Presentation et evaluation de la complexite en moyenne d'algorithmes de filtrage dans les moteurs d'inferences (algorithmes RETE et arbre d'unification)

Luc Albert

▶ **To cite this version:**

Luc Albert. Presentation et evaluation de la complexite en moyenne d'algorithmes de filtrage dans les moteurs d'inferences (algorithmes RETE et arbre d'unification). [Rapport de recherche] RR-0837, INRIA. 1988. inria-00075716

HAL Id: inria-00075716
https://inria.hal.science/inria-00075716

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France

Tél. (1) 39 63 55 11

Rapports de Recherche

N° 837

PRESENTATION ET EVALUATION DE LA COMPLEXITE EN MOYENNE D'ALGORITHMES DE FILTRAGE DANS LES MOTEURS D'INFERENCES (ALGORITHMES RETE ET ARBRE D'UNIFICATION)

Luc ALBERT

AVRIL 1988

RR 837

PRESENTATION ET EVALUATION DE LA COMPLEXITE EN MOYENNE D'ALGORITHMES DE FILTRAGE DANS LES MOTEURS D'INFERENCES (ALGORITHMES RETE ET ARBRE D'UNIFICATION)

LUC ALBERT[1]

Résumé. *Le formalisme des Systèmes de Règles de Productions est très utilisé aujourd'hui en Intelligence Artificielle pour la représentation des connaissances et la création de Systèmes Experts. L'efficacité de la phase de pattern-matching (ou semi-unification) de l'interpréteur de règles de production (ou Moteur d'Inférences) est fondamentale dans leur bon fonctionnement. L'algorithme* RETE *de multi-Pattern Matching [Forg 82] est un algorithme efficace pour effectuer cette comparaison d'un grand nombre de motifs avec un grand nombre d'objets. Il produit toutes les combinaisons d'objets qui s'unifient avec une conjonction de motifs. Cet algorithme est très utilisé pour améliorer les performances des systèmes experts à base de règles. Dans cet article, on emploie la théorie des séries génératrices afin de déterminer un coût en moyenne précis de l'algorithme* RETE. *On établit tout d'abord des résultats avec un modèle de termes aléatoires, puis on étend ce modèle afin de prendre en compte les différentes fréquences d'apparition des symboles de façon à modéliser fidèlement les applications réelles. Ensuite on présente une version de l'algorithme* RETE *basée sur l'*ARBRE D'UNIFICATION *et on analyse de même sa complexité. Enfin on applique cette étude à un exemple, les résultats numériques étant obtenus à l'aide du calculateur formel MAPLE.*

SEMI-UNIFICATION ALGORITHMS (RETE AND ARBRE D'UNIFICATION) : AN AVERAGE CASE ANALYSIS

Abstract. *Production systems, or more generally rule-based systems, are widely used in Artificial Intelligence for representing knowledge and building expert systems. The efficiency of the pattern match phase of the production system interpreter is an essential condition of its "well-running". The* RETE *multi-pattern match algorithm [Forg 82] is an efficient method for comparing a large collection of patterns to a large collection of objects. It finds all the combinations of objects which match with a conjunction of patterns. This algorithm is widely used to improve the run-time performance of rule-based expert systems. In this paper we employ generating functions theory in order to perform a precise average case complexity analysis of* RETE *algorithm. Our results are first established under a "simple" random term model, and later extended to take into account different frequency coefficients for symbols in a way that can closely model real-life applications. Then we present a variant of the* RETE *algorithm based on the "*ARBRE D'UNIFICATION*" and we present its complexity analysis with the same methods. Lastly, we apply our results to an example using the symbolic manipulation program MAPLE to obtain numerical values.*

[1] Institut National de Recherche en Informatique et Automatique, Domaine de Voluceau, Rocquencourt, BP 105, 78150 Le Chesnay Cedex France

1.INTRODUCTION

L'algorithme de multi-Pattern Matching RETE [Forg 79] [Forg 82] a été introduit par C. Forgy à la suite de ses travaux sur les Systèmes de Règles de Production [Forg 81].

Les Systèmes de Règles de Production sont très utilisés en Intelligence Artificielle pour modéliser un comportement intelligent [LNR 87] et réaliser des systèmes experts. Ils sont d'une utilisation assez souple et possèdent de nombreux avantages: modularité et indépendance relative des règles et grande puissance expressive (ils possèdent la généralité d'une machine de Turing). Toutefois, en contrepartie, le principal inconvénient réside dans l'inefficacité algorithmique des Moteurs d'Inférences. La phase la plus coûteuse dans un Moteur d'Inférences est la phase de Pattern Matching qui consiste à maintenir l'ensemble de toutes les instanciations des règles satisfaites, ou Ensemble de Conflit, durant les changements de la Base de Faits. Le système peut consacrer plus de 90 % de son temps à cette opération [DNM 78].

L'algorithme RETE détermine de façon incrémentale l'Ensemble de Conflit. En effet, pour les applications des systèmes experts le déclenchement d'une règle affecte un petit nombre de faits en comparaison avec le nombre de faits total de la Base de Faits. Cette détermination est donc efficace car il n'y a pas à chaque cycle du Moteur d'Inférences d'itération sur l'ensemble de la Base de Faits. D'autre part les conditions communes à plusieurs règles, qui sont fréquentes (en définissant une égalité au renommage des variables près), sont partagées de façon à ce qu'en testant seulement quelques conditions on puisse déterminer la satisfiabilité de plusieurs règles.

Forgy a montré, sous certaines hypothèses simplificatrices, dans [Forg 79] que la complexité dans le cas le pire pour déterminer l'ensemble des règles instanciées de l'algorithme RETE est linéaire en fonction du nombre de règles de la Base de Règles et polynomial en le nombre de faits de la Base de Faits (avec pour degré le nombre supposé constant de conditions dans une règle). Dans le meilleur des cas la complexité en temps est constante. Entre ces deux extrèmes la complexité en temps dépend complètement des caractéristiques des règles.

L'analyse en moyenne précise de l'algorithme RETE présente de nombreux intérêts. Tout d'abord, l'algorithme RETE admet beaucoup de variantes et d'optimisations: en ce qui concerne la représentation des mémoires locales [Forg 79], le calcul des jointures [Mir 87], la compilation totale du graphe [Fag 86], la parallelisation de l'algorithme [Gupt 84], le partage des conditions (qui conduit à l'établissement de l'arbre d'unification que nous étudierons par la suite) [Gha 87], etc ... On peut utiliser une analyse en moyenne de l'algorithme pour évaluer ces différentes optimisations et même en proposer de nouvelles. Ensuite, il est nécessaire de pouvoir prédire le temps d'exécution de l'algorithme pour le développement de systèmes experts temps réels [WGF 86], [SF 83]. Avec un modèle mathématique du temps d'exécution on peut extrapoler les performances d'un prototype aux performances d'un système expert grandeur nature et définir son domaine d'applicabilité en fonction du nombre d'objets qui peuvent être traités à un instant donné. Enfin à l'aide d'un modèle mathématique peuvent être conçus des benchmarks significatifs pour comparer différentes implémentations en fonction des paramètres caractéristiques d'une base de connaissances [GF 83].

Dans ce papier nous présentons l'analyse en moyenne de l'algorithme RETE classique et de la variante comportant l'ARBRE D'UNIFICATION. C'est, à notre connaissance, une des premières fois, qu'un algorithme de ce type utilisé en IA est analysé totalement et précisément. De plus les méthodes mathématiques développées dans cet article ne sont pas seulement intéressantes pour leur résultats. L' étude de la complexité en moyenne à l'aide de la théorie des familles simples d'arbres et du modèle pondéré est intéressante en elle-même [Alb 87].

Nous représentons les faits (ou *termes*) sous leur forme arborescentes et les conditions (ou *motifs* en anglais *patterns*) par des arbres avec variables (ce qui constitue une généralisation des

PAPIER RECUPERÉ ET RECYCLÉ

termes et des faits acceptés par OPSV, système développé par Forgy). On ne considerera que des tests d'égalité. Dans la partie 2, on présente de façon générale l'algorithme RETE. Ensuite dans la partie 3, on introduit la théorie des fonctions génératrices qui est employée pour analyser la complexité en moyenne des algorithmes. On obtient alors un résultat avec le modèle des familles simples d'arbres (3.4). Dans la partie 4, on expose les résultats obtenus en prenant en compte des fréquences d'apparitions différentes pour les symboles. Puis dans la section 5 on étend les résultats en considérant la négation. Enfin dans la partie 6, on présente la variante de l'algorithme RETE utilisant l'ARBRE D'UNIFICATION que l'on étudie également. On trouvera en Annexe une étude numérique complète menée sur un exemple précis.

2. L'ALGORITHME RETE

Les Systèmes de Règles de Production que nous allons considérer par la suite sont composés d'un ensemble fixe, noté BR (pour Base de Règles), d'assertions du type *"si-alors"* appelées *règles de production*, et d'un ensemble variable de *faits* appelé Base de Faits ou BF. On représente les faits sous la forme d'arbres formés à partir d'un alphabet fini F de symboles fonctionnels donnés avec leur arité. Les constantes sont des symboles d'arités 0. Par exemple, étant donnés le symbole f d'arité 2, g d'arité 1 et une constante a on peut former les termes suivants: a, $(p\ \ a)$, $(f\ \ a\ a)$, $(f\ \ (p\ \ a)\ a)$, etc ... On note l'ensemble des termes ainsi formés $T(F)$. La Base de Faits est représentée par un uplet de tels termes.

La partie $-si-$ d'une règle (son *membre gauche*) est une conjonction de conditions (ou *motifs*) représentée par le uplet $(P_1, \ldots, P_n)$. Un motif est un arbre dont certaines des feuilles peuvent être variables. Les variables forment un ensemble dénombrable V et sont notées par $x, y, \ldots$ L'ensemble des motifs formés à partir de F et V est noté $T(F,V)$. Les conditions filtrent les faits (*termes*). Plus précisement un terme t s'unifie avec un motif P si l'on peut trouver une *substitution* sur les variables du motif, $\sigma : V \rightarrow T(F)$, telle que $\sigma P = t$. Par exemple le motif $(f\ \ (p\ \ x)\ x)$ dans lequel on substitue x par $(p\ \ b)$ s'unifie avec le terme $(f\ \ (p\ \ (p\ \ b))\ (p\ \ b))$.

On dit que le membre gauche d'une règle $(P_1, \ldots, P_n)$ est *instancié* (ou que la règle est *satisfaite*), s'il existe un uplet de termes $(t_1, \ldots, t_n)$ avec $t_i \in BF$, appelé l'instance, et une substitution σ telle que $\sigma P_i = t_i$. Remarquons que puisqu'un motif dans une règle peut filtrer plusieurs faits de la Base de Faits, une règle peut être instanciée de multiples façons. La partie $-alors-$ d'une règle (son *membre droit*) est une suite d'actions qui peuvent avoir comme effet d'ajouter ou de supprimer un (ou plusieurs) fait de la Base de Faits.

Exemple : Soient les symboles fonctionnels h d'arité 3, f et g d'arité 2 et les constantes $a0$, $a1$, $a2$, $a3$, $a4$, $a5$. On peut considérer les 3 règles suivantes, où l'on a représenté les variables par des majuscules et omis les membres droits :

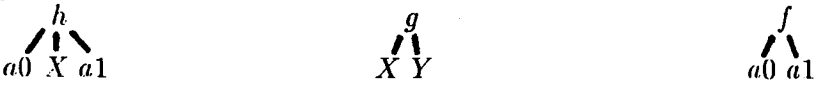
$$(R1 : \quad (g\ \ X\ Y)\ \neg(f\ \ a0\ X)\ (h\ \ a1\ X\ Y) \longrightarrow \ \ldots)$$

$$(R2 : \quad (h\ \ a0\ X\ a1)\ (g\ \ X\ Y)\ (f\ \ a0\ a1) \longrightarrow \ \ldots)$$

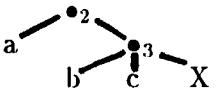
$$(R3 : \quad (f\ \ X\ a1)\ (f\ \ Y\ a2)\ (h\ \ a2\ X\ Y) \longrightarrow \ \ldots)$$

(La négation "¬" est étudiée au 5.)

