N° d'ordre : 2317

THÈSE

PRÉSENTÉE À

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par Cedric CHAUVE

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Structures arborescences : problèmes algorithmiques et combinatoires

---

Soutenue le : 11 décembre 2000

Après avis de : MM. Dominique Gouyou-Beauchamps Rapporteurs
Mireille Régnier ....
Volker Strehl ....

Devant la Commission d'examen formée de :

MM.	Robert Cori	Professeur	Président
	Mireille Bousquet-Mélou	Chargée de recherche au CNRS ..	Rapporteur
	Maxime Crochemore	Professeur	Examinateur
	Serge Dulucq	Professeur	Directeur
	Dominique Gouyou-Beauchamps	Professeur	Examinateur
	Mireille Régnier	Directrice de recherche à l'INRIA	Examinatrice
	Volker Strehl	Professeur	Examinateur

– 2000 –

Au cours de ces trois années de thèse, j'ai croisé de nombreuses personnes que je me suis promis de remercier... que les absents de la (pourtant longue) liste qui suit veuillent bien me pardonner.

Le professeur Volker Strehl a assisté à mes premiers pas balbutiants hors du doux cocon du LaBRI, lors d'un séminaire lotharingien de combinatoire en Allemagne, et sa gentillesse fut d'un grand secours. Je suis donc particulièrement sensible au fait qu'il ait accepté le lourd travail de rapporter cette thèse et d'assister à cette soutenance.

À la sortie de mon DEA, j'ai passé de longues semaines à sécher sur les tableaux de Young de hauteur bornée. Une des motivations de cette tentative résidait dans les beaux articles de Dominique Gouyou-Beauchamps sur ce sujet (qui ont donc accompagné mes premiers échecs). Je suis très heureux de le compter parmi mes rapporteurs et dans mon jury.

Je remercie Mireille Régnier d'avoir apporté le point de vue de l'algorithmicienne sur ce travail.

Le livre de Maxime Crochemore sur l'algorithmique des mots fut mon livre de chevet durant ma première année de thèse. Je suis donc très heureux qu'il ait accepté de participer à ce jury.

Il est à la fois étrange et agréable de retrouver dans son jury ses anciens enseignants (de "quand on était petit"). Ma passion pour l'algorithmique remontant au cours de licence de Robert Cori, je suis particulièrement heureux de sa participation à ce jury. J'ajoute que ses questions et commentaires lors de mes divers exposés en groupe de travail ont sensiblement contribué au développement de cette thèse.

Il y aurait vraiment trop à dire pour exprimer la joie que j'éprouve à compter Mireille Bousquet-Mélou (Mimi) dans mon jury, tant pour des raisons scientifiques (je ne cite que les constellations) qu'extra-scientifiques (vélocipédiques notamment). Je m'arrête donc là avant de noircir une page entière.

Enfin, mon "chef", Serge. Pour exprimer toute ma gratitude à son égard, je dirais qu'en me retournant sur ces trois années écoulées, je m'aperçois qu'il m'a permis de m'épanouir scientifiquement, encadrant de réunions hebdomadaires mes débuts hésitants avant de me laisser progressivement voler de mes propres ailes vers d'autres horizons (à la fois coloriés et québécois), sans toutefois jamais me perdre de vue. J'ajoute que sa relecture attentive (doux euphémisme) de ce mémoire est pour beaucoup dans le fait qu'il soit lisible (je l'espère).

En parlant de chef, je ne peux m'empêcher de dire ici toute l'admiration que je porte à Pierre Leroux, qui m'a gentiment accueilli au LaCIM et m'a guidé dans le monde fascinant de la théorie des espèces. Je m'estime privilégié d'avoir pu travailler avec lui et son compère Gilbert Labelle.

Continuons en mélangeant joyeusement les nombreux combinatoriciens que j'ai régulièrement croisés que ce soit au LaBRI ou au LaCIM : Sacha (dont la gentillesse semble n'avoir d'égal que l'immensité de ses connaissances), Bétréma (Canaris 0 - Girondins 5), Srećko Brlek, François Bergeron, Olivier Guibert, Jean-Guy Penaud, Xavier Viennot, ...

L'un des aspects les plus agréables du petit monde de la combinatoire est que chaque conférence est l'occasion de retrouver, en général autour d'une bonne bière, un groupe de jeunes camarades : Gilles (en concaténant les nombreuses questions que je lui ai posées on doit pouvoir couvrir la distance séparant Bordeaux de Nancy), Dominique (compagne d'infortune lors du séminaire lotharingien sus-cité ... ça crée des liens), Sylvie (que je n'aurais jamais cru remercier après notre première rencontre), Mike (à qui je dois notamment la découverte du Rocky Horror Picture Show), Étienne, Luigi, Marni (qui m'a presque persuadé que le tofu peut s'apprêter), Michel (dont l'appartement à Montréal fut un nid douillet durant 4 mois) et les nombreux étudiants du LaCIM. Je ne peux oublier de mentionner les amis de Michel : Francis, Michel2 et Nathalie.

Si j'ai beaucoup apprécié mon temps au LaBRI, les gens que j'y ai croisés y sont sans doute pour beaucoup. Je remercie donc en vrac TiNico et GrandNico (qui m'ont initié au culte du dieu Tipunch), Driss (dont la recette de tajine fait encore partie de mes préparations favorites), Akka (dangereux compagnon de voyage), Sami, Laurent, Guillaume, Anne (douce Anne), Olivier (qui finira bien par trouver un vélo), JC, Christophe, Jaco, Isa... Une mention spéciale à Pierre (pour Cortazar, Coltrane, Cassavetes, et bien d'autres choses), Hélène (qui a fini par céder aux sirènes du grand capital), Duduche (dangereux probabiliste) et Stéphane (à qui je dois la recette de ma désormais célébrissime tarte au citron). Pour clore le chapitre LaBRI, je ne peux qu'évoquer les amis qui ont partagé mes divers bureaux (les bars bordelais fréquentés en commun en faisant partie) : Olive (un jour on finira bien par ne plus être dans le même bureau et tu pourras fumer), Jeff (maintenant indissociable de la délicieuse Élisa), Manu (le squatter du jeudi) et Andrew (sur qui j'aurais trop de choses à dire pour le peu de place qui me reste).

Cette thèse fut aussi l'occasion de découvrir le beau métier d'enseignant ... que les collègues et étudiants qui ont accompagné cette découverte soient remerciés à la fois pour leur indulgence face à mes balbutiements et pour le plaisir que j'ai eu à travailler avec eux.

Enfin, pour passer certains caps difficiles au cours de ces dernières années, j'ai toujours su pouvoir compter sur le soutien indéfectible des mes amis au long cours (Henri, Yannis, Lionel, Xavier, Sandrine1, Sandrine2, Christine, Thierry), de mes parents et de mon globetrotter de frère.

Table des matières

Introduction

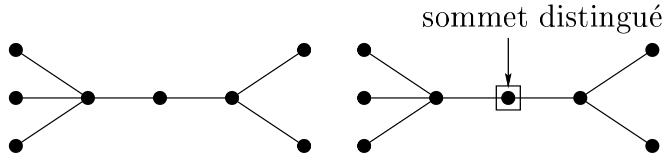
Les travaux que nous présentons dans ce mémoire se situent dans deux domaines connexes de l'informatique, la combinatoire, et plus précisément la combinatoire énumérative, et l'algorithmique. Nous nous sommes intéressés à divers problèmes ayant en commun le fait de manipuler des *structures arborescentes*.

Si la notion de structure arborescente n'a été formalisée, en termes mathématiques, que tardivement (au XIX^{ème} siècle), la présentation arborescente d'un ensemble de données est une pratique bien antérieure. On peut par exemple citer les arbres généalogiques, mais aussi la représentation de vertus par des arbres de la sagesse dans certains livres médiévaux, ou encore les arbres de classification d'espèces animales des naturalistes des XVII^{ème} et XVIII^{ème} siècles.

La première formalisation mathématique du concept d'arbre est généralement attribuée à Kirchhoff et est issue de travaux portant sur l'étude des circuits électriques.

Un *arbre* est un graphe connexe acyclique.
Une *arborescence* est un arbre ayant un sommet distingué[1].

La figure suivante représente un arbre (à gauche) et une arborescence (à droite).



C'est cependant Cayley qui a introduit le terme d'arbre, dans une série d'articles (le premier datant de 1857), portant principalement sur l'énumération d'arbres et d'arborescences en relation avec l'étude de formules algébriques [30]. À la suite de ces premiers travaux de Cayley, l'étude des propriétés des structures arborescentes s'est rapidement imposée comme un domaine classique et fécond de la combinatoire, notamment de la combinatoire énumérative (on peut se reporter aux ouvrages de Moon [120], Harary et Palmer [88, chapitre 3], Stanley [148, chapitre 5], Bergeron, Labelle et Leroux [16]).

Cet intérêt pour les structures arborescentes a ensuite connu un nouvel essor avec le développement de l'informatique. Selon Knuth, les arborescences sont en effet "les plus importantes structures non linéaires apparaissant en algorithmique" [102, page 308]. De nombreux algorithmes manipulant des arborescences (notamment cette structure de donnée

1. On peut noter que cette terminologie, issue des travaux de théorie des graphes (voir par exemple le livre de Berge [14]), est différente de la terminologie utilisée couramment dans les travaux traitant d'algorithmique des structures arborescentes, où on appelle arbre (resp. arbre non enraciné) ce que nous appelons arborescence (resp. arbre).

fondamentale qu'est l'arborescence binaire de recherche), l'analyse des performances en temps de ces algorithmes (en moyenne ou dans le cas le pire) nécessite souvent une connaissance fine de propriétés combinatoires de ces structures (voir les ouvrages de Knuth [102, section 2.3] et Sedgewick et Flajolet [139, chapitre 5]).

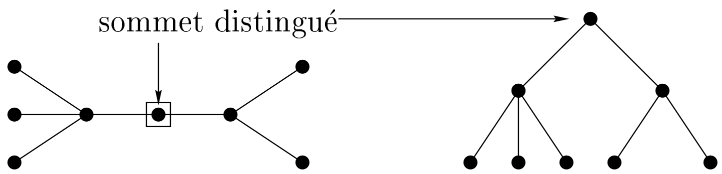
Finalement, on peut noter que de nombreux problèmes (mathématiques, biologiques, etc) pouvant être exprimés (ou modélisés) en termes d'arborescences, la question de la résolution algorithmique de ces problèmes implique la mise au point d'algorithmes efficaces manipulant des structures arborescentes. On peut citer par exemple le problème de la reconstruction phylogénétique source de nombreux travaux algorithmiques récents manipulant des arborescences phylogénétiques (voir l'ouvrage de Gusfield [86, chapitre 17] ou la thèse de Berry [19]).

Notre travail se situe dans ces deux domaines, à savoir l'étude des propriétés combinatoires de structures arborescentes et la mise au point d'algorithmes manipulant de telles structures. Bien que ces deux types problèmes aient comme point commun de manipuler des structures arborescentes, les problématiques, tout comme les méthodes employées, sont de natures différentes, ce qui nous a conduit à séparer ce mémoire en deux parties, une première partie consacrée aux problèmes combinatoires et une seconde partie consacrée aux problèmes algorithmiques.

Terminologie et conventions. Nous reprenons la terminologie classique composée de termes provenant essentiellement du vocabulaire de la généalogie et de la botanique.

Soit A une arborescence. On appelle *racine* de A son sommet distingué. Soient x et y deux sommets de A. Si le chemin allant de la racine à y passe par x, on dit que x est un *ancêtre* de y et que y est un *descendant* de x. Si de plus x et y sont reliés par une arête, on dit alors que x est le père de y et que y est un fils de x. Si deux sommets ont le même père, on dit qu'ils sont *frères*. On appelle *degré* d'un sommet x de A le nombre de fils de x. On dit que x est une *feuille* si son degré est nul, et que x est un *nœud* sinon.

Il existe de nombreuses façons de représenter une structure arborescente, mais dans ce mémoire, nous convenons, à l'instar de ce qui est le plus courant dans la littérature, de dessiner une arborescence en plaçant le sommet distingué en haut, toutes les arêtes s'éloignant de ce sommet (dessin de droite de la figure ci-dessous).



Énumération de structures arborescentes. La première partie de ce mémoire est consacrée à l'énumération de diverses familles de structures arborescentes, en général selon le nombre de sommets. La plupart des articles initiaux de Cayley traitant de structures arborescentes étaient consacrés à leur énumération. Dans un article célèbre [29], ce dernier s'est notamment intéressé aux arborescences étiquetées (chacun des n sommets est muni d'un étiquette entière appartenant à $[n]$, deux sommets ne pouvant avoir la même étiquette) libres (seules les relations d'adjacence entre sommets sont prises en compte) et a montré le résultat suivant[2] :

le nombre d'arborescences libres à n sommets et de racine 1 est n^{n-2}, le corollaire suivant en découlant immédiatement :

le nombre d'arborescences libres à n sommets est n^{n-1}.

Ces arborescences, qui sont parfois appelées arborescences de Cayley, plutôt que libres, sont un objet central apparaissant dans de nombreux problèmes combinatoires, comme par exemple les factorisations minimales de permutations circulaires en produit de transposition [53], l'étude des fonctions de parking ou, plus récemment, le dénombrement des régions de la famille d'arrangements d'hyperplans défini par Shi [147, 124].

Cette partie débute par un chapitre d'introduction, dans lequel nous fixons des notations et définitions que nous utilisons dans les chapitres ultérieurs. Puis, toujours dans ce chapitre introductif, nous présentons deux preuves du résultat de Cayley utilisant des outils que nous reprenons dans les chapitres suivants : le codage de Prüfer, qui illustre l'approche bijective et le codage des arborescences par des mots, et la formule d'inversion de Lagrange, qui illustre l'approche formelle et l'utilisation de séries génératrices.

Le chapitre suivant est consacré à l'étude combinatoire de la famille $\mathcal{A}_{n,k}$ des arborescences de Cayley à n sommets telles que la racine possède exactement k fils qui lui sont inférieurs. Nous montrons notamment de manière bijective le résultat suivant (théorème 16).

$$\text{Le nombre d'arborescences de } \mathcal{A}_{n+1,k} \text{ est } \binom{n}{k} n^{n-k}.$$

Dans le cas $k = 0$, ce résultat implique que le nombre d'arborescences de Cayley à $n + 1$ sommets telles que la racine est inférieure à tous ses fils est n^n. Nous poursuivons par l'étude de divers paramètres classiques sur les structures arborescentes (l'étiquette de la racine, le degré de la racine ou encore le nombre d'arêtes décroissantes) pour la famille $\mathcal{A}_{n,k}$, qui permet de donner des explications combinatoires pour plusieurs identités faisant intervenir n^n, nouvelles pour la plupart à notre connaissance. Nous proposons ensuite une nouvelle description de la bijection due à Moszkowski [122] prouvant que le nombre de factorisations de la permutation cyclique $(1,2,\dots,n)$ en produit de $n-1$ transpositions est n^{n-2}, description qui nous permet de raffiner ce résultat et de relier un sous-ensemble de ces factorisations à $\mathcal{A}_{n,0}$. Finalement, nous concluons ce chapitre en proposant un lemme général pour l'énumération d'arborescences telles que la racine est inférieure à ses fils, que nous appliquons aux arborescences ordonnées, planes et cycliques. Les résultats de ce chapitre ont été obtenus en collaboration avec Serge Dulucq et Olivier Guibert [36].

Dans le chapitre 3, nous poursuivons l'étude combinatoire de la famille d'arborescences $\mathcal{A}_{n,0}$. Nous nous intéressons au polynôme énumérateur des inversions de cette famille (une *inversion* dans une arborescence est un couple (x,y) de sommets tels que x est un ancêtre de y et $x > y$). Dans le cas des arborescences de Cayley de racine 1, ce polynôme a été étudié par Kreweras [106], qui démontre une formule de récurrence qu'il satisfait (permettant par exemple de calculer le nombre d'arborescences ayant un nombre fixé d'inversions) et relie son évaluation en 0, 1, 2 et -1 à des objets combinatoires classiques (graphes et nombres d'Euler). Nous proposons une généralisation des résultats de Kreweras pour la famille des arborescences de $\mathcal{A}_{n,0}$ de racine r fixée (en posant $r = 1$ dans nos résultats, nous obtenons ceux de Kreweras). Les résultats de ce chapitre ont été obtenus en collaboration avec Philippe Duchon et Serge Dulucq [34].

Le troisième chapitre est consacré à l'énumération d'arborescences alternantes. Une arborescence est alternante si toute suite $x_1,x_2,x_3\dots$ de sommets successivement reliés

2. Bien que ce résultat soit en général attribué à Cayley, il avait été auparavant découvert par Sylvester en 1857.

par une arête dans cette arborescence vérifie $x_1 < x_2 > x_3 < \cdots$ ou $x_1 > x_2 < x_3 > \cdots$. Les arborescences de Cayley alternantes ont été introduites par Postnikov [124, 125] en relation avec la famille d'arrangements d'hyperplans de Linial. Dans ce chapitre, nous donnons une preuve bijective de la formule dénombrant le nombre d'arborescences de Cayley alternantes, répondant ainsi à une question de Postnikov [125]. Nous poursuivons en montrant que le nombre d'arborescences ordonnées alternantes à $n+1$ sommets est $2n^n$. Nous donnons deux preuves de ce résultat : une preuve formelle basée sur la composition de séries génératrices et la formule d'inversion de Lagrange, suivie d'une preuve basée sur une bijection avec les arborescences de $\mathcal{A}_{n,0}$. Les résultats de ce chapitre ont été obtenus en collaboration avec Serge Dulucq et Andrew Rechnitzer [37].

Les deux derniers chapitres de cette partie sont centrés autour de l'extension aux séries multivariées de la formule d'inversion de séries formelles de Lagrange, particulièrement adaptée à l'énumération de structures arborescentes coloriées (chaque sommet est muni, en plus de son étiquette, d'une couleur choisie parmi un ensemble de couleurs fixé). La formule de Lagrange est un outil particulièrement utile pour traiter de nombreux problèmes d'énumération de structures arborescentes caractérisées par des séries génératrices à une variable [16, section 3.1]. L'énumération de structures arborescentes coloriées faisant appel à des séries génératrices multivariées, Good [72] a proposé en 1960 une extension de la formule de Lagrange permettant de prendre en compte le cas de séries multivariées. Parmi les différentes formes équivalentes de la formule de Good (voir [16, section 3.2]), il en est une, introduite indépendamment par Goulden et Kulkarni [80] et Bender et Richmond [11], qui est particulièrement intéressante car elle ne fait pas intervenir de calcul de déterminant (contrairement à toutes les autres formes) et s'interprète naturellement en termes d'objets combinatoires classiques (arborescences coloriées et endofonctions coloriées). Goulden et Kulkarni utilisent cette propriété pour donner une preuve bijective de cette formule. Dans les chapitres 5 et 6 nous proposons une approche bijective de cette formule, approche qui nous permet de clarifier et d'expliquer combinatoirement certains résultats obtenus par application des formes classiques de la formule de Good, et de déduire de ces explications des algorithmes de génération aléatoire uniforme pour les structures considérées. Ces deux aspects représentent, à notre avis, un point de vue intéressant sur la formule de Good et l'énumération de structures arborescentes coloriées.

Dans le chapitre 5 nous présentons différentes formes de la formule de Good, dont la forme due à Goulden-Kulkarni et Bender-Richmond, que nous appelons forme arborescente car basée sur la notion de dérivée d'un vecteur de séries formelles multivariées relativement à une arborescence. Nous proposons ensuite une nouvelle preuve bijective de cette formule, plus simple que la seule preuve bijective connue (due à Goulden et Kulkarni [80]), basée sur une variante de la preuve de la formule de Lagrange pour les séries à une variable due à G. Labelle [108]. Comme la preuve de Goulden et Kulkarni, notre preuve repose sur une bijection entre arborescences coloriées et endofonctions coloriées. Nous illustrons ensuite un premier intérêt d'une telle preuve en proposant des explications combinatoires pour plusieurs formules dénombrant des arborescences coloriées dont la partition des arêtes est fixée. Si la plupart de ces formules étaient déjà présentes dans la littérature, les preuves que nous en donnons sont, à notre connaissance, les premières à être purement combinatoires. Nous poursuivons en expliquant deux formules dénombrant des cactus planaires, une famille de graphes coloriés reliés aux factorisations minimales de permutations cycliques en produit de permutations [22, 21] et à la classification topologique des polynômes complexes [87, 58]. Nous déduisons de ces preuves des algorithmes de génération aléatoire uniformes de cactus planaires selon le nombre de sommets de chaque couleur ou selon la partition des degrés. Les résultats de ce chapitre ont été obtenus en collaboration avec Michel Bousquet, Gilbert Labelle, Pierre Leroux et Gilles Schaeffer [23, 24].

Le dernier chapitre de cette partie est consacré à l'énumération et à la génération aléatoire de constellations planaires. Une m-constellation planaire est une carte planaire obtenue en collant des polygones à m sommets de sorte que pour chaque face de la carte ainsi obtenue, le nombre d'arêtes la bordant soit un multiple de m. Ces objets sont en relation avec les factorisations minimales de permutations en produits de permutations et ont été notamment étudiés par N. Hanusse [87], et M. Bousquet-Mélou et G. Schaeffer [135, 25]. Ces derniers montrent notamment, à l'aide de la conjugaison d'arborescence, technique introduite dans [135], que les m-constellations sont en bijection avec une famille particulière d'arborescences : les arborescences eulériennes. Nous montrons alors que l'on peut mettre ces arborescences en bijection avec une famille d'arborescences pour lesquelles on peut appliquer la formule de Good. Nous obtenons ainsi une formule dénombrant le nombre de m-constellations ayant n sommets et f faces et en déduisons un algorithme de génération aléatoire pour ces structures. Dans le cas $m = 2$, nous obtenons de plus une nouvelle formule dénombrant les cartes biparties selon le nombre de sommets et de faces.

Recherche de motifs. Dans la seconde partie de ce mémoire, nous nous intéressons au problème algorithmique constitué par la recherche de motifs dans une arborescence. Le problème de la recherche de motifs ("pattern matching") est un problème classique en algorithmique (voir par exemple le récent livre édité par Apostolico et Galil [7]) que l'on peut définir de la façon suivante.

> Étant données deux structures de même type, un texte T et un motif M, on cherche à repérer toutes les occurrences (cette notion dépendant des structures considérées) de M présentes dans T.

Dans le cas des mots, lorsque par exemple, $T = bacacbacbabc$ et $M = acba$, on a deux occurrences de M dans T débutant respectivement en position 4 et 7. Dans le cas des arborescences, on considère habituellement des arborescences ordonnées (pour chaque sommet, on impose un ordre linéaire sur ses fils) telles que tout sommet est muni d'une étiquette appartenant à un alphabet Σ, deux sommets pouvant ici avoir la même étiquette. Intuitivement, on dit qu'on a une occurrence de M (le *motif*) en un sommet x de T (le *texte*) si en plaçant la racine de M en x, on arrive à une situation (que l'on devra préciser) dans laquelle tout sommet (resp. arête) de M recouvre un sommet (resp. arête) de T ayant la même étiquette.

Dans nos travaux, nous nous sommes plus particulièrement intéressés au problème de la recherche de motif dans un texte *statique*. Revenons aux mots et considérons par exemple le cas d'une encyclopédie sur CD-ROM. Une des fonctionnalités indispensables à un tel logiciel est une procédure permettant de repérer toutes les occurrences d'un mot M dans le texte de cette encyclopédie. En remarquant que dans toute instance (T,M) de ce problème le paramètre T est le même (le texte de l'encyclopédie), il est naturel de vouloir traiter ces problèmes de recherche de motifs de la manière suivante : prétraiter le texte T une fois pour toute lors de la création du CD-ROM et utiliser les informations de ce prétraitement pour rechercher rapidement les occurrences d'un motif[3]. La solution généralement employée dans ce cas consiste à construire une structure de données annexe indexant les suffixes

3. Parmi les nombreuses autres applications où l'on retrouve la situation d'un texte statique, on peut citer la recherche de motifs dans des banques de données regroupant des milliers de séquence issues de l'analyse des génomes (une fois qu'une séquence est ajoutée dans la banque, elle n'est plus modifiée et elle peut donc être prétraitée lors de cette insertion).

du texte T : en remarquant que toute occurrence du motif M dans T est préfixe d'un des suffixes de T, cet index permet de déterminer rapidement les occurrences de M dans T (en général en temps proportionnel à la seule taille du motif, c'est à dire indépendamment de la taille du texte). Dans le cas des mots, les principales structures de données utilisées pour indexer les suffixes d'un texte sont l'automate des suffixes, le tableau des suffixes et l'arborescence des suffixes [52, chapitres 5 et 6].

Notre but est de traiter le problème de la recherche de motifs dans une arborescence *statique* en nous inspirant essentiellement de l'utilisation de l'arborescence des suffixes dans le cas des mots. Dans un premier chapitre, nous définissons formellement la notion d'occurrence d'un motif M dans une arborescence T et une restriction de cette notion (que nous appelons occurrence compacte). On peut cependant noter que la notion générale d'occurrence est équivalente à la notion d'occurrence compacte si les arborescences considérées sont des termes (encore appelés arborescences d'expression). Nous poursuivons par une rapide présentation du problème de la recherche de motifs dans les mots, en nous intéressant surtout à une structure de données souvent utilisée dans le cas d'un texte statique : *l'arborescence des suffixes d'un mot*, qui est à la base des résultats que nous proposons par la suite. Nous effectuons ensuite une revue des principaux articles traitant de recherche de motifs dans une arborescence. Cette étude nous amène notamment à faire les remarques suivantes :

- le sujet a été relativement peu étudié, notamment d'un point de vue pratique, la plupart des algorithmes proposés étant assez complexes,
- tous les algorithmes proposés recherchent les occurrences compactes du motif dans le texte et ne semblent pas pouvoir être aisément étendus pour rechercher les occurrences générales,
- aucun des algorithmes proposés ne se place dans le cas d'un texte statique.

Nous concluons ce chapitre par une adaptation simple d'un algorithme proposé Hoffmann et O'Donnell [91] pour la recherche des occurrences compactes qui nous permet de rechercher les occurrences générales du motif dans le texte.

Le second chapitre propose une approche du cas statique pour les termes reposant sur le codage d'un terme par un mot et l'utilisation de l'arborescence des suffixes de ce mot. La complexité dans le pire des cas de l'algorithme que nous proposons n'améliore pas les résultats théoriques connus, mais des résultats expérimentaux nous laissent penser qu'il est très intéressant d'un point de vue pratique. Nous proposons aussi une variante de cet algorithme pour la recherche des occurrences compactes du motif dans le texte.

Le dernier chapitre de cette partie propose une approche plus théorique du problème que nous considérons, dans laquelle nous étendons la notion d'arborescence des suffixes des mots aux arborescences. En effet, étant donnée une famille de structures de données et une notion de suffixes de ces structures, il est naturel de vouloir définir une notion d'arborescence des suffixes pour ces structures. Giancarlo et Grossi ont par exemple suivi cette démarche dans le cas des matrices carrées [66, 67]. Récemment, Kosaraju [105] a introduit une notion d'arborescence des suffixes d'une arborescence, dans laquelle un suffixe est un chemin allant d'un sommet vers la racine. Dans ce chapitre, nous introduisons une nouvelle notion de suffixe, inspirée de la notion de suffixe d'un mot : un suffixe d'une arborescence T est une sous-arborescence maximale de T. On a donc exactement un suffixe par sommet x de T (la sous-arborescence de T de racine x), ce qui permet de définir une notion d'arborescence des suffixes d'une arborescence T occupant un espace linéaire par rapport au nombre de sommets de T. Nous proposons ensuite un algorithme permettant de construire une telle arborescence des suffixes, dérivé de l'algorithme de McCreight pour l'arborescence des suffixes d'un mot [118]. Nous concluons en évoquant les applications de de cette structure en termes de recherche de motifs dans une arborescence.

Première partie

Énumération de structures arborescentes

Chapitre 1

Énumération d'arborescences

Dans cette première partie, nous considérons les arborescences d'un point de vue combinatoire, en nous attachant à *énumérer* plusieurs familles d'entre elles. Dans ce premier chapitre, nous présentons quelques résultats classiques sur ce sujet.

1.1 Définitions

Dans ce premier paragraphe, nous introduisons les définitions et notations nécessaires. On rappelle tout d'abord la définition la plus générale d'une arborescence.

> *Une arborescence à n sommets est un graphe connexe sans cycle ayant n sommets (et donc $(n-1)$ arêtes) dont on a distingué un sommet, appelé la racine.*

Définition 1. Une arborescence est dite *étiquetée* si chacun de ses sommets est muni d'une étiquette de sorte que l'ensemble des sommets est totalement ordonné. En général, on étiquette une arborescence à n sommets avec les entiers de 1 à n (on parle alors d'une arborescence sur $[n]$).

Notation. Sauf mention contraire, toutes les arborescences que nous considérons sont étiquetées et nous omettons donc de le préciser.

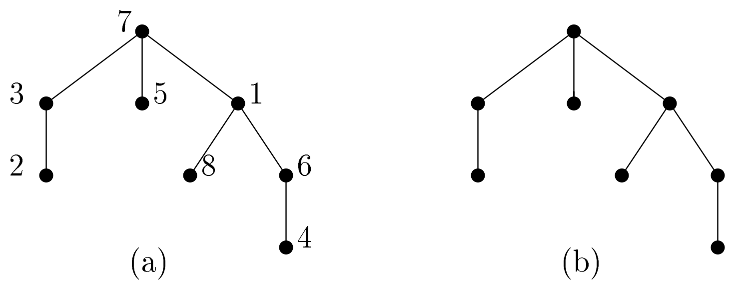


Fig. 1.1 – (a) Une arborescence étiquetée sur [8] - (b) Une arborescence non étiquetée

Dans la suite de cette partie, pour introduire la notion de *famille d'arborescences*, nous nous inspirons de la théorie des espèces de structures [16], développée par l'école montréalaise de combinatoire à la suite des travaux d'André Joyal [96], particulièrement adaptée à l'énumération de structures arborescentes.

Définition 2. Soit A une arborescence et x un sommet de A. On appelle *fibre de x* l'ensemble des fils de x.

Nous considérons diverses familles d'arborescences qui se distinguent par la *structure combinatoire (étiquetée) dont on munit la fibre de chaque sommet.*

Définition 3. Soit $\mathcal{F}$ une famille (ou classe) d'objets combinatoires étiquetés et $\mathcal{F}_n$ l'ensemble des objets de $\mathcal{F}$ sur $[n]$. On appelle $\mathcal{F}$-arborescences l'ensemble des arborescences telles que pour tout sommet x, si sa fibre comporte n sommets, elle est munie d'une structure de $\mathcal{F}_n$ (on dit alors que la fibre de x est munie d'une $\mathcal{F}$-structure).

Par exemple, les familles d'arborescences les plus couramment étudiées dans la littérature sont

- les arborescences de Cayley, pour lesquelles seules comptent les relations d'adjacence entre sommets (la fibre de chaque sommet est munie d'une structure d'ensemble et il y a donc plusieurs façons de dessiner une arborescence de Cayley),
- les arborescences ordonnées, pour lesquelles les fils d'un sommet sont totalement ordonnés (la fibre de tout sommet est munie d'une structure de liste ordonnée).

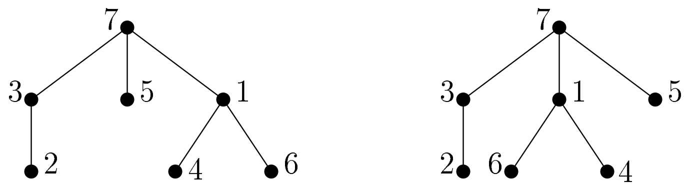


Fig. 1.2 – Deux arborescences ordonnées différentes, mais identiques si on les considère comme arborescences de Cayley

Notation. Nous notons $\mathcal{A}$ (resp. $\mathcal{O}$) l'ensemble des arborescences de Cayley (resp. ordonnées), $\mathcal{A}_n$ (resp. $\mathcal{O}_n$) l'ensemble des arborescences de Cayley (resp. ordonnées) à n sommets et a_n (resp o_n) le nombre d'arborescences de Cayley (resp. ordonnées) à n sommets.

On peut bien entendu utiliser toute classe de structures combinatoires étiquetées pour définir une famille d'arborescences (rien n'interdit par exemple de munir la fibre d'un sommet d'une structure d'arborescence ordonnée binaire, structure que l'on retrouve dans diverses structures de données), ou d'utiliser diverses structures pour définir une famille d'arborescences (on peut par exemple munir la fibre des sommets pairs d'une structure de liste ordonnée et la fibre des sommets impairs d'une structure d'ensemble).

Notation. Les principales classes de structures que nous utilisons pour définir des familles d'arborescences sont celles d'ensemble, notée $\mathcal{E}$, celle de liste ordonnée, notée $\mathcal{L}$ et celle de liste circulairement ordonnée, notée $\mathcal{C}$. Les arborescences de Cayley sont donc les $\mathcal{E}$-arborescences et les arborescences ordonnées les $\mathcal{L}$-arborescences.

Définition 4. Soient $\mathcal{F}$ et $\mathcal{G}$ deux familles de structures combinatoires. On appelle $\mathcal{G}$-assemblée de $\mathcal{F}$-arborescences tout ensemble de $\mathcal{F}$-arborescences muni d'une $\mathcal{G}$-structure. Si les étiquettes des arborescences présentes dans une $\mathcal{G}$-assemblée sont deux à deux distinctes et correspondent à l'ensemble $[n]$, on parle alors de $\mathcal{G}$-assemblée de $\mathcal{F}$-arborescences sur $[n]$. On appelle *forêt* de $\mathcal{F}$-arborescences les $\mathcal{E}$-assemblées de $\mathcal{F}$-arborescences.

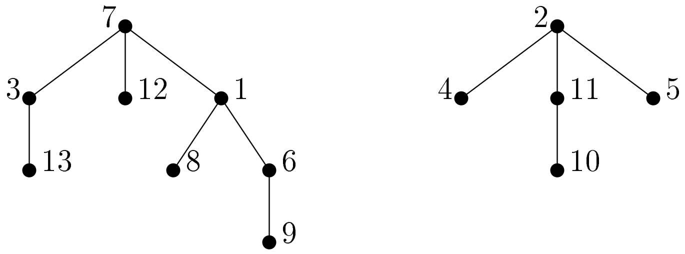


FIG. 1.3 – Une forêt à 13 sommets

Bien que la structure de la fibre des sommets soit la principale condition permettant de définir des familles d'arborescences, on peut bien entendu imposer d'autres conditions. Par exemple, on peut considérer des arborescences *croissantes* (pour tout sommet différent de la racine, son étiquette est supérieure à celle de son père) [41, 15, 132], des arborescences alternantes (chapitre 4), etc.

1.2 Énumération d'arborescences de Cayley

Le premier - et le plus fameux - résultat concernant l'énumération d'arborescences de Cayley est généralement attribué à Arthur Cayley [29] et donne une formule étonnement simple pour le nombre de telles arborescences à n sommets.

Théorème 5. *Le nombre d'arborescences de Cayley à n sommets est* n^{n-1}.

Il existe de nombreuses preuves du théorème 5 (voir notamment les travaux de Moon [119, 120] pour un récapitulatif des preuves connues à la fin des années 60, la preuve de Joyal [96] ou encore Shor [142] pour une nouvelle preuve récente). Dans cette section, nous présentons successivement deux preuves de ce résultat, qui illustrent deux approches différentes dans la résolution de problèmes d'énumération : l'approche bijective et l'approche formelle par la manipulation de séries génératrices.

1.2.1 L'approche bijective : le codage de Prüfer

Le principe de l'approche bijective est le suivant. Soit $\mathcal{F}$ une classe d'objets combinatoires dont on veut déterminer le cardinal $|\mathcal{F}|$ et $\mathcal{G}$ une autre classe d'objets combinatoires dont on connaît le cardinal. Si on peut décrire deux algorithmes $AlgoF$ et $AlgoG$ tels que

- $AlgoF$ transforme tout objet de $\mathcal{F}$ en un objet de $\mathcal{G}$,
- $AlgoG$ transforme tout objet de $\mathcal{G}$ en un objet de $\mathcal{F}$,
- pout tout objet $F \in \mathcal{F}$, $AlgoG(AlgoF(F)) = F$,
- pout tout objet $G \in \mathcal{G}$, $AlgoF(AlgoG(G)) = G$,

on a établi une bijection entre les ensembles $\mathcal{F}$ et $\mathcal{G}$, et on obtient alors une preuve du fait que $|\mathcal{F}| = |\mathcal{G}|$.

Le théorème 5 suggère l'existence d'une bijection entre les arborescences de Cayley à n sommets et les mots de longueur $(n-1)$ sur l'alphabet $[n]$. Dans [129] Prüfer propose une telle bijection, à la fois simple et élégante. Nous commençons par décrire cette bijection, puis nous montrons son intérêt pour affiner le résultat du théorème 5 et engendrer aléatoirement et uniformément des arborescences de Cayley.

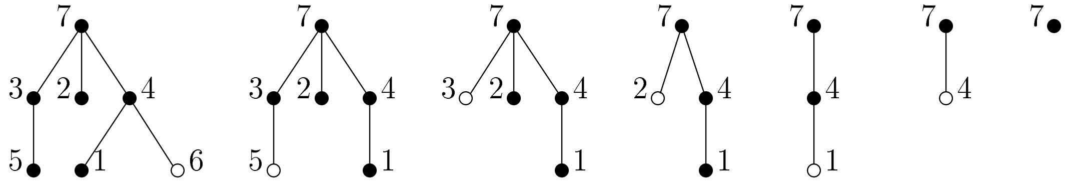
Dans un premier temps, examinons la transformations des arborescences en mots. Soit A une arborescence de $\mathcal{A}_n$. On construit alors un mot m_A de $(n-1)$ lettres sur l'alphabet $[n]$ par l'algorithme suivant.

Algorithme 1: Codage de Prüfer : des arborescences vers les mots

Données : $A \in \mathcal{A}_n$
Résultat : m_A : mot de $(n-1)$ lettres sur l'alphabet $[n]$
début
- $m_A := \epsilon$ (le mot vide);
- **tant que** A n'est pas réduite à sa racine **faire**
  - soit x la plus grande feuille de A et y le père de x;
  - $m_A := m_A.y$ (ajouter y à la fin de m_A);
  - supprimer le sommet x et l'arête (y,x) de A;
- **retourner** m_A

fin

Dans la figure suivante, nous donnons un exemple détaillé d'exécution de cet algorithme. À chaque étape, la plus grande feuille (qui sera retirée de A) est indiquée par un sommet blanc.



m_A : 4 4.3 4.3.7 4.3.7.7 4.3.7.7.4 4.3.7.7.4.7

En raisonnant par induction sur le nombre de sommets de A, on vérifie immédiatement que deux arborescences de Cayley différentes ne peuvent donner le même mot. De plus, par construction, on vérifie les propriétés suivantes.

Propriété 6. La dernière lettre de m_A est la racine de A.

Propriété 7. La lettre x apparaît k fois dans m_A si et seulement si le sommet x a k fils dans A. En particulier, les feuilles de A n'apparaissent pas dans m_A.

Ces deux propriétés prouvent alors que l'algorithme 2 que nous donnons ci-dessous, est bien l'inverse du précédent et reconstruit l'arborescence A à partir du mot m_A.

Il n'est pas difficile de vérifier que cet algorithme produit toujours une arborescence et que deux mots ne peuvent donner la même arborescence, ce qui établit la bijection entre mots et arborescences, et par conséquent le théorème 5.

Un des intérêts des preuves bijectives réside dans le fait qu'elles impliquent souvent une meilleure compréhension de la structure des objets que l'on compte et qu'elles permettent

Algorithme 2: Codage de Prüfer : des mots vers les arborescences

Données : m : mot de $(n-1)$ lettres sur l'alphabet $[n]$
Résultat : $A_m \in \mathcal{A}_n$
début
  soient A_m la forêt vide et F l'ensemble des lettres de $[n]$ non présentes dans m;
  pour i allant de 1 à $(n-1)$ **faire**
    soient x le plus grand élément de F et y la première lettre de m;
    ajouter l'arête (y,x) à la forêt A_m;
    supprimer x de F et la première lettre (y donc) de m;
    si y n'apparaît plus dans m **alors** ajouter y à F;
  retourner A_m
fin

donc d'obtenir des raffinements de résultats d'énumération et des preuves élégantes de certaines identités combinatoires. Nous allons en voir maintenant quelques exemples.

Corollaire 8. *Soit $r \in [n]$. Le nombre d'arborescences de Cayley sur $[n]$ de racine r est n^{n-2}.*

Preuve. On déduit directement de la propriété 6 que, les arborescences de Cayley sur $[n]$ ayant pour racine $k \in [n]$ sont en bijection avec les mots de $(n-1)$ lettres sur l'alphabet $[n]$ dont la dernière lettre est r, d'où le résultat. □

Théorème 9. [44] *Le nombre d'arborescences de Cayley sur $[n]$ telles que la racine a d fils est*

$$n\binom{n-2}{d-1}(n-1)^{n-1-d}.$$

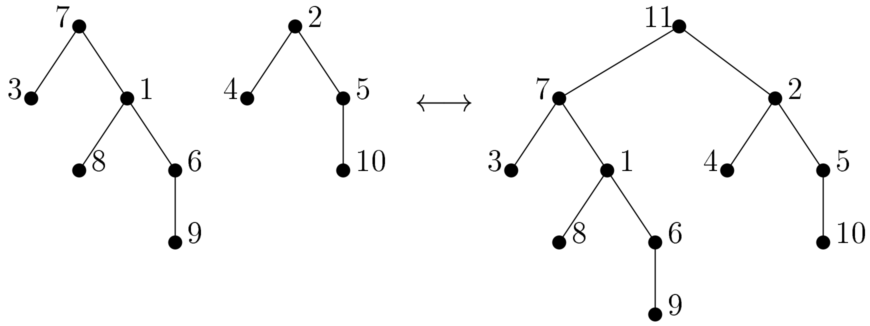
Preuve. Des propriétés 6 et 7, on déduit qu'une arborescence de Cayley A dont la racine possède exactement d fils correspond à un mot de longueur $(n-1)$ sur $[n]$ qui contient exactement d occurrences de sa dernière lettre et le résultat en découle. □

On peut déduire immédiatement du résultat précédent une formule pour le nombre de forêts d'arborescences de Cayley.

Théorème 10. *Le nombre de forêts de Cayley sur $[n]$ comportant k arborescences ($k > 0$) est*

$$\binom{n-1}{k-1}n^{n-k}. \tag{1.1}$$

Preuve. Une forêt de Cayley sur $[n]$ comportant k arborescences correspond à une arborescence de Cayley sur $[n+1]$ dont la racine $(n+1)$ a k fils :



Le résultat découle directement de la formule pour le nombre d'arborescences dont la racine a degré k. □

Le résultat de Cayley [29] est en fait le suivant (voir aussi Takács [154, 153] pour deux preuves récentes de ce résultat), que nous utiliserons dans le chapitre 2.

Théorème 11. *Soit* $\{r_1, \cdots, r_k\} \subset [n]$*. Le nombre de forêts de Cayley sur* $[n]$ *comportant* k *arborescences de racines respectives* $r_1, \cdots, r_k$ $(k > 0)$ *est*

$$k\ n^{n-k-1}.$$

Preuve. Ce résultat découle directement du théorème 10, en divisant (1.1) par $\binom{n}{k}$. □

Un autre intérêt des preuves bijectives réside dans leur application à la mise au point d'algorithmes de génération aléatoire de structures combinatoires, qui s'avèrent souvent utiles pour l'étude expérimentale d'algorithmes ou l'obtention de propriétés asymptotiques des objets engendrés. Pour engendrer aléatoirement une arborescence de Cayley à n sommets, uniformément parmi toutes les arborescences de $\mathcal{A}_n$, il suffit d'engendrer aléatoirement et uniformément un mot de longueur $(n-1)$ sur $[n]$, ce qui ne pose aucun problème et se fait efficacement, puis d'appliquer l'algorithme 2 transformant un mot en une arborescence.

1.2.2 L'approche formelle : séries génératrices et formule de Lagrange

Il n'est pas toujours facile d'exhiber une preuve bijective pour un résultat énumératif. De plus, l'approche bijective telle que présentée ci-dessus a le défaut de ne pas être très générale : il faut trouver une bijection pour chaque problème traité. Le principal outil permettant, si ce n'est "d'automatiser" la résolution des problèmes d'énumération, tout au moins d'en traiter une vaste classe est la *série génératrice.*

Définition 12. Soient $\mathcal{F}$ une famille d'objets combinatoires munis d'un paramètre entier positif $p\ : \mathcal{F} \to N$ (appelé la taille). On note f_n le nombre (fini) d'objets de $\mathcal{F}$ de taille n. On appelle *série génératrice ordinaire* de $\mathcal{F}$ la série formelle

$$\sum_{n\geq 0} f_n x^n,$$

et *série génératrice exponentielle* de $\mathcal{F}$ la série formelle

$$\sum_{n\geq 0} f_n \frac{x^n}{n!}.$$

Notation. On utilise la notation $[x^n]F(x)$ pour désigner le coefficient de x^n dans la série formelle $F(x)$, c'est-à-dire le coefficient de x^n dans le développement de Taylor de la série $F(x)$ au voisinage de 0.

Il existe d'autres familles de séries formelles utilisées en combinatoire énumérative, comme par exemple les séries de Dirichlet. On peut également, comme on le verra à partir du chapitre 5, considérer plusieurs paramètres sur une famille d'objets combinatoires, ce qui se traduit par l'introduction de séries formelles multivariées (à chaque paramètre est assignée une variable).

On utilise habituellement les séries génératrices exponentielles pour traiter les structures étiquetées et les séries génératrices ordinaires pour traiter les structures non étiquetées. Comme par la suite, nous nous intéressons à des objets étiquetés (les arborescences par exemple), sauf mention contraire, nous emploierons le terme série génératrice pour désigner une série génératrice exponentielle.

Remarque 13. Les séries génératrices exponentielles associées respectivement à $\mathcal{E}$ (la famille des ensembles), $\mathcal{L}$ (la famille des listes ordonnées) et $\mathcal{C}$ (la famille des listes circulairement ordonnées) sont $E(x) = e^x$, $L(x) = 1/(1-x)$ et $C(x) = 1 + \ln(1/(1-x))$.

Un des intérêts des séries génératrices réside dans le fait que l'on peut traduire les opérations usuelles de construction combinatoire en termes d'opérations sur les séries génératrices des objets considérés (cela vaut aussi bien pour les séries génératrices ordinaires qu'exponentielles).

Par exemple, soient $\mathcal{F}$ et $\mathcal{G}$ deux familles d'objets combinatoires, de séries génératrices respectives $F(x)$ et $G(x)$. Si $\mathcal{H} = \mathcal{F} \cup \mathcal{G}$ (on suppose cette union disjointe : un objet H de $\mathcal{H}$ est soit un objet de $\mathcal{F}$ soit un objet de $\mathcal{G}$, la taille de H étant sa taille dans $\mathcal{F}$ si $H \in \mathcal{F}$ ou sa taille dans $\mathcal{G}$ si $H \in \mathcal{G}$), la série génératrice $H(x)$ de $\mathcal{H}$ est

$$H(x) = F(x) + G(x).$$

Si $\mathcal{H} = \mathcal{F} \times \mathcal{G}$ (un objet de $H \in \mathcal{H}$ est un couple (F,G) formé d'un objet de $F \in \mathcal{F}$ et d'un objet de $G \in \mathcal{G}$, sa taille étant la somme des tailles de F et G), la série génératrice $H(x)$ de $\mathcal{H}$ est

$$H(x) = F(x).G(x).$$

Mais l'opération qui s'avère la plus utilisée dans le cas des structures arborescentes est la composition de séries. Soit $\mathcal{G}$ une classe de structures donnée, de série génératrice $G(x)$. Pour une famille $\mathcal{H}$ d'objets combinatoires, de série associée $H(x)$, la série associée aux $\mathcal{G}$-assemblées d'objets de $\mathcal{H}$ (la taille d'une telle assemblée étant la somme des tailles des objets présents dans cette assemblée) est alors

$$G(H(x)).$$

Cette opération est particulièrement adaptée à l'énumération de structures arborescentes. En effet, pour une classe $\mathcal{G}$ donnée (de série génératrice $G(x)$), une $\mathcal{G}$-arborescence peut se décomposer en deux parties : un sommet isolé (la racine) et une $\mathcal{G}$-assemblée de $\mathcal{G}$-arborescences (figure 1.4). La série génératrice $A_G(x)$ de la famille des $\mathcal{G}$-arborescences (dans ce cas le paramètre taille est le nombre de sommets) vérifie alors l'équation fonctionnelle

$$A_G(x) = xG(A_G(x)). \tag{1.2}$$

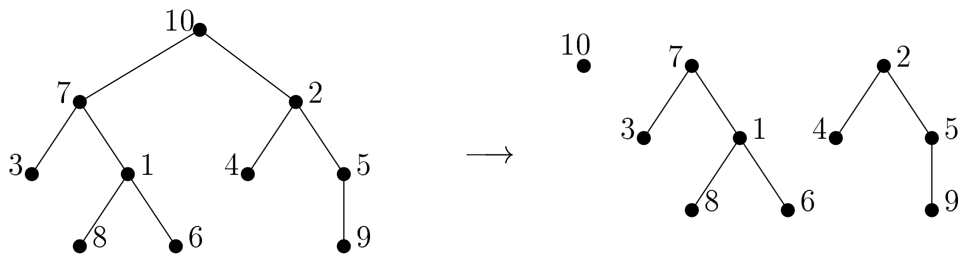


FIG. 1.4 – Décomposition d'une arborescence de Cayley

Dans certains cas, on peut déduire directement d'une telle équation fonctionnelle une expression pour la série considérée. Par exemple, pour la série génératrice $O(x)$ des arborescences ordonnées $\mathcal{O}$, on déduit de l'équation fonctionnelle

$$O(x) = xL(O(x)),$$

où

$$L(x) = \frac{1}{1-x}$$

est la série associée à la famille des ordres linéaires, que

$$O(x) = \frac{1-\sqrt{1-4x}}{2} \tag{1.3}$$

et un rapide calcul permet d'extraire les coefficients de cette série et de montrer que le nombre d'arborescences ordonnées sur $[n]$ est

$$\frac{n!}{n+1}\binom{2n}{n} = n!C_n$$

où C_n est le $n^{ème}$ nombre de Catalan (voir [148] pour une présentation de ces nombres, que l'on rencontre très souvent en combinatoire).

On ne peut cependant pas toujours obtenir une forme close pour la série génératrice recherchée. C'est notamment le cas des arborescences de Cayley, dont la série $A(x)$ vérifie l'équation fonctionnelle

$$A(x) = xe^{A(x)}. \tag{1.4}$$

C'est là qu'intervient un outil fondamental en combinatoire énumérative pour traiter les séries vérifiant des équations de la forme (1.2) (que nous désignons par le terme d'*équations lagrangiennes*) : la formule de Lagrange pour l'inversion de séries formelles (sur la formule de Lagrange et ses applications à l'énumération de structures arborescentes, on peut se reporter à [77, 146, 16]).

Théorème 14. *Soit $F(x)$, $G(x)$ et $H(x)$ trois séries formelles telles que $G(0) \neq 0$. L'équation*

$$F(x) = xG(F(x)) \tag{1.5}$$

détermine alors la série formelle en $H(F(x))$ par :

$$[x^n]H(F(x)) = \frac{1}{n}[x^{n-1}]\frac{\partial H(x)}{\partial x}(F(x))^n. \tag{1.6}$$

En appliquant cette formule aux arborescences de Cayley ($G(x) = e^x$ et $H(x) = x$), on obtient alors immédiatement le résultat du théorème 5.

De même, en posant $H(x) = x^k/k!$, c'est-à-dire la série génératrice des ensembles de taille k (cette série se réduit donc à un seul terme) on obtient le résultat du théorème 10 sur les forêts d'arborescences de Cayley.

Pour une description détaillée de l'utilisation des séries génératrices pour l'énumération de structures décomposables on peut se reporter aux ouvrages de Goulden et Jackson [77], Wilf [165], Stanley [146, chapitre 5] ou Bergeron, Labelle et Leroux [16]. Ces séries sont aussi très utilisées en analyse d'algorithmes, et dans ce domaine, on peut se reporter au livre de Flajolet et Sedgewick [139] ou à l'article de Steyaert et Flajolet [149] qui présente une très belle illustration de ces techniques dans l'analyse d'un algorithme de recherche de motifs dans une arborescence.

Pour conclure ce chapitre, il faut cependant mentionner que, contrairement à ce que peut laisser supposer le titre de cette section ("l'approche formelle"), l'utilisation de séries génératrices et de la formule d'inversion de Lagrange n'est pas purement formelle. En effet, l'obtention d'une équation "lagrangienne" (de la forme de l'équation (1.5)) pour une série implique une compréhension de la structure des objets correspondants. De plus, il existe plusieurs preuves combinatoires de cette formule qui permettent souvent de traduire de manière bijective les résultats obtenus par application de l'inversion de Lagrange. La première preuve combinatoire de la formule de Lagrange est due à Raney [131], qui montre que le membre droit de (1.6) énumère les mots du langage de Lukasiewicz pondéré par les coefficients de la série $G(x)$. En utilisant le principe de la conjugaison de mots, il en déduit une preuve combinatoire de la formule d'inversion de Lagrange. Cette preuve est basée sur l'énumération de mots d'un langage algébrique, et donc sur la manipulation d'objets non étiquetés et de séries génératrices ordinaires en variables non commutatives. La première preuve bijective de la formule de Lagrange manipulant des structures étiquetées et des séries génératrices exponentielles apparaît dans les deux premiers articles fondant la théorie des espèces de structures [96, 108] (voir aussi [77] et [16, section 3.1]), et est basée sur une élégante bijection entre certaines familles d'arborescences et certaines familles d'endofonctions, que nous présentons au chapitre 5. Parmi les travaux portant sur l'interprétation combinatoire de l'inversion de Lagrange, on peut aussi mentionner les travaux de Chen [38, 39], basés sur l'utilisation d'arborescences de Schröder et dans lesquels Chen réunit sous un même formalisme une preuve manipulant des structures non étiquetées et une preuve manipulant des structures étiquetées.

Chapitre 2

Arborescences de Cayley et fils de la racine

À l'origine des problèmes abordés dans ce chapitre se trouvent deux identités concernant respectivement $(n+1)^n$ et n^n. La première n'est autre qu'une application de la formule du binôme :

$$(n+1)^n = \sum_{k=0}^{n} \binom{n}{k} n^{n-k}. \tag{2.1}$$

De manière surprenante, nous n'avons trouvé dans la littérature aucune interprétation simple et directe de l'identité (2.1) en termes d'arborescences de Cayley de racine non fixée (on explique en effet facilement cette identité en distribuant les arborescences de Cayley de racine étiquetée 1 et telles que le plus grand sommet est une feuille suivant le paramètre “degré de la racine”, comme on le verra dans le paragraphe 2.1.1). La seconde identité qui a motivé cette étude est un cas particulier d'un résultat dû à Goulden et Jackson [79, proposition 3.1], relié à un problème de factorisations dans le groupe symétrique :

$$n^n = n(n-1)^{n-1} + \sum_{k=1}^{n-1} \binom{n}{k} \; k^k \; (n-1-k)^{n-1-k}. \tag{2.2}$$

Pour aborder ces deux problèmes, nous avons été amenés à étudier les arborescences de Cayley selon un paramètre nouveau, *le nombre de fils de la racine qui lui sont inférieurs.*

Remarque 15. Récemment, la famille des arborescences de Cayley telles que la racine est inférieure à ses fils a été introduite par Gessel dans le cadre de travaux (non publiés mais cités par Athanasiadis et Linusson [9]) sur l'énumération de régions de certains arrangements d'hyperplans (voir la section 2.5 de ce chapitre). Gessel montre, de manière formelle, qu'il y a n^n arborescences de Cayley sur $[n+1]$ telles que la racine est inférieure à ses fils. La démonstration du lemme 17 fournit une preuve bijective de ce résultat.

Dans la première section, nous montrons de manière bijective que l'énumération des arborescences de Cayley distribuées suivant le paramètre “nombre de fils de la racine qui lui sont inférieurs” donne une explication de l'identité (2.1). On peut noter que si cette identité et l'explication que nous en proposons sont simples, la preuve passe par une construction nécessitant l'examen de plusieurs cas non triviaux, ce qui explique sans doute que cette interprétation n'ait pas été trouvée plus tôt. La section suivante est consacrée à une étude combinatoire des arborescences de Cayley suivant ce paramètre. Cette étude permet de donner des preuves combinatoires d'identités, la plupart nouvelles à notre connaissance, et notamment de (2.2). Dans une troisième section, nous relions la famille des arborescences de Cayley telles que la racine est inférieure à tous ses fils à un problème de factorisation de permutations circulaires en un nombre minimal de transpositions. Nous concluons par l'utilisation de l'inversion de Lagrange pour compter les arborescences ordonnées, cycliques et planes dont la racine est inférieure à ses fils.

Notation. Dans les sections 2.1, 2.2 et 2.3, toutes les arborescences considérées sont des arborescences de Cayley et nous nous contentons donc de parler d'arborescences.

2.1 Arborescences de Cayley

Dans cette section, nous proposons une interprétation de l'identité (2.1) en termes d'arborescences, via le résultat suivant.

Théorème 16. *Le nombre d'arborescences sur $[n+1]$ telles que la racine a exactement k fils qui lui sont inférieurs est*

$$\binom{n}{k} n^{n-k}.$$

Dans le cas $k = 0$, ce résultat, qui montre que le nombre d'arborescences sur $[n]$ telles que la racine est inférieure à ses fils est $(n-1)^{n-1}$, peut être vu en fait comme un résultat intermédiaire entre les formules donnant le nombre d'arborescences de racine 1 (n^{n-2}) et le nombre d'arborescences sans restriction (n^{n-1}).

La suite de cette section est consacrée à une preuve bijective du théorème 16. Nous procédons en deux étapes. Dans un premier temps, nous prouvons le cas particulier $k = 0$ (i.e. nous montrons que le nombre d'arborescences sur $[n+1]$ telles que la racine est plus petite que tous ses fils est n^n). Dans un second temps, nous étendons la bijection précédente au cas général pour prouver le théorème 16.

Notation. On note $\mathcal{A}_{n,k}$ la famille des arborescences sur $[n]$ telles que la racine a exactement k fils qui lui sont inférieurs.

2.1.1 Arborescences dont la racine est inférieure à ses fils

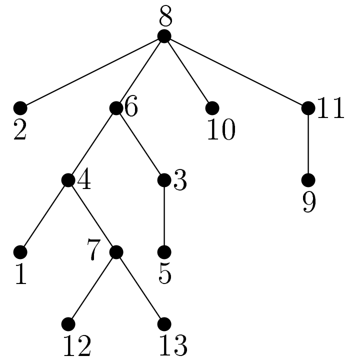
Soit $\mathcal{B}_n$ la famille des arborescences sur $[n]$ telles que n est une feuille. On déduit immédiatement de la formule de Cayley (théorème 5), que $|\mathcal{B}_n| = (n-1)^{n-1}$ (pour obtenir une arborescence de $\mathcal{B}_n$ il suffit en effet de considérer une arborescence de Cayley sur $[n-1]$ ayant un sommet distingué et d'insérer n comme fils du sommet distingué).

Lemme 17. *Il existe une bijection ϕ entre $\mathcal{B}_n$ et $\mathcal{A}_{n,0}$.*

Définition 18. Soit A une arborescence, x et y deux sommets de A tels que x est le père de y. L'arête (x,y) est *décroissante* (resp. *croissante*) si $x > y$ (resp. $x < y$). Une arborescence est *décroissante* (resp. *croissante*) si toutes ses arêtes sont décroissantes (resp. croissantes).

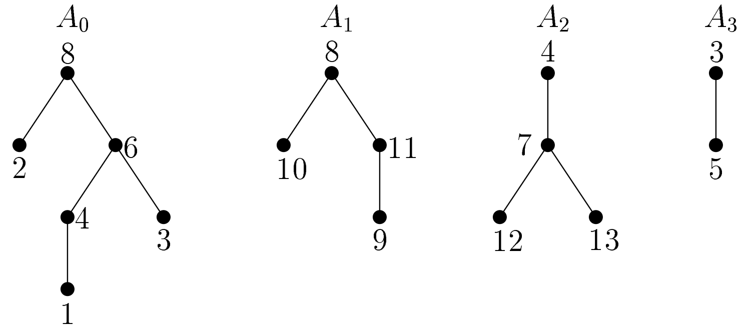
La bijection ϕ se décompose en trois étapes, que nous illustrons au fur et à mesure avec un exemple. Soit A une arborescence de $\mathcal{B}_n$, de racine r. On cherche à construire une arborescence $A' = \phi(A)$ appartenant à $\mathcal{A}_{n,0}$.

Considérons par exemple l'arborescence A sur 13 sommets suivante, de racine $r = 8$.

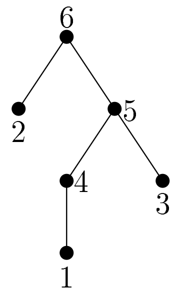


Étape 1. La première étape de la bijection ϕ consiste à décomposer A en un ensemble non ordonné d'arborescences $(A_0,A_1,\dots,A_k)$ défini comme suit :

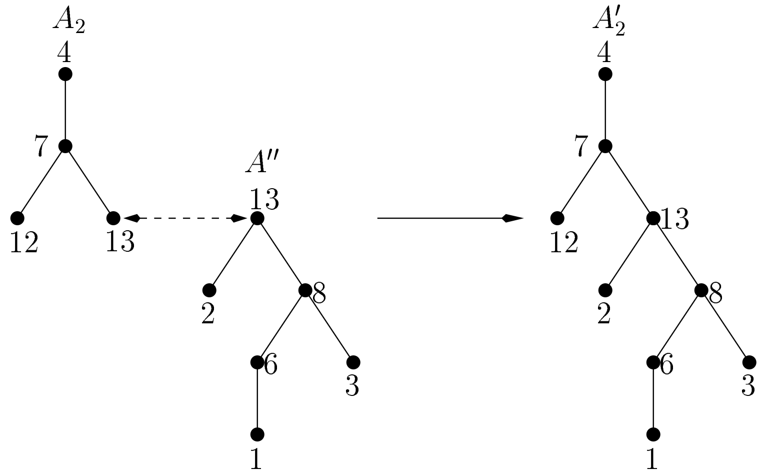
- A_0 est la sous-arborescence décroissante maximale de A enracinée en r,
- $A_1,\dots,A_k$ sont les arborescences ayant au moins deux sommets et obtenues à partir de A par suppression de toutes les arêtes appartenant à A_0.



On obtient alors un ensemble d'arborescences telles que les racines des A_i, pour $i = 0,\dots,k$, sont toutes des sommets de A_0. Par construction, on peut dire que, pour $i = 1,\dots,k$, chacune des arborescences A_i est telle que la racine est inférieure à ses fils. De plus, si l'arborescence décroissante A_0 a m sommets, elle peut être codée par un couple (A'_0,S) où A'_0 est une arborescence décroissante sur $[m]$ et $S \subset [n]$ est le sous-ensemble de $[n]$ formé des étiquettes de A_0.
Dans notre exemple, nous avons $m = 6$, $S = \{1,2,3,4,6,8\}$ et A'_0 est l'arborescence suivante.

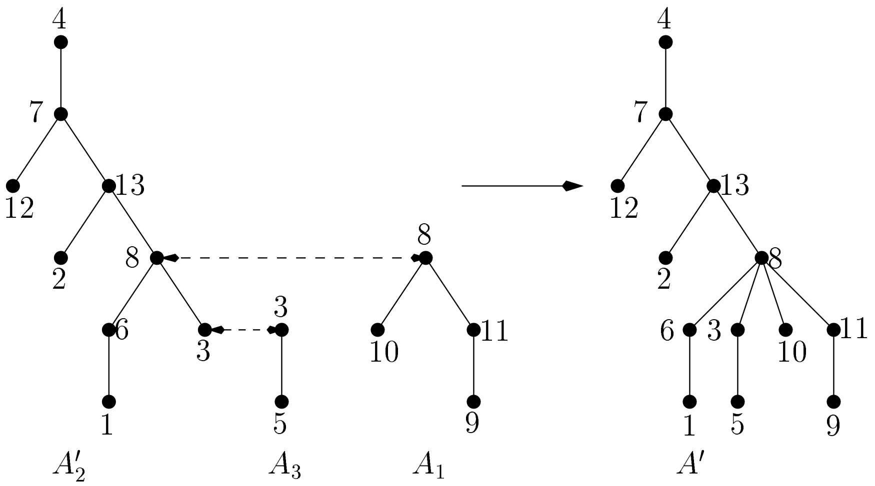


Étape 2. Soit A_j l'arborescence contenant le sommet étiqueté n, et r_j sa racine. On définit un ensemble S' en remplaçant r_j par n dans S ($S' = \{n\} \cup S/\{r_j\}$). Dans notre exemple, $j = 2$, $r_j = 4$ et $S' = \{1,2,3,6,8,13\}$). On considère alors l'arborescence décroissante A'' codée par le couple (A'_0,S') (qui a donc n pour racine). Cette arborescence A'' est greffée à l'arborescence A_j (par identification du sommet n) pour donner l'arborescence A'_j.



On dispose ainsi d'un ensemble de k arborescences, $(A_1,\dots,A_{j-1},A'_j,A_{j+1},\dots,A_k)$, tel que la racine de A'_j est inférieure à ses fils et A'_j contient toutes les étiquettes de A_0. On va alors rassembler ces arborescences en une seule arborescence.

Étape 3. Chacune des arborescences restantes A_i ($i > 0$, $i \neq j$), est greffée à A'_j en identifiant la racine de l'arborescence avec le sommet de même étiquette dans A'_j, ce qui donne par construction une arborescence appartenant à $\mathcal{A}_{n,0}$.



Cette construction possède clairement la propriété suivante, que nous utilisons dans la section suivante.

Propriété 19. Le nombre d'arêtes décroissantes dans $\phi(A)$ est égal au nombre d'arêtes décroissantes dans A.

Pour prouver le lemme 17, il suffit alors de décrire l'inverse de la construction précédente, qui consiste à reprendre ses 3 étapes. Soit A' une arborescence de $\mathcal{B}_n$ de racine r.

1. Dans A', soit A'' la sous-arborescence décroissante maximale enracinée en n. En supprimant toutes les arêtes de cette sous-arborescence, on obtient l'ensemble $(A'',A_1,$

$\dots, A_j, \dots, A_k$).

2. Si A'' est codée par le couple (A'_0, S'), on considère l'arborescence A_0 correspondant au couple (A'_0, S), où $S = \{r\} \cup S'/\{n\}$.
3. Finalement, la construction de A à partir de $(A_0, \dots, A_j, \dots, A_k)$ est immédiate.

2.1.2 Le cas général

Soit $\mathcal{F}_{n,k}$ l'ensemble des forêts sur $[n]$ constituées de k arborescences dans lesquelles n est une feuille. Le cardinal de $\mathcal{F}_{n,k}$ est clairement égal au nombre d'arborescences sur $[n+1]$ dont la racine d'étiquette 1 a k fils, et dans lesquelles $n+1$ est une feuille.

Par application du codage de Prüfer, on sait que ces arborescences sont codées par des mots de longueur n sur l'alphabet $[n+1]$ possédant k occurrences de 1 (dont une comme dernière lettre du mot) et aucune occurrence de $n+1$ (qui est une feuille). On a donc :

$$|\mathcal{F}_{n,k}| = \binom{n-1}{k-1}(n-1)^{n-k}. \tag{2.3}$$

La preuve du théorème 16 est alors une conséquence directe du lemme suivant et de (2.3).

Lemme 20. *Il existe une bijection Φ entre $\mathcal{F}_{n,k+1}$ et $\mathcal{A}_{n,k}$.*

La suite de ce paragraphe est consacrée à la preuve de ce lemme. Soit $F \in \mathcal{F}_{n,k+1}$, c'est-à-dire une forêt de $(k+1)$ arborescences $(A_1, A_2, \dots, A_{k+1})$ sur $[n]$, de racines respectives $r_1, r_2, \dots, r_{k+1}$, avec $r_1 < r_2 < \cdots < r_{k+1}$. Soit A_j l'arborescence de F contenant la feuille étiquetée n. On distingue alors deux cas.

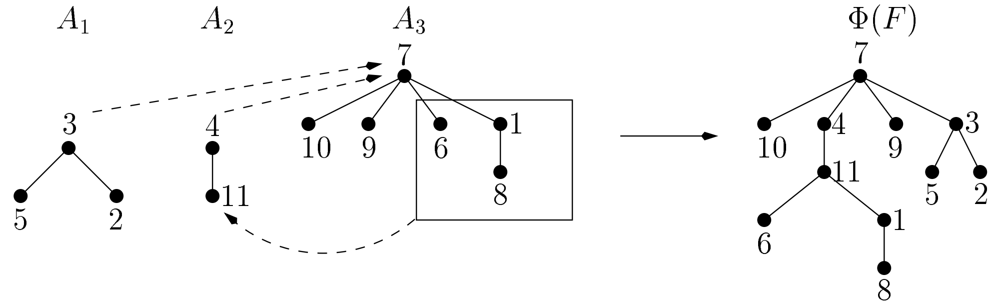
Cas 1. Si

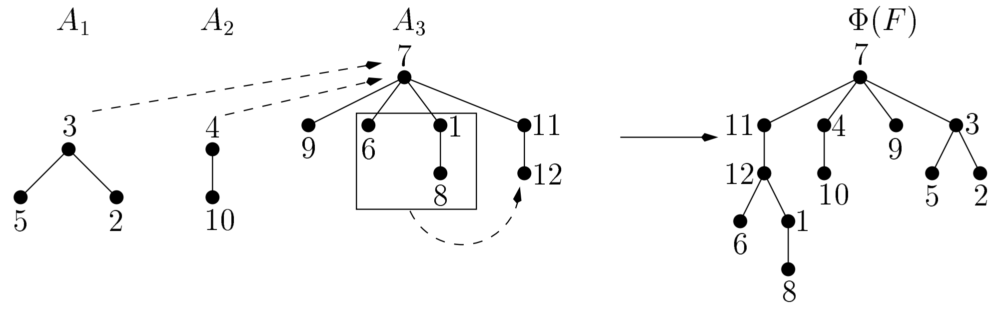
– $r_{k+1} \neq r_j$,

ou

– $r_{k+1} = r_j$ et le chemin allant de r_j à n débute par une arête croissante,

alors $\Phi(F)$ est obtenue, à partir de F, en greffant sous la feuille n toutes les sous-arborescences de A_{k+1} dont les racines sont les fils de r_{k+1} qui lui sont inférieurs, puis en greffant sous le sommet r_{k+1} les k arborescences restantes, à savoir $A_1, A_2, \dots, A_k$. On obtient alors une arborescence appartenant à $\mathcal{A}_{n,k}$. Les deux exemples qui suivent illustrent ces deux situations et la transformation résultante.



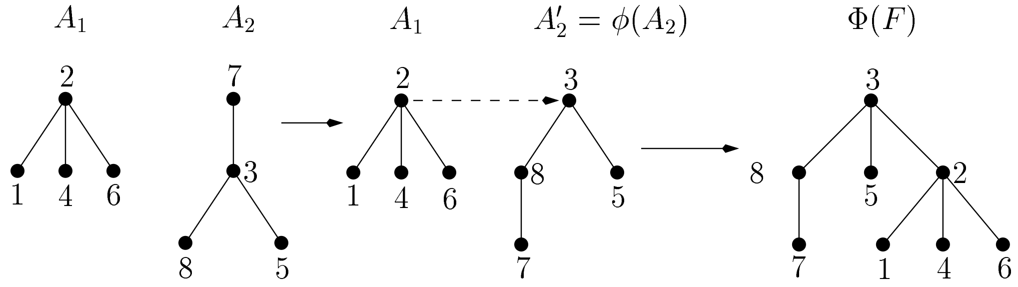


Propriété 21. Dans l'arborescence $\Phi(F)$ ainsi obtenue, tous les fils de n sont inférieurs à la racine r.

Cas 2. $r_{k+1} = r_j$ et le chemin allant de r_{k+1} à n commence par une arête décroissante.

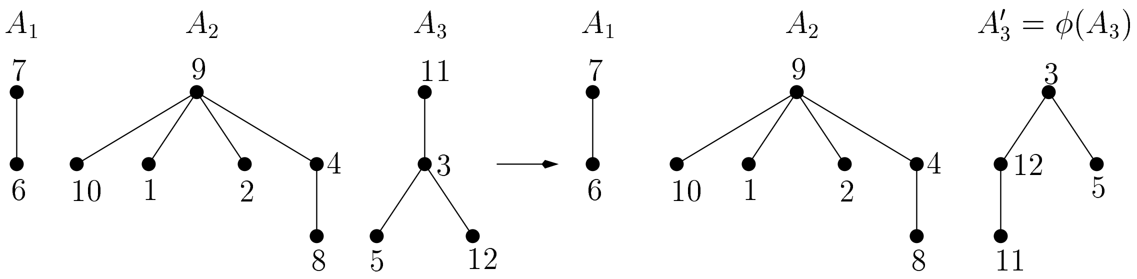
Dans un premier temps, n étant une feuille, on applique à A_{k+1} la bijection ϕ définie précédemment pour le cas $k = 0$, et on note A'_{k+1} l'arborescence ainsi obtenue et r'_{k+1} sa racine qui, par construction, est plus petite que tous ses fils. On distingue alors deux sous-cas.

Sous-cas 2.a. Si $r'_{k+1} > r_i$ pour $i = 1,2,\dots,k$, alors $\Phi(F)$ est obtenue, à partir de F, en greffant sous r'_{k+1} les k arborescences $A_1,A_2,\dots,A_k$.



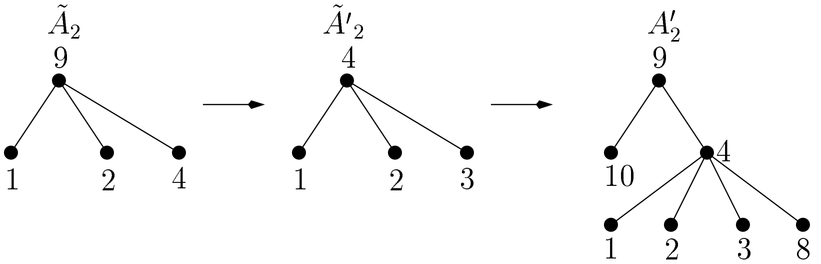
Propriété 22. Soit r la racine de l'arborescence $\Phi(F)$ ainsi obtenue. Au moins un fils de n est supérieur à r. De plus, le chemin allant de r à n débute par une arête croissante.

Sous-cas 2.b. Sinon, $r_k > r'_{k+1}$, et r_k est donc la plus grande racine de la forêt constituée de $A_1,A_2,\dots,A_k,A'_{k+1}$.

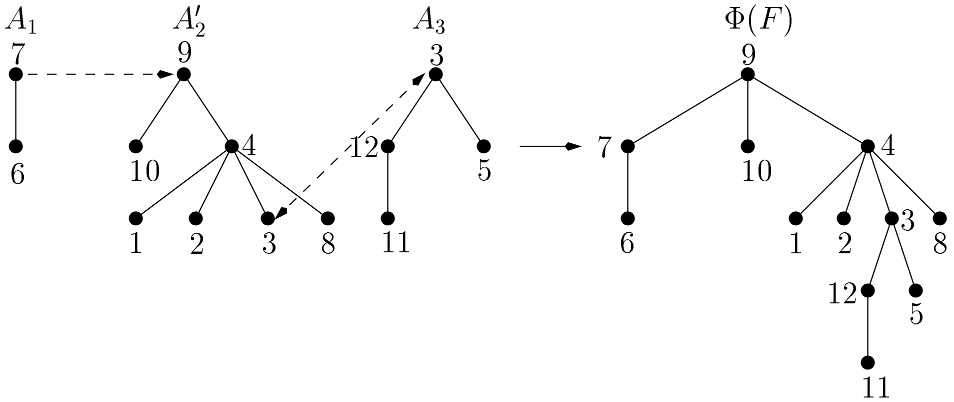


On procède en deux étapes. Soit $\tilde{A}_k$ la sous-arborescence décroissante maximale de A_k enracinée en r_k. Comme on l'a vu dans la description de ϕ, on peut coder $\tilde{A}_k$ par un couple $(\tilde{A}'_k,S)$. On définit alors $S' = \{r'_{k+1}\} \cup S/\{r_k\}$ et A'_k comme l'arborescence A_k où les arêtes de $\tilde{A}_k$ ont été supprimées et la sous-arborescence décroissante codée par $(\tilde{A}'_k,S')$

a été greffée sous r_k, les autres arêtes restant inchangées. Dans notre exemple, on a donc $S = \{1,2,4,9\}$, $S' = \{1,2,3,4\}$ et on effectue les transformations suivantes.



Dans un deuxième temps, on greffe l'arborescence A'_{k+1} sous r'_{k+1} (dans A'_k donc) et, toujours dans A'_k, on greffe sous r_k les arborescences restantes, à savoir $A_1, A_2, \ldots, A_{k-1}$. On obtient alors une arborescence de $\mathcal{A}_{n,k}$.



Propriété 23. Soit r la racine de l'arborescence $\Phi(F)$ ainsi obtenue. Cette racine possède k fils qui lui sont inférieurs et au moins un fils de n est supérieur à r. De plus, le chemin allant de r à n débute par une arête décroissante.

Pour définir la construction inverse Φ^{-1}, il suffit de remarquer que toute arborescence vérifie une et une seule des propriétés 21, 22 ou 23. Soit donc A une arborescence de $\mathcal{A}_{n,k}$, de racine r.

Cas 1. Si A vérifie la propriété 21, en supprimant les arêtes entre r et ses fils qui lui sont inférieurs, puis en transférant les fils de n sous r, on obtient la forêt $F = \Phi^{-1}(A)$.

Cas 2. Si A vérifie la propriété 22, on commence par supprimer les arêtes reliant r et ses fils qui lui sont inférieurs. On obtient alors une forêt dont l'arborescence de racine r est telle que cette racine est inférieure à tous ses fils. Il suffit alors d'appliquer ϕ^{-1} à cette arborescence pour obtenir $F = \Phi^{-1}(A)$.

Cas 3. Si A vérifie la propriété 23, on commence par supprimer les arêtes reliant r et ses fils qui lui sont inférieurs, sauf celle menant vers n (on note x le fils de r extrémité de cette arête). Ensuite, soit y le premier sommet sur le chemin de x à n tel que tous ses fils lui soient supérieurs. On extrait la sous-arborescence de racine y. Ensuite on applique la transformation décrite dans le sous-cas 2.b à l'arborescence de racine r puis ϕ^{-1} à l'arborescence de racine y.

2.2 Quelques identités

Dans cette section, nous utilisons les résultats précédents pour donner des interprétations combinatoires de plusieurs identités, la plupart nouvelles à notre connaissance.

Étiquette de la racine. Le corollaire 8 nous a permis de rappeler que, pour un entier $r \in [n]$ fixé, le nombre d'arborescences sur $[n]$ de racine r est n^{n-2}. Nous proposons maintenant un résultat analogue pour les arborescences de $\mathcal{A}_{n,k}$.

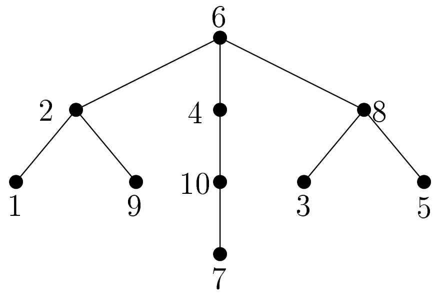
Théorème 24. *Le nombre d'arborescences de $\mathcal{A}_{n+1,k}$ de racine $r+1$ $(r = k,k+1,\ldots,n)$ est*

$$(n+1)^{n-r-1}n^{r-k-1}\binom{r-1}{k-1}\left(\frac{(n-r)(r-k)}{k}+n(n+1)\right),\ \textit{si } k>0, \tag{2.4}$$

$$(n+1)^{n-r-1}n^{r-1}(n-r),\ \textit{si } k=0. \tag{2.5}$$

Preuve. Nous prouvons ce résultat par une variante du codage de Prüfer. Le codage de Prüfer, présenté au chapitre précédent, met en correspondance une arborescence sur $[n]$ et un mot de longueur $(n-1)$ sur l'alphabet $[n]$. Dans la variante que nous proposons, nous mettons en correspondance une arborescence sur $[n]$ et un couple (m,M) de deux mots sur l'alphabet $[n]$, de longueurs cumulées $(n-1)$. Dans un premier temps, nous décrivons cette variante du codage de Prüfer, puis nous énumérons les couples de mots en correspondance avec une arborescence $\mathcal{A}_{n+1,k}$ de racine $r+1$.

Le principe guidant notre variante du code de Prüfer (voir algorithme 1) est que lorsque l'on retire une arête d'une arborescence (l'arête (y,x) contenant la plus grande feuille restante x), on insère y dans le mot m si x est inférieur à la racine et dans le mot M sinon.



$\longleftrightarrow$ $m = 8.6.8.2.6$, $M = 2.10.4.6$

Pour une arborescence de $\mathcal{A}_{n+1,k}$ de racine $r+1$, le couple (m,M) obtenu en appliquant cet algorithme vérifie les propriétés suivantes (pour $k \geq 0$) :

(a) $|m| = r$ et $|M| = n-r$,

(b) il y a exactement k occurrences de $r+1$ dans m (car $r+1$ a exactement k fils qui lui sont inférieurs),

(c) si x est la dernière lettre de m, $x \geq r+1$ ($x \geq r+2$ si $k=0$ car dans ce cas $r+1$ n'apparaît pas dans m),

(d) $r+1$ est la dernière lettre de m ou la dernière lettre de M (éventuellement des deux mots), et uniquement la dernière lettre de M si $k=0$.

Pour construire une arborescence A de $\mathcal{A}_{n+1,k}$ de racine $r+1$ à partir d'un couple (m,M) vérifiant ces propriétés, on applique le principe de l'algorithme 2. Soit F l'ensemble des éléments de $[n+1]$ n'apparaissant ni dans M ni dans m : tant que l'un des deux mots M

ou m n'est pas vide, retirer le plus grand élément x de F et si $x > r+1$ (resp. $x < r+1$), retirer de M (resp. m) sa première lettre y, l'ajouter à F si y n'apparaît plus ni dans M ni dans m, et ajouter une arête de y vers x dans A.

Il reste alors à compter les couples de mots vérifiant les propriétés ci-dessus. On distingue deux cas :

1. si la dernière lettre de m est $r+1$, il faut choisir $k-1$ positions dans m, parmi $r-1$ (cf. (a)), recevant les $k-1$ occurrences restantes de $r+1$ (cf. (b)), puis remplir les positions restantes de m avec la seule contrainte que $r+1$ n'y apparaît pas et il n'y a aucune contrainte sur le mot M (cf. (d)), ce qui donne
$$\binom{r-1}{k-1} n^{r-k}(n+1)^{n-r} \tag{2.6}$$
couples (m,M) possibles dans ce cas là ;
2. si la dernière lettre de m est différente de $r+1$, il faut choisir k positions dans m, parmi $r-1$, recevant les k occurrences de $r+1$, puis choisir la dernière lettre de m parmi $\{r+2,\dots,n+1\}$ (cf. (c)), remplir les positions restantes de m sans y faire apparaître $r+1$ et enfin la seule contrainte sur M est que sa dernière lettre est $r+1$, ce qui donne
$$\binom{r-1}{k}(n-r)n^{r-k-1}(n+1)^{n-r-1} \tag{2.7}$$
couples (m,M) possibles dans ce cas.

La formule (2.4) s'obtient en sommant les deux expressions (2.6) et (2.7). Pour obtenir (2.5), il suffit de remarquer que seul le cas 2 intervient lorsque $k=0$. □

En considérant le cas $k=0$ du théorème 24 et en sommant sur les valeurs possibles pour r, on obtient alors une preuve combinatoire de l'identité suivante.

Corollaire 25.
$$n^n = \sum_{r=0}^{n-1}(n-r)(n+1)^{n-r-1}n^{r-1}.$$

Degré de la racine. Dans le même esprit, on peut étudier la distribution des arborescences de $\mathcal{A}_{n+1,k}$ selon le degré de la racine.

Proposition 26. *Le nombre d'arborescences de $\mathcal{A}_{n+1,k}$ dont la racine a d fils ($d \geq k$) est*
$$\binom{n+1}{d+1}k(d-k)\sum_{s=0}^{n-d}\binom{n-d}{s}(s+k)^{s-1}(n-s-k)^{n-s-d-1},\ \textit{si}\ k>0, \tag{2.8}$$

$$\binom{n+1}{d+1}d\ n^{n-d-1},\ \textit{si}\ k=0. \tag{2.9}$$

Preuve. Une arborescence de $\mathcal{A}_{n+1,k}$ de racine r de degré d peut être décomposée en 3 parts :

– un sommet isolé (la racine r),

– un ensemble de k arborescences, comportant $s+k$ sommets, de racines $r_1 < \cdots < r_k < r$,
– un ensemble de $d-k$ arborescences, comportant $n-k-s$ sommets, de racines $r < t_1 < \cdots < t_{d-k}$.

On a vu (théorème 11) que le nombre de forêts de Cayley à n sommets et k arborescences de racines fixées est $k\, n^{n-k-1}$ si $k > 0$. On en déduit qu'après avoir choisi les $d+1$ sommets correspondant aux r_i, aux t_i et la racine r, et s sommets différents des r_i, des t_i et de r pour les arborescences de racines $r_1,\ldots,r_k$, on a deux forêts à construire :
– une forêt de k arborescences de racines $r_1,\ldots,r_k$ à $s+k$ sommets,
– une forêt de $d-k$ arborescences de racines $t_1,\ldots,t_{d-k}$ à $n-s-k$ sommets,

ce qui prouve (2.8). Lorsque $k=0$, le même raisonnement permet d'obtenir (2.9) : choix de $d+1$ sommets et construction d'une forêt de d arborescences sur n sommets. □

Remarque 27. En comparant le résultat pour $k=0$ avec son analogue pour les arborescences sans condition sur la racine (théorème 9), on remarque la relation suivante, quelque peu surprenante à première vue, entre n^n et $(n+1)^n$:

$$\begin{aligned} (n+1)^n &= \sum_{d=1}^{n} \frac{(n+1)}{n} \binom{n}{d}\, d\, n^{n-d-1}, \\ n^n &= \sum_{d=1}^{n} \frac{(n+1)}{(d+1)} \binom{n}{d}\, d\, n^{n-d-1}. \end{aligned}$$

Nombre d'arêtes décroissantes. Nous nous intéressons maintenant successivement au nombre d'arêtes décroissantes dans $\mathcal{A}_{n+1,0}$, puis aux arborescences de $\mathcal{A}_{n+1,k}$ ayant exactement k arêtes décroissantes, ce qui nous permet d'obtenir une identité, semble-t-il nouvelle, concernant les *nombres de Stirling de première espèce.* Pour une présentation détaillée des nombres de Stirling, on pourra se reporter à [82, section 6.1].

Définition 28. Les nombres de Stirling de première espèce $s(n,m)$ (pour $n,m > 0$) sont définis par

$$s(n,m) = (-1)^{n-m} c(n,m),$$

où $c(n,m)$ est le nombre de permutations de $[n]$ comportant m cycles.

Proposition 29. *Le nombre d'arborescences de $\mathcal{A}_{n+1,0}$ ayant k arêtes décroissantes ($k = 0,1,\ldots,n-1$) est*

$$(2.10) \qquad (-1)^{k+1} \sum_{m=k+1}^{n} \binom{m}{k+1} s(n+1,n+1-m)\; n^{n-m}.$$

Preuve. Pour prouver ce résultat, nous introduisons les notions *d'endofonction* et de *descente* dans une endofonction :
– une *endofonction* sur $[n]$ est une fonction de $[n]$ dans $[n]$;
– une *descente* dans une endofonction f est un entier $x \in [n]$ tel que $f(x) \geq x$.

Dans [100], Khidr et El-Desouky prouvent (bien que le résultat ne soit pas exprimé en termes d'endofonctions, mais en termes de chemins dans des matrices carrées à valeurs dans $\{0,1\}$) que le nombre d'endofonctions sur $[n]$ à k descentes est

$$(-1)^k \sum_{m=k}^{n} \binom{m}{k} \; s(n+1,n+1-m) \; n^{n-m}.$$

Dans [96] (voir aussi G. Labelle [108]), Joyal décrit une bijection entre les endofonctions sur $[n]$ et les arborescences sur $[n]$ ayant un sommet distingué (ou de manière équivalente les arborescences sur $[n+1]$ telles que le sommet $n+1$ est une feuille), cette bijection ayant la propriété que le nombre de descentes d'une endofonction est égal au nombre d'arêtes décroissantes de l'arborescence correspondante plus 1. La proposition découle alors directement de ces deux résultats, de la bijection ϕ (lemme 17) et de la propriété 19. □

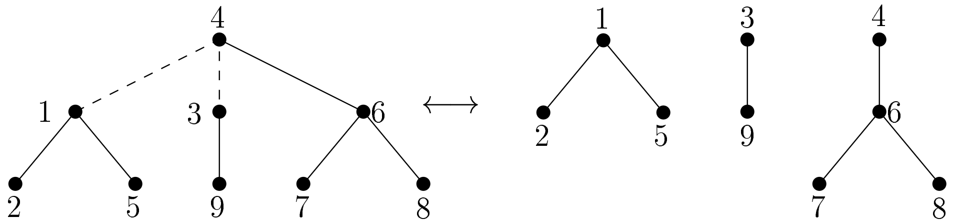
Proposition 30. *Le nombre d'arborescences de $\mathcal{A}_{n+1,k}$ ayant k arêtes décroissantes (toutes les arêtes décroissantes partent donc de la racine), pour $k = 0,1,\ldots,n$, est*

$$(-1)^{n-k} \; s(n+1,k+1). \tag{2.11}$$

Preuve. La preuve repose sur les résultats classiques suivants :

- le nombre d'arborescences croissantes sur $[n+1]$ est égal au nombre de permutations de $[n+1]$ ayant un seul cycle, c'est-à-dire $n!$ (dans une arborescence croissante sur $[n]$ on peut greffer le sommet $n+1$ comme feuille en exactement n positions différentes, ce qui, par induction, prouve que le nombre d'arborescences croissantes sur $[n+1]$ est $n!$) ;
- le nombre de permutations sur $[n+1]$ ayant m cycles est donné par $(-1)^{n+1-m}s(n+1,m)$ (définition 28).

Il suffit alors de remarquer qu'une arborescence de $\mathcal{A}_{n+1,k}$ ayant exactement k arêtes décroissantes est équivalente à une forêt de $k+1$ arborescences croissantes sur $[n+1]$ (en supprimant les arêtes reliant la racine à ses fils qui lui sont inférieurs) et le résultat en découle directement.



□

En posant $k = 0$ dans les propositions 29 et 30, on obtient alors l'identité suivante.

Corollaire 31.

$$(-1)^{n-1}s(n+1,1) = \sum_{m=1}^{n} m \; s(n+1,n+1-m) \; n^{n-m} = n!.$$

Sur une identité de Goulden et Jackson. Dans ce dernier paragraphe, nous nous intéressons à l'identité (2.2). Dans [79], Goulden et Jackson démontrent un résultat de Hurwitz [92] (voir aussi les travaux de V. Strehl [151]) exprimant le nombre de factorisations minimales transitives, en produit de transpositions, d'une permutation de type cyclique donné. Ce résultat a été étendu par M. Bousquet-Mélou et G. Schaeffer [25, 135].

La preuve de Goulden et Jackson est basée sur une utilisation astucieuse de la formule d'inversion de Lagrange, et fait notamment apparaître une identité intrigante concernant n^n.

Proposition 32. *Soit $T_{k,m}$ ($k \geq 1$ et $m \geq 1$) donné par la formule suivante (dans laquelle on suppose que $0^0 = 1$) :*

$$T_{k,m} = \frac{k^{k+1}}{k!} \sum_{i=1}^{m} \frac{i^i}{i!} \; \frac{(m-i)^{m-i}}{(m-i)!} \; \frac{1}{k+m-i}.$$

On a alors

$$(k+m)^{k+m} = (k+m)! \; (T_{k,m} + T_{m,k}). \tag{2.12}$$

La preuve de cette identité exposée dans [79] étant purement formelle (voir le mémoire de D. Poulalhon [126] pour un exposé plus détaillé de cette preuve), la question d'une explication combinatoire de cette identité est toujours ouverte. Une telle preuve serait notamment utile pour espérer obtenir une preuve bijective du résultat d'Hurwitz. Dans ce paragraphe, nous donnons une preuve combinatoire pour le cas $m = 1$, avant de reprendre la preuve de Goulden et Jackson pour mettre en évidence une propriété de $T_{k,m}$ pouvant mener à une preuve bijective du cas général.

Dans le cas $m = 1$, l'identité (2.12) se simplifie et on obtient l'identité (2.2) :

$$n^n = n(n-1)^{n-1} + \sum_{k=1}^{n-1} \binom{n}{k} \; k^k \; (n-1-k)^{n-1-k}.$$

Preuve de l'identité (2.2). On rappelle que le membre gauche de cette identité est le cardinal de $\mathcal{A}_{n+1,0}$.

Le membre droit de cette expression se compose de deux parties que nous traitons séparément: $n(n-1)^{n-1}$ et la sommation.

D'après le théorème 16, le nombre d'arborescences de $\mathcal{A}_{n+1,0}$ telles que $n+1$ est une feuille est clairement égal à $n(n-1)^{n-1}$, ce qui fournit un explication de la première partie de (2.2).

Considérons maintenant les arborescences de $\mathcal{A}_{n+1,0}$ pour lesquelles $n+1$ est un nœud. On note $\mathcal{R}_{n+1,k+1}$ l'ensemble des couples (A_1,A_2) d'arborescences telles que :

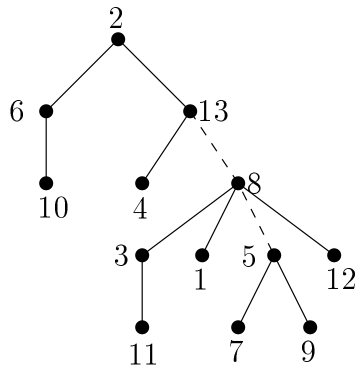
- la racine de A_1 (resp. A_2) est plus petite que ses fils,
- les ensembles S_1 et S_2 des étiquettes de A_1 et A_2 forment une partition de $[n+1]$, avec $|S_1| = k+1$, $|S_2| = n-k$ et $n+1 \in S_1$ ($n+1$ est donc un sommet de A_1).

Le théorème 16 nous permet d'affirmer que, pour $k = 1, \ldots, n-1$,

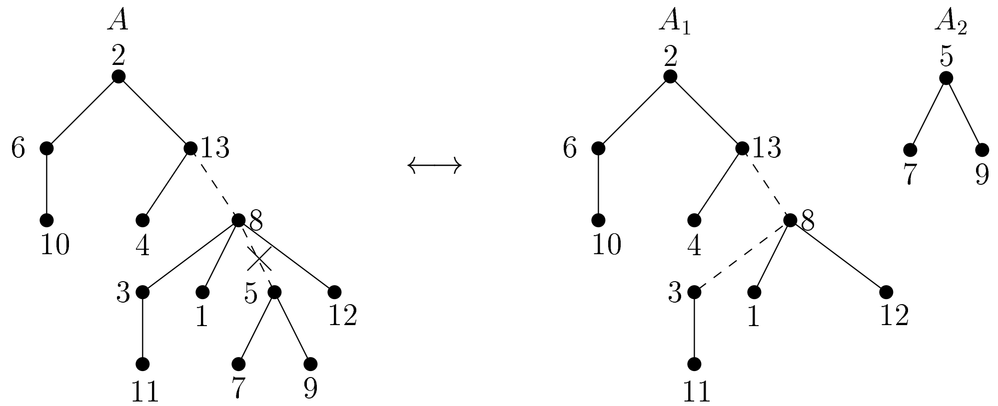
$$|\mathcal{R}_{n+1,k+1}| = \binom{n}{k} \; k^k \; (n-1-k)^{n-1-k}.$$

Pour prouver (2.2), il suffit donc d'établir une bijection entre les arborescences de $\mathcal{A}_{n+1,0}$ telles que $n+1$ est un nœud et l'union des $\mathcal{R}_{n+1,k+1}$, pour $k = 1, \ldots, n-1$.

Nous décrivons maintenant une telle bijection, qui utilise la notion de *branche décroissante maximale* issue d'un sommet : la *branche décroissante maximale* d'un sommet x appartenant à une arborescence A est l'ensemble des sommets $(x_0,x_1,x_2,\dots,x_j)$ tels que $x_0 = x$ et x_i est le plus grand fils de x_{i-1} inférieur à x_{i-1} (pour $i = 1,\dots,j$). Dans la figure suivante par exemple, les arêtes de la branche décroissante maximale de 13, sont hachurées, et cette branche est constituée des sommets (13,8,5).



Soient A une arborescence de $\mathcal{A}_{n+1,0}$ telle que $n+1$ est un nœud et $(x_0,\dots,x_j)$ la branche maximale décroissante de $x_0 = n+1$ dans A. En supprimant l'arête reliant x_j à son père x_{j-1} (comme $n+1$ est un nœud, on a toujours $j \geq 1$), on obtient un couple (A_1,A_2) d'arborescences (où A_1 est l'arborescence contenant le sommet $n+1$). Si A_1 contient $k+1$ sommets ($k \geq 1$ car A_1 contient $n+1$, qui ne peut être racine, et $k \leq n-1$ car A_2 est non vide), il est clair que $(A_1,A_2) \in \mathcal{R}_{n+1,k+1}$.



La construction inverse, permettant de reconstruire A à partir de (A_1,A_2) est très simple. Soit $(x_0,\dots,x_l)$ la branche maximale décroissante de $x_0 = n+1$ dans A_1. En parcourant cette branche de x_0 vers x_l, on greffe A_2 comme fils du premier sommet x_j tel que la racine r_2 de A_2 est supérieure à x_{j+1} et inférieure à x_j (si $x_j = x_l$, on greffe A_2 sous x_l). Cette construction bijective conclut la preuve de (2.2). $\square$

Pour conclure ce paragraphe consacré à l'identité (2.12), nous proposons une "relecture" de la preuve de Goulden et Jackson, qui suggère une "piste" pour une preuve bijective de cette identité.

On peut réécrire $T_{k,m}$ de la façon suivante.

$$T_{k,m} = \sum_{i+j=m,\ i\geq 1,\ j\geq 0} \frac{i^i}{i!}\frac{j^j}{j!}\frac{k^k}{k!}\frac{k}{k+j}.$$

On introduit alors la série génératrice ordinaire bivariée des coefficients $T_{k,m}$:

$$T(x,y) = \sum_{k,m\geq 1} T_{k,m} x^k y^m.$$

Considérons

$$u(x) = \sum_{i\geq 1} i^{i-1} \frac{x^i}{i!},$$

la série génératrice exponentielle des arborescences de Cayley non vides,

$$\tilde{u}(x) = \sum_{i\geq 1} i^i \frac{x^i}{i!} = x \frac{\partial u}{\partial x}(x),$$

la série génératrice exponentielle des arborescences de Cayley pointées (un deuxième sommet, qui peut être la racine, est distingué) non vides, et

$$\hat{u}(x) = \sum_{i\geq 0} i^i \frac{x^i}{i!} = 1 + \tilde{u}(x).$$

Le terme "gênant" dans $T_{k,m}$ est le facteur $\frac{k}{k+j}$. Pour l'éliminer, on introduit une nouvelle variable formelle q et on considère la série $T(xq,yq)$. On a alors

$$T(xq,yq) = \sum_{k,m\geq 1} T_{k,m} x^k y^m q^{k+m}$$

$$\Longrightarrow T(xq,yq) = \sum_{k\geq 1} k^{k+1} \frac{x^k q^k}{k!} \left(\sum_{i\geq 1} i^i \frac{y^i q^i}{i!} \right) \left(\sum_{j\geq 0} j^j \frac{y^j q^j}{j!} \frac{1}{k+j} \right)$$

$$\Longrightarrow T(xq,yq) = \tilde{u}(yq) \sum_{k\geq 1, j\geq 0} k^{k+1} \frac{x^k}{k!} j^j \frac{y^j}{j!} \frac{q^{k+j}}{k+j}$$

$$\Longrightarrow T(xq,yq) = \tilde{u}(yq) \sum_{k\geq 1, j\geq 0} k^{k+1} \frac{x^k}{k!} j^j \frac{y^j}{j!} \int_0^q t^{k+j-1} dt$$

$$\Longrightarrow T(xq,yq) = \tilde{u}(yq) \int_0^q \sum_{k\geq 1, j\geq 0} k^{k+1} \frac{x^k}{k!} j^j \frac{y^j}{j!} \; t^{k+j-1} dt$$

$$\Longrightarrow T(xq,yq) = \tilde{u}(yq) \int_0^q \hat{u}(yt) \left(\sum_{k\geq 1} k^{k+1} \frac{x^k}{k!} t^{k-1} \right) dt$$

$$T(xq,yq) = \tilde{u}(yq) \int_0^q \hat{u}(yt) \frac{\partial \tilde{u}(xt)}{\partial t} dt.$$

On peut alors relier cette expression à une construction classique sur les séries génératrices exponentielles, le *boxed product* (voir par exemple [61]).

Lemme 33. *Soient $\mathcal{U}$ et $\mathcal{V}$ deux familles d'objets étiquetés de séries génératrices exponentielles respectives $U(x)$ et $V(x)$, et $\mathcal{W}$ la famille des objets $W = (U,V)$ (W est un couple de deux objets U et V appartenant respectivement à $\mathcal{U}$ et $\mathcal{V}$) tels que la plus petite étiquette de W appartient à U. La série génératrice $W(x)$ de $\mathcal{W}$ vérifie alors la relation*

$$W(x) = \int_0^x V(t)\frac{\partial U(t)}{\partial t}dt.$$

Souhaitant montrer que

$$[x^k y^m q^{k+m}](T(xq,yq) + T(yq,xq)) = \frac{(k+m)^{k+m}}{(k+m)!},$$

une preuve combinatoire de cette identité nécessiterait de mettre en évidence deux familles $\mathcal{F}$ et $\mathcal{G}$ d'objets vérifiant les propriétés suivantes.

- Pour $k,m \geq 1$, on a $(k+m)^{k+m}$ objets de $\mathcal{F}$ étiquetés sur $[k+m]$ et pour chacun de ces objets F, il existe une partition canonique de ses étiquettes en deux ensembles, notés $E_1(F)$ et $E_2(F)$, de tailles respectives k et m : la série génératrice trivariée de $\mathcal{F}$ est alors
$$\sum_{k,m\geq 1} (k+m)^{k+m} \frac{x^k y^m q^{k+m}}{(k+m)!}.$$
- Il existe deux familles $\mathcal{G}_1$ et $\mathcal{G}_2$ de structures étiquetées, de séries génératrices respectives $\tilde{u}(x)$ et $\hat{u}(x)$, telles que
  - tout objet $G \in \mathcal{G}$ sur $[k+m]$ est un triplet (X,Y,Z) vérifiant $X,Z \in \mathcal{G}_1$ et $Y \in \mathcal{G}_2$
  - on peut partitionner les étiquettes de G en deux ensembles notés $E_1(G)$ et $E_2(G)$, de tailles respectives k et m,
  - les étiquettes de $E_1(G)$ sont les étiquettes de X et Y et celles de $E_2(G)$ celles de Z,
  - la plus petite étiquette de X et Z appartient à Z.

Il faudrait alors établir une bijection entre $\mathcal{F}$ et $\mathcal{G}$ vérifiant la propriété suivante : si $F \in \mathcal{F}$ est en bijection avec $G \in \mathcal{G}$, alors $|E_1(F)| = |E_1(G)|$ et $|E_2(F)| = |E_2(G)|$.

2.3 Factorisations du grand cycle

Dans [53], Dénes a établi une correspondance entre arborescences de Cayley sur $[n]$ de racine 1 et factorisations minimales de la permutation circulaire $(1,2,\ldots,n)$ (que nous appelons le *grand cycle*) comme produit de transpositions. Dans cette section, nous proposons une relecture de la preuve du résultat de Dénes due à Moszkowski [122] qui nous permet de relier la famille $\mathcal{A}_{n+1,0}$ à ce problème. On rappelle qu'une transposition sur $[n]$ est une permutation sur $[n]$ qui échange exactement deux éléments, c'est-à-dire une permutation comportant $n-1$ cycles (et donc un cycle de longueur 2 et $(n-2)$ cycles de longueur 1, les *points fixes*).

Définition 34. Soit σ une permutation circulaire sur $[n]$. Une *factorisation minimale* de σ en produit de transpositions est un $(n-1)$-uplet $(\tau_1,\ldots,\tau_{n-1})$ de transpositions sur $[n]$ tel que $\tau_1 \cdots \tau_{n-1} = \sigma_0$, le produit étant effectué de gauche à droite.

Dans la suite de cette section, nous utilisons le terme factorisation pour désigner une factorisation minimale en produit de transpositions. Par exemple, le produit $\tau_1.\tau_2.\tau_3.\tau_4$ avec $\tau_1 = (3,5)$, $\tau_2 = (3,4)$, $\tau_3 = (2,4)$, $\tau_4 = (1,2)$, est une factorisation de $\sigma = (1,2,4,3,5)$.

Théorème 35. [53] *Le nombre de factorisations minimales de* $(1,2,\ldots,n)$ *est* n^{n-2}.

La preuve donnée par Dénes repose sur la représentation d'un produit de transpositions par un graphe. Soit $\tau_1 \cdots \tau_k$ un produit de k transpositions sur $[n]$. On peut le représenter par un graphe à n sommets étiquetés dans $[n]$ et k arêtes étiquetées dans $[k]$: si $\tau_i = (x,y)$, on a alors une arête étiquetée i reliant x à y.

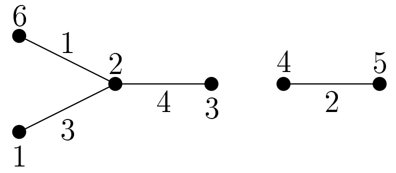


FIG. 2.1 – Graphe associée à (2,6).(4,5).(1,2).(2,3)

Dans [53] (voir aussi les travaux d'Eden et Schützenberger [56]), Dénes montre que la permutation $\sigma = \tau_1 \cdots \tau_{n-1}$ est un cycle de longueur n si et seulement si ce graphe est une arborescence (on prend 1 pour racine). Par exemple, le produit des 5 transpositions (4,5).(1,6).(2,4).(3,6).(1,4) est une factorisation de (1,3,6,4,5,2) et correspond à l'arborescence représentée dans la figure suivante.

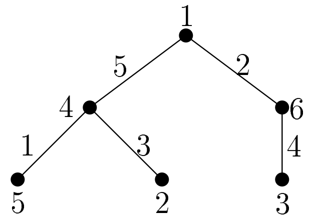


FIG. 2.2 – Arborescence associée à (4,5).(1,6).(2,4).(3,6).(1,4)

On a donc une correspondance entre les factorisations de permutations circulaires sur $[n]$ et les arborescences doublement étiquetées, sur $[n]$ pour les sommets et $[n-1]$ pour les arêtes, de racine 1. Clairement, ces arborescences sont au nombre de $(n-1)!n^{n-2}$. D'autre part, le nombre de permutations circulaires sur $[n]$ est $(n-1)!$. Le résultat découle directement de ces deux observations.

Alors que le théorème 35 suggère qu'il existe une bijection directe entre les factorisations du grand cycle et les arborescences de Cayley de racine 1, la preuve précédente est basée sur une correspondance entre deux ensembles plus larges (factorisations des n-cycles et arborescences doublement étiquetées). La première preuve bijective directe du théorème 35 est due à Moszkowski [122] (voir aussi les travaux de Goulden et Pepper [81]). Nous proposons une nouvelle description de la bijection de Moszkowski, qui présente les avantages suivants.

- Elle est plus intuitive, car basée uniquement sur des manipulations d'arborescences : la preuve de sa validité est immédiate contrairement à la preuve originale de Moszkowski.

– Elle permet de donner une interprétation de n^n en termes de factorisations du grand cycle.

La bijection de Moszkowski. Cette bijection s'appuie sur la propriété suivante que vérifie une arborescence représentant une factorisation de $(1,2,\ldots,n)$ (propriété découlant directement de la définition d'une factorisation du grand cycle).

Propriété 36. Soit A une arborescence sur $[n]$ doublement étiquetée et de racine 1. Pour un sommet x de A, on note $C_x = (a_0,\ldots,a_k)$ la suite des arêtes croissantes constituant le plus long chemin issu de x où a_0 est la plus petite arête incidente à x (éventuellement l'arête reliant x à son père) et, si l'arête a_j $(j < k)$ joint y à z, alors a_{j+1} est la plus petite arête incidente à z supérieure à a_j. Ainsi A représente une factorisation du grand cycle si et seulement, pour tout sommet x de A, le sommet terminal de C_x est $x+1$.

Cette propriété des arborescences associées à une factorisation du grand cycle induit la description suivante de la bijection de Moszkowski, basée sur un réétiquetage des sommets de A.

Algorithme 3: Des arborescences vers les factorisations

Données : A: arborescence de racine 1
début
  pour chaque sommet x de A différent de 1 **faire**
    étiqueter l'arête reliant x à son père par $x-1$;
  $x := 1$;
  pour i allant de 2 à n **faire**
    déterminer le chemin C_x;
    réétiqueter le sommet terminal y de C_x par i;
    $x := y$ (x est le sommet que l'on vient d'étiqueter i);
fin

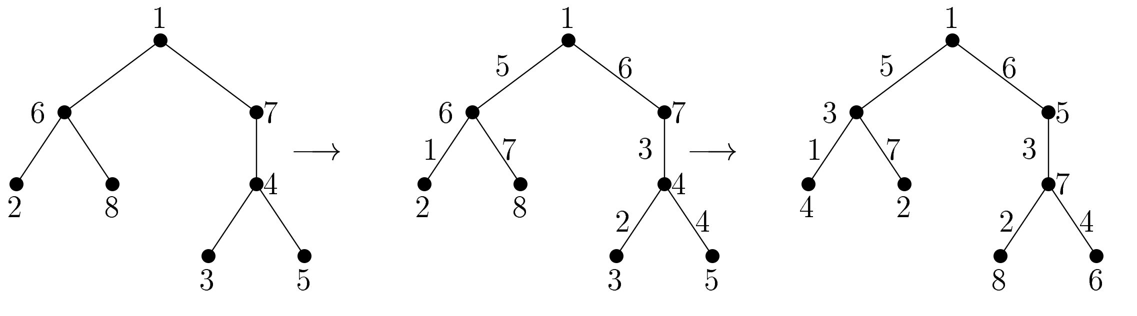


FIG. 2.3 – Transformation d'une arborescence en factorisation du grand cycle

Pour prouver la validité de cet algorithme, il suffit de s'assurer qu'il n'existe pas deux sommets x et y tels que C_x et C_y ont même sommet terminal. Supposons qu'il existe deux sommets x et y tels que C_x et C_y aient au moins un sommet en commun, et soit z leur premier sommet commun. Supposons de plus que C_x (resp. C_y) arrive en z par l'arête a (resp. b).

– z ne peut être le sommet terminal des deux chemins C_x et C_y : en effet si $a > b$ (resp.

$b > a$), C_y (resp. C_x) ne s'arrête pas en z.

- Si z n'est le sommet terminal d'aucun des deux chemins, C_x et C_y ne quittent pas z par la même arête c (sinon, $c > a$ et $c > b$, et on a donc soit $c > a > b$, ce qui implique que C_x (resp. C_y) se poursuit par une arête d'étiquette comprise entre $a+1$ et c (resp. $b+1$ et a)), soit $c > b > a$, ce qui implique que C_x (resp. C_y) se poursuit par une arête d'étiquette comprise entre $a+1$ et b (resp. $b+1$ et c)), et A étant acyclique, il ne peuvent avoir le même sommet terminal.

On a donc réétiqueté tous les sommets de A, et par construction, A' est bien une factorisation du grand cycle. De plus, le réétiquetage des sommets de A étant induit par l'étiquetage des arêtes de A, et ce dernier étant donné par l'étiquetage des sommets de A, deux arborescences différentes ne peuvent produire la même factorisation du grand cycle.

La construction inverse est immédiate. Soit A' une arborescence doublement étiquetée représentant une factorisation de $(1,2,\ldots,n)$. Pour obtenir l'arborescence de racine 1 engendrant A' par l'algorithme précédent, il suffit de remplacer chaque sommet x différent de 1 par l'étiquette de l'arête menant à son père, augmentée de 1 (l'étiquette de la racine reste donc 1).

Interprétation de n^n en termes de factorisations du grand cycle. Il découle de ce qui précède que le nombre de factorisations de $(1,2,\ldots,n)$ ne comportant qu'une seule occurrence de 1 (c'est-à-dire une seule transposition agissant sur 1) est égal au nombre d'arborescences sur $[n]$ de racine 1 n'ayant qu'un fils, c'est-à-dire au nombre d'arborescences sur $[n-1]$, soit $(n-1)^{n-2}$. On peut alors déduire de l'algorithme précédent le résultat suivant, concernant les factorisations du grand cycle ne comportant qu'une seule transposition contenant 1.

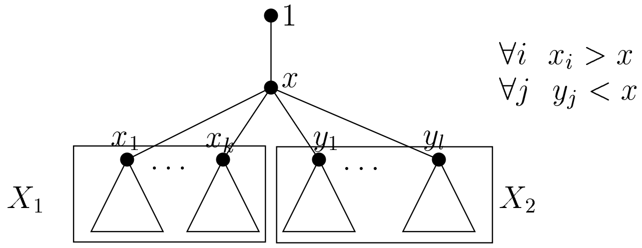
Proposition 37. *Soit $f_{n,k}$ ($k \in [n]$ et $k \neq 1$) le nombre de factorisations de $(1,2,\ldots,n)$ ne comportant qu'une seule transposition contenant 1 et de la forme $(1,k)$.*

$$f_{n,2} = f_{n,n} = (n-2)^{n-2},$$

et si $(n-1) \geq k \geq 3$

$$f_{n,k} = \sum_{l=1}^{n-1} \sum_{r=1}^{l} \binom{n-1}{l} \binom{n-1-l}{k-r+1} (r-1)^{k-2-r} (l-r)(n-k)^{n-1-k-l+r}.$$

Preuve. On a vu précédemment qu'une factorisation de $(1,2,\ldots,n)$ ne comportant qu'une seule transposition contenant 1 est représentée par une arborescence sur $[n]$ de racine 1 n'ayant qu'un seul fils (ce qui est équivalent à une arborescence sur $[n-1]$). Par définition de la représentation d'une factorisation du grand cycle par une arborescence, l'image k de 1 dans la transposition contenant 1 correspond à l'étiquette du fils de la racine dans cette arborescence, après application de l'algorithme de réétiquetage. Soit x le fils de 1 dans l'arborescence initiale A (qui sera ensuite réétiqueté par k) et X_1 (resp. X_2) l'ensemble des sommets y de A tels que le chemin de x vers y commence par une arête croissante (resp. décroissante).



En se souvenant que l'étiquetage des arêtes de A est donné par celui des sommets et de la définition des chemins C_y, on obtient les propriétés suivantes.

- Tout chemin C_y passant par un sommet de X_1 a son sommet terminal dans $X_1 \cup \{x\}$: si C_y passe par x, soit il se termine en x, soit il arrive en x par une arête d'étiquette supérieure ou égale à x et ne peut donc pas se poursuivre par une arête menant à un sommet de X_2 ou au sommet 1.
- On en déduit que le sommet terminal de C_1 est soit x, soit un sommet de X_1, car la première arête suivie après l'arête $x-1$ ne peut mener qu'à un sommet de X_1.
- Pour tout $y \in X_2$, le sommet terminal de C_y appartient à $X_2 \cup \{1\}$: si C_y passe par x, il se poursuit obligatoirement par l'arête menant de x à 1 (étiquetée par $x-1$) et se termine ainsi. Il ne peut donc passer par un sommet de X_1.
- On en déduit que si $X_2 \neq \emptyset$, le sommet terminal de C_x est dans X_2 car la première arête de C_x mène vers un sommet de X_2 (si $X_2 = \emptyset$, C_x est constitué de la seule arête reliant x à la racine 1).

De ces propriétés, on déduit que la nouvelle étiquette du sommet x fournie par l'algorithme précédent est la valeur $2 + |X_1|$. Ainsi, la seule transposition contenant 1 est (1,2) si tous les fils de x sont inférieurs à x dans A. Le théorème 16 implique alors que $f_{n,2} = (n-2)^{n-2}$. Inversement, x est réétiqueté n par l'algorithme si tous les fils de x sont supérieurs à x et on a bien $f_{n,n} = (n-2)^{n-2}$. Pour $k \neq 2$ et $k \neq n$, $f_{n,k}$ est le nombre d'arborescences sur $[n]$ de racine 1 ayant un seul fils noté x et telles que $|X_1| = k$. On construit une telle arborescence de la façon suivante :

- choisir un ensemble $L = (x_1, \dots, x_l)$ de l sommets différents de 1 qui correspondront au sommet x et à ses fils (on explique ainsi la somme sur l) ;
- pour chacun des sommets x_r de L, considérer le cas des arborescences telles que $x = x_r$ et les fils de x sont $L/\{x_r\}$ (on explique ainsi la somme sur r et on remarque que le sommet x a alors exactement $r-1$ fils qui lui sont inférieurs et $l-r$ fils qui lui sont supérieurs) ;
- en rappelant qu'on énumère les arborescences telles que $|X_1| = k$, il reste à choisir $(k-r+1)$ sommets parmi les $(n-1-l)$ restants, à greffer sous les sommets $(x_1, \dots, x_{r-1})$ une forêt de k sommets comportant $(r-1)$ arborescences de racines respectives $x_1, \dots, x_{r-1}$, et à greffer sous les sommets $(x_{r+1}, \dots, x_l)$ une forêt de $n-1-k$ sommets comportant $(l-r)$ arborescences de racines respectives $x_{r+1}, \dots, x_l$: le résultat découle directement de cette construction et du théorème 11.

□

2.4 Arborescences ordonnées, cycliques et planes

La simplicité du résultat établissant que le nombre d'arborescences de Cayley sur $[n+1]$ dont la racine est inférieure à ses fils est n^n incite à penser qu'il existe une preuve formelle, qui devrait pouvoir s'appliquer à d'autres familles d'arborescences. C'est le sujet de cette section. Dans un premier temps, nous énonçons un lemme général pour traiter ce problème, avant de l'appliquer aux arborescences ordonnées, cycliques et planes.

2.4.1 Un lemme général

Soient $\mathcal{U}$ et $\mathcal{V}$ deux familles d'objets combinatoires étiquetés, de séries génératrices exponentielles respectives

$$U(x) = \sum_{k\geq 0} u_k \frac{x^k}{k!} \text{ et } V(x) = \sum_{k\geq 0} v_k \frac{x^k}{k!},$$

et $u_0 \neq 0$. On introduit alors la série formelle $W(x)$ définie par

$$W(x) = \int_0^x V(t)\ dt = \sum_{k\geq 1} v_{k-1} \frac{x^k}{k!}$$

et les familles suivantes de structures arborescentes.

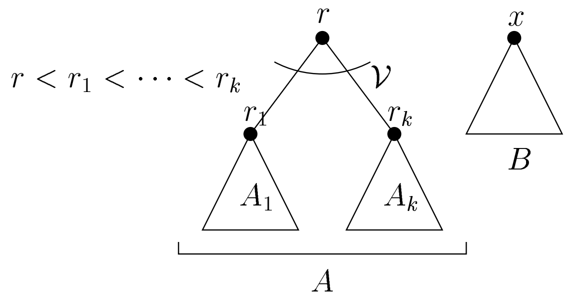
- $\mathcal{T}$ est la famille des $\mathcal{U}$-arborescences, de série génératrice $T(x) = \sum_{k\geq 0} t_k \frac{x^k}{k!}$.
- $\mathcal{R}$ est la famille des arborescences telles que les fils d'un sommet sont munis d'une $\mathcal{V}$-structure si ce sommet est la racine et d'une $\mathcal{U}$-structure sinon.
- $\mathcal{R}_0$ est la famille des arborescences de $\mathcal{R}$ telles que la racine est inférieure à ses fils. On note $R_0(x) = \sum_{k\geq 0} r_{k,0} \frac{x^k}{k!}$ sa série génératrice.

Lemme 38. *La série génératrice exponentielle des arborescences de $\mathcal{R}_0$ vérifie l'équation fonctionnelle*

$$R_0(x) = \frac{x}{T(x)} W(T(x)).$$

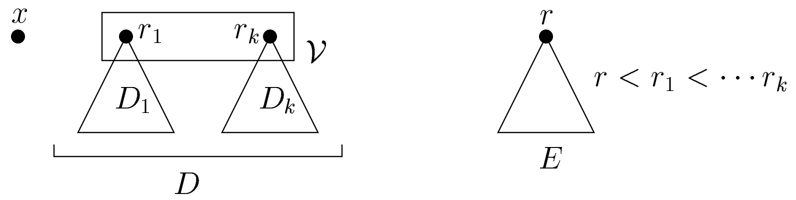
Preuve. On veut donc prouver que $R_0(x)T(x) = xW(T(x))$ [1]. Pour cela, on interprète les deux membres de cette identité de la façon suivante.

Une $(\mathcal{R}_0.\mathcal{T})$-structure sur $[n]$ est un couple (A,B) tel que A est une arborescence dont la racine r est inférieure à ses fils, ces derniers, structurés en une $\mathcal{V}$-assemblée, étant les racines d'arborescences de $\mathcal{T}$ $A_1,\ldots,A_k$, et B est une arborescence de $\mathcal{T}$.



1. La preuve que nous donnons ici, différente de la preuve présentée dans [35], nous a été indiquée par Volker Strehl.

Pour les structures comptées par la série $xW(T(x))$ (on note $\mathcal{W}$ la famille de structures comptées par $W(x)$), on peut tout d'abord déduire du lemme 33 (boxed product) qu'une $\mathcal{W}$-structure sur $[n]$ est un couple $(1,C)$ où C est une $\mathcal{V}$-structure sur $\{2,\ldots,n\}$. On en déduit que $xW(T(x))$ est la série génératrice des triplets (x,D,E) tels que $x \in [n]$, E est une arborescence de $\mathcal{T}$ et D une $\mathcal{V}$-assemblée d'arborescences de $\mathcal{T}$ $(D_1,\ldots,D_k)$ et que la racine y de E est inférieure aux racines des D_i.



Pour prouver le résultat voulu, il suffit alors de remarquer les faits suivants :

- dans (x,D,E), en échangeant les sommets x et r du triplet (x,D,E) et en enracinant la $\mathcal{V}$-assemblée $D_1,\ldots,D_k$ en r, on obtient le couple (A,B) (avec $A_i = D_i$) ;
- dans (A,B), en déracinant A et en échangeant les sommets x et r, on obtient le triplet (x,D,E) (avec $D_i = A_i$).

$\square$

Corollaire 39. *Le nombre d'arborescences de $\mathcal{R}_0$ à n sommets est*

$$[x^n]R_0(x) = \frac{1}{n}[x^{n-1}]\left\{\left(V(x)U(x) - W(x)\frac{\partial U(x)}{\partial x}\right)(U(x))^{n-2}\right\}.$$

Preuve. En remarquant que

$$T(x) = xU(T(x))$$

on en déduit que

$$R_0(x) = \left(\frac{W}{U}\right)(T(x)).$$

Il suffit alors d'appliquer la formule d'inversion de Lagrange (théorème 14, page 18), avec $F(x) = T(x)$, $G(x) = U(x)$ et $H(x) = (W/U)(x)$. $\square$

2.4.2 Application aux arborescences ordonnées, cycliques et planes

Prenons par exemple les arborescences de Cayley. On a alors $U(x) = V(x) = e^x$ et $W(x) = e^x - 1$. On déduit alors du Corollaire 39 que le nombre d'arborescences de Cayley à n sommets telles que la racine est inférieure à ses fils est

$$\frac{n!}{n}[x^{n-1}]e^{(n-1)x} = (n-1)^{n-1}.$$

Proposition 40. *Le nombre d'arborescences ordonnées à n sommets telles que la racine est inférieure à ses fils est*

$$\frac{(2n-2)!}{(n-1)!} - \sum_{k=0}^{n-2} \frac{(n+k-1)!}{(n-k-1)k!}.$$

Preuve. Dans le cas des arborescences ordonnées de racine inférieure à ses fils, on a

$$U(x) = V(x) = \frac{1}{1-x} \Longrightarrow W(x) = -\ln(1-x).$$

On déduit du Corollaire 39, que le nombre d'arborescences ordonnées à n sommets ($n \geq 1$) de racine inférieure à ses fils est

$$\frac{n!}{n}[x^{n-1}]\left(\frac{1}{1-x}\right)^n (1+\ln(1-x)) =$$

$$(n-1)!\left([x^{n-1}](1-x)^{-n} + [x^{n-1}](1-x)^{-n}\ln(1-x)\right) =$$

$$(n-1)!\left(\binom{2n-2}{n-1} - \sum_{k=0}^{n-2}\frac{1}{n-1-k}\binom{n+k-1}{k}\right) =$$

$$\frac{(2n-2)!}{(n-1)!} - \sum_{k=0}^{n-2}\frac{(n+k-1)!}{(n-k-1)k!}.$$

□

$\mathcal{C}$ désignant la famille des listes circulairement ordonnées, on appelle *arborescences cycliques* les arborescences $\mathcal{C}$-enrichies et on note $AC(x)$ leur série génératrice exponentielle.

Proposition 41. *Le nombre d'arborescences cycliques sur $[n]$ telles que la racine est inférieure à ses fils est*

$$\left(\sum_{k=0}^{n}\frac{n!}{(n-k)!}(-1)^{n-1-k}s(n-1,k)\right) - \left(\sum_{k=0}^{n-1}\frac{(n-2)!}{(n-1-k)!}(-1)^{n-k}s(n,k)\right),$$

où les $s(n,k)$ sont les nombres de Stirling de première espèce.

Preuve. On a (voir [16] par exemple)

$$U(x) = V(x) = 1 - \ln(1-x) \Longrightarrow W(x) = (1-x)\ln(1-x) + x.$$

On déduit alors du Corollaire 39, que le nombre d'arborescences cycliques à n sommets ($n \geq 1$) de racine inférieure à ses fils est

$$\frac{n!}{n}[x^{n-1}]\left\{(1-\ln(1-x))^n - \frac{(1-\ln(1-x))^{n-2}}{(1-x)}\right\}. \tag{2.13}$$

En remarquant que

$$\frac{1}{n-1}\frac{\partial((1-\ln(1-x))^{n-1})}{\partial x} = \frac{(1-\ln(1-x))^{n-2}}{(x-1)},$$

on a

$$[x^{n-1}]\frac{(1-\ln(1-x))^{n-2}}{(x-1)} = \frac{n}{n-1}[x^n](1-\ln(1-x))^{n-1}. \tag{2.14}$$

Il suffit maintenant de rappeler (voir [82, Table 351] par exemple) que

$$\left(\ln\left(\frac{1}{1-x}\right)\right)^k = \sum_{i\geq 0} k!(-1)^{i-k} s(i,k)\frac{x^i}{i!},$$

pour obtenir

$$[x^{n-1}](1-\ln(1-x))^n = \sum_{k=0}^{n}\binom{n}{k} k!(-1)^{n-1-k} s(n-1,k)\frac{1}{(n-1)!}, \tag{2.15}$$

et le résultat est une conséquence directe de (2.13), (2.14) et (2.15). □

On appelle *arborescences planes* les arborescences plongées dans un plan orienté [16]. Cette famille est caractérisée par l'enrichissement suivant : la fibre de la racine est munie d'une $\mathcal{C}$-structure et la fibre de tout autre sommet est munie d'une $\mathcal{L}$-structure. On note $\mathcal{P}$ la famille des arborescences planes et $P(x)$ leur série génératrice.

Proposition 42. *Le nombre d'arborescences planes sur* $[n]$ *à* n *sommets telles que la racine est inférieure à ses fils est*

$$\left(2\sum_{k=0}^{n-2}\frac{(n-2+k)!}{k!}\frac{n-1}{n-1-k}\right) - \frac{(2n-3)!}{(n-2)!}.$$

Preuve. On a

$$U(x) = \frac{1}{(1-x)},\; V(x) = 1-\ln(1-x) \Longrightarrow W(x) = (1-x)\ln(1-x)+x.$$

On déduit alors du Corollaire 39, que le nombre d'arborescences planes à n sommets ($n \geq 1$) de racine inférieure à ses fils est

$$\frac{n!}{n}[x^{n-1}]\left\{-x\left(\frac{1}{1-x}\right)^n - 2\left(\frac{1}{1-x}\right)^{n-1}\ln(1-x)\right\}.$$

On a

$$[x^{n-1}]\left\{-x\left(\frac{1}{1-x}\right)^n\right\} = -[x^{n-2}]\left(\frac{1}{1-x}\right)^n = -\binom{2n-3}{n-2}.$$

D'autre part,

$$-[x^{n-1}]\left(\frac{1}{1-x}\right)^{n-1}\ln(1-x) = [x^{n-1}]\left(\sum_{k\geq 0}\binom{n-2+k}{k}x^k\right)\left(\sum_{k\geq 1}\frac{x^k}{k}\right)$$

$$\Longrightarrow -[x^{n-1}]\left(\frac{1}{1-x}\right)^{n-1}\ln(1-x) = \sum_{k=0}^{n-2}\binom{n-2+k}{k}\frac{1}{n-1-k}$$

et on obtient le résultat voulu. □

2.5 Conclusion

Le principal résultat de ce chapitre a trait à la prise en compte d'un paramètre sur les arborescences qui n'a pas été considéré jusqu'à présent, le nombre de fils de la racine qui lui sont inférieurs. Son étude s'avère intéressante du point de vue énumératif, notamment pour la famille $\mathcal{A}_{n,0}$, car ce paramètre permet d'expliquer combinatoirement plusieurs identités faisant intervenir n^n.

Deux questions. Parmi les principales questions relatives à l'étude que nous avons menée dans ce chapitre, il serait intéressant de trouver une explication combinatoire de l'intrigante identité (2.2), due à Goulden et Jackson. Il nous semble que les éléments que nous avons apportés dans ce chapitre peuvent permettre de résoudre ce problème.

Le lemme 38 permet d'exprimer la série génératrice d'une famille d'arborescences telles que la racine est inférieure à ses fils en fonction de la série de la même famille d'arborescences sans conditions sur la racine. Il serait intéressant d'étendre ce résultat aux arborescences dont le nombre de fils de la racine qui lui sont inférieurs est fixé.

Arrangements d'hyperplans. Nous avons rapidement mentionné en introduction que la famille $\mathcal{A}_{n,0}$ apparaissait dans des travaux de Gessel sur les arrangements d'hyperplans (non publiés mais cités par Athansiadis et Linusson dans [9]), et plus particulièrement dans l'arrangement d'hyperplans défini par Shi dans [140] :

– on note S_n ($n > 0$) l'arrangement d'hyperplans de R^n définis par les équations $x_i - x_j = 0$ et $x_i - x_j = 1$, pour $1 \leq i < j \leq n$.

Une *région* de S_n est une composante connexe de l'espace obtenu en supprimant de R^n les hyperplans de S_n. Stanley [145, 147] (voir aussi [9]) a montré que le nombre de régions de S_n est $(n+1)^{n-1}$ en établissant une bijection entre ces régions et les *fonctions de parking sur* $[n]$ que l'on sait être au nombre de $(n+1)^{n-1}$.

Gessel a introduit la notion de *région bornée* : une région de S_n est bornée si son intersection avec l'hyperplan défini par $x_1 + x_2 + \cdots + x_n = 0$ est bornée en tant que région de l'espace Euclidien. Il a établi une bijection entre ces régions et un sous-ensemble des fonctions de parking, *les fonctions de parking premières sur* $[n]$ et a montré, de manière formelle que ces fonctions sont au nombre de n^n (Kalikow a donné une preuve bijective de ce résultat [97]). Il a de plus établi une bijection entre les fonctions de parking premières et les forêts d'arborescences de Cayley sur $n+1$ telles que la racine de la plus petite arborescence n'a aucun fils. Ces forêts sont clairement équivalentes à $\mathcal{A}_{n+1,0}$ et la preuve du lemme 17 fournit donc une preuve bijective de la formule énumérant les régions bornées de S_n.

De plus, si on note R_0 l'unique région de S_n telle que $x_1 > x_2 > \cdots > x_n$ et $x_1 - x_n < 1$, Stanley a introduit une notion de distance d'une région R de S_n par rapport à R_0 et a montré que le nombre de régions de S_n à distance d de R_0 est égal au nombre d'arborescences de Cayley de racine 1 ayant $\binom{n}{2} - d$ inversions. Dans le chapitre suivant, qui poursuit l'étude combinatoire de $\mathcal{A}_{n,0}$, nous nous intéressons à l'énumération des inversions dans les arborescences $\mathcal{A}_{n,0}$, que l'on peut donc interpréter en termes de distance des régions bornées de S_n par rapport à R_0.

Cependant, tous les résultats sus-cités établissent un lien indirect entre régions de S_n et arborescences de Cayley, car ils sont basés sur des bijections entre ces régions et les fonctions de parking (il existe plusieurs bijections classiques entre arborescences et fonctions de parking [103, pages 732-733]). Il serait donc intéressant d'établir un lien combinatoire direct entre arborescences et régions de S_n, qui fasse naturellement apparaître le lien entre les régions bornées et la famille $\mathcal{A}_{n,0}$ et de regarder comment sont transportés les paramètres classiques des arborescences (racine, degré de la racine, arêtes décroissantes, etc) sur les régions de S_n.

Chapitre 3

Polynôme énumérateur des inversions

Dans ce chapitre, nous nous intéressons au *polynôme énumérateur des inversions* des arborescences de Cayley dont la racine est inférieure à ses fils. Nous obtenons plusieurs généralisations de résultats de Kreweras, concernant le polynôme énumérateur des inversions des arborescences de Cayley de racine 1, en particulier une nouvelle interprétation combinatoire des nombres d'Entringer et des nombres d'Euler.

Notation. Tout au long de ce chapitre, nous considérons des arborescences de Cayley que nous nommons simplement arborescences.

3.1 Inversions et arborescences

Définition 43. Une *inversion* dans une arborescence A est un couple (x,y) de sommets tels que $x > y$ où y est un descendant de x. On note $inv(A)$ le nombre d'inversions de A.

Par exemple, l'arborescence de la figure suivante comporte 6 inversions : (7,2), (7,3), (4,1), (4,2), (4,3), (9,6).

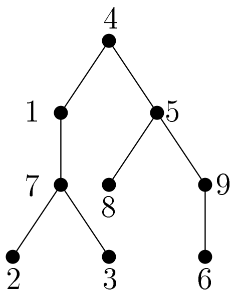


FIG. 3.1 – Une arborescence à 6 inversions

Mallows et Riordan [116] ont été les premiers à considérer ce paramètre, dans le cas des arborescences de racine 1, et à introduire la notion de *polynôme énumérateur des inversions.*

Définition 44. Soit $\mathcal{F}_n$ une famille d'arborescences sur $[n]$. Le polynôme énumérateur des inversions de $\mathcal{F}_n$ est défini par

$$\mathcal{P}_{\mathcal{F}_n}(q) = \sum_{A \in \mathcal{F}_n} q^{inv(A)}.$$

On note $\mathcal{P}_n(q)$ le polynôme énumérateur des inversions des arborescences de racine 1.

Le polynôme énumérateur des inversions des arborescences de racine 1 est relié à de nombreux problèmes combinatoires, comme le polynôme de Tutte pour certaines familles de graphes [63], les fonctions de parking [106, 121] et les fonctions de k-parking [166], l'activité externe d'une arborescence [10], ou les arrangements d'hyperplans [147, 145], comme nous l'avons déjà mentionné en conclusion du chapitre précédent. Pour des familles autres que les arborescences de Cayley, il existe peu de résultats. On peut essentiellement citer les travaux de Gessel, Sagan et Yeh [64] et de Chen [40] pour les arborescences ordonnées, cycliques et planes de racine 1.

La première étude détaillée du polynôme $\mathcal{P}_n(q)$ est due à Kreweras [106] qui obtient notamment la formule de récurrence suivante.

Proposition 45. *Le polynôme énumérateur des inversions des arborescences de racine* 1 *vérifie la relation de récurrence :*

$$\mathcal{P}_1(q) = 1, \tag{3.1}$$

$$\mathcal{P}_{n+1}(q) = \sum_{m=1}^{n} \binom{n-1}{m-1} \left(\sum_{i=0}^{m-1} q^i\right) \mathcal{P}_m(q)\mathcal{P}_{n+1-m}(q). \tag{3.2}$$

Cette formule est obtenue en supprimant l'arête reliant la racine 1 au fils qui contient le plus grand sommet dans la sous-arborescence qui en est issue. Nous reprenons cette technique dans la section suivante pour l'étendre au cas des arborescences de $\mathcal{A}_{n,0}$. Kreweras montre qu'évalué en $q = -1,0,1,2$, le polynôme $\mathcal{P}_n(q)$ prend des valeurs remarquables, qui font notamment intervenir les nombres d'Euler et le nombre de graphes étiquetés simples connexes.

Définition 46. On note e_n le $n^{ème}$ nombre d'Euler ; ces nombres sont donnés par la série génératrice

$$\sum_{n \geq 0} e_n \frac{x^n}{n!} = \frac{1 + \sin(x)}{\cos(x)},$$

ou par la récurrence

$$e_0 = 1,\ e_{n+1} = \sum_{m=0}^{\lfloor n/2 \rfloor} \binom{n}{2m} e_{2m} e_{n-2m}.$$

Proposition 47. *Soit* $n > 0$.

$$\mathcal{P}_n(0) = (n-1)!, \tag{3.3}$$

$$\mathcal{P}_n(1) = n^{n-2}, \tag{3.4}$$

$$\mathcal{P}_n(-1) = e_{n-1}. \tag{3.5}$$

Les preuves des identités (3.3) et (3.4) sont immédiates :

- $\mathcal{P}_n(0)$ est le nombre d'arborescences sur $[n]$ ne comportant aucune inversion, c'est-à-dire le nombre d'arborescences croissantes : $(n-1)!$ (voir la preuve de la proposition 30),
- $\mathcal{P}_n(1)$ est le nombre d'arborescences sur $[n]$ de racine 1, c'est-à-dire n^{n-2} (Corollaire 8).

La preuve de l'identité (3.5) donnée par Kreweras est basée sur les récurrences satisfaites par $\mathcal{P}_n(q)$ et par les nombres d'Euler. Or les nombres d'Euler énumèrent de nombreux objets combinatoires, comme par exemple les permutations alternantes [107] ou les arborescences croissantes paires [162]. Pansiot [123] a utilisé ce dernier point pour donner une preuve combinatoire de (3.5), basée sur la technique suivante :

- exhiber une sous-famille $\mathcal{B}_n$ de $\tilde{\mathcal{A}}_n$ (les arborescences sur $[n]$ de racine 1) de cardinal e_{n-1} telle que toute arborescence de $\mathcal{B}_n$ a un nombre pair d'inversions ($\mathcal{B}_n$ est la famille des arborescences croissantes paires, que nous définissons dans la section suivante),
- définir une involution ψ sur $\tilde{\mathcal{A}}_n$ privé de $\mathcal{B}_n$ telle que le nombre d'inversions de A et $\psi(A)$ diffère de 1.

Nous utilisons une technique similaire dans le section suivante pour prouver un résultat analogue pour le polynôme associé aux arborescences de $\mathcal{A}_{n,0}$.

Définition 48. On appelle graphe simple étiqueté sur $[n]$ un graphe à n sommets étiquetés par $\{1,\dots,n\}$, n'ayant ni boucle ni arête multiple. On note $\mathcal{G}_n$ l'ensemble de tels graphes connexes sur $[n]$ et $g_n = |\mathcal{G}_n|$.

Proposition 49. *Soit $n > 0$.*

$$\mathcal{P}_n(1+q) = \sum_{G\in\mathcal{G}_n} q^{a(G)-n+1} \tag{3.6}$$

où $a(G)$ est le nombre d'arêtes de G. On en déduit alors que

$$\mathcal{P}_n(2) = g_n. \tag{3.7}$$

La première preuve de ce résultat est due à Kreweras qui montre que le nombre de graphes connexes simples étiquetés sur $[n]$ satisfait une récurrence similaire à celle de $\mathcal{P}_n(2)$ et en déduit (3.7). Dans [65], Gessel et Wang proposent une preuve bijective de (3.6) basée sur la notion de parcours en profondeur d'un graphe. Nous reprendrons ce principe pour prouver un résultat analogue pour le polynôme associé aux arborescences de $\mathcal{A}_{n,0}$.

Finalement, on peut relier le polynôme des inversions des arborescences au polynôme de Tutte des graphes complets.

Définition 50. Soit G un graphe aux arêtes totalement ordonnées et A une arborescence couvrante de G. Une arête (x,y) de G n'appartenant pas à A (resp. appartenant à A) est *externe* (resp. *interne*) si elle est la plus grande arête de l'unique cycle du graphe $A\cup(x,y)$ (resp. cocycle de $\{G-A\}\cup(x,y)$) contenant (x,y). On note $EA_G(A)$ (resp. $IA_G(A)$) l'ensemble des arêtes externes (resp. internes) de G relativement à A.

Le polynôme de Tutte de G est alors défini par

$$\mathcal{T}_G(u,q) = \sum_A u^{|IA_G(A)|} q^{|EA_G(A)|},$$

où la somme est sur toutes les arborescences couvrantes A de G.

Tutte [156] a notamment montré que ce polynôme ne dépendait pas de l'ordre choisi sur les arêtes de G. Si on note K_n le graphe complet à n sommets, on a alors le résultat suivant [10, 63].

Proposition 51. *Soit $n > 0$.*

$$\mathcal{T}_{K_n}(1,q) = \mathcal{P}_n(q) \tag{3.8}$$

3.2 Polynôme des inversions de $\mathcal{A}_{n,0}$

Dans cette section, nous nous intéressons au polynôme des inversions des arborescences dont la racine est inférieure à ses fils, que nous notons $\tilde{\mathcal{P}}_n(q)$. Plus précisément, nous étudions la famille $\mathcal{A}_{n,0,r}$ des arborescences de $\mathcal{A}_{n,0}$ ayant pour racine $r \in [n-1]$, dont nous notons $\tilde{\mathcal{P}}_{n,r}(q)$ le polynôme énumérateur des inversions (on a donc $\tilde{\mathcal{P}}_{n,1}(q) = \mathcal{P}_n(q)$). Pour ces polynômes, nous obtenons des généralisations des propositions 45, 47, 49 et 51 (en posant $r = 1$ dans nos résultats, on retrouve les résultats de Kreweras).

Dans un premier temps, nous nous intéressons à une formule de récurrence pour ce polynôme et à son évaluation en 0, 1 et 2. Nous relions notamment cette dernière valeur $\tilde{\mathcal{P}}_{n,r}(2)$ à l'énumération de graphes simples connexes étiquetés sur $[n]$ tels que le sommet r est inférieur à ses voisins. Nous concluons ce paragraphe en reliant $\tilde{\mathcal{P}}_{n,r}$ au polynôme de Tutte de certains graphes dérivés du graphe complet. Dans un deuxième temps, nous montrons que la valeur de ce polynôme en -1 est liée aux nombres d'Entringer (qui raffinent les nombres d'Euler). On peut noter que la plupart des résultats de Kreweras sur $\mathcal{P}_n(q)$ s'interprètent en termes de graphes étiquetés, et que les généralisations que nous proposons pour $\tilde{\mathcal{P}}_n(q)$ s'interprètent naturellement en termes de graphes tels que le sommet r est inférieur à tous ses voisins.

3.2.1 Récurrence pour $\tilde{\mathcal{P}}_{n,r}$, évaluation en 0, 1 et 2 et lien avec le polynôme de Tutte

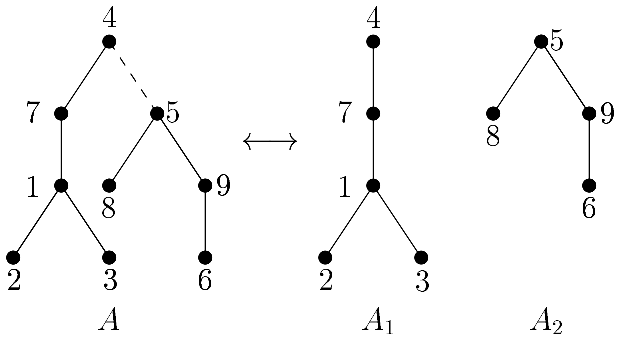
Proposition 52. *Le polynôme énumérateur des inversions des arborescences de $\mathcal{A}_{n,0,r}$ vérifie la relation de récurrence :*

$$\tilde{\mathcal{P}}_{r,r}(q) = \begin{cases} 1 \text{ si } r = 1, \\ 0 \text{ sinon,} \end{cases} \tag{3.9}$$

$$\tilde{\mathcal{P}}_{n+1,r}(q) = \sum_{m=1}^{n+1-r} \sum_{k=0}^{r-1} \binom{n-r}{m-1}\binom{r-1}{k}\left(\sum_{i=0}^{m-1} q^{2k+i}\right)\tilde{\mathcal{P}}_{k+m,1}(q)\tilde{\mathcal{P}}_{n',r'}(q), \tag{3.10}$$

où $n' = (n+1-m-k)$ et $r' = (r-k)$.

Preuve. La preuve de ce résultat est basée sur le même principe que la démonstration du cas $r = 1$ exposée par Kreweras dans [106]. Tout d'abord, la preuve de (3.9) est immédiate. Ensuite, soit A une arborescence de $\mathcal{A}_{n+1,0,r}$ et x le seul fils de la racine r situé sur le chemin allant de r à $n+1$. En supprimant l'arête reliant r et x, on obtient une forêt à deux arborescences (A_1,A_2) telle que r est la racine de A_1 et est inférieure à ses fils, x est la racine de A_2 et $n+1$ appartient à A_2 (on peut avoir $x = n+1$).



Si on note, pour une arborescence A et un entier r, $|A|_{>r}$ (resp. $|A|_{<r}$) le nombre de sommets de A supérieurs (resp. inférieurs) à r, on a alors

$$inv(A) = inv(A_1) + inv(A_2) + |A_2|_{<r}.$$

En posant $k = |A_2|_{<r}$ et $m = |A_2|_{>r}$ $(m \geq 1)$, on peut dire que A_1 comporte $n' = (n+1-m-k)$ sommets, dont exactement $r' = (r-1-k)$ sont inférieurs à la racine r, ce qui explique ainsi le terme $\tilde{\mathcal{P}}_{n',r-k}(q)$ de (3.10), polynôme énumérant les inversions de A_1. Soient $x_1 < x_2 < \cdots < x_{k+m}$ les sommets de A_2 et x_i sa racine. En échangeant successivement les sommets x_i et x_{i-1}, puis x_{i-1} et x_{i-2}, ..., puis x_2 et x_1, on obtient une arborescence A_2' de racine x_1 (c'est-à-dire dont la racine est le plus petit sommet). En remarquant que chaque échange de deux sommets consécutifs x_j et x_{j-1} diminue le nombre d'inversions de 1 (x_j, situé à la racine, est échangé avec x_{j-1} qui lui est inférieur), on constate que $inv(A_2') = inv(A_2) + i - 1$, les valeurs possibles pour i étant comprises entre $k+1$ et $k+n$ car la racine x_i de A_2 est supérieure à r et $k = |A_2|_{<r}$. En rappelant que le nombre d'inversions de la forme (r,x) avec x dans A_2 est $k = |A_2|_{<r}$, on obtient l'explication du terme

$$\left(\sum_{i=0}^{m-1} q^{2k+i}\right) \tilde{\mathcal{P}}_{k+m,1}(q),$$

et donc du résultat annoncé. □

En posant $r = 1$, on a alors $k = 0$ et on retrouve bien le résultat de la proposition 45.

Proposition 53. *Soient $n > 0$ et $0 < r < n$.*

$$\tilde{\mathcal{P}}_{n,r}(0) = \begin{cases} (n-1)! \text{ si } r = 1, \\ 0 \text{ sinon}, \end{cases} \tag{3.11}$$

$$\tilde{\mathcal{P}}_{n,r}(1) = (n-r)n^{n-r-1}(n-1)^{r-2}. \tag{3.12}$$

Preuve. $\tilde{\mathcal{P}}_{n,r}(0)$ est le nombre d'arborescences croissantes de racine r inférieure à ses fils. Une arborescence croissante étant obligatoirement de racine 1, on a immédiatement la preuve de (3.11). $\tilde{\mathcal{P}}_{n,r}(1)$ est le nombre d'arborescences de $\mathcal{A}_{n,0,r}$. L'identité (3.12) découle alors du théorème 24. □

Le corollaire suivant est une conséquence immédiate du fait que $|\mathcal{A}_{n,0}| = (n-1)^{n-1}$.

Corollaire 54. *Soit $n > 0$.*

$$\tilde{\mathcal{P}}_n(0) = (n-1)!, \tag{3.13}$$
$$\tilde{\mathcal{P}}_n(1) = (n-1)^{n-1}. \tag{3.14}$$

On note $\mathcal{G}_{n,r}$ l'ensemble des graphes simples connexes étiquetés sur $[n]$ tels que le sommet r est inférieur à tous ses voisins (sommets reliés à r par une arête), et $g_{n,r} = |\mathcal{G}_{n,r}|$.

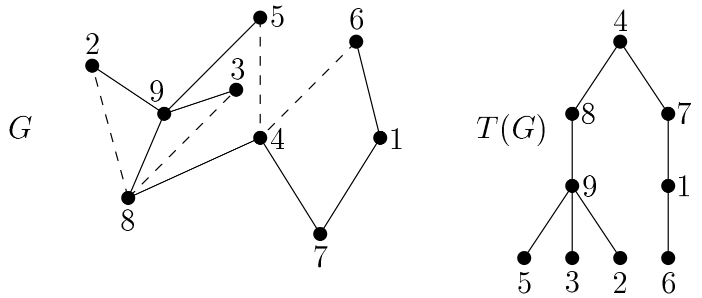
Proposition 55. *Soient $n > 0$ et $0 < r < n$.*

$$\tilde{\mathcal{P}}_{n,r}(1+q) = (1+q)^{2(r-1)} \sum_{G \in \mathcal{G}_{n,r}} q^{a(G)-n+1} \tag{3.15}$$

où $a(G)$ est le nombre d'arêtes de G. Il s'en suit que

$$\tilde{\mathcal{P}}_{n,r}(2) = 4^{(r-1)} g_{n,r}. \tag{3.16}$$

Preuve. Nous étendons la méthode utilisée par Gessel et Wang [65] pour prouver combinatoirement la proposition 49. Soit G un graphe de $\mathcal{G}_{n,r}$. On appelle *arborescence couvrante maximale de* G, notée $T(G)$, l'arborescence obtenue en réalisant un parcours en profondeur de G à partir du sommet r de la manière suivante : si x est le dernier sommet rencontré lors du parcours, le sommet que l'on visite ensuite est le plus grand voisin de x non encore rencontré. Dans l'exemple suivant, les arêtes pleines sont les arêtes empruntées lors d'un tel parcours du graphe G lorsque $r = 4$.



Soit A une arborescence de racine r inférieure à ses fils. On note $C(A)$ l'ensemble des arêtes que l'on peut ajouter à A pour obtenir un graphe de $\mathcal{G}_{n,r}$ d'arborescence couvrante maximale A. Formellement, $C(A)$ est l'ensemble des couples (x,y) ne correspondant pas à des arêtes de A tels que

- si $x = r$ alors (r,y) ne constitue pas une inversion et le fils z de r situé sur le chemin menant à y est tel que (z,y) est une inversion,
- si $x \neq r$ alors il existe z tel que (x,z) est une arête de A et (z,y) est une inversion.

Dans l'exemple, $C(T(G)) = \{(4,5),(4,6),(8,2),(8,3),(8,5)\}$. On constate que chaque inversion (u,v) de A induit un couple de $C(A)$ sauf lorsque

- $u = r$: $r-1$ inversions sont de cette forme
- u est un fils de la racine r et $v < r$ (les couples (r,v) où $v > r$ ne constituant pas une inversion sont dans $C(A)$) : il y a $r-1$ couples de cette forme puisque $u > r$.

Dans notre exemple, les inversions (8,5), (7,6), (9,2), (9,3) et (9,5) correspondent respectivement à (4,5), (4,6), (8,2), (8,3) et (8,5). On en déduit donc que $|C(A)| = inv(A) - 2(r-1)$. Soit $\mathcal{G}_A$ l'ensemble des graphes G tels que $T(G) = A$. Par définition de l'arborescence couvrante maximale d'un graphe et de $C(A)$, on peut dire que, pour une arborescence A donnée, un graphe G appartient à $\mathcal{G}_A$ si et seulement si toutes les arêtes de G non présentes dans A appartiennent à $C(A)$. On en déduit que, si $|A| = n$,

$$\sum_{G \in \mathcal{G}_A} q^{a(G)} = q^{n-1}(1+q)^{inv(A)-2(r-1)},$$

le terme q^{n-1} correspondant aux arêtes de A et le terme $(1+q)^{inv(A)-2(r-1)}$ correspondant aux autres arêtes choisies dans $C(A)$. En sommant sur toutes les arborescences couvrantes maximales, c'est-à-dire sur toutes les arborescences de $\mathcal{A}_{n,0,r}$, on obtient le résultat attendu. □

La preuve du corollaire suivant est alors immédiate.

Corollaire 56. *Soit $n > 0$.*

$$\tilde{\mathcal{P}}_n(1+q) = \sum_{G \in \mathcal{G}_n} q^{a(G)-n+1} \mathcal{Q}_G(1+q) \tag{3.17}$$

où, si on note $S(G)$ l'ensemble des sommets de G inférieurs à leurs voisins,

$$\mathcal{Q}_G(q) = \sum_{x \in S(G)} q^{2(x-1)}.$$

Si on note $K_{n,r}$ le graphe complet privé des arêtes entre r et les sommets $1,2,\ldots,r-1$, on relie alors $\tilde{\mathcal{P}}_{n,r}(q)$ à $\mathcal{T}_{K_{n,r}}(1,q)$.

Proposition 57. *Soient $n > 0$ et $0 < r < n$.*

$$q^{r-1}\mathcal{T}_{K_{n,r}}(1,q) = \tilde{\mathcal{P}}_{n,r}(q) \tag{3.18}$$

Preuve. Soit G un graphe à n sommets et F une forêt couvrante de G. On note $C_G(F)$ l'ensemble des couples (x,y) tels que

- y est un descendant de x dans F, mais pas un fils de x,
- si le chemin allant de x à y dans F commence par l'arête (x,z) alors $z > y$.

Dans [63], Gessel et Sagan ont montré que

$$\mathcal{T}_G(1+u,q) = \sum_F u^{c(F)-1} q^{|C_G(F)|},$$

où $c(F)$ est le nombre d'arborescences de F, la somme portant sur toutes les forêts couvrantes de G. En posant $u = 0$, on en déduit que

$$\mathcal{T}_G(1,q) = \sum_A q^{|C_G(A)|},$$

la somme portant sur toutes les arborescences couvrantes de G. De la même manière que dans la preuve précédente, il suffit alors de remarquer que si $G = K_{n,r}$, toute arborescence couvrante de $K_{n,r}$ vérifie $|C_G(A)| = inv(A) - r + 1$. Comme toute arborescence de $\mathcal{A}_{n,0,r}$ est une arborescence couvrante de $K_{n,r}$, on en déduit le résultat annoncé. □

3.2.2 Évaluation de $\tilde{\mathcal{P}}_{n,0,r}(-1)$

L'évaluation de $\mathcal{P}_n(-1)$ fait apparaître des nombres bien connus, les nombres d'Euler. Il existe plusieurs spécialisations des nombres d'Euler, dont la suite des nombres d'Entringer [59], qui ont de nombreuses interprétations combinatoires en termes de permutations alternantes ou d'arborescences [107, 127, 128].

Définition 58. Les nombres d'Entringer $e_{n,k}$ sont définis par la récurrence.

$$e_{n,n} = \begin{cases} 1 \text{ si } n = 1 \\ 0 \text{ sinon,} \end{cases}$$

et, pour $k = 1, \ldots, n$,

$$e_{n+1,k+1} = e_{n+1,k} - e_{n,n+1-k}.$$

Ils sont reliés aux nombres d'Euler par la relation

$$e_n = e_{n+1,1} = \sum_{k=1}^{n} e_{n,k}. \tag{3.19}$$

Proposition 59. *Soient $n > 0$ et $0 < r < n$.*

$$\tilde{\mathcal{P}}_{n,r}(-1) = e_{n,r}. \tag{3.20}$$

Avant de présenter la preuve de cette proposition, on remarque qu'en posant $r = 1$ on retrouve le fait que $\mathcal{P}_n(-1) = e_{n-1}$. De plus, en utilisant la relation (3.19), on obtient le corollaire suivant.

Corollaire 60. *Soit $n > 0$.*

$$\tilde{\mathcal{P}}_n(-1) = e_n. \tag{3.21}$$

Tout le reste de cette section est consacrée à la preuve de la proposition 59, pour laquelle nous allons nous inspirer de la preuve de l'identité (3.5) due à Pansiot [123]. Nous allons procéder en trois étapes :

1. exhiber un sous-ensemble $\mathcal{B}_{n,0,r}$ de $\mathcal{A}_{n,0,r}$ de cardinal $e_{n,r}$, dont chaque arborescence a un nombre pair d'inversions,
2. partitionner $\mathcal{A}_{n,0,r}$ en sous-familles $C_1, \ldots, C_l$ telles que pour $i = 1, \ldots, l$, C_i contient au plus une arborescence de $\mathcal{B}_{n,0,r}$ (on aura alors $\tilde{\mathcal{P}}_{n,r}(-1) = \sum_{i=1}^{l} \mathcal{P}_{C_i}(-1)$),
3. montrer que pour $i = 1, \ldots, l$, $\mathcal{P}_{C_i}(-1) = 1$ si C_i contient une arborescence de $\mathcal{B}_{n,0,r}$ et $\mathcal{P}_{C_i}(-1) = 0$ sinon, ce qui prouvera la proposition 59.

Une sous-famille $\mathcal{B}_{n,0,r}$ de $\mathcal{A}_{n,0,r}$ de cardinal $e_{n,r}$. Pour définir la famille d'arborescences $\mathcal{B}_{n,0,r}$, il est nécessaire d'introduire la notion d'arborescence paire, utilisée notamment par Pansiot [123], Viennot [162] ou Kuznetsov, Pak et Postnikov [107].

Définition 61. Une arborescence est *paire* si tout sommet différent de la racine a un nombre pair de fils. On note $\mathcal{B}_{n,0,r}$ la famille des arborescences A de $\mathcal{A}_{n,0,r}$ telles que, si $x_1,x_2,\ldots,x_k$ sont les fils de r et $A_1,A_2,\ldots,A_k$ les sous-arborescences de A de racines respectives $x_1,x_2,\ldots,x_k$, les A_i sont des *arborescences décroissantes paires contenant chacune un nombre impair de sommets supérieurs à r.*

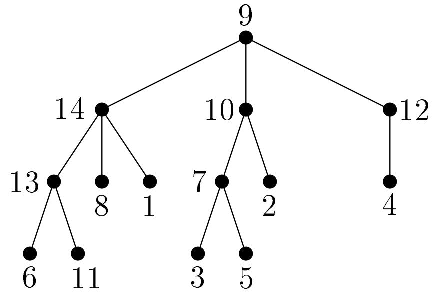


FIG. 3.2 – Une arborescence de $\mathcal{B}_{14,0,9}$.

Lemme 62. *Pour tout arborescence A de $\mathcal{B}_{n,0,r}$, $inv(T)$ est pair.*

Preuve. Soit $A \in \mathcal{B}_{n,0,r}$. Les A_i étant des arborescences décroissantes paires, chaque sommet x tel que $x \neq r$ et $x \neq x_i$, a un nombre pair de descendants, et est supérieur à tous ses descendants. L'arborescence possède donc un nombre pair d'inversions de la forme (x,y) avec $x \neq r$ et $x \neq x_i$. La parité de $inv(A)$ est donc celle du nombre d'inversions de la forme (r,y) ou (x_i,y), pour $i = 1,2,\ldots,k$. Les A_i étant décroissantes, le nombre d'inversions de la forme (x_i,y) est égal à $n-k-1$. De plus, le nombre d'inversions de la forme (r,y) est égal à

$$r - 1 = n - 1 - \sum_{i=1}^{k} |A_i|_{>r}.$$

Il s'en suit que la parité de $inv(A)$ est celle de

$$2n - 2 - k - \sum_{i=1}^{k} |A_i|_{>r} = 2n - 2 - \sum_{i=1}^{k}(|A_i|_{>r} + 1).$$

Il suffit alors de rappeler que par définition de l'ensemble $\mathcal{B}_{n,0,r}$, $|A_i|_{>r}$ est impair pour $i = 1,2,\ldots,k$, pour conclure la preuve de ce lemme. □

Soit $\overline{\mathcal{B}}_{n,0,r}$ la famille des arborescences A telles que, si $x_1,x_2,\ldots,x_k$ sont les fils de la racine r et $A_1,A_2,\ldots,A_k$ les sous-arborescences de A de racines respectives $x_1,x_2,\ldots,x_k$, les A_i sont des *arborescences croissantes paires contenant chacune un nombre impair de sommets supérieurs à r*. Il existe une bijection immédiate entre $\overline{\mathcal{B}}_{n,0,r}$ et $\mathcal{B}_{n,0,r}$ (elle consiste, pour chaque A_i, à échanger son plus grand sommet et son plus petit sommet, puis son second plus grand et son second plus petit, etc). La figure 3.3, page suivante, représente une arborescence de $\mathcal{B}_{n,0,r}$ et l'arborescence correspondante de $\overline{\mathcal{B}}_{n,0,r}$. On constate qu'une arborescence de $\overline{\mathcal{B}}_{n,0,r}$ n'appartient pas en général à $\mathcal{A}_{n,0,r}$.

Soit $\mathcal{C}_{n,r}$ la famille des arborescences croissantes paires sur $[n]$ telles que le père du sommet n (nécessairement une feuille) est le sommet r. Kuznetsov, Pak et Postnikov ont montré bijectivement le résultat suivant.

Théorème 63. [107, théorème 4] *Soient $n > 0$ et $0 < r < n$.*

$$|\mathcal{C}_{n+1,r}| = e_{n,r}.$$

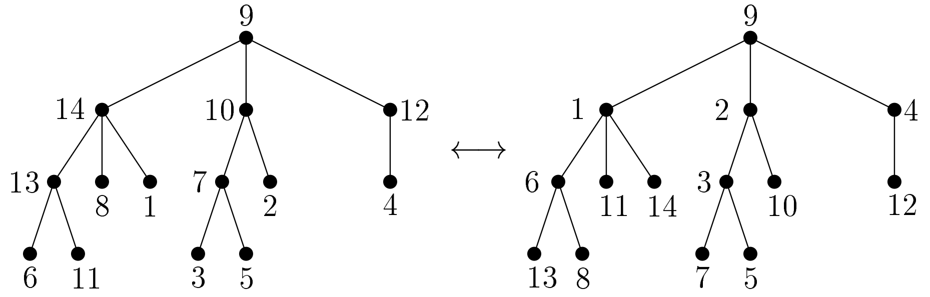


FIG. 3.3 – La correspondance entre $\overline{\mathcal{B}}_{n,0,r}$ et $\mathcal{B}_{n,0,r}$

Pour prouver que $\mathcal{B}_{n,0,r} = e_{n,r}$, il suffit donc d'établir une bijection entre $\mathcal{C}_{n+1,r}$ et $\overline{\mathcal{B}}_{n,0,r}$.

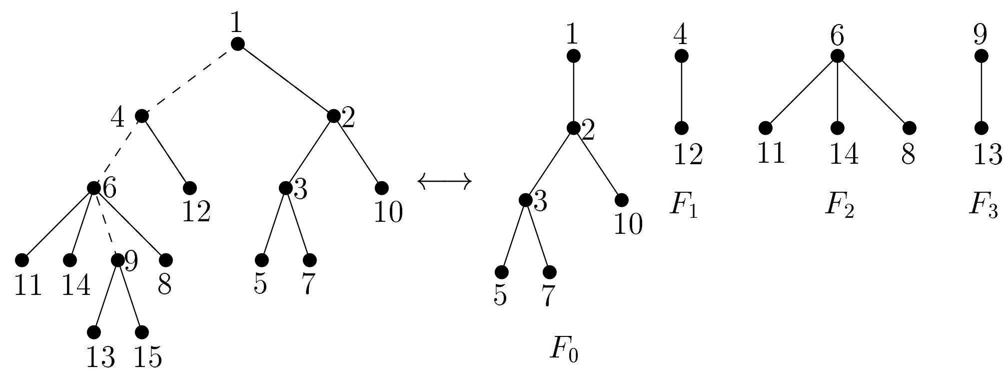
Lemme 64. *Il existe une bijection ϕ entre $\mathcal{C}_{n+1,r}$ et $\overline{\mathcal{B}}_{n,0,r}$.*

Preuve. Soit A une arborescence de $\mathcal{C}_{n+1,r}$. On distingue deux cas.
Cas 1. Dans un premier temps, on traite le cas où $r = 1$ (et donc la racine de A). Dans ce cas $\phi(A)$ est obtenue en supprimant le sommet $n+1$ de A. Du fait que A est une arborescence croissante et paire, on a bien $\phi(A) \in \overline{\mathcal{B}}_{n,0,r}$. On a donc dans ce cas une bijection entre les arborescences de $\mathcal{C}_{n+1,1}$ et les arborescences de $\overline{\mathcal{B}}_{n,0,1}$.
Cas 2. On suppose maintenant que $r \neq 1$. La bijection ϕ, qui se décompose alors en 4 étapes que nous illustrons au fur et à mesure par un exemple, est basée sur les notions de de *r-forêt paire* et de *r-forêt impaire* d'une arborescence.

Soient B une arborescence quelconque, $x_1,\dots,x_k$ les fils de la racine de B et $B_1,\dots,B_k$ les sous-arborescences de B de racines respectives $x_1,\dots,x_k$: pour un r donné, la r-forêt paire (resp. impaire) de B est le sous-ensemble des B_i ayant un nombre pair (resp. impair) de sommets supérieurs à r.
Étape 1. On supprime dans A la feuille $n+1$, puis toutes les arêtes du chemin allant de la racine 1 à r, ce qui donne une forêt ordonnée $F = \{F_0,F_1,\dots, F_k\}$ de $k+1$ arborescences de racines respectives $x_0(=1) < x_1 < \cdots < x_k(=r)$. La construction inverse, produisant A à partir de F, est immédiate du fait de l'ordre sur les racines des F_i.

Dans notre exemple, on considère une arborescence de $\mathcal{C}_{15,9}$. La figure suivante illustre cette décomposition de A en F (les arêtes en pointillées représentent le chemin de la racine 1 au sommet r). La forêt F vérifie alors la propriété suivante.

(a) *Pour $i = 0,1,\dots,k$, F_i est une arborescence croissante paire. De plus, si $i > 0$, la racine x_i de F_i a un nombre impair de fils.*



Étape 2. On partitionne $F = \{F_0,F_1,\ldots, F_k\}$ en trois sous-ensembles : l'arborescence F_k, l'ensemble $F' = \{F_0,F'_1,\ldots,F'_l\}$ et l'ensemble $F'' = \{F''_1,\ldots,F''_{k-l-1}\}$ où les F'_i (resp. F''_j) sont les arborescences de F autres que F_0 (resp. que F_k) ayant un nombre pair (resp. impair) de sommets supérieurs à r. On peut avoir F'' vide (notamment lorsque $k = 2$). Dans notre exemple, on a $F' = \{F_0,F_2\}$ et $F'' = \{F_1\}$. La construction inverse, produisant F à partir de F_k, F' et F'' est immédiate, et F' vérifie les propriétés suivantes.

(b) *Pour tout $i > 0$, la r-forêt impaire (resp. paire) de F'_i contient un nombre pair (resp. impair) d'arborescences.*

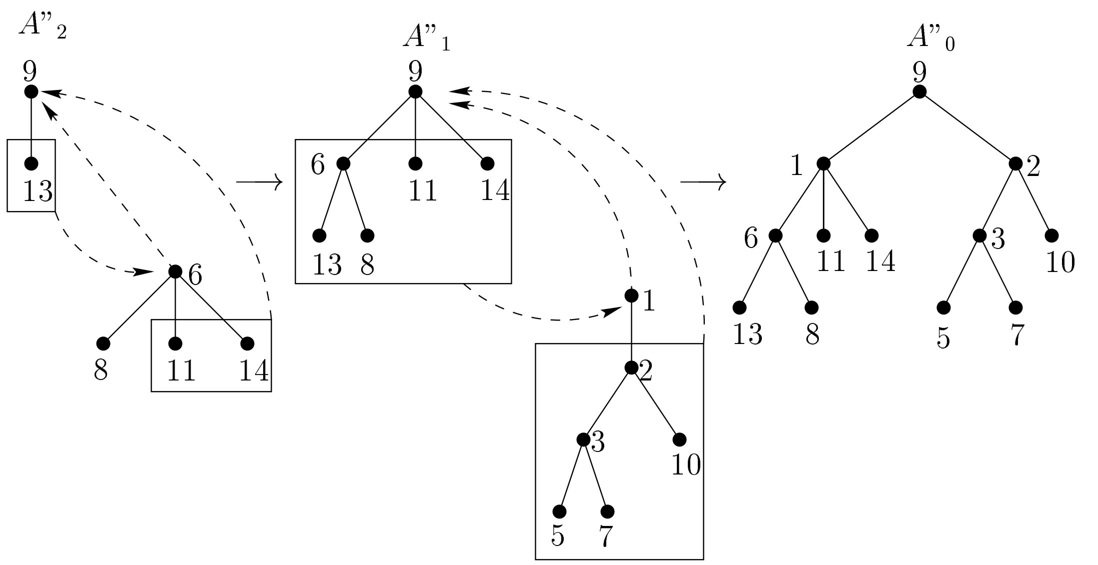
En effet, comme F'_i contient un nombre pair de sommets supérieurs à r, la r-forêt impaire de F'_i ne peut contenir qu'un nombre pair d'arborescences. Et comme la racine de F'_i a un nombre impair de fils (propriété (a)), la r-forêt paire de F'_i contient un nombre impair d'arborescences.

(c) *Pour tout i, la r-forêt impaire (resp. paire) de F''_i contient un nombre impair (resp. pair) d'arborescences.*

On prouve (c) de la même manière que (b).

Étape 3. La troisième étape consiste à construire de manière incrémentale une arborescence A'' à partir de F_k en “insérant” les arborescences de F' de la manière suivante. On commence par poser $A''_{l+1} = F_k$, puis pour i allant de l à 0, on échange les r-forêts impaires de F'_i et A''_{i+1} (les arborescences de la r-forêt impaire de F'_i sont insérées sous la racine de A''_{i+1} pendant que, simultanément, les arborescences de la r-forêt impaire de A''_{i+1} sont insérées sous la racine x'_i de F'_i) et enfin on insère l'arborescence F'_i ainsi modifiée sous la racine de A''_{i+1} (x'_i devient un fils de cette racine) pour obtenir A''_i. Finalement, on pose $A'' = A''_0$.

Sur notre exemple, on effectue donc les opérations décrites par la figure suivante (on entoure les r-forêts impaires qu'on va échanger). On rappelle que $F' = \{F_0,F_2\}$ et $F_k = F_3$.



On vérifie alors les propriétés suivantes.

(d) *Pour $i = 0,\ldots,l+1$, la racine de A''_i est r.*

En effet, la racine de $F_k = A''_{l+1}$ est r, par définition de F_k, et aucune des opérations effectuées lors du passage de A''_{i+1} à A''_i ne modifie cette racine.

(e) *Pour $i = 1, \ldots, l+1$, la racine r de A''_i a un nombre impair de fils et pour $i = 0, \ldots, l+1$, la r-forêt paire de A''_i est vide.*

Pour le cas $i = l+1$, cela découle directement de la définition de $F_k = A''_{l+1}$, qui est une arborescence croissante paire de racine r, et de la propriété (a). Pour le cas général, cela découle de la transformation de A''_{i+1} en A''_i :

- on remplace la r-forêt impaire de F'_i, qui contient un nombre pair d'arborescences, sauf éventuellement si $i = 0$ (propriété (b)), par celle de A''_{i+1}, qui en contient un nombre impair (par induction), ce qui entraîne que parmi ses descendants, x'_i a maintenant un nombre impair de sommets supérieurs à r,
- on remplace la r-forêt impaire de r (par induction, cela revient à enlever tous les fils de r) par celle de F'_i (qui contient un nombre pair d'arborescences) et on greffe ensuite une arborescence ayant un nombre impair de sommets supérieurs à r comme on vient de le voir.

(f) *Pour $i = 0, \ldots, l$, le plus petit fils de la racine r de A''_i est x'_i (la racine de F'_i).*

Les fils de r dans A''_i sont les racines de la r-forêt impaire de F'_i (c'est-à-dire des sommets supérieurs à x'_i car F'_i est une arborescence croissante) et x'_i lui-même.

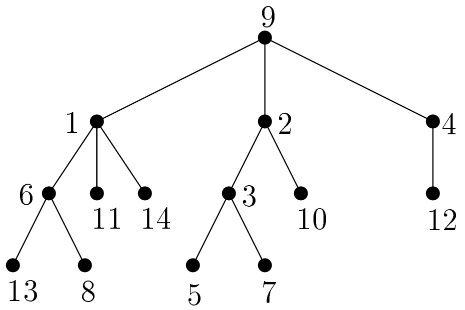
(g) *Pour $i = 0, \ldots, l+1$, les sous-arborescences de A''_i ayant pour racines les fils de r sont croissantes paires.*

Cette propriété découle directement de la construction de A''_i et de la propriété (a) qui assure que les F'_i sont des arborescences croissantes paires.

La propriété (f) permet de définir la construction inverse, donnant F_k et F' à partir d'une arborescence A'' de r-forêt paire vide : tant que le plus petit fils x de la racine r de A'' vérifie $x < r$ (on note A_x la sous-arborescence de A'' de racine x), on supprime l'arête reliant r et x, on échange simultanément les r-forêts impaires de A'' (privé de A_x) et de A_x, puis on ajoute l'arborescence A_x ainsi modifié à F' ; la dernière arborescence A'' restante pour laquelle r est inférieur à tous ses fils est F_k.

Étape 4. Finalement, la dernière étape consiste à insérer sous la racine r de A'' toutes les arborescences de F''. On déduit des propriétés (c), (d), (e) et (g) que l'arborescence A' ainsi obtenue appartient bien à $\overline{\mathcal{B}}_{n,0,r}$.

A'



La construction inverse, donnant F'' et A'' à partir de A', est basée sur les propriétés (c) et (e) et sur le fait que F_0 est de racine 1 et n'appartient pas à F''. Elle consiste à déterminer toutes les sous-arborescences de A' de racine autre que 1 ayant une r-forêt paire contenant un nombre pair d'arborescences. Les arborescences ainsi déterminées de A' forment F''. □

Nous avons donc achevé la première étape de la preuve de la proposition 59, en exhibant un sous-ensemble de $\mathcal{A}_{n,0,r}$ de cardinal $e_{n,r}$ ne contenant que des arborescences ayant un nombre pair d'inversions.

Partition de la famille $\mathcal{A}_{n,0,r}$. Cette deuxième étape a pour objectif de partitionner $\mathcal{A}_{n,0,r}$ en sous-ensembles $C_1, \ldots, C_l$ contenant chacun au plus une arborescence de $\mathcal{B}_{n,0,r}$. Nous allons définir cette partition en termes de classes d'équivalence d'une relation basée sur la notion de *sommets échangeables*, notion introduite par Pansiot [123].

Définition 65. Soit A une arborescence, x et y deux sommets de A tels que y est un descendant de x. On dit que x et y sont *échangeables* si il n'existe pas de sommet z descendant de x vérifiant $x < z < y$ ou $y < z < x$.

Par exemple, dans l'arborescence A_1 de la figure 3.4 ci-dessous, les sommets 5 et 6 sont échangeables, alors que les sommets 5 et 8 ne le sont pas, ni les sommets 1 et 3. Il est clair qu'échanger deux sommets échangeables inverse la parité du nombre d'inversions.

Définition 66. Soient A et A' deux arborescences de $\mathcal{A}_{n,0,r}$. On dit que A et A' sont *équivalentes pour l'échange racine-contraint* si il existe k arborescences $A_1, \ldots, A_k$ de $\mathcal{A}_{n,0,r}$ telles que $A_1 = A$, $A_k = A'$ et, pour $0 < i < k$, A_i et A_{i+1} diffèrent par l'échange de deux sommets échangeables (racine-contraint signifie ici que la racine reste inchangée et toujours inférieure à ses fils).

Notation. Pour une arborescence A de $\mathcal{A}_{n,0,r}$, on note $C_r(A)$ la clôture transitive de A pour la relation d'équivalence définie par l'échange racine-contraint.

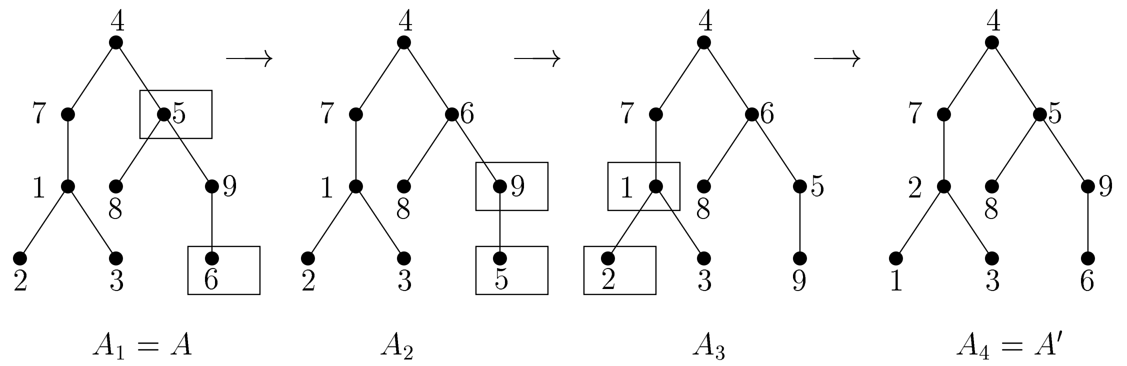


FIG. 3.4 – Illustration de l'équivalence pour l'échange racine-contraint

La relation d'équivalence ainsi définie à partir de la notion d'échange racine-contraint induit donc une partition de $\mathcal{A}_{n,0,r}$ en classes d'équivalence, les classes $C_r(A)$ pour toutes les arborescences A de $\mathcal{A}_{n,0,r}$. Pour vérifier que toute classe contient au plus une arborescence de $\mathcal{B}_{n,0,r}$, on introduit une deuxième relation d'équivalence.

Définition 67. Soit $k \in [n]$. On dit que deux arborescences A et A' sont *équivalentes pour l'échange k-contraint* si il existe m arborescences $A_1, \ldots, A_m$ telles que $A_1 = A$, $A_m = A'$, pour tout i la racine de A_i est supérieure ou égale à k et, pour $0 < i < m$, on passe de A_i à A_{i+1} par l'échange de deux sommets échangeables.

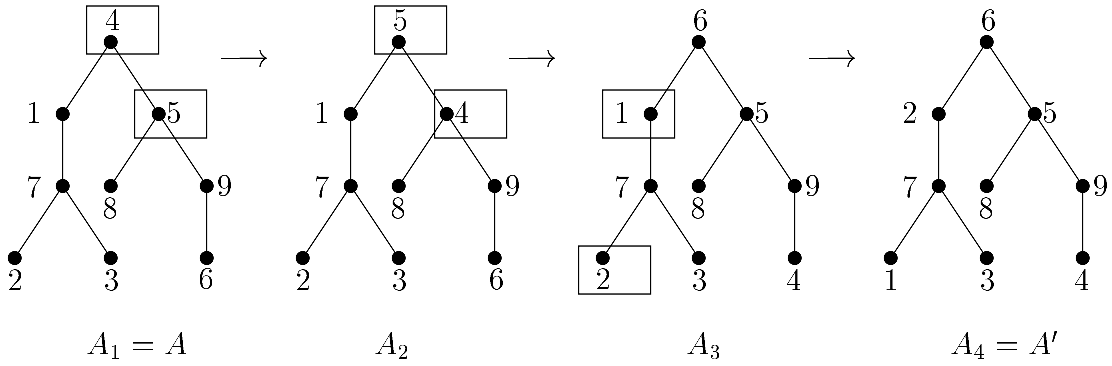


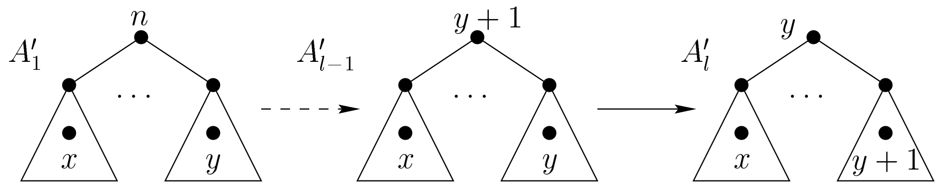
FIG. 3.5 – Illustration de l'équivalence pour l'échange 4-contraint

Notation. Pour une arborescence A donnée, on note $C^k(A)$ la clôture transitive de A pour la relation d'équivalence définie par l'échange k-contraint.

Lemme 68. *Soient k et n deux entiers tels que $k \in [n]$, et A une arborescence sur $[n]$ de racine supérieure ou égale à k. Alors $C^k(A)$ contient* une et une seule *arborescence décroissante.*

Preuve. Il est clair que la classe $C^k(A)$ contient au moins une arborescence décroissante. En effet, par une série d'échanges k-contraint, on peut faire "remonter" n à la racine : échange de la racine x avec $x+1$, puis $x+1$ avec $x+2$, etc et on réitère ce procédé pour chacune des sous-arborescences en "remontant" le plus grand sommet, ... jusqu'à obtenir une arborescence décroissante.

Il reste alors à montrer qu'il n'y en a qu'une seule. Pour cela prenons A' et A'' deux arborescences décroissantes (de racine n donc) équivalentes pour l'échange k-contraint. Supposons qu'il existe deux sommets x, $y > k$ tels que le plus proche ancêtre commun de x et y est la racine n dans A', et un sommet différent de n dans A''. Il existe donc $A' = A'_1, \ldots, A'_m = A''$ telles que A'_i et A'_{i+1} diffèrent par un échange k-contraint. Du fait que y doit être "transporté" dans la sous-arborescence de la racine contenant x, on en déduit qu'il existe $l < m$ tel que la racine de A'_l est y et $y+1$ n'est pas dans la sous-arborescence contenant x. En effet, y devant passer par la racine, n doit la quitter et ne peut être échangé qu'avec $n-1$, puis $n-1$ seulement avec $n-2$, ..., et enfin $y+1$ avec y.



Comme n est la racine de A'_m, un raisonnement analogue conduit à affirmer qu'il faudra effectuer de nouveau l'échange $(y,y{+}1)$ et donc extraire y de la sous-arborescence contenant x. Donc, si A' et A'' sont deux arborescences décroissantes appartenant à $C^k(A)$, le plus proche ancêtre commun du couple de sommets (x,y) dans A' est n si et seulement si le plus proche ancêtre commun du couple de sommets (x,y) dans A'' est n. En répétant récursivement ce raisonnement sur les sous-arborescences de A' et A'' ayant pour racines les fils de n, on montre que tout couple de sommets (x,y) a même plus proche ancêtre commun dans A' et A'', et donc que $A' = A''$. □

Notation. Pour une arborescence A, de racine supérieure ou égale à k, on note $Dec^k(T)$ l'unique arborescence décroissante de $C^k(T)$.

Lemme 69. *Soit $A \in \mathcal{A}_{n,0,r}$. $C_r(A)$ contient* une et une seule *arborescence A' telle que sa racine r est inférieure à ses fils $x_1,\dots,x_k$ et ses sous-arborescences $A'_1,\dots,A'_k$ de racines respectives $x_1,\dots,x_k$ sont décroissantes.*

Preuve. Comme on effectue seulement des échanges racine-contraints, si deux arborescences A' et A'' appartiennent à $C_r(A)$, le plus proche ancêtre commun de (x,y) dans A' est r si et seulement si le plus proche ancêtre commun de (x,y) dans A'' est r. En appliquant le lemme 68 aux sous-arborescences de A' et A'' ayant pour racines les fils de r, on prouve bien qu'au plus une des deux arborescences A' et A'' peut vérifier les conditions du lemme. Le fait qu'il en existe une s'obtient par un raisonnement identique à celui employé dans la preuve précédente. □

Notation. Pour une arborescence $A \in \mathcal{A}_{n,0,r}$, on note $Dec_r(A)$ l'unique arborescence de $C_r(A)$ vérifiant les propriétés du lemme 69.

Nous avons donc complété la deuxième étape de la preuve de la proposition 59, en décrivant une partition de $\mathcal{A}_{n,0,r}$ en ensembles tels que chacun d'entre eux contient au plus une arborescence de $\mathcal{B}_{n,0,r}$ (chacun contient une seule arborescence dont la racine est inférieure à ses fils et les sous-arborescences sont décroissantes).

Évaluation en -1 du polynôme énumérateur des inversions des classes $C_r(A)$. Dans cette dernière partie de la preuve, pour toute arborescence A on note $\mathcal{P}_{C_r(A)}(q)$ (resp. $\mathcal{P}_{C^k(A)}(q)$) le polynôme énumérateur des inversions de l'ensemble d'arborescences $C_r(A)$ (resp. $C^k(A)$). Comme précédemment, on s'intéresse d'abord aux classes d'équivalence pour l'échange k-contraint.

Lemme 70. *Soit A une arborescence sur $[n]$ de racine n et $B = Dec^n(A)$. $\mathcal{P}_{C^n(A)}(-1) = 0$ si B est une arborescence décroissante paire et $\mathcal{P}_{C^n(A)}(-1) = (-1)^{inv(B)}$ sinon.*

Preuve. Dans [123], Pansiot établit une involution ψ sur les arborescences non décroissantes paires et de racine n telle que A et $\psi(A)$ diffèrent par un échange n-contraint ($inv(A)$ et $inv(\psi(A))$ n'ont donc pas même parité). Pour toute arborescence de racine n, la classe $C^n(A)$ contenant au plus une arborescence décroissante (lemme 68), le lemme est une conséquence directe du résultat de Pansiot. □

Lemme 71. *Soient $k \in [n]$, A une arborescence sur $[n]$ de racine k et $B = Dec^k(A)$. $\mathcal{P}_{C^k(A)}(-1) = \mathcal{P}_{C^n(A)}(-1) \times (|B|_{\geq k} \mod 2)$.*

Preuve. On note D_i la classe des arborescences de $C^k(A)$ de racine i (pour $i = k,\dots,n$). On a alors $D_n = C^n(B)$ et on construit D_i à partir de D_{i+1} en échangeant les sommets i et $i+1$ dans toutes les arborescences de D_{i+1}. Par construction des D_i, on vérifie immédiatement que $\mathcal{P}_{D_i}(-1) = -\mathcal{P}_{D_{i+1}}(-1)$. Le lemme découle alors du fait que $\mathcal{P}_{C^k(A)}(-1) = \sum_{i=k}^{n} \mathcal{P}_{D_i}(-1)$. □

Lemme 72. *Soit* $A \in \mathcal{A}_{n,0,r}$. $\mathcal{P}_{C_r(A)}(-1) = 1$ *si* $Dec_r(A) \in \mathcal{B}_{n,0,r}$ *et* $P_{C_r(A)}(-1) = 0$ *sinon.*

Preuve. Soient $x_1, \ldots, x_k$ les fils de r dans $Dec_r(A)$ et $A'_1, \ldots, A'_k$ les sous-arborescences de $Dec_r(A)$ de racines respectives $x_1, \ldots, x_k$. On note D_i (resp. D'_i) l'ensemble d'arborescences $C^r(A'_i)$ (resp. $C^{x_i}(A'_i)$). Toute arborescence A'' de $C_r(A)$ est constituée d'une racine r et de k arborescences $A''_1, \ldots, A''_k$ appartenant respectivement à $D_1, \ldots, D_k$. On a donc

$$(-1)^{inv(A'')} = (-1)^{r-1} \times \prod_{i=1}^{k} (-1)^{inv(A''_i)}.$$

On en déduit que

$$\mathcal{P}_{C_r(A)}(-1) = (-1)^{r-1} \times \prod_{i=1}^{k} \mathcal{P}_{C^r(A'_i)}(-1)$$

ou de manière équivalente (lemme 71)

$$\mathcal{P}_{C_r(A)}(-1) = (-1)^{r-1} \times \left(\prod_{i=1}^{k} \mathcal{P}_{C^{x_i}(A'_i)}(-1) \times (|A'_i|_{\geq r} \mod 2) \right),$$

ce qui est aussi équivalent à

$$\mathcal{P}_{C_r(A)}(-1) = (-1)^{inv(Dec^r(A))} \times \left(\prod_{i=1}^{k} |\mathcal{P}_{C^{x_i}(A'_i)}(-1)| \times (|A'_i|_{\geq r} \mod 2) \right), \tag{3.22}$$

On a donc $\mathcal{P}_{C_r(A)}(-1) = 0$ si et seulement si il existe $i \in [k]$ tel que $\mathcal{P}_{C^{x_i}(A'_i)}(-1) = 0$ ou $(|A'_i|_{\geq r} \mod 2) = 0$, ce qui est équivalent à dire que $Dec_r(A) \notin \mathcal{B}_{n,0,r}$. Donc si $Dec_r(A) \in \mathcal{B}_{n,0,r}$ on a $\mathcal{P}_{C_r(A)}(-1) = 1$ ou $P_{C_r(A)}(-1) = -1$, et si $Dec_r(A) \notin \mathcal{B}_{n,0,r}$ on a $\mathcal{P}_{C_r(A)}(-1) = 0$. Or si $Dec_r(A) \in \mathcal{B}_{n,0,r}$, les A'_i sont des arborescences décroissantes paires. Donc, d'après le lemme 70, qui permet d'affirmer que pour tout i, $|\mathcal{P}_{C^{x_i}(A'_i)}(-1)| = 1$, et le lemme 62, qui permet d'affirmer que $inv(Dec_r(A))$ est pair, on a bien $\mathcal{P}_{C_r(A)}(-1) = 1$. □

Ceci conclut la preuve de la proposition 59, qui est une conséquence du théorème 63, des lemmes 64, 69 et 72, et par là-même ce chapitre consacré au polynôme énumérateur des inversions des arborescences de Cayley dont la racine est inférieure à ses fils.

Chapitre 4

Arborescences alternantes

Dans ce chapitre, nous nous intéressons à l'énumération d'arborescences alternantes, définies de la manière suivante.

Définition 73. Une arborescence A est dite *alternante* si pour tout chemin x_1,x_2, x_3, x_4, $\ldots$ dans A, on a $x_1 < x_2 > x_3 < x_4 > \cdots$ ou bien $x_1 > x_2 < x_3 > x_4 < \cdots$.

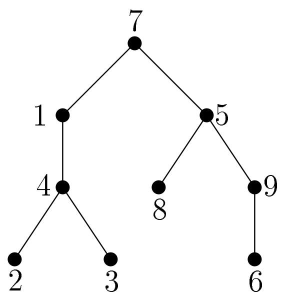


FIG. 4.1 – Une arborescence alternante

Les arborescences de Cayley alternantes ont été introduites par Postnikov [125, 124], qui montre que le nombre de telles arborescences sur $[n-1]$ et de racine 1 est égal au nombre de régions de l'arrangement d'hyperplans de R^n défini par Linial. Postnikov prouve notamment, à l'aide de l'inversion de Lagrange, une formule donnant le nombre de ces arborescences, et pose la question d'une preuve bijective de ce résultat.

Dans un premier temps nous répondons à cette question. Dans la section suivante, nous nous intéressons aux arborescences ordonnées alternantes et nous relions ces objets aux arborescences de la famille $\mathcal{A}_{n,0}$.

4.1 Arborescences de Cayley alternantes

4.1.1 Les résultats de Postnikov

On note $\mathcal{F}_n$ la famille des arborescences de Cayley alternantes sur $[n]$ et $f_n = |\mathcal{F}_n|$.

Théorème 74. [125, 124] *Le nombre f_n d'arborescences de Cayley alternantes sur $[n]$ est*

$$\frac{1}{2^{n-1}}\sum_{k=1}^{n}\binom{n}{k}k^{n-1}. \tag{4.1}$$

Pour prouver ce résultat, Postnikov considère les arborescences de Cayley alternantes de racine étiquetée 1 (nous notons cette famille $\tilde{\mathcal{F}}_n$) et la série génératrice exponentielle (décalée) de $\tilde{\mathcal{F}}_n$

$$\tilde{F}(x)=\sum_{k\geq 0}\tilde{f}_{k+1}\frac{x^k}{k!}.$$

Il montre que cette série vérifie l'équation fonctionnelle

$$\tilde{F}(x)=e^{\frac{x}{2}(\tilde{F}(x)+1)}, \tag{4.2}$$

et en déduit (4.1) à l'aide de la formule d'inversion de Lagrange.

D'autre part Postnikov relie ces arborescences et les *arborescences LBS* (LBS pour *Local Binary Search*), introduites par Gessel.

Définition 75. Une *arborescence LBS* est une arborescence ordonnée binaire (chaque noeud a un ou deux fils, un *fils droit* et/ou un *fils gauche*) telle que pour chaque noeud x, son fils droit, s'il existe, a une étiquette supérieure à x et son fils gauche, s'il existe, a une étiquette inférieure à x.

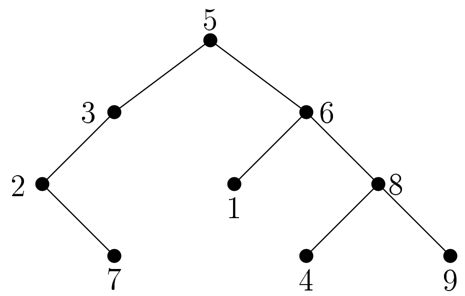


FIG. 4.2 – Une arborescence LBS à 9 sommets.

Remarque 76. Dans le cas des arborescences binaires de recherche, la contrainte sur l'ordre des sommets est globale : un sommet est supérieur (resp. inférieur) à tous les sommets de sa sous-arborescence gauche (resp. droite). Ici cette contrainte n'est que locale.

On note $\mathcal{LBS}_n$ la famille des arborescences LBS sur $[n]$ pour lesquelles la racine n'a qu'un seul fils. Dans [125], Postnikov établit le résultat suivant.

Proposition 77. *Il existe une bijection ϕ entre $\mathcal{F}_n$ et $\mathcal{LBS}_n$.*

La bijection ϕ procède comme suit. Soit $A \in \mathcal{F}_n$. Pour chaque sommet x de A, soient $x_1 < \cdots < x_k$ les fils de x. Si $x < x_1$, alors on place $x_1, x_2, \ldots, x_k$ consécutivement sur une branche droite issue de x. Sinon, $x > x_k$, et on place $x_1, x_2, \ldots, x_k$ consécutivement sur une branche gauche issue de x. La construction inverse est claire.

Dans le paragraphe suivant nous décrivons une preuve bijective du théorème 74, répondant ainsi à la question posée par Postnikov.

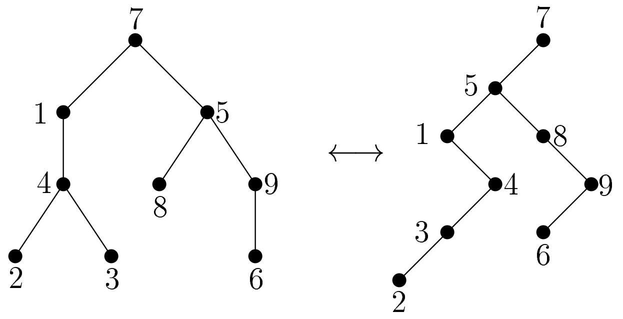


FIG. 4.3 – Illustration de la bijection ϕ

4.1.2 Une preuve bijective du théorème 74

Soit $\mathcal{LBS}_n^c$ la famille des arborescences LBS bicoloriées sur $[n]$ telles que la racine a un seul fils : tout sommet autre que la racine est soit noir, soit blanc, la racine étant toujours noire. Une conséquence directe de la proposition 77 est :

$$|\mathcal{LBS}_n^c| = 2^{n-1} f_n. \tag{4.3}$$

Soit $\mathcal{A}_n^c$ la famille des arborescences de Cayley sur $[n]$ dont les feuilles sont bicoloriées (blanches ou noires), tous les noeuds étant noirs. Pour une arborescence A de $\mathcal{A}_n^c$, on note $n(A)$ son nombre de noeuds, $f_N(A)$ (resp. $f_B(A)$) son nombre de feuilles noires (resp. blanches). En utilisant le codage de Prüfer, une arborescence A de $\mathcal{A}_n^c$ vérifiant $n(A) + f_N(A) = k$ $(1 \leq k \leq n)$ est codée par un couple (S,m) où $S \subseteq [n]$ est un sous-ensemble de $[n]$ de taille k (les noeuds et les feuilles noirs) et m un mot de longueur $n-1$ sur l'alphabet S (les feuilles noires sont les lettres de S n'apparaissant pas dans le mot m, les feuilles blanches étant les lettres de $[n]/S$). On en déduit que

$$|\mathcal{A}_n^c| = \sum_{k=1}^{n} \binom{n}{k} k^{n-1}. \tag{4.4}$$

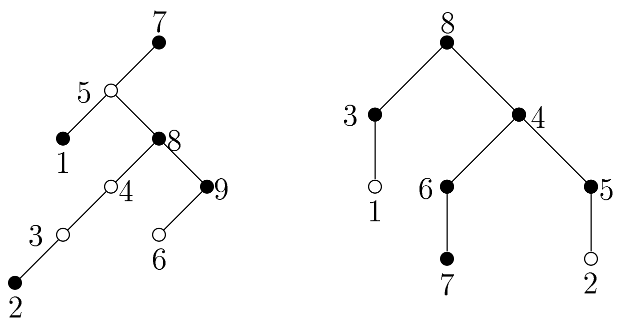


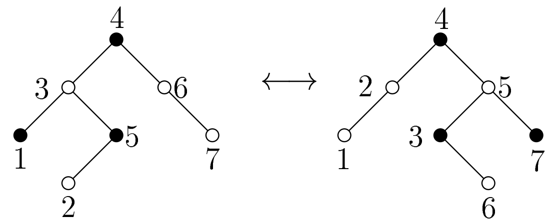
FIG. 4.4 – Une arborescence de $\mathcal{LBS}_9^c$ et une arborescence de $\mathcal{A}_8^c$

Proposition 78. *Il existe une bijection ψ entre $\mathcal{LBS}_n^c$ et $\mathcal{A}_n^c$.*

Preuve. Nous commençons par introduire une construction intermédiaire, à savoir une involution θ sur $\mathcal{LBS}_n^c$. Soit A une arborescence de $\mathcal{LBS}^c$ sur l'ensemble d'étiquettes $\{x_1,x_2,\ldots,x_k\}$, avec $x_1 < x_2 < \cdots < x_k$. L'arborescence LBS bicoloriée $\theta(A)$ est obtenue en effectuant les opérations suivantes :

- pour $1 \leq i \leq k$, échanger les étiquettes x_i et x_{k-i+1}, les couleurs restant attachées aux sommets ;
- pour chaque noeud, échanger les sous-arborescences droite et gauche (l'ordre sur les étiquettes ayant été inversé, il est nécessaire de procéder à cet échange gauche-droite pour obtenir une arborescence LBS).

La figure suivante illustre cette involution.



Soit $A \in \mathcal{A}_n^c$. On transforme récursivement A de manière à obtenir une arborescence de $\mathcal{LBS}_n^c$ de la façon suivante.

Cas 1. Si A est réduite à un sommet, alors $\psi(A) = A$, quelle que soit la couleur de ce sommet.

Cas 2. Si A a deux sommets, une racine et une feuille, on distingue deux cas :

- si la racine x de A est inférieure à son fils y, alors $\psi(A)$ est l'arborescence binaire de racine x ayant un fils droit y (la couleur de y restant inchangée et x restant noir),
- si la racine x de A est supérieure à son fils y, alors $\psi(A)$ est l'arborescence binaire de racine x ayant un fils gauche y (la couleur de y restant inchangée et x demeurant noir).

Propriété 79. Dans le cas où A a au plus deux sommets, la racine de $\psi(A)$ a au plus un fils et a la même étiquette que la racine de A. Si A a deux sommets, la racine de $\psi(A)$ est noire.

Cas 3. Supposons que A ait plus de deux sommets. Soient r sa racine, $x_1,\ldots,x_k$ les fils de r et $A_1,\ldots,A_k$ les sous-arborescences de racines respectives $x_1,\ldots,x_k$. On procède récursivement de la manière suivante. Pour chaque A_i, soit $B_i = \psi(A_i)$. Par hypothèse d'induction (propriété 79), la racine de B_i est x_i, cette racine ayant au plus un fils. On transforme alors les B_i en B_i' en utilisant les règles suivantes :

- si B_i a un seul sommet x_i, alors $B_i' = B_i$;
- si x_i a un fils droit (qui, par hypothèse, est son seul fils), alors B_i' est obtenue à partir de B_i en coloriant la racine, étiquetée x_i, en noir ;
- sinon x_i a seulement un fils gauche : on pose $B_i' = \theta(B_i)$ et on colorie sa racine (qui peut alors avoir une étiquette différente de x_i) en noir.

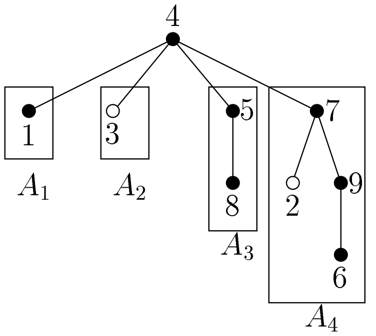
On a alors un ensemble de k arborescences LBS telles que chaque racine a un seul fils, qui est un fils droit, et chaque sommet est colorié en noir ou en blanc. On note $y_1,y_2,\ldots,y_k$ leurs racines respectives, avec $y_1 < y_2 < \cdots < y_k$. On conclut alors la construction de $\psi(T)$ de la manière suivante, qui s'inspire de la bijection ϕ de la proposition 77 :

- si $y_k < r$, alors y_k devient fils gauche de r, y_{k-1} fils gauche de y_k, etc ;

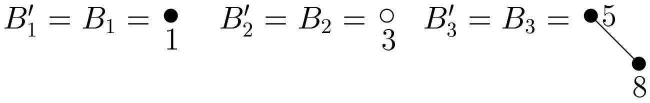
– si $y_k > r$, alors y_k devient fils droit de r, y_{k-1} fils gauche de y_k, y_{k-2} fils gauche de y_{k-1}, etc.

On vérifie immédiatement l'invariant (propriété 79) selon lequel la racine de A, de couleur noire, est aussi la racine de $\psi(A)$ et que cette dernière a un seul fils. □

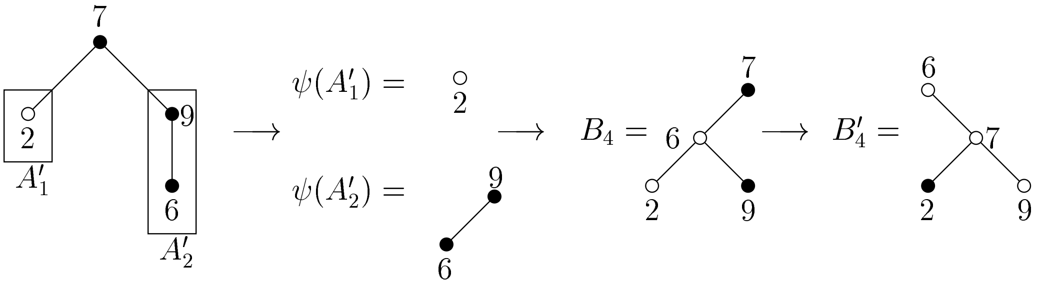
Nous illustrons maintenant cette construction par un exemple détaillé. Soit A l'arborescence de $\mathcal{A}_9^c$ suivante.



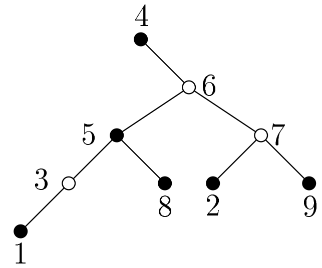
On a alors, pour A_1, A_2 et A_3,



et pour A_4 :



Et finalement, $\psi(A)$ est l'arborescence binaire suivante appartenant à $\mathcal{LBS}_9^c$.



4.1.3 Deux résultats supplémentaires

Pour conclure cette section, nous donnons deux résultats d'énumération sur les arborescences que nous venons de considérer.

Corollaire 80. *Le nombre $f_{n,p}$ d'arborescences LBS sur $[n]$ telles que la racine a un seul fils x et la* branche gauche *issue de x (le chemin débutant en x et n'empruntant que des* *arêtes gauches) comporte p sommets est*

$$\frac{n}{2^{n-1}}\binom{n-2}{p-1}\sum_{k=0}^{n-1}\binom{n-1}{k}k^{n-1-p}.$$

Preuve. On déduit immédiatement de la définition de ψ que $2^{n-1}f_{n,k}$ est le nombre d'arborescences de $\mathcal{A}_n^c$ telles que la racine a exactement k fils. Le codage de Prüfer de telles arborescences de racine $r \in [n]$ est un mot de longueur $n-1$, sur un alphabet $S \subseteq [n]$ (de taille k et comportant la lettre r), tel que la lettre r apparaît exactement p fois dont une fois en dernière position. Le résultat en découle directement. □

Corollaire 81. *Le nombre d'arborescences de Cayley sur $[n]$ telles que chaque noeud a au moins un fils qui lui est supérieur (en d'autres termes, pour tout noeud x, il existe un chemin croissant allant de x à une feuille) est*

$$\frac{f_n}{2} = \frac{1}{2^n}\sum_{k=1}^{n}\binom{n}{k}k^{n-1}.$$

Preuve. On déduit de la bijection ψ que ces arborescences sont en correspondance avec les arborescences de $\mathcal{LBS}_n^c$ telles que le fils de la racine est un fils droit et que tout sommet est colorié en noir. Le résultat provient alors de la proposition 77. On reconnaît ici la correspondance classique entre arborescences binaires et arborescences. □

4.2 Arborescences ordonnées alternantes

Dans cette section, nous nous intéressons à l'énumération des arborescences ordonnées alternantes. On note $\mathcal{K}_n$ la famille des arborescences ordonnées alternantes sur $[n]$ et k_n leur nombre. Nous présentons deux preuves, l'une formelle et l'autre bijective, du résultat suivant.

Théorème 82. *Le nombre d'arborescences ordonnées alternantes sur $[n]$ est*

$$2(n-1)^{n-1}.$$

4.2.1 Une preuve formelle

Pour prouver ce résultat nous nous intéressons à la famille $\mathcal{K}_{n,0}$ des arborescences ordonnées alternantes sur $[n]$ telles que la racine est inférieure à ses fils (on note $k_{n,0}$ le nombres de telles arborescences) et nous montrons que $k_{n,0} = (n-1)^{n-1}$. Soit $K_0(x)$ la série génératrice de ces arborescences :

$$K_0(x) = \sum_{n\geq 1} k_{n,0}\frac{x^n}{n!}.$$

Lemme 83. *La série génératrice $K_0(x)$ satisfait l'équation fonctionnelle*

$$K_0(x) - 1 = -e^{\frac{x}{K_0(x)-1}}.$$

Preuve. Soient $\mathcal{P}_n$ la famille des arborescences planes (voir la section 2.4 pour une définition des arborescences planes) alternantes sur $[n]$ (on pose $p_n = |\mathcal{P}_n|$) et $\mathcal{L}_n$ la famille des arbres ordonnés (graphes - dessinés au sens des cartes - connexes sans cycles *non enracinés*) alternants sur $[n]$ (on pose $l_n = |\mathcal{L}_n|$) [1]. On note $L(x)$ et $P(x)$ les séries génératrices

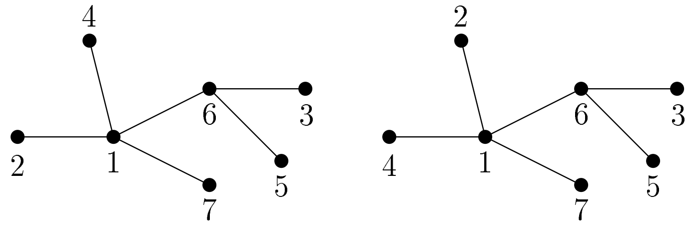


Fig. 4.5 – Deux arbres ordonnés alternants différents

exponentielles respectives des arbres ordonnés alternants $\mathcal{L}$ et des arborescences planes alternantes $\mathcal{P}$. On rappelle de plus que $\mathcal{C}$ désigne la famille des listes circulairement ordonnées.

En remarquant qu'une arborescence de $\mathcal{L}_n$ peut se décomposer en une liste circulairement ordonnée d'arborescences de $\mathcal{K}_{n-1,0}$ reliées au sommet n et que la série formelle

$$\frac{\partial L(x)}{\partial x} = \sum_{n \geq 0} l_{n+1} \frac{x^n}{n!}$$

est en fait la série génératrice exponentielle des arbres ordonnés alternants dans lesquels le sommet maximal n'est pas "compté" par la variable formelle x, on en déduit que

$$\frac{\partial L(x)}{\partial x} = \ln \left(\frac{1}{1 - K_0(x)} \right).$$

Comme on obtient une arborescence de $\mathcal{P}_n$ en pointant un sommet d'un arbre de $\mathcal{L}_n$, on a alors

$$P(x) = x \frac{\partial L(x)}{\partial x} = x \ln \left(\frac{1}{1 - K_0(x)} \right). \tag{4.5}$$

D'autre part, une arborescence plane alternante de $\mathcal{P}_n$ est équivalente à un arbre de $\mathcal{L}_n$ si sa racine est n, et à une arborescence de $\mathcal{K}_{n,0}$ si sa racine est inférieure à n (il suffit d'ordonner linéairement les fils de la racine en considérant comme fils le plus à gauche le seul fils de la racine situé sur le chemin allant de la racine au sommet n). En remarquant que dans le cadre de cette correspondance, si $n = 1$, on considère deux fois la seule arborescence de $\mathcal{P}_1$ (en tant qu'arbre de $\mathcal{L}_1$ et arborescence de $\mathcal{K}_{1,0}$), on obtient l'identité suivante :

$$P(x) = K_0(x) + L(x) - x. \tag{4.6}$$

En dérivant (4.5) et (4.6), on obtient alors

$$1 + \frac{x}{1 - K_0(x)} \frac{\partial K_0(x)}{\partial x} = \frac{\partial K_0(x)}{\partial x}.$$

1. La preuve que nous donnons ici, différente de la preuve présentée dans [37], nous a été indiquée par Volker Strehl.

En multipliant les deux membres de l'identité précédente par $1/(1-K_0(x))$ et en intégrant, on obtient alors

$$\frac{x}{1-K_0(x)} = \ln\left(\frac{x}{1-K_0(x)}\right),$$

le lemme en découlant immédiatement. □

Cette équation permet d'exprimer la série $K_0(x)$ en fonction de la série génératrice $A(x)$ des arborescences de Cayley. En effet, on déduit du lemme 83 que

$$-\frac{x}{K_0(x)-1} = x\ e^{\frac{-x}{K_0(x)-1}}.$$

Comme $A(x) = x\ e^{A(x)}$, on en déduit que

$$-\frac{x}{K_0(x)-1} = A(x),$$

ce qui entraîne immédiatement que

$$K_0(x) - 1 = -\frac{1}{e^{A(x)}}.$$

On peut alors applique l'inversion de Lagrange (théorème 14) avec $G(x) = e^x$ et $H(x) = -e^{-x}$:

$$[x^n](K_0(x)-1) = \frac{1}{n}[x^{n-1}]e^{(n-1)x}$$

$$\Longrightarrow [x^n](K_0(x)-1) = [x^{n-1}]\sum_{k\geq 0}\frac{(n-1)^k x^k}{n\ (k)!} \Longrightarrow [x^n](K_0(x)-1) = \frac{(n-1)^{n-1}}{n!}.$$

Ceci termine la preuve du théorème 82.

4.2.2 Une preuve bijective

Le nombre d'arborescences ordonnées alternantes sur $[n]$ telles que la racine est inférieure à ses fils étant $(n-1)^{n-1}$, les résultats du chapitre 2, et notamment le lemme 17, incitent à chercher une bijection entre $\mathcal{K}_{n,0}$ et $\mathcal{A}_{n,0}$. Dans ce paragraphe, nous décrivons une telle correspondance.

Définition 84. Soit A une arborescence et x un sommet de A, différent de la racine et ayant y pour père : x est une *double-descente* (resp. *double-montée* si $y > x$ (resp. $y < x$) et x a un fils $z < x$ (resp. $z > x$).

Propriété 85. Une arborescence est alternante si et seulement si elle ne possède ni double-descente, ni double-montée.

Définition 86. Un *parcours en profondeur préfixe* d'une arborescence A est un parcours récursif débutant à la racine de A et dans lequel, après avoir visité un sommet x (ayant k fils ordonnés (de gauche à droite) $x_1, \ldots, x_k$), on visite successivement A_{x_1}, puis A_{x_2}, etc.

Algorithme 4: De $\mathcal{A}_{n,0}$ vers $\mathcal{K}_{n,0}$

Données : A : arborescence de $\mathcal{A}_{n,0}$
Résultat : A' : arborescence de $\mathcal{K}_{n,0}$
début
  pour chaque sommet x de A **faire**
    ordonner ses fils en croissant de gauche à droite;
  tant que A n'est pas une arborescence ordonnée alternante **faire**
    effectuer un parcours en profondeur préfixe de A et soit x le premier sommet visité tel que son père y est double-montée (resp. double-descente);
    soient $x_1, \ldots ,x_k$ les fils de x tels que $x_i > x$ (resp. $x_i < x$) et $A_1, \ldots ,A_k$ les sous-arborescences de racines respectives $x_1, \ldots ,x_k$;
    placer les A_i à gauche (resp. droite) de y : x_k devient le frère gauche de x, x_{k-1} celui de x_k, etc (resp. x_1 devient le frère droit de x, x_2, celui de x_1, etc);
fin

Cet algorithme consiste donc à éliminer les doubles-descentes et les doubles-montées jusqu'à obtenir une arborescence ordonnée alternante. La figure qui suit présente une exécution de cet algorithme, où à chaque étape les arêtes reliant x aux sommets x_i mis en jeu sont données en pointillé.

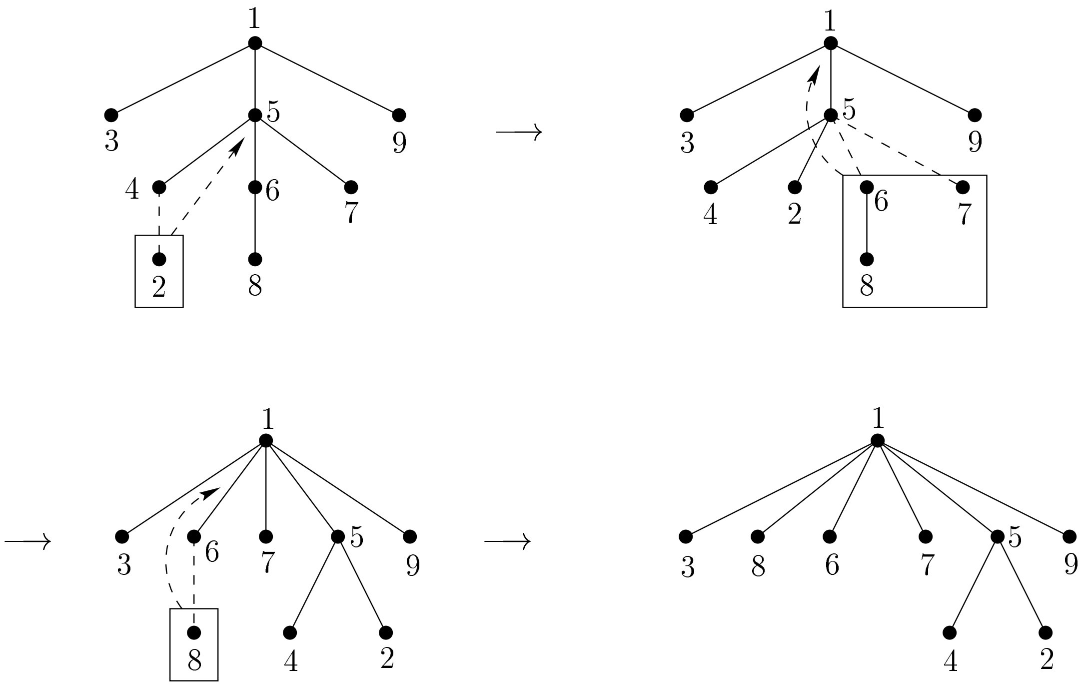


FIG. 4.6 – Exemple d'exécution de l'algorithme 4

Algorithme 5: De $\mathcal{K}_{n,0}$ vers $\mathcal{A}_{n,0}$

Données : A : arborescence de $\mathcal{K}_{n,0}$
Résultat : A' : arborescence de $\mathcal{A}_{n,0}$
début
  pour chaque sommet x de A différent de la racine **faire**
    si le père de x est inférieur (resp supérieur) à x **alors**
      soient y le premier frère de x situé à sa gauche (resp. droite) tel que $y < x$ (resp. $y > x$), $x_1 < \cdots < x_k$ les sommets situés entre y et x (resp. entre x et y) et $A_1, \ldots ,A_k$ les sous-arborescences de racines respectives $x_1, \ldots ,x_k$;
      déplacer les A_i à la droite du dernier (resp. gauche du premier) fils de x : x_k est le dernier fils de x, x_{k-1} le frère gauche de x_k, etc (resp. x_1 est le premier fils de x, x_2 le frère droit de x, etc);
fin

On vérifie aisément que ces deux algorithmes définissent bien une bijection entre $\mathcal{K}_{n,0}$ et $\mathcal{A}_{n,0}$, ce qui prouve le théorème 82.

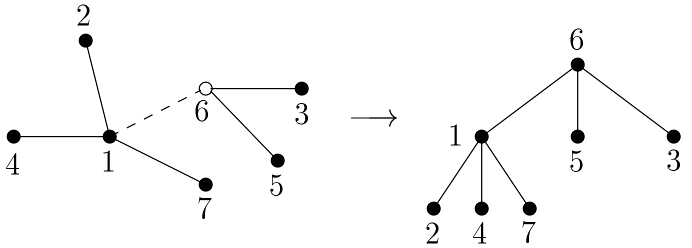
4.2.3 Arbres ordonnés et arborescences planes alternants

Si on considère les arbres (arborescences non enracinées) ordonnés alternants, on obtient le résultat suivant.

Corollaire 87. *Le nombre d'arbres ordonnés alternants non enracinés sur* $[n]$ *est*

$$(n-1)^{n-2}.$$

Preuve. On rappelle que k_n est le nombre d'arborescences ordonnées alternantes et l_n le nombre d'arbres ordonnés alternants. On a une bijection entre les triplets $(A,(x,y),z)$ où A est un arbre ordonné alternant sur $[n]$, (x,y) une arête de A, et z un des sommets de cette arête (soit x, soit y), et les arborescences ordonnées alternantes sur $[n]$: pour obtenir une arborescence ordonnée à partir de $(A,(x,y),z)$, on enracine A en z et on impose que son premier fils est l'autre sommet de l'arête (x,y).



On en déduit l'identité

$$k_1 = l_1 = 1,\ k_n = 2(n-1)l_n\ (n > 1),$$

qui, combinée au théorème 82 conclut la preuve. □

Considérons maintenant les arborescences planes alternantes. Tout arbre ordonné alternant détermine, par le choix d'un sommet racine, une arborescence plane alternante. Il s'ensuit le corollaire suivant.

Corollaire 88. *Le nombre d'arborescence planes alternantes sur* $[n]$ *est*

$$n(n-1)^{n-2}.$$

Il est alors naturel de se poser la question suivante.

Question 89. Quel est le nombre d'arborescences cycliques alternantes sur $[n]$? Pour $n = 9$, on obtient les premières valeurs suivantes calculées expérimentalement.

$$1 \quad 2 \quad 6 \quad 30 \quad 216 \quad 2026 \quad 23406 \quad 321678 \quad 5097760$$

La consultation de l'ouvrage classique de Sloane et l'interrogation du serveur Web associé, [144, 143], ne donnent aucune information supplémentaire sur cette suite de nombres.

Chapitre 5

La formule de Lagrange multidimensionnelle

Dans ce chapitre, nous présentons une extension de la formule de Lagrange aux structures combinatoires coloriées et aux séries multivariées, connue sous le nom de formule de Good. Après une présentation de cette formule et d'une de ses variantes (section 5.1), nous proposons une nouvelle preuve combinatoire de cette variante (section 5.2). Nous concluons en illustrant l'intérêt des preuves combinatoires de la formule de Lagrange multidimensionnelle, qui permettent de fournir des explications simples de formules obtenues par application de la formule de Good (sections 5.3 et 5.4), et de déduire de ces explications des algorithmes de génération aléatoire (section 5.4).

5.1 Introduction

Dans les chapitres précédents, nous avons utilisé la formule d'inversion de Lagrange pour énumérer des structures arborescentes selon un seul paramètre, à savoir le nombre de sommets. Il est cependant fréquent de vouloir énumérer des structures combinatoires selon plusieurs paramètres. Pour les arborescences, on peut par exemple penser au nombre de sommets et au nombre de feuilles. On peut aussi considérer des arborescences dont les sommets sont de diverses sortes, que l'on représente habituellement par des couleurs (à chaque sorte de sommets correspond une couleur), la sorte d'un sommet pouvant influencer la structure de sa fibre. On parle alors d'arborescences coloriées (ou multisortes).

Par exemple la famille des arborescences de Cayley bichromatiques est la famille des arborescences telles que

- chaque sommet est colorié avec une couleur choisie parmi 2 (blanc ou noir, par exemple),
- la fibre de chaque sommet est munie d'une $\mathcal{E}$-structure (structure d'ensemble),
- la fibre d'un sommet blanc (resp. noir) ne contient pas de sommet blanc (resp. noir).

Si une telle arborescence a n_1 (resp. n_2) sommets blancs (resp. noirs), on étiquette ces sommets par $\{1,2,\ldots,n_1\}$ (resp. $\{1,2,\ldots,n_2\}$), la couleur permettant de distinguer les sommets ayant même étiquette.

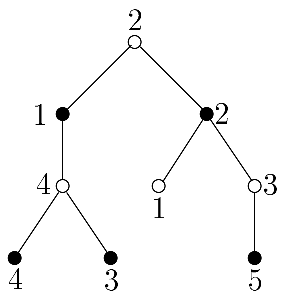


FIG. 5.1 – Une arborescence de Cayley bichromatique

Il est alors naturel de vouloir énumérer ces arborescences non plus seulement selon le nombre de sommets (le nombre d'arborescences bichromatiques à n sommets est $2n^{n-1}$, la couleur de la racine déterminant celle de tous les autres sommets), mais suivant le nombre de sommets de chaque couleur. Pour cela, on associe à une telle famille d'arborescences une *série génératrice multivariée*: si a_{n_1,n_2} est le nombre d'arborescences de Cayley bichromatiques ayant n_1 (resp. n_2) sommets blancs (resp. noirs), la série génératrice de ces arborescences comptées selon le nombre de sommets de chaque couleur est

$$A(x_1,x_2) = \sum_{n_1,n_2 \geq 0} a_{n_1,n_2} \frac{x_1^{n_1}}{n_1!} \frac{x_2^{n_2}}{n_2!}.$$

Si on note $A_1(x_1,x_2)$ (resp. $A_2(x_1,x_2)$) la série génératrice des arborescences de Cayley bichromatiques de racine de couleur blanche (resp. noire), les séries $A(x_1,x_2)$, $A_1(x_1,x_2)$, $A_2(x_1,x_2)$ vérifient le système d'équations fonctionnelles suivant :

$$A(x_1,x_2) = A_1(x_1,x_2) + A_2(x_1,x_2), \tag{5.1}$$

$$A_1(x_1,x_2) = x_1 e^{A_2(x_1,x_2)}, \tag{5.2}$$

$$A_2(x_1,x_2) = x_2 e^{A_1(x_1,x_2)}. \tag{5.3}$$

5.1.1 La formule de Good

Pour traiter ce type de problèmes d'énumération de structures arborescentes coloriées, on peut utiliser la généralisation suivante de la formule de Lagrange, communément appelée *inversion de Lagrange multidimensionnelle* ou *formule de Good*, puisqu'elle est due à Good [72].

Théorème 90. [72] *Soient m un entier, $\mathbf{x}$ le vecteur $(x_1,\ldots,x_m)$ de variables formelles, $H(\mathbf{x}),G_1(\mathbf{x}),\ldots,G_m(\mathbf{x})$ des séries formelles multivariées telles que $G_i(0,\ldots,0) \neq 0$ pour $i = 1,\ldots,m$. L'ensemble d'équations*

$$F_i(\mathbf{x}) = x_i G_i(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x})),\ i = 1,\ldots,m \tag{5.4}$$

détermine alors la série formelle $H(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x}))$ par :

$$[\mathbf{x}^\mathbf{n}] \left\{ \frac{H(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x}))}{\det\left(\delta_{i,j} - x_i \dfrac{\partial G_i(\mathbf{x})}{\partial x_j}\right)_{m\times m}} \right\} = [\mathbf{x}^\mathbf{n}] \left\{ H(\mathbf{x}) \left(\prod_{i=1}^{m} (G_i(\mathbf{x}))^{n_i} \right) \right\}, \tag{5.5}$$

où $[\mathbf{x}^\mathbf{n}] = [x_1^{n_1} \cdots x_m^{n_m}]$ *et* $\delta_{i,j} = 1$ *si* $i = j$ *et* 0 *sinon, ou bien*

(5.6)

$$\{[\mathbf{x}^\mathbf{n}]H(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x}))\} = [\mathbf{x}^\mathbf{n}]H(\mathbf{x})\left(\prod_{i=1}^m (G_i(\mathbf{x}))^{n_i}\right) \det\left(\delta_{i,j} - \frac{x_i}{G_j(\mathbf{x})}\frac{\partial G_j(\mathbf{x})}{\partial x_i}\right)_{m\times m}$$

On remarque alors que dans le cas $m = 1$, les deux identités (5.5) et (5.6) sont équivalentes, et on retrouve, après un rapide calcul, le théorème 14. Les identités (5.5) et (5.6), qui sont appelées respectivement *forme implicite* et *forme explicite* de la formule de Good, sont en fait équivalentes (cf [16, section 3.2]). Dans les problèmes d'énumération, la forme explicite est la plus fréquemment utilisée.

Revenons à notre exemple des arborescences de Cayley bichromatiques. Le système d'équations fonctionnelles (5.1), (5.2) et (5.3) satisfait les conditions du théorème 90 et on peut donc appliquer la forme explicite de la formule de Good avec $H(x_1,x_2) = x_1 + x_2$, $G_1(x_1,x_2) = e^{x_2}$ et $G_2(x_1,x_2) = e^{x_1}$. Le déterminant du membre droit de (5.6) vaut $1 - x_1x_2$, et le nombre d'arborescences de Cayley bichromatiques à n_1 sommets blancs et n_2 sommets noirs, avec $n_1,n_2 > 0$ est donc

$$n_1!n_2!\left([x_1^{n_1}x_2^{n_2}](x_1+x_2)e^{n_1x_2}e^{n_2x_1}(1-x_1x_2)\right),$$

ce qui donne

$$(n_1+n_2)n_1^{n_2-1}n_2^{n_1-1},$$

un résultat dû à Scoins [138].

Le théorème 90 est dû à Good [72], qui l'a utilisé dans le cadre de l'énumération d'arborescences [73, 74] (l'exemple des arborescences de Cayley bichromatiques est d'ailleurs tiré de [73]). Il a ensuite été suivi par plusieurs auteurs, notamment Tutte [158], Knuth [101], Goulden et Jackson [95, 76, 77], qui ont introduit une nouvelle forme de la formule de Lagrange multivariée (dont nous proposons une preuve bijective dans la section 5.2) permettant d'énumérer diverses familles d'arborescences dont la partition des arêtes est fixée, ou encore certaines familles de chemins dans des graphes [75]. Récemment, cette technique a été utilisée dans le cadre de l'énumération de cactus planaires par Goulden et Jackson [78] et Bóna, Bousquet, Labelle et Leroux [21].

Comme nous l'avons mentionné dans le chapitre 1, il existe plusieurs preuves de la formule de Lagrange unidimensionnelle dues notamment à Raney [131] et Labelle [108]. La première preuve combinatoire d'une formule de Lagrange multidimensionnelle est due à Chottin [42]. Cette preuve est une extension de la preuve de Raney, dans le cas $m = 2$ (c'est-à-dire des séries bivariées). Chottin a été le premier à faire le lien entre cette preuve (et donc celle de Raney) et l'énumération d'arborescences planes non étiquetées dont les sommets sont pondérés par les coefficients des séries $G_1(x_1,x_2)$ et $G_2(x_1,x_2)$ [43] (on pourra se référer à Cori [49] pour un exposé de synthèse présentant les résultats de Raney et Chottin). La première preuve combinatoire de la formule de Good pour un nombre quelconque de variables est due à Gessel [62]. Il propose une extension de la preuve de Labelle de la formule unidimensionnelle et démontre ainsi la forme implicite de la formule de Good. Cette preuve de Gessel, basée sur une bijection entre arborescences coloriées et endofonctions coloriées a été reprise dans la cadre de la théorie des espèces par Bergeron, Labelle et Leroux [16, section 3.2]. La première preuve combinatoire de la forme explicite est due à Ehrenborg et Méndez [57], mais cette preuve n'est pas directe. On peut aussi citer les travaux de Strehl [150] (voir aussi [16, exercice 3.2.16]). Goulden et Kulkarni [80] en ont une nouvelle preuve en introduisant une nouvelle expression de la formule de Good, que nous présentons maintenant.

5.1.2 La forme arborescente de la formule de Good

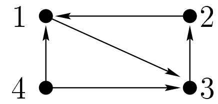
Pour prouver la forme explicite de la formule de Good, Goulden et Kulkarni [80] introduisent une formule d'inversion appelée *forme arborescente de la formule de Good*, car basée sur la notion de dérivée d'un vecteur de séries formelles par rapport à une arborescence orientée.

Définition 91. Soient G un graphe orienté étiqueté ayant pour ensemble de sommets $S = \{1, \ldots, m\}$ et ensemble d'arcs A, $\mathbf{x} = (x_1, \cdots, x_m)$ un vecteur de variables formelles et $\mathbf{R}(\mathbf{x}) = (R_1(\mathbf{x}), \cdots, R_m(\mathbf{x}))$ un vecteur de séries formelles en les variables $\mathbf{x}$. On définit la dérivée de $\mathbf{R}(\mathbf{x})$ par rapport à G par

$$\frac{\partial \mathbf{R}(\mathbf{x})}{\partial G} = \prod_{j \in S} \left\{ \left(\prod_{(i,j) \in A} \frac{\partial}{\partial x_i} \right) R_j(\mathbf{x}) \right\},$$

où $(i,j) \in A$ fait référence à un arc allant du sommet i vers le sommet j.

Par exemple, si $m = 4$ et G est le graphe suivant,



alors,

$$\frac{\partial \mathbf{R}(\mathbf{x})}{\partial G} = \left\{ \left(\frac{\partial}{\partial x_2} \frac{\partial}{\partial x_4} \right) R_1(\mathbf{x}) \right\} \left\{ \frac{\partial R_2(\mathbf{x})}{\partial x_3} \right\} \left\{ \left(\frac{\partial}{\partial x_1} \frac{\partial}{\partial x_4} \right) R_3(\mathbf{x}) \right\} R_4(\mathbf{x}).$$

Goulden et Kulkarni, utilisant cette notion, ont montré que la forme explicite de la formule de Good (5.6) est équivalente à la formule (5.7) qui suit, dans laquelle le déterminant du membre droit de (5.6) est remplacé par une somme de termes positifs définis à partir de dérivées selon des arborescences orientées.

Théorème 92. [80] *Soient m un entier, $\mathbf{x}$ le vecteur $(x_1, \cdots, x_m)$ de variables formelles, $H(\mathbf{x}), G_1(\mathbf{x}), \ldots, G_m(\mathbf{x})$ des séries formelles multivariées telles que $G_i(0, \ldots, 0) \neq 0$ pour $i = 1, \ldots, m$. L'ensemble d'équations*

$$F_i(\mathbf{x}) = x_i G_i(F_1(\mathbf{x}), \ldots, F_m(\mathbf{x})),\ i = 1, \ldots, m$$

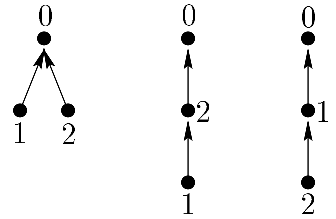
détermine alors de manière unique la série formelle $H(F_1(\mathbf{x}), \ldots, F_m(\mathbf{x}))$ par : .

(5.7)
$$\{[\mathbf{x}^{\mathbf{n}}] H(F_1(\mathbf{x}), \ldots, F_m(\mathbf{x}))\} = \left(\prod_{i=1}^{m} \frac{1}{n_i} \right) [\mathbf{x}^{\mathbf{n}-\mathbf{1}}] \sum_{T} \frac{\partial (H(\mathbf{x}), (G_1(\mathbf{x}))^{n_1}, \ldots, (G_m(\mathbf{x}))^{n_m})}{\partial T},$$

où $\mathbf{n} - \mathbf{1} = (n_1 - 1, \ldots, n_m - 1)$ et la somme porte sur toutes les arborescences orientées T ayant $\{0,1, \ldots, m\}$ pour ensemble de sommets, 0 pour racine et dont tous les arcs sont orientés vers la racine.

On remarque que si $m = 1$, il y a une seule arborescence T (de racine 0 ayant un seul fils, étiqueté 1) et on retrouve la formule d'inversion de Lagrange unidimensionnelle (théorème 14).

Revenons encore à notre exemple des arborescences bichromatiques (équations (5.1), (5.2) et (5.3)). On a $m = 2$ et il y a 3 arborescences sur les sommets $\{0,1,2\}$ de racine 0 dont tous les arcs sont orientées vers 0 :



En appliquant (5.7), le nombre d'arborescences bichromatiques ayant n_1 (resp. n_2) sommets blancs (resp. noirs) est

$$(n_1-1)!(n_2-1)![x_1^{n_1-1}x_2^{n_2-1}]\left(\frac{\partial}{\partial x_1}\frac{\partial}{\partial x_2}(x_1+x_2)\right)e^{n_1x_2}e^{n_2x_1}+$$

$$(n_1-1)!(n_2-1)![x_1^{n_1-1}x_2^{n_2-1}]\frac{\partial(x_1+x_2)}{\partial x_2}e^{n_1x_2}\frac{\partial e^{n_2x_1}}{\partial x_1}+$$

$$(n_1-1)!(n_2-1)![x_1^{n_1-1}x_2^{n_2-1}]\frac{\partial(x_1+x_2)}{\partial x_1}\frac{\partial e^{n_1x_2}}{\partial x_2}e^{n_2x_1},$$

ce qui donne immédiatement

$$(n_1+n_2)n_1^{n_2-1}n_2^{n_1-1}.$$

Goulden et Kulkarni [80] proposent une preuve bijective de la formule 5.7, basée sur une extension de la bijection de Labelle entre arborescences et endofonctions, ce qui constitue la première preuve bijective directe de la forme explicite de la formule de Good. On peut noter qu'une preuve du théorème 92 restreint au cas des séries à deux variables avait été ébauchée par Leroux [110] et que le théorème 92 a été obtenu indépendamment par Bender et Richmond [11]. Ces derniers ont notamment utilisé le fait que le membre droit de (5.7) est constitué d'une somme de termes positifs (contrairement au membre droit de (5.6) faisant intervenir un déterminant) pour en déduire des propriétés asymptotiques de structures "lagrangiennes" coloriées [12].

5.2 Preuve de la forme arborescente de la formule de Good

Dans cette section, nous proposons une nouvelle preuve combinatoire du théorème 92. Cette preuve est basée sur le même principe que celle de Goulden et Kulkarni, à savoir une bijection entre arborescences coloriées et endofonctions coloriées, mais présente l'avantage d'être plus simple à présenter et à prouver.

Nous considérons ici des structures combinatoires m-coloriées. Soient $\mathbf{c} = (c_1,\dots,c_m)$ un vecteur de m couleurs et $\mathbf{x} = (x_1,\dots,x_m)$ un vecteur de m variables formelles. On introduit une classe $\mathcal{H}$ de structures m-coloriées, de série génératrice multivariée $H(\mathbf{x})$ (x_i comptant les éléments de couleur c_i), et un vecteur $\mathcal{G} = (\mathcal{G}_1,\dots,\mathcal{G}_m)$ de m classes de structures combinatoires m-coloriées de séries génératrices multivariées respectives $G_1(\mathbf{x}),\dots,G_m(\mathbf{x})$. On note $[\mathbf{n}] = [n_1]\cup\cdots\cup[n_m]$, $\mathbf{n}! = \prod_{i=1}^m n_i!$, et on appelle structure sur $[\mathbf{n}]$ une structure m-coloriée ayant n_i sommets de couleur c_i, pour $i = 1,\dots,m$.

5.2.1 Arborescences et endofonctions coloriées

($\mathcal{H}$,$\mathcal{G}$)-arborescences. On a vu au chapitre 1 que, lorsque $m = 1$, le nombre de $\mathcal{H}$-assemblées de $\mathcal{G}_1$-arborescences sur $[n_1]$ est égal à $n_1![x_1^{n_1}]H(A_{G_1}(x_1))$, où $A_{G_1}(x_1)$ est la série génératrice exponentielle des $\mathcal{G}_1$-arborescences, qui vérifie l'équation fonctionnelle $A_{G_1}(x_1) = x_1 G_1(A_{G_1}(x_1))$. On peut donc, dans le cas $m = 1$, interpréter le membre gauche de la formule (5.7) en termes de $\mathcal{H}$-assemblées de $\mathcal{G}_1$-arborescences sur $[n_1]$. Pour traiter le cas général ($m \geq 1$) on introduit les notions de $\mathcal{G}$-arborescences et ($\mathcal{H}$,$\mathcal{G}$)-arborescences.

Définition 93. Une $\mathcal{G}$-*arborescence* A sur $[\mathbf{n}]$ est une arborescence m-coloriée sur l'ensemble de sommets $[\mathbf{n}]$ telle que, pour $i = 1,\ldots,m$, la fibre de tout sommet de couleur c_i est munie d'une $\mathcal{G}_i$-structure.

Définition 94. Une ($\mathcal{H}$,$\mathcal{G}$)-*arborescence* sur $[\mathbf{n}]$ est une $\mathcal{H}$-assemblée de $\mathcal{G}$-arborescences. Une ($\mathcal{H}$,$\mathcal{G}$)-arborescence sur $[\mathbf{n}]$ est représentée par une arborescence sur l'ensemble de sommets $\{0\} \cup [\mathbf{n}]$, noté $[0,\mathbf{n}]$, de racine 0, telle que la fibre de cette racine est munie d'une $\mathcal{H}$-structure et, pour $i = 1,\ldots,m$, la fibre de tout sommet de couleur c_i est munie d'une $\mathcal{G}_i$-structure.

Par exemple, la figure 5.2 représente une ($\mathcal{H}$,$\mathcal{G}$)-arborescence avec $m = 3$ (les sommets sont blancs, gris ou noirs). Dans cette figure, on représente la structure de la fibre d'un sommet par un arc de cercle indicé par la structure dont est munie cette fibre. Par la suite, on omettra cet arc de cercle, la structure étant implicitement déterminée par la couleur du sommet.

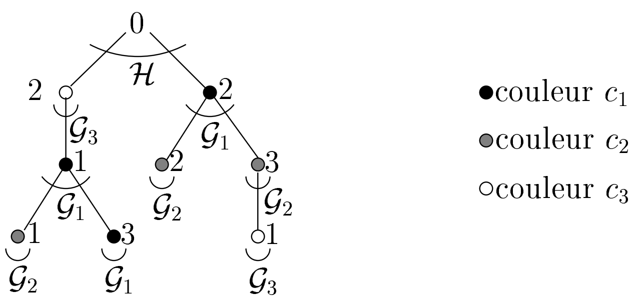


FIG. 5.2 – ($\mathcal{H}$,$\mathcal{G}$)-arborescence à trois couleurs ($m = 3$)

Si on note $A_{(H,\mathbf{G})}(\mathbf{x})$ (resp. $A_{G_i}(\mathbf{x})$, pour $i = 1,\ldots,m$) la série génératrice exponentielle des ($\mathcal{H}$,$\mathcal{G}$)-arborescences (resp. des $\mathcal{G}$-arborescences de racine de couleur c_i), on obtient par une application directe de la composition de séries génératrices (cf section 1.2.2)

$$A_{(H,\mathbf{G})}(\mathbf{x}) = H(A_{G_1}(\mathbf{x}),\ldots,A_{G_m}(\mathbf{x}))$$

$$A_{G_i}(\mathbf{x}) = x_i G_i(A_{G_1}(\mathbf{x}),\ldots,A_{G_m}(\mathbf{x})),\ i = 1,\ldots,m,$$

et on en déduit le lemme suivant, interprétant le membre gauche de (5.7) en termes de ($\mathcal{H}$,$\mathcal{G}$)-arborescences.

Lemme 95. *Le nombre de ($\mathcal{H}$,$\mathcal{G}$)-arborescences sur $[\mathbf{n}]$ est*

$$\mathbf{n}![\mathbf{x}^{\mathbf{n}}]H(A_{G_1}(\mathbf{x}),\ldots,A_{G_m}(\mathbf{x})), \tag{5.8}$$

($\mathcal{H}$,$\mathcal{G}$)-endofonctions restreintes. On peut interpréter combinatoirement le membre droit de (5.7) à l'aide d'une généralisation de la notion d'endofonction, que nous avons déjà vue au chapitre 2 (preuve de la proposition 29).

Définition 96. Une ($\mathcal{H}$,$\mathcal{G}$)-*endofonction partielle* sur $[\mathbf{n}]$ est une fonction f de $[\mathbf{n}]$ dans $[0,\mathbf{n}]$ telle que la préimage de 0, $f^{-1}(0)$, est munie d'une $\mathcal{H}$-structure et, pour $i = 1,\ldots,m$, la préimage de tout élément u de couleur c_i ($u \in [n_i]$), $f^{-1}(u)$, est munie d'une $\mathcal{G}_i$-structure.

On représente habituellement une endofonction f par un graphe orienté dans lequel un arc de u vers v indique que $f(u) = v$. Tout comme pour les arborescences, on peut indiquer la structure des préimages par un arc de cercle indicé par la structure dont est munie une préimage. Par la suite, on omettra cet arc de cercle, la structure étant implicitement déterminée par la couleur du sommet image.

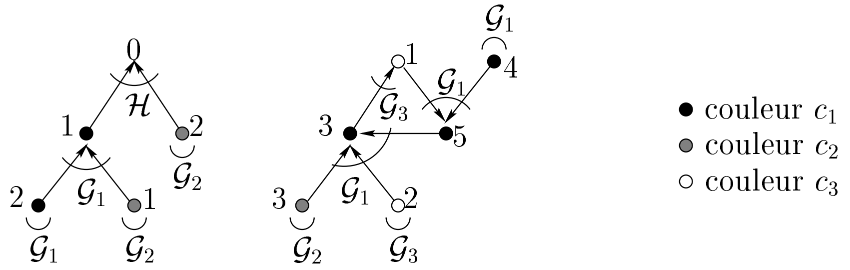


FIG. 5.3 – ($\mathcal{H}$,$\mathcal{G}$)-endofonction avec $m = 3$

Définition 97. Soit f une ($\mathcal{H}$,$\mathcal{G}$)-endofonction partielle sur $[\mathbf{n}]$. On appelle *graphe des couleurs de* f le graphe orienté à $(m+1)$ sommets étiquetés par $\{0\} \cup [m]$, ayant un arc de i vers 0 (resp. j) si et seulement si l'élément 1 de couleur c_i a pour image 0 (resp. un élément de couleur c_j).

Le graphe des couleurs d'une endofonction est donc déterminé par les sommets d'étiquette 1 de chaque couleur.

Définition 98. Une ($\mathcal{H}$,$\mathcal{G}$)-*endofonction restreinte* sur $[\mathbf{n}]$ est une ($\mathcal{H}$,$\mathcal{G}$)-endofonction partielle f dont le graphe des couleurs est une arborescence de racine 0 ayant tous les arcs orientés vers la racine 0.

L'endofonction de la figure 5.3 est une ($\mathcal{H}$,$\mathcal{G}$)-endofonction restreinte comme le montre la figure qui suit.

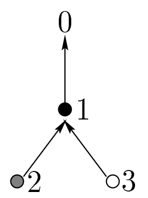


FIG. 5.4 – Graphe des couleurs de l'endofonction de la figure 5.3

Dans la suite, on considère les endofonctions comme des graphes orientés et, pour tout élément u, on appelle parfois fibre sa préimage $f^{-1}(u)$. On relie ces endofonctions au membre droit de (5.7) par le résultat suivant [80].

Lemme 99. *Le nombre de* $(\mathcal{H},\mathcal{G})$*-endofonctions restreintes sur* $[\mathbf{n}]$ *est*

$$(\mathbf{n}-\mathbf{1})![\mathbf{x}^{\mathbf{n}-\mathbf{1}}]\sum_T \frac{\partial(H(\mathbf{x}),(G_1(\mathbf{x}))^{n_1},\ldots,(G_m(\mathbf{x}))^{n_m})}{\partial T}, \tag{5.9}$$

la somme portant sur toutes les arborescences orientées T *ayant* $\{0,1,\ldots,m\}$ *pour ensemble de sommets,* 0 *pour racine et dont les arcs sont tous orientés vers la racine* 0.

Preuve. Si $\mathcal{C}$ est une classe de structures combinatoires m-coloriées telle que $c_{\mathbf{n}}$ est le nombre de $\mathcal{C}$-structures sur $[\mathbf{n}]$, $\mathcal{C}$ a pour série génératrice exponentielle

$$C(\mathbf{x}) = \sum_{n_1,\ldots,n_m\geq 0} c_{\mathbf{n}}\frac{\mathbf{x}^{\mathbf{n}}}{\mathbf{n}!}.$$

En notant $C_i(\mathbf{x})$ la série $\partial C(\mathbf{x})/\partial x_i$, on a

$$C_i(\mathbf{x}) = \sum_{n_1,\ldots,n_{i-1},n_{i+1},\ldots,n_m\geq 0, n_i\geq 1} c_{\mathbf{n}}\frac{x_1^{n_1}}{n_1!}\cdots\frac{x_i^{n_i-1}}{(n_i-1)!}\cdots\frac{x_m^{n_m}}{n_m!}.$$

$C_i(\mathbf{x})$ représente donc la série génératrice exponentielle des $\mathcal{C}$-structures ayant au moins un élément de couleur c_i, l'un de ces éléments de couleur c_i n'étant pas “compté” par la variable x_i. On peut, sans perte de généralité, supposer que l'élément non pris en compte est le plus petit élément de couleur c_i, et on en déduit que le nombre de $\mathcal{C}$-structures sur $[\mathbf{n}]$ telles que $n_i \geq 1$ est

$$n_1!\cdots(n_i-1)!\cdots n_m![x_1^{n_1}\cdots x_i^{n_i-1}\cdots x_m^{n_m}]\frac{\partial C_i(\mathbf{x})}{\partial x_i}.$$

Une endofonction peut aussi être représentée comme la liste ordonnée des préimages de ses éléments, ces éléments étant ordonnés par couleurs croissantes ($c_i > c_j$ si et seulement si $i > j$), puis par ordre croissant des éléments d'une même couleur, 0 étant inférieur à tout autre élément, comme l'illustre la figure suivante.

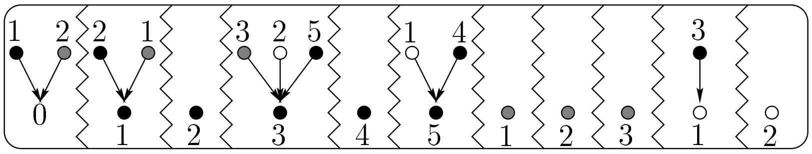


FIG. 5.5 – Représentation de l'endofonction de la figure 5.3

On en déduit donc qu'une $(\mathcal{H},\mathcal{G})$-endofonction partielle (non forcément restreinte) sur $[\mathbf{n}]$ est une $(\mathcal{H}\prod_{i=1}^m \mathcal{G}_i^{n_i})$-structure sur $[\mathbf{n}]$ (voir [16, section 3.2]) et, par application des résultats classiques sur la multiplication de séries génératrices (voir chapitre 1), que le nombre de $(\mathcal{H},\mathcal{G})$-endofonctions partielles sur $[\mathbf{n}]$ est

$$\mathbf{n}![\mathbf{x}^{\mathbf{n}}]H(\mathbf{x})\left(\prod_{i=1}^m (G_i(\mathbf{x}))^{n_i}\right).$$

Le graphe des couleurs d'une endofonction étant déterminé par les éléments 1 de chaque couleur, en appliquant les résultats précédents aux endofonctions restreintes (le graphe des couleurs est une arborescence orientée de racine 0 aux arcs orientés vers la racine), on en déduit le résultat voulu. □

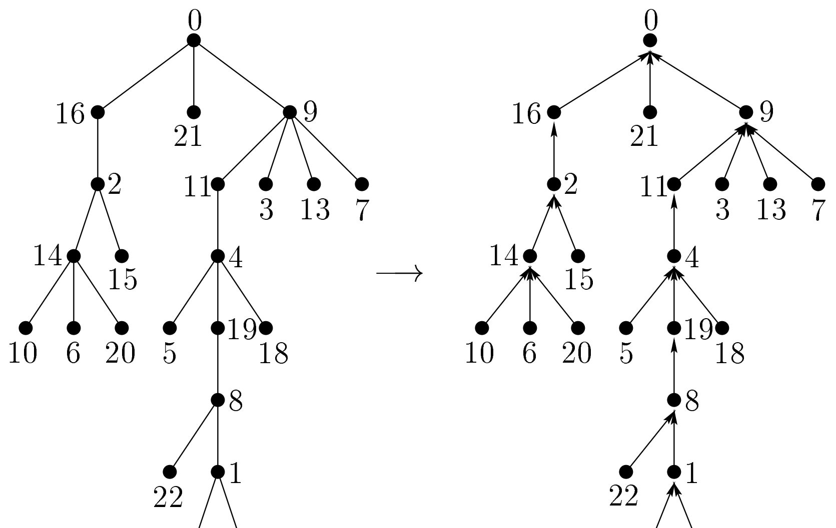
Il découle de (5.7), (5.8) et (5.9), que l'obtention d'une bijection entre les $(\mathcal{H},\mathcal{G})$-arborescences sur $[\mathbf{n}]$ et les $(\mathcal{H},\mathcal{G})$-endofonctions restreintes sur $[\mathbf{n}]$ induit une preuve bijective du théorème 92. Goulden et Kulkarni ont proposé une telle bijection. Dans la suite de cette section, nous présentons une nouvelle bijection plus simple que la leur.

5.2.2 Le cas unidimensionnel : la formule de Lagrange

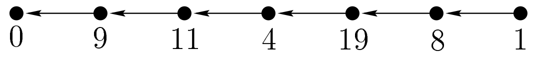
Nous commençons par traiter le cas unidimensionnel, c'est-à-dire la formule de Lagrange classique. Nous présentons dans un premier temps la preuve de Labelle [108], qui sert de base à la preuve de Goulden et Kulkarni dans le cas multidimensionnel. Nous en proposons ensuite une variante, que nous étendons à notre tour au cas multidimensionnel dans le paragraphe suivant.

Dans le cas unidimensionnel ($m = 1$), le vecteur $\mathcal{G}$ se réduit à $\mathcal{G}_1$. Les $(\mathcal{H},\mathcal{G})$-arborescences sur $[\mathbf{n}]$ se réduisent alors à des arborescences sur $[0,n_1]$ de racine 0 telles que la fibre de 0 est munie d'une $\mathcal{H}$-structure et la fibre de tout autre sommet est muni d'une $\mathcal{G}_1$-structure. Les $(\mathcal{H},\mathcal{G})$-endofonctions restreintes sur $[\mathbf{n}]$ se réduisent aux fonctions f de $[n_1]$ dans $[0,n_1]$ telles que $f(1) = 0$ (cette condition est due à la contrainte sur le graphe des couleurs de f qui doit être dans ce cas l'arborescence $1 \to 0$), la fibre de 0 est munie d'une $\mathcal{H}$-structure et la fibre de tout autre élément est munie d'une $\mathcal{G}_1$-structure.

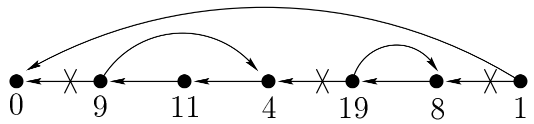
La bijection de G. Labelle. Soit A une $(\mathcal{H},\mathcal{G})$-arborescence. On définit tout d'abord une endofonction f'_A en orientant toutes les arêtes de A vers la racine 0 (la structure dont est munie la fibre d'un élément se transporte évidemment de A vers f'_A). f'_A est donc une endofonction partielle, mais n'est pas forcément restreinte (on n'a pas en général $f'_A(1) = 0$).



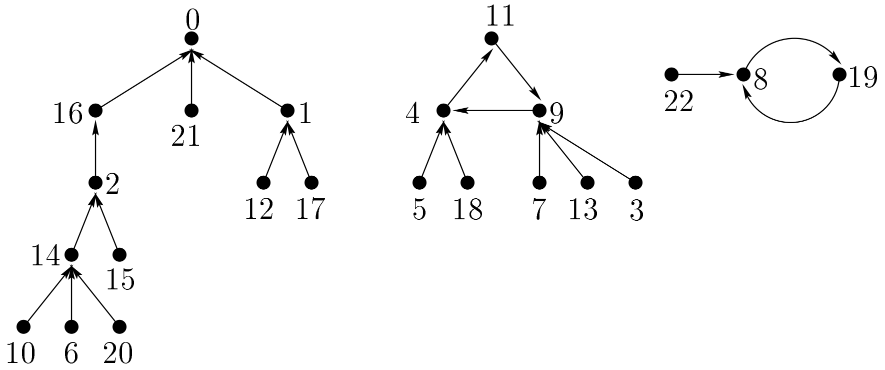
On appelle *colonne vertébrale* de f'_A le chemin dans l'arborescence reliant le sommet 0 au sommet 1. On note $(u_0,u_1,\ldots,u_k)$ ce chemin ($u_0 = 0$, $u_k = 1$ et u_i est fils de u_{i-1}, pour $1 \leq i \leq k$). La colonne vertébrale de l'exemple ci-dessus est le chemin suivant.



Pour $1 \leq i < k$, on dit que u_i est un *minimum à droite* de l'ensemble ordonné $(u_0,\ldots,u_{k-1})$ si il n'existe pas d'entier j tel que $0 \leq i < j < k$ et $u_j < u_i$. On note l le nombre de minima à droite et $(v_0 = 0,\ldots,v_{l-1})$ l'ensemble des minima à droite. Dans notre exemple, $l = 3$ et $(v_0,v_1,v_2) = (0,4,8)$. Soit w_i le successeur de v_i sur le chemin allant de 0 à 1 ($(w_0,w_1,w_2) = (9,19,1)$). On transforme alors f'_A en une endofonction restreinte f_A en permutant circulairement les w_i : pour i allant de 0 à $l-1$, on remplace, dans la fibre de v_i (c'est-à-dire dans la $\mathcal{H}$-structure ou $\mathcal{G}_1$-structure correspondante), w_i par w_{i-1} si $i > 0$ et w_0 par w_{l-1}. Cela revient à modifier la colonne vertébrale de la façon suivante (ces modification s'effectuant donc dans l'arborescence de départ) :



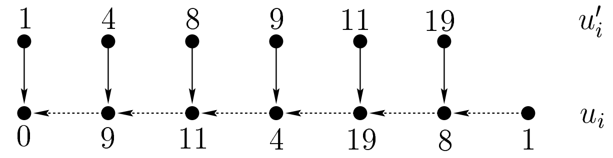
et on obtient l'endofonction suivante :



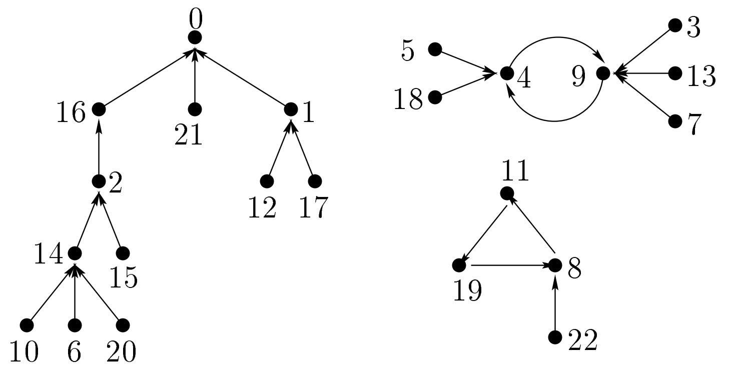
L'endofonction f_A ainsi obtenue est bien une endofonction restreinte car $f_A(1) = 0$. De plus, les cycles (il y en a exactement $l-1$) sont constitués uniquement des éléments $u_1,\ldots,u_{k-1}$, chacun contenant exactement un minimum à droite différent de 0.

La construction inverse, transformant f_A en A, se déduit immédiatement de ces constats. Soient $l-1$ le nombre de cycles de f_A, $v_1 < \cdots < v_{l-1}$ les éléments minimaux de chaque cycle et, pour $1 \leq i \leq l-1$, w_i l'élément de la fibre de v_i situé sur son cycle. On pose de plus $v_0 = 0$ et $w_0 = 1$. Pour obtenir une arborescence, il suffit alors d'effectuer la transformation suivante : pour i allant de 0 à $l-1$, remplacer, dans la fibre de v_i, w_i par w_j où $j = (i+1) \mod l$.

Une variante de cette bijection. La variante que nous proposons est basée elle aussi sur une transformation de la colonne vertébrale de l'endofonction obtenue en orientant les arêtes de A vers la racine. Soient $(u_0 = 0,u_1,\ldots,u_k = 1)$ cette colonne vertébrale et $u'_0 < \cdots < u'_k$ ces mêmes sommets en ordre croissant. Pour i allant de 0 à $k-1$, on remplace, dans la fibre de u_i, u_{i+1} par u'_{i+1}. Dans l'exemple, on peut représenter cette construction par le schéma suivant,

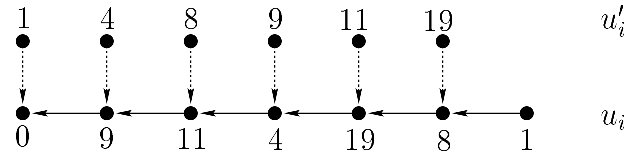


ce qui donne l'endofonction f_A suivante.



Là encore, du fait que $f_A(1) = 0$, f_A est bien une endofonction restreinte. On vérifie de plus que les cycles de f_A sont constitués des seuls éléments $u_1, \dots ,u_{k-1}$.

La construction inverse consiste elle aussi à modifier les fibres de 0, 1, et des éléments des cycles de f_A. Soient $u'_0 < \cdots < u'_k$ ces éléments ($u'_0 = 0$ et $u'_1 = 1$). Pour $i = 1, \dots ,k$, on note u_i le sommet $f_A(u'_i)$. On remplace alors, dans la fibre de u_i, le sommet u'_i par u_{i+1} (u'_k par 1 dans la fibre de u_k).



On peut illustrer cette construction par le schéma précédent, qui montre qu'on transforme les cycles de f_A en un chemin commençant en 0 et se terminant en 1 et qu'on obtient ainsi une arborescence de racine 0.

5.2.3 Le cas multidimensionnel

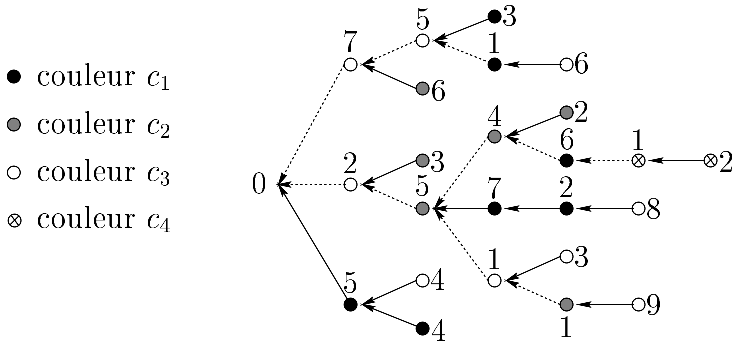
L'extension de la bijection précédente au cas multidimensionnel repose sur la notion de *squelette*, une généralisation de la notion de colonne vertébrale :

- le squelette d'une $(\mathcal{H},\mathcal{G})$-arborescence est constitué par l'ensemble des chemins allant de 0 à chaque sommet 1 des m couleurs ;
- le squelette d'une $(\mathcal{H},\mathcal{G})$-endofonction f est constitué par l'ensemble des cycles de f et l'ensemble des chemins allant des sommets 1 de chaque couleur vers un sommet d'un cycle ou vers le sommet 0.

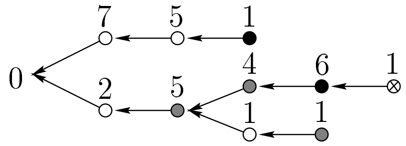
Notation. Dans la suite de cette section, on note k_i le sommet k de couleur c_i.

Transformation d'une $(\mathcal{H},\mathcal{G})$-arborescence en une $(\mathcal{H},\mathcal{G})$-endofonction restreinte. Soit A une $(\mathcal{H},\mathcal{G})$-arborescence. On commence par orienter toutes ses arêtes vers la racine 0, pour obtenir une endofonction partielle non forcément restreinte. Dans la figure suivante

($m = 4$, $n_1 = 7$, $n_2 = 6$, $n_3 = 9$ et $n_4 = 2$), on a représenté en pointillés les arêtes du squelette de A.

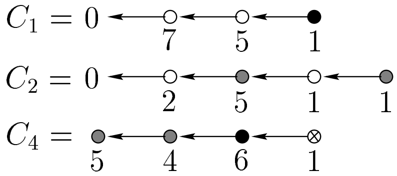


On dit alors que le sommet 1_i est *initial* si la restriction de sa fibre au squelette de A est vide.



Comme on le voit sur l'exemple, les sommets initiaux sont 1_1, 1_2 et 1_4.

La première étape de la transformation consiste à assigner un chemin C_i à chaque sommet initial 1_i : pour i allant de 1 à m, si 1_i est initial, on note C_i le chemin (dans le squelette) allant du sommet 1_i au premier de ses ancêtres appartenant déjà à un chemin C_j avec $j < i$ (ou à 0 si aucun sommet n'a déjà été obtenu). Pour l'exemple considéré, on obtient :



Remarque 100. Les C_i constituent ce que l'on pourrait appeler les branches (ou les filaments) du squelette prises relativement à l'ordre sur les couleurs.

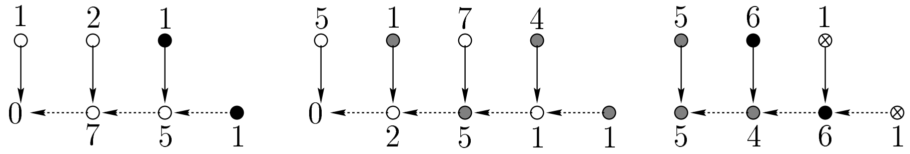
Pour chaque chemin C_i, on note f_i l'extrémité finale de C_i (l'extrémité initiale est le sommet 1_i). On vérifie alors la propriété suivante, conséquence de la définition des C_i.

(a) *Tout sommet $u \neq 0$ appartenant à plusieurs chemins C_i est l'extrémité finale de tous ces chemins excepté du premier. Le sommet* 0 *est pour sa part l'extrémité finale de tous les chemins auxquels il appartient.*

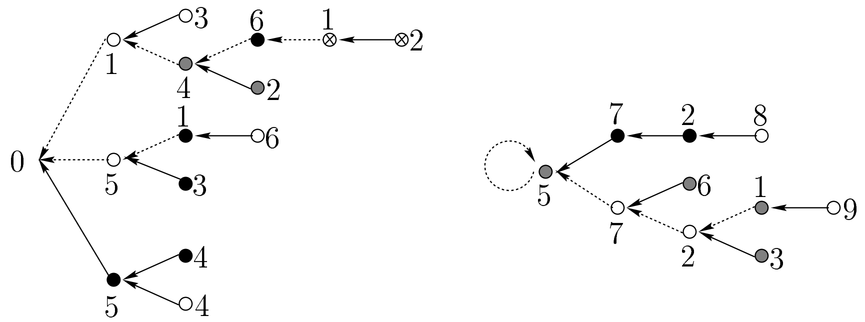
Soit P_i l'ensemble des sommets de couleur c_i appartenant au squelette de A :

$$P_1 = \{1{,}6\},\ P_2 = \{1{,}4{,}5\},\ P_3 = \{1{,}2{,}5{,}7\},\ P_4 = \{1\}.$$

On effectue alors la transformation suivante : pour i allant de 1 à m, si 1_i est initial, poser $u = f_i$ et tant que $u \neq 1_i$, si v est le prédécesseur de u dans C_i et c_j la couleur de v, remplacer, dans la fibre de u, l'élément v par le plus petit sommet de P_j (que l'on retire de P_j) puis poser $u = v$. Cela revient à dire que l'on parcourt les C_i à contre-sens et que les préimages sont déterminées en prenant les sommets de ces couleurs en ordre croissant.



On obtient de cette manière l'endofonction f_A suivante (les arcs en pointillé sont les arcs modifiés).



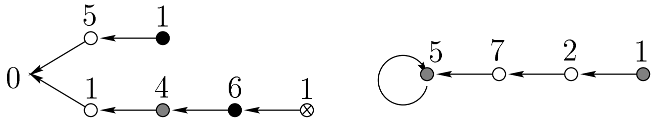
Il reste à vérifier que le graphe des couleurs de l'endofonction ainsi construite est une arborescence orientée de racine 0 dont tous les arcs sont orientés vers la racine. Cela découle du raisonnement suivant :

– si le prédécesseur u de 0 dans le premier chemin C_i suivi est de couleur c_j, alors on remplace u par 1_j dans la fibre de 0,
– lorsqu'on remplace, dans la fibre d'un sommet u de couleur c_i, le prédécesseur v de u par un sommet 1_j (v est le premier sommet de couleur c_j rencontré), cela implique que $1_i \notin P_i$ (on a retiré 1_i de P_i lorsqu'on a rencontré le premier sommet ayant un successeur de couleur c_i, qui est soit u soit un sommet rencontré avant u) et que dans le graphe des couleurs le sommet j a pour image le sommet i.

On en déduit que pour toute couleur c_i, il existe k tel que $f_A^k(1_i) = 0$, et par conséquent que le graphe des couleurs de f_A est une arborescence de racine 0 dont tous les arcs sont orientés vers la racine. De plus, on vérifie immédiatement les propriétés suivantes satisfaites par f_A.

(b) *Pour tout sommet u de f_A, u appartient au squelette de f_A si et seulement si u appartient au squelette de A.*
(c) *Le sommet 1_i est initial dans f_A si et seulement il l'est dans A.*
(d) *Pour tout élément non initial u du squelette de f_A, la restriction de la fibre de u au squelette de f_A comporte k éléments si et seulement si il existe un ensemble $\{i_1, \ldots, i_k\} \subseteq [m]$ maximal tel que u appartient à chaque chemin C_{i_j}, pour $j = 1, \ldots, k$.*

Pour l'exemple considéré, le squelette de l'endofonction f_A obtenue est le suivant.



Transformation d'une $(\mathcal{H},\mathcal{G})$-endofonction restreinte en une $(\mathcal{H},\mathcal{G})$-arborescence. Soit f une $(\mathcal{H},\mathcal{G})$-endofonction restreinte. En se référant à la propriété (b), pour obtenir une arborescence A_f telle que $f = f_{A_f}$, il suffit de reconstruire les chemins C_i du squelette de A_f à partir du squelette de f. Soit R le multi-ensemble comportant, pour chaque élément u, k occurrences du sommet u si et seulement si la restriction de la fibre de u au squelette de f comporte k éléments (R ne contient donc que des éléments du squelette de f). Pour l'exemple précédent, nous avons

$$R = \{0,0,6_1,4_2,5_2,5_2,1_3,2_3,5_3,7_3\}.$$

Pour $i = 1,\dots,m$, soit S_i l'ensemble des sommets de couleur c_i du squelette de f :

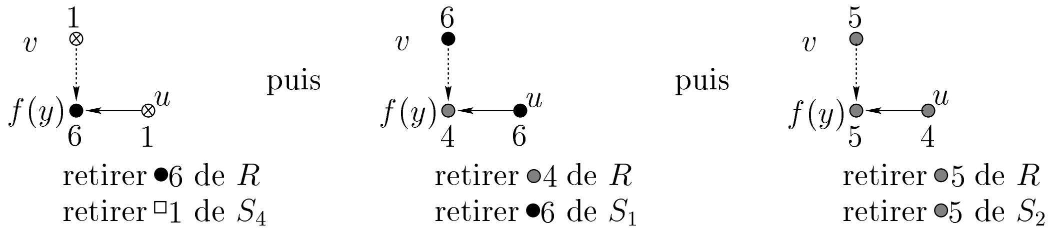
$$S_1 = \{1,6\},\ S_2 = \{1,4,5\},\ S_3 = \{1,2,5,7\},\ S_4 = \{1\}.$$

On peut maintenant construire les chemins C_i. Pour $i = m,\dots,1$, si le sommet 1_i est initial dans f, on construit C_i de la façon suivante.

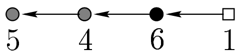
---

poser $u = 1_i$;
répéter
  soit c_j la couleur de u et v le plus grand élément de $S_j(f)$;
  remplacer, dans la fibre de $f(v)$, v par u;
  retirer $f(v)$ de R et v de S_j, puis poser $u = f(v)$;
tant que $u \neq 0$ et u n'apparaît pas dans R;

Dans notre exemple, pour $i = 4$, on effectue les opérations suivantes pour reconstruire C_4,



on a alors $R = \{0,0,5_2,1_3,2_3,5_3,7_3\}$ et on s'arrête car 5_2 apparaît encore dans R. On retrouve bien C_4 de cette façon.

$C_4 =$ 

Pour vérifier qu'on obtient bien une arborescence, il suffit de constater qu'on a transformé le squelette de f en une arborescence de racine 0, ce qui découle directement de la condition d'arrêt de la boucle construisant les C_i : on s'arrête lorsque l'on arrive sur 0 ou sur un élément qui sera présent dans un autre chemin car il apparaît encore dans le multi-ensemble R. Le fait que $f_{A_f} = f$ est une conséquence des propriétés (a), (b), (c) et (d).

Cette bijection vérifie de plus la propriété suivante, que nous appelons *conservation de la structure des fibres.*

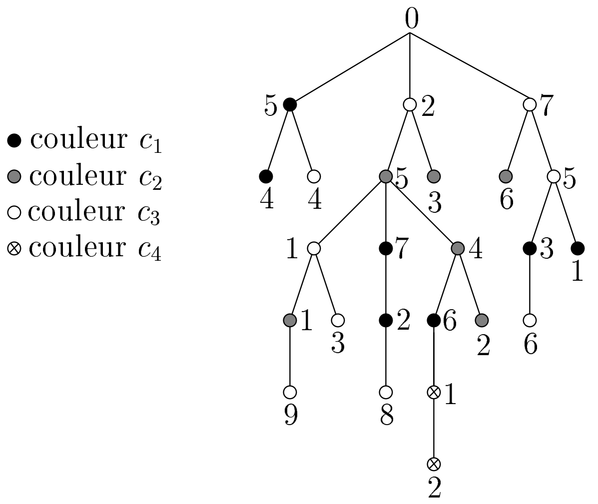
Propriété 101. Pour tout sommet u et toute couleur c_i, la fibre de u dans A comporte k éléments de couleur c_i si et seulement si la fibre de u dans f_A comporte aussi k éléments de couleur c_i.

Dans les deux dernières sections de ce chapitre, nous appliquons cette bijection entre arborescences et endofonctions restreintes à l'énumération d'arborescences et de cactus.

5.3 Arborescences dont la partition des arêtes est fixée

Dans deux articles parus au début des années 80 [95, 76], Goulden et Jackson se sont intéressés aux arborescences dont la partition des arêtes (voir définition 102 ci-dessous) est fixée, en utilisant une variante de la formule de Good dont ils ne donnent pas de preuve combinatoire. Dans cette section, nous reprenons certains de leurs résultats, notamment la variante de la formule de Good, et nous en donnons une preuve combinatoire basée sur la bijection entre arborescences et endofonctions présentée précédemment. Ces preuves illustrent le fait que, selon nous, *les endofonctions restreintes sont des structures souvent plus faciles à décrire et à compter que les arborescences*, car la structure arborescente à considérer est réduite au graphe des couleurs.

Définition 102. Soit A une arborescence à m couleurs sur $[0,\mathbf{n}]$, de racine 0. La *partition des arêtes de A* est une matrice carrée P de taille $(m+1) \times (m+1)$ $(P = (p_{i,j})_{0 \leq i,j \leq m})$ telle que $p_{i,j}$ (resp. $p_{i,0}$), est égal au nombre d'arêtes reliant, dans A, un sommet u de couleur c_i à un sommet v de couleur c_j (resp. $y = 0$), v étant le père de u. La partition des arêtes d'une arborescence sur $[0,\mathbf{n}]$ de racine 0 vérifie $p_{0,j} = 0$ pour $1 \leq j \leq m$ et $\sum_{j=0}^{m} p_{i,j} = n_i$ pour $1 \leq i \leq m$.



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 2 & 2 & 0 \\ 0 & 0 & 2 & 4 & 0 \\ 2 & 3 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

FIG. 5.6 – Une arborescence et sa partition des arêtes

Remarque 103. On étend naturellement cette notion de partition des arêtes aux endofonctions en les considérant comme des graphes orientés ($p_{i,j}$ est le nombre d'arcs allant d'un sommet de couleur c_i vers un sommet de couleur c_j).

Notation. On note $\mathbf{p_j}$ le vecteur correspondant à la $j^{ème}$ colonne $(p_{0,j}, p_{1,j}, \ldots, p_{m,j})$ de la matrice P et $\mathbf{P}! = \prod_{i,j} p_{i,j}!$ (par convention $0! = 1$).

Pour énumérer les $(\mathcal{H},\mathcal{G})$-arborescences dont la partition des arêtes P est fixée il suffit d'énumérer les $(\mathcal{H},\mathcal{G})$-endofonctions restreintes de partition des arêtes P, compte-tenu de la bijection entre arborescences et endofonctions et de la propriété de conservation des fibres (propriété 101). Pour prendre en compte la contrainte sur le graphe des couleurs des endofonctions, on utilise le théorème "Matrix-Tree" pour les graphes orientés (voir [77, section 3.3.24] ou [31]).

Théorème 104. *Soit $\lambda = (\lambda_{i,j})_{1\leq i,j\leq n}$ la matrice d'adjacence d'un graphe orienté G étiqueté à n sommets ($\lambda_{i,j}$ est le nombre d'arcs allant de i vers j). Le nombre d'arborescences couvrantes orientées de G de racine $c \in [n]$ dont tous les arcs sont orientés vers c est égal à*

$$\det\left(\left\{\delta_{i,j}\sum_{k=1}^{n}\lambda_{i,k}\right\} - \lambda_{i,j}\right)_{1\leq i,j\leq n, i\neq c, j\neq c}.$$

Pour une famille d'objets combinatoires étiquetés $\mathcal{F}$ et un entier k, on rappelle que $\mathcal{F}^k[n]$ désigne la famille des objets étiquetés sur $[n]$ constitués d'une liste ordonnée de k $\mathcal{F}$-structures. On note alors $\mathcal{H}[\mathbf{n}]$ (resp. $\mathcal{G}_i^{n_i}[\mathbf{n}]$) l'ensemble des $\mathcal{H}$-structures (resp. $\mathcal{G}_i^{n_i}$-structures) sur $[\mathbf{n}]$.

Théorème 105. *Soit P une matrice vérifiant les conditions de la définition 102 et $\delta(P)$ la matrice $(\delta_{i,j}n_i - p_{i,j})_{1\leq i,j\leq m}$. Le nombre de $(\mathcal{H},\mathcal{G})$-arborescences sur $[\mathbf{n}]$ admettant P comme partition des arêtes est*

$$\det(\delta(P))\frac{(\mathbf{n}-\mathbf{1})!}{\mathbf{P}!}\left(\prod_{i=1}^{m}|\mathcal{G}_i^{n_i}[\mathbf{p_i}]|\right)|\mathcal{H}[\mathbf{p_0}]|\,. \tag{5.10}$$

Preuve. Comme nous l'avons déjà mentionné, il suffit de montrer que le nombre de $(\mathcal{H},\mathcal{G})$-endofonctions restreintes sur $[\mathbf{n}]$ admettant P comme partition des arêtes est égal à (5.10).

On dit qu'une arborescence orientée T sur $[0,m]$ est *compatible avec* P si le fait qu'il existe dans T un arc reliant i à j, i étant le fils de j implique que $p_{i,j} \neq 0$ (une telle arborescence a donc forcément 0 pour racine, par définition de P).

Soit T une arborescence orientée compatible avec P. On cherche dans un premier temps à calculer le nombre de $(\mathcal{H},\mathcal{G})$-endofonctions restreintes f sur $[\mathbf{n}]$ admettant T comme graphe des couleurs. On remarque que si T possède un arc reliant i à j (avec i fils de j et donc $i > 0$), le nombre de façons de répartir les $n_i - 1$ éléments de couleur c_i (l'élément 1_i a pour image un élément de couleur c_j) en respectant la partition des couleurs de leurs images est

$$\binom{n_i - 1}{p_{i,0},\ldots,p_{i,j-1},p_{i,j}-1,p_{i,j+1},\ldots,p_{i,m}} = \frac{p_{i,j}}{n_i}\binom{n_i}{p_{i,0},\ldots,p_{i,j},\ldots,p_{i,m}}.$$

Pour l'élément 1_i la couleur de son image est fixée par l'arc reliant i et j dans T : c_j si $j > 0$, 0 si $j = 0$. On en déduit que le nombre d'endofonctions f admettant une arborescence T compatible avec P comme graphe des couleurs est

$$\left(\prod_{(i,j)\in T}p_{i,j}\right)\left(\prod_{i=1}^{m}\left\{\frac{1}{n_i}\binom{n_i}{p_{i,0},\ldots,p_{i,m}}|\mathcal{G}_i^{n_i}[\mathbf{p_i}]|\right\}\right)|\mathcal{H}[\mathbf{p_0}]|\,. \tag{5.11}$$

Soit $\mathcal{T}_P$ l'ensemble des arborescences sur $[0,m]$ de racine 0 compatibles avec P. Le théorème 104 donne

$$\det\left(\delta(P)\right) = \sum_{T\in\mathcal{T}_P}\left(\prod_{(i,j)\in T}p_{i,j}\right),$$

ce qui, combiné à (5.11), permet de déduire que le nombre de $(\mathcal{H},\mathcal{G})$-endofonctions restreintes admettant P comme partition des arêtes est

$$\det(\delta(P)) \left(\prod_{i=1}^{m} \left\{ \frac{1}{n_i} \binom{n_i}{p_{i,0},\ldots,p_{i,m}} |\mathcal{G}_i^{n_i}[\mathbf{p_i}]| \right\} \right) |\mathcal{H}[\mathbf{p_0}]|,$$

et conclut la preuve de la proposition. □

En rappelant que $|\mathcal{H}[\mathbf{p_0}]| = \mathbf{p_0}![\mathbf{x}^{\mathbf{p_0}}]H(\mathbf{x})$ et, que pour $i = 1,\ldots,m$, $|\mathcal{G}_i^{n_i}[\mathbf{p_i}]| = \mathbf{p_i}![\mathbf{x}^{\mathbf{p_i}}](G_i(\mathbf{x}))^{n_i}$, on a, compte-tenu du lemme 95 et de la proposition précédente, une preuve combinatoire immédiate de la variante suivante de la formule de Good, due à Goulden et Jackson [95] (voir aussi [77, section 1.2.13]).

Corollaire 106. *Soient m un entier, $\mathbf{x}$ le vecteur $(x_1,\ldots,x_m)$ de variables formelles, $H(\mathbf{x}),G_1(\mathbf{x}),\ldots,G_m(\mathbf{x})$ des séries formelles multivariées telles que $G_i(0,\ldots,0) \neq 0$ pour $i = 1,\ldots,m$. L'ensemble d'équations*

$$F_i(\mathbf{x}) = x_i G_i(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x})),\ i = 1,\ldots,m$$

détermine de manière unique alors la série formelle en $\mathbf{x}$ $H(F_1,\ldots,F_m)$ par :

$$[\mathbf{x}^{\mathbf{n}}]\left\{H(F_1(\mathbf{x}),\ldots,F_m(\mathbf{x}))\right\} =$$

$$\left(\prod_{i=1}^{m} \frac{1}{n_i}\right) \sum_{P} \left\{ \det(\delta(P)) \left(\prod_{i=1}^{m} [\mathbf{x}^{\mathbf{p_i}}]\left\{(G_i(\mathbf{x}))^{n_i}\right\} \right) [\mathbf{x}^{\mathbf{p_0}}]\left\{H(\mathbf{x})\right\} \right\},$$

la somme portant sur toutes les matrices P vérifiant les conditions de la définition 102.

Pour terminer, nous nous intéressons aux arborescences de Cayley et ordonnées sur $[\mathbf{n}]$, m-coloriées, dont la racine est de couleur c_k fixée et dont la partition des arêtes P est fixée. Nous aurons besoin du résultat (immédiat) suivant.

Lemme 107. *Soient S un ensemble de k éléments indistinguables et l un entier. Le nombre de partitions de S en l parts, éventuellement vides, ordonnées entre elles, est*

$$\binom{k+l-1}{k}.$$

Des arborescences sur $[\mathbf{n}]$, on se ramène aux arborescences sur $[0,\mathbf{n}]$ considérées précédemment dans ce chapitre en ajoutant un sommet 0 devenant racine et père de l'ancienne racine de couleur c_k. Ainsi, la partition des arêtes P vérifie $p_{k,0} = 1$ et pour tout $i \neq k$, $p_{i,0} = 1$. Dans les deux corollaires qui suivent, on suppose que la matrice P vérifie ces conditions. On note $q_j = \sum_{i=1}^{m} p_{i,j}$ le nombre de sommets présents dans les fibres des sommets de couleur c_j, et on rappelle que pour tout $1 \leq i \leq m$, le nombre de sommets de couleur c_i est $n_i = \sum_{j=0}^{m} p_{i,j}$. Pour une matrice $M = (m_{i,j})_{1\leq i,j\leq l}$, on note $\mathrm{cof}_{k,k}(M)$ le déterminant de la matrice M privée de sa $k^{ème}$ ligne et de sa $k^{ème}$ colonne.

Corollaire 108. *Le nombre d'arborescences ordonnées m-coloriées sur $[\mathbf{n}]$ dont la racine est de couleur c_k et admettant P pour partition des arêtes est*

$$\mathrm{cof}_{k,k}(\delta(P)) \frac{(\mathbf{n}+\mathbf{q}-\mathbf{1})!}{\mathbf{P}!},$$

Le nombre d'arborescences de Cayley m-coloriées sur $[\mathbf{n}]$ dont la racine est de couleur c_k et admettant P pour partition des arêtes est

$$\mathrm{cof}_{k,k}(\delta(P))\frac{(\mathbf{n}-\mathbf{1})!}{\mathbf{P}!}\mathbf{n}^{\mathbf{q}}.$$

Preuve. Elle résulte de la preuve du théorème 105 en remarquant les points suivants.

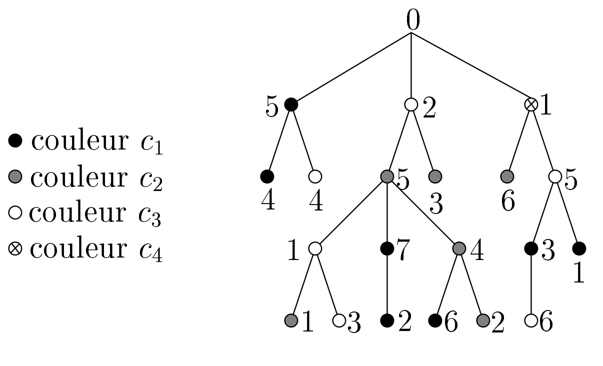
- Les arborescences A considérées (sur $[0,\mathbf{n}]$ après ajout du sommet 0) étant telles que la racine 0 a un unique fils de couleur c_k, il leur correspond des endofonctions restreintes f_A pour lesquelles $f_A(1_k)=0$ et $f_A(1_i)\neq 0$ pour $i\neq k$ (ce constat résulte immédiatement de la construction du paragraphe 5.2.3). Il s'ensuit que le graphe des couleurs de ces endofonctions est une arborescence de racine 0 n'ayant qu'un seul fils 1_k. En utilisant cette propriété dans la preuve du théorème 105 et en appliquant le théorème 104, on en déduit que le terme $\det(\delta(P))$ de (5.10) vaut $\mathrm{cof}_{k,k}(\delta(P))$.
- Le lemme 107 nous assure que pour les arborescences ordonnées, le nombre de façons d'ordonner les q_i sommets présents dans la fibre des n_i sommets de couleur c_i (ou, de manière équivalente, le nombre de $\mathcal{L}^{n_i}$-structures sur $[\mathbf{p_i}]$) est

$$q_i!\binom{n_i+q_i-1}{q_i}.$$

  Pour les arborescences de Cayley, ce nombre (ou le nombre de $\mathcal{E}^{n_i}$-structures sur $[\mathbf{p_i}]$) est $n_i^{q_i}$ (chacun des q_i sommets se trouve dans la fibre de l'un des n_i sommets de couleur c_i). $\square$

Nous nous intéressons maintenant aux arborescences dont la couleur de la racine et la partition des arêtes sont fixées, ainsi que la *partition des degrés*, que nous définissons ci-dessous.

Définition 109. Soit A une arborescence sur $[0,\mathbf{n}]$ de racine 0. La partition des degrés de A est une matrice $D=(d_{i,j})_{1\leq i,j}$ de taille $m\times\infty$, où $d_{i,j}$ est le nombre de sommets de couleur c_i ayant $j-1$ fils. Une telle matrice vérifie pour $1\leq i\leq m$, $\sum_{j\geq 1}d_{i,j}=n_i$ et $q_i=\sum_{j\geq 1}(j-1)d_{i,j}$, q_i désignant le nombre total de sommets présents dans les fibres des sommets de couleur c_i.



$$\begin{bmatrix} 4 & 1 & 0 & 0 & 0 & \cdots \\ 4 & 1 & 1 & 1 & 0 & \cdots \\ 3 & 3 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & 0 & \cdots \end{bmatrix}$$

FIG. 5.7 – Une arborescence et sa partition des degrés

Corollaire 110. *Le nombre d'arborescences ordonnées m-coloriées sur $[\mathbf{n}]$, dont la racine est de couleur c_k, admettant P pour partition des arêtes et D comme partition des degrés, est*

$$\mathrm{cof}_{k,k}(\delta(P))\frac{\mathbf{q}!\mathbf{n}!(\mathbf{n}-\mathbf{1})!}{\mathbf{P}!\mathbf{D}!}.$$

Le nombre d'arborescences de Cayley m-coloriées sur $[\mathbf{n}]$, dont la racine est de couleur c_k, admettant P pour partition des arêtes et D comme partition des degrés, est

$$\mathrm{cof}_{k,k}(\delta(P))\frac{\mathbf{q}!\mathbf{n}!(\mathbf{n}-\mathbf{1})!}{\mathbf{P}!\mathbf{D}!}\left(\prod_{i,j\geq 1}\frac{1}{((j-1)!)^{d_{i,j}}}\right).$$

Preuve. La preuve de ces deux formules est similaire à la preuve du Corollaire précédent. La seule différence réside dans le fait que le nombre de façons d'ordonner les q_i éléments présents dans les fibres des sommets de couleur c_i en respectant la partition des degrés (c'est-à-dire le nombre de $\mathcal{L}^{n_i}$-structures sur $[\mathbf{p_i}]$ telles que $d_{i,j}$ sommets ont degré $j-1$) est

$$q_i!\binom{n_i}{d_{i,1},\, d_{i,2},\, d_{i,3},\ldots}.$$

La formule pour les arborescences de Cayley est une conséquence directe de la formule pour les arborescences ordonnées (en effet, le nombre de façons d'ordonner les fils d'un sommet de degré d est $d!$). □

5.4 Énumération et génération aléatoire de cactus m-aires

Nous nous intéressons maintenant aux cactus planaires. Nous montrons comment la bijection entre arborescences et endofonctions permet d'expliquer combinatoirement deux formules d'énumération relatives à ces objets, qui n'avaient pas reçu à ce jour d'explication combinatoire[1], et nous en déduisons un algorithme de génération aléatoire pour ces structures.

5.4.1 Généralités sur les cactus.

Définition 111. Un *m-cactus* est un graphe simple connexe dans lequel chaque arête appartient à exactement un cycle élémentaire de taille m. De façon équivalente, chaque composante 2-connexe d'un cactus est un cycle élémentaire à m sommets, c'est-à-dire un m-gone (un polygone à m côtés).

Dans cette section, nous nous intéressons aux m-cactus plongés dans le plan (les polygones adjacents à un sommet sont circulairement ordonnés), nommés *m-cactus planaires*, et plus particulièrement aux *cactus m-aires*.

Définition 112. Un *cactus m-aire* est un m-cactus planaire tel que les sommets autour de chaque m-gone sont coloriés successivement par les couleurs c_1,c_2,… ,c_m dans le sens trigonométrique, et dans lequel un polygone est distingué (ou de manière équivalente une arête reliant un sommet de couleur c_1 et un sommet de couleur c_2). On parle parfois de

1. Nous avons récemment appris que D. Zvonkin a obtenu une autre preuve combinatoire de ces formules [168].

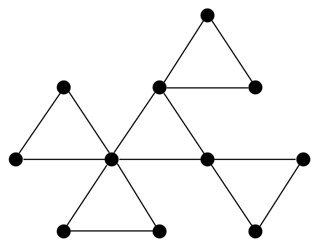


FIG. 5.8 – Un 3-cactus

cactus m-aire enraciné. Un cactus m-aire sur $\mathbf{n} = (n_1, \ldots ,n_m)$ est un cactus m-aire ayant n_i sommets de couleur i (pour $i = 1, \ldots ,m$).

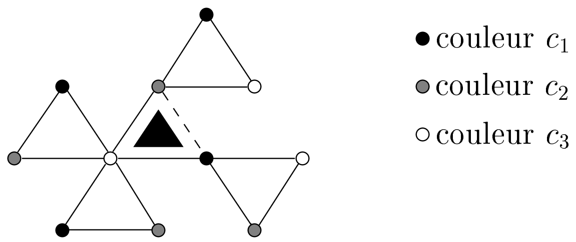


FIG. 5.9 – Un cactus ternaire

Les premières études concernant ces graphes sont dues à Harary et Uhlenbeck [89] et Harary et Palmer [88], suite à un article de Husimi [93]. Les cactus m-aires apparaissent dans plusieurs champs des mathématiques, comme par exemple la classification topologique des polynômes complexes [58, 87] ou encore les factorisations transitives minimales de permutations circulaires [78] (voir chapitre 6 de cette thèse).

5.4.2 Énumérations de cactus m-aires

Cactus, arborescences et endofonctions. Dans ce paragraphe, nous mettons en évidence la structure arborescente des cactus m-aires, ce qui, combiné à la bijection entre arborescences et endofonctions, nous permettra de mener à bien leur énumération.

Définition 113. Une arborescence *m-cactacée* sur $\mathbf{n}$ est une arborescence m-coloriée non étiquetée ayant n_i sommets de couleur c_i $(i = 1, \ldots ,m)$, de racine un sommet de couleur c_1, et telle que la fibre de tout sommet u de couleur c_i est structurée en une liste ordonnée de *i-blocs*, un i-bloc étant un ensemble (non ordonné donc) de $(m - 1)$ sommets dont les couleurs sont les $m - 1$ autres que c_i. La figure 5.10 présente une arborescence 3-cactacée.

Lemme 114. *Il existe une bijection entre les arborescences m-cactacées sur $\mathbf{n}$ et les cactus m-aires sur $\mathbf{n}$.*

Il s'agit d'un résultat classique concernant les cactus m-aires enracinés, basé sur un parcours en profondeur des polygones d'un cactus, en commençant par le polygone contenant l'arête racine (son sommet de couleur c_1 est la racine de l'arborescence créée), au cours duquel on remplace chaque polygone rencontré par un i-bloc si on est arrivé sur ce polygone par un sommet de couleur c_i. On associe ainsi un bloc à chaque polygone. La figure suivante, dans laquelle on a représenté les blocs par des rectangles et on a "fusionné" les arêtes joignant les sommets d'un bloc à leur père en une seule arête, illustre cette construction (on remarque que le polygone racine du cactus correspond au premier fils de la racine de l'arborescence).

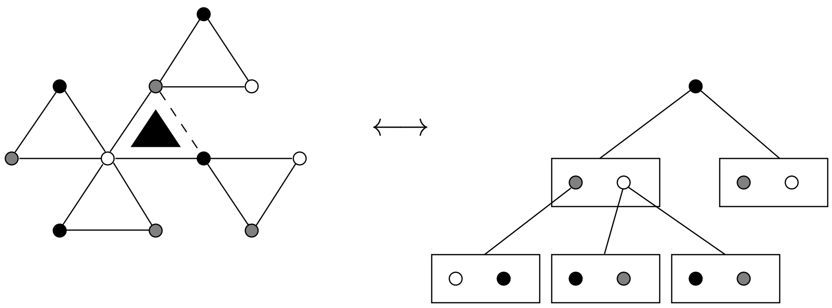


FIG. 5.10 – Cactus ternaire et arborescence 3-cactacée

On appelle degré d'un sommet dans une arborescence cactacée le nombre de blocs présents dans sa fibre, et degré d'un sommet dans un cactus le nombre de polygones auxquels ce sommet est incident. On vérifie alors immédiatement la propriété suivante.

Propriété 115. Soit C un cactus m-aire. Si u est un sommet de C autre que le sommet distingué (appartenant à l'arête racine) de couleur c_1, et u est incident à k polygones dans C ($k > 0$), alors u a degré $(k-1)$. Le sommet distingué de couleur c_1 de C et la racine de l'arborescence cactacée associée ont, pour leur part, même degré.

Pour énumérer les cactus m-aires, on se ramène donc à l'énumération des arborescences m-cactacées. Pour cela nous allons utiliser la bijection entre arborescences et endofonctions après avoir étiqueté les sommets de ces arborescences : les n_i sommets de couleur c_i d'une arborescence m-cactacée sur $\mathbf{n}$ sont étiquetés sur $[n_i]$ et on parle alors *d'arborescence m-cactacée étiquetée sur* $[\mathbf{n}]$ (ou d'arborescences m-cactacées sur $[\mathbf{n}]$).

Remarque 116. Si $a_{\mathbf{n}}$ (resp. $b_{\mathbf{n}}$) est le nombre d'arborescences m-cactacées sur $\mathbf{n}$ (resp. étiquetées sur $[\mathbf{n}]$), alors $b_{\mathbf{n}} = \mathbf{n}!a_{\mathbf{n}}$. Ainsi un algorithme engendrant aléatoirement et uniformément les arborescences m-cactacées étiquetées sur $[\mathbf{n}]$ induit un algorithme engendrant aléatoirement et uniformément les arborescences m-cactacées sur $\mathbf{n}$.

En appliquant la bijection entre arborescences et endofonctions de la section 5.2, on obtient une bijection entre arborescences cactacées étiquetées et endofonctions cactacées, dont la définition est la suivante.

Définition 117. Une endofonction *cactacée* sur $[\mathbf{n}]$ est une endofonction restreinte m-coloriée telle que la fibre de tout sommet u de couleur c_i est structurée en une liste ordonnée de *i-blocs*, la fibre de 0 comportant un seul sommet, le sommet 1 de couleur c_1.

Remarque 118. Le fait que la fibre de 0 est constituée du seul sommet 1 de couleur c_1 est une conséquence directe de la construction permettant de passer d'une arborescence à une endofonction restreinte (paragraphe 5.2.3).

Nous nous intéressons dans un premier temps au nombre de cactus m-aires ayant n_i sommets de couleur c_i, pour un vecteur $\mathbf{n}$ donné. On peut commencer par remarquer que ce vecteur doit vérifier certaines conditions.

Lemme 119. *Soient* $\mathbf{n}$ *donné et* $n = \sum_{i=1}^{m} n_i$. *Il existe au moins un cactus* m*-aire sur* $\mathbf{n}$ *si et seulement si* $(n-1) = p(m-1)$ *(*p *est alors égal au nombre de polygones) et* $1 \le n_i \le p$, *pour* $i = 1, \dots, m$.

Preuve. La condition est clairement nécessaire car chaque polygone comporte m sommets. Pour prouver qu'elle est suffisante, on raisonne par induction sur p. Si $p = 0$, $n = 1$ et on a le cactus à un seul sommet. Sinon, si $p = 1$, alors $n_i = 1$ pour $i = 1, \dots, m$ et on a le cactus à un seul polygone. Sinon ($p > 1$), il existe i tel que $n_i < p$. Supposons que $n_m < p$ et définissons le vecteur $\mathbf{n}'$ par $n'_m = n_m$ et $n'_i = n_i - 1$ pour $i \neq m$. Il suffit alors de construire un m-cactus de distribution des couleurs $\mathbf{n}'$ et d'y attacher en un sommet de couleur c_m un polygone pour obtenir un m-cactus de distribution des couleur $\mathbf{n}$. □

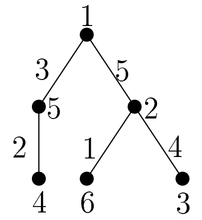
Le résultat suivant, donnant le nombre de cactus m-aires selon le nombre de sommets de chaque couleur, est dû à Goulden et Jackson [78] (voir aussi [21, 22]).

Théorème 120. *Soit* $\mathbf{n} = (n_1, \dots, n_m)$ *un vecteur d'entiers positifs vérifiant les conditions du lemme 119. Le nombre de cactus* m*-aires sur* $\mathbf{n}$ *est*

$$\frac{1}{p} \prod_{i=1}^{m} \binom{p}{n_i}, \tag{5.12}$$

où $n = \sum_{i=1}^{m} n_i$, *et* $p = (n-1)/(m-1)$ *est le nombre de polygones.*

Preuve. Il suffit de dénombrer les endofonctions cactacées sur $[\mathbf{n}]$. Afin de prendre en compte les contraintes du graphe des couleurs de ces endofonctions restreintes, on commence par remarquer que le codage de Prüfer, présenté au chapitre 1, induit un étiquetage des arêtes d'une arborescence de Cayley : si lors de la $i^{ème}$ étape du codage (algorithme 2) on crée une arête entre les sommets u et v, on étiquette cette arête par i.

2.5.1.2.1 $\longleftrightarrow$ 

Nous réalisons l'énumération des endofonctions cactacées en trois étapes : partition des blocs, placement des éléments minimaux (éléments 1_i) et enfin placement des éléments non minimaux. On rappelle qu'une endofonction cactacée ayant n_i sommets de couleur i comporte exactement p blocs, et que la préimage de l'élément 0 est constituée du seul élément 1_1. Un i-bloc étant constitué de $m-1$ sommets de couleurs deux à deux distinctes et autres que c_i, on en déduit la propriété suivante.

(a) *Le nombre de* 1*-blocs est* $p - n_1 + 1$, *et le nombre de* i*-blocs, pour* $i > 1$, *est* $p - n_i$.

Étape 1. Partition des blocs. Elle consiste à répartir les $p - n_1 + 1$ 1-blocs en n_1 parts (les n_1 fibres des sommets de couleur c_1) et les $p - n_i$ i-blocs ($i > 1$) en n_i parts (les fibres des sommets de couleur c_i). On déduit du lemme 107 que le nombre de façons de répartir les blocs dans les endofonctions cactacées sur $[\mathbf{n}]$ est

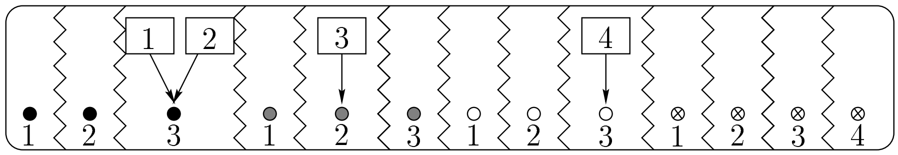
$$\binom{p}{n_1-1}\prod_{i=2}^{m}\binom{p-1}{n_i-1} = \frac{1}{(p-n_1+1)p^{m-1}}\prod_{i=1}^{m} n_i\binom{p}{n_i}. \tag{5.13}$$

Maintenant, étant donnée une partition des blocs, (on connaît, pour chaque sommet, le nombre de blocs que contient sa fibre) il nous faut déterminer le nombre de façons de répartir les sommets dans les blocs de manière à ce que la contrainte sur le graphe des couleurs d'une endofonction cactacée soit satisfaite.

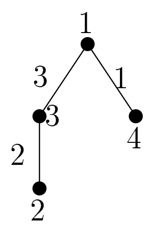
Étape 2. Placement des sommets minimaux. On considère dans un premier temps les sommets minimaux 1_i, $i = 2,\dots,m$ (1_1, le seul sommet présent dans la fibre de 0, n'appartient à aucun bloc), qui sont les sommets déterminant le graphe des couleurs de l'endofonction. Le nombre façons de placer les sommets minimaux afin que le graphe des couleurs soit une arborescence orientée de racine 0 dont tous les arcs sont orientés vers 0 est

$$p^{m-2}(p-n_1+1). \tag{5.14}$$

En effet, supposons les p blocs numérotés de manière arbitraire de 1 à p et considérons un mot w de longueur $(m-1)$ sur l'alphabet $[p]$: à toute lettre de w correspond un des p blocs de l'endofonction. À partir de w, on construit le mot w' en remplaçant chaque numéro de bloc par son type : si le bloc a dans w est un i-bloc, on remplace a par i dans w'. Ce mot w' est alors un mot de longueur $(m-1)$ sur l'alphabet $[m]$. La justification de cette construction transformant w en w' est due au fait que le graphe des couleurs est, comme son nom l'indique, seulement fonction des couleurs : ainsi, le fait que le sommet 1_j soit dans un quelconque des i-blocs se traduit par un arc unique $j \to i$ dans le graphe des couleurs. En appliquant le codage de Prüfer au mot w' tel que décrit précédemment, on obtient une arborescence sur $[m]$ dont les arêtes sont étiquetées sur $[m-1]$ (on les oriente vers la racine et on parle donc d'arcs). On peut alors associer à cette arborescence le placement suivant des éléments minimaux : si on a un arc de i vers j étiqueté k, on place le sommet 1_i dans le bloc correspondant à la $k^{ème}$ lettre de w. Ce bloc, donné par la $k^{ème}$ lettre de w ne peut être un i-bloc : en effet, l'arc d'étiquette k allant de i à j ($i \neq j$), la $k^{ème}$ lettre de w' est donc j et ce bloc est un j-bloc. Par exemple, pour $m = 4$ et $\mathbf{n} = (3,3,3,4)$, soit la partition suivante des blocs (les 4 blocs sont numérotés de 1 à 4 de gauche à droite).



Pour $w = 2.4.1$, on obtient $w' = 1.3.1$ et le codage de Prüfer appliqué à w' donne l'arborescence suivante.



On place alors le sommet 1_2 dans le bloc 4, le sommet 1_3 dans le bloc 1 et le sommet 1_4 dans le bloc 2.

Pour que la contrainte sur le graphe des couleurs soit vérifiée par ce placement des éléments minimaux, il faut et il suffit que sa racine soit le sommet 1, ce qui équivaut à imposer que la dernière lettre de w' est la lettre 1 (on obtient ainsi une arborescence de racine 1 que l'on relie à 0 par un arc joignant 1 à 0) et donc que le bloc associé à la dernière lettre de w est un 1-bloc. On a donc p choix pour chacune des $(m-2)$ premières lettres de w et $(p-n_1+1)$ pour sa dernière lettre, ce qui prouve (5.14).

Étape 3. Placement des éléments non minimaux. Une fois que les éléments minimaux sont placés, la partition des $n_i - 1$ autres sommets dans les blocs ne pose aucun problème. On déduit le résultat voulu de (5.13), (5.14) et du fait qu'il y a $(\mathbf{n}-\mathbf{1})! = \prod_{i=1}^{m}(n_i-1)!$ façons de répartir les éléments non minimaux (en tenant compte de la remarque 116). $\square$

La *distribution des degrés* d'un cactus est une matrice $D = (d_{i,j})_{1,\leq i,j}$, de taille $m \times \infty$, où $d_{i,j}$ est le nombre de sommets de couleur i et degré j. On considère chaque ligne $(d_{i,1}, d_{i,2}, \ldots)$ comme un vecteur $\mathbf{d_i}$. Par définition de D, on a $n = \sum_{i,j} d_{i,j}$ et $n_i = \sum_j d_{i,j}$, pour $i = 1, \ldots, m$. Par exemple, la distribution des degrés du cactus de la figure 5.9 est donnée par la matrice

$$D = \begin{bmatrix} 3 & 1 & 0 & 0 & \cdots \\ 3 & 1 & 0 & 0 & \cdots \\ 2 & 0 & 1 & 0 & \cdots \end{bmatrix}.$$

Théorème 121. [78, 21] *Soit $D = (d_{i,j})$ une matrice d'entiers positifs ou nuls de taille $m \times \infty$ avec $n_i = \sum_{j\geq 1} d_{i,j}$ et $\mathbf{n} = (n_1, \ldots, n_m)$ vérifiant les conditions du lemme 119. Le nombre de cactus m-aires sur $\mathbf{n}$ ayant D pour distribution des degrés est*

$$p^{m-1} \prod_{i=1}^{m} \frac{1}{n_i} \binom{n_i}{\mathbf{d_i}}. \tag{5.15}$$

où $n = \sum_{i=1}^{m} n_i$ et $p = (n-1)/(m-1)$.

Preuve. La preuve de ce résultat repose sur le même principe que la précédente, compte-tenu de la bijection entre arborescences cactacées et endofonctions cactacées. Si le degré d'un élément d'une endofonction cactacée est le nombre de blocs présents dans sa fibre, grâce à la propriété 101 de conservation de la structure des fibres et à la propriété 115, l'ensemble des cactus m-aires ayant pour partition des degrés D et une racine de degré l (le sommet de couleur c_1 du polygone racine est incident à l polygones) est en bijection avec l'ensemble des endofonctions cactacées ayant

- $d_{i,j}$ éléments de couleur c_i et de degré $j-1$ si $i > 1$ ou ($i = 1$, $j \neq l$ et $j \neq (l+1)$),
- $d_{1,l} - 1$ éléments de couleur c_1 et de degré $l-1$ et
- $d_{1,l+1} + 1$ éléments de couleur c_1 et de degré l.

Étape 1.a. Partition des i-blocs pour $i > 1$. Pour chaque couleur c_i, avec $i > 1$, le nombre de façons de répartir les i-blocs dans les fibres des n_i sommets de couleur i est

$$\binom{n_i}{d_{i,1}, d_{i,2}, \ldots}.$$

On place en effet $j-1$ i-blocs dans chaque fibre de $d_{i,j}$ éléments de couleur c_i, choisis parmi n_i, et ce pour tout $j > 1$.

Étape 1.b. Partition et pointage des 1-blocs. En ce qui concerne la couleur c_1, on commence de la même manière que pour les autres couleurs : on place $j-1$ 1-blocs dans chacune des fibres de $d_{1,j}$ éléments de couleur c_1 choisis parmi n_1, pour chaque valeur de $j > 1$. On obtient alors la même expression que précédemment avec $i = 1$. Mais ici, il reste encore un 1-bloc à placer dans la fibre d'un élément de couleur c_1, bloc que l'on distingue. Comme il y a exactement $d+1$ façons de placer un bloc distingué dans la fibre d'un élément de degré d, on a donc

$$n_1 + \sum_{j\geq 1}(j-1)d_{1,j} - 1 = n_1 + (p - n_1 + 1) - 1 = p$$

façons de placer ce 1-bloc distingué dans la fibre d'un élément de couleur c_1.

On en déduit que le nombre de façons de placer l'ensemble des p-blocs et de distinguer un 1-bloc est donné par

$$p\prod_{i=1}^{m}\binom{n_i}{d_{i,1},d_{i,2},\ldots}. \tag{5.16}$$

Étape 2. Placement des éléments minimaux. De la même manière que pour la preuve précédente, le nombre de façons de placer les éléments minimaux 1_i $(i > 1)$, en respectant la contrainte sur le graphe des couleurs, est donné, après numérotation arbitraire des blocs sur $[p]$, par le nombre de mots de longueur $(m-1)$ sur l'alphabet $[p]$ pour lesquels la dernière lettre est le numéro du 1-bloc distingué. Il y'en a donc

$$p^{m-2}. \tag{5.17}$$

Étape 3. Placement des éléments non minimaux. Finalement, comme précédemment, il y a $(\mathbf{n-1})!$ façons de placer les éléments non minimaux.

Ce dernier point, combiné à (5.16), (5.17) et à la remarque 116 donne la formule (5.15). □

5.4.3 Génération aléatoire de cactus m-aires

Les preuves précédentes, outre le fait qu'elles donnent la première explication combinatoire, à notre connaissance, des formules d'énumération de cactus m-aires, induisent des algorithmes efficaces de génération aléatoire uniforme de cactus m-aires, selon la distribution des couleurs ou des degrés.

Pour énumérer les endofonctions cactacées selon la distribution des couleurs ou des degrés, nous avons suivi un cheminement en trois étapes qui sous-tendent les grandes lignes d'un algorithme de génération aléatoire de cactus m-aires. Pour engendrer aléatoirement et uniformément une telle structure selon la distribution des couleurs (resp. des degrés), on peut appliquer l'algorithme suivant.

Algorithme 6: Génération aléatoire de cactus m-aires

Donnée : distribution $\mathbf{n}$ (resp. D) des couleurs (resp. des degrés)

Résultat : cactus m-aire

début

génération d'une endofonction cactacée ayant distribution des couleurs $\mathbf{n}$ (resp. des degrés D) par les 3 étapes suivantes;

1. partition des blocs;
2. placement des éléments minimaux;
3. placement des éléments non minimaux;

application de la bijection entre endofonctions cactacées et arborescences cactacées étiquetées;

application de la bijection entre arborescences cactacées et cactus m-aires;

fin

Lemme 122. *L'Algorithme 6 permet d'engendrer aléatoirement et uniformément un cactus m-aire à n sommets selon la distribution des couleurs ou des degrés, en temps et espace $O(n)$.*

Preuve. Par les preuves des théorèmes 120 et 121, il est clair que cet algorithme engendre un cactus m-aire. Il reste à vérifier que cette génération est uniforme et que les complexités en temps et en espace sont linéaires.

1. Génération uniforme. Le fait que cet algorithme engendre uniformément les cactus m-aires, c'est-à-dire que tous les cactus ayant une distribution des couleurs $\mathbf{n}$ (resp. des degrés D) ont la même probabilité d'être engendrés, provient du fait que lors de l'étape de génération aléatoire d'une endofonction cactacée :

- toutes les partitions des blocs ont la même probabilité d'être engendrées,
- le nombre de façons de placer les éléments minimaux ne dépend pas de la partition des blocs (et tous ces placements sont donc équiprobables),
- le nombre de façons de placer les éléments non minimaux ne dépend ni de la partition des blocs, ni du placement des éléments minimaux (et tous ces placements sont donc équiprobables).

2. Complexité linéaire en temps et en espace. Dans les deux cas, la phase de partition des blocs consiste essentiellement à partitionner un ensemble de taille fixée de i-blocs en n_i parts, ce qui se fait en temps et espace linéaires. Le placement des éléments minimaux consiste à choisir un mot de longueur fixée sur un alphabet fixé, puis à appliquer le codage de Prüfer, ce qui là aussi se fait en temps et espace linéaires. Le placement des éléments non minimaux consiste à choisir aléatoirement une permutation sur les éléments non minimaux, ce qui se fait également en temps et espace linéaires.

La transformation d'une arborescence cactacée en un cactus s'effectuant en temps et espace linéaires par un parcours de l'arborescence, la complexité de l'algorithme est donc déterminée par la complexité de la transformation d'une endofonction cactacée en arborescence cactacée.

Supposons qu'on code les sommets s des endofonctions cactacées et des arborescences cactacées avec la même structure de donnée, comportant les informations suivantes : la couleur c et l'étiquette e du sommet s, l'adresse p de son père, un tableau F stockant sa fibre, et la position de s dans le tableau F_p de son père. La transformation se décompose en trois étapes.

- On marque tout d'abord les sommets du squelette et on calcule le degré de la restriction de leur fibre dans ce squelette, ce qui se fait en au plus deux parcours effectués en remontant le long des chemins de l'endofonction.
- Une fois les sommets du squelette marqués, en utilisant un tableau de taille n_i pour chaque couleur, on parcoure une nouvelle fois tous les sommets de l'endofonction pour construire les ensembles S_i.
- On transforme finalement le squelette en modifiant les fibres de ses éléments, ce qui ne pose pas de difficulté car chaque sommet connaît sa position dans le tableau F de son père.

Cette transformation s'effectue donc en temps et espace linéaires. □

Remarque 123. La preuve de la forme arborescente de la formule de Good due à Goulden et Kulkarni [80] a les mêmes propriétés algorithmiques que la preuve présentée ici et peut être utilisée de la même manière pour obtenir un algorithme linéaire de génération aléatoire et uniforme de cactus m-aires.

5.5 Conclusion

Nous terminons ce chapitre en présentant quelques problèmes liés à la formule de Lagrange multidimensionnelle et qui nous semblent intéressants à examiner.

Énumération d'arborescences chromatiques ordonnées. Une arborescence m-coloriée est dite *m-chromatique* si toute arête relie deux sommets de couleurs différentes.

L'énumération d'arborescences de Cayley (la fibre de chaque sommet est munie d'une $\mathcal{E}$-structure) m-chromatiques selon la distribution des couleurs est simple à mener. Le nombre d'arborescences de Cayley m-chromatiques sur $[\mathbf{n}]$ est

$$\left(\sum_{i=1}^{m} n_i\right)^{m-1} \left(\prod_{i=1}^{m} \left(\sum_{j=1,\ j\neq i}^{m} n_j\right)^{n_i-1}\right).$$

On explique aisément cette formule en termes d'endofonctions correspondant aux arborescences de Cayley m-chromatiques : le premier terme de cette expression exprime le nombre de façons de choisir les images des éléments 1_i de sorte que l'endofonction soit restreinte (par une technique utilisant le codage de Prüfer analogue à la technique employée dans l'énumération de cactus planaires), le deuxième terme exprime le nombre de façons de choisir les images des éléments non minimaux.

Dans le cas des arborescences ordonnées m-chromatiques, le problème est plus difficile. Soit $\mathcal{OC}$ la famille des arborescences ordonnées m-chromatiques. Dans le cas $m = 2$, on déduit de la propriété 101 de conservation de la structure des fibres que les endofonctions correspondant à ces arborescences sont telles que la fibre d'un sommet de couleur c_1 ne comporte que des éléments de couleur c_2 et inversement. On en déduit presque immédiatement que le nombre d'arborescences 2-chromatiques ordonnées ayant n_1 (resp. n_2) sommets de couleur c_1 (resp. c_2) est

$$\frac{1}{n-1}\left(\frac{1}{n_1}\frac{(n-1)!}{(n_2-1)!}\frac{(n-1)!}{(n_1-2)!} + \frac{1}{n_2}\frac{(n-1)!}{(n_1-1)!}\frac{(n-1)!}{(n_2-2)!}\right),$$

où $n = n_1 + n_2$.

Cependant, dès que $m > 2$, il devient plus difficile de décrire les endofonctions restreintes correspondant aux arborescences de $\mathcal{OC}$, et il n'existe pas, à notre connaissance, une formule d'énumération pour ces arborescences. On peut toutefois utiliser le Corollaire 108 et sommer sur toutes les partitions des arêtes. Le résultat suivant, qui est le meilleur que nous ayons pu établir pour ce problème, est basé sur ce principe, avec l'avantage d'avoir éliminé le calcul du cofacteur.

Proposition 124. *Le nombre d'arborescences ordonnées m-chromatiques sur $[\mathbf{n}]$ est*

$$\left(\prod_{i=2}^{m}(2n-i)\right)\sum_{P}\frac{(\mathbf{n}+\mathbf{q}-\mathbf{1})!}{\mathbf{P}!},$$

où la somme est sur toutes les matrices $P = (p_{i,j})_{m\times m}$ telles que $p_{i,i} = 0$, $\sum_{j=1}^{m} p_{i,j} = n_i - 1$, $n = \sum_{i=1}^{m} n_i$ et $q_j = \sum_{i=1}^{m} p_{i,j}$.

Preuve. On énumère les endofonctions restreintes correspondant aux arborescences ordonnées m-chromatiques en considérant la matrice P donnant la partition des arcs excluant ceux ayant les éléments minimaux pour origine ($p_{i,j}$ est le nombre d'arcs allant d'un sommet de couleur c_i différent de 1_i à un sommet de couleur c_j). Ainsi, q_i est le nombre d'éléments non minimaux présents dans la fibre des éléments de couleur c_i. On a donc

$$\binom{n_i - 1}{p_{i,1}, \ldots ,p_{i,m}}$$

façons de choisir la couleur des images des éléments non minimaux de couleur c_i. On déduit du lemme 107 qu'une fois ces choix faits pour $i = 1,\ldots ,m$ (chaque élément non minimal connaît la couleur de son image), on a

$$q_i!\binom{n_i + q_i - 1}{n_i - 1}$$

façons de choisir les préimages (hors éléments minimaux) des éléments de couleur c_i et de les ordonner linéairement. Il reste alors à déterminer les images des éléments minimaux de sorte que l'on ait une endofonction restreinte. On utilise le codage de Prüfer exactement de la même façon que pour l'énumération de cactus m-aires, et obtient ainsi le résultat voulu. □

Dans le cas $m = 3$, cette proposition fournit la formule suivante pour le nombre d'arborescences ordonnées 3-chromatiques sur $[n_1,n_2,n_3]$ ($n = n_1 + n_2 + n_3$), certes lourde, mais programmable pour calculer les premières valeurs plus rapidement que par extraction de coefficients de séries définies par des équations lagrangiennes :

$$(2n-2)(2n-3)\sum_{k_1=0}^{n_1-1}\sum_{k_2=0}^{n_2-1}\sum_{k_3=0}^{n_3-1}\frac{(n_1+n_2-k_2+k_3-2)!}{k_1!(n_1-k_1-1)!}\times \frac{(n_2+n_3-k_3+k_1-2)!}{k_2!(n_2-k_2-1)!}\frac{(n_1+n_3-k_1+k_2-2)!}{k_3!(n_3-k_3-1)!}. \tag{5.18}$$

Cependant, lorsque m croît, ce mode de calcul devient rapidement trop lourd, en raison notamment du grand nombre de matrices P à considérer.

Le problème de déterminer (de calculer) le nombre d'arborescences ordonnées m-chromatiques ($m > 2$) selon la distribution des couleurs se pose et semble être difficile.

Formule de Lagrange multidimensionnelle et objets non étiquetés. La preuve de la formule de Lagrange multivariée que nous avons présentée, comme celle de Goulden et Kulkarni [80], est basée sur une bijection entre deux ensembles d'objets coloriés étiquetés. Comme on l'a vu, cela ne cause pas de difficulté pour la génération aléatoire de cactus m-aires car les arborescences m-cactacées sont des structures plongées dans le plan, et donc asymétriques.

On peut cependant vouloir engendrer les cactus (ou d'autres structures "lagrangiennes" coloriées) non pas aléatoirement, mais *exhaustivement*, c'est-à-dire engendrer tous les cactus ayant une distribution des couleurs (ou des degrés) donnée. Ce problème, apparaît notamment dans le cadre de la classification topologique des polynômes complexes [58, 87] et revient à engendrer exhaustivement toutes les arborescences m-cactacées de partition des couleurs (ou des degrés) donnée (lemme 114). Dans ce contexte, le fait que pour une distribution des couleurs donnée $\mathbf{n}$, il y ait $\mathbf{n}!$ fois plus d'arborescences m-cactacées sur $[\mathbf{n}]$ que d'arborescences m-cactacées non étiquetées sur $\mathbf{n}$ entraîne un surcoût prohibitif.

Dans cette optique, il serait intéressant d'avoir une preuve combinatoire de la formule de Good basée sur la manipulation d'objets non étiquetées et utilisant des séries génératrices ordinaires.

Comme nous l'avons déjà mentionné, il existe pour le cas unidimensionnel deux preuves de la formule de Lagrange manipulant des arborescences non étiquetées et dérivées de preuves "étiquetées". Elles sont dues à Chen [38, 39] et Joyal [96]. La preuve de Joyal est particulièrement intéressante car elle est basée sur le même principe que la preuve de Labelle [108] (Joyal utilise la théorie des espèces linéaires, voisine de la théorie des espèces classiques utilisée par Labelle pour sa preuve "étiquetée").

Dans le cas multidimensionnel, la seule preuve "non étiquetée" que nous connaissons est due à Chottin [42, 43], mais se limite au cas $m = 2$. Si la technique employée ne semble pas pouvoir s'étendre au delà, les objets considérés dans cette preuve fournissent cependant une piste intéressante. En effet, on déduit aisément des travaux de Chottin les points suivants.

- Le membre gauche de (5.7) est le nombre d'arborescences non étiquetées m-coloriées ayant n_i sommets de couleur c_i, ayant pour racine 0, et telles que
  - la fibre de chaque sommet (y compris la racine) est munie d'une structure d'ordre linéaire,
  - pour tout sommet de couleur c_i, aucun de ses frères gauches ne peut être de couleur c_j avec $j > i$,
  - un sommet de couleur c_i (resp. la racine 0) possédant dans sa fibre $f_1, \dots, f_m$ fils de couleurs respectives $c_1, \dots, c_m$ est pondéré par le coefficient de $[\mathbf{x}^\mathbf{f}]$ dans $F_i(\mathbf{x})$ (resp. $H(\mathbf{x})$).
- On peut interpréter le membre droit de (5.7) en termes d'*esquisse* d'une $(\mathcal{H},\mathcal{G})$-endofonction restreinte. Soit f une telle endofonction : son esquisse est obtenu en "oubliant", pour chaque élément u, les étiquettes de sa fibre de $f^{-1}(u)$, pour n'en conserver que la structure (le nombre de sommets de chaque couleur) et, si la fibre de u comporte f_i éléments de couleur c_i, on pondère u par le coefficient de $[\mathbf{x}^\mathbf{f}]$ dans $F_i(\mathbf{x})$ si $u \neq 0$ et de $H(\mathbf{x})$) si $u = 0$.

Remarque 125. Cette présentation de la formule de Lagrange due à Chottin souligne le fait que les arborescences de Cayley sont le modèle naturel pour la formule de Lagrange dans le cadre des séries génératrices exponentielles, alors que les arborescences ordonnées semblent être le modèle naturel pour la formule de Lagrange dans le cadre des séries génératrices ordinaires.

Cela suggère une piste séduisante pour tenter établir une preuve combinatoire "non étiquetée" de la formule de Good. Si on note $\mathcal{A}_{\mathcal{G},\mathcal{H}}$ (resp. $\mathcal{S}_{\mathcal{G},\mathcal{H}}$) les arborescences ordonnées (resp. esquisse d'endofonctions) définies précédemment, il faudrait définir deux algorithmes étiquetant canoniquement les arborescences de $\mathcal{A}_{\mathcal{G},\mathcal{H}}$ et les esquisses d'endofonctions de $\mathcal{S}_{\mathcal{G},\mathcal{H}}$ (on disposerait alors de structures étiquetées permettant d'appliquer la bijection Φ entre arborescences et endofonctions). Ces algorithmes devraient être tels que pour toute arborescence A obtenue en étiquetant une arborescence de $\mathcal{A}_{\mathcal{G},\mathcal{H}}$ (resp. toute endofonction f obtenue en étiquetant une esquisse d'endofonction de $\mathcal{S}_{\mathcal{G},\mathcal{H}}$), il existe une esquisse d'endofonction de $\mathcal{S}_{\mathcal{G},\mathcal{H}}$ tel que son étiquetage donne $\Phi(A)$ (resp. il existe une arborescence de $\mathcal{A}_{\mathcal{G},\mathcal{H}}$ telle que son étiquetage donne $\Phi(f)$).

Chapitre 6

Énumération de m-constellations

Ce dernier chapitre est à nouveau consacré à l'énumération et à la génération aléatoire de structures arborescentes coloriées à l'aide de la formule de Lagrange multidimensionnelle. Plus précisément, nous nous intéressons à une famille de cartes planaires m-coloriées, appelées *m-constellations*, qui généralisent les cactus m-aires étudiés dans le chapitre précédent et apparaissent notamment lors de l'énumération des factorisations minimales transitives de permutations [87, 25, 135]. Le but de ce chapitre est de donner une formule pour le nombre de m-constellations ayant n sommets et f faces, et d'en fournir une preuve combinatoire permettant d'obtenir un algorithme de génération aléatoire pour ces cartes. Dans le cas $m = 2$, nous obtenons une nouvelle formule dénombrant les cartes biparties, différente de celle due à Walsh [163].

Ce chapitre se décompose en trois parties.

- Dans une première section, nous présentons les m-constellations et leurs duales, les cartes m-eulériennes, ainsi que leur interprétation en termes de factorisations de permutations.
- Nous poursuivons par la présentation des arborescences m-eulériennes et de l'opération de clôture, une technique introduite par Schaeffer et Bousquet-Mélou (voir [25, 135] [1]) qui établit une bijection entre arborescences m-eulériennes équilibrées et cartes m-eulériennes.
- Finalement, nous introduisons une famille d'arborescences lagrangiennes, les *arborescences constellées*, qui généralisent les arborescences cactacées introduites lors de l'énumération de cactus m-aires, et nous établissons une bijection entre arborescences m-eulériennes et arborescences constellées. Nous terminons en appliquant la formule de Lagrange multivariée, afin d'énumérer les m-constellations ayant n sommets et f faces et nous déduisons de la preuve de ce résultat un algorithme de génération aléatoire.

1. La clôture d'arborescences, non limitée aux seules arborescences eulériennes et constellations, a été introduite par Schaeffer dans sa thèse [135, 136]. Elle permet d'énumérer et d'engendrer efficacement de nombreuses familles de cartes planaires.

6.1 Constellations et cartes eulériennes

On rappelle qu'une *carte planaire* C (cf. Cori [48] par exemple) est un plongement propre d'un graphe connexe sur la sphère et vérifie la relation suivante (formule d'Euler pour les cartes planaires) :

$$s(C) + f(C) - a(C) = 2, \tag{6.1}$$

où $s(C)$ est le nombre de sommets de C, $f(C)$ le nombre de faces de C et $a(C)$ le nombre d'arêtes de C. On appelle *degré* d'une face de C le nombre d'incidences d'arêtes à C.

6.1.1 Généralités sur les constellations

Nous reprenons la définition des *m-constellations* donnée dans [135].

Définition 126. Soit $m \geq 2$. Une *m-constellation* est une carte planaire dont chaque face est coloriée en noir ou en blanc de sorte que

- les faces adjacentes à une face blanche (resp. noire) sont noires (resp. blanches),
- les faces noires (resp. blanches) sont de degré m (resp. multiple de m).

Une m-constellation est enracinée si une de ses arêtes est distinguée.

Dans la suite de ce chapitre, on considère uniquement des m-constellations enracinées (on omet donc le qualificatif enracinée) et on désigne par le terme *polygones* les faces noires. De même que pour les cactus, les sommets situés autour de chaque polygone (à m côtés) sont coloriés successivement par les couleurs $c_1, c_2, \ldots, c_m$ dans le sens trigonométrique, avec la convention que l'arête racine relie un sommet de couleur c_{m-1} à un sommet de couleur c_m. On appelle m-constellation sur $\mathbf{n} = (n_1, \ldots, n_m)$ une m-constellation ayant n_i sommets de couleur c_i ($i = 1, \ldots, m$).

Remarque 127. Un cactus m-aire ayant p polygones est une m-constellation ayant p polygones et une seule face blanche.

Notation. Dans les figures présentant des m-constellations, on représente les polygones en grisé et l'arête racine est en pointillés.

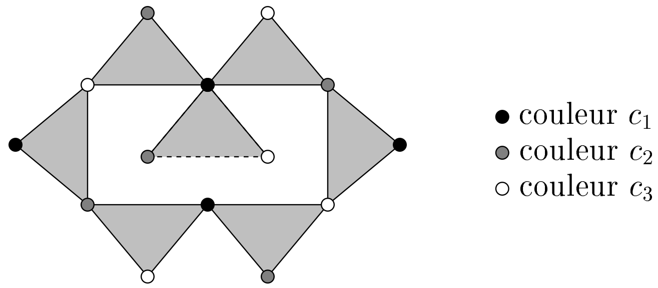


FIG. 6.1 – Une 3-constellation

Énumération de m-constellations. Dans [25, 135], Bousquet-Mélou et Schaeffer prouvent bijectivement les résultats suivants.

Théorème 128. [25, 135] *Le nombre de m-constellations C ayant d_i faces blanches de degré mi, pour $i = 1, \dots, m$ est*

$$m(m-1)^{f-1}\frac{[(m-1)n]!}{[(m-1)n-f+2]!}\prod_{i=1}^{m}\frac{1}{d_i!}\binom{mi-1}{i-1}^{d_i},$$

où $n = \sum_{i=1}^{m} id_i$ est le nombre de sommets de C et $f = \sum_{i=1}^{m} d_i$ est le nombre de faces blanches de C.

Corollaire 129. [25, 135] *Le nombre de m-constellations ayant p polygones est*

$$\frac{(m+1)m^{p-1}}{[(m-1)p+2][(m-1)p+1]}\binom{pm}{p}.$$

De plus, dans [135], Schaeffer déduit de la preuve du théorème 128, basée sur la clôture d'arborescences eulériennes, un algorithme de génération aléatoire uniforme de m-constellations ayant p polygones.

Dans la suite de ce chapitre nous nous intéressons aux m-constellations distribuées suivant le nombre de sommets et le nombre de faces (blanches) et nous prouvons combinatoirement le résultat suivant.

Théorème 130. *Soient m, n, f des entiers tels que $n + f - 2 = p(m-1)$ et $1 \leq f \leq p$. Le nombre de m-constellations ayant n sommets et f faces blanches est*

$$\left(\frac{(m-1)^{f-1}}{nf}\right)\sum_{k=0}^{p-f} m^{k+1}\binom{p-k-1}{f-1}\binom{p+n-k-2}{n-1}\binom{k+f-2}{f-2}, \tag{6.2}$$

si $f > 1$ et

$$\frac{1}{(m-1)p+1}\binom{mp}{p} \tag{6.3}$$

si $f = 1$.

On déduit de la remarque 127 que le cas $f = 1$ correspond en fait à la formule dénombrant les cactus m-aires suivant le nombre de sommets (voir [22, section 4.1] par exemple).

Nous déduirons alors de la preuve de ce résultat un algorithme de génération aléatoire uniforme de m-constellations ayant n sommets et f faces.

Avant de donner la preuve du théorème 130, nous pouvons rapidement justifier les conditions imposées à m, n et f dans l'énoncé du théorème 130. Soit C une constellation ayant n sommets, f faces blanches et p polygones (C a donc $f + p$ faces et pm arêtes). Par la formule d'Euler (6.1), on a alors $n + f + p - pm = 2$, ce qui est équivalent à $n + f - 2 = p(m-1)$. De plus si on associe à chaque polygone la face blanche à laquelle est incidente son arête reliant les sommets de couleur c_1 et c_2 (par exemple), on en déduit qu'on associe chaque polygone à une seule face blanche, et on a donc bien $1 \leq f \leq p$.

Remarque 131. Une m-constellation ayant n sommets et f faces blanches (avec $(n+f-2) = p(m-1)$) a p m-gones (et donc $p + f$ faces).

Définition 132. Une m-constellation est dite *régulière* si et seulement si toutes ses faces blanches ont degré m.

Proposition 133. [25, 135] *Il existe une bijection entre les m-constellations ayant n_i sommets de couleur c_i ($1 \leq i \leq m$), et f faces blanches, et les $(m+1)$-constellations régulières ayant n_i sommets de couleur c_i ($1 \leq i \leq m$) et f sommets de couleur c_{m+1}.*

La construction reliant m-constellations et $(m+1)$-constellations régulières est la suivante. Soit C une m-constellation : on ajoute au centre de chaque face blanche de C un sommet x de couleur c_{m+1} et on transforme chaque arête adjacente à cette face reliant un sommet y de couleur c_1 et un sommet z de couleur c_m en deux arêtes reliant respectivement x à y et x à z. On obtient de cette façon une $(m+1)$-constellation régulière, l'arête distinguée étant l'arête joignant le sommet de couleur c_m distingué dans C au sommet de couleur c_{m+1} ajouté au centre de la face blanche incidente à l'arête distinguée de C. La transformation inverse est immédiate (voir figure 6.2).

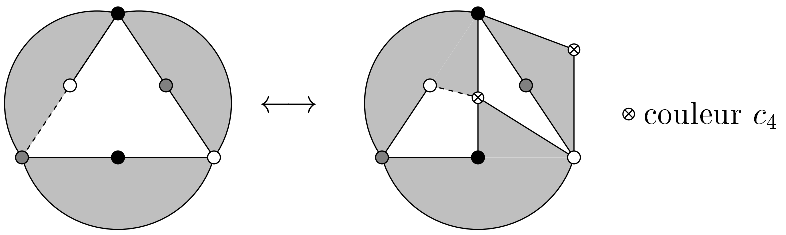


FIG. 6.2 – Bijection entre m-constellations et $(m+1)$-constellations régulières

Observation 134. Ainsi, pour énumérer (resp. engendrer) les m-constellations sur $\mathbf{n}$ ayant f faces, il suffit d'énumérer (resp. d'engendrer) les $(m+1)$-constellations régulières sur $(\mathbf{n},f) = (n_1,\dots,n_m,f)$.

Factorisations minimales transitives de permutations. Les constellations apparaissent notamment lors de l'énumération des factorisations minimales de permutations dans le groupe symétrique, problème que nous avons abordé au chapitre 2 (section 2.3) dans le cas particulier des factorisations d'une permutation circulaire en produit de transpositions.

Définition 135. Soit σ_0 une permutation sur $[n]$. Une *factorisation minimale transitive* de σ_0 est un ensemble ordonné $(\sigma_1,\dots,\sigma_m)$ de m permutations sur $[n]$ telles que

– $\sigma_0 = \sigma_1 \cdots \sigma_m$ (le produit étant effectué de gauche à droite),
– le groupe engendré par $\sigma_1,\dots,\sigma_m$ agit transitivement sur $[n]$,
– $\sum_{i=0}^{m} z(\sigma_i) = n(m-1)+2$, où $z(\sigma_i)$ est le nombre de cycles de σ_i.

Par exemple, si $\sigma_1 = (1,3)(2)$, $\sigma_2 = (1,2,3)$ et $\sigma_3 = (1)(2,3)$, alors $(\sigma_1,\sigma_2,\sigma_3)$ est une factorisation minimale transitive de $\sigma_0 = (1)(2)(3)$.

Remarque 136. On vérifie alors que la définition 34 définit en fait les factorisations minimales transitives de *permutations circulaires* en produit de *transpositions*. De plus, l'identité (2.2) que nous avons démontrée combinatoirement au chapitre 2 apparaît dans le cadre des factorisations minimales transitives de *permutations quelconques* en produit de *transpositions* [79].

Proposition 137. [25, 135] *Soient $p \geq 1$ et $m \geq 2$. Il existe une bijection entre*

- *les m-constellations C ayant p polygones étiquetés par $\{1,\dots,p\}$ telles que le polygone incident à l'arête racine soit étiqueté par 1, et*
- *les ensembles $(\sigma_0,\dots,\sigma_m)$ de permutations sur $[p]$ tels que $(\sigma_1,\dots,\sigma_m)$ est une factorisation minimale transitive de σ_0.*

De plus, si C a d_i faces (blanches) de degré $m \times i$, alors σ_0 a d_i cycles de longueur i.

Compte-tenu de la remarque 127, on déduit de ce résultat que les cactus m-aires étiquetés ayant p polygones sont en bijection avec les factorisations minimales transitives des permutations circulaires sur $[p]$.

6.1.2 Cartes m-eulériennes

Définition 138. Soit C une carte ayant s sommets, f faces et a arêtes. La *carte duale de C* est la carte C^* ayant f sommets, s faces et a arêtes telle que

- les sommets de C^* correspondent aux faces de C,
- les faces de C^* correspondent aux sommets de C,
- à toute arête e de C correspond une arête de C^* joignant les deux sommets associés aux deux faces incidentes à e.

Ainsi, la carte duale C^* de C décrit les relations d'adjacence entre les faces de C. On remarque que pour la carte duale d'une carte planaire la formule d'Euler (6.1) reste vraie que que cette carte duale est donc elle aussi planaire. Dans la figure suivante, nous avons superposé à la carte C sa carte duale en pointillés (partie gauche de la figure), avant de la redessiner sous une forme plus esthétique (partie droite de la figure).

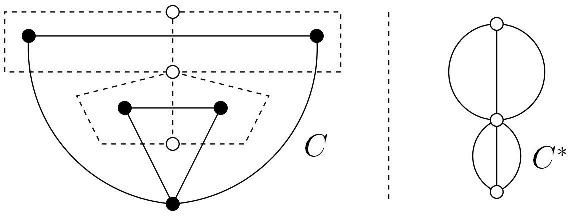


FIG. 6.3 – Deux cartes duales l'une de l'autre

Définition 139. Une *carte m-eulérienne* est une carte planaire bipartie (ses sommets sont de deux types[2], notés A et B, et toute arête relie un A-sommet à un B-sommet) telle que

- le degré de chaque A-sommet est égal à m,
- le degré de chaque B-sommet est un multiple de m.

Une carte m-eulérienne est dite *enracinée* si une de ses arêtes est distinguée.

Dans la suite de ce chapitre on ne considère que des cartes m-eulériennes enracinées, que l'on appelle donc simplement cartes m-eulériennes. La dénomination de carte m-eulérienne provient du fait que si $m = 2$, en lissant les A-sommets d'une carte m-eulérienne, on obtient une carte pour laquelle tout sommet a un degré pair, ce qui en fait une carte eulérienne [157, 134].

Notation. Dans les figures présentant des cartes m-eulériennes, on représente les A-sommets par des carrés et les B-sommets par des triangles et l'arête distinguée en pointillés.

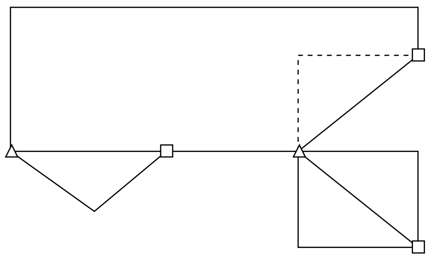


Fig. 6.4 – Une carte 3-eulérienne

On vérifie immédiatement que toute carte m-eulérienne E est la carte duale d'une m-constellation C : on fait correspondre aux polygones (resp. faces blanches) de C les A-sommets (resp. B-sommets) de E. De plus, dans le cas d'une m-constellation C sur $\mathbf{n}$, la coloration des sommets de C induit une coloration des faces de la carte m-eulérienne E. On dit alors que E est une *carte m-eulérienne sur* $\mathbf{n}$.

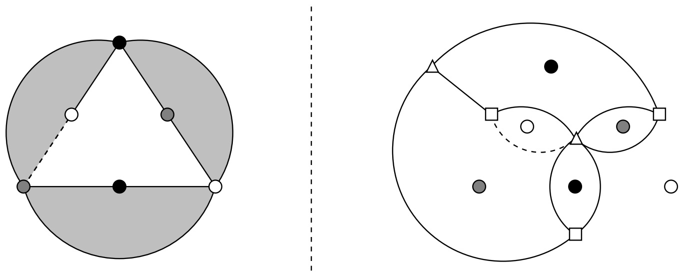


Fig. 6.5 – Une 3-constellation et la carte 3-eulérienne duale

Propriété 140. Si E est une carte m-eulérienne, l'arête racine de E sépare une face f' de couleur c_{m-1} et une face f de couleur c_m (en parcourant cette arête de son A-sommet vers son B-sommet, f est située à gauche et f' à droite).

Définition 141. Une carte m-eulérienne est dite *régulière* si tous ses sommets ont degré m.

On vérifie que la carte duale d'une m-constellation régulière est une carte m-eulérienne régulière. On a de plus la propriété suivante.

Propriété 142. Tout A-sommet (resp. B-sommet) x d'une carte m-eulérienne régulière est adjacent à exactement m faces et en tournant dans le sens trigonométrique autour de x, les couleurs des faces rencontrées sont successivement $c_1, c_2, \dots, c_m$ (resp. $c_m, c_{m-1}, \dots, c_1$).

2. Habituellement, les sommets d'une carte bipartie sont coloriés en blanc ou en noir. Nous utilisons ici la terminologie A-sommets et B-sommets pour éviter toute confusion avec une notion de coloration que nous introduisons section 6.2.

On déduit des propriétés 140 et 142 que la coloration des faces d'une carte m-eulérienne régulière est complètement déterminée par son arête racine.

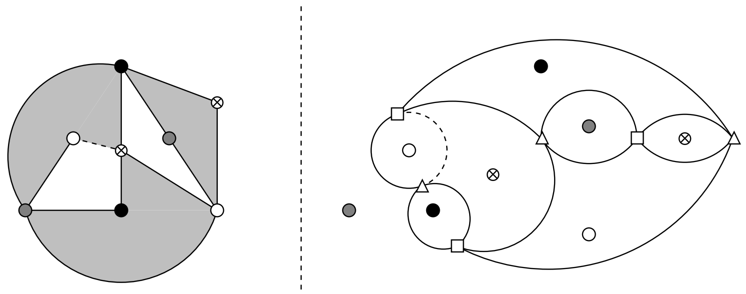


FIG. 6.6 – Une 4-constellation régulière et la carte 4-eulérienne régulière duale

Observation 143. On déduit alors de ce paragraphe et de la proposition 133 que pour énumérer (resp. engendrer) les m-constellations sur $\mathbf{n}$ ayant f faces, il suffit d'énumérer (resp. d'engendrer) les cartes $(m+1)$-eulériennes régulières sur $(\mathbf{n},f) = (n_1, \dots ,n_m,f)$.

6.2 Arborescences m-eulériennes régulières

La preuve du théorème 128 donnée dans [25, 135] est basée sur une bijection entre cartes m-eulériennes (non nécessairement régulières) et arborescences m-eulériennes. Dans cette section, nous donnons un aperçu de cette correspondance dans le cas particulier des cartes m-eulériennes régulières et des *arborescences m-eulériennes régulières.*

Arborescences m-eulériennes régulières. Une arborescence est dite *plantée* si et seulement si sa racine a un seul fils. Dans ce cas, on considère habituellement que le sommet racine est une feuille et non un nœud, et nous adoptons cette convention : une feuille d'une arborescence plantée est soit la racine soit un sommet sans fils, et un nœud est un sommet différent de la racine ayant au moins un fils.

Définition 144. Une *arborescence m-eulérienne régulière* est une arborescence plantée ordonnée non étiquetée dont les sommets peuvent être de deux sortes (A-sommets et B-sommets) telle que :
- toute arête relie un A-sommet et un B-sommet,
- sa racine est une A-feuille,
- tout nœud a $m-1$ fils,
- tout A-nœud a un et un seul B-nœud pour fils (les $m-2$ autres sont des B-feuilles).

Notation. Dans les figures présentant des arborescences m-eulériennes, on conserve la convention utilisée pour les cartes m-eulériennes, à savoir des carrés pour représenter les A-sommets et des triangles pour les B-sommets.

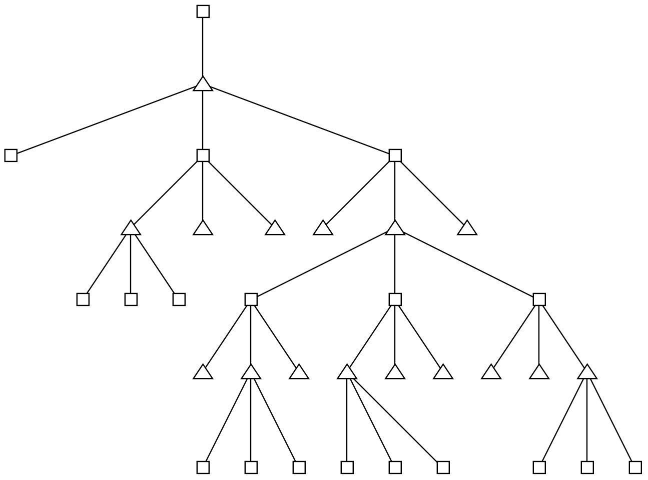


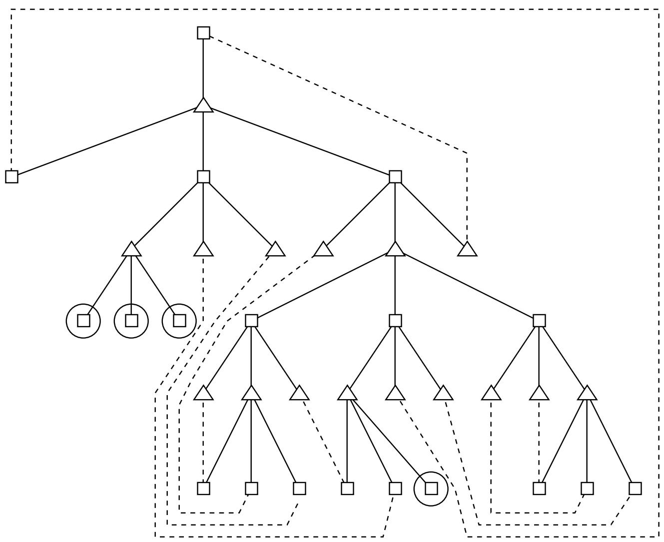
FIG. 6.7 – Une arborescence 4-eulérienne régulière

Clôture d'une arborescence m-eulérienne régulière. On rappelle que le *parcours préfixe en profondeur* d'une arborescence ordonnée T dont la racine a k fils consiste en la visite de la racine de A suivie, récursivement, du parcours des k sous-arborescences de T ayant pour racine les fils de r. On définit alors l'opération de *clôture* d'une arborescence m-eulérienne de la manière suivante.

Algorithme 7: *Clôture d'une arborescence m-eulérienne*

Donnée : T : arborescence m-eulérienne régulière
Résultat : $C(T)$: carte planaire
début
  soient P une pile vide et $C(T) := T$;
  pour tout sommet x visité lors du parcours préfixe en profondeur de T **faire**
    si x est une B-feuille **alors** empiler x;
    si x est une A-feuille et P est non vide **alors**
      soit y le sommet de la pile P que l'on dépile;
      relier x et y par une arête que l'on ajoute à $C(T)$ (les feuilles x et y sont dites appariées);
  tant que P est non vide **faire**
    soit y le sommet de la pile P que l'on dépile;
    relier la première (relativement au parcours préfixe en profondeur) A-feuille x non encore appariée à y par une arête que l'on ajoute à $C(T)$;
fin

La figure suivante donne un exemple d'exécution de cet algorithme (les arêtes ajoutées sont représentées en pointillés).



Définition 145. Une A-feuille x d'une arborescence m-eulérienne est dite *libre* si et seulement si, à l'issue de sa clôture, x n'est reliée (appariée) à aucune B-feuille.

Sur la figure précédente, les A-feuilles libres sont entourées d'une cercle. On vérifie la propriété suivante, conséquence de la définition des arborescences m-eulériennes et de l'opération de clôture d'une arborescence m-eulérienne (voir [25, 135]).

Propriété 146. Une arborescence m-eulérienne régulière T a exactement m A-feuilles libres, toutes situées sur la même face de $C(T)$.

Définition 147. Une arborescence m-eulérienne régulière est dite *équilibrée* si et seulement si sa racine est une A-feuille libre.

Proposition 148. [25, 135] *Il existe une bijection entre les arborescences m-eulériennes régulières équilibrées ayant k A-nœuds et l B-nœuds et les cartes m-eulériennes régulières ayant $k+1$ A-sommets et l B-sommets.*

Cette correspondance est basé sur la construction suivante. Soient T une arborescence m-eulérienne régulière équilibrée et $C(T)$ la carte planaire obtenue par clôture de T. Dans un premier temps, on ajoute dans la face de $C(T)$ contenant les m A-feuilles libres un A-nœud relié à m B-feuilles (cet ensemble de sommets est appelé une *étoile*) et on apparie ces B-feuilles avec les m A-feuilles libres, dans un ordre circulaire de sorte que les arêtes ainsi créées ne se croisent pas. On obtient ainsi une carte planaire notée $C'(T)$. Dans un second temps, pour chaque paire (x,y) de feuilles appariées, on remplace les trois arêtes (u,x), (x,y) et (v,y) par une arête (u,v) (u et v étaient les pères respectifs de x et y dans l'arborescence T ou l'étoile). Pour terminer, on distingue l'arête reliant le A-sommet ajouté au fils de la racine de T et passant par la racine. On obtient ainsi une carte m-eulérienne régulière E.

Propriété 149.

1. Les A-sommets de E (et donc les polygones de la m-constellation régulière duale de E) correspondent aux A-nœuds de T et au A-nœud de l'étoile (ce dernier correspondant au polygone contenant l'arête racine).

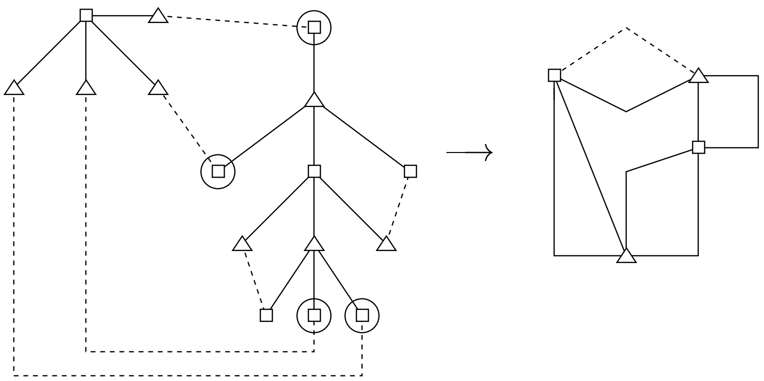


FIG. 6.8 – Clôture puis transformation d'une arborescence 4-eulérienne régulière équilibrée en une carte 4-eulérienne régulière

2. Les B-sommets de E (et donc les faces blanches de la m-constellation régulière duale de E) correspondent aux B-nœuds de T.

On remarque que pour toute face f de $C'(T)$, il existe une et une seule arête (x,y) la bordant où x est une A-feuille et y est une B-feuille et telle que son parcours de x vers y laisse la face f à main droite : on dit dans ce cas que la A-feuille x ferme la face f. On en déduit la propriété suivante.

Propriété 150. Soient T une arborescence m-eulérienne régulière équilibrée et E la carte m-eulérienne régulière régulière lui correspondant. La fermeture des faces de E par les A-feuilles de T induit une correspondance entre faces de la carte E et A-feuilles de l'arborescence T.

La construction inverse est plus complexe à décrire et, comme elle ne nous est pas utile par la suite, nous renvoyons le lecteur intéressé aux travaux [25, 135].

Coloration d'une arborescence m-eulérienne régulière. Dans ce paragraphe, nous décrivons un algorithme affectant une couleur de $\{c_1,\dots,c_m\}$ à chaque sommet d'une arborescence m-eulérienne régulière (non nécessairement équilibrée) de sorte que, dans le cas d'une arborescence équilibrée, la couleur d'une A-feuille est la couleur de la face fermée par cette A-feuille dans la carte m-eulérienne régulière lui correspondant. Nous en déduisons que l'énumération (resp. la génération) des cartes m-eulériennes régulières ayant n_i faces de couleur c_i (pour $i = 1,\dots,m$) se ramène à l'énumération (resp. la génération) des arborescences m-eulériennes régulières (non nécessairement équilibrées) ayant n_i A-feuilles de couleur c_i.

Soit T une arborescence m-eulérienne régulière. On colorie les sommets de T de la manière suivante (la figure 6.9 présente un exemple de cette construction) :
- la racine r et son fils sont coloriés par la couleur c_m,
- si un A-sommet (resp. B-sommet) x autre que la racine est colorié par la couleur c_i, ses $m-1$ fils sont successivement coloriés de gauche à droite (resp. droite à gauche) par les couleurs $c_{i+1},\dots,c_m,c_1,c_{i-1}$.

Une arborescence m-eulérienne régulière sur $\mathbf{n}$ est une arborescence ayant n_i A-feuilles de couleur c_i.

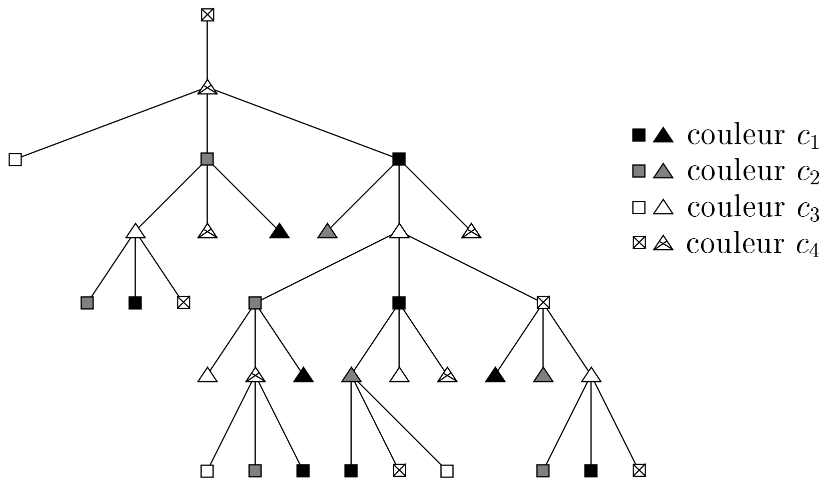
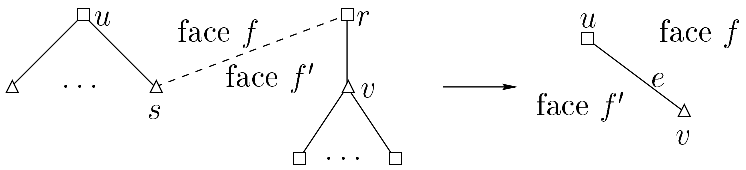


FIG. 6.9 – Coloration d'une arborescence 4-eulérienne régulière

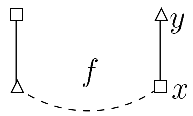
Lemme 151. *Soit E une carte m-eulérienne régulière et T l'arborescence m-eulérienne régulière équilibrée correspondante, coloriée par l'algorithme défini ci-dessus. Pour chaque face f de E, la couleur de la A-feuille de T fermant f est la couleur de f.*

Preuve. Considérons tout d'abord la A-feuille r racine de T. Après clôture de T et ajout de l'étoile (dont on note u le A-nœud), l'arête e racine de E est l'arête joignant le A-nœud u au fils v de la racine de T (voir figure ci-après).



La A-feuille r ferme la face distinguée notée f sur la figure (située à droite lorsque l'on parcourt l'arête de la A-feuille r vers s), cette face f étant située à gauche de l'arête e dans E lorsque cette arête est parcourue du A-sommet u vers le B-sommet v. La propriété 140 nous assure que cette face f est coloriée par c_m, ce qui justifie le choix de la couleur c_m pour la A-feuille racine de T effectuée dans l'algorithme de coloration.

Considérons maintenant une A-feuille x de T autre que la racine r et soient y le B-nœud père de x et f la face de $C'(T)$ fermée par x (voir figure ci-après).



La face f fermée par la A-feuille x est la face située à main gauche lorsque l'on remonte l'arête (x,y) de x vers y. Compte-tenu de la définition de l'algorithme de coloriage de T, conséquence directe de la propriété 142 et du fait que les A et B-nœuds de T correspondent aux A et B-sommets de E, il est clair que f et x ont même couleur puisqu'il en est ainsi pour la A-feuille racine de T et la face distinguée de E. □

Soit T une arborescence m-eulérienne régulière de racine r. On dit qu'on "repique" T en une de ses feuilles x si on plante T en x (x devient la racine) de la façon suivante : si le parcours préfixe en profondeur de T visite les sommets dans l'ordre $x_0 = r, x_1, \ldots, x_i = x, \ldots, x_n$, alors ce même parcours sur l'arborescence T repiquée en x visite les sommets dans l'ordre $x_i = x, x_{i+1}, \ldots, x_n, x_0 = r, x_1, \ldots, x_{i-1}$.

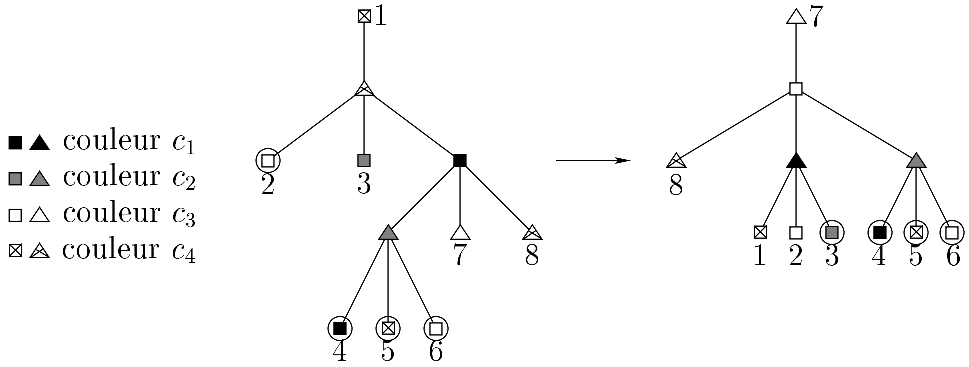


FIG. 6.10 – Repiquage d'une arborescence 4-eulérienne régulière en la B-feuille 7, conservation des A-feuilles libres (entourées d'un cercle) et des couleurs des feuilles

Compte-tenu des principes des algorithmes de clôture et de coloration, on obtient la propriété suivante.

Lemme 152. *Repiquer une arborescence m-eulérienne régulière de racine r en une feuille x ne modifie*

1. *ni les appariements de feuilles,*
2. *ni la coloration de ses feuilles (en coloriant bien sûr x à l'identique à l'initialisation de l'algorithme).*

Preuve. Le premier point est immédiat, le repiquage conservant l'ordre préfixe des feuilles et les appariements de feuilles étant obtenus par un parcours préfixe en profondeur de l'arborescence.

Pour le second point, supposons que la couleur d'une arête (x,y) de l'arborescence soit donnée par la couleur du sommet y fils de x. Ainsi, l'algorithme de coloration est tel que, lorsque l'on tourne autour d'un A-sommet (resp. B-sommet) dans le sens trigonométrique positif (resp. négatif), les arêtes sont coloriées $c_1, c_2, \ldots, c_m$. Le repiquage d'une arborescence ne modifie évidemment pas cette coloration des arêtes de l'arborescence et donc celle des feuilles (il n'en est pas exactement de même pour les nœuds). □

Il s'ensuit le résultat suivant.

Lemme 153. *Soit T une arborescence m-eulérienne régulière (non nécessairement équilibrée) coloriée par l'algorithme décrit précédemment. Pour tout $i = 1, \ldots, m$, il existe une et une seule A-feuille libre de T de couleur c_i.*

Preuve. Lors de la construction de la carte m-eulérienne régulière associée à l'arborescence T, chacune des faces incidentes au A-nœud de l'étoile est fermée par une A-feuille libre de T. Compte-tenu du lemme 151 et de la propriété 142, le résultat est donc vrai si T est équilibrée.

Supposons que T ne soit pas équilibrée. Soient x une feuille libre de T et T' l'arborescence obtenue en repiquant T en x. On déduit du lemme 152 que que T' est équilibrée, que ses A-feuilles libres sont celles de T et que leurs couleurs sont inchangées. T'

étant équilibrée, ses A-feuilles libres sont coloriées $c_1, c_2, \ldots, c_m$ d'après le raisonnement précédent. Ceci conclut la preuve du lemme. □

Lemme 154. *Si $a_{\mathbf{n}}$ est le nombre d'arborescences m-régulières sur $\mathbf{n}$ et b_n le nombre d'entre elles équilibrées, alors*

$$a_{\mathbf{n}} = n_m \times b_{\mathbf{n}}.$$

Preuve. Soit T une arborescence m-eulérienne régulière sur $\mathbf{n}$. On note $F(T)$ l'ensemble des arborescences obtenues en repiquant T en une des ses A-feuilles de couleur c_m [3]. On constate alors que

- $F(T)$ comporte exactement n_m arborescences (les appariements des feuilles étant conservés, on ne peut avoir deux arborescences identiques de cette façon),
- toutes les arborescences de $F(T)$ sont des arborescences sur $\mathbf{n}$ (lemme 152),
- seule l'arborescence de $F(T)$ repiquée en la A-feuille libre de couleur c_m (le lemme 153 assure qu'une telle A-feuille existe) est équilibrée,
- T appartient à $F(T)$ et pour toute arborescence T' de $F(T)$, $F(T') = F(T)$.

Le résultat découle de ces points. □

Observation 155. On déduit ainsi de l'observation 134 et des lemmes 154 et 151 que pour énumérer (resp. engendrer) les m-constellations sur $\mathbf{n}$ ayant f faces blanches, il suffit d'énumérer (resp. engendrer) les arborescences $(m+1)$-eulériennes régulières (non nécessairement équilibrées) dont les A-feuilles sont coloriées sur $(\mathbf{n}, f)$.

6.3 Énumération et génération aléatoire de constellations

Dans cette dernière section, nous commençons par montrer que les arborescences m-eulériennes régulières sont en bijection avec une famille d'arborescences qui généralisent les arborescences cactacées et que nous appelons *arborescences m-constellées*. On en déduit que l'énumération de m-constellations se ramène à l'énumération d'arborescences $(m+1)$-constellées. Cela ne nous a malheureusement pas donné la possibilité d'établir une formule satisfaisante résolvant le problème général, à savoir l'énumération suivant la distribution des couleurs des sommets et le nombre de faces blanches. Toutefois, cette construction reliant arborescences eulériennes régulières et arborescences constellées nous a permis de résoudre une version simplifiée du problème, l'énumération et la génération de constellations suivant le nombre n de sommets et le nombre f de faces blanches.

6.3.1 Arborescences constellées

Définition 156. Une arborescence *m-constellée* sur $\mathbf{n}$ est une arborescence non étiquetée m-coloriée sur $\mathbf{n}$ (ayant donc n_i sommets de couleur c_i, pour $i = 1, \ldots, m$), de racine de couleur c_m, telle que

- la fibre de la racine comporte $m-1$ fils, de couleurs respectives $c_{m-1}, \ldots, c_1$, ainsi ordonnés de gauche à droite par couleur décroissante,

3. Cette opération consistant à repiquer une arborescence, après sa clôture, en une de ses feuilles libres est à la base de la *conjugaison d'arborescences*, introduite par Schaeffer [135]. Pour une arborescence T, on peut en effet considérer l'ensembles $F(T)$ comme une classe de conjugaison d'arborescences.

– la fibre de tout sommet x de couleur c_i est structurée en une liste ordonnée de (i,j)-*blocs*, avec $i \neq j$, un (i,j)-bloc étant un ensemble (non ordonné) de $(m-2)$ sommets de couleurs deux à deux distinctes et autres que c_i et c_j.

Dans les figures, on représente les blocs par des rectangles et on “fusionne” les arêtes joignant les sommets d'un bloc à leur père en une seule arête.

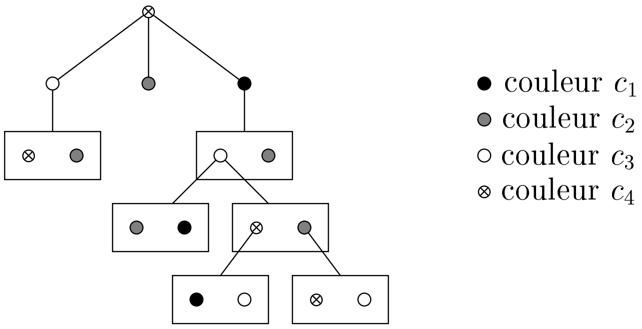
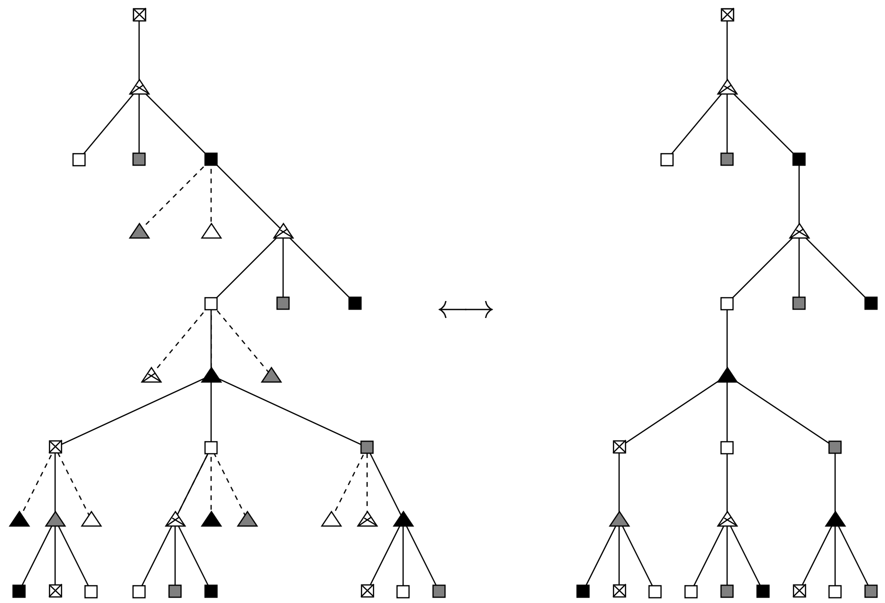


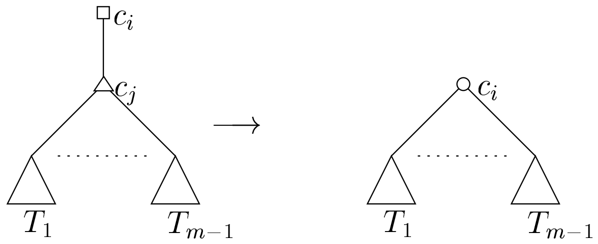
FIG. 6.11 – Une arborescence 4-constellée

Proposition 157. *Il existe une bijection entre les arborescences m-eulériennes régulières dont les A-feuilles sont coloriées sur* $\mathbf{n}$ *et les arborescences m-constellées sur* $\mathbf{n}$.

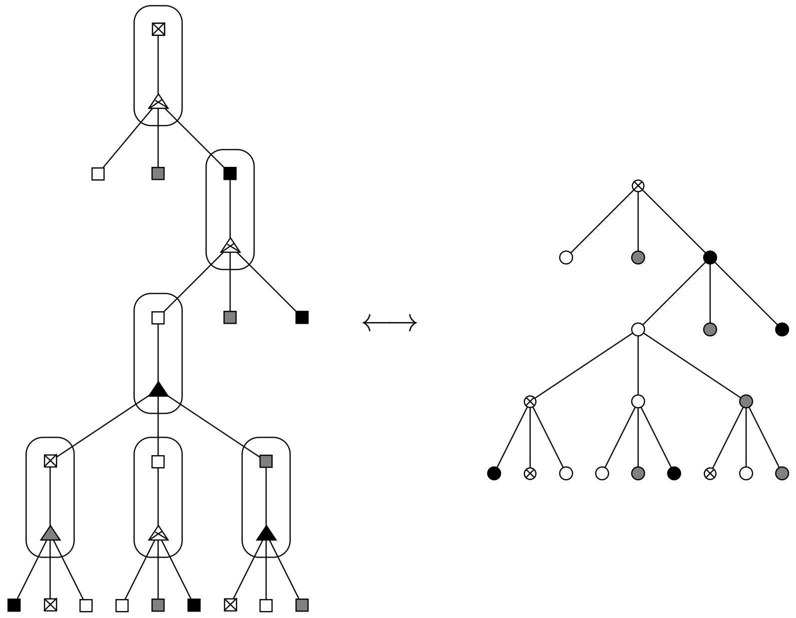
Preuve. Le principe de cette bijection consiste à supprimer peu à peu tous les nœuds et toutes les B-feuilles d'une arborescence m-eulérienne régulière. La construction se compose de 3 étapes. Soit T une arborescence m-eulérienne régulière aux A-feuilles coloriées sur $\mathbf{n}$ (on a n_i A-feuilles de couleur c_i, pour $i = 1, \ldots, m$).
Étape 1. Suppression des B-feuilles. On construit une arborescence T_1 à partir de T en supprimant toutes les B-feuilles de T. Comme la fibre de tout A-nœud de T comporte exactement un B-nœud (définition 144), la fibre de tout A-nœud (ainsi que celle de la racine) de T_1 est donc réduite à un B-nœud. La construction inverse, donnant T à partir de T_1 est immédiate, la position et les couleurs des B-feuilles à ajouter à la fibre d'un A-nœud x de T_1 se déduisant de la couleur de x, de la couleur du B-nœud fils de x et de l'algorithme de coloration d'une arborescence m-eulérienne régulière.



Étape 2. Suppression des B-nœuds. Chaque B-nœud x, situé dans la fibre d'un A-sommet y (x est donc le seul fils de y) est supprimé et sa fibre est greffée à son père y qui conserve sa couleur. On obtient ainsi une arborescence T_2 ne comportant que des A-sommets (on parle alors simplement de sommets, nœuds et feuilles, que l'on représente par des cercles). L'arborescence T_2 n'étant plus plantée, on considère sa racine comme un nœud et non plus comme une feuille.



L'arborescence de notre exemple devient :

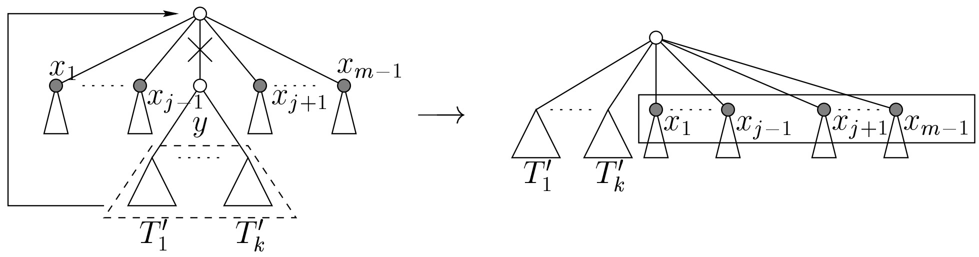


L'arborescence T_2 vérifie alors les propriétés suivantes, conséquences de l'algorithme de coloriage.

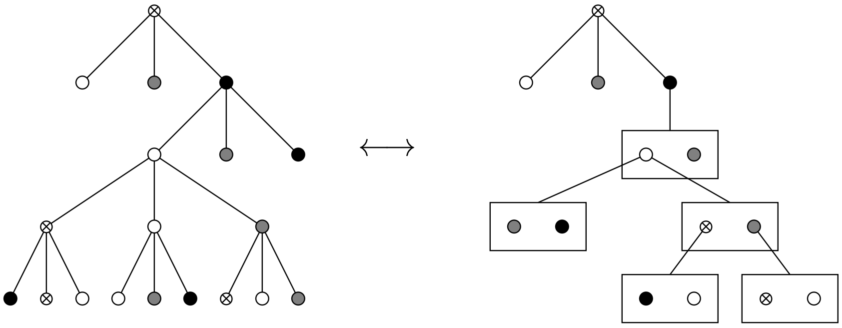
- La racine r est de couleur c_m et sa fibre est composée de $m-1$ sommets de couleurs $c_{m-1},\dots,c_1$, ainsi ordonnés de gauche à droite par couleur décroissante.
- Tout nœud $x \neq r$ de couleur c_i, a sa fibre composée de $m-1$ sommets de couleurs deux à deux distinctes, $c_k,c_{k-1},\dots,c_1,c_m,c_{m-1},\dots,c_{k+2}$ ainsi ordonnés de gauche à droite par couleur décroissante, une couleur autre que c_i étant absente (ainsi x de couleur c_i possède un fils de couleur c_i).
- L'arborescence T_2 a n_i feuilles de couleur c_i, pour $i=1,\dots,m-1$, et $n_m - 1$ feuilles de couleur c_m.

La construction inverse consiste à "dilater" tout nœud x de couleur c_i en une arête reliant x à un B-nœud y, à greffer la fibre de x sous y et à donner à y la couleur qui n'apparaissait pas dans la fibre de x.

Étape 3. Suppression des nœuds de T_2. Soient x de couleur c_i un nœud de T_2 autre que la racine, y le fils de x de même couleur c_i et c_j la couleur non présente dans la fibre de x. On supprime alors y en greffant sa fibre à gauche des autres fils de x qui, privés de y forment un (i,j)-bloc. En répétant cette opération pour chaque nœud de T_2, on obtient ainsi une arborescence m-constellée T' sur $\mathbf{n}$.



Pour reconstruire T_2 à partir de A', il convient de parcourir les sommets de T' en ordre préfixe (parcours en profondeur). Pour chaque nœud x (de couleur c_i) autre que la racine, il suffit de supprimer de sa fibre tous les blocs à l'exception du dernier (situé le plus à droite), d'ajouter un sommet y de couleur c_i à la fibre de x et de transférer les blocs supprimés de la fibre de x à celle de y.



Ceci termine la preuve de la proposition. □

La propriété suivante est une conséquence immédiate de la construction précédente. On la vérifie en suivant les étapes une à une.

Propriété 158. Soit k le nombre de blocs d'une arborescence m-constellée sur $\mathbf{n}$. L'arborescence m-eulérienne régulière coloriée sur $\mathbf{n}$ lui correspondant a alors $k+1$ B-nœuds.

La proposition précédente peut être considérée comme une généralisation du lemme 114, reliant cactus m-aires et arborescences m-cactacées. En effet, un cactus m-aire étant une m-constellation comportant une seule face blanche (remarque 127), il lui correspond une arborescence $(m+1)$-eulérienne régulière ayant exactement une A-feuille de couleur c_{m+1}. On déduit du lemme 153 que toute arborescence $(m+1)$-eulérienne régulière ayant une seule A-feuille de couleur c_{m+1} est équilibrée (cette unique A-feuille de couleur c_{m+1} étant libre et constitue donc la racine). Ainsi, compte-tenu de la proposition précédente, les cactus m-aires sont en bijection avec les arborescences $(m+1)$-constellées ayant un et un seul sommet de couleur c_{m+1}, qui par définition est la racine. Or, si T est une arborescence $(m+1)$-constellée sur $(\mathbf{n},1)$ (comportant donc exactement un sommet de couleur c_{m+1}, qui est sa racine), tous ses blocs sont nécessairement des $(i,m+1)$-blocs. Considérons alors le fils x de la racine de couleur c_1. En regroupant les $m-1$ autres fils de la racine (différents de x), on constitue un $(1,m+1)$-bloc. En transférant ce bloc dans la fibre de x, à droite de tous les blocs déjà présents dans cette fibre, et en supprimant le sommet racine, on obtient alors une arborescence m-cactacée sur $\mathbf{n}$.

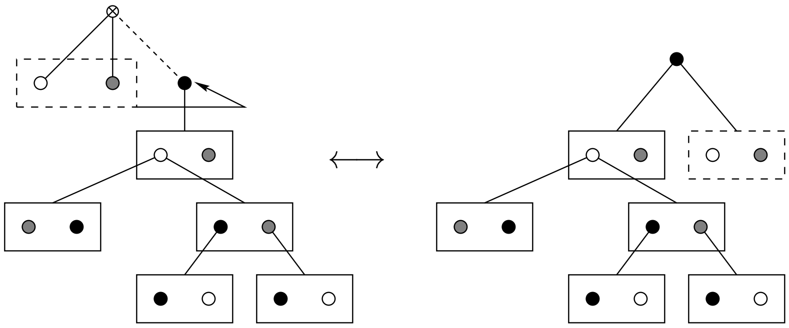


FIG. 6.12 – Une arborescence 4-constellée ayant un sommet de couleur c_4 et l'arborescence 3-cactacée correspondante

On peut cependant remarquer que pour un cactus m-aire donné, l'arborescence m-cactacée obtenue en appliquant la construction du lemme 114 n'est en général pas la même que l'arborescence m-cactacée obtenue en appliquant la construction précédente.

6.3.2 Énumération et génération aléatoire de m-constellations

Tous les éléments étant maintenant réunis, nous pouvons aborder le problème de l'énumération des m-constellations suivant le nombre de sommets et le nombre de faces. Plus précisément, nous allons démontrer le résultat énoncé lors de la présentation des constellations (théorème 130) et que nous rappelons ici.

Soient m, n et f des entiers tels que $n + f - 2 = p(m-1)$ et $1 \leq f \leq p$. Le nombre de m-constellations ayant n sommets et f faces blanches est

$$\left(\frac{(m-1)^{f-1}}{nf}\right)\sum_{k=0}^{p-f} m^{k+1}\binom{p-k-1}{f-1}\binom{p+n-k-2}{n-1}\binom{k+f-2}{f-2},$$

si $f > 1$ et

$$\frac{1}{(m-1)p+1}\binom{pm}{p}$$

si $f = 1$.

Nous allons donner une preuve combinatoire de cette formule, faisant intervenir la preuve combinatoire de la formule de Lagrange multidimensionnelle. Nous en déduirons un algorithme de génération aléatoire de m-constellations ayant un nombre de sommets et de faces blanches fixés.

6.3.2.1 Énumération de constellations

On déduit

- de la correspondance entre m-constellations et $(m+1)$-constellations régulières (proposition 133),
- de la correspondance entre m-constellations régulières et cartes m-eulériennes régulières (dualité, observation 143),
- de la correspondance entre cartes m-eulériennes régulières et arborescences m-eulériennes régulières équilibrées (proposition 148) et de la relation entre leurs colorations (lemme 151)
- du lemme 154 exprimant le nombre d'arborescences m-eulériennes régulières équilibrées en fonction du nombre d'arborescences m-eulériennes régulières,
- de la correspondance entre arborescences m-eulériennes régulières et arborescences m-constellées (proposition 157),

que, si $a_{n,f}$ est le nombre d'arborescences $(m+1)$-constellées sur $(\mathbf{n}, f)$, le nombre de m-constellations ayant m sommets et f faces blanches est

$$\frac{a_{n,f}}{f}. \qquad (6.4)$$

Pour énumérer les arborescences m-constellées selon les deux seuls paramètres n et f, nous introduisons une famille d'arborescences bicoloriées qui constituent une restriction des arborescences constellées. Nous les appelons *squelettes d'arborescences m-constellées*, ou de manière plus concise *m-squelettes.*

Définition 159. Soit $l \geq 3$. Un l-squelette est une arborescence non étiquetée dont les sommets sont de deux types (A-sommets et B-sommets) telle que

- la racine est un B-sommet, dont la fibre, qui comporte $l-1$ A-sommets, est munie d'une structure de liste ordonnée,
- la fibre de tout A-sommet est structurée en une liste ordonnée de *A-blocs* et de *B-blocs*, un A-bloc (resp. B-bloc) étant une liste ordonnée de $(l-2)$ A-sommets (resp. $(l-3)$ A-sommets et 1 B-sommet, qui est alors le dernier élément du bloc),
- la fibre de tout B-sommet est structurée en une liste ordonnée de A-blocs (un B-sommet n'a pas de B-sommet dans sa fibre).

Dans les figures, comme précédemment, nous représentons les A-sommets par des carrés, les B-sommets par des triangles et nous identifions toutes les arêtes joignant les sommets d'un bloc à son père en une seule (une arête d'un l-squelette relie donc un sommet à un bloc de sa fibre). On distingue de plus quatre types d'arêtes dans un squelette :

1. les arêtes de type 1 reliant la racine à un de ses fils,
2. les arêtes de type 2 reliant un A-sommet à un A-bloc,
3. les arêtes de type 3 reliant un A-sommet à un B-bloc,
4. les arêtes de type 4 reliant un B-sommet à un A-bloc.

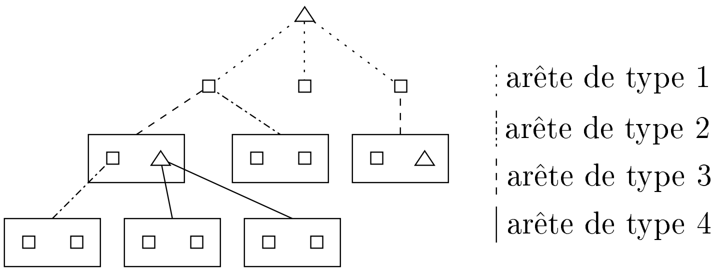


FIG. 6.13 – Un 4-squelette

Remarque 160. Soit T une arborescence l-constellée ayant n sommets de couleur $c_1, \ldots,$ c_{l-1} et n' sommets de couleur c_l. En en remplaçant chaque sommet de couleur $c_1, \ldots, c_{l-1}$ par un A-sommet et chaque sommet de couleur c_l par un B-sommet, on obtient un l-squelette S ayant n A-sommets et n' B-sommets. On appelle alors S le *squelette de A.*

Lemme 161. *Soient $l \geq 3$ et S un l-squelette ayant f B-sommets et k arêtes de type* 4. *Le nombre d'arborescences l-constellées dont S est le squelette est* $(l-2)^{f-1}(l-1)^k$.

Preuve. Pour obtenir une arborescence l-constellée à partir de S, on peut utiliser l'algorithme suivant. Tout d'abord on colorie la racine par la couleur c_l, les fils de la racine, de gauche à droite, par les couleurs $c_{l-1}, \ldots, c_1$. On effectue ensuite un parcours en profondeur de S et pour chaque bloc b rencontré, dont le père x a été précédemment colorié c_i $(i \neq l)$ s'il s'agissait d'un A-sommet, c_l dans le cas d'un B-sommet :

- si l'arête reliant x à b est de type 2 (x était un A-sommet et b ne contient pas de B-sommet), colorier les $l-2$ sommets de b par les $l-2$ couleurs c_j autres que c_i et c_l, de gauche à droite par couleur croissante,

- si l'arête reliant x à b est de type 3 (x était un A-sommet et b contient un B-sommet y), choisir un entier $k \in [l-1]$ différent de i, colorier y par c_l et les $l-3$ A-sommets de b par les couleurs c_j autres que c_i, c_k et c_l, de gauche à droite par couleur croissante,
- si l'arête reliant x à b est de type 4 (x de couleur c_l était un B-sommet) choisir un entier $k \in [l-1]$ et colorier les $l-2$ A-sommets de b par les couleurs c_j autres que c_k et c_l, de gauche à droite par couleur croissante.

La preuve du lemme découle directement de cet algorithme de coloration du squelette S. □

Lemme 162. *Soient $l \geq 3$, n et $f > 1$ trois entiers tels qu'il existe un entier p vérifiant $(n+f-2) = p(l-2)$ et $f \leq p$. Le nombre de l-squelettes ayant n A-sommets, f B-sommets et k arêtes de type 4 est*

$$\frac{(l-1)}{n}\binom{p-k-1}{f-1}\binom{p+n-k-2}{n-1}\binom{k+f-2}{f-2}.$$

Preuve. Tout d'abord, on peut remarquer qu'il existe un l-squelette ayant n A-sommets et f B-sommets si et seulement si il existe un entier p tel que $(n+f-2) = p(l-1)$, $p-1$ étant alors le nombre de blocs de ce squelette (donc de l'endofonction correspondante), et $1 \leq f \leq p$.

Ensuite, pour prouver le lemme, nous allons considérer des l-squelettes étiquetés, les A-sommets sur $[n]$ et les B-sommets sur $[f]$. De même que pour les arborescences l-cactacées (remarque 116), si $a_{l,n,f}$ (resp. $b_{l,n,f}$) est le nombre de l-squelettes sur (n,f) (resp. étiquetés sur $[n,f]$), nous avons alors

$$b_{l,n,f} = n! \times f! \times a_{l,n,f}. \tag{6.5}$$

Considérons les endofonctions sur des éléments de deux types (les A-éléments étiquetés sur [n] et les B-éléments étiquetés sur [f]) et notons $c_{l,n,f}$ le nombre de fonctions g de $[0,n,f]$ dans $[n,f]$ telles que

- la fibre $g^{-1}(0)$ de 0 comporte exactement $l-1$ A-éléments linéairement ordonnés, dont l'élément étiqueté 1 (ces endofonctions sont donc restreintes), et aucun B-élément,
- la fibre $g^{-1}(x)$ de tout A-élément x est structurée en une liste ordonnée de A-blocs et de B-blocs (définition 159),
- la fibre de tout B-sommet est structurée en une liste ordonnée de A-blocs.

On déduit alors de la preuve bijective de la formule de Lagrange bivariée que

$$b_{l,n,f} = f \times c_{l,n,f-1}, \tag{6.6}$$

le terme $f \times c_{l,n,f-1}$ provenant du fait que pour pouvoir appliquer la bijection entre arborescences et endofonctions restreintes, on doit transformer le B-sommet racine (dont l'étiquette appartient à $[f]$) en un sommet 0, ce qui donne une endofonction ayant $f-1$ B-éléments.

Il nous reste maintenant à énumérer ces endofonctions. Plus précisément, il nous faut déterminer le nombre d'endofonctions $c_{l,n,f-1}$ ayant k A-blocs dans les fibres des B-éléments. En effet, pour ce dernier point, on utilise le fait que la bijection entre arborescences et endofonctions préserve exactement le nombre d'éléments de chaque couleur (type) dans toutes les fibres (propriété 101). Procédons par étapes.

1. La fibre de l'élément 0. Examinons dans un premier temps la fibre $g^{-1}(0)$ de 0 : elle doit comporter $l-1$ A-éléments, dont le A-élément étiqueté 1, ordonnés linéairement. L'élément 0 peut donc avoir

$$(l-1)!\binom{n-1}{l-2} \tag{6.7}$$

fibres différentes.

2. Répartition des blocs. Les endofonctions considérées ont k A-blocs dans les fibres des $f-1$ B-éléments. Ainsi, les $p-1-k$ autres blocs sont dans les fibres des n A-éléments, $f-1$ étant des B-blocs (on rappelle que $f>1$). En utilisant le lemme 107 on a

$$\binom{k+f-2}{f-2} \tag{6.8}$$

façons de répartir les k A-blocs dans les fibres des f B-éléments et

$$\binom{p+n-k-2}{n-1}\binom{p-k-1}{f-1} \tag{6.9}$$

façons de répartir les $p-1-k$ autres blocs dans les fibres des n A-éléments, $f-1$ parmi ces $p-1-k$ blocs étant des B-blocs.

3. Placement des éléments dans les blocs. La fibre de l'élément 0 ayant été construite, il reste à placer $n-(l-1)$ A-éléments et les $f-1$ B-éléments, ce qui se fait de

$$(n-l+1)!(f-1)! \tag{6.10}$$

façons.

Le nombre d'endofonctions $c_{l,n,f-1}$ s'obtient en effectuant le produit des quatre termes (6.7), (6.8), (6.9), (6.10). Combiné à (6.5) et (6.6), on en déduit la formule du lemme. □

Preuve du théorème 130. La formule (6.2) du théorème 130 est une conséquence des lemmes 162 et 161, de (6.4), en remplaçant l par $m+1$ et du fait que, si $f>1$, le nombre d'arêtes de types 4 d'un squelette ayant $p-1$ blocs et $f-1$ B-sommets varie de 0 (les $f-1$ B-sommets sont des feuilles) à $p-f$ (tous les A-blocs sont des fils des B-sommets).

Pour le cas $f=1$ et la formule (6.3) du théorème, qui dénombre les cactus m-aires suivant le nombre de sommets (remarque 127), on applique le même raisonnement, en remarquant toutefois que les squelettes correspondant ne peuvent avoir d'arête de type 4 ($f=1$ et $k=0$ dans le lemme 162). □

Remarque 163. Compte-tenu de la formule d'Euler liant les trois paramètres nombre de sommets n, nombre de m-gones p et nombre de faces blanches f par $n+f-2=p(m-1)$ le théorème 130 exprime le dénombrement des m-constellations selon deux quelconques de ces trois paramètres.

Le cas $m=2$: les cartes biparties. Une carte bipartie est une carte planaire enracinée dont les sommets sont de deux types (A-sommets et B-sommets), telle que que toute arête relie un A-sommet à un B-sommet. Walsh [163] a prouvé le résultat suivant.

Théorème 164. [163] *Le nombre de cartes biparties ayant n ($n\geq 2$) sommets et f ($f>1$) faces est*

$$2\frac{(n+f-3)!}{(f-1)!n!}\sum_{k=0}^{n-2}\binom{2n+2f-4}{n-2-k}\binom{f+k}{k}.$$

Or, en remplaçant chaque A-sommet (resp. B-sommet) d'une carte bipartie par un sommet de couleur c_1 (resp. c_2), et chaque arête par un 2-gone, on obtient une 2-constellation.

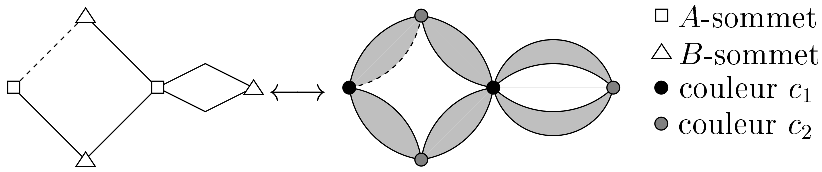


FIG. 6.14 – Une carte bipartie et la 2-constellation correspondante

En posant $m = 2$ dans le théorème 130 et en remarquant que $p = n + f - 2$, on obtient immédiatement le corollaire suivant, qui donne une formule différente de celle obtenue par Walsh pour l'énumération de cartes biparties.

Corollaire 165. *Le nombre de cartes biparties ayant n ($n \geq 2$) sommets et f faces ($f > 1$) est*

$$\frac{2}{n\ f} \sum_{k=0}^{n-2} 2^k \binom{n+f-3-k}{f-1} \binom{2n+f-k-4}{n-1} \binom{k+f-2}{f-2}.$$

6.3.2.2 Génération de constellations

Dans ce paragraphe, nous déduisons de la preuve du théorème 130 un algorithme de génération de m-constellations ayant n sommets et f faces. Le principe de cet algorithme est similaire à celui de l'algorithme de génération aléatoire de cactus présenté dans le chapitre précédent et comporte 4 étapes :

- engendrer une endofonction,
- appliquer la bijection entre endofonctions et arborescences, pour obtenir un squelette
- colorier ce squelette pour obtenir une arborescence constellée,
- appliquer le transformation entre arborescences constellées et constellations.

Cependant, il y a deux différences majeures qui compliquent l'application de ce principe. La première réside dans le fait que les endofonctions définies dans la preuve du lemme 162 le sont non seulement en fonctions des paramètres m, n et f, mais aussi en fonction du nombre k d'arêtes de type 4, qui influe lui-même sur le nombre de façons de colorier le squelette correspondant. Cette contrainte oblige, comme on le verra dans l'algorithme 8 à manipuler des nombres entiers longs. La seconde est liée à la transformation d'une arborescence constellée en constellation, qui est moins directe que dans le cas des cactus, sans toutefois alourdir la complexité asymptotique de l'algorithme.

Algorithme 8: Génération aléatoire de m-constellations

Données : trois entiers m, n et f

Résultat : m-constellation ayant n sommets et f faces blanches

début

$p := (n+f-2)/(m-1)$, $t := 0$;

pour k allant de 0 à $p-f$ **faire**

$$t_k := m^{k+1}\binom{p-k-1}{f-1}\binom{p+n-k-2}{n-1}\binom{k+f-2}{f-2};$$

$t := t + t_k$;

tirer un entier $0 \le k \le p-f$ avec probabilité t_k/t;

engendrer une endofonction g sur $[n,f-1]$ ayant k arêtes de type 4 (lemme 162);

1. engendrer la fibre de 0;
2. répartir les $p-1$ blocs;
3. placer les éléments dans les blocs;

transformer g en $(m+1)$-squelette S (par application de la bijection entre endofonctions et arborescences);

choisir l'étiquette $x \in [f]$ de la racine du $(m+1)$-squelette S;

transformer S en une arborescence $(m+1)$-constellée A en le coloriant aléatoirement (lemme 161);

transformer A en une arborescence $(m+1)$-eulérienne régulière B (proposition 157);

transformer B en carte $(m+1)$-eulérienne régulière E (proposition 148);

transformer E en $(m+1)$-constellation régulière D;

transformer D en m-constellation C (proposition 133);

retourner C

fin

Lemme 166. *L'Algorithme 8 permet d'engendrer aléatoirement et uniformément une m-constellation à n sommets et f faces blanches,en temps et espace $O^*(n+f)$ (la notation $O^*(n+f)$ indiquant que la complexité est linéaire en terme d'opérations manipulant des entiers arbitrairement longs).*

Preuve. Il est clair que cet algorithme engendre une m-constellation ayant n sommets et f faces blanches. Il reste à vérifier que cette génération est uniforme et que les complexités en temps et en espace sont linéaires.

1. Génération uniforme. L'uniformité de la génération est assurée par les premières étapes consistant à choisir le nombre k d'arêtes de type 4 de l'endofonction (et donc du squelette) que l'on va engendrer ensuite. En effet, on remarque immédiatement que t_k/t est la probabilité qu'une arborescence constellée ait un squelette ayant k arêtes de type 4. De plus, pour un squelette S donné, l'algorithme de coloration aléatoire d'un squelette (lemme 161) produit chaque arborescence constellée ayant S pour squelette avec la même probabilité (qui est $1/((m-1)^{f-1}m^k)$). Finalement, de la même manière que dans la preuve du lemme 122, on montre aisément que la génération aléatoire des endofonctions ayant sur $[n,f-1]$ et k arêtes de type 4 (étapes 1, 2 et 3) est uniforme.

2. Complexité linéaire en temps et en espace. La première partie de l'algorithme, consistant à choisir l'entier k, nécessite d'effectuer $O(p-f)$ opérations sur des entiers arbitrairement longs, où $p-1$ est le nombre de polygones de la constellation finale, et est donc inférieur à $(n+f)$. Le choix de k se fait donc en temps, $O^*(n+f)$. Comme dans la preuve du lemme 122, on montre aisément que la complexité de la génération du squelette S se fait en temps $O(n)$. La coloration de S se fait lors d'un parcours de ce squelette en temps $O(n+f)$. Chaque étape de la transformation de l'arborescence constellée A ainsi obtenue en arborescence eulérienne régulière B s'effectue en un parcours en profondeur de A et en temps $O(n)$. La transformation de B en carte eulérienne régulière E se fait en cloturant B, opération linéaire en temps et espace [25, 135]. Finalement, calculer la carte duale de E et appliquer la transformation entre $(m+1)$-constellations régulières et m-constellations ne pose aucun problème algorithmique et s'effectue en temps linéaire. □

6.4 Conclusion

L'intérêt de ce chapitre est de combiner deux constructions combinatoires intéressantes, la bijection entre arborescences et endofonctions et la conjugaison d'arborescences. Il reste cependant plusieurs questions que nous n'avons pu résoudre.

Énumération de constellations selon la distribution des couleurs. La première est une généralisation naturelle du théorème 130 : quel est le nombre de m-constellations ayant n_i sommets de couleur c_i $(i=1,\dots,m)$ et f faces blanches ? Dans le cas $f=1$, c'est-à-dire les cactus m-aires, le théorème 120 répond à cette question. Cependant, dès que $f>1$, nous n'avons pas pu étendre de manière satisfaisante la construction utilisée dans le cas des arborescences cactacées aux arborescences constellées. En effet, si on peut définir, de manière analogue aux endofonctions cactacées, une famille d'endofonctions *constellées* correspondant aux arborescences constellées, le fait que dans ces endofonctions il "manque" deux couleurs parmi les $(m+1)$ possibles (dans les blocs des endofonctions cactacées il ne manque qu'une couleur), introduit un degré de liberté difficile à gérer :

- dans le cas des endofonctions cactacées, un i-bloc est forcément dans la fibre d'un sommet de couleur c_i (voir la propriété (a) de la preuve du théorème 120), et on déduit ainsi de $\mathbf{n}$ le nombre de blocs présents dans les fibres des sommets de chaque couleur,
- dans le cas des endofonctions constellées, un (i,j)-bloc peut appartenir à la fibre d'un sommet de couleur c_i ou c_j et, pour une distribution $(\mathbf{n},f)$ donnée, le nombre de blocs présents dans les fibres des sommets de couleur c_i n'est donc pas unique.

Une des voies permettant de résoudre ce problème consiste peut-être à étudier le cas non trivial le plus simple, c'est-à-dire le cas des constellations à 2 faces $(f=2)$. On peut aussi noter que dans le cas $m=2$ (c'est-à-dire les cartes biparties), les blocs des arborescences 3-constellées (on rappelle qu'énumérer les 2-constellations revient à énumérer les arborescences 3-constellées) se réduisent à un seul sommet, et ces arborescences sont donc très proches des arborescences 3-chromatiques ordonnées, introduites en conclusion du chapitre précédent. Plus précisément, une arborescence 3-constellée est une arborescence 3-chromatique ordonnée telle que la racine est de couleur c_3 et a deux fils, un de chaque couleur c_3 exceptée, ordonnés par couleur décroissante. En utilisant la même technique que pour la formule (5.18) (technique basée sur la proposition 124), on peut ainsi obtenir une formule, particulièrement lourde et inélégante, pour le nombre de cartes biparties ayant

n_1 A-sommets, n_2 B-sommets et n_3 faces :

$$(n_1+q_1)(n-3)\sum_{k_1=0}^{n_1-1}\sum_{k_2=0}^{n_2-2}\sum_{k_3=0}^{n_3-1}\frac{(n_1+q_1-1)!(n_2+q_2-2)!(n_3+q_3-1)!}{(n_2-1)\mathbf{k}!(\mathbf{n}-\mathbf{k}-\mathbf{1})!}+$$

$$(n_2+q_2)(n-3)\sum_{k_1=0}^{n_1-2}\sum_{k_2=0}^{n_2-1}\sum_{k_3=0}^{n_3-1}\frac{(n_1+q_1-2)!(n_2+q_2-1)!(n_3+q_3-1)!}{(n_1-1)\mathbf{k}!(\mathbf{n}-\mathbf{k}-\mathbf{1})!}+$$

$$(n_1+q_1)(n_2+q_2)\sum_{k_1=0}^{n_1-1}\sum_{k_2=0}^{n_2-1}\sum_{k_3=0}^{n_3-1}\frac{(n_1+q_1-1)!(n_2+q_2-1)!(n_3+q_3)!}{\mathbf{k}!(\mathbf{n}-\mathbf{k}-\mathbf{1})!},$$

où $n=n_1+n_2+n_3$, $q_1=k_2+k_3$, $q_2=n_3+k_1-k_3-1$, $q_3=n_1+n_2-k_1-k_2-3$, $\mathbf{n}=(n_1,n_2,n_3)$ et $\mathbf{q}=(q_1,q_2,q_3)$.

Distribution des degrés. Il serait aussi intéressant d'interpréter la distribution des degrés des sommets de chaque couleur et des degrés des faces d'une constellation en termes d'arborescences constellées, comme on peut le faire dans le cas des cactus.

Constellations non enracinées. Finalement, il est naturel de vouloir énumérer les constellations non enracinées non étiquetées. Dans le cas des cactus (constellations à une seule face blanche), Bousquet [22] a remarqué qu'une méthode introduite par Liskovets [111] permettait de résoudre ce problème. Il semble que l'on puisse appliquer la méthode de Liskovets aux constellations [25, Théorème 3.1].

Deuxième partie

Recherche de motifs dans une arborescence

Chapitre 7

Recherche de motifs dans une arborescence

Ce chapitre est une introduction au problème de la recherche de motifs dans une arborescence. Après quelques définitions (section 7.1), nous nous intéressons au cas des mots et nous présentons deux structures de données classiques, l'automate de Aho-Corasick et l'arborescence des suffixes, que nous utilisons par la suite. Nous poursuivons par une rapide revue des principaux articles parus sur le sujet de la recherche de motifs dans une arborescence, avant de conclure par l'extension d'un de ces algorithmes dû à Hoffmann et O'Donnell [91].

Notation. Dans la suite, toutes les complexités considérées, sauf mention contraire, sont relatives à la complexité dans le pire des cas.

7.1 Introduction

Dans cette seconde partie de notre mémoire, nous considérons des arborescences ordonnées et étiquetées sur un alphabet Σ fixé, que nous désignons par le terme arborescences sur Σ, définies récursivement de la manière suivante.

Définition 167. Une arborescence A sur Σ est constituée d'un sommet, appelé racine de A, muni d'une étiquette appartenant à Σ, relié à un *ensemble totalement ordonné* (éventuellement vide) d'arborescences sur Σ.

On remarque que contrairement aux arborescences considérées dans la première partie de ce mémoire, une même étiquette peut apparaître plusieurs fois dans une arborescence sur Σ.

Définition 168. Soit A une arborescence et x un noeud de A. Si x est relié à la liste $A_1, \ldots, A_k$ d'arborescences de racines respectives $x_1, \ldots, x_k$, on appelle x_i le $i^{ème}$ fils de x. On dit que x_i est un frère gauche (resp. droit) de x_j si et seulement si $i < j$ (resp. $i > j$).

Recherche de motifs dans une arborescence quelconque. Soient M et T deux arborescences. Intuitivement, on dit que T (le *texte*) possède une occurrence de M (le *motif*) en un sommet x si, en plaçant la racine de M en x, on arrive à une situation dans laquelle tout sommet (resp. arête) de M recouvre un sommet (resp. arête) de T ayant la même étiquette. On traduit formellement cette notion de la façon suivante.

Définition 169. Soient M et T deux arborescences sur Σ, et x un sommet de T. Le texte T possède une *occurrence de M en x* si et seulement si il existe une correspondance entre les sommets de M et ceux de T telle que :

1. tout sommet de M est en correspondance avec un sommet de T ;
2. la racine de M est en correspondance avec x ;
3. si un sommet y de M est en correspondance avec un sommet z de T, y et z ont la même étiquette ;
4. un noeud y de M est en correspondance avec un noeud z de T, alors chaque fils de y correspond à un fils de z ;
5. si y_1 et y_2 sont deux sommets de M tels que y_1 est un frère gauche (resp. droit) de y_2, et si z_1 et z_2 sont les sommets de T correspondant respectivement à y_1 et y_2, alors z_1 est un frère gauche (resp. droit) de z_2.

Par exemple, dans la figure suivante, le texte T comporte une seule occurrence du motif M ("matérialisée" par les arêtes en pointillé).

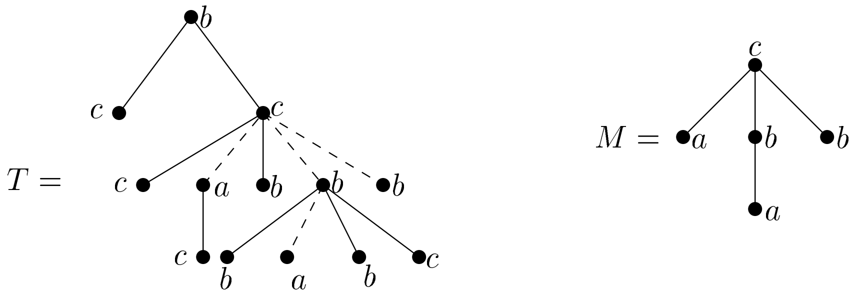


FIG. 7.1 – Illustration de la notion d'occurrence d'un motif dans un texte

Définition 170. Le problème de la recherche de motifs dans une arborescence, que nous appelons problème RMA, consiste, étant données deux arborescences T et M, à trouver tous les sommets x de T tels que T admette au moins une occurrence de M en ce sommet x.

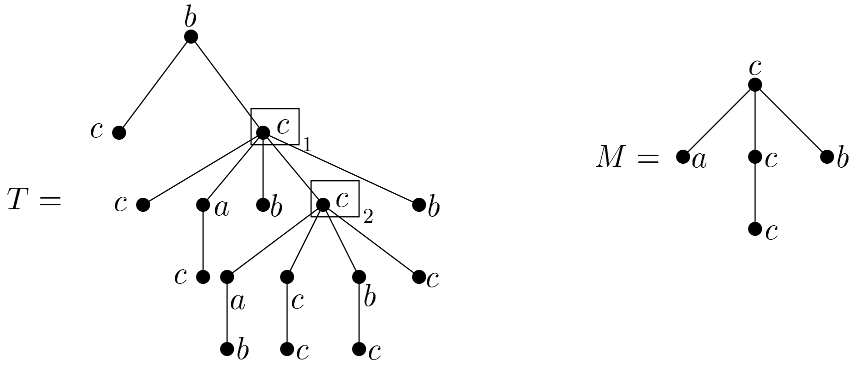


FIG. 7.2 – Le motif M apparaît deux fois dans le texte (en les sommets encadrés)

Cependant, comme nous le verrons dans la section suivante, les différents travaux traitant de recherche de motifs dans une arborescence s'intéressent à une restriction de ce problème, basée sur la notion d'*occurrence compacte d'un motif dans un texte.*

Définition 171. Soient M et T deux arborescences sur Σ, et x un sommet de T. On dit que le texte T possède une *occurrence compacte de* M *en* x si et seulement si il existe une correspondance entre les sommets de M et ceux de T telle que :

1. tout sommet de M est en correspondance avec un sommet de T ;
2. la racine de M est en correspondance avec x ;
3. si un sommet y de M est en correspondance avec un sommet z de T, y et z ont la même étiquette ;
4. si le nœud y de M correspond au nœud z de T, alors le $i^{ème}$ fils de y correspond au $i^{ème}$ fils de z.

On appelle alors *recherche de motifs compacts* dans une arborescence (noté RMCA) le problème RMA dans lequel on ne cherche à détecter que les occurrences compactes du motif dans le texte. Par exemple, dans la figure 7.2 seule la l'occurrence de M dans T (de racine le sommet encadré indicé 2) est une occurrence compacte.

Cette version restreinte du problème de la recherche de motifs a reçu plus d'attention que le problème général, en partie car elle peut être reformuler immédiatement en termes d'arborescences binaires, qui se prêtent mieux à un traitement algorithmique que les arborescences quelconques.

Définition 172. Une arborescence binaire A sur Σ est une arborescence dans laquelle tout noeud a un ou deux fils, appelés ses fils gauche et droit : dans le cas où un sommet x de A n'a qu'un seul fils, ce dernier est soit fils gauche de x, soit fils droit de x.

Il existe une bijection classique entre les arborescence sur Σ à n sommets et les arborescences binaires sur Σ à n sommets telles que la racine n'a qu'un fils gauche. Étant donnée une arborescence A sur Σ, cette bijection consiste à disposer la liste des fils $\{y_1,y_2,\dots,y_p\}$ d'un sommet x sur une même branche droite de sorte que

- y_1 soit fils gauche de x,
- y_{i+1} soit fils droit de y_i pour $1 \leq i < p$.

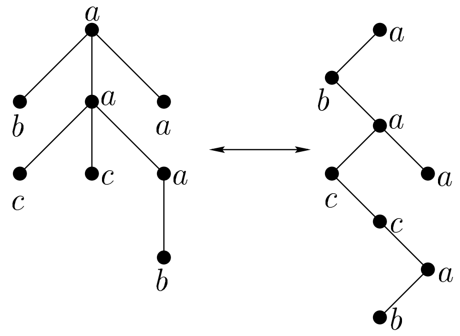


FIG. 7.3 – La transformation fondamentale de Knuth

La construction inverse est immédiate et le caractère bijectif de cette transformation (souvent appelée *transformation fondamentale de Knuth* [102, section 2.3.2]) se vérifie sans difficulté, tout comme la propriété suivante.

Propriété 173. Soient T et M deux arborescences sur Σ, T' et M' les arborescences binaires sur Σ correspondantes, x un sommet de T et x' le sommet correspondant de T'. Il y a une occurrence compacte de M en x si et seulement il y a une occurrence compacte de M' en y.

Cette propriété permet de ramener le problème RMCA pour les arborescences quelconques au problème équivalent pour les arborescences binaires telles que la racine a un fils gauche. Par contre, on vérifie immédiatement que le problème RMA ne possède pas de traduction directe en termes d'arborescences binaires.

Recherche de motifs dans un terme. La notion de *terme* (ou *arborescence d'expression*) est une extension de la notion d'arborescence, qui apparaît notamment dans les langages de programmations, les systèmes de preuve automatique, ou plus généralement les systèmes de réécriture. On considère dans ce cas chaque symbole s de Σ comme un symbole fonctionnel auquel est associé une *arité* notée $\nu(s)$.

Définition 174. Un terme sur (Σ,ν) est une arborescence sur Σ telle que chaque sommet x a un nombre de fils qui est exactement égal à l'arité $\nu(e(x))$ de son étiquette $e(x)$ appartenant à Σ.

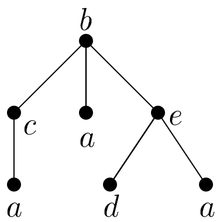


FIG. 7.4 – Un terme avec $\nu(a) = \nu(d) = 0$, $\nu(c) = 1$, $\nu(e) = 2$ et $\nu(b) = 3$

Dans le cas des termes, on le fait que l'arité d'un sommet est fixée par son étiquette implique que le problème RMA est équivalent au problème RMCA moyennant l'ajout d'une étiquette dite "variable" pour le motif M.

Définition 175. Un *motif* sur (Σ,ν) est un terme sur $(\Sigma\cup\{v\},\nu)$ (on appelle v la *variable*), avec $\nu(v) = 0$, et un *texte* sur (Σ,ν) est un terme sur (Σ,ν).

On définit alors la notion d'occurrence d'un motif M dans un terme T en remplaçant la condition 2 de la définition 169 (ou de la définition 171, les deux définitions devenant alors équivalentes pour les termes) par

2. si un sommet y de M est en correspondance avec un sommet z de T, soit y et z ont même étiquette, soit l'étiquette de y est v.

Dans la figure suivante, on a $\Sigma = \{a,b,c,d\}$ et $\nu(a) = 4$, $\nu(b) = 1$ et $\nu(c) = \nu(d) = 0$.

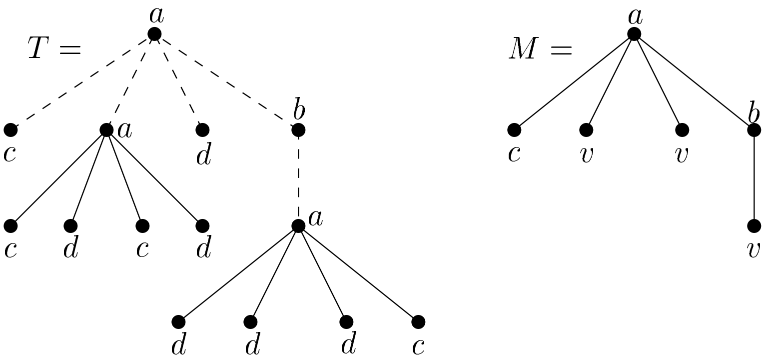


FIG. 7.5 – Une occurrence de M dans T (en la racine)

7.2 Recherche de motifs dans un mot

Soient M et T deux mots sur un alphabet Σ. Rechercher les occurrences du motif M dans le texte T consiste à repérer l'ensemble des facteurs de T identiques à M. Du fait de ses nombreuses applications pratiques (logiciels pour l'édition de texte, analyse de séquences biologiques, commandes de systèmes d'exploitation, etc), ce problème a été largement étudié comme l'illustre l'ouvrage de synthèse de Crochemore et Rytter [52].

Un mot de longueur n pouvant naturellement être considéré comme une arborescence à n sommets dont chaque noeud a exactement un fils (une telle arborescence comporte donc une seule feuille), le problème de la recherche de motifs dans une arborescence est une extension naturelle du problème restreint aux mots. Comme il existe plusieurs manières de coder une arborescence par un ensemble de mots, la plupart des algorithmes résolvant le problème RMA sont basés sur des algorithmes ou des structures de données issus du problème de la recherche de motifs dans les mots.

On peut diviser les algorithmes opérant sur les mots en deux familles. La première regroupe les algorithmes basés sur un prétraitement du motif M qui permet d'accélérer le parcours du texte T. Ce principe est notamment adapté au cas où le texte n'est pas fixé et ne peut donc être prétraité. On dit alors que le texte est *dynamique*. Ce sont des algorithmes appartenant à cette famille qui sont implantés dans les logiciels d'édition de texte ou la commande `grep` du système UNIX par exemple (les algorithmes les plus célèbres de cette nature sont dûs à Knuth, Morris et Pratt [104], Aho et Corasick [1], Boyer et Moore [26] et permettent, après un prétraitement en temps et espace $O(|M|)$ du motif M, de détecter les occurrences de ce motif en temps $O(|T| + |M|)$).

La seconde famille regroupe les algorithmes opérant sur un texte fixé (dit *statique*). Ils sont basés sur un prétraitement du texte T, consistant à construire une structure de données (occupant un espace en $O(|T|)$) indexant les suffixes de T, à partir de laquelle on peut rechercher rapidement (en temps linéaire ou quasi-linéaire par rapport à $|M|$) les occurrences du motif M dans T. Les structures de données classiques utilisées dans ce cas sont l'arborescence des suffixes [118], l'automate des suffixes [20, 51] ou encore le tableau des suffixes [117].

Dans les deux paragraphes qui suivent, nous présentons deux structures de données utilisées pour résoudre le problème de la recherche de motifs dans les mots. Nous utiliserons ces structures de données dans les algorithmes de recherche de motifs dans une arborescence décrites dans la dernière section de ce chapitre et dans les deux derniers chapitres.

On rappelle qu'on appelle *préfixe* (resp. *suffixe*) d'un mot M tout mot M_1 tel qu'il existe un mot M_2 vérifiant $M = M_1M_2$ (resp. $M = M_2M_1$). Si $M_1 \neq M$, on dit que M_1 est un préfixe (resp. suffixe) *propre* de M. Par exemple, $abca$ est un préfixe propre de $abcacba$ et un suffixe propre de $acbabca$.

7.2.1 Le cas dynamique et l'automate de Aho-Corasick

Cet algorithme aborde le problème plus général consistant à se donner un ensemble $\{M_1, \ldots, M_k\}$ de mots sur Σ (les motifs) et à rechercher toutes les occurrences de l'un des motifs $M_1, \ldots, M_k$ dans le texte T (voir [52, Section 7.1] et [1]). Le prétraitement des motifs $\{M_1, \ldots, M_k\}$ est basé sur une structure de donnée classique, appelée *arbre digital de recherche* (que nous notons ADR).

Définition 176. Soit $\{M_1, \ldots, M_k\}$ un ensemble de mots sur Σ. L'arborescence digitale de recherche (ADR) associée à $\{M_1, \ldots, M_k\}$ est l'arborescence dont les arêtes sont étiquetées par des lettres de Σ et vérifiant :

- deux arêtes issues d'un même sommet ont des étiquettes différentes,
- tout chemin dans l'ADR débutant à la racine est étiqueté par un mot préfixe de l'un des mots $\{M_1, \ldots, M_k\}$,
- tout mot de $\{M_1, \ldots, M_k\}$ étiquette un chemin débutant à la racine de l'ADR.

Pour chaque sommet u de cette ADR, on note $m(u)$ le mot formé des étiquettes des arêtes du chemin allant de la racine à u (on dit alors que le sommet u reconnaît $m(u)$ et que u est un *sommet final* si il existe i tel que $M_i = m(x)$). De plus, en orientant les arêtes de la racine vers les feuilles, on transforme cette ADR en un automate déterministe (mais non forcément complet ou minimal) reconnaissant le langage formé des mots $M_1, \ldots, M_k$: on parle d'état au lieu de sommets (la racine devient l'état initial et les sommets finaux, les états finaux) et de transitions au lieu d'arêtes.

Le prétraitement de $\{M_1, \ldots, M_k\}$ comporte deux étapes. Tout d'abord, on construit l'automate correspondant à l'ADR associée à $\{M_1, \ldots, M_k\}$, que l'on complète ensuite. Ainsi, pour chaque état u et pour chaque lettre $a \in \Sigma$ telle qu'aucune transition étiquetée a ne quitte u, on ajoute une transition (étiquetée a) de u vers l'état reconnaissant le plus long suffixe propre du mot $m(u).a$ (qui est donc aussi préfixe d'un des M_i). On note $AC(M_1,\ldots,M_k)$ l'automate ainsi construit.

Dans la figure suivante, on représente l'état de $AC(M_1,\ldots,M_k)$) (resp. le sommet de l'ADR associé à $\{M_1,\ldots,M_k\}$) reconnaissant M_i par un carré contenant l'entier i et les transitions ajoutées lors de la seconde phase de prétraitement en pointillé.

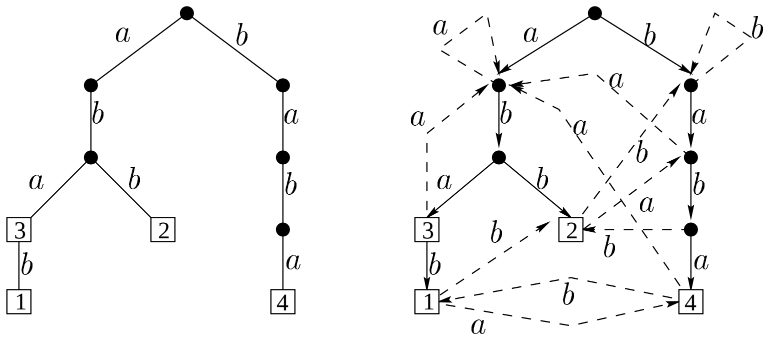


FIG. 7.6 – L'ADR de ($abab$, abb, aba, $baba$) et l'automate $AC(abab, abb, aba, baba)$

Pour rechercher les positions des occurrences des motifs $(M_1,\ldots,M_k)$ dans un texte T, il suffit d'appliquer l'algorithme suivant (on note $T[i]$ la $i^{ème}$ lettre d'un mot T).

Algorithme 9: Recherche de motifs avec l'automate de Aho-Corasick.

Données : $AC(M_1,\ldots,M_k)$ et T de longueur n
début
  Soit r l'état initial de $AC(M_1,\ldots,M_k)$ et $u := r$;
  pour i allant de 1 à n **faire**
    $u := \delta(u,T[i])$;
    si x est un état final étiqueté j **alors**
      Afficher "Occurrence de M_j en position $i - |M_j| + 1$";
fin

Il reste à étudier la complexité en temps et en espace de cet algorithme (on pose $|T| = n$ et $\sum_i |M_i| = m$). La phase de recherche de motifs s'effectue clairement en temps $O(n)$ (l'automate étant complet, on peut stocker les transitions quittant un état par un tableau indexé par les lettres de Σ assurant ainsi un accès en temps $O(1)$ à chacun de ces arcs). L'automate $AC(M_1,\ldots,M_k)$ occupe un espace en $O(m \times |\Sigma|)$, et Aho et Corasick ont proposé un algorithme permettant de construire cet automate temps $O(m \times |\Sigma|)$ [1].

Remarque 177. Pour être plus précis, on remarque que si l'ADR associée à $\{M_1,\ldots,M_k\}$ comporte p sommets, la transformation de cette ADR en l'automate $AC(M_1,\ldots,M_k)$ s'effectue en temps $O(p \times |\Sigma|)$.

Remarque 178. Dans le cas d'un alphabet Σ de taille importante, un automate complet occupe souvent trop de place. On peut cependant ramener cet espace à $O(m)$ en ne complétant pas l'automate correspondant à l'ADR et en organisant les transitions quittant un sommet dans une arborescence binaire de recherche équilibrée par exemple (voir [50] pour une présentation de cette structure de données). En calculant pour chaque état

x une transition spéciale (car non étiquetée) reliant u à l'état v tel que $m(v)$ est le plus grand suffixe propre de $m(u)$ reconnu par l'automate, on peut alors effectuer la phase de recherche de motifs en temps $O(n \times \log(|\Sigma|))$. Cependant, cette technique ne peut être employée efficacement dans le cas de la recherche de motifs dans une arborescence, comme nous le verrons dans les deux prochaines sections. Pour cette raison nous privilégions cette version utilisant un automate complet.

Notation. Dans la suite de ce mémoire, nous appelons *structure de branchement* la structure de données utilisée pour stocker des transitions étiquetées sur Σ dans un automate ou une arborescence aux arêtes étiquetées, cette structure pouvant être de deux types (dans les applications qui nous intéressent) :

- des tableaux indexés par les lettres de Σ (chaque état/sommet nécessite un espace en $O(|\Sigma|)$ pour stocker les transitions/arêtes le quittant et l'accès à une transition/arête s'effectue en temps $O(1)$),
- des arborescences binaires de recherche équilibrées (un état/sommet ayant k transitions/arêtes le quittant nécessite un espace en $O(k)$ et l'accès à chaque transition/arête s'effectue en temps $O(\log(|\Sigma|))$).

7.2.2 Le cas statique et l'arborescence des suffixes

Arborescence des suffixes. Soit T un mot de longueur n sur Σ. On note $Suf(T) = \{T[1,n], \ldots ,T[n,n]\}$ (où $T[i,j]$ est le facteur de T commençant en position i et finissant en position j) l'ensemble des suffixes de T, ordonnés par longueur décroissante. Par exemple, pour $T = acbab$, on a $Suf(T) = \{acbab,cbab,bab,ab,b\}$.

Définition 179. L'arborescence des suffixes (AS) d'un mot T, notée $AS(T)$, est obtenue en comprimant chaque chemin de l'ADR de $Suf(T)$ ne comportant que des sommets non finaux ayant un seul fils en une seule arête étiquetée par le facteur de T correspondant au chemin ainsi comprimé. On obtient ainsi une arborescence dans laquelle chaque arête est étiquetée par un facteur $T[i,j]$ de T qui peut donc être codée par le couple d'entiers (i,j) appelé désormais l'étiquette de l'arête.

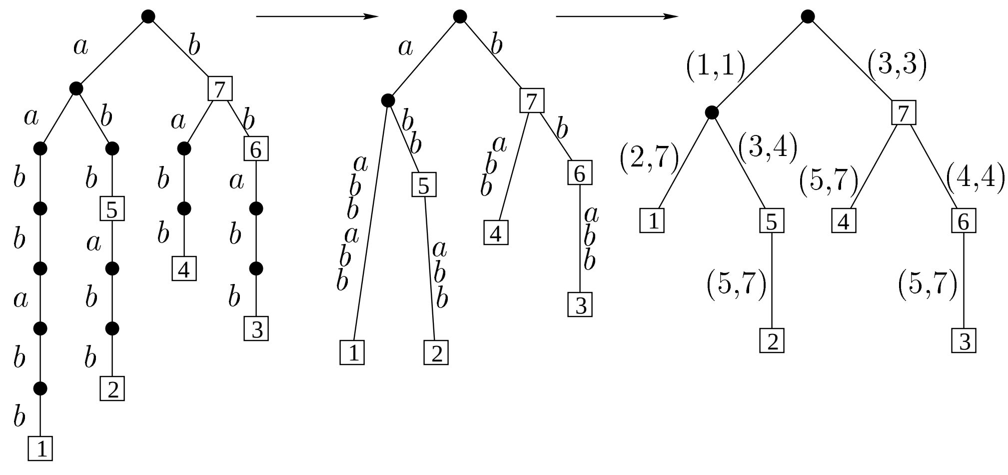


FIG. 7.7 – De l'ADR de $Suf(aabbabb)$ à l'arborescence des suffixes de $aabbabb$

Recherche de motifs. Dans le cas statique, le prétraitement de l'algorithme de recherche d'un motif dans un mot T (le texte fixé) consiste alors à construire l'arborescence des suffixes $AS(T)$. La phase de recherche consiste, étant donné un motif M, à appliquer l'algorithme suivant.

Algorithme 10: Recherche de motif avec l'arborescence des suffixes.

Données : T, $AS(T)$ et M
début
  soit r la racine de $AS(T)$, $u := r$ et $i := 1$;
  tant que $i \leq |M|$ **faire**
    si il existe une arête (j,k) (allant de u vers v) telle que $T[j] = M[i]$ **alors**
      $l := j$;
      tant que $i \leq |M|$, $l \leq k$ et $T[l] = M[i]$ **faire** $l := l + 1$ et $i := i + 1$;
1   **si** $i > |M|$ **alors** parcourir la sous-arborescence de $AS(T)$ de racine v pour obtenir toutes les positions des occurrences de M dans T (elles sont données par les sommets finaux de cette sous-arborescence);
      sinon si $l > k$ **alors** $u := v$ **sinon** sortir;
    sinon sortir;
fin

En remarquant que l'arborescence des suffixes comporte exactement $n = |T|$ sommets finaux et que tout nœud non final possède au moins deux fils, on peut affirmer que $AS(T)$ a au plus $2n$ sommets. Tout comme pour l'automate AC introduit précédemment, la complexité en espace de $AS(T)$ dépend de la structure de branchement utilisée. L'utilisation de tableaux indexés par les lettres de Σ implique une complexité en espace en $O(n \times |\Sigma|)$, alors qu'utiliser des arborescences binaires de recherche équilibrées entraîne une complexité en espace en $O(n)$[1]. La phase de recherche de motifs proprement dite (l'algorithme 10) ne nécessite aucun espace additionnel et, si il y a k occurrences de M dans T, elle s'effectue clairement en temps $O(m + k)$ ou $O(m \times \log(|\Sigma|) + k)$ selon la structure de branchement utilisée (le terme k provient du fait qu'une fois un mot reconnu, la sous-arborescence de $AS(T)$ parcourue lors de l'étape 1 a au plus k sommets finaux et donc au plus $2k$ sommets). Il reste à examiner la complexité en temps du prétraitement du texte T, c'est-à-dire la complexité en temps de la construction de l'arborescence des suffixes $AS(T)$. Il existe plusieurs algorithmes permettant de construire l'arborescence des suffixes d'un mot de longueur n sur Σ en temps $O(n \times \log(|\Sigma|))$ si on utilise des arborescences binaires de recherche équilibrées, comme nous le verrons dans le paragraphe suivant et le chapitre 9).

Quelques rappels sur l'arborescence des suffixes. L'arborescence des suffixes, introduite par Weiner [164], s'est imposée comme une structure de données fondamentale en algorithmique des mots (comme le montrent par exemple les articles de synthèse d'Apostolico [5, 6]). Dans le paragraphe précédent, nous avons vu son utilité dans le cadre de la recherche de motifs dans un mot. L'arborescence des suffixes s'avère aussi utile pour

1. Gusfield [86] remarque que, dans la plupart des cas, les nœuds situés à faible distance de la racine peuvent avoir un degré important alors que les autres nœuds ont un degré faible. Il semble donc intéressant de mélanger les structures de branchement au sein d'une même arborescence pour obtenir un compromis entre espace et temps de branchement.

résoudre un problème proche, la recherche de motifs approchés [32, 159, 45, 109]. On peut aussi remarquer que l'arborescence des suffixes d'un mot T permet de trouver les facteurs de T qui se répètent : si un noeud u de $AS(T)$ a, parmi ses descendants, k sommets finaux, on en déduit que le facteur $m(u)$ apparaît k fois dans T (ces occurrences pouvant cependant se chevaucher). Cette propriété peut s'avérer extrêmement utile pour la compression de textes [133].

Le premier algorithme non trivial pour la construction de l'arborescence des suffixes d'un mot T de longueur n est dû à Weiner [164]. Cet algorithme construit l'arborescence des suffixes $AS(T)$ incrémentalement en insérant les suffixes de T dans une ADR initialement vide, du plus petit au plus long suffixe, en temps et espace $O(n \times |\Sigma|)$. Par la suite, McCreight [118] a proposé un nouvel algorithme, basé sur un principe analogue à celui de Weiner, mais en insérant les suffixes de T du plus long au plus petit, ce qui a pour effet de construire $AS(T)$ en temps $O(n \times \log(|\Sigma|))$. Nous présentons en détail cet algorithme dans le prochain chapitre. Plus récemment, Ukkonen [160] a proposé un algorithme élégant, linéaire en temps, qui présente l'avantage d'être "en ligne" : l'arborescence des suffixes $AS(T)$ est construite incrémentalement au fur et à mesure que l'on lit le texte T. Dans un article de synthèse, Giegerich et Kurtz [71] ont unifié la présentation de ces trois algorithmes. Enfin, parmi les algorithmes de construction d'une arborescence des suffixes, on peut aussi noter les travaux d'Apostolico et al. [8] proposant un algorithme parallèle de construction de $AS(T)$, qui, dans le cas séquentiel, est linéaire en temps (modulo le temps de branchement), et de Farach [60], qui donne le premier algorithme indépendant de la taille de l'alphabet Σ à condition que Σ soit l'alphabet constitué des symboles $\{1,2,\dots,n\}$.

Plus récemment, plusieurs auteurs se sont intéressés aux propriétés pratiques de cette structure de données. En effet, bien qu'elle ait des propriétés théoriques optimales, l'arborescence des suffixes s'avère souffrir de certaines limitations pratiques. Par exemple, pour un texte anglais classique, l'espace occupé en moyenne par l'arborescence des suffixes est d'environ 15 octets par lettre du texte initial. Le tableau des suffixes, proposé par Manber et Myers [117] et le cactus des suffixes défini par Kärkkäinen [98] semblent être des alternatives à l'arborescence des suffixes ayant un intérêt d'un point de vue pratique. Dans ce même domaine, on peut aussi se référer aux travaux d'Andersson [4], qui propose une implantation efficace de l'arborescence des suffixes, ou à ceux de Giegerich et Kurtz [70]. Toujours dans une optique d'optimisation de l'espace occupé par l'arborescence des suffixes, certains auteurs se sont intéressés au cas où on ne veut indexer qu'un sous-ensemble des suffixes d'un texte (par exemple dans le cas d'un texte composé de mots) et ont proposé des algorithmes efficaces pour ce problème [3, 99].

7.3 Résultats connus pour les arborescences

Dans ce paragraphe, nous passons rapidement en revue les principaux travaux et résultats obtenus pour le problème de la recherche de motifs dans une arborescence. Le but de cette section n'est pas tant de décrire dans le détail les algorithmes connus, ces derniers étant souvent assez complexes, mais de mettre en avant les principes sur lesquels ils reposent.

Notation. Par la suite, n représente le nombre de sommets du texte T et m le nombre de sommets du motif M. De plus, pour un sommet x d'une arborescence A, on note A_x la sous-arborescence de A de racine x et $e(x)$ son étiquette.

L'algorithme naïf. L'algorithme le plus naturel (aussi appelé *algorithme naïf* pour détecter les occurrences d'une arborescence M dans une arborescence T est basé sur une procédure récursive de comparaison d'arborescences.

Algorithme 11: Algorithme naïf

Données : M et T
début
  pour Chaque sommet x de T **faire**
    si Comparer(M**,** T_x**)** $=$ Vrai **alors** afficher "Occurrence de M en x";
fin

Algorithme 12: Comparer

Données : deux arborescences A et A'
début
1 soient x la racine de A et y celle de A';
  si $e(x) \neq e(y)$ **alors retourner** Faux;
  sinon
2   soient $x_1, \ldots, x_k$ les fils de x et $y_1, \ldots, y_l$ ceux de y, $i := 1$ et $j := 1$;
    tant que $i \leq k$ et $j \leq l$ **faire**
      si Comparer(A_{x_i}**,** A'_{y_j}**)** $=$ Vrai **alors** i := i+1;
3      $j := j + 1$;
    si $i = k + 1$ **alors retourner** Vrai **sinon retourner** Faux;
fin

Pour traiter le problème RMCA, on peut simplifier la procédure **Comparer** en remplaçant le bloc situé entre les lignes 2 et 3 par le bloc suivant.

---

soient $x_1, \ldots, x_k$ les fils de x, $y_1, \ldots, y_l$ ceux de y et $i := 1$;
tant que $i \leq k$, $i \leq l$ et **Comparer(**A_{x_i}**,** A'_{y_i}**)** $=$ Vrai **faire** $i := i + 1$;

La complexité en temps de cet algorithme est déterminée, dans les deux cas, par le nombre d'appels à la procédure **Comparer**, ou de manière équivalente, par le nombre de comparaisons entre les étiquettes des sommets de T et celles des sommets de M (ligne 1 de l'algorithme 12).

Définition 180. Soient A une arborescence et x un sommet de A. On appelle *hauteur de* x, noté $h(x)$, le nombre de sommets situés sur le chemin allant de la racine de A à x. On appelle hauteur de A, noté $h(A)$, le nombre de sommets présents sur le plus long chemin reliant la racine de A à une de ses feuilles.

Proposition 181. [54] *Soient T et M deux arborescences de tailles respectives n et m. L'algorithme naïf détecte toutes les occurrences (compactes ou non) de M dans T en temps $O(n \times h(M))$.*

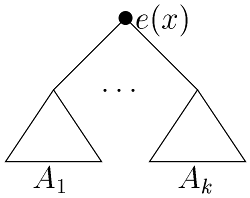
Flajolet et Steyaert [149], dans un article illustrant l'utilisation de techniques de combinatoire analytique pour l'analyse en moyenne d'algorithmes, ont prouvé le résultat suivant.

Proposition 182. *Soient T et M deux termes de tailles respectives n et m. L'algorithme naïf détecte toutes les occurrences (compactes ou non) de M dans T en temps $O(n+m)$* en moyenne.

Remarque 183. Dans le problème restreint qui consiste à rechercher l'occurrence de M en un sommet donné x de T, la complexité en temps de l'algorithme naïf dépend fortement de la notion d'occurrence prise en compte (compacte ou non). Pour le cas d'une occurrence compacte, l'algorithme naïf résoud ce problème en temps $O(m)$, alors que dans le cas d'une occurrence quelconque, la complexité en temps est en $O(|T_x|)$.

L'algorithme montant de Hoffmann et O'Donnell. Les premiers algorithmes non triviaux pour la recherche de motifs dans une arborescence sont dûs à Hoffmann et O'Donnell [91]. Dans cet article, les auteurs proposent deux algorithmes, aux performances très différentes.

Le premier est dédié aux termes, et plus précisément au problème de la recherche de motifs dont au moins une des feuilles est étiquetée par la variable v (le cas où M ne comporte aucun sommet étiqueté v peut être résolu en temps linéaire en utilisant le codage d'un terme par un mot [115]). Le principe de cet algorithme consiste à considérer l'ensemble $F(M)$ des sous-arborescences de M (pour chaque arborescence A de $F(M)$ il existe au moins un sommet y de M tel que $M_y = A$) et à associer à chaque sommet x de T le sous-ensemble des arborescences de $F(M)$ ayant une occurrence en x. On note $MS(x)$ cet ensemble d'arborescences. Pour une feuille x d'étiquette a, l'ensemble $MS(x)$ comporte les deux arborescences réduites à un seul sommet, étiqueté v (la variable) ou a. Partant de cette constatation, l'algorithme détermine les ensembles $MS(x)$ récursivement (en remontant des feuilles vers la racine) : si x est un sommet de degré k (ayant k fils notés $x_1, \ldots, x_k$, dont on suppose connaître les ensembles $MS(x_i)$), $MS(x)$ comprend l'arborescence réduite au sommet étiqueté v et toutes les arborescences de $F(M)$ de la forme



telles que A_i appartient à $MS(x_i)$. En repérant les ensembles $MS(x)$ pour lesquels M appartient à $MS(x)$, on réduit ainsi la phase de recherche à un parcours de T, qui s'effectue en temps $O(n)$. Le point crucial consiste donc à calculer, pour tout noeud x, $MS(x)$ à partir de l'étiquette a de x et des ensembles $MS(x_i)$ de ses fils x_i. Pour cela, l'algorithme d'Hoffmann et O'Donnell nécessite la calcul, pour chaque symbole a de Σ, d'une table de dimension k si a est d'arité k, indexée par des sous-ensembles de $F(M)$. La principale tâche réside dans le calcul de ces tables, qui peuvent cependant occuper un espace en $O(2^m)$. Hoffmann et O'Donnell introduisent une sous-classe de l'ensemble des motifs possibles, appelée classe des motifs simples, pour lesquels le calcul de ces tables peut être effectué en temps $O(m^2 \times d + m^{d+1} \times |\Sigma|)$ et en espace $O(m^2 + |\Sigma| \times m^d)$, où d est l'arité maximale des symboles de Σ. Hoffmann et O'Donnell affirment de plus que dans les applications pour lesquelles ils ont utilisé cet algorithme (notamment l'implantation d'un interpréteur LISP), le fait de se restreindre à cette classe des motifs simples n'est pas contraignant. Cet algorithme a ensuite suscité plusieurs articles donnant une amélioration de la complexité en espace du prétraitement, travaux dûs notamment à Chase [33], Cai, Paige et Tarjan

[28] ou Thorup [155]. Les techniques mises en œuvre sont cependant toujours complexes, et il est clair qu'elles sont étroitement liées à la notion d'arité d'un sommet fixée par son étiquette et ne peuvent donc être appliquées qu'aux termes.

L'algorithme descendant de Hoffmann et O'Donnell. Le second algorithme proposé dans [91] aborde le problème de la recherche des occurrences compactes d'un motif dans un texte. Il suit un principe différent du précédent et privilégie un prétraitement plus léger au détriment, bien sûr, de la phase de recherche. Dans un premier temps, chaque sommet x de M autre que la racine est dédoublé de la manière suivante : si x est le $i^{ème}$ fils de son père, on remplace x par deux sommets étiquetés i et $e(x)$, et reliés par une arête (voir figure 7.8). On construit ensuite, à partir de cette arborescence, l'ADR des mots correspondant aux chemins allant de la racine vers les feuilles. Dans un second temps, à partir de cette ADR (dédoubler les sommets assure notamment que deux de ces mots ne peuvent être égaux), on construit l'automate de Aho-Corasick de cet ensemble de mots en le complétant, avec toutefois la restriction que pour chaque état u, si la dernière lettre de $m(u)$ est un entier (resp. une lettre de Σ), on ne complète les transitions issues de u que par les lettres de Σ (resp. les entiers) nécessaires. Le fait d'avoir dédoublé les sommets assure que l'automate ainsi obtenu est déterministe. Finalement, dans une dernière phase de prétraitement, on associe à tout état final u la longueur du mot $m(u)$ reconnu en ce sommet et une transition non étiquetée vers l'état final v reconnaissant le plus long suffixe propre de $m(u)$ (vers l'état initial si aucun suffixe propre de $m(u)$ n'est reconnu par l'automate) et on note $s(u)$ cette transition. Dans la figure suivante, on a représenté les états finaux par des carrés (l'entier situé à l'intérieur indiquant la longueur du mot reconnu), les transitions ajoutées à l'ADR associée à M par des pointillés longs et les transitions $s(u)$ par des pointillés courts. On note $AC(M)$ l'automate ainsi obtenu.

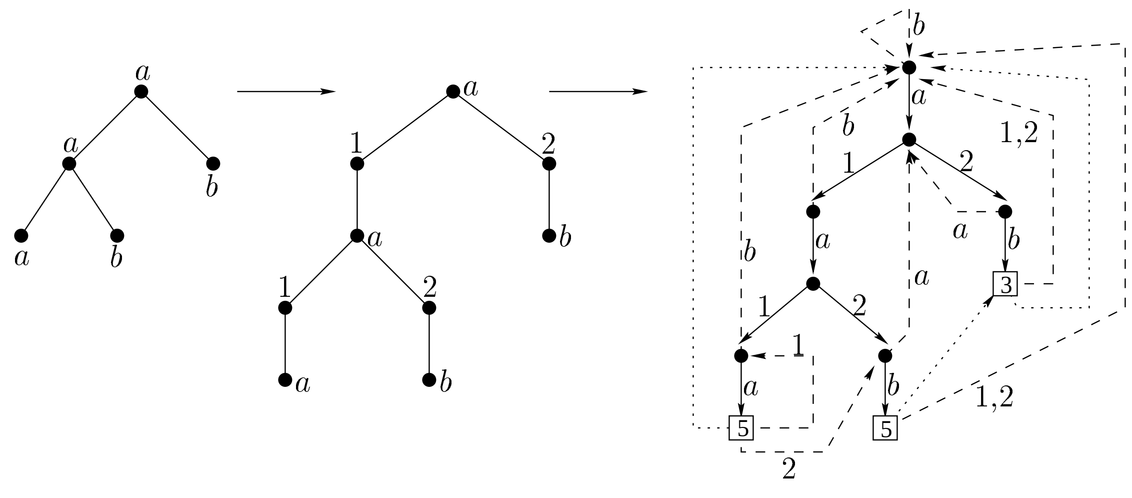


FIG. 7.8 – Une arborescence et l'automate correspondant

La construction de l'ADR se fait en temps $O(m)$, puisqu'elle repose uniquement sur une modification de M. Cette ADR comportant $2m$ sommets on déduit de la remarque 177 que le calcul des arcs complétant cet automate se fait en temps $O(m \times |\Sigma|)$. Finalement le calcul des liens $s(u)$ s'effectue en temps $O(m)$ et le prétraitement nécessite donc un espace $O(m \times |\Sigma|)$ et un temps $O(m \times |\Sigma|)$.

La phase de recherche de motif consiste, après avoir dédoublé les sommets du texte T de la même manière que pour M, à "lire" l'arborescence T dans l'automate $AC(M)$ (tout comme on lisait un texte T dans l'automate $AC(M)$). On associe cependant à tout sommet de T un compteur (initialisé à 0) enregistrant le nombre de chemins issus de ce sommet et correspondant à un chemin racine-feuille du motif. On a ainsi une occurrence de M en x si et seulement si à la fin de la phase de recherche de motifs, le compteur associé à x est égal au nombre de feuilles de M. On lit T dans $AC(M)$ en effectuant un parcours en profondeur de T (dont on a dédoublé les sommets) et en suivant, à chaque sommet x visité, la transition correspondant à son étiquette. On maintient de plus une pile contenant les couples (y,u) où y est un ancêtre de x et u l'état de l'automate atteint après avoir visité y (on utilise un tableau pour stocker cette pile, notée P). La principale opération concerne les états finaux de $AC(M)$: si en lisant le sommet x de T, on arrive sur un état final u de $AC(M)$, pour tous les sommets finaux v de $AC(M)$ reconnaissant un suffixe (non forcément propre) de $m(u)$ (on accède à ces sommets par les transitions s), si $m(v)$ est de longueur k, on incrémente le compteur associé au $k^{ème}$ ancêtre de x (on accède à ce sommet en temps $O(1)$ à l'aide de la pile P). On appelle cet algorithme **HOD**. Hoffmann et O'Donnell ont montré que l'algorithme **HOD** détecte les occurrences compactes de M dans T en temps $O(n \times l(M))$, où $l(M)$ est la taille du plus grand ensemble de feuilles $\{f_1,\dots,f_k\}$ de M telles que le mot porté par le chemin menant de la racine à f_i $i < n$) est un suffixe du mot associé à la feuille f_{i+1}. En remarquant qu'il existe des motifs à m sommets tels que $l(M)$ est en $O(m)$, on constate que cet algorithme n'améliore en fait pas la borne quadratique en $O(n \times m)$ dans le pire des cas de l'algorithme naïf.

Le fait que **HOD** soit limité aux occurrences compactes de M provient de la façon dont on dédouble les sommets de M lors du prétraitement, qui fait intervenir l'ordre des fils d'un sommet de manière trop stricte. Dans la section suivante, nous proposons une adaptation de cet algorithme au cas des occurrences quelconques du motif.

Les algorithmes de Kosaraju et Dubiner-Galil-Magen. La première amélioration de la borne quadratique pour la recherche de motifs dans une arborescence est due à Kosaraju [105], qui propose un algorithme en temps $O(n \times m^{3/4} \times \log(m))$ pour le problème RMCA. En reprenant le même principe que l'algorithme de Kosaraju, Dubiner, Galil et Magen [54] (voir aussi la présentation de cet algorithme par Crochemore et Rytter [52, section 15.2]) ont amélioré cette borne et obtenu un algorithme en temps $O(n \times \sqrt{m} \times \log(m))$.

Ces deux algorithmes nécessitent que l'étiquette de chaque sommet (du texte et du motif) contienne l'information "ordre parmi ses frères" : si le sommet x, étiqueté a, est le $i^{ème}$ fils de son père, on transforme l'étiquette de x en un couple (a,i). Si M est un motif à k feuilles $f_1,\dots,f_k$, on note $m(f_i)$ le mot correspondant au chemin allant de la racine de M à f_i. On a ainsi une occurrence de M en un sommet x de T, si et seulement si il existe k descendants $x_1,\dots,x_k$ de x tels que le mot allant de x à x_i est $m(f_i)$.

Pour obtenir une borne en $O(n \times q \times \log(m))$ ($q = m^{3/4}$ chez Kosaraju [105] et $q = \sqrt{m}$ chez Dubiner et al [54]), ces auteurs proposent des algorithmes opérant en trois étapes. La première consiste à calculer un ensemble d'arborescences $M_1,\dots,M_l$ obtenues en restreignant M aux sommets présents sur les chemins allant de la racine à un sous-ensemble des feuilles $f_1,\dots,f_k$ de façon à ce que les M_i vérifient des propriétés permettant de détecter les occurrences des M_i dans T en temps $O(n \times q)$ en utilisant un algorithme simple (l'algorithme naïf si $h(M_i)$ est faible, ou l'algorithme **HOD** si $l(M_i)$ est faible, où $l(M_i)$ est le paramètre défini dans la description de l'algorithme **HOD**). Si on note M' l'arborescence obtenue à partir de M en supprimant les sommets dont tous les descendants appartiennent aux M'_i, la deuxième étape consiste à rechercher les occurrences de M' dans T. Il s'agit là de l'étape difficile de ces deux algorithmes, nécessitant de faire appel à un algorithme de recherche de motifs dans un mot avec symboles "don't care" et, dans le cas de l'algorithme de Dubiner et al, à des propriétés de périodicité des chemins racine-feuilles de M' (intuitivement, il est naturel de jouer sur cette information car les chemins de faible longueur ont été traités avec les M'_i). Finalement,comme l'information "ordre parmi ses frères" a été ajoutée aux étiquettes des sommets, on peut dire qu'on a une occurrence de M en un sommet x de T si et seulement si on a une occurrence de chacune des arborescences M_i et M' en x.

On peut noter que pour calculer l'ensemble des arborescences M_i, Kosaraju introduit une notion d'arborescence des suffixes d'une arborescence dans laquelle un suffixe d'une arborescence M est le mot formé des étiquettes du chemin reliant la racine de M à l'une de ses feuilles. Kosaraju calcule cette arborescence des suffixes en temps $O(m \times \log(m))$, en utilisant la version séquentielle de l'algorithme parallèle mis au point dans [8] (la construction de cette arborescence des suffixes a ensuite été améliorée par Breslauer [27] et Shibuya [141]).

Comme dans le cas de l'algorithme **HOD**, il ne semble pas possible de pouvoir étendre ces algorithmes au problème RMA du fait de la présence nécessaire de l'information "ordre" dans l'étiquette des sommets de M et T.

Les algorithmes de Cole, Hariharan et Indyk. Dans [46], Cole et Hariharan empruntent une nouvelle voie, en proposant un algorithme probabiliste permettant de résoudre le problème RMCA en temps $O(n \times \log^3(m))$. Dans un premier temps, Cole et Hariharan introduisent le problème de la recherche de motifs dans les *arborescences binaires vertébrées*, c'est-à-dire les arborescences binaires dans lesquelles un chemin reliant la racine à un sommet ayant au plus un fils est distingué, ce chemin étant appelé la colonne vertébrale de l'arborescence. Ainsi, pour ces arborescences binaires vertebrées, l'existence d'une occurrence de M en un sommet x de T, nécessite que la colonne vertébrale de M corresponde à une partie de la colonne vertébrale de T (x doit donc appartenir à cette colonne). Les auteurs montrent ensuite que l'on peut réduire le problème RMCA à un ensemble d'instances du problème de la recherche de motifs dans des arborescences binaires vertébrées de tailles cumulées $O(n + m)$, cette réduction s'effectuant en temps $O(n + m)$. Cole et Hariharan introduisent ensuite une généralisation du problème de la recherche de motifs dans des mots, appelé problème du "subset matching" que l'on peut définir comme suit. Soit $P(\Sigma)$ l'alphabet constitué de tous les sous-ensembles de Σ, alphabet sur lequel on définit la notion de mot généralisé. Par exemple, si $\Sigma = \{a,b,c,d\}$, le mot $M = (a,c)()(a,c)()(b)$ est un mot généralisé sur Σ. Ainsi, si M et T sont deux mots généralisés sur Σ, tels que M est constitué de m sous-ensembles de Σ (M est de longueur m), on a une occurrence de M en $T[i]$ si et seulement si, pour $1 \leq j \leq m$, $M[j] \subseteq T[i+j-1]$. Par exemple, si $T = (c,d)(b,d)(a,b,c)()(a,c)()(b,c)()(a,b,c)(a,d)(a,d)(b)$, on a une seule occurrence de M en T, située en position 3. Cole et Hariharan montrent alors que la recherche de motifs dans des arborescences binaires vertébrées se réduit en temps linéaire au problème précédent. Finalement, ils proposent deux algorithmes pour résoudre le problème "subset matching" : un algorithme déterministe en temps $O(n \times \sqrt{m} \times \log(m))$ et un algorithme non déterministe en temps $O(n \times \log^3(m))$. En utilisant des résultats algorithmiques sur le codes superimposés [94], Cole, Hariharan et Indyk [47] ont ensuite proposé un algorithme déterministe permettant de résoudre le problème "subset matching", et donc le problème de la recherche de motifs dans une arborescence, en temps $O(n \times \log^3(n))$.

On peut noter que la notion de colonne vertébrale introduite dans cette approche impose que chaque sommet de la colonne vertébrale d'une arborescence a au plus un fils n'appartenant pas à cette colonne vertébrale, cette condition étant nécessaire lors la phase de recherche de motifs. Cet algorithme semble donc limité aux arborescences binaires et au problème RMCA.

Coder une arborescence par des mots. Dans une série de trois articles [112, 113, 114] (voir aussi les travaux de Grossi, Luccio et Pagli [85]), Luccio et Pagli introduisent une notion de distance entre deux arborescences A_1 et A_2, correspondant au nombre d'arborescences à insérer ou supprimer dans A_1 pour obtenir A_2. Ils proposent un algorithme simple, qui, étant donnés deux arborescences T et M et un entier k, détecte tous les sommets x de T pour lesquels T_x est à distance au plus k de M[2]. Cet algorithme se compose de trois étapes.

La première étape consiste à coder T (resp. M) par trois mots[3] S_T, F_T et P_T (resp. S_M, F_M et P_M), chacun de taille $O(n)$ (resp. $O(m)$), puis à concaténer les deux mots S_T et S_M en un seul mot S, et à calculer l'arborescence des suffixes $AS(S)$ de ce mot.

La deuxième étape consiste à prétraiter $AS(S)$ de sorte qu'étant donnés deux sommets u et v de $AS(S)$, on puisse calculer le plus proche ancêtre commun de u et v en temps $O(1)$ (et donc le plus long préfixe commun de $m(u)$ et $m(v)$). Harel et Tarjan [90] ont montré que ce prétraitement peut être effectué en temps et espace linéaire, soit en $O(n+m)$ dans notre cas.

Remarque 184. Ce résultat, désormais classique, est souvent utilisé pour la mise au point d'algorithmes sur les mots utilisant l'arborescence des suffixes (voir par exemple [86, partie II]). Il est dû à Harel et Tarjan [90], et a été ensuite amélioré par Schieber et Vishkin [137] et Berkman et Vishkin [17]. Récemment, Bender et Farach-Colton [13] ont proposé une nouvelle technique dont la mise en œuvre est plus aisée.

Finalement, la phase de recherche de motifs proprement dite consiste à lire n fois en parallèle un suffixe de S_T et S_M en utilisant $AS(S)$, F_T, P_T, F_M et P_M pour éviter de parcourir des facteurs identiques, ce qui permet d'effectuer chacun de ces parcours en temps $O(k)$. Cette phase a une complexité en temps en $O(k \times n)$.

On peut noter que l'idée du codage d'une arborescence par des mots a été utilisée notamment par Ramesh et Ramakrishnan [130] pour le problème de la recherche de motifs dans des termes, ou par Mäkinen [115] (voir aussi les articles de Dublish [55], Grossi [84, 83] et Verma [161]).

Quelques commentaires. Pour conclure ce rapide tour d'horizon des travaux abordant le problème de la recherche de motifs dans une arborescence, on peut souligner trois points essentiels.

Premièrement, il n'existe pas, à notre connaissance, d'algorithme traitant le problème RMA dans le cas des arborescences générales. Tous les algorithmes proposés portent soit sur les termes, soit sur le problème RMCA. Dans la section suivante, nous proposons une adaptation de l'algorithme descendant de Hoffmann et O'Donnell pour le problème RMA.

---

2. Cette notion de distance est une simplification de la notion de distance entre deux arborescences introduite par Tai [152] et Zhang et Shasha [167].

3. Nous décrivons ce codage dans le chapitre suivant, dans lequel nous présentons un algorithme utilisant cette approche.

Deuxièmement, on remarque que même le problème RMCA a reçu peu d'attention (comme nous venons de le voir, les travaux sur ce sujet sont peu nombreux), ce qui s'explique sans doute par sa difficulté. Ainsi les techniques employées pour obtenir un algorithme en temps quasi-linéaire (le seul algorithme ayant une complexité de cet ordre est celui de Cole, Hariharan et Indyk) sont complexes et lourdes à implanter. C'est pourquoi, bien qu'aucune étude expérimentale sur ce sujet n'ait été réalisée, on peut penser que ces algorithmes sont peu compétitifs, relativement au temps d'exécution moyen, en comparaison avec l'algorithme naïf qui se caractérise par sa simplicité et son bon comportement en moyenne (les résultats expérimentaux que nous fournissons dans la section suivante semblent indiquer un comportement linéaire en moyenne). C'est ainsi que dans la plupart des applications nécessitant d'effectuer une recherche de motifs dans une arborescence, l'algorithme implanté est d'ailleurs l'algorithme naïf (à plus forte raison lorsque le problème traité est le problème RMA et non le problème RMCA).

Finalement, on peut remarquer qu'aucun des algorithmes proposés dans la littérature ne semble adapté au cas statique. En effet, comme nous l'avons vu dans la deuxième section de ce chapitre, on peut, dans le cas des mots, prétraiter “une fois pour toute” le texte (en construisant son arborescence des suffixes, ou son tableau des suffixes qui possède de meilleures propriétés pratiques), en temps et espace linéaires, et effectuer ainsi une recherche du motif en temps quasi-linéaire. Cette technique est particulièrement utile pour rechercher rapidement un ou des motifs dans une banque de données de textes statiques (comme des séquences biologiques par exemple). Dans le chapitre suivant, nous proposons une adaptation de l'algorithme naïf basée sur le codage d'une arborescence par des mots et l'utilisation de l'arborescence des suffixes, qui si elle ne modifie pas la complexité quadratique de l'algorithme naïf, semble posséder de bonnes propriétés pratiques dans le cas des termes et du problème RMCA.

7.4 Adaptation de l'algorithme HOD

Dans cette section, nous décrivons une modification de l'algorithme **HOD** (qui traite le problème RMCA) afin de résoudre le problème RMA. Cette modification, bien que très simple, ne semble figurer dans aucun article antérieur sur le sujet. Nous commençons par rappeler la définition du *parcours en profondeur préfixe* d'une arborescence.

Définition 185. Le parcours en profondeur préfixe d'une arborescence A est le parcours consistant à visiter la racine de A, puis à parcourir récursivement chacun des sous-arborescences issues de cette racine dans l'ordre où ils apparaissent. Lors de ce parcours, on associe un indice à chaque sommet lors de ce parcours (indice i pour le $i^{ème}$ sommet visité).

On suppose ici que tout sommet x du texte T connaît la taille $|T_x|$ de la sous-arborescence dont il est racine et l'indice qui lui est associé lors d'un parcours en profondeur préfixe (on note cette information $ind(x)$). Ce prétraitement du texte s'effectue en temps et espace $O(n)$.

Dans un premier temps, on associe à toute feuille x du motif les informations suivantes (voir figure 7.9) :

- si x est la $i^{ème}$ feuille visitée lors d'un parcours en profondeur préfixe de M, on note $num(x) = i$ (on dit que x est la $i^{ème}$ feuille de M) ;
- on note $l1(x)$ le nombre d'arêtes du chemin allant de x à la racine de M ;

– si $num(x) = i$ et $i > 1$ (x n'est pas la première feuille), on note $l2(x)$ le nombre d'arêtes du chemin allant de x au plus proche ancêtre commun de x et de la $(i-1)^{ème}$ feuille de M (on a donc $l2(x) > 0$) ; sinon, $l2(x) = 0$, pour la première feuille de x ($num(x) = 1$).

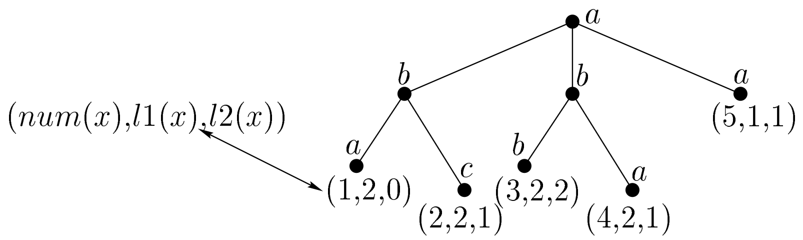


FIG. 7.9 – Informations associées aux feuilles du motif

Notation. Soient A une arborescence et x, y deux sommets de A tels que y est un descendant de x. On note $C_A(x,y)$ ($C_A(y)$ si x est la racine de A) le mot formé des étiquettes des sommets du chemin allant de x à y, les sommets x et y inclus (si x est le $k^{ème}$ ancêtre de y, ce mot comporte donc $k+1$ lettres).

On construit ensuite l'automate de Aho-Corasick de l'ensemble des mots $C_M(f_i)$ associés à chaque feuille f_i de M (f_i désignant la $i^{ème}$ feuille de M). On associe de plus à tout état final u de cet automate (représenté par un carré dans la figure suivante) l'ensemble des triplets $(i,l1(f_i),l2(f_i))$ tels que le mot correspondant au chemin allant de la racine de M à la feuille f_i est égal à $m(u)$. Finalement, on ajoute les transitions $s(u)$ de la même manière que pour l'automate AC calculé par l'algorithme **HOD**, et on obtient ainsi un automate que l'on note aussi $AC(M)$. Dans la figure 7.10, on n'a pas représentée les transitions $s(u)$, celles-ci reliant chaque état final à l'état initial (pour cet exemple).

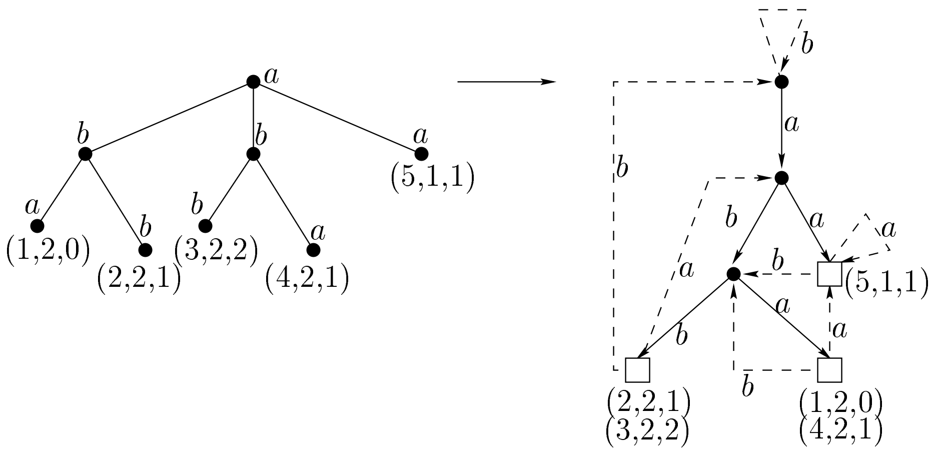


FIG. 7.10 – Un motif M et l'automate $AC(M)$ correspondant

Définition 186. Soit M une arborescence ayant k feuilles. On note M_i l'arborescence associée à la feuille f_i et obtenue en supprimant de M tous les sommets n'ayant, parmi leurs descendants, aucune feuille f_j telle que $j \leq i$. On appelle M_i le $i^{ème}$ préfixe de M car cette arborescence correspond à la suppression dans M de tous les sommets ayant un indice supérieur à celui de la feuille f_i.

On peut remarquer que, pour un sommet x de T, on détecte une occurrence de M_1 en x si et seulement si il existe un descendant y de x tel que $C_T(x,y) = C_M(f_1)$. Supposons maintenant que l'on ait trouvé une occurrence de M_i en un sommet x de T (on note z le sommet de T correspondant à f_i), et qu'il existe un descendant y de x tel que $C_T(x,y) = C_M(f_{i+1})$. On a alors une occurrence de M_{i+1} en x telle que y corresponde à la feuille f_{i+1} si et seulement si les conditions suivantes sont satisfaites (voir figure 7.11) : si $k = l2(f_{i+1})$, si y_1 est le $k^{ème}$ ancêtre de y et y_2 le $(k-1)^{ème}$ ancêtre de y (y_1 est le père de y_2) alors z appartient à la sous-arborescence T_{y_1} et le chemin allant de y_1 à z passe par un frère gauche de y_2, ce qui se traduit formellement par

$$ind(y_1) < ind(z) < ind(y_1) + |T_{y_1}| - 1 \text{ et } ind(y_2) > ind(z). \tag{7.1}$$

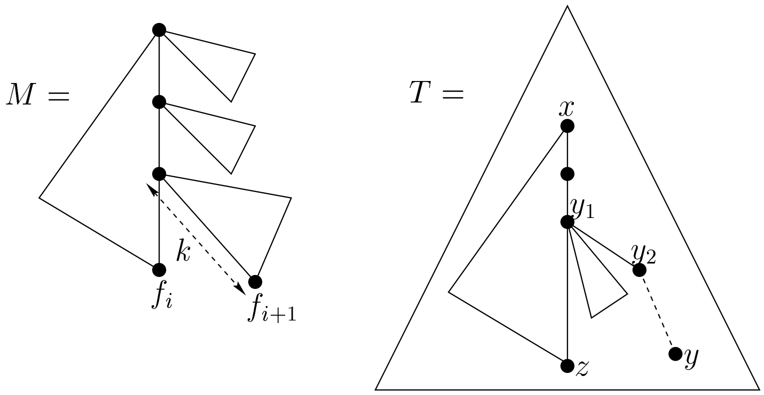


FIG. 7.11 – $C_T(x,y) = C_M(f_{i+1})$

Le principe de la phase de recherche de motifs consiste à parcourir en parallèle l'arborescence T (par un parcours en profondeur préfixe) et l'automate $AC(M)$ de sorte qu'après avoir visité un sommet s de T, on se trouve dans un état t de l'automate $AC(M)$ correspondant à la lecture du mot $C_T(s)$. Durant ce parcours , on maintient une pile P (gérée à l'aide d'un tableau) telle qu'à tout instant, le sommet de pile contienne le quadruplet (s,j,k,t) tel que

- s est le sommet courant de T (le dernier sommet visité lors du parcours en profondeur préfixe de T),
- j est l'indice du plus grand préfixe de M (définition 186) reconnu en s,
- k est l'indice du sommet z de T_s correspondant à la $j^{ème}$ feuille de M ($ind(z) = k$),
- t est l'état de $AC(M)$ atteint après avoir visité le sommet s,

et on maintient dans ce tableau (cette pile) les informations correspondant aux ancêtres du dernier sommet s visité (le quadruplet correspondant à s est le dernier élément du tableau alors que la quadruplet correspondant à la racine de T en est le premier élément). On en déduit que P contient à tout instant au plus $h(T)$ quadruplets (on rappelle que $h(T)$ est la hauteur de T). Finalement, pour détecter les occurrences de M dans T, on maintient les informations données par les quadruplets de P de la manière suivante : si, après avoir visité le sommet s on arrive sur un état final t de $AC(M)$, pour chaque triplet associé à un état final de l'automate reconnaissant un suffixe de longueur k de $m(t)$ (on parcourt ces états finaux grâce aux transitions $s(t)$ de l'automate), on modifie le quadruplet de P associé au $k^{ème}$ ancêtre de s lorsque la condition (7.1) est vérifiée. On obtient ainsi l'algorithme suivant, que nous appelons **HOD2**.

Algorithme 13: HOD2 : recherche du motif M dans T avec l'automate $AC(M)$.

Données : $AC(M)$ et T
début
  soient $init$ l'état initial de $AC(M)$ et r la racine de T;
  $top := 1$, $t := \delta(init,e(x))$ et $P[top] := (r,0,0,t)$;
  tant que $top > 0$ **faire**
    $(s,j,k,t) := P[top]$;
    si tous les fils de s ont été visités **alors**
      $top := top - 1$;
      si j est égal au nombre de feuilles de M **alors** afficher "Occurrence de M en s";
    sinon
      soit y le premier fils de s non visité;
      $top := top + 1$, $t := \delta(t,e(y))$ et $P[top] := (y,0,0,t)$;
      si t est un état final **alors**
        tant que $t \neq init$ **faire**
          pour chaque triplet (i,l_1,k) associé à l'état t **faire**
1           $(x,j_1,k_1,t_1) := P[top - l_1]$;
            $(y_1,j_2,k_2,t_2) := P[top - k]$;
            $(y_2,j_3,k_3,t_3) := P[top - k + 1]$;
            si $(j_1 = i - 1)$ et $((k_1 = 0)$ ou $(ind(y_1) < k_1 < ind(y_1) + |T_{y_1}| - 1$ et $ind(y_2) > k_1))$ **alors**
2             $P[top - l_1] := (x,j_1 + 1,ind(y),t_1)$;
          $t := s(t)$;
fin

La complexité du prétraitement est identique à celle du prétraitement de l'algorithme descendant de Hoffmann et O'Donnell: la transformation de M en ADR est réalisée en temps $O(m)$, la complétion de cette ADR pour obtenir un automate complet s'effectue en temps $O(m \times |\Sigma|)$ et le calcul des transitions $s(t)$ se fait en temps $O(m)$. Lors de la phase de recherche du motif, le nombre d'opérations effectuées lorsque l'on visite un sommet x de T est proportionnel au nombre d'exécutions du bloc situé entre les lignes marquées 1 et 2, qui vaut au plus $l(M)$. On obtient alors le résultat suivant.

Proposition 187. *Soient T et M deux arborescences de tailles respectives n et m. L'algorithme* **HOD2** *détecte toutes les occurrences de M dans T en temps $O(n \times l(M))$.*

Nous donnons maintenant quelques résultats expérimentaux qui laissent penser que cet algorithme est malheureusement moins efficace que l'algorithme naïf. Nous avons mis en œuvre ces deux algorithmes en C et nous avons procédé de la manière suivante pour réaliser des tests. Nous avons engendré aléatoirement et uniformément une arborescence T sur Σ [2], puis une centaine d'arborescences M sur Σ de taille comprise entre 5 sommets et 20 sommets, et enfin, pour chacune d'entre elles, nous avons recherché les occurrences de M dans T en utilisant l'algorithme naïf, puis l'algorithme **HOD2**. Les statistiques suivantes ont été mesurées : nombre $Comp1$ de comparaisons entre les étiquettes d'un sommet de T et d'un sommet de M effectuées par l'algorithme naïf, nombre $Comp2$ d'exécutions du bloc situé entre les lignes marquées 1 et 2 de **HOD2**, temps t_1 pris par l'algorithme naïf et temps t_2 pris par l'algorithme **HOD2**. Nous obtenons les résultats suivants, synthétisés par le tableau suivant. On peut de plus ajouter qu'en nous limitant à des motifs ayant peu de feuilles, nous avons obtenu des résultats semblables.

$\|T\|$	$\|\Sigma\|$	$Comp2/Comp1$	t_2/t_1
100	1	2,73	2,68
100	3	2,52	3,04
100	5	3,60	3,71
100	7	4,79	3,84
1000	1	2,15	2,93
1000	3	2,05	3,85
1000	5	2.97	4,57
1000	7	3,99	4,82
10000	1	1.97	4,03
10000	3	1,92	4,79
10000	5	2,80	4,88
10000	7	3,76	5,52

TAB. 7.1 – Comparaison de l'algorithme naïf et de l'algorithme **HOD2**

Chapitre 8

Coder une arborescence par des mots

Comme nous l'avons mentionné dans le chapitre précédent, en conclusion de la revue des principaux travaux traitant de problème de la recherche de motifs dans une arborescence, aucun des algorithmes proposés n'est dédié au cas statique (le texte T est fixé). Dans ce chapitre, nous nous intéressons à ce problème et nous proposons une adaptation de l'algorithme naïf basée sur le codage d'une arborescence par des mots et l'utilisation de l'arborescence des suffixes. Si la complexité en temps (toujours dans le pire des cas) de l'algorithme que nous proposons en $O(n \times m)$, les résultats expérimentaux que nous obtenons, dans le cas de termes binaires ou du problème RMCA, semblent indiquer des performances en moyenne intéressantes (d'un point de vue pratique et théorique).

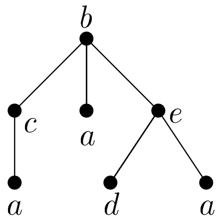
Dans la première section nous nous intéressons au cas des termes. Nous poursuivons en décrivant un algorithme pour le problème RMCA dans le cas des arborescences générales basé sur le même principe mais utilisant un codage d'une arborescence par des mots légèrement différent. Nous concluons en exposant quelques résultats expérimentaux.

Remarque 188. On peut noter que le codage d'une arborescence par des mots que nous utilisons dans les deux prochaines sections ci-dessous provient des travaux de Luccio et Pagli [112, 113, 114]. Ces auteurs l'utilisent pour résoudre certains problèmes de recherche de motifs dans une arborescence légèrement différents des problèmes que nous considérons dans ce mémoire.

8.1 Le cas des termes

Définition 189. Soit A un terme de taille n sur l'alphabet de symboles Σ. On note S_A et L_A les mots de longueur n sur les alphabets respectifs Σ et $[n]$ définis de la manière suivante : si x est le $i^{ème}$ sommet visité lors du parcours en profondeur préfixe de A, alors $S_A[i] = e(x)$ et $L_A[i] = i + |T_x|$.

On remarque que si x est le $i^{ème}$ sommet visité lors du parcours en profondeur de A, $L_A[i]$ donne la position dans S_A du symbole associé au sommet rencontré immédiatement après que toute la sous-arborescence de racine x ait été visitée.



$S_A = b.c.a.a.e.d.a$
$L_A = 8.4.4.5.8.7.8$

FIG. 8.1 – Codage d'un terme par 2 mots

Observation 190. Soit x un sommet de T et i l'indice du symbole $e(x)$ de S_T associé à x. On a une occurrence de M en x si et seulement si il existe m entiers $i_1 = i < i_2 < \cdots < i_m \leq n$ tels que, pour $1 \leq j \leq n$,

- si $S_M[j] \neq v$ alors $S_T[i_j] = S_M[j]$,
- si $S_M[j] \neq v$ alors $i_{j+1} = i_j + 1$, sinon $i_{j+1} = L_T[i_j]$.

On en déduit la version suivante de l'algorithme naïf. Comme nous nous plaçons dans le cas statique, nous supposons que le texte T a été prétraité de sorte que pour chaque symbole de Σ, on connaît ses occurrences dans S_T. On rappelle qu'on note n la taille du texte T et m la taille du motif M.

Algorithme 14: Algorithme naïf amélioré (pour les termes)

Données : S_T, L_T, S_M
début
  pour chaque $1 \leq i \leq n$ tel que $S_T[i] = S_M[1]$ **faire**
    $j := 2$ et $k := i + 1$;
    tant que $j \leq m$ **faire**
      si $S_T[k] = S_M[j]$ **alors**
        $j := j + 1$ et $k := k + 1$;
      sinon si $S_M[j] = v$ **alors**
        $j := j + 1$ et $k := L_T[k]$;
      sinon sortir de la boucle **tant que**;
    si $j = m + 1$ **alors**
      afficher "Occurrence de M en i";
fin

On peut cependant remarquer que si il existe i_1 et i_2 tels que $S_T[i_1,n]$ et $S_T[i_2,n]$ ont un préfixe commun égal à un préfixe de S_M, ces deux facteurs identiques de S_T seront lus par cet algorithme. Le principe de l'amélioration que nous proposons consiste à éviter cette lecture multiple de certains facteurs identiques en utilisant l'arborescence des suffixes de S_T. Notre algorithme repose sur la définition de la reconnaissance d'un préfixe de S_M dans $AS(S_T)$, que nous énonçons ci-dessous avant de l'illustrer par la figure 8.2.

Définition 191. Soient (d,f) l'étiquette d'une arête de $AS(S_T)$ allant d'un sommet u vers un sommet w (à cette arête correspond le facteur $S_T[d,f]$) et l un entier tel que

$0 \le l \le f - d$. On dit que $((d,f),l)$ reconnaît le $k^{ème}$ préfixe de S_M si il existe k indices $(d - |m(u)|) = i_1 < i_2 < \cdots < i_k = (d + l)$ tels que, pour $1 \le j \le k$,

- si $S_M[j] \neq v$ alors $S_T[i_j] = S_M[j]$,
- si $S_M[j] \neq v$ alors $i_{j+1} = i_j + 1$, sinon $i_{j+1} = L_T[i_j]$.

Dans la figure suivante, on a représenté de manière abstraite les symboles de Σ par de petits carrés blancs.

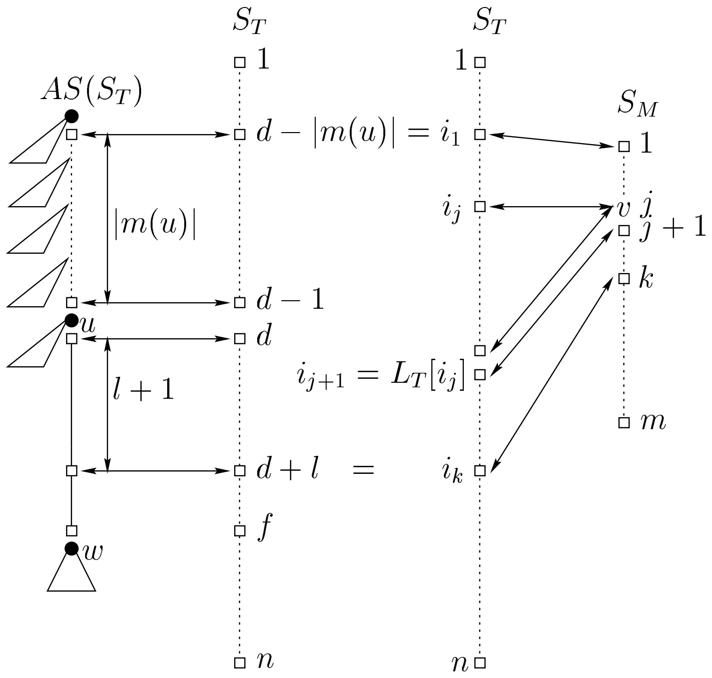


FIG. 8.2 – Illustration de la définition 191

Le principe de notre algorithme est de lire S_M dans $AS(S_T)$, la seule difficulté étant de gérer efficacement le cas où $S_M[j] = v$ (on dit alors qu'on effectue un *saut*). Pour cela on s'appuie sur l'observation suivante, que l'on illustre par la figure 8.3.

Observation 192. Soient $k < m$ tel que $S_M[k+1] = v$ et $((d,f),l)$ un couple reconnaissant le $k^{ème}$ préfixe de S_M (au sens de la définition 191). On suppose que (d,f) est l'étiquette d'une arête de $AS(S_T)$ allant d'un sommet u à un sommet w. Soient (d',f') l'étiquette d'une arête de $AS(S_T)$ allant d'un sommet u' descendant de u à un sommet w' descendant de w, h la longueur du mot étiquetant le chemin allant de $((d,f),l)$ à u' ($h = 0$ si $u' = u$ et $l = 0$, et $h = |m(u')| - |m(u)| - l$ sinon) et l' un entier tel que $0 \le l' \le f' - d'$. $((d',f'),l')$ reconnaît le $(k+1)^{ème}$ préfixe de S_M si et seulement si

$$L_T[d' - h + 1] = d' + l' + 1.$$

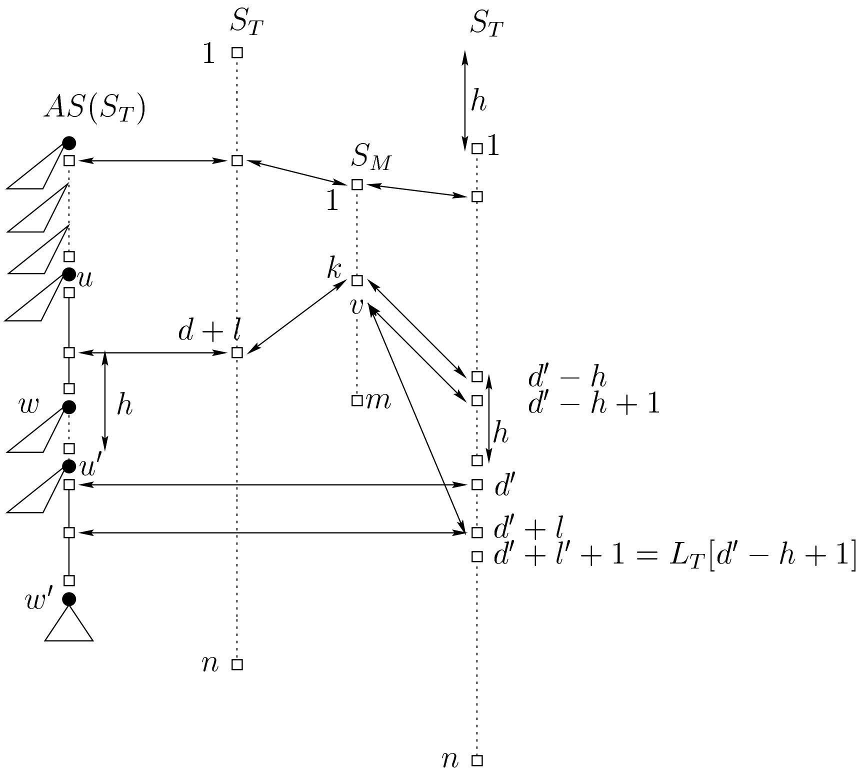


FIG. 8.3 – Illustration de l'observation 192

On définit alors deux procédures pour parcourir $AS(S_T)$. La première, appelée **Lecture**, est une modification de la procédure de recherche d'un mot dans une arborescence des suffixes, tant que le caractère courant de S_M est différent de v et fait appel à une procédure récursive **Sauts** dès que le caractère courant de S_M est la variable v (on compare $S_M[j] = v$ avec le $(l+1)^{ème}$ caractère de l'arc (d,f)). La procédure **Sauts** cherche alors tous les couples $((d',f'),l')$ reconnaissant le $(j+1)^{ème}$ préfixe de S_M (on dit alors qu'on a effectué un *saut* de $((d,f),l)$ vers $((d',f'),l')$). Dans les procédures décrites ci-dessous, on suppose que l'arête (d,f) (resp. (d',f')) va du sommet u (resp. u') vers le sommet w (resp. w').

Algorithme 15: Recherche de motifs accélérée (pour les termes)

Données : S_T, L_T, S_M et $AS(S_T)$
début
  soit r la racine de $AS(S_T)$;
  si une arête (d,f) telle que $S_T[d] = S_M[1]$ quitte r **alors**
    Lecture$(1,\ ((d,f),0))$;
fin

Algorithme 16: Lecture (pour les termes)

Données : j, $((d,f),l)$
début
    $k := j$;
1   **tant que** $k \leq m$, $l \leq (f-d)$ et $S_T[d+l] = S_M[k]$ **faire**
        $k := k+1$ et $l := l+1$;
    si $k > m$ **alors**
2   parcourir la sous-arborescence de $AS(S_T)$ de racine w, tous les sommets finaux correspondant à une occurrence de M dans T;
    sinon si $l > (f-d)$ **alors**
        pour chaque arête (d',f') quittant w **faire**
            Lecture(k, $((d',f'),0)$);
    sinon si $S_M[k] = v$ **alors**
        Sauts(k, $((d,f),l)$, 0) **sinon** Échec;
fin

Algorithme 17: Sauts (pour les termes)

Données : j, $((d,f),l)$ et h (voir figure 8.3
début
    $g := L_T[d+l-h+1]$;
    si $g \leq f+1$ **alors**
        Lecture($j+1$, $((d,f),g-d)$);
    sinon pour toute arête (d',f') quittant w **faire**
        Sauts(j, $((d',f'),0)$, $h+(f-d+1)-l$);
fin

Proposition 193. *Soient T et M deux termes de tailles respectives n et m. L'algorithme 15 détecte toutes les occurrences de M dans T en temps $O(n \times m)$ dans le pire des cas.*

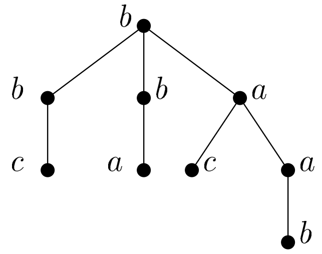
Preuve. La complexité en temps de cet algorithme dépend de trois paramètres : le nombre de comparaisons entre les caractères de S_T et S_M (nombre d'itérations de l'étape marquée 1 de **Lecture**), le nombre d'appels à **Sauts** et le parcours des sous-arborescences de $AS(S_T)$ en cas de succès (étape marquée 2 de **Lecture**). Le nombre de comparaisons de caractères, étant clairement au plus égal à celui de l'algorithme naïf, est en $O(n \times m)$. Le nombre de sommets de $AS(S_T)$ visités lors de l'étape marquée 2 de **Lecture** est en $O(k)$ si on a k occurrences de M dans T, et $k \leq n$. Finalement, en remarquant qu'il ne peut y avoir deux appels à **Sauts** concernant la même arête (d,f) que si on a effectué une comparaison entre un caractère de S_T et un caractère de S_M entre ces deux appels, on a bien $O(n \times m)$ appels à **Sauts**. □

Nous donnons en fin de chapitre des résultats expérimentaux qui semblent indiquer que sur des termes aléatoires, cet algorithme est sensiblement plus rapide que l'algorithme naïf.

8.2 Le cas des arborescences quelconques

On peut bien entendu appliquer le même principe aux arborescences quelconques, mais le codage par des mots doit être modifié. On utilise en effet 3 mots de longueur $2n$ chacun pour coder une arborescence à n sommets.

Définition 194. Soit A une arborescence de taille n sur Σ. On définit les trois mots S_A, L_A et P_A de longueur $2n$ sur l'alphabet $\Sigma \cup \{0\}$ comme suit :
- le mot S_A est obtenu en codant, lors d'un parcours en profondeur préfixe de A, le premier (resp. dernier) passage sur le sommet x par $e(x)$ (resp. 0) : on dit que x est le sommet associé à $S_A[i] = e(x)$;
- si $S_A[i] = 0$ alors $L_A[i] = 0$; sinon, x étant le sommet associé à $S_A[i]$, $L_A[i] = i + 2|A_x|$;
- si $S_A[i] = 0$ ou $i = 1$, alors $P_A[i] = 0$; sinon, x étant le sommet associé à $S_A[i]$ et y le père de x (qu'on suppose associé à $S_A[j]$), on pose $P_A[i] = j$.

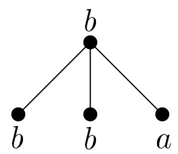


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$S_A =$	b	b	c	0	0	b	a	0	0	a	c	0	a	b	0	0	0	0
$L_A =$	19	6	5	0	0	10	9	0	0	18	13	0	17	16	0	0	0	0
$P_A =$	0	1	2	0	0	1	6	0	0	1	10	0	10	13	0	0	0	0

FIG. 8.4 – Codage d'une arborescence quelconque par des mots

Observation 195. Soient x un sommet de T et i l'indice du symbole $e(x)$ de S_T associé à x. On a une occurrence de M en x si et seulement si il existe $2m$ entiers $i_1 = i < i_2 < \cdots < i_{2m} \leq 2n$ tels que
- pour $1 \leq j \leq 2m$, $S_T[i_j] = S_M[j]$,
- si $j < 2m$ et $S_M[j+1] \neq 0$ alors $i_{j+1} = i_j + 1$
- si $j < 2m$ et $S_M[j+1] = 0$:
  - si $S_T[i_j + 1] = 0$ alors $i_{j+1} = i_j + 1$,
  - sinon, $i_{j+1} = L_T[P_T[i_j + 1]] - 1$.

Par exemple, si T est l'arborescence A de la figure précédente et M est l'arborescence suivante,



alors

$$S_M = b\ b\ 0\ b\ 0\ a\ 0\ 0,$$

et on a une occurrence de M à la racine de T, correspondant aux symboles de S_T entourés de crochets ci-dessous

$$[b]\ [b]\ c\ 0\ [0]\ [b]\ a\ 0\ [0]\ [a]\ c\ 0\ a\ b\ 0\ 0\ [0]\ [0].$$

On remarque que les lettres 0 du motif jouent un rôle analogue à la variable v pour les termes et on peut traduire l'algorithme naïf pour le problème RMCA de la manière suivante.

Algorithme 18: Algorithme naïf amélioré (problème RMCA et arborescences quelconques)

Données : S_T, L_T, P_T et S_M

début
- **pour** chaque $1 \leq i \leq 2n$ tel que $S_T[i] = S_M[1]$ **faire**
  - $j := 2$ et $k := i + 1$;
  - **tant que** $j \leq 2m$ **faire**
    - **si** $S_T[k] = S_M[j]$ **alors**
      - $j := j + 1$ et $k := k + 1$;
    - **sinon si** $S_M[j] = 0$ **alors**
      - $j := j + 1$ et $k := L_T[P_T[k]]$;
    - **sinon** sortir de la boucle **tant que**;
  - **si** $j = 2m + 1$ **alors**
    - afficher "Occurrence en i";

fin

Nous pouvons procéder par analogie au cas des termes en considérant $AS(S_T)$ pour accélérer cet algorithme en évitant la comparaison de préfixes communs. On définit la notion de reconnaissance d'un préfixe de S_M dans $AS(S_T)$ de la manière suivante, conséquence de l'observation 195.

Définition 196. Soient (d,f) une arête de $AS(S_T)$ (allant d'un sommet u vers un sommet w) et l un entier tel que $0 \leq l \leq f - d$. On dit que $((d,f),l)$ reconnaît le $k^{ème}$ préfixe de S_M si il existe k entiers $(d - |m(u)|) = i_1 < i_2 < \cdots < i_k = (d + l)$ tels que :
- pour $1 \leq j \leq k$, $S_T[i_j] = S_M[j]$,
- si $j < k$ et $S_M[j+1] \neq 0$ alors $i_{j+1} = i_j + 1$
- si $j < k$ et $S_M[j+1] = 0$:
  - si $S_T[i_j + 1] = 0$ alors $i_{j+1} = i_j + 1$,
  - sinon, $i_{j+1} = L_T[P_T[i_j + 1]] - 1$.

Observation 197. Soient $k < m$ tel que $S_M[k+1] = 0$ et $((d,f),l)$ un couple reconnaissant le $k^{ème}$ préfixe de S_M. On suppose que (d,f) est l'étiquette d'une arête de $AS(S_T)$ allant d'un sommet u à un sommet w. Soient (d',f') l'étiquette d'une arête de $AS(S_T)$ allant d'un sommet u' descendant de u à un sommet w' descendant de w, h la longueur du mot étiquetant le chemin allant de $((d,f),l)$ à u' ($h = 0$ si $u' = u$ et $l = 0$, et $h =$

$|m(u')| - |m(u)| - l$ sinon) et l' un entier tel que $0 \leq l' \leq f' - d'$. $((d',f'),l')$ reconnaît le $(k+1)^{ème}$ préfixe de S_M si et seulement si

$$d' = d,\ l' = l+1 \text{ et } S_T[d'+l'] = 0$$
$$\text{ou}$$
$$u' = w,\ l' = 0 \text{ et } S_T[d'] = 0$$
$$\text{ou}$$
$$L_T[P_T[d'-h+1]] = d'+l'+1.$$

En s'appuyant sur cette observation, on obtient, de la même façon que pour les termes, les procédures suivantes. On suppose que l'arête (d,f) (resp. (d',f')) va du sommet u (resp. u') vers le sommet w (resp. w').

Algorithme 19: Recherche de motifs accélérée (problème RMCA et arborescences quelconques)

```
début
  Soit r la racine de AS(S_T);
  si une arête (d,f) telle que S_T[d] = S_M[1] quitte r alors
    Lecture(1, ((d,f),0));
fin
```

Algorithme 20: Lecture (problème RMCA et arborescences quelconques)

```
Données : j, ((d,f),l)
début
  k := j;
  tant que k ≤ 2m, l ≤ (f − d) et S_T[d + l] = S_M[k] faire
    k := k + 1 et l := l + 1;
  si k > 2m alors
    parcourir la sous-arborescence de AS(S_T) de racine w, tous les sommets
    finaux correspondant à une occurrence de M dans T;
  sinon
    si l > (f − d) alors
      si il existe une arête (d',f') quittant w telle que S_T[d'] = S_M[k] alors
        Lecture(k + 1, ((d',f'),0));
      sinon si S_M[k] = 0 alors
        pour toute arête (d',f') quittant w faire
          Sauts(j, ((d',f'),0), d' − P_T[d']);
    sinon si S_M[j] = 0 alors
      Sauts(j, ((d,f),l), d + l − P_T[d + l]);
fin
```

Algorithme 21: Sauts(problème RMCA et arborescences quelconques)

Données : j, $((d,f),l)$ et h
début
  $g := L_T[P_T[d+l-h+1]]$;
  si $g \leq f$ **alors**
    Lecture$(j+1,\ ((d,f),g-d))$;
  sinon
    pour toute arête (d',f') quittant y **faire**
      Sauts$(j,\ ((d',f'),0),\ h+(f-d+1)-l)$;
fin

Proposition 198. *Soient T et M deux arborescences de tailles respectives n et m. L'algorithme 19 détecte toutes les occurrences compactes de M dans T en temps $O(n \times m)$ dans le pire des cas.*

Preuve. La preuve de ce résultat est identique à celle de la proposition 193.

8.3 Résultats expérimentaux

Nous avons implanté les algorithmes 14 et 15 (en C sur un PC-Linux) et les avons expérimentés sur des termes binaires. Pour cela nous avons suivi un processus expérimental analogue à celui du chapitre précédent. Nous avons engendré aléatoirement et uniformément un terme binaire T sur Σ, puis une centaine de motifs binaires M sur Σ de taille comprise entre 5 sommets et 20 sommets, et enfin, pour chacun d'entre eux, recherché leurs occurrences dans T en utilisant l'algorithme 14, puis par l'algorithme 15. Nous avons alors réalisé les statistiques suivantes : nombre $Comp1$ de comparaisons entre les étiquettes de deux sommets de T et M effectuées par l'algorithme naïf, nombre $Comp2$ de comparaisons effectuées par l'algorithme 15, nombre $Sauts$ d'appels à la procédure **Sauts**, temps t_1 pris par l'algorithme 14 et temps t_2 pris par l'algorithme 15. Dans la table 8.1, $mult$ désigne le nombre d'étiquettes possibles pour un nœud (resp. feuille) de T et de M (on a donc $|\Sigma| = 2 \times mult$). Nous présentons ces statistiques en comparant $Comp1$ à $Comp2$ et $Sauts$, et t_1 à t_2.

On remarque alors que l'algorithme 14 effectue effectivement beaucoup moins de comparaisons entre caractères de S_T et S_M que l'algorithme 15 (on rappelle que Steyaert et Flajolet ont montré que $Comp1$ est en $O(n)$ [149]), et que le nombre d'appels à la procédure **Sauts** est en $O(n)$. Cela se traduit en pratique par le fait que dès que $mult > 1$, notre algorithme semble être sensiblement plus intéressant, en termes de temps d'exécution, que l'algorithme 14. En remarquant de plus que l'algorithme 15 est très simple à implanter, il nous semble naturel de considérer cet algorithme comme un bon candidat "en pratique" pour traiter le problème de la recherche de motifs dans des termes statiques. Il serait cependant intéressant de vérifier cette affirmation non plus seulement sur des termes binaires aléatoires, mais sur des jeux de test issus de problèmes réels. Il serait également intéressant d'essayer de caractériser le comportement en moyenne des paramètres $Comp2$ (en utilisant des techniques analogues à celles utilisées par Steyaert et Flajolet dans [149]) et $Sauts$ (même si pour cette dernière statistique, les techniques de Steyaert et Flajolet ne semblent pas pouvoir être utilisées).

$\|T\|$	$mult$	$Comp1/Comp2$	$Comp1/Sauts$	t_1/t_2
100	1	13	5,3	0,6
100	2	23,9	13,9	1,2
100	3	24,4	21,5	1,6
100	5	26,4	33,9	2,2
100	7	17,5	44,3	3,4
1000	1	18,1	6,2	0,76
1000	2	60,6	17,7	2,2
1000	3	92,9	27,2	2,4
1000	5	104,1	44,4	3
1000	7	110	62,6	3,3
10000	1	20,9	7,1	0,8
10000	2	90,2	20,9	2,2
10000	3	185,2	32,2	3,1
10000	5	371,4	54,9	4,8
10000	7	483,6	74,1	7,1

TAB. 8.1 – Comparaison de l'algorithme naïf et de l'algorithme utilisant l'arborescence des suffixes - Cas des termes

Dans le cas des arborescences quelconques et du problème RMCA, nous avons obtenu un facteur d'accélération similaire pour l'algorithme utilisant l'arborescence des suffixes par rapport à l'algorithme naïf. On peut naturellement modifier légèrement l'algorithme utilisé dans le cadre du problème RMCA pour traiter le problème RMA. Cependant, les premières expérimentations réalisées sur des arborescences aléatoires semblent indiquer cet algorithme est légèrement plus lent que l'algorithme naïf.

Chapitre 9

Arborescence des suffixes d'une arborescence

Nous avons vu au chapitre précédent une première approche du problème de la recherche de motifs dans une arborescence (texte) statique, basée sur le codage de cette arborescence par un mot et l'utilisation de l'arborescence des suffixes de ce mot. Dans ce chapitre, nous proposons une autre approche de ce problème en introduisant le concept d'arborescence des suffixes d'une arborescence.

Dans la première section, nous définissons cette notion d'arborescence des suffixes d'une arborescence, la structure de données que nous proposons occupant un espace en $O(n)$ pour une arborescence A ayant n sommets. Dans la seconde section, nous rappelons l'algorithme de construction de l'arborescence des suffixes d'un mot dû à McCreight [118], dont nous nous inspirons pour définir un algorithme de construction de l'arborescence des suffixes d'une arborescence telle que nous l'avons définie. L'analyse de la complexité en temps de cet algorithme permet d'affirmer qu'il construit l'arborescence des suffixes d'une arborescence ayant n sommets en temps $O(n^2)$, mais nous conjecturons que cette complexité est en $O(n \times \log(n))$. Cette structure de donnée nous permet ensuite de résoudre certains problèmes de recherche d'un motif M de taille m dans un texte T de taille n en temps $O(m \times \log(n))$.

9.1 La structure de données

Arborescence des suffixes d'une structure abstraite. Dans ce paragraphe, nous adoptons un point de vue général et nous considérons une structure abstraite T appartenant à une famille $\mathcal{F}$ de structures possédant les propriétés suivantes :

- on peut définir une notion de *suffixe* sur les structures de $\mathcal{F}$,
- on peut associer à chaque suffixe T' de T un mot sur un alphabet Γ, noté $S(T')$.

On peut ainsi définir l'arborescence des suffixes $AS(T)$ de T comme étant l'ADR compressée des mots $S(T')$ associés aux suffixes T' de T, ou de manière équivalente par le fait que $AS(T)$ est la seule arborescence vérifiant les propriétés suivantes.

Propriété 199.

1. Chaque arête de $AS(T)$ est étiquetée par un mot sur Γ ; pour un sommet u de $AS(T)$, on note $m(u)$ le mot porté par le chemin allant de la racine de $AS(T)$ à u.
2. Pour tout suffixe T' de T, il existe un et un seul sommet u de $AS(T)$ tel que $m(u) = S(T')$ (on dit que u est le sommet final acceptant T').
3. Soient u et v deux sommets de $AS(T)$, ayant pour plus proche ancêtre commun le sommet w. Le mot $m(w)$ est alors le plus long préfixe commun de $m(u)$ et $m(v)$ (on en déduit que deux arêtes quittant un même nœud ne peuvent commencer par la même lettre de Γ et que tout nœud non final a au moins deux fils).
4. Pour tout sommet u de $AS(T)$, il existe au moins un suffixe T' de T tel que $m(u)$ est préfixe de $S(T')$.

On déduit immédiatement de cette propriété 199 le résultat suivant, qui exprime l'efficacité en espace de cette structure de données (moyennant un codage efficace de ses arêtes).

Proposition 200. *Soit T appartenant à $\mathcal{F}$. Si T comporte n suffixes, l'arborescence des suffixes $AS(T)$ de T comporte au plus $2n$ sommets.*

On qualifie "d'algorithme simple de recherche dans $AS(T)$" l'algorithme recherchant un mot sur Γ dans $AS(T)$ de manière similaire à l'algorithme 10 (page 141). Le résultat suivant montre l'intérêt de cette notion d'arborescence des suffixes pour la recherche de motif dans une structure T statique ou pour la compression de $\mathcal{F}$.

Proposition 201. *Soient T appartenant à $\mathcal{F}$, M un mot de longueur m sur Γ et $AS(T)$ l'arborescence des suffixes de T. L'algorithme simple consistant à "lire" M dans $AS(T)$ permet de déterminer en temps $O(m \times \log(min\{n,|\Gamma|\}))$ si il existe un suffixe T' de T tel que M est un préfixe de $S(T')$.*

Mots, matrices et arborescences. Comme nous l'avons vu lors du chapitre précédent, la notion d'arborescence des suffixes a été introduite à l'origine pour les mots par Weiner [164].

Dans les années 90, Giancarlo et Grossi [66, 67, 68] ont défini la notion d'arborescence des suffixes pour une matrice carrée, afin de détecter toutes les occurrences d'une matrice carrée $m \times m$ dans un texte (matrice carrée $n \times n$) en temps quasi-linéaire [1]relativement à m^2, après avoir construit l'arborescence des suffixes du texte. Ils utilisent cette approche pour l'analyse d'images représentées par des matrices de pixels (voir aussi l'article récent de Giancarlo et Guaiana [69]).

La première notion d'arborescence des suffixes d'une arborescence présente dans la littérature est due à Kosaraju [105]. Elle repose sur la notion suivante de suffixe d'une arborescence : pour une arborescence A sur Σ et x un sommet de A, le suffixe A_x de A correspond au mot porté par le chemin allant de x à la racine de A. Pour une arborescence A ayant n sommets, Breslauer [27] a proposé un algorithme construisant l'arborescence des suffixes de A en temps $O(n \times \log(|\Sigma|))$ (Shibuya [141] donne un algorithme en temps $O(n \times \log(\log(n)))$ si l'alphabet $\Sigma = \{1, \ldots, n\}$). Cette structure de données permet en particulier de trouver les k occurrences d'un mot ayant m lettres dans une arborescence ayant n sommets en temps $O(m \times \log(|\Sigma|) + k)$. Cependant il ne semble pas qu'elle permette

1. Par *quasi-linéaire*, nous entendons une complexité en temps linéaire pondérée par un facteur logarithmique, souvent dû au temps de branchement.

de résoudre des problèmes de recherche de motifs autre que des mots de façon plus efficace que l'algorithme naïf (c'est-à-dire en temps $O(n \times m)$).

Très récemment, Shibuya [141] a introduit la notion de B-arborescence des suffixes d'une arborescence k-aire A (chaque nœud a exactement k fils), reposant sur la notion suivante de suffixe : le suffixe de A enraciné en un de ses sommets x est la plus grande sous-arborescence de A complète (les sommets situés à la même hauteur sont soit tous des nœuds, soit tous des feuilles) de racine x. En se référant à la proposition 201, on peut alors détecter un motif M ayant m sommets dans un texte T (une arborescence k-aire) en temps $O(m \times \log(|\Sigma|))$ si M est une arborescence k-aire complète sur Σ.

Dans la dernière section du chapitre précédent, nous avons en fait déjà introduit une notion d'arborescence des suffixes d'une arborescence, basée sur les définitions suivantes :

- un suffixe d'une arborescence A est une sous-arborescence maximale de A (les feuille de cette sous-arborescence sont des feuilles de A) : on a donc un suffixe A_x pour chaque sommet x de A,
- le mot associé à un suffixe est obtenu par un parcours en profondeur de A_x.

Si, comme nous l'avons vu, cette structure de données semble s'avérer utile, en pratique, pour accélérer l'algorithme naïf de recherche de motifs dans une arborescence, son utilisation ne permet pas d'avoir un résultat analogue à la proposition 201 et n'améliore pas la complexité théorique de l'algorithme naïf (complexité quadratique).

Dans la suite de ce chapitre, nous proposons une nouvelle définition de l'arborescence des suffixes (conforme à la propriété 199), reposant sur la même notion de suffixe que précédemment (à savoir une sous-arborescence), mais utilisant le codage d'un suffixe par un mot calculé lors d'un *parcours en largeur* de ce suffixe. La structure de données ainsi obtenue permet de traiter en temps quasi-linéaire certains problèmes de recherche de motifs dans une arborescence.

Suffixe et préfixe d'une arborescence, et mots associés. Le principe guidant la définition de l'arborescence des suffixes d'une arborescence que nous proposons est de considérer un mot de longueur h comme une arborescence de hauteur h ayant un seul sommet par niveau (arborescence *filiforme*). Une arborescence de hauteur h (voir définition 180) peut donc être vue comme un "mot généralisé" (chaque niveau comporte non plus un sommet, mais un ensemble de sommets).

Définition 202. Soient A une arborescence de hauteur h et $1 \leq i \leq h$.

- On note $A[i]$ l'ensemble des sommets x de A situés à hauteur i : on qualifie $A[i]$ de $i^{ème}$ niveau de A.
- Si $A[i]$ comporte k sommets, on note $A[i,j]$ le $j^{ème}$ sommet ($1 \leq j \leq k$), à partir de la gauche (ou par ancienneté), de $A[i]$.

Définition 203. Soient A une arborescence et $x = A[i,j]$ un sommet de A.

- On appelle *suffixe de A enraciné en x*, noté $Suf(A,x)$ ou A_x, la sous-arborescence maximale de A de racine x.
- On appelle *préfixe de A finissant en x*, noté $Pref(A,x)$, la sous-arborescence de A constituée des sommets $x' = A[i',j']$ de A tels que soit $i' < i$, soit $i' = i$ et $j' \leq j$.

Il reste à définir l'alphabet Γ des mots $S(A_x)$ associés aux suffixes A_x d'une arborescence A. Pour cela, nous introduisons la notion de décalage d'un sommet et de T-lettre associée à un sommet.

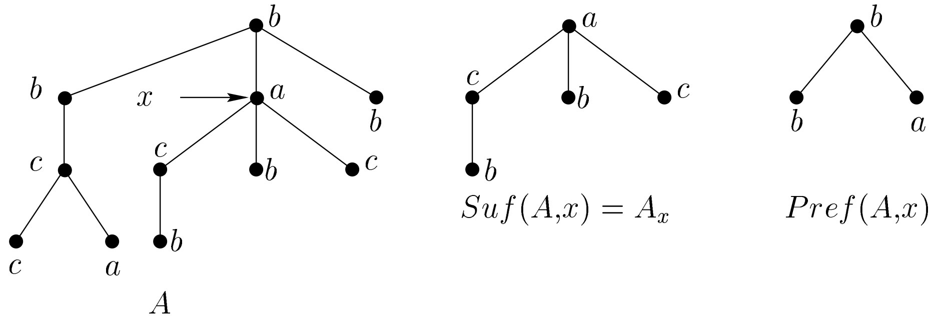


FIG. 9.1 – Illustration de la notion de suffixe et de préfixe d'une arborescence

Définition 204. Soient A une arborescence sur Σ et $x = A[i,j]$ un sommet de A. On appelle *décalage de x dans A*, noté $D_A(x)$, l'entier défini par

- si $x = A[1,1]$ est la racine de A, alors $D_A(x) = 1$,
- si $j = 1$, alors $D_A(x) = j'$ où $A[i-1,j']$ est le père de x,
- sinon, $D_A(x) = j' - j''$ où $A[i-1,j']$ est le père de x et $A[i-1,j'']$ le père de $A[i,j-1]$.

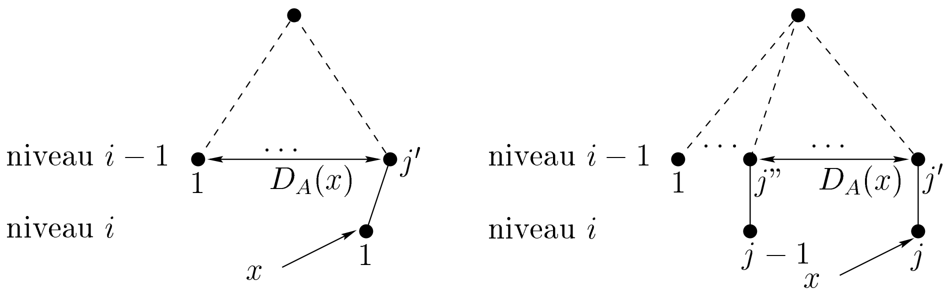


FIG. 9.2 – Le décalage d'un sommet

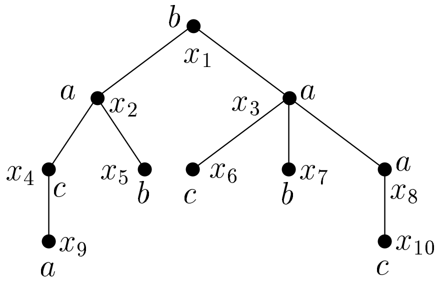
Définition 205. Soient A une arborescence sur Σ et $x = A[i,j]$ un sommet de A. La T-lettre associée à x dans A, notée x_A, est le triplet $(a_x,D_A(x),e(x))$ où $a_x = 1$ si $j = 1$ et $a_x = 0$ sinon, et $e(x)$ est l'étiquette de x.

L'alphabet Γ correspondant à une arborescence A est alors l'alphabet des T-lettres associées à A, c'est-à-dire des triplets (a,b,c) où a est un booléen, b un entier positif ou nul et c une lettre de Σ.

Définition 206. Le *parcours en largeur* d'une arborescence ordonnée A de hauteur h est le parcours consistant à visiter d'abord la racine, puis les sommets à hauteur 2 (on visite successivement $A[2,1]$, $A[2,2]$, …, $A[2,k]$, …), puis les sommets à hauteur 3, …, et pour finir les sommets à hauteur h.

Définition 207. Soit A une arborescence sur Σ ayant n sommets. On note $S(A)$ le mot de longueur n sur Γ tel que, si x est le $i^{ème}$ sommet visité lors du parcours en largeur de A, la $i^{ème}$ lettre de $S(A)$ est la T-lettre x_A associée à x dans A.

Notation. Par la suite, étant donnée une arborescence T sur Σ ayant n sommets, on note x_i le $i^{ème}$ sommet visité lors du parcours en largeur de T. De plus on désigne par S le mot $S(T)$ associé à T et, pour chaque sommet x_i de T, on note T_i l'arborescence suffixe T_{x_i} et S_i le mot $S(T_i)$.



$$S = (1,1,b)(1,1,a)(0,0,a)(1,1,c)(0,0,b)(0,1,c)(0,0,b)(0,0,a)(1,1,a)(0,4,c)$$

$$S_3 = (1,1,a)(1,1,c)(0,0,b)(0,0,a)(1,3,c)$$

FIG. 9.3 – Exemples de mot associé à une arborescence

L'arborescence des suffixes d'une arborescence. L'arborescence des suffixes d'une arborescence T ayant n sommets, notée $AS(T)$, est obtenue en compressant l'ADR des mots S_i pour $i = 1, \dots ,n$. Comme plusieurs suffixes (mots S_i) d'une même arborescence peuvent être identiques, on associe à chaque sommet final u la liste des indices i tels que $m(u) = S_i$. Dans la figure suivante, on représente le sommet final acceptant le mot S_i par un rectangle contenant i.

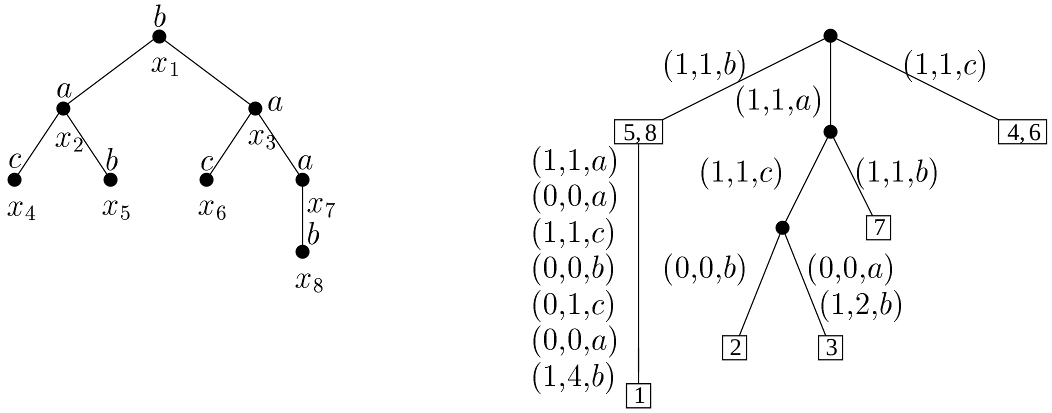
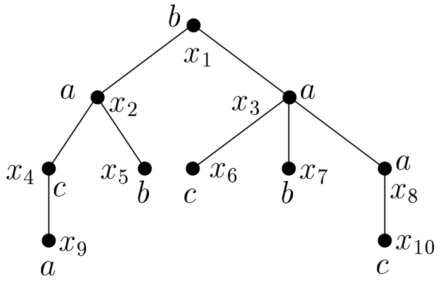


FIG. 9.4 – L'ADR compressée des suffixes d'une arborescence

Si la proposition 200 nous assure que $AS(T)$ comporte au plus $2n$ sommets, lorsque T en comprend n, pour obtenir une structure de donnée occupant un espace en $O(n)$, il est nécessaire de coder chaque mot sur Γ associé à une arête en espace $O(1)$. Dans le cas des mots, le codage d'un arc par un couple d'entiers vérifie cette condition tout en permettant de déterminer le facteur correspondant à une arête (d,f) en temps $O(f-d+1)$.

Dans le cas des arborescences, on peut remarquer que pour tout sommet x_i de T, tout facteur $S_i[l..l']$ de S_i peut être codé par le triplet (x_i,x',x'') où x' (resp. x'') est le sommet de T_i correspondant à la T-lettre $S_i[l]$ (resp. $S_i[l']$).



$$S_3 = (1,1,a)(1,1,c)(0,0,b)(0,0,a)(1,3,c)$$

$$S_3[2..4] = (1,1,c)(0,0,b)(0,0,a) \text{ est codé par } (x_3,x_6,x_8)$$

FIG. 9.5 – Codage d'un facteur par un triplet de sommets

On peut donc représenter le facteur associé à une arête de $AS(T)$ par un tel triplet, ce qui, dans le cas de notre exemple, donne l'arborescence représentée figure 9.6.

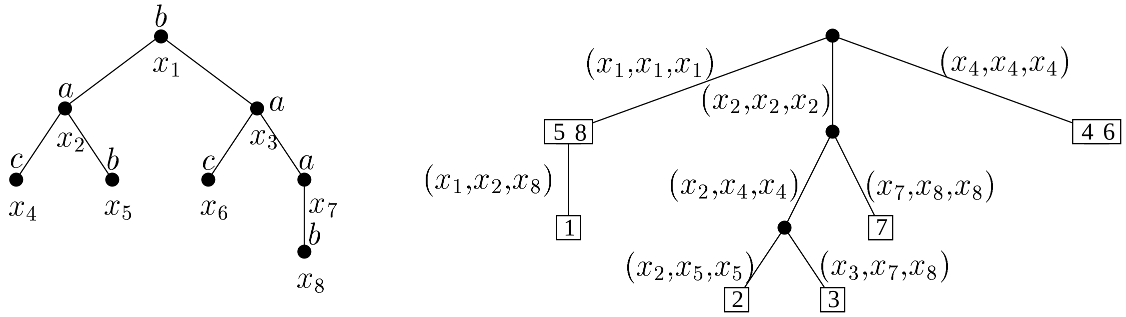


FIG. 9.6 – L'arborescence des suffixes d'une arborescence

On obtient ainsi une structure de données occupant un espace en $O(n)$. Les deux sections suivantes présentent un algorithme permettant de construire cette structure.

Remarque 208. Nous verrons dans la section 9.3 un prétraitement de l'arborescence T permettant, étant donnés trois sommets x, x' et x'' codant un mot comportant k T-lettres, d'obtenir ce mot en temps $O(k)$.

Notation. Par la suite, on considère tout sommet final associé à au moins deux suffixes comme un nœud.

Notation. Dans la suite de ce chapitre, on utilise la notation (x,x',x'') pour représenter indifféremment un mot composé de T-lettres et l'ensemble de sommets de l'arborescence

T_x lui correspondant. De plus, lorsqu'il n'y a pas ambiguïté, on identifie un sommet de cet ensemble avec la T-lettre correspondante.

9.2 L'algorithme de McCreight

Dans cette section, nous présentons l'algorithme de construction de l'arborescence des suffixes d'un mot T (de longueur n sur un alphabet Σ) dû à McCreight [118], algorithme que nous reprenons dans la section suivante pour construire l'arborescence des suffixes d'une arborescence. La présentation que nous en donnons est inspirée du livre de Crochemore et Rytter [52, section 5.3].

Un algorithme simple mais non optimal. Le principe de cet algorithme est de débuter avec l'ADR correspondant au seul mot T (on note cet ADR AS_1), puis de construire incrémentalement $AS(T)$ en insérant les suffixes propres de T dans cette ADR par ordre de longueur décroissante (on note AS_i l'ADR compressé ainsi obtenu après insertion du suffixe $T[i..n]$ et f_i le sommet final correspondant au suffixe $T[i..n]$).

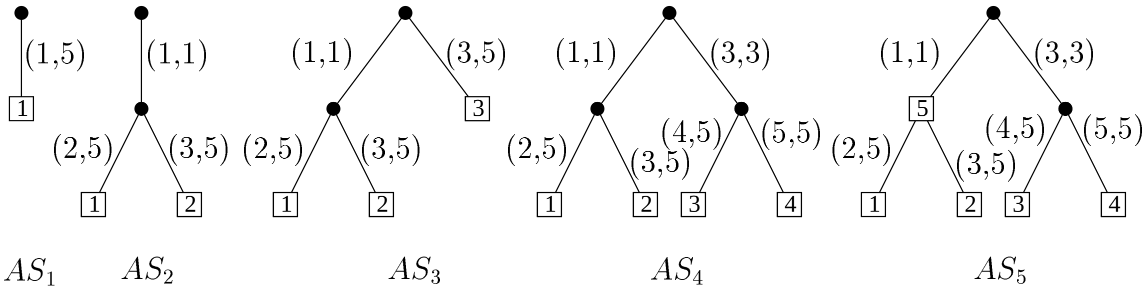


FIG. 9.7 – Les arborescences successives créées par l'algorithme de McCreight si $T = aabba$

Pour insérer le mot $T[i..n]$ dans AS_{i-1}, une solution simple consiste à utiliser une procédure qui, étant donnés un sommet u de AS_{i-1} (normalement la racine) et un mot W, suit lettre à lettre le plus long préfixe de W correspondant à un chemin issu de u et renvoie le couple (v,k) où v est le sommet final de ce chemin (v est créé si nécessaire) et k est la longueur du suffixe de W non reconnu. On appelle cette opération la *procédure de recherche lente*, notée **ChercherL**(u, W).

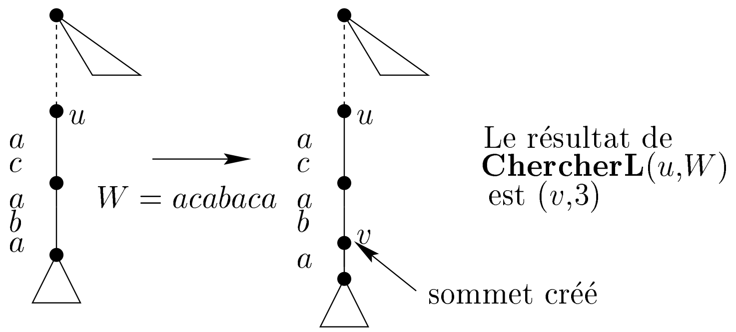


FIG. 9.8 – La procédure **ChercherL**

En effectuant les mêmes remarques que celles faisant suite à l'algorithme 10, il apparaît clairement que la complexité en temps de cette procédure est en $O((|m(v)| - |m(u)|) \times \log(|\Sigma|))$. Pour construire l'arborescence des suffixes $AS(T)$, on peut alors utiliser l'algorithme suivant, qui a une complexité en temps en $O(n^2)$.

Algorithme 22: Construction de l'arborescence des suffixes de T : algorithme simple

Donnée : mot T sur Σ de longueur n
début
créer AS_1 (ADR à deux sommets, la racine r et la feuille f_1 dont l'arête est étiquetée $(1,n)$);
pour i allant de 2 à n **faire**
$(v,k) :=$ **ChercherL**$(r, T[i..n])$;
si $k = 0$ **alors** $f_i := v$;
sinon créer f_i comme fils de v relié à celui-ci par l'arête $(n - k + 1,n)$;
fin

Une première amélioration : les liens suffixes.

Notation. Dans la suite de cette section, on note g_i le premier nœud ancêtre de f_i (f_i est le sommet correspondant au suffixe $T[i..n]$) dans AS_i ($g_i = f_i$ si f_i est un nœud et g_i est le père de f_i si f_i est une feuille) et h_i le père[2] de g_i dans AS_i.

Pour accélérer ce premier algorithme, on s'appuie sur les résultats suivants.

Lemme 209. *Soient $1 \leq i \leq n$ et u un nœud de AS_i autre que g_i. Il existe dans AS_i un sommet v tel que $m(u) = a.m(v)$ où a est une lettre de Σ ($m(v)$ est le plus long suffixe propre de $m(u)$).*

Preuve. Posons $m(u) = a.s$, avec $a \in \Sigma$ et $s \in \Sigma^*$. Si $u \neq g_i$, u étant un nœud de AS_i, il existe deux suffixes $T[j_1..n]$ et $T[j_2..n]$ avec $j_1 < j_2 < i$ tels que $m(u)$ est le plus long préfixe commun de ces deux mots. On en déduit que le plus long préfixe commun des deux suffixes $T[j_1+1..n]$ et $T[j_2+1..n]$ est le mot s et, par définition de AS_i, il existe un sommet v de AS_i tel que $m(v) = s$. $\square$

2. Dans la suite de cette section, nous ne prenons pas en compte les cas $g_i = r$ ou $h_i = r$, où r est la racine de AS_i. Ces deux cas alourdissent la présentation de l'algorithme de McCreight, bien que leur prise en compte ne pose pas de problème particulier.

Propriété 210. Pour $2 \leq i \leq n$, AS_i a au plus un ou deux sommets de plus que AS_{i-1}, les sommets f_i et g_i, qui peuvent être confondus.

On peut utiliser ces résultats et supposer ainsi qu'après avoir inséré le suffixe $T[i+1..n]$ dans AS_i (et donc construit AS_{i+1}), tout sommet u, autre que la racine, qui était présent dans AS_i possède une arête spéciale (non étiquetée) reliant u au sommet v tel que $m(u) = a.m(v)$. On appelle cette arête *le lien suffixe de* u, noté $v = LS(u)$. Pour que cette hypothèse soit vérifiée, compte-tenu de la propriété 210, il suffit d'ajouter lors de la construction de AS_{i+1} un seul lien suffixe, à savoir $LS(g_i)$.

On représente les liens suffixes en pointillés sur la figure suivante.

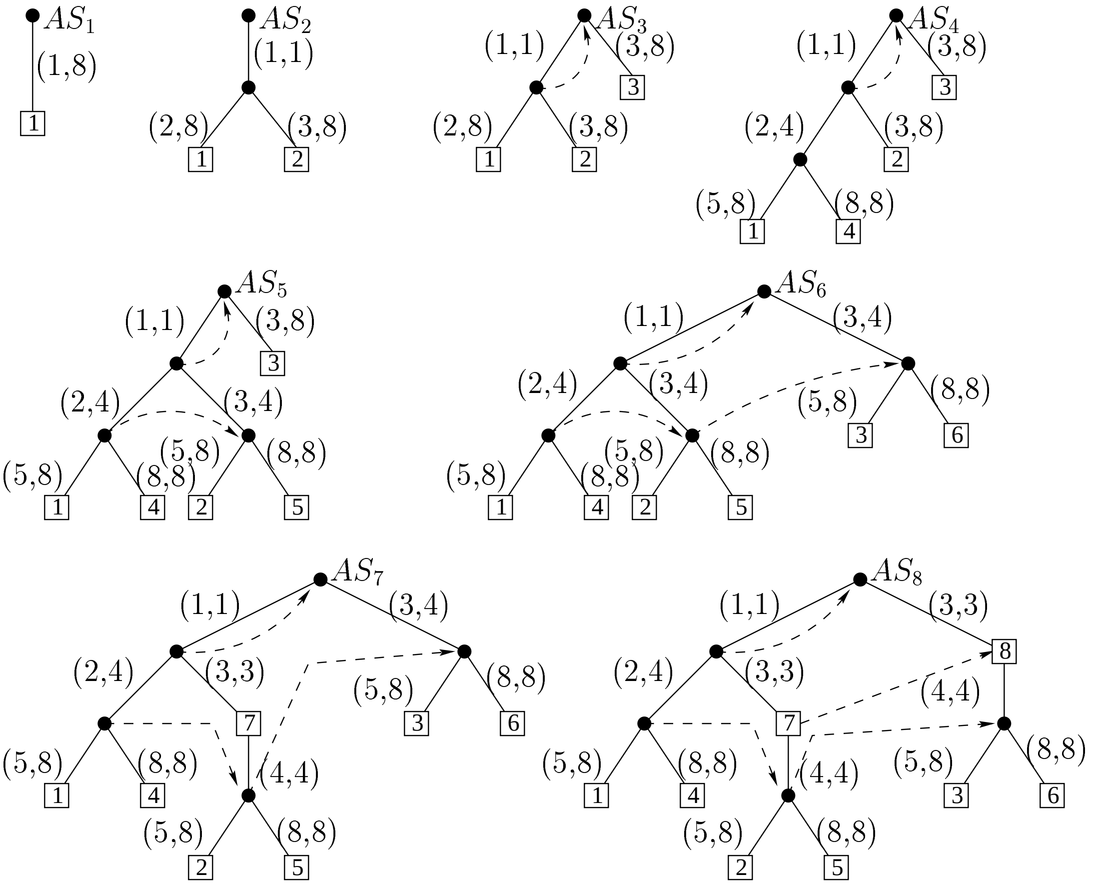


FIG. 9.9 – Les liens suffixes lors de la construction de $AS(aabaabab)$

Pour insérer le suffixe $T[i+1..n]$ dans AS_i, on peut alors suivre un schéma *Remonter-Traverser-Descendre* :

1. *remonter* jusqu'au père h_i de g_i (h_i possède un lien suffixe d'après le lemme 209),
2. suivre ce lien suffixe (*traverser*) pour aboutir au sommet $u = LS(h_i)$,
3. insérer depuis u le suffixe $T[i+1+|m(u)|..n]$ (*descendre*).

Ce schéma permet ainsi d'éviter d'avoir à relire le préfixe $m(u)$ de $T[i..n]$ comme cela était le cas dans l'algorithme simple présenté auparavant. Il nécessite cependant, comme on l'a remarqué précédemment, qu'après l'insertion de $T[i+1..n]$ dans AS_i, le nœud g_i

possède un lien suffixe. De plus, pour un mot T composé de n occurrences de la même lettre de Σ, on remarque que la complexité en temps reste quadratique en n. Pour améliorer cela, on combine l'utilisation des liens suffixes avec le recours à une procédure de "recherche" d'un mot dans un ADR plus efficace que **ChercherL**.

Une seconde amélioration : la recherche rapide. Cette amélioration repose sur le résultat suivant.

Lemme 211. *Dans l'arborescence AS_i, si $m(g_i) = a.s$, avec $a \in \Sigma$ et $s \in \Sigma^*$, alors il existe un sommet u tel que s est un préfixe de $m(u)$.*

Preuve. Si g_i est un sommet déjà présent dans AS_{i-1}, le sommet $v = LS(g_i)$ vérifie cette propriété et on a même $m(v) = s$. Sinon, g_i a été créé lors de l'insertion de $T[i..n]$. Comme g_i est par définition un nœud, il existe un suffixe $T[j..n]$ de T avec $j < i$ tel que $m(g_i)$ est un préfixe de $T[j..n]$. En prenant pour u le sommet f_{j+1} de AS_i, alors on constate que s est un préfixe de $m(u)$. $\square$

Corollaire 212. *Soient (k,l) l'étiquette de l'arête reliant h_i à g_i dans AS_i et $u = LS(h_i)$. Il existe un descendant v de u tel que le mot correspondant à (k,l) est un préfixe du mot correspondant au chemin allant de u à v.*

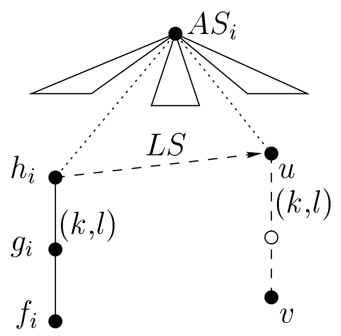


FIG. 9.10 – Illustration du Corollaire 212

Pour déterminer, lors de la partie *descendre*, le préfixe de $T[i+1+|m(u)|..n]$ correspondant à (k,l), il n'est donc pas nécessaire de le chercher en le lisant lettre à lettre car on sait qu'il sera entièrement reconnu à partir du sommet $u = LS(h_i)$. On utilise plutôt la procédure suivante de *recherche rapide* à partir d'un sommet u d'un mot W que l'on sait à priori être reconnu. Cette procédure renvoie comme résultat le sommet v (en le créant si nécessaire) tel que le chemin de u à v correspond au mot W.

Algorithme 23: ChercherR

Données : AS_i, un sommet u de AS_i et un mot W de longueur p
début
  soit (i,j) l'arête quittant u (vers un sommet v) telle que $T[i] = W[1]$;
  si $j - i + 1 = p$ **alors retourner** v;
  sinon si $j - i + 1 < p$ **alors ChercherR**(v, $W[j - i + 2..p]$);
  sinon insérer un sommet x entre u et v de sorte que l'arête reliant u à x (resp. x à v) est étiquetée $(i,i + p - 1)$ (resp. $(i + p,j)$);
fin

Propriété 213. La complexité d'un appel à **ChercherR**(AS_i, u, W) renvoyant comme résultat un sommet v est en $O(k \times \log(|\Sigma|))$, où k est le nombre de nœuds situés sur le chemin allant de u à v.

Ainsi, McCreight [118] a pu proposer l'algorithme suivant pour construire $AS(T)$, basé sur l'utilisation des liens suffixes et de **ChercherR** (rappelons que nous passons sous silence les cas particuliers $g_i = r$ ou $h_i = r$).

Algorithme 24: Algorithme de McCreight

Donnée : mot T sur Σ de longueur n
début
  créer AS_1 (ADR à deux sommets, la racine r et la feuille f_1 dont l'arête est étiquetée $(1,n)$);
  pour i allant de 2 à n **faire**
    soit (l_1,l'_1) l'arête reliant g_{i-1} à f_{i-1};
    soit (l_2,l'_2) l'arête reliant h_{i-1} à g_{i-1};
    $u := LS(h_{i-1})$;
    $x :=$**ChercherR**(u, $T[l_2..l'_2]$);
    $LS(g_{i-1}) := x$;
    $(g_i,k) :=$**ChercherL**(v, $T[l_1..l'_1]$);
    si $k = 0$ **alors** $f_i := g_i$;
    sinon créer f_i comme fils de g_i relié à celui-ci par une arête étiquetée $(n - k + 1,n)$;
fin

Observation 214. Si lors de l'étape de recherche rapide le nœud x est créé, il est clair que la seule arête quittant ce nœud ne commence pas par $T[l_2]$ et il n'est pas nécessaire d'effectuer la phase recherche lente. On peut de suite poser $g_i = x$ et insérer f_i comme fils de g_i, relié à ce dernier par une arête étiquetée (l_2,l'_2).

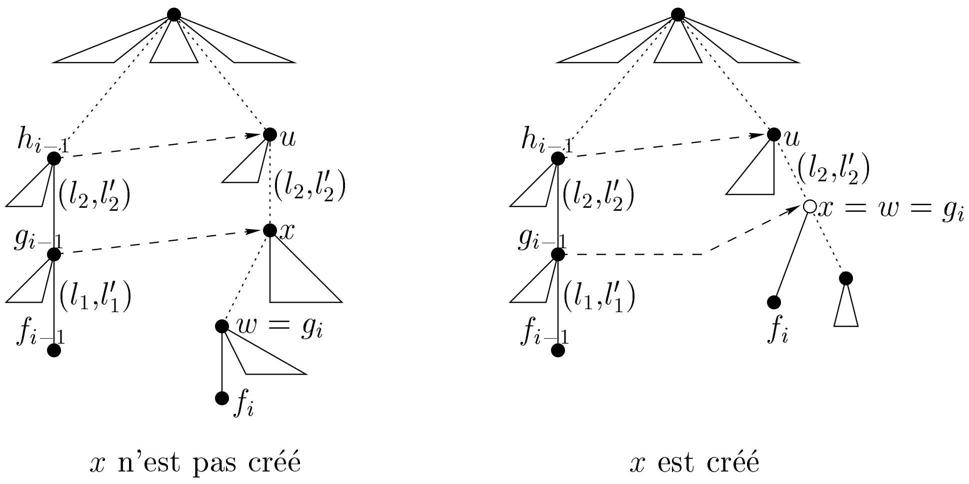


FIG. 9.11 – Illustration de l'observation 214

Proposition 215. *Soit T un mot de longueur n sur l'alphabet Σ. L'algorithme de McCreight calcule l'arborescence des suffixes $AS(T)$ en temps $O(n \times \log(|\Sigma|))$.*

Preuve. La complexité en temps d'un appel à **ChercherL** est proportionnelle au nombre de comparaisons entre un caractère du mot cherché et un caractère de T. On peut remarquer que, lors de l'insertion de $T[i..n]$ dans AS_{i-1}, le nombre de caractères lus par **ChercherL** est $|m(g_i)| - |m(g_{i-1})| + 2$. On en déduit que le nombre total de comparaisons réalisées par les appels à **ChercherL** est exactement

$$\sum_{i=2}^{n}(|m(g_i)| - |m(g_{i-1})| + 2) = |m(g_n)| - |m(g_1)| + 2(n-1) = 3n.$$

La complexité totale des appels à **ChercherL** est donc en $O(n \times \log(|\Sigma|))$.

Examinons maintenant la procédure **ChercherR**. La complexité de l'appel à **ChercherR** lors de la construction de l'arborescence AS_i est proportionnelle au nombre de sommets de AS_i traités lors de cet appel, nombre que nous notons int_i. On remarque que chaque sommet rencontré (sauf éventuellement le dernier) depuis le sommet $u = LS(h_{i-1})$ est un ancêtre de h_i. On en déduit que

$$int_i - 1 \leq |m(h_i)| - |m(u)| \leq |m(h_i)| - (|m(h_{i-1})| - 1)$$

(on ne peut transformer le second $\leq$ en $=$ car h_i et h_{i+1} peuvent être égaux à la racine). Ainsi

$$int_i \leq |m(h_i)| - |m(h_i)| + 2.$$

Un calcul similaire au précédent permet d'affirmer que la complexité des appels à **ChercherR** est en $O(n \times \log(|\Sigma|))$. □

9.3 Le cas des arborescences

Nous présentons maintenant une extension de l'algorithme de McCreight pour construire l'arborescence des suffixes $AS(T)$ d'une arborescence T.

Prétraitement de T. Comme nous l'avons mentionné en présentant la structure de données, nous considérons une arborescence T de hauteur h comme un "mot généralisé" de longueur h. Dans ce paragraphe, nous décrivons un prétraitement permettant de manipuler efficacement T et notamment de décrire un mot (x,x',x'') de k T-lettres en temps $O(k)$ (remarque 208). Dans un premier temps, on s'inspire de la représentation d'un mot par un tableau pour coder T (voir figure 9.12) :

- chaque sommet $x = T[i,j]$ autre que la racine est codé par le couple $(e(x),j')$ si le père de x est le sommet $T[i-1,j']$, la racine étant codée $(e(x),0)$;
- si le niveau $T[i]$ comporte k sommets $(1 \leq i \leq h(T))$, il est représenté par le tableau des couples associés aux sommets de $T[i]$ (ce tableau comportant k éléments) ;
- l'arborescence T est représentée par le tableau des h tableaux $T[i]$.

On suppose de plus qu'on connaît le nombre de sommets n_i présents sur chaque niveau i $(n_1 = 1)$.

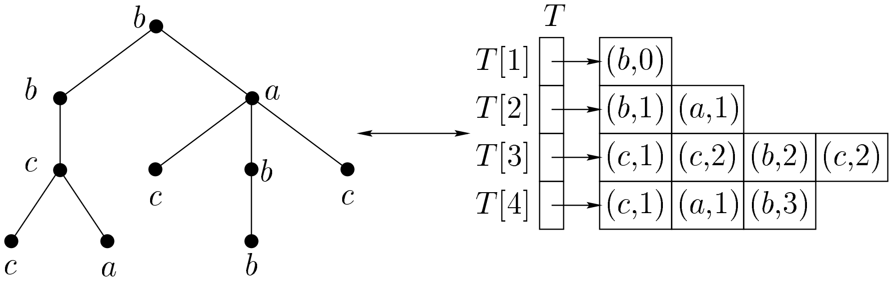


FIG. 9.12 – Codage d'une arborescence par des tableaux

Ce codage d'une arborescence est une généralisation de la représentation d'un mot sous forme de tableau (en considérant un mot comme une arborescence ayant une seule feuille, chaque niveau comporte exactement un sommet). Cependant, pour pouvoir se "déplacer" rapidement dans les tableaux codant T, nous avons besoin du résultat suivant dû à Berkman et Vishkin [18].

Définition 216. Soient $x = T[i,j]$ un sommet de T, p un entier tel que $0 \leq p \leq i-1$ et q un entier tel que $1 \leq q \leq h(T) - i$. On note $Anc(x,p)$ l'ancêtre de x situé sur le niveau $T[i-p]$ et $Succ(x,k)$ le premier sommet y, s'il existe, du niveau $T[i+q]$ tel que le sommet $Anc(y,q) = T[i,j']$ vérifie $j' \geq j$.

Ainsi, on peut noter que le successeur d'un sommet n'est pas toujours un de ses descendants.

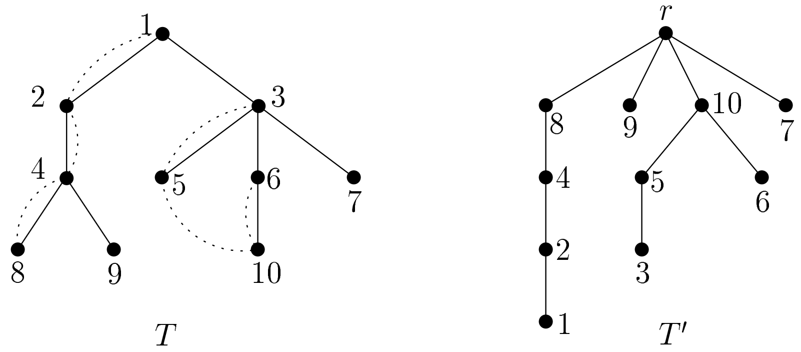
Théorème 217. [18] *Soient T une arborescence à n sommets, $x = T[i,j]$ et $Anc(x,k) = T[i',j']$. On peut prétraiter T en temps et espace $O(n)$ de sorte que, connaissant i, j et k, la détermination de i' et j' s'effectue en temps $O(1)$.*

L'algorithme de Berkman et Vishkin [18] étant extrêmement complexe (à décrire et à mettre en œuvre), il ne nous est pas possible, dans le cadre de ce mémoire, d'en faire une présentation synthétique.

Corollaire 218. *Soient T une arborescence à n sommets, $x = T[i,j]$ un sommet de T et $Succ(x,k) = T[i',j']$. On peut prétraiter T en temps et espace $O(n)$ de sorte que, connaissant i, j et k, la détermination de i' et j' s'effectue en temps $O(1)$.*

Preuve. Lors d'un parcours en profondeur de T, on associe à chaque sommet $x = T[i,j]$ de T le sommet $s_x = T[i+1,j']$ dont le père $T[i,j'']$ vérifie $j'' \geq j$, lorsqu'un tel sommet existe (ces associations sont matérialisées par les arêtes en pointillées sur la figure ci-dessous). Puis on construit une arborescence T' ayant $n+1$ sommets (les sommets de T et une racine r) de la manière suivante (illustrée par la figure ci-dessous dans laquelle nous avons numéroté les sommets de T) :

- si, pour un sommet x, s_x n'est pas défini, on crée une arête entre r et x,
- sinon on crée une arête entre s_x et x.



Par construction, on a alors $y = Succ(x,k)$ dans T si et seulement si $x = Anc(y,k)$ dans T'. La construction de T' s'effectuant en temps $O(n)$, on déduit du théorème 217 le résultat annoncé. □

Lemme 219. *Soient T une arborescence à n sommets, et x et x' deux sommets de T. On peut prétraiter T en temps et espace $O(n)$ de sorte qu'il est possible de déterminer si x' est un ancêtre de x en temps $O(1)$.*

Preuve. Il suffit d'associer à tout sommet y son indice $ind(y)$ lors d'un parcours en profondeur de T et la taille $|T_y|$ de la sous-arborescence T_y dont il est racine (voir section 7.4). □

Lemme 220. *Soit T une arborescence à n sommets. Après avoir appliqué les prétraitements des théorème 217, Corollaire 218 et lemme 219, on peut effectuer les opérations suivantes en temps $O(1)$:*

1. *étant donnés un sommet $x = T[i,j]$ et deux entiers k et l, déterminer i' et j' tels que $T_x[k,l] = T[i',j']$;*
2. *étant donnés un sommet $x = T[i,j]$ et $y = T[k,l]$ un ancêtre de x, calculer la T-lettre x_{T_y} ;*
3. *étant donnés un sommet $x = T[i,j]$ et $y = T[k,l]$ un ancêtre de x, déterminer i' et j' tels que $x' = T[i',j']$ soit le sommet suivant (resp. précédant) x lors d'un parcours en largeur de T_y ;*
4. *étant donnés un sommet $x = T[i,j]$ et un entier k, calculer le nombre de sommets présents dans le $k^{\text{ème}}$ niveau de T_x.*

Preuve. Examinons la première opération. Pour calculer i' et j', il suffit de déterminer le sommet $y = T_x[k,1] = T[i'',j''] = Succ(x,k)$ (Corollaire 218) et si x est ancêtre de ce sommet (lemme 219), alors $i' = i''$ et $j' = j'' + l$.
Pour la seconde opération, notons $x_{T_y} = (a_x, D_{T_y}(x), e(x))$. Pour calculer a_x, il suffit de

tester si $Succ(y,i-k) = x$ (Corollaire 218). Pour calculer $D_{T_y}(x)$, si $x = Succ(y,i-k)$ (et donc $a_x = 1$) alors $D_{T_y}(x) = j'$ où $T[i-1,j']$ est le père de $x = T[i,j]$ (théorème 217). Sinon ($a_x = 0$), $D_{T_y}(x) = j' - j''$ où $T[i-1,j']$ est le père de $x = T[i,j]$ et $T[i-1,j'']$ est le père de $T[i,j-1]$ (théorème 217).
Pour la troisième opération, si x est le dernier sommet de son niveau dans T_y (ce qui équivaut à $j = n_i$ ou $ind(T[i,j+1]) > ind(y) + |T_y|$), le sommet suivant x est $x' = Succ(t,i-k+1)$ et on peut déterminer i' et j' en temps constant grâce au Corollaire 218. Le raisonnement dans le cas où x' précède x est similaire.
Finalement, pour calculer le nombre de sommets présents dans le niveau $T_x[k]$, on détermine $Succ(x,k) = T[i+k,j']$. Si $j = n_j$ (x est le dernier sommet de son niveau dans T) ou $T[i+k,n_{i+k}]$ est un descendant de x, il y a $n_{i+k} - j' + 1$ sommets dans $T_x[k]$. Sinon $x' = T[i,j+1]$ étant le sommet suivant x sur son niveau, on détermine $Succ(x',k) = T[i+k,j'']$. Il y a alors $j'' - j'$ sommets dans $T_x[k]$. □

Le principe de l'algorithme de construction de l'arborescence des suffixes $AS(T)$ d'une arborescence. L'algorithme que nous décrivons par la suite est basé sur le même principe que l'algorithme de McCreight présenté section précédente, à savoir une construction incrémentale de $AS(T)$ basée sur un schéma "Remonter-Traverser-Descendre". Elle utilise une notion de liens suffixes et une procédure de recherche rapide d'un mot à partir d'un sommet de l'arborescence des suffixes. Rappelons les notations suivantes :

- x_i est le $i^{ème}$ sommet de l'arborescence T visité lors d'un parcours en largeur, T_i la sous-arborescence suffixe de T de racine x_i et S_i le mot de T-lettres $S(T_i)$;
- AS_i est l'ADR compressé des mots de T-lettres $S_1, \dots ,S_i$;
- f_i est le sommet final de AS_i correspondant à la lecture du mot S_i, g_i le premier nœud ancêtre de f_i dans AS_i (g_i est égal à f_i si f_i est un nœud) et h_i le père de g_i dans AS_i.

Sommets implicites et liens suffixes. Dans ce paragraphe, nous étendons la notion de lien suffixe à l'arborescence des suffixes d'une arborescence. Dans un premier temps, nous introduisons la notion de sommet implicite.

Notation. Pour tout sommet u de AS_i, il existe une sous-arborescence de T, notée $T(u)$, telle que $m(u) = S(T(u))$ (il peut y avoir plusieurs occurrences de $T(u)$ dans T).

Définition 221. On appelle *sommet implicite* de AS_i un couple (u,l) où u est un sommet de AS_i autre que la racine et l un entier tel que, si $T(u)$ est de hauteur h, alors $0 \leq l \leq h$. On note $T(u,l)$ l'arborescence obtenue en supprimant les l derniers niveaux de $T(u)$ et $m(u,l)$ le mot associé à $T(u,l)$.

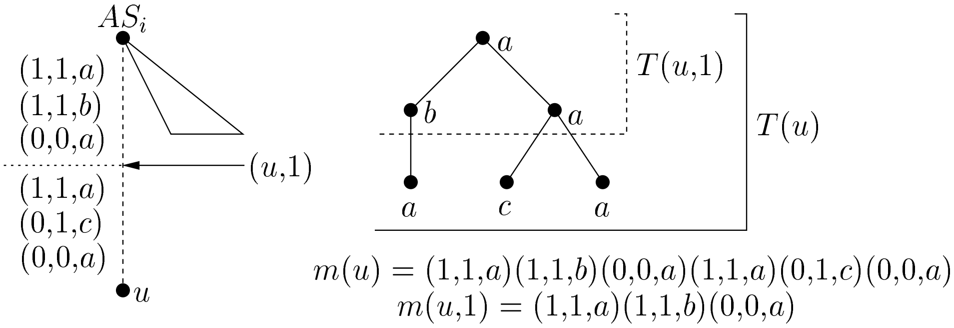
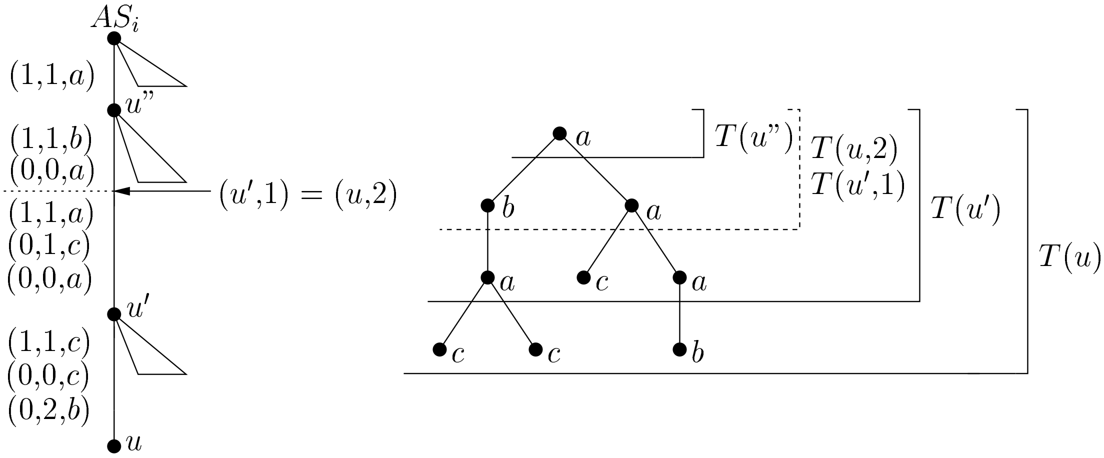


FIG. 9.13 – Illustration de la notion de sommet implicite

On remarque alors immédiatement que si $l = 0$, le sommet implicite $(u{,}l)$ correspond en fait au sommet u ($T(u{,}0) = T(u)$). De plus, plusieurs sommets implicites pouvant représenter la même arborescence, on introduit la notion de sommet implicite minimal.

Définition 222. Un sommet implicite $(u{,}l)$ de AS_i est *minimal* si et seulement si il n'existe pas de sommet implicite $(u'{,}l')$ où u' est un ancêtre de u tel que $m(u'{,}l') = m(u{,}l)$.

Par exemple, soient u, u' et u'' trois sommets de AS_i tels que u' est le père de u et u'' le père de u', correspondant aux arborescences $T(u)$, $T(u')$ et $T(u'')$ représentées par la figure suivante.



Les sommets implicites $(u'{,}1)$ et $(u{,}2)$ correspondent à la même arborescence ($T(u'{,}1) = T(u{,}2)$), mais $(u{,}2)$ n'est pas minimal, alors que $(u'{,}1)$ l'est.

On associe naturellement à la notion de sommet implicite minimal une procédure de minimisation d'un sommet implicite $(u{,}l)$, qui consiste, étant donné $(u{,}l)$ à déterminer l'unique sommet implicite minimal $(u'{,}l')$ tel que $T(u'{,}l') = T(u{,}l)$. Son principe consiste simplement à remonter depuis le sommet u à son ancêtre u' en calculant, pour chaque arête suivie le nombre de niveaux remontés lors de cette étape.

Algorithme 25: Minimiser

Donnée : AS_p et un sommet implicite (u,l)

début

  $u' := u$ et $l' := l$;

  soient v le père de u' et $(x = T[i,j],T[i',j'],T[i'',j''])$ le mot de T-lettres porté par l'arête reliant v à u';

  tant que $i' > i'' - l'$ **faire**

    $u' := v$;

    si $T[i',j']$ est le premier sommet du niveau $i' - i + 1$ de T_x **alors**

      $l' := l' - (i'' - i' + 1)$;

    sinon $l' := l' - (i'' - i')$;

    soient v le père de u' et $(x = T[i,j],T[i',j'],T[i'',j''])$ l'arête reliant v à u';

  retourner (u',l');

fin

Pour deux sommets de T, $x = T[i,j]$ et $y = T[i',j']$ où y est un descendant de x, le sommet y est le premier sommet du niveau $i' - i + 1$ de T_x si et seulement si la T-lettre $y_{T_x} = (a_y,D_{T_x}(y),e(y))$ vérifie $a_y = 1$. Il suffit donc de calculer cette T-lettre y_{T_x} pour déterminer si y est le premier sommet d'un niveau dans T_x, ce qui peut être fait en temps constant (lemme 220). Le nombre d'itérations de la boucle **tant que** étant égal au nombre de sommets situés sur le chemin allant de u' à u, on en déduit la propriété suivante.

Propriété 223. La complexité en temps d'un appel **Minimiser**$((u,l))$ renvoyant comme résultat un sommet implicite (u',l') est en $O(k)$, où k est le nombre de sommets présents dans AS_p sur le chemin allant de u' à u.

Nous introduisons maintenant la notion de restriction d'une arête, illustrée par la figure 9.14.

Définition 224. Soient (x,x',x'') un mot de T-lettres et $\{x' = z_1, \ldots ,z_l = x''\}$ les sommets de la sous-arborescence T_x correspondant à ce mot. Supposons que le sommet x possède d fils $\{y_1, \ldots ,y_d\}$. Pour $1 \leq k \leq d$, on appelle $k^{\text{ème}}$ *restriction de* (x,x',x''), notée $Rest(k,(x,x',x''))$, le facteur (éventuellement vide) de $S(T_{y_k})$ composé des sommets de $\{z_1, \ldots ,z_l\}$ présents dans T_{y_k}.

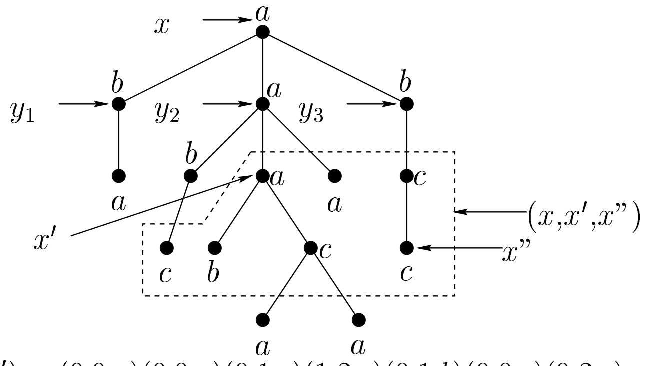
Propriété 225. Pour un mot (x,x',x'') donné, on calcule le triplet (y,y',y'') correspondant à $Rest(k,(x,x',x''))$ en temps $O(1)$.

En effet, soient $(x = T[i,j],x' = T[i',j'],x'' = T[i'',j''])$ et y le $k^{\text{ème}}$ fils de x :

- si x' est un descendant de y, alors $y' = x'$,
- si x' est un descendant d'un frère gauche (resp. droit) de y, alors $y' = T_y[i' - i,1]$ (resp. $y' = T_y[i' - i + 1,1]$),

et on détermine y'' de la façon similaire. Les lemmes 219 et 220 assurent que l'on peut réaliser ces calculs en temps $O(1)$.

Le lemme suivant est l'analogue du lemme 209 pour l'arborescence des suffixes et permet d'introduire la notion de lien suffixe. On peut cependant noter une différence



$(x,x',x'') = (0{,}0{,}a)(0{,}0{,}a)(0{,}1{,}c)(1{,}2{,}c)(0{,}1{,}b)(0{,}0{,}c)(0{,}2{,}c)$
$Rest(1,(x,x',x'')) = \epsilon$ (le mot vide)
$Rest(2,(x,x',x'')) = (0{,}0{,}a)(0{,}0{,}a)(1{,}1{,}c)(0{,}1{,}b)(0{,}0{,}c)$
$Rest(3,(x,x',x'')) = (1{,}1{,}c)(1{,}1{,}c)$

FIG. 9.14 – Restrictions d'un mot de T-lettres

fondamentale par rapport aux mots : le lien suffixe d'un sommet peut ne pas correspondre à un sommet de AS_i, mais à un sommet implicite.

Lemme 226. *Soient x_i un sommet de T et x_j son père ($j < i$) possédant d fils. Pour tout nœud u de AS_i autre que la racine et déjà présent dans AS_j, et tout entier k tel que $1 \leq k \leq d$, il existe un sommet implicite (v,l) dans AS_i tel que $m(v,l) = Rest(k,m(u))$. On dit alors que le couple (u,k) admet un lien suffixe valide, noté $LS(k,u)$.*

Preuve. Si u est un nœud de AS_j, il existe $j_1 < j$ tel que le préfixe de longueur $|m(u)|$ de S_{j_1} est égal à $m(u)$. x_{j_1} a donc degré d, et, si x_p est son $k^{ème}$ fils, on aura inséré S_p avant S_i. Comme $Rest(k,m(u))$ est préfixe de S_p, on en déduit qu'il existe un sommet v de AS_i tel que le mot $Rest(k,m(u))$ est préfixe de $m(v)$. □

Lemme 227. *Soit T une arborescence à n sommets. Le nombre de couples (u,k), où u est un sommet de $AS(T)$ et k un entier, qui admettent un lien suffixe valide est au plus de $2(n-1)$.*

Preuve. Comme dans le cas des mots, on peut remarquer que lors de l'insertion du suffixe S_i (de racine x_i) dans l'ADR compressé AS_{i-1}, on crée au plus deux sommets. Supposons que l'on associe à x_i l'ensemble des sommets de $AS(T)$ créés lors de l'insertion de S_i. On a donc, pour tout $i \in [n]$, au plus deux sommets associés à x_i. Soit u l'un de ces éventuels sommets et supposons x_i de degré d. Le nombre de couples (u,k) admettant un lien suffixe valide est au plus d, c'est-à-dire le nombre d'arêtes quittant x_i dans T. On a donc au plus $2d$ couples (u,k) admettant un lien suffixe tels que u est un sommet de $AS(T)$ associé à x_i, ce qui implique le résultat annoncé. □

Nous avons maintenant un des éléments indispensables au schéma "Remonter-Traverser-Descendre", à savoir les liens suffixes, et nous avons montré que l'espace additionnel nécessaire au stockage de ces liens est linéaire. Dans le paragraphe suivant, nous nous intéressons aux procédures de recherche d'un mot dans AS_i.

Les procédures de recherche lente et rapide. La procédure de recherche lente d'un mot de T-lettres (x,x',x'') dans AS_i est similaire à la procédure correspondante dans le cas des mots (la procédure **ChercherL**). Elle présente les caractéristiques suivantes :

- le point de départ de la recherche est un sommet implicite (u,l),
- le résultat retrouvé est le couple (v,z) où v est un sommet (créé si nécessaire) tel que $m(v)$ est le plus long préfixe de $m(u,l).(x,x',x'')$ reconnu dans AS_i, et z le premier sommet de (x,x',x'') n'appartenant pas à ce préfixe reconnu ($z = NULL$, si (x,x',x'') est entièrement reconnu).

On note **A-ChercherL**((u,l), (x,x',x'')) cette procédure. En remarquant que

- pour un sommet implicite (u,l) donné, le prétraitement de T permet d'obtenir en temps constant la première T-lettre de l'arête reliant u à son père n'appartenant pas à $m(u,l)$,
- on peut lire un mot (x,x',x'') de k T-lettres en temps $O(k)$ (conséquence du lemme 220),

on obtient le résultat suivant.

Propriété 228. Soit T une arborescence à n sommets. La complexité en temps d'un appel à **A-ChercherL** lors de la construction de $AS(T)$ est en $O(k \times \log(n))$, où k est la longueur du plus long préfixe de (x,x',x'') reconnu lors de cet appel.

La procédure de recherche rapide d'un T-mot (x,x',x'') dans une sous-arborescence de AS_i, sachant que ce T-mot est reconnu, suit le même principe que dans le cas des mots (procédure **ChercherR**). Toutefois, cette recherche débute en un sommet implicite (u,l) et elle renvoie comme résultat le sommet implicite minimal (u',l') tel que $m(u',l') = m(u,l).(x,x',x'')$. On la note **A-ChercherR**((u,l), (x,x',x'')). Elle s'appuie sur le résultat suivant.

Propriété 229. Soient deux mots $W = (x,x',x'')$ et $W' = (y,y',y'')$ dont on sait que l'un est préfixe de l'autre. On peut, en temps $O(1)$, déterminer

- si W est préfixe de W' et quel est le suffixe (y,z,y'') de W' non présent dans W ;
- si W' est préfixe de W et quel est le suffixe (x,z,x'') de W non présent dans W' ;
- si $W = W'$.

En effet, on déduit du point 4 du lemme 220 que pour un mot $(x = T[i,j],x' = T[i',j'],x'' = T[i'',j''])$ donné et pour $i' \leq k \leq i''$, on peut calculer le nombre de sommets de $T_x[k]$ en temps $O(1)$. La propriété découle de ce constat et des faits suivants :

- $W = W'$ si et seulement si ces deux mots comportent le même nombre de niveaux et le même nombre de sommets sur leurs derniers niveaux ;
- W (resp. W') est préfixe de W' (resp. W) si et seulement si W (resp. W') "possède moins de niveaux" que W' (resp. W) ou le même nombre de niveaux mais moins de sommets sur son dernier niveau.

Algorithme 26: A-ChercherR

Données : AS_p, un sommet implicite (u,l) de AS_p et un mot (x,x',x'')
début
  si $l = 0$ **alors**
    soit (y,y',y'') l'arête quittant u (vers un sommet v) telle que $y' = x'$ et $u' := u$;
  sinon
    soient (y,z,y'') l'arête reliant u à son père u', (y,y',y'') le suffixe de ce mot correspondant aux l derniers niveaux et $v := u$;
  si $(x,x',x'') = (y,y',y'')$ **alors**
    retourner $(v,0)$;
  sinon
    si (y,y',y'') est préfixe (x,x',x'') **alors**
      soit z le premier sommet de (x,x',x'') n'appartenant pas à (y,y',y'');
      A-ChercherR($(v,0)$, (x,z,x''));
    sinon
      soit $z = T[i,j]$ le premier sommet de $(y,y',y'' = T[i'',j''])$ n'appartenant pas à (x,x',x'');
      retourner $(v,j'' - j + 1)$;
fin

Comme dans le cas des mots, on vérifie aisément la propriété suivante.

Propriété 230. Soit T une arborescence à n sommets. La complexité en temps d'un appel à **A-ChercherR** lors de la construction de $AS(T)$ est en $O(k \times \log(n))$, où k est le nombre de sommets de AS_p visités lors de cet appel.

L'algorithme final et sa complexité en temps. Nous disposons maintenant de presque tous les éléments permettant de décrire l'algorithme de construction de $AS(T)$. Cet algorithme suit le même schéma que l'algorithme de McCreight, excepté sur le point suivant : les fils x_i d'un sommet x_j n'étant pas,en général, traités immédiatement après x_j, si g_j désigne le père de f_j dans AS_j, de nouveaux sommets étant insérés dans $AS_{j+1}, \ldots$, AS_{i-1}, le père de f_j dans AS_{i-1} peut être différent de g_j. On doit donc associer à tout sommet x_j de T les informations suivantes :

- le sommet f_j acceptant T_{x_j},
- le père g_j de f_j dans AS_j et l'arête (x_j,x'_j,x''_j) reliant g_j à f_j,
- le père h_j de g_j dans AS_j et l'arête (y_j,y'_j,y''_j) reliant h_j à g_j.

Comme dans le cas de l'algorithme 24, pour simplifier la présentation de notre algorithme, nous passons sous silence le traitement des cas où $g_i = r$ ou $h_i = r$, r désignant la racine.

Algorithme 27: Construction de $AS(T)$

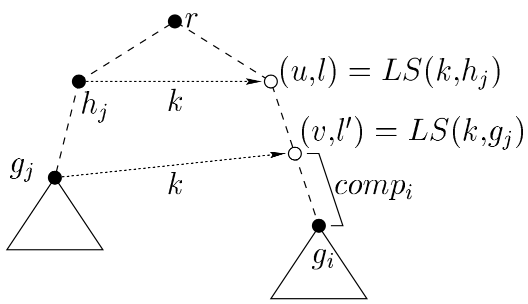
Donnée : T
début
créer AS_1 (ADR compressé à deux sommets, la racine r et la feuille f_1 associée au T-mot $S_1 = S(T_1)$);
pour chaque sommet x_i de T autre que la racine visité lors d'un parcours en largeur **faire**
soient x_j le père de x_i (x_i est le $k^{ème}$ fils de x_j) et, tels que définis précédemment les sommets f_j, g_j et h_j, et les arêtes (x_j,x'_j,x''_j) et (y_j,y'_j,y''_j);
1 $LS(k,h_j) :=$ **Minimiser**$(LS(k,h_j))$;
$(u,l) := LS(k,h_j)$;
2 $(v,l') :=$**A-ChercherR**$((u,l),\ (y_j,y'_j,y''_j))$;
3 $LS(k,g_j) := (v,l')$;
4 $(w,x) :=$**A-ChercherL**$((v,l'),\ (x_j,x'_j,x''_j))$;
si $x = NULL$ **alors** $f_i := w$;
sinon créer f_i comme fils de w relié à celui-ci par une arête (x_j,x,x''_j);
5 $LS(k,g_j) :=$ **Minimiser**$(LS(k,g_j))$;
fin

Pour terminer, nous cherchons à déterminer la complexité en temps de cet algorithme, qui nécessite de calculer la complexité des appels à **A-ChercherL**, la complexité des appels à **Minimiser** et la complexité des appels à **A-ChercherR**.

Dans les figures suivantes, un sommet blanc correspond à un sommet éventuellement implicite, les arêtes en pointillés à un chemin dans l'arborescence et les flêches en pointillés aux liens suffixes.

Lemme 231. *Soit T une arborescence à n sommets. La complexité en temps des appels à* **A-ChercherL** *lors de la construction de $AS(T)$ est en $O(n \times \log(n))$.*

Preuve. La complexité d'un appel à **A-ChercherL**$((u,l),\ (x,x',x''))$ est déterminée par la longueur du plus long préfixe de (x,x',x'') reconnu à partir de (u,l) (propriété 228). Notons $comp_i$ la longueur de ce préfixe lors de l'appel à **A-ChercherL** effectué lors de l'insertion de S_i dans AS_{i-1} (étape 4).



Le sommet x_i désignant le $k^{ème}$ fils de x_j, on a clairement

$$comp_i \leq |m(g_i)| - |Rest(k,m(g_j))| + 1.$$

On en déduit que

$$\sum_{i=2}^{n} comp_i \leq (n-1) + \sum_{i=2}^{n} |m(g_i)| - \sum_{j=1}^{n} \left(\sum_{k=1}^{deg(x_j)} |Rest(k,m(g_j))| \right)$$

où $deg(x_j) =$ est le nombre de fils de x_j. En remarquant que

$$|m(g_j)| = 1 + \sum_{k=1}^{deg(x_j)} |Rest(k,m(g_j))|$$

on en déduit que

$$\sum_{i=2}^{n} comp_i \leq (n-1) + \sum_{i=2}^{n} |m(g_i)| - \sum_{j=1}^{n} (|m(g_j)| - 1)$$

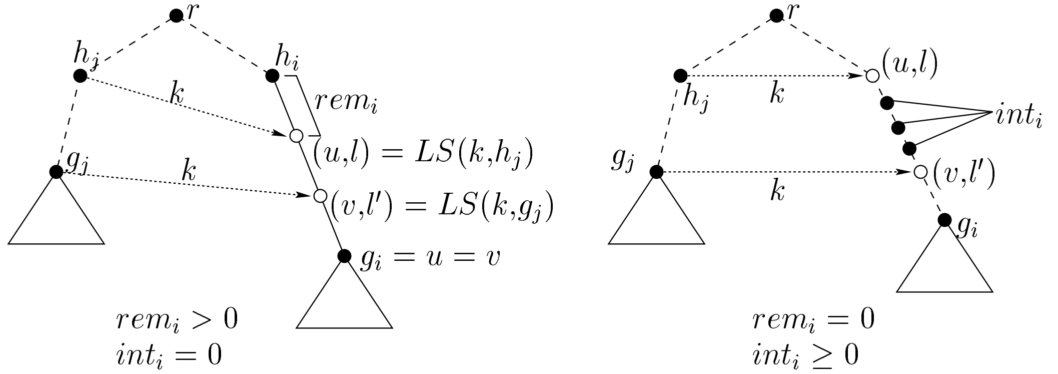
$$\Longrightarrow \sum_{i=2}^{n} comp_i \leq (2n-1) - |m(g_1)| < 2n.$$

Le facteur $\log(n)$ correspond pour sa part au temps de branchement (il y a au plus n arêtes quittant un sommet de AS_i). □

Lemme 232. *Soit T une arborescence à n sommets. La complexité en temps des appels à* **Minimiser** *lors de la construction de $AS(T)$ est en $O(n)$.*

Preuve. Le second appel à **Minimiser** (étape 5) a pour seul but de prendre en compte le fait que lors de l'appel à **A-ChercherL** (étape 3), un sommet a pu être créé, ce qui fait que le lien suffixe $LS(k,g_j)$ peut ne plus être minimal, alors qu'il l'était lors de sa création (étape 4). Comme on a ajouté au plus un sommet, cet appel s'effectue clairement en temps $O(1)$. Pour borner la complexité du premier appel à **Minimiser** (étape 1), on peut remarquer que, pour un sommet implicite (u,l) et un descendant v de u, le nombre total de sommets visités lors des minimalisations de (u,l) est au plus $|m(v)| - |m(u,l)|$. En remarquant que lors de la création du lien suffixe $LS(k,g_j) = (w,l')$, g_i est un descendant de w, on en déduit que la complexité totale des appels à **Minimiser** correspondant à l'étape 1 est au plus $\sum_{i=2}^{n} comp_i$, ce qui, compte tenu de la preuve du lemme précédent prouve le résultat annoncé. □

Il reste à examiner la complexité des appels à **A-ChercherR**. On rappelle que la complexité de l'appel à **A-ChercherR** lors de l'insertion de S_i dans AS_{i-1} est déterminée par le nombre int_i de sommets de AS_{i-1} visités lors de cet appel (propriété 230). On distingue deux cas, illustrés par la figure suivante



et on introduit la notation suivante (on rappelle qu'un sommet est considéré comme faisant partie de ses ancêtres) : si h_i est un ancêtre de u et $h_i \neq u$ (figure de gauche), $rem_i = |m(u,l)| - |m(h_i)| + 1$, sinon, $rem_i = 0$.

Lemme 233. *Soit T une arborescence à n sommets. La complexité en temps des appels à* **A-ChercherR** *lors de la construction de $AS(T)$ est en $O((n+k) \times \log(n))$, où $k = \sum_{i=2}^{n} rem_i$.*

Preuve. Comme dans le cadre des mots, pour $i = 2, \ldots ,n$, on note $res_i = |m(f_i)| - |m(u,l)| + 1$. En remarquant que

$$\sum_{x_i \ fils \ de \ x_j} res_i \leq res_j + rem_j - int_j,$$

on en déduit que

$$\sum_{j=2}^{n} int_j \leq \sum_{j=2}^{n} (res_j + rem_j) - \sum_{j=2}^{n} \left(\sum_{x_i \ fils \ de \ x_j} res_i \right),$$

ce qui, si on note p le nombre de sommets présents sur les niveaux $T[1]$ (la racine de T) et $T[2]$ (les fils de cette racine), est équivalent à

$$\sum_{j=2}^{n} int_j \leq \sum_{j=2}^{n} res_j - \sum_{j=p+1}^{n} res_j + \sum_{j=2}^{n} rem_j.$$

En notant que $\sum_{j=1}^{p} res_j \leq n$, on obtient alors le résultat annoncé. $\square$

Nous n'avons cependant pas été en mesure de montrer que $\sum_{i=2}^{n} rem_i$ est en $O(n)$, ce qui induirait une construction de $AS(T)$ en temps $0(n \times \log(n))$. Les nombreux tests que nous avons menés, combinés à de vains efforts pour construire un contre-exemple, nous amènent toutefois à formuler la conjecture suivante.

Conjecture 234. *$\sum_{i=2}^{n} rem_i$ est en $O(n)$.*

Le tableau suivant présente les résultats en moyenne de nos tests, effectués sur des arborescences quelconques. Dans ce tableau, N désigne le nombre de sommets de $AS(T)$, ACL la complexité des appels à **A-ChercherL**, ACR la complexité des appels à **A-ChercherR**, et Min la complexité des appels à **Minimiser**.

n	$\|\Sigma\|$	N	ACL	ACR	Min
100	1	44	102	22	1
100	3	54	100	13	1
100	5	59	98	10	0
100	7	62	95	7	0
1000	1	368	1018	186	3
1000	3	439	1017	122	1
1000	5	475	1017	109	1
1000	7	480	1013	88	0
10000	1	3222	10135	1613	24
10000	3	3816	10134	1093	7
10000	5	4063	10132	965	4
10000	7	4148	10131	833	2

TAB. 9.1 – Construction de $AS(T)$: résultats expérimentaux moyens

Applications à la recherche de motifs. Comme nous l'avons évoqué en début de chapitre, notre but en définissant une telle structure de données est de traiter le problème de la recherche d'un motif M de m sommets dans une arborescence statique T à n sommets, en utilisant une structure de données efficace en espace.

Dans le cas général, tel que nous l'avons défini dans le chapitre précédent, il ne semble pas que l'utilisation de $AS(T)$ permette d'améliorer la borne quadratique $O(n \times m)$ de l'algorithme naïf. En effet, soient M un motif, $S(M)$ le mot de T-lettres correspondant et u un sommet de $AS(T)$ (on rappelle qu'on note $T(u)$ la sous-arborescence maximale de T telle que $S(T(u)) = m(u)$). On dit que u est un sommet minimal acceptant M si et seulement si

– il existe une occurrence de M en la racine de $T(u)$,

– pour tout ancêtre $v \neq u$ de u, il n'existe pas d'occurrence de M en la racine de $T(v)$.

On en déduit que si w est un sommet final de $AS(T)$ descendant de u acceptant S_i, alors on a une occurrence de M en x_i dans T. On peut alors construire des couples (M,T) tels que le nombre de sommets minimaux acceptant M de $AS(T)$ est de l'ordre de $O(n)$ et que deux chemins quelconques dans $AS(T)$ allant de la racine r vers deux de ces sommets ont un plus long préfixe commun de taille constante. Dans ce cas, en utilisant seulement $AS(T)$, repérer tous les sommets minimaux acceptant M nécessite un temps $O(n \times m)$.

On peut cependant, dans le cas de la recherche de motifs dans des termes, pour le problème RMCA, utiliser $AS(T)$ pour accélerer en pratique l'algorithme naïf, et ce en reprenant le principe des algorithmes présentés dans le chapitre 8. Les tests que nous avons effectués dans ce cas indiquent toutefois que les algorithmes du chapitre 8, qui se caractérisent par leur simplicité, sont plus efficaces en pratique.

On peut aussi utiliser $AS(T)$ dans le cadre de la proposition 201, pour résoudre efficacement des cas restreints de recherche de motifs dans une arborescence. Dans le cas des termes, on déduit par exemple immédiatement de la proposition 201 le résultat suivant (on rappelle que le texte T est étiqueté sur Σ et le motif M sur $\Sigma \cup \{v\}$).

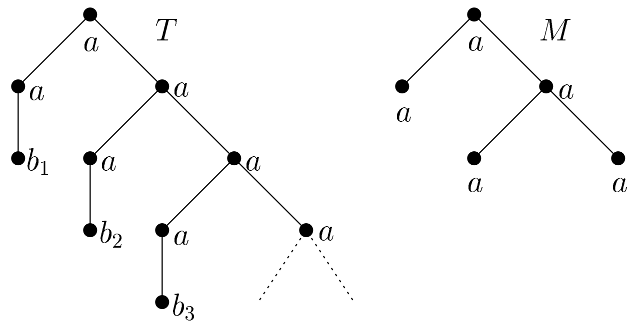


FIG. 9.15 – Un couple (T,M) maximisant la recherche des sommets minimaux acceptant M

Proposition 235. *Soit T un terme à n sommets et M un motif à m sommets tel que tous les sommets de son dernier niveau sont étiquetés v et aucun sommet étiqueté v n'est situé ailleurs que sur ce dernier niveau. Après construction de $AS(T)$, on peut détecter les k occurrences de M dans T en temps $O(m \times \log(n) + k)$.*

Dans le cas des arborescences quelconques, le motif M est dit *complet* si et seulement si toutes ses feuilles sont situées sur son dernier niveau (cette notion est une extension de la notion d'arborescence k-aire complète). Une occurrence de M dans T est dite *complète* si tout nœud de M de degré $d > 0$ correspond à un nœud de T de même degré d. Dans la figure suivante, la seule occurrence complète de M dans T est matérialisée par les arêtes en pointillés.

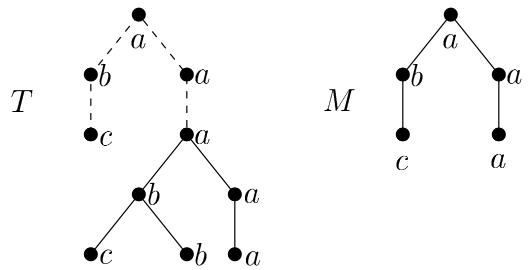


FIG. 9.16 – Illustration de la notion d'occurrence complète

On déduit de la proposition 201 et de la définition de $AS(T)$ le résultat suivant, qui généralise le résultat de Shibuya sur les arborescences k-aires complètes [141].

Proposition 236. *Soit T un texte à n sommets et M un motif complet à m sommets. Après construction de $AS(T)$, on peut détecter les k occurrences complètes de M dans T en temps $O(m \times \log(n) + k)$.*

9.4 Conclusion

Dans ce chapitre, nous avons proposé une approche du problème de la recherche de motifs dans une arborescence statique basée sur l'utilisation d'une structure de données indexant les suffixes du texte statique, l'arborescence des suffixes d'une arborescence. Si la notion de suffixe que nous avons choisie, consistant à voir une arborescence comme un mot généralisé et à regrouper les sommets de T par niveau, permet de définir une structure de données efficace en espace, elle ne permet cependant de traiter efficacement (en termes de complexité dans le pire des cas) que des instances restreintes du problème de la recherche de motifs dans une arborescence. De plus, les techniques nécessaires à la construction de $AS(T)$ sont lourdes, et d'un point de vue pratique, les algorithmes proposés dans le chapitre 8 semblent plus intéressants, du fait de leur simplicité et de leurs performances pratiques.

Cependant, nous pensons que d'un point de vue théorique, la structure de données que nous avons introduite est intéressante en elle-même, car elle correspond à une généralisation naturelle de l'arborescence des suffixes d'un mot. Il nous a ainsi paru naturel de définir un algorithme de construction de cette structure s'appuyant sur un algorithme de construction de l'arborescence des suffixes d'un mot, et l'algorithme de McCreight nous a semblé être le meilleur candidat. Toutefois, nous n'avons pas pu prouver une borne optimale pour notre algorithme et le problème de la construction de l'arborescence des suffixes d'une arborescence en temps quasi-linéaire reste à ce jour ouvert. Cependant, comme nous l'avons mentionné en début de chapitre, Shibuya a récemment traité le cas simplifié des arborescences régulières (tous les nœuds ont le même degré) et, se basant sur l'algorithme de Farach [60], a proposé un algorithme quasi-linéaire pour l'arborescence des suffixes d'une arborescence régulière. Il serait intéressant d'essayer d'étendre sa technique au cas des arborescences quelconques.

Bibliographie

[1] Aho (A. V.) et Corasick (M. J.). – Efficient string matching: an aid to bibliographic search. *Comm. ACM*, vol. 18, 1975, pp. 333–340.

[2] Alonso (L.) et Schott (R.). – *Random generation of trees.* – Kluwer Academic Publishers, Boston, 1995.

[3] Andersson (A.), Larsson (N. J.) et Swanson (K.). – Suffix trees on words. *In : Combinatorial Pattern Matching, CPM'96. Lecture Notes in Comput. Sci.*, volume 1075, pp. 102–115. – Springer, 1996.

[4] Andersson (A.) et Nilsson (S.). – Efficient implementation of suffix trees. *Software: Practice and Experience*, vol. 25, n° 2, 1995, pp. 129–141.

[5] Apostolico (A.). – The myriad virtues of subword trees. *In : Combinatorial algorithms on words. NATO Adv. Sci. Inst. Ser. F Comput. Systems Sci.*, volume 12, pp. 85–96. – Springer, 1985.

[6] Apostolico (A.). – *Pattern matching algorithms*, chap. Online string searching, pp. 89–122. – In Apostolico et Galil [7].

[7] Apostolico (A.) et Galil (Z.) (édité par). – *Pattern matching algorithms.* – Oxford University Press, New York, 1997.

[8] Apostolico (A.), Iliopoulos (C.), Landau (G. M.), Schieber (B.) et Vishkin (U.). – Parallel construction of a suffix tree with applications. *Algorithmica*, vol. 3, n° 3, 1988, pp. 347–365.

[9] Athanasiadis (C.) et Linusson (S.). – A simple bijection for the regions of the Shi arrangement of hyperplans. *Discrete Math.*, vol. 204, n° 1-3, 1999, pp. 27–39.

[10] Beissinger (J. S.). – On external activity and inversions in trees. *J. Combin. Theory Ser. B*, vol. 33, n° 1, 1982, pp. 87–92.

[11] Bender (E. A.) et Richmond (L. B.). – A multivariate Lagrange inversion formula for asymptotic calculations. *Electron. J. Combin.*, vol. 5, n° 1, 1998, Research Paper 33, 4 pp.

[12] Bender (E. A.) et Richmond (L. B.). – Multivariate asymptotics for products of large powers with applications to Lagrange inversion. *Electron. J. Combin.*, vol. 6, n° 1, 1999, Research Paper 8, 21 pp.

[13] Bender (M. A.) et Farach-Colton (M.). – The LCA problem revisited. *In : Latin American Theoretical Informatics, LATIN'2000. Lecture Notes Comput. Sci.*, volume 1776, pp. 89–94. – Springer, 2000.

[14] Berge (C.). – *Graphes et hypergraphes.* – Dunod, Paris, 1973. Deuxième édition.

[15] Bergeron (F.), Flajolet (P.) et Salvy (B.). – Varieties of increasing trees. *In : Colloque sur les Arbres en Algèbre et en Programmation, CAAP'92. Lecture Notes in Comput. Sci.*, volume 581, pp. 24–48. – Springer, 1992.

[16] Bergeron (F.), Labelle (G.) et Leroux (P.). – *Combinatorial species and tree-like structures.* – *Encyclopedia of mathematics and its applications*, volume 67, Cambridge University Press, Cambridge (MA), 1998.

[17] Berkman (O.) et Vishkin (U.). – Recursive star-tree parallel data structure. *SIAM J. Comput.*, vol. 22, n° 2, 1993, pp. 221–242.

[18] Berkman (O.) et Vishkin (U.). – Finding level-ancestors in trees. *J. Comput. System Sci.*, vol. 48, n° 2, 1994, pp. 214–230.

[19] Berry (V.). – *Méthodes mathématiques et informatiques pour la reconstruction phylogénétique.* – Thèse de Doctorat, LIRMM, Université Montpellier II, 1997.

[20] Blumer (A.), Blumer (J.), Haussler (D.), Ehrenfeucht (A.), Chen (M. T.) et Seiferas (J.). – The smallest automaton recognizing the subwords of a text. *Theoret. Comput. Sci.*, vol. 40, n° 1, 1985, pp. 31–55.

[21] Bóna (M.), Bousquet (M.), Labelle (G.) et Leroux (P.). – Enumeration of m-ary cacti. *Adv. in Appl. Math.*, vol. 24, n° 1, 2000, pp. 22–56.

[22] Bousquet (M.). – *Théorie des espèces et applications au dénombrement de cartes et de cactus planaires.* – Thèse de Doctorat, LaCIM, Université du Québec à Montréal, 1999.

[23] Bousquet (M.), Chauve (C.), Labelle (G.) et Leroux (P.). – A bijective proof of the arborescent form of the multivariable lagrange inversion formula. *In : Colloquium Mathematics and Computer Science.* Trends in mathematics, pp. 89–100. – Birkhäuser, 2000.

[24] Bousquet (M.), Chauve (C.) et Schaeffer (G.). – Enumération et génération aléatoire de cactus m-aires. *In : Colloque LaCIM 2000. Publications du LaCIM*, volume 27, pp. 81–91. – Université du Québec à Montréal, 2000.

[25] Bousquet-Mélou (M.) et Schaeffer (G.). – Enumeration of planar constellations. *Adv. in Appl. Math.*, vol. 24, n° 4, 2000, pp. 337–368.

[26] Boyer (M. S.) et Moore (J. S.). – A fast string-searching algorithm. *Comm. ACM*, vol. 20, 1977, pp. 762–772.

[27] Breslauer (D.). – The suffix tree of a tree and minimizing sequential transducers. *Theoret. Comput. Sci.*, vol. 191, n° 1-2, 1998, pp. 131–144.

[28] Cai (J.), Paige (R.) et Tarjan (R. E.). – More efficient bottom-up multi-pattern matching in trees. *Theoret. Comput. Sci.*, vol. 106, n° 1, 1992, pp. 21–60.

[29] Cayley (A.). – A theorem on trees. *Quart. J. Math.*, vol. 23, 1889, pp. 376–378, Collected Mathematical Papers of Arthur Cayley, vol XIII, p 26-28, Cambridge University Press, Cambridge (MA), 1897.

[30] Cayley (A.). – *Collected mathematical papers. Vol. I-XIII.* – Cambridge University Press, Cambridge (MA), 1897.

[31] Chaiken (S.). – A combinatorial proof of the all minors matrix tree theorem. *SIAM J. Algebraic Discrete Methods*, vol. 3, n° 3, 1982, pp. 319–329.

[32] Chang (W. I.) et Lawler (E. L.). – Approximate string matching in sublinear expected time. *In : IEEE Symposium on Foundations of Computer Science, FOCS'90.* pp. 116–124. – IEEE Comput. Soc. Press, 1990.

[33] Chase (D. R.). – An improvement to bottom-up tree pattern matching. *In : ACM Symposium on Principles of Programming Languages, POPL'87.* pp. 168–177. – Assoc. Comput. Mach. Press, 1987.

[34] Chauve (C.), Duchon (P.) et Dulucq (S.). – *Une généralisation des résultats de Kreweras sur le polynôme énumérateur des inversions dans les arborescences.* – Rapport

technique n° RR-1241-00, LaBRI, Université Bordeaux I, 2000. (à paraître dans Ann. Sci. Math. Québec).

[35] Chauve (C.), Dulucq (S.) et Guibert (O.). – *Enumeration of some labelled trees.* – Rapport technique n° RR-1226-99, LaBRI, Université Bordeaux I, 1999.

[36] Chauve (C.), Dulucq (S.) et Guibert (O.). – Enumeration of some labelled trees. *In : Formal Power Series and Algebraic Combinatorics, FPSAC'00.* pp. 146–157. – Springer, 2000.

[37] Chauve (C.), Dulucq (S.) et Rechnitzer (A.). – *Enumerating alternating trees.* – Rapport technique n° RR-1221-99, LaBRI, Université Bordeaux I, 1999. (à paraître dans J. Combin. Theory. Series A).

[38] Chen (W. Y. C.). – A general bijective algorithm for trees. *Proc. Nat. Acad. Sci. U.S.A.*, vol. 87, n° 24, 1990, pp. 9635–9639.

[39] Chen (W. Y. C.). – A bijection for enriched trees. *European J. Combin.*, vol. 15, n° 4, 1994, pp. 337–343.

[40] Chen (W. Y. C.). – The pessimistic search and the straightening involution for trees. *European J. Combin.*, vol. 19, n° 5, 1998, pp. 553–558.

[41] Chen (W. Y. C.). – A general bijective algorithm for increasing trees. *Systems Sci. Math. Sci.*, vol. 12, n° 3, 1999, pp. 193–203.

[42] Chottin (L.). – Une démonstration combinatoire de la formule de Lagrange à deux variables. *Discrete Math.*, vol. 13, n° 3, 1975, pp. 215–224.

[43] Chottin (L.). – Énumération d'arbres et formules d'inversion de séries formelles. *J. Combin. Theory Ser. B*, vol. 31, n° 1, 1981, pp. 23–45.

[44] Clarke (L. E.). – On Cayley's formula for counting trees. *J. London Math. Soc.*, vol. 33, 1958, pp. 471–474.

[45] Cobbs (A. L.). – Fast approximate matching using suffix trees. *In : Combinatorial Pattern Matching, CPM'95. Lecture Notes in Comput. Sci.*, volume 937, pp. 41–54. – Springer, 1995.

[46] Cole (R.) et Hariharan (R.). – Tree pattern matching and subset matching in randomized $O(n \log^3 m)$ time. *In : ACM Symposium on Theory of Computing, STOC'97.* pp. 66–75. – Assoc. Comput. Mach. Press, 1997.

[47] Cole (R.), Hariharan (R.) et Indyk (P.). – Tree pattern matching and subset matching in deterministic $O(n \log^3 n)$-time. *In : ACM-SIAM Symposium on Discrete Algorithms, SODA'99.* pp. 245–254. – Assoc. Comput. Mach. Press, 1999.

[48] Cori (R.). – *Un code pour les graphes planaires et ses applications.* – *Astérisque*, volume 27, Société Mathématique de France, 1975.

[49] Cori (R.). – Formules d'inversion de séries formelles. *In : Séries formelles en variables non commutatives et applications (5e École de Printemps Informat. Théorique).* pp. 55–67. – École Nat. Sup. Tech. Avancées, Paris, 1978.

[50] Cormen (T. H.), Leiserson (C. E.) et Rivest (R. L.). – *Introduction to algorithms.* – The MIT Press, Cambridge (MA), 1990.

[51] Crochemore (M.). – Transducers and repetitions. *Theoret. Comput. Sci.*, vol. 45, n° 1, 1986, pp. 63–86.

[52] Crochemore (M.) et Rytter (W.). – *Text algorithms.* – Oxford University Press, New York, 1994.

[53] Dénes (J.). – The representation of a permutation as the product of a minimal number of transpositions, and its connection with the theory of graphs. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, vol. 4, 1959, pp. 63–71.

[54] Dubiner (M.), Galil (Z.) et Magen (E.). – Faster tree pattern matching. *J. Assoc. Comput. Mach.*, vol. 41, n° 2, 1994, pp. 205–213.

[55] Dublish (P.). – Some comments on: "On the subtree isomorphism problem for ordered trees" by E. Mäkinen. *Inform. Process. Lett.*, vol. 36, n° 5, 1990, pp. 273–275.

[56] Eden (M.) et Schützenberger (M. P.). – Remark on a theorem of Dénes. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, vol. 7, 1962, pp. 353–355.

[57] Ehrenborg (R.) et Méndez (M.). – A bijective proof of infinite variated Good's inversion. *Adv. in Math.*, vol. 103, n° 2, 1994, pp. 221–259.

[58] El Marraki (M.), Hanusse (N.), Zipperer (J.) et Zvonkin (A.). – Cacti, braids and complex polynomials. *Sém. Lothar. Combin.*, vol. 37, 1996, Art. B37b, 36 pp.

[59] Entringer (R. C.). – A combinatorial interpretation of the Euler and Bernoulli numbers. *Nieuw Arch. Wisk. (3)*, vol. 14, 1966, pp. 241–246.

[60] Farach (M.). – Optimal suffix tree construction with large alphabets. *In: IEEE Symposium on Foundations of Computer Science, FOCS'97.* pp. 137–143. – IEEE Comput. Soc. Press, 1997.

[61] Flajolet (P.) et Sedgewick (R.). – *The average case analysis of algorithms: counting and generating functions.* – Rapport technique n° 1888, INRIA, 1993.

[62] Gessel (I. M.). – A combinatorial proof of the multivariable Lagrange inversion formula. *J. Combin. Theory Ser. A*, vol. 45, n° 2, 1987, pp. 178–195.

[63] Gessel (I. M.) et Sagan (B. E.). – The Tutte polynomial of a graph, depth-first search, and simplicial complex partitions. *Electron. J. Combin.*, vol. 3, n° 2, 1996, Research Paper 9, 36 pp.

[64] Gessel (I. M.), Sagan (B. E.) et Yeh (Y. N.). – Enumeration of trees by inversions. *J. Graph Theory*, vol. 19, n° 4, 1995, pp. 435–459.

[65] Gessel (I. M.) et Wang (D. L.). – Depth-first search as a combinatorial correspondence. *J. Combin. Theory Ser. A*, vol. 26, n° 3, 1979, pp. 308–313.

[66] Giancarlo (R.). – A generalization of the suffix tree to square matrices, with applications. *SIAM J. Comput.*, vol. 24, n° 3, 1995, pp. 520–562.

[67] Giancarlo (R.) et Grossi (R.). – On the construction of classes of suffix trees for square matrices: algorithms and applications. *Inform. and Comput.*, vol. 130, n° 2, 1996, pp. 151–182.

[68] Giancarlo (R.) et Grossi (R.). – Parallel construction and query of index data structures for pattern matching on square matrices. *J. Complexity*, vol. 15, n° 1, 1999, pp. 30–71.

[69] Giancarlo (R.) et Guaiana (D.). – On-line construction of two-dimensional suffix trees. *J. Complexity*, vol. 15, n° 1, 1999, pp. 72–127.

[70] Giegerich (R.) et Kurtz (S.). – A comparison of imperative and purely functional suffix tree constructions. *Sci. Comput. Programming*, vol. 25, n° 2-3, 1995, pp. 187–218.

[71] Giegerich (R.) et Kurtz (S.). – From Ukkonen to McCreight and Weiner: a unifying view of linear-time suffix tree construction. *Algorithmica*, vol. 19, n° 3, 1997, pp. 331–353.

[72] Good (I. J.). – Generalizations to several variables of Lagrange's expansion, with applications to stochastic processes. *Proc. Cambridge Philos. Soc.*, vol. 56, 1960, pp. 367–380.

[73] Good (I. J.). – The generalization of Lagrange's expansion and the enumeration of trees. *Proc. Cambridge Philos. Soc.*, vol. 61, 1965, pp. 499–517.

[74] Good (I. J.). – Correction: "The generalization of Lagrange's expansion and the enumeration of trees". *Proc. Cambridge Philos. Soc.*, vol. 64, 1968, p. 489.

[75] Goulden (I. P.) et Jackson (D. M.). – The enumeration of directed closed Euler trails and directed Hamiltonian circuits by Lagrangian methods. *European J. Combin.*, vol. 2, n° 2, 1981, pp. 131–135.

[76] Goulden (I. P.) et Jackson (D. M.). – The application of Lagrangian methods to the enumeration of labelled trees with respect to edge partition. *Canad. J. Math.*, vol. 34, n° 3, 1982, pp. 513–518.

[77] Goulden (I. P.) et Jackson (D. M.). – *Combinatorial enumeration.* – John Wiley & Sons, New York, 1983.

[78] Goulden (I. P.) et Jackson (D. M.). – The combinatorial relationship between trees, cacti and certain connection coefficients for the symmetric group. *European J. Combin.*, vol. 13, n° 5, 1992, pp. 357–365.

[79] Goulden (I. P.) et Jackson (D. M.). – Transitive factorisations into transpositions and holomorphic mappings on the sphere. *Proc. Amer. Math. Soc.*, vol. 125, n° 1, 1997, pp. 51–60.

[80] Goulden (I. P.) et Kulkarni (D. M.). – Multivariable Lagrange inversion, Gessel-Viennot cancellation, and the matrix tree theorem. *J. Combin. Theory Ser. A*, vol. 80, n° 2, 1997, pp. 295–308.

[81] Goulden (I. P.) et Pepper (S.). – Labelled trees and factorizations of a cycle into transpositions. *Discrete Math.*, vol. 113, n° 1-3, 1993, pp. 263–268.

[82] Graham (R. L.), Knuth (D. E.) et Patashnik (O.). – *Concrete mathematics.* – Seconde édition, Addison-Wesley Publishing Company, Reading, (MA), 1994.

[83] Grossi (R.). – Further comments on the subtree isomorphism for ordered trees: "On the subtree isomorphism problem for ordered trees" by E. Mäkinen. *Inform. Process. Lett.*, vol. 40, n° 5, 1991, pp. 255–256.

[84] Grossi (R.). – A note on the subtree isomorphism for ordered trees and related problems. *Inform. Process. Lett.*, vol. 39, n° 2, 1991, pp. 81–84.

[85] Grossi (R.), Luccio (F.) et Pagli (L.). – Coding trees as strings for approximate tree matching. *In : Sequences, II (Positano, 1991).* pp. 245–259. – Springer, 1993.

[86] Gusfield (D.). – *Algorithms on strings, trees and sequences.* – Cambridge University Press, Cambridge (MA), 1997.

[87] Hanusse (N.). – *Cartes, constellations et groupes : questions algorithmiques.* – Thèse de Doctorat, LaBRI, Université Bordeaux I, 1997.

[88] Harary (F.) et Palmer (E. M.). – *Graphical enumeration.* – Academic Press, New York, 1973.

[89] Harary (F.) et Uhlenbeck (G. E.). – On the number of Husimi trees. *Proc. Nat. Acad. Sci. U.S.A.*, vol. 39, 1953, pp. 315–322.

[90] Harel (D.) et Tarjan (R. E.). – Fast algorithms for finding nearest common ancestors. *SIAM J. Comput.*, vol. 13, n° 2, 1984, pp. 338–355.

[91] Hoffmann (C. M.) et O'Donnell (M. J.). – Pattern matching in trees. *J. Assoc. Comput. Mach.*, vol. 29, n° 1, 1982, pp. 68–95.

[92] Hurwitz (A.). – Über Riemann'sche Flächen mit gegebenen Verzeigungspunkten. *Math. Ann.*, vol. 39, 1891, pp. 1–66.

[93] Husimi (K.). – Note on Mayer's theory of cluster integrals. *J. Chem. Phys.*, vol. 18, 1950, pp. 682–684.

[94] Indyk (P.). – Deterministic superimposed coding with applications to pattern matching. *In : IEEE Symposium on Foundations of Computer Science, FOCS'97.* pp. 127–136. – IEEE Comput. Soc. Press, 1997.

[95] Jackson (D. M.) et Goulden (I. P.). – The generalisation of Tutte's result for chromatic trees, by Lagrangian methods. *Canad. J. Math.*, vol. 33, n° 1, 1981, pp. 12–19.

[96] Joyal (A.). – Une théorie combinatoire des séries formelles. *Adv. in Math.*, vol. 42, n° 1, 1981, pp. 1–82.

[97] Kalikow (L. H.). – *Enumeration of parking functions, allowable permutation pairs, and labeled trees.* – Thèse de Doctorat, Brandeis University, 1999.

[98] Kärkkäinen (J.). – Suffix cactus: A cross between suffix tree and suffix array. *In : Combinatorial Pattern Matching, CPM'95. Lecture Notes in Comput. Sci.*, volume 937, pp. 191–204. – Springer, 1995.

[99] Kärkkäinen (J.) et Ukkonen (E.). – Sparse suffix trees. *In : Computing and Combinatorics, COCOON'96. Lecture Notes in Comput. Sci.*, volume 1090, pp. 219–230. – Springer, 1996.

[100] Khidr (A. M.) et El-Desouky (B. S.). – A symmetric sum involving the Stirling numbers of the first kind. *European J. Combin.*, vol. 5, n° 1, 1984, pp. 51–54.

[101] Knuth (D. E.). – Another enumeration of trees. *Canad. J. Math.*, vol. 20, 1968, pp. 1077–1086.

[102] Knuth (D. E.). – *The art of computer programming.* – Troisième édition, Addison-Wesley Publishing Company, Reading (MA), 1997. Volume 1: Fundamental algorithms.

[103] Knuth (D. E.). – *The art of computer programming.* – Troisième édition, Addison-Wesley Publishing Company, Reading (MA), 1997. Volume 3: Sorting and Searching.

[104] Knuth (D. E.), Morris (J. H. Jr.) et Pratt (V. R.). – Fast pattern matching in strings. *SIAM J. Comput.*, vol. 6, n° 2, 1977, pp. 323–350.

[105] Kosaraju (S. R.). – Efficient tree pattern matching. *In : IEEE Symposium on Foundations of Computer Science, FOCS'89.* pp. 178–183. – IEEE Comput. Soc. Press, 1989.

[106] Kreweras (G.). – Une famille de polynômes ayant plusieurs propriétés énumératives. *Period. Math. Hungar.*, vol. 11, n° 4, 1980, pp. 309–320.

[107] Kuznetsov (A. G.), Pak (I.) et Postnikov (A.). – Increasing trees and alternating permutations. *Russian Math. Survey*, vol. 49, 1994, pp. 79–110.

[108] Labelle (G.). – Une nouvelle démonstration combinatoire des formules d'inversion de Lagrange. *Adv. in Math.*, vol. 42, n° 3, 1981, pp. 217–247.

[109] Landau (G. M.) et Vishkin (U.). – *Pattern matching algorithms*, chap. Approximate string searching, pp. 185–199. – In Apostolico et Galil [7].

[110] Leroux (P.). – L'inversion de Lagrange à plusieurs variables : l'approche combinatoire des espèces de structures. – Février 1981. manuscrit non publié.

[111] Liskovets (V. A.). – A census of nonisomorphic planar maps. *In : Algebraic methods in graph theory. Colloq. Math. Soc. János Bolyai*, volume 25, pp. 479–494. – North-Holland, 1981.

[112] Luccio (F.) et Pagli (L.). – An efficient algorithm for some tree matching problems. *Inform. Process. Lett.*, vol. 39, n° 1, 1991, pp. 51–57.

[113] Luccio (F.) et Pagli (L.). – Simple solutions for approximate tree matching problems. *In: Theory and Practice of Software, TAPSOFT '91. Lecture Notes in Comput. Sci.*, volume 493, pp. 193–201. – Springer, 1991.

[114] Luccio (F.) et Pagli (L.). – Approximate matching for two families of trees. *Inform. and Comput.*, vol. 123, n° 1, 1995, pp. 111–120.

[115] Mäkinen (E.). – On the subtree isomorphism problem for ordered trees. *Inform. Process. Lett.*, vol. 32, n° 5, 1989, pp. 271–273.

[116] Mallows (C. L.) et Riordan (J.). – The inversion enumerator for labeled trees. *Bull. Amer. Math. Soc.*, vol. 74, 1968, pp. 92–94.

[117] Manber (U.) et Myers (G.). – Suffix arrays: a new method for on-line string searches. *SIAM J. Comput.*, vol. 22, n° 5, 1993, pp. 935–948.

[118] McCreight (E. M.). – A space-economical suffix tree construction algorithm. *J. Assoc. Comput. Mach.*, vol. 23, n° 2, 1976, pp. 262–272.

[119] Moon (J. W.). – Various proofs of Cayley's formula for counting trees. *In: A seminar on Graph Theory.* pp. 70–78. – Rinehart and Winston, 1967.

[120] Moon (J. W.). – *Counting labelled trees.* – *Canadian Mathematical Monographs*, volume 1, Canadian Mathematical Congress, Montréal, 1970.

[121] Moszkowski (P.). – Arbres et suites majeures. *Period. Math. Hungar.*, vol. 20, n° 2, 1989, pp. 147–154.

[122] Moszkowski (P.). – A solution to a problem of Dénes: a bijection between trees and factorizations of cyclic permutations. *European J. Combin.*, vol. 10, n° 1, 1989, pp. 13–16.

[123] Pansiot (J.-J.). – Nombres d'Euler et inversions dans les arbres. *European J. Combin.*, vol. 3, n° 3, 1982, pp. 259–262.

[124] Postnikov (A.). – *Enumeration in algebra and geometry.* – Thèse de Doctorat, M.I.T., 1997.

[125] Postnikov (A.). – Intransitive trees. *J. Combin. Theory Ser. A*, vol. 79, n° 2, 1997, pp. 360–366.

[126] Poulalhon (D.). – *Graphes et décompositions de permutations.* – Mémoire de DEA, LIX, Ecole Polytechnique, 1997.

[127] Poupard (C.). – De nouvelles significations énumératives des nombres d'Entringer. *Discrete Math.*, vol. 38, n° 2-3, 1982, pp. 265–271.

[128] Poupard (C.). – Deux propriétés des arbres binaires ordonnés stricts. *European J. Combin.*, vol. 10, n° 4, 1989, pp. 369–374.

[129] Prüfer (H.). – Never Beweis eines Satzes über Permutationen. *Arch. Math. Phys. Sci.*, vol. 27, 1918, pp. 742–744.

[130] Ramesh (R.) et Ramakrishnan (I. V.). – Nonlinear pattern matching in trees. *J. Assoc. Comput. Mach.*, vol. 39, n° 2, 1992, pp. 295–316.

[131] Raney (G. N.). – Functional composition patterns and power series reversion. *Trans. Amer. Math. Soc.*, vol. 94, 1960, pp. 441–451.

[132] Riordan (J.). – Forests of label-increasing trees. *J. Graph Theory*, vol. 3, n° 2, 1979, pp. 127–133.

[133] Rodeh (M.), Pratt (V. R.) et Even (S.). – Linear algorithm for data compression via string matching. *J. Assoc. Comput. Mach.*, vol. 28, n° 1, 1981, pp. 16–24.

[134] Schaeffer (G.). – Bijective census and random generation of Eulerian planar maps with prescribed vertex degrees. *Electron. J. Combin.*, vol. 4, n° 1, 1997, Research Paper 20, 14 pp.

[135] Schaeffer (G.). – *Conjugaison d'arbres et cartes combinatoires aléatoires.* – Thèse de Doctorat, LaBRI, Université Bordeaux I, 1999.

[136] Schaeffer (G.). – Random sampling of large planar maps and convex polyhedra. *In: ACM Symposium on Theory of Computing, STOC'99.* pp. 760–769. – Assoc. Comput. Mach. Press, 1999.

[137] Schieber (B.) et Vishkin (U.). – On finding lowest common ancestors: simplification and parallelization. *SIAM J. Comput.*, vol. 17, n° 6, 1988, pp. 1253–1262.

[138] Scoins (H. I.). – The number of trees with nodes of alternate parity. *Proc. Cambridge Philos. Soc.*, vol. 58, 1962, pp. 12–16.

[139] Sedgewick (R.) et Flajolet (P.). – *An introduction to the analysis of algorithms.* – Addison-Wesley Publishing Company, Reading (MA), 1996.

[140] Shi (J. Y.). – *The Kazhdan-Lusztig cells in certain affine Weyl groups.* – *Lecture Notes in Math.*, volume 1179, Springer, 1986.

[141] Shibuya (T.). – Constructing the suffix tree of a tree with a large alphabet. *In: International Symposium on Algorithms and Computation, ISAAC'99. Lecture Notes in Comput. Sci.*, volume 1741, pp. 225–236. – Springer, 1999.

[142] Shor (P. W.). – A new proof of Cayley's formula for counting labeled trees. *J. Combin. Theory Ser. A*, vol. 71, n° 1, 1995, pp. 154–158.

[143] Sloane (N. J. A.). – An on-line version of the encyclopedia of integer sequences. *Electron. J. Combin.*, vol. 1, 1994, Feature 1, 5 pp.

[144] Sloane (N. J. A.) et Plouffe (S.). – *The encyclopedia of integer sequences.* – Academic Press, San Diego, 1995.

[145] Stanley (R. P.). – Hyperplane arrangements, interval orders, and trees. *Proc. Nat. Acad. Sci. U.S.A.*, vol. 93, n° 6, 1996, pp. 2620–2625.

[146] Stanley (R. P.). – *Enumerative combinatorics. Vol. 1.* – *Cambridge Studies in Advanced Mathematics*, volume 49, Cambridge University Press, Cambridge (MA), 1997.

[147] Stanley (R. P.). – Hyperplane arrangements, parking functions and tree inversions. *In: Mathematical essays in honor of Gian-Carlo Rota. Progr. Math.*, volume 161, pp. 359–375. – Birkhäuser, 1998.

[148] Stanley (R. P.). – *Enumerative combinatorics. Vol. 2.* – *Cambridge Studies in Advanced Mathematics*, volume 62, Cambridge University Press, Cambridge (MA), 1999.

[149] Steyaert (JM.) et Flajolet (P.). – Patterns and pattern-matching in trees: an analysis. *Inform. and Control*, vol. 58, n° 1-3, 1983, pp. 19–58.

[150] Strehl (V.). – Zykel-Enumeration bei Lokal-Strukturierten Funktionen. – Habilitationsschrift, Institut für Mathematische Maschinen und Datenverarbeitung der Universität Erlangen-Nürnberg, Allemagne, 1989.

[151] Strehl (V.). – Minimal transitive products of transpositions—the reconstruction of a proof of A. Hurwitz. *Sém. Lothar. Combin.*, vol. 37, 1996, Art. S37c, 12 pp.

[152] Tai (K. C.). – The tree-to-tree correction problem. *J. Assoc. Comput. Mach.*, vol. 26, n° 3, 1979, pp. 422–433.

[153] Takács (L.). – Counting forests. *Discrete Math.*, vol. 84, n° 3, 1990, pp. 323–326.

[154] Takács (L.). – On Cayley's formula for counting forests. *J. Combin. Theory Ser. A*, vol. 53, n° 2, 1990, pp. 321–323.

[155] Thorup (M.). – Efficient preprocessing of simple binary pattern forests. *J. Algorithms*, vol. 20, n° 3, 1996, pp. 602–612.

[156] Tutte (W. T.). – A contribution to the theory of chromatic polynomials. *Canadian J. Math.*, vol. 6, 1954, pp. 80–91.

[157] Tutte (W. T.). – A census of slicings. *Canad. J. Math.*, vol. 14, 1962, pp. 708–722.

[158] Tutte (W. T.). – The number of planted plane trees with a given partition. *Amer. Math. Monthly*, vol. 71, 1964, pp. 272–277.

[159] Ukkonen (E.). – Approximate string-matching over suffix trees. *In : Combinatorial Pattern Matching, CPM'93. Lecture Notes in Comput. Sci.*, volume 684, pp. 228–242. – Springer, 1993.

[160] Ukkonen (E.). – On-line construction of suffix trees. *Algorithmica*, vol. 14, n° 3, 1995, pp. 249–260.

[161] Verma (R. M.). – Strings, trees, and patterns. *Inform. Process. Lett.*, vol. 41, n° 3, 1992, pp. 157–161.

[162] Viennot (G.). – Une interprétation combinatoire des coefficients des développements en série entière des fonctions elliptiques de Jacobi. *J. Combin. Theory Ser. A*, vol. 29, n° 2, 1980, pp. 121–133.

[163] Walsh (T. R. S.). – Hypermaps versus bipartite maps. *J. Combin. Theory Ser. B*, vol. 18, 1975, pp. 155–163.

[164] Weiner (P.). – Linear pattern matching algorithms. *In : 14th Annual IEEE Symposium on Switching and Automata Theory*. pp. 1–11. – IEEE Comput. Soc. Press, 1973.

[165] Wilf (H. S.). – *Generatingfunctionology*. – Seconde édition, Academic Press, Boston, 1994.

[166] Yan (C. H.). – Generalized tree inversions and k-parking functions. *J. Combin. Theory Ser. A*, vol. 79, n° 2, 1997, pp. 268–280.

[167] Zhang (K.) et Shasha (D.). – Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, vol. 18, n° 6, 1989, pp. 1245–1262.

[168] Zvonkin (D.), 2000. Communication personnelle.

Structures arborescentes : problèmes algorithmiques et combinatoires

Résumé : La première partie de ce mémoire est consacrée à l'énumération de diverses familles de structures arborescentes, en général selon le nombre de sommets. Les trois premiers chapitres sont consacrés à l'étude des arborescences de Cayley telles que la racine est inférieure à ses fils et des arborescences alternantes. La plupart de nos résultats sont prouvés bijectivement. Nous nous intéressons ensuite aux arborescences coloriées, et plus particulièrement à la formule d'inversion de séries formelles multivariées de Good-Lagrange. Nous donnons une nouvelle preuve bijective d'une variante de cette formule et utilisons cette preuve pour prouver combinatoirement diverses formules d'énumération de structures arborescentes et en déduire des algorithmes de génération aléatoire pour ces structures (notamment les cactus planaires). Nous concluons cette première partie par un chapitre consacré aux constellations : en combinant notre preuve de la formule de Good-Lagrange et la conjugaison d'arborescences (due à Bousquet-Mélou et Schaeffer), nous prouvons bijectivement une formule (nouvelle) pour l'énumération de constellations selon le nombre de sommets et de faces.

Dans la seconde partie, nous étudions le problème de la recherche de motifs dans une arborescence, en utilisant une structure de données classique pour les mots : l'arborescence des suffixes. Nous proposons notamment un algorithme de recherche de motifs dans une arborescence, basé sur un codage d'une arborescence par des mots et sur l'utilisation de l'arborescence des suffixes d'un de ces mots, qui semble avoir de bonnes propriétés expérimentales. Nous concluons en étendant la notion d'arborescence des suffixes des mots aux arborescences et en décrivant un algorithme de construction pour cette structure.

Mots-clés : Combinatoire, algorithmique, arborescences, énumération, bijections, génération aléatoire, séries formelles multivariées, recherche de motifs, arborescence des suffixes.

Arborescent structures: algorithmic and combinatorial problems

Abstract : The first part is devoted to enumerative combinatorics. In the third first chapters, we study the families of Cayley trees such that the root is lower than its sons under some combinatorial parameters (number of vertices, label of the root, inversion polynomial), and of alternating trees. Most of our proofs are based on bijections. In the next chapter, we are interested in the enumeration of colored trees, with the formula of Good-Lagrange (inversion of multivariated formal power series). We give a new bijective proof of a variant of this formula and apply this proof in the enumeration and random generation of arborescent structures (like planar cacti). We conclude this part by a proof of a (new) formula for the enumeration of constellation under the number of vertices and faces.

In the second part, we study the problem of tree pattern matching, using a classical data structure (for words): the suffix tree. We propose a new tree pattern matching algorithm, based on encoding a tree with words and using the suffix tree of one of these words, which seems to have good experimental properties. We conclude by proposing an notion of suffix tree of a tree and an algorithme computing such a structure.

Keywords: Combinatorics, algorithmic, trees, enumeration, bijection, random generation, multivariated formal power series, tree pattern matching, suffix tree.

Thèse en informatique, préparée au LaBRI (Université Bordeaux I, 351 cours de la Libération, 33405 Talence).