IFT339
Structures de données

Thème 12 : Monceaux (Tas), Monceaux binomiaux

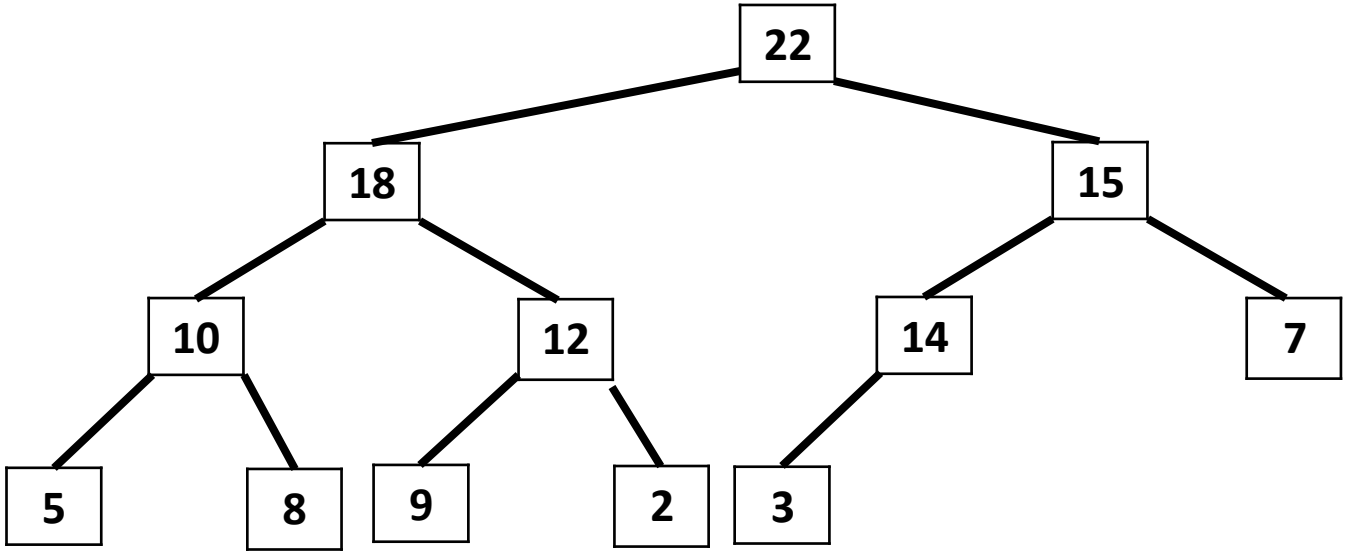
Aïda Ouangraoua

Département d'informatique

UNIVERSITÉ DE
SHERBROOKE

Monceau

- Arbre binaire complet sauf le dernier niveau qui n'est pas nécessairement complet
- Tout élément a une valeur supérieure à celle de ses descendants
- Utile pour représenter des files de priorité



Insertion de x dans un monceau

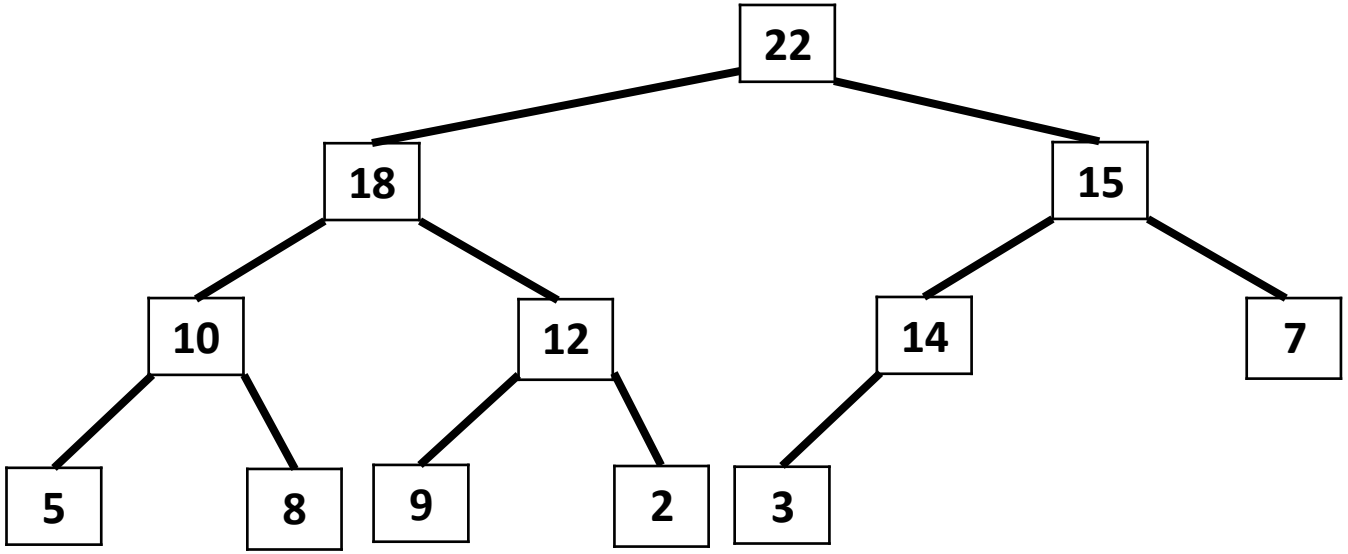
- Insérer x dans une nouvelle feuille à droite au dernier niveau
- Faire percoler x vers le haut à la bonne place en répétant:
  - Si x est supérieur à son parent p, échanger x et p

Complexité en temps O(log(n))

Insertion de x dans un monceau

- Insérer x dans une nouvelle feuille à droite au dernier niveau
- Faire percoler x vers le haut à la bonne place en répétant:
  - Si x est supérieur à son parent p, échanger x et p

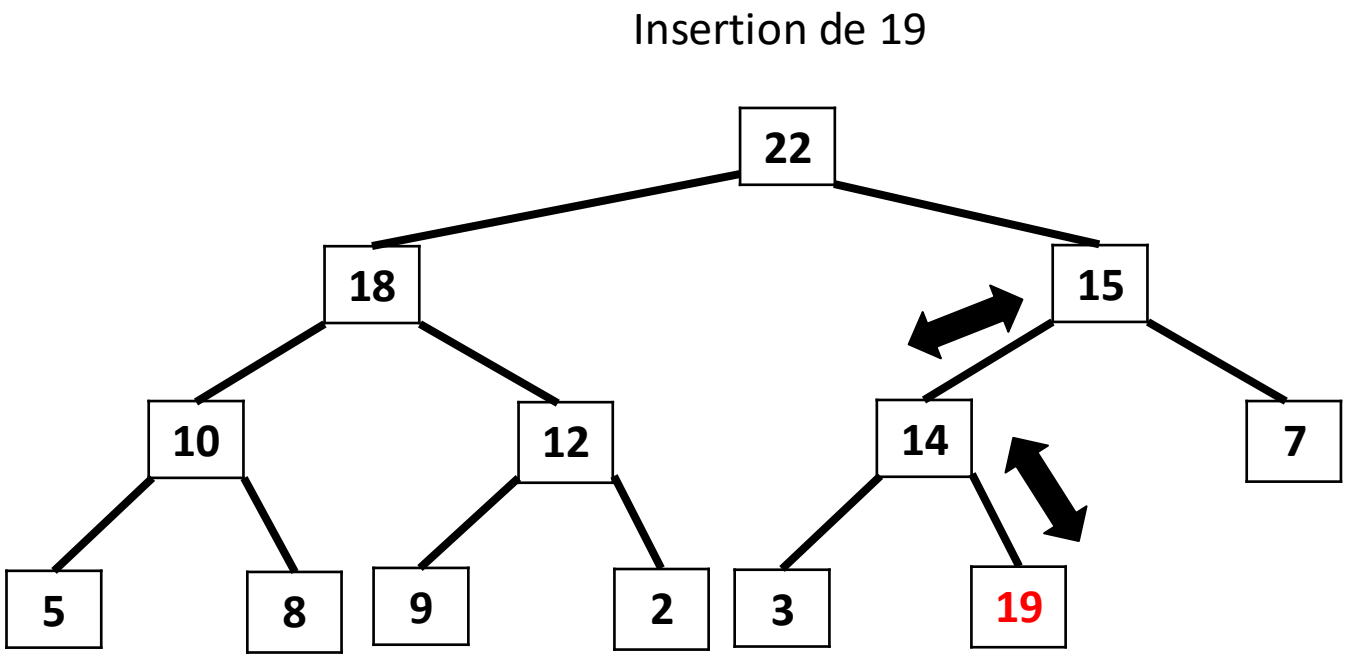
Insertion de 19



Complexité en temps O(log(n))

Insertion de x dans un monceau

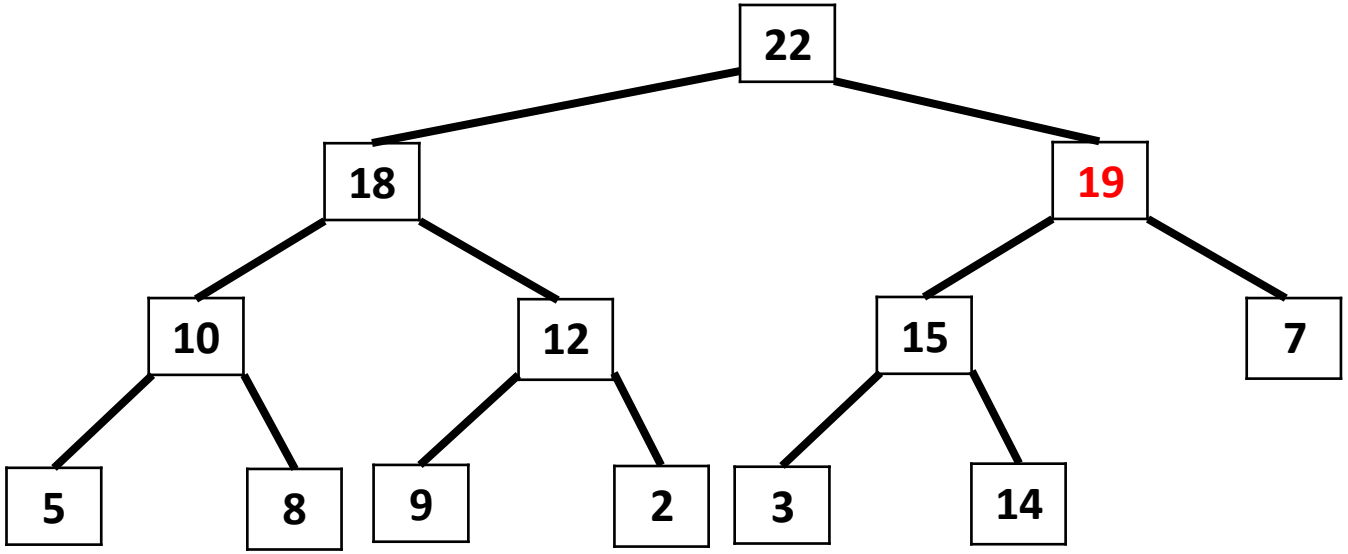
- Insérer x dans une nouvelle feuille à droite au dernier niveau
- Faire percoler x vers le haut à la bonne place en répétant:
  - Si x est supérieur à son parent p, échanger x et p



Insertion de x dans un monceau

- Insérer x dans une nouvelle feuille à droite au dernier niveau
- Faire percoler x vers le haut à la bonne place en répétant:
  - Si x est supérieur à son parent p, échanger x et p

Insertion de 19



Suppression (sommet) dans un monceau

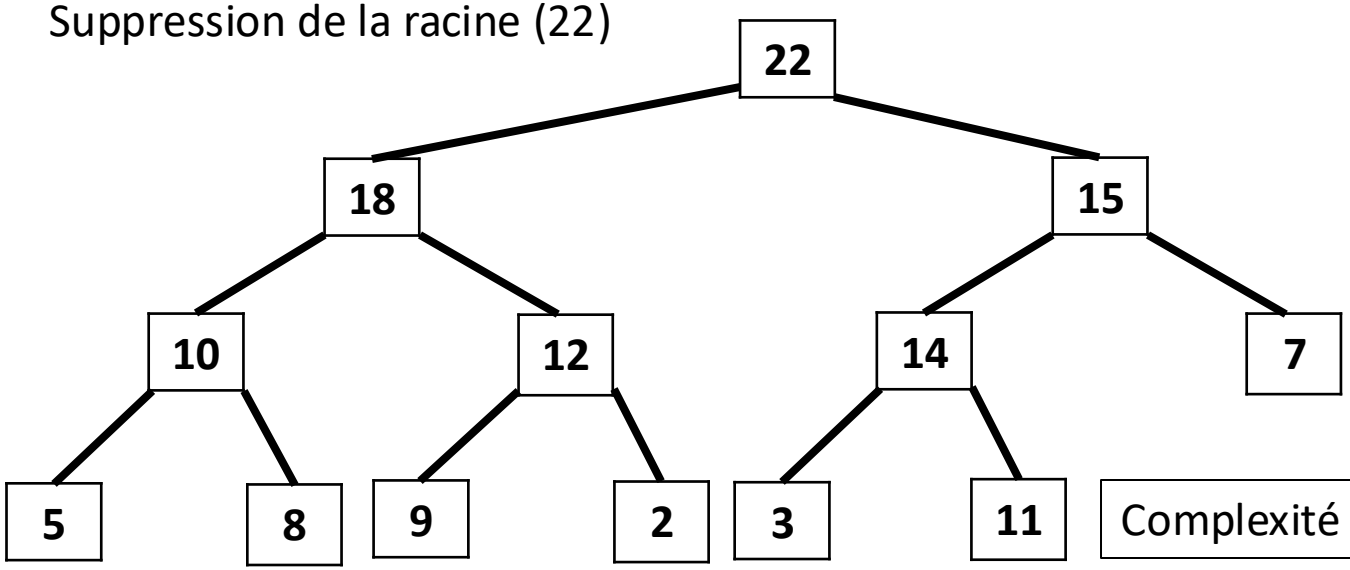
- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

