REVUE FRANÇAISE D'AUTOMATIQUE INFORMATIQUE
RECHERCHE OPÉRATIONNELLE. INFORMATIQUE THÉORIQUE

JEAN FRANÇON

Arbres binaires de recherche : propriétés combinatoires et applications

Revue française d'automatique informatique recherche opérationnelle. Informatique théorique, tome 10, n° R3 (1976), p. 35-50

<http://www.numdam.org/item?id=ITA_1976__10_3_35_0>

R.A.I.R.O. Informatique Théorique
(vol. 10, n° 12, décembre 1976, p. 35 à 50).

ARBRES BINAIRES DE RECHERCHE : PROPRIÉTÉS COMBINATOIRES ET APPLICATIONS (*)

par Jean FRANÇON (1)

Communiqué par R. Cori

Résumé. — *On présente de nouvelles propriétés combinatoires des arbres binaires de recherche ("binary search trees" dans la terminologie anglaise consacrée), qui permettent de dénombrer certaines classes de ces arbres. Ces résultats sont utilisés pour analyser certains algorithmes relatifs à ces arbres.*

1. INTRODUCTION

La méthode de stockage et recherche d'information par arbre binaire (en anglais « binary search trees ») est très connue des programmeurs et fréquemment utilisée. Notre propos est de donner des propriétés combinatoires de ces arbres en vue d'analyser les algorithmes qui s'y rapportent.

Rappelons, tout d'abord, l'essentiel de cette méthode. A partir d'une suite d'articles caractérisés chacun par une clé, on construit un arbre binaire tel que chaque sommet soit étiqueté par une clé. Ayant construit un tel arbre, on peut retrouver un article connaissant sa clé par un algorithme analogue à celui qui a permis de construire l'arbre. Ces arbres binaires de recherche ne dépendent, en fait, que de l'ordre des clés dans la suite. On est donc amené, pour analyser les performances des algorithmes qui se rapportent à ces arbres, à étudier l'ensemble des arbres binaires de recherche construits à partir de toutes les permutations d'un même ensemble de clés distinctes.

Certains résultats sont déjà connus sur ce sujet. On trouvera dans l'ouvrage de D. Knuth [10] un exposé de l'état des connaissances acquises sur cette question et une bibliographie complète. Nous nous y référerons constamment. Les auteurs qui ont jusqu'ici étudié ce type d'arbres se sont surtout attachés à énumérer le nombre de comparaisons effectuées dans la construction de ces arbres et dans la recherche d'un article dans un arbre déjà construit (*voir* [12] et les références dans [10]). Ici nous nous sommes efforcés non seulement de donner de nouveaux résultats mais aussi d'introduire des tech-

(*) Reçu février 1976.
(1) Centre de Calcul du C.N.R.S., Strasbourg-Cronenbourg et Institut de Recherche Mathématique Avancée, Strasbourg.

niques puissantes généralisables à des problèmes voisins. Dans le même temps il apparaît que les problèmes d'énumération relatifs à ces arbres se ramènent souvent à des problèmes classiques et résolus d'énumération de certaines classes de permutations et font apparaître des suites classiques de nombres, nombres eulériens, nombres d'Euler, par exemple.

Un informaticien confronté au calcul des performances d'un algorithme relatif aux arbres binaires de recherche doit d'abord ramener ce calcul à un problème de dénombrement. Ensuite il doit mettre en jeu des méthodes combinatoires pour résoudre ce problème de dénombrement, méthodes qui sont spécifiques, comme l'est la traduction de ce problème en un dénombrement d'une famille de permutations. C'est pourquoi les applications des résultats combinatoires obtenus sont, dans cet article, peu nombreuses et succinctes, au profit des développements combinatoires et, en particulier, la mise en évidence du « bon objet » combinatoire pour l'étude des arbres binaires de recherche : les arbres binaires décroissants.

Dans le présent travail nous reprenons l'étude de l'ensemble des arbres binaires de recherche construits à partir de toutes les permutations d'un même ensemble de clés *distinctes*, en associant à chaque suite de clés, de façon *biunivoque*, un arbre binaire dit *décroissant* qui est le même arbre que l'arbre binaire de recherche construit à partir de la suite donnée sauf que l'étiquette de chaque sommet est le rang d'une clé dans la suite au lieu d'être la clé elle-même. Cette correspondance est décrite à la section suivante. La suite des étiquettes des sommets d'un arbre binaire décroissant pris en ordre symétrique, que nous appelons la *projection* de l'arbre, apparaît comme ayant des propriétés remarquables : les sommets de l'arbre ayant soit deux fils vides, soit un seul fils vide à gauche, soit un seul fils vide à droite, soit aucun fils vide, sont « projetés » respectivement sur les pics, montées, descentes, creux de la projection. L'énumération des arbres par la nature de leurs sommets est ainsi ramenée à un problème déjà résolu pour les permutations. La section 3 est consacrée à cet aspect.

Nous développons plus avant l'étude combinatoire des arbres binaires décroissants à la section 4 en leur associant une famille de forêts enracinées étiquetées; ceci permet, en particulier, de résoudre certains problèmes liés à la hauteur de la pile dans les algorithmes de parcours d'un arbre en ordre symétrique à l'aide d'une pile, problèmes traités à la dernière section.

Ensuite, à la section 5, nous étudions un groupe de transformations opérant sur les arbres binaires décroissants, ce qui permet de ramener tout problème d'insertion d'un nouveau sommet à celui de son insertion dans la position la plus à droite possible. Cette méthode permet de raffiner un résultat connu.

Les méthodes introduites et les résultats combinatoires obtenus permettent d'analyser de nombreux algorithmes relatifs aux arbres binaires de recherche. Comme il n'est pas possible dans le cadre de cet article de développer de telles analyses nous nous contenterons d'en détailler une seule à la dernière

section : celle relative à un algorithme de parcours d'un arbre binaire de recherche en ordre symétrique à l'aide d'une pile.

