Chapitre 4

Arbres 2-3-4 et arbres bicolores

Exercice 4.1 Hauteur d'un arbre 2-3-4
Soit t un *arbre 2-3-4* contenant n nœuds et soit $h(t)$ sa hauteur.

Montrer que

$$\log_4(3n+1) - 1 \leqslant h(t) \leqslant \log_2(n+1) - 1$$

2. En déduire que si m est le nombre d'**éléments** de l'arbre, on a

$$\log_4(m+1) - 1 \leqslant h(t) \leqslant \log_2(m+1) - 1$$

Exercice 4.2 Insertion dans un arbre 2-3-4 avec éclatements en remontée
Soit la suite des entiers $4, 35, 10, 13, 3, 30, 15, 12, 7, 40, 20, 11, 6$.
Donner la construction de l'arbre 2-3-4 par insertions successives aux feuilles, à partir de l'arbre vide.
L'insertion dans un arbre vide entraîne la création d'un 2-nœud.
L'insertion dans un i-nœud, $i = 2, 3$, entraîne la création d'un $i+1$-nœud.
L'insertion dans un 4-nœud, provoque l'éclatement de la feuille en deux 2-nœuds contenant respectivement le plus petit et le plus grand élément de la feuille, et l'élément médian doit être ajouté au nœud père de la feuille.
Les éclatements peuvent remonter en cascade, éventuellement sur toute la hauteur de l'arbre si le chemin suivi pour déterminer l'endroit de l'insertion n'est formé que de 4-nœuds.

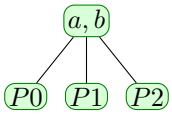
Exercice 4.3 Insertion dans un arbre 2-3-4 avec éclatements à la descente
Pour éviter la propagation des éclatements de bas en haut, il suffit de travailler sur les *arbres 2-3-4* qui ne contiennent jamais deux 4-nœuds à la suite : dans ce cas toute insertion provoque au plus un éclatement. Ceci peut être réalisé en éclatant les 4-nœuds à la descente, lors de la recherche de la place de l'élément à ajouter.
Reprendre l'exemple de l'éxercice précédent en utilisant la méthode d'éclatement à la descente.

Exercice 4.4 Éclatements des 4-nœuds.
Isoler sous forme de schémas d'*arbres 2-3-4* les modifications locales possibles lors d'éclatements des 4-nœuds.

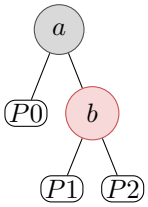
Exercice 4.5 Représentation d'arbre 2-3-4
Donner des exemples de représentation des différents types de nœuds. Quels sont les inconvénients des ces représentations ?

Exercice 4.6 Un arbre 2-3-4
Proposer un type OCaml `type 'a tree234` pour définir le type *arbre 2-3-4*.

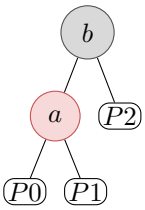
Exercice 4.7 Recherche dans un arbre 2-3-4
Écrire une fonction `tree234_search` qui recherche un élément dans un *arbre 2-3-4*, et renvoie 0 s'il n'est pas présent, et sa multiplicité dans le cas contraire.
Quelle est sa complexité ?



(a) 3-nœud

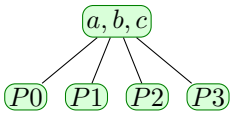


(b) arbre bicolore "penché à droite"

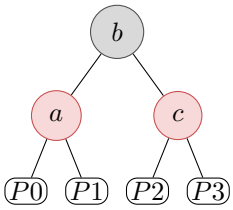


(c) arbre bicolore "penché à gauche"

FIGURE 4.1: Transformation d'un 3-nœud, Fig. 4.1(a), en arbres bicolores, Fig. 4.1(b)(c)



