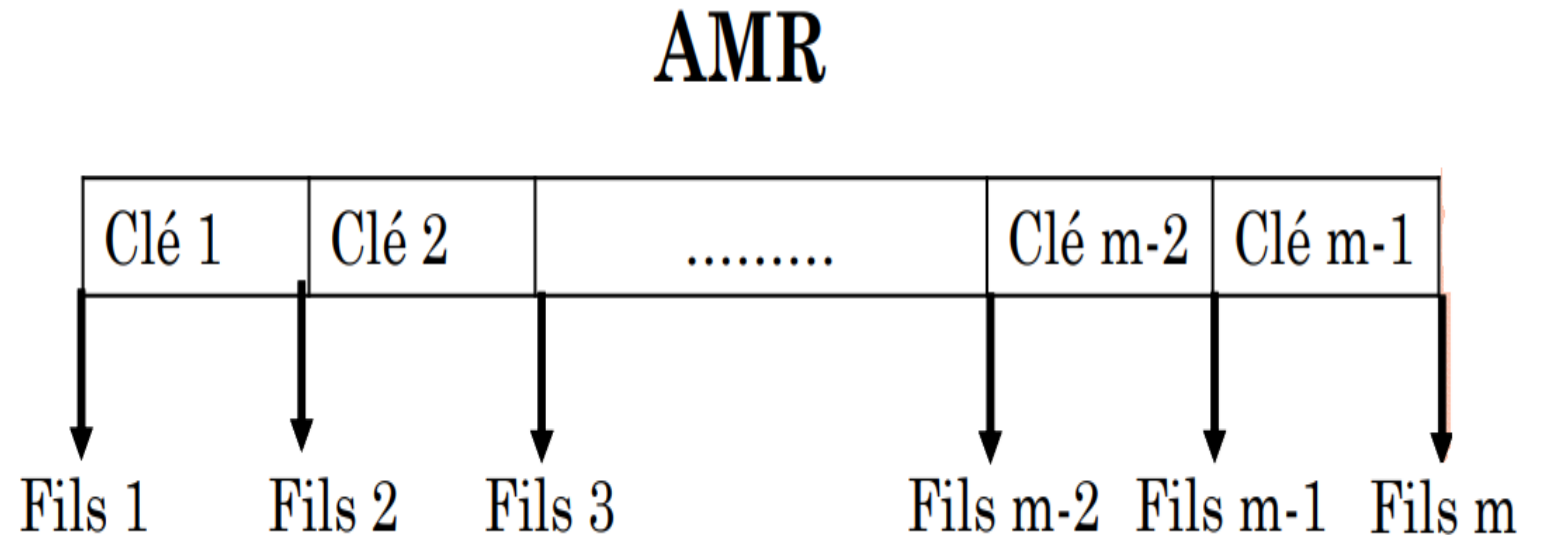
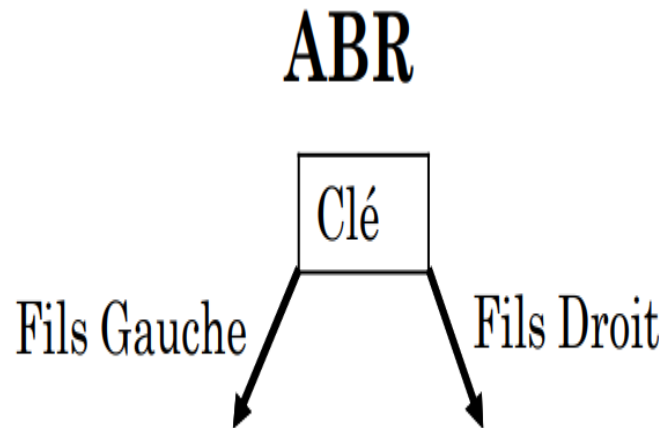


ARBRES M-AIRE DE RECHERCHE (AMR)

Master 1 CSIA

DEFINITION

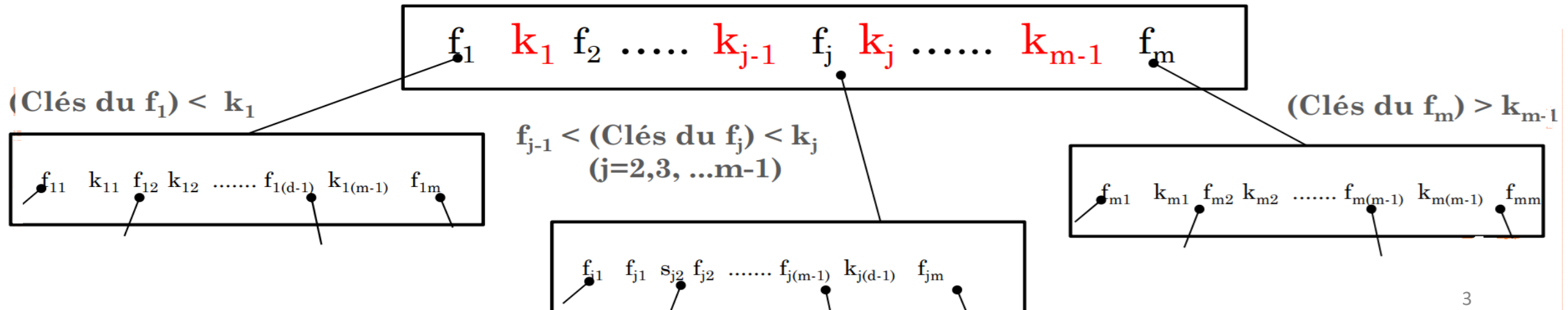
- Un Arbre M-aire de Recherche (AMR) peut être défini comme une généralisation de l'Arbre Binaire de Recherche (ABR):
- au lieu d'avoir une clé et deux fils, on aura $\ll m-1 \gg$ clés et $\ll m \gg$ fils pointeurs.



DEFINITION

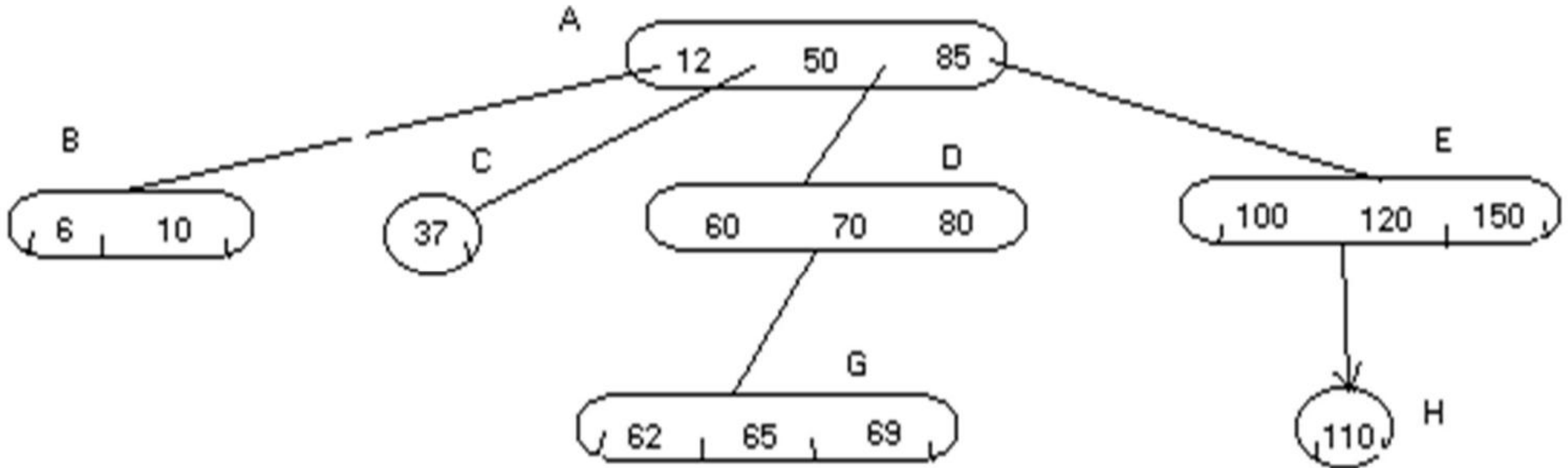
- Si f_1, f_2, \dots, f_m sont les $\ll m \gg$ fils issus d'un nœud donne avec les clés k_1, k_2, \dots, k_{m-1} dans l'ordre ascendant, alors :
- Toutes les clés dans f_1 sont inférieures a k_1
- Toutes les clés dans f_j ($j=2,3, \dots, m-1$) sont supérieures a k_{j-1} et inférieures a k_j
- Toutes les clés dans f_m sont supérieures k_{m-1} .

