UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

Département d'informatique et d'ingénierie

SCIENCES ET TECHNOLOGIES DE L'INFORMATION

T H È S E

pour l'obtention du grade de

Philosophiae Doctor (Ph.D.)

en Sciences et technologies de l'information

de l'Université du Québec en Outaouais

Présentée et soutenue par

Abdélilah Balamane

Découverte et gestion de motifs en analyse formelle de concepts

Thèse sous la supervision de Pr. Rokia Missaoui

soutenue le 15 Septembre 2017

Membres du jury :

Karell Bertet	**Maître de conférence HDR**	**Examinatrice externe**
Université de La Rochelle.	La Rochelle	France
Jean Vaillancourt	**Professeur affilié**	**Examinateur externe**
Hautes études commerciales	Montréal	Canada
Ana-Maria Crétu	**Professeur**	**Membre du jury**
Université du Québec	En Outaouais	Canada
Karim El Guemhioui	**Professeur**	**Président et membre du jury**
Université du Québec	En Outaouais	Canada
Rokia Missaoui	**Professeur**	**Directrice de thèse**
Université du Québec	En Outaouais	Canada

Remerciements

À la mémoire de mon épouse.
À mes très chers et adorables enfants Saad et Ali.

Cette thèse n'aurait pas été possible sans le soutien et l'aide de nombreuses personnes. Je leur témoigne ici toute ma gratitude.

Je dois ma profonde gratitude au Professeur Rokia Missaoui pour sa patience, ses très nombreux conseils et surtout son humanisme. Elle incarne la bonté et l'intelligence. Sans son soutien, son aide et ses suggestions, cette thèse n'aurait pas été possible. C'est un honneur pour moi d'avoir réalisé ce travail sous sa direction. Elle inspire le respect et l'admiration. Merci Rokia d'avoir cru en moi. Je t'en suis profondément reconnaissant.

J'aimerais aussi témoigner ma profonde reconnaissance à un homme qui inspire l'admiration et le respect pour sa simplicité, sa grande culture et l'étendue de son expertise. Merci Docteur Jean Vaillancourt d'avoir accepté d'évaluer mon travail de recherche et de faire partie de mon jury de thèse. Je suis très honoré et reconnaissant envers votre personne.

Ma gratitude va aux membres du jury : Professeur Karell Bertet de l'université de La Rochelle ainsi que les professeurs Karim Guemhioui et Ana-Maria Crétu de l'université du Québec en Outaouais.

Je tiens également à remercier Docteur Léonard Kwuida, professeur à L'université de Berne des sciences appliquées pour les nombreux et fructueux échanges en recherche.

Résumé : La découverte et la gestion de motifs font référence à un ensemble d'activités de prétraitement de données ainsi que d'extraction, de manipulation et de stockage de motifs à partir de données d'une manière similaire, mais plus élaborée que la gestion de bases de données. Un motif (*pattern*) fait partie du résultat d'une fouille de données laquelle est une étape du processus d'extraction de connaissances à partir des données. En analyse formelle de concepts, le motif prend deux principales formes : a) des concepts formels décrivant des objets/individus avec leurs attributs communs et représentant des nœuds d'un treillis de concepts (Galois), et b) des règles d'association entre des groupes d'attributs, y compris des implications.

Dans le cadre de cette recherche doctorale, nous nous sommes intéressé et avons contribué à deux thèmes en découverte et gestion de motifs. Le premier thème concerne la production de motifs (et spécialement des concepts formels) généralisés en procédant à une agrégation d'objets et/ou d'attributs d'un contexte formel initial et selon trois modes de généralisation : $\exists$, $\forall$ et α. Cinq cas de généralisation simultanée sur les objets et les attributs sont également considérés. La seconde contribution concerne le développement d'une procédure générique de co-clustering, appelée BiP, qui part d'un contexte formel binaire pour produire quatre types de co-clusters : des concepts formels décrivant la présence d'attributs, des concepts formels décrivant l'absence d'attributs, des co-clusters dont les valeurs sont une combinaison bien définie de 0 et de 1, et des blocs dont les colonnes sont pleines de 0 ou de 1 indiquant la présence de certaines propriétés mais l'absence d'autres attributs. Le cas de la production de co-clusters à partir de données discrètes ou catégorielles est également étudié.

Les divers résultats de recherche théoriques obtenus sont validés sur des données en phytothérapie et des collections de données connues.

Mots clés : Fouille de données, découverte de connaissances, analyse formelle de concepts, gestion de motifs, motifs généralisés, co-clustering.

Abstract : Pattern discovery and management cover a set of activities related to data preprocessing as well as pattern extraction, manipulation and storage from data in a similar but more elaborated way that database management. A pattern is a part of the output of a data mining task as a step of the discovery process. In Formal Concept Analysis, a pattern is mainly of two types : a) formal concepts that describe object sets with their common properties and represent nodes of a concept (Galois) lattice, and b) association rules between attributes, including implications.

In this doctoral research we were concerned and contributed to two topics in knowledge discovery and management. The first one concerns the computation of generalized patterns, mainly formal concepts, by an aggregation of objects and/or attributes of an initial formal context and according to three kinds of generalization : $\exists$, $\forall$ et α. Five cases of simultaneous generalization on both objects and attributes are also considered. The second contribution covers the design of a generic biclustering procedure called BiP which starts from a binary context and generates four types de co-clusters : formal concepts with their common properties, formal concepts with the properties they do not have, coclusters with a well-defined combination of 0 and 1 values, and coclusters with columns completely full with either 0 or 1 values to highlight the presence of some attributes but the absence of other ones. The production of coclusters from discrete and nominal data is also studied.

All the theoretical results of the present research work are validated using well-known data sets as well as phytotherapy data.

Keywords : Data mining, knowledge discovery, formal concept analysis, pattern management, generalized patterns, coclustering.

Table des Matières

Chapitre 1

Introduction

Sommaire

À l'ère des données massives (*big data*), la grande disponibilité d'importants volumes de données à moindre coût combinée avec la nécessité d'extraire des informations concises et des connaissances pertinentes, posent de nouvelles exigences pour les analystes de données dans les diverses applications du monde réel que ce soit dans les activités industrielles, scientifiques, socio-culturelles, professionnelles ou même de loisir comme les réseaux sociaux. De toute évidence, les données dans des volumes aussi énormes ne constituent pas une connaissance en soi. Il en résulte que peu de connaissances utiles peuvent être extraites simplement par leur observation. Ainsi, des techniques sont nécessaires pour extraire les connaissances cachées et les rendre accessibles pour les utilisateurs et principalement les gestionnaires. Une approche habituelle pour traiter une grande quantité de données est de les réduire aux artefacts de connaissances sous forme de *clusters*, *biclusters* ou règles d'association par des méthodes de traitement de données tout en préservant autant que possible les informations cachées/intéressantes que ces données englobent. Ces artefacts de connaissances, appelés aussi motifs, correspondent à une représentation concise et sémantiquement riche de données brutes.

La découverte et la gestion de motifs se réfèrent à un ensemble d'activités liées au prétraitement de données, à l'extraction, la description, la manipulation et au stockage de motifs d'une manière similaire (mais plus élaborée) aux données gérées par des applications de base de données. Toutefois, l'opération de découverte de motifs s'accompagne souvent d'un certain nombre de problèmes incluant principalement l'extraction et la description des motifs à partir de données de plus en plus massives et bruitées ainsi que leur manipulation et leur stockage. L'opération d'extraction produit souvent un nombre élevé de motifs qui rend difficile la sélection de ceux qui sont pertinents pour l'utilisateur. En effet, dans de nombreuses applications, les utilisateurs tendent à être noyés par un nombre de plus en plus élevé de motifs alors qu'ils sont réellement intéressés par un ensemble plus restreint de connaissances. De plus, chaque utilisateur a ses propres besoins au niveau des données et des connaissances qui y sont extraites et a tendance à utiliser un processus exploratoire et itératif de fouille de données pour découvrir des motifs sous différents scénarios et selon différentes hypothèses.

Dans le cadre de l'analyse formelle des concepts (AFC), différentes solutions ont été élaborées pour remédier au problème du nombre élevé des motifs produits. Des méthodes de

simplification, décomposition et regroupement ont été proposées afin de générer des sous-matrices plus compactes qui expriment des sous-ensembles d'objets avec leurs attributs. Rappelons que l'AFC est un formalisme de représentation et d'extraction de la connaissance se basant sur la formalisation des concepts et la hiérarchie des concepts [23]. En AFC, le motif prend deux principales formes : a) des concepts formels décrivant des objets/individus avec leurs attributs communs et représentant des nœuds d'un treillis de concepts (Galois), et b) des règles d'association entre des groupes d'attributs, y compris des implications.

Cependant, un problème récurrent en AFC est le nombre de concepts qui peut être exponentiel en fonction de la taille des données d'entrée. Pour contrôler la taille du contexte et celle du treillis de Galois correspondant, plusieurs techniques ont été proposées [3]. Une de ces techniques est de montrer seulement les concepts fréquents du treillis à la place du treillis complet [40]. D'autres techniques essaient de produire un treillis réduit en faisant une sélection au niveau des objets et/ou une projection sur les attributs [28, 45] ou en exploitant la taxonomie présente au niveau des objets et/ou des attributs pour effectuer une généralisation et donc une réduction de la dimensionnalité. En plus de potentiellement (mais pas systématiquement) réduire la taille du treillis, la généralisation sur les objets ou les attributs peut révéler des motifs généralisés, significatifs et inattendus tel que discuté dans le chapitre 3.

1.1 Objectifs

Le but principal de cette recherche doctorale est de proposer de nouvelles solutions en découverte et gestion de motifs en vue d'une part, d'obtenir un ensemble plus réduit de connaissances succinctes et pertinentes par application de la généralisation sur les objets et/ou leurs attributs, et d'autre part, de produire un ensemble de motifs générés par une nouvelle approche de co-clustering se déclinant en plusieurs variantes dont celles qui font ressortir aussi bien la présence que l'absence de propriétés. À cette fin, nous avons contribué à deux thèmes en découverte et gestion de motifs. Le premier thème concerne la production de motifs (et spécialement des concepts formels) généralisés en procédant à une agrégation d'objets et/ou d'attributs d'un contexte formel initial et selon trois modes de généralisation : $\exists$, $\forall$ et α. Cinq cas de généralisation simultanée sur les objets et les attributs sont également considérés. La seconde contribution concerne le développement d'une procédure générique de co-clustering, appelée BiP, qui part d'un contexte formel binaire pour produire quatre types de co-clusters : des concepts formels décrivant la présence d'attributs, des concepts formels décrivant l'absence d'attributs, des co-clusters dont les valeurs sont une combinaison bien définie de 0 et de 1, et des blocs dont les colonnes sont pleines de 0 ou de 1 indiquant la présence de certaines propriétés mais l'absence d'autres attributs. Le cas de la production de co-clusters à partir de données discrètes ou catégorielles est également étudié sans la nécessité d'effectuer une recodification préalable de ces données.

1.2 Organisation de la thèse

La thèse est organisée comme suit : le chapitre 2 fait d'abord un rappel sur l'analyse formelle de concepts et propose une démarche de génralisation sur les objets et/ou les attributs selon trois formes : $\exists$, $\forall$, et α ainsi que différents scénarios de généralisation simultanée sur les objets et les attributs. Le chapitre 3 illustre non seulement l'intérêt de la généralisation à découvrir de nouveaux motifs pertinents sur des données réelles en phytothérapie mais également fournit une étude empirique sur la réduction du nombre de concepts formels à la suite d'une généralisation même de type $\exists$. Quant au chapitre 4, il introduit le co-clustering, rappelle les travaux connexes ainsi que notre propre approche de co-clustering avec ses variantes en mettant l'accent sur la production de concepts formels. Le chapitre 5 présente d'autres variantes de notre procédure permettant de produire des co-clusters dits de types 2, 3 et 4 à partir d'une matrice à valeurs binaires. Il fournit également plusieurs illustrations et discussions des co-clusters produits, y compris la manière dont ces derniers peuvent servir au processus de généralisation d'objets. Le chapitre 6 traite du calcul de co-clusters à partir de données discrètes ou catégorielles alors que le chapitre 7 illustre la gestion de motifs de type co-clusters selon les besoins de l'utilisateur à travers une structure d'arbre Patricia. Finalement, une conclusion générale est donnée par le chapitre 8 avec indication des travaux que nous comptons poursuivre dans le futur.

Chapitre 2

Extraction de motifs généralisés

Sommaire

2.1 Introduction

Dans de nombreuses applications de la vie réelle, la sémantique associée aux données peut être avantageusement exploitée pour découvrir de nouvelles connaissances inattendues et pertinentes. Tel que rappelé en introduction, un motif extrait des données est une représentation concise et sémantiquement riche des données. Il peut être représenté par un groupe (*cluster*) ou concept, une règle d'association, un cas aberrant, une tendance et ainsi de suite. Dans ce chapitre, nous allons tout d'abord faire un rappel sur l'analyse formelle de concepts pour ensuite présenter quelques façons d'effectuer des agrégations par généralisation des objets et/ou des attributs à différents niveaux de granularité pour produire des concepts généralisés à l'aide de taxonomies sur des attributs et/ou des objets.

L'analyse formelle de concepts (AFC) est un formalisme de représentation de la connaissance, basé sur la formalisation des concepts et la hiérarchie de concepts [23]. Un problème récurrent en AFC est le nombre de concepts qui peut être exponentiel en fonction de la taille du contexte. Pour contrôler le volume du contexte et celui du treillis de Galois correspondant, plusieurs techniques ont été proposées [3]. Une de ces techniques est de montrer

seulement les concepts fréquents du treillis à la place du treillis complet [40]. D'autres techniques essaient de produire un treillis réduit en faisant une sélection au niveau des objets et/ou une projection sur les attributs [45] ou en exploitant la taxonomie présente au niveau des objets et/ou des attributs. En plus de potentiellement (mais pas systématiquement) réduire la taille du treillis, la généralisation sur les objets ou les attributs peut conduire à des motifs généralisés potentiellement pertinents et inattendus tel qu'illustré par le chapitre 3.

Ce chapitre est organisé comme suit. En section 2.2, nous introduisons les notions de base de l'AFC. La section 2.3 présente trois opérateurs de généralisation ainsi que différents scénarios de généralisation sur les attributs et les objets. Nous discutons aussi des problèmes de la visualisation des motifs généralisés et nous discutons du sens réel des trois cas de généralisation. En section 2.4, la taille de l'ensemble des concepts généralisés est comparée à celle de l'ensemble des concepts initiaux (c.-à-d. avant la généralisation). Un survol des travaux reliés au couplage de l'AFC avec les ontologies est brièvement fourni en section 2.5.

2.2 Analyse formelle de concepts

En analyse formelle des concepts, un contexte formel est un triplet $\mathbb{K} := (G, M, I)$ où G, M et I représentent respectivement un ensemble d'objets, un ensemble d'attributs et une relation binaire entre les éléments de G et M. Un concept formel est une paire (A, B) tel que B est l'ensemble de toutes les propriétés partagées par les objets en A, et A est l'ensemble de tous les objets qui ont l'ensemble des propriétés en B. L'expression $(g, m) \in I$ ou encore gIm signifie que l'objet g possède l'attribut m. On note $A' := \{m \in M \mid gIm \textit{ pour tout } g \in A\}$ et $B' := \{g \in G \mid gIm \textit{ pour tout } m \in B\}$. On dit que (A, B) est un concept de $\mathbb{K}$ si et seulement si $A' = B$ et $B' = A$. L'extension de (A, B) est A (notée aussi $Ext(A, B)$) et B en est l'intention. Notons respectivement $\mathfrak{B}(\mathbb{K})$, Int($\mathbb{K}$) et Ext($\mathbb{K}$) l'ensemble des concepts, l'ensemble des intentions et la collection des extensions de $\mathbb{K}$ respectivement. Un sous-ensemble X est dit fermé si $X'' = X$. Les fermetures des sous-ensembles de G constituent Ext($\mathbb{K}$) alors que les fermetures des sous-ensembles de M déterminent Int($\mathbb{K}$).

La figure 2.1 représente un contexte qui décrit huit transactions (clients) relatives à l'achat des articles $a, ..., h$ par des clients, un cas typique de l'analyse du panier de consommateurs (*basket market analysis*).

La hiérarchie des concepts est formalisée à l'aide de la relation $\leq$ définie sur $\mathfrak{B}(\mathbb{K})$ par $A \subseteq C \Longleftrightarrow : (A, B) \leq (C, D) : \Longleftrightarrow B \supseteq D$. C'est une relation d'ordre partiel appelée aussi relation de spécialisation/généralisation sur les concepts. En fait, un concept (A, B) est une spécialisation du concept (C, D), ou (C, D) est une généralisation de (A, B) ssi $(A, B) \leq (C, D)$ est vérifiée.

Pour tout ensemble $\mathcal{C}$ de concepts de $\mathbb{K}$, il y a un concept $\mathfrak{u}$ de $\mathbb{K}$ qui est plus général que tout autre concept de $\mathcal{C}$ et une spécialisation de toute généralisation de tous les concepts dans $\mathcal{C}$ ($\mathfrak{u}$ est le *supremum* de $\mathcal{C}$ et est noté par $\bigvee \mathcal{C}$), et il y a un concept $\mathfrak{v}$ de $\mathbb{K}$ lequel est une spécialisation de tout concept dans $\mathcal{C}$ et une généralisation de toute spécialisation

$\mathbb{K}$	a	b	c	d	e	f	g	h
1	×				×		×	
2	×				×	×		×
3	×	×			×	×	×	
4		×			×	×	×	×
5	×		×	×				
6	×	×	×	×				
7		×	×				×	
8		×	×	×			×	

FIGURE 2.1 – Contexte Formel

de tous les concepts dans $\mathcal{C}$ ($\mathfrak{v}$ est l'*infimum* de $\mathcal{C}$ et est noté par $\bigwedge \mathcal{C}$)[1]. $\mathfrak{B}(\mathbb{K})$ est alors un *treillis complet* du contexte formel $\mathbb{K}$.

Pour $g \in G$ et $m \in M$, on établit $g' := \{g\}'$ and $m' := \{m\}'$. Les concepts objet $(\gamma g := (g'', g'))_{g \in G}$ et les concepts attributs $(\mu m := (m', m''))_{m \in M}$ forment les blocs de base de $\mathfrak{B}(\mathbb{K})$. En effet, chaque concept de $\mathbb{K}$ est un supremum de quelques γg et un infimum de quelques μm[2].

La taille du treillis de concepts peut être extrêmement grande et théoriquement exponentielle en fonction du nombre d'objets du contexte. Pour prendre en charge une telle taille, plusieurs techniques ont été proposées pour la décomposition de contextes (Atlas, directe ou sous-directe) [34], ou l'élagage ou la réduction de treillis (treillis iceberg, diagrammes imbriqués). Nous pensons que l'emploi de la taxonomie sur les objets et les attributs peut contribuer à extraire des motifs généralisés, pertinents et imprévus et dans la plupart des cas contribuer à réduire la taille de l'ensemble des motifs générés.

Un treillis de concept peut être représenté par un diagramme de Hasse où chaque nœud est un concept et un arc entre deux nœuds décrit une relation d'ordre partiel. L'étiquetage des nœuds peut être complet ou réduit. Dans le premier cas, chaque nœud affiche au complet l'extension et l'intention d'un concept. Dans le deuxième cas (cf. figure 2.2), l'extension du concept représentée par le nœud a est obtenue par les objets de G rencontrés en parcourant les chemins allant du nœud a vers l'infimum. Quant à l'intention du nœud, elle est obtenue par tous les attributs de M retrouvés lors du parcours des chemins de a vers le supremum. Par exemple, l'étiquette 5 dans la partie gauche de la figure 2.2 représente le concept objet $\gamma 5 = (\{5, 6\}, \{a, c, d\})$. Le diagramme de Hasse est un outil précieux pour la visualisation d'un treillis. Cependant, produire un bon schéma pour des structures complexes est un grand défi [21]. Par conséquent, des travaux de recherche ont été menés pour avoir un résultat plus compact en réduisant la taille de l'entrée ou en proposant une structure imbriquée des diagrammes (*nested line diagrams*) comme moyen de visualiser les concepts selon diverses perspectives et groupes d'attributs [34]. Avant de passer aux formes de

[1]Pour deux concepts x_1 et x_2, $x_1 \vee x_2 := \bigvee\{x_1, x_2\}$ et $x_1 \wedge x_2 := \bigwedge\{x_1, x_2\}$.

[2]Pour $(A, B) \in \mathfrak{B}(G, M, \mathrm{I})$ nous avons $\bigvee_{g \in A} \gamma g = (A, B) = \bigwedge_{m} \mu m$.

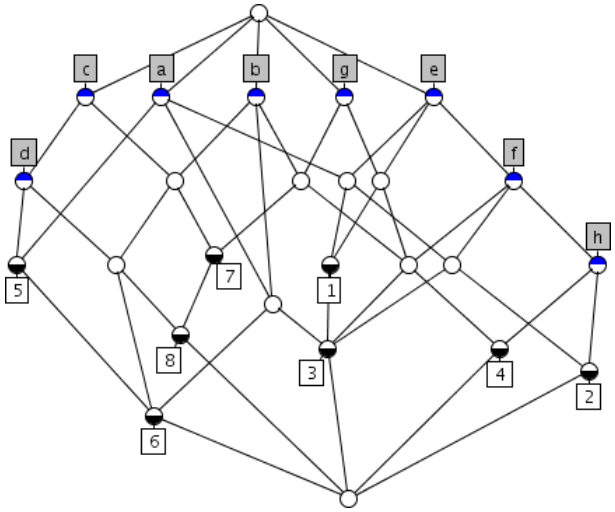


FIGURE 2.2 – Treillis de concepts du contexte de la figure 2.1

généralisation, voyons comment les données sont transformées en contextes binaires en AFC.

Fréquemment, les données ne sont pas codées dans un format binaire, mais plutôt dans un format multi-valué, *i.e.*, un tuple (G, M, W, I) tel que G est un ensemble d'objets, M un ensemble d'attributs, W un ensemble des valeurs d'attributs, $I \subseteq G \times M \times W$ et chaque $m \in M$ est un passage partiel de G vers W avec $(g, m, w) \in \mathrm{I}$ ssi $m(g) = w$. Un contexte multi-valué peut être transformé en un contexte binaire à travers une opération appelée échelonnage conceptuel (*conceptual scaling*).

Une échelle conceptuelle pour un attribut m de (G, M, W, I) est un contexte binaire $\mathbb{S}_m := (G_m, M_m, \mathrm{I}_m)$ tel que $m(G) \subseteq G_m$. Intuitivement, M_m discrétise ou groupe les valeurs d'attributs en $m(G)$ et I_m décrit comment chaque valeur d'attribut $m(g)$ est reliée aux éléments de M_m. Pour un attribut m de (G, M, W, I) et une échelle conceptuelle $\mathbb{S}_m$, nous dérivons un contexte binaire $\mathbb{K}_m := (G, M_m, \mathrm{I}^m)$ avec $g \,\mathrm{I}^m\, s_m : m(g) \,\mathrm{I}_m\, s_m$, où $s_m \in M_m$. Cela signifie qu'un objet $g \in G$ est dans une relation avec un attribut échelonné (*scaled attribute*) s_m ssi la valeur de m sur g est en relation avec s_m dans $\mathbb{S}_m$. Avec une échelle conceptuelle *(conceptual scale)* pour chaque attribut, nous obtenons le contexte dérivé $\mathbb{K}^S := (G, N, \mathrm{I}^S)$ où $N := \bigcup\{M_m \mid m \in M\}$ et $g \,\mathrm{I}^S\, s_m \iff m(g) \,\mathrm{I}^m\, s_m$. Dans la pratique, l'ensemble des objets reste inchangé et chaque nom d'attribut est remplacé par les attributs $s_m \in M_m$. Le choix d'une échelle de transformation dépend de l'interprétation désirée et se fait souvent sur recommandation d'un expert du domaine. Les méthodes présentées dans la section sur la généralisation des motifs sont en fait une forme d'échelonnage conceptuel.

2.3 Motifs généralisés

Dans un processus de fouille de données, les motifs généralisés représentent des connaissances extraites des données suite à la prise en compte de taxonomies sur les objets et/ou

les attributs. Dans ce qui suit, nous formalisons la façon dont les motifs généralisés sont produits. Soit $\mathbb{K} := (G, M, \mathrm{I})$ un contexte, les attributs de $\mathbb{K}$ peuvent être groupés ensemble pour former un autre ensemble S d'attributs. En analyse du panier du consommateur, les articles peuvent être généralisés en lignes de produits et par la suite en catégories de produits. De même, les clients peuvent être agrégés pour former des groupes répondant à des caractéristiques spécifiques (ex : analystes, programmeurs et techniciens). Le contexte $\mathbb{K}$ est alors remplacé par un contexte (G, S, J) comme décrit dans le processus d'échelonnage conceptuel où S peut être vu comme un ensemble d'indices tels que $\{m_s \mid s \in S\}$ couvre M. Dans la suite, nous identifions le groupe m_s par l'indice s.

2.3.1 Types de généralisation

Il y a principalement trois façons d'exprimer la relation J mentionnée précédemment :

($\exists$) $g \,\mathrm{J}\, s : \exists m \in s,\ g \,\mathrm{I}\, m$. Considérons une table d'information décrivant des compagnies et leur succursales aux États-Unis. Nous avons d'abord mis en place un contexte dont les objets sont les entreprises et dont les attributs sont les villes où ces entreprises ont des succursales. S'il y a trop de villes, nous pouvons décider de les regrouper en états pour réduire le nombre d'attributs. Alors, le nouvel ensemble d'attributs est maintenant un ensemble S dont les éléments sont des états. Il est tout à fait naturel de dire qu'une entreprise g a une succursale dans un état s si g a une succursale dans une ville m qui appartient à l'état s.

($\forall$) $g \,\mathrm{J}\, s \iff \forall m \in s,\ g \,\mathrm{I}\, m$. Considérons un système d'information relatif à des étudiants de doctorat et les composantes de l'examen de synthèse (ES). Supposons que les composantes sont les suivantes : la partie écrite (p), la partie orale (q), et la proposition de thèse (r). Aussi, un étudiant obtient un succès à son examen s'il réussit toutes les composantes de cet examen. Les objets du contexte sont les étudiants de doctorat et les attributs sont les différents examens pris par les étudiants. Si nous regroupons ensemble les différentes composantes, par exemple

$$ES.p, ES.q, ES.r \mapsto ES.examen,$$

alors il devient naturel d'affirmer qu'un étudiant g réussit son examen complet $ES.examen$ s'il réussit toutes les parties de ES.

($\alpha\%$) $g \,\mathrm{J}\, s \iff \frac{|\{m \in s \mid g \,\mathrm{I}\, m\}|}{|s|} \geq \alpha_s$ où α_s est un seuil fourni par l'usager pour l'attribut généralisé s. Ce cas généralise le cas-($\exists$) avec $\alpha = \frac{1}{|M|}$ et le cas-($\forall$) avec $\alpha = 1$.

Pour illustrer ce cas, considérons un contexte décrivant différentes spécialisations dans un programme de maîtrise. Pour chaque programme, il y a un ensemble de cours obligatoires et un autre de ceux qui sont optionnels. En outre, il y a un nombre prédéfini de cours que l'élève doit réussir pour obtenir le diplôme dans une spécialisation donnée. Supposons que, pour obtenir une maîtrise en informatique avec une spécialisation en "génie logiciel", l'étudiant doit réussir sept cours à partir des

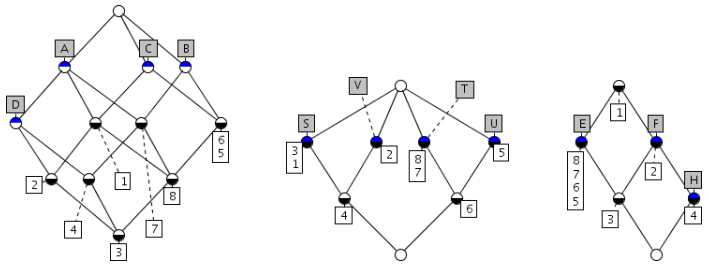


FIGURE 2.3 – Contexte généralisé des treillis de concepts dans la Table 2.1. ∃-généralisation (à gauche), ∀-généralisation (milieu) and α-généralisation avec $\alpha := 60\%$ (à droite).

cours obligatoires du groupe s_1 et trois cours à partir d'un ensemble de cours optionnels contenus dans s_2. Alors, nous pouvons introduire deux attributs généralisés s_1 et s_2. Aussi, l'étudiant g passe le groupe s_1 s'il réussit au moins sept cours de s_1 et passe le groupe s_2 s'il réussit dans au moins trois cours de s_2. Aussi, $\alpha_{s_1} := \frac{7}{|s_1|}$,$\alpha_{s_2} := \frac{3}{|s_2|}$, et

$$g \,\mathrm{J}\, s_i \iff \frac{|\{m \in s_i \mid g \,\mathrm{I}\, m\}|}{|s_i|} \geq \alpha_{s_i},\ 1 \leq i \leq 2.$$

Le tableau 2.1 illustre les trois types de généralisation sur le contexte initial.

Contexte initial									∃-généralisation				∀-généralisation				α-généralisation		
	a	b	c	d	e	f	g	h	A	B	C	D	S	T	U	V	E	F	H
1	×				×		×		×		×		×						
2	×				×	×		×	×		×	×				×		×	
3	×	×			×	×	×		×	×	×	×	×				×	×	
4		×			×	×	×	×	×	×		×	×			×		×	×
5	×		×	×						×	×				×		×		
6	×	×	×	×						×	×			×	×		×		
7		×	×					×		×	×			×			×		
8		×	×	×			×		×	×	×			×			×		

TABLE 2.1 – Trois généralisations du contexte de la figure 2.1. Le ∃-attributs généralisés sont $A := \{e, g\}$, $B := \{b, c\}$, $C := \{a, d\}$ et $D := \{f, h\}$. Le ∀-attributs généralisés sont $S := \{e, g\}$, $T := \{b, c\}$, $U := \{a, d\}$ et $V := \{f, h\}$. Le α-attributs généralisés sont $E := \{a, b, c\}$, $F := \{d, e, f\}$ et $H := \{g, h\}$ avec $\alpha = 60\%$.

La généralisation d'attributs réduit le nombre d'attributs dans le cas de la partition. On peut donc s'attendre à une réduction du nombre de concepts. Malheureusement, ce n'est pas toujours le cas. Par conséquent, il serait intéressant d'étudier sous quelle(s)condition(s) la généralisation des objets et/ou des attributs conduit à un treillis de plus petite taille que le treillis initial avant généralisation. Si les données représentent des clients (transactions) et les articles (des produits), l'utilisation d'une taxonomie sur les attributs conduit à de nouveaux motifs utiles qui pourraient ne pas être perçus avant la généralisation sur les attributs. À titre d'exemple, le cas-∃ (voir la figure 2.3 à gauche) permet d'acquérir les connaissances suivantes :

- Le client 3 (en bas du treillis) achète au moins un élément de chaque ligne de produits.
- Chaque fois qu'un client achète au moins un élément de la gamme de produits D, alors il achète nécessairement au moins un élément de la gamme de produits A.

À partir du cas-$\forall$ de la figure 2.3 (au milieu), on peut noter par exemple que les clients 4 et 6 ont des comportements distincts, dans le sens que le premier achète tous les articles des lignes de produits V et S, tandis que le deuxième achète tous les éléments des lignes de produits U et T.

Une illustration du cas-α est montrée dans la figure 2.3 (droite). On peut noter que tout client qui paie au moins 60% des articles dans H achète nécessairement au moins 60% des articles dans F. En outre, la ligne de produit E (respectivement H) semble être la plus (respectivement la moins) populaire parmi les quatre lignes de produits puisque cinq des huit clients (resp. un seul client) achètent au moins 60% d'articles dans E (resp. H). Le fait que le client 1 apparaît au niveau du supremum du treillis signifie qu'il achète moins de 60% d'article dans chacune des lignes de produits. Notons que si tous les groupes dans le cas-α ont deux éléments, alors n'importe quelle généralisation α serait soit une généralisation de type $\exists$ ($\alpha \leq 0.5$) ou une généralisation $\forall$ ($\alpha > 0.5$).

Jusqu'à maintenant, nous avons discuté de la généralisation au niveau des attributs. Un raisonnement similaire peut être effectué pour les objets. Dans ce qui suit, nous proposons quelques scénarios de généralisation simultanée sur les objets et les attributs.

2.3.2 Généralisation simultanée sur les objets et les attributs

Faite simultanément sur les attributs et les objets, la généralisation produit une sorte de *hypercontexte* (similaire à des hypergraphes [9]) puisque les objets sont des sous-ensembles de G et les attributs sont des sous-ensembles de M. Soit A un groupe d'objets et B un groupe d'attributs dans $\mathbb{K}$. La relation J entre les groupes d'objets et les groupes d'attributs peut être définie selon les cas identifiés suivants :

(1) $A \,\mathrm{J}_1\, B$ iff $\exists a \in A$, $\exists b \in B$ de sorte que $a \,\mathrm{I}\, b$, c.-à-d. quelques objets issus du groupe A sont en relation avec quelques attributs du groupe B. Autrement dit, $A{\times}B \cap \mathrm{I} \neq \emptyset$. C'est la condition la plus faible à imposer sur A et B, et probablement un cas peu utile dans des situations réelles. Son dual, J_1^d, est défini par : $A \,\mathrm{J}_1^d\, B$ ssi $\forall a \in A$, $\forall b \in B$ $a \,\mathrm{I}\, b$, ce qui signifie que $A{\times}B \cap \mathrm{I} = A{\times}B$. Les déclarations suivantes sont alors équivalentes :

(a) $A \,\mathrm{J}_1^d\, B$, (b) $A{\times}B \subseteq I$, (c) $A \subseteq B'$, (d) $B \subseteq A'$, et (e) (A, B) est un pré-concept [75].