Complexité en temps O(log(n))

Suppression (sommet) dans un monceau

- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

Suppression de la racine (22)

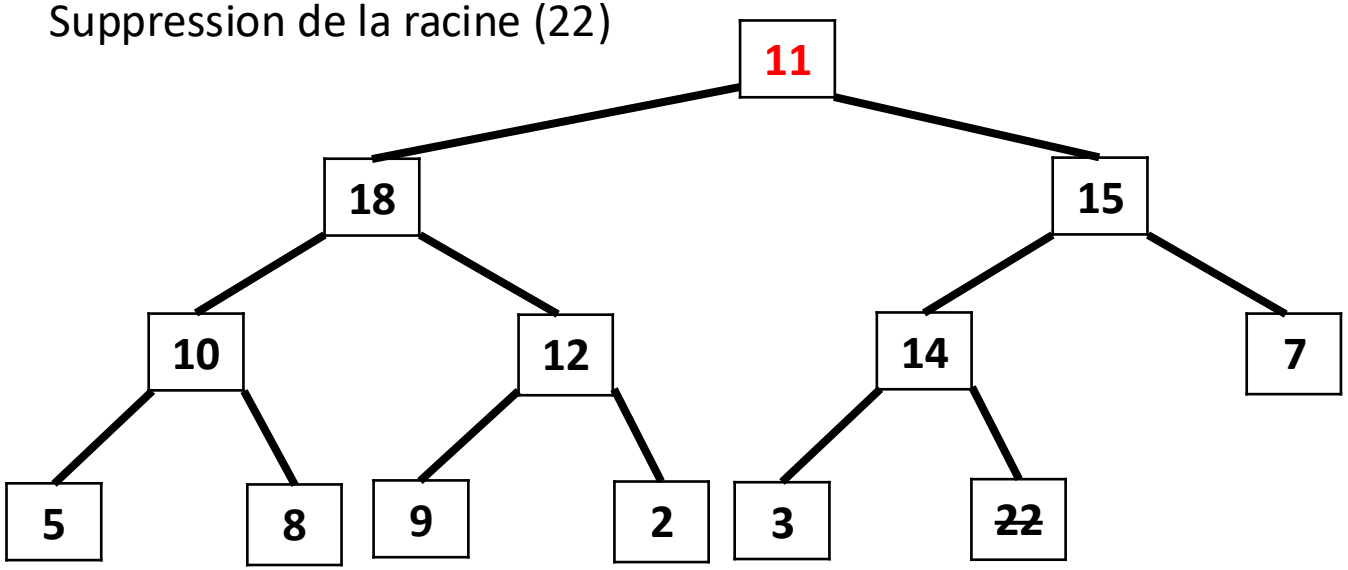


Complexité en temps O(log(n))

Suppression (sommet) dans un monceau

- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

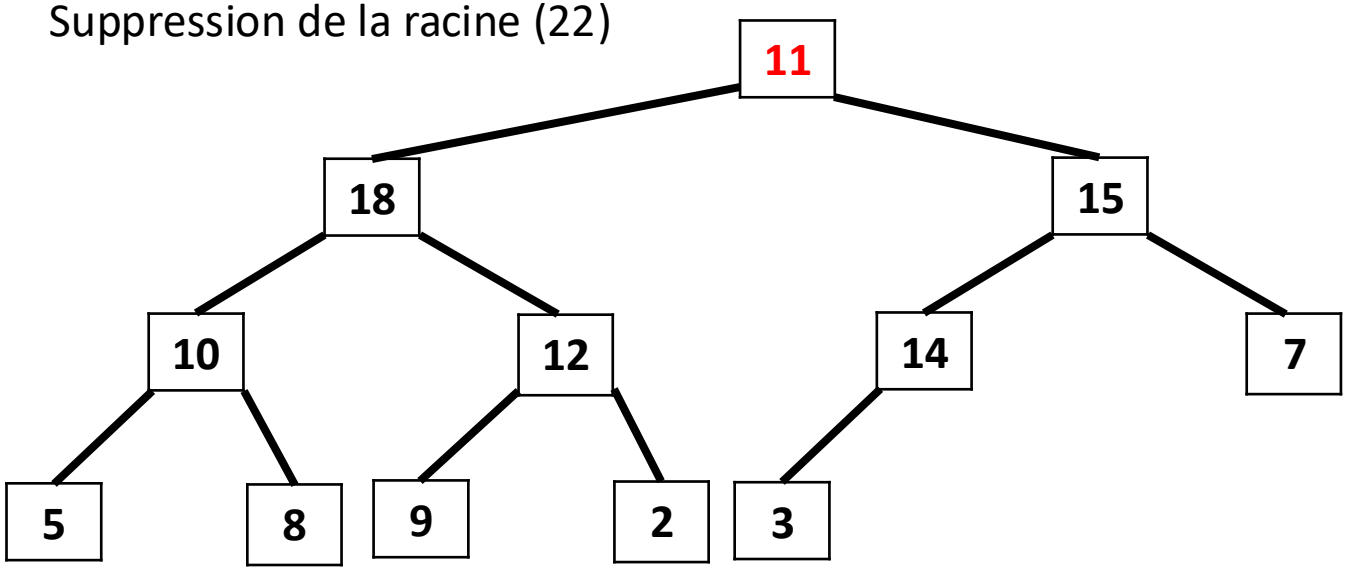
Suppression de la racine (22)



Suppression (sommet) dans un monceau

- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

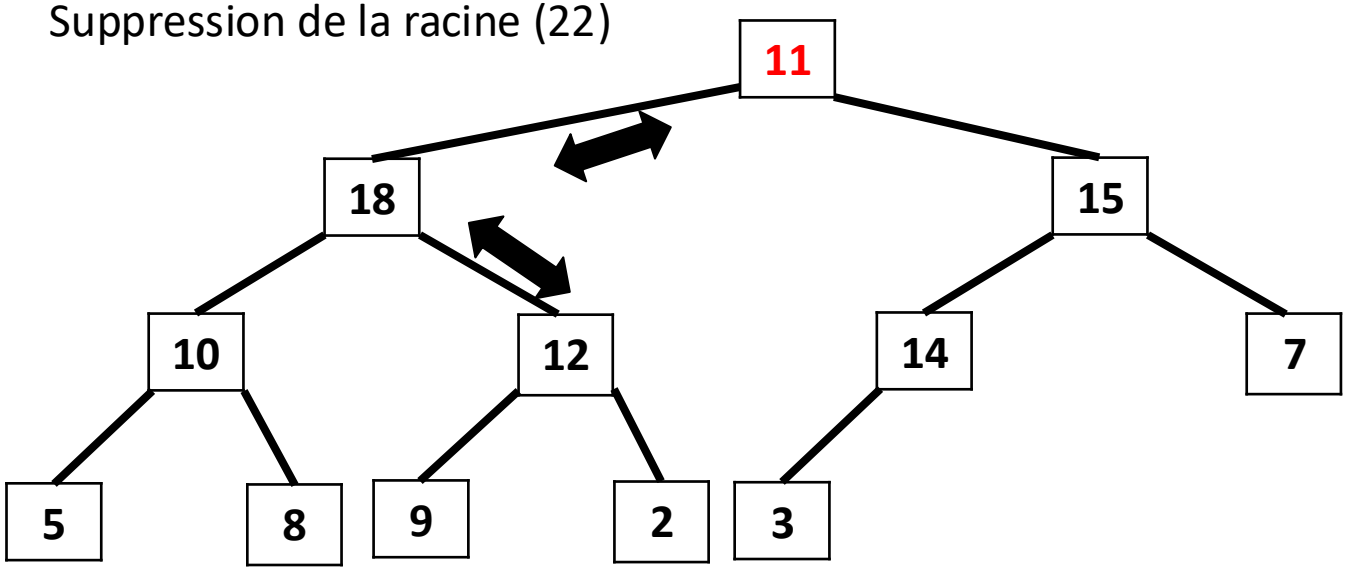
Suppression de la racine (22)



Suppression (sommet) dans un monceau

- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

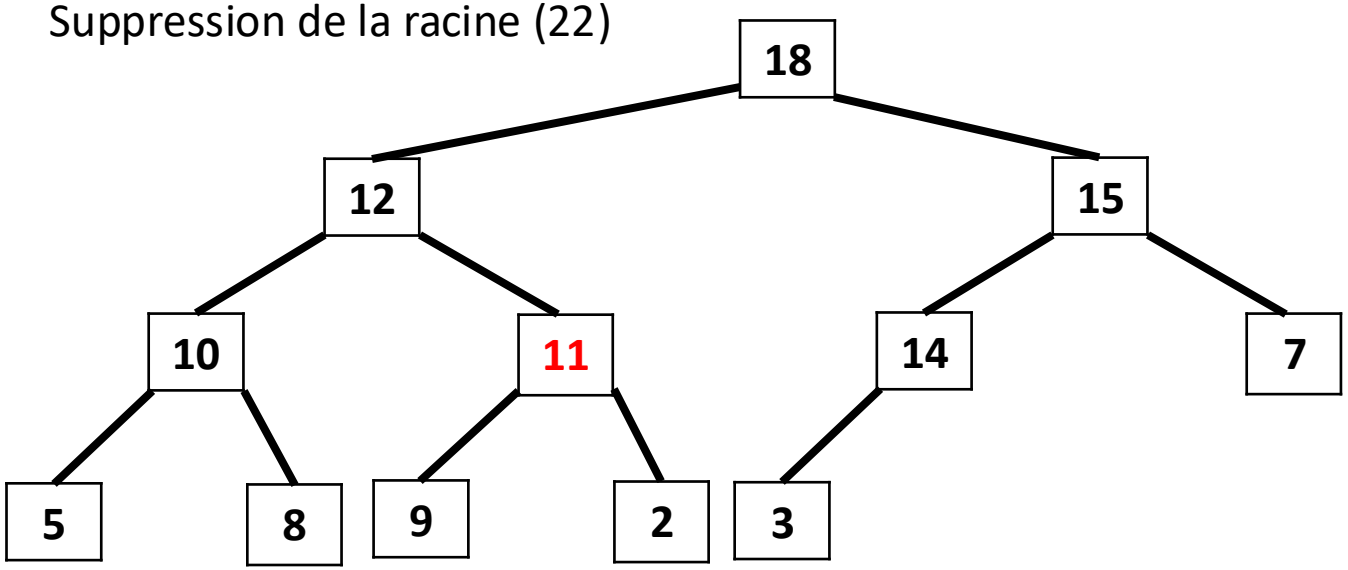
Suppression de la racine (22)



Suppression (sommet) dans un monceau

- Suppression logique: échanger la valeur avec celle du dernier nœud du dernier niveau, et supprimer ce dernier
- Faire percoler la nouvelle racine x vers le bas jusqu'à la bonne place en répétant:
  - Si x est à inférieur à au moins un de ses enfants, échanger x avec la valeur de son plus grand enfant y

Suppression de la racine (22)



Trier un ensemble E en utilisant un monceau (heap sort)

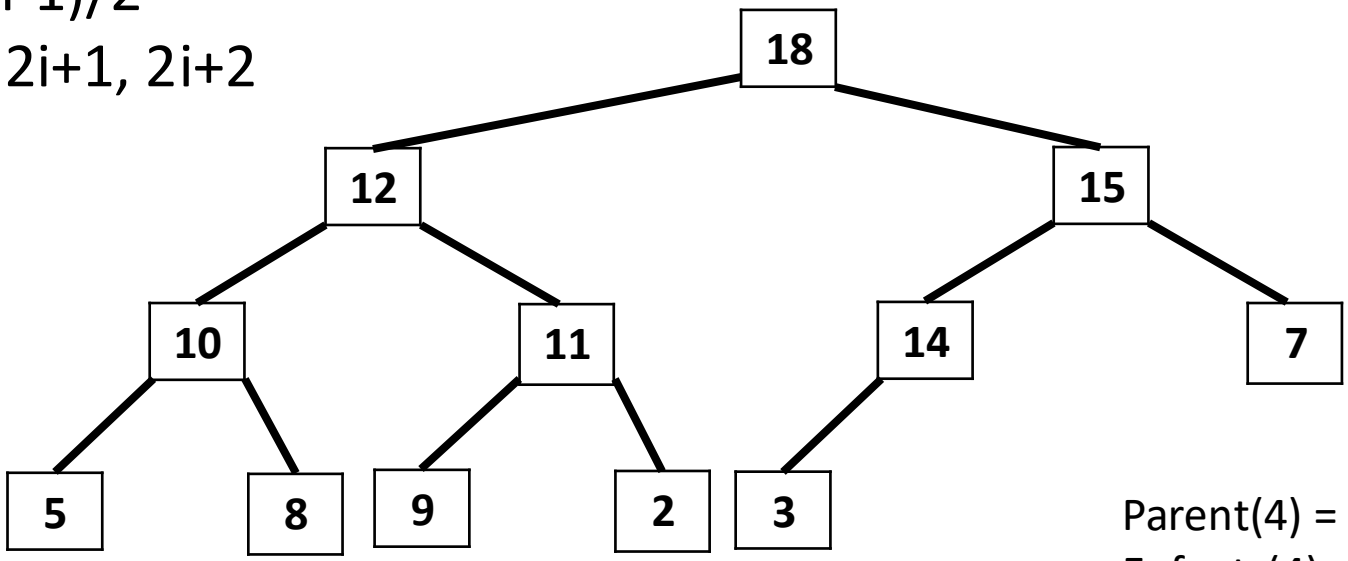
- (1) Créer un monceau vide
- (2) Insérer les éléments de E un à un dans le monceau
- (3) Retirer les éléments un à un