Ordre ascendant: $k_1 < k_2 < \dots < k_{m-1}$



DEFINITION

- **Exemple:** Soit l'AMR d'ordre 4 suivant



RECHERCHE

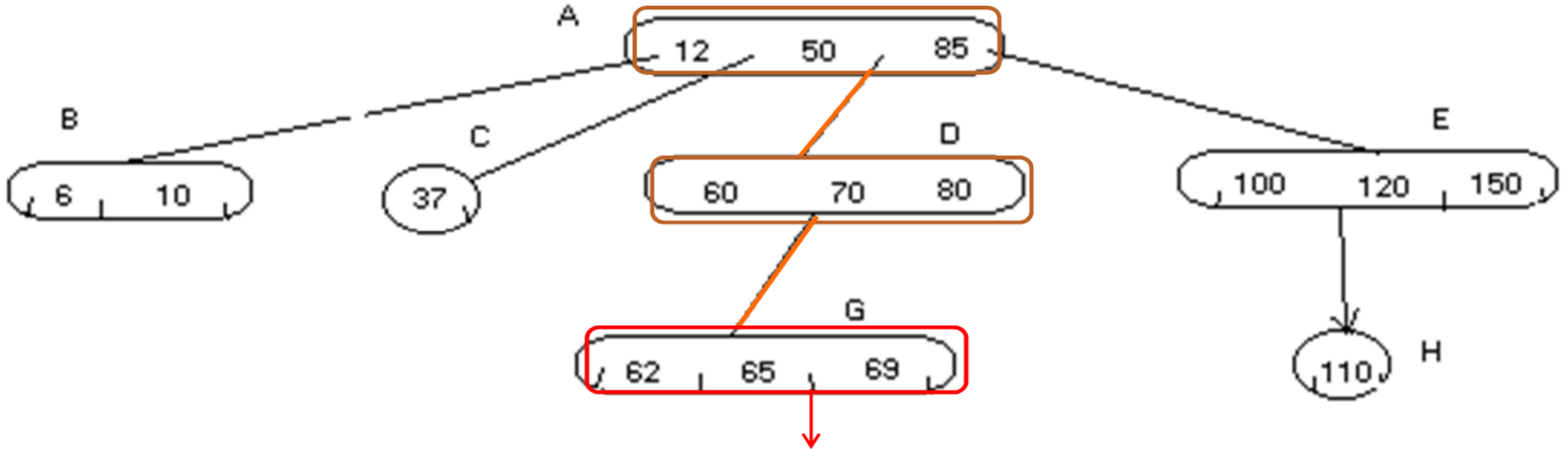
- La recherche dans un AMR ressemble beaucoup à celle effectuée dans un ABR, excepte qu'au lieu de prendre à chaque nœud une décision de branchement binaire (Fils gauche ou droit), on prend une décision à options multiples, selon le nombre de fils du nœud:
- Soit R la racine. Supposant que chaque nœud contient entre autres, un tableau de taille $m-1$ pour les clés et un tableau de taille m pour les pointeurs de fils. La recherche d'une clé $\ll x \gg$ se déroule comme suit:

RECHERCHE

- Soit R la racine. Supposant que chaque nœud contient, entre autre, un tableau de taille $m-1$ pour les clés et un tableau de taille m pour les pointeurs de fils. La recherche d'une clé $\ll x \gg$ se déroule comme suit:
 1. Dans le nœud R, rechercher la clé $\ll x \gg$ dans le tableau des clés (en faisant soit une recherche séquentielle soit une recherche dichotomique).
 - Si la clé existe, fin de recherche.
 - Sinon, retourner l'indice i tel que $k_i > x$ et passer a l' étape 2.
 2. Soit F_i le fils correspond a l'indice i .
 - Si ce fils est a Nil alors la clé n'existe pas, fin de recherche.
 - Sinon, remplacer R par F_i et aller l'etape 1.

RECHERCHE

- **Exemple:** Rechercher l'élément 68.



Nil i.e: L'élément n'existe pas

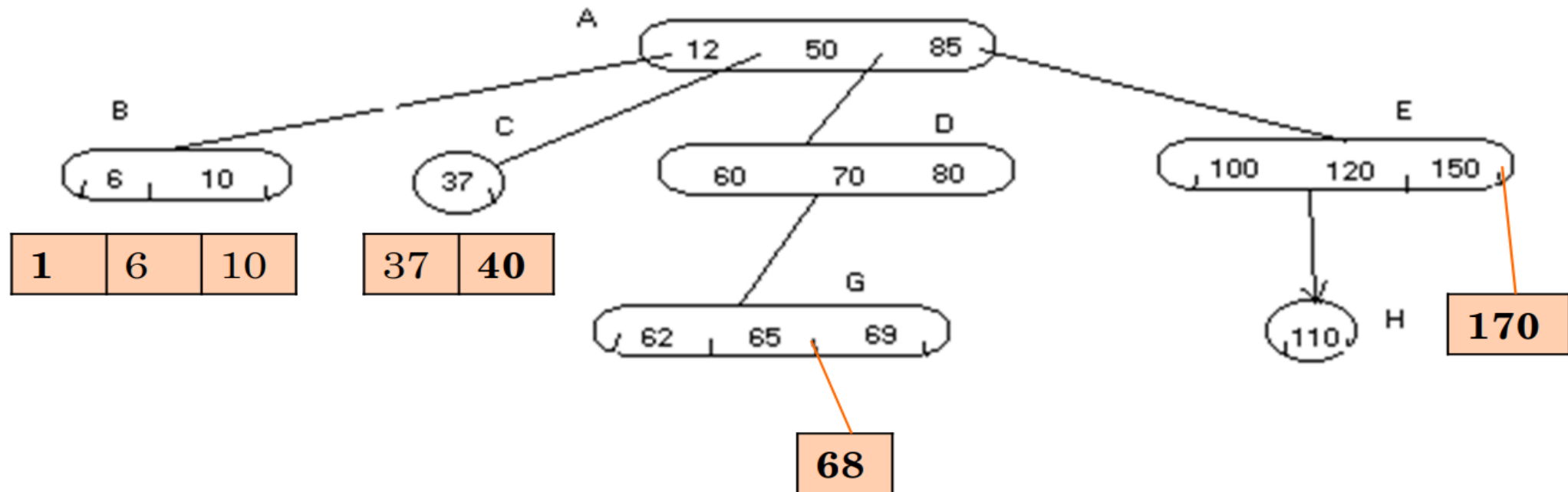
INSERTION

Soit $\ll P \gg$ la racine de l'arbre. L'insertion de la clé $\ll x \gg$ se déroule comme suit:

1. Si $\ll P \gg$ est à Nil (l'arbre est vide) alors remplacer $\ll P \gg$ par un nouveau nœud contenant la clé $\ll x \gg$. Sinon,
2. Si la clé $\ll x \gg$ existe, stop. Sinon
3. Si $\ll P \gg$ n'est pas plein (le nombre des clés insérées est strictement inférieur à $m-1$) alors insérer la clé à sa bonne position dans $\ll P \gg$ (décalage ou déplacement des clés et/ou des fils). Sinon
4. $\ll P \gg$ est plein (le nombre des clés insérées est égale à $m-1$) :
 - a) a. Recherche le fils pouvant contenir la clé $\ll x \gg$. Soit $\ll Fi \gg$ ce fils ou $x < k_i$
 - b) b. Si $\ll Fi \gg$ est à Nil alors remplacer $\ll Fi \gg$ par un nouveau nœud contenant la clé $\ll x \gg$;
 - c) c. Sinon Remplacer $\ll P \gg$ par $\ll Fi \gg$ et recommencer de l'étape 3.

INSERTION

- Exemple: Soit R un AMR d'ordre 4. Insérer les éléments suivants dans R: 1, 40, 68, 170.



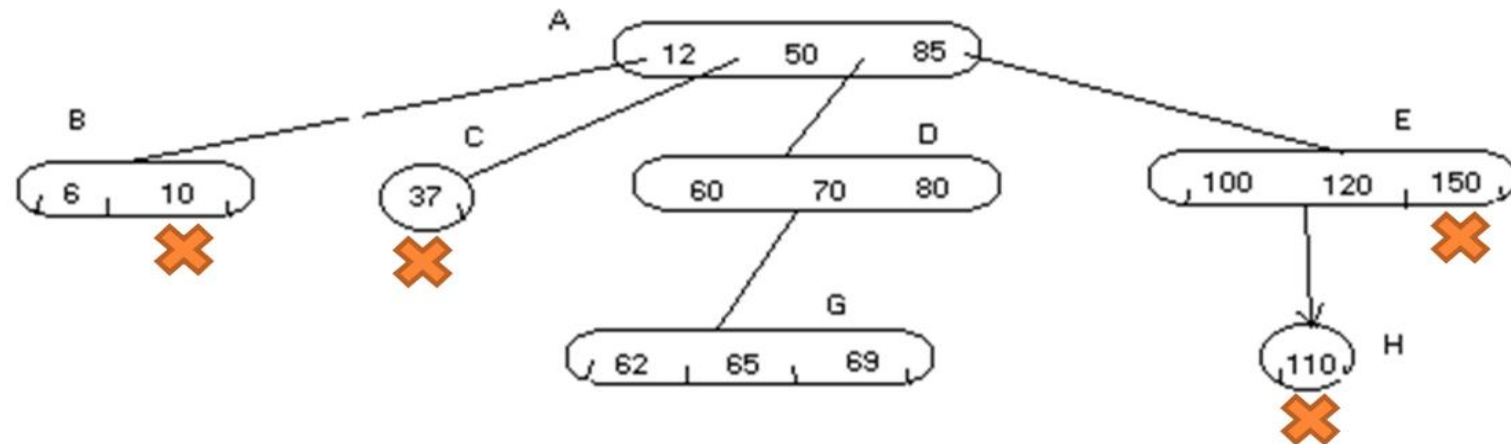
SUPPRESSION

- **Technique similaire a celle des ABR:**

1. Rechercher la clé a supprimer « **x** ». Si elle existe (soit « **P** » le nœud trouve) on passe a l' étape suivante, sinon on s'arrete
2. On distingue trois cas:
 - a) Si la clé « **x** » n'a aucun sous arbre (i.e. les deux sous arbres gauche ou droit sont vides)
 - b) Si la clé « **x** » a exactement un seul sous arbre (gauche ou droit) vide.
 - c) Si la clé « **x** » a les deux sous arbres non vides

SUPPRESSION

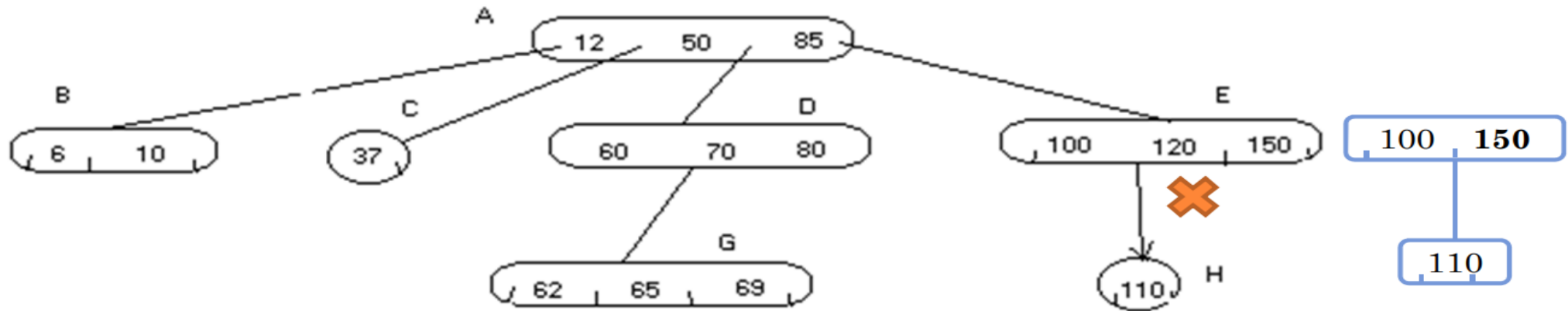
- Si la clé $\ll x \gg$ n'a aucun sous arbre (i.e. les deux sous arbres gauche ou droit sont vides) alors supprimer la clé, ensuite tasser (ou organiser) le nœud (décalage ou déplacement des clés et/ou des fils).
- **Exemple:** supprimer 10, 37, 110 ou 150,



➤ Si la clé à supprimer est la seule clé dans la feuille « **P** » alors libérer la feuille.