Les motifs-condition de la règle $R2$ sont donc $(h\ \ a0\ X\ a1)$, $(g\ \ X\ Y)$ et $(f\ \ a0\ a1)$ ou sous leur forme arborescente :



Signalons que l'on peut représenter une liste imbriquée $(a\ \ (b\ c\ X))$ par:



Le cycle d'inférences exécute les trois opérations suivantes:

- Il *unifie* les motifs avec les faits de la Base de Faits afin de déterminer l'ensemble des règles instanciées (l' *Ensemble de Conflit*), c'est la phase de *pattern-matching.*
- Il *sélectionne* une instance particulière d'une règle dans l'Ensemble de Conflit;
- Il *déclanche* les actions de la règle choisie.

En fait, dans l'algorithme RETE la phase de déclanchement n'est pas séparée de la phase de pattern matching. Celle-ci est exécutée à chaque modification de la Base de Faits, c-à-d à l'initialisation du système quand on entre les faits puis à chaque ajout ou suppression durant l'étape de déclanchement. Puisque l'algorithme de multi-Pattern Matching RETE s'intéresse uniquement aux membres gauches des règles, on identifiera à présent les règles avec leur membre gauche. Les règles sont représentées sous la forme d'un graphe de discrimination : le GRAPHE DE RETE. Pendant l'exécution, quand un ajout ou une suppression est réalisé, un élément de modification parcourt le graphe depuis sa racine. Dès que celui-ci s'unifie avec un motif il est mémorisé (ou enlevé) du graphe et si d'autres faits mémorisés s'unifient avec lui à un membre gauche de règle, l'instance est ajoutée (ou supprimée) de l'Ensemble de Conflit. En ce sens la Base de Règles et la Base de Faits sont toutes les deux représentées dans le graphe.

On peut distinguer deux types de tests dans le membre gauche des règles:

- Les tests *mono-motifs* qui sont les tests effectués au sein d'un motif pour le caractériser. Ils testent l'égalité de symboles entre un fait et le motif ou l'égalité de certains sous-termes des faits lorsqu'une même variable a plusieurs occurrences dans le motif.
- Les tests *multi-motifs* qui testent différents motifs d'un même membre gauche de règle. Ils vérifient les les différentes occurrences de la même variable dans ces motifs. Ils effectuent une *jointure* au sens des Bases de Données (c-à-d ces nœuds produisent en sortie un produit cartésien des données en entrée discriminées par des tests).

Dans l'algorithme RETE de multi-Pattern Matching, on effectue tout d'abord les tests mono-motifs, ensuite on exécute les tests multi-motifs à chaque jointure sur des faits préalablement mémorisés. Le graphe de RETE se compose donc de deux parties (Cf Figure 1).

La première partie est un arbre composé de tests mono-motifs, appelé *arbre test.* L'arbre test a autant de feuilles qu'il y a de motifs distincts dans les membres gauches des règles (l'égalité entre motifs étant entendue au renommage des variables près), soit 8 feuilles dans notre exemple. On définit le *i-motif* en un nœud i de l'arbre test comme le motif correspondant à la suite des tests effectués sur le chemin de l'arbre test allant de la racine à i compris, chemin que l'on appelle la *i-branche.* Le nœud racine est une position non testée, c-à-d un motif non sélectif (qui s'unifie avec tous les faits), une variable. Les i-motifs des feuilles de l'arbre test sont tous les motifs distincts qui apparaissent dans tous les membres gauches des règles. Les différents successeurs d'un nœud correspondent aux différentes branches du graphe à parcourir. Lors de l'exécution un fait atteint les différentes feuilles qui correspondent aux différents motifs avec lesquels il s'unifie.

Ainsi dans notre exemple, au nœud 8 correspond le motif $(h\ \ ao\ X\ Y)$ et le terme $(f\ \ a0\ a1)$ parvient aux nœuds 4, 5 et 10.

La seconde partie du graphe de RETE est composée de jointures binaires entre les feuilles de l'arbre test. En ces nœuds-jointure, le i-motif est un uplet de motifs $(P_1, \ldots, P_l)$, dans lequel

la présence de variables communes inter-motifs est testée par les tests multi-motifs. Les nœuds terminaux de ce *graphe de jointure* correspondent bijectivement avec les règles de la Base de Règles. Les i-motifs des nœuds terminaux sont les membres gauches de toutes les règles au renommage des variables près. Ce sont donc les instances des règles qui atteignent les nœuds terminaux. Lors de l'exécution, lorsqu'un uplet parvient en un nœud-jointure, il est mémorisé dans une mémoire locale un *nœud-mémoire*, et l'on parcourt alors le nœud-mémoire de l'autre entrée du nœud-jointure afin de trouver un uplet satisfaisant aux tests multi-motifs. Ainsi au nœud **17** de la Figure 1, en entrée gauche sont mémorisés des doublets du type $(f\ \ X\ a1)(f\ \ Y\ a2)$, en entrée droite sont mémorisés des $(h\ \ a2\ U\ V)$ et on sélectionne des triplets suivants les tests multi-motifs $X = U$ et $Y = V$. Dans beaucoup d'implémentations on restreint souvent l'entrée droite d'un nœud-jointure à être directement une feuille de l'arbre test (et non une sortie d'un autre nœud-jointure) comme dans la Figure 1 de sorte que le graphe de jointure est un ensemble de *peignes*. Ceci élimine une possibilité d'optimisation consistant à regrouper ensemble les motifs les plus "sélectifs" (nous nous placerons donc dans le cas général pour mener les calculs).

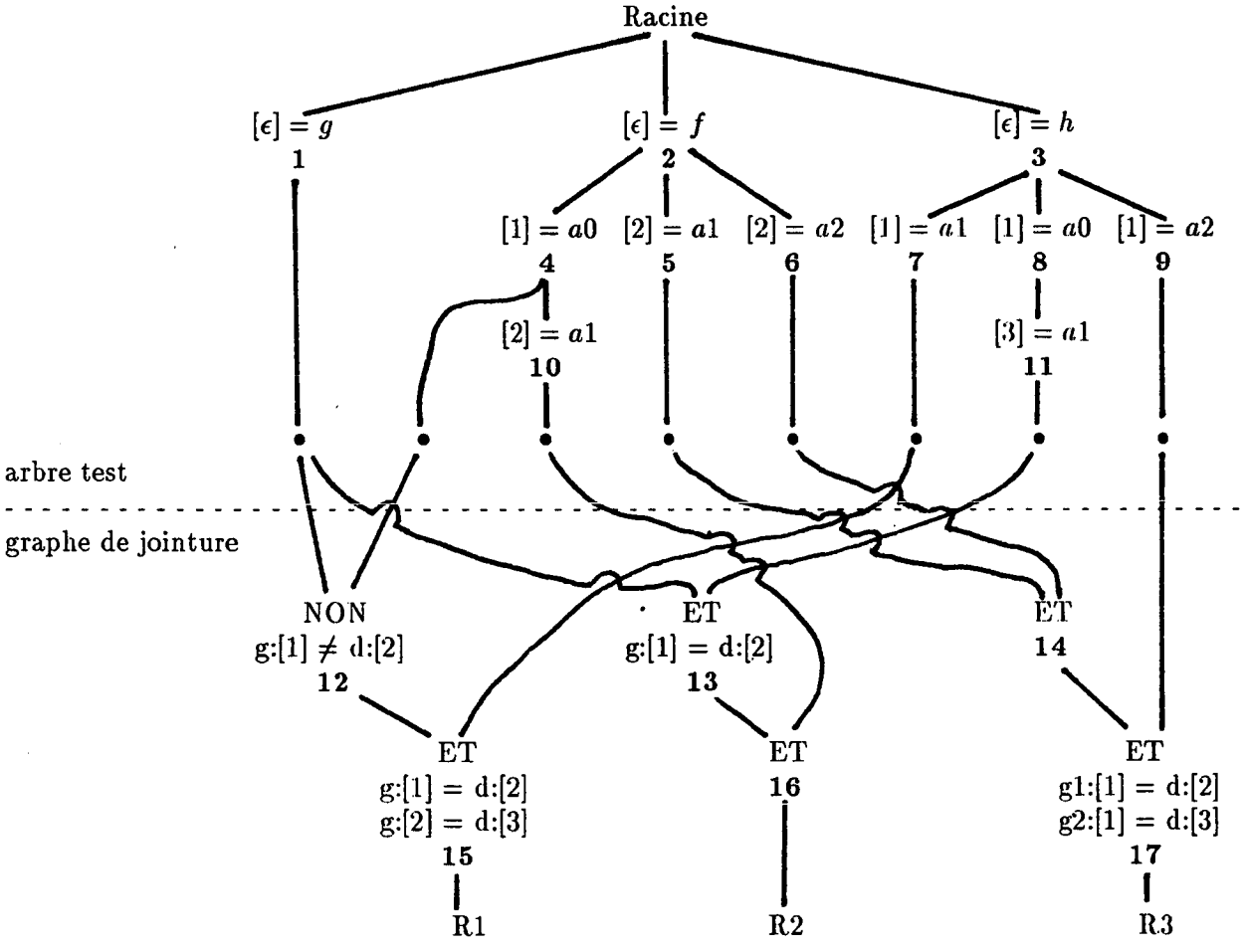


FIGURE 1 † : Graphe de RETE

† Notation : On note par [j] le j^{ieme} fils le plus à gauche de la racine ([ϵ]) du i-motif; d (g) désigne le motif en entrée droite (gauche) d'un nœud-jointure (gj précise le j^{ieme} motif de la liste en entrée gauche).

La complexité en temps de calculs en un nœud-jointure est quadratique en la taille de ses entrées. Dans le pire des cas la taille mémoire des feuilles de l'arbre test est proportionnelle à la taille de la Base de Faits $|BF|$. Donc dans le pire des cas le calcul de l'Ensemble de Conflit est polynomial en fonction de $|BF|$, le degré étant le nombre maximum de motifs dans une règle.

On peut utiliser les techniques de hachage pour réduire la complexité de parcours des nœuds mémoire, en hachant les mémoires d'entrées suivant les tests d'égalités. De même si l'on considère des prédicats d'inégalité on peut structurer les nœuds mémoires en arbres de recherche.

Nous n'avons pas encore parlé de la négation de motifs. Un motif nié traduit la non-existence d'un fait s'unifiant avec le motif. Les motifs niés sont traités dans l'algorithme RETE par un second type de nœuds jointures (nœud **12**). En ces nœuds le nœud-mémoire de l'entrée gauche conserve pour chaque uplet le nombre d'éléments de la mémoire droite (les éléments niés) qui lui sont compatibles relativement aux test multi-motifs. Quand un compteur atteint 0 (resp. 1) le uplet de gauche est propagé dans le graphe sous la forme d'un ajout (resp d'une suppression). Dans notre exemple, tant qu'il n'y a pas dans la Base de Faits de termes du type $(f\ ao\ X)$ le terme en entrée du nœud **12** est transmis au nœud **15**. Le quantificateur existanciel peut être traité de façon analogue. En fait il est possible de généraliser l'algorithme RETE de façon à accepter dans les membres gauches des règles toute formule logique du premier ordre avec un degré d'imbrication de quantificateurs quelconque.

3. LE COUT DE L'ALGORITHME

Dans cette partie, on détermine le coût en moyenne de l'algorithme RETE et l'évaluation de la taille moyenne des nœuds mémoire du graphe de RETE.

3.1 LE MODELE DES FAMILLES SIMPLES D'ARBRES

Dans le modèle des familles simples d'arbres on considère la structure arborescente usuelle des termes. Les motifs sont donc représentés par des arbres aux feuilles éventuellement variables. On sait par [SF 83] et [Stey 84] que le coût moyen pour unifier un motif de taille p (c-à-d le nombre de nœuds de sa représentation arborescente) à un terme de taille n est constant. Dans le cas de l'algorithme RETE on considère des semi-unifications avec une conjonction de motifs.