(2) $A \,\mathrm{J}_2\, B$ ssi $\forall a \in A$, $\exists b \in B$ de sorte que $a \,\mathrm{I}\, b$, c.-à-d. chaque objet dans le groupe A a au moins un attribut dans B ; Ainsi $A \,\mathrm{J}_2\, B$ ssi $A \subseteq \bigcup\{b' \mid b \in B\}$.

Le dual, J_2^d est défini par : $A \,\mathrm{J}_2^d\, B$ iff $B \subseteq \bigcup\{a' \mid a \in A\}$. Notez que $\begin{cases} A \,\mathrm{J}_2\, B \implies |A{\times}B \cap \mathrm{I}| \geq |A| \\ A \,\mathrm{J}_2^d\, B \implies |A{\times}B \cap \mathrm{I}| \geq |B|. \end{cases}$

La réciproque est fausse.

(3) $A\,\mathrm{J}_3\,B$ ssi $\exists b \in B$ de sorte que $\forall a \in A\ a\,\mathrm{I}\,b$, c.-à-d. il y a un attribut dans le groupe B qui appartient à tous les objets du groupe A. (b est un type d'attribut *fédérateur* pour les objets dans A.) Ainsi $A\,\mathrm{J}_3\,B$ ssi $B \cap A' \neq \emptyset$. Le dual, J_3^d est défini par : $A\,\mathrm{J}_3^d\,B$ iff $A \cap B' \neq \emptyset$. En outre,

$\Big\{ A\,\mathrm{J}_3\,B \implies |A{\times}B \cap \mathrm{I}| \geq |A| A\,\mathrm{J}_3^d\,B \implies |A{\times}B \cap \mathrm{I}| \geq |B|$. La réciproque est fausse.

(4) $A\,\mathrm{J}_4\,B$ ssi $\frac{\left|\{a \in A | \frac{|a' \cap B|}{|B|} \geq \beta_B\}\right|}{|A|} \geq \alpha_A$, c.-à-d. au moins une fraction α_A d'objets du groupe A ont chacun au moins une fraction β_B d'attributs dans le groupe B. Réglons $\mathcal{A}_{\beta_B} := \{a \in A \mid |a' \cap B| \geq |B|\beta_B\}$, nous obtenons $A\,\mathrm{J}_4\,B$ iff $|\mathcal{A}_{\beta_B}| \geq \alpha_A|A|$. Ainsi $A\,\mathrm{J}_4\,B$ implique

$$|A{\times}B \cap \mathrm{I}| = \sum_{a \in A} |a' \cap B| \geq \sum_{a \in \mathcal{A}_{\beta_B}} |a' \cap B| \geq |\mathcal{A}_{\beta_B}||B|\beta_B \geq \alpha_A|A||B|\beta_B.$$

Cela signifie que la densité de $A \times B$ est au moins $\alpha_A\beta_B$. Le dual de J_4 est défini par : $A\,\mathrm{J}_4^d\,B$ ssi $\left|\left\{b \in B \mid \frac{|b' \cap A|}{|A|} \geq \alpha_A\right\}\right| \geq |B| \cdot \beta_B$. De façon similaire, $A\,\mathrm{J}_4^d\,B \implies \frac{|A{\times}B \cap \mathrm{I}|}{|A{\times}B|} \geq \alpha_A\beta_B$.

(5) $A\,\mathrm{J}_5\,B$ ssi $\frac{|A{\times}B \cap \mathrm{I}|}{|A{\times}B|} \geq \alpha$, c.-à-d. la densité du rectangle $A \times B$ est au moins α. Tous les cas ci-dessus satisfont J_5 pour une α donnée, mais ces cas ne peuvent pas être récupérés par un simple réglage de la valeur de α.

Remarque 1 : Un cas particulier ennuyeux que nous pourrions rencontrer est lorsque nous avons à déclarer qu'un groupe A d'objets est en relation avec un groupe B d'attributs bien que certains objets dans A n'ont aucun attribut dans B ou que certains attributs dans B ne sont satisfaits par aucun objets en A. Afin d'éviter cette situation, nous devrions exiger que toutes les relations que nous avons définies sur la généralisation des objets/attributs devront être des sous-relations de $\tilde{\mathrm{J}}_2 := \mathrm{J}_2 \cap \mathrm{J}_2^d$. Cela signifie que $A\tilde{\mathrm{J}}_2B$ ssi $A{\times}B$ ne contient aucune ligne ou colonne vide. Ainsi, $A\tilde{\mathrm{J}}_2B$ ssi $A \subseteq \bigcup_{b \in B} b'$ et $B \subseteq \bigcup_{a \in A} a'$.

Remarque 2 : Les relations J_4 et J_4^d sont suffisantes pour généraliser $\mathrm{J}_1, \mathrm{J}_2, \mathrm{J}_3$ ainsi que leurs relations duales. En outre, si $A\,\mathrm{J}_4\,B$ ou $A\,\mathrm{J}_4^d\,B$ alors $A\,\mathrm{J}_5\,B$ pour toute $\alpha \leq \alpha_A\beta_B$.

Un exemple de cas de généralisation sur les objets et les attributs serait celui où des clients groupés selon certaines caractéristiques communes et les articles groupés eux aussi dans des catégories de produits. Nous pouvons aussi affecter à chaque groupe tous les articles achetés par leurs membres (cas de la généralisation de type $\exists$) ou seulement leurs articles communs (cas de la généralisation $\forall$), ou juste quelques-uns des éléments fréquents parmi leurs membres (cas similaire à généralisation α). Nous pouvons également décider,

comme dans $\tilde{\mathrm{J}}_2$, d'assigner une catégorie de produit B à un groupe de clients A seulement si chaque client de A achète au moins un produit dans la catégorie B et que chaque produit de la catégorie B est acheté par au moins un client dans A.

2.3.3 Visualisation des motifs généralisés

Soit $\mathbb{K}$ un contexte formel et (G, S, J) un contexte obtenu à partir de $\mathbb{K}$ par une généralisation sur les attributs. L'opération habituelle consiste à construire directement le diagramme de Hasse de $\mathfrak{B}(G, S, \mathrm{J})$ tel qu'indiqué par la figure 2.3. On peut être intéressé à raffiner le diagramme de $\mathfrak{B}(G, S, \mathrm{J})$ par rapport aux attributs dans M, par exemple afin de récupérer le treillis de concepts de $\mathbb{K}$. Dans le cas où l'espace de stockage n'est pas une contrainte, les attributs dans M et les attributs généralisés peuvent être conservés ensemble. Ceci est réalisé en utilisant l'apposition de (G, M, I) et (G, S, J) pour obtenir $(G, M \cup S, \mathrm{I} \cup \mathrm{J})$.

Un diagramme imbriqué [34] peut être utilisé pour afficher le treillis résultant, avec (G, S, J) au premier niveau et $\mathbb{K}$ au second niveau, *c.-à-d.*, le diagramme construit pour $\mathfrak{B}(G, S, \mathrm{J})$ avec des nœuds assez larges pour contenir des copies du diagramme en lignes de $\mathfrak{B}(\mathbb{K})$. Ceci est probablement pas si intéressant ici, puisque chaque nœud de $\mathfrak{B}(G, S, \mathrm{J})$ contiendra une copie de $\mathfrak{B}(\mathbb{K})$, qui est probablement déjà assez grande. Une façon alternative de visualiser les concepts généralisés est de faire une projection de $(G, M \cup S, \mathrm{I} \cup \mathrm{J})$ sur les attributs généralisés dans S et joindre à chaque nœud de $\mathfrak{B}(G, M \cup S, \mathrm{J})$ sa classe d'équivalence. Ici, deux nœuds sont équivalents si et seulement si leurs intentions ont la même restriction sur S [45]. Pour notre exemple de la généralisation $\forall$, nous affichons dans la figure 2.4 la projection de $\mathfrak{B}(G, M \cup S, \mathrm{I} \cup \mathrm{J})$ sur S en marquant les classes d'équivalence sur $\mathfrak{B}(G, M \cup S, \mathrm{I} \cup \mathrm{J})$. Ceci est une amélioration du treillis affiché au milieu de la figure 2.3.

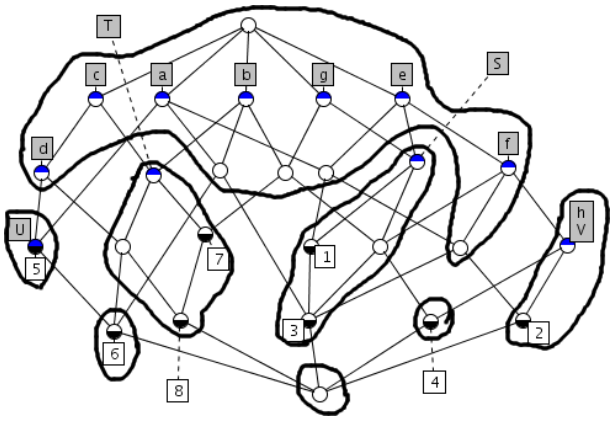


FIGURE 2.4 – Projection du treillis de la figure 2.2 sur les attributs généralisés par $\forall$.

2.3.4 Interprétation de la généralisation sur les attributs

Pour les attributs $a, b \in M \cup S$, nous devons normalement affirmer que a est une généralisation de b (ou b est une spécialisation de a) chaque fois qu'il y a plus d'objets qui vérifient a que b, autrement dit $\mu a \geq \mu b$.

Pour le cas $\exists$, nous avons $m'_s = \bigcup\{m' \mid m \in m_s\}$. Ainsi, $\mu m_s \geq \mu m$ pour tout $m \in m_s$; ce qui signifie que m_s *généralise* réellement tout attribut $m \in m_s$.

Pour the cas $\forall$, nous avons $m'_s = \bigcap\{m' \mid m \in m_s\}$. Ainsi, $\mu m_s \leq \mu m,\ \forall m \in m_s$; c.-à-d. m_s *spécialise* tout attribut $m \in m_s$.

Pour le cas α, $g \,\mathrm{J}\, m_s$ ssi $\alpha \leq \frac{|\{m \in m_s | g \,\mathrm{I}\, m\}|}{|m_s|}$. Pour la généralisation de type α contenue dans la table 2.1, le treillis de concepts de $(G, M \cup S, \mathrm{I} \cup \mathrm{J})$ est illustré par la figure 2.5 et indique que :

- Il y a une généralisation d'attributs $m_s \in S$ avec au moins un attribut $m \in m_s$ de telle sorte que $\mu m_s \not\leq \mu m$ dans $\mathfrak{B}(G, M \cup S, I \cup J)$; c.-à-d. μm_s n'est pas une *spécialisation* de μm. Ex., $m_s := E = \{a, b, c\}$ et $m = b$.

- Il y a une généralisation d'attributs $m_s \in S$ avec au moins un attribut $m \in m_s$ tel que $\mu m \not\leq \mu m_s$ dans $\mathfrak{B}(G, M \cup S, I \cup J)$; c.-à-d. μm_s n'est pas une *généralisation* de μm. Exemple : $m_s := E = \{a, b, c\}$ et $m = b$.

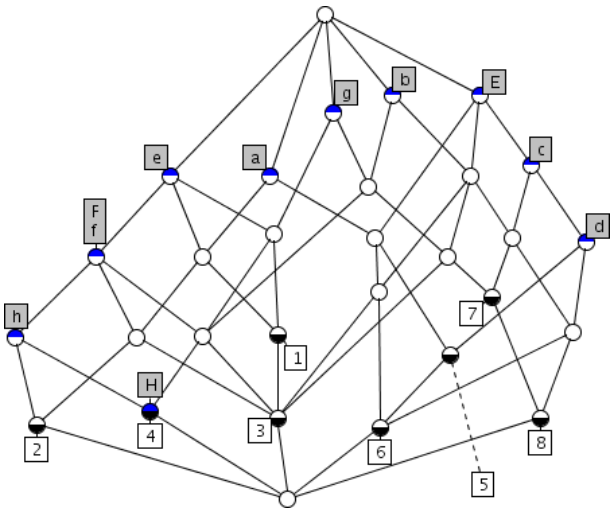


FIGURE 2.5 – généralisation α avec $\mu E \| \mu b$. $E = \{a, b, c\}$, $F = \{d, e, f\}$, $H = \{g, h\}$, $\alpha = 0.6$, où $x \parallel y$ signifie que x et y sont incomparables.

Par conséquent, dans le cas-α, il y a des attributs généralisés m_s qui sont ni une *généralisation* des éléments de m ni une *spécialisation* de ces mêmes éléments. Ainsi, nous devons mieux appeler le cas-α une *approximation* d'attributs, le cas-$\forall$ une *spécialisation* et seulement le cas-$\exists$ une *généralisation*.

2.4 Contrôle de la taille des concepts généralisés

Un concept généralisé est un concept dont l'intention (ou l'extension) contient des attributs généralisés (ou objets). La figure 2.6 affiche une généralisation de type $\exists$ qui conduit à un plus grand nombre de concepts. Dans ce qui suit, nous analysons l'impact des généralisations $\exists$ et $\forall$ d'attributs sur la taille de l'ensemble des concepts généralisés résultants.

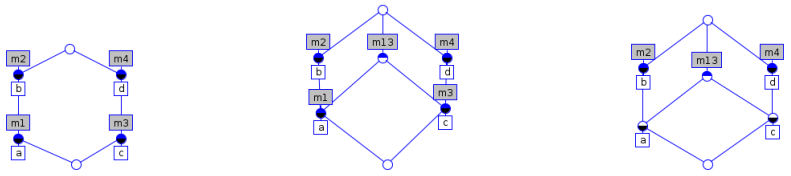


FIGURE 2.6 – B_4 (gauche) et une généralisation$\exists$: m_1 et m_3 sont généralisés en m_{13} et par la suite enlevés du contexte. La taille n'augmente pas.

2.4.1 Une généralisation de type $\exists$ sur les attributs

Soit (G, M, I) un contexte et (G, S, J) un contexte obtenu par une généralisation de type $\exists$ sur les attributs. Nous posons $S = \{m_s \mid s \in S\}$, avec $m_s \subseteq M$. Alors, $g \,\mathrm{J}\, m_s$ ssi $\exists m \in m_s, g \,\mathrm{I}\, m$. Pour comparer la taille du treillis de concepts correspondant, nous définissons quelques fonctions. Nous supposons que $(m_s)_{s\in S}$ forme une partition de M. Alors pour chaque $m \in M$, il y a un seul attribut généralisé m_s tel que $m \in m_s$, et $g \,\mathrm{I}\, m$ implique $g \,\mathrm{J}\, m_s$, pour tous $g \in G$. Pour faire la distinction entre les dérivations dans chacun des deux contextes (G, M, I) et dans (G, S, J), nous remplacerons $'$ par le nom de la relation binaire correspondante.

Comme exemple, $g^{\mathrm{I}} = \{m \in M \mid g \,\mathrm{I}\, m\}$ et $g^{\mathrm{J}} = \{s \in S \mid g \,\mathrm{J}\, s\}$. Les deux fonctions canoniques $\bar{\gamma}$ et $\bar{\mu}$ définies par

$$\begin{array}{rcl} \bar{\gamma}\colon G & \to & \mathfrak{B}(G,S,J) \\ g & \mapsto & (g^{\mathrm{J}\,\mathrm{J}}, g^{\mathrm{J}}) \end{array} \quad \text{et} \quad \begin{array}{rcl} \bar{\mu}\colon M & \to & \mathfrak{B}(G,S,J) \\ m & \mapsto & (s^{\mathrm{J}}, s^{\mathrm{J}\,\mathrm{J}}), \text{ où } m \in m_s \end{array}$$

induisent deux correspondances φ et ψ qui préservent l'ordre et sont définies par :

$$\begin{array}{rcl} \varphi : \mathfrak{B}(G,M,I) & \to & \mathfrak{B}(G,S,J) \\ (A,B) & \mapsto & \bigvee\{\bar{\gamma}g \mid g \in A\} \end{array} \quad \text{et} \quad \begin{array}{rcl} \psi : \mathfrak{B}(G,M,I) & \to & \mathfrak{B}(G,S,J) \\ (A,B) & \mapsto & \bigwedge\{\bar{\mu}m \mid m \in B\}. \end{array}$$

Si φ ou ψ est surjective, alors le treillis de concepts du contexte généralisé est de cardinalité inférieure ou égale à celle du treillis initial. Comme nous l'avons vu sur la figure 2.6, ces fonctions peuvent toutes les deux ne pas être subjectives. Évidemment,

$\varphi(A,B) \leq \psi(A,B)$ puisque $g\,\mathrm{I}\,m$ implique $g\,\mathrm{J}\,m_s$ et $\bar{\gamma}g \leq \bar{\mu}m_s$. La question que l'on pourrait se poser est la suivante : quand est-ce que nous avons-nous l'égalité ? Est-ce que l'égalité implique la surjectivité ?

Nous avons identifié les deux cas spéciaux suivants où le nombre de concepts n'augmente pas après une généralisation de type $\exists$.

Cas 1 tout m_s possède un plus grand élément $\top_s$. Alors, le contexte (G,S,J) est une projection de (G,M,I) sur l'ensemble $M_S := \{\top_s \mid s \in S\}$ des plus grands éléments de m_s. Ainsi, $\mathfrak{B}(G,S,J) \cong \mathfrak{B}(G,M_S,I \cap (G \times M_S))$ et est un sous-ordre de $\mathfrak{B}(G,M,I)$. D'où $|\mathfrak{B}(G,S,J)| = |\mathfrak{B}(G,M_S,I \cap G \times M_S)| \leq |\mathfrak{B}(G,M,I)|$.

Cas 2 $\bigcup\{m^{\mathrm{I}} \mid m \in m_s\}$ est une extension pour tout $m_s \in S$. Alors, aucun groupement d'attributs ne produit de nouveaux concepts et donc le nombre de concepts ne peut pas augmenter.

Le résultat suivant fournit une importante classe de treillis pour laquelle la généralisation de type $\exists$ ne fait pas augmenter la taille du treillis.

Theorme 2.4.1 *La généralisation de type $\exists$ sur des treillis de concepts distributifs dont les contextes sont réduits selon les objets*[3] *ne font pas augmenter la taille du treillis de concepts.*

Preuve Soit $\mathbb{K} := (G,M,\mathrm{I})$ un contexte réduit selon les objets de sorte que $L := \mathfrak{B}(\mathbb{K})$ est un treillis distributif. Soit (G,S,J) un contexte obtenu par une généralisation de type $\exists$ sur les attributs de M. Soit m_s un attribut généralisé. Il suffit de prouver que m_s^{J} est l'extension d'un concept issu du contexte $\mathbb{K} := (G,M,\mathrm{I})$.

Par définition,

$$m_s^{\mathrm{J}} = \bigcup\{m^{\mathrm{I}} \mid m \in m_s\} \subseteq \Big(\bigcup\{m^{\mathrm{I}} \mid m \in m_s\}\Big)^{\mathrm{II}} = \mathrm{ext}\,\Big(\bigvee\{\mu m \mid m \in m_s\}\Big)$$

Pour n'importe quel $g \in \mathrm{ext}(\bigvee\{\mu m \mid m \in m_s\})$ nous avons $\gamma g \leq \bigvee\{\mu m \mid m \in m_s\}$ et

$$\begin{aligned}
\gamma g &= \gamma g \wedge \bigvee\{\mu m \mid m \in m_s\}, && \text{puisque } \gamma g \leq \bigvee\{\mu m \mid m \in m_s\} \\
&= \bigvee\{\gamma g \wedge \mu m \mid m \in m_s\}, && \text{dès que } L \text{ est distributive} \\
&= \gamma g \wedge \mu m \text{ for some } m \in m_s, && \text{dès que } \mathbb{K} \text{ est un contexte réduit.}
\end{aligned}$$

Donc $\gamma g \leq \mu m$, and $g \in m^{\mathrm{I}}$. Ceci prouve que $\mathrm{ext}(\bigvee\{\mu m \mid m \in m_s\}) \subseteq m_s^{\mathrm{J}}$, et $m_s^{\mathrm{J}} = \mathrm{ext}\,(\bigvee\{\mu m \mid m \in m_s\})$. ∎

Dans [23] Marcel Erné a trouvé des conditions nécessaires et suffisantes sur les contextes pour que des treillis de concepts soient distributifs (voir aussi [34]). Comme les cas discutés

[3]Un contexte est dit réduit selon les objets si aucune ligne du contexte ne peut être obtenue par intersection d'autres lignes du contexte. Dans ce cas, les concepts objets γg sont irréductibles-$\bigvee$.

auparavant ne sont pas les seuls où la taille n'augmente pas, il serait intéressant de décrire les classes de treillis sur lesquelles une généralisation de type ∃ ne fait pas augmenter la taille du treillis. La figure 2.7 fournit quelques exemples ne satisfaisant aucune des conditions ci-dessus.

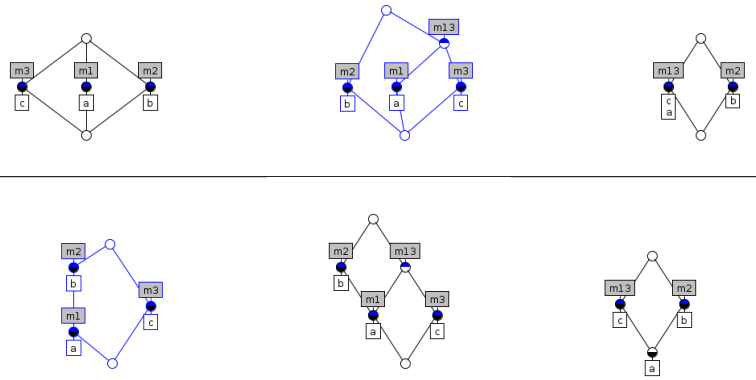


FIGURE 2.7 – généralisation de type ∃ sur M_3 (au dessus de la ligne, à gauche) et sur N_5 (au-dessous de la ligne, à gauche) : m_1 et m_3 sont généralisés en m_{13} et ainsi supprimés du contexte. La taille n'augmente pas.

Les treillis M_3 et N_5 sont les treillis minimaux non distributifs. N'importe quelle généralisation sur ces treillis ne fera pas augmenter la taille.

Sur le treillis B_4 (voir la figure 2.6), il y a une généralisation de type ∃ qui augmente la taille du treillis de concepts. En effet, en mettant les attributs m_1 et m_3 ensemble, ils génèrent exactement un seul nouveau concept : μm_{13} sur le contexte $(\{a, b, c, d\}, \{m_1, m_2, m_3, m_4, m_{13}\}, \mathrm{I})$[4]. Cependant, les concepts attributs m_1 et m_3 deviennent réductibles et leur suppression ne fait pas réduire la taille du treillis de concepts. Cela semble être la principale configuration qui force la généralisation de type ∃ de faire augmenter la taille, comme nous pouvons le voir dans la Proposition 2.4.2.

Proposition 2.4.2

i) Le treillis B_4 est le plus petit treillis sur lequel il y a une généralisation de type ∃ qui fait augmenter la taille du treillis de concepts initial.

ii) Si le contexte contient les attributs m_1, m_2, m_3, m_4 tel que $\mu m_1 < \mu m_2$, $\mu m_3 < \mu m_4$, $\mu m_2 \wedge \mu m_3 \leq \mu m_1$ et $\mu m_1 \wedge \mu m_4 \leq \mu m_3$, alors il y a une généralisation de type ∃ qui ne fait pas augmenter la taille du treillis de concepts.

Preuve Soit (G, M, I) un contexte et $m_1, m_2, m_3, m_4 \in M$ tel que $\mu m_1 < \mu m_2$, $\mu m_3 < \mu m_4$, $\mu m_2 \wedge \mu m_3 \leq \mu m_1$, $\mu m_1 \wedge \mu m_4 < \mu m_3$. La généralisation de m_1 et m_3 en m_{13} produira un nouveau contexte $(G, M \cup \{m_{13}\})$. Les attributs m_1 et m_3 deviennent réductibles puisque

[4]Nous ne ferons pas la distinction ici entre I et sa restriction ou son extension.

$\mu m_1 = \mu m_2 \wedge \mu m_{13}$ et $\mu m_3 = \mu m_4 \wedge \mu m_{13}$. En effet, pour tout $g \in G$ nous avons $g \in m_1' \implies g \in m_2' \cap m_{13}'$. Inversement, $g \in m_2' \cap m_{13}' = m_2' \cap (m_1' \cup m_3') = (m_2' \cap m_1') \cup (m_2' \cap m_3') \subseteq m_1'$ puisque $\mu m_2 \wedge \mu m_3 \leq \mu m_1$. Le même raisonnement peut être effectué sur m_3. Donc $|\mathfrak{B}(G, M \cup \{m_{13}\} \setminus \{m_1, m_2\}, \mathrm{I})| = |\mathfrak{B}(G, M \cup \{m_{13}\}, \mathrm{I})| \geq |\mathfrak{B}(G, M, \mathrm{I})|$.

La proposition 2.4.2 stipule que si un treillis de concepts contient une copie de B_4 étiqueté comme indiqué dans la figure 2.6, alors il y a une généralisation de type $\exists$ qui ne fait pas diminuer la taille du treillis. Cette copie ne doit pas être un sous-treillis comme nous pouvons le voir sur la figure 2.8.

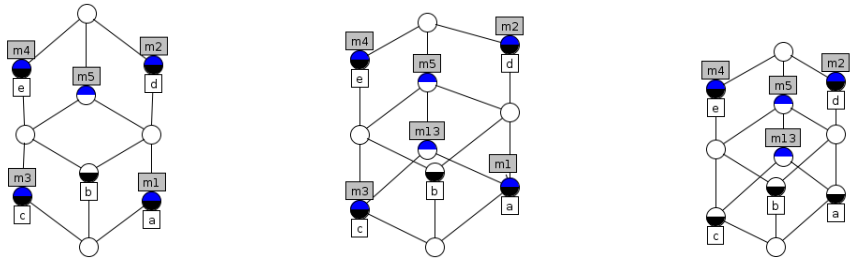


FIGURE 2.8 – Une généralisation de type $\exists$ avec B_4 comme sous-poset mais pas sous-treillis

Pour compléter la caractérisation, voici quelques questions que l'on peut se poser :

- Combien de nouveaux concepts peuvent être générés par une généralisation de type $\exists$ sur juste deux attributs ?
- Est-ce que la réciproque de la proposition 2.4.2 ii) tient toujours ? c.-à-d. existe-il une généralisation de type $\exists$ qui ne fait pas diminuer la taille initiale du treillis de concepts seulement si elle contient une copie de B_4 ? Bien que cela semble être plausible, une preuve nécessite d'être établie.

2.4.2 Une généralisation de type $\forall$ sur les attributs

Soit (G, S, J) un contexte obtenu de (G, M, I) par une généralisation $\forall$. Dans le contexte $(G, M \cup S, \mathrm{I} \cup \mathrm{J})$, chaque concept d'attribut μm_s est réductible. Ce qui signifie que $m_s^{\mathrm{J}} = \bigcap\{m^{\mathrm{J}} \mid m \in m_s\} = \bigcap\{m^{\mathrm{I}} \mid m \in m_s\}$, et est une extension de (G, M, I). Donc, $|\mathfrak{B}(G, S, \mathrm{J})| \leq |\mathfrak{B}(G, M \cup S, \mathrm{I} \cup \mathrm{J})| = |\mathfrak{B}(G, M, \mathrm{I})|$.

Theorme 2.4.3 *La généralisation de type $\forall$ sur les attributs ne fait pas augmenter la taille du treillis de concepts.*

2.5 Les travaux connexes

La présente section donne une vue d'ensemble sur les travaux reliés soit au processus de traitement des descriptions généralisées (au lieu de contextes binaires) en analyse formelle des concepts [27, 31, 44, 53], soit à l'exploitation d'une ontologie (incluant les taxonomies) pour découvrir des motifs généralisés à partir des données. Notre présent travail est davantage relié au second sujet. Quant au premier volet, nous citons en particulier les travaux de [44, 53] sur la généralisation de la construction de treillis de concepts à partir de contextes ayant une structure d'ordre additionnelle sur les objets et/ou les attributs. À ce sujet, Ganter et Kuznetsov [31] proposent une approche où les objets avec leurs descriptions de données partiellement ordonnées (*e.g.*, graphes étiquetés) forment des structures de motifs *(pattern structures)* qui sont exploitées en AFC. Pour simplifier les calculs, une projection est utilisée sur les structures de motifs pour générer les concepts, les implications et les règles de classification. Cette approche a été utilisée avec succès dans [41] pour analyser l'expression des gènes lorsque les motifs sont exprimés comme des tuples d'intervalles. Pernelle *et al.* [64] ont utilisé de façon indépendante la notion de projection avec la même motivation pour obtenir un plus petit treillis comme dans [31]. Ferré [27] définit l'analyse logique de concepts comme une généralisation de l'AFC avec des ensembles d'attributs représentant des expressions logiques. Dans un tel cadre de logique [28], des relations de subsomption peuvent être définies et des opérations sur le treillis de concepts telles que la navigation, l'interrogation et les mises à jour peuvent être conduites d'une manière uniforme et puissante.

Pour le second thème de recherche, il y a eu un ensemble d'études [16, 17, 26, 29, 37, 69, 74] sur la collaboration possible entre l'analyse formelle de concepts et le génie ontologique comme la création, la fusion, l'exploration et l'alignement d'ontologies. En partant de l'observation que le domaine des ontologies et l'AFC ont tous les deux pour objectif de modéliser des concepts, [16] montre comment l'AFC pourrait être exploitée en génie ontologique, et inversement, comment les ontologies pourraient être utilisées de façon bénéfique dans les applications de l'AFC, comme l'extraction de nouvelles connaissances. Dans [69], les auteurs proposent une approche ascendante appelée *FCA-MERGE* pour fusionner des ontologies en utilisant un ensemble de documents en entrée. La méthode repose sur des techniques de traitement du langage naturel et l'AFC pour produire un treillis de concepts. [29] étudie le rôle de l'AFC dans la réutilisation d'ontologies du domaine. Wang propose en [74] une approche de construction d'une ontologie du domaine par l'utilisation de l'AFC. L'ontologie résultante est représentée par un treillis de concepts et exprimée par SWRL (*Semantic Web Rule Language*) pour faciliter le raisonnement et le partage d'ontologies.

Il y a eu plusieurs efforts pour intégrer la connaissance dans les processus de fouille de données. Par exemple, l'étude [68] utilise la taxonomie sur les attributs pour produire des règles généralisées alors que [1] emploie une ontologie du domaine, incluant les relations entre entités pour découvrir des motifs séquentiels généralisés. Dans le domaine de l'AFC, on peut citer le travail de [7] qui utilise des formules de dépendance d'attributs pour une fouille de données basée sur des contraintes *(constraint-based data mining)*, et plus

précisément, pour selectionner uniquement les concepts formels qui vérifient les contraintes exprimées par ces formules.

2.6 Conclusion

Dans ce chapitre, nous avons étudié l'utilisation de taxonomies sur les objets et/ou les attributs dans le cadre de l'analyse formelle des concepts sous trois principaux cas de généralisation ($\exists$, $\forall$, et α) et nous avons considéré différents scénarios de généralisation simultanée sur les objets et les attributs. Nous avons montré que (i) l'ensemble des concepts généralisés est certainement (respectivement souvent) plus petit que l'ensemble des motifs extraits en présence des attributs originaux dans le cas de la généralisation de type $\forall$ (respectivement $\exists$), et (ii) le treillis de concepts généralisé génère non seulement de nouveaux motifs sur les attributs généralisés mais révèle également des particularités d'objets et peut dévoiler une nouvelle forme de taxonomie sur les objets.

Une analyse minutieuse des trois cas de généralisation d'attributs conduit à la conclusion suivante : Le cas α est une approximation d'attribut, le cas $\forall$ est une spécialisation d'attribut alors que le cas $\exists$ est une généralisation d'attribut.

Chapitre 3

Validation

Sommaire

Dans ce chapitre, nous illustrons d'abord l'usage de la généralisation avec un cas pratique relié à la phytothérapie. Ensuite, nous présentons une étude empirique à la section 3.2 sur le lien entre la taille du treillis avant et après une opération de généralisation obtenue par application des opérateurs $\exists$ et α.

3.1 Cas pratique de la phytothérapie

Depuis l'antiquité, les hommes se sont soignés avec les plantes qu'ils avaient à leur disposition. Qu'est-ce qui les a guidés à employer une plante plutôt qu'une autre ? Le hasard ? La religion ? La superstition ? L'expérience, certainement. Plusieurs théoriciens ont entrepris d'expliquer l'action des plantes sur l'organisme. Dans l'Antiquité gréco-romaine, mentionnons les grands médecins grecs : Hippocrate (460-v. 377 av.J.-C.) ; Dioscoride (l° siècle apr.J.-C.), Galien (v. 131-v. 201). L'ouvrage de Dioscoride sur la matière médicale (*De materia medica*) qui décrivait tous les médicaments en usage à son époque demeura l'une des sources les plus consultées par les médecins jusqu'à l'aube du XIXe siècle. Au XVIe siècle, la célèbre école italienne de Salerne a marqué la médecine de son temps. Elle conseillait au roi de conserver un esprit gai, de prendre du repos et de se contenter d'une alimentation modeste. Aujourd'hui, ces conseils pourraient être suivis judicieusement par chacun d'entre nous. Jusqu'au XIXe siècle, les médecins se limitaient pratiquement à l'emploi des plantes comme moyen de soulager les maux de leurs patients. C'est alors que les chimistes ont réussi à isoler les principes actifs de certaines plantes importantes (la quinine du quinquina, la digitaline de la digitale, etc.). Poursuivant leurs recherches au début du XXe siècle, ils ont fabriqué des molécules synthétiques. Désormais, on allait prescrire exclusivement des médicaments issus des plantes en se servant uniquement des réserves

moléculaires chimiques utiles. Récemment, des médecins et des professeurs ont créé des centres de formation en phytothérapie pour expérimenter de nouvelles plantes (comme *Harpagophytum procumbens*), modernisant ainsi la présentation des médicaments et rendant ceux-ci plus efficaces. C'est le cas des extraits secs de plantes prescrits sous forme de gélules. En outre, on procède à des expériences en milieu hospitalier. Au CHRU (Centre hospitalier de recherche universitaire) de Clermont-Ferrand, le professeur Pierre Bastide a, entre autres, testé les vertus curatives des huiles essentielles de cannelle et de girofle sur les infections de l'appareil urinaire.