(a) 4-nœud



(b) arbre bicolore

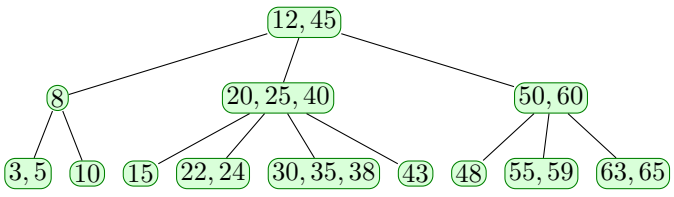
FIGURE 4.2: Transformation d'un 4-nœud, Fig. 4.2(a), en arbre bicolore, Fig. 4.2(b)

Exercice 4.8 Visualization d'un arbre 2-3-4
Écrire une fonction `tree234_to_dot` qui produit la représentation de l'*arbre 2-3-4* au format `dot`, syntaxe `Graphviz` consultable à l'adresse :
`http://www.graphviz.org/pdf/dotguide.pdf`

Exercice 4.9 Représentation d'un *arbre 2-3-4* par un arbre bicolore
Soit l'arbre 2-3-4 donné sur la Fig. 4.3. Dessiner sa représentation sous la forme d'un arbre bicolore. On utilisera les transformations des k-nœuds, $k = 2, 3, 4$ comme illustré sur la Fig. 4.1 et la Fig. 4.2

Exercice 4.10 Un arbre bicolore
Proposer un type OCaml `type 'a bct` pour définir le type *arbre bicolore*.

Exercice 4.11 Recherche dans un arbre bicolore
Écrire une fonction `bct_search` qui recherche un élément dans un arbre bicolore.



(a)

FIGURE 4.3: Arbre 2-3-4

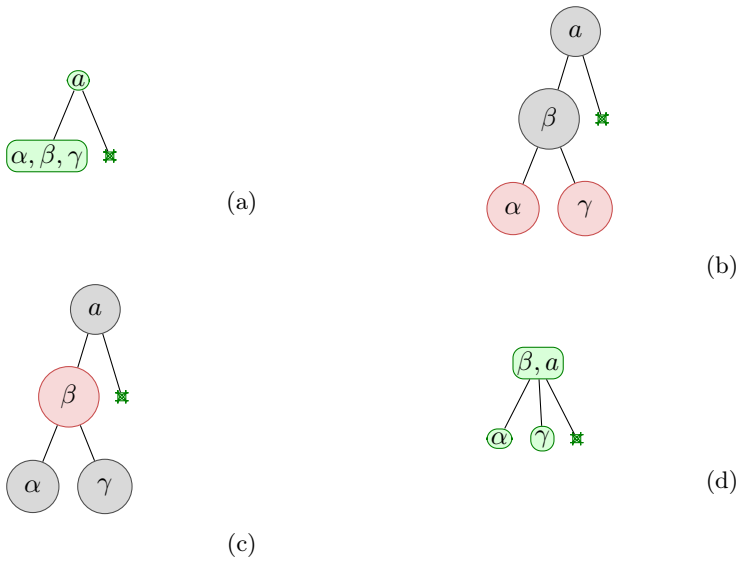


FIGURE 4.4: Eclatement d'un 4-nœud, fils gauche d'un 2-nœud, Fig. 4.4(a), en un 3-nœud et deux 2-nœuds, Fig. 4.4(d)

Exercice 4.12 Simulation des éclatements des nœuds d'un *arbre 2-3-4*
Pour chaque type d'éclatement d'un 4-nœud, intervenant dans l'insertion à la descente dans un *arbre 2-3-4*, montrer comment il peut être simulé sur un *arbre bicolore*.
Des exemples sont donnés sur les figures Fig. 4.4, Fig. 4.5, Fig. 4.6, Fig. 4.7 et Fig. 4.8

Exercice 4.13 Rotations
Pour maintenir les propriétés des arbres bicolores on rééquilibre l'arbre par des transformations locales appelées rotations. Il existe quatre types de rotation : les rotations simples, à droite (Fig. 4.9) et à gauche (Fig. 4.10), et les rotations doubles, gauche-droite (Fig. 4.11) et droite-gauche (Fig. 4.12).
En utilisant des rotations écrire une fonction `bct_combine_left e3 l r` qui transforme l'arbre bicolore de la Fig. 4.13(a) en l'arbre de la Fig. 4.13(b).
Par analogie, écrire une fonction `bct_combine_right e1 l r` qui transforme l'arbre bicolore de la Fig. 4.14(a) en l'arbre de la Fig. 4.14(b).