Comme nous l'avons dit, la Base de Faits est représentée par un uplet quelconque de tels termes appelé une *forêt*. Cette représentation de la Base de Faits sous forme d'une liste ordonnée de faits représente bien la succession d'entrée dans le graphe de RETE des termes de modification et leur possible répétition au cours des cycles (suppression puis ajout d'un même fait). On va donc évaluer dans cette partie le nombre moyen de ces termes ou l-uplets de termes qui sont filtrés en un nœud du graphe de RETE. Afin de déterminer le temps moyen d'établissement de l'Ensemble de Conflit, nous allons utiliser comme mesure le nombre de tests multi-motifs effectués en un nœud-jointure (comme nous l'avons dit le temps de parcours dans l'arbre test reste négligeable par rapport au temps consacré aux jointures).

Nous avons choisi de représenter les termes sous la forme d'arbres construits à partir d'une famille finie F de symboles d'arité fixée: c'est le formalisme des familles simples d'arbres (Cf [Stey 84] [Fla 85] [FS 86]). Soit a_n le nombre de termes de taille n de la famille simple d'arbres. On associe à la famille simple d'arbres la série génératrice:

$$A(z) = \sum_{n \geq 0} a_n z^n = \sum_{t \in T(F)} z^{|t|}$$

avec $|t|$ la taille de t et $T(F)$ l'ensemble des termes construits avec les symboles de F. L'utilisation des séries génératrices est très efficace, on peut en effet systématiser beaucoup d'opérations sur des objets combinatoires comme par exemple traduire sous forme d'équations le produit cartesien, l'union ... (Cf [Fla 87]). Ces séries servent beaucoup pour le dénombrement d'un grand nombre d'objets : des études sur les arbres de recherche, sur les algorithmes de tri ont ainsi été menées.

Exemple : Considérons la famille simple des arbres binaires. On a deux symboles: une constante $-c-$ et un symbole d'arité 2: $- \bullet -$ (penser au "cons" en LISP). On peut alors déterminer $A(z)$ en partitionnant les termes:

$$A(z) = \sum_{t=c} z^{|t|} + \sum_{t=t_1.t_2} z^{|t_1.t_2|}$$

Puisque $|t_1 - \bullet - t_2| = 1 + |t_1| + |t_2|$:

$$A(z) = z + z \sum_{t_1 \in T(F)} \sum_{t_2 \in T(F)} z^{|t_1|} z^{|t_2|} = z(1 + A^2(z))$$

Dans le cas général, en notant c_k le nombre de symboles d'arités k dans F et $\Phi(X)$ le polynome $\Phi(X) = \sum_{k=0}^{p} c_k X^k$ (avec p la plus grande arité parmi les symboles), la série génératrice $A(z)$ vérifie l'équation fonctionnelle:

$$A(z) = z\Phi(A(z))$$

Ce résultat combinatoire classique exprime une transposition systématique de la structure des termes à l'équation fonctionnelle vérifiée par la série génératrice. On peut trouver une preuve de ce résultat dans [SF 83] et dans [MM 78]. On supposera sur Φ que:

- $\exists\, c_k > 0$ avec $K \geq 2$, c-à-d il existe au moins un symbole d'arité ≥ 2.
- et il n'existe pas $\Psi(X)$ polynome et d entier $d \geq 2$ tel que $\Phi(X) = \Psi(X^d)$ (condition de P.G.C.D). Cette condition entraîne l'existence d'une unique solution τ à l'équation fondamentale: $\Phi(\tau) = \tau\Phi'(\tau)$, τ réel et positif. Cette hypothèse est vérifiée dès qu'il y a dans F un symbole unaire. Et quand bien même, l'existence d'un tel d n'est pas génante, les calculs suivants sont identiques. d apparaît comme facteur multiplicatif dans quelques résultats mais disparaît dès qu'il s'agit du calcul d'une moyenne (Cf [Stey 84]), les résultats principaux sont donc inchangés.

Dans notre exemple, $\Phi(X) = X^3 + 2X^2 + 6$.

3.2 FORMULES COMBINATOIRES EXACTES

Nous allons considérer que l'algorithme RETE prend en entrée une Base de Faits quelconque et nous allons déterminer la quantité moyenne de termes ou de l-uplets de termes (c-à-d une liste ordonnée de l termes) qui sont filtrés à un nœud i du graphe de RETE. Nous effectuerons une moyenne sur toutes les Bases de Faits possibles en entrée.

Dans cette section nous allons établir des résultats combinatoires exacts dont on donnera dans la section 3.3 une expression simplifiée en effectuant de l'analyse asymptotique sur la taille de la Base de Faits d'entrée. Nous allons considérer une forêt de k termes de taille globale n (n est la somme des tailles des k termes composant la forêt). La série génératrice d'une forêt de k termes d'une famille simple d'arbres déterminée par $A(z)$ est donnée par:

$$F(z) = A^k(z)$$

Ainsi le n^{ieme} coefficient de z dans $A^k(z)$, $[z^n]A^k(z)$ représente le nombre de forêts de taille n avec k éléments.

Pour un nœud i de l'arbre test, on pose: $B_i(z) = \sum_{m\geq 0} b_{i,m} z^m$ avec $b_{i,m}$ le nombre de termes de taille m qui sont filtrés en i.

Pour un nœud i du graphe de jointure qui produit en sortie des l-uplets de termes qui instancient partiellement le membre gauche d'une règle, on pose: $B_i(z) = \sum_{m\geq 0} b_{i,m} z^m$ avec $b_{i,m}$ le nombre de l-uplets de taille totale m qui sont filtrés en ce nœud-jointure i.

Pour un nœud i de l'arbre test, on a noté i-branche de l'arbre test, la branche de tests qui va de la racine à i compris. Celle-ci détermine le i-motif et on note:

- h_i la hauteur du nœud i dans l'arbre test ($h(racine) = 0$)
- ω_i l'arité du symbole de F testé en i (0 s'il s'agit d'un test d'égalité entre deux variables)
- $\Omega_i = \sum_{j=racine}^{i} \omega_i$
- $x_{i,j}$ = le nombre de j-uplets ($j \geq 2$) de variables identiques testées depuis la racine jusqu'à i compris
- $X_i = \sum_j x_{i,j}$
- $Y_i = \sum_j (j-1)x_{i,j}$ = nombre de tests d'égalité entre deux variables nécessaires pour déterminer le i-motif.

On a donc sur la i-branche Y_i nœuds-test testant l'égalité de deux variables du i-motif et $P_i = h_i - Y_i$ nœuds testant un symbole non variable du i-motif. On peut poser $P_i = |i\text{-motif}|$ et on a $\Omega_i - h_i + 1$ feuilles variables (ou position non encore testée) dans le i-motif.

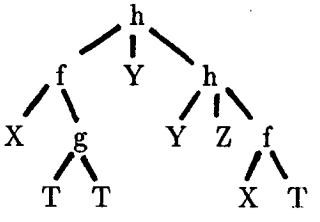
On peut à présent déterminer $B_i(z)$:

THÉORÈME 1 : *La série génératrice des termes filtrés en un nœud i de l'arbre test est:*

$$B_i(z) = z^{P_i} \prod_{j=1}^{\Omega_i - h_i + 1} A^{x_{i,j}}(z^j)$$

avec $x_{i,1} = \Omega_i - h_i + 1 - X_i$ le nombre de variables distinctes dans le i-motif.

Exemple : La série génératrice des termes filtrés par le motif correspondant au nœud **11** de la Figure 1 est $B_{11}(z) = z^3 A(z)$, car $h = 3$, $\omega = 0$, $\Omega = 3$, $x_1 = 1$ et les autres $x_{i,j}$ sont nuls. Si l'on désire un exemple moins élémentaire, on peut considérer le motif suivant:



Pour cet exemple la suite de tests de la i-branche est:

$$\begin{cases} [\epsilon] = h \\ [1] = f \\ [3] = h \\ [1.2] = g \\ [3.1] = [2] \\ [3.3] = f \\ [1.2.1] = [1.2.2] \\ [3.3.1] = [1.1] \\ [3.3.2] = [1.2.1] \end{cases}$$

avec $[m.n]$ qui désigne le n^{ieme} fils le plus à gauche du m^{ieme} fils le plus à gauche de la racine du motif. On a alors:

$$h_i = 9; x_{i,1} = 1; x_{i,2} = 2; x_{i,3} = 1 \quad \text{et les autres } x_{i,j} \text{ sont nuls}$$

$$X_i = 3; Y_i = 4; \Omega_i = 12 \quad \text{et} \quad P_i = 5$$

La série génératrice des termes qui s'unifient avec ce motif est donc:

$$B_i(z) = z^5 A(z)\ A^2(z^2) A(z^3)$$

PREUVE DU THÉORÈME: On a $B_i(z) = \sum_{t \in T_i} z^{|t|}$ avec T_i l'ensemble des termes qui sont filtrés en i.
Soit $t \in T_i$. On a

$$|t| = P_i + \sum_{j=1}^{\Omega_i - h_i + 1} j \left(\sum_{k=1}^{x_{i,j}} |t_{j,k}| \right)$$

avec $t_{i,j}$ qui représente un des sous-arbres qui sont répétés j fois dans t.
d'où

$$B_i(z) = \sum_{t \in T_i} z^{|t|} = z^{P_I} \prod_{j=1}^{\Omega_i - h_i + 1} \left(\prod_{k=1}^{x_{i,j}} \sum_{t_{j,k}} z^{j|t_{j,k}|} \right)$$

Et puisque t est quelconque, les $t_{j,k}$ sont tous quelconques et on obtient donc:

$$B_i(z) = z^{P_I} \prod_{j=1}^{\Omega_i - h_i + 1} \left(\prod_{k=1}^{x_{i,j}} (\sum_{t \in T} z^{j|t|}) \right) = z^{P_I} \prod_{j=1}^{\Omega_i - h_i + 1} A^{x_{i,j}}(z^j) \quad \blacksquare$$

Remarque : On a bien $B_{racine}(z) = A(z)$.

Considérons à présent un nœud i du graphe de jointure. i produit en sortie des listes de l-uplets. Pour un nœud i donné on connait les l motifs qui filtrent chacuns une des composantes des l-uplets de sortie (en oubliant les tests multi-motifs). Soit L_i cet ensemble de l motifs. Notons alors:

- $P_i = \sum_{p \in L_i} P_p$
- $\Omega_i = \sum_{p \in L_i} \Omega_p$
- $h_i = \sum_{p \in L_i} h_p$
- pour $j \geq 1$, $x_{i,j} = \left(\sum_{p \in L_i} x_{i,p} \right) + c_{i,j}$

avec $c_{i,j}$ le terme correctif dû à l'apparition de variables identiques entre les l motifs. *Si l'on considère l'ensemble global de toutes les variables de tous les différents l motifs, $x_{i,j}$ compte encore le nombre de j-uplets de variables identiques dans cet ensemble.* On détermine donc aisément les $c_{i,j}$ en fonction des variables identiques testées au nœud i, on a: $c_{i,1} \leq 0$ car la jointure accroît *a priori* le nombre de variables identiques.
Ainsi pour le nœud **13** de la Figure 1, on a $P = 4$, $\Omega = 5$, $h = 4$, $x_1 = 1$ et $x_2 = 1$.

On obtient alors de la même façon que précédemment:

THÉORÈME 2 : *La série génératrice des l-uplets de termes qui sont filtrés au nœud i du graphe de jointure est:*

$$B_i(z) = z^{P_i} \prod_{j=1}^{\Omega_i - h_i + l} A^{x_{i,j}}(z^j)$$

Déduisons de ceci le nombre moyen $\overline{b_{i,n,k}}$ de termes ou de l-uplets de termes filtrés en i en considérant comme donnée de l'algorithme RETE une forêt quelconque de k termes de taille n.

THÉORÈME 3 : *Le nombre moyen de termes filtrés en un nœud i de l'arbre test est:*

$$\overline{b_{i,n,k}} = \frac{k[z^{n-P_i}]\prod_{j=1}^{\Omega_i - h_i + 1} A^{\nu_{i,j}}(z^j)}{[z^n]A^k(z)}$$

avec $\nu_{i,1} = k + \Omega_i - h_i - X_i$ et pour $j \geq 2$, $\nu_{i,j} = x_{i,j}$.