Signalons qu'une partie des résultats présentés ici a été publiée dans la série des séminaires de l'I.R.I.A. [8].

2. ARBRES BINAIRES DE RECHERCHE ET ARBRES BINAIRES DÉCROISSANTS

Dorénavant on désignera par n un entier positif, par $[n]$ l'ensemble des entiers de 1 à n, par S_n l'ensemble des permutations de $[n]$ et on considérera qu'un ensemble de n clés distinctes est $[n]$. Une suite de n clés distinctes sera alors considérée comme une permutation σ de S_n ou comme le mot $\sigma(1)\,\sigma(2)\ldots\sigma(n)$ sur l'alphabet $[n]$.

Un *arbre binaire* est soit vide (et alors on dit que le nombre de ses sommets est nul et que sa racine est vide), soit un triplet (g, r, d) où r est un sommet (dit aussi *sommet interne*) appelé la *racine* de l'arbre, g (respectivement d) est un arbre binaire appelé *sous-arbre à gauche* (respectivement *droite*) de r. Les racines de d et g sont dits *fils* de r. Un fils vide est encore appelé un *sommet externe*. Précisons que les sommets externes ne sont pas comptés dans le décompte des sommets d'un arbre.

Un arbre binaire de recherche correspondant à un mot $\sigma(1)\,\sigma(2)\ldots\sigma(n)$, $\sigma \in S_n$ est construit par récurrence : pour $i = 0$ l'arbre est vide; pour $i > 0$ supposons construit un arbre binaire de recherche à partir du mot $\sigma(1)\,\sigma(2)\ldots\sigma(i-1)$; on construit un arbre binaire de recherche pour le mot $\sigma(1)\,\sigma(2)\ldots\sigma(i)$ en « insérant » $\sigma(i)$ par l'algorithme d'insertion suivant (nous empruntons la façon de décrire un algorithme à [9]), ou v est la clé à insérer.

Algorithme P d'insertion

P 1 (Initialisation) prendre pour racine-en-cours la racine de l'arbre.

P 2 (Insertion) si la racine-en-cours est vide, insérer v à sa place (i. e. créer un sommet portant l'étiquette v et ayant ses deux fils vides), fin de l'algorithme.

P 3 (Comparaison) si v est supérieur à l'étiquette de la racine-en-cours aller en P 5.

P 4 (Aller à gauche) prendre pour nouvelle racine-en-cours le fils à gauche de la racine-en-cours, aller en P 2.

P 5 (Aller à droite) prendre pour nouvelle racine-en-cours le fils à droite de la racine-en-cours, aller en P 2.

La figure 1 montre l'arbre binaire de recherche à 9 sommets obtenu à partir du mot $\sigma = 5\,2\,6\,1\,9\,3\,8\,4\,7$. Les sommets internes sont représentés par des ronds et les sommets externes par des carrés; les étiquettes sont écrites à côté des sommets internes.

décembre 1976.

Rappelons que l'algorithme de recherche d'un article dans un arbre binaire de recherche est le même que l'algorithme d'insertion P aux modifications suivantes près : v est la clé de l'article cherché et il faut substituer à P 2 la séquence

P' 2 Si la racine-en-cours est vide alors v n'est pas dans l'arbre, fin de l'algorithme; si l'étiquette de la racine-en-cours est égale à v alors on a trouvé l'article cherché, fin de l'algorithme.

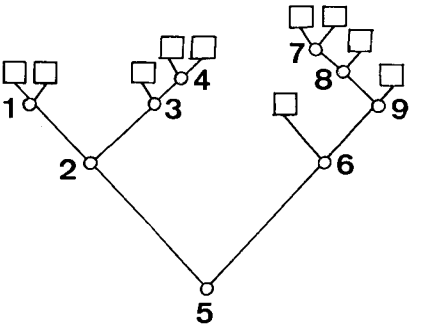

Figure 1.
Arbre binaire de recherche correspondant au mot 526193847.

Rappelons encore que parcourir un arbre binaire en ordre symétrique consiste à parcourir d'abord le sous-arbre de gauche en ordre symétrique, puis visiter la racine, puis parcourir le sous-arbre de droite en ordre symétrique (*cf.* [9]).

DÉFINITION : On appelle *projection* d'un arbre binaire étiqueté la suite des étiquettes des sommets prises dans l'ordre symétrique.

On peut aussi donner la définition plus formelle suivante : la projection d'un arbre vide est le mot vide; la projection d'un arbre binaire (g, r, d) est le mot $g'\, r'\, d'$ où g' (respectivement d') est la projection du sous-arbre g (respectivement d) et r' est l'étiquette de la racine r.

L'intérêt de la projection vient de la proposition suivante, bien connue et facile à vérifier :

PROPOSITION 2.1 : *La projection de tout arbre binaire de recherche construit à partir d'une permutation de S_n est la suite croissante* 1, 2, ..., n.

On sait aussi qu'un même arbre binaire de recherche peut être obtenu en général à partir de différentes permutations; par exemple l'arbre de la figure 1 peut être obtenu à partir du mot 5 2 1 3 4 6 9 8 7.

Nous allons maintenant introduire les outils essentiels de cette étude : les arbres binaires décroissants et leur correspondance avec les permutations.

Ces notions se trouvent sous une forme déguisée dans un travail de D. Foata et M.-P. Schützenberger [5]. Elle ont été introduites indépendamment par W. H. Burge [2].

DÉFINITION : On appelle *arbre binaire décroissant* un arbre binaire dont les sommets internes sont étiquetés de façon que l'étiquette d'un sommet soit inférieure à celle de chacun de ses fils, s'il a un fils non vide. On notera B_n l'ensemble des arbres binaires décroissants à n sommets étiquetés par 1, 2, ..., n.

