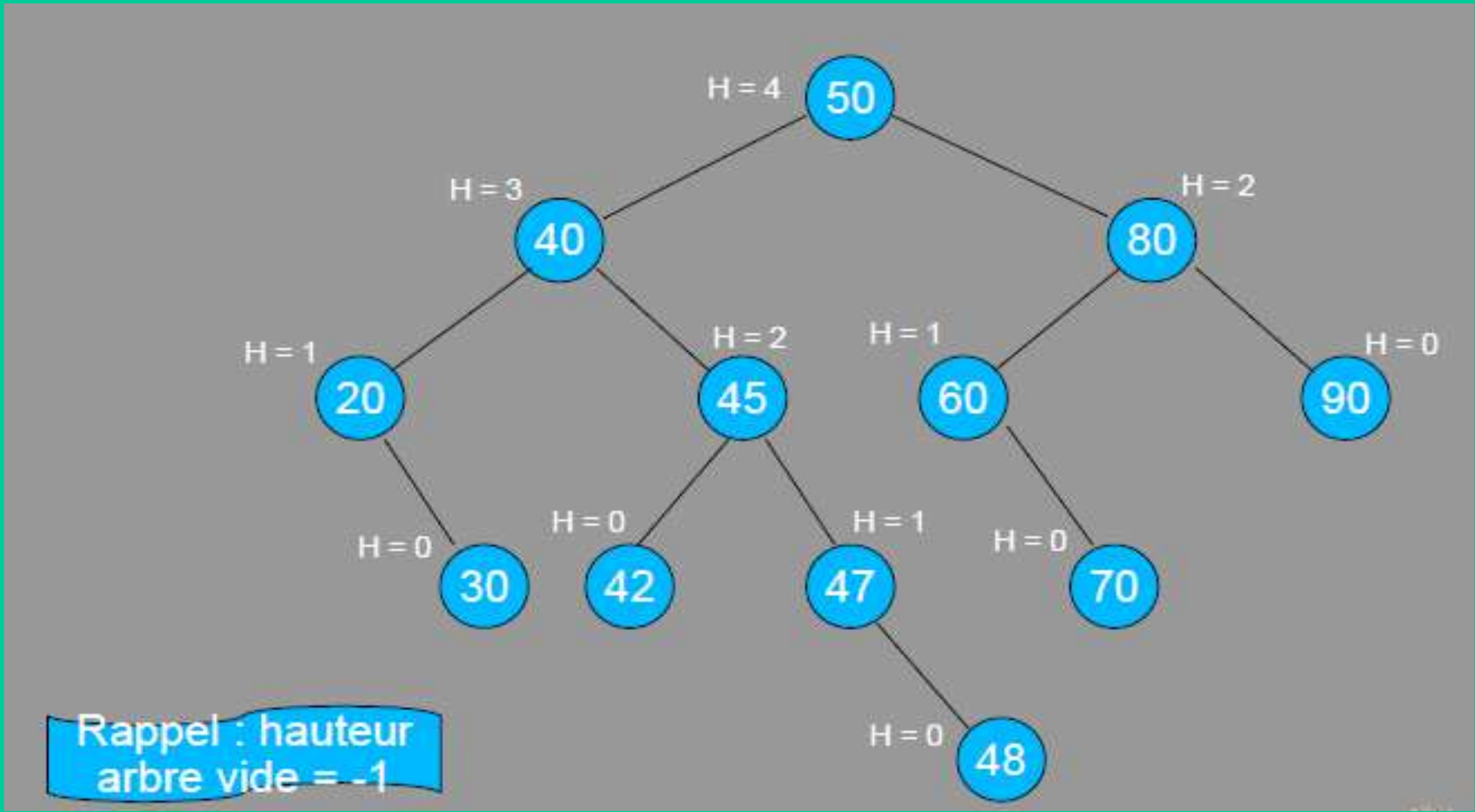
Insérer 18, 98, 51, 70, 62 dans un arbre vide.