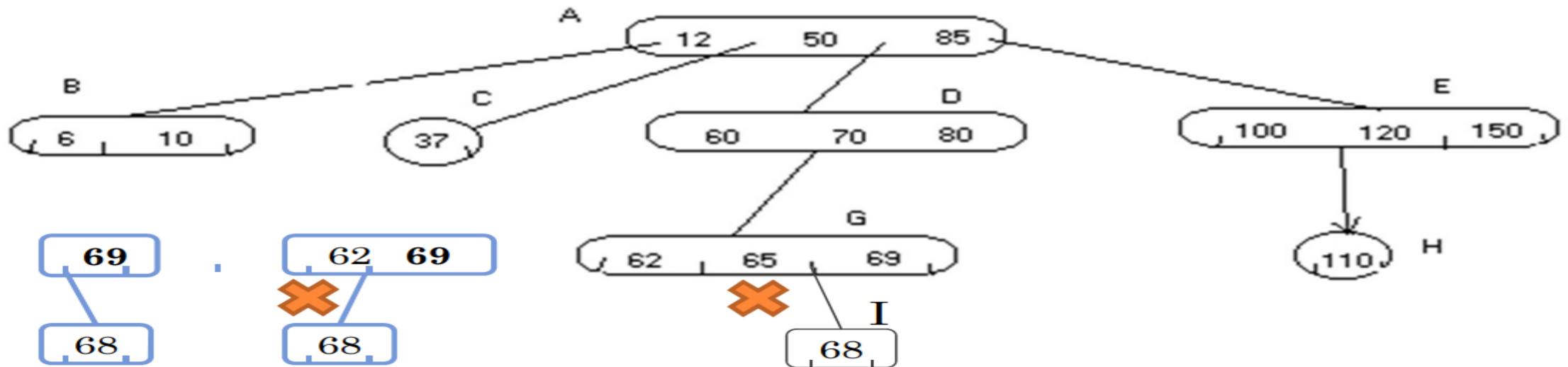
SUPPRESSION

- Si la clé « **x** » a un seul sous arbre vide alors supprimer la clé, ensuite tasser (ou organiser) le nœud (décalage ou déplacement des clés et/ou des fils).
- **Exemple:** la suppression du 120 entraine le décalage de 150 a la position de 120.

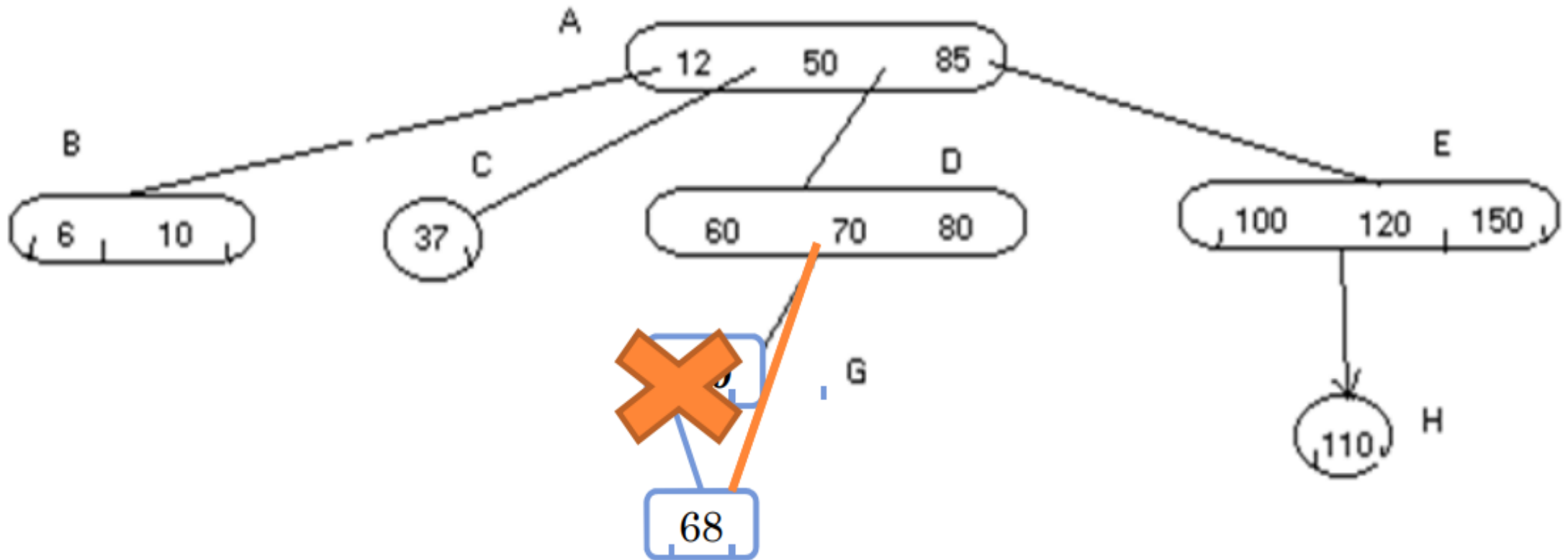


SUPPRESSION

- Si la clé « **x** » a un seul sous arbre vide alors supprimer la clé, ensuite tasser (ou organiser) le nœud (décalage ou déplacement des clés et/ou des fils).
- **Exemple:** supprimer 65, 62 et 69 de l'arbre ci dessous

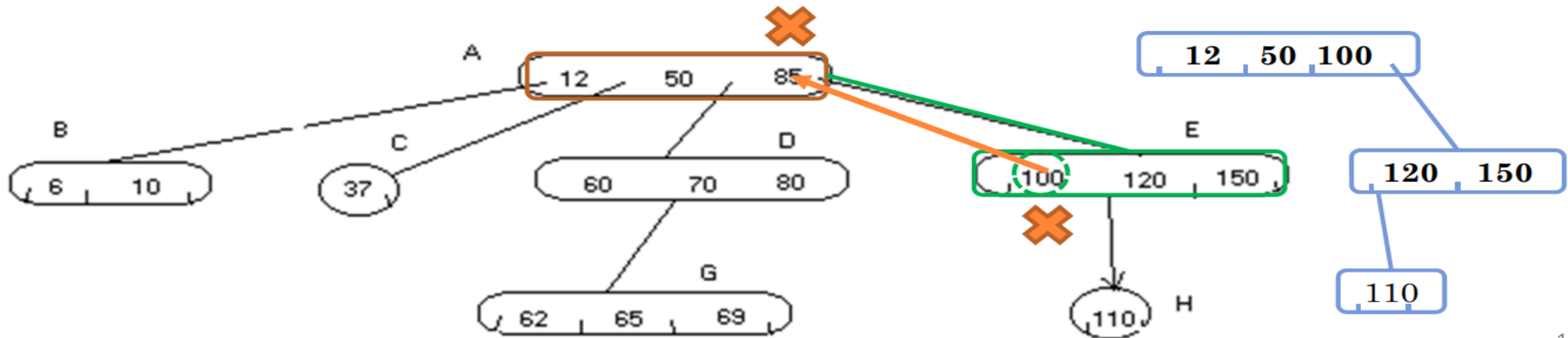


SUPPRESSION



SUPPRESSION

- Si la clé « x » a les deux sous arbres non vides alors remplacer cette clé par son successeur/prédécesseur le plus proche (soit la clé « y »), ensuite supprimer la clé « y » (aller au cas a ou b).
- **Exemple:** la suppression du 85 entraine le remplacement du 85 par 100 , ensuite la suppression du 100.



B-ARBRES: AMR EQUILIBRÉS

Introduction

- Les principaux problèmes avec les AMR sont
- Équilibrage après MAJ (insertion & suppression)
- Gaspilla d'espace dû à la suppression
- Bayer et McCreight (en 1970) ont fourni une solution à ces problèmes par l'invention des B-arbres (B pour Bayer / Boeng / Balanced).