De cette définition on déduit immédiatement que l'étiquette de la racine d'un arbre binaire décroissant est le minimum des étiquettes de l'arbre.

Soit σ une permutation de S_n, soit b_σ l'arbre binaire de recherche construit à partir de σ; les sommets internes de b_σ sont étiquetés par $\sigma(1)$, $\sigma(2)$, ..., $\sigma(n)$. Si pour tout $i \in [n]$ nous remplaçons l'étiquette $\sigma(i)$ par i, nous obtenons un arbre binaire d_σ qui est décroissant par construction. Notons que b_σ et d_σ ont « la même forme ».

Prenons maintenant la projection de d_σ; elle n'est autre que la permutation inverse σ^{-1}; en effet, si $p_1 p_2 \ldots p_n$ est la projection de d_σ, pour tout $i \in [n]$ la lettre $\sigma(p_i)$ est par construction la i-ième lettre de la projection de b_σ qui est égale à i par la proposition 2.1.

Si σ et τ sont deux permutations distinctes de S_n, alors on a $\sigma^{-1} \neq \tau^{-1}$, et alors les arbres binaires décroissants d_σ et d_τ sont distincts puisqu'ils ont des projections distinctes.

Rassemblons ces résultats dans la :

PROPOSITION 2.2 : *L'arbre binaire décroissant construit à partir d'une permutation σ est caractéristique de cette permutation et sa projection est la permutation inverse σ^{-1}.*

Exemple : Sur la figure 2 se trouve l'arbre binaire décroissant correspondant à la permutation $\sigma = 5\,2\,6\,1\,9\,3\,8\,4\,7$ dont l'arbre binaire de recherche est représenté sur la figure 1. On vérifiera que sa projection est la permutation $\sigma^{-1} = 4\,2\,6\,8\,1\,3\,9\,7\,5$.

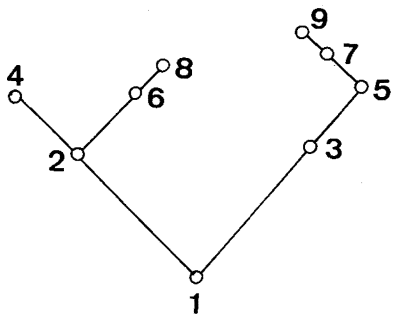

Figure 2.
Arbre binaire décroissant correspondant au mot 426813975.

décembre 1976.

Donnons maintenant la construction de l'arbre binaire décroissant dont la projection est une permutation donnée $\tau \in S_n$. Le mot $\tau = \tau(1)\,\tau(2) \ldots \tau(n)$ factorise toujours en $w\ r\ w'$ où w et w' sont des mots éventuellement vides et r est la plus petite lettre de τ; alors r est l'étiquette de la racine de l'arbre cherché et w (respectivement w') est la projection du sous-arbre à gauche (respectivement droite); on recommence la même construction sur w et w' s'ils ne sont pas vides.

Exemple : Avec $\tau = 4\,2\,6\,8\,1\,3\,9\,7\,5$ on a $r = 1$, $w = 4\,2\,6\,8$, $w' = 3\,9\,7\,5$; la même construction opérant sur w' donne une racine d'étiquette 3, un sous-arbre à gauche vide et un sous-arbre à droite de projection 9 7 5; etc. L'arbre cherché est celui de la figure 2.

Notons que L. J. Guibas [15] a récemment énoncé un principe d'indépendance qui se déduit facilement de la proposition précédente.

PROPOSITION 2.3. : *Étant donné un arbre binaire à n sommets internes, le nombre de permutations de S_n dont cet arbre est l'arbre binaire de recherche est égal à n! divisé par le produit du nombre de sommets de chaque sous-arbre.*

Preuve : Le nombre cherché est, d'après la proposition 2.2, le nombre de façons d'étiqueter un arbre binaire de façon à en faire un arbre binaire décroissant. La réponse est fournie par D. Knuth ([10] 5.1.4 exercice 20).

3. PROPRIÉTÉS DE LA PROJECTION

DÉFINITION : Soit σ une permutation de S_n; pour simplifier les définitions qui suivent posons par convention $\sigma(0) = \sigma(n+1) = 0$. Pour $i \in [n]$ on dira que

$\sigma(i)$ est un *pic* si $\sigma(i) > \sigma(i+1)$ et $\sigma(i) > \sigma(i-1)$;
$\sigma(i)$ est un *creux* si $\sigma(i) < \sigma(i+1)$ et $\sigma(i) < \sigma(i-1)$;
$\sigma(i)$ est une *montée* si $\sigma(i-1) < \sigma(i) < \sigma(i+1)$;
$\sigma(i)$ est une *descente* si $\sigma(i-1) > \sigma(i) > \sigma(i+1)$.

DÉFINITION : Soit v un sommet d'un arbre binaire; on dira que

v est *vide à droite* si son fils à droite est vide;
v est *vide à gauche* si son fils à gauche est vide;
v est une *feuille* si ses deux fils sont vides;
v est un *point double* si aucun de ses fils n'est vide;
v est un *point simple* si un seul de ses fils est vide.

Exemple : Dans la permutation 4 2 6 8 1 3 9 7 5 les pics sont 4, 8 et 9, les creux sont 1 et 2, les descentes sont 5 et 7, les montées sont 3 et 6; dans l'arbre binaire décroissant de la figure 2 les feuilles sont 4, 8 et 9, les points doubles sont 1 et 2, les points simples vides à droite sont 5 et 7, les points simples vides à gauche sont 3 et 6.

Il est facile de vérifier la :

PROPOSITION 3.1 : *La projection d'un arbre binaire décroissant envoie les feuilles sur les pics, les points doubles sur les creux, les points simples vides à gauche sur les montées et les points simples vides à droite sur les descentes.*

Il est clair que, dans une permutation, entre deux pics consécutifs il y a toujours un et un seul creux; il suit que si une permutation a k pics alors elle a $k-1$ creux. En corollaire de la proposition précédente, tout arbre binaire décroissant ayant k feuilles a $k-1$ points doubles, propriété bien connue des arbres binaires (*voir*, par exemple [9]).