Exercice 4.14 Implémentation d'arbres bicolores
Écrire une fonction `bct_insert e t` pour l'insertion de l' élément `e` dans un arbre bicolore `t`.

Exercice 4.15 Construction d'arbres bicolores
Écrire une fonction `list_to_tree l insert_fun` qui à partir de la liste `l` et la fonction d' insertion `insert_fun` construit l'arbre en insérant successivement les éléments de la liste.
Tester cette fonction pour construire l' arbre bicolore à partir de la liste $4, 10, 35, 13, 15, 7, 12, 40, 20$ et 6.

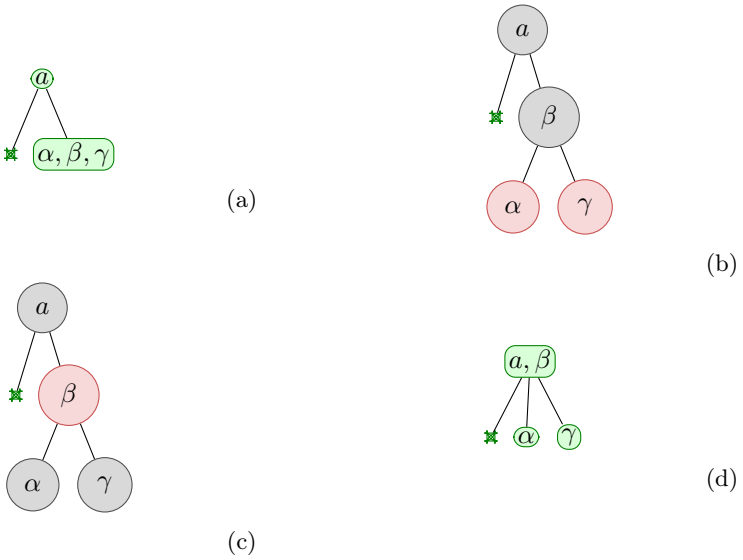


FIGURE 4.5: Eclatement d'un 4-nœud, fils droit d'un 2-nœud, Fig. 4.5(a), en un 3-nœud et deux 2-nœuds, Fig. 4.5(d)

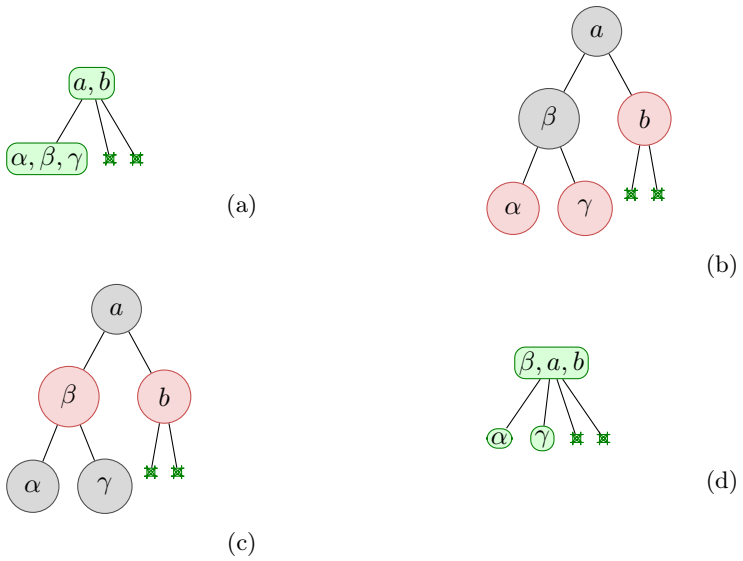


FIGURE 4.6: Eclatement d'un 4-nœud, fils gauche d'un 3-nœud, Fig. 4.6(a), en un 4-nœud et deux 2-nœuds, Fig. 4.6(d)

