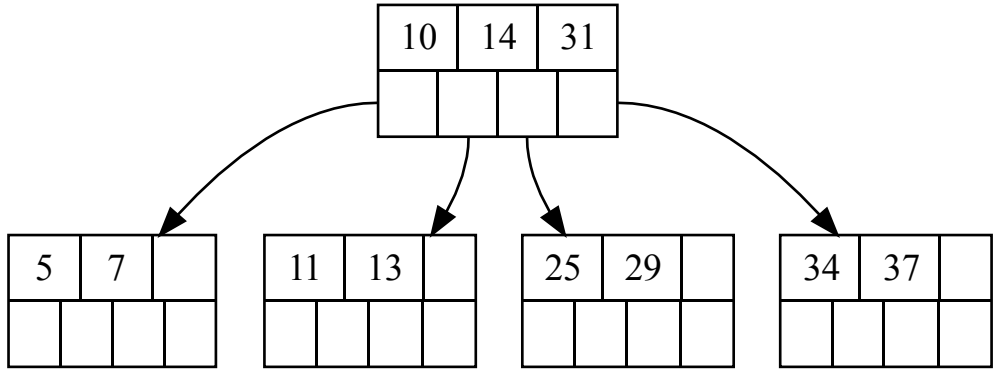
3.4 – Les B-arbres

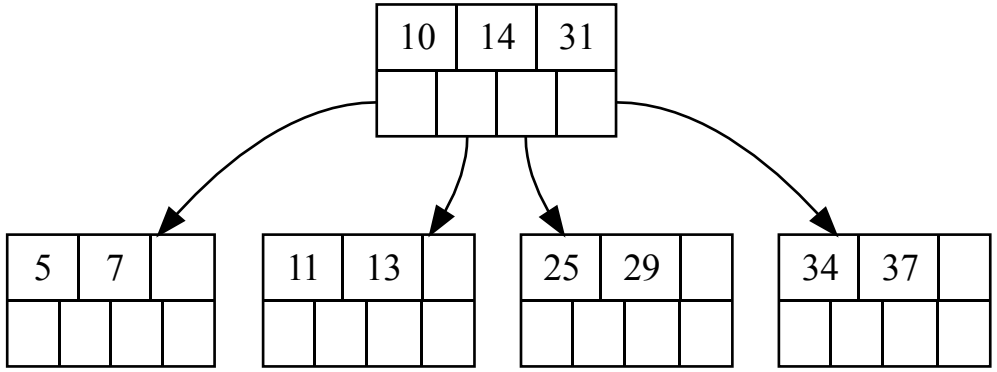
Définition

- Inventés par R. Bayer et E. M. McCreight en 1970
  - « *The more you think about what the B in B-trees means, the better you understand B-trees.* » [McCreight, 2013]
- Un B-arbre d'ordre m est un arbre m-aire auto-équilibré
  - Un B-arbre est ainsi une généralisation des ABR auto-équilibrés (comme les AVL ou les arbres rouge-noir)
- Chaque nœud possède k<=m-1 clés
- Exemple pour m=4 :
  - Les valeurs inférieures à a_1 sont dans le sous-arbre 1
  - Les valeurs entre a_1 et a_2 sont dans le sous-arbre 2
  - Les valeurs entre a_2 et a_3 sont dans le sous-arbre 3
  - Les valeurs supérieures à a_3 sont dans le sous-arbre 4



Propriétés

- Un B-arbre d'ordre m vérifie les propriétés suivantes :
  - Chaque nœud a au plus m fils
  - Chaque nœud interne (= tous les nœuds sauf la racine et les feuilles) ont au moins ceil(m/2) fils
  - Chaque nœud non-feuille a au moins 2 fils
  - Toutes les feuilles sont au même niveau
  - Tout nœud non-feuille avec k fils a k-1 clés



Complexité

- La recherche, l'insertion et la suppression dans un B-arbre sont en O(log(n)) dans le pire des cas