L'aromathérapie, l'art de soigner par les huiles essentielles, est devenue une science méthodique depuis qu'elle repose sur une classification de ces huiles selon leur capacité à lutter contre les bactéries. Il y a une vingtaine d'années, les docteurs Maurice Girault et Paul Belaiche ont mis au point l'aromatogramme, méthode comparable à l'antibiogramme, qui permet de déceler les huiles essentielles qui sont les plus efficaces sur un germe donné.

Dans ce qui suit, nous présentons une série de tests que nous avons menés sur un exemple issu du monde de la phytothérapie. Nous voulons illustrer étape par étape l'utilisation de l'opérateur $\exists$ présenté dans le chapitre précédent afin de mettre en évidence la capacité de celui-ci à pouvoir réduire, dans la majorité des cas, la taille d'un contexte et en même temps permettre de mieux cerner des motifs importants que nous n'aurions pas pu (ou difficilement) extraire sans cet opérateur.

Nous avons consulté le livre intitulé *le guide familial de la phytothérapie* édité en France en 2010 [2], pour construire notre exemple, lequel va être représenté par une matrice de 142 objets et 108 attributs. Les lignes de cette matrice représentent la liste des maladies et les colonnes, quant à elles, les plantes utilisées pour les traiter. L'intersection d'une ligne et d'une colonne de cette matrice contient la valeur 1 ou la valeur 0, selon que la maladie est soignable ou non par cette plante.

Nous avons utilisé le logiciel Conexp [77] pour conduire l'ensemble de ces tests.

3.1.1 Première généralisation

Dans ce premier test, nous avons déterminé le nombre de concepts formels issus de cette matrice avant généralisation et nous en avons obtenu 304. La figure 3.1 illustre le treillis de concepts obtenu à partir du contexte initial sur les données de la phytothérapie.

On remarque sur la figure 3.1 qu'il serait difficile voire impossible d'extraire facilement des motifs à cause des relations multiples qui existent entre les différents concepts et ce malgré leur nombre relativement petit. C'est une des raisons pour lesquelles plusieurs méthodes de visualisation basées sur des techniques de décomposition de contextes ont étaient créées [35]. Cependant, notre approche pour tenter de réduire la taille de ce contexte va être basée sur les principes de généralisation présentés dans le chapitre précédent. Pour cela, nous avons regroupé les 142 maladies de la liste de départ pour constituer 41 groupes. Notons que ces groupes ont été formés manuellement en se basant sur l'information que nous avons pu obtenir à partir du dictionnaire français intitulé *Larousse des plantes médicinales* [15]. Voici ci-après la liste de ces groupes :

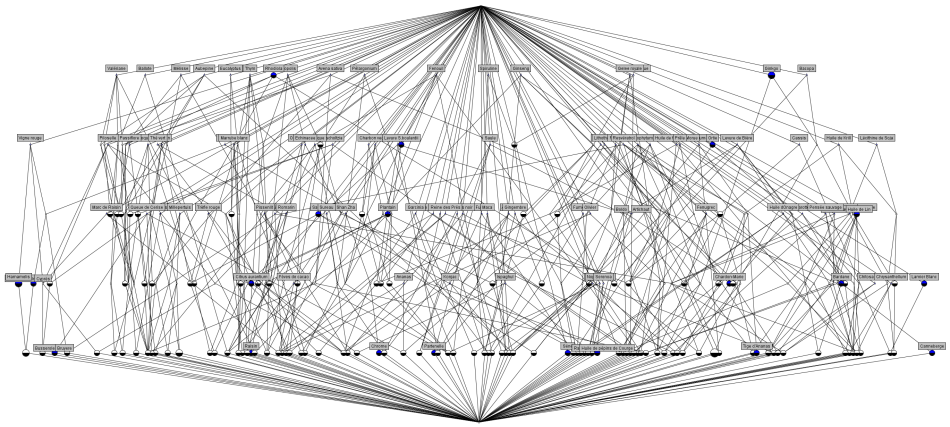


FIGURE 3.1 – treillis du contexte obtenu sur l'exemple de phytothérapie

ACIDITÉ GASTRIQUE ET INDIGESTION	ACNÉ ET FURONCLES	ACOUPHÈNES
ADÉNOME PROSTATIQUE	AFFECTIONS ARTICULAIRES	ANÉMIE
ANXIÉTÉ, ÉTAT DÉPRESSIF ET TENSION NERVEUSE	ASTHME	CONSTIPATION ET DIARRHÉE
DOULEURS ABDOMINALES	DOULEURS ARTICULAIRES ET ANKYLOSE	DOULEURS DORSALES
DOULEURS MUSCULAIRES ET CRAMPES	ENGELURES	ERUPTIONS CUTANÉES
FATIGUE	FLATULENCES ET GAZ	HERPÈS, VARICELLE ET ZONA
HYPERTENSION ARTÉRIELLE	INFECTIONS URINAIRES	INSOMNIE
MALADIE DU MÉTABOLISME	MALADIE HÉPATHIQUE	MANQUE D'APÉTIT
MAUX DE GORGE ET ANGINES	NAUSÉES ET VOMISSEMENTS	NÉVRALGIES
OEDÈMES INFLAMMATOIRES	PROBLÈMES DE FERTILITÉ CHEZ L'HOMME	PROBLÈMES Du CUIR CHEVELURE
PROBLÈMES DU POIDS	PROBLÈMES POSÉS PAR LA MÉNOPAUSE	PROBLÈMES VASCULAIRES
REVITALISANT	RHUMES, GRIPPE ET ÉTATS FÉBRILES	SECRAGE TABAGIQUE
SÉCRÉTIONS ABONDANTES DES MUQUEUSES	SYMPTÔMES DU VIEILLISSEMENT	TOUX ET BRONCHITES
TROUBLES MENSTRUELS	VARICES ET HÉMORROÏDES	

TABLE 3.1 – Liste de 41 groupes de maladies obtenue à partir des 142 maladies du contexte initial

Ainsi, le contexte généralisé obtenu par ce regroupement est une matrice de la forme 41 x 108 au lieu de 142 x 108. Un groupe de maladies i a la valeur 1 en association avec la plante j si cette dernière soigne au moins une maladie du groupe i. Sinon, c'est la valeur 0 qui est attribuée à la cellule. On est donc en présence d'une généralisation de type $\exists$.

D'un côté, la figure 3.2 permet de constater visuellement que le treillis produit est plus réduit. En effet, le nombre de concepts dans le treillis généralisé est de seulement 154 éléments au lieu des 304 de départ. Nous arrivons ainsi à réduire la taille du treillis initial de moitié. D'un autre côté, cette généralisation nous a permis aussi de mettre en évidence certains motifs que nous n'aurions jamais pu extraire sans avoir constitué des groupes de maladies. La figure 3.3 permet de mettre en évidence de tels motifs. Toutes les plantes qui traitent l'hypertension artérielle servent également à soigner les symptômes de vieillissement. Il s'agit de l'ail, l'olivier et l'aubépine. De même, les plantes qui soignent les problèmes vasculaires traitent également les œudèmes inflammatoires, les problèmes de poids ainsi que les varices et les hémorroïdes.

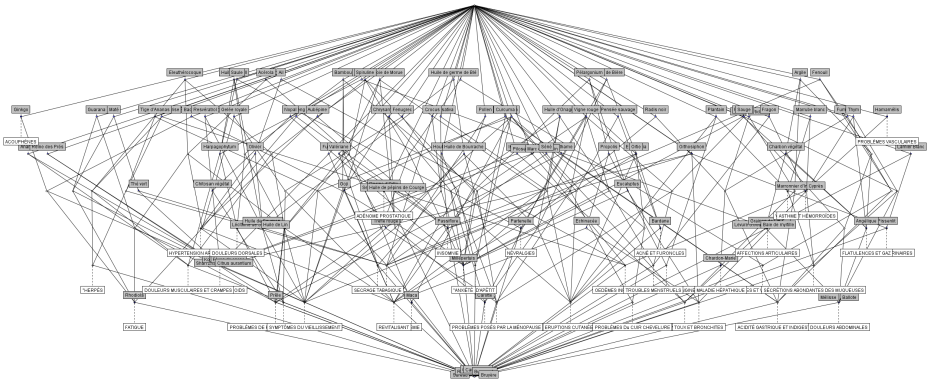


FIGURE 3.2 – Treillis de concepts après un premier regroupement sur les maladies

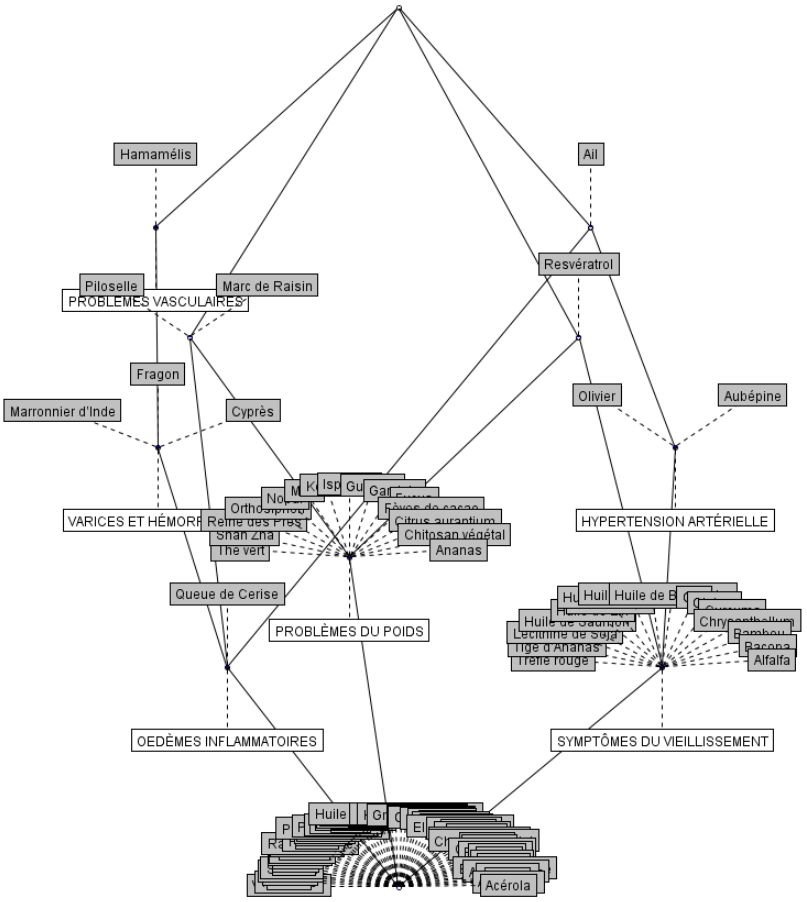


FIGURE 3.3 – Partie du treillis de concepts après un premier regroupement sur les maladies

Nous avons ensuite conduit un deuxième test en procédant à un second regroupement effectué au niveau de la liste des groupes de maladies précédemment créée. L'idée est de tenter de réduire davantage la taille du treillis et découvrir d'autres motifs que l'on n'a pas pu extraire avec la première généralisation.

3.1.2 Deuxième généralisation

Dans ce deuxième test, nous avons procédé à la formation de 18 nouveaux groupes de maladies (table 3.2) tel que proposé dans [15].

ACOUPHÈNES	ADÉNOME PROSTATIQUE	ALLERGIES
INFECTIONS URINAIRES ET MYCOSES	MALADIE HÉPATHIQUE	MANQUE D'APÉTIT
PROBLÈME DU SYSTÈME CARDIO VASCULAIRES	PROBLÈME DU SYSTÈME IMMUNITAIRE	PROBLÈMES DE L'APPAREIL RESPIRATOIRE
PROBLÈMES DERMATOLOGIQUES	PROBLÈMES MÉTABOLIQUES	PROBLÈMES OSSEUX ET MUSCULAIRES
RÈGLES ET FERTILITÉ	SECRAGE TABAGIQUE	TROUBLES DE LA CIRCULATION SANGUINE
TROUBLES DIGESTIFS	TROUBLES NERVEUX ET TROUBLES LIÉS AU STRESS	VIEILLISSEMENT ET TROISIÈME ÂGE

TABLE 3.2 – Liste de 18 groupes de maladies obtenue à partir des 41 groupes initiaux

Le treillis de concepts obtenu à partir de ce deuxième regroupement est illustré par la figure 3.4. Visiblement, cette deuxième généralisation sur les maladies a permis de réduire encore la taille du treillis. Effectivement, le nombre de concepts formels obtenus à partir de ce second contexte généralisé est de seulement 83 au lieu de 154 éléments. Ainsi, une réduction totale de l'ordre de 73% de la taille initiale du treillis sur la phytothérapie a été obtenue suite à cette deuxième généralisation. Précisons que lors de ce deuxième test, c'est à nouveau une généralisation utilisant l'opérateur $\exists$ que nous avons employée et qui a permis de réduire la taille du treillis alors que ce type de généralisation ne le garantit pas systématiquement comme le fait l'opérateur $\forall$.

Nous pouvons plus facilement identifier sur la figure 3.4 les concepts obtenus à partir du deuxième contexte généralisé. Aussi, on peut avec une facilité accrue distinguer certains motifs. Une telle distinction ne pouvait pas s'effectuer sur les treillis précédents. Toutefois, nous avons projeté le contexte obtenu à partir de la deuxième généralisation sur neuf groupes de maladies parmi l'ensemble des 18 existants. Un résultat remarquable a été obtenu. Il met en évidence le fait que les plantes qui traitent certaines maladies du système cardio-vasculaire tel qu'illustré par la figure 3.5, traitent aussi certaines maladies de la peau. Docteur Anne Dompmartin du CHU de CAEN a même mentionnée d'ailleurs dans une étude scientifique que le psoriasis (maladie de la peau) pourrait en réalité cacher des problèmes de santé plus graves comme ceux du cœur ou une mauvaise circulation sanguine (maladie cardio-vasculaire). Ainsi, il pourrait possiblement y avoir un lien étroit entre les maladies du système cardio-vasculaire et celles de la peau.

Ayant illustré l'intérêt de la généralisation d'objets dans la découverte de nouveaux concepts, nous allons consacrer la section suivante à des tests expérimentaux sur la réduction de la taille d'un treillis suite à l'application d'une généralisation de type α ou $\exists$ sur des attributs.

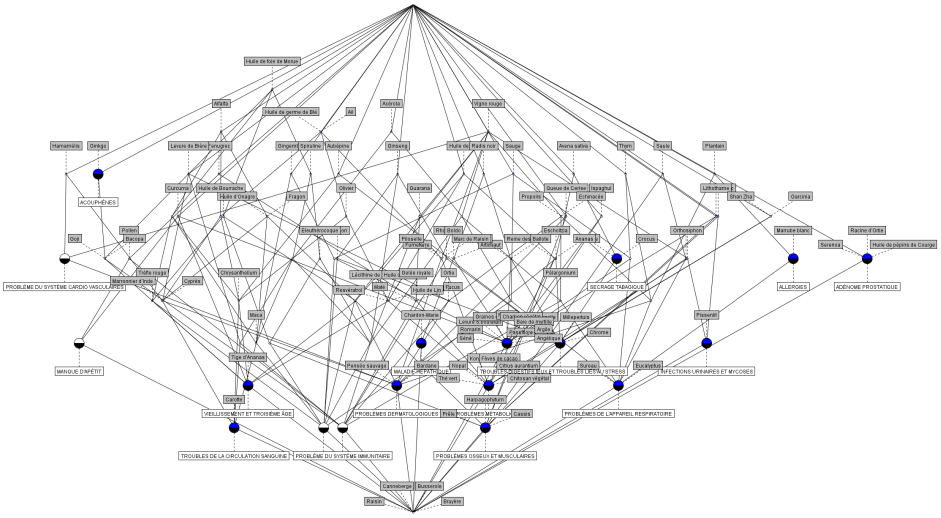


FIGURE 3.4 – Treillis de concepts après un deuxième regroupement sur les maladies

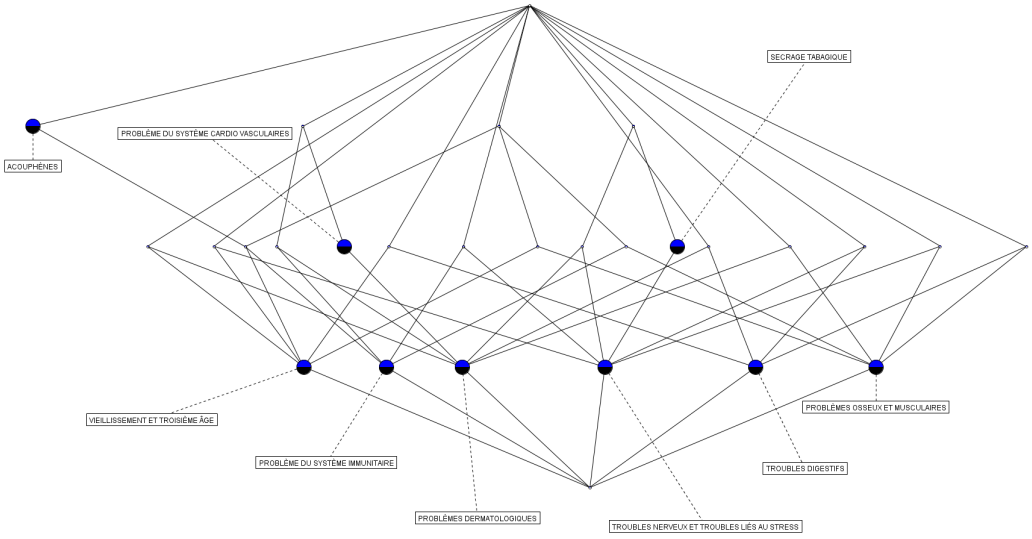


FIGURE 3.5 – Projection du treillis sur quelques groupes de maladies

3.2 Tests expérimentaux

Le but de ces expériences est double : (i) montrer que la généralisation peut amener de nouveaux *motifs sémantiquement riches*, et (ii) mettre en évidence le fait que dans les cas α et $\exists$, la *taille du treillis* après une généralisation sur les attributs est *généralement réduite* à l'exception de quelques cas spécifiques tel que dans la généralisation-$\exists$ sur les groupes de deux attributs qui ne sont pas communs à plus d'un objet.

Nous avons conduit des expériences sur l'ensemble de données bien connu appelé *mushroom* du *UCI Repository* [51]. Les données de cet exemple représentent 23 espèces de 8126 champignons à lamelles en terme de 22 attributs nominaux. Chaque espèce est identifiée comme comestible ou toxique. Le contexte binaire obtenu par conversion contient 119 attributs incluant deux attributs pour les deux classes de champignons : comestible (e) et toxique (p). Le contexte binaire a une densité de 19% et génère un treillis complexe de 238711 concepts.

3.2.1 Nouveaux motifs généralisés

Pour mettre en évidence le potentiel de la généralisation, découvrir des motifs nouveaux et pertinents, nous avons effectué différentes généralisations sur les attributs dans l'ensemble de données *mushroom*. La figure 3.6 montre un cas où deux attributs généralisés sont créés. Le premier cas existentiel généralise l'habitat $= \{d, g, m\}$ et la surface du chapeau $= \{g, s, y\}$ pour représenter la propriété du champignon d'avoir un habitat soit de bois (d), herbes (g), ou prairies (m) parmi sept valeurs possibles, ou d'avoir une surface de chapeau parmi les trois sortes : rainures (g), écailleuse (y), ou lisse (s). La seconde généralisation concerne la propriété d'avoir une odeur d'amande (a) ou d'anis (l) parmi les neufs cas d'odeurs possibles. Après généralisation et projection sur les attributs généralisés ainsi que sur les deux valeurs de l'attribut de classification et la valeur de la taille de la lamelle (*narrow gill-size*), nous obtenons la figure 3.6 laquelle montre clairement que le premier attribut généralisé est détenu par 96% de champignons, mais ne permet pas de faire la distinction entre les deux classes : champignons comestibles ou toxiques. Cependant, nous pouvons conclure que les champignons avec l'odeur d'amande ou d'anis sont nécessairement comestibles. Un tel motif serait difficilement détectable dans le grand treillis.

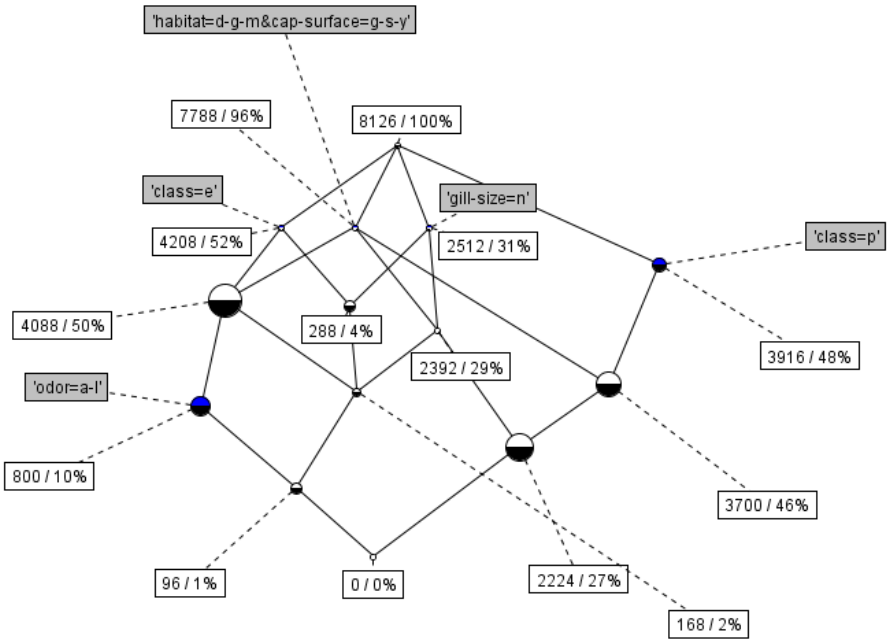


FIGURE 3.6 – Deux attributs généralisés dans l'ensemble des données sur les champignons.

TABLE 3.3 – Généralisation-α sur les attributs des données de *mushroom*.

Nb. d'attributs Gén.	Nb. de concepts Gén.	% de réduction	Densité
6	145	99.9	0.60
9	477	99.8	0.46
20	23138	90.3	0.68
58	185138	22.4	0.34

3.2.2 Variation de la taille du treillis

Afin d'analyser de manière empirique la variation de la taille du treillis avant une généralisation $\exists$ ou α, nous avons conduit deux types d'expériences sur l'ensemble des données de *mushroom*. Le premier type de tests consiste en une génération aléatoire de 6, 9, 20 et finalement 58 attributs généralisés parmi les 117 attributs élémentaires initiaux. Nous avons ensuite appliqué la généralisation-α avec comme valeur pour $\alpha = 19\%$ au sein de chaque groupe d'attributs sur la totalité de l'ensemble des 8126 champignons. Par exemple, dans le cas de 9 attributs généralisés, chaque groupe représente 13 attributs élémentaires. Chaque génération aléatoire a été répétée 10 fois et la moyenne des valeurs a été comptabilisée pour le nombre de concepts généralisés et la densité du contexte généralisé comme indiqué dans la table 3.3. Comme on peut s'y attendre, il y a une plus petite réduction de la taille du treillis si le nombre d'attributs élémentaires par attribut généralisé devient plus petit et que le nombre d'attributs généralisés augmente. En effet, ceci est observé quand nous prenons 58 attributs généralisés par groupes de deux attributs élémentaires identifiés au hasard dix fois.

Nos essais antérieurs sur un large contexte généré artificiellement montrent que le processus de généralisation ne réduit pas seulement la taille du contexte, mais peut également réduire considérablement la taille du treillis correspondant. En outre, le nombre de concepts généralisés est presque inversement proportionnel au nombre d'attributs simples par attribut généralisé. Cependant, nous avons constaté que lorsque ce nombre est égal à 2, le nombre de concepts généralisés peut être plus grand que le nombre original de concepts. Ceci est observé dans le deuxième type de tests (voir table 3.4).

Le deuxième type de tests empiriques liés à la variation de la taille du treillis vise à observer l'impact d'une généralisation de type$\exists$ appliquée à seulement une seule paire d'attributs. Vingt attributs parmi les 117 attributs initiaux sont d'abord choisis au hasard cinq fois. Pour chacun des groupes de 20 attributs, deux parmi eux sont identifiés à 10 reprises de façon aléatoire pour constituer un attribut généralisé. Nous obtenons donc 50 contextes de 19 attributs et 8126 objets. La table 3.4 montre que parmi les cinquante contextes avec un seul attribut généralisé obtenu à partir de deux attributs élémentaires, il existe des situations où la taille du treillis diminue et d'autres où la taille augmente. Par exemple, parmi les cinquante contextes générés, nous observons une diminution de 13 à 41% dans la taille du treillis, tandis qu'il y a huit cas où la taille du treillis a à peine

TABLE 3.4 – Variation de la taille du treillis après une généralisation-$\exists$ sur une paire d'attributs dans l'ensemble de *mushroom*.

Dimunution de la taille	Nb de tests
$[-0.007, -0.004[$	8
$[-0.004, 0[$	7
$[0, 6[$	17
$[6, 13[$	7
$[13, 41[$	11

augmenté de 0.04% à 0.07%.

3.3 Conclusion

Nous avons présenté dans ce chapitre une illustration de l'opérateur de généralisation $\exists$ sur un exemple concret, celui de la phytothérapie, une science qui étudie l'emploi des plantes pour soigner certaines maladies. Nous avons ensuite montré à travers deux tests qu'il était possible de réduire de façon significative ($\pm 60\%$) la taille du treillis et aussi faciliter considérablement la détection de motifs pertinents qui n'auraient pas pu être mis en évidence avant la généralisation.

Chapitre 4

Découverte de co-clusters

Sommaire

4.1 Introduction

Pour mieux gérer l'énorme volume de données à traiter, des méthodes de décomposition de matrices de données et de généralisation au niveau surtout des propriétés et parfois des objets (cf. chapitre 2) ont été proposées pour générer des blocs plus compacts et homogènes composés soit de sous-ensembles d'objets et de propriétés, soit d'objets et d'attributs à un plus haut niveau d'abstraction (ex. des attributs généralisés) de sorte à faciliter l'analyse des données et la visualisation des motifs qui en sont extraits [32, 33, 22]. Pour produire des motifs généralisés, on utilise fréquemment une taxonomie sur les objets et/ou les propriétés [47]. Que fait-on alors si une telle taxonomie est absente ou n'est pas évidente ? Une approche consiste à utiliser une des méthodes de regroupement (*clustering*) qui, lorsqu'elle est appliquée aux objets et/ou aux propriétés, permet d'identifier ceux à regrouper pour constituer un élément généralisé.

Cependant, les méthodes de regroupement emploient l'ensemble des propriétés ou des objets pour créer les groupes [66]. Elles peuvent se révéler très efficaces dans certains cas et bien moins dans d'autres. Cela se produit quand des groupes d'objets plus homogènes et/ou plus intéressants peuvent être formés sur la base d'un sous-ensemble de propriétés

plutôt que sur l'ensemble de ces dernières, ou lorsqu'une propriété et/ou un objet peut faire partie de plusieurs groupes. On devrait alors être en mesure de constituer des groupes d'objets et/ou de propriétés chevauchants.

Les méthodes classiques de regroupement ne permettent pas de créer de tels groupes à la différence des techniques de clustering flou (*Fuzzy clustering*) [76] ou de co-clustering (*biclustering*) [38, 54]. Toutefois, ces dernières nécessitent souvent l'emploi de certains paramètres tels le degré d'appartenance à un groupe de même que la détermination préalable du nombre et des centres des différents groupes à créer. Un mauvais choix de l'un de ces paramètres peut conduire à la création de groupes dont les éléments ont un faible degré d'homogénéité et où les frontières entre l'ensemble ou quelques-uns des groupes générés sont mal définies. Dans ce chapitre ainsi que dans les trois chapitres suivants, nous décrivons une nouvelle procédure de co-clustering qui se décline en plusieurs variantes, y compris celle qui produit des concepts formels.

4.2 État de l'art

En 1972, Hartigan [38] avait proposé une nouvelle méthode appelée le co-clustering (ou classification croisée) qui permet de créer à partir d'une matrice binaire des sous-matrices composées d'objets et de propriétés et appelées co-clusters. Une telle méthode permet d'effectuer un regroupement simultané sur les objets et les propriétés d'une matrice (ou contexte formel). Son principal avantage est l'interprétation directe des co-clusters. Ces derniers correspondent à un ensemble d'objets fortement corrélés avec un ensemble de propriétés. Notons que le niveau élevé de cohésion au sein des différents co-clusters s'explique par le fait que seules les propriétés jugées très pertinentes pour la création d'un groupe d'objets sont utilisées plutôt que l'ensemble des propriétés disponibles. De plus, une méthode de co-clustering permet souvent d'avoir un objet ou une propriété qui se retrouve plus d'une fois dans des groupes d'objets ou des groupes de propriétés respectivement. Cela est également le cas pour quelques méthodes de regroupement qui produisent des groupes chevauchants.

Plusieurs algorithmes de co-clustering ont suivi [11, 55, 19, 48, 14]. Ils ont été proposés dans différents domaines dont principalement celui de la génétique mais aussi et de façon plus restreinte en marketing, traitement d'image, fouille de texte et exploration du Web. Dans l'état de l'art produit par [55], les auteurs comparent non seulement les caractéristiques du clustering avec le co-clustering mais présentent également un grand nombre d'approches de co-clustering ainsi que le type de co-clusters qu'elles produisent. Contrairement aux procédures de *clustering*, les algorithmes de co-clustering déterminent les groupes d'objets qui exhibent des spécificités pour un sous-ensemble spécifique de propriétés. Ainsi donc, les techniques de co-clustering sont considérées comme un bon choix d'analyse de données lorsque ces dernières comportent des sous-ensembles d'objets (ex. gènes) dont les membres peuvent être codécrits par un sous-ensemble d'attributs mais sont indépendants d'autres attributs [55].

Les algorithmes de co-clustering les plus connus produisent des co-clusters à partir d'une

matrice binaire où toutes les valeurs sont égales à 1 [50]. Chacun des co-clusters produits est constitué d'un ensemble d'objets partageant un ensemble de propriétés. Lorsque la matrice est binaire, ces algorithmes ignorent ou n'exploitent pas l'information pouvant être extraite à partir des 0 présents dans cette matrice. La présence d'un zéro dans une matrice binaire donne une information à ne pas négliger sur l'absence d'une propriété pour un objet. Ainsi, un co-cluster contenant un agencement adéquat de 0 et de 1 peut générer des motifs beaucoup plus riches et plus pertinents qu'un co-cluster composé uniquement de 1. Dans certains cas, nous pouvons à partir d'un co-cluster, connaître les objets possédant un ensemble de propriétés et, sans recours à une tierce information, connaître aussi ceux qu'ils ne possèdent pas. Ce type de co-cluster serait utile en marketing et en génétique pour ne citer que ces deux domaines. On pourrait ainsi découvrir en génétique les gènes qui se comportent de façon radicalement opposée lorsqu'ils sont soumis à certaines conditions, et en marketing les personnes qui adoptent des comportements différents vis-à-vis de certains produits du marché. Dans le domaine du marketing, la recherche commanditée (*sponsored search*) fait usage de plusieurs techniques de fouille de données, y compris le co-clustering. Cela permet d'identifier des groupes dans lesquels une collection de mots clés (objets) est choisie un certain nombre de fois par des utilisateurs pour accéder à des sites publicitaires (attributs).

On dénombre un effectif assez limité de méthodes de co-clustering permettant de générer des co-clusters composés de 0 et de 1. C'est le cas de [4] et de l'approche de projection duale proposée par [25]. Cette dernière a été développée pour produire des co-clusters disjoints à valeurs binaires et dont la densité des 1 est variable. Toutefois, le nombre de co-clusters produits par cette méthode doit être fixé au préalable. En fait, rares sont les algorithmes de co-clustering qui n'emploient pas de paramètres d'entrée pour créer les blocs. Généralement, c'est le nombre de co-clusters à produire et/ou la densité de ces derniers qui sont requis par ces algorithmes. Par conséquent, la qualité des résultats obtenus est tributaire du choix de la valeur des paramètres d'entrée.

Une autre catégorie d'algorithmes de co-clustering existe pour les matrices multivaluées [8, 61]. Généralement, ces algorithmes procèdent à une transformation des valeurs de la matrice multivaluée en valeurs binaires avant de pouvoir procéder à la production de co-clusters. Toutefois, cette transformation augmente la dimensionnalité des données et risque de conduire à des motifs peu ou pas pertinents si le découpage effectué au niveau de l'ensemble des propriétés est mal ou peu adapté au type d'analyse requis.

4.2.1 Principales classes de co-clusters

La table 4.1 illustre les cinq grandes classes de co-clusters [55] lesquelles correspondent aux différents types de motifs qui peuvent être extraits et mis en évidence à partir d'une matrice de données. De gauche à droite, on trouve la classe des co-clusters à une seule valeur constante, à valeurs constantes au niveau des lignes, à valeurs constantes au niveau des colonnes, à valeurs cohérentes (cas additif), à valeurs cohérentes (cas multiplicatif) et finalement les co-clusters à évolutions cohérentes.