DÉFINITION

- Un B-arbre d'ordre m (m est impair, i.e. $m = 2*d + 1$, $d \geq 1$) est un AMR qui vérifie les propriétés suivantes:
- La racine, si elle n'est pas une feuille, a au moins 2 fils.
- Tous les nœuds feuilles sont au même niveau (dernier niveau).
- Chaque nœud contient k clés avec:
 - $1 \leq k \leq 2*d$ (nœud racine)
 - $d \leq k \leq 2*d$ (nœud non racine). Autrement dit, le taux de remplissage de chaque nœud (non racine) est maintenu entre 50% et 100%

Introduction

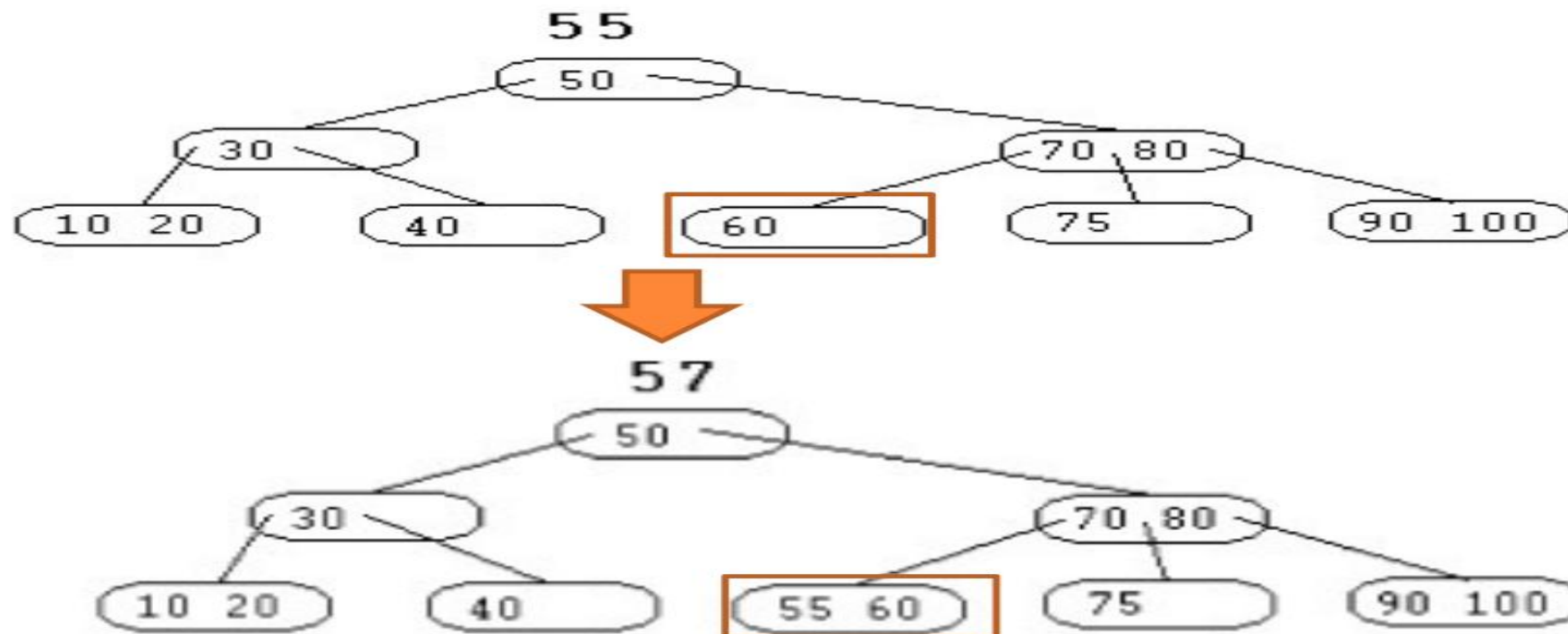
- Les insertions et les suppressions deviennent simples à réaliser sauf dans deux cas particuliers:
 - Insertion dans un nœud déjà plein (possédant $2*d$ clés, i.e. taux de remplissage égale à 100%)
 - Suppression dans un nœud entraînant un taux de remplissage inférieur à 50%

INSERTION

- **L'insertion se fait toujours au niveau des feuilles.**
- Ainsi, l'insertion de la clé « x » se déroule comme suit:
 1. Si l'arbre est vide, alors créer un nouveau nœud contenant « x ». Ce nœud sera la nouvelle racine. Sinon,
 2. Rechercher la feuille pouvant contenir « x », soit « P » la feuille trouvée.
 3. Si le nœud « P » n'est pas plein alors insérer la clé à sa bonne position.

INSERTION

- **Exemple:** Considérons le B-arbre suivant d'ordre 3, insérer les valeurs suivantes dans l'ordre : **55, 57, 95, 85, 87**



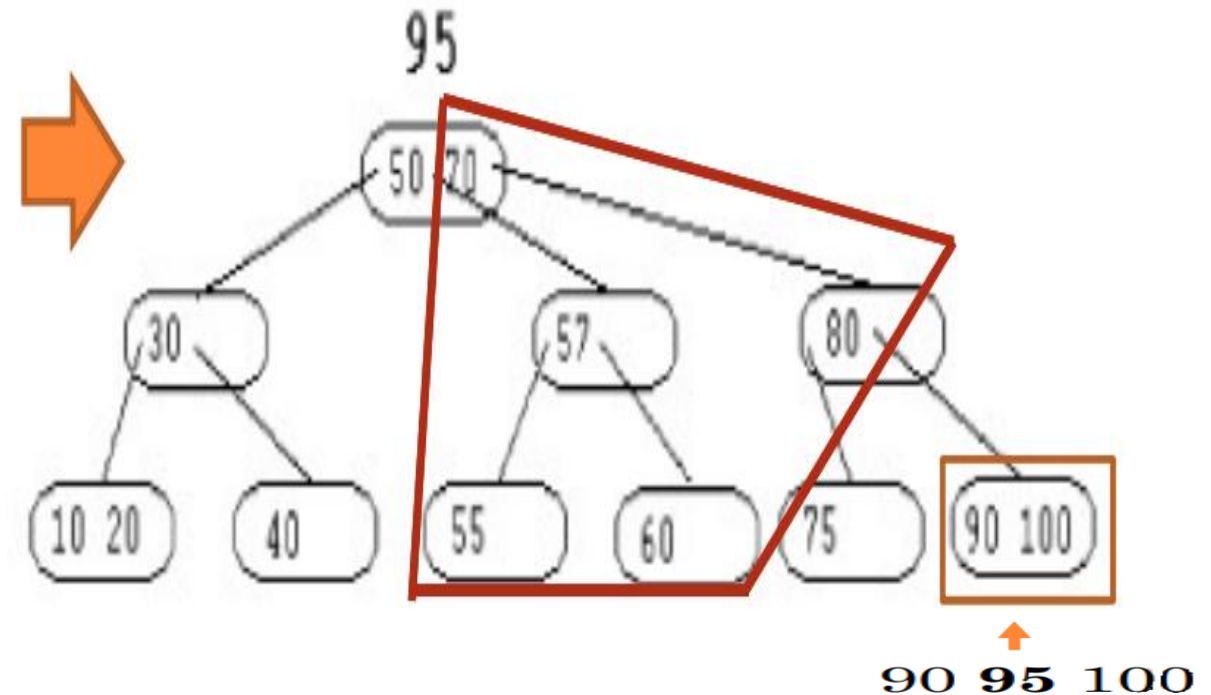
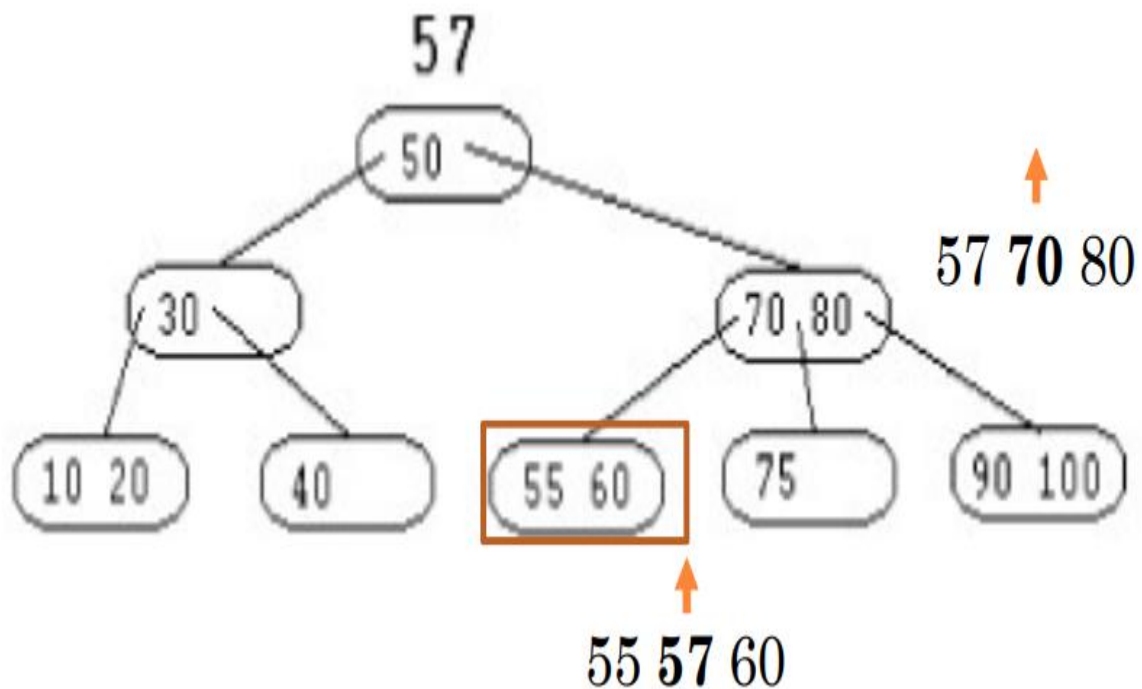
INSERTION

4. Sinon (le nœud « P » est plein), l'éclatement se fait en cascade (approche ascendante) comme suit :

- 1) Classer les clés du nœud « P » ainsi que la clé « x » dans l'ordre croissant : k_1, k_2, \dots, k_m ; soit k_{mil} la clé du milieu ou la clé médiane
- 2) Laisser les d plus petites clés ($k_1 \dots k_{mil-1}$) dans le nœud « P »
- 3) Déplacer les d plus grandes clés ($k_{mil+1} \dots k_m$) dans un nouveau nœud (soit « Q »)
- 4) Remonter la clé du milieu k_{mil} dans le nœud père (soit « Pere ») de telle sorte que le nœud « P » se trouvera à sa gauche et « Q » à sa droite.
- 5) Remplacer « P » par le nœud père « Pere ».
- 6) Si « P » est différent de Nil aller à 3 (application récursive de ce principe d'éclatement, éventuellement jusqu'à arriver à la racine),
- 7) Sinon, créer un nouveau nœud contenant la clé du milieu k_{mil} . Ce nœud sera la nouvelle racine (fin de l'algorithme).