Exemple d’applications

- Manipulation d’un large index de fichiers, qui n’entre pas entièrement en mémoire
  - Exemple : NTFS (Windows, 1993), EXT4 (Linux, 2006), APFS (Apple, 2017)
- Optimisation de la recherche de données, sachant que le temps de lecture d’un enregistrement depuis un disque dur est beaucoup plus élevé que le temps de comparaison

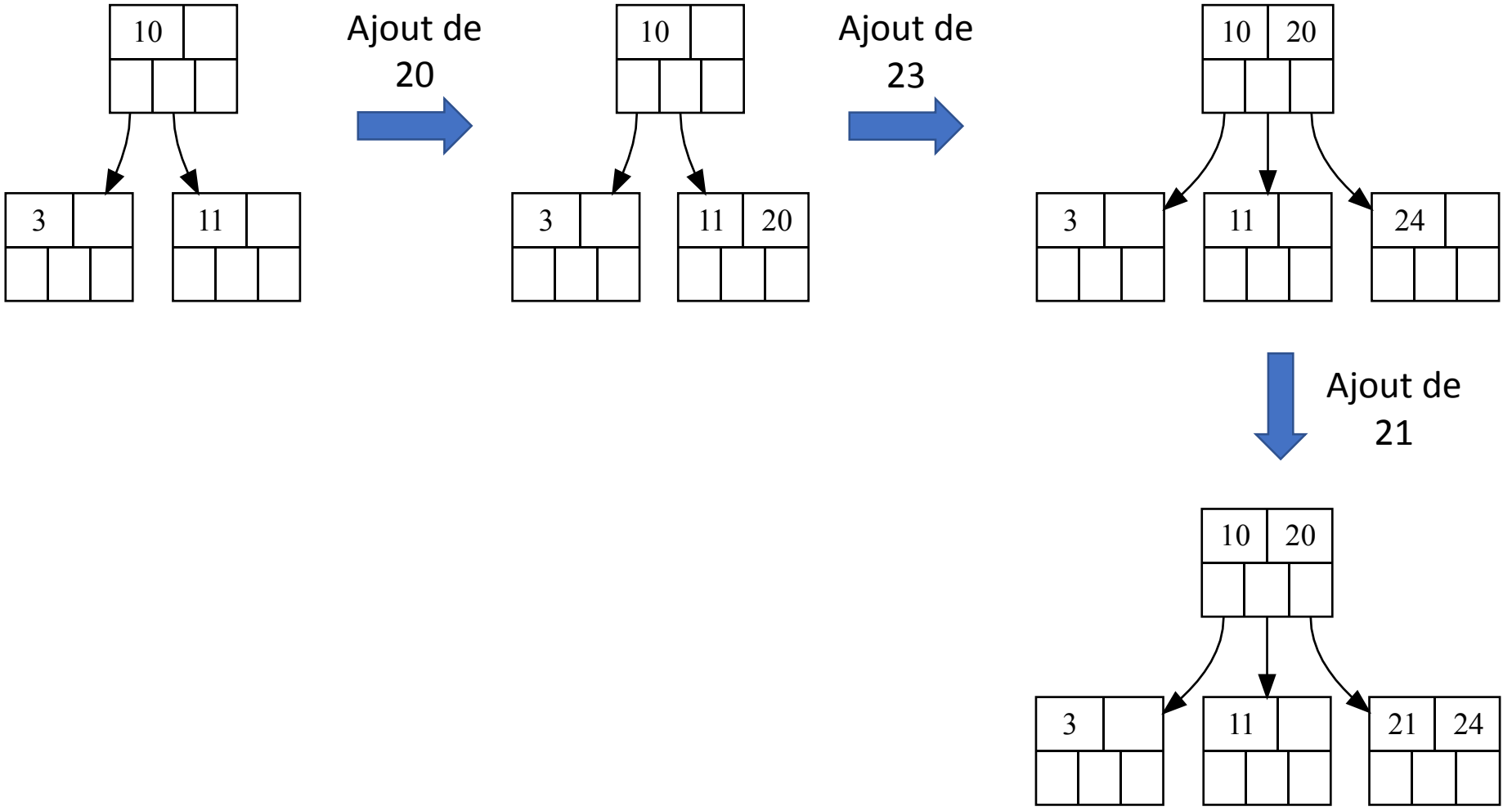
Recherche dans un B-arbre

- La recherche est similaire à celle dans un ABR
  - A l'intérieur de chaque nœud, on fait généralement une recherche dichotomique sur les clés