PREUVE : L'idée de base de cette démonstration est de séparer $A(z)$ en: $A(z) = B_i(z) + C_i(z)$ avec $C_i(z)$ la "série reste" c-à-d la série génératrice de tous les termes qui ne s'unifient pas avec le i-motif. Alors on marque avec la variable u les éléments filtrés en i et on pose:

$$A(z,u) = uB_i(z) + C_i(z) = A(z) + (u-1)B_i(z)$$

Ainsi $f_{i,n,p,k} = [u^p z^n](A^k(z,u))$ représente le nombre de forêts à k termes de taille n avec exactement p termes qui sont filtrés en i.
Donc $\overline{b_{i,n,k}}$ est le quotient de $\sum_{p=0}^{+\infty} p f_{i,n,p,k}$ $\left(= \sum_{p=0}^{k} p f_{i,n,p,k}\right)$ par le nombre total de forêts de taille n à k éléments, soit:

$$\overline{b_{i,n,k}} = \frac{\sum_p p f_{i,n,p,k}}{[z^n]A^k(z)}$$

On a $(A^k(z,u)) = \sum_{p,n} f_{i,n,p,k} u^p z^n$ d'où

$$\frac{\partial}{\partial u}\left(A^k(z,u)\right) = \sum_{p,n} f_{i,n,p,k} p u^{p-1} z^n$$

d'où

$$\frac{\partial}{\partial u}\left(A^k(z,u)\right)\Big|_{u=1} = \sum_n \left(\sum_p p f_{i,n,p,k}\right) z^n$$

Donc

$$\overline{b_{i,n,k}} = \frac{[z^n]\frac{\partial}{\partial u}\left(A^k(z,u)\right)|_{u=1}}{[z^n]A^k(z)}$$

Et

$$\begin{aligned}\frac{\partial}{\partial u}\left(A^k(z,u)\right)\Big|_{u=1} &= kA^{k-1}(z,u)\frac{\partial}{\partial u}\left(A(z,u)\right)\Big|_{u=1} \\ &= kA^{k-1}(z)B_i(z)\end{aligned}$$

et donc le Théorème est démontré. ∎

THÉORÈME 4 : *Le nombre moyen de l-uplets de termes filtrant en un nœud i du graphe de jointure est:*

$$\overline{b_{i,n,k}} = \frac{k^l[z^{lk^{l-1}n - P_i}]\prod_{j=1}^{\Omega_i - h_i + l} A^{\nu_{i,j}}(z^j)}{[z^{lk^{l-1}n}]A^{lk^l}(z)}$$

avec $\nu_{i,1} = l(k^l - 1) + x_{i,1}$ et pour $j \geq 2$, $\nu_{i,j} = x_{i,j}$.

Remarque : Les notations ont été choisies de façon à ce que l'expression obtenue dans le Théorème 3 corresponde exactement au cas $l = 1$.

PREUVE : A partir d'une forêt f de taille n avec k termes on obtient k^l l-uplets d'éléments de f au niveau du nœud i. Parmi ces l-uplets nous devons déterminer ceux qui sont filtrés en i. Dans l'algorithme RETE classique on a vu qu'un fait de la Base de Faits peut parcourir plusieurs chemins dans l'arbre test et se retrouver dupliqué aux feuilles de celui-ci. En raison de cette duplication on ne va pas faire de moyenne sur les k-uplets de termes représentant les Bases de Faits possibles mais sur toutes les k^l-listes de l-uplets de termes quelconques. Ceci suppose donc une répartition uniforme des l-uplets de termes quelconques issus de f. A ce moment, en considérant donc que f conduit au niveau de i à une k^l-liste de l-uplets de termes quelconques de taille totale $lk^{l-1}n$, on obtient le Théorème avec un raisonnement complètement identique au précédent. ∎

Remarque : On verra qu'avec l'ARBRE D'UNIFICATION, un fait parcourt de façon unique l'arbre test, les calculs peuvent être alors menés en effectuant une moyenne sur les Bases de Faits et le résultat obtenu reste identique à celui du Théorème 4.

Remarquons également que les résultats sont obtenus en fonction des caractéristiques du graphe de RETE qui est une donnée fixe de l'algorithme. Ces résultats nous permettrons donc une optimisation dans la création du graphe de RETE (construction pour laquelle il y a encore beaucoup de choix arbitraires dans l'ordonnancement des nœuds).

Ces résultats combinatoires exacts peuvent être étudiés en utilisant par exemple la formule d'inversion de Lagrange mais les expressions obtenues sont difficiles à utiliser (coefficients multinomiaux ...). De façon à les exprimer simplement nous allons effectuer une évaluation asymptotique en fonction de k et n, car on peut en effet considérer ces 2 paramètres comme arbitrairement grand si l'on se réfère aux grosses Bases de Faits des systèmes réels ou si l'on considère ce qui revient au même beaucoup de cycles du Moteur d'Inférences.

3.3 EVALUATION ASYMPTOTIQUE

Afin de simplifier les expressions obtenues, nous allons utiliser des méthodes d'analyse de singularités. Nous allons estimer la valeur d'intégrales avec des méthodes d'analyse complexe (comme la méthode du point col). On considère comme donnée de l'algorithme une Base de Faits de taille globale n avec k termes. Et donc nous allons considérer que n et k croissent tous les deux.

D'après la section 3.2 nous devons déterminer des coefficients tels que: $[z^n]\prod_{j=1} A^{\nu_{i,j}}(z^j)$. Nous allons tout d'abord évaluer $[z^n]A^k(z)$, puis nous montrerons comment on peut en déduire l'expression de $\overline{b_{i,n,k}}$.
Soit $a_n^k = [z^n]A^k(z)$. Par le Théorème de Cauchy on peut exprimer cette quantité par l'intégrale suivante

$$a_n^k = \frac{1}{2i\pi}\int_\Gamma (A^k(z))\frac{dz}{z^{n+1}}$$

où Γ est un lacet qui entoure simplement l'origine en restant dans le domaine d'analycité de $A(z)$. Afin d'obtenir un équivalent de cette intégrale quand k et n tendent tous deux vers $+\infty$, nous allons utiliser la méthode du col (Cf [dB 58] ou [Henr 74]). Auparavant de façon à ne conserver qu'un paramètre sous l'intégrale, nous allons établir une relation entre k et n dans les familles simples d'arbres. Avec des méthodes combinatoires classiques (Cf [Alb 87]), on obtient le résultat suivant:

THÉORÈME 5 : *n étant fixé, le nombre moyen d'arbres $\bar{k}_n$ dans une forêt de taille n est:*

$$\bar{k}_n = \frac{[z^n]\left(\frac{A(z)}{(1-A(z))^2}\right)}{[z^n]\left(\frac{1}{1-A(z)}\right)}$$

Rappelons les notations qui vont être utilisées par la suite:

PROPOSITION: *Considérons une famille simple d'arbres. Notons $A(z)$ sa série génératrice, comme nous l'avons déjà vu, elle vérifie*

$$A(z) = z\Phi(A(z))$$

On appelle τ la plus petite racine réelle positive de l'équation

$$\Phi(X) = X\Phi'(X)$$

(l'unicité d'une telle solution étant assurée par la condition de P.G.C.D.) et $\rho = \tau/\Phi(\tau)$ est le rayon de convergence de $A(z)$. On a $0 < \rho < 1$ et $\tau = A(\rho)$.

Avec ces notations, on a dans le voisinage de $z = \rho$ (Cf [MM 78])

$$A(z) = \tau - \sqrt{\frac{2\Phi(\tau)}{\Phi''(\tau)}}\left(1-\frac{z}{\rho}\right)^{\frac{1}{2}} + \gamma\left(1-\frac{z}{\rho}\right) + O\left(\left(1-\frac{z}{\rho}\right)^{\frac{3}{2}}\right)$$

D'où l'on déduit

THÉORÈME 6 : *Considérons une forêt de termes de cette famille simple d'arbres. Le nombre moyen $\bar{k}_n$ de termes dans une forêt de taille globale n fixée est:*

1) Si $\tau < 1$

$$\bar{k}_n = \frac{1+\tau}{1-\tau}\left(1+O\left(\frac{1}{n}\right)\right)$$

2) Si $\tau = 1$

$$\bar{k}_n = \sqrt{\frac{\pi\Phi''(1)}{\Phi(1)}}n^{\frac{1}{2}}\left(1+O\left(\frac{1}{\sqrt{n}}\right)\right)$$

3) Si $\tau > 1$ $\exists\, \omega < \rho$ / $A(\omega) = 1$ $(\omega = \Phi(1)^{-1})$ et

$$\bar{k}_n = \underbrace{\frac{1}{2}\left(1-\frac{\Phi'(1)}{\Phi(1)}\right)}_{<1} n\left(1+O\left(\frac{1}{n^2}\right)\right)$$

La taille des termes est approximativement constante dans les implémentations c-à-d $\bar{k}_n$ est proportionnel à n. Le troisième cas reflète donc bien la réalité. Théoriquement c'est le cas où

le nombre de constantes $\Phi(0)$ est assez grand (il est alors facile de vérifier que la constante de proportionalité entre k et n est bien inférieure à 1 dans ce cas là, [Alb 87]).

On va donc évaluer à présent a_n^k dans les trois cas suivants:

1) $k = \alpha + \mu/n$ et $\tau < 1$
2) $k = \lambda\sqrt{n} + \mu$ et $\tau = 1$
3) $k = \lambda n + \alpha + \mu/n$ et $\tau > 1$

(Pour plus de détails sur les calculs qui vont suivre se raporter à [Alb 87] ou à [SF 83] où un calcul similaire est développé dans le cas où $k \leq \sqrt{n}/\log^3 n$).

Expliquons succinctement le principe des calculs. On a

$$a_n^k = \frac{1}{2i\pi}\int_\Gamma (A^k(z))\frac{dz}{z^{n+1}}$$

Nous allons effectuer un changement de variable et faire de $y = A(z)$ la variable indépendante. Avec l'équation fonctionnelle, a_n^k devient:

$$a_n^k = \frac{1}{2i\pi}\int_C \Phi^n(y)\left(1 - y\frac{\Phi'(y)}{\Phi(y)}\right)\frac{dy}{y^{n-k+1}}$$

avec C image de Γ qui peut être pris quelconque dans le domaine d'analycité de Φ. Nous allons utiliser la *méthode de col.* Cela va nous conduire à choisir un lacet C particulier.

- Dans les cas 1) et 2) celui-ci va être

$$C = \{z/|z| = \tau\}$$

ceci car dans ces cas τ est le col de la partie principale de l'intégrale, c-à-d $(\Phi(y)/y)^{n-1}$, quand n croît.

Dans le troisième cas nous prendrons comme point col σ la plus petite racine réelle positive de l'équation fondamentale:

$$y\Phi'(y) = (1 - A)\Phi(y)$$

L'idée de base ensuite consiste à couper l'intégrale en deux:

$$a_n^k = \int_C = \int_{C1} + \int_{C2} = I_1 + I_2$$

avec $C1 = \{z/|z| = \tau \; et \; -\epsilon \leq Arg(z) \leq \epsilon\}$, $C2 = C/C1$ et $\epsilon = \log^2 n/\sqrt{n}$. Ce choix de ϵ est tel que d'une part l'intégrale le long de $C2$ est exponentiellement négligeable et que d'autre part ϵ est assez petit pour que l'on puisse réaliser des développements sous le signe somme afin d'estimer la valeur de $C1$.