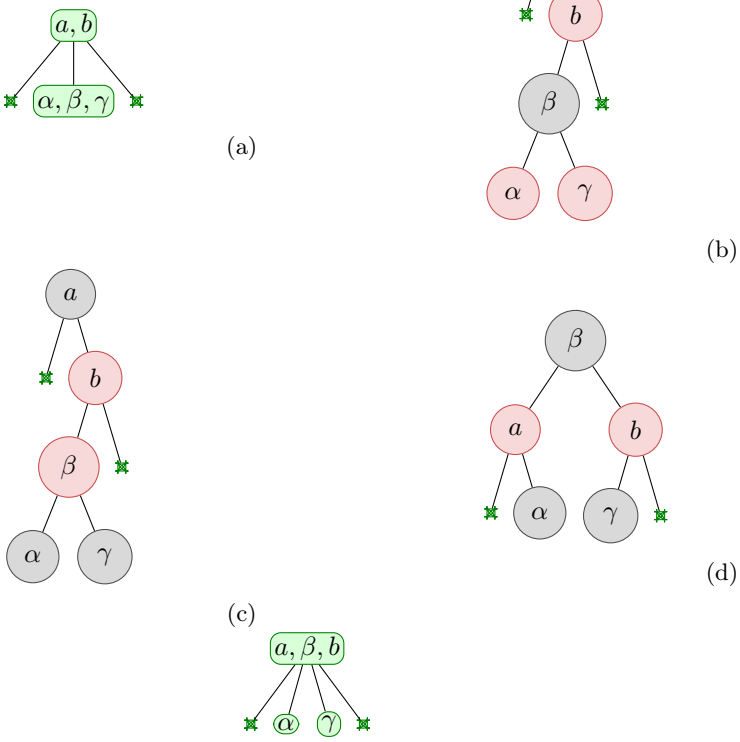


FIGURE 4.7: Eclatement d'un 4-nœud, fils médian d'un 3-nœud, Fig. 4.7(a), en un 4-nœud et deux 2-nœuds, Fig. 4.7(d)