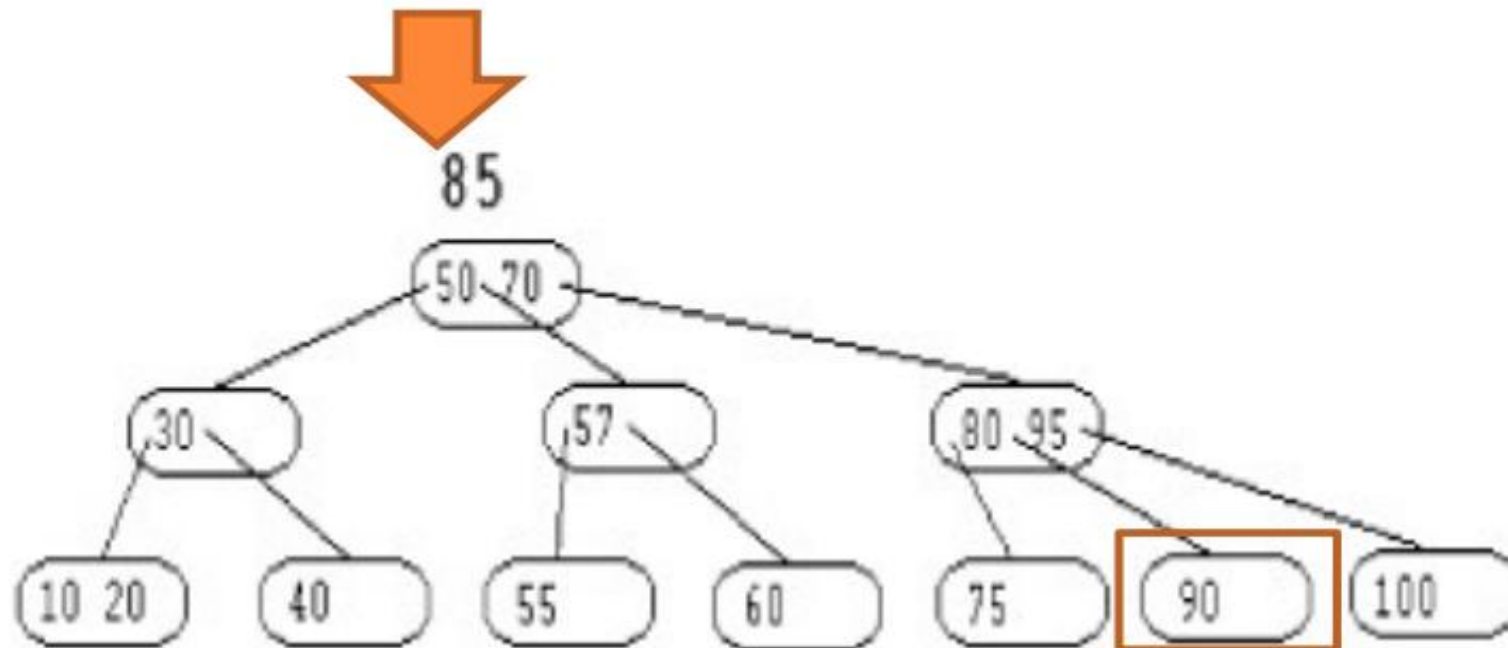
INSERTION

- **Exemple:** Considérons le B-arbre suivant d'ordre 3, insérer les valeurs suivantes dans l'ordre : **57, 95, 85, 87**



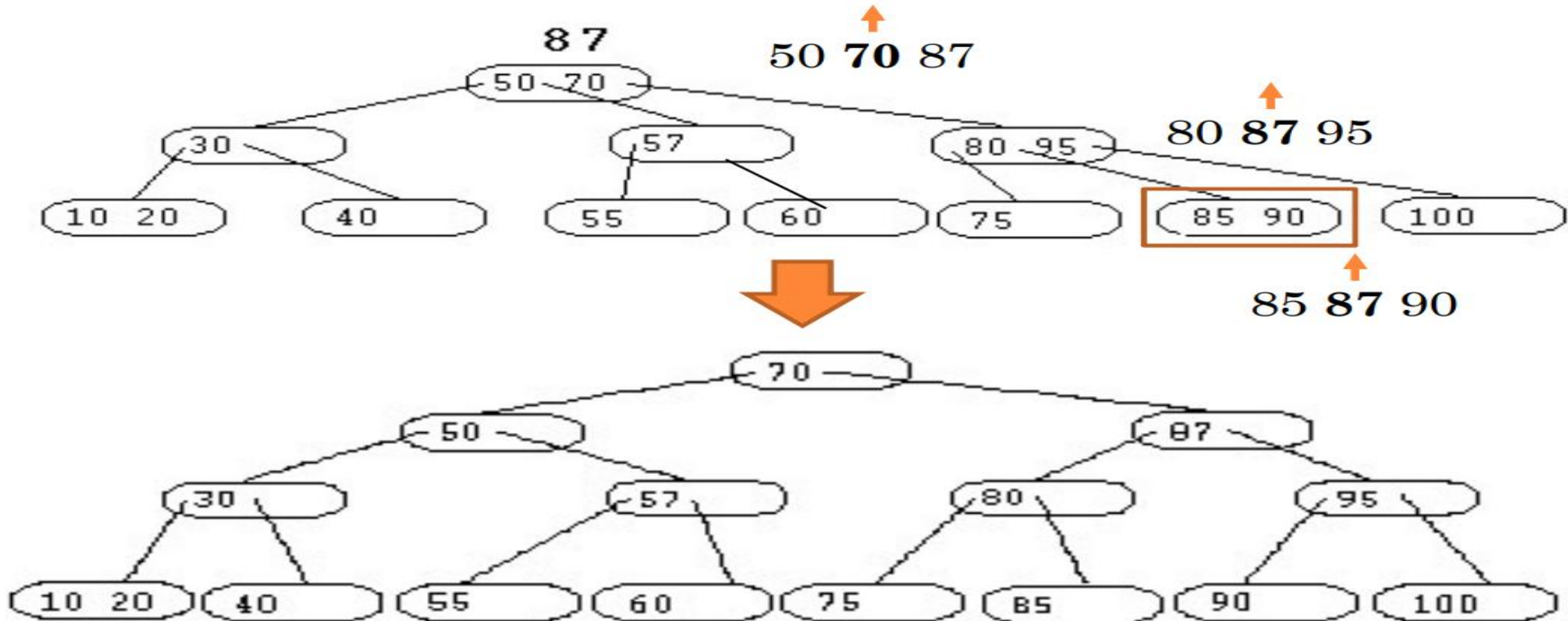
INSERTION

Exemple: Considérons le B-arbre suivant d'ordre 3, insérer les valeurs suivantes dans l'ordre : **85, 87**



INSERTION

Exemple: Considérons le **B-arbre** suivant d'ordre 3, insérer les valeurs suivantes dans l'ordre : **87**

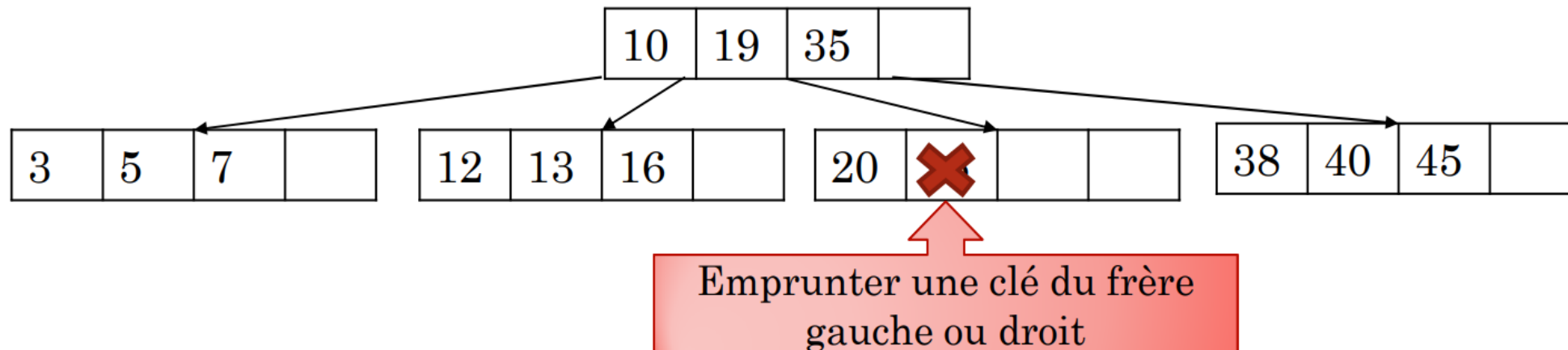


SUPPRESSION

- La suppression dans un B-arbre est similaire à la suppression dans un AMR où on distingue deux cas seulement:
 - Suppression d'une clé ayant aucun sous arbre (suppression au niveau d'une feuille)
 - Suppression d'une clé ayant deux sous arbres non vide (remplacement par le successeur/prédécesseur qui se trouve au niveau de feuille)
 - ✓ Impossible de trouver dans un B-arbre une clé ayant un seul sous arbre (vu la technique d'éclatement utilisée dans l'insertion)
- Il faut supprimer l'élément tout en préservant la qualité de B-arbre, c'est à dire en gardant au moins d clés dans le nœud (non racine).
- Ainsi, si le nœud feuille (soit « P ») qui contenait la clé à supprimer a moins de d clés, alors l'action suivante est entreprise :