A la suite de ces calculs on obtient:

THÉORÈME 7 : *Avec* $\bar{k}_n = \alpha(1 + O(\frac{1}{n}))$ *(cas où* $\tau < 1$*):*

$$\begin{aligned} a_n^k &= [z^n](A(z)^k) \\ &= \sqrt{\frac{\Phi(\tau)}{2\pi\Phi''(\tau)}}\rho^{-n}\tau^{\alpha-1}\alpha n^{-\frac{3}{2}}\left(1 + O\left(\frac{1}{n}\right)\right) \end{aligned}$$

THÉORÈME 8 : *Avec* $\bar{k}_n = \lambda\sqrt{n} + O(1)$, *et* $\tau = 1$, *on a*

$$a_n^k = \frac{\Phi(1)^n}{2} e^{-\pi} n^{-1} \left(1 + O\left(\frac{1}{\sqrt{n}}\right)\right)$$

THÉORÈME 9 : *Avec* $\bar{k}_n = \lambda n + \alpha + O\left(\frac{1}{n}\right)$, $\tau > 1$, *on a*

$$a_n^k = \frac{\lambda}{4\sqrt{\pi}} \left(\frac{\sigma^2\Phi''(\sigma)}{2\Phi(\sigma)} + \frac{\lambda(1-\lambda)}{2}\right)^{-\frac{1}{2}} \sigma^{-n(1-\lambda)} \Phi(\sigma)^n n^{-\frac{1}{2}} \left(1 + O\left(\frac{1}{n}\right)\right)$$

avec σ *la plus petite racine réelle positive de*

$$y\Phi'(y) = (1-\lambda)\Phi(y)$$

Avec ces résultats, en se plaçant dans le cas le plus proche de la réalité, on peut à présent déterminer l'estimation asymptotique du dénominateur de l'expression précédente donnant $\overline{b_{i,n,k}}$. Pour le numérateur nous allons utiliser le lemme suivant qui découle des calculs précédents:

LEMME : *Soit* ρ *le rayon de convergence de* $A(z)$. *Si* $g(z)$ *est une fonction analytique au voisinage de* ρ *alors dans les trois cas précédents, on a:*

$$[z^n]\{A^k(z)g(z)\} = g(\rho)[z^n]A^k(z)$$

Le rayon de convergence de $A(z^j)$ est $\rho^{\frac{1}{j}}$. Comme $\rho < 1$, on a $\rho = min\left\{\rho^{\frac{1}{j}}\ ,\ j \geq 1\right\}$, d'où

$$[z^n]\prod_{j=1} A^{\nu_{i,j}}(z^j) = \prod_{j\geq 2} A^{\nu_{i,j}}(\rho^j)\left([z^n]A^{\nu_1^i}(z)\right)$$

Les calculs sont alors presque immédiats et on obtient le Théorème:

THÉORÈME 10 : *Soit une forêt de* k *termes de taille* n *avec* $\bar{k}_n = \frac{1}{2}\left(1 - \frac{\Phi'(1)}{\Phi(1)}\right)n\left(1 + O\left(\frac{1}{n^2}\right)\right)$ *(cas où* $\tau > 1$*), on a:*

- *pour un nœud* i *de l'arbre test:*

$$\overline{b_{i,n,k}} = \Pi_i n\left(1 + O\left(\frac{1}{n}\right)\right)$$

$$\text{avec} \qquad \Pi_i = \frac{1}{2}\left(1 - \frac{\Phi'(1)}{\Phi(1)}\right)\left(\frac{\sigma^{\frac{1}{2}\left(1+\frac{\Phi'(1)}{\Phi(1)}\right)}}{\Phi(\sigma)}\right)^{h_i - Y_i}\left(\prod_{j=2}^{\Omega_i - h_i + 1} A^{\nu_{i,j}}(\rho^j)\right)$$

- *pour un nœud* i *du graphe de jointure:*

$$\overline{b_{i,n,k}} = \Pi_i n^l\left(1 + O\left(\frac{1}{n^2}\right)\right)$$

$$\text{avec} \qquad \Pi_i = \frac{1}{2}\left(1 - \frac{\Phi'(1)}{\Phi(1)}\right)^l\left(\frac{\sigma^{\frac{1}{2}\left(1+\frac{\Phi'(1)}{\Phi(1)}\right)}}{\Phi(\sigma)}\right)^{h_i - Y_i}\left(\prod_{j=2}^{\Omega_i - h_i + l} A^{\nu_{i,j}}(\rho^j)\right)$$

avec σ la plus petite racine réelle de l' équation

$$y\Phi'(y) = \frac{1}{2}\left(1 - \frac{\Phi'(1)}{\Phi(1)}\right)\Phi(y)$$

On peut à présent définir un taux de filtrage $\overline{\alpha_{i,n}}$ en un nœud i du graphe de RETE qui traduit la probabilité pour un terme ou un l-uplet de termes d'être filtré en un nœud i du graphe de RETE. On obtient facilement le résultat suivant:

THÉORÈME 11 : *Le taux de filtrage pour un nœud i du graphe de* RETE *est*

$$\overline{\alpha_{i,n}} \simeq \left(\frac{\sigma^{\frac{1}{2}\left(1+\frac{\Phi'(1)}{\Phi(1)}\right)}}{\Phi(\sigma)}\right)^{h_i - Y_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A^{\nu_{i,j}}(\rho^j)\right)$$

avec l le nombre de composantes des uplets en sortie d'un nœud i du graphe de jointure en posant pour un nœud i de l'arbre test $l = 1$.
(On gardera cette notation par la suite pour des commodités d'écriture).

Remarque : Notons que:

$$\frac{\sigma^{\frac{1}{2}\left(1+\frac{\Phi'(1)}{\Phi(1)}\right)}}{\Phi(\sigma)} \leq 1$$

et donc Π_i décroît rapidement avec h_i.

On donne en Annexe les résultats numériques des taux de filtrage des nœuds du graphe de la Figure 1.

3.4 RESULTATS AVEC LE MODELE DES FAMILLES SIMPLES D'ARBRES

On peut à présent déterminer le coût moyen de l'algorithme RETE étant donnés une Base de Faits quelconque et un graphe de RETE fixé. Nous allons évaluer le temps moyen nécessaire à l'algorithme RETE pour déterminer l'Ensemble de Conflit à partir d'une Base de Faits d'entrée donnée. On détermine le nombre moyen de tests multi-motifs réalisés aux nœuds-jointure (ce qui représente comme on l'a déjà dit la partie principale du temps total d'exécution). Notons l_i le nombre de composantes des uplets de sortie en un nœud-jointure i et β_i le nombre de tests multi-motifs realisés en i (en considérant seulement un l_i-uplet en entrée, dans notre exemple $\beta_{17} = 2$, $\beta_{14} = 0$, $\beta_{13} = 1$ ...). Le nombre moyen de tests réalisés en i est le nombre moyen de l_i-uplets testés en i multipliés par β_i. Pour obtenir le nombre moyen de l_i-uplets qui arrive en entrée en i il suffit de considérer qu'en i aucun test multi-motifs n'est réalisé (c-à-d $\forall j$, $c_{i,j} = 0$) et d'évaluer le $\overline{b'_{i,n,k}}$ ainsi modifié. Notons $\Pi'_i n^{l_i}$ cette quantité (remarquons que celle-ci est précisément égale à $\overline{b_{i1,n,k}} \times \overline{b_{i2,n,k}}$ où $i1$ et $i2$ désignent les deux nœuds entrées de i).

On peut alors proposer

$$\overline{coût_n} \simeq K \sum_{i \ nd-jt} \beta_i \Pi'_i n^{l_i} = K \sum_{i \ nd-jt} \beta_i (\Pi_{i1}\Pi_{i2}) n^{l_i} \qquad (1)$$

avec K une constante dépendant de l'implémentation.

Bien sûr, asymptotiquement en n, on obtient la partie prépondérente de cette expression en gardant seulement les nœuds qui ont le plus grand l_i.

Remarque : En hachant les mémoires locales suivant les tests d'égalité réalisés aux nœuds-jointure, le nombre moyen de semi-unifications réalisées en un nœud-jointure i est seulement le nombre de l_i-uplets de sortie c-à-d $\overline{b_{i,n,k}}$. Et donc dans ce cas nous avons juste à remplacer dans la formule ci-dessus Π'_i par Π_i.

A partir de cette expression,*qui peut être complètement précisée sur n'importe quel exemple*, on peut émettre *a posteriori* les hypothèses de Forgy (Cf [Forg 79]). Par exemple, si l'on suppose que le nombre de motifs-condition par membre gauche dans la Base de Règles est constant égal à c, le résultat précédent devient:

$$\overline{coût_{|BF|}} \simeq K\lambda^c|BF|^c\left(\sum_{j=1}^{|BR|}\beta_j\Pi'_j\right) = K\lambda^c|BF|^c\left(\sum_{j=1}^{|BR|}\beta_j\Pi_{j1}\Pi_{j2}\right)$$

avec $j1$ et $j2$ les deux entrées du dernier nœud-jointure j de la règle Rj.
Et l'on retrouve le même type de relations proposées par Forgy: un coût linéaire en fonction du nombre de règles et un coût polynomial en fonction de la taille de la Base de Faits $|BF|^c$ (on a déterminé le temps d'exécution de l'algorithme RETE avec en entrée $|BF|$ termes-modification, Forgy trouvait un temps en $|BF|^{c-1}$ parce qu'il considérait seulement un terme-modification comme entrée).

4. LE MODELE PONDERE

4.1 INTRODUCTION

Le modèle étudié dans la dernière partie considérait une distribution uniforme sur tous les arbres de la même taille n. Afin d'obtenir une modélisation plus fine et plus vraisemblable, nous allons considérer une modèle prenant en compte le fait que pour beaucoup d'applications certains symboles sont plus fréquents que d'autres. Nous présentons donc ici un modèle probabiliste appelé *modèle de branchement* (Cf [FSS]).

Soit F un ensemble de symboles fonctionnels, tous munis d'une *arité*. L'ensemble des termes correspondant est défini de la façon usuelle. Soit w une *fonction de poids* qui associe à chaque symbole $f \in F$ un réel positif $w[f]$. On étend ce poids multiplicativement aux arbres. Si t est un arbre, son poids $w[t]$ est défini comme

$$w[t] = \prod_{f \in t} w[f]$$

où le produit est étendu à tous les symboles de t. Si on désire utiliser un modèle de probabilité pour les symboles, on prend

$$\sum_{f \in F} w[f] = 1$$

Ces modèles pondérés sont intéressants pour plusieurs raisons. Par construction, ils sont à même de représenter des situations réelles mieux que des modèles uniformes. Ces modèles sont aussi naturels d'un point de vue mathématique car ils reviennent à supposer que les arbres sont générés par un *processus de branchement* conditionné par la taille n du résultat. On peut encore associer

à (F,w) un polynome de structure: $\Phi_1(y) = \sum_{f\in F} w[f]y^{ari[f]}$. La série génératrice est alors $A_1(z) = \sum_{t\in T} w[t]z^{|t|}$ et elle vérifie encore: $A_1(z) = z\Phi_1(A_1(z))$. Avec ce modèle $B_i(z)$ devient:

THÉORÈME 12 : *Pour un nœud i du graphe de jointure, on a*

$$B_i(z) = W_i z^{P_i} \prod_{j=1}^{\Omega_i - h_i + l} A_j^{\nu_{i,j}}(z^j)$$

avec W_i le poids du i-motif c-à-d *pour un nœud i de l'arbre test $W_i = \prod_{f\in i-mot} w[f]$ et pour un nœud i du graphe de jointure, puisque le i-motif est un l_i-uplet de motifs, on pose $W_i = \prod_{j_i=1}^{l_i} W_{j_i}$ avec W_{j_i} le poids de la j^{ieme} composante du i-motif,*
et avec $A_j(z)$ qui vérifie

$$A_j(z) = z\Phi_j(A_j(z)) \qquad \text{avec} \qquad \Phi_j(y) = \sum_F w^j[f]y^{ari[f]}$$

On démontre ce Théorème avec le même raisonnement qu'auparavant. On introduit la série génératrice $A_j(z)$ à cause de l'apparition dans les calculs de séries du type:
$\sum_{t\in T} w^j[t]z^{j|t|}$ $(= A_j(z^j))$.
On déduit de tout ceci:

THÉORÈME 13 : *Pour un nœud i du graphe de* RETE, *avec le modèle pondéré le nombre moyen de termes filtrés en i est:*

$$\overline{b_{i,n,k}} = \frac{k^l W_i [z^{lk^{l-1}n - P_i}] \prod_{j=1}^{\Omega_i - h_i + l} A_j^{\nu_{i,j}}(z^j)}{[z^{lk^{l-1}n}] A^{lk^l}(z)}$$

Tous les $A_j(z)$ ont un rayon de convergence $\rho_j < 1$ qui peut être déterminé (ainsi que τ_j) comme précédemment.
Il y aura donc deux cas dans les évaluations asymptotiques qui vont suivre:
Soit $\rho_1 = min\{\rho_j^{\frac{1}{j}} \ , \ j \geq 1\}$ et les résultats seront similaires à ceux du modèle uniforme, soit $\exists\ m \neq 1\ /\ \rho_m^{\frac{1}{m}} = min\{\rho_j^{\frac{1}{j}} \ , \ j \geq 1\}$. Dans ce cas l'évaluation du dénominateur de $\overline{b_{i,n,k}}$ est bien sûr inchangée mais pour le numérateur la série de rayon de convergence minimal est $A_m^{\nu_{i,m}}(z^m)$ et l'évaluation asymptotique utilisera le Théorème 7.