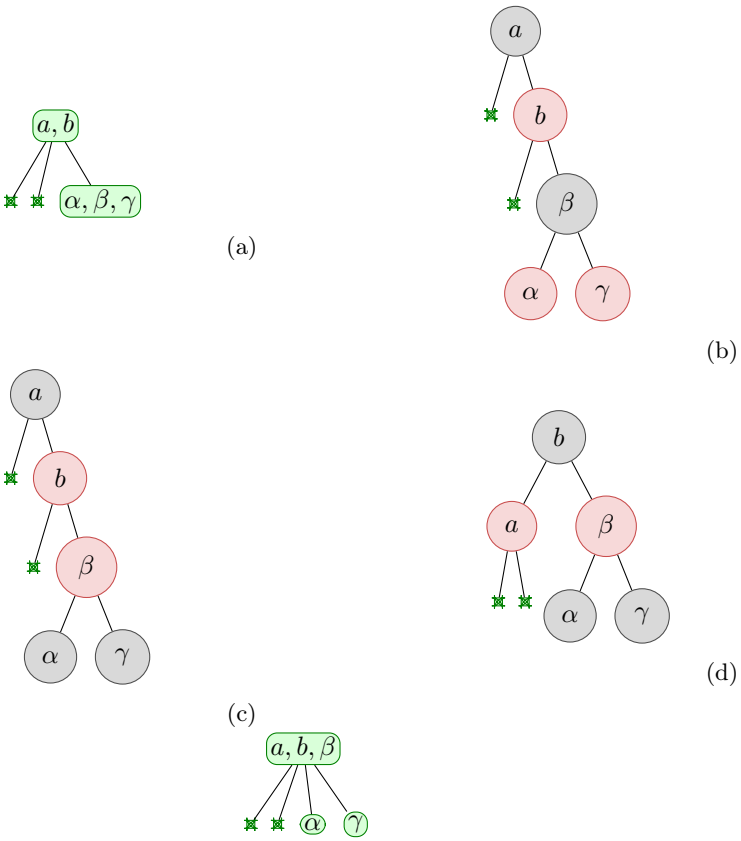


FIGURE 4.8: Eclatement d'un 4-nœud, fils droit d'un 3-nœud, Fig. 4.8(a), en un 4-nœud et deux 2-nœuds, Fig. 4.8(d)

FIGURE 4.9: Rotation droite

FIGURE 4.10: Rotation gauche

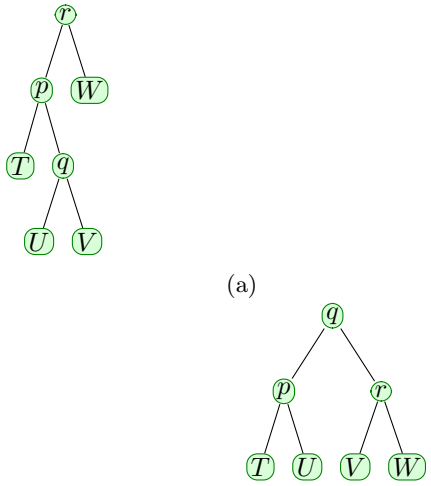


FIGURE 4.11: Rotation gauche-droite

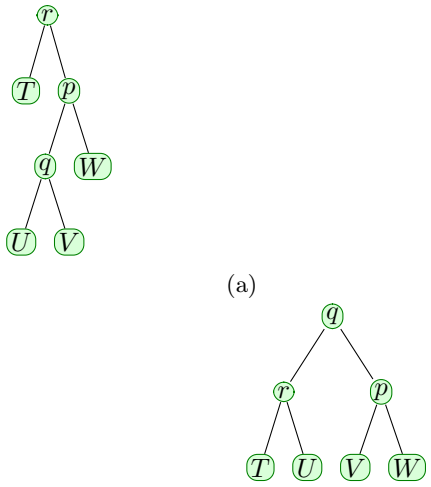


FIGURE 4.12: Rotation droite-gauche

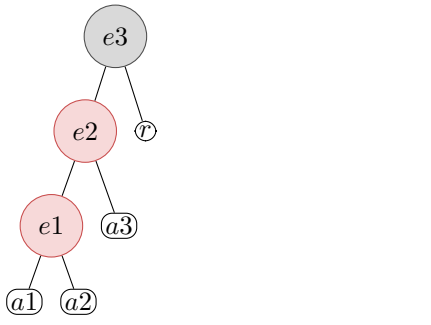


FIGURE 4.13: Rééquilibrage à droite

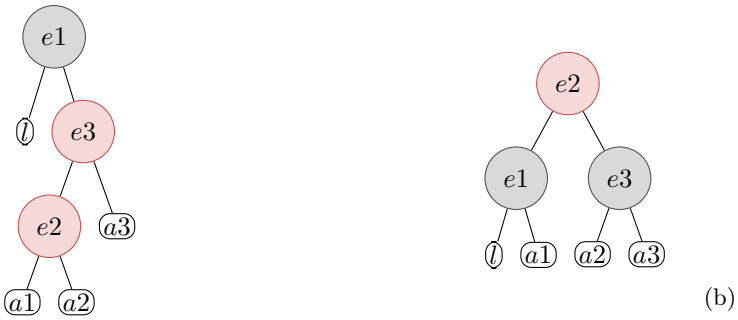


FIGURE 4.14: Rééquilibrage à gauche