HAL
archives-ouvertes.fr

Connaissance inter-entreprises et optimisation combinatoire

Mohamed Ould Mohamed Lemine

▶ **To cite this version:**

Mohamed Ould Mohamed Lemine. Connaissance inter-entreprises et optimisation combinatoire. Autre [cs.OH]. Université Paris Dauphine - Paris IX, 2014. Français. <NNT : 2014PA090015>. <tel-01552184>

HAL Id: tel-01552184
https://tel.archives-ouvertes.fr/tel-01552184

Submitted on 1 Jul 2017

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

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

DAUPHINE
UNIVERSITÉ PARIS

Université Paris-Dauphine

ÉCOLE DOCTORALE DE DAUPHINE

THÈSE

préparée au **LAMSADE** et à **Altares D&B**

présentée par

Mohamed OULD MOHAMED LEMINE

pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ
Spécialité : INFORMATIQUE

Connaissance Inter-Entreprises et Optimisation Combinatoire

Soutenue publiquement le 17 juin 2014 devant le jury :

A. R.	Mahjoub	Directeur de thèse	Université Paris-Dauphine, France
G.	Marteau	Co-encadrant	Altares D & B, Nanterre, France
H.	Kheddouci	Rapporteur	Université Lyon I, Lyon, France
A.	Quilliot	Rapporteur	Université Blaise Pascal, Clermont II, France
A.	Haouba	Examinateur	Université de Nouakchott, Mauritanie
S.	Martin	Examinateur	Université de Lorraine, France
E.	Uchoa	Examinateur	Universidade Federal Fluminense, Brasil

altares

Résumé

La connaissance inter-entreprises permet à chaque société de se renseigner sur ses clients, ses fournisseurs et de développer son activité tout en limitant le risque lié à la solvabilité ou retard de paiement de ses partenaires. Avec les tensions de trésorerie, la nécessité de la croissance et l'augmentation de la concurrence, ce domaine devient plus que jamais stratégique aussi bien pour les PME que pour les grands groupes. La quantité de données traitée dans ce domaine, les exigences de qualité et de fraîcheur, la nécessité de croiser ces données pour déduire des nouvelles informations et indicateurs, posent plusieurs problèmes pour lesquels l'optimisation en général et l'optimisation combinatoire en particulier peuvent apporter des solutions efficaces.

Dans cette thèse, nous utilisons l'optimisation combinatoire, l'algorithmique du texte et la théorie des graphes pour résoudre efficacement des problèmes issus du domaine de la connaissance inter-entreprises et posés par Altares D&B.

Dans un premier temps, nous nous intéressons à la qualité de la base de données des dirigeants. Ce problème combine la détection et suppression des doublons dans une base de données et la détection d'erreurs dans une chaîne de caractères. Nous proposons une méthode de résolution basée sur la normalisation des données et l'algorithmique de texte et de comparaison syntaxique entre deux chaînes de caractères. Les résultats expérimentaux montrent non seulement que cette méthode est pertinente dans la détection et la suppression des doublons mais aussi qu'elle est efficace de point du vue temps de traitement.

Nous nous focalisons par la suite sur les données des liens capitalistiques et nous considérons le problème de calcul des liens indirects et l'identification des têtes des groupes. Nous présentons une méthode de résolution basée sur la théorie des graphes. Nous testons cette méthode sur plusieurs instances réelles. Nous prouvons l'efficacité de cette méthode par son temps de traitement et par l'espace de calcul qu'elle utilise. Enfin, nous remarquons que le temps de calcul de celui-ci augmente de façon logarithmique en fonction de la taille d'instance.

Enfin, nous considérons le problème de l'identification des réseaux d'influence. Nous formalisons ce problème en termes de graphes et nous le ramenons à un problème de partitionnement de graphe qui est NP-difficile dans ce cas général. Nous proposons alors une formulation en programme linéaire en nombre entier pour ce problème. Nous étudions le polyèdre associé et décrivons plusieurs classes de contraintes valides. Nous donnons des conditions nécessaires pour que ces contraintes définissent

des facettes et discutons des algorithmes de séparations de ces contraintes. En utilisant les résultats polyédraux obtenus, nous développons un algorithme de coupes et branchements. Enfin, nous donnons quelques résultats expérimentaux qui montrent l'efficacité de notre algorithme de coupes et branchements.

Mots clés : Connaissance inter-entreprises, optimisation combinatoire, déduplication des données, similarité syntaxique, complexité, graphe, polytope, facette, séparation, algorithme de coupes et branchements.

Abstract

The inter-companies knowledge allows to every partner to learn about its customers, its suppliers and to develop its activity. Also this permits to limit the risk related to the creditworthiness, or the late payment of its partners. With the cash flow pressures, the need for growth and increased competition, this area becomes more strategic than ever, for both small (PME) and large groups. The amount of data processed in this domain, the requirements of quality and freshness, the need to cross these data to obtain new information and indicators, yield several optimization problems for which the recent techniques and computational tools can bring effective solutions.

In this thesis, we use combinatorial optimization, text algorithms as well as graph theory to solve efficiently problems arising in the field of inter-companies knowledge. In particular, such problems was encountered in Altares D&B.

First, we focus on the quality of the managers database. This problem combines the detection and removal of duplicates in a database, as well as the error detection in a string. We propose a method for solving this problem, based on data normalization, text algorithms and syntactic comparison between two strings. Our experimental results show that this method is relevant for the detection and removal of duplicates, and it is also very efficient in terms of processing time.

In a second part of the thesis, we address a problem related to the data of ownership links. We compute the indirect links, and identify the group heads. We propose a method for solving this problem using graph theory and combinatorial optimization. We then perform a set of experiments on several real-world instances. The computational results show the effectiveness of our method in terms of CPU-time and resource allocation. In fact, the CPU time for computation increases logarithmically with the size of the instances.

Finally, we consider the problem of identifying influence networks. We give a description of this problem in terms of graphs, and show that it can reduce to a graph partitioning problem. The latter is NP-hard. We then propose an integer linear programming formulation to model the problem. We investigate the associated polyhedron and describe several classes of valid inequalities. We give some necessary and sufficient conditions for these inequalities to define facets of the considered polyhedron, and we discuss the related separation problems. Based on the obtained polyhedral results, we devise a Branch-and-Cut algorithm to solve the problem. Some numerical results are presented to show the efficiency of our algorithm.

Key words : inter-companies knowledge, combinatorial optimization, data duplication, syntactic similarities, computational complexity, graph, polytope, facet, separation, Branch-and-Cut algorithm.

Table des matières

Chapitre 1

Introduction

Face à la mondialisation galopante et à l'accélération des mutations dans le monde professionnel, les besoins des décideurs en matière d'informations sur les entreprises ne cessent d'évoluer. Par conséquent, l'intégration, le stockage et la traçabilité des données deviennent des enjeux quotidien. De plus, dans un monde interconnecté, chaque évènement affectant une entreprise finit par avoir des conséquences directes sur ses partenaires, clients, fournisseurs ou concurrents. Par exemple, suite aux difficultés vécues par les constructeurs automobiles lors de la crise économique en 2008, plusieurs sous-traitants ont rapidement connu des problèmes dûs au manque de contrats de la part de ces constructeurs.

Obtenir des informations sur les autres entreprises représente donc un enjeu stratégique pour chaque société. Ces informations doivent en outre être fiables, récentes et recoupées afin de pouvoir prendre les décisions adéquates à chaque situation. En ce qui concerne la gestion des risques, l'importance des informations est encore plus grande. En effet, faire un crédit à une entreprise en faillite ou qui ne paie pas dans les temps peut s'avérer dangereux et nécessite de prendre certaines assurances. Des études ont montré que près d'une défaillance sur quatre est dûe au défaut de paiement d'un client. D'autre part, dans le contexte de crise actuelle, la gestion des risques doit, elle aussi, être plus précise car les entreprises en mauvaises santés financières sont plus nombreuses et docn les risques sont plus grands. Il convient alors de surveiller plus attentivement les entreprises liées à la sienne mais aussi d'examiner les sociétés qui ne sont pas directement en contact avec l'activité de l'entreprise. L'information d'entreprises est, donc devenue, avec la crise un domaine sensible et stratégique.

C'est dans ce contexte qu'opère Altares, spécialiste de l'information BtoB. Altares propose des solutions d'information à valeur ajoutée pour aider les entreprises à optimiser leurs processus de décision et à développer leurs activités. Altares est membre exclusif du réseau Dun & Bradstreet. Ce dernier possède des informations sur 231 millions d'entreprises réparties dans 220 pays. Le cœur de métier d'Altares, appelé la connaissance inter-entreprise, permet d'informer les entreprises sur l'identité, la solvabilité et la santé financière de leurs clients, fournisseurs et prospects. En temps

de crise, ce domaine est devenu critique et ses enjeux ont dépassé le simple service BtoB. Les entreprises dont certaines décisions vont dépendre des informations fournies par Altares ou de l'un de ses concurrents vont être d'autant plus attentives, du fait de la crise, au choix de leurs fournisseurs.

Les bases de données d'Altares contiennent toutes les informations sur la totalité des entreprises françaises, actives et radiées avec un historique de 20 ans. De ce fait, ces bases de données sont gigantesques. D'autre part, chaque jour Altares reçoit, de la part de ses fournisseurs, une centaine de milliers de mises à jour concernant les entreprises françaises et leurs dirigeants. Toutes ces informations doivent être vérifiées, croisées entre elles et intégrées dans les bases de données d'Altares le plus rapidement possible. Ensuite, Altares doit calculer, à la base de ces informations, de nouvelles informations à grande valeur ajoutée, qu'elle met à la disposition de ses clients. Pour cela, Altares a besoin d'algorithmes efficaces et rapides. C'est dans ce cadre que s'inscrit cette thèse dont l'objectif est de résoudre efficacement des problèmes opérationnels d'Altares.

Cette thèse, réalisée dans le cadre d'un contrat de recherche entre Altares et le laboratoire LAMSADE de l'université Paris-Dauphine, porte sur l'optimisation dans le domaine de la connaissance inter-entreprise. La thèse s'articule autour de trois axes. Dans un premier temps, nous nous intéressons à l'amélioration de la qualité de la base de données des dirigeants. En particulier, nous cherchons à identifier et supprimer les doublons qui existent dans cette base. Ce problème sera appelé Problème de Déduplication des Dirigeants (PDD). Ensuite, nous nous focalisons sur les données des liens capitalistiques. Les liens capitalistiques peuvent être définis comme les liens d'actionnariat et participation des entreprises. Altares dispose de tous les liens capitalistiques directs. Elle a, cependant, besoin de calculer les liens indirects et d'identifier le groupe d'entreprises qui possède le même actionnaire ultime. Cette actionnaire est appelée *tête de groupe*. Enfin, nous cherchons à caractériser les réseaux d'influence. En combinant les informations sur les dirigeants et les informations des liens capitalistiques, nous construisons un graphe qui donne les relations entre les personnes (dirigeant ou actionnaire). L'identification des réseaux d'influence revient à partitionner ce graphe en minimisant la plus grande coupe entre n'importe quel deux ensembles de la partition. Pour résoudre ce problème, nous proposons une formulation à l'aide de la programmation mathématique. Cette formulation sera améliorée à l'aide de l'approche polyédrale.

Le manuscrit est organisé comme suit. Dans le Chapitre 2, nous introduisons quelques notions de base sur les graphes, les polyèdres et la théorie de la complexité. Nous donnons aussi les notations qui seront utiles tout au long de la thèse. Le Chapitre 3 présente les bases de données en général, les bases de données d'Al-

tares et donne quelques exemples de problèmes d'optimisation dans les bases de données. Le Chapitre 4 est consacré au problème au problème de déduplication des dirigeants. Nous verrons que ce problème fait intervenir le calcul de la similarité syntaxique entre des chaînes de caractères. Nous donnons un état de l'art des méthodes de similarité syntaxique. Puis nous présentons la méthode de résolution ainsi que les résultats obtenus. Le Chapitre 5 est dédié au problème du calcul des liens capitalistiques indirects et au problème de l'identification des têtes de groupe. Après une présentation des données des liens capitalistiques, nous introduisons les deux problèmes étudiés ainsi que leurs motivations. Ensuite, nous présentons la méthode de résolution proposée ainsi que les résultat obtenus. Finalement, le Chapitre 6 est centré de l'identification des réseaux d'influence. Nous ramenons ce problème à un problème de partitionnement de graphe pour lequel nous donnons une formulation en programme linéaire en nombres entiers. Nous étudions ensuite le polytope associé à ce problème et nous introduisons des inégalités valides pour ce polytope. Enfin, nous proposons un algorithme de coupes et branchements et donnons des résultats expérimentaux afin de le valider.

Chapitre 2

Notions préliminaires

Sommaire

Dans ce chapitre, nous donnons quelques notions de base sur les polyèdres combinatoires et la théorie de la complexité. Nous effectuons ensuite un bref rappel sur les méthodes de coupes et branchements. Nous terminons ce chapitre par les notations et définitions qui seront utilisées tout au long de ce manuscrit.

2.1 Problèmes d'optimisation combinatoire

L'*Optimisation combinatoire* est une des branches de l'informatique et des mathématiques appliquées. Elle concerne les problèmes pouvant se formuler de la façon suivante : soit $E = \{e_1, \ldots, e_n\}$ un ensemble fini appelé *ensemble de base*, où chaque élément e_i possède un *poids* $c(e_i)$. Soit $\mathcal{S}$ une famille de sous-ensembles de E. Si $S \in \mathcal{S}$, alors $c(S) = \sum_{e_i \in S} c(e_i)$ est le poids de S. Le problème consiste à déterminer un élément de $\mathcal{S}$, ayant le plus petit (ou le plus grand) poids. Un tel problème est appelé un *problème d'optimisation combinatoire*. L'ensemble $\mathcal{S}$ est appelé l'*ensemble des solutions* du problème.