Les co-clusters à une seule valeur constante sont des co-clusters dont toutes les valeurs

2	2	2	2	2	2	2	2	2	3	4	5	1	2	5	0	1.0	2.0	0.5	1.5
2	2	2	2	3	3	3	3	2	3	4	5	2	3	6	1	2.0	4.0	1.0	3.0
2	2	2	2	4	4	4	4	2	3	4	5	4	5	8	3	4.0	8.0	2.0	6.0
2	2	2	2	5	5	5	5	2	3	4	5	5	6	9	4	3.0	6.0	1.5	4.5

TABLE 4.1 – Illustration des cinq principales classes de co-clusters

sont égales à une même constante au sein du bloc. Les co-clusters à valeurs constantes au niveau des lignes sont ceux où chaque ligne du co-cluster correspond à une valeur constante. Il en est de même pour les co-clusters à valeurs constantes au niveau des colonnes, mais dans ce cas ce sont les colonnes du co-clusters qui sont constantes. Dans le cas des co-clusters à valeurs cohérentes de type additif, le passage de la valeur d'une colonne à l'autre se fait par addition d'une même valeur positive ou négative. Le même principe est utilisé pour les co-clusters à valeurs cohérentes pour le cas multiplicatif. Toutefois, l'addition est remplacée par la multiplication. Finalement, les co-clusters à évolutions cohérentes sont ceux où le passage entre les différentes valeurs du co-cluster s'effectue suivant une règle de calcul bien définie.

Comme nous allons montrer par la suite notre approche de co-clustering permet de produire quelques nouveaux types de co-clusters.

4.2.2 Principales approches de co-clustering

Plusieurs algorithmes basés sur les techniques de co-clustering ont été proposés et utilisés dans différents domaines tels que le marketing [39, 73], la fouille de texte [20, 49], les réseaux sociaux [62] et la génétique [3]. Ils permettent de partitionner un contexte représenté par une matrice à deux dimensions en un ensemble de sous-matrices appelées co-clusters qui peuvent être chevauchants ou non. Le but est de constituer des blocs composés de groupes d'objets et de propriétés de sorte que chacun de ces groupes révèle l'existence d'un lien très étroit entre ses membres et leurs attributs.

Il existe deux principales approches de co-clustering. L'approche directe et celle indirecte [6]. Dans l'approche directe, aucune transformation préalable du contexte n'est pratiquée lors de l'utilisation de l'algorithme de co-clustering. Cependant, dans la seconde approche, le contexte initial bidimensionnel (*two-modes*) composé d'un ensemble d'objets et leurs attributs est transformé en deux contextes unidimensionnels (*One-mode*) représentant la projection sur l'ensemble des objets et celle sur l'ensemble d'attributs. La littérature sur le co-clustering compte plus de méthodes directes qu'indirectes. La principale raison est que l'on croit que les méthodes indirectes sont entachées d'erreurs causées par une perte d'information liée à la transformation préalable du contexte via l'opération de projection. Cependant, Borgatti et Everett [24] ont montré que rares sont les cas où une telle perte peut se produire. Ils ont montré qu'il était possible de transformer par projection un contexte bidimensionnel en deux contextes unidimensionnels sur lesquels une des méthodes de regroupement classique pourrait être appliquée afin de pouvoir extraire des co-clusters.

Toutefois, il est souvent difficile lorsqu'on applique ce type d'approche de déterminer les bons clusters à créer et de choisir le bon nombre de ces derniers. Borgatti et Everett proposent de fixer le nombre à 3×3 sans aucune justification de ce choix. De plus, rien ne nous permet de confirmer que les blocs générés sont optimaux du point de vue nombre et qualité.

D'autres méthodes de co-clustering dites directes ont été développées [70]. Ces méthodes emploient des heuristiques pour éviter de faire une recherche exhaustive de tous les co-clusters du contexte et réduire ainsi la complexité des algorithmes bâtis autour de ces méthodes. Elles peuvent se baser sur la maximisation d'un facteur de gain afin de réduire le temps de production des co-clusters. Différentes approches heuristiques ont été employées et sont au nombre de cinq :

1. Le regroupement itératif des lignes et des colonnes *(Iterative row and column clustering)* : Dans un tel cas, une méthode de regroupement standard *(clustering)* est appliquée itérativement aux lignes et aux colonnes du contexte pour ensuite combiner des groupes *(clusters)* obtenus afin de former des co-clusters.

2. La méthode générale "diviser pour régner" *(divide and conquer)* où on divise récursivement un problème en sous-problèmes.

3. La méthode de recherche itérative gloutonne *(greedy iterative search)* : Les algorithmes bâtis autour de cette méthode tentent de choisir une solution optimale locale à chacune de leurs étapes pour avoir une solution optimale au problème global.

4. L'énumération exhaustive des co-clusters *(exhaustive bicluster enumeration)* : Cette méthode énumère de façon exhaustive l'ensemble de tous les co-clusters.

5. L'identification des paramètres de la distribution *(distribution parameter identification)* : Cette méthode suppose que la structure des données du contexte suit un modèle statistique dont il faut déterminer les paramètres

La plupart des algorithmes de co-clustering construits autour de l'une des méthodes discutées ci-dessus nécessitent la connaissance préalable d'un certain nombre de paramètres. À titre d'exemple, on peut citer le nombre de co-clusters à créer par l'algorithme ou la densité désirée pour ces derniers.

4.2.3 Classification des algorithmes de co-clustering

Les algorithmes de co-clustering se distinguent principalement par la structure des co-clusters qu'ils produisent (cf. figure 4.1), l'approche heuristique qu'ils emploient et la classe à laquelle ils appartiennent parmi les cinq classes mentionnées ci-dessus [55].

La figure. 4.1 illustre les neuf différentes structures possibles qu'un algorithme de co-clustering permet de générer. La première structure a) représente un co-cluster unique, la seconde b) est une structure où les co-clusters ont des lignes et des colonnes exclusives.

Dans le cas c), les co-clusters sont non chevauchants et non exclusifs, le cas d) est celui où les co-clusters sont exclusifs au niveau des lignes. Les co-clusters dans le cas e) sont exclusifs au niveau des colonnes alors que le cas f) représente des co-clusters non chevauchants ayant une structure d'arbre *(non overlapping co-clusters with tree structure)*. Le cas g) illustre une autre situation de co-clusters non chevauchants et non exclusifs. Quant au cas h), c'est celui où les co-clusters sont chevauchants et présentent une structure hiérarchique. Enfin le dernier cas i) est celui où les co-clusters sont arbitraires.

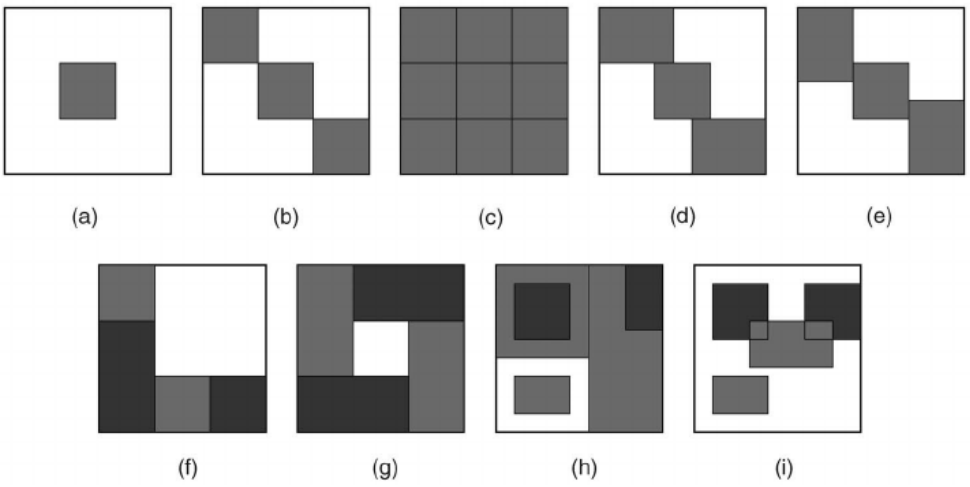


FIGURE 4.1 – Principales structures des co-clusters.

La méthode générique de co-clustering que nous proposons s'appelle BiP (***Bi****clustering* ***P****rocedure*) et possède les caractéristiques suivantes : a) elle permet le traitement d'une matrice dont le type de données peut être soit binaire, qualitatif ou multivalué sans aucune nécessité de recodification préalable des données de la matrice, b) elle se décline en quatre variantes où chaque co-cluster généré peut représenter soit des objets avec les propriétés qu'ils possèdent, des objets avec les propriétés qu'ils ne possèdent pas, une juxtaposition de sous-groupes ayant des profils distincts (et parfois purement opposés) pour un sous-ensemble d'attributs, ou finalement des colonnes pleines de 0 ou pleines de 1 pour exprimer le fait que des objets ont des attributs sans toutefois en posséder d'autres. La première variante correspond à la production de concepts formels dont la procédure est décrite ci-après.

4.3 Génération des concepts formels

Dans ce qui suit, nous proposons la définition d'un co-cluster de type 1, c.-à-d. représentant un concept formel. Cette définition ne produit que les concepts dont la taille de l'intention dépasse 1. Il est toutefois possible de la modifier et d'adapter les algorithmes pour générer l'ensemble des concepts du treillis, y compris l'infimum et le supremum.

4.3.1 Définition

Partons d'une matrice binaire ou contexte formel $\mathbb{K} = (G, M, I)$ avec $G = \{o_1, o_2, ...o_n\}$ l'ensemble des objets représentant les données, $M = \{m_1, m_2, ...m_m\}$ les propriétés de ces objets et $I \subseteq G \times M$ une relation binaire. Un co-cluster de type 1 issu de $\mathbb{K}$ est un couple (X, Y) avec $X \subseteq G$ tel que $|X| \geq 1$ et $Y \subseteq M$ tel que $|Y| > 1$ représentant un rectangle maximal pour l'inclusion dont toutes les valeurs sont égales à 1. On dira que (X, Y) est maximal pour l'inclusion si : $X \times Y \subset I$ et $\forall X_1 \supset X$, $\forall Y_1 \supset Y$, $(X_1 \times Y_1 \subset I$, (X_1, Y_1) a toutes ses valeurs égales à 1 et vérifie la condition $(X = X_1, Y = Y_1)$.

4.3.2 Illustration

Dans le but de clarifier la notion de co-cluster, nous allons l'illustrer par l'exemple d'un contexte $\mathbb{K}$ de huit objets $\{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8\}$ et quatre propriétés $\{m_1, m_2, m_3, m_4\}$.

$$\begin{pmatrix} \mathbb{K} & m_1 & m_2 & m_3 & m_4 \\ O_1 & 0 & 0 & 0 & 0 \\ O_2 & 0 & 1 & 0 & 0 \\ O_3 & 1 & 0 & 0 & 0 \\ O_4 & 1 & 1 & 1 & 1 \\ O_5 & 0 & 1 & 1 & 1 \\ O_6 & 1 & 1 & 0 & 0 \\ O_7 & 1 & 0 & 0 & 0 \\ O_8 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} H_1 & \mathrm{m}_3 & \mathrm{m}_4 \\ o_4 & 1 & 1 \\ o_5 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} H_2 & \mathrm{m}_2 & \mathrm{m}_3 & \mathrm{m}_4 \\ o_4 & 1 & 1 & 1 \\ o_5 & 1 & 1 & 1 \\ o_8 & 1 & 1 & 1 \end{pmatrix}$$

L'exemple ci-dessus traite du cas d'un contexte $\mathbb{K}$ à partir duquel on a extrait deux sous-matrices H_1 et H_2. La première sous-matrice n'est pas un co-cluster alors que la seconde l'est. Notons que H_1 n'est pas un co-cluster puisque il n'est pas maximal.

4.3.3 L'arbre Patricia

L'algorithme BiP que nous nous apprêtons à présenter dans la suite de ce chapitre et les chapitres suivants utilise une structure de données efficace, nommée arbre Patricia, pour stocker et gérer les co-clusters. Cette structure est pratique pour le stockage et la recherche de l'information codée en alphanumérique. Elle a été introduite par [60] en 1968 et est une représentation compacte de l'arbre trie. Cependant, contrairement à un arbre trie régulier, les arêtes de l'arbre Patricia sont étiquetées avec des séquences de caractères plutôt qu'avec

des caractères simples. Il peut s'agir de chaînes de caractères, de chaînes de bits comme des nombres entiers, ou des séquences généralement arbitraires d'objets stockés dans un ordre lexicographique. L'idée est de fusionner les branches en ne gardant que les nœuds internes avec au moins deux fils. Les arêtes de l'arbre ne sont donc plus étiquetées par des lettres mais par des mots. Ainsi, chaque nœud de l'arbre possédant un fils unique sera fusionné avec son parent. Un avantage de l'arbre Patricia est qu'aucun réaménagement de données ou d'index n'est nécessaire lorsque de nouvelles données doivent être ajoutées. Cette structure permet de récupérer des informations en réponse à des clés fournies par l'utilisateur avec un temps de calcul borné linéairement par la longueur des clés et du nombre de leurs occurrences.

- **La recherche d'une chaîne dans l'arbre Patricia**

La recherche d'une chaîne s dans un arbre Patricia est semblable à la recherche de s dans une structure trie sauf que lorsqu'on traverse une arête de l'arbre, on vérifie l'étiquette de l'arête avec un préfixe de s et pas seulement un seul caractère. S'il y a une correspondance, l'arête est traversée. Sinon, la recherche échoue sans avoir trouvé s. Toutefois, si la recherche utilise tous les caractères de s, alors on atteint une feuille correspondant à s. Dans tous les cas, la recherche d'une chaîne dans l'arbre Patricia prend un temps égal à O($|s|$).

- **L'insertion d'une chaîne dans l'arbre Patricia**

L'opération d'insertion d'une chaîne s dans un arbre Patricia est semblable à celle de la recherche décrite ci-dessus. On commence par scruter s dans l'arbre. Deux cas sont envisagés : (i) s n'est pas présent dans l'arbre, ou (ii) la recherche échoue sur une arête e. Dans le premier cas, on insère s dans l'arbre. Dans le second cas, on teste si e partage un préfixe commun avec la chaîne s. Deux cas sont alors à considérer. Si l'arête e n'a pas de préfixe commun avec s, alors on ajoute un fils au nœud sur lequel on se trouve et l'arête les reliant est la fin de la chaîne s à insérer. Sinon, on choisit l'un des préfixes en commun, on supprime l'arête correspondante, puis on crée une arête ayant pour étiquette ce préfixe et on lui ajoute deux fils avec pour étiquettes respectivement le suffixe de l'arête que l'on a supprimée, et la fin de la chaîne s à insérer privée du préfixe commun. Dans tous les cas de figures, l'opération d'insertion prend un temps égal à O ($|s| + |\sum|$) puisqu'elle implique une recherche de s suivie de la création d'au plus deux nouveaux nœuds, chacun de taille O ($|\sum|$), avec ($|\sum|$) représentant la cardinalité de l'alphabet utilisé [59].

Dans le cadre de cette thèse, les mots sont une combinaison d'objets du contexte $\mathbb{K}$ tandis que les valeurs stockées dans les nœuds de l'arbre sont les attributs du contexte. Un exemple de l'arbre Patricia est donné à la figure 7.3. Dans la suite de ce chapitre, les co-clusters de types 3 et 4 sont appelés co-clusters mixtes, et nous utilisons une structure d'arbre Patricia (*Radix-tree*) pour gérer tout les types de coclusters produits par notre algorithme BiP.

4.3.4 Algorithmes

Algorithme 1 : BiP - version concept formel

Entrées : $\mathbb{K} := (G, M, I)$ avec $M := \{m_1, m_2, \ldots, m_m\}$
Sorties : $\mathbb{B}$: arbre Patricia contenant l'ensemble de co-clusters

```
1  m ← |M|
2  n ← |G|
3  V_i := [K[1,i], ..., K[n,i]]^T avec 1 ≤ i ≤ m
4  C ← ∅
5  pour i ← 1 à m − 1 faire
6  |  pour j ← i + 1 à m faire
7  |  |  si |Commun(V_i, V_j)| ≥ 1 alors
8  |  |  |  X ← Commun(V_i, V_j)
9  |  |  |  Y ← {m_i, m_j}
10 |  |  |  C ← C ∪ {(X, Y)}
11 |  fin
12 fin
13 B ← Extraire(C)
14 Retourner B
```

Algorithme 2 : Procédure Commun

Entrées : $V_i[n]$ et $V_j[n]$
Sorties : X : Ensemble des objets communs à V_i et V_j

```
1  X ← ∅
2  pour k ← 1 à n faire
3  |  si V_i[k] = V_j[k] = 1 alors
4  |  |  X ← X ∪ {k}
5  fin
6  Retourner X
```

Algorithme 3 : Procédure Extraire

Entrées : $\mathbb{C}$
Sorties : $\mathbb{B}$: arbre Patricia, ensemble de co-clusters

$\mathbb{B} \leftarrow UnElement(\mathbb{C})$
$\mathbb{C} \leftarrow \mathbb{C} \setminus UnElement(\mathbb{C})$
pour *Chaque* $(X_a, Y_a) \in \mathbb{C}$ **faire**
  pour *Chaque* $(X_b, Y_b) \in \mathbb{B}$ **faire**
    CAS :
    $X_b \subset X_a : \mathbb{B}[X_b] \leftarrow Y_b \cup Y_a$; $\mathbb{B}[X_a] \leftarrow Y_a$
    $X_a \subseteq X_b : \mathbb{B}[X_a] \leftarrow Y_b \cup Y_a$
    $(X_a \cap X_b \neq \emptyset) : \mathbb{B}[X_a \cap X_b] \leftarrow Y_b \cup Y_a$; $\mathbb{B}[X_a] \leftarrow Y_a$
    DÉFAUT : $\mathbb{B}[X_a] \leftarrow Y_a$
    FIN CAS :
  fin
fin
Retourner $\mathbb{B}$

$\mathbb{K}$	m_1	m_2	m_3	m_4	m_5
1	1	1	1	1	1
2	1	1	1	1	1
3	0	1	0	1	0
4	0	1	0	1	1
5	0	1	0	1	1

TABLE 4.2 – Contexte binaire $\mathbb{K}$

Nous proposons ci-après une illustration détaillée de l'algorithme 1 pour le cas de la production de concepts formels.

La première étape de l'algorithme 1 permet de bâtir un ensemble $\mathbb{C}$ composé de couples (X, Y) où $X \subseteq G$ et $Y \subseteq M$, avec $Y = \{m_i, m_j\}$ pour m_i et $m_j \in$ M et $X = \{o_h \in G \mid o_h I m_i = o_h I m_j = 1, m_i, m_j \in Y\}$. Ainsi, pour deux propriétés quelconques de M, l'ensemble X est formé par des objets où chacun d'eux possède la valeur 1 pour la paire d'éléments dans Y. C'est la procédure $\mathbb{C}ommun$ qui permet de bâtir l'ensemble X à partir des éléments de l'ensemble Y. En partant de l'exemple de la table 4.2, l'ensemble $\mathbb{C}$ est constitué au départ des diverses paires possibles d'attributs parmi les cinq attributs existants :
$\mathbb{C} = \{(\{1,2\}, \{m_1, m_2\}), (\{1,2\}, \{m_1, m_3\}), (\{1,2\}, \{m_1, m_4\}), (\{1,2\}, \{m_1, m_5\}), \ldots, (\{1,2,4,5\}, \{m_4, m_5\})\}$.

L'étape suivante consiste à créer l'ensemble $\mathbb{B}$ des co-clusters (cf. la procédure $\mathbb{E}$xtraire de l'algorithme 3) dans une structure d'arbre Patricia [60] dont chaque nœud non terminal identifie un objet et chaque nœud terminal identifie un ensemble de propriétés (attributs) possédées par les objets se trouvant sur le chemin entre cette feuille et la racine de l'arbre.

Le choix d'une telle structure se justifie par son efficacité de stockage des données et de recherche par clé des éléments. La procédure Extraire part de $\mathbb{C}$ pour générer l'ensemble $\mathbb{B}$ des co-clusters maximaux. On commence par placer le premier élément de $\mathbb{C}$ dans l'ensemble $\mathbb{B}$. Ensuite, on va comparer l'élément suivant de $\mathbb{C}$ avec l'ensemble des éléments qui se trouvent déjà dans $\mathbb{B}$ afin de créer de nouveaux blocs qui doivent vérifier la condition définissant un co-cluster. Notons qu'il y a quatre cas à considérer lors de la comparaison de deux couples (X, Y) quand l'ensemble des objets n'est pas identique. Lorsque tous les éléments de $\mathbb{C}$ sont traités, on retourne le contenu de l'arbre $\mathbb{B}$.

Voici une illustration de cette deuxième étape : on commence par placer le premier couple $(\{1,2\}, \{m_1, m_2\})$ dans l'arbre $\mathbb{B}$. À l'étape qui suit, on doit comparer la paire $(X_a, Y_a) = (\{1,2\}, \{m_1, m_3\})$ avec tous les éléments de $\mathbb{B}$. Cependant, l'arbre $\mathbb{B}$ est réduit au seul couple $(X_b, Y_b) = (\{1,2\}, \{m_1, m_2\})$. Comme on est dans le cas où $X_b = X_a$ (cf. ligne 7 de l'algorithme 3, cas 2), on doit ajouter l'attribut m_3 à l'ensemble $\{m_1, m_2\}$ pour former la paire $(\{1,2\}, \{m_1, m_2, m_3\})$. Le même raisonnement sera appliqué aux autres couples contenant l'attribut m_1. Comme c'est toujours le cas 2 avec égalité, on doit alors ajouter les attributs m_4 et m_5 à l'ensemble $(\{1,2\}, \{m_1, m_2, m_3\})$ pour former finalement le couple $(\{1,2\}, \{m_1, m_2, m_3, m_4, m_5\})$. Nous devons à présent comparer le couple $(X_a, Y_a) = (\{1,2\}, \{m_2, m_3\})$ avec l'ensemble du contenu des différents nœud de l'arbre $\mathbb{B}$. On est encore dans le cas 2 de l'algorithme 3 et les deux attributs m_2, m_3 existent déjà dans l'ensemble des attributs de la seule feuille de l'arbre. Donc, aucun changement ne sera apporté à $\mathbb{B}$. La comparaison suivante concerne le couple $(X_a, Y_a) = (\{1,2,3,4,5\}, \{m_2, m_4\})$ de $\mathbb{C}$ avec les éléments de $\mathbb{B}$. On est dans le cas 1 (cf. ligne 6 de l'algorithme 3). On doit ajouter les attributs m_2, m_4 à l'ensemble des attributs du contenu de la seule feuille de l'arbre et ajouter le couple (X_a, Y_a) à cet arbre. Le raisonnement appliqué au couple $(X_a, Y_a) = (\{1,2,4,5\}, \{m_2, m_5\})$ de $\mathbb{C}$ génère une seule et unique modification de l'arbre $\mathbb{B}$ qui consiste à créer un nouveau nœud lequel contient ce couple. Les autres couples $(X_a, Y_a) = (\{1,2\}, \{m_3, m_4\})$ et $(X_a, Y_a) = (\{1,2\}, \{m_3, m_5\})$ de $\mathbb{C}$ n'apportent aucune modification à l'arbre. Toutefois, le dernier couple $(X_a, Y_a) = (\{1,2,4,5\}, \{m_4, m_5\})$ de $\mathbb{C}$ va faire augmenter l'ensemble des attributs du nœud existant d'extension $1,2,4,5$ de l'attribut m_4 pour aboutir au couple $(\{1,2,4,5\}, \{m_2, m_4, m_5\})$.

On constate que le nombre de co-clusters (ici des concepts formels) pouvant être extraits du contexte $\mathbb{K}$ est de trois tel qu'indiqué par la table 4.3. Notons que la figure 4.3 vient aussi confirmer ce résultat.

	m_1	m_2	m_3	m_4	m_5
O_1	1	1	1	1	1
O_2	1	1	1	1	1

	m_2	m_4
O_1	1	1
O_2	1	1
O_3	1	1
O_4	1	1
O_5	1	1

	m_2	m_4	m_5
O_1	1	1	1
O_2	1	1	1
O_4	1	1	1
O_5	1	1	1

TABLE 4.3 – Les trois co-clusters obtenus à partir du contexte $\mathbb{K}$

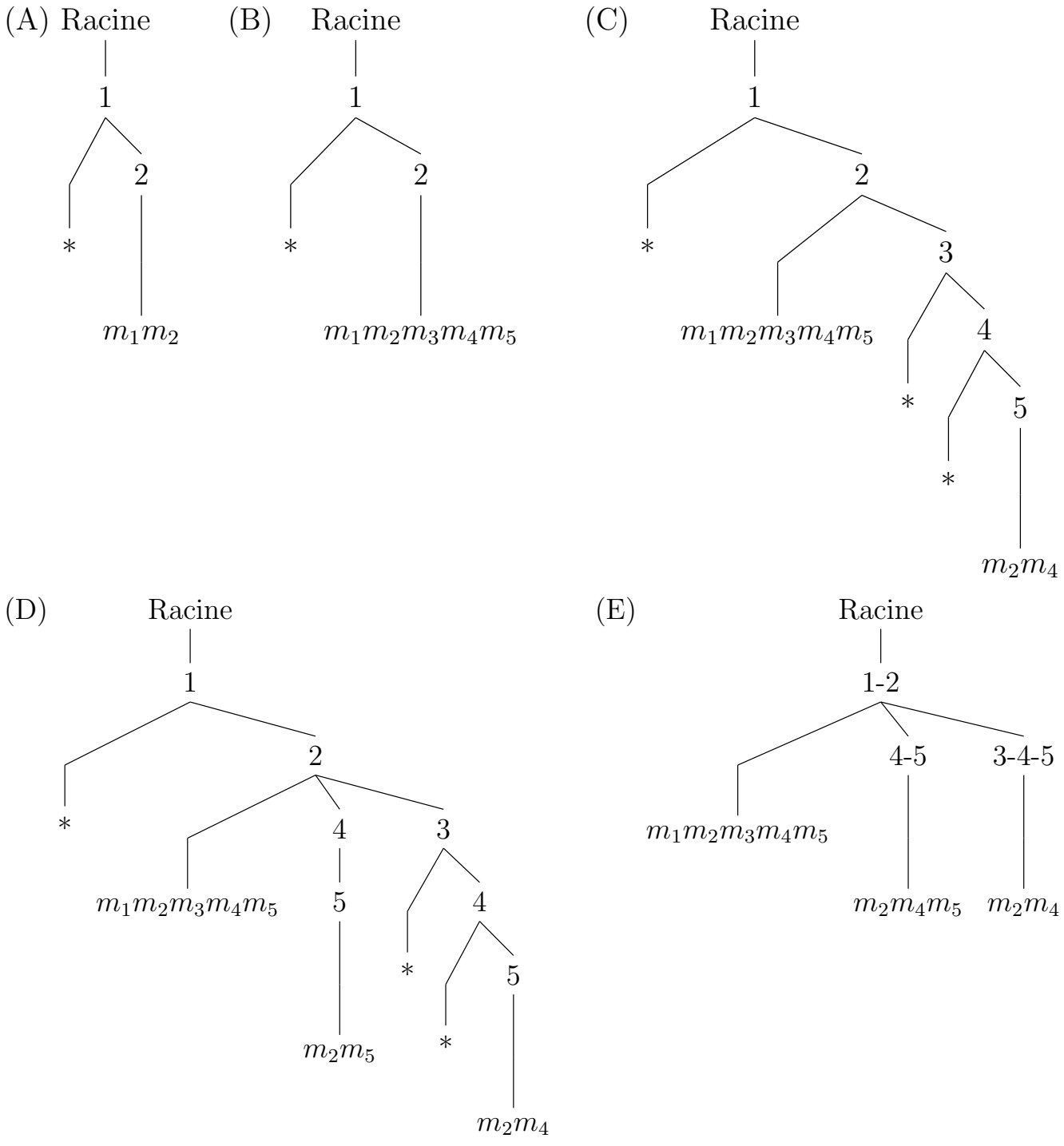


FIGURE 4.2 – États de l'arbre $\mathbb{B}$ durant les différentes phases de l'algorithme

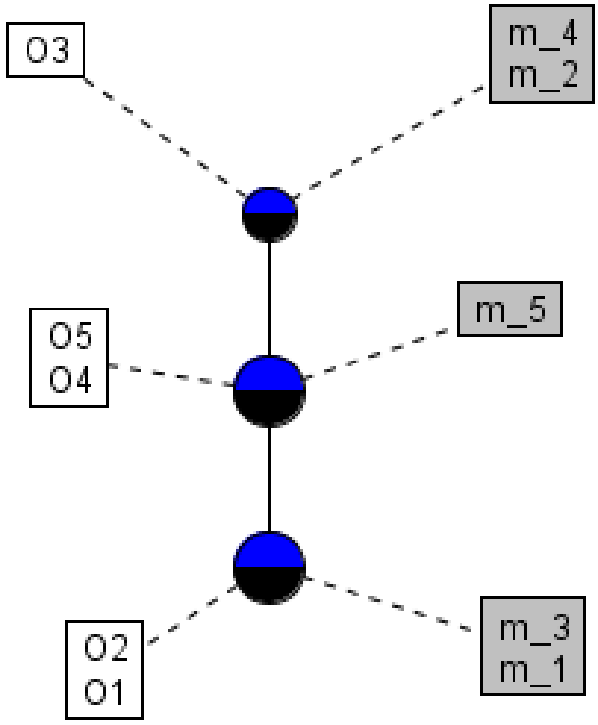


FIGURE 4.3 – Treillis de concepts du contexte $\mathbb{K}$

Dans ce qui suit, nous fournissons une preuve que tout co-cluster de type 1 est un concept formel.

4.3.5 Preuve

Supposons que (X_b, Y_b) dans $\mathbb{B}$ n'est pas un concept formel. Donc, soit $X_b^{'} \neq Y_b$ ou $Y_b^{'} \neq X_b$. Prenons le premier cas, $X_b^{'} \neq Y_b$. Donc, $\exists m \in M$ et $m \in X_b^{'}$ mais $m \notin Y_b$. Donc, il existe des objets de G qui ont comme attribut m mais qui ne sont pas dans X_b. La raison étant que si de tels objets étaient présents dans X_b, alors nécessairement $m \in Y_b$. Soit A, l'ensemble de ces objets. Posons $X_c = X_b \cup A$, on aura $X_b \subset X_c$. D'après (cf. ligne 6 algorithme 3), on aura à additionner à l'ensemble $\mathbb{B}$ le couple $(X_b, Y_b \cup \{m\})$. Ce raisonnement étant valable pour tous les m $\in X_b^{'}$ qui ne sont pas dans Y_b. Il en résulte que le couple $(X_b, X_b^{'}) \in \mathbb{B}$. Le même raisonnement peut être appliqué à $Y_b^{'} \neq X_b$. Ainsi, n'importe quel couple $(X_b, Y_b) \in \mathbb{B}$ est soit un concept formel ou bien il en donne naissance. Toutefois, comme les éléments de $\mathbb{B}$ sont maximaux, alors il ne peut y avoir de couple (X_b, Y_b) dans $\mathbb{B}$ qui n'est pas un concept formel. Ainsi, tous les couples de $\mathbb{B}$ sont des concepts formels.

Il reste à vérifier que les concepts formels issus du contexte $\mathbb{K}$ sont bel et bien tous (à l'exception de ceux dont l'intention a une taille égale à 1, et parfois le supremum et l'infimum) dans $\mathbb{B}$. Pour cela, supposons qu'il existe un concept formel $(X_b, Y_b) \notin \mathbb{B}$. Ainsi, il ne peut exister de couple $(X_a, Y_a) \in \mathbb{C}$ avec $X_a \subset X_b$ et $Y_a \subset Y_b$. Notons que l'existence d'un tel couple générera systématiquement un concept dans $\mathbb{B}$. Cependant, n'importe quel Y_a composé de deux éléments m_1 et $m_2 \in Y_b$ formera un couple $(X_b, Y_a) \in \mathbb{C}$. On en déduit donc, que l'algorithme 3 génère bel et bien des concepts formels contenus dans le contexte $\mathbb{K}$.

4.4 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle méthode de co-clustering nommée BiP qui accepte en entrée une matrice de données binaires pour produire des co-clusters qui ne sont rien d'autre que des concepts formels. Nous avons fourni la preuve de la validité de notre approche pour la production de ce type 1 de co-clusters. Le chapitre suivant est consacré à la version de BiP qui permet de créer des co-clusters de types 2, 3 et 4. Son application à des exemples concrets permet d'obtenir des co-clusters sémantiquement riches et parfois pertinents.

Chapitre 5

Découverte d'autres types de co-clusters

Sommaire

5.1 Introduction

La plupart des algorithmes de co-clustering mettent l'accent sur la présence de propriétés que possèdent les objets ou les individus analysés. Cependant, il pourrait être fort utile dans plusieurs domaines d'application de montrer des groupes d'objets présentant des similarités tant au niveau de la présence que de l'absence (négation) d'attributs. Après avoir présenté dans le chapitre 4 une première variante de l'algorithme BiP permettant de créer des concepts formels (type 1), nous allons décrire dans le présent chapitre les trois autres variantes de notre procédure générique qui permet de produire les co-clusters de types 2, 3 et 4. Le type 2 correspond à un co-cluster dont les cellules sont remplies uniquement de 0, le type 3 est celui où le co-cluster contient un mélange de valeurs 0 et 1, et le dernier type est celui où le co-cluster contient des colonnes complètement pleines de 1

ou[1] complètement pleines de 0. Chacune de ces trois instanciations donne lieu à des motifs ayant une signification spécifique. Rappelons que les co-clusters de type 1 correspondent à la notion de concepts en analyse formelle de concepts [36].