Complexité en temps O(nlog(n))

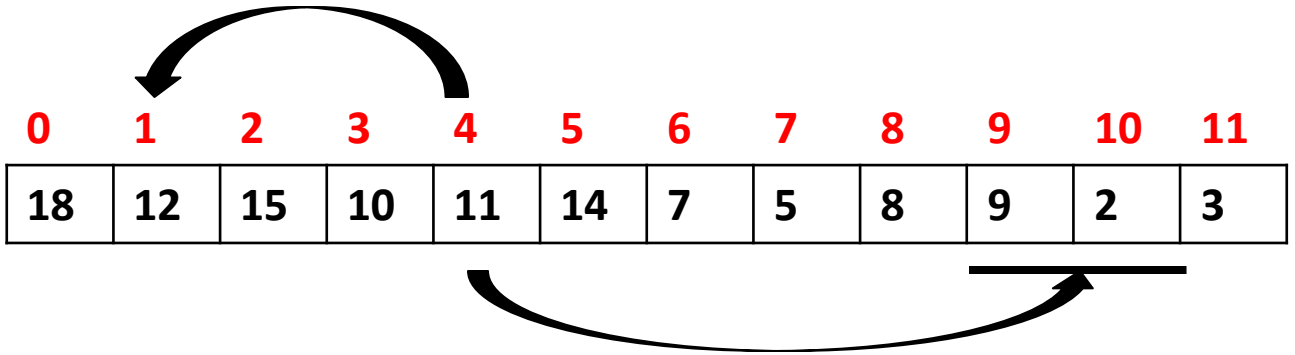
Représentation d'un monceau

- Dans un tableau (élément dans l'ordre du parcours en largeur)

- Parent(i) = (i-1)/2
- Enfants(i) = 2i+1, 2i+2



Parent(4) = 3/2 =1
Enfants(4) = 9,10



Insertion de 17

0	1	2	3	4	5	6	7	8	9	10	11
18	12	15	10	11	14	7	5	8	9	2	3

0	1	2	3	4	5	6	7	8	9	10	11	12
18	12	15	10	11	14	7	5	8	9	2	3	17

Parent(12) = (12-1)/2
= 5

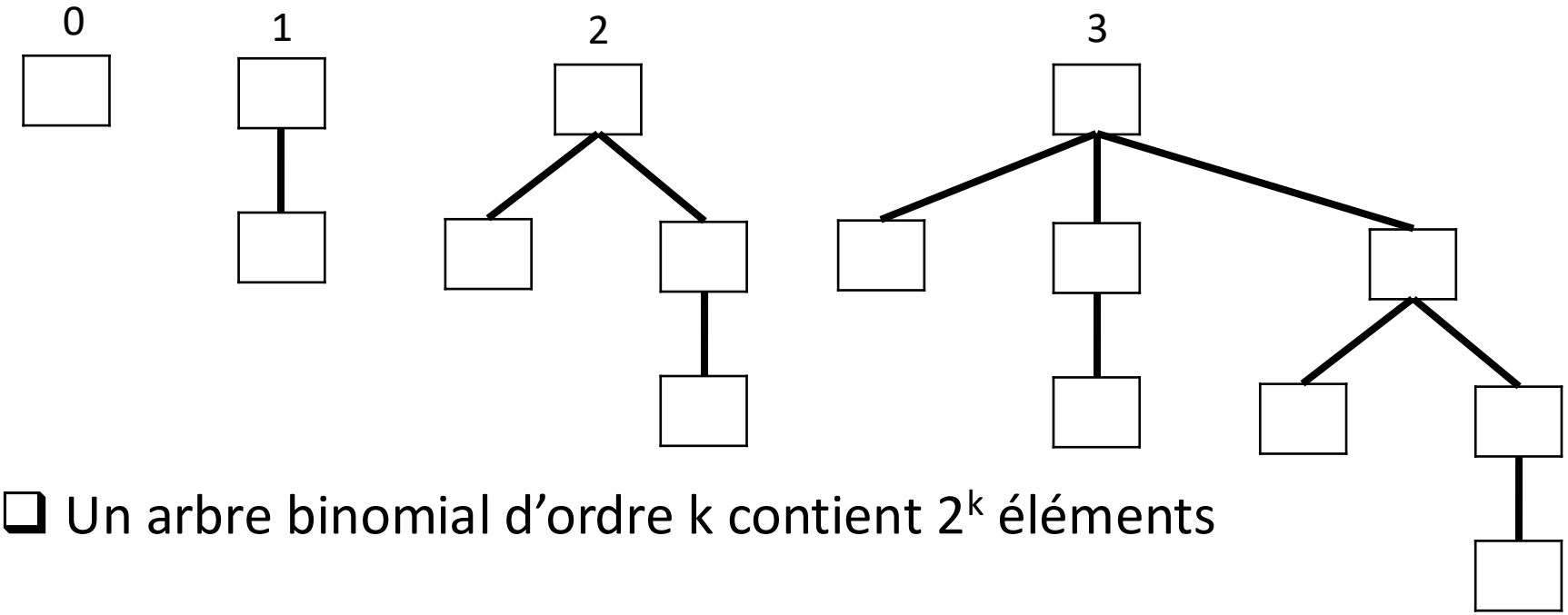
0	1	2	3	4	5	6	7	8	9	10	11	12
18	12	15	10	11	17	7	5	8	9	2	3	14

Parent(5) = (5-1)/2
= 2

0	1	2	3	4	5	6	7	8	9	10	11	12
18	12	17	10	11	15	7	5	8	9	2	3	14

Monceau binomial

- **Objectif**: fusion de deux monceaux en temps O(log(n))
- **Arbre binomial**
  - Définition récursive:
    - Ordre 0 : un seul nœud
    - Ordre k : une racine avec k sous-arbres qui sont des arbres binomiaux d'ordre 0, 1, 2, …, k-1

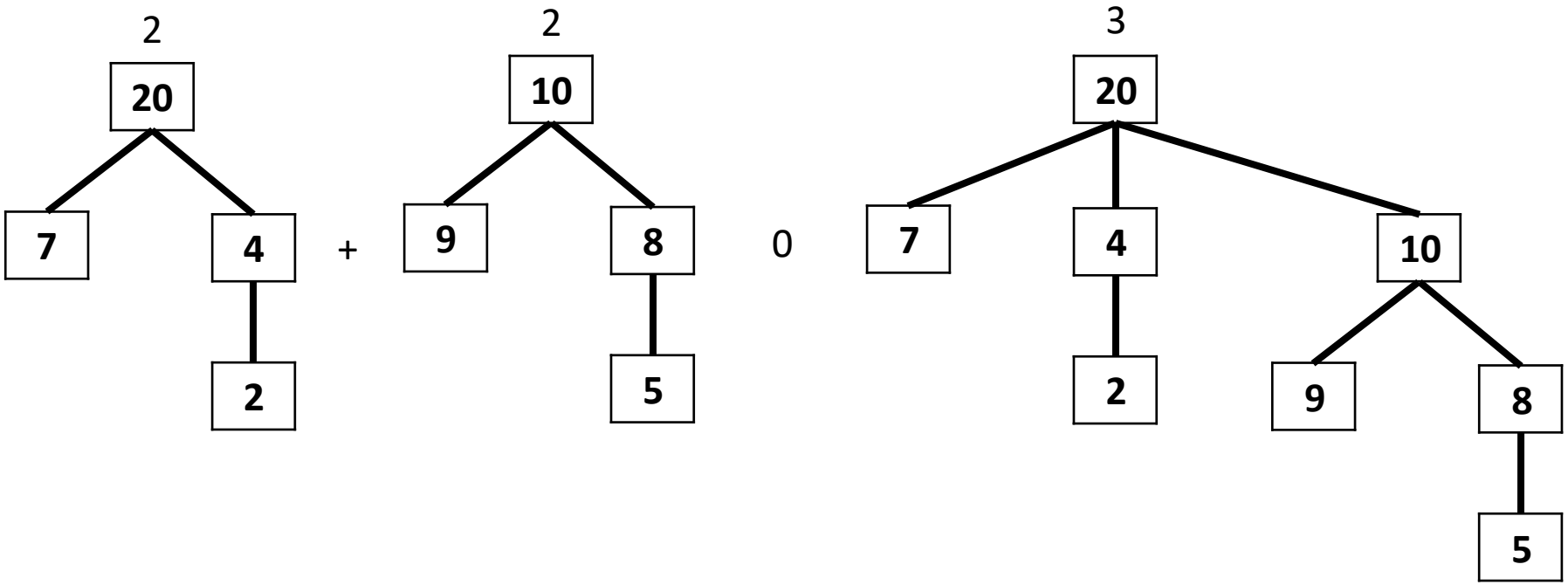


- Un arbre binomial d'ordre k contient 2^k éléments

Fusion de deux arbres binomiaux d'ordre k

- Prendre l'arbre avec la plus petite racine et l'ajouter comme sous-arbre de l'autre racine

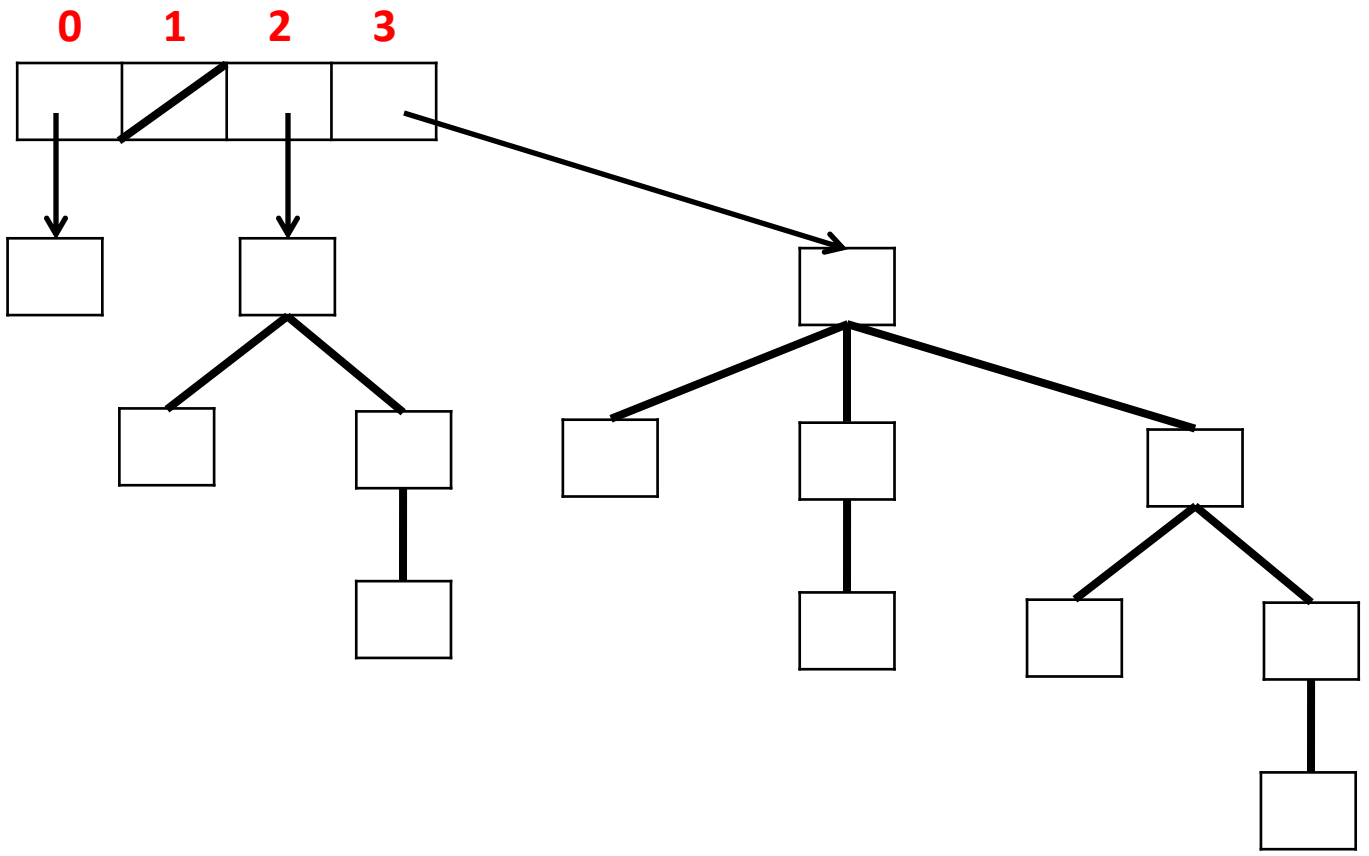
Exemple:



Monceau binomial

- Collection d'arbres binomiaux d'ordres différents

Exemple: 1 + 4 + 8 = 13 éléments ($2^0 + 2^2 + 2^3$)



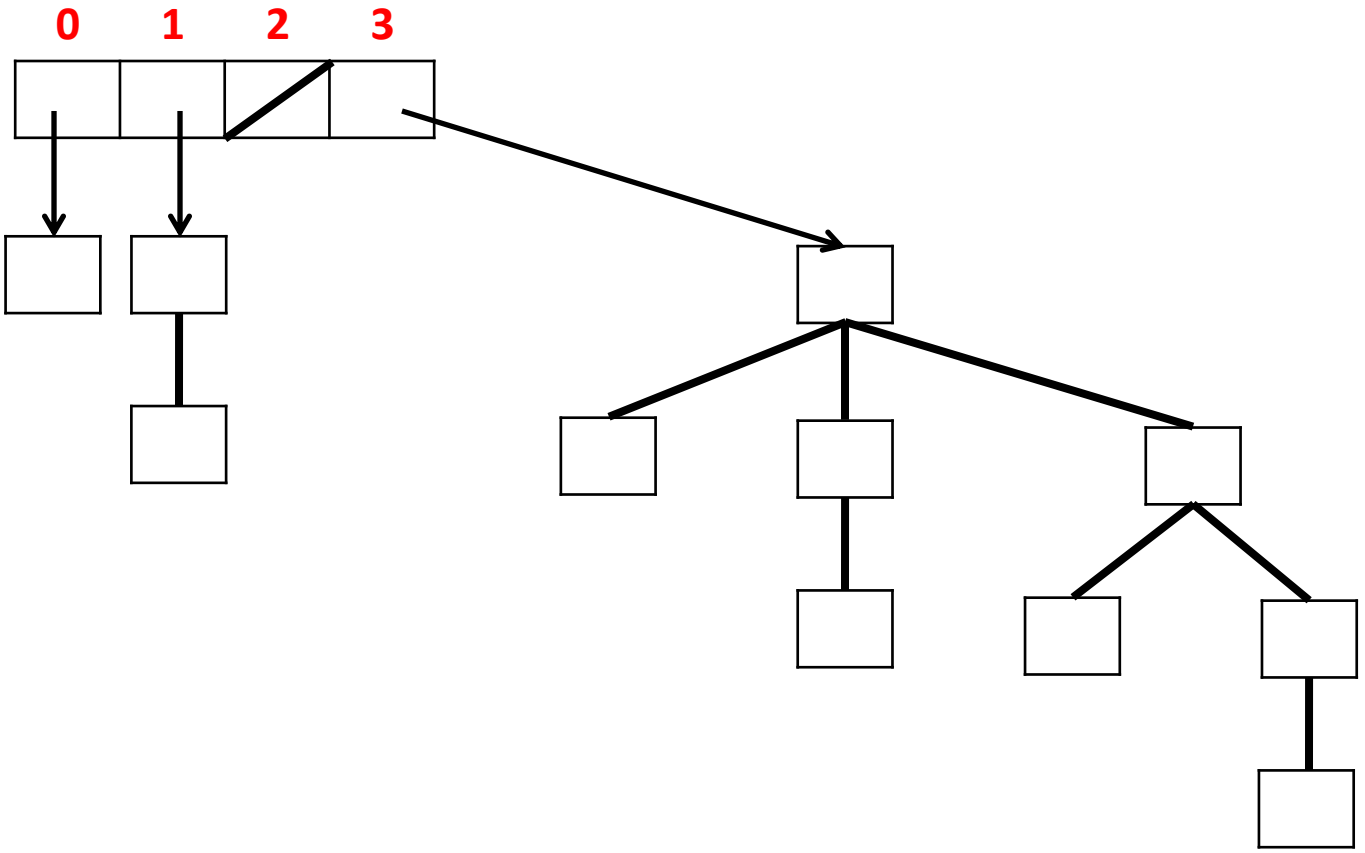
Monceau binomial

- Quel monceau binomial contient 11 éléments ?

Monceau binomial

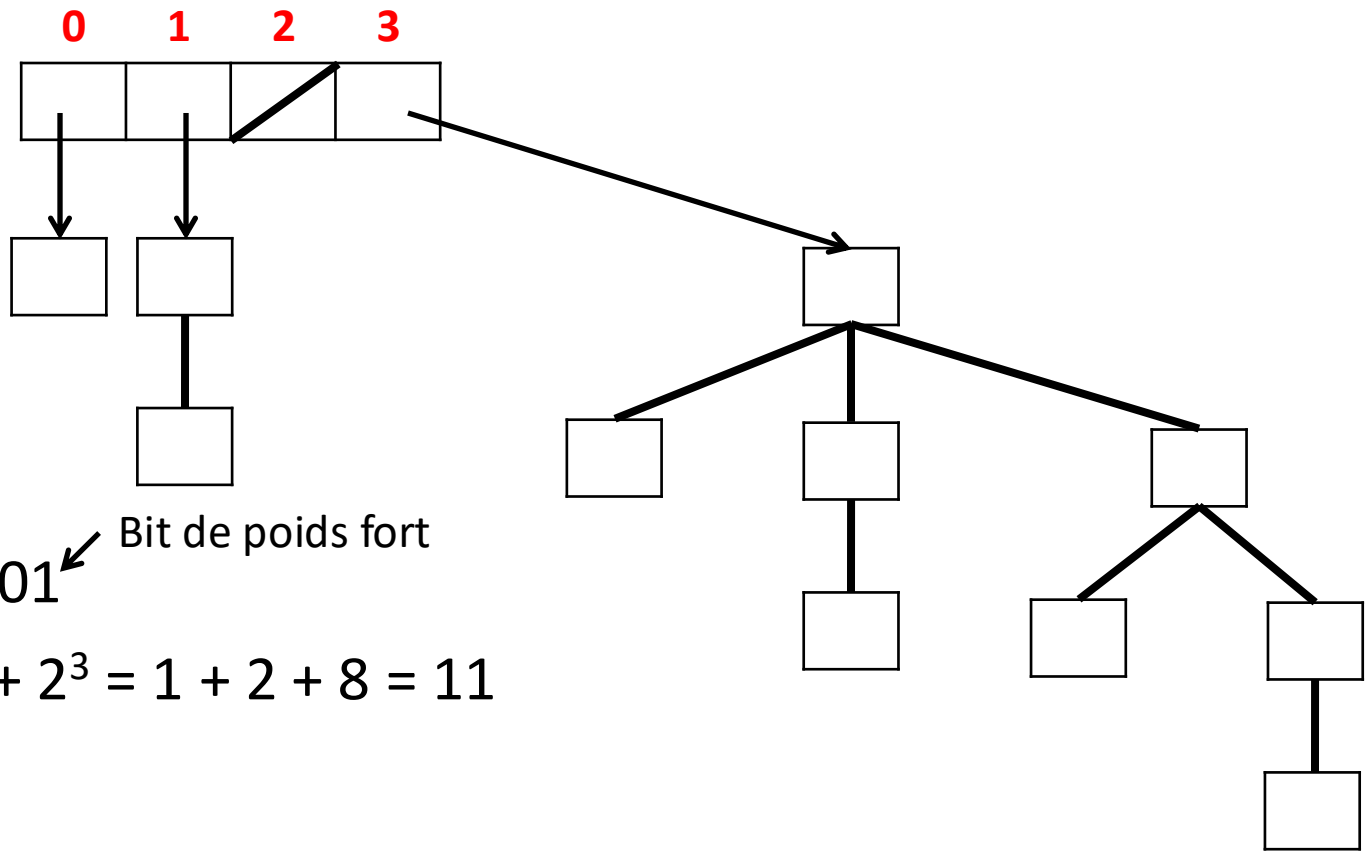
- Quel monceau binomial contient 11 éléments ?

  $2^0 + 2^1 + 2^3 = 1 + 2 + 8 = 11$



Monceau binomial

- Quel monceau binomial contient n éléments ?
  Écrire n en base binaire, et garder les positions (ordres) où Il y a 1.



$11 = 1101$

$2^0 + 2^1 + 2^3 = 1 + 2 + 8 = 11$

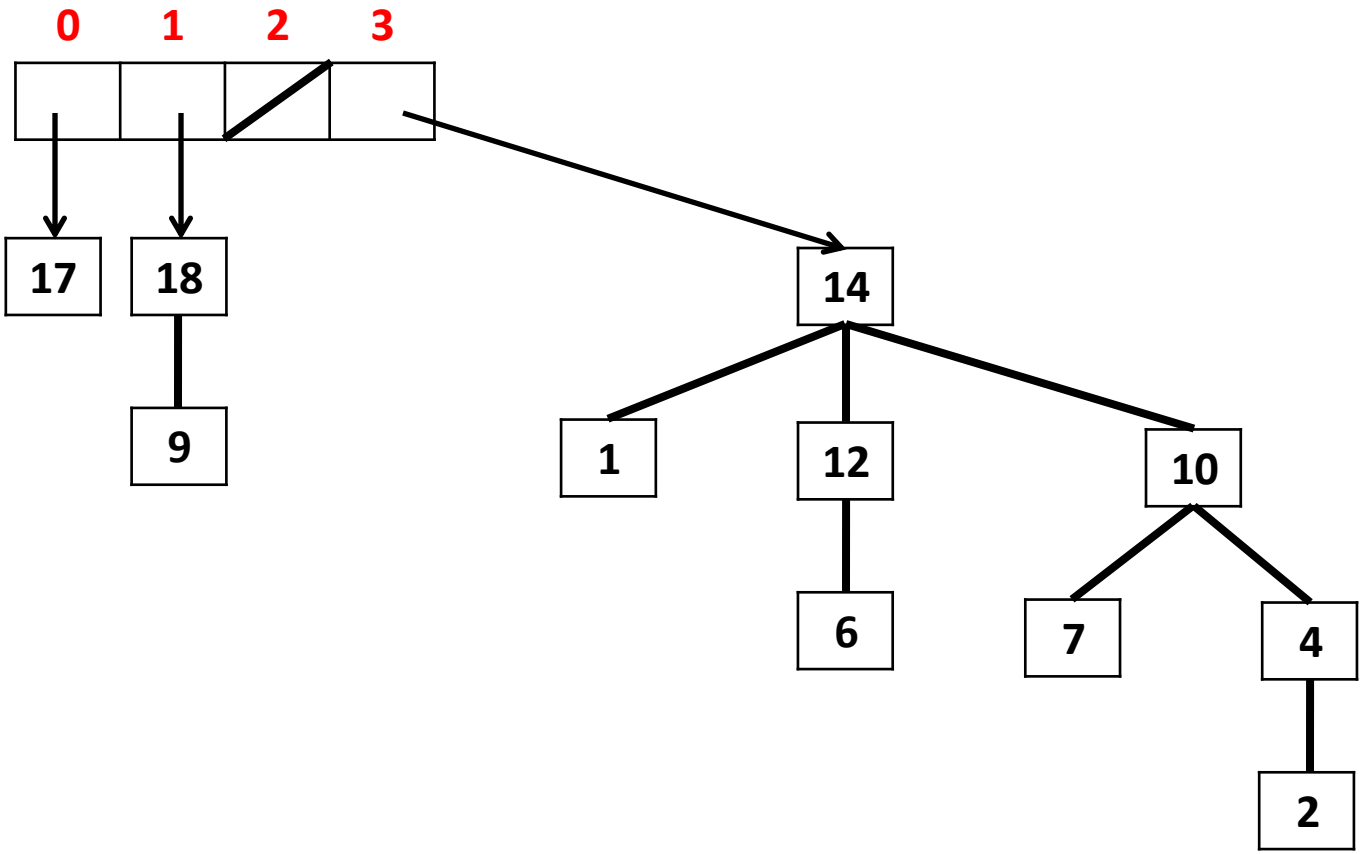
Insérer x dans un monceau binomial

- Équivalent à l'incrémentation dans un nombre binaire

- Créer un arbre d'ordre 0 contenant x
- Si il n'y a pas d'arbre d'ordre 0 dans le monceau binomial, ajouter l'arbre
- Sinon: fusionner les deux arbres d'ordre 0 pour obtenir un arbre d'ordre 1
- Recommencer jusqu'à ce qu'on puisse insérer l'arbre