Le mot *combinatoire*, qui désigne la discipline des mathématiques concernée par les structures discrètes ou finies, évoque l'existence d'une structure sous-jacente discrète (généralement un graphe). L'ensemble de solutions $\mathcal{S}$ est défini dans cette structure et peut avoir un nombre exponentiel d'éléments. Le mot *optimisation* signifie que l'on recherche le meilleur élément de l'ensemble de solutions $\mathcal{S}$.

De nombreux problèmes peuvent se formuler comme des problèmes d'optimisation combinatoire, par exemple le problème du sac à dos, du voyageur de commerce, de conception de réseaux, de transport, de localisation... L'optimisation combinatoire se trouve au carrefour de la théorie des graphes, de la programmation linéaire et de la programmation linéaire en nombres entiers. Elle est très liée à la théorie de la complexité d'algorithmes.

2.2 Rappels sur la théorie de la complexité

La théorie de la complexité est basée sur les travaux d'Edmonds [33] et de Cook [26]. Elle permet de classer un problème donné parmi les problèmes faciles ou difficiles. Dans cette section, nous rappelons quelques éléments de base. La théorie de la NP-complétude est présentée en détail dans Garey et Johnson [39].

Un *problème* est une question générale possédant des paramètres dont la valeur n'est pas connue. Un problème est décrit en donnant : une description générale de tous les paramètres, et une énumération des propriétés que la solution doit satisfaire. Une *instance* d'un problème est obtenue en spécifiant la valeur de chaque paramètre du problème. Un *algorithme* de résolution d'un problème donné est une procédure, décomposable en opérations élémentaires, qui pour chaque instance du problème, produit une solution. La *taille* d'un problème reflète le nombre de données nécessaires pour décrire une instance.

Un algorithme est dit en $O(f(n))$ s'il existe un scalaire c et un entier n_0 tels que nombre d'opérations élémentaires nécessaires pour résoudre une instance de taille n au est plus $c.f(n)$ pour tout $n \geq n_0$. Si f est une fonction polynomiale, alors l'algorithme est dit *polynomial*. Un problème appartient à la *classe* P s'il existe, pour toute instance du problème, un algorithme polynomial en la taille de l'instance permettant de résoudre le problème. Les problèmes de la classe P sont dits *faciles*.

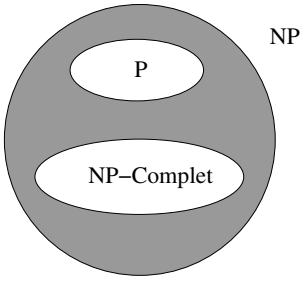

FIGURE 2.1 – Relations entre P, NP et NP-Complet

Un *problème de décision* est un problème ayant deux réponses possibles : *oui*

ou *non*. Soient $\mathcal{P}$ un problème de décision et $\mathcal{I}$ les instances de ce problème pour lesquelles la réponse est oui. $\mathcal{P}$ appartient à la *classe NP* (Nondeterministic Polynomial) s'il existe un algorithme polynomial qui permet de vérifier que la réponse est oui pour toute instance de $\mathcal{I}$. Il est clair que la classe P est contenue dans la classe NP (voir figure 2.1). La différence entre P et NP n'a pas été prouvée, mais la conjecture est considérée comme hautement probable.

Nous distinguons également dans la classe NP, la classe des problèmes NP-complets. La NP-complétude s'appuie sur la notion de réduction polynomiale. Un problème de décision P_1 se réduit polynomialement en un problème de décision P_2 s'il existe une fonction polynomiale f telle que, pour toute instance I de P_1, la réponse est oui si et seulement si la réponse de $f(I)$ pour P_2 est oui. Nous noterons alors $P_1 \alpha P_2$. Un problème $\mathcal{P}$ est *NP-complet*, s'il appartient à la classe NP et s'il existe un problème Q connu comme étant NP-complet tels que $Q \alpha P$. Cook a été le premier à montrer la NP-complétude d'un problème, celui de la satisfiabilité [26].

À tout problème d'optimisation combinatoire peut être associé un problème de décision. Tout problème d'optimisation combinatoire dont le problème de décision associé est NP-complet est dit *NP-difficile*.

2.3 Approche polyédrale et méthode de coupes et branchements

Un grand nombre de problèmes issus du monde réel peuvent être formulés sous la forme de problèmes d'optimisation combinatoire. A première vue, leur résolution semble facile. Nous pouvons par exemple faire une énumération de toutes les solutions possibles et choisir la meilleure. Cependant, le nombre de ces solutions devient vite exponentiel et la méthode énumérative n'est plus possible. D'où la nécessité de développer des techniques permettant de résoudre ces problèmes plus efficacement. Une des méthodes les plus puissantes est la méthode dite *polyédrale* qui a été introduite par Edmonds en 1965 [34] pour le problème du couplage. Nous allons à présent présenter brièvement cette approche. Pour plus de détails, voir par exemple [82, 56].

2.3.1 Définitions

Nous allons tout d'abord rappeler quelques définitions et propriétés liées à la théorie des polyèdres.

Soit $n \in \mathbb{N}$. Le symbole $\mathbb{R}^n$ représente l'ensemble des vecteurs ayant n composantes réelles. L'ensemble des nombres réels non négatifs sera noté $\mathbb{R}^+$.

Étant donné un ensemble de points $x^1, \ldots, x^m \in \mathbb{R}^n$, un point $x \in \mathbb{R}^n$ est dit *combinaison linéaire* de $x^1, \ldots, x^m$ s'il existe $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ tels que

$$x = \sum_{i=1}^{m} \lambda_i x^i.$$

Si de plus

$$\sum_{i=1}^{m} \lambda_i = 1$$

$$\text{(resp. } \lambda_i \in \mathbb{R}_+ \text{ pour i=1,...,m et } \sum_{i=1}^{m} \lambda_i = 1),$$

x est dit *combinaison affine* (resp. *combinaison convexe*) de ces points.

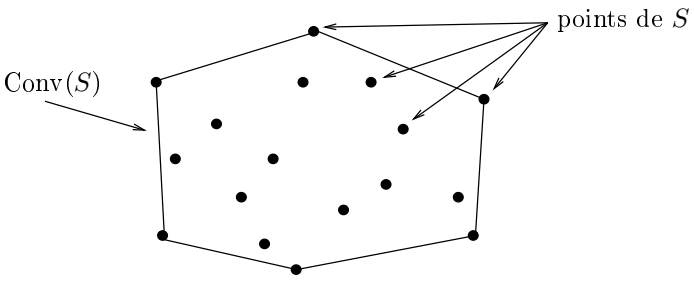

FIGURE 2.2 – Enveloppe convexe

Étant donné un ensemble S de points $x^1, \ldots, x^m \in \mathbb{R}^n$, l'*enveloppe convexe* de $x^1, \ldots, x^m$ est

$$conv(S) = \{x \in \mathbb{R}^n \mid x \text{ combinaison convexe de } x^1, \ldots, x^m\}.$$

La figure 2.2 illustre cette notion.

Des points $x^1, \ldots, x^m \in \mathbb{R}^n$ sont *linéairement indépendants* (resp. *affinement indépendants*) si le système

$$\sum_{i=1}^{m} \lambda_i x^i = 0$$

$$\text{(resp. } \sum_{i=1}^{m} \lambda_i x^i = 0 \text{ et } \sum_{i=1}^{m} \lambda_i = 0)$$

admet une solution unique $\lambda_i = 0$ pour $i = 1, \ldots, m$.

Un *polyèdre* P est l'ensemble des solutions d'un système linéaire $Ax \leq b$ c'est-à-dire $P = \{x \in I\!R^n \mid Ax \leq b\}$, où A est une matrice à m lignes et n colonnes, et b un vecteur à m composantes. Un *polytope* est un polyèdre borné.

Un polyèdre $P \subseteq I\!R^n$ est dit de *dimension* p si le nombre maximum de points de P affinement indépendants est $p+1$. Nous noterons alors $dim(P) = p$. Un polyèdre P de $I\!R^n$ est *de pleine dimension* si $dim(P) = n$.

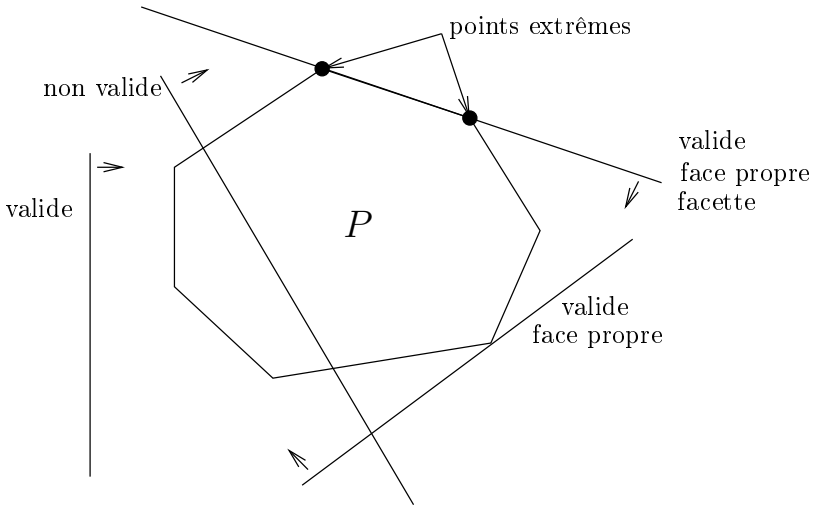

FIGURE 2.3 – Contraintes valides, faces et facettes

Si a et x sont deux vecteurs colonne à n composantes, on note par $a^T x$, le produit scalaire de a et x. Une contrainte $a^T x \leq \alpha$ est dite *valide* pour un polyèdre P de $I\!R^n$ si elle est vérifiée pour toute solution de P. Soit $x^* \in I\!R^n$. L'inégalité $a^T x \leq \alpha$ est dite *serrée* pour x^* si $a^T x^* = \alpha$. Une contrainte est dite *violée* par x^* si x^* ne satisfait pas la contrainte (voir figure 2.3).

Étant donné un polyèdre P et une contrainte $a^T x \leq \alpha$ valide pour P, le sous-ensemble $F = \{x \in P \mid a^T x = \alpha\}$ est appelé *face* de P *définie* par $a^T x \leq \alpha$. De plus nous avons $dim(F) \leq dim(P)$. Une face F est dite *propre* si $F \neq P$ et $F \neq \emptyset$. Une face propre F est une *facette* de P si $dim(F) = dim(P) - 1$.

Un *point extrême* d'un polyèdre P est une face de P de dimension 0. Il est facile de voir qu'un point $x \in I\!R^n$ est un point extrême d'un polyèdre P, s'il ne peut pas être écrit comme combinaison convexe d'autres points de P.

2.3.2 Approche polyédrale

Soient $\mathcal{P}$ un problème d'optimisation combinatoire, $\mathcal{S}$ l'ensemble des solutions de $\mathcal{P}$, E l'ensemble de base de $\mathcal{P}$ et c la fonction poids associée aux variables du problème. Le problème $\mathcal{P}$ s'écrit donc $max\{cx \mid x \in \mathcal{S}\}$.

Si T est un sous-ensemble de E, le vecteur $x^T \in I\!R^E$ (ayant $|E|$ composantes associées aux éléments de E) tel que

$$x^T(e) = \begin{cases} 1 & \text{si } e \in T, \\ 0 & \text{sinon,} \end{cases}$$

est appelé *vecteur d'incidence* de T. Le polyèdre

$$P(\mathcal{S}) = conv\{x^S \mid S \in \mathcal{S}\}$$

est appelé *polyèdre des solutions* de $\mathcal{P}$ (ou *polyèdre associé* à $\mathcal{P}$).

Le problème $\mathcal{P}$ est donc équivalent au programme linéaire $max\{cx \mid x \in P(\mathcal{S})\}$. $P(\mathcal{S})$ peut être caractérisé par un ensemble de contraintes linéaires où chaque contrainte définit une facette. Si on peut décrire entièrement le polyèdre $P(\mathcal{S})$ par un système d'inégalités linéaires, le problème $\mathcal{P}$ se ramène donc à la résolution d'un programme linéaire.

L'*approche polyédrale* consiste à ramener la résolution du problème à la résolution d'un (ou d'une séquence) de programmes linéaires. Cette transformation nécessite, par conséquent, une étude approfondie du polyèdre associé au problème. Néanmoins, la caractérisation complète de ce polyèdre est généralement difficile à établir (voire impossible si le problème est NP-difficile). De plus, le nombre de contraintes nécessaires pour décrire le polyèdre, est souvent exponentiel. Cependant, en utilisant une méthode de coupes et branchements (Branch-and-Cut method), une description partielle peut être suffisante pour résoudre le problème à l'optimum. Cette méthode combine la méthode de génération de nouvelles contraintes (Cutting plane method) et la méthode de *séparations et évaluations* (Branch & Bound method). Dans la suite, nous discutons de cette méthode.

2.3.3 Méthode de coupes et branchements

La méthode de coupes et branchements pour un problème d'optimisation combinatoire est basée sur le problème dit *de séparation*. Soit P un polyèdre dans $I\!R^n$. Le *problème de séparation* associé à P consiste à vérifier pour un point $x^* \in I\!R^n$ si x^* appartient à P, et dans le cas contraire, à trouver une contrainte $a^T x \leq b$ valide

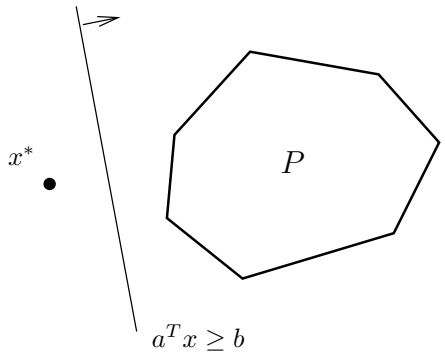

FIGURE 2.4 – Hyperplan séparant x^* et P

pour P et violée par x^*. Dans ce deuxième cas, l'hyperplan $a^T x = b$ sépare P et x^* (voir la figure 2.4).