Afin d'établir un pont entre nos deux principales contributions (généralisation d'objets et/ou d'attributs et co-clustering), l'exemple de la phytothérapie (cf. chapitre 3) est alors utilisé pour décrire toutes les étapes de la procédure de co-clustering qui permet de constituer des blocs pouvant servir à bâtir une taxonomie sur les deux dimensions d'un contexte. Ensuite, cette taxonomie peut être employée entre autres pour généraliser les objets (maladies) des données de la phytothérapie et créer en conséquence des motifs généralisés.

Après un peu plus de quatre décennies de recherche sur les techniques de co-clustering, nous pensons qu'il y a matière à amélioration par l'inclusion de la négation (absence) des propriétés des objets au sein des co-clusters. C'est justement le cas des types 2, 3 et 4. Le développement de méthodes de production de motifs avec négation en analyse formelle des concepts [58, 67] est relativement récent et peu exploré et concerne principalement les implications. Or, la prise en compte d'aussi bien la présence que l'absence de propriétés lors de la génération de motifs (concepts formels et règles d'association) peut être fort utile.

Au mieux de notre connaissance, il existe quelques méthodes qui produisent des co-clusters combinant des valeurs 0 et 1 comme par exemple [4, 25]. Toutefois, la configuration de tels co-clusters ne vise pas nécessairement à faire ressortir tout aussi bien la présence que l'absence de propriétés.

5.2 Définitions

- **Attribut négatif**

Soit $\mathbb{K} := (G, M, I)$ un contexte formel où G est un ensemble d'objets et M les attributs de ces objets. Chaque élément de M est noté par la lettre minuscule m et cette notation sera réservée pour représenter les attributs positifs. Aussi, nous utilisons $\overline{m}$ pour noter la négation (l'absence) de l'attribut m et $\overline{M} = \{\overline{m} | m \in M\}$ dont les éléments seront nommés attributs négatifs.

Un élément arbitraire de $M \cup \overline{M}$ sera noté par les premières lettres de l'alphabet a,b,c... etc et $\overline{a}$ dénote l'opposé de a. Ainsi, l'attribut a peut représenter autant un attribut positif que négatif. Aussi, si a = m $\in$ M, alors $\overline{a} = \overline{m}$ et si $a = \overline{m} \in \overline{M}$, alors $\overline{a} = m$. Les lettres majuscules A, B, C dénotent des sous ensembles de $M \cup \overline{M}$. Si $A \subset M \cup \overline{M}$, alors $\overline{A}$ représente l'ensemble $\{\overline{a} | a \in A\}$.

- **Contexte complémentaire**

Soit $\mathbb{K} := (G, M, I)$ un contexte formel, on définit par $\overline{\mathbb{K}} := (G, \overline{M}, (M \times G) \backslash I)$ le contexte complémentaire de $\mathbb{K}$. Le contexte formel complémentaire $\overline{\mathbb{K}}$ a pour but de représenter l'absence des attributs pour les objets du contexte $\mathbb{K}$. Dans une représentation matricielle du contexte $\mathbb{K}$, les 1 seront représentés par des 0 dans $\overline{\mathbb{K}}$ et vice et versa.

[1] Il s'agit du OU inclusif

- **Apposition de contextes**

Soit $\mathbb{K}_1 := (G, M_1, I_1)$ et $\mathbb{K}_2 := (G, M_2, I_2)$ deux contextes formels. On définit par $\mathbb{K} := \mathbb{K}_1|\mathbb{K}_2 = (G, M_1 \cup M_2, I_1 \cup I_2)$ l'apposition de $\mathbb{K}_1$ et $\mathbb{K}_2$.

$$\begin{pmatrix} \mathbb{K}|\overline{\mathbb{K}} & a & b & c & d & \overline{a} & \overline{b} & \overline{c} & \overline{d} \\ O_1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ O_2 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ O_3 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ O_4 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$\mathbb{K}|\overline{\mathbb{K}}$: Exemple de l'apposition de deux contextes

Dans la suite de ce chapitre, nous décrivons et illustrons dans la section 5.3 le fonctionnement de l'algorithme de co-clustering BiP en mettant l'accent sur la production de co-clusters des types 3 et 4. Quant au calcul des co-clusters de type 2 (remplis de 0) et représentant les objets qui ne possèdent pas un ensemble d'attributs, il se fait d'une manière similaire au type 1 en remplaçant simplement la valeur 1 par 0 dans la procédure *Commun* décrite au chapitre 4. Une analyse empirique préliminaire est présentée en section 5.4. Finalement, la section 5.6 conclut ce chapitre et énumère les travaux futurs.

5.3 Autres co-clusters

Dans cette section, nous présentons les variantes 2, 3 et 4 de l'algorithme BiP qui permettent de générer des co-clusters avec des 0 et dont le type peut être parmi l'un des suivants.

1. Toutes les valeurs du co-cluster sont égales à 0
2. Les valeurs du co-cluster sont une combinaison bien définie de 0 et de 1.
3. Les valeurs du co-cluster sont une combinaison de colonnes pleines de 0 et/ou de 1.

Le premier type va générer des rectangles maximaux comportant uniquement des 0, c.-à-d. des concepts formels présentant l'absence de propriétés pour une collection d'objets et donc des concepts formels du contexte $\overline{\mathbb{K}} := (G, \overline{M}, (M \times G) \backslash I)$. Dans ce chapitre, nous allons nous concentrer sur la définition et la génération des co-clusters (X, Y) des types 3 et 4 tels que décrits ci-après. Il est important de noter que l'ensemble des co-clusters de type 4 est équivalent à l'ensemble de concepts formels produit à partir de l'apposition (juxtaposition) du contexte initial $\mathbb{K}$ avec son complémentaire $\overline{\mathbb{K}}$. Toutefois, l'algorithme 7 le génère d'une manière plus astucieuse et efficace. Ainsi, il sera possible d'identifier des sous-groupes d'objets dans X ayant des profils totalement opposés.

5.3.1 Définitions

Étant donné que toutes les valeurs au sein d'un co-cluster de type 1 sont égales à 0, il n'y a pas lieu de donner une définition pour ce type de co-cluster puisqu'il s'agit de produire des rectangles maximaux complètement pleins de 0. Les co-clusters de type 3 de la forme (X, Y) contiennent des sous-groupes d'objets $X_1, \ldots, X_p$ au sein de l'ensemble X avec une variation au niveau de la présence des propriétés de Y. Formellement, un co-cluster de type 3 est une paire (X, Y) avec $X \subseteq G$, $Y \subseteq M$, $|X| \geq 1$, $|Y| > 1$ vérifiant les deux conditions suivantes :

1. Pour tout attribut $m_t \in Y$, il existe au moins un attribut $m_l \neq m_t \in Y$ tel que $\forall o \in X \ (oIm_t \wedge oIm_l) \vee (o \not I m_t \wedge o \not I m_l)$

2. (X, Y) est maximal pour l'inclusion. Cela signifie que : $X \times Y \subset I$ et $\forall X_1 \supset X$, $\forall Y_1 \supset Y$, $(X_1 \times Y_1 \subset I$, et (X_1, Y_1) vérifie la condition 1 $\Rightarrow X = X_1, Y = Y_1)$.

Pour les co-clusters de type 4, seule la condition 1 change :

1. Pour tout attribut $m_t \in Pos(Y)$, il existe au moins un attribut $m_l \neq m_t \in Pos(Y)$ tel que $(\forall o \in X \ oIm_t \wedge oIm_l)$ ou $(\forall o \in X \ o \not I m_t \wedge o \not I m_l)$

où $Pos(Y)$ représente l'ensemble des propriétés de Y exprimées en éléments positifs. À titre d'exemple, $Pos(\{m_2, \overline{m_3}\}) = \{m_2, m_3\}$.

5.3.2 Algorithmes

Partant du contexte (matrice) $\mathbb{K}$, l'algorithme 4 permet d'obtenir des co-clusters de type 3. La première étape consiste à créer l'ensemble $\mathbb{C}$ composé des couples (X, Y) où Y est une paire d'attributs dont les valeurs deux à deux sont identiques pour les objets contenus dans X. Ce traitement est fait à travers les lignes 5 à 12 de l'algorithme 4 et de la procédure *Commun* (algorithme 5). Ensuite, la procédure *Extraire* est lancée pour construire progressivement l'ensemble des co-clusters à partir de $\mathbb{C}$. Les cas considérés dans cette procédure permettent d'assurer la condition 2 de maximalité et de générer tout nouveau bloc.

Algorithme 4 : BiP - Cas 3

Entrées : $\mathbb{K} := (G, M, I)$ avec $M := \{m_1, m_2, \ldots, m_m\}$
Sorties : $\mathbb{B}$: arbre Patricia contenant l'ensemble de co-clusters

```
1  m ← |M|
2  n ← |G|
3  V_i := [K[1,i], ..., K[n,i]]^T avec 1 ≤ i ≤ m
4  C ← ∅
5  pour i ← 1 à m − 1 faire
6  |  pour j ← i + 1 à m faire
7  |  |  si |Commun(V_i, V_j)| ≥ 1 alors
8  |  |  |  X ← Commun(V_i, V_j)
9  |  |  |  Y ← {m_i, m_j}
10 |  |  |  C ← C ∪ {(X, Y)}
11 |  fin
12 fin
13 B ← Extraire(C)
14 Retourner B
```

Algorithme 5 : Procédure Commun

Entrées : $V_i[n]$ et $V_j[n]$
Sorties : X : Ensemble des objets communs à V_i et V_j

```
1  X ← ∅
2  pour k ← 1 à n faire
3  |  si V_i[k] = V_j[k] alors
4  |  |  X ← X ∪ {k}
5  fin
6  Retourner X
```

Algorithme 6 : Procédure Extraire

Entrées : $\mathbb{C}$
Sorties : $\mathbb{B}$: arbre Patricia, ensemble de co-clusters

$\mathbb{B} \leftarrow UnElement(\mathbb{C})$
$\mathbb{C} \leftarrow \mathbb{C} \setminus UnElement(\mathbb{C})$
for ***Chaque*** $(X_a, Y_a) \in \mathbb{C}$ **do**
  for ***Chaque*** $(X_b, Y_b) \in \mathbb{B}$ **do**
    CAS :
    $X_b \subset X_a : \mathbb{B}[X_b] \leftarrow Y_b \cup Y_a \,;\, \mathbb{B}[X_a] \leftarrow Y_a$
    $X_a \subseteq X_b : \mathbb{B}[X_a] \leftarrow Y_b \cup Y_a$
    $(X_a \cap X_b \neq \emptyset) : \mathbb{B}[X_a \cap X_b] \leftarrow Y_b \cup Y_a \,;\, \mathbb{B}[X_a] \leftarrow Y_a$
    DÉFAUT : $\mathbb{B}[X_a] \leftarrow Y_a$
    FIN CAS :
  end
end
Retourner $\mathbb{B}$

Dans ce qui suit, nous présentons une procédure permettant de produire des co-clusters correspondant aux concepts formels obtenus quand on prend en considération aussi bien la présence que l'absence (la négation) d'attributs. Autrement dit, on cherche à obtenir des co-clusters contenant une combinaison de colonnes pleines de 1 et de colonnes pleines de 0. On adoptera tout au long de cette illustration la convention suivante : si m est un attribut alors on notera $\overline{m}$ sa négation. Aussi, lorsque deux objets o_i et o_j partagent la même valeur 1 pour deux attributs m_l et m_k, ils sont représentés dans le couple utilisé à produire un des co-clusters par $(\{o_i, o_j\} ; \{m_l, m_k\})$. Toutefois, si la valeur partagée est 0, le couple sera noté $(\{o_i, o_j\} ; \{\overline{m_l}, \overline{m_k}\})$.

Pour la production des co-clusters de type 4, l'algorithme 7 est différent de l'algorithme 4 au niveau uniquement du calcul des paires d'attributs communs à un ensemble d'objets tel que décrit par la procédure *CommunPN*.

Algorithme 7 : BiP - Cas 4

Entrées : $\mathbb{K} := (G, M, I)$ avec $M := \{m_1, m_2, \ldots, m_m\}$
Sorties : $\mathbb{B}$: arbre Patricia contenant l'ensemble de co-clusters

$m \leftarrow |M|$
$n \leftarrow |G|$
$V_i := [\mathbb{K}[1, i], \ldots, \mathbb{K}[n, i]]^T$ *avec* $1 \leq i \leq m$
$\mathbb{C} \leftarrow \emptyset$
pour $i \leftarrow 1$ ***à*** $m - 1$ **faire**
  pour $j \leftarrow i + 1$ ***à*** m **faire**
    si $|CommunPN\,(V_i, V_j, 1)\,| \geq 1$ **alors**
      $X \leftarrow CommunPN\,(V_i, V_j, 1)$
      $Y \leftarrow \{m_i, m_j\}$
      $\mathbb{C}_1 \leftarrow \mathbb{C}_1 \cup \{(X, Y)\}$
    sinon si $|CommunPN\,(V_i, V_j, 0)\,| \geq 1$ **alors**
      $X \leftarrow CommunPN\,(V_i, V_j, 0)$
      $Y \leftarrow \{\overline{m_i}, \overline{m_j}\}$
      $\mathbb{C}_0 \leftarrow \mathbb{C}_0 \cup \{(X, Y)\}$
  fin
fin
$\mathbb{C} \leftarrow \mathbb{C}_0 \cup \mathbb{C}_1$
$\mathbb{B} \leftarrow Extraire\,(\mathbb{C})$
Retourner $\mathbb{B}$

Algorithme 8 : Procédure CommunPN

Entrées : $V_i[n], V_j[n]$ et P
Sorties : X : Ensemble des objets communs à V_i et V_j

$X \leftarrow \emptyset$
pour $k \leftarrow 1$ ***à*** n **faire**
  si $V_i[k] = V_j[k] = P$ **alors**
    $X \leftarrow X \cup \{k\}$
fin
Retourner X

5.3.3 Illustration des algorithmes

$\mathbb{K}$	m_1	m_2	m_3	m_4	m_5
1	1	1	1	1	1
2	1	1	1	1	1
3	0	1	0	1	0
4	0	1	0	1	1
5	0	1	0	1	1

TABLE 5.1 – Contexte binaire $\mathbb{K}$

Nous proposons ci-après une illustration détaillée de l'algorithme 4 pour le cas de la production des co-clusters de type 3 pour ensuite discuter brièvement de la génération des co-clusters de type 4.

Au départ, l'ensemble $\mathbb{C}$ est constitué des diverses paires possibles d'attributs parmi les cinq attributs existants :

$$\mathbb{C} = \{(\{1,2\},\{m_1,m_2\}),(\{1,2,3,4,5\},\{m_1,m_3\}), \quad \ldots,(\{1,2,4,5\},\{m_4,m_5\})\}.$$

L'étape suivante consiste à créer l'ensemble $\mathbb{B}$ des co-clusters dans une structure d'arbre Patricia [60] dont chaque nœud non terminal identifie un objet et chaque nœud terminal identifie un ensemble de propriétés (attributs) possédées par les objets se trouvant sur le chemin entre cette feuille et la racine de l'arbre. Le choix d'une telle structure se justifie par son efficacité de stockage des données et de recherche par clé des éléments. Le délai de traitement requis pour trouver une occurrence d'une clé (séquence d'objets) ou pour en conclure qu'il n'y en a pas, a une borne qui dépend linéairement de la longueur de la clé et est indépendante de la taille de l'ensemble des objets ou des attributs.

On commence par placer le premier couple $(\{1,2\},\{m_1,m_2\})$ dans l'arbre $\mathbb{B}$. À l'étape qui suit, on doit comparer la paire $(X_a,Y_a) = (\{1,2,3,4,5\},\{m_1,m_3\})$ avec tous les éléments de $\mathbb{B}$. Cependant, l'arbre $\mathbb{B}$ est réduit au seul couple $(X_b,Y_b) = (\{1,2\},\{m_1,m_2\})$. Comme on est dans le cas où $X_b \subset X_a$ (cf. ligne 6 de l'algorithme 6), on doit ajouter l'attribut m_3 à l'ensemble $\{m_1,m_2\}$ pour former la paire $(\{1,2\},\{m_1,m_2,m_3\})$. En outre, une autre feuille sera créée au niveau de l'arbre $\mathbb{B}$ afin d'y insérer le couple $(\{1,2,3,4,5\},\{m_1,m_3\})$. L'élément suivant à considérer dans $\mathbb{C}$ est le couple $(\{1,2\},\{m_1,m_4\})$ qui sera alors comparé avec les deux éléments existants dans $\mathbb{B}$, c.-à-d. les deux branches menant aux deux feuilles de l'arbre. Notons qu'au niveau de la première feuille, il y a égalité entre l'ensemble des objets de ce couple et celui de la feuille. On est donc dans le cas d'égalité des extensions (cf. ligne 7 de l'algorithme 6). Ainsi, le contenu de cette feuille est augmenté de l'attribut m_4. La comparaison du même couple $(\{1,2\},\{m_1,m_4\})$ avec la deuxième feuille représentant $(\{1,2,3,4,5\},\{m_1,m_3\})$ montre que l'on est dans le cas où $X_a \subseteq X_b$ (cf. ligne 7 de l'algorithme 6). Toutefois, un nœud avec l'ensemble des objets de ce couple existe déjà au sein de l'arbre $\mathbb{B}$ et son ensemble d'attributs contient déjà le sous-ensemble $\{m_1,m_4\}$. Aucune modification ne sera alors faite sur le contenu de ce nœud. Le dernier couple de l'ensemble $\mathbb{C}$ contenant l'attribut m_1 va être comparé avec les éléments de l'arbre $\mathbb{B}$ est $(\{1,2,3\},\{m_1,m_5\})$. Le premier élément de l'arbre avec lequel on doit faire la comparaison est la feuille contenant le couple $(\{1,2\},\{m_1,m_2,m_3,m_4\})$. Étant donné que $\{1,2\} \subset \{1,2,3\}$, on applique alors les instructions de la ligne 6 de l'algorithme 6. L'attribut m_5 devra alors être ajouté au contenu de cette feuille et un nouveau nœud devra être créé pour le couple $(\{1,2,3\},\{m_1,m_5\})$. De même, la comparaison sera faite avec le contenu de la feuille $(\{1,2,3,4,5\},\{m_1,m_3\})$, laquelle va ajouter l'attribut m_3 à l'intention du nœud $(\{1,2,3\},\{m_1,m_5\})$ pour aboutir à $(\{1,2,3\},\{m_1,m_3,m_5\})$. Le résultat final de cette dernière comparaison va modifier successivement la structure de l'arbre (A) pour aboutir à la structure de l'arbre (D).

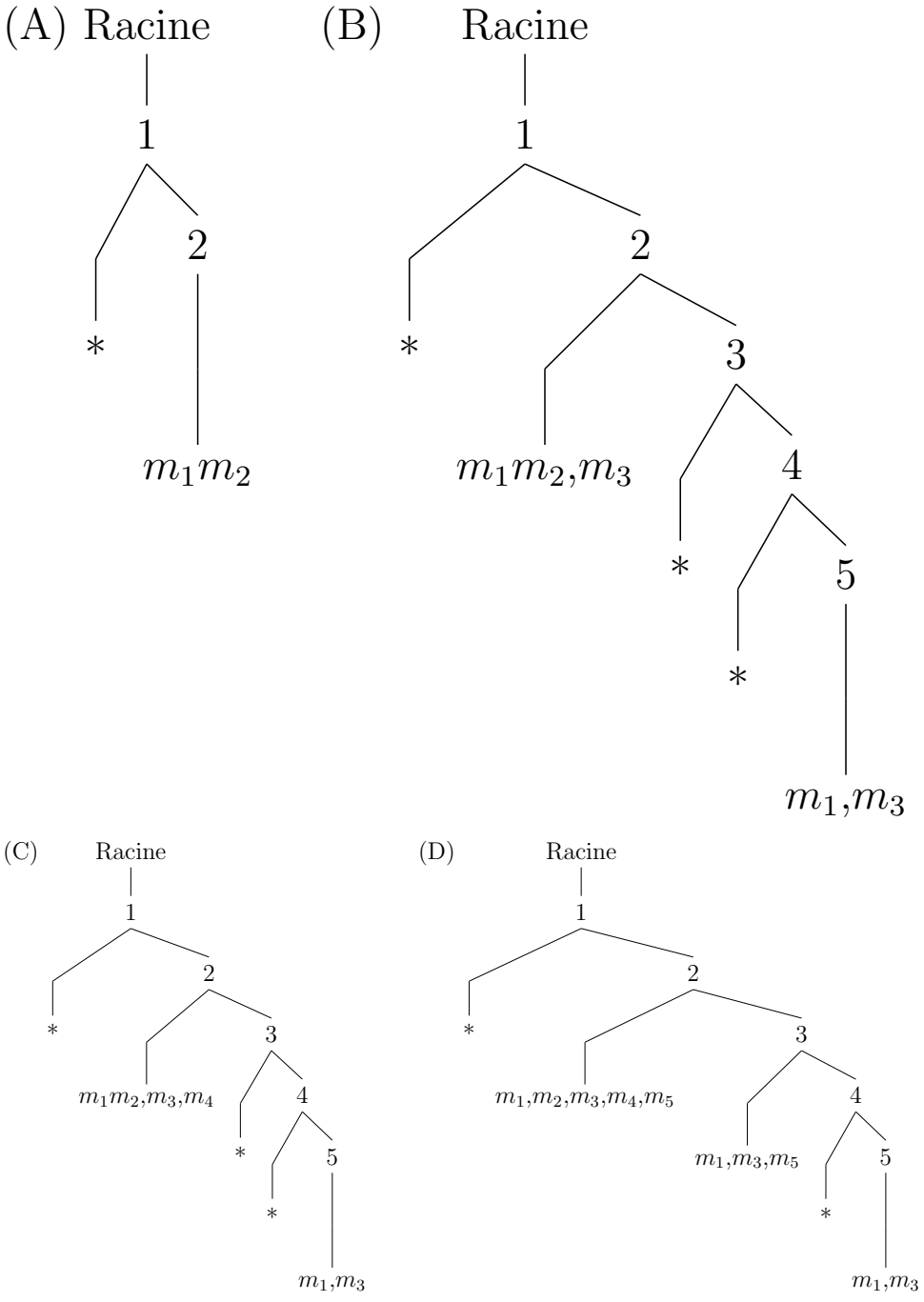


FIGURE 5.1 – États de l'arbre $\mathbb{B}$ lors de la prise en compte des couples contenant l'attribut m_1

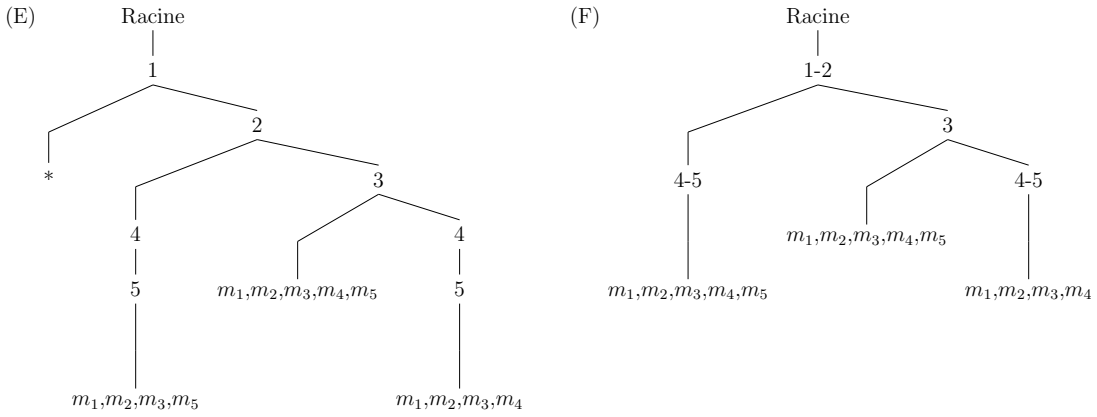


FIGURE 5.2 – États final de l'arbre $\mathbb{B}$

Les comparaisons suivantes sont celles impliquant les paires d'attributs de l'ensemble $\mathbb{C}$ qui contiennent m_2. Le même raisonnement que celui utilisé avec les couples contenant m_1 devra être employé pour les paires contenant l'attribut m_2. Notons aussi que ce sont ces mêmes traitements qui devront être appliqués pour le reste des paires d'attributs. Ainsi, nous aboutissons à la structure (E) de l'arbre $\mathbb{B}$ une fois que les couples contenant m_2 sans m_1 sont traités. En utilisant le même raisonnement pour les paires restantes contenant m_3 puis m_4, on constate que plus aucun changement n'est apporté à l'arbre $\mathbb{B}$ lorsqu'on considère l'attribut m_3 mais que l'attribut m_4 génère un arbre $\mathbb{B}$ dont la structure est (F). Voici ci-après les trois co-clusters extraits de l'arbre Patricia et du contexte initial par sélection sur les objets du co-cluster et par projection sur leurs attributs.

C_1	m_1	m_2	m_3	m_4	m_5
1	1	1	1	1	1
2	1	1	1	1	1
3	0	1	0	1	0

C_2	m_1	m_2	m_3	m_4	m_5
1	1	1	1	1	1
2	1	1	1	1	1
4	0	1	0	1	1
5	0	1	0	1	1

C_3	m_1	m_2	m_3	m_4
1	1	1	1	1
2	1	1	1	1
3	0	1	0	1
4	0	1	0	1
5	0	1	0	1

TABLE 5.2 – Les trois co-clusters extraits du contexte binaire $\mathbb{K}$

Si nous reprenons le même exemple présenté dans la table 5.1 pour produire les co-clusters de type 4, on va d'abord faire appel à la procédure *CommunPN* qui génère les couples (X_i, Y_i) où X_i est l'ensemble d'objets ayant les attributs de Y_i. Ce dernier sous-ensemble représente les diverses paires possibles d'attributs de l'ensemble M pour $\mathbb{C}_1$ et de l'ensemble $\overline{M}$ pour $\mathbb{C}_0$.

On aboutit alors à :

$\mathbb{C}_1 = \{(\{1,2\},\{m_1,m_2\});(\{1,2\},\{m_1,m_3\});(\{1,2\},\{m_1,m_4\});(\{1,2\},\{m_1,m_5\});$
$(\{1,2\},\{m_2,m_3\});(\{1,2,3,4,5\},\{m_2,m_4\});(\{1,2,4,5\},\{m_2,m_5\});(\{1,2\},\{m_3,m_4\});$

$(\{1,2\},\{m_3, m_5\}); (\{1, 2, 4, 5\}, \{m_4, m_5\})\}$.

$\mathbb{C}_0 = \{(\{3, 4, 5\}, \{\overline{m_1}, \overline{m_3}\}); (\{3\}, \{\overline{m_1}, \overline{m_5}\}); (\{3\}, \{\overline{m_3}, \overline{m_5}\})\}$.

À l'étape suivante, on applique la procédure **Extraire**. Le résultat est illustré sur l'arbre Patricia (A). Six co-clusters seront crées. Ils sont illustrés ci-après :

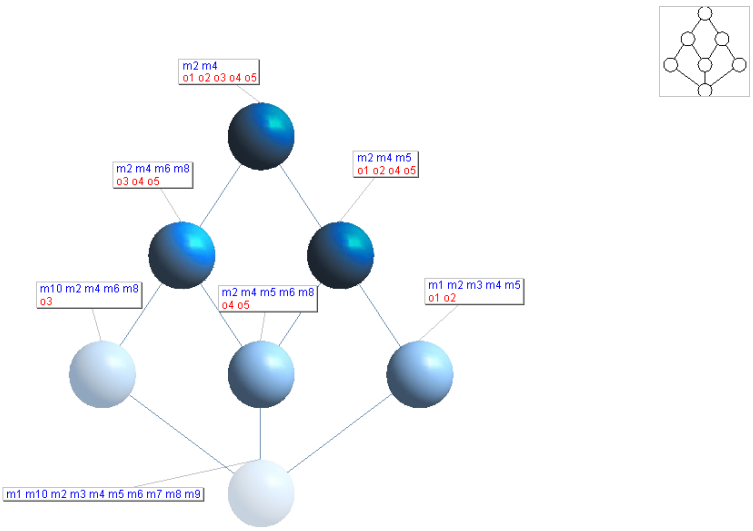
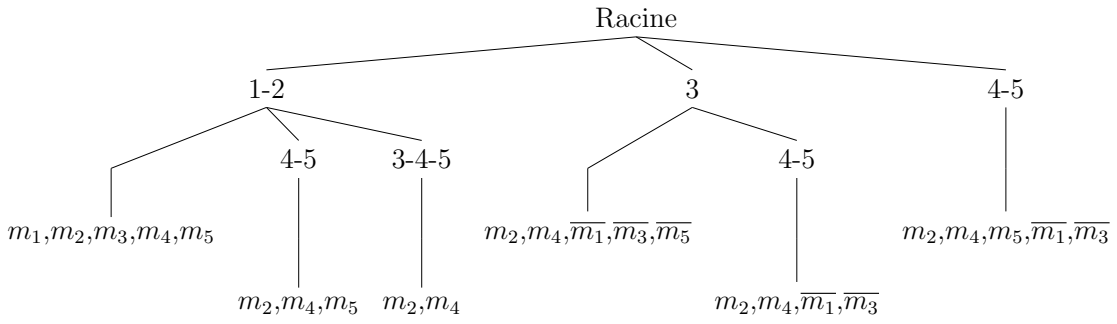


FIGURE 5.3 – Treillis mettant en évidence les concepts formels obtenus par la concaténation du contexte $\mathbb{K}$ avec son complémentaire. Avec $m_6 = \overline{m_1}$ .... $m_{10} = \overline{m_5}$

(A)



C_1	m_1	m_2	m_3	m_4	m_5
1	1	1	1	1	1
2	1	1	1	1	1

C_2	m_1	m_2	m_3	m_4	m_5
4	0	1	0	1	1
5	0	1	0	1	1

C_3	m_1	m_2	m_3	m_4
3	0	1	0	1
4	0	1	0	1
5	0	1	0	1

C_4	m_2	m_4	m_5
1	1	1	1
2	1	1	1
4	1	1	1
5	1	1	1

C_5	m_2	m_4
1	1	1
2	1	1
3	1	1
4	1	1
5	1	1

C_6	m_1	m_2	m_3	m_4	m_5
3	0	1	0	1	0

TABLE 5.3 – Les six co-clusters extraits du contexte binaire $\mathbb{K}$

Ainsi, le co-cluster C_3 nous indique que les objets 3, 4 et 5 se ressemblent dans le sens où ils partagent les propriétés m_2 et m_4 mais ne possèdent nullement les propriétés m_1 et m_3. Il peut être alors exprimé sous forme de la paire $(\{3,4,5\}, \{\overline{m_1}, m_2, \overline{m_3}, m_4\})$.

5.3.4 Analyse théorique de complexité

Afin de conduire cette analyse, nous avons choisi comme procédure de référence l'un des algorithmes de co-clustering les plus connus produisant des concepts formels. Il s'agit de l'algorithme Bimax qui est considéré comme étant rapide et fiable par rapport à d'autres algorithmes énumérant l'ensemble des co-clusters maximaux remplis de 1. Dans la sous-section 5.3.4.1, on décrit le fonctionnement de Bimax et on l'emploie comme référence dans la sous-section 5.3.5 pour conduire notre étude comparative.

5.3.4.1 Algorithme Bimax

Bimax est un algorithme opérant uniquement sur une matrice binaire. Cependant, en présence d'une matrice qui ne l'est pas, les données peuvent être converties de différentes façons dont la plus simple est l'utilisation d'un seuil qui peut être égal à la moyenne ou la médiane des données de la matrice. Toutefois, ce seuil peut aussi être fixé par l'utilisateur

de l'algorithme. Notons qu'il existe d'autres méthodes d'échellonnage qui peuvent être employées.

Bimax est une algorithme de type "diviser pour régner" utilisant une approche récursive et non directe d'énumération de l'ensemble des co-clusters remplis de 1 (concepts formels). Il commence par choisir n'importe quelle ligne l de la matrice A contenant un mélange de 0 et de 1. Dans le cas où une telle ligne n'existe pas, alors soit que toutes les valeurs de A sont égales à 1 auquel cas c'est un co-cluster au sens de la définition de Bimax ou c'est juste des 0 et dans ce cas, il ne l'est pas. Pour trouver les sous-matrices de A, les colonnes $C = \{1, .., m\}$ sont divisées verticalement en deux sous-matrices : une où toutes les valeurs de la ligne l sont égales à 1 et l'autre est celle où toutes les valeurs de cette même ligne sont égales à 0. On aboutit alors à deux sous-matrices $C_U = \{c : A[l, c] = 1\}$ et $C_V = C - C_U$.

L'étape suivante consiste à diviser les n lignes de la matrice A en trois sous-ensembles R_U, R_W et R_V. Le premier sous-ensemble contient les lignes de C_U avec comme seule valeur 1. Quant à R_W, il contient les lignes de C_U et C_V où toutes les valeurs sont égales à 1. Finalement, R_V contient l'ensemble des lignes avec comme seule valeur 1 dans C_V. On génère alors deux nouvelles sous-matrices $U = (R_U \cup R_W, C_U)$ et $V = (R_W \cup R_V, C_U \cup C_V)$. La même procédure reprend récursivement sur les deux matrices U et V. L'algorithme s'arrête lorsque plus aucun réaménagement n'est possible.