Un autre corollaire de la proposition 3.1 est le suivant :

COROLLAIRE 3.2 : *Les arbres binaires décroissants à n sommets et sans points simples sont dénombrés par le nombre d'Euler (ou nombre tangent) A_n dont la série génératrice exponentielle est* $\sum_{n \geq 1} A_n (u_n/n!) = \operatorname{tg} u$.

Preuve : Un arbre binaire décroissant sans points simples a un nombre impair de sommets, puisque s'il a k feuilles il a $k-1$ points doubles donc $2k-1$ nœuds. Sa projection est une permutation sans montées ni descentes, dite encore permutation alternante; depuis un résultat célèbre de Désiré André [1] on sait que les permutations alternantes de S_{2k-1} ($k \geq 1$) sont dénombrées par le nombre tangent A_{2k-1}.

Le dénombrement des permutations par nombre de pics, creux, montées et descentes a été faite par D. Foata et V. Strehl ([7] où on trouvera les références aux travaux antérieurs sur cette question). Les preuves utilisent les propriétés d'un groupe de transformation de S_n dont on trouvera l'analogue à la section 5. De leurs résultats et de la proposition 3.1 on déduit immédiatement la distribution des points doubles, points simples, feuilles des arbres binaires décroissants ainsi que des propriétés remarquables de ces distributions.

DÉFINITION : On appelle *monotonie* d'une permutation $\sigma \in S_n$ tout plus grand facteur croissant du mot $\sigma(1)\,\sigma(2) \ldots \sigma(n)$.

Exemple : Dans $\sigma = 4\,2\,6\,8\,1\,3\,9\,7\,5$ il y a 5 monotonies, à savoir 4, 2 6 8, 1 3 9, 7, 5.

Il est bien connu que le nombre de permutations de S_n ayant k monotonies est le nombre eulérien $A_{n,k}$ (*voir*, par exemple [6] ou [10]). Or une monotonie se termine soit par un pic soit par une descente. Donc l'entier $A_{n,k}$ compte le nombre de permutations de S_n qui ont k pics et descentes. En associant à toute permutation $\sigma \in S_n$ la permutation σ' définie par $\sigma'(i) = \sigma(n+1-i)$ pour tout $i \in [n]$, on voit que $A_{n,k}$ compte aussi les permutations de S_n qui ont k pics et montées; d'après la relation entre nombre de pics et nombre de creux, $A_{n,k}$ compte aussi les permutations de S_n qui ont $k-1$ creux et

descentes ou $k-1$ creux et montées. D'après la proposition 3.1 et le caractère biunivoque de la projection on a le :

COROLLAIRE 3.3. *Le nombre d'arbres binaires décroissants à n sommets dont k sont vides à droite (respectivement à gauche) [ou $k-1$ sont non vides à droite (respectivement à gauche)] est le nombre eulérien $A_{n,k}$.*

Remarquons qu'on peut démontrer directement ce résultat par l'intermédiaire de la relation de récurrence des nombres eulériens

$$A_{n+1,k} = k\,A_{n,k} + (n+2-k)\,A_{n,k-1}$$

en ajoutant un sommet d'étiquette $n+1$ à un arbre de B_n ayant soit k soit $k-1$ sommets vides à droite pour obtenir un arbre de B_{n+1} ayant k sommets vides à droite.

4. ARBRES BINAIRES DÉCROISSANTS ET FONCTIONS DÉCROISSANTES

DÉFINITION : On appelle *fonction décroissante* une application de $[n]$ dans $[n]$ telle que pour tout $i \in [n]$ on ait $f(i) \leqq i$. On notera D_n l'ensemble des fonctions décroissantes sur $[n]$.

DÉFINITION : On appelle *graphe linéaire* associé à une application de $[n]$ dans $[n]$ un graphe orienté à n sommets étiquetés par les entiers 1, 2, ..., n et où un arc joint le sommet i au sommet j si et seulement si $f(i) = j$; sauf si $i = j$, on dira alors que le sommet i est *fils* du sommet j ou que le sommet j est *père* du sommet i.

PROPOSITION 4.1 : *Le graphe linéaire associé à une fonction décroissante est une forêt enracinée étiquetée telle que l'étiquette d'un sommet est supérieure à l'étiquette de son père et telle que les points fixes de la fonction sont associés aux racines de la forêt.*

Preuve : il suffit de montrer que le graphe associé à une fonction $f \in D_n$ est sans cycle (les boucles mises à part), le reste étant évident. Il faut montrer que pour tout $i \in [n]$, sauf si $f(i) = i$, il n'existe pas d'entier positif m tel que $f^m(i) = i$. En effet, si $f(i) \neq i$ alors $f(i) < i$, et, par itération, $f^m(i) < i$ quel que soit $m > 1$; donc il n'existe pas d'entier positif m tel que $f^m(i) = i$.

Exemple : Pour $n = 9$ définissons une fonction $f \in D_9$ par le tableau suivant :

i.....	1	2	3	4	5	6	7	8	9
$f(i)$.....	1	1	3	2	5	1	5	1	7

Le graphe linéaire de cette fonction est donné à la figure 3. Les sommets sont figurés par des ronds et les boucles ne sont pas figurées.

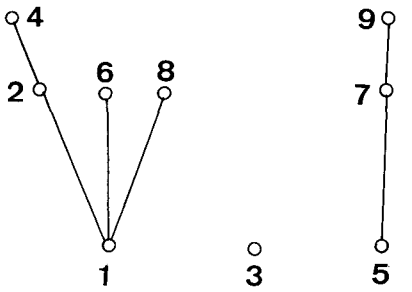

Figure 3.
Graphe linéaire associé à la fonction décroissante 1 1 3 2 5 1 5 1 7.