Une des grandes avancées de l'optimisation combinatoire est le parallèle entre séparation et optimisation. Grötschel, Lovász et Schrijver [40] ont montré qu'un problème d'optimisation sur un polyèdre P est polynomial si et seulement si le problème de séparation associé à P peut être résolu en temps polynomial.

Considérons un problème d'optimisation combinatoire $\mathcal{P}$ de la forme $max\{cx \mid Ax \leq b, x \text{ entier}\}$ et soit P le polyèdre associé à $\mathcal{P}$. Supposons que l'on dispose d'un système $\bar{A}x \leq \bar{b}$ de contraintes valides pour $\mathcal{P}$ et qui contient comme sous-système les contraintes de base du problème. La méthode de coupes et branchements commence par résoudre un programme linéaire $P_1 = max\{cx \mid \bar{A}_1 x \leq \bar{b}_1\}$ où $\bar{A}_1 x \leq \bar{b}_1$ est un sous-système de $\bar{A}x \leq \bar{b}$ contenant un nombre raisonnable de contraintes. Si la solution optimale disons x_1 de P_1 est entière et solution de $Ax \leq b$, alors elle est optimale pour $\mathcal{P}$. Sinon, on résout le problème de séparation associé à $\bar{A}x \leq \bar{b}$ et x_1. Si l'on trouve une contrainte $a_1 x \leq \alpha_1$ violée par x_1, elle est rajoutée au système $\bar{A}_1 x \leq \bar{b}_1$ et nous obtenons un nouveau programme linéaire $P_2 = max\{cx \mid \bar{A}_1 x \leq \bar{b}_1, a_1 x \leq \alpha_1\}$. On résout P_2. Si la solution optimale x_2 de P_2 est entière et solution de $Ax \leq b$, comme pour x_1 on résout le problème de séparation associé à $\bar{A}x \leq \bar{b}$ et x_2. Si $a_2 x \leq \alpha_2$ est une contrainte violée par x_2, elle est rajoutée au système $\bar{A}_2 x \leq \bar{b}_2$ et ainsi de suite. En continuant ce processus appelé *phase de coupe*, nous pouvons trouver, soit une solution optimale pour $\mathcal{P}$, soit une solution x^* qui soit fractionnaire et pour laquelle nous ne pouvons pas générer de contrainte violée. Dans ce cas, nous commençons une phase dite de *branchement* qui consiste à construire un arbre de Branch & Bound. Nous choisissons une variable fractionnaire x_i^*. Nous résolvons deux nouveaux programmes linéaires (nouveaux sommets de l'arbre) en ajoutant soit la contrainte $x_i = 0$, soit $x_i = 1$ au programme linéaire obtenu à la fin de la phase de coupe. Pour chacun de ces nouveaux sommets,

nous appliquons une procédure de coupes. Si une solution optimale de $\mathcal{P}$ n'a pas été trouvée, nous sélectionnons une feuille de l'arbre et nous recommençons une phase de branchement.

Cette approche est maintenant largement utilisée pour les problèmes d'optimisation combinatoire difficiles. La méthode de coupes et branchements peut être utilisée lorsque le nombre de variables du problème n'est pas très grand (polynomial). Par contre, si le problème comporte un nombre exponentiel de variables, la méthode de génération de colonnes et branchement est plus appropriée pour approcher le problème. La section suivante présente les grandes lignes de cette méthode.

2.4 Notations et Définitions de la thèorie des graphes

Un *graphe non orienté* est noté $G = (V, E)$ où V est l'ensemble de *sommets* et E l'ensemble des *arêtes*. Si e est une arête reliant deux sommets u et v, alors u et v seront appelés les *extrémités* de e, et nous écrirons $e = uv$ ou $e = \{u, v\}$. Si u est une extrémité de e, alors u (resp. e) est dit *incident* à e (resp. u). De même, deux sommets u et v formant une arête sont dits *adjacents*.

Un *graphe orienté* est noté $D = (V, A)$ où V est l'ensemble de *sommets* et A l'ensemble des *arcs*. Si a est un arc allant du sommet u au sommet v, alors u et v seront appelés respectivement le *sommet origine* et le *sommet destination* de a, et nous écrirons $a = (u, v)$.

Soit $G = (V, E)$ un graphe non orienté. Si $F \subseteq E$ est un sous-ensemble d'arêtes, alors $V[F]$ représente l'ensemble des extrémités des arêtes de F. Si $W \subseteq V$ est un sous-ensemble de sommets, alors $E[W]$ denote l'ensemble des arêtes ayant leurs deux extrémités dans W.

Soit $W \subseteq V$, $H = (W, E(W))$ est dit sous-graphe de G *induit* par W et sera noté par $G[W]$.

Soient u et v deux sommets de V. Une *chaîne* (resp. *un chemin*) P entre u et v (resp. allant de u à v) est une séquence alternée de sommets et d'arêtes (resp. d'arcs) $(v_0, e_1, v_1, e_2, v_2, \ldots, v_{k-1}, e_k, v_k)$ (resp. $(v_0, a_1, v_1, a_2, v_2, \ldots, v_{k-1}, a_k, v_k)$) où $v_0 = u$, $v_k = v$, $e_i = v_{i-1}v_i$ (resp. $a_i = (v_{i-1}, v_i)$) pour $i = 1, \ldots, k$ et $v_0, \ldots, v_k$ sont des sommets distincts de V. S'il n'y a pas de confusion possible, on peut également noter P par sa séquence d'arêtes $(e_1, \ldots, e_p)$ (resp. sa séquence d'arcs $(a_1, \ldots, a_p)$) ou sa séquence de sommets $(v_1, \ldots, v_{p+1})$. Deux chaînes entre u et v sont dites *arête-disjointes* (resp. *sommet-disjointes*) s'il n'existe pas d'arête (resp. de sommet différent de u et v) apparaissant dans les deux chaînes.

Un *cycle* est un chemin dont les deux extrimités sont identiques.

Un graphe G est *connexe* si, pour toute paire de sommets u, v de V, il existe au moins une chaîne entre u et v.

Dans un graphe $G = (V, E)$, un *arbre* est un sous-graphe connexe et sans cycle.

Si $F \subset E$, on notera par $G\backslash F$ le graphe obtenu à partir de G en supprimant les arêtes de F. Si F est réduit à une seule arête e, nous écrirons $G\backslash e$ au lieu de $G\backslash\{e\}$.

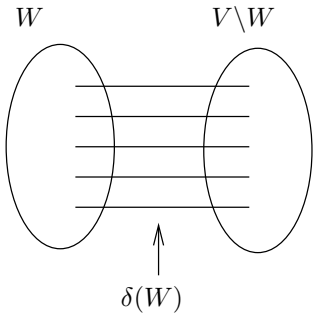

FIGURE 2.5 – Une coupe $\delta(W)$

Soit $W \subseteq V$, $W \neq \emptyset$, un sous-ensemble de sommets de V. L'ensemble des arêtes ayant une extrémité dans W et l'autre dans $V\backslash W$ est appelé *coupe* et noté $\delta(W)$. En posant $\overline{W} = V\backslash W$, nous avons $\delta(W) = \delta(\overline{W})$. Si W est réduit à un seul sommet v, nous écrirons $\delta(v)$ au lieu de $\delta(\{v\})$.

Etant donnés W et W' deux sous-ensembles disjoints de V, alors $\delta_G(W, W')$ représente l'ensemble des arêtes de G qui ont une extrémité dans W et l'autre dans W'. Lorsque W et W' sont réduits à un seul sommet, on notera $[u, u']$ à la place de $\delta_G(\{u\}, \{u'\})$.

Si $V_1, \ldots, V_p$, $(p \geq 2)$ est une partition de V, alors $\delta(V_1, \ldots, V_p)$ représente l'ensemble des arêtes ayant leurs extrémités dans des éléments différents de la partition (*i.e* $e = uv$ telles que $u \in V_i$, $v \in V_j$ et $i \neq j$).

Chapitre 3

L'optimisation dans Les bases de données

Sommaire

3.1 Concepts de bases de données

Les problèmes étudiés dans cette thèse sont issues des bases de données. Dans cette section, nous nous proposons de donner quelques notions sur les bases de données ainsi que leur traitement en se basant pricipalement sur la référence [38]. Pour des définitions plus détaillées l'auteur peut se référer aux livres [27, 38, 41, 49].

3.1.1 Introduction aux bases de données

Avec l'invention des disques durs à la fin des années 50, les ordinateurs ont été utilisés pour collecter, classer et stocker de grandes quantités de données. Dans les

années 60, un nouveau terme désignant une collection d'informations partagées par différents utilisateurs d'un système d'informations militaire est apparu, à savoir le terme database (base de données). Ce terme a par la suite, pris une grande ampleur dans le domaine informatique. En effet, au cours des quarantes dernières années des concepts, méthodes, modèles et algorithmes ont été développés pour concevoir et gérer les dites bases de données.

3.1.1.1 Description

Une *base de données* (*database*) est un ensemble de données modélisant les objets d'une partie du monde réel et servant de support à une application informatique. Plus précisément, il s'agit d'un conteneur informatique permettant de stocker des données souvent non indépendantes dans le but de les traiter et en extraire des informations pertinentes.

L'utilisation d'une base de données sert à collecter de grandes quantités de données qui doivent être facilemet interrogeables et modifiables. En effet, une base de donnée doit permettre une gestion sûre d'accès fréquents et multiples aux données. Elle doit aussi permettre de fournir des informations cohérentes et synthétiques à partir de renseignements bruts.

3.1.1.2 Terminologie

Dans cette section, nous présentons quelques définitions de termes fréquemment utilisées dans les bases de données.

Un *Modèle de description de données* est l'ensemble de concepts et de règles de composition de ces concepts, permettant de décrire des données. On distingue deux types de modèle de données, le modèle de données *logique* et le modèle de données *physique*. Un modèle logique ou conceptuel est la description des données telles qu'elles le sont dans la pratique. Le modèle physique est un modèle dérivé du modèle logique qui décrit comment les données seront techniquement stockées dans la base de données.

Une *Table* est un ensemble de données organisées sous forme d'un tableau où les colonnes correspondent à des catégories d'information (une colonne peut stocker des noms de personnes, des prénoms...) et les lignes à des enregistrements. Une table peut être aussi défini comme un ensemble de données relatives à un même concept, ou permettant de lier d'autres tables.

Une *Colonne* est un élément vertical dans une table représentant un ensemble de valeurs d'un attribut.

Une *Ligne* est un élément horizontal dans une table représentant une énumération des valeurs des différents attributs (colonnes) pour une même référence (identifiant).

Une *Clef primaire* est un identifiant d'une table composée d'un ou plusieurs attributs ou colonnes. Une clef permet de retrouver sans ambiguïté la ligne dans une table. Les valeurs d'une clef doivent être uniques au sein de la table.

Une *Clef étrangère* est une colonne, ou groupe de colonne représentant une clef primaire d'une autre table afin d'assurer la relation entre les deux tables.

Une *Entité* est un ensemble d'éléments informatif relatif à un même concept dans un modèle. Correspond généralement à une table dans l'univers des bases de données.

Un *Identifiant* est un autre nom du concept de clef primaire.

Une *Intégrité référentielle* est un mécanisme assurant la cohérence des dépendances de données entre les différentes tables.

Une *contrainte* est un mécanisme de contrôle de validité des données interdisant l'insertion de données violant les contraintes.

Un *index* est une construction physique d'une structure de données relative à une ou plusieurs colonnes d'une table, permettant d'accélérer les recherches au sein d'une table.

Une *association* est un lien logique entre entités dont le type est défini par un verbe et une liste éventuelle de propriétés.

Un *attribut* est une propriété d'une entité ou d'une association caractérisée par un nom et un type élémentaire.

Un *enregistrement* est une donnée composite qui comporte plusieurs attributs où sont enregistrées les données.

Une *violation* (de contrainte, de clef) est une erreur générée lorsque l'on tente d'insérer une ligne avec une valeur de clef déjà utilisée (*violation de clef*), ou pour des valeurs de colonne hors des contraintes définies (*violation de contrainte*).

Pour gérer efficacement de grandes bases de données, des logiciels appelés *SGBD* (Systèmes de Gestion de Bases de Données sont apparus).

3.1.2 Système de gestion de bases de données

Un *Système de Gestion des Bases de Données* est un ensemble de programmes et logiciels systèmes permettant de stocker et d'interroger un ensemble de données interdépendants. Dans le cas d'une entreprise, il s'agit d'un outil qui permet de modéliser et de gérer les données de cette entreprise.

Parmi les objectifs et fonctions des SGBD on peut citer :
- assurer l'indépendance physique et logique des programmes aux données,
- permettre un usage multiple des données,
- contrôler la cohérence, l'exactitude et la redondance de données,
- protéger et sécuriser les données,
- réduire le coût de stockage, de mise à jour et de saisie des données,
- garantir un accès efficace, rapide et protégé aux données.

Pour assurer ce dernier objectif, et afin de retrouver rapidement les données, plusieurs SGBD utilisent une structure de données appropriée appelée *index*. Dans la pratique, l'utilisation d'un index simplifie et accélère les opérations de recherche, de tri, de jointure ou d'agrégation effectuées par le SGBD.

Aussi, un SGBD doit garantir la cohérence des données lors des mises à jour de la base. Cette propriété s'appelle *contrainte d'intégrité*.

3.1.3 Modèle relationnel des bases de données

Depuis l'apparition des bases de données, leurs structures, conceptions et modèles ne cessent de se développer. L'un des modèles les plus couramment utilisés pour la réalisation des bases de données est *le modèle relationnel*. Dans un modèle relationnel, une base de données est composée d'un ensemble de *tables* appelées aussi *relations*, contenant des données et des liens. Avec l'apparition des modèles relationnels dans les bases de données, une nouvelle terminologie a émergé.