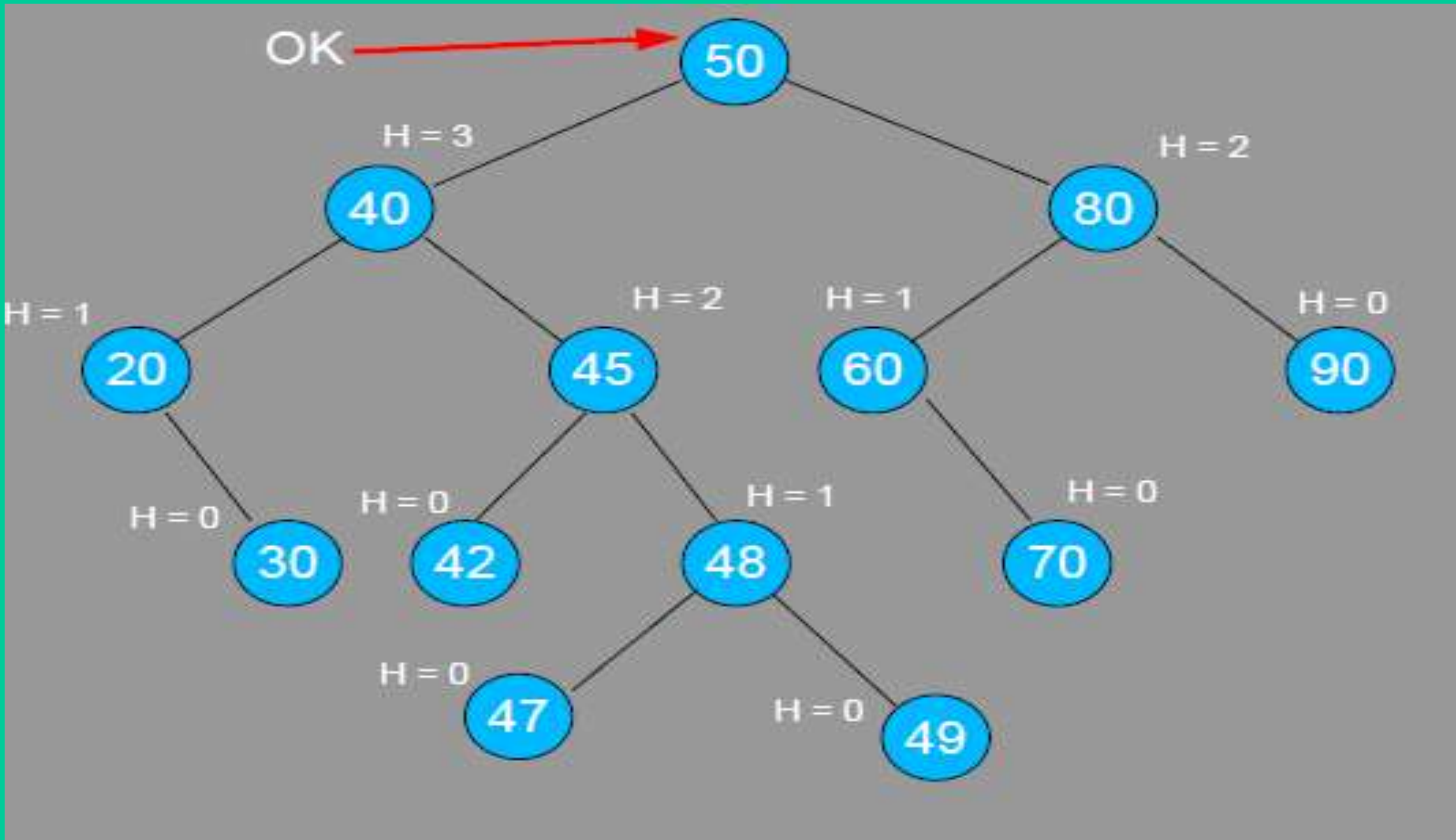

51
18
98
70
51
18
98
18
51
98


51
18
98
70
62
51
18
70
62
98

Ajouter 49 à l’arbre suivant :



Résultat :


OK
50
H = 3
40
H = 2
80
H = 1
20
H = 2
45
H = 1
60
H = 0
90
H = 0
30
H = 0
42
H = 1
48
H = 0
70
H = 0
47
H = 0
49

Ajouter 46 à l’arbre précédent :