SUPPRESSION

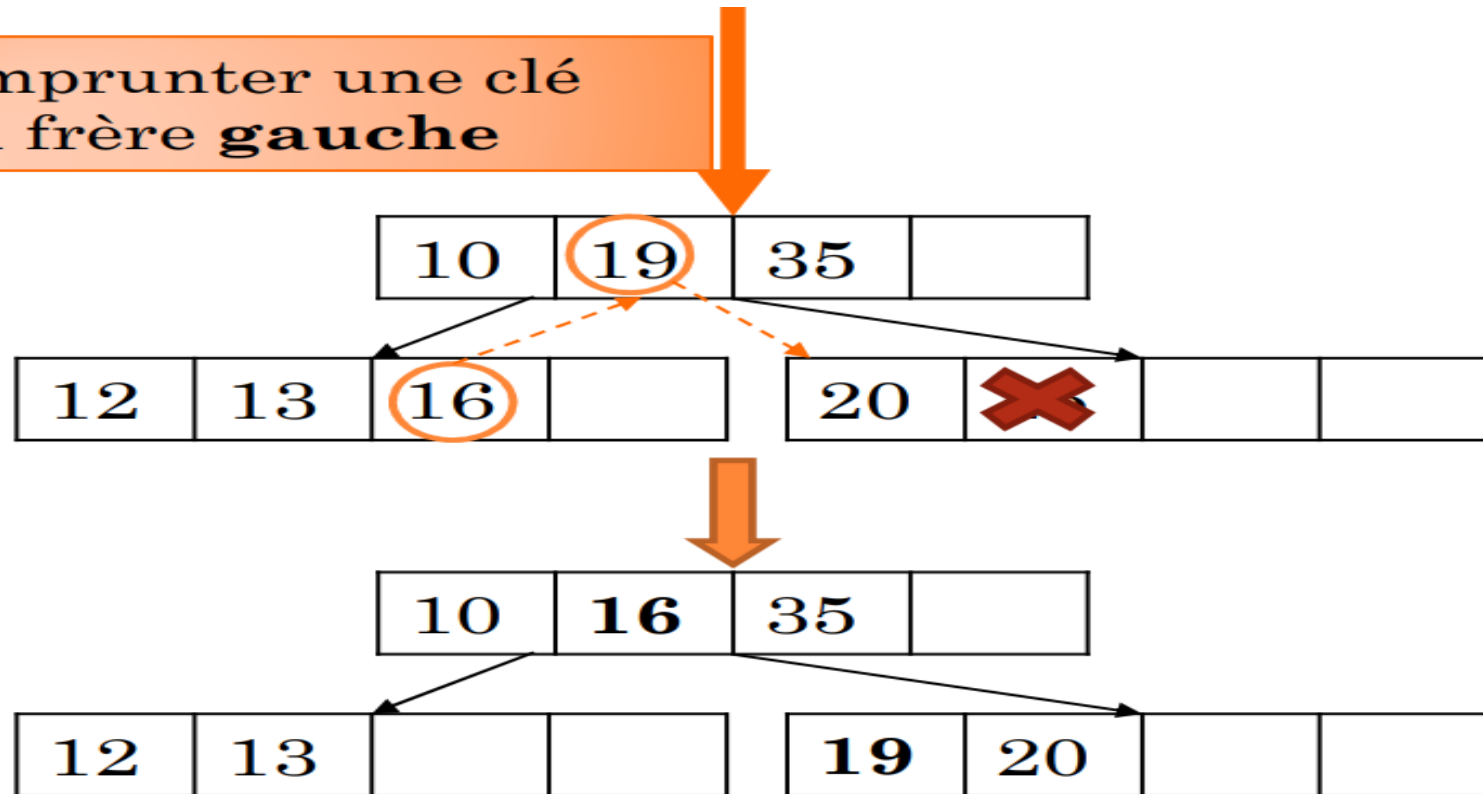
- **Cas 1:** Si l'un des frères (gauche ou droit) contient plus de d clés, alors la clé, soit K_s , dans le nœud père qui sépare entre les deux frères est ajoutée au nœud « P » et le dernier (si frère gauche) ou le premier élément (si frère droit) est ajoutée au père à la place de K_s .
- Exemple: Supprimer 25 du B-arbre d'ordre 5 suivant:



SUPPRESSION

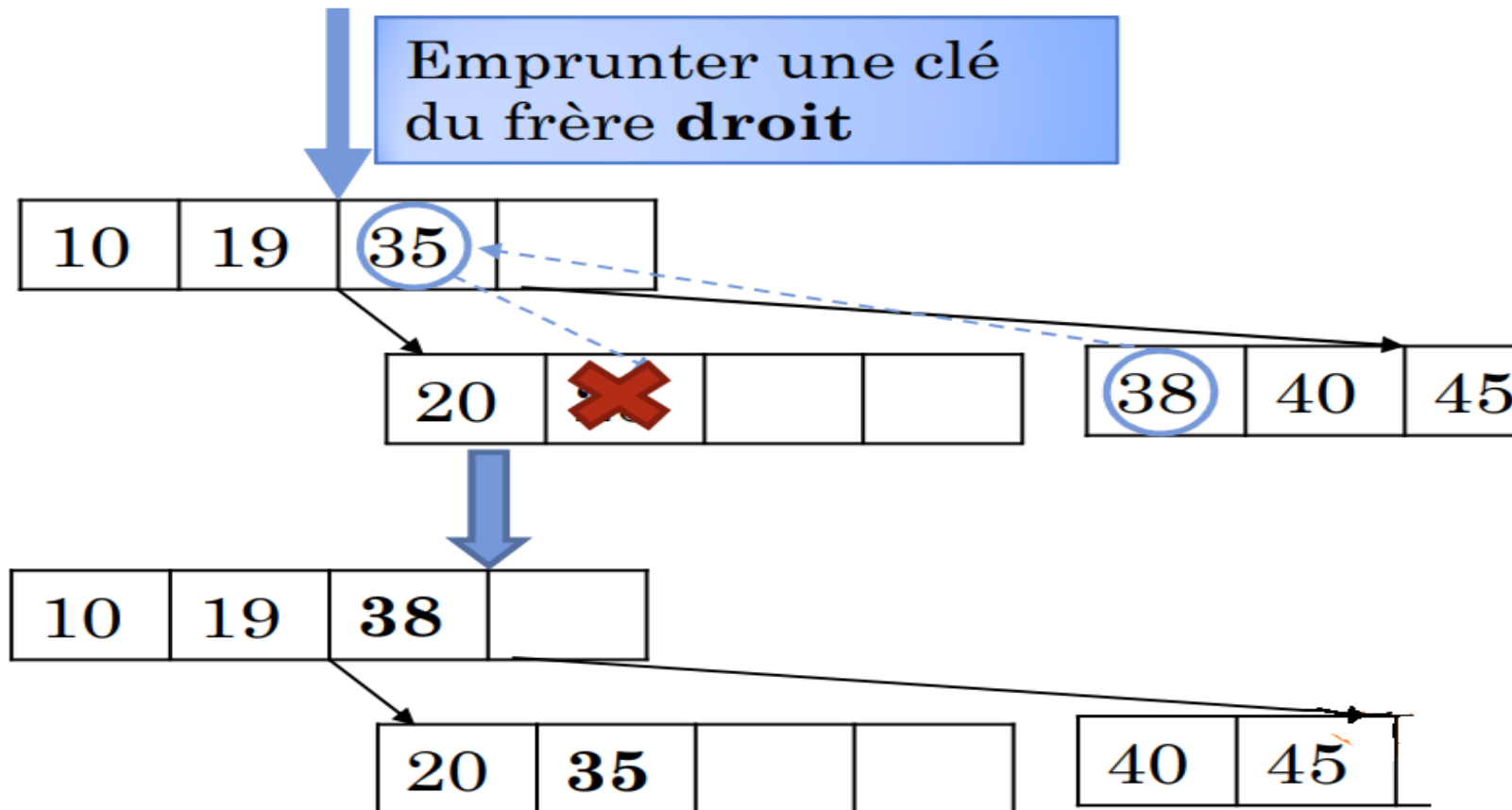
- Exemple (cas 1): Supprimer 25 du B-arbre d'ordre 5 suivant:

Emprunter une clé
du frère **gauche**



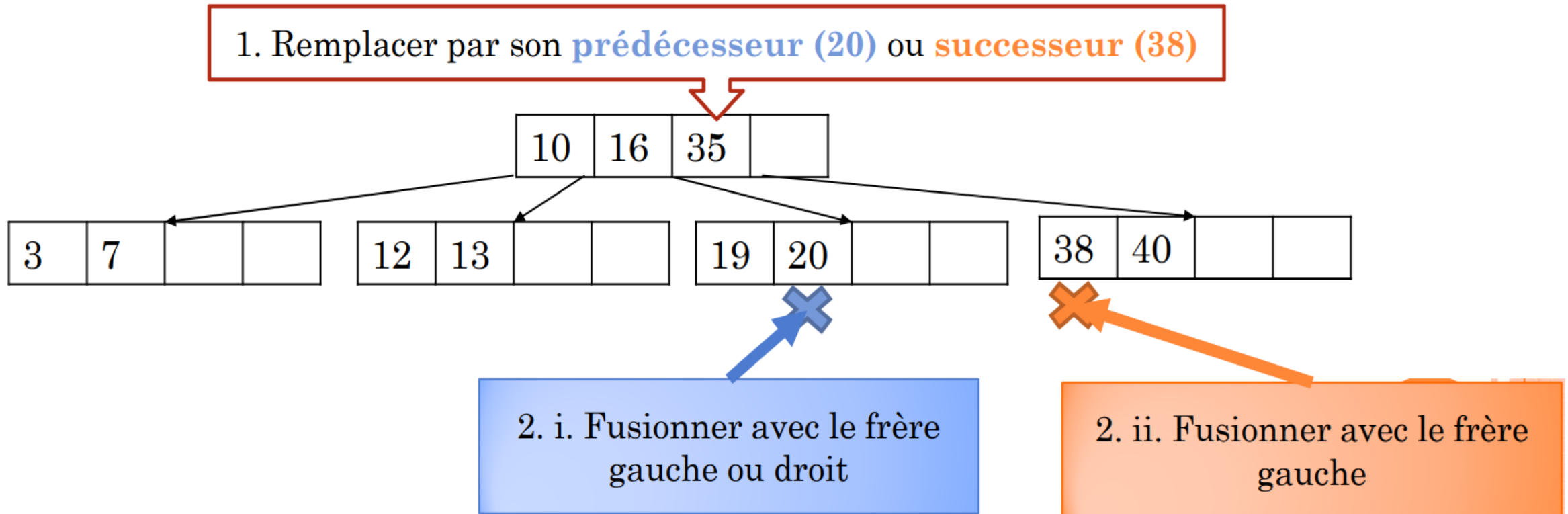
SUPPRESSION

- Exemple (cas 1): Supprimer 25 du B-arbre d'ordre 5 suivant:



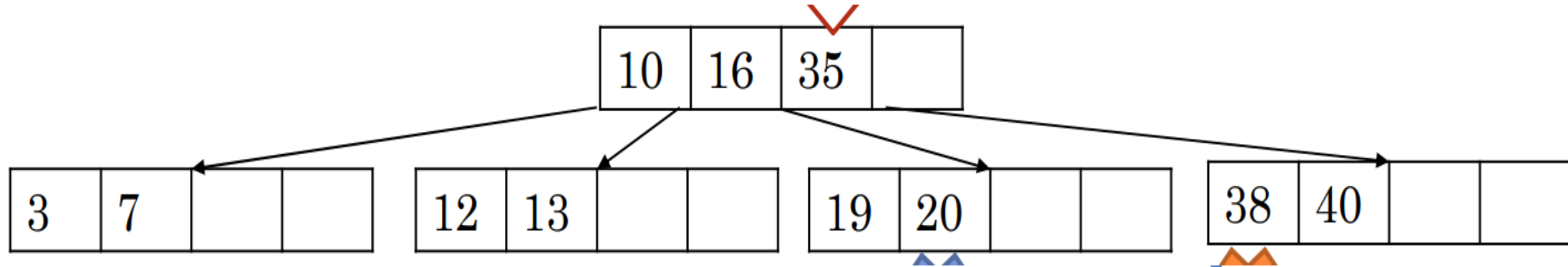
SUPPRESSION

- **Cas 2-a:** Si les deux frères contenaient exactement d clés, le nœud « P » et l'un de ses frères seront concaténés (fusionnés ou consolidés) en un seul nœud qui contient aussi la clé séparatrice de leur père.
- Exemple (cas 2-a): Supprimer **35** du B-arbre d'ordre 5 suivant:

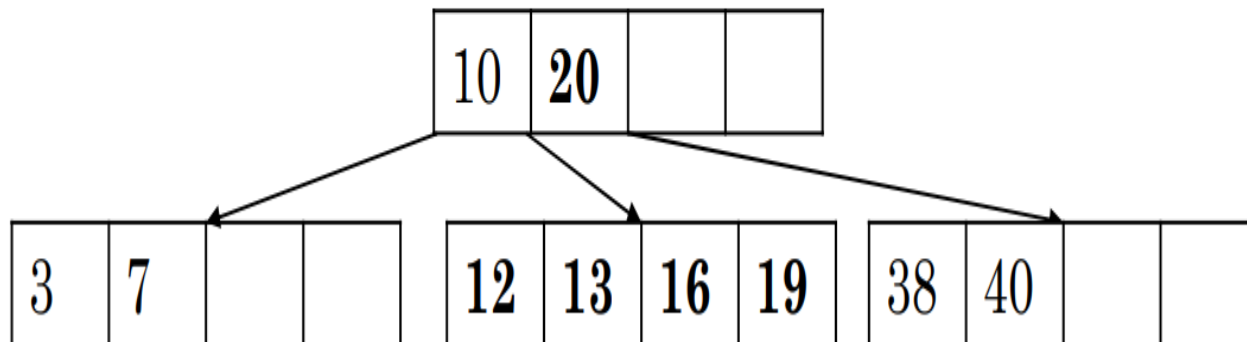


SUPPRESSION

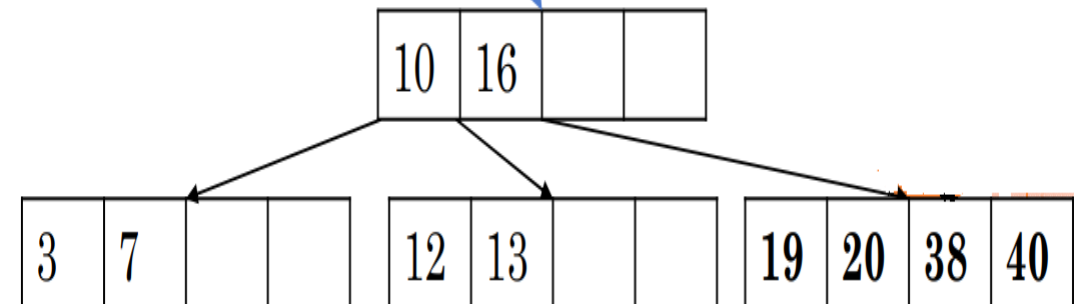
Exemple (cas 2-a): Supprimer **35** du B-arbre d'ordre 5 suivant:



2. Fusionner avec le frère gauche

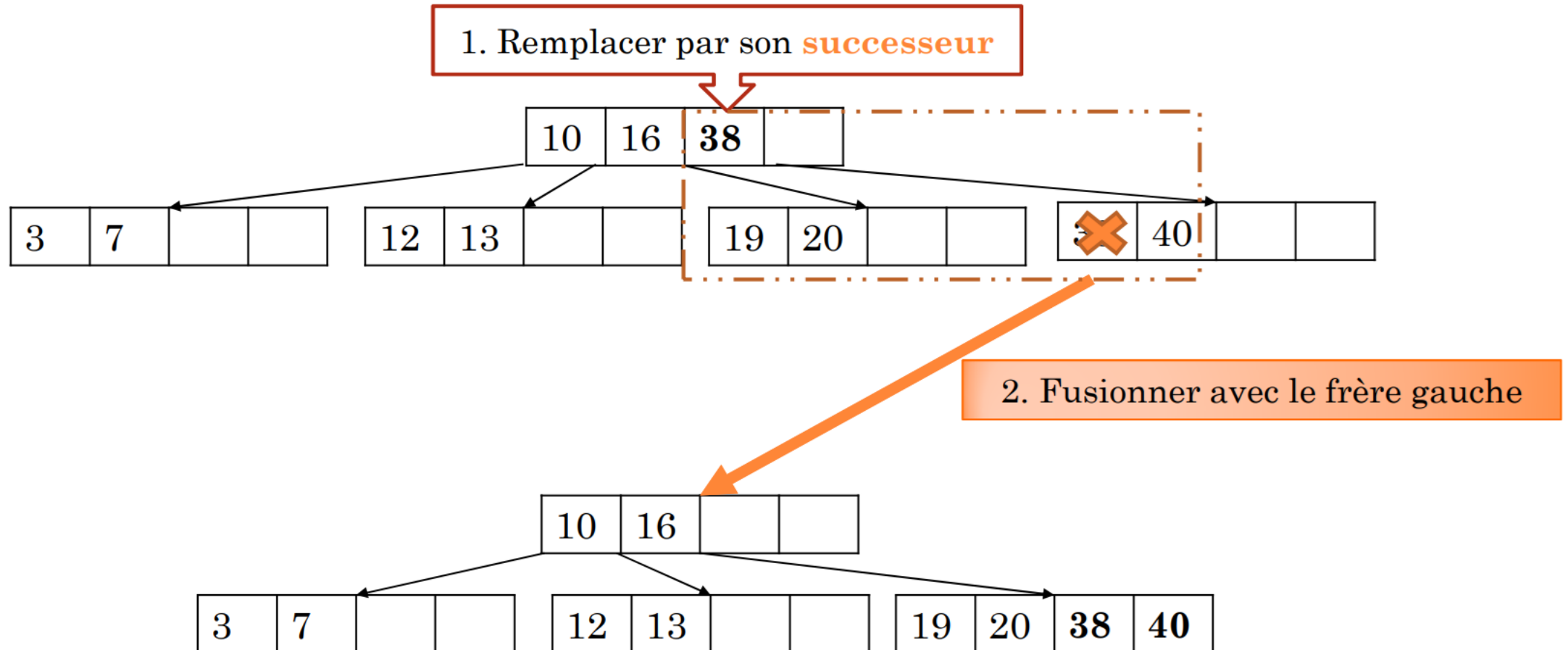


2. Fusionner avec le frère droit



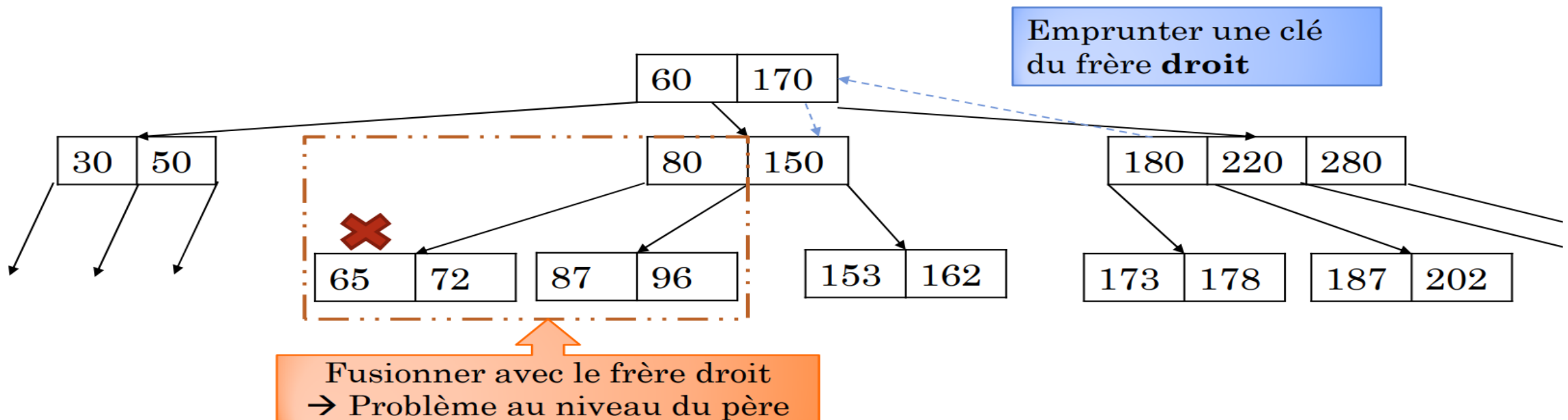
SUPPRESSION

Exemple (cas 2-a): Supprimer **35** du B-arbre d'ordre 5 suivant:



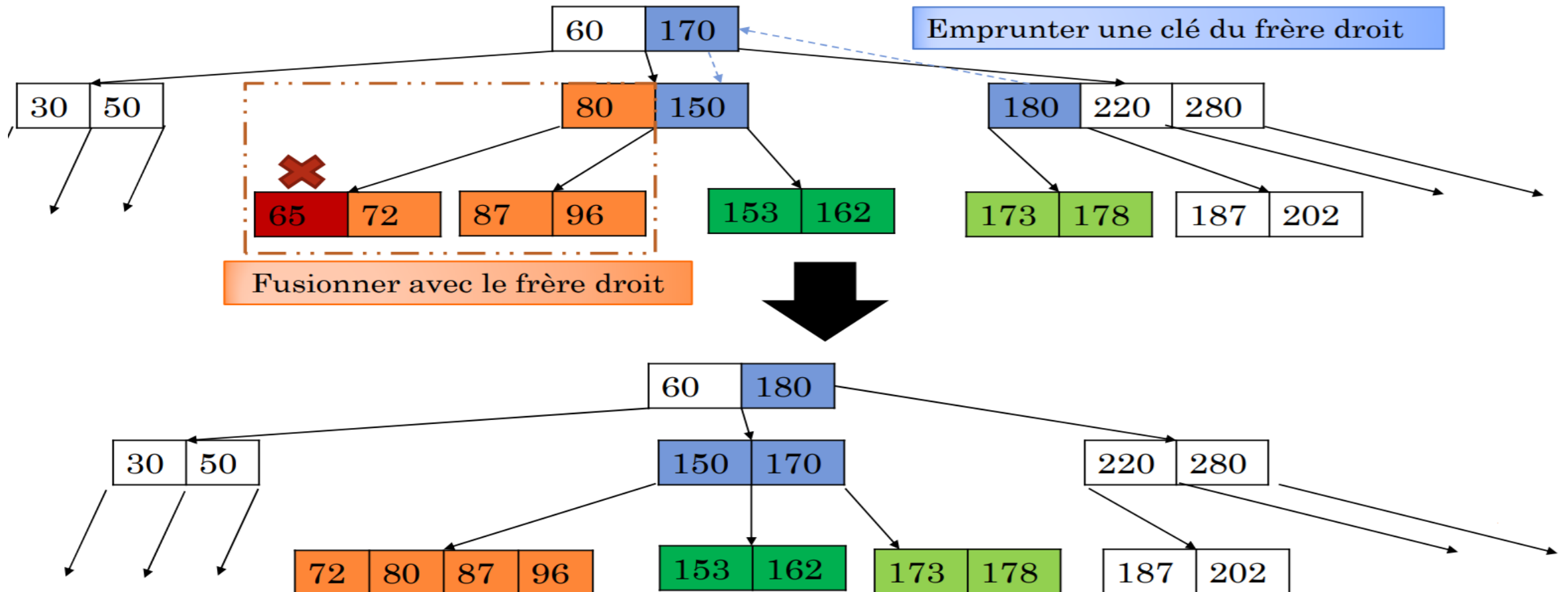
SUPPRESSION

- **Cas 2-b:** Il est aussi possible que le père contienne seulement des clés et par conséquent il n'a pas de clé à donner. Dans ce cas, il peut emprunter de son père et frère.
- Exemple (Cas 2-b) : Supprimer 65 du B-arbre d'ordre 5 suivant:



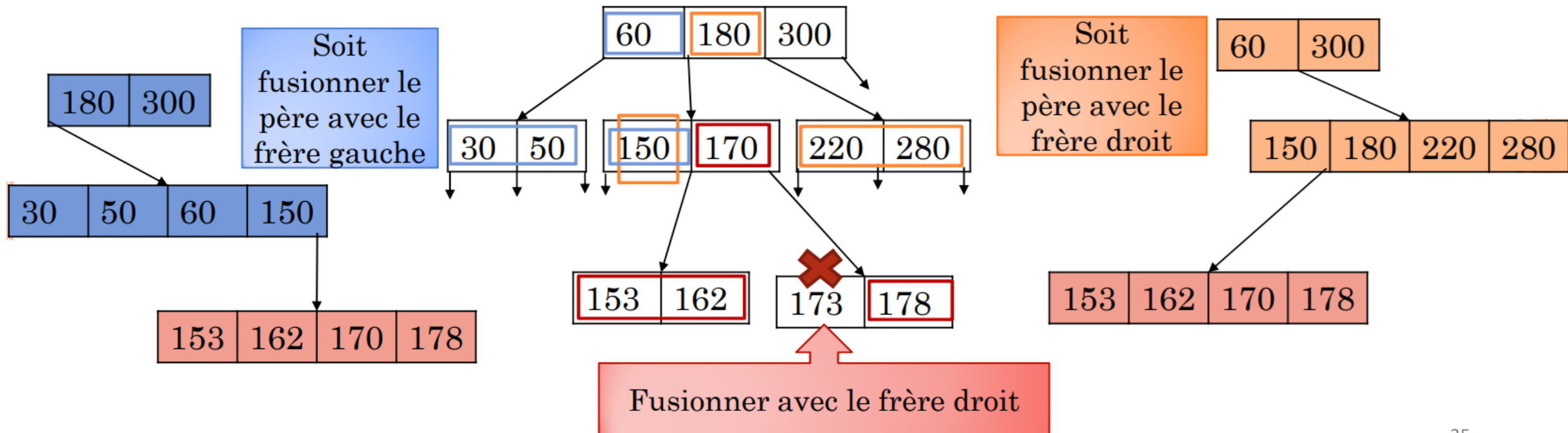
SUPPRESSION

- Exemple (Cas 2-b) : Supprimer 65 du B-arbre d'ordre 5 suivant:



SUPPRESSION

- **Cas 2-c:** Il est aussi possible que les frères du père n'ont pas des clés à donner, le père et son frère peuvent aussi être concaténés et une clé est prise du grand père.
- Exemple : Supprimer 173 du B-arbre d'ordre 5 suivant:

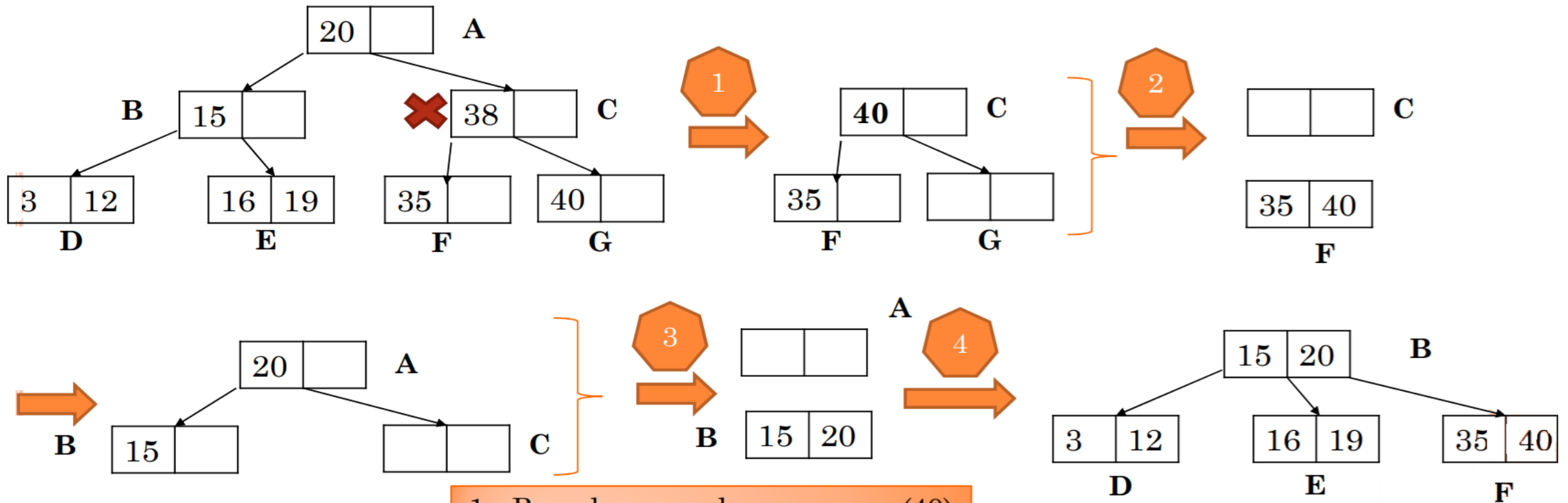


SUPPRESSION

- **Cas 2-d:** Dans le pire des cas, si tous les antécédents d'un nœud et leurs frères contiennent exactement d clés, une clé sera prise de la racine (due aux concaténations en cascades):
- Si la racine avait plus d'une clé, ceci termine le processus.
- Si la racine contenait une seule clé, elle serait utilisée dans la concaténation. Le nœud racine est libéré et la profondeur de l'arbre est réduit d'une unité.

SUPPRESSION

- Exemple (**cas 2-d**): Supprimer 38 du B-arbre d'ordre 3 suivant



1. Remplacer par le successeur (40)
2. Fusionner F et G avec la clé de C
3. Fusionner C et B avec la clé de A
4. Supprimer le nœud A