Dans les modèles de données relationnels, une *clé primaire* est un attribut unique et spécifique à chaque enregistrement (ligne) de la table. Toujours concernant les mêmes modèles, une *clé étrangère* est un attribut qui contient une référence à une donnée connexe.

La définition d'un modèle relationnel implique l'apparition de nouvelles contraintes telles que la contrainte d'*intégrité référentielle*. L'intégrité référentielle est assurée lorsque toutes les données référencées par les clés étrangères sont présentes dans la base de données.

Dans la pratique les entreprises disposent d'une surabondance de données. Celles-ci présentent une mine d'informations qui doivent être bien triées, classées et interprétées. Pour ce faire, il a été nécessaire de mettre en place une nouvelle informatique décissionnelle.

3.1.4 Les bases de données et le décisionnel

Pour obtenir une meilleur compréhension des données, définir des indicateurs business pertinents et faciliter la prise de décisions opérationnelles, l'informatique décisionnelle avait pour rôle d'intégrer une nouvelle architecture. Cette architecture est dite le *Data Warehouse* (*entrepôt de données*).

3.1.4.1 Entrepôts de données

La définition classique du Data Warehouse donnée par Inmon and Hackathorn dans [43] est la suivante : *le data warehouse est une collection de données orientées sujet, intégrées, non volatiles et historisées, organisées pour le support d'un processus d'aide à la décision.*

Le terme entrepôt de données (ou base de données décisionnelle) désigne une base de données utilisée pour regrouper, ordonner et stocker des informations provenant de bases de données opérationnelles d'une entreprise. Son objectif est de fournir un ensemble de données servant de référence unique, utilisée pour la prise de décisions dans l'entreprise à travers des analyses de données, des études statistiques, etc... La conception d'entrepôts de données est un processus en perpétuelle évolution. Dans ce contexte, on peut voir l'entrepôt de données comme une architecture décisionnelle capable d'une part de gérer l'hétérogénéité et le changement des données, et d'autres part de transformer les données en informations directement exploitables par les utilisateurs.

En conclusion, les fonctions essentielles réalisées par les entrepôts de données comme résumées dans [38] sont :
- la collecte de données à partir de bases existantes et chargement de l'entrepôt,
- la gestion de l'organisation des données dans l'entrepôt,
- l'analyse de données pour la prise de décision en interaction avec les anlystes.

3.1.4.2 Fouille de données

La *fouille de données* (*data mining*), connue aussi sous le nom de *forage de données* ou *prospection de données*, a pour objet l'extraction de connaissances à partir

de grandes quatités de données. Il s'agit d'un ensemble de techniques d'exploration de larges bases de données afin d'en tirer les liens sémantiques significatifs et plus généralement des règles et des modèles pour la compréhension et l'aide à la décision [38]. L'utilisation opérationnelle du data mining dans le monde professionnel permet de résoudre des problèmes divers, allant de l'analyse de risque, au contrôle de qualité passant par l'analyse financière ou encore la gestion de stocks, le domaine médical, etc...

L'approche consiste souvent à induire des règles avec des coefficients de vraisemblance à partir d'un large ensemble de données. Les techniques de base sont issues de l'intelligence artificielle et de l'analyse de données (analyse statistique, modèles fonctionnels, réseaux de neurones, classification, segmentation, etc...).

3.2 Bases de données d'Altares

En octobre 2004, la société BIL (Base d'Informations Légales), créée en 1987, acquiert D&B France qui est la filiale française de D&B. D&B, appelée aussi Dun & Bradstreet, est le leader mondial dans le domaine des informations sur les entreprises. Les deux entités, BIL et D&B France vont alors mettre en place une organisation commune, et devenir BIL/D&B France. Dans un souci de lisibilité, BIL/D&B France décide de changer de nom, et d'adopter l'appellation Altares D&B en février 2006. Cette nouvelle société est aujourd'hui membre du réseau Dun & Bradstreet. L'appartenance à ce réseau, qui est constitué d'une entreprise par pays, lui permet de bénéficier des données recueillies par les autres entreprises membres, et d'occuper ainsi une place sur le marché du renseignement international. Le réseau D&B couvre les informations sur 230 millions d'entreprises dans 220 pays différents.

Dans cette section, nous commençons par présenter l'activité d'Altares. Ensuite nous décrivons les sources des données qui alimentent les bases de données d'Altares. Enfin, nous présentons les différentes bases de données qui constituent le nouveau système d'information d'Altares.

3.2.1 Le métier d'Altares

Altares est un acteur indépendant qui consacre toute son activité à l'information sur les entreprises. La société emploie ainsi son savoir-faire à récupérer toutes les informations qui concernent l'ensemble des entreprises françaises, à s'assurer de leurs qualités et de leurs fraicheurs et à les communiquer le plus rapidement possible aux clients d'Altares.

Ces données couvrent les informations légales, pour permettre à chaque client d'Altares de suivre l'évolution de ses partenaires, les informations financières, pour mieux analyser la solvabilité des entreprises. Les solutions proposées par Altares s'organisent autour de trois axes majeurs :

- Fiabiliser et enrichir les référentiels métiers en données à valeur ajoutée
- Optimiser les axes de développement commercial
- Mesurer et gérer le risque financier, clients ou fournisseurs

Altares fournit ainsi une vision 360° sur la vie et les événements jalonnant le parcours des entreprises et procure les informations pertinentes pour accroitre l'efficacité des prises de décisions. Il peut s'agir, par exemple, d'un accès aux données légales, financières et marketing, d'analyses de solvabilité, de surveillance ou d'évaluations du risque, le tout à l'échelle nationale et internationale.

Les solutions d'Altares peuvent être regroupées dans deux grandes catégories qui sont

- Gestion de risque : Les prestations de gestion du risque, expertise historique d'Altares, regroupent l'ensemble des solutions et outils pour minimiser les risques financiers inhérents au crédit interentreprises et améliorer les délais de règlement. Ces solutions se déclinent en interfaces web, traitements et transferts de fichiers ou solutions web services pour intégration dans les systèmes d'information des entreprises. Les accès en ligne d'Altares permettent d'identifier et de valider l'existence d'une entreprise, en France comme à l'international, et de consulter toutes les données disponibles la concernant.
- Un référentiel unique : La création d'un référentiel permet de structurer et d'organiser les données internes à une entreprise. Grâce à une clé unique d'enregistrement, le SIREN en France et le D-U-N-S® Number à l'international, ces données deviennent cohérentes, combinables entre elles et analysables. La source d'information est unique et transversale, facilement mise à jour en données structurelles et spécifiques. La mise à jour quotidienne est couplée à une collecte en amont des informations légales avant même leur publication au journal officiel BODACC.

Il est primordial pour Altares de connaitre, comprendre et évaluer les enjeux auxquels sont confrontées toutes les directions opérationnelles dans une entreprise. Comment travaillent-elles ? Quels sont leurs besoins ? Ces questions en apparence simples appellent en réalité des réponses multiples et variées à retranscrire dans les différentes communications. L'entreprise doit effectuer une approche structurée par métier afin de traduire les bénéfices que les différentes directions d'une entreprise peuvent retirer en utilisant les solutions Altares.

Dans la section suivante, nous présentons les sources des données qu'intégre Altares afin de répondre aux besoins de ses clients.

3.2.2 Les sources des données

La vocation d'Altares est d'accéder à la bonne information et de la rendre accessible à ses clients. Le souci constant en la matière est la qualité de ces informations. Cette qualité est caractérisée par l'exhaustivité, l'exactitude, la fraîcheur et la transparence. Altares, diffuseur agréée d'un ensemble de sources officielles, fédère, recoupe, diffuse et archive les informations des entreprises collectées depuis 1987.

Altares exploite ainsi la pleine richesse des informations officielles, légales et financières sur les entreprises françaises. Ces données sont stockées dans des bases de données propriétaires d'Altares permettent de répondre au besoin des clients d'Altares. Pour l'acquisition de ces données, Altares investit plus 5,5 millions d'euros chaque année. Altares est diffuseur agréé des sources légales françaises. Ces licences garantissent l'historique de la base de données, le rythme de mise à jour ainsi que l'exhaustivité des informations transmises.

Les données dont dispose Altares proviennent de trois type de source qui sont les sources officielles, fournisseurs privés des données et les collectes qu'effectuent Altares sur certaines données.

Les principales sources officielles de données sont :

- L'Institut National de la Statistique et des Etudes Economiques (INSEE) : l'INSEE centralise les informations identitaires des entreprises. Ces dernières sont légalement obligées de déclarer tout changement concernant leurs identité (nom, adresse, activité, forme juridique etc). L'INSEE propose plusieurs flux informatiques qui contiennent ces changements. Altares est abonné aux deux principaux flux que sont les mises à jour quotidiennes et les fichiers de calages.
- Le Registre National du Commerce et des Sociétés (RNCS) : Toutes les entreprises ne sont pas concernées par ce flux. En effet, celui-ci ne concerne que les entreprises commerciales et les commerçants. Ce flux est le seul à fournir les informations sur les dirigeants. Son contenu et son fonctionnement sont détaillés dans le chapitre suivant.
- Le Bulletin Officiel des Annonces Civiles et Commerciales (Bodacc) : depuis 1987, Altares reçoit quotidiennement les annonces publiées par le BODACC dans les éditions A, B et C. Le bodacc A est composé de deux sections. La première contient les informations de procédures collectives. La seconde comportent les annonces de créations, d'immatriculation et de vente et sessions des sociétés.

- Le Journal Officiel des Associations et des Fondations d'Entreprises (JOAFE) : Le JOAFE recense toutes les annonces publiées relevant de la vie des associations, des associations syndicales de propriétaires, des fondations d'entreprise et des fonds de dotation dans un registre appelé Registre National des Associations (RNA). Altares reçoit toutes les mises à jour et insertions concernant le registre RNA.

Pour compléter les sources officielles, Altares fait appel à des fournisseurs privées des données dont nous citons par exemple, les bilans reçus de la part de Bilan Service, les RIB reçus de la part de la Banque de France, les incidents de paiement reçus de la part de la société FactoFrance.

Enfin, et afin de compléter sa vision 360° sur les entreprises, Altares collecte elle-même certaines typologies des données dont nous citons par exemple : les annonces parues dans les journaux légales et les informations des liens capitalistiques. Notons que les informations des liens capitalistiques sont détaillées dans le chapitre 5.

En plus de toutes ces données reçues de la part des fournisseurs officiels, des fournisseurs privés et des données collectées par les équipes Altares, cette dernière calcule pour ces clients un ensemble des données à grande valeur ajoutée dont nous citons par exemple le score et les indicateurs. Le Score est une note qu'attribue Altares à chaque entreprise. Cette note correspond à la probabilité de défaillance des entreprises. Son algorithme de calcul est très complexe et fait intervenir l'ensemble des données dont dispose Altares mais aussi l'évolution de ces données. Les clients d'Altares se basent sur cette note pour prendre des décisions importantes concernant leurs clients et/ou fournisseurs.

Les deux défis majeurs d'Altares sont :

- La qualité des données : Altares, spécialiste de la « connaissance inter-entreprises » consacre l'essentiel de ses ressources et de son activité à l'intégration, la collecte, le calcul et la diffusion des données sur les entreprises. Au-delà des exigences quotidiennes en termes de maintien de la qualité des bases de données d'Altares et des informations qu'elles contiennent, les équipes d'Altares sont en permanence mobilisées autour de la question exhaustive et de qualité de données envoyées ou mises à disposition des clients.
- rapidité des traitements : chaque nuit Altares intègre des centaines de milliers des données reçues de la part des fournisseurs ou collectées par Altares. Selon le types de mise à jour, Altares calcule un ensemble d'indicateurs et de métadonnées qui doivent ensuite être envoyés ou mise à disposition des clients intéressés, le plus tôt possible. Les événements les plus sensibles doivent être envoyés par mails ou mise à disposition par web-service ou sous forme de fichier aux clients concernés. Les clients doivent recevoir leurs alertes et leurs fichiers avant 8h00 afin qu'ils puissent organiser le travail de leurs équipes concernées et prendre les décisions qu'ils jugent nécessaires.

3.2.3 Bases de données

Altares dispose de plusieurs bases de données. Certaines sont dédiées aux données opérationnelles, d'autres sont destinées aux consommations des clients et à la facturation et enfin d'autres sont consacrées aux analyses de données effectuées par les équipes d'Altares. Nous nous intéressons ici exclusivement aux bases de données dédiées aux données opérationnelles, c'est-à-dire les données que reçoivent, collectent ou calculent Altares et qu'elle met à la disposition de ses clients.

Altares a lancé, fin 2007, un projet de refonte du système d'information. Ce projet avait comme objectif de moderniser et fiabiliser le système d'information d'Altares qui était basé sur le mainframe AS400. Suite à ce projet, Altares a mis en place une architecture basée essentiellement sur des bases de données Oracle et des applications Java. Dans ce cadre, elle a créé trois grosses bases de données appelées : IWH (Infor WareHousse), Repository (le référentiel métier d'Altares) et DTM (qui est orienté besoin clients).

Le schéma 3.1 donne une vision simplifiée de l'alimentation de ces bases de données opérationnelles d'Altares. Les données que reçoivent ou collectent Altares sont stockées dans IWH. Cet entrepôt de données sert à conserver les données telles qu'elles étaient envoyées par les sources. Cette traçabilité permet à Altares de répondre aux réclamations clients et de remonter jusqu'à la source qui a envoyé l'information erronée. Ensuite, une application Java valide la qualité de ces données et les intègrer dans Repository. Ce dernier sert à alimenter les différents Datamart en fonction des règles des clients.