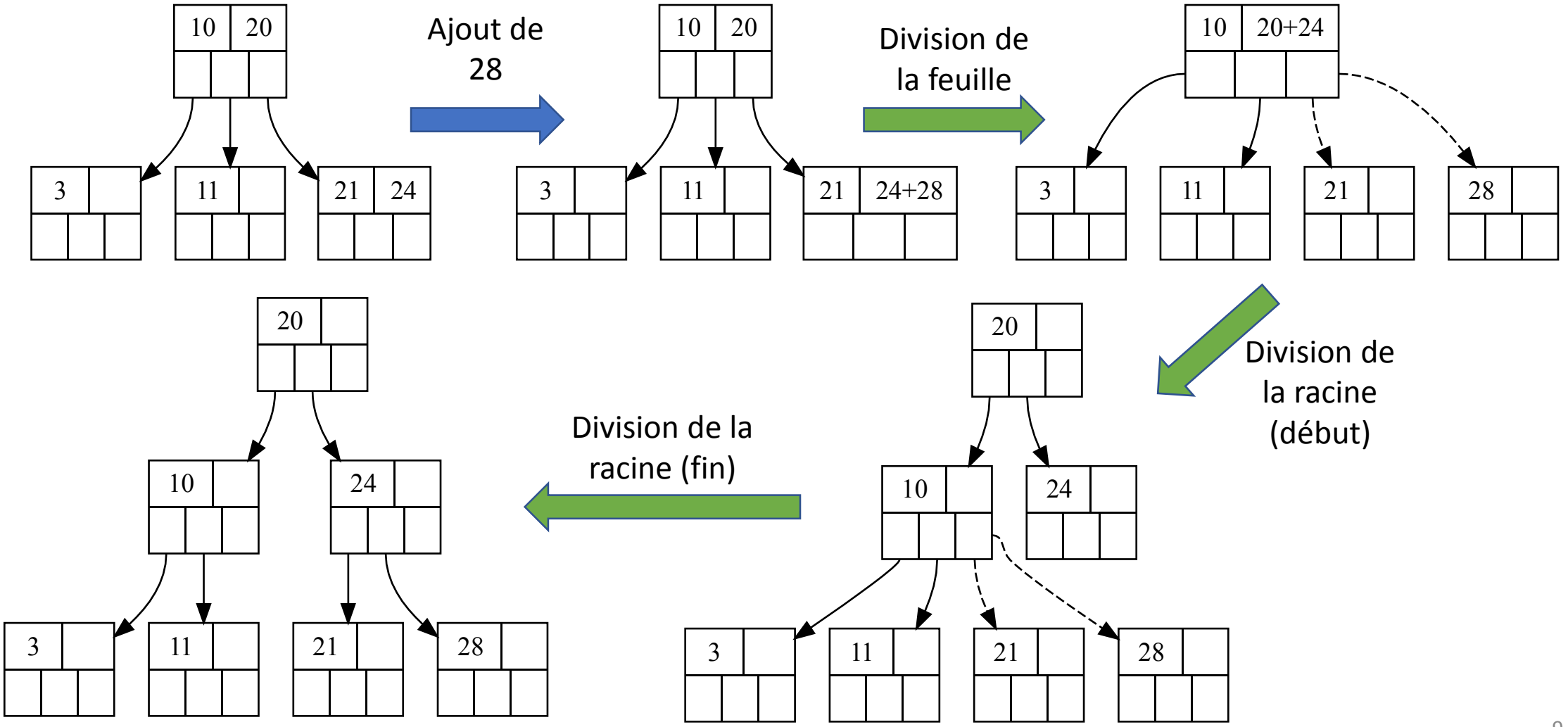
Insertion dans un B-arbre

- Chaque insertion démarre sur une feuille
  - Si la feuille n’est pas pleine, on lui ajoute la clé
  - Si la feuille est pleine (donc ne pouvant pas contenir la (m+1)-ème clé), on divise la feuille en deux frères ayant chacun m/2 clés, et on insère la clé intermédiaire dans le parent (en continuant récursivement si besoin)
- Remarque : la profondeur d’un B-arbre n’augmente que quand on doit diviser la racine

Exemple d’insertions (1/2)



Exemple d'insertions (2/2)