CONVENTION : Dans la suite on identifiera toute fonction de D_n avec son graphe associé.

REMARQUE : il est facile de vérifier que les fonctions f de D_n ayant un seul point fixe, qu'on appellera *arborescences décroissantes*, vérifient $f(1) = 1$ et $f(i) < i$ pour $i > 1$. Soit E_n l'ensemble des arborescences décroissantes sur $[n]$. Un argument simple montre que Card $E_n = (n-1)!$ et Card $D_n = n!$.

Considérons une forêt enracinée étiquetée et ordonnons de gauche à droite suivant l'ordre croissant des étiquettes, les racines et tous les fils d'un même père (ce qui est fait sur la figure 3). On obtient alors une forêt « ordonnée » (*voir* [9], p. 306), qui est transformée par la « correspondance naturelle » (*voir* [9], p. 333) en un arbre binaire. La proposition suivante est alors immédiate.

PROPOSITION 4.2 : *La correspondance naturelle est une bijection de D_n sur B_n.*

Cette bijection a aussi été mise en évidence par W. H. Burge [2].

Exemple : L'image du graphe de la figure 3 par la correspondance naturelle est l'arbre de la figure 2.

Nous allons maintenant donner quelques propriétés de la correspondance naturelle appliquée aux fonctions décroissantes.

DÉFINITION : Un entier $i \in [n]$ est dit une *feuille* d'une fonction $f \in D_n$ s'il n'existe pas $j \neq i$, $j \in [n]$, tel que $f(j) = i$.

Exemple : pour la fonction de la figure 3, les entiers (ou sommets) 3, 4, 6, 8, 9 sont les seules feuilles.

Nous laissons au lecteur la preuve de la :

PROPOSITION 4.3 : *Par la correspondance naturelle les feuilles d'une fonction décroissante sont envoyées sur les sommets sans fils à gauche de l'arbre binaire décroissant associé.*

décembre 1976.

En corollaire de cette proposition et du corollaire 3.3, les nombres eulériens dénombrent les fonctions décroissantes par nombre de feuilles.

DÉFINITION : On appelle *basse branche droite* d'un arbre binaire décroissant la suite $(i_1, i_2, ..., i_k)$, $k > 0$, telle que

(1) $i_1 = 1$;

(2) i_j pour $j = 2, 3, ..., k$ est l'étiquette du fils à droite du sommet d'étiquette i_{j-1};

(3) le fils à droite du sommet d'étiquette i_k est vide.

Exemple : La basse branche droite de l'arbre binaire décroissant de la figure 2 est la suite (1, 3, 5).

Nous laissons encore au lecteur la preuve de la :

PROPOSITION 4.4 : *La correspondance naturelle associe la suite croissante des points fixes d'une fonction décroissante à la basse branche droite de l'arbre binaire décroissant associé.*

COROLLAIRE 4.5 : *Le nombre d'arbres binaires décroissants à n noeuds dont la basse branche droite est* $(i_1, i_2, ..., i_k)$ est

$$\prod_{j \in [n] \setminus \{i_1, i_2, ..., i_k\}} (j-1).$$

Preuve : D'après la proposition 4.4 il suffit de dénombrer les fonctions de D_n dont les points fixes sont $i_1, i_2, ..., i_k$. Si f est une telle fonction, on a $f(i_1) = i_1, f(i_2) = i_2, ..., f(i_k) = i_k, f(j) < j$ si j n'appartient pas à l'ensemble $\{ i_1, i_2, ..., i_k \}$. Le nombre cherché est donc le nombre d'applications de $[n]$ dans $[n]$ telles que $f(j) \in [j-1]$ pour $j \notin \{ i_1, i_2, ..., i_k \}$ et $f(j) = j$ sinon. Le résultat est alors immédiat.

Remarque : Ce résultat est un raffinement du dénombrement suivant qui figure dans [10] : le nombre d'arbres binaires de recherche à n sommets dont k sur la basse branche droite est $s_{n,k}$ le nombre de Stirling de première espèce en valeur absolue. On aurait aussi pu effectuer le dénombrement des fonctions décroissantes à k points fixes, par exemple en faisant apparaître la relation de récurrence des nombres de Stirling de première espèce. Quoiqu'il en soit, on a ainsi une preuve combinatoire de l'identité classique (*voir*, par exemple [3]) :

$$s_{n,k} = \sum_{1 \leq j_1 < j_2 < ... < j_{n-k} < n} j_1 j_2 \cdots j_{n-k}.$$

Nous allons terminer cette section en dénombrant les fonctions décroissantes de hauteur donnée, dénombrement nécessaire au calcul de la hauteur de la pile dans certains algorithmes de parcours d'arbres à l'aide d'une pile.

DÉFINITION : On appelle *hauteur d'un sommet* i dans une forêt enracinée étiquetée un entier $h(i)$ ainsi défini :

$h(i) = 0$ si i est une racine;

$h(i) = 1+h(j)$ si i n'est pas une racine et j est père de i.

En d'autres termes, la hauteur de i pour $f \in D_n$ est $h(i) = m$ où m est le plus petit entier tel que $f^m(i) = f^{m+1}(i)$.

DÉFINITION : On appelle *hauteur* d'une fonction décroissante $f \in D_n$ l'entier $h(f) = \max_{i \in [n]} h(i)$ où $h(i)$ est la hauteur de i pour f.

On notera $D_n(h)$ [respectivement $E_n(h)$] l'ensemble des fonctions de D_n (respectivement E_n) de hauteur inférieure ou égale à h, entier donné. On notera $D_{n,k}(h)$ l'ensemble des fonctions de $D_n(h)$ ayant k points fixes [donc on a $E_n(h) = D_{n,1}(h)$]. On s'intéresse, en vue des applications à la dernière section, à dénombrer les fonctions de D_n de hauteur h donnée. Ce nombre est évidemment Card $D_n(h)$ − Card $D_n(h-1)$. Le calcul effectif de Card $D_n(h)$ par des relations de récurrence peut être obtenu par les relations de la proposition 4.5.