Les bases de données d'Altares sont gigantesques. Chacune de ces trois bases de données nécessitent un stockage d'environ 1,5 Tera Octet. Par exemple DTM est composé d'environ 5300 objets dont 1000 tables. La taille moyenne de chaque table est de dizaine de millions d'enregistrements. Une des tables de DTM contient plus de 2 milliards d'enregistrements. Vu leurs tailles, le traitement de ces bases de données doit être optimisé afin de répondre aux exigences d'Altares et de ces clients. D'autre part, la qualité de ces bases de données doit être la meilleure possible afin que les décisions prises à la base des informations, que contiennent ces bases de données, soient fiables et efficaces.

La section suivante présente quelques exemples d'utilisation de la recherche opérationnelle et de l'optimisation combinatoire pour répondre à des problématiques des bases de données.

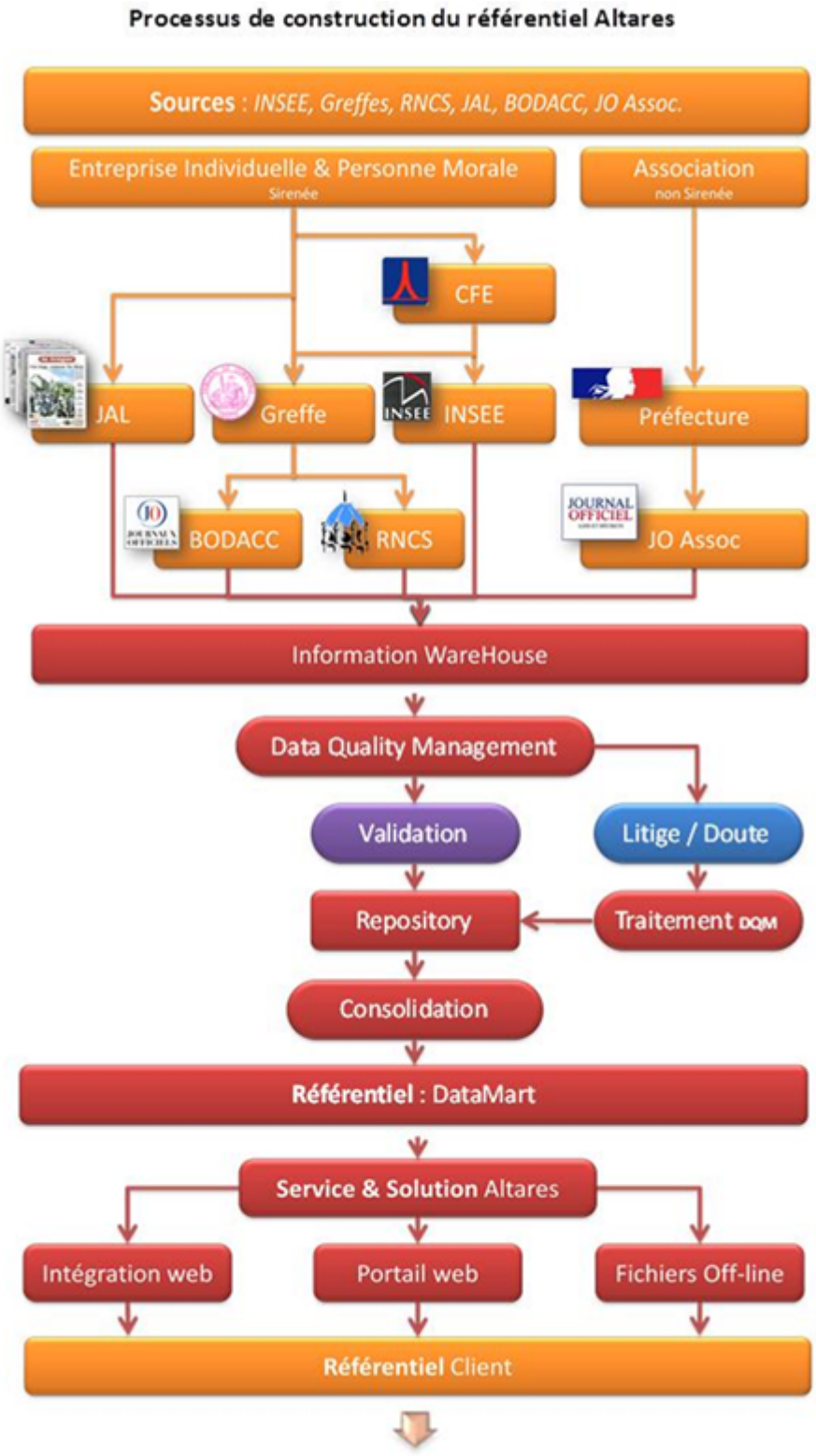

FIGURE 3.1 – Les bases de données d'Altares : rôle et alimentation

3.3 La recherche opérationnelle dans les bases de données

Plusieurs techniques de la recherche opérationnelle sont utilisées pour la résolution de certains problèmes d'optimisation dans les bases de données. L'une des applications directes de la recherche opérationnelle est l'optimisation en terme de temps de calcul et de complexité des requêtes (*query optimisation*).

3.3.1 Les requêtes dans les bases de données

Une requête est une expression qui décrit des données à récupérer à partir d'une base de données. Il s'agit d'une interrogation de la base de données qui reçoit comme réponse des éléments tirés de cette base. L'un des problèmes cruciaux dans les bases de données est l'extraction des données nécessaires en un temps minimal et avec une complexité optimale.

3.3.1.1 Optimisation des requêtes

Le problème d'optimisation des requêtes dans les bases de données a suscité l'intérêt des chercheurs depuis longtemps. Jarck et Koch étaient parmi les premiers à écrire un survey sur cette problèmatique [46]. Ensuite, les travaux se sont succédés pour une investigation optimale des données. Ces travaux étaient en majorité basés sur les techniques d'optimisation et de recherche opérationnelle. Par exemple, Bennet et al. [9] et Stillger et al. [85] ont utilisé les algorithmes génétiques pour l'optimisation des requêtes dans les bases de données. D'autres approches heuristiques et métaheuristiques ont aussi été utilisées. Dans [87] et [44], les auteurs ont traité le problème d'optimisation de requêtes de jointure en utilisant les méthodes de recherche locale et de recuit simulé. Dans le même contexte, Swami [86] propose une combinaison de certaines heuristiques avec les techniques d'optimisation combinatoire pour une meilleure résolution du problème.

Le problème d'optimisation des requêtes devient d'autant plus complexe que les données sont incohérentes dans la base de données.

3.3.1.2 Requêtes dans les bases de données incohérentes

Une base de données est dite *incohérente* si elle viole une ou plusieurs contraintes d'intégrité. Les incohérences surviennent sous plusieurs circonstances. Par exemple,

des erreurs de saisie peuvent être involontairement introduites ou encore une même information peut être écrite différemment dans la même base de données. Les incohérences peuvent également survenir lors de la combinaison de données provenant de différentes sources qui représentent les mêmes informations de manière différente. Par conséquent, l'optimisation des requêtes dans de telle base passe par une phase appelée de *nettoyage* (*cleaning*), une problèmatique à laquelle nous nous intéressons dans le Chapitre suivant.

Dans [62], les auteurs proposent une approche efficace pour mener des requêtes dans les bases de données incohérentes. L'approche se base sur une stratégie pertinente de réparation des bases de données à l'issue de laquelle on obtient une base de données cohérente. Les auteurs montrent que la base obtenue leur permet d'effectuer tout type de requête en temps polynomial. Pour résoudre le même problème, Kolaitis et al. [50] ont eu recours à la programmation en nombres entiers. Ils développent des algorithmes basés sur une formulation en $0-1$. Ces algorithmes ont prouvé de bonnes performances pour des requêtes conjonctives sous contraintes de clé primaire et ont surpassé de manière significative les résultats obtenus par les systèmes existants.

L'efficacité de l'interrogation d'un entrepôt de données est fortement liée à sa conception physique. Cette conception repose sur la sélection d'index pertinents, un problème dit de *sélection d'index* (*Index Selection Problem)* [6].

3.3.1.3 Problème de sélection d'index

Le problème de sélection d'index tel qu'il est défini dans [5] consiste à construire une configuration d'index optimisant le coût d'exécution d'une charge donnée. Cette optimisation peut être réalisée sous certaines contraintes, comme l'espace de stockage alloué aux index à sélectionner. Chaque index construit (sélectionné) nécessite un temps de maintien et chaque requête peut être effectuée dans un temps de réponse qui dépend de l'ensemble des index construits. L'objectif principal est donc de minimiser le temps global d'exécution défini comme la somme des temps de maintien plus la somme des temps de réponse à toutes les requêtes. Le problème de sélection d'index est l'un des problèmes les plus importants dans la conception physique des bases de données. Il a été montré NP-complet dans [25]. De ce fait, plusieurs travaux l'ont résolu de façon approchée. Choenni et al. [21] ont résolu le problème par des méthodes heuristique mais en adoptant une idée intelligente. Cette idée consiste à diviser le problème en deux sous-problèmes : le problème de sélection d'index primaire et le problème de sélection d'index secondaire.

Dans [51], les auteurs ont opté plutôt pour un algorithme génétique afin de résoudre ce problème. Les résultats expérimentaux de l'algorithme génétique ont été

menés sur des instances standard du problème de sélection d'index et comparés aux résultats obtenus par un algorithme de Branch-and-Cut ainsi que d'autres résultats de la littérature obtenus par CPLEX et OSL. Ces résultats montrent que l'algorithme génétique est efficace et robuste.

Papadomanolakis et Ailamaki [75] se sont à leur tour intéressés à ce problème, mais en optant plutôt pour les méthodes de résolution exacte. Dans [75], les auteurs proposent une fomulation en programme linéaire en nombre entiers (PLNE) du problème de sélection d'index. Cette formulation leur a permis d'utiliser les techniques d'optimisation combinatoire afin d'assurer une meilleure résolution du problème. En se basant sur la formulation en PLNE, les auteurs ont présenté dans le contexte des systèmes de bases de données commerciales un outil de sélection efficace. Cette approche leur a permis d'une part d'avoir une meilleure qualité de la solution, et d'autre part de garantir l'efficacité et l'évolutivité, sans pour autant sacrifier la précision qu'offre les outils de sélection existants.

En plus de l'optimisation des requêtes, la recherche opérationnelle est largement utilisée pour l'analyse et la fouille de données (*data mining*). En effet, il existe une grande intéraction entre le domaine de data mining et la recherche opérationnelle. Dans la suite, nous détaillons d'avantage cette intéraction.

3.3.2 Recherche opérationnelle pour la fouille de données

Grâce aux progrès technologiques récents dans les différents domaines, les chercheurs font face de plus en plus à des défis pour extraire des informations pertinentes à partir d'énormes volumes de données. L'optimisation combinatoire et plus généralement la recherche opérationnelle ont toujours constitué des outils efficaces pour le traitement et l'exploitation des bases de données. Ces outils interviennent dans tout le processus du data mining en commençant par les préprocessing et exploration des données vierges, passant par la classification et le partitionnement des données jusqu'à l'analyse finale des relations et intéractions des partitions de données. Une étude approfondie de ces étapes ainsi que l'importance de la recherche opérationnelle comme un outil efficace à leurs réalisations sont détaillées dans le survey [73]. Dans la suite, nous nous limitons à la présentation du problème de partitionnement et nous nous attardons sur l'efficacité de la programmation mathématique pour l'étude des problèmes de data mining. Pour plus de détails le lecteur peut se référer à [16] et [73].

3.3.2.1 Problème de clustering

Pour analyser les données, il est souvent souhaitable de les partitionner selon des critères de similarité, un problème connu sous le nom de *clustering*. Le problème de partionnement consiste à diviser les données en groupes en se basant sur un critère spécifique. Il constitue un outil puissant pour la découverte des connaissances dans les bases de données, plus connue sous le nom *Knowledge Discovery in Databases* (KDD).

Dans [78], les auteurs présentent une étude bibliographique détaillée sur les méthodes, les techniques et les outils de partitionnement dans les bases de données. Ils discutent en particulier de la différence entre le format de stockage de données et celui que les algorithmes de clustering utilisent en entrée, et proposent un modèle unifié pour les deux. Dans [52], les auteurs se basent sur la théorie des graphes pour résoudre le problème de clustering sur des données de grande dimension. D'autres travaux se sont plutôt inspirés des métaheuristiques. Mohamadi et al. [61] ont combiné la métaheuristique de recuit simulé (Simulated Annealing) avec les techniques des systèmes flous (Fuzzy Systems) pour résoudre le problème de classification des données. Dans [89], Tsai et al. propose un algorithme de classification basée sur une autre métaheuristique à savoir la métaheuristique de colonie de fourmies (Ant Colony). Ils montrent que leur algorithme permet de trouver des partitions efficaces pour des bases de données de taille importante. Dans [54], les auteurs utilisent les algorithmes évolutionnaires pour le partitionnement de bases de données en groupes en se basant sur des critères spécifiques. Pour une étude détaillée sur le problème de partitionnement dans les bases de données, le lecteur peut se référer au survey [10].

En étudiant un ensemble de données, il est possible et parfois nécessaire d'analyser non seulement les propriétés d'échantillons, mais aussi de leurs composants (généralement appelés attributs). Par conséquent, il est naturel de s'attendre à ce que chaque élément de la partition appelée aussi *cluster* soit à son tour partitionné en sous-ensembles. Le problème du partitionnement d'un cluster en sous-ensembles est appelé problème de *biclustering*. Pour plus de détails sur le problème du biclustering, le lecteur est référé au survey [18].

Dans la pratique, la technique du partitionnement a prouvé son efficacité pour le traitement de bases de données de très grande taille. Dans [95], Zahng et al. ont proposé une méthode efficace pour le clustering dans les bases de données de grande taille, appelée BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies). Ils prouvent que la méthode BIRCH est meilleure que tous les autres algorithmes utilisées pour le clustering dans les bases de données du point de vue du temps d'exécution, de l'espace mémoire utilisé mais aussi de la qualité de partitionnement.

3.3.2.2 Programmation mathématique et fouille de données

Des études récentes ont montré l'efficacité des algorithmes de factorisation des matrices non-négatives (Non-negative Matrix Factorization) pour le partitionnement des données. Dans [29], les auteurs proposent plusieurs algorithmes basés sur les techniques de factorisation des matrices non-négatives pour l'analyse et l'exploration des données. Ces algorithmes se basent sur le "multi-way normalized cut spectral clustering", le problème du couplage dans les graphes ainsi que le problème de clique maximum. Les auteurs montrent que ces algorithmes sont extrêmement simples à mettre en oeuvre et prouvent leurs efficacités sur le partitionnement des bases de données.

Les approches de programmation mathématique ont aussi été utilisées pour d'autres types de problèmes associés à l'analyse des données. Dans [57], Mangasarian utilise les méthodes puissantes de la programmation mathématique pour la résolution de trois problèmes fondamentaux dans le data mining. Le premier problème est le problème de sélection d'attributs (*feature selection*). Ce problème est utile pour la comparaison de deux ensembles de données en retenant juste les attributs les plus pertinents de la base des données. L'auteur a aussi étudié le problème de partitionnement des données (clustering) en utilisant les techniques de l'optimisation combinatoire. Plus particulièrement, il a proposé une formulation du problème et développé un algorithme de partitionnement basé sur les algorithmes de k-Median et k-Mean. Le dernier problème traité par Mangasarian dans son article [57] est le problème de présentation robuste (*robust representation*). Ce problème consiste à établir un système de relations dans la base de données de façon à ce que cette présentation soit toujours valide et fiable en cas de changement de données dans la base. Les méthodes de résolution proposées par l'auteur pour ces trois problèmes ont été tester sur une base de données médicales (Wisconsin Prognosis Breast Cancer) et ont permis de diminuer la marge d'erreurs pour le traitement des dossiers des patients.

D'autres travaux dans la littérature se sont aussi servis des approches de la programmation mathématique pour la résolution de problèmes déjà traités comme la classification et le clustering, mais aussi pour le traitement des données après partitionnement. Plus de détails peuvent être trouvé dans l'article [4].

3.3.3 Fouille de données pour la recherche opérationnelle

Paralèllement aux services fournis par la recherche opérationnelle pour la résolution des problèmes de data mining, cette dernière a aussi fourni un support efficace

pour la résolution de problème de la recherche opérationnelle. En effet, dans les dernières années, plusieurs travaux se sont servis des techniques de la fouille de données en les combinant avec des méthodes d'optimisation afin de résoudre des problèmes d'optimisation. Dans ce cadre, plusieurs travaux ont proposé une hybridation de certaines métaheuristiques en utilisant le process du data mining. Dans [79], Santos et al. ont proposé une hybridation d'un algorithme génétique pour la résolution du problème de tournées de véhicules. Ils ont montré que l'intégration du data mining a contribué à une meilleure résolution des instances en comparaison à l'algorithme génétique brut. Des méthodes d'hybridation similaires ont été proposées dans [32] pour la métaheuristique du Particle Swarm et dans [8] pour la métaheuristique Grasp.

D'autre part, le data mining a aussi été utile pour la résolution du problème de Job-Shop dynamique [83] et aussi pour un traitement efficace de l'aspect nonlinéare dans les problèmes issus de l'optimisation des systèmes électriques [65].

Chapitre 4

Problème de dé-duplication des dirigeants

Sommaire

4.1 Introduction

De nos jours, les bases de données jouent un rôle très important dans l'économie. De nombreuses industries et entreprises dépendent de l'exactitude de leurs bases de données pour effectuer des opérations plus ou moins sensibles et prendre des décisions importantes.

Les problèmes de qualité des données stockées dans les bases ou les entrepôts de données s'étendent aux domaines gouvernemental, commercial, industriel et scientifique.

La découverte des connaissances et la prise de décision à partir de données de qualité médiocre (c'est-à-dire contenant des erreurs, des doublons, des incohérences, des valeurs manquantes, etc.) ont des conséquences directes et significatives pour les entreprises et pour tous leurs utilisateurs.

Il devient de plus en plus courant dans les bases de données de trouver des entités représentées par deux ou plusieurs enregistrements. Ceci est principalement dû à l'une ou à la combinaison des raisons suivantes : erreurs de transcription, informations incomplètes, manque de format standard.

Les données dont dispose Altares concernent deux types d'entitées qui sont les entreprises et les personnes physiques (dirigeants, actionnaires,..). En France, les entreprises disposent d'un identifiant unique SIREN (Système Informatique du Répertoire des Entreprises) permettant d'identifier et de structurer leurs données de façon fiable et efficace. L'équivalent du SIREN pour les personnes physiques est le NIR Numéro d'Inscription au RNIPP (Répertoire National d'Identification des Personnes Physiques), communément appelé le numéro de sécurité sociale. Les craintes suscitées par la généralisation d'un identifiant national et unique qui rendrait plus aisées les possibilités de rapprochements de fichiers, ont conduit le législateur à encadrer strictement l'utilisation de ce numéro. En conséquence, la loi "Informatique et Libertés" a donc toujours soumis à des exigences procédurales particulières l'utilisation du NIR. Ce numéro n'est donc pas fourni à Altares par l'INPI qui n'a pas le droit de le stocker dans sa base de données et encore moins de l'envoyer dans un flux informatique.

Du fait de l'absence d'un identifiant unique pour les personnes, celles-ci sont identifiées grâce à leurs attributs : le nom, le prénom, la date de naissance et le lieu de naissance. Pour savoir si deux personnes sont identiques, il faut comparer leurs attributs deux à deux. C'est-à-dire le nom de la première personne avec le nom de la seconde, le prénom de la première avec le prénom de la seconde, la date de naissance de la première avec la date de naissance de la seconde et le lieu de naissance de la première personne avec le lieu de naissance de la seconde. Or trois de ces quatre attributs sont des chaines de caractères et sont donc susceptibles d'être considérés comme différente non pas parce qu'elles apportent des informations différentes mais à cause des fautes d'orthographe, des erreurs de saisie et de manque de normalisation.

Une étude réalisée sur la base de données BDD_DIR, qui contenait au départ 10 millions de personnes, avait montré que des nombreuses personnes ont été créées plusieurs fois. En conséquence, dans ce chapitre, nous nous intéressons au problème d'identification et d'élimination des doublons des dirigeants. Ce problème est appelé le problème de PDD (Problème de Déduplication des Dirigeants). La résolution de ce problème passe par la mise en place d'une méthode efficace de comparaison

des personnes. Cette méthode qui servira à identifier et éliminer les doublons des dirigeants, doit être intégré au processus de mises à jour des données pour éviter de créer des doublons.

La première section de ce chapitre est dédiée à la compréhension du problème et à ces motivations. Pour cela, nous commençons par décrire la source des données en expliquant les informations qui manquent à cette source. Puis, nous présentons la structure simplifiée de la base de données des dirigeants. Ensuite, nous donnons les motivations qui ont poussées à étudier le problème de PDD. Enfin, nous définissons le problème de PPD de façon plus formelle.

La deuxième section est destinée à l'état de l'art. Elle met l'accent sur les mesures de la similarité syntaxique. Dans un premier temps, nous présentons les méthodes d'encodage phonétique. Puis nous donnons quelques métriques qui permettent de mesurer la similarité syntaxique entre deux chaines de caractères ou entre deux documents. Enfin, nous présentons quelques logiciels de déduplication des bases de données.

La troisième et dernière section est consacrée à la méthode de résolution que nous avons proposée pour le problème PDD. Dans un premier temps, nous décrivons la normalisation des lieux de naissance ainsi que les raisons de la mise en place de cette normalisation. Puis nous présentons la méthode de comparaison proposée pour le nom et prénom. Cette méthode est basée sur des métriques de mesures de similarité syntaxique. Enfin, nous donnons les résultats obtenus sur les bases de données d'Altares.

Notations et Définitions

Notations

On note par S, l'ensemble des sociétés (entreprises). Dans les bases de données d'Altares, nous disposons d'environ 14 millions de sociétés.

L'ensemble des identités des personnes physiques disponibles dans notre base de données est noté par P. Deux identitées appartenant à P peuvent désigner la même personnes.

Soient p_i et p_j deux personnes appartenant à P telque $i \neq j$, on considère que p_i et p_j sont égaux si et seulement si p_i et p_j désigne une seule et même personne. Cette personne est représentée par deux entregistrement : on parle donc de *doublons*.

Les personnes physiques présentes dans notre base de données, sont les dirigeants et les actionnaires. Au début de notre travail, la base de données BDD_DIR contenait environ 10 millions de personnes. Dans ce chapitre, nous présentons une méthode de déduplication des enregistrements qui représentent les personnes. Cette méthode a été appliquée sur la base de données des dirigeants d'Altares. Cela a permis de réduire considérablement la taille de cette base de données et d'améliorer sa qualité.

Définitions

Le travail présenté dans ce chapitre se base sur des techniques de comparaison de chaîne de caractère. Nous donnons, dans cette section, quelques définitions qui seront nécessaires pour la compréhension de ce travail.

Définition 1 *Un mot vide est un mot ayant un faible (ou n'ayant pas) de contenu informatif.*

Par exemple, les déterminants (Le, La ....) sont des mots vides.

Définition 2 *Soient C_1 et C_2 deux chaînes de caractères,*

- *C_1 et C_2 sont identiques si et seulement si $C_1 = C_2$. Dans ce cas on dit aussi que C_1 et C_2 sont parfaitement similaires.*
- *C_1 et C_2 sont parfaitement différentes si et seulement si elles n'ont aucun caractère commun*
- *C_1 et C_2 sont syntaxiquement similaires, si elles s'écrivent de façon proche. Nous verrons par la suite qu'il existe plusieurs métriques pour mesurer cette similarité syntaxique. Nous en présentons quelques unes dans ce chapitre.*

Par exemple Paris et Lyon sont deux chaînes de caractère parfaitement différentes. Chateau et Chatou sont deux chaînes de caractères similaires.

Définition 3 *En droit français, une* **personne morale** *est un groupement doté de la personnalité juridique. Généralement une personne morale se compose d'un groupe de personnes physiques réunies pour accomplir quelque chose en commun. Ce groupe peut aussi réunir des personnes physiques et des personnes morales. Il peut également n'être constitué que d'un seul élément. La personnalité juridique donne à la personne morale des droits et des devoirs.*

Le droit français distingue :

- *Les personnes morales de droit public : l'État, les collectivités territoriales, les établissements publics... ;*
- *Les personnes morales de droit privé : les plus courantes étant les entreprises, les sociétés civiles, les groupements d'intérêt économique, les associations. Certaines personnes morales de droit privé sont chargées de la gestion d'un service public.*

Définition 4 Les dirigeants *d'une entreprise sont les personnes physiques ou morales qui dirigent, gèrent et représentent légalement l'entreprise. La définition de dirigeants recouvre tous les organes de gestion (président du conseil d'administration, conseil d'administration, gérants, directeurs généraux, ...) mais aussi les représentants légaux d'une entreprise.*

Le rôle d'un dirigeant est d'assurer la rentabilité économique de l'entreprise dans l'idée de constituer des fonds propres pour l'avenir et rémunérer le risque pris par l'actionnaire.

Il détient aussi une responsabilité sociale, devant ainsi développer l'emploi et la richesse sociale de l'entreprise.

4.2 Base de données BDD_DIR

Altares dispose des données sur les dirigeants d'entreprises françaises dont l'historique remonte à 1998. Les données des dirigeants sont stockées dans la base de données référentielle d'Altares appelée *Repository*. Il s'agit d'une gigantesque base de données qui contient toutes les informations métiers d'Altares. Cette base de données contient plus de deux cents tables ayant en moyenne une taille de dizaines de millions de ligne. La plus grosse table contient environ 2 milliards de ligne. Cette base de données est structurée par source de données et par métier. Dans la suite de ce chapitre, nous désignons par *la Base De Données des DIRigeants BDD_DIR*, l'ensemble des objets de Repository dédiées aux données des dirigeants.

Dans cette section, nous présentons la source qui alimente BDD_DIR ainsi que son Modèle Physique de Données. Puis nous donnons les motivations du problème de déduplication des dirigeants. Enfin, nous définissons ce problème de façon plus détaillée.

4.2.1 Source des données

Rappellons que la BDD_DIR contient les informations des dirigeants des entreprises françaises (personnes morales et commerçants). Les personnes morales (sociétés, GIE...) et les commerçants sont immatriculés au registre du commerce et des sociétés, tenu localement par les greffiers des tribunaux de commerce. Ces registres locaux sont consolidés dans le *Registre National du Commerce et des Sociétés (RNCS)* qui est tenu par l'*INPI* (Institut national de la propriété industrielle).

Le RNCS centralise et archive électroniquement les formalités déposées par les entreprises aux greffes, à savoir :

- toutes les inscriptions (immatriculations, modifications et radiations),
- les dépôts d'actes et les comptes annuels pour les personnes morales.

La mission de l'INPI est de centraliser et diffuser l'ensemble des informations du RNCS afin de les porter à la connaissance du public.

Le flux *IMR (Immatriculations, Modifications et Radiations)*, que reçoit électroniquement Altares de la part de l'INPI, contient toutes les mises à jour du RNCS : c'est-à-dire toutes les nouvelles déclarations concernant les sociétés de ce registre. Ce flux est un fichier plat de format fixe composé de 6 sections :

- **Données entreprise** : contiennent les informations sur les entreprises comme le SIREN, la raison sociale (le nom de l'entreprise), le NAF (la Nomenclature d'Activités Française), le capital social et la catégorie juridique de l'entreprise.
- **Données établissement** : donnent la liste des établissements et donne pour chaque établissement, l'adresse, l'activité et la nature de l'établissement (établissement secondaire ou établissement principal) ainsi que le NIC (le Numéro Interne de Classement) qui combiné avec le SIREN constitue un identifiant unique de l'établissement
- **Données des dirigeants** : donnent la liste des dirigeants actuels de l'entreprise. Ces dirigeants peuvent être des personnes physiques ou des personnes morales. Le tableau 4.1 détaille la structure des données sur les dirigeants.
- **Données procédures collectives** : contiennent la liste des procédures collectives en cours pour l'entreprise. En droit français, une procédure collective place sous contrôle judiciaire le fonctionnement d'une entreprise en difficulté. Elle rassemble tous les créanciers et les prive du droit d'agir individuellement.
- **Données actes et statuts** : contiennent la liste des actes déposés par l'entreprise tels que le K-Bis. Le K-Bis (qui s'écrit aussi K bis) est un document officiel attestant de l'existence juridique d'une entreprise commerciale en France. Il constitue la « carte d'identité » de l'entreprise. Le terme K-bis provient de la référence de l'imprimé qui était utilisé à l'origine

- **Données EIRL** : Le statut Entrepreneur Individuel à Responsabilité Limité (EIRL) est un statut qui permet à un entrepreneur individuel de limiter l'étendue de sa responsabilité. Cela passe par la constitution et la déclaration d'un patrimoine d'affectation dédié à son activité professionnelle.

En analysant les données fournisseurs, qui composent le flux IMR, nous pouvons faire les observations suivantes :

- Altares ne reçoit pas l'information de date de début de fonction des dirigeants. En effet, le flux IMR contient la liste des dirigeants actuels pour l'entreprise sans la date de prise de fonction des dirigeants en poste (la date de début de fonction). De même aucune distinction n'est faite entre les nouveaux dirigeants et ceux qui étaient en poste auparavant.
- Le flux IMR ne donne pas la liste des dirigeants qui étaient en poste et qui ne le sont plus. Pour avoir cette information très importante Altares doit identifier, parmi les dirigeants dont elle dispose pour l'entreprise, lesquels ne sont pas présents dans ce flux afin d'en déduire qu'ils ne sont plus en poste. Ensuite elle doit les supprimer de la liste des dirigeants actuellement en poste en les passant dans l'historique des dirigeants. Enfin, Altares calcule la date de fin de fonction, pour ces dirigeants qui ne sont plus en poste, doit être calculée par Altares.
- Le lieu de naissance est une chaîne de caractères qui peut aller jusqu'à 100 caractères. C'est une zone de saisie libre sans table de correspondance qui imposerait que cette information soit normalisée. En effet, les greffes se contentent de saisir l'information déclarée par l'entreprise. De ce fait, la même information pourrait être écrite de plusieurs façons différentes.
- L'INPI, comme les autres fournisseurs de données officielles, n'a pas de concurrents. Son fonctionnement est défini par le législateur. De ce fait, Altares et ses concurrents (re-diffuseur d'informations officielles sur les entreprises) ont très peu de marge de négociation avec l'INPI. Ils doivent donc s'adapter à son mode de fonctionnement et aux structures des données qu'ils leurs impose.

Dans la section suivante nous verrons comment ces données sont structurées dans la base BDD_DIR.

4.2.2 Modèle Physique de Données

Dans cette section nous présentons le modèle physique de données en se limitant à la description des tables dont la présentation est nécessaire pour la compréhension de notre problème. Ces tables sont les suivantes :

- La table **EB_PERSONNE** : cette table contient toutes les informations des personnes à savoir le nom d'usage, le nom de naissance, le prénom, la

Champ	longueur	Commentaire
Numéro SIREN	9	Identifiant unique de l'entreprise
Code enregistrement = 03	2	
Code fonction	10	Il s'agit des dirigeants des personnes morales immatriculées au Registre du Commerce. Le terme « Dirigeant » comprend les représentants légaux, les administrateurs et certains organes de gestion. La liste des codes fonctions est fournie.
Dénomination ou nom de naissance	160	Le dirigeant peut être soit une personne physique, soit une personne morale. (Code Type de Dirigeant : 2 caractères) 1. Personne physique : – Nom de naissance : 160 caractères (à compter de Mai 2002). – Nom : 80 caractères (nom d'usage) . – Prénom : 80 caractères (premier prénom) . – Date de naissance : 8 caractères . – Lieu de naissance : 100 caractères . – Sexe : 1 caractère. 2. Personne morale : 2 structures possibles : – Structurée – Nom patronymique (position 693) – Prénom (position 182) – Nom d'usage (262) – Sexe (position 450) – Date naissance (position 342) – Lieu de naissance (position 350) – Adresse (position 453) – Non Structurée – Non structurée – Dénomination : (position 22) Représentant personne physique : (position 453)
Prénom	80	
Nom d'usage	80	
Date de naissance (SSAAMMJJ)	8	
Lieu de naissance	100	
Sexe	1	
Type de dirigeant	2	
Représentant permanent si PM	240	
Nom personne physique représentant personne morale	80	

TABLE 4.1 – La composition des données sur les dirigeants reçues de la part de l'INPI

date de naissance, le lieu de naissance et la civilité de la personne. Cette table constitue un référentiel des personnes présentes dans les bases de données d'Altares. Contrairement aux autres tables présentées dans cette section, cette table n'est pas propre aux dirigeants. Elle a pour vocation de centraliser les informations de toutes les personnes quelque soit la source qui les envoient. Par exemple, nous verrons dans le chapitre suivant que cette table contient aussi les personnes phsiques actionnaires.

- La table **EX_DIRIGEANT_RNCS_PP** : Cette table est dédiée aux dirigeants personnes physiques actuellement en fonction. Elle contient la référence de la personne dirigeante et de l'entreprise dirigée. Cette table contient aussi la fonction du dirigeant et la date de début de fonction. Elle permet de lister les dirigeants personnes physiques actuels d'une entreprise donnée. Elle permet aussi, pour une personne donnée, d'avoir la liste des entreprises qu'elle dirige.
- La table **IP_DIRIGEANT_RNCS_PP** : Cette table est destinée aux dirigeants personnes physiques qui ne sont plus en fonction. Elle contient les champs de la table EX_DIRIGEANT_RNCS_PP plus la date de fin de fonction et une date d'historisation. Cette table permet, pour une entreprise donnée, d'avoir la liste des anciens dirigeants personnes physiques. Elle permet aussi, pour une personne donnée, d'avoir la liste des entreprises qu'elle a dirigée dans le passé.
- La table **EX_DIRIGEANT_RNCS_PM** : En plus de la raison sociale de l'entreprise dirigeante, de la référence de la personne qui la représente et de la référence de l'entreprise dirigée, cette table contient aussi la fonction et la date de début de fonction. C'est cette table qui permet de lister les dirigeants personnes morales actuels d'une entreprise donnée. Elle permet aussi, pour une personne morale donnée, d'avoir la liste des entreprises qu'elle dirige.
- La table **IP_DIRIGEANT_RNCS_PM** : cette table contient les champs de la table EX_DIRIGEANT_RNCS_PM plus la date de fin de fonction et une date d'historisation. Cette table permet, pour une entreprise donnée, d'avoir la liste des anciens dirigeants personnes morales. Elle permet aussi, pour une personne morale donnée, d'avoir la liste des entreprises qu'elle a dirigées dans le passé.

En plus des informations citées plus haut, chaque table contient les informations techniques suivantes :

- Un identifiant unique (clé primaire) permettant une exploitation optimale des données.
- Une date d'insertion et une date de mise à jour.
- Un identifiant technique qui permet de remonter jusqu'au fichier fournisseur. Ceci garantit une traçabilité très importante pour Altares. Cette traçabilité

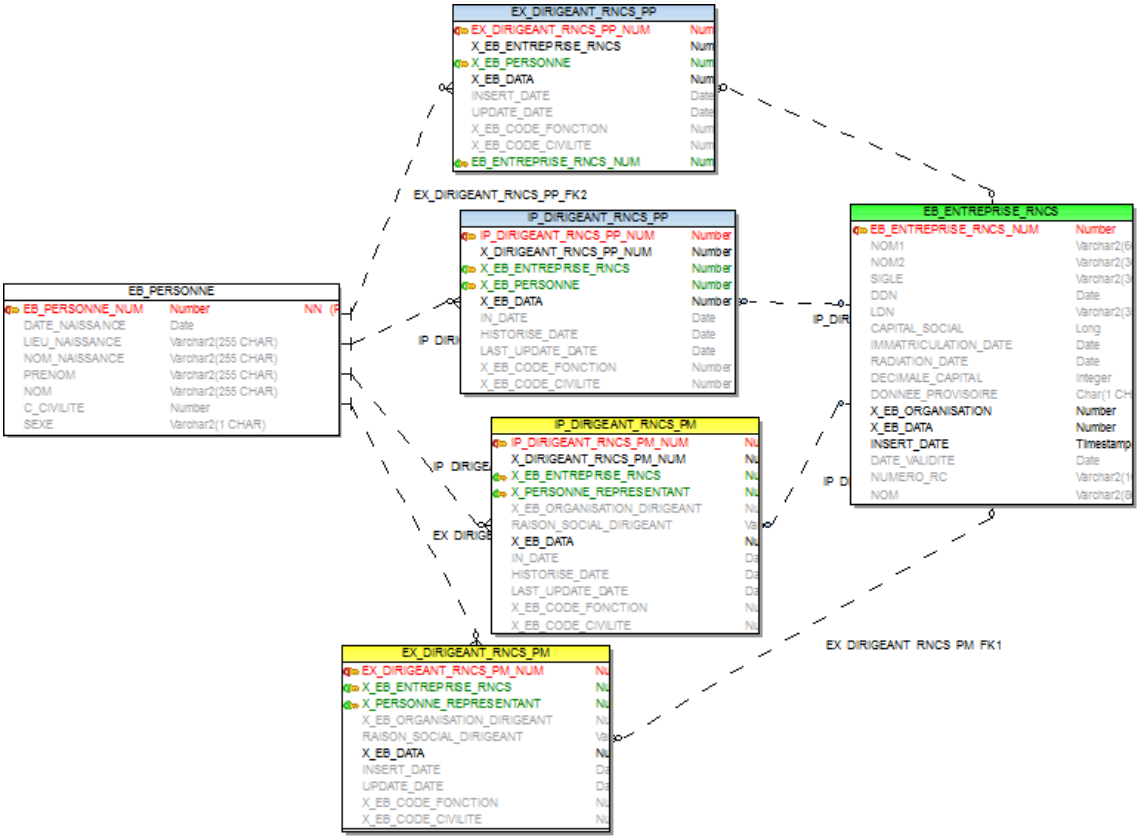

FIGURE 4.1 – Modèle Physique des données de la BDD_DIR

permet, entre autres, de répondre aux réclamations que peuvent formuler les clients d'Altares.

La Figure 4.1 donne la représentation graphique du modèle physique des données des dirigeants composées des 5 tables citées au-dessus et la table d'entreprise nommée **EB_ENTREPRISE_RNCS**. Les tables **EX_DIRIGEANT_RNCS_PP** et **IP_DIRIGEANT_RNCS_PP** sont en bleu clair. Les tables **EX_DIRIGEANT_RNCS_PM** et **IP_DIRIGEANT_RNCS_PM** sont en couleur jaune. La table **EB_PERSONNE** est en couleur blanc et la table **EB_ENTREPRISE_RNCS** est en couleur vert.

Notons que seulement 2% des dirigeants sont des personnes morales. Par conséquence, dans la suite de ce chapitre, nous nous intéressons exclusivement aux dirigeants personnes physiques.

4.2.3 Motivations du problème

Le flux IMR fonctionne en mode annule et remplace. Cela veut dire que les informations reçues de la part de la source viennent à chaque fois annuler et remplacer les informations dont disposaient Altares. Ce flux donne l'état de l'entreprise à un instant t. Cet instant correspond à la date de valeur (la date de la modification dans le registre RNCS) mais sans préciser la différence entre cet état et l'état précédant ni donner les anciennes valeurs.

Pour pallier au manque de certaines informations importantes non envoyées par la source, comme la date de début et de fin de fonction, Altares doit les calculer elle-même (de façon plus au moins exacte). Pour cela, elle doit tenir compte des informations envoyées par la source et des informations déjà disponibles en base.

Pour chacune des sociétés présentes dans le flux et chacun de ces dirigeants envoyés par la source, nous nous posons la question suivante : Ce dirigeant occupait-il déjà le même poste pour cette entreprise ? Autrement dit, avons-nous dans notre base de donnée cette personne en tant que dirigeant de cette même entreprise pour la même fonction ? La réponse à cette question est très importante pour la suite du processus d'intégration.

- Si la réponse est oui nous nous contentons de mettre à jour des informations techniques (la date de mise à jour et les informations de traçabilité).
- Si la réponse est non, nous considérons qu'il s'agit d'un nouveau dirigeant pour cette société. Sa date de début de fonction est positionnée à la date de valeur du fichier en entrée. Ensuite, nous envoyons une alerte à nos clients qui suivent cette société indiquant que celle-ci vient d'avoir un nouveau dirigeant

Supposons que la réponse à cette question devrait être oui et que nous considérons à tort que la réponse est non. Cela peut avoir des conséquences lourdes dont nous citons par exemple :

- Nos dates de début de fonction deviennent erronées. Nos clients peuvent donc prendre des décisions plus au moins importantes sur des informations incorrectes.
- Les alertes reçues par nos clients sont incorrectes. Pour les clients qui traitent nos alertes manuellement, des alertes incorrectes ont un coût de traitement manuel inutile. Pour les clients qui traitent nos alertes de façon automatique, ces alertes peuvent avoir un impact sur la qualité de leur base de données et par la suite sur les décisions qu'ils vont prendre sur la base de ces données.

Or pour répondre à cette question, il faut pouvoir comparer « efficacement » la personne en entrée avec la liste des personnes qui dirigent cette société. Les difficultés de cette comparaison, en l'occurence la comparaison de deux personnes, sont décrites dans la section suivante. Dans le cas où la réponse à la question précédente est non.

Nous nous posons une deuxième question qui est : cette personne existait-elle déjà dans notre base de données BDD_DIR ou non ?

- Si la réponse est oui, nous ne créons pas une nouvelle personne. Mais considérons que cette personne, déjà existante dans notre base de données, vient d'être dirigeante d'une nouvelle société.
- Si la réponse est non, nous créons une nouvelle personne dans la table des personnes EB_PERSONNE.

Supposons que la réponse à cette question devait être oui et que nous considérons à tort que la réponse est non. Cela aura les conséquences suivantes :

- Cette personne existerait deux fois dans notre système d'information (doublons) mais avec deux identités différentes.
- Les expériences de cette personne seront divisées entre ces deux identités. Nous ne pouvons plus fournir à nos clients la liste de ces expériences (les entreprises qu'elle dirige actuellement et les entreprises qu'elle a dirigée dans le passé). Autrement-dit, son parcours ne se sera pas complet.
- Les liens entre les différentes sociétés qu'elle dirige, ne seront pas établis. C'est-à-dire, si avec la première identitée, cette personne est connue comme étant dirigeante de la société s_1 et avec la deuxième identitée elle est connue comme étant dirigeante de la société s_2, nous perdons l'information que s_1 et s_2 ont un dirigeant en commun.

Pour que la réponse à cette question soit correcte, il faut pouvoir comparer « efficacement » la personne en entrée avec la liste des personnes déjà connues en base.

Pour qu'elle soit efficace et qu'elle permette d'apporter des réponses pertinantes aux deux questions posées plus haut dans cette section, cette comparaison doit tenir compte des erreurs de saisie, des fautes d'orthographe et toutes les autres anomalies que peuvent subir les attributs de type chaîne de caractères (le nom, le prénom et le lieu de naissance).

Le fait qu'une même personne dirigeante d'une ou plusieurs entreprises, existe dans nos bases de données uniquement une seule fois et ait un seul identifiant unique, est nécessaire pour une exploitation optimale de ses données. Ceci permet, entre autres, de répondre efficacement à plusieurs questions dont par exemple :

- Pour une société donnée, qui sont ses dirigeants actuels ? Et pour chacun d'eux, depuis quand est-il en poste ?
- Pour une société donnée, qui sont ses dirigeants passés ? Et pour chacun de ces dirigeants, quel est la date de début ainsi que la date de fin de fonction ?
- Quelles sont les sociétés qui sont dirigées, ou qui ont été dirigées par Monsieur $p_1 \in P$ actuellement dirigeant de la société $s_1 \in S$?

- Quelles sont les sociétés qui sont dirigées, ou qui ont été dirigées par Monsieur $p_1 \in P$ anciennement dirigeant de la société $s_1 \in S$?
- Quelles sont les sociétés dont Monsieur $p_1 \in P$, dirigeant de la société $s_1 \in S$, est actionnaire ?
- Quel est le parcours d'un dirigeant donné, c'est-à-dire l'ensemble de ces postes actuels avec la date de début de fonction et l'ensemble de ces postes passés avec les dates de début et de fin de fonction.

Par ailleurs, consciente du rôle que joue le dirigeant d'une entreprise dans la réussite et l'échec de celle-ci, Altares souhaite proposée une solution dédiée aux informations des dirigeants. Cette solution doit comporter deux parties. Une première partie consacrée aux informations des dirigeants, leurs parcours, les personnes avec qui ils dirigent d'autres entreprises. La seconde consolide les informations sur toutes les entreprises que la personne dirige afin d'avoir une idée plus précise de son efficacité en tant que dirigeant. Pour pouvoir mettre en place cette solution, il est primordial d'éliminer les doublons des dirigeants et faire en sorte qu'un même n'existe dans la base des données qu'une seule fois.

4.2.4 Définition du problème

Dans la suite de ce chapitre, nous nous intéressons à la détéction et la suppression des doublons de BDD_DIR afin de garantir l'unicité des personnes. C'est-à-dire, en partant de P, nous nous proposons de créer un ensemble P' de personne unique tel que :

1. $P' \subseteq P$,
2. Pour tout $p_i \in P$ il existe $p'_i \in P'$ tel que $p_i = p'_i$,
3. pour chaque couple $p'_i, p'_j \in P'$ tel que $i \neq j$ et p'_i est différent de p'_j,

En effet, 1 implique de P' est inclus dans P. Toutes le personnes de P' sont issues de P. 2 implique que pour toute personne $p_i \in P$ il existe une personne $p'_i \in P'$ tel que p_i et p'_i désignent la même personne.

Ce problème trouve des applications dans d'autres domaines par exemple la détéction de pages Web similaires ou identiques [17, 20] appelée *détection miroire* (mirror detection), ou encore la détection du même mot ou entité dans un texte [59] connue par *résolution d'anaphores* (anaphora resolution).

Cette problèmatique a été largement étudiée dans la littérature. On la trouve sous plusieurs noms tels que *couplage d'enregistrements* (record linkage or record mat-

ching) [70, 72, 88, 71], *déduplication des données* (data deduplication) [80], *identification de l'instance* (instance identification) [92] ou encore *couplage des noms* (name matching) [15]. De plus, le problème de détection des doublons a, par ailleurs, plusieurs applications dans diverses domaines. En particulier, dans le domaine de la santé, le problème a suscité l'intérêt des chercheurs ainsi que les praticiens [55, 58].

Cependant, par rapport aux problèmes de détection des doublons étudiés dans la littérature, notre problème présente certaines particularités dont nous citons :

- La solution doit pouvoir être appliquée sur la totalité de notre base de données pour en éliminer les doublons, mais aussi ajoutée au processus d'intégration des données pour éviter de créer des doublons lors de la mise à jour quotidienne de la base de données. C'est-à-dire, vu l'impact de création des doublons, il ne suffit pas d'avoir un processus d'élémination des doublons qu'on pourra exécuter à une fréquence défini, mais il faut éviter de créer des doublons à chaque mise à jour de notre base de données.
- La très grande taille de notre base des données.
- A cause des anomalies, des fautes d'orthographe et des erreurs de saisie, les doublons que nous cherchons à identifier et supprimer ne sont pas identiques au sens informatique du terme. Ces doublons peuvent en effet différer d'un ou plusieurs caractères. Par conséquence, notre problème combine la problématique de détection des doublons dans une base de données avec la problèmatique de calcul de similarité de chaîne de caractères.

Dans la section suivante, nous présentons les méthodes de calcul de similarité syntaxique les plus connues dans la littérature pour en déduire celles que nous utiliserons pour la comparaison de nos chaînes de caractères lors de la détection des doublons de la BDD_DIR.

4.3 État de l'art

L'évaluation de similarité entre deux chaînes de caractères ou deux documents est une problèmatique présente dans plusieurs disciplines dont par exemple la recherche ou l'extraction des connaissances à partir des données textuelles. Cette similarité pourra être étudiée selon deux axes : la similarité syntaxique et la similarité sémantique. La première définit la similarité en se basant sur la façon dont on écrit les mots. Alors que la seconde définit la similarité en se basant sur le sens des mots.

Pour le PDD, nous nous intéressons à la similarité syntaxique. Dans cette section nous commençons par donner de façon non exhaustive quelques mesures (distance et algorithmes) qui permettent de mésurer la similarité syntaxique entre deux chaînes

de caractères. Ensuite nous citons, à titre d'exemple, quelques logiciels de déduplication.

Pour une étude bibliographique approfondie, le lecteur peut se référer aux surveys [36, 23, 68] et au livre [22].

4.3.1 Encodage phonétique

L'encodage phonétique fut la première technique développée pour la comparaison des chaînes de caractères pour pallier aux fautes d'orthographe. Cette technique s'avère très efficace pour éliminer les fautes d'orthographe et les erreurs syntaxiques mais présentent le défaut de considérer parfois de fausses similitudes.

4.3.1.1 Soundex

Le terme Soundex remonte à 1918. Le premier algorithme de ce type a été inventé par Margeret O'Dell et Robert C.Russell [76, 77], lors des études des problèmes liées au recensement américain. En effet, de par leur construction, les Etats-Unis d'Amérique sont tenus a recenser leur population tous les 10 ans. Á la fin du 19ième siècle, le problème du recensement était devenu un problème majeur. Le fait de traiter des informations concernant une population de plusieurs dizaines de millions d'américains à la main demandait un travail phénoménal.

Le Soundex est un encodage extrêmement simple d'un mot sous une forme phonétique simplifiée. Il permet de réaliser des comparaisons grossières entre des champs, car l'orthographie est réduite à son strict minimum. Cet encodage provoque par contre beaucoup de fausses ressemblances.

Le Soundex d'un mot est un code composé d'un caractère alphanumérique suivi de trois caractères numérique. Le caractère en alphanumérique est le premier caractère du mot à codifier. Les trois caractères numériques sont un encodage simple du reste du mot. L'Algorithme 1 donne en détail les étapes du Soundex.

Compte tenu que les tables de caractères modernes, permettent maintenant de saisir des lettres majuscules accentuées, il est nécessaire de transcrire ces lettres en lettres simples. En particulier, dans la langue française, le c majuscule avec cédille (Ç) sera transformé en S. De même le caractère OE (dans le mot cœur par exemple) sera transformé en E. De plus il est nécessaire de supprimer les espaces morts, c'est-à-dire les espaces qui peuvent être avant et après le mot ainsi que les blancs et le tiret. L'encodage étant extrêmement grossier, il ne peut pas être utilisé pour comparer seul deux valeurs d'un même champ : une comparaison sur le nom

Algorithme 1 : L'algorithme de Soundex

Data : Une chaîne de caractère C_1

Result : Un code de longueur 4 qui commence par un alpha-numérique suivi de trois chiffres

Ecrire la chaîne C_1 en majuscules;
Conserver la première lettre du mot;
Eliminer toutes les voyelles, le H et le W;
Coder les lettres restantes à l'aide de la Table 4.2;
Eliminer toutes les paires consécutives de chiffres dupliquées;
Retourner 4 caractères du Soundex ainsi obtenus (compléter par des zéros s'il y a moins de 4 caractères);

Lettre	**Code**
B F P V	1
C G J K Q S X Z	2
D T	3
L	4
M N	5
R	6

TABLE 4.2 – Tableau de correspondance Soundex

d'une entreprise ramènerait trop de fiches similaires. Par contre vu sa simplicité et sa rapidité, il peut être envisagé lors d'un pré-traitement qui vise à identifier les informations comparables.

4.3.1.2 Soundex2

Le Soundex2 est une version francisée du Soundex mais aussi plus performante. Contrairement au Soundex qui ne fait appel qu'à des chiffres à l'exception du premier caractère, le Soundex2 conserve la plupart des lettres. En comparant les deux versions, on trouve, pour la première un nombre de combinaisons possibles de 26x10x10x10 = 26 000 alors que dans cette version améliorée le nombre de combinaisons différentes monte jusqu'à 20x20x20x20 = 160 000 (le code est composé de 4 caractères, chacun de ces caractère peut être l'un des 20 consonnes de la langue française). Il se révèle donc plus performant dans de nombreux cas, c'est à dire qu'il permet de considérer moins de fausses similarités. L'Algorithme 2 détaille les étapes du Soundex2.

Algorithme 2 : L'algorithme de Soundex2

Data : Une chaîne de caractère C_1
Result : Un code alphanumérique de longueur 4
Éliminer les blancs à droite et à gauche du nom;
Convertir le nom en majuscule;
Convertir les lettres accentuées et le c cédille en lettres non accentuées;
Eliminer les blancs et les tirets;
Remplacer les groupes de lettres par leurs correspondants dans le Tableau (4.3);
Remplacer toutes les voyelles sauf le Y par A exceptée s'il y a un A en tête;
Remplacer les préfixes présents dans le Tableau (4.4) par leur correspondants dans ce même tableau;
Supprimer les H sauf s'ils sont précédés par C ou S;
Supprimer les Y sauf s'il est précédé d'un A;
Supprimer les terminaisons suivantes A, T, D et S;
Enlever tous les A sauf le A de tête s'il y en a un;
Enlever toutes les sous chaînes de lettres répétitives;
Retourner les 4 premiers caractères du mot et si besoin le compléter avec des blancs pour obtenir 4 caractères;

GUI	KI
GUE	KE
GA	KA
GO	KO
GU	K
CA	KA
CO	KO
CU	KU
Q	K
CC	K
CK	K

TABLE 4.3 – Tableau de correspondance Soundex2, prononciation K

MAC	MCC
ASA	AZA
KN	NN
PF	FF
SCH	SSS
PH	FF

TABLE 4.4 – Tableau de correspondance Soundex2

4.3.1.3 Phonex

Phonex est un algorithme basé sur le phonétique et encore plus perfectionné que Soundex2 (la version francisée du Soundex). Le Phonex est optimisé pour le langage français. Il sait reconnaître différents types de sons comme les sons 'on', 'ai', 'ein', etc... et place son résultat sous la forme d'un réel de type double (précision 5.0 x 10-324 .. 1.7 x 10308 sur 15 à 16 chiffres significatifs). Cependant son temps de calcul est double de celui du Soundex et 30% supérieure de celui du Soundex2

4.3.2 Algorithme de comparaison syntaxique

Ces algorithmes servent à comparer deux chaînes de caractères afin d'évaluer la distance entre elles. Ils peuvent être utilisés sur le texte brut ou après un pré-traitement (par exemple Soundex). Leur nombre et leur richesse proviennent des recherches réalisées en génétique, cherchant à apparier des codes ADN.

4.3.2.1 Distance de Hamming

La distance de Hamming, appelée aussi la différence de Hamming, est le nombre de caractères non identiques à la même position dans deux chaînes de caractères de même longueur. Par exemple les chaînes suivantes : "D823" et "M843" ont une différence de HAMMING de 2. Ainsi deux Soundex sont identiques si la différence de HAMMING vaut zéro. Ils sont semblables si cette différence est 1. Ils sont totalement dissemblables si