5.3.4.2 Calcul de la complexité temporelle de BiP

Après avoir présenté Bimax, nous allons calculer la complexité temporelle de BiP et la comparer à celle de Bimax. Notons que Bimax [65] a une complexité $\Theta(n.m.\beta.min(n, m))$ où β est le nombre de co-clusters générés. Si nous supposons que le contexte $\mathbb{K}$ a plus de lignes que de colonnes ($n > m$), alors Bimax a une complexité maximale de $\Theta(n.m^2.\beta)$. Dans le cas de notre procédure BiP dans sa généralité, la complexité de la procédure *Commun* est $\Theta(n)$. Le temps d'exécution des lignes 5 à 12 de l'algorithme 1 est alors $\Theta(n.m^2)$ du fait que l'ensemble $\mathbb{C}$ a au plus $\frac{m(m-1)}{2}$ éléments. L'exécution de la procédure *Extraire* est faite p fois où la variable p est bornée par le nombre β de co-clusters. Comme certaines instructions de cette boucle sont des opérations d'intersection, de comparaison et d'union d'ensembles, elles peuvent être exécutées en temps linéaire. En plus, l'ajout et le remplacement des éléments à l'arbre $\mathbb{B}$ peuvent être réalisés dans un temps $\Theta(1)$ et au pire cas $\Theta(n)$. Aussi, vu que $\mathbb{C}$ a au plus $\frac{m(m-1)}{2}$ éléments, la boucle externe (cf. ligne 5) s'exécute dans un temps $\Theta(m^2)$, et la complexité de calcul des instructions des deux boucles imbriquées est alors $\Theta\left((m + |\Sigma|).\beta.\frac{m(m-1)}{2}\right)$. Ainsi, la complexité totale de l'algorithme BiP est $\Theta\left(\frac{n.m^2}{2} + (m + |\Sigma|).\beta.\frac{m(m-1)}{2}\right)$. Toutefois $m + |\Sigma| \leq m + n$. Donc la complexité de BiP est comparable à celle de Bimax $\Theta(n.m^2.\beta)$.

5.3.5 Étude comparative

Après avoir présenté l'algorithme BiP avec ses différentes variantes et prouvé son exactitude à produire des concepts formels, il serait important de comparer la complexité de notre

algorithme avec celle des meilleurs algorithmes de l'analyse formelle des concepts tant sur le plan théorique qu'expérimental. Dans cette comparaison, nous allons nous limiter au cas des co-clusters de type 1 car au meilleur de notre connaissance aucun algorithme connu de l'AFC n'est en mesure de produire des co-clusters de type 3 ou 4. Les seuls travaux réalisés par la communauté de l'AFC qui tiennent compte de la négation sont ceux ayant porté sur la production des implications avec négation [58, 67].

Dans leur étude comparative des algorithmes produisant tous les concepts formels et/ou le treillis des concepts, les auteurs [43] considèrent qu'un algorithme est optimal s'il génère le treillis des concepts avec une complexité temporelle polynomiale et une complexité en espace linéaire par rapport au nombre de concepts formels produits. Dans cette étude, un certain nombre d'algorithmes de l'AFC connus pour leur performance ont été sélectionnés. Nous utiliserons ces mêmes algorithmes et Bimax pour conduire notre comparaison théorique alors que seul Bimax sera employé pour la comparaison expérimentale étant donné que c'est l'algorithme de référence le plus utilisé dans les études comparatives des méthodes de co-clustering.

5.3.5.1 Comparaison sur le plan théorique

Des tests empiriques antérieurs de comparaison des performances de BiP avec Bimax ont fourni des résultats intéressants en faveur de notre algorithme. Contrairement à l'algorithme Bimax qui génère des co-clusters remplis seulement de 1, nous générons des co-clusters selon les quatre types : 1, 2, 3 et 4, puis nous interprétons la description des groupes obtenus. Rappelons que Bimax utilise les colonnes avec des 1 d'une certaine rangée du contexte comme modèle pour effectuer le fractionnement de la matrice représentant le contexte. Dans ce cas particulier, si la matrice est peu dense, le fractionnement pourrait conduire à une division déséquilibrée. Ce fait, conjugué à la forme carrée de la matrice, pourrait conduire à une complexité du temps de fonctionnement le plus défavorable de $\Theta\left(n^3\beta\right)$ [63].

Nom de l'algorithme	Complexité théorique
NextClosure	$\Theta\left(\lvert G\rvert^2 . \lvert M\rvert . \lvert L\rvert\right)$
Bordat	$\Theta\left(\lvert G\rvert . \lvert M\rvert^2 . \lvert L\rvert\right)$
Chein	$\Theta\left(\lvert G\rvert^3 . \lvert M\rvert . \lvert L\rvert\right)$
Nourine	$\Theta\left((\lvert G\rvert + \lvert M\rvert) . \lvert G\rvert . \lvert L\rvert\right)$
BiP	$\Theta\left(\lvert G\rvert . \lvert M\rvert^2 . \lvert L\rvert\right)$
BiMax	$\Theta\left(\lvert G\rvert . \lvert M\rvert^2 . \lvert L\rvert\right)$

TABLE 5.4 – Tableau de la complexité théorique de six algorithmes de l'AFC. $|L|$ taille du treillis.

Les complexités temporelles théoriques indiquées dans la table 5.4 montrent que la meilleure d'entre elles est obtenue par l'algorithme Nourine suivi de celles des algorithmes Bordat, BiP et Bimax lesquels possèdent la même complexité. Comme BiP emploie une structure d'arbre Patrica laquelle utilise moins d'espace que l'arbre trie, sa complexité spatiale est plus faible que celle des algorithmes utilisant une structure de trie. Rappelons que pour l'arbre Patricia, un nœud qui a moins de deux fils est fusionné avec son parent. On aboutit ainsi à une structure plus compacte que le trie.

5.3.5.2 Comparaison sur le plan empirique

Cette comparaison à été menée avec du code en C++ pour les deux algorithmes BiP et Bimax. Les tests ont été conduits sur un PC Intel 2.4GHz avec une faible mémoire RAM de 4 GO et le système d'exploitation Windows 10. Dans un premier temps, nous avons comparé les temps d'exécution des deux algorithmes sur des jeux de données générés aléatoirement. Chaque jeu produit correspond à une matrice de 100 objets et 100 attributs. Au total, on a produit 50 jeux constitués en 5 groupes de 10 jeux chacun. Tous les contextes d'un même groupe ont la même densité. Ainsi, le premier groupe a une densité de 10, le suivant 15, ensuite 20, 25 et finalement 30. Pour chacun des algorithmes, une valeur moyenne des temps d'exécution obtenus sur l'ensemble des dix contextes d'un même groupe a été déterminée. Ci-dessous les résultats obtenus.

Densité	Temps moyen BiP	Temps moyen Bimax
10	5.29	2.9265
15	7.1249	3.1915
20	9.0727	3.7043
25	15.4034	4.9566
30	47.634	11.8456

TABLE 5.5 – Tableau comparant les temps d'exécution en seconde de BiP et Bimax

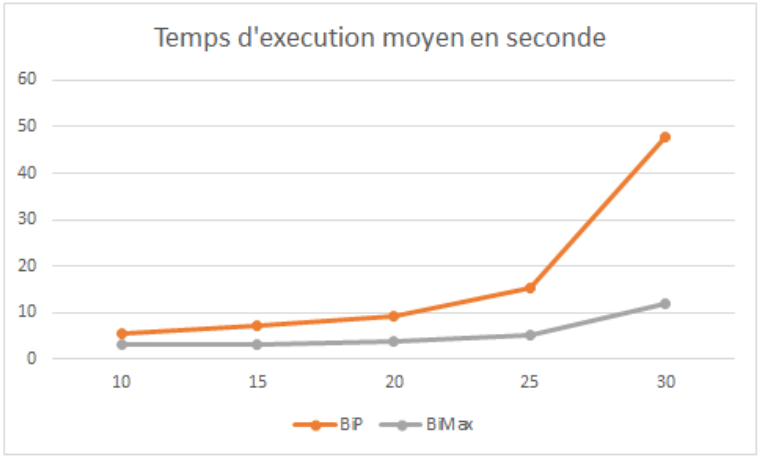


FIGURE 5.4 – Graphe comparant les temps d'exécution de BiP et de Bimax

La table 5.5 indique des temps d'exécution meilleurs pour Bimax. Toutefois, il faut noter que l'algorithme BiP stocke les co-clusters produits dans une structure d'arbre Patricia. Donc après création des co-clusters, on est en mesure de chercher et d'afficher uniquement ceux répondant à un besoin spécifique d'un utilisateur. Ce dernier pourra par exemple vouloir extraire uniquement les co-clusters contenant un certain nombre d'objets. L'algorithme Bimax n'offre pas cette fonctionnalité. Dans un deuxième temps, nous avons exécuté BiP et Bimax sur la base de données Mushroom. Comme les données de cette base sont discrètes, il nous a fallu procéder à une opération d'échellonnage des données. Rappelons que cette transformation n'est pas nécessaire pour BiP puisqu'il est en mesure d'utiliser directement des données discrètes en entrée.

La base Mushroom est composée de 8124 champignons partageant 19 attributs chacun dont deux sont des attributs de classification. L'opération d'échellonnage de cette base conduit à un contexte de 8124 champignons et de 119 attributs dont deux sont utilisés pour la classification. Pour extraire les co-clusters du contexte Mushroom, BiP a consommé un temps d'exécution bien inférieur à celui utilisé par Bimax. Ainsi, il a fallu à BiP 159.92 secondes pour produire les co-clusters alors que Bimax a pris un temps bien supérieur de $1.048s$ pour les générer.

5.4 Expérimentation

Le but de cette section est de (i) se servir de l'exemple [18] (cf. annexe) pour illustrer la qualité et la richesse sémantique des co-clusters générés de type 3, (ii) utiliser les données de la phytothérapie pour produire des co-clusters pouvant servir à suggérer une taxonomie sur les objets, et (iii) estimer les temps d'exécution de la procédure BiP en faisant varier la taille et la densité du contexte (de la matrice) ainsi que la distribution des 1 au sein du contexte. Dans ce qui suit, nous présentons et discutons la signification des co-clusters obtenus en appliquant notre méthode sur l'exemple de Davis fourni en annexe.

5.4.1 L'exemple de Davis

L'exemple connu de Davis décrit la participation de dix-huit femmes du sud des États-Unis à quatorze événements sociaux [18] et a été utilisé par Freeman pour mener une méta-analyse des outils et des techniques d'identification de groupes sociaux [30].

Devant le grand nombre de co-clusters générés au cours de cette opération de co-clustering, nous avons choisi d'illustrer seulement certains d'entre eux qui montrent clairement l'importance de considérer autant les 0 que les 1 lors de la création de co-clusters. On peut aussi voir que les motifs qui en découlent sont clairement identifiables et ont un haut niveau de pertinence.

Davis	E3	E5	E11	E14
1	1	1	0	0
2	1	1	0	0
3	1	1	0	0
4	1	1	0	0
5	1	1	0	0
6	1	1	0	0
8	0	0	0	0
10	0	0	0	0
11	0	0	0	0
14	0	0	1	1
15	0	0	1	1
16	0	0	0	0

TABLE 5.6 – Co-cluster de type 3 extrait de l'exemple de Davis

Le co-cluster ci-dessus dévoile l'existence de trois groupes de femmes :
I= $\{1, 2, 3, 4, 5, 6\}$, II= $\{8, 10, 11, 16\}$ et III = $\{14, 15\}$. Alors que les femmes du premier groupe assistent aux événements E_3 et E_5, celles des deux autres groupes n'assistent pas à ces événements. En outre, seules les femmes du troisième groupe assistent aux événements E_{11} et E_{14}.

5.5 Création d'une taxonomie

Dans ce qui suit, nous allons montrer comment on peut utiliser les co-clusters produits par BiP pour construire une taxonomie lorsque cette dernière n'est pas évidente ou inexistante au sein du contexte initial. À cette fin, on va mener une série de tests sur l'exemple de la phytothérapie précédemment utilisé dans le chapitre 3. Notre but est double, d'un côté on veut extraire de ce contexte des sous-ensembles de maladies à regrouper, et de l'autre, montrer l'utilité d'avoir créé ces groupes en mettant en évidence certains motifs qu'on n'aurait pas pu extraire de ce contexte sans avoir créé cette taxonomie et l'avoir utilisée pour généraliser ce contexte.

5.5.0.1 Démarche

La version originale du contexte sur la phytothérapie est composée de 143 maladies ou stimulants et de 107 plantes. Rappelons que ce contexte a été créé en se basant sur une information extraite de [2]. Dans un premier temps, nous avons supprimé de ce contexte toutes les plantes qui soignent moins de deux maladies. Nous avons aussi enlevé de la liste des objets de ce contexte tous les stimulants en ne laissant que les objets qui correspondent à une maladie. Le contexte final obtenu peut ainsi être représenté par une matrice binaire de 111 maladies et 87 plantes (Contexte réduit). La figure 5.5 illustre le treillis de Galois représentant ce contexte. On a pu extraire de ce contexte 222 co-clusters de type 1.

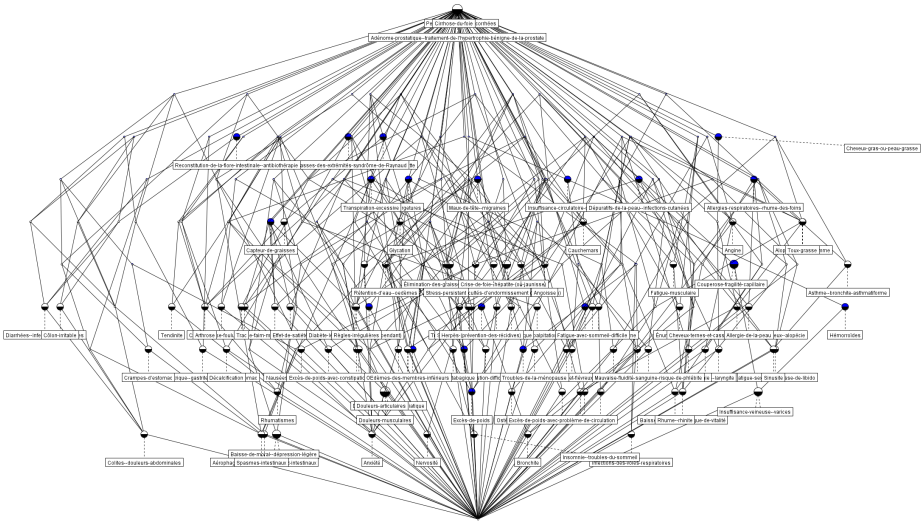


FIGURE 5.5 – Treillis de concepts obtenu sur l'exemple de phytothérapie (réduit)

L'étape qui suit nous a permis d'identifier certains sous-ensembles de maladies qui pourraient être regroupées. Chacun de ces sous-ensembles est en fait l'ensemble des objets d'un des co-clusters produits de type 1 à partir de ce contexte. Voici ci-après la liste des sous-ensembles de maladies identifiées pour constituer une taxonomie sur les objets. 14

groupes de maladies ont ainsi été identifiées. Précisons aussi que les 14 groupes constitués sont disjoints deux à deux.

Bloc 1	Bloc 2	Bloc 3
Douleurs articulaires	Anémie Ostéoporose	Problèmes hormonaux
Arthrite	Arthrose	Anémie, asthénie
Douleurs articulaires	Alopécie chez l'homme	Douleurs musculaires
Rhumatismes	Cheveux ternes et cassants, chute des cheveux, alopécie	Diabète léger (non insulino-dépendant)
Douleurs torticolis et sciatique	Ongles ternes et cassants	

Le contexte généralisé obtenu à partir de ces 14 groupes de maladies et 87 plantes a été formé comme suit. Nous avons attribué la valeur 1 à un de ces groupes pour une plante donnée s'il existe au sein de ce groupe au moins une maladie pour laquelle la valeur 1 a été attribuée pour cette plante. Autrement, c'est la valeur 0 qui est attribuée au groupe pour la plante considérée. Nous avons ainsi obtenu 39 concepts formels à partir de ce sous-contexte généralisé. La figure 5.6 illustre le treillis de Galois représentant ce sous-contexte. Un premier résultat déduit de cette figure est qu'aucune des maladies faisant parties des 14 groupes de maladies n'est traitable par aucune des plantes suivantes : ail, huile de saumon, huile de lin et lécithine de soja.

Aussi, sur la figure 5.7, on peut identifier clairement les motifs suivants :

1. Toutes les plantes qui soignent des troubles de la ménopause soignent également l'anémie et l'ostéoporose

2. Toutes les plantes qui soignent l'herpès et des maladies des voies respiratoires traitent également des troubles gastriques

3. Toutes les plantes qui soignent la cellulite et les œdèmes traitent également l'excès de poids

4. Les plantes suivantes : cyprès, fragon, hamamélis et Marronnier-d'inde ne soignent que les maladies associées à la mauvaise circulation sanguine causée par un excès de poids (cf. figure 5.8).

À l'exception du troisième motif, le trois autres n'auraient certainement pas ou très difficilement pu être identifiés sur le contexte réduit avant généralisation. Donc, l'extraction des quatre motifs mentionnés ci-dessus a été rendue possible grâce à la généralisation du contexte sur la phytothérapie, sans laquelle il serait difficile voire impossible d'extraire de tels motifs.

À l'étape suivante, nous avons voulu savoir s'il y avait des motifs pertinents qu'on pourrait extraire à partir d'un nouveau contexte où on se baserait sur la taxonomie créée et l'ensemble des autres maladies du contexte réduit ne faisant pas partie des maladies des 14 groupes. Ce nouveau contexte est alors une matrice binaire constituée de 67 objets dont chacun désigne soit une maladie ou un groupe de maladies et de 87 plantes du contexte réduit. Le treillis de Galois correspondant est illustré par la figure 5.9. Il y a 135 co-clusters de type 1 extraits de ce contexte généralisé au lieu des 222 de départ. On peut distinguer clairement sur le treillis de cette figure trois motifs dont le premier est très pertinent et non évident :

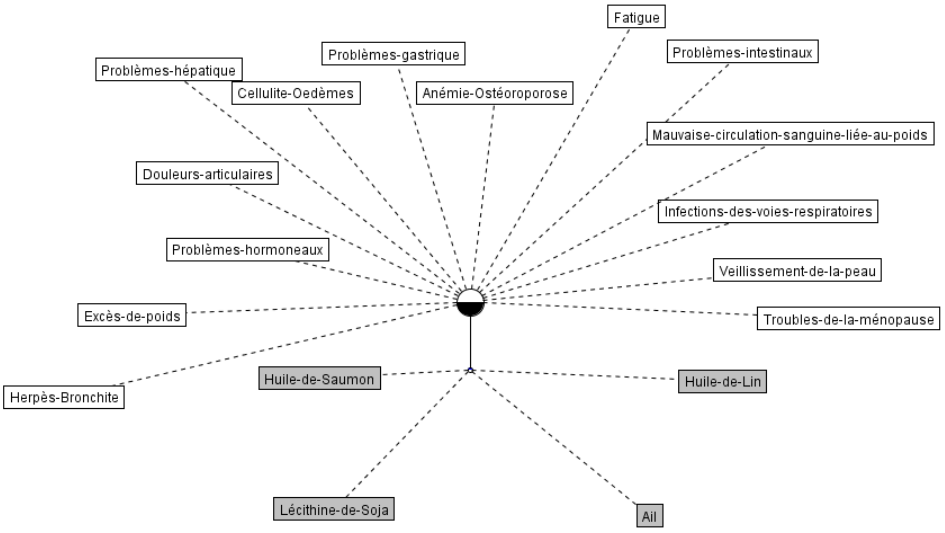


FIGURE 5.6 – Les quatre plantes qui ne traitent pas les 14 groupes de maladies.

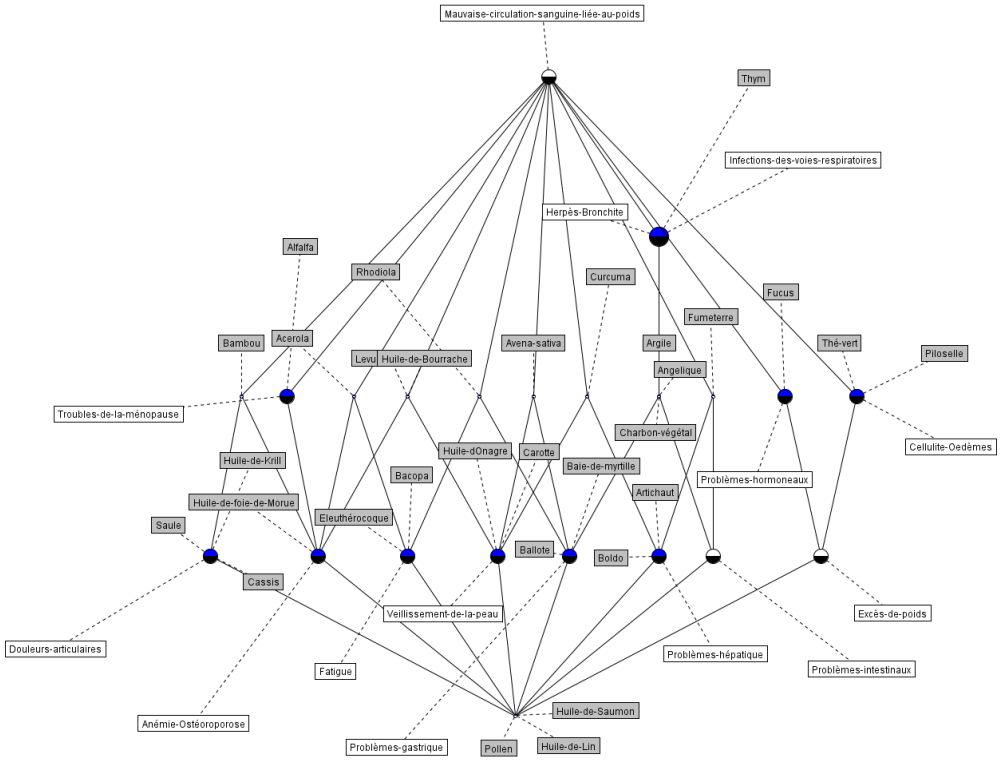


FIGURE 5.7 – Treillis du sous-contexte limité aux 14 groupes de maladies.

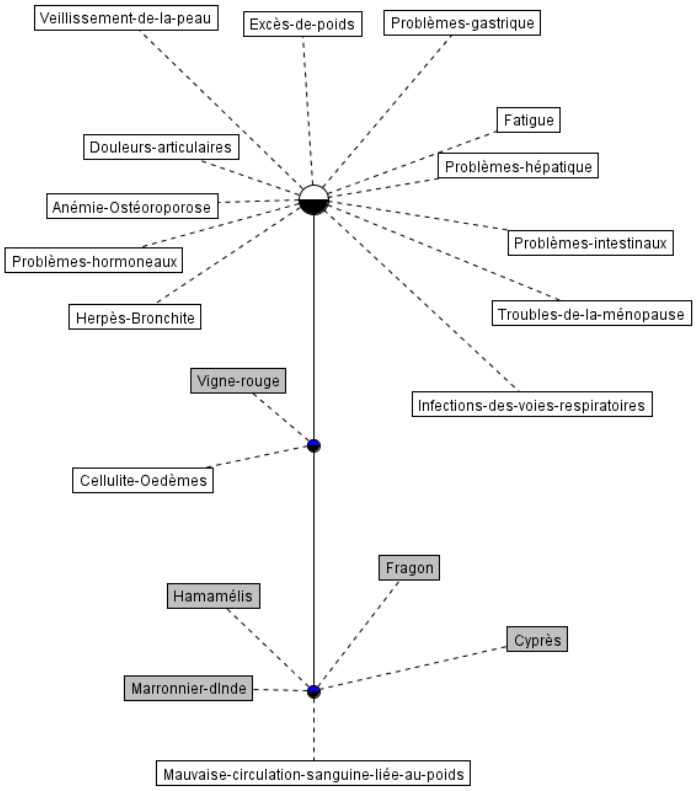


FIGURE 5.8 – Portion du treillis mettant en évidence le lien entre l'excès de poids, la cellulite/œdèmes et une mauvaise circulation sanguine.

1. Les plantes soignant un problème d'insuffisance circulatoire cérébrale traitent également des troubles de la mémoire

2. Les plantes soignant une angine traitent aussi les maux de gorge et la laryngite

3. Les plantes qui soignent l'asthme, une bronchite ou une allergie respiratoire servent aussi à soigner une sinusite.

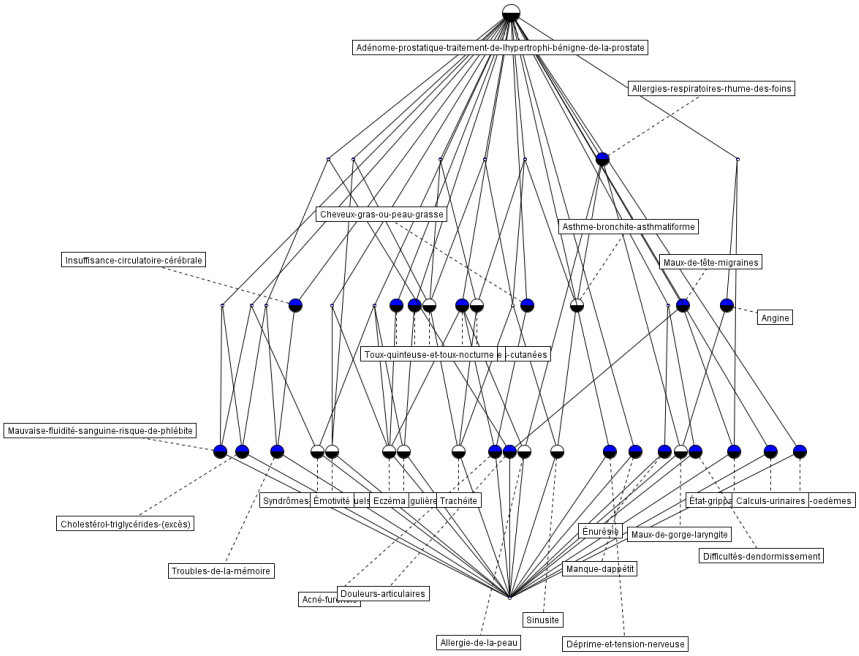


FIGURE 5.9 – Treillis mettant en évidence des liens entre les groupes de maladies

Pour finir, nous avons projeté le nouveau contexte sur d'autres groupes de maladies. La figure 5.10 met ainsi en évidence plusieurs autres motifs ayant un haut degré de pertinence. À titre d'exemple, on voit que toutes les plantes qui soignent un excès d'acide urique dans le sang (maladie de la goutte) traitent également les douleurs articulaires.

5.5.1 Analyse empirique de complexité

Les tests ont été menés avec du code en C++ sur un PC Intel 2.1GHz avec une mémoire RAM de 8 GO et le système d'exploitation Windows 10. Afin d'analyser la complexité temporelle de BiP en fonction du nombre d'attributs, de la densité et de la distribution des données au sein du contexte pour la production de co-clusters de type 1, nous avons généré synthétiquement $5 \times 3 \times 3$ contextes binaires en utilisant le système Coron [42]. La valeur 5 représente le nombre de fois qu'un même contexte de même taille et de même

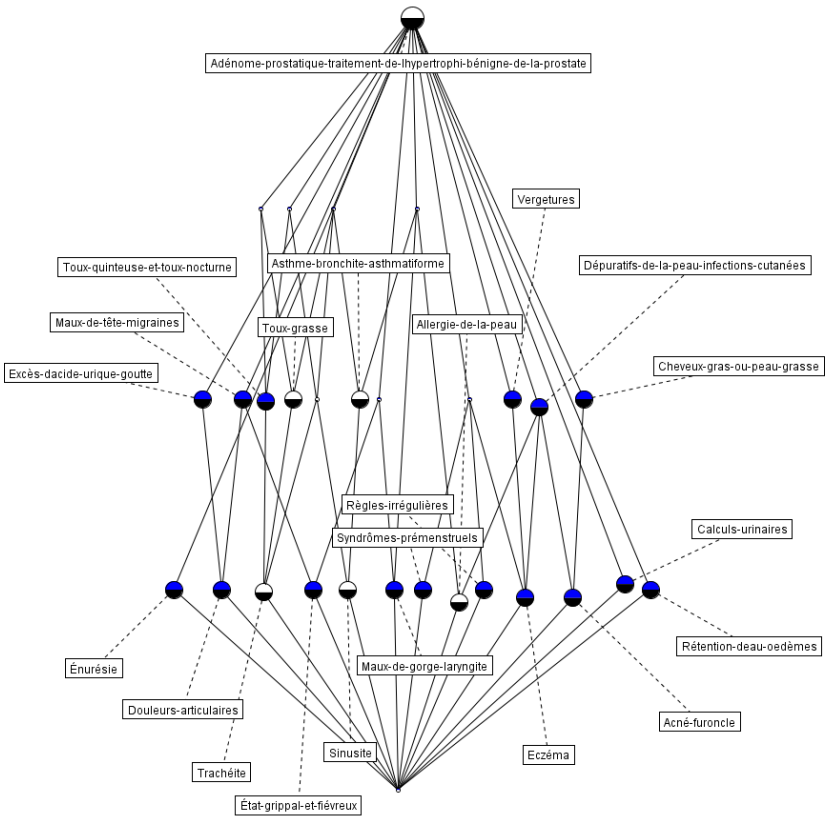
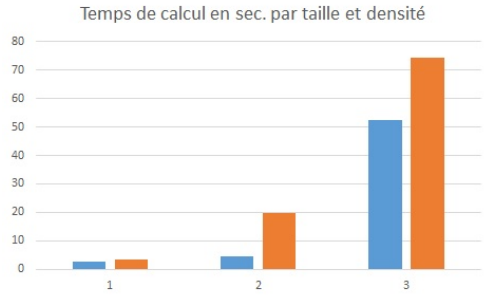


FIGURE 5.10 – Treillis mettant en évidence d'autres liens entre les groupes de maladies

densité est généré pour plus tard prendre la moyenne des temps d'exécution et tenir compte de diverses distributions des 1 dans les contextes. Pour chacune des trois tailles suivantes de matrices : (50x50), (100x100) et (150x150), nous avons considéré des densités de 10 et 20% du fait que les matrices d'entrée de plusieurs applications réelles sont assez éparses et leur densité dépasse rarement les 20%. L'histogramme suivant montre les résultats obtenus avec des temps de traitement relativement raisonnables qui augmentent avec un accroissement de la taille et de la densité de la matrice.



Taille et densité	10	20
50 x 50	2.6	3.39
100 x 100	4.6	19.94
150 x 150	52.6	74.44

FIGURE 5.11 – Temps d'exécution de BiP selon la taille et la densité

Le test suivant consiste à déterminer les temps d'exécution de BiP en fonction du nombre de colonnes du contexte. Ainsi, nous avons généré 50 contextes de 1000 lignes chacun et d'une même densité de 20%. On a fait varier le nombre d'attributs de 10, 20, 25, 30 puis 40. Ainsi, cinq groupes de 10 contextes de même taille chacun ont été constitués. Les résultats agrégés obtenus sont indiqués sur la table 5.7.

Attributs	Minimum	Moyenne	Maximum
10	0.201	0.2234	0.288
20	7.947	8.9653	9.43
25	26.443	27.8791	29.246
30	78.035	86.4405	94.028
40	367.789	389.6998	400.01

TABLE 5.7 – Temps d'exécution en secondes pour des matrices de 1000 objets

Les résultats des tests indiquent des temps d'exécution relativement faibles pour des contextes composés de 1000 objets et moins de 30 attributs. Cependant, à mesure que le nombre d'attributs augmente, le temps d'exécution augmente significativement.

5.6 Conclusion

Dans ce chapitre ainsi que dans le chapitre précédent, nous avons présenté une nouvelle méthode de co-clustering pouvant prendre en entrée une matrice binaire à partir de laquelle on peut générer des co-clusters de quatre types distincts. À travers les deux derniers cas

de cohabitation de 0 et de 1, nous désirons introduire aussi bien la présence que l'absence de propriétés au sein des co-clusters. Cela peut être particulièrement utile dans divers domaines d'application comme la bio-informatique, le marketing et l'analyse des réseaux sociaux pour mieux révéler la notion de groupes par rapport à l'absence ou la présence de certains attributs. Nous avons mis l'accent sur le troisième cas et montré qu'il est possible de mettre en évidence au sein d'un même co-cluster des sous-groupes d'objets ou d'individus totalement ou partiellement identiques, ou même opposés vis-à-vis d'un ensemble donné de propriétés. Pour le type 4, nous voulions illustrer le fait que des objets sont similaires non seulement par rapport à la présence mais également à l'absence de propriétés. Nous avons montré aussi à travers l'exemple de la phytothérapie les différentes étapes permettant de créer une taxonomie sur un contexte lorsque cette dernière est inexistante ou n'est pas évidente. Nous avons montré l'intérêt d'employer une telle démarche dont la production de motifs pertinents. Plusieurs exemples de motifs ont été produits à partir du contexte sur les données de la phytothérapie.

L'application de notre approche à l'exemple très connu de Davis pour la production de co-clusters du 3ème type révèle l'existence de sous-groupes de femmes, au sein d'un même co-cluster, qui fréquentent totalement, partiellement ou aucunement un ensemble d'événements.

Nos travaux futurs visent à définir un ensemble d'opérations d'interrogation des co-clusters produits par le parcours de la structure d'arbre Patricia et correspondant à des besoins et requêtes spécifiques de l'utilisateur. À titre d'exemple, l'utilisateur peut demander à trouver un co-cluster dont l'extension comporte au moins (au plus, ou exactement) un ensemble donné d'objets afin d'en connaître ses sous-groupes et les propriétés qui les définissent. En outre, si la matrice représentant le contexte a une densité faible des 1, le nombre de co-clusters issus des types 2, 3 et 4 peut être très grand comparativement à l'effectif des co-clusters de type 1. Nous envisageons alors de produire les co-clusters à la volée à la suite de toute requête explicite de l'utilisateur portant sur un sous-ensemble précis d'objets et/ou d'attributs par projection et sélection sur la matrice initiale. La structure d'arbre Patricia combinée avec la présence d'un index sur les attributs faciliterait la production rapide d'une réponse.