Notons en passant que Card $D_n(1)$ est le nombre d'involutions de S_n (la preuve est laissée au lecteur).

PROPOSITION 4.6 : *Si t et u sont des indéterminées qui commutent, on a, pour h entier positif donné, les identités*

$$(1) \qquad \sum_{n>0} \frac{u^n}{n!} \operatorname{Card} D_{n,k}(h) = \frac{1}{k!}\left(\sum_{n>0} \frac{u^n}{n!} \operatorname{Card} E_n(h)\right)^k \qquad k > 0,$$

$$(2) \qquad \operatorname{Card} E_n(h) = \operatorname{Card} D_{n-1}(h-1) \qquad n \geqq 1,$$

$$(3) \qquad 1 + \sum_{n>0} \frac{u^n}{n!} \sum_{0<k\leqq n} t^k \operatorname{Card} D_{n,k}(h) = \exp\left[t \sum_{n>0} \frac{u^n}{n!} \operatorname{Card} D_{n-1}(h-1)\right]$$

en particulier

$$1 + \sum_{n>0} \frac{u^n}{n!} \operatorname{Card} D_n(h) = \exp\left[\sum_{n>0} \frac{u^n}{n!} \operatorname{Card} D_{n-1}(h-1)\right],$$

$$(4) \qquad \text{conditions initiales : } \operatorname{Card} D_n(0) = 1, \qquad n \geqq 0.$$

Preuve : Tout d'abord les relations (3) sont conséquence directe des relations (1) et (2) et de l'égalité Card $D_n(h) = \sum_k$ Card $D_{n,k}(h)$. L'idée qui préside à la preuve de (1) est de considérer une fonction de $D_{n,k}(h)$ comme « composée » de k fonctions de $E_n(h)$. Nous renvoyons le lecteur à [4], [13], [14] où les mêmes techniques sont mises en jeu pour le dénombrement des forêts enracinées étiquetées de hauteur donnée. Quant à la relation (2), la

méthode est très classique (*voir*, par exemple [3], [4]) : à tout $f \in D_n(h)$ associons la fonction f' définie pour $i \in [n]$ par les relations

$$f'(i) = 0 \quad \text{si} \quad f(i) = i,$$
$$f'(i) = f(i) \quad \text{si} \quad f(i) \neq i,$$
$$f'(0) = 0.$$

On vérifie facilement que f' est une arborescence décroissante sur l'ensemble $\{0, 1, 2, ..., n\}$ et que la correspondance entre f et f' est biunivoque; de plus f' est de hauteur $h+1$ si f est de hauteur h.

Enfin les conditions initiales (4) résultent du fait que $D_n(0)$ ne contient que l'application identique.

REMARQUE : On pourrait, par des techniques analogues, dénombrer les fonctions décroissantes de hauteur totale donnée (la hauteur totale étant la somme des hauteurs de tous les sommets). Nous renvoyons le lecteur à l'ouvrage de D. Foata [4] où on trouvera les techniques nécessaires, des références à d'autres travaux antérieurs sur des problèmes voisins, ainsi que des indications sur d'autres dénombrements en rapport avec les hauteurs.

5. L'OPÉRATION D'ÉCHANGE DANS UN ARBRE BINAIRE DÉCROISSANT

DÉFINITION : Soit i l'étiquette d'un sommet d'un arbre binaire décroissant b. On dit qu'on effectue *un échange par rapport* à i sur b si on transpose les sous-arbres à droite et à gauche du sommet i dans b. On notera $\varepsilon_i(b)$ le nouvel arbre obtenu.

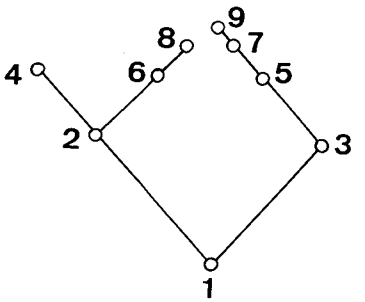

Figure 4.

Arbre binaire décroissant obtenu à partir de l'arbre de la figure 2 par échange par rapport au sommet d'étiquette 3.

Exemple : l'échange par rapport à $i = 3$ opérant sur l'arbre de la figure 2 donne l'arbre de la figure 4.

La proposition suivante, directement inspirée de [7], est laissée au lecteur.

PROPOSITION 5.1 : *Pour n donné, l'ensemble des échanges $\{ \varepsilon_i, i \in [n] \}$ forme un groupe commutatif d'involutions opérant sur B_n; de plus on a pour tout $i \in [n]$, pour tout $b \in B_n$ l'égalité $\varepsilon_i(b) = b$ si et seulement si i est feuille de b.*

On trouvera dans le travail de D. Foata et V. Strehl [7] une étude du groupe $\{ p \varepsilon_i p^{-1}, i \in [n] \}$ opérant sur S_n où p est la projection définie à la deuxième section.

Nous allons maintenant, à l'aide de l'opération d'échange, étudier le chemin allant de la racine au sommet d'étiquette n dans un arbre de B_n.

Soit $b \in B_n$, soit $(i_1, i_2, ..., i_k, n)$ la suite des étiquettes des sommets de l'unique chemin allant de la racine au sommet d'étiquette n dans b; on appelera cette suite le *chemin étiqueté* de n dans b. On a les relations

$$1 = i_1 < i_2 < ... < i_k < n.$$

En supposant $n > 1$, on a $k \geqq 1$. On posera par convention $i_{k+1} = n$.

On appelle *suite des directions* de n dans b la suite $(d_1, d_2, ..., d_k)$ où pour $j = 1, 2, ..., k$ d_j est égal à une variable D (respectivement G) si i_{j+1} est fils à droite (respectivement à gauche) de i_j dans le chemin étiqueté de n dans b.

Exemple : Reprenons l'arbre de la figure 2; ici n vaut 9; le chemin étiqueté de 9 dans cet arbre est (1, 3, 5, 7, 9) et la suite des directions est (D, D, G, G).

PROPOSITION 5.2 : *Le nombre d'arbres binaires décroissants à $n > 1$ sommets internes dont le chemin étiqueté de n est la suite $(i_1, i_1, ..., i_k, n)$ donnée $(k < n)$ et dont la suite des directions est donnée, est égal à*

$$\prod_{j \in [n-1] \setminus \{i_1, i_2, ..., i_k\}} (j-1)$$

Preuve : Montrons d'abord que deux ensembles d'arbres de B_n ayant même chemin étiqueté de n, $(i_1, i_2, ..., i_k, n)$, mais dont les suites des directions de n sont distinctes, ont même cardinal. Soient Δ (respectivement Δ') ces deux ensembles ayant pour suite des directions de n $d = (d_1, d_2, ..., d_k)$ [respectivement $d' = (d'_1, d'_2, ..., d'_k)$]. Soient $j_1, j_2, ..., j_r$ $(1 \leqq r \leqq k)$ les seuls indices où les éléments de d et d' diffèrent (i. e. $d_i \neq d_i$ si $i \in \{ j_1, j_2, ..., j_r \}$ et $d_i = d'_i$ sinon). Posons $m_1 = i_{j_1}, m_2 = i_{j_2}, ..., m_r = i_{j_r}$ les éléments du chemin étiqueté de n où les directions diffèrent. Alors il est facile de vérifier qu'en composant (dans un ordre quelconque) les échanges $\varepsilon_{m_1}, \varepsilon_{m_2}, ..., \varepsilon_{m_r}$ on obtient une bijection entre Δ et Δ'.

Pour terminer la preuve il suffit de dénombrer les arbres de B_n ayant pour chemin étiqueté de n la suite donnée $(i_1, i_2, ..., i_k, n)$ et tels que la suite des directions de n n'a aucune occurrence de la variable G; cet ensemble est celui des arbres de B_n dont la basse branche droite est $(i_1, i_2, ..., i_k, n)$ et dont le dénombrement est immédiat à partir du corollaire 4.5.

décembre 1976.

COROLLAIRE 5.3 : *Le nombre d'arbres binaires décroissants à $n > 1$ sommets internes tels que la suite des directions de n est donnée et de longueur k,* est *égal à $s_{n-1,k}$, nombre de Stirling de première espèce non signé.*

Preuve : Il suffit d'utiliser la remarque qui suit le corollaire 4.5.

COROLLAIRE 5.4 : *Le nombre d'arbres binaires décroissants à $n > 1$ sommets internes tels que le sommet d'étiquette n est de hauteur k, est égal à $2^k s_{n-1,k}$.*

Preuve : Dire que le sommet n est à la hauteur k dans un arbre $b \in B_n$ est équivalent à dire que la suite des directions de n dans b est de longueur k. Il suffit alors de sommer le résultat du corollaire précédent sur l'ensemble des suites de directions de longueur k qui est de cardinal 2^k.

Ce dernier résultat est dû à W. C. Lynch [11] (*voir*, aussi [10]) et sert à analyser le comportement de l'algorithme d'insertion et de l'algorithme de recherche dans un arbre binaire de recherche (*cf.* section 2) du point de vue du nombre de comparaisons effectuées.

Application : Nous terminons cette section par un dénombrement relatif à l'algorithme d'insertion donné à la section 2 : le nombre de permutations σ de S_n pour lesquelles l'insertion de $\sigma(n)$ dans l'arbre binaire de recherche formé à partir de $\sigma(1)\,\sigma(2) \ldots \sigma(n-1)$ requiert i passages en P 4 et $k-$ passages en P 5 ($0 \leqq i \leqq k \leqq n-1$) est $\binom{k}{i} s_{n-1,k}$. En effet, dans l'arbre binaire décroissant correspondant à σ, le nœud n est à la hauteur k et dans la suite des directions de n la variable G a exactement i occurrences. Le dénombrement cherché est alors conséquence immédiate du corollaire 5.3.

6. APPLICATION

Certains résultats présentés ci-dessus vont être utilisés dans l'étude détaillée d'un algorithme de parcours d'un arbre binaire de recherche en ordre symétrique dont le principe a été rappelé à la section 2. Cet algorithme sert en particulier à trier une suite de clés après avoir construit l'arbre binaire de recherche correspondant. Décrivons cet algorithme dans la forme empruntée à D. Knuth [9].

On note Q le sommet courant, Λ tout sommet vide, LLINK (Q) [respectivement RLINK (Q)] le fils à gauche (respectivement à droite) de Q. On dispose aussi d'une pile, initialement vide.

Algorithme S

S 1 (Initialisation) affecter à Q la racine de l'arbre :

S 2 Si LLINK (Q) = Λ, aller en S 4.

S 3 Mettre Q sur la pile, (aller à gauche) affecter LLINK (Q) à Q, aller en S 2.

S 4 Visiter Q.

S 5 Si RLINK $(Q) = \Lambda$, aller en S 7.

S 6 (Aller à droite) affecter RLINK (Q) à Q, aller en S 2.

S 7 Si la pile est vide, l'algorithme est terminé.

S 8 Affecter à Q le sommet de la pile, aller en S 4.

Remarquons qu'il existe une variante, plus simple, de cet algorithme, mais où on met systématiquement tout sommet sur la pile, ce qui n'est guère économique (*voir*, par exemple [9]).

L'application des lois de Kirchhoff à l'algorithme S montre que, mis à part le nombre n de sommets de l'arbre parcouru par S, il n'y a qu'un seul paramètre qui est le nombre k de fois où on passe en S 3; en effet, on passe une fois en S 1, n fois en S 2, S 4 et S 5, k fois en S 3 et S 8, $k+1$ fois en S 7 et $n-k-1$ fois en S 6.

Pour un arbre binaire de recherche donné, k est le nombre de sommets dont le fils à gauche n'est pas vide. D'après le corollaire 3.3, le nombre d'arbres binaires de recherche à n sommets qui ont k sommets dont le fils à gauche n'est pas vide est le nombre eulérien $A_{n,k+1}$. Nous renvoyons à l'ouvrage de D. Knuth [10] pour le calcul de la moyenne et de la variance de cette distribution.

Nous nous intéressons maintenant à la distribution de la hauteur maximale de la pile dans l'algorithme S. Plus précisément, pour un arbre binaire de recherche donné b, parcouru par S, la pile atteint une hauteur maximale h_b. Le problème est de compter combien il y a d'arbres binaires de recherche b à n sommets, parmi les $n!$ possibles, tels que $h_b = h$, h entier donné. La réponse va être donnée par les résultats de la section 4. Ces mêmes résultats ou les techniques introduites à la section 4 permettent de répondre à d'autres problèmes du même type, par exemple : distribution de la hauteur moyenne de la pile, distribution des hauteurs de pile excédant un nombre donné, ...

La propriété suivante donne la clé du problème : si b est un arbre binaire de recherche, h_b la hauteur maximale de la pile lorsque l'algorithme S est appliqué à b, si b' est la fonction décroissante associée à b par la correspondance naturelle (*voir* section 4), alors h_b est la hauteur de b'. Cette propriété est une conséquence directe du résultat suivant : pour tout sommet x d'un arbre binaire décroissant b, la hauteur $h_b(x)$ de x sur la pile, quand S est appliqué à b (par convention $h_b(x) = 0$ si x n'est jamais sur la pile) est la hauteur $h'_{b'}(x')$ du sommet x' correspondant à x dans la fonction décroissante b' correspondant à b. En effet, si x est fils à gauche de y dans b, alors $h_b(x) = 1 + h_b(y)$; mais alors x' est fils de y' dans b', donc $h'_{b'}(x') = 1 + h'_{b'}(y')$. Si x est fils à droite de y dans b alors $h_b(x) = h_b(y)$; mais alors soit x' et y' sont deux points fixes de b', soit x' et y' ont même père dans b', donc $h'_{b'}(x') = h'_{b'}(y')$. Dans tous les cas l'égalité $h_b(y) = h'_{b'}(y')$ entraîne $h_b(x) = h'_{b'}(x')$. La propriété est démontrée par récurrence en remarquant que la hauteur de la racine

de b dans la pile de S est nulle ainsi que la hauteur de tout point fixe de b', et que la racine de b correspond au plus petit point fixe de b',

D'après la section 4, le nombre d'arbres binaires de recherche à n sommets et de hauteur h est Card $D_n(h)$ − Card $D_n(h-1)$ où Card $D_n(h)$ vérifie les relations de la proposition 4.6, Il serait intéressant d'avoir une expression simple de la moyenne par rapport à h de cette variable et son comportement asymptotique en n,

REMERCIEMENTS

L'auteur doit ce travail à l'enseignement stimulant du Professeur D. Foata. Il tient aussi à remercier L. Hyafil, d'avoir attiré son attention sur le travail de W. H. Burge.

BIBLIOGRAPHIE

1. D. ANDRÉ, *Sur les permutations alternées*, J. Math. Pures Appl., vol. 7, 1881, p. 167-184.
2. W. H. BURGE, *An Analysis of a Tree Sorting Method and Some Properties of a Set of Trees*, First U.S.A.-Japan Computer Conference, 1972.
3. L. COMTET, *Analyse Combinatoire*, Paris, P.U.F., 1970.
4. D. FOATA, *La série génératrice exponentielle dans les problèmes d'énumération*, Les Presses de l'Université de Montréal, Montréal, 1974.
5. D. FOATA et M.-P. SCHÜTZENBERGER, *Polynômes de Kempner* (à paraître).
6. D. FOATA et M.-P. SCHÜTZENBERGER, *Théorie des polynômes eulériens*. Lectures Notes in Math., n° 138, Berlin, Springer-Verlag, 1970.
7. D. FOATA et V. STREHL, *Euler Numbers and Variations of Permutations*, Atti del Colloquio sulle teorie combinatorie, Roma, Accademia dei Lincei, septembre 1973 (à paraître).
8. J. FRANÇON, *Séminaires de l'I.R.I.A.* (Institut de Recherches en Informatique et Automatique, Rocquencourt, France), 1974.
9. D. E. KNUTH, *The Art of Computer Programming*, vol. 1, Reading, Massachusetts, Addison-Wesley, 1969.
10. D. E. KNUTH, *The Art of Computer Programming*, vol. 3, Reading, Massachusetts, Addison-Wesley, 1973.
11. W. C. LYNCH, *More Combinatorial Properties of Certain Trees*, Computer J., vol. 7, 1965, p. 299-302.
12. E. M. PALMER, M. A. RAHIMI et R. W. ROBINSON, *Efficiency of a Binary Comparison Storage Technique*, Journal of the A.C.M., vol. 21, 1974, p. 376-384.
13. A. RÉNYI et G. SZEKERES, *On the Height of Trees*, J. Austral. Math. Soc., vol. 7, 1967, p. 497-507.
14. J. RIORDAN, *The Enumeration of Trees by Height and Diameter*, I.B.M. J. Research and Development, vol. 4, 1960, p. 473-478.
15. L. J. GUIBAS, *A Principle of Independence for Binary Search Trees*, Acta Informatica, vol. 4, 1975, p. 293-298.