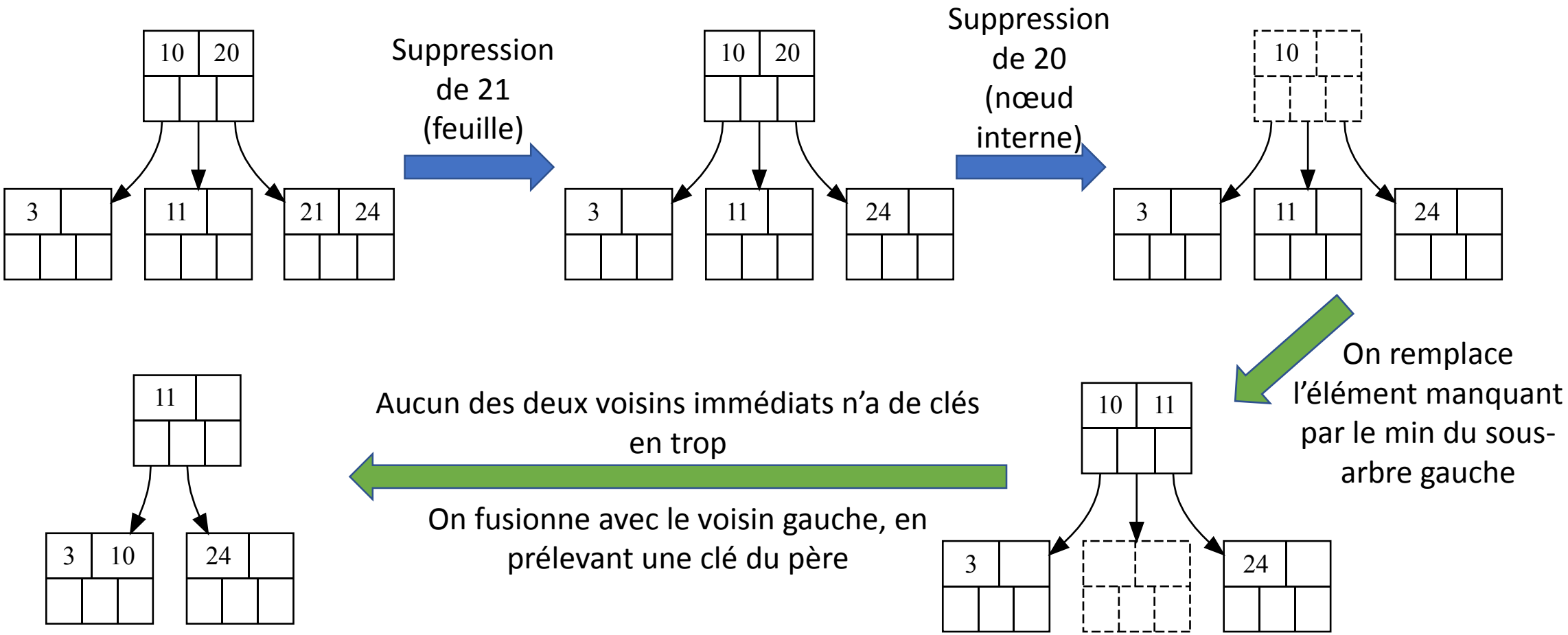
Suppression dans un B-arbre

- On cherche le nœud contenant la clé à supprimer
- Si c'est une feuille :
  - On supprime la clé
  - S'il n'y a plus assez de clés dans la feuille, on doit rééquilibrer l'arbre (voir après)
- Si c'est un nœud interne :
  - On remplace la clé à supprimer par le plus grand élément du sous-arbre gauche (ou par le plus petit élément du sous-arbre droit), qui est forcément dans une feuille ; il faudra potentiellement rééquilibrer l'arbre (voir après)