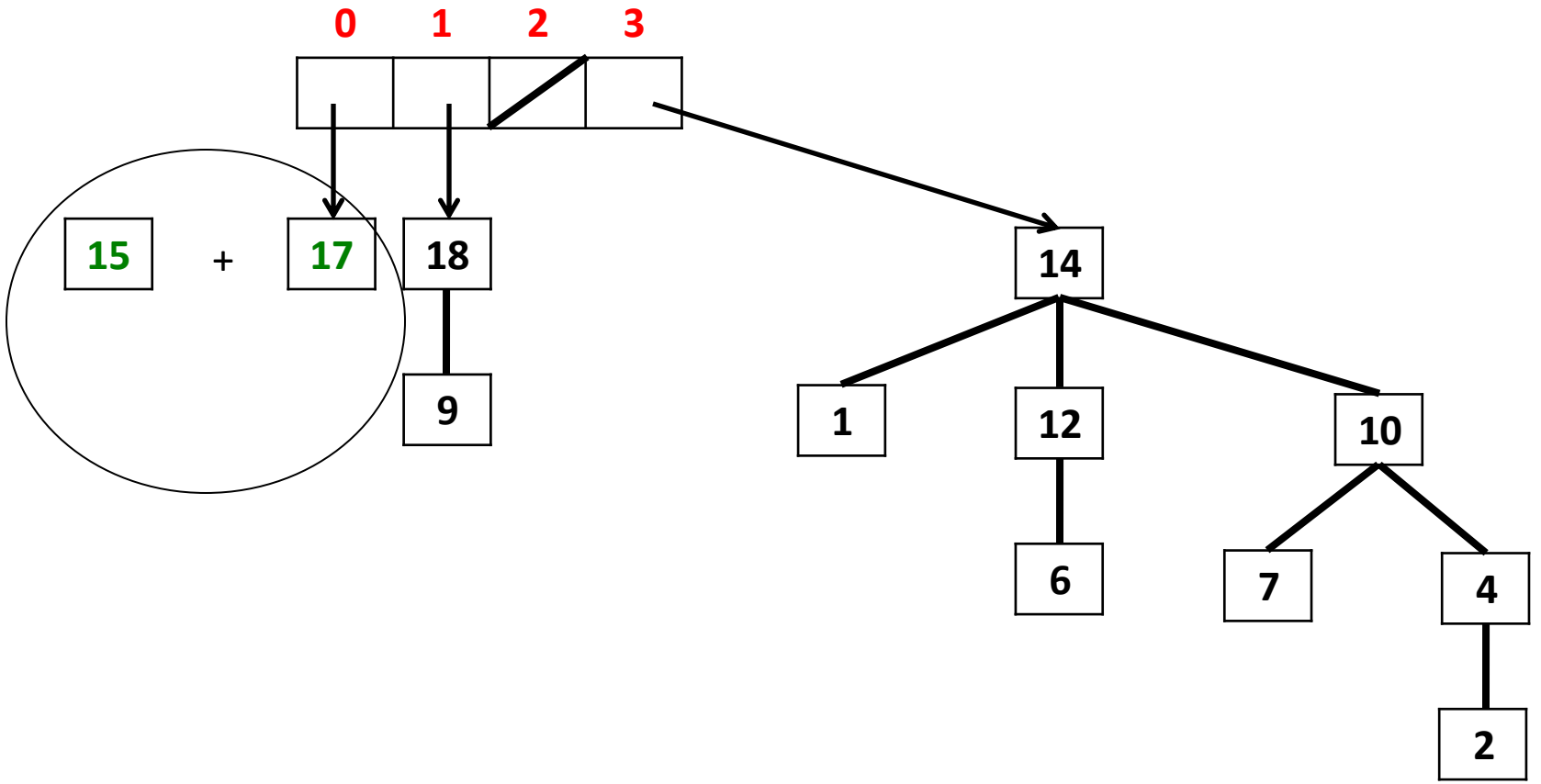
Complexité en temps O(log(n))

Insérer 15 dans le monceau binomial



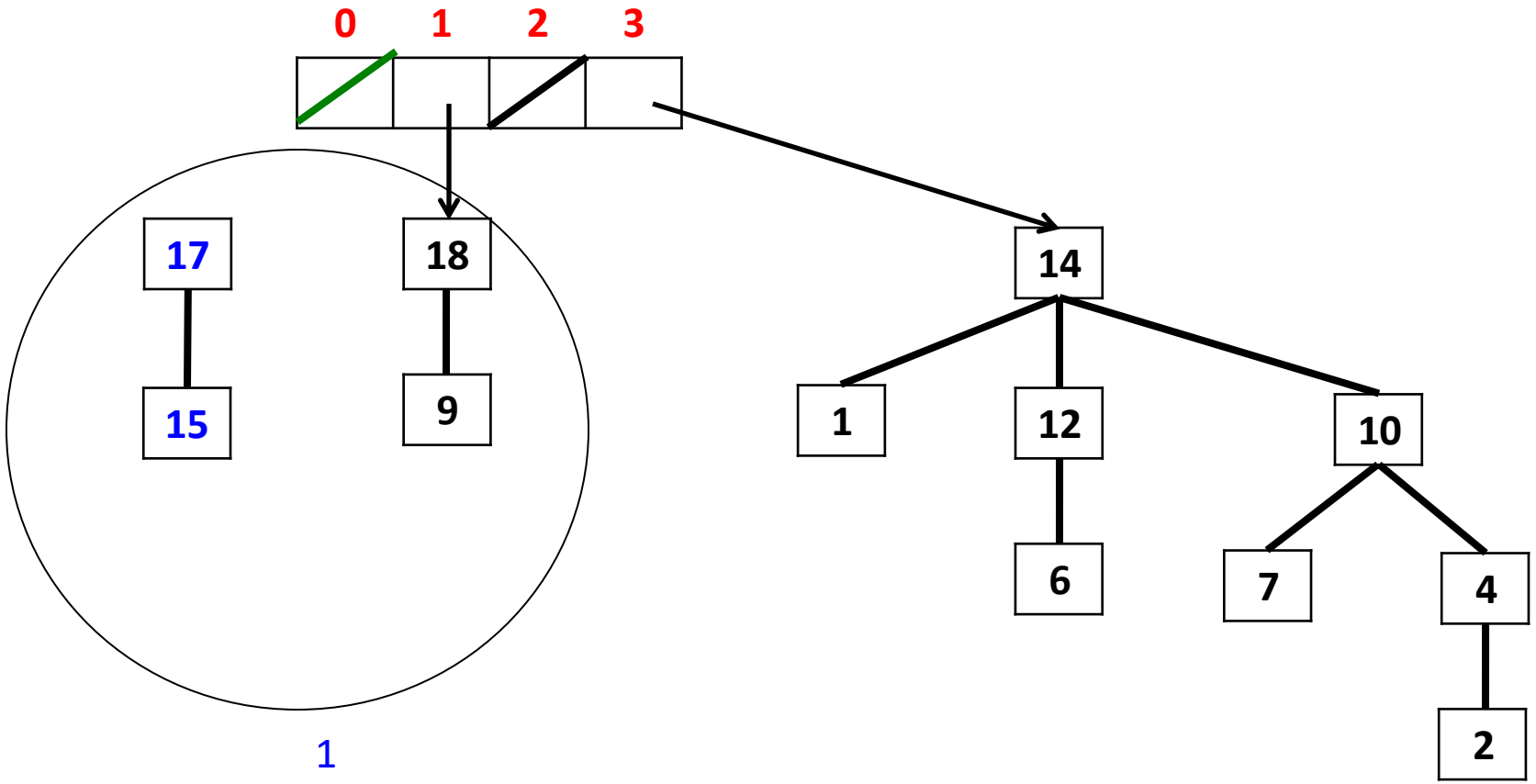
$$\begin{array}{r} 1\ 1\ 0\ 1 \\ +\ 1\ 0\ 0\ 0 \\ \hline \end{array}$$