4.2 RESULTATS AVEC LE MODELE PONDERE

On obtient le Théorème suivant:

THÉORÈME 14 : *Avec le modèle pondéré, considérons une forêt de taille n de k termes avec*

$$\bar{k}_n = \frac{1}{2}\left(1 - \frac{\Phi_1'(1)}{\Phi_1(1)}\right) n \left(1 + O\left(\frac{1}{n^2}\right)\right) = \lambda n \left(1 + O\left(\frac{1}{n^2}\right)\right)$$

on a:

- *Si* $\rho_1 = min\{\rho_j^{\frac{1}{j}} \ , \ j \geq 1\}$
  - *pour un nœud i de l'arbre test:*

$$\overline{b_{i,n,k}} = \Pi_i n\left(1 + O\left(\frac{1}{n}\right)\right)$$

  - *pour un nœud i du graphe de jointure:*

$$\overline{b_{i,n,k}} = \Pi_i n^l\left(1 + O\left(\frac{1}{n^2}\right)\right)$$

avec dans les deux cas $\Pi_i = W_i \lambda^l \left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{P_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{\nu_{i,j}}(\rho_1^j)\right)$

et le taux de filtrage $\overline{\alpha_{i,n}} \simeq W_i \left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{P_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{\nu_{i,j}}(\rho_1^j)\right)$

- *Si* $\exists\ m \neq 1\ /\ \rho_m^{\frac{1}{m}} = min\{\rho_j^{\frac{1}{j}} \ , \ j \geq 1\}$
  - *pour un nœud i de l'arbre test:*

$$\overline{b_{i,n,k}} = M_{i,n}\left(1 + O\left(\frac{1}{n}\right)\right)$$

  - *pour un nœud i du graphe de jointure:*

$$\overline{b_{i,n,k}} = M_{i,n}\left(1 + O\left(\frac{1}{n^2}\right)\right)$$

avec dans les deux cas

$$M_{i,n} = \frac{W_i}{l}\left(\prod_{\substack{j=1 \\ \text{sauf } m}}^{\Omega_i - h_i + 1} A_j^{\nu_{i,j}}(\rho_m^{\frac{j}{m}})\right)\sqrt{\frac{m\Phi_m(\tau_m)}{2\pi\Phi_m''(\tau_m)}}\rho_m^{-\frac{l\lambda^{l-1}n^l - P_i}{m}}\tau_m^{\nu_{i,m}-1}4\nu_{i,m}\sqrt{\pi}$$

$$\times\left(\frac{\sigma^2\Phi_1''(\sigma)}{2\Phi_1(\sigma)} + \frac{\lambda(1-\lambda)}{2}\right)^{\frac{1}{2}}\left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{l\lambda^{l-1}n^l}$$

Et toutes les constantes dans ces expressions peuvent être complètement déterminées sur un exemple. On connaît toutes les constantes dépendant des caractéristiques du graphe de RETE et des caractéristiques du modèle pondéré (w, $\Phi_j \ldots$). Et on a: τ_j la plus petite racine réelle de $\Phi_j(y) = y\Phi_j'(y)$, qui détermine les ρ_j ($\rho_j = \frac{\tau_j}{\Phi_j(\tau_j)} = \frac{1}{\Phi_j'(\tau_j)}$), et σ la plus petite racine de l'équation $y\Phi'(y) = (1-\lambda)\Phi(y)$, et $A_j(\rho_m^{\frac{j}{m}})$ est la limite de la suite $u_0 = A(0) = 0$ et $u_{h+1} = \rho_m^{\frac{j}{m}}\Phi_j(u_h)$ (qui converge géométriquement). Toutes ces valeurs sont faciles à déterminer avec l'assistance d'un programme de calcul formel tel que MAPLE (Cf [MAPLE 85]). L'utilisateur de ces résultats n'a donc pas à savoir manipuler les séries formelles, ces résultats sont sous la forme de *formules prêtes à l'emploi* !

En ce qui concerne le coût en moyenne de l'algorithme RETE, dans le premier cas, le modèle pondéré ne fait qu'introduire qu'un facteur multiplicatif W_i dans les Π_i. Les résultats avec cette modification sont donc les mêmes. Dans le second cas, qui reste à mon avis un seul cas théorique, le coût moyen global de la semi-unification est la somme pour tous les nœuds i de graphe de jointure des $\beta_i\overline{b'_{i,n,k}}$.

On donne en Annexe les taux de filtrage des nœuds du graphe de la Figure 1 où l'on a pondéré les symboles de F en fonction d'une Base de Faits particulière.

5. LA NEGATION

Nous pouvons aisément étendre les calculs précédents à un modèle prenant en compte la *négation*. Plus précisément nous pouvons considérer la négation de motifs-condition dans le membre gauche des règles ou aussi des tests de non-égalité entre variables.

Les résultats que l'on a obtenus sont assez souples, par exemple si l'on veut calculer le nombre moyen de termes filtrés par le motif suivant (dans lequel X est un symbole de variable et $TETE$ la partie non variable du motif) :

$$TETE[X, \neg X]$$

celui-ci est égal au nombre moyen de termes filtrés par $TETE[X, Y]$ moins le nombre moyen de termes filtrés par $TETE[X, X]$ (ce qui s'étend facilement au cas de plusieurs variables niées simultanément).

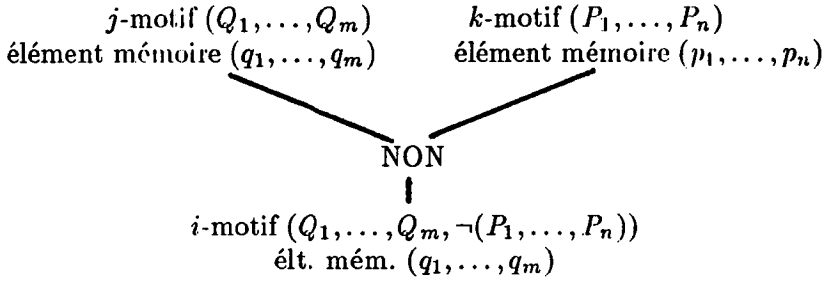


FIGURE 2

Considérons à présent la négation d'un motif-condition ou d'un l-uplet de motifs-condition. Pour un nœud i (Cf Figure 2), on connaît le taux de filtrage $\overline{\alpha_{k,n}}$ au nœud k et le taux de filtrage au nœud j $\overline{\alpha_{j,n}}$. Le taux de filtrage du motif $\neg(P_1, \ldots, P_n)$ est $1 - \overline{\alpha_{k,n}}$ et donc le taux de filtrage du nœud i correspondant au i-motif $(Q_1, \ldots, Q_m, \neg(P_1, \ldots, P_n))$ est $\overline{\alpha_{i,n}} = \overline{\alpha_{j,n}}(1 - \overline{\alpha_{k,n}})$ (quand il n'y a pas de variables communes entre les P_i et les Q_j). Et les taux de filtrage des nœuds-jointure qui sont sous ce nœud (c-à-d qui peuvent être atteints depuis i) sont modifiés de la même façon: on détermine normalement leur taux de filtrage en oubliant l'existence du uplet de motifs $\neg(P_1, \ldots, P_n)$ et on multiplie le résultat obtenu par le coefficient $(1 - \overline{\alpha_{k,n}})$ pour obtenir le taux de filtrage réel.

Le cas du nœud 12 de la Figure 1 requiert les deux raisonnements ci-dessus à la fois.

6. L'ARBRE D'UNIFICATION

Dans cette partie, on étudie une version de l'algorithme RETE introduite dans [Gha 80], [GD 84] et [Duf 84], l'ARBRE D'UNIFICATION, dont on évalue la performance présentée dans le Théorème 17.

Il s'agit principalement d'une restructuration de l'arbre test fondée sur les remarques suivantes:

- Dans le graphe de RETE classique, chaque donnée traverse le graphe par plusieurs chemins et l'information acquise sur cette donnée dans un chemin n'est pas exploitée dans une autre partie du graphe,
- l'utilisation de tests binaires fait que la même information (par exemple la valeur d'un attribut) peut être extraite plusieurs fois de la même donnée: il y a répétition de tests,
- enfin il n'y a pas, comme on l'a déjà dit, d'ordonnancement des tests associés aux motifs et aux jointures de façon à optimiser le graphe.

L'ARBRE D'UNIFICATION est un arbre de discrimination des motifs-condition tenant compte de ces remarques. Son établissement suppose que tout élément de la Base de Faits s'unifie nécessairement avec un motif-condition du membre gauche d'une règle (cette hypothèse est en général vérifiée lorsque les faits qui sont ajoutés à la Base de Faits durant l'exécution proviennent seulement des membres droits des règles et non du milieu extérieur au système expert, d'ailleurs une modification mineure de l'ARBRE D'UNIFICATION nous ramène au cas général). L'arbre test ainsi construit possède comme caractéristique fondamentale le fait que chaque donnée parcourt le graphe par un seul chemin. Ceci permet d'avoir aux feuilles de l'arbre test une partition de la Base de Faits d'entrée. Le graphe de jointure est lui aussi ordonné de façon particulière. L'algorithme de création de ce graphe est donnée dans [GD 87] et [Duf 84].

La Figure 3 ci-dessous représente l'ARBRE D'UNIFICATION obtenu pour les 3 règles de production déjà étudiées:

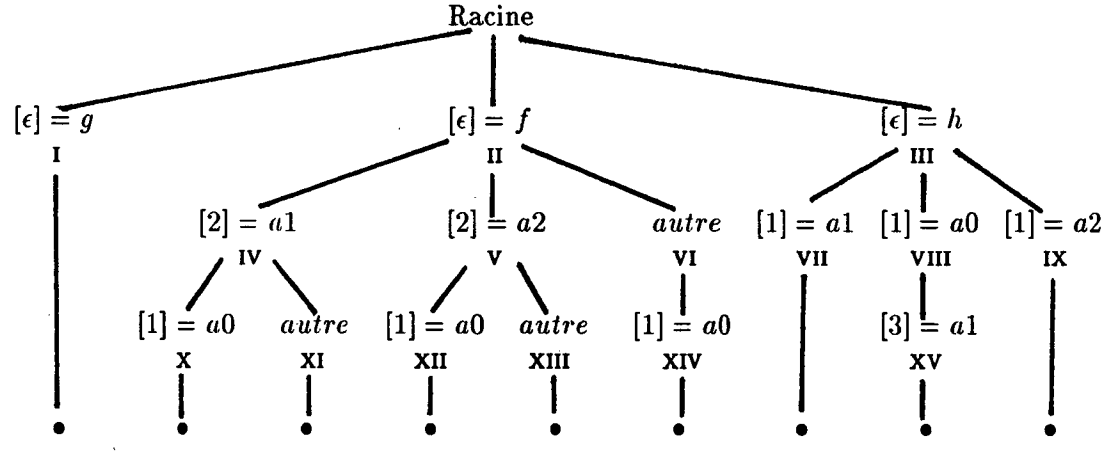


FIGURE 3 :ARBRE D'UNIFICATION

On n'a pas représenté la partie graphe de jointure, qui reste identique à celle de la Figure 1; en entrée droite du nœud **12**, en entrée gauche et droite du nœud **14**, il faut réunir respectivement les mémoires des nœuds (X, XII, XIV), (X, XI) et (XII, XIII).