Chapitre 6

Coclusters avec des données discrètes ou catégorielles

Sommaire

6.1 Introduction

Nous avons présenté et discuté dans les chapitres 4 et 5 quatre versions de BiP qui génèrent des co-clusters de quatre types distincts à partir d'une matrice à valeurs binaires. Dans ce chapitre, on généralise la procédure de production de co-clusters de type 1 au cas de données discrètes ou catégorielles. L'intérêt de cette version réside dans le fait qu'aucune transformation préalable des données multivaluées du contexte initial n'est requise. Donc, la dimensionnalité du contexte reste inchangée et aucune perte d'information n'est constatée. Cependant, lorsque les données sont continues, une transformation préalable des données est nécessaire afin de les amener dans un format de données directement exploitable par BiP : binaire, discret ou catégoriel.

Pour valider notre approche, deux contextes issus du monde réel sont utilisés. Le premier emploie des données discrètes et le second des données catégorielles. Une discussion concernant la sémantique et la pertinence des co-clusters produits est fournie.

6.2 Définition d'un co-cluster à valeurs discrètes ou catégorielles

Partant d'un contexte multi-valué $\mathbb{K} = (G, M, V, I)$ où $G = \{o_1, o_2, ...o_n\}$ est un ensemble d'objets représentant les données, $M = \{m_1, m_2, ...m_m\}$ les propriétés de ces objets et $I \subseteq G \times M \times V$. L'expression $(o, m, v) \in I$ signifie que l'objet o a la valeur v pour l'attribut m. Un co-cluster est alors une paire (X, Y) avec $X \subseteq G$ ($|X| \geq 1$) et $Y \subseteq M$ ($|Y| > 1$) vérifiant les deux conditions suivantes :

1. Pour tout attribut $m_t \in Y$, il existe au moins un attribut $m_l \neq m_t \in Y$ tel que $\forall o \in X\ (m_t(o) = m_l(o) = v)$.

2. (X, Y) est maximal. On dira que (X, Y) est maximal pour l'inclusion, lorsque : $X \times Y \subset I$ et $\forall X_1 \supset X$, $\forall Y_1 \supset Y$, $(X_1 \times Y_1 \subset I$, (X_1, Y_1) a toutes ses valeurs égales à v et vérifie la condition $(X = X_1, Y = Y_1)$.

Notons qu'un co-cluster de type 1 est un cas particulier du type de co-cluster défini ci-dessus avec $V = \{1\}$.

6.3 Illustration du cas de données discrètes

Pour montrer la capacité de BiP à produire des co-clusters à partir d'un contexte dont les données sont discrètes sans avoir à les transformer en binaire, nous avons choisi l'exemple réel tiré de *UCI Machine Learning Repository* [52]. Cet exemple porte sur l'utilisation d'un moyen contraceptif par un échantillon d'une population Indonésienne supposée non enceinte au moment de la conduite de cette étude.

L'étude a été menée sur un sous-ensemble des données extraites de l'enquête nationale de prévalence contraceptive conduite en 1987 en Indonésie. L'échantillon est composé de femmes mariées qui n'étaient pas enceintes ou ne savaient pas si elles l'étaient au moment de l'entrevue. Le but de cette étude est de prédire la méthode contraceptive actuelle (aucune méthode, méthodes à long terme, ou méthodes à court terme) d'une femme sur la base de ses caractéristiques démographiques et socio-économiques.

L'enquête a été menée auprès de 1473 femmes pour lesquelles les informations suivantes ont été requises : l'âge de la femme, son niveau d'éducation ainsi que celui du mari, le nombre d'enfants nés, la religion de la femme, la détention d'un travail, l'occupation du mari, le standard de vie, l'exposition aux médias, et finalement le moyen contraceptif utilisé. Des codes ont été utilisés pour exprimer les réponses (cf. annexe).

Vu le nombre important de co-clusters générés, nous avons choisi de visualiser uniquement deux des co-clusters créés desquels on peut prédire aisément la méthode de contraception utilisée au moment de l'enquête.

Le co-cluster de gauche met clairement en évidence le fait que 9 femmes sur 12 qui le constituent ont plus de 39 ans et ont un nombre assez élevé d'enfants (supérieur à 5 dans $66,6\%$ des cas). Le niveau d'éducation de chacune de ces femmes et de leur mari est bas. Le couple est de confession musulmane et a une mauvaise exposition aux médias. On constate aussi que 10 fois sur 12, ces femmes n'utilisent pas de moyen contraceptif.

Le co-cluster de droite nous permet de tirer les conclusions suivantes : toutes les femmes ont moins de 30 ans, le couple a un bon niveau d'éducation, le nombre d'enfants est de trois par femme à part un seul cas, ce qui pourrait peut-être expliquer la raison pour laquelle toutes ces femmes ont eu recours récemment à un moyen de contraception à l'exception de la première qui est âgée seulement d'une vingtaine d'années.

1	2	3	4	5	6	7	8	9	10
45	**1**	**1**	8	**1**	1	2	2	**1**	1
47	**1**	**1**	7	**1**	1	2	3	**1**	1
23	**1**	**1**	3	**1**	1	2	1	**1**	1
44	**1**	**1**	1	**1**	1	4	1	**1**	1
39	**1**	**1**	5	**1**	0	3	1	**1**	1
48	**1**	**1**	8	**1**	0	2	2	**1**	1
47	**1**	**1**	8	**1**	1	3	4	**1**	2
31	**1**	**1**	3	**1**	1	4	2	**1**	3
48	**1**	**1**	1	**1**	0	3	3	**1**	1
48	**1**	**1**	6	**1**	1	2	3	**1**	1
33	**1**	**1**	5	**1**	1	3	1	**1**	1
48	**1**	**1**	8	**1**	1	1	1	**1**	1

1	2	3	4	5	6	7	8	9	10
20	**2**	2	**2**	1	1	**2**	**2**	1	1
33	**3**	3	**3**	1	1	**3**	**3**	0	3
26	**3**	3	**3**	1	1	**3**	**3**	0	3
25	**3**	3	**3**	1	1	**3**	**3**	0	3
26	**3**	4	**3**	0	1	**3**	**3**	0	3
30	**3**	4	**3**	1	1	**3**	**3**	0	3
30	**3**	2	**3**	1	1	**3**	**3**	0	3
30	**3**	4	**3**	1	1	**3**	**3**	0	3
22	**3**	3	**3**	1	1	**3**	**3**	0	3

TABLE 6.1 – Exemple de co-clusters générés par BiP sur l'ensemble de données de la contraception. **(Cf. l'annexe pour la signification des codes)**

6.4 Illustration du cas de données catégorielles

Nous utilisons la base de données sur les champignons [51] pour illustrer la capacité de BiP à produire des co-clusters à partir de données de type catégoriel sans recours à une transformation préalable de ces données en binaire. Cette base de données est composée de 8124 champignons définis par 23 propriétés dont l'une d'elles est la classe d'appartenance du champignon testé (e : comestible et p : non comestible). À la différence de la majorité des algorithmes de co-clustering, nous n'avons pas eu besoin de transformer les données de cette base de données en binaire avant de pouvoir créer les co-clusters. Rappelons qu'une telle transformation aurait abouti à une matrice avec une dimension beaucoup plus importante que celle d'origine (123 au lieu de 23 attributs).

Comme on pouvait s'y attendre, plusieurs co-clusters ont été produits à partir de cet exemple. Nous avons choisi de discuter quelques-uns d'entre eux qui appartiennent à l'une ou l'autre des deux classes e et p. À partir des résultats obtenus, nous pouvons émettre les observations suivantes :

1. Les champignons dont l'attribut couleur-calotte est e (rouge) et l'attribut couleur-tige-au-dessus-de-la-bague est aussi égal à e (rouge) sont de type comestible.

2. Les champignons sans odeur (l'attribut odeur est n) et l'attribut couleur-tige-sous-la-bague est w (blanc) sont de type comestible.

3. Les champignons dont l'attribut surface-calotte est égal à f (fibreux) et l'attribut Bleus (*bruises* ou ecchymoses) vaut f (non) et l'attribut couleur-branlette a la valeur h (couleur chocolat) et finalement l'attribut type-bague a pour valeur l (grand) sont de type vénéneux.

4. Les champignons dont l'attribut surface-calotte a la valeur s (lisse) et dont l'attribut forme-tige a la valeur e (élargi) et finalement l'attribut racine-tige est aussi égal à e (égal) sont de type vénéneux.

Sur la base de ces résultats, nous constatons que notre approche permet d'identifier des groupes de champignons de type comestible ou non en considérant uniquement un nombre très réduit de leurs propriétés. Dans le premier cas par exemple, on a besoin uniquement de deux propriétés : la couleur de la calotte et la couleur de la tige au-dessus de la bague parmi les 22 attributs de départ (autres que l'attribut de classification) pour identifier la classe d'appartenance des champignons. Ainsi donc, notre procédure de co-clustering peut s'avérer intéressante même dans une tâche d'apprentissage supervisé.

6.5 Conclusion

Dans le présent chapitre, nous avons montré que la procédure BiP peut être utilisée en généralisant la variante de type 1 pour produire des co-clusters de données discrètes ou catégorielles sans avoir à transformer les données en binaire. Ainsi, la dimensionnalité du contexte reste inchangée et aucune perte d'information n'est produite causée par cette transformation.

Chapitre 7

Gestion de motifs

Sommaire

7.1 Introduction

La grande disponibilité à moindre coût d'importants volumes de données, combinée avec la nécessité d'extraire des informations concises pour être manipulées et analysées efficacement, posent de nouvelles exigences pour les systèmes de gestion de données dans les applications industrielles et scientifiques. De toute évidence, les données massives et brutes ne constituent pas une information ou une connaissance en soi. Il en résulte que peu d'informations utiles peuvent être extraites simplement par leur observation. Ainsi, des techniques sont nécessaires pour la découverte des connaissances cachées à partir de ces données.

Tel qu'indiqué auparavant, la découverte et la gestion de motifs se réfère à un ensemble d'activités liées à l'extraction, la description, la manipulation et le stockage de motifs. Rappelons que dans la gestion des motifs, les instances sont des artefacts de connaissances (des règles d'association, des clusters, des co-clusters) extraits par des procédures de fouille de données et récupérés à la demande de l'utilisateur.

Dans de nombreuses applications, les utilisateurs tendent à être noyés non seulement par les données massives mais également par un nombre impressionnant de motifs alors qu'ils sont réellement intéressés par un ensemble très limité de connaissances. De plus, les besoins en motifs pertinents diffèrent d'un utilisateur à l'autre et évoluent avec le temps.

Enfin, la fouille de données devrait être davantage un processus exploratoire et itératif de découverte de motifs sous différents scénarios et différentes hypothèses selon les propres besoins de chaque utilisateur.

7.2 État de l'art

Dans la littérature, des études récentes sur la gestion des motifs [46, 5, 13, 57] fournissent un cadre uniforme à la gestion des données et des motifs et définissent des liens entre des données et les motifs par des opérations de mise en correspondance et de requêtes croisées telles que la recherche de données couvertes par un motif donné ou l'identification de motifs liés à un ensemble de données. Bien que de nombreuses études limitent la gestion des motifs aux règles d'association seulement, les travaux menés par [12, 72] couvrent différents types de motifs.

Dans [71], un système de gestion d'une base de motifs *(BPMS)* est utilisé pour stocker, traiter et interroger des motifs. De plus, des langages pour la définition et la manipulation des motifs sont proposés, et les aspects temporels des motifs sont traités. Dans [13], une algèbre de fouille de données et un modèle dit *3-World* sont définis, ainsi qu'un ensemble réduit d'opérateurs de fouille de données primitives sont proposés pour formuler des requêtes complexes. Le modèle proposé comprend trois composantes : *D-World* pour la définition et la manipulation des données via des opérations algébriques comme la projection et la jointure, *I-World* pour la définition et la manipulation de la région (ensemble de contraintes) et *E-World* pour les opérations sur les données contenues dans les régions. Une librairie de modèles de fouille de données est définie en fonction d'une approche générique d'exploration de données pour contrôler certains aspects de cette exploration à l'aide d'un ensemble de propriétés. Du côté de l'industrie, les travaux ont principalement porté sur la conception de langages de description, de manipulation et d'échange de motifs. C'est le cas de PMML et de SQL/MM [10].

Une autre recherche [57] a permis de développer deux nouveaux opérateurs de manipulation de motifs. Le premier de ces opérateurs permet de sélectionner un sous-ensemble de motifs à partir de l'ensemble de tous les motifs produits en imposant certaines conditions aux motifs extraits. Le second permet de sélectionner les motifs sur la base d'un sous-ensemble d'attributs pris parmi l'ensemble de tous les attributs existants. En fait, les deux opérateurs proposés ne sont rien d'autre que les opérations de sélection et de projection utilisées dans les bases de données relationnelles. Ils sont définis comme suit :

Partant d'un contexte $\mathbb{K} := (G, M, I)$ où G, M et I représentent respectivement un ensemble d'objets, un ensemble d'attributs, et une relation binaire entre les éléments de G et ceux de M avec r un ensemble de concepts formels et $N \subseteq M$. La projection de l'ensemble des concepts r sur l'ensemble N est définie par : $\Pi_N(r) = Project(r, N) = \{c_1 = (Ext(c), Int(c) \cap N) \text{ — } c \in r$ et c_1 est maximal dans sa classe d'équivalence. On dit que deux concepts c_1 et c_2 sont équivalents si $Int(c_1) \cap N = Int(c_2) \cap N$.

La figure 7.2 illustre la projection sur $\{S, T, U, V\}$ du treillis de la figure 7.1. Sur la gauche de cette figure, nous pouvons voir des classes d'équivalence marquées sur le treillis

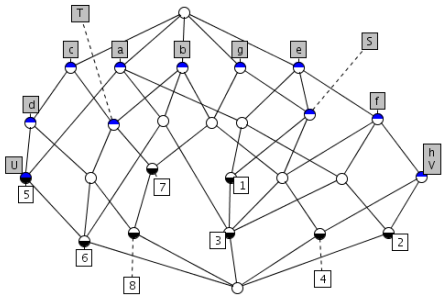


FIGURE 7.1 – treillis d'un contexte $\mathbb{K}$

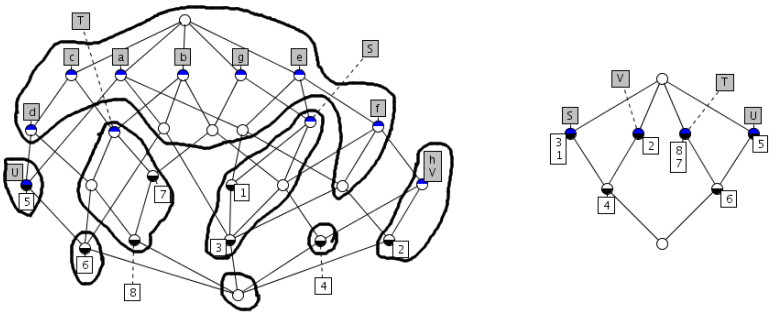


FIGURE 7.2 – Projection sur $\{S, T, UV\}$ du contexte associé au treillis de la figure : 7.1

$\mathbb{K}$	a	b	c	d	e	f	g	h	S	T	U	V
1	x				x		x		x		x	
2	x				x	x		x				x
3	x	x			x	x	x		x			
4		x			x	x	x	x	x			x
5	x		x	x							x	
6	x	x	x	x						x	x	
7		x	x				x			x		
8		x	x	x			x			x		

TABLE 7.1 – $\mathbb{K}$: Contexte avec attributs de projection

initial. Sur la droite, nous notons que chaque classe d'équivalence est représentée par un seul nœud auquel est attachée une classe d'équivalence.

Quant à l'opérateur de sélection, il est défini d'une manière duale comme suit : étant donné un ensemble r de concepts, on peut sélectionner des concepts parmi ceux de r en se limitant à un sous-ensemble d'attributs $F = \{A_1 \wedge ... \wedge A_n\}$ extraits de l'ensemble des attributs initiaux par l'opérateur : $Select(r, F = \{A_1 \wedge ... \wedge A_n\}) = \{c$ tel $c \in r$ et $c| = F$.

À moins d'effectuer une apposition du contexte initial avec son complémentaire, les deux opérateurs de sélection et projection ne permettent nullement de sélectionner les objets du contexte $\mathbb{K}$ qui possèdent certains attributs et n'ont pas explicitement d'autres propriétés. En outre, la taille du treillis initial peut être très volumineuse rendant illisible même le résultat de la projection sur des attributs ou la sélection d'objets. Par ailleurs, l'utilisateur pourrait être intéressé à poser des questions du genre : existe t-il des concepts formels possédant exactement ou approximativement [56] un ensemble d'objets et/ou un ensemble d'attributs ? Quels sont les successeurs ou prédécesseurs immédiats d'un concept donné ?

En analyse formelle de concepts, la notion d'approximation de concepts coïncide avec celle d'un pré-concept d'un contexte formel $\mathbb{K} := (G, M, I)$. Ce dernier est un couple $c := (A, B)$ tel que $A \subseteq G$ et $B \subseteq M$ et $A \subseteq B'$ ($\Leftrightarrow A' \supseteq B$). En fait, $(A'', A) \leq c \leq (B', B'')$.

Dans ce qui suit, nous proposons une démarche pour sélectionner des motifs de type co-clusters en faisant appel à la structure de l'arbre Patricia *(radix-tree)* pour leur stockage et gestion. Précisons que cette structure nous permet de remplacer les concepts ne répondant pas à la requête de l'usager du système d'information par un ou plusieurs pré-concepts permettant ainsi de répondre approximativement à la demande formulée par l'usager.

Nous avons vu précédemment que la pertinence d'un co-cluster est fonction non seulement de l'information ou la connaissance que celui-ci véhicule mais aussi et surtout de l'importance de celle-ci pour l'utilisateur. Donc, il serait nécessaire de connaître les objets et/ou les attributs par lesquels l'utilisateur est intéressé et qu'il aimerait voir figurer dans les co-clusters produits. Cependant, si l'utilisateur part à la découverte des données et de ce qu'ils peuvent contenir comme information, le système pourrait chercher et proposer les combinaisons de paires d'attributs les plus pertinentes dans le sens de celles qui mettent

en relation un plus grand nombre d'objets à la fois. Cette approche pourrait être appliquée spécifiquement aux co-clusters de types 3 et 4 qui sont généralement plus nombreux que ceux des types 1 et 2 à cause de la prise en compte à la fois de la présence que de l'absence de propriétés. Aussi, plusieurs des ces co-clusters seront chevauchants. Par conséquent, en se limitant juste aux attributs mettant en jeu plusieurs objets à la fois, on va pouvoir limiter le nombre de co-clusters, réduire le temps nécessaire à leurs création, diminuer les chevauchements et augmenter les chances d'obtenir des co-clusters pertinents. Aussi, l'intérêt d'utiliser cette approche plutôt que celle où on construit le treillis des concepts pour ensuite le projeter sur un ensemble d'attributs réside dans le fait de commencer par le choix des attributs. Cette approche va générer des treillis de taille réduite montrant uniquement les objets qui intéressent l'utilisateur. C'est cette approche que nous allons employer dans la suite de ce chapitre.

Pour pouvoir calculer le nombre d'objets partagés par une paire d'attributs ou le nombre d'attributs partagés par deux d'objets dans un contexte binaire, une méthode consiste à utiliser la matrice représentant ce contexte et sa transposée. À titre d'exemple, si $\mathbb{R}$ est un contexte binaire et $\mathbb{R}^t$ sa transposée (voir annexe), le produit $\mathbb{R}^t * \mathbb{R}$ nous donne le nombre d'objets partagés par un couple d'attributs et $\mathbb{R} * \mathbb{R}^t$ le nombre d'attributs partageant une paire d'objets. L'usager du système peut alors sur la base de l'information contenue dans ces deux produits matriciels choisir la (les) paire(s) d'attributs ou d'objets qui partagent respectivement un nombre minimal d'objets versus d'attributs qu'il veut voir apparaître dans les co-clusters créés. À titre d'exemple, le chiffre 9 à l'intersection de la ligne m_2 et de la colonne m_4 représente le nombre d'objets du contexte $\mathbb{R}$ que ces deux attributs partagent (voir annexe).

7.3 Illustration

Tel qu'on a vu dans les précédents chapitres sur le co-clustering, la première étape des différentes versions de l'algorithme BiP emploie la procédure *commun* pour générer tous les couples (X,Y) où $Y = \{m_i, m_j\}$ avec $m_i, m_j \in M$ et X l'ensemble des objets de G possédant la même valeur pour les attributs m_i et m_j. Ces couples servent par la suite à former les co-clusters. Toutefois, les besoins de l'usager peuvent se limiter à un type particulier de co-cluters. Il peut vouloir extraire uniquement les concepts formels (type 1), des co-clusters à valeurs nulles (type 2) ou mixtes (types 3 et 4).

7.3.1 Concepts formels

Pour ce premier cas, nous allons supposer que l'usager est intéressé uniquement par les paires d'attributs dont le nombre d'objets partagés est supérieur ou égal à 9. Il faudrait bien sûr considérer tous les couples d'attributs dont le nombre d'objets partagés est supérieur ou égal à 1 pour obtenir l'ensemble des concepts formels. Ainsi, les paires d'attributs du contexte $\mathbb{R}$ qui partagent au minimum 9 objets sont : (m_2, m_4) - (m_2, m_8) - (m_4, m_8). On peut alors déterminer les éléments qui vont constituer l'ensemble $\mathbb{C}$.

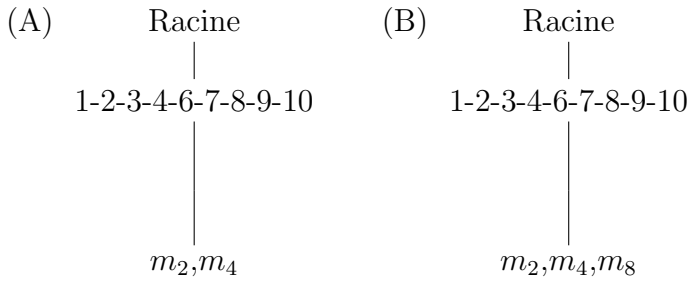


FIGURE 7.3 – Arbre Patricia $\mathbb{B}$ - Cas des concepts formels

$$X_1 = \{o_1, o_2, o_3, o_4, o_6.o_7, o_8, o_9, o_{10}\} \text{ et } Y_1 = \{m_2, m_4\}$$
$$X_2 = \{o_1, o_2, o_3, o_4, o_6.o_7, o_8, o_9, o_{10}\} \text{ et } Y_2 = \{m_2, m_8\}$$
$$X_3 = \{o_1, o_2, o_3, o_4, o_6.o_7, o_8, o_9, o_{10}\} \text{ et } Y_3 = \{m_4, m_8\}$$

$\mathbb{B}$	m_2	m_4	m_8
o_1	1	1	1
o_2	1	1	1
o_3	1	1	1
o_4	1	1	1
o_6	1	1	1
o_7	1	1	1
o_8	1	1	1
o_9	1	1	1
o_{10}	1	1	1

TABLE 7.2 – Concept formel obtenu à partir des attributs m_2, m_4, et m_8

La figure 7.3 (A) illustre l'état de l'arbre Patricia $\mathbb{B}$ après l'insertion du couple (X_1, Y_1). L'étape suivante consiste à comparer X_1 avec X_2. Comme ces deux ensembles d'objets sont égaux, on applique le cas 3 de la procédure *Extraire* (cf. algorithme 3 de la section 4.3. La structure de l'arbre Patricia devient alors La fig. 7.3 (B). On poursuit les comparaisons du contenu de l'arbre avec le dernier couple (X_3, Y_3). Comme il y a égalité encore avec l'ensemble des objets du seul nœud existant de $\mathbb{B}$ et que l'union de Y_3 avec le contenu de ce nœud n'apporte aucune modification à ce contenu, il s'ensuit que la structure de l'arbre reste inchangée. Ainsi un seul concept formel va être formé avec l'intention $\{m_2, m_4, m_8\}$ tel qu'indiqué sur la table 7.2.

7.3.2 Co-clusters pleins de zéros

Penchons-nous maintenant sur le deuxième cas, c.-à-d. celui où l'usager du système d'information désire extraire des co-clusters composés uniquement de valeurs nulles (des zéros). En

d'autre termes, l'usager désire extraire du contexte $\mathbb{R}$ des blocs composés d'une liste d'objets ne possédant pas un ensemble d'attributs. Ce genre de motifs peut servir à répondre à des questions du genre : quels sont les clients d'une grande surface qui n'achètent pas un ensemble de produits ou en médecine quels sont les patients qui ne répondent pas à une médication donnée. Nous illustrons ce cas sur le contexte $\mathbb{R}$. Aussi, comme cela a été indiqué ci-dessus, nous devons calculer l'apposition du contexte $\mathbb{R}$ ($\overline{\mathbb{R}}$) et le produit $\overline{\mathbb{R}^t} * \overline{\mathbb{R}}$. Comme nous pouvons le voir sur la matrice produite, plusieurs couples peuvent être formés. Pour des besoins d'illustration, on suppose que l'usager du système est intéressé uniquement par les co-clusters de type 2 contenant l'attribut m_1 et/ou m_6. À partir du produit $\overline{\mathbb{R}^t} * \overline{\mathbb{R}}$, ils sont au nombre de quatre couples pour m_1 et trois pour m_6.

$$X_1 = \{o_1, o_2, o_3, o_4\} \text{ et } Y_1 = \{m_1, m_3\} \,;\, X_2 = \{o_1, o_2, o_3, o_4\} \text{ et } Y_2 = \{m_1, m_5\}$$
$$X_3 = \{o_1, o_2, o_3, o_4\} \text{ et } Y_3 = \{m_1, m_6\} \,;\, X_4 = \{o_1, o_2, o_3, o_4\} \text{ et } Y_4 = \{m_1, m_9\}$$
$$X_5 = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_5 = \{m_6, m_7\}$$
$$X_6 = \{o_1, o_2, o_3, o_4, o_5\} \text{ et } Y_6 = \{m_6, m_9\}$$
$$X_7 = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_7 = \{m_6, m_{10}\}$$

La première étape consiste à créer l'arbre $\mathbb{B}$ et à placer le couple (X_1, Y_1). On obtient alors un arbre dont la structure est illustrée sur la figure 7.4 (C). L'étape qui suit consiste à comparer X_2 à X_1. Ces deux ensembles sont égaux et donc on doit appliquer le cas 3 de la procédure *Extraire*. Ainsi, aucune nouvelle branche ne sera créée au sein de l'arbre $\mathbb{B}$ excepté le fait qu'on doit ajouter au nœud de l'arbre l'attribut m_5. Aussi, remarquons que X_3 et X_4 sont aussi des ensembles égaux à X_1. Donc, nous devons là aussi appliquer le cas 3 de la procédure *Extraire*. Le résultat final de toutes ces comparaisons nous amène à un arbre $\mathbb{B}$ dont la structure est donnée par la figure 7.4 (D). Le couple suivant à comparer au contenu de l'arbre $\mathbb{B}$ est (X_5, Y_5). Comme nous pouvons le voir sur la procédure *Extraire*, on doit appliquer le cas 4. Ainsi une nouvelle branche va être créée au sein de $\mathbb{B}$ figure 7.4 (E).

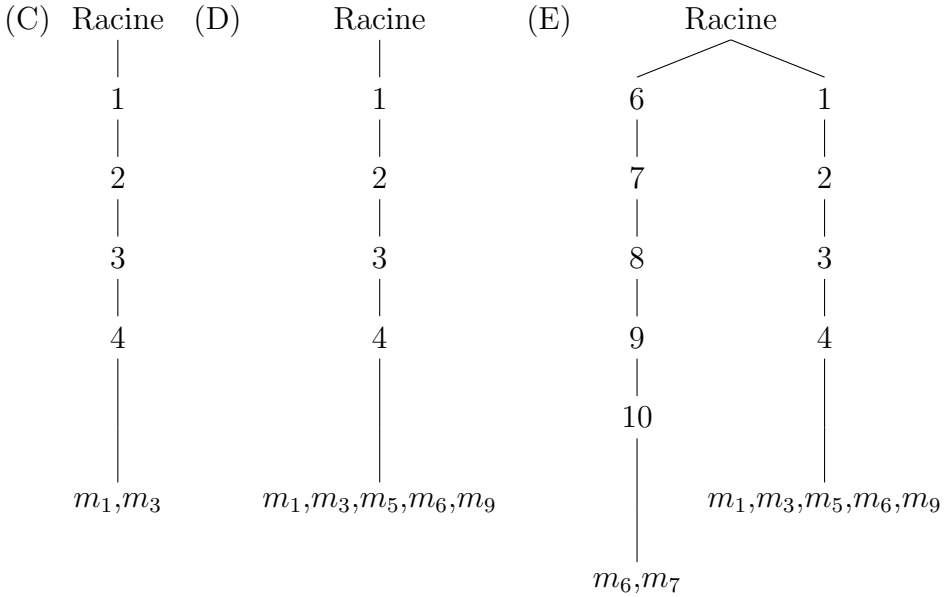


FIGURE 7.4 – Arbre Patricia $\mathbb{B}$ - Cas des co-clusters à valeurs nulles

Nous devons comparer (X_6, Y_6) avec le contenu de l'arbre $\mathbb{B}$. On est alors dans le cas 1 de la procédure *Extraire*. Toutefois, la feuille correspondant à la clé $o_1o_2o_3o_4$ contient déjà les attributs m_6, m_9. Donc, aucune modification ne sera apportée à cette branche de l'arbre. On doit cependant étendre cette branche en ajoutant l'arête o_5 et en créant un nouveau nœud avec les attributs m_6, m_9. On aboutit alors à un arbre $\mathbb{B}$ dont la structure est illustrée par la figure 7.5.

$\mathbb{P}$	m_1	m_3	m_5	m_6	m_9
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

$\mathbb{H}$	m_6	m_7
6	0	0
7	0	0
8	0	0
9	0	0
10	0	0

$\mathbb{E}$	m_6	m_9
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0

TABLE 7.3 – les trois co-clusters à valeurs nulles

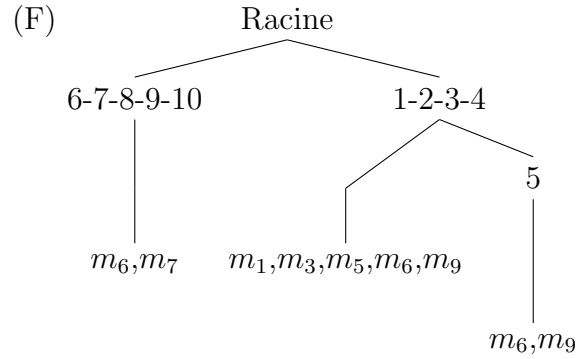


FIGURE 7.5 – Arbre Patricia $\mathbb{B}$ - Cas des co-clusters à valeurs nulles

La table 7.3 montre les trois co-clusters à valeurs nulles correspondant aux couples générés par les deux attributs m_1 et m_6. Ces co-clusters ont été également illustrés sur l'arbre Patrica par la figure 7.5.

7.3.3 Co-clusters mixtes

Pour les co-clusters mixtes, nous donnons l'illustration seulement pour le type 3. Le type 4 diffère du type 3 uniquement par la façon dont les attributs négatifs sont pris en charge. De plus, l'illustration pour le cas des co-clusters type 4 a été donnée dans le chapitre 5. Dans ce cas, nous devons nous servir de la matrice $\mathbb{X}$ afin de choisir quelles paires d'attributs devraient être considérées pour produire de tels co-clusters. À partir de la matrice $\mathbb{X} = \overline{\mathbb{R}^t} * \overline{\mathbb{R}} + \mathbb{R}^t * \mathbb{R}$, nous allons extraire, comme dans le cas précédent, les couples contenant respectivement les attributs m_1 et m_6. Ceux-ci sont au nombre de dix. Ils génèrent cinq co-clusters mixtes tel qu'indiqué sur l'arbre de figure 7.6.

$$X_1 = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_1 = \{m_1, m_2\}$$
$$X_2 = \{o_1, .., o_{10}\} \text{ et } Y_2 = \{m_1, m_3\}$$
$$X_3 = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_3 = \{m_1, m_4\}$$
$$X_4 = \{o_1, .., o_{10}\} \text{ et } Y_4 = \{m_1, m_5\}$$
$$X_5 = \{o_1, o_2, o_3, o_4\} \text{ et } Y_5 = \{m_1, m_6\}$$
$$X_6 = \{o_5, o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_6 = \{m_1, m_8\}$$
$$X_7 = \{o_1, o_2, o_3, o_4, o_6, .., o_{10}\} \text{ et } Y_7 = \{m_1, m_9\}$$
$$X_8 = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_8 = \{m_6, m_7\}$$
$$X_9 = \{o_1, o_2, o_3, o_4, o_5\} \text{ et } Y_9 = \{m_6, m_9\}$$
$$X_{10} = \{o_6, o_7, o_8, o_9, o_{10}\} \text{ et } Y_{10} = \{m_6, m_{10}\}$$

La même démarche que celle utilisée précédemment pour produire des co-clusters à valeurs nulles devrait être employée pour créer des co-clusters mixtes. Ainsi, dans le cas de notre exemple et en se limitant aux dix couples $(X_1, Y_1), ..(X_{10}, Y_{10})$, on aboutit à la structure fournie par la figure 7.6.