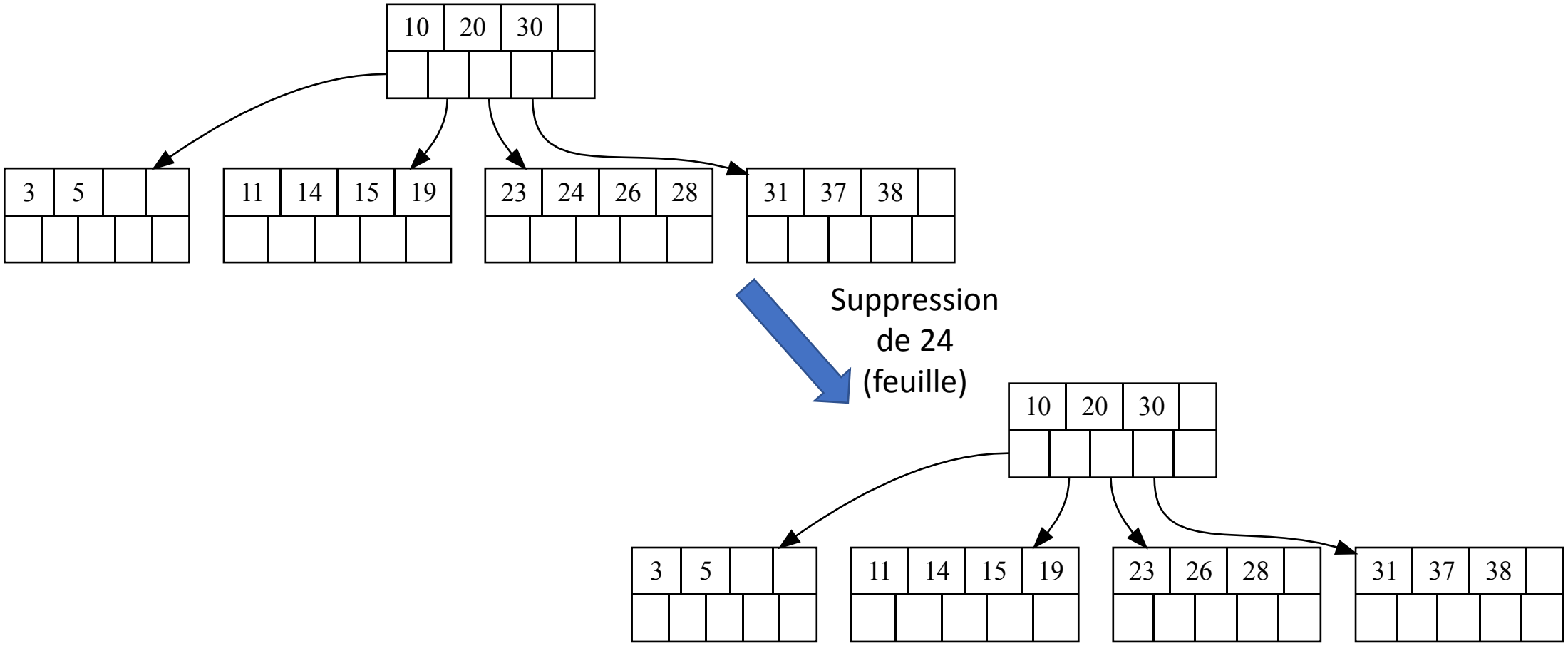
Rééquilibrage après suppression

- Le rééquilibrage part d'une feuille et remonte les ancêtres
- Pour rééquilibrer, on doit redistribuer des éléments de nœuds frères ayant assez de clés ➔ rotation (gauche ou droite)
- Si ce n'est pas possible ➔ fusion du nœud avec un frère, en enlevant un fils du père, mais il faut alors potentiellement rééquilibrer le père

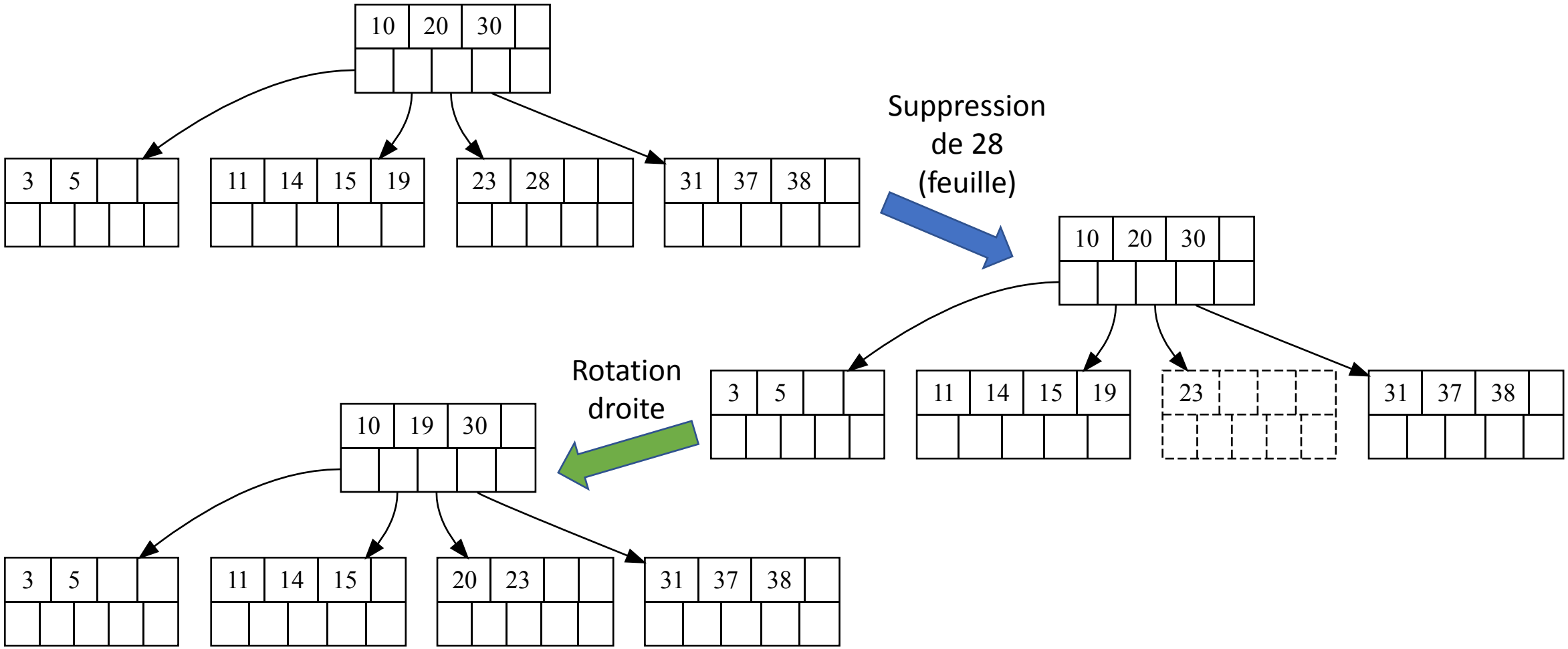
Exemple de suppressions (1/3)



Exemple de suppressions (2/3)



Exemple de suppressions (3/3)



Conclusion

- Les arbres sont des structures de données très utiles
- Les arbres binaires de recherche sont utilisés pour rechercher rapidement des données
- Les B-arbres sont utilisés dans les bases de données