Dans l'ARBRE D'UNIFICATION, un terme est filtré par un nœud "*autre*" fils d'un nœud i, si les tests de tous les autres fils de i ont été des échecs; ainsi le terme $(f\ a0\ a5)$ effectue le parcours II, VI, XIV. On remarque que le seul chemin du terme $(f\ a0\ a1)$ est II, IV, X.

En ce qui concerne l'évaluation de la performance de cet algorithme, l'intérêt n'est pas tant dans ce parcourt unique du graphe (puisque l'on sait que de toute façon ce sont les opérations effectuées aux jointures qui constituent la partie principale du temps d'exécution) mais dans cette partition de la Base de Faits aux feuilles de l'ARBRE D'UNIFICATION. Celle-ci va en effet nous permettre de reprendre le calcul combinatoire du Théorème 4 du nombre moyen de l-uplets de termes qui sont filtrés en un nœud i du graphe de jointure.

On conserve les notations précédentes. On a jusqu'à présent considéré une Base de Faits en entrée quelconque qui pouvait donc contenir notamment des faits filtrés par aucun des motifs des règles de la Base de Règles. On va donc pour s'adapter à l'hypothèse de l'ARBRE D'UNIFICATION considérer encore une Base de Faits quelconque et on va décomposer $A(z)$ en $A(z) = U(z) + R(z)$ avec $U(z)$ la série génératrice qui compte tous les motifs filtrés aux feuilles de l'ARBRE D'UNIFICATION

et $R(z)$ la "série reste". De cette façon, et puisqu'il y a une partition de tous les éléments filtrés aux feuilles de l'ARBRE D'UNIFICATION on peut écrire en numérotant les feuilles de l'ARBRE D'UNIFICATION de 1 à p:

$$A(z) = \sum_{j=1}^{p} B_j(z) + R(z)$$

(ce que la duplication éventuelle dans l'algorithme RETE classique empêchait d'écrire).

On cherche à déterminer le nombre moyen de termes filtrés en un nœud i du graphe de jointure. Pour cela on va déterminer dans un premier temps le nombre moyen de l-uplets de termes dont chacune des composantes filtre une des l feuilles de l'ARBRE D'UNIFICATION associées à i c-à-d le nombre moyen de l-uplets de termes qui passeraient par i si dans le graphe de jointure ne s'effectuait aucun test multi-motifs ou encore si aux nœuds-jointure ne s'effectuaient que des produits cartésiens des 2 entrées. Une fois cette quantité moyenne de l-uplets déterminée on calculera le nombre moyen de ceux-ci qui sont filtrés jusqu'en i et l'on pourra alors déduire finalement le nombre moyen réellement filtré en un nœud i du graphe de jointure. On a:

THÉORÈME 15 : *En considérant comme entrée du graphe une forêt de k termes de taille globale n, le nombre moyen de l-uplets de termes dont chacune des composantes est filtrée par une des l feuilles de l'*ARBRE D'UNIFICATION *associées à un nœud i du graphe de jointure est:*

$$\overline{c_{i,n,k}} = \frac{k(k-1)\dots(k-l+1)[z^n]A^{k-l}(z)\prod_{j^i=1}^{l} B_{j^i}(z)}{[z^n]A^k(z)}$$

*les j^i étant les l feuilles de l'*ARBRE D'UNIFICATION *qui filtrent chacune une des composantes des l-uplets testés en i.*

PREUVE : Puisqu'il n'y a pas de duplication des termes aux feuilles de l'ARBRE D'UNIFICATION, on peut écrire:

$$A(z) = \sum_{j^i=1}^{l} B_{j^i}(z) + R'(z)$$

avec $R'(z)$ la série reste.
Comme dans la démonstration du Théorème 3, on marque par la variable u_{j^i} les termes filtrés par la j^i ^{eme} feuille de l'ARBRE D'UNIFICATION. Ainsi on définit la série génératrice

$$A(z, u_{1^i}, \dots, u_{l^i}) = \sum_{j^i=1}^{l} u_{j^i} B_{j^i}(z) + R'(z) = A(z) + \sum_{j^i=1}^{l} (u_{j^i} - 1) B_{j^i}(z)$$

Alors $f_{i,n,p_1,\dots,p_l,k} = [z^n u_{1^i}^{p_1} \dots u_{l^i}^{p_l}](A^k(z, u_{1^i}, \dots, u_{l^i}))$ est le nombre de forêts de taille n de k termes parmi lesquels $\forall j$ il y a exactement p_j termes qui sont filtrés en la j^i ^{eme} feuilles de l'ARBRE D'UNIFICATION. D'où puisque ceci conduit à $p_1 \times \dots \times p_l$ l-uplets:

$$\overline{b_{i,n,k}} = \frac{\sum_{p_1,\dots,p_l} p_1 \times \dots \times p_l f_{i,n,p_1,\dots,p_l,k}}{[z^n]A^k(z)}$$

Or on a

$$\sum_{p_1,\dots,p_l} p_1 \times \dots \times p_l f_{i,n,p_1,\dots,p_l,k} = \frac{\partial}{\partial u_{1^i}} \dots \frac{\partial}{\partial u_{l^i}} (A^k(z, u_{1^i}, \dots, u_{l^i}))\Big|_{u_{j^i}=1}$$

$$= k(k-1)\dots(k-l+1)[z^n]A^{k-l}(z) \prod_{j^i=1}^{l} B_{j^i}(z)$$

on a donc bien le résultat. ∎

Déterminons à présent, en un nœud i du graphe de jointure réel (avec les tests multi-motifs), le nombre moyen de l-uplets de termes qui filtrent en i à partir de L l-uplets de termes dont chacune des composantes filtre un des $B_{j^i}(z)$. On a:

THÉORÈME 16 : *En un nœud i du graphe de jointure, le nombre moyen de l-uplets de termes filtrés en i à partir d'une liste de taille n de L l-uplets de termes dont chacune des composantes filtre une des feuilles de l'*ARBRE D'UNIFICATION *associée à i est:*

$$\overline{\gamma_{i,L,n}} = L\frac{[z^n]B_i(z)\Big(\prod_{j^i=1}^{l} B_{j^i}(z)\Big)^{L-1}}{[z^n]\Big(\prod_{j^i=1}^{l} B_{j^i}(z)\Big)^{L}}$$

*les j^i étant les l feuilles de l'*ARBRE D'UNIFICATION *qui filtrent chacune une des composantes des l-uplets testés en i.*

PREUVE : La série génératrice des l-uplets dont chacune des composantes filtre une feuille de l'ARBRE D'UNIFICATION associée à i est:

$$D(z) = \prod_{j^i=1}^{l} B_{j^i}(z)$$

si l'on marque par w les l-uplets qui filtrent jusqu'en i (avec les tests multi-motifs) on obtient:

$$D(z,w) = (w-1)B_i(z) + \prod_{j^i=1}^{l} B_{j^i}(z)$$

D'où, si l'on considère une liste de L l-uplets, le nombre de telles L-listes de taille n qui comportent r l-uplets exactement qui filtrent en i est $[w^r z^n]D^L(z,w)$.
On a donc

$$\overline{\gamma_{i,L,n}} = \frac{\sum_{r=1}^{L} r[w^r z^n]\Big((w-1)B_i(z) + \prod_{j^i=1}^{l} B_{j^i}(z)\Big)^{L}}{[z^n]\Big(\prod_{j^i=1}^{l} B_{j^i}(z)\Big)^{L}}$$

Et donc comme pour le Théorème 3, on obtient bien l'expression de $\overline{\gamma_{i,L,n}}$. ∎

On déduit facilement de ces Théorèmes le nombre moyen de l-uplets de termes filtrés en un nœud i du graphe de jointure à partir d'une forêt d'entrée de l'algorithme de k termes de taille globale n. En effet, on applique le Théorème 16 avec une liste de $\overline{c_{i,n,k}}$ l-uplets de taille $\overline{c_{i,n,k}}/\lambda$. Sous la forme combinatoire exacte cela ne semble pas aisée, on simplifie donc les expressions obtenues en effectuant de l'analyse asymptotique sur la taille de la forêt d'entrée. On obtient alors:

THÉORÈME 17 : *Avec une forêt d'entrée de k termes de taille n et dans le cas (pratique) où $\rho_1 = min\{\rho_j^{\frac{1}{j}}\ ,\ j \geq 1\}$,*

*- Le nombre moyen de l-uplets de termes dont chacune des composantes est filtrée par une des l feuilles de l'*ARBRE D'UNIFICATION *associée à un nœud i du graphe de jointure est:*

$$\overline{c_{i,n,k}} = W_i \lambda^l \left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{P_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{\nu_{i,j} - c_{i,j}}(\rho_1^j)\right) n^l \left(1 + O\left(\frac{1}{n^2}\right)\right)$$

- *En un nœud i du graphe de jointure, le nombre moyen de l-uplets de termes filtrés en i à partir d'une liste de L l-uplets de termes donc chacune des composantes filtre une des feuilles de l'*ARBRE D'UNIFICATION *associée à i est:*

$$\overline{\gamma_{i,L,n}} = L \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{c_{i,j}}(\rho_1^j)\right) \left(1 + O\left(\frac{1}{n^2}\right)\right)$$

- *D'où, pour l'*ARBRE D'UNIFICATION*, le nombre moyen de l-uplets de termes filtrés en un nœud i du graphe de jointure est:*

$$\overline{b_{i,n,k}} = W_i \lambda^l \left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{P_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{\nu_{i,j}}(\rho_1^j)\right) n^l \left(1 + O\left(\frac{1}{n^2}\right)\right)$$

PREUVE : En utilisant le fait que

$$\prod_{j'=1}^{l} B_{j'}(z) \prod_{j=2}^{\Omega_i - h_i + l} A_j^{c_{i,j}}(z) = B^l(z) A^{-c_{i,1}}(z)$$

les calculs, similaires à ceux du Théorème 10 sont (presque) immédiats. ∎

Remarque : Le lecteur ne manquera pas de constater que le résultat obtenu est le même que dans le cas de l'algorithme RETE classique. En effet il est normal de retrouver la même formule pour l'estimation moyenne des mémoires dans le graphe de jointure puisque celui-ci, dans la version de l'ARBRE D'UNIFICATION, reste classique. Mais en ce qui concerne le calcul de la complexité globale de l'algorithme, il faut prendre en compte, pour comparer les versions, "l'allègement" apporté par l'ARBRE D'UNIFICATION dans la partie arbre test.

Cette identité entre les deux résultats nous apporte notamment deux informations:

- D'une part, que l'hypothèse de distribution uniforme du Théorème 4 est bonne puisque les calculs dans le cas particulier de l'ARBRE D'UNIFICATION ont été menés sans elle;
- Et d'autre part une interprétation intéressante du résultat général du Théorème14, en effet on sait que la taille de la mémoire locale de sortie des nœuds-jointure est conditionnée par deux phénomènes: l'expansion due au produit cartésien des données d'entrée et la diminution due aux tests multi-motifs effectués en ce nœud-jointure. Le raisonnement précédent nous permet de distinguer ces deux composantes dans l'expression de la taille moyenne des l-uplets de termes filtrés en un nœud-jointure i, on a

$$\overline{b_{i,n,k}} = \underbrace{W_i \lambda^l \left(\frac{\sigma^{1-\lambda}}{\Phi_1(\sigma)}\right)^{P_i} \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{\nu_{i,j} - c_{i,j}}(\rho_1^j)\right)}_{expansion} \underbrace{n^l \left(\prod_{j=2}^{\Omega_i - h_i + l} A_j^{c_{i,j}}(\rho_1^j)\right)}_{diminution} \left(1 + O\left(\frac{1}{n^2}\right)\right)$$

On trouvera en Annexe l'étude numérique menée sur l'ARBRE D'UNIFICATION de la Figure 3.

7. CONCLUSION

La complexité en moyenne de l'algorithme RETE est donnée par la formule (1) de la section 3.4 avec une loi de distribution uniforme des symboles dans les faits et les motifs. La formule (1) nous donne le coût moyen en temps de l'algorithme RETE pour déterminer l'ensemble des règles instanciées à partir de la Base de Faits initiale et d'une Base de Règles fixée. Ce résultat est donné en fonction de la taille moyenne des mémoires locales du graphe de RETE. L'expression de ces résultats est donnée dans le Théorème 10 et peut être calculée en fonction des paramètres de n'importe quelle Base de Faits et Base de Règles (avec par exemple un outil de calcul formel comme Maple ou Macsyma).