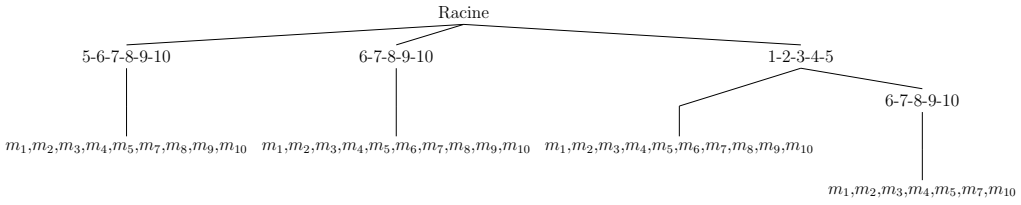


FIGURE 7.6 – Arbre Patricia $\mathbb{B}$ - Cas des co-clusters mixtes (type 3)

7.4 Interrogation des motifs

Dans cette section, nous allons nous concentrer sur la façon dont nous pouvons parcourir la structure d'arbre Patricia pour retrouver des motifs. L'exemple Davis (cf. le tableau 8.1 de l'annexe) va servir d'illustration.

	m1	m2	m3	m4	m5	m6	m7	m8	m9	m10
O1	1	0	1	0	1	1	0	0	1	0
O2	1	0	1	0	1	1	0	0	1	0
O3	1	0	1	0	1	1	0	0	1	0
O4	1	0	1	0	1	1	0	0	1	0
O5	0	1	0	1	0	1	0	0	1	0
O6	0	0	0	0	0	1	1	0	0	1
O7	0	0	0	0	0	1	1	0	0	1
O8	0	0	0	0	0	1	1	0	0	1
O9	0	0	0	0	0	1	1	0	0	1
O10	0	0	0	0	0	1	1	0	0	1

	m1	m2	m3	m4	m5	m6	m7	m8	m9	m10
O1	1	0	1	0	1	1	0	0	1	0
O2	1	0	1	0	1	1	0	0	1	0
O3	1	0	1	0	1	1	0	0	1	0
O4	1	0	1	0	1	1	0	0	1	0
O5	0	1	0	1	0	1	0	0	1	0
O6	0	0	0	0	0	1	1	0	0	1
O7	0	0	0	0	0	1	1	0	0	1
O8	0	0	0	0	0	1	1	0	0	1
O9	0	0	0	0	0	1	1	0	0	1
O10	0	0	0	0	0	1	1	0	0	1

	m1	m2	m3	m4	m5	m6	m7	m8	m9	m10
O1	1	0	1	0	1	1	0	0	1	0
O2	1	0	1	0	1	1	0	0	1	0
O3	1	0	1	0	1	1	0	0	1	0
O4	1	0	1	0	1	1	0	0	1	0
O5	0	1	0	1	0	1	0	0	1	0
O6	0	0	0	0	0	1	1	0	0	1
O7	0	0	0	0	0	1	1	0	0	1
O8	0	0	0	0	0	1	1	0	0	1
O9	0	0	0	0	0	1	1	0	0	1
O10	0	0	0	0	0	1	1	0	0	1

	m1	m2	m3	m4	m5	m6	m7	m8	m9	m10
O1	1	0	1	0	1	1	0	0	1	0
O2	1	0	1	0	1	1	0	0	1	0
O3	1	0	1	0	1	1	0	0	1	0
O4	1	0	1	0	1	1	0	0	1	0
O5	0	1	0	1	0	1	0	0	1	0
O6	0	0	0	0	0	1	1	0	0	1
O7	0	0	0	0	0	1	1	0	0	1
O8	0	0	0	0	0	1	1	0	0	1
O9	0	0	0	0	0	1	1	0	0	1
O10	0	0	0	0	0	1	1	0	0	1

FIGURE 7.7 – Exemple de co-clusters mixtes de $\mathbb{R}$

7.4.1 Requête 1

Dans cette requête, nous voulons connaître ce qui caractérise les femmes Evelyn, Laura, Charlotte, Frances, Pearl, Nora et Helen en terme de leur participation ou non aux événements. Pour cela, on procède à l'exploration de l'arbre représentant les co-clusters mixtes de type 3. Notons que dans le contexte Davis, ce groupe de femmes se trouvent respectivement sur le lignes 0, 1, 4, 5, 7, 13 et 14 de la matrice. Pour effectuer une telle recherche sur l'arbre $\mathbb{B}$, une fonction du package ($Patricia$) est alors utilisée.

La figure 7.8 affiche le résultat de cette requête et permet de conclure que :

1. Le groupe de femmes impliquées dans cette requête peut être départagé en quatre sous-groupes : $\{Evelyn, Laura\}$, $\{Charlotte, Frances\}$, $\{Pearl\}$ et $\{Nora, Helen\}$.
2. Alors que Evelyn et Laura fréquentent les événements E_1,E_2,E_3 et E_5 et n'assistent aucunement aux événements E_{10}, E_{11}, E_{12} et E_{14}, Nora et Helen adoptent un comportement complètement opposé à celui de Evelyn et Laura.
3. Le groupe de femmes $\{Charlotte, Frances\}$ se comporte différemment de celui du premier et dernier groupe. Ainsi, Charlotte et Frances participent aux événements E_3 et E_5 mais ne participent pas aux autres événements, à savoir : E_1, E_2 et E_{10},E_{11},E_{12} et E_{14}. Le troisième groupe, composé uniquement d'une seule femme Pearl, ne participe à aucun des événements retournés par cette requête.

7.4.2 Requête 2

Nous voulons savoir quelles sont les fréquentations des femmes pour les événements suivants : $E_2, E_3, E_5, E_6, E_9, E_{11}, E_{12}, E_{14}$. Le résultat de cette requête est affiché dans la figure. 7.9. Trois co-clusters sont générés impliquant tous ces attributs. On peut identifier sur le co-cluster de gauche cinq motifs. Le premier de ces motifs nous informe que Evelyn et Theresa participent aux événements E_2 et E_6 auxquels Charlotte, Myrna et Dorothy ne

```
greedy_match("0 1 4 5 7 13 14"):
    0 1 4 5 7 13 14====0,1,2,4,9,10,11,13,
```

	1	2	3		5				10	11	12		14
Evelyn	1	1	1	1	1	1	0	1	1	0	0	0	0
Laura	1	1	1	0	1	1	1	1	0	0	0	0	0
	0	1	1	1	1	1	1	1	1	0	0	0	0
	1	0	1	1	1	1	1	1	0	0	0	0	0
Charlotte	0	0	1	1	1	0	1	0	0	0	0	0	0
Frances	0	0	1	0	1	1	0	1	0	0	0	0	0
	0	0	0	0	1	1	1	1	0	0	0	0	0
Pearl	0	0	0	0	0	1	0	1	1	0	0	0	0
	0	0	0	0	1	0	1	1	1	0	0	0	0
	0	0	0	0	0	0	1	1	1	0	0	1	0
	0	0	0	0	0	0	0	1	1	1	0	1	0
	0	0	0	0	0	0	0	1	1	1	0	1	1
	0	0	0	0	0	0	1	1	1	1	0	1	1
Nora	0	0	0	0	0	1	1	0	1	1	1	1	1
Helen	0	0	0	0	0	0	1	1	0	1	1	1	1
	0	0	0	0	0	0	0	1	1	1	0	1	0
	0	0	0	0	0	0	0	0	1	0	1	0	0
	0	0	0	0	0	0	0	0	1	0	1	0	0

FIGURE 7.8 – Résultat de la requête 1 sur l'exemple Davis

participent pas. Le deuxième motif montre que Evelyn, Theresa et Charlotte participent aux trois événements E_3, E_4 et E_5 auxquels Myrna et Dorothy ne participent pas. Quant au troisième motif, il montre que parmi les cinq femmes, seule Charlotte ne participe pas aux deux événements E_8 et E_9. Le quatrième motif nous laisse savoir qu'aucune de ces femmes ne participe aux deux événements E_{11}, E_{14}. Le dernier motif montre que seules Myrna et Dorothy participent aux deux événements E_{10} et E_{12}. De la même façon, on peut extraire quatre motifs à partir du co-cluster du milieu et cinq motifs à partir du co-cluster de droite.

Ainsi, juste à partir de trois co-clusters, nous sommes en mesure d'extraire quatorze motifs qui nous dévoilent le comportementt des femmes par rapport aux événements ci-haut identifiés. Cet exemple met en évidence l'importance et la richesse de la connaissance qui peut être extraite à partir des co-clusters mixtes. Avec des mécanismes de visualisation sophistiqués, ces divers motifs peuvent être davantage mis en relief.

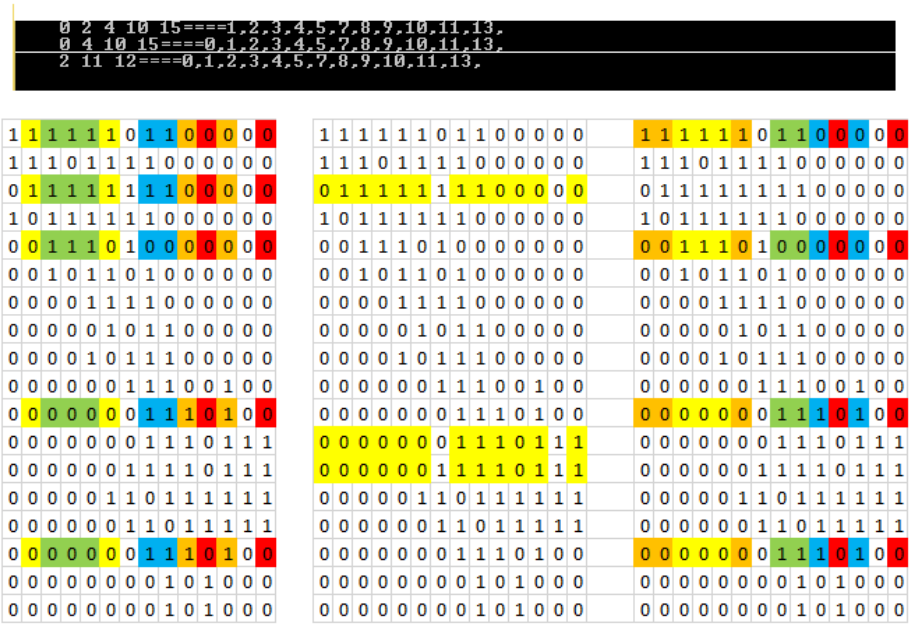


FIGURE 7.9 – Résultat de la requête 2 sur l'exemple Davis

7.4.3 Requête 3

Dans cette requête, nous voulons savoir quelles sont les femmes qui ne participent pas aux événements E_1, E_2, E_3, E_4. Pour répondre à cette requête, nous devons employer l'apposition du contexte Davis et appliquer l'approche présentée dans la sous-section 6.4.2. Ainsi l'usager du système d'information sera en mesure d'extraire du contexte Davis la liste des femmes qui ne participent pas à ces événements. Il obtiendra ainsi : Ruth, Verne, Myrna, Katherine, Sylvia, Nora, Helen, Dorothy, Olivia et Flora.

7.5 Conclusion

L'illustration de l'approche de co-clustering à travers des exemples concrets montre ses caractéristiques et les types de motifs qui peuvent être générés. En particulier, l'application de notre approche à des exemples concrets et à des ensembles de données connus comme le réseau des femmes-événements de Davis conduit à des groupes qui révèlent la coexistence de sous-groupes d'individus qui ont les mêmes caractéristiques ou présentent l'absence partielle ou totale de telles caractéristiques. Cela peut également aider à constituer une taxonomie sur les objets et/ou les attributs par consultation des groupes formés par co-clustering. Puisque l'utilisateur peut ne pas être intéressé par un ensemble massif de co-clusters, nous nous avons montré à travers trois requêtes comment on peut interroger la base des co-clusters stockés dans une structure d'arbre Patricia pour connaître par exemple le profil de fréquentation des femmes d'un ensemble donné d'événements ou les co-clusters liés à un sous-ensemble d'événements.

Chapitre 8
Conclusion Générale

La présente thèse de doctorat comporte deux principales contributions à la découverte et à la gestion de motifs principalement dans le cadre de l'analyse formelle de concepts. La première contribution concerne la production de motifs (et spécialement des concepts formels) généralisés en procédant à une agrégation d'objets et/ou d'attributs d'un contexte formel initial et selon trois modes de généralisation : $\exists$, $\forall$ et α. Cinq cas de généralisation simultanée sur les objets et les attributs sont également considérés. La seconde contribution concerne le développement d'une procédure générique de co-clustering, appelée BiP, qui part d'un contexte formel binaire pour produire quatre types de co-clusters : des concepts formels décrivant la présence d'attributs, des concepts formels décrivant l'absence d'attributs, des co-clusters dont les valeurs sont une combinaison bien définie de 0 et de 1, et des blocs dont les colonnes sont pleines de 0 ou de 1 indiquant la présence de certaines propriétés mais l'absence d'autres attributs. Le cas de la production de co-clusters à partir de données discrètes ou catégorielles est également étudié. Les divers résultats de recherche théoriques obtenus sont validés sur des collections de données connues et en phytothérapie.

Nos travaux futurs concernent les aspects suivants :

1. Identification d'autres cas de la réduction sûre de la taille du treillis de concepts suite à une généralisation $\exists$ sur les objets et/ou les attributs
2. Passage d'une base d'implications sur les attributs originaux vers la base d'implications sur les attributs produits suite à une généralisation
3. Définition rigoureuse d'un ensemble d'opérations d'interrogation des co-clusters produits par le parcours de la structure d'arbre Patricia et correspondant à des besoins et requêtes spécifiques de l'utilisateur. À titre d'exemple, l'utilisateur peut demander à trouver un co-cluster dont l'extension comporte au moins (ou plus, ou exactement) un ensemble donné d'objets afin d'en connaître ses sous-groupes et les propriétés qui les définissent
4. Comme le nombre de co-clusters issus du type 3 peut être très grand comparativement à l'effectif des autres types de co-clusters, nous envisageons de produire les co-clusters à la volée à la suite de toute requête explicite de l'utilisateur portant sur un sous-ensemble précis d'objets et/ou d'attributs par projection et sélection sur la matrice initiale.

Annexe

Signification des attributs dans la base de données : choix d'une méthode de contraception.

1. Âge de l'épouse (numérique)
2. Niveau d'éducation de l'épouse (catégoriel) 1=bas, 2, 3, 4=élevé
3. Niveau d'éducation de l'époux (catégoriel) 1=bas, 2, 3, 4=élevé
4. Nombre d'enfants (numérique)
5. Religion de l'épouse (binaire) 0=Non-Islam, 1=Islam
6. L'épouse travail ? (binaire) 0=Oui, 1=Non
7. L'occupation du mari (catégoriel) 1, 2, 3, 4
8. Index du niveau de vie standard (catégoriel) 1=bas, 2, 3, 4=élevé
9. Exposition aux média (binaire) 0=bon, 1=mauvais
10. Méthode de contraception utilisée (classe d'attribut) 1=jamais, 2=Longue période, 3=courte période

Davis	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14
EVELYN	1	1	1	1	1	1	0	1	1	0	0	0	0	0
LAURA	1	1	1	0	1	1	1	1	0	0	0	0	0	0
THERESA	0	1	1	1	1	1	1	1	1	0	0	0	0	0
BRENDA	1	0	1	1	1	1	1	1	0	0	0	0	0	0
CHARLOTTE	0	0	1	1	1	0	1	0	0	0	0	0	0	0
FRANCES	0	0	1	0	1	1	0	1	0	0	0	0	0	0
ELEANOR	0	0	0	0	1	1	1	1	0	0	0	0	0	0
PEARL	0	0	0	0	0	1	0	1	1	0	0	0	0	0
RUTH	0	0	0	0	1	0	1	1	1	0	0	0	0	0
VERNE	0	0	0	0	0	0	1	1	1	0	0	1	0	0
MYRNA	0	0	0	0	0	0	0	1	1	1	0	1	0	0
KATHERINE	0	0	0	0	0	0	0	1	1	1	0	1	1	1
SYLVIA	0	0	0	0	0	0	1	1	1	1	0	1	1	1
NORA	0	0	0	0	0	1	1	0	1	1	1	1	1	1
HELEN	0	0	0	0	0	0	1	1	0	1	1	1	1	1
DOROTHY	0	0	0	0	0	0	0	1	1	1	0	1	0	0
OLIVIA	0	0	0	0	0	0	0	0	1	0	1	0	0	0
FLORA	0	0	0	0	0	0	0	0	1	0	1	0	0	0

TABLE 8.1 – **Exemple de Davis des femmes participant à des événements**

Contextes : $\mathbb{R}; \mathbb{R}^t$

$\mathbb{R}$	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
o_1	0	1	0	1	0	0	1	1	0	1
o_2	0	1	0	1	0	0	1	1	0	1
o_3	0	1	0	1	0	0	1	1	0	1
o_4	0	1	0	1	0	0	1	1	0	1
o_5	1	0	1	0	1	0	1	1	0	1
o_6	1	1	1	1	1	0	0	1	1	0
o_7	1	1	1	1	1	0	0	1	1	0
o_8	1	1	1	1	1	0	0	1	1	0
o_9	1	1	1	1	1	0	0	1	1	0
o_{10}	1	1	1	1	1	0	0	1	1	0

$\mathbb{R}^t$	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}
m_1	0	0	0	0	1	1	1	1	1	1
m_2	1	1	1	1	0	1	1	1	1	1
m_3	0	0	0	0	1	1	1	1	1	1
m_4	1	1	1	1	0	1	1	1	1	1
m_5	0	0	0	0	1	1	1	1	1	1
m_6	0	0	0	0	0	0	0	0	0	0
m_7	1	1	1	1	1	0	0	0	0	0
m_8	1	1	1	1	1	1	1	1	1	1
m_9	1	1	1	1	1	0	0	0	0	0
m_{10}	1	1	1	1	1	0	0	0	0	0

$\mathbb{R}^t * \mathbb{R}$	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
m_1	6	5	6	5	6	0	1	6	5	1
m_2	5	9	5	9	5	0	4	9	5	4
m_3	6	5	6	5	6	0	1	6	5	1
m_4	5	9	5	9	5	0	4	9	5	4
m_5	6	5	6	5	6	0	1	6	5	1
m_6	0	0	0	0	0	0	0	0	0	0
m_7	1	4	1	4	1	0	5	5	0	5
m_8	6	9	6	9	6	0	5	10	5	5
m_9	5	5	5	5	5	0	0	5	5	0
m_{10}	1	4	1	4	1	0	5	5	0	5

$\overline{\mathbb{R}}$	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
o_1	1	0	1	0	1	1	0	0	1	0
o_2	1	0	1	0	1	1	0	0	1	0
o_3	1	0	1	0	1	1	0	0	1	0
o_4	1	0	1	0	1	1	0	0	1	0
o_5	0	1	0	1	0	1	0	0	1	0
o_6	0	0	0	0	0	1	1	0	0	1
o_7	0	0	0	0	0	1	1	0	0	1
o_8	0	0	0	0	0	1	1	0	0	1
o_9	0	0	0	0	0	1	1	0	0	1
o_{10}	0	0	0	0	0	1	1	0	0	1

$\overline{\mathbb{R}^t}$	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}
m_1	1	1	1	1	0	0	0	0	0	0
m_2	0	0	0	0	1	0	0	0	0	0
m_3	1	1	1	1	0	0	0	0	0	0
m_4	0	0	0	0	1	0	0	0	0	0
m_5	1	1	1	1	0	0	0	0	0	0
m_6	1	1	1	1	1	1	1	1	1	1
m_7	0	0	0	0	0	1	1	1	1	1
m_8	0	0	0	0	0	0	0	0	0	0
m_9	1	1	1	1	1	0	0	0	0	0
m_{10}	0	0	0	0	0	1	1	1	1	1

$\overline{\mathbb{R}^t} * \overline{\mathbb{R}}$	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
m_1	4	0	4	0	4	4	0	0	4	0
m_2	0	1	0	1	0	1	0	0	1	0
m_3	4	0	4	0	4	4	0	0	4	0
m_4	0	1	0	1	0	1	0	0	1	0
m_5	4	0	4	0	4	4	0	0	4	0
m_6	4	1	4	1	4	10	5	0	5	5
m_7	0	0	0	0	0	5	5	0	0	5
m_8	0	0	0	0	0	0	0	0	0	0
m_9	4	1	4	1	4	5	0	0	5	0
m_{10}	0	0	0	0	0	5	5	0	0	5

$\mathbb{X}$	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
m_1	10	5	10	5	10	4	1	6	9	1
m_2	5	10	5	10	5	1	4	9	6	4
m_3	10	5	10	5	10	4	1	6	9	1
m_4	5	10	5	10	5	1	4	9	6	4
m_5	10	5	10	5	10	4	1	6	9	1
m_6	4	1	4	1	4	10	5	0	5	5
m_7	1	4	1	4	1	5	10	5	0	10
m_8	6	9	6	9	6	0	5	10	5	5
m_9	9	6	9	6	9	5	0	5	10	0
m_{10}	1	4	1	4	1	5	10	5	0	10

Bibliographie

[1] Adda, M., Valtchev, P., Missaoui, R., and Djeraba, C. (2007). Toward recommendation based on ontology-powered web-usage mining. *IEEE Internet Computing*, 11(4) :45–52.

[2] Alpen, E. (2010). *Précis de Phytotérapie*. Edition Alpen.

[3] Ayadi, W., Elloumi, M., and Hao, J.-K. (2009). A biclustering algorithm based on a bicluster enumeration tree : application to dna microarray data. *BioData mining*, 2(1) :9.

[4] Balamane, A., Missaoui, R., Kwuida, L., and Vaillancourt, J. (2016). Descriptive group detection in two-mode data networks using biclustering. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 744–749.

[5] Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., and Lakhal, L. (2002). Pascal : un algorithme d'extraction des motifs fréquents. *Techniques et Sciences Informatiquess*, 21(1) :65–95.

[6] Batagelj, V., Ferligoj, A., and Doreian, P. (1992). Direct and indirect methods for structural equivalence. *Social Networks*, 14(12) :63 – 90. Special Issue on Blockmodels.

[7] Belohlávek, R. and Vychodil, V. (2008). Adding background knowledge to formal concept analysis via attribute dependency formulas. In *SAC*, pages 938–943.

[8] Ben-Dor, A., Chor, B., Karp, R., and Yakhini, Z. (2003). Discovering local structure in gene expression data : the order-preserving submatrix problem. *Journal of computational biology*, 10(3-4) :373–384.

[9] Berge, C. (1979). *Graphs and hypergraphs*, volume 6 of *North - Holland Mathematical Library*. North-Holland, Elsevier, Amsterdam, repr. of the 2., rev. ed. edition.

[10] Bertino, E., Catania, B., and Maddalena, A. (2004). Towards a language for pattern manipulation and querying. In *Proceedings of the Intl. Workshop on Pattern Representation and Management, Heraklion, Hellas, March 18, 2004, PaRMa 2004 was held in conjunction with the 9th Int. Conference on Extending Database Technology (EDBT 2004)*.

[11] Busygin, S., Prokopyev, O., and Pardalos, P. M. (2008). Biclustering in data mining. *Computers & Operations Research*, 35(9) :2964–2987.

[12] Calders, T., Rigotti, C., and Boulicaut, J.-F. (2006). A survey on condensed representations for frequent sets. In *Constraint-based mining and inductive databases*, pages 64–80. Springer.

[13] Catania, B. and Maddalena, A. (2006). Pattern management : Practice and challenges. *Processing and managing complex data for decision support*, pages 280–317.

[14] Charrad, M., Lechevallier, Y., Saporta, G., and Ben Ahmed, M. (2008). Le bi-partitionnement : état de l'art sur les approches et les algorithmes. *EcolIA*, 8.

[15] Chevallier, A. (2001). *Encyclopédie des plantes médicinales*. Larousse, 2ième edition.

[16] Cimiano, P., Hotho, A., Stumme, G., and Tane, J. (2004). Conceptual knowledge processing with formal concept analysis and ontologies. In Eklund, P. W., editor, *ICFCA*, volume 2961 of *Lecture Notes in Computer Science*, pages 189–207. Springer.

[17] Curé, O. and Jeansoulin, R. (2008). An fca-based solution for ontology mediation. In *ONISW '08 : Proceeding of the 2nd int. workshop on Ontologies and nformation systems for the semantic web*, pages 39–46, New York, NY, USA. ACM.

[18] Davis, A., Gardner, B. B., and Gardner, M. R. (1941). *Deep South*. The University of Chicago Press.

[19] Dhillon, I. S., Mallela, S., and Modha, D. S. (2003a). Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM.

[20] Dhillon, I. S., Mallela, S., and Modha, D. S. (2003b). Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM.

[21] Duquenne, V. (2010a). Lattice drawings and morphisms. In Kwuida, L. and Sertkaya, B., editors, *Formal Concept Analysis*, volume 5986 of *Lecture Notes in Computer Science*, pages 88–103. Springer Berlin Heidelberg.

[22] Duquenne, V. (2010b). Lattice drawings and morphisms. In *Formal Concept Analysis*, pages 88–103. Springer.

[23] Erné, M. (1993). Distributive laws for concept lattices. *Algebra Universalis*, 30 :538–580.

[24] Everett, M. and Borgatti, S. (2013a). The dual-projection approach for two-mode networks. *Social Networks*, 35(2) :204 – 210. Special Issue on Advances in Two-mode Social Networks.

[25] Everett, M. G. and Borgatti, S. P. (2013b). The dual-projection approach for two-mode networks. *Social Networks*, 35(2) :204–210.

[26] Fan, L. and Xiao, T. (2007). An automatic method for ontology mapping. In Apolloni, B., Howlett, R. J., and Jain, L. C., editors, *Knowledge-Based Intelligent Information and Engineering Systems*, volume 4694 of *Lecture Notes in Computer Science*, pages 661–669. Springer.

[27] Ferré, S. and Ridoux, O. (2000). A logical generalization of formal concept analysis. In *8th International Conference on Conceptual Structures*, pages 371–384.

[28] Ferré, S. and Ridoux, O. (2004). Introduction to logical information systems. *Inf. Process. Manage.*, 40(3) :383–419.

[29] Formica, A. (2006). Ontology-based concept similarity in formal concept analysis. *Inf. Sci.*, 176(18) :2624–2641.

[30] Freeman, L. C. (2003). Finding social groups : A meta-analysis of the southern women data. In Ronald Breiger, K. C. and Pattison, P., editors, *Dynamic Social Network Modelling and Analysis : Workshop Summary and Papers*, pages 1–60. The National Academies Press.

[31] Ganter, B. and Kuznetsov, S. O. (2001a). Pattern structures and their projections. In *9th International Conference on Conceptual Structures*, pages 129–142.

[32] Ganter, B. and Kuznetsov, S. O. (2001b). Pattern structures and their projections. In *Conceptual Structures : Broadening the Base*, pages 129–142. Springer.

[33] Ganter, B. and Wille, R. (1999a). Decompositions of concept lattices. In *Formal Concept Analysis/Mathematical Foundations*, pages 129–181. Springer Berlin Heidelberg.

[34] Ganter, B. and Wille, R. (1999b). *Formal Concept Analysis : Mathematical Foundations*. Springer-Verlag New York, Inc. Translator-C. Franzke.

[35] Ganter, B. and Wille, R. (1999c). *Formal Concept Analysis : Mathematical Foundations*. Springer-Verlag New York, Inc. Translator-C. Franzke.

[36] Ganter, B. and Wille, R. (1999d). *Formal concept analysis : mathematical foundations*. Springer Science & Business Media.

[37] Haav, H.-M. (2004). A semi-automatic method to ontology design by using fca. In Snásel, V. and Belohlávek, R., editors, *Concept Lattices and Applications (CLA)*, volume 110 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[38] Hartigan, J. A. (1972). Direct clustering of a data matrix. *Journal of the american statistical association*, 67(337) :123–129.

[39] Hofmann, T. and Puzicha, J. (1999). Latent class models for collaborative filtering. In *IJCAI*, volume 99, pages 688–693.

[40] Kaytoue, M., Kuznetsov, S. O., and Napoli, A. (2011a). Biclustering numerical data in formal concept analysis. In *Formal Concept Analysis*, pages 135–150. Springer.

[41] Kaytoue, M., Kuznetsov, S. O., Napoli, A., and Duplessis, S. (2011b). Mining gene expression data with pattern structures in formal concept analysis. *Inf. Sci.*, 181(10) :1989–2001.

[42] Kaytoue, M., Marcuola, F., Napoli, A., Szathmary, L., and Villerd, J. (2010). The coron system. In *ICFCA Supplementary Proceedings*, pages 55–58.

[43] Kuznetsov, S. and Obiedkov, S. (2002). Comparing performance of algorithms for generating concept lattices. *Journal of Experimental and Theoretical Artificial Intelligence*, 14 :189–216.

[44] Kuznetsov, S. O. (1999). Learning of simple conceptual graphs from positive and negative examples. In *PKDD*, pages 384–391.

[45] Kwuida, L., Missaoui, R., Amor, B. B., Boumedjout, L., and Vaillancourt, J. (2010a). Restrictions on concept lattices for pattern management. In Kryszkiewicz, M. and Obiedkov, S. A., editors, *Concept Lattices and Applications (CLA)*, volume 672 of *CEUR Workshop Proceedings*, pages 235–246. CEUR-WS.org.

[46] Kwuida, L., Missaoui, R., Amor, B. B., Boumedjout, L., and Vaillancourt, J. (2010b). Restrictions on concept lattices for pattern management. In *CLA*, volume 672, pages 235–246.

[47] Kwuida, L., Missaoui, R., Balamane, A., and Vaillancourt, J. (2014). Generalized pattern extraction from concept lattices. *Annals of Mathematics and Artificial Intelligence*, 72(1-2) :151–168.

[48] Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004a). Rcv1 : A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5 :361–397.

[49] Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004b). Rcv1 : A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5 :361–397.

[50] Li, L., Guo, Y., Wu, W., Shi, Y., Cheng, J., and Tao, S. (2012). A comparison and evaluation of five biclustering algorithms by quantifying goodness of biclusters for gene expression data. *BioData mining*, 5(8) :8.

[51] Lichman, M. (2013). UCI machine learning repository.

[52] Lim, T.-S. (1987). Contraceptive method choice data set. https ://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice.

[53] Liquire, M. and Sallantin, J. (1998). Structural machine learning with galois lattice and graphs. In *Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin*, pages 305–313.

[54] Madeira, S. and Oliveira, A. (2004a). Biclustering algorithms for biological data analysis : a survey. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 1(1) :24–45.

[55] Madeira, S. C. and Oliveira, A. L. (2004b). Biclustering algorithms for biological data analysis : a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1) :24–45.

[56] Meschke, D.-M. C. (2012). *Concept Approximations*. PhD thesis, Higher School of Economics.

[57] Missaoui, R., Kwuida, L., Quafafou, M., and Vaillancourt, J. (2009). Algebraic operators for querying pattern bases. *arXiv preprint arXiv :0902.4042*.

[58] Missaoui, R., Nourine, L., and Renaud, Y. (2010). An inference system for exhaustive generation of mixed and purely negative implications from purely positive ones. In *CLA*, pages 271–282.

[59] Morin., P. (2014). Arbre patricia. cglab.ca/ morin/teaching/5408/notes/strings.pdf.

[60] Morrison, D. R. (1968). PATRICIA - practical algorithm to retrieve information coded in alphanumeric. *J. ACM*, 15(4) :514–534.

[61] Murali, T. and Kasif, S. (2003). Extracting conserved gene expression motifs from gene expression data. In *Pacific Symposium on Biocomputing*, volume 8, pages 77–88. World Scientific.

[62] Murata, T. (2009). Detecting communities from bipartite networks based on bipartite modularities. In *Computational Science and Engineering, 2009. CSE '09. International Conference on*, volume 4, pages 50–57.

[63] Orzechowski, P. and Boryczko, K. (2016). Propagation-based biclustering algorithm for extracting inclusion-maximal motifs. *Computing and Informatics*, 35(2) :391–410.

[64] Pernelle, N., Rousset, M.-C., Soldano, H., and Ventos, V. (2002). Zoom : a nested galois lattices-based system for conceptual clustering. *J. Exp. Theor. Artif. Intell.*, 14(2-3) :157–187.

[65] Prelić, A., Bleuler, S., Zimmermann, P., Wille, A., Búhlmann, P., Gruissem, W., Hennig, L., Thiele, L., and Zitzler, E. (2006). A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9) :1122–1129.

[66] Rai, P. and Singh, S. (2010). A survey of clustering techniques. *International Journal of Computer Applications*, 7(12) :1–5.

[67] Rodriguez-Jimenez, J., Cordero, P., Enciso, M., and Mora, A. (2014). Negative attributes and implications in formal concept analysis. *Procedia Computer Science*, 31 :758 – 765.

[68] Srikant, R. and Agrawal, R. (1995). Mining generalized association rules. In Dayal, U., Gray, P. M. D., and Nishio, S., editors, *VLDB*, pages 407–419. Morgan Kaufmann.

[69] Stumme, G. and Maedche, A. (2001). FCA-MERGE : Bottom-up merging of ontologies. In *Seventeenth International Joint Conference on Artificial Intelligence (IJCAI) Seattle, Washington*, pages 225–234.

[70] Tanay, A., Sharan, R., and Shamir, R. (2005). Biclustering algorithms : A survey. *Handbook of computational molecular biology*, 9 :26–1.

[71] Terrovitis, M. and Vassiliadis, P. (2003). Architecture for pattern-base management systems. In *PANDA Workshop*.

[72] Terrovitis, M., Vassiliadis, P., Skiadopoulos, S., Bertino, E., Catania, B., Maddalena, A., and Rizzi, S. (2007). Modeling and language support for the management of pattern-bases. *Data Knowledge Engineering*, 62(2) :368 – 397.

[73] Wang, H., Wang, W., Yang, J., and Yu, P. S. (2002). Clustering by pattern similarity in large data sets. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 394–405. ACM.

[74] Wang, J. and He, K. (2006). Towards representing fca-based ontologies in semantic web rule language. In *CIT '06 : Proceedings of the Sixth IEEE International Conference on Computer and Information Technology*, page 41, Washington, DC, USA. IEEE Computer Society.

[75] Wille, R. (2004). Preconcept algebras and generalized double boolean algebras. In *ICFCA*, pages 1–13.

[76] Yang, M.-S. (1993). A survey of fuzzy clustering. *Mathematical and Computer modelling*, 18(11) :1–16.

[77] Yevtushenko., S. A. (2000). System of data analysis "concept explorer". (in russian). *Proceedings of the 7th national conference on Artificial Intelligence*, pages 127–134.