Insérer 15 dans le monceau binomial



$$\begin{array}{r} 1\ 1\ 0\ 1 \\ +\ 1\ 0\ 0\ 0 \\ \hline \end{array}$$

Insérer 15 dans le monceau binomial



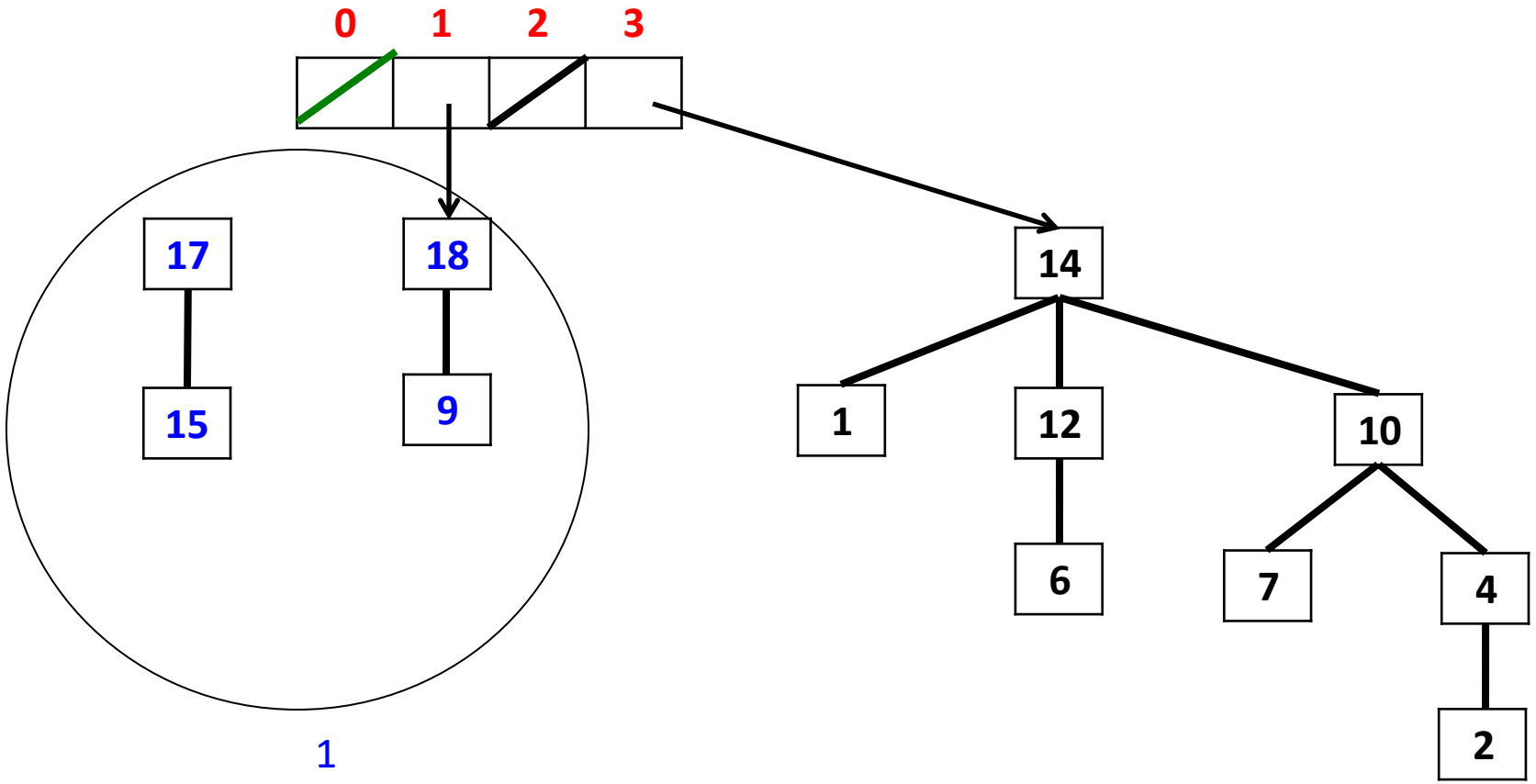
1

1 1 0 1

\+ 1 0 0 0

0

Insérer 15 dans le monceau binomial



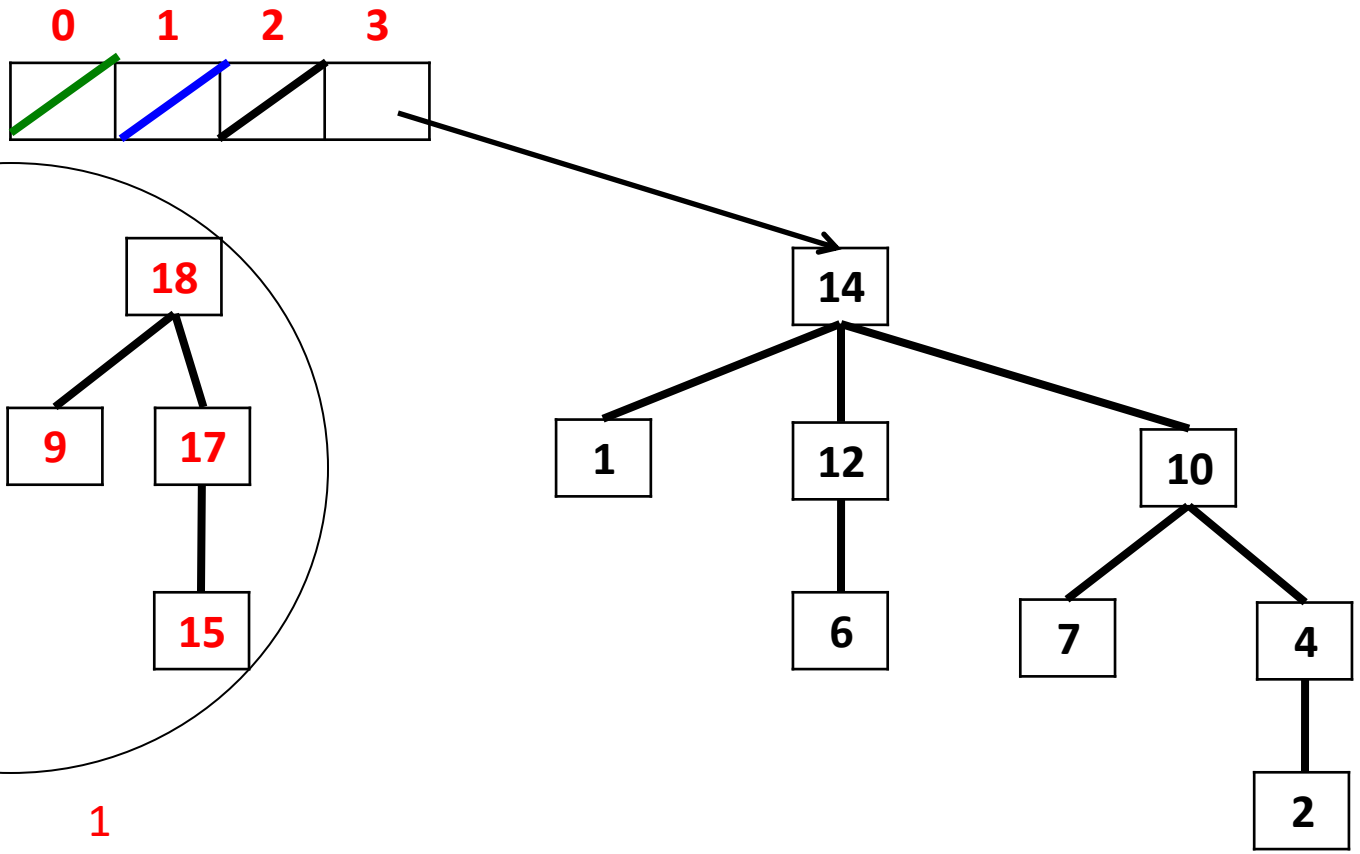
1

1 1 0 1

\+ 1 0 0 0

0

Insérer 15 dans le monceau binomial



1

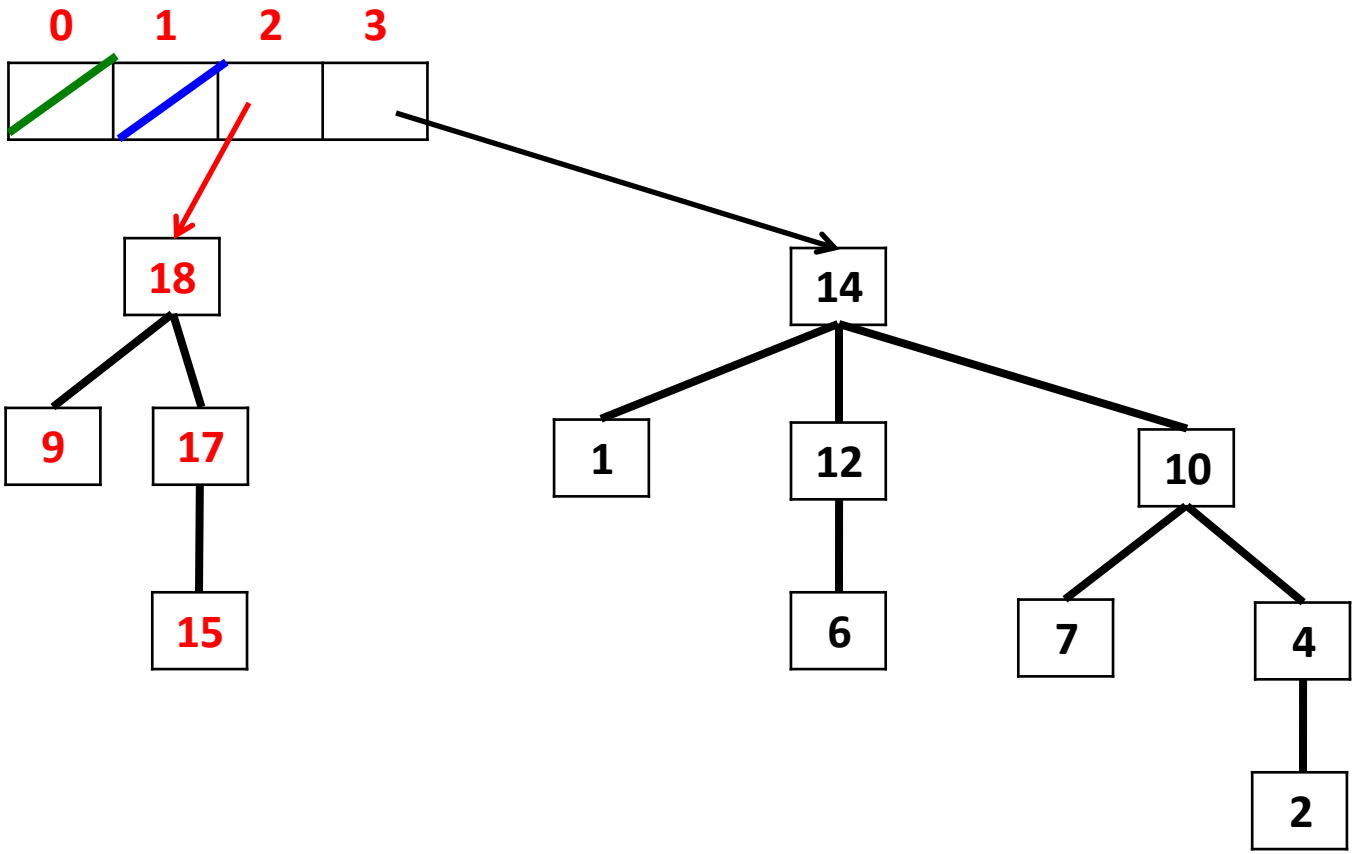
1 1 0 1

\+ 1 0 0 0

---

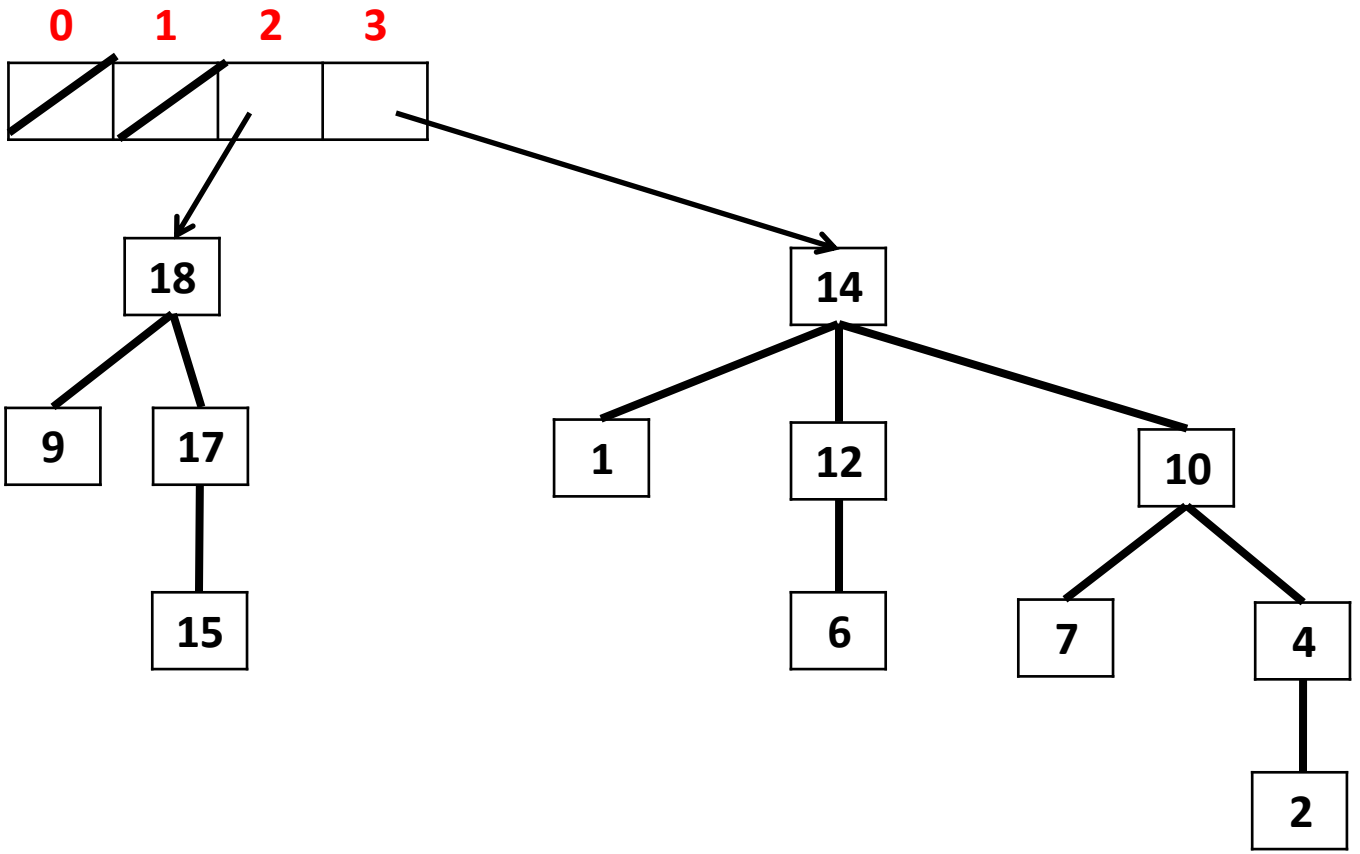
0 0

Insérer 15 dans le monceau binomial



$$
\begin{array}{rcccc}
 & 1 & 1 & 0 & 1 \\
+ & 1 & 0 & 0 & 0 \\
\hline
 & 0 & 0 & 1 & 1
\end{array}
$$

Insérer 15 dans le monceau binomial



Fusionner deux monceaux binomiaux B1 et B2

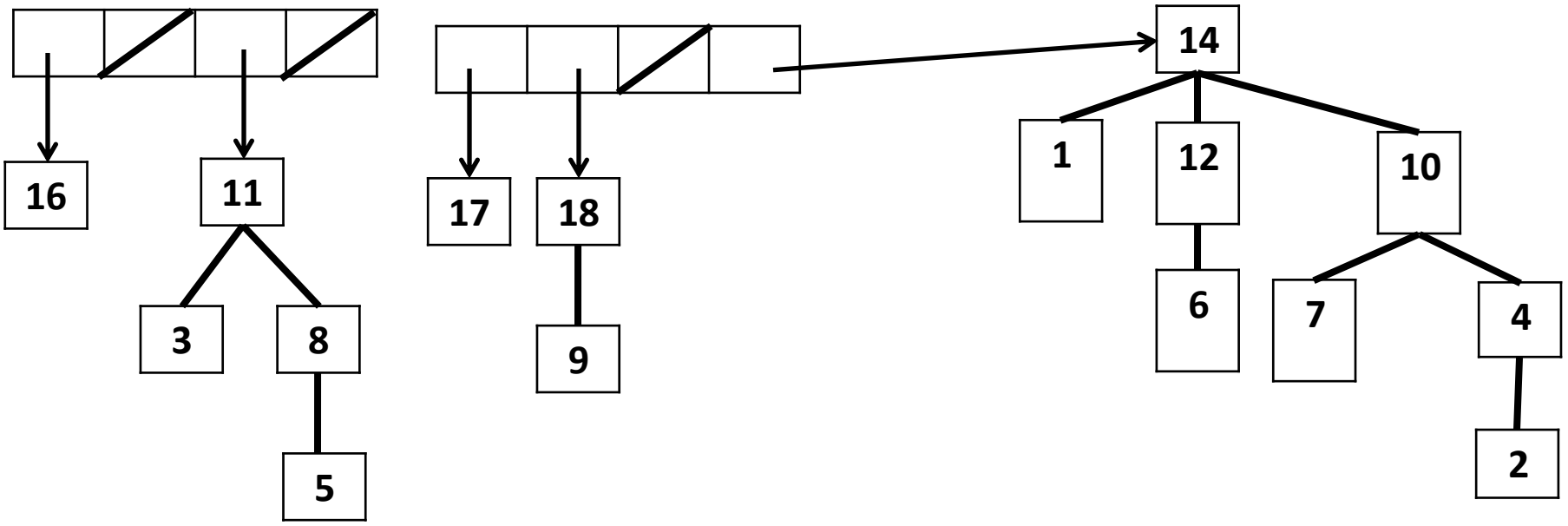
- Équivalent à l'addition en base binaire

- Retenue = nullptr
- Pour i allant de 0 à n
  - Si B1[i] != nullptr et B2[i] != nullptr:
    - B[i] = Retenue
    - Retenue = B1[i] + B2[i]
  - Sinon si Retenue != nullptr:
    - Si B1[i] != nullptr:
      - B[i] = nullptr
      - Retenue = Retenue + B1[i]
    - Sinon :
      - B[i] = nullptr
      - Retenue = Retenue + B2[i]
  - Sinon:
    - Si B1[i] != nullptr:
      - B[i] = B1[i]
    - Sinon :
      - B[i] = B2[i]
- Retourner: B

Complexité en temps O(log(n))

Fusionner deux monceaux binomiaux

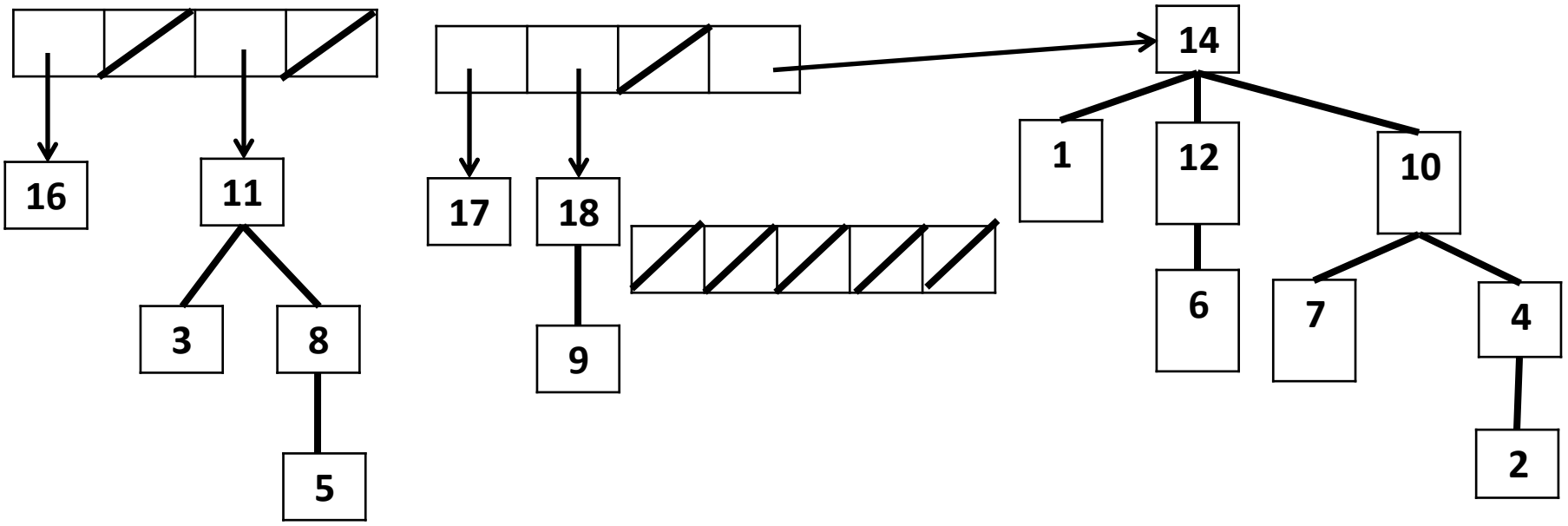
- Équivalent à l'addition en base binaire



$$\begin{array}{r} 1\ 1\ 0\ 1 \\ +\ 1\ 0\ 1\ 0 \\ \hline \end{array}$$

Fusionner deux monceaux binomiaux

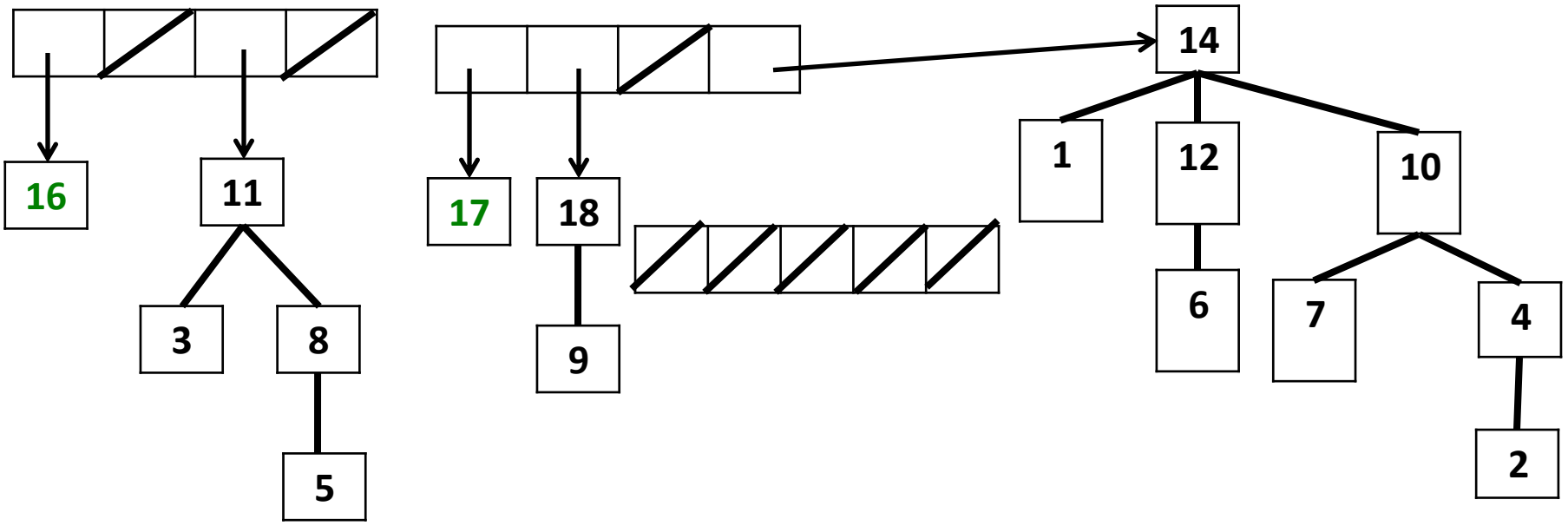
- Équivalent à l'addition en base binaire



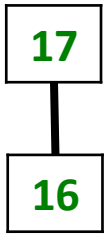

```
  1 1 0 1
+ 1 0 1 0
---------
  0 0 0 0 1
```

Fusionner deux monceaux binomiaux

- Équivalent à l'addition en base binaire

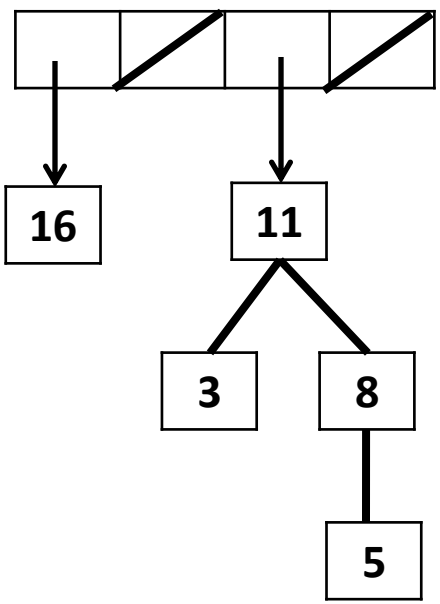


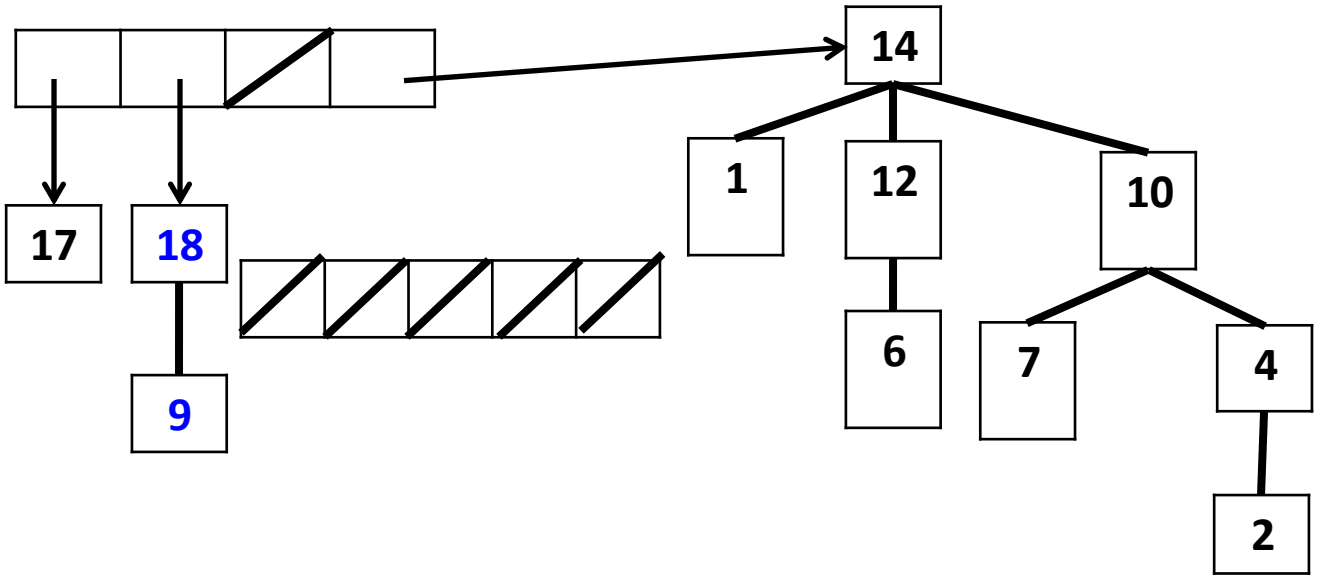

```
  1  1  0 1
+ 1  0  1 0
-----------
  0  0  0 0 1
```



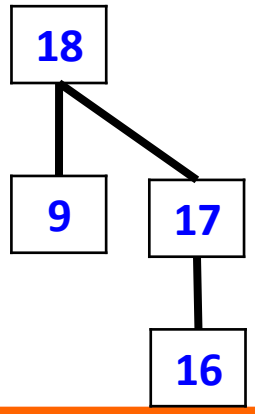
Fusionner deux monceaux binomiaux

- Équivalent à l'addition en base binaire



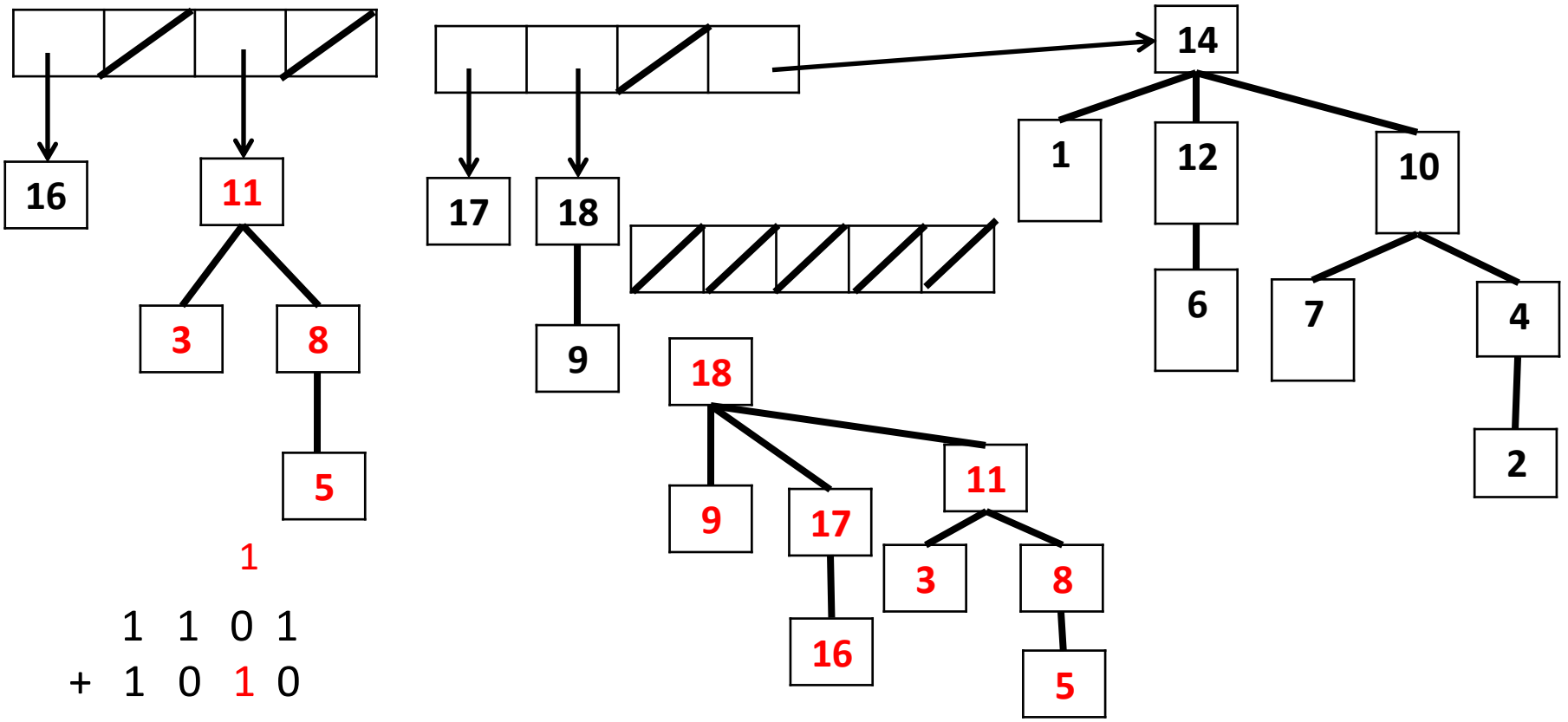



```
    1
  1 1 0 1
+ 1 0 1 0
---------
  0 0 0 0 1
```



Fusionner deux monceaux binomiaux

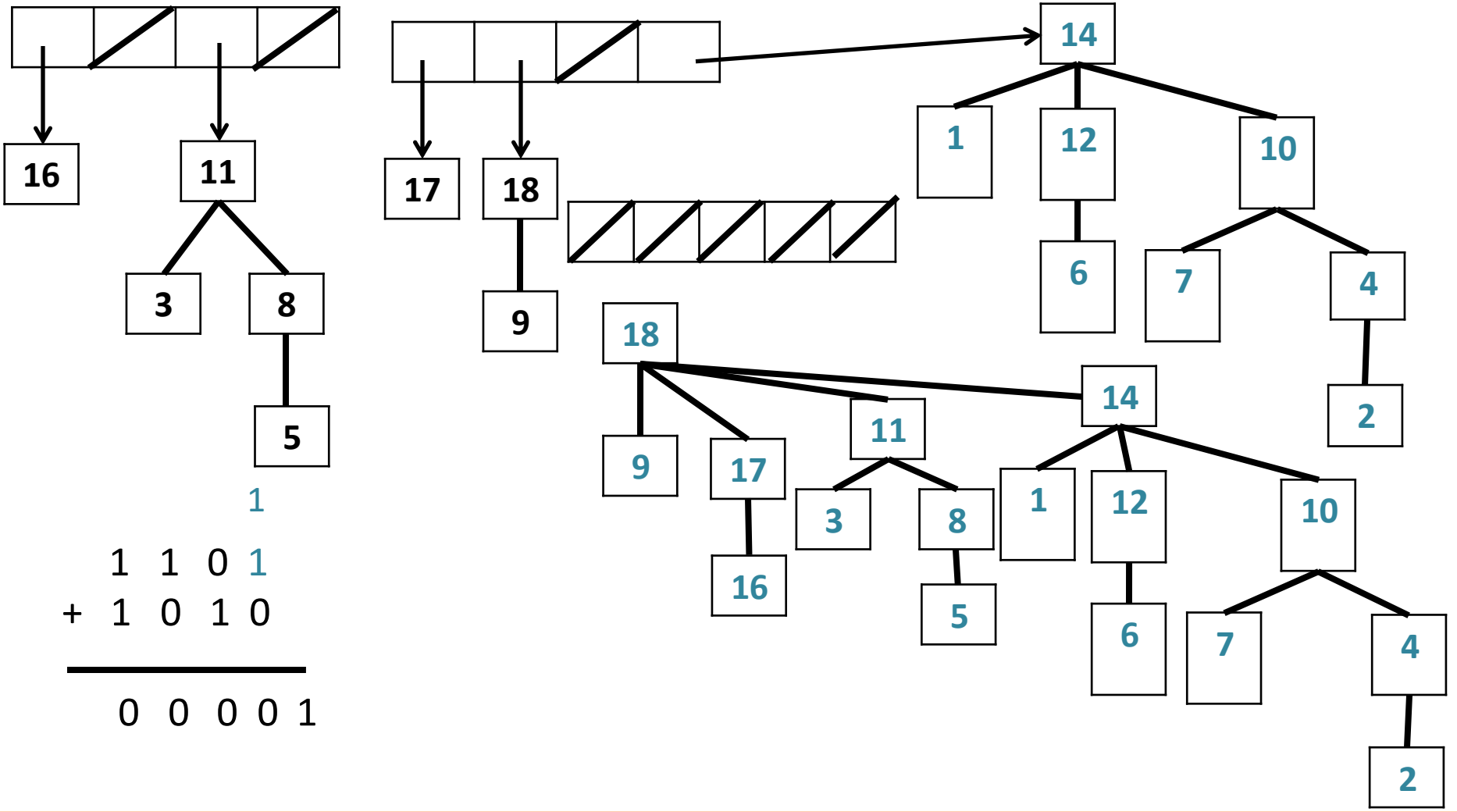
- Équivalent à l'addition en base binaire




```
    1
  1 1 0 1
+ 1 0 1 0
---------
  0 0 0 0 1
```

Fusionner deux monceaux binomiaux

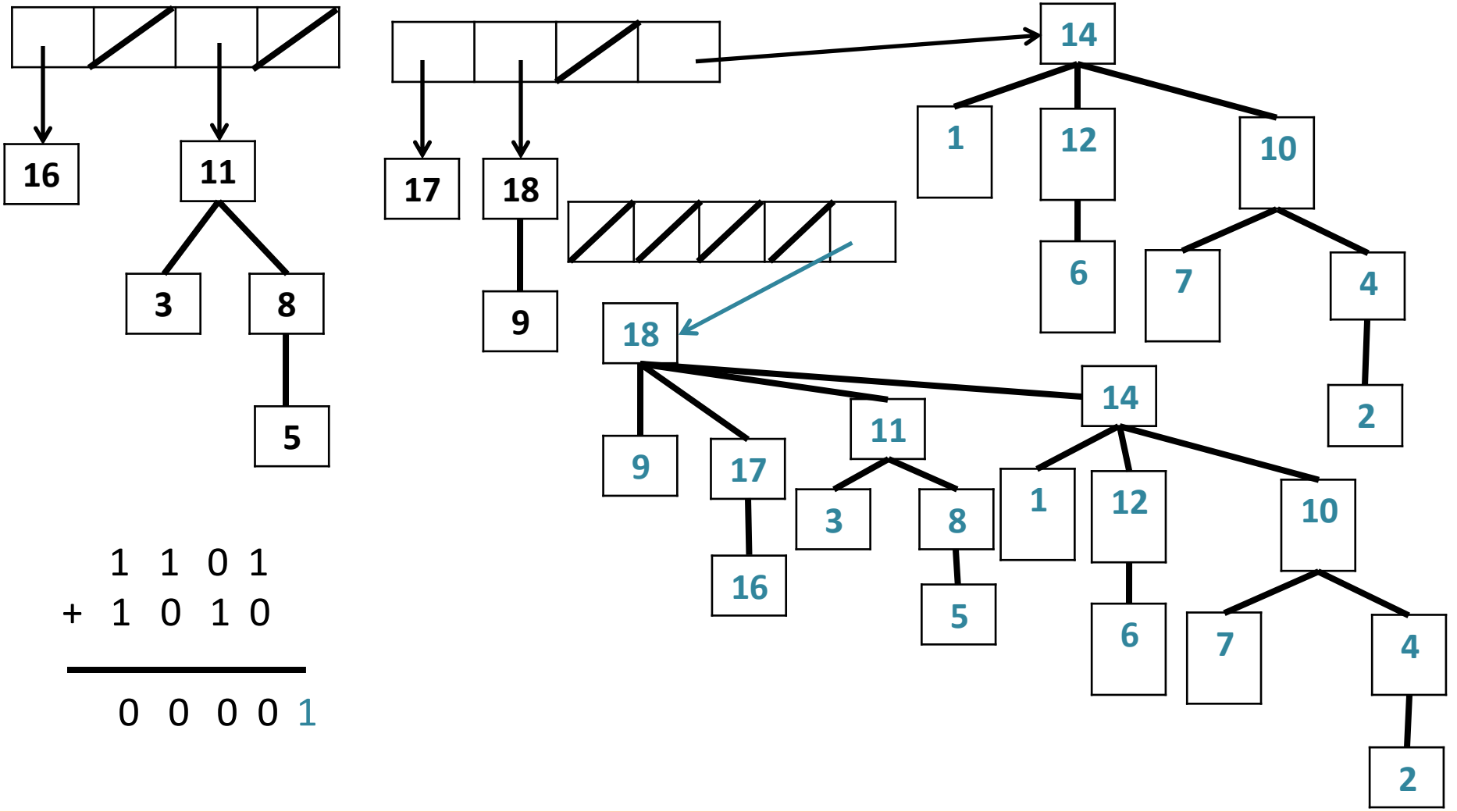
- Équivalent à l'addition en base binaire




```
    1
  1 1 0 1
+ 1 0 1 0
---------
  0 0 0 0 1
```

Fusionner deux monceaux binomiaux

- Équivalent à l'addition en base binaire




```
  1 1 0 1
+ 1 0 1 0
---------
  0 0 0 0 1
```

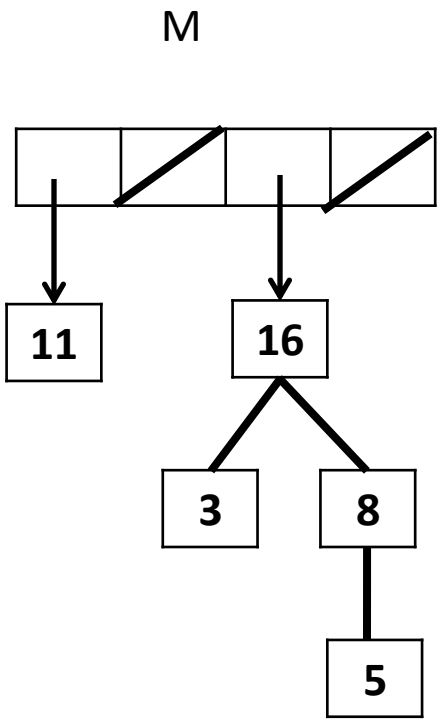
Suppression (plus grand) dans un monceau binomial M

- revient à l'addition en base binaire

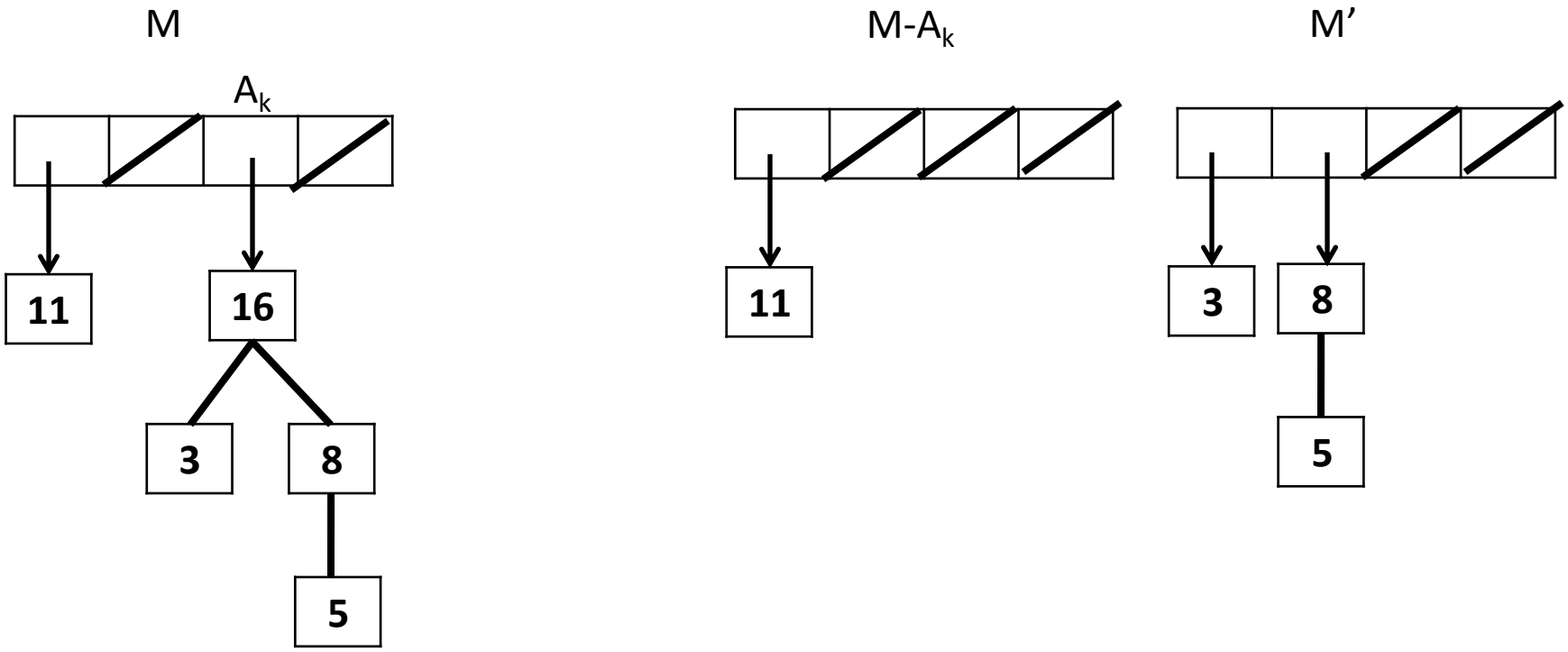
- k = l'ordre de l'arbre A_k qui a la plus grande valeur à sa racine
- M' = monceau obtenu en supprimant la racine de l'arbre d'ordre k (M' contient donc des sous-arbres d'ordres 0…. k-1)
- Fusionner M-A_k et M'

Complexité en temps O(log(n))

Suppression du plus grand



Suppression du plus grand



Suppression du plus grand