On considère ensuite une fonction de poids qui associe à chaque symbole sa fréquence d'apparition dans les faits ou les motifs. Le théorème 14 donne alors avec ce modèle une estimation de la taille moyenne des mémoires locales aux jointures et la probabilité pour un terme quelconque d'être mémorisé en un nœud donné. Ici encore ces quantités peuvent être déterminées à partir des caractéristiques de n'importe quelle Base de Faits et Base de Règles. Ces résultats théoriques seront naturellement très bientot confrontés à l'expérimentation (des expérimentations voisines ont déjà été menées dans le cas du langage LISP par Clark, Cf [Clark 79]). Ces résultats théoriques nous permettent d'ailleurs de proposer, comme optimisation simple de l'algorithme, de réordonner les nœuds du graphe de RETE afin de diminuer les tailles des mémoires locales de la racine aux nœuds terminaux.

On a étudié la négation dans la partie 5. On pourra encore travailler afin de prendre en compte d'autres prédicats (comme l'inégalité arithmétique ...) ou d'autres tests quelconques. Comme on l'a dit, les résultats obtenus s'étendent sans peine lorsque l'on effectue un hachage des mémoires locales; les résultats peuvent s'adapter à beaucoup de variantes de l'algorithme RETE. La variante utilisant l'ARBRE D'UNIFICATION est présentée et étudiée dans la partie 6 de cet article. On montre dans le théorème 17 que la formule donnant la taille moyenne des mémoires locales est inchangée. Ce résultat nous permet de mieux expliquer l'expression obtenue, et cette interprétation de la formule est prometteuse pour les heuristiques optimisantes du graphe de RETE.

On montre donc que la théorie des fonctions génératrices est une méthode très générale pour l'analyse précise (et en moyenne !) des algorithmes. Le modèle développé dans cet article est donc intéressant pour les Théorèmes combinatoires et analytiques démontrés pour l'achèvement des calculs et comme partie de cette vaste et puissante théorie.

8. REMERCIEMENTS

Je tiens à remercier Ph. FLAJOLET, Fr. FAGES pour m'avoir dirigé pendant ce travail si intéressant, toute l'équipe du projet ALGO de l'INRIA ainsi que l'équipe de l'école Polytechnique pour leur aide et enfin Ma. GHALLAB du LAAS Toulouse pour son aide sur l'ARBRE D'UNIFICATION et ses encouragements.

9. BIBLIOGRAPHIE

[Alb 87] L. Albert "Présentation et évaluation de la complexité de l'algorithme RETE de multi-pattern matching dans les systèmes de règles de production", rapport de DEA Université de PARIS XI, ENS, rapport de recherche 87-8 LCR Thomson-CSF, 1987.

[AF 87] L. Albert, F. Fages "Average case complexity analysis of the RETE pattern match algorithm", rapport de recherche INRIA 773, décembre 87.

[dB 58] NG de Bruijn *Asymptotic Methods in Analysis* Dover 1958.

[CKS] C.Choppy, S.Kaplan,M.Soria "Algorithmic complexity of term rewriting systems", proceedings of the first Conference on Rewriting Techniques and Applications, Dijon, France. 1986.

[Clark 79] D.W. Clark "Measurements of Dynamic List Structures Use in Lisp". IEEE Transactions on Software Engineering SE-5(1), pp.51-59. (Jan. 1979).

[Dieu 68] J. Dieudonné *Calcul infinitésimal* Hermann 1968.

[Duf 84] P. Dufresne "Contribution algorithmique à l'inférence par règles de production", Thèse Université Paul Sabatier Toulouse 1984.

[DNM 78] J. McDermott, A. Newell, J. Moore "The efficiency of certain production system implementations", dans *Pattern-Directed Inference Systems* (Waterman et Hayes-Roth ed.) Academic Press, New York, 1978, pp 155-176. de production", Thèse Université Paul Sabatier Toulouse 1984.

[Fag 86] F. Fages "On the proceduralization of rules in expert systems", First France-Japan Symposium on Artificial Intelligence, Tokyo, Nov. 86. In Programming of Future Generation Computers, Addison-Wesley, Eds. M. Nivat and K. Fuchi.

[Fla 85] P. Flajolet "Mathematical methods in the analysis of algorithms and data structures", INRIA, Research Report 400,1985. To appear in the Mathematical Handbook of Theoretical Computer Science, North-Holland.

[Fla 87] P. Flajolet "The symbolic operator method", *in Mathematical methods in the analysis of algorithms and data structure*, L.N.C.S., Springer Verlag, to appear 1987.

[Forg 79] C. Forgy "On the efficient implementation of production systems", PhD Thesis Carnegie Mellon University 1979.

[Forg 81] C. Forgy "OPS-V user's manual", Computer Science Department, Carnegie Mellon University, Pittsburgh, MA, 1981.

[Forg 82] C. Forgy "RETE, a fast algorithm for the many patterns many objects Match problem", *Artificial Intelligence* 19, 1982,pp 17-37.

[FS 86] P. Flajolet, R. Sedgewick "Mathematical analysis of algorithms", Computer Science 504, lecture Notes for Princeton University 1986.

[FSS] P. Flajolet, P. Sipala et J.M. Steyaert "The analysis of tree compaction in symbolic manipulations", preprint.

[GD 84] M. Ghallab, P. Dufresne "Moteurs d'inférences pour systèmes de règles de production : techniques de compilation et d'interprétation", Colloque d'Intelligence Artificielle, Marseille,Oct. 1984, pp 89-103.

[GF 83] A. Gupta et C.L. Forgy "Measurements on production systems", Carnegie Mellon University Technical Report CMU-CS-83-167,1983.

[Gha 80] M. Ghallab "New optimal decision tree for matching patterns in inference and planning system", 2nd Int. Meeting on Artificial Intelligence, Leningrad, Oct. 1980.

[Gupt 84] A. Gupta "Parallelism in production Systems : the sources and the expected Speed-up", Carnegie Mellon University Technical Report CMU-CS-84-169,1984.

[Henr 74] P. Henrici *Applied and Computational complex Analysis* Volumes 1-3 Wiley New-York.

[HWL 83] F. Hayes-Roth, D.A.Waterman and D.A. Lenat. *Building Expert Systems.* Addison-Wesley, Reading, M.A. (1983).

[LNR 87] J.E. Laird, A. Newell and P.S. Rosenbloom. "SOAR: An Architecture for General Intelligence", *Artificial Intelligence* 33, pp 1-64. (1987).

[MAPLE 85] B.W. Char, K.O. Geddes, G.H. Gonnet, S.M. Watt, *MAPLE: Reference Manual*, University of Waterloo, 1985.

[Mir 87] D.P. Miranker "TREAT: A Better Match Algorithm for AI Production Systems" Proceedings of the 1987 National Conference on Artificial Intelligence. Seattle, Washington. (1987).

[MM 78] A. Meier, J.W. Moon "On the altitude of nodes in random trees",*Canadian Journal of Math* 30,1978,pp 997-1015.

[SBM 83] Stefik M., Bobrow D., Mittal S. and Conway L "Knowledge Programming in LOOPS: Report on an Experimental Course" The AI magazine Fall 83, pp 3-13. (1983).

[SF 88] Schang T. and Fages F "A Real-Time Expert System for On-Board Radar Identification" 55th Symposium AVP-AGARD on Software Engineering and its Applications to Avionics.(1988).

[SF 83] J.M. Steyaert, P. Flajolet "Patterns and Pattern Matching in trees : an analysis", *Information and Control* 58,1983,pp 19-58.

[Stey 84] J.M. Steyaert "Complexité et structures des algorithmes", Thèse d'Etat Université de Paris 7 1984.

[Vien 86] G. Viennot "La combinatoire bijective par l'exemple", Université de Bordeaux 1, 1986.

[War 86] Warren D.H.D. An abstract Prolog Instruction Set. Technical Note 309, SRI International. 1985.

[WGF 86] Wright, M.L., Green M.W., Fiegl G., Cross P.F., "An Expert System for Real-Time Control" SRI International, in IEEE Software. (March 1986).

ANNEXE

Calcul des taux de filtrage des différents nœuds des Figures 1 et 3 dans le modèle uniforme.

Comme on l'a dit on a

$$\Phi(X) = X^3 + 2X^2 + 6$$

et on calcule la taille moyenne escomptée $\lambda = 9$, $\tau = 1.17456$, $\sigma = 1.08669$ et $\rho = 0.11316$. On a:

nœud	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
$\overline{\alpha_{i,n}}$	1.1163e-1	1.1163e-1	1.1163e-1	1.2461e-2	1.2461e-2	1.2461e-2	1.2461e-2	1.2461e-2
nœud	**9**	**10**	**11**	**12** *	**13**	**14**	**15**	**16**
$\overline{\alpha_{i,n}}$	1.2461e-2	1.3911e-3	1.3911e-3	1.1162e-1	1.1956e-5	1.5529e-4	8.2448e-6	1.6632e-8
nœud	**17**	I	II	III	IV	V	VI	VII
$\overline{\alpha_{i,n}}$	1.1471e-8	1.1163e-1	1.1163e-1	1.1163e-1	1.2461e-2	1.2461e-2	8.6708e-2	1.2461e-2
nœud	VIII	IX	X	XI	XII	XIII	XIV	XV
$\overline{\alpha_{i,n}}$	1.2461e-2	1.2461e-2	1.3911e-3	1.1069e-2	1.3911e-3	1.1069e-2	9.6788e-3	1.3911e-3

* : On obtient le taux de filtrage au nœud **12** sachant que:

$$\overline{\alpha}((g\ \ X\ Y), \neg(f\ \ a0\ X)) = \overline{\alpha}(g\ \ X\ Y) - \overline{\alpha}((g\ \ X\ Y), (f\ \ a0\ X))$$

Calcul des taux de filtrage des différents nœuds des Figures 1 et 3 dans le modèle pondéré.

On considère la Base de Faits particulière suivante:

$$(f\ \ a0\ a1)\quad (f\ \ a0\ a2)\quad (f\ \ a4\ a3)$$

$$(g\ \ a1\ a2)\quad (g\ \ a3\ a4)\quad (g\ \ a1\ a3)$$

$$(h\ \ a0\ a5\ a1)\quad (h\ \ a1\ a3\ a2)\quad (h\ \ a2\ a1\ a4)\quad (h\ \ a1\ a3\ a4)$$

On choisit comme pondération la fréquence d'apparition des différents symboles dans la Base de Faits, soit: h:4/34, f:3/34, g:3/34, a0:3/34, a1:7/34, a2:4/34, a3:5/34, a4:4/34 et a5:1/34. On obtient alors

$$\Phi_1(X) = 2/17X^3 + 3/17X^2 + 12/17$$

et $\lambda = 6.8$, $\tau_1 = 1.23069$, $\rho_1 = 1,03206$, $\sigma = 1.11519$ et on a bien ρ_1 le minimum des $\rho_j{}^{1/j}$. On a:

nœud	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
$\overline{\alpha_{i,n}}$	8.8959e-2	8.8959e-2	1.1861e-1	7.9138e-3	1.8465e-2	1.0551e-2	2.4621e-2	1.0551e-2
nœud	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
$\overline{\alpha_{i,n}}$	1.4069e-2	1.6427e-3	2.1902e-3	8.8953e-2	2.0866e-5	1.9484e-4	2.5117e-5	3.4277e-8
nœud	**17**	I	II	III	IV	V	VI	VII
$\overline{\alpha_{i,n}}$	3.1438e-8	8.8959e-2	8.8959e-2	1.1861e-1	1.8465e-2	1.0551e-2	5.9942e-2	2.4621e-2
nœud	VIII	IX	X	XI	XII	XIII	XIV	XV
$\overline{\alpha_{i,n}}$	1.0551e-2	1.4069e-2	1.6427e-3	1.6822e-2	9.3868e-4	9.6131e-3	5.3325e-3	2.1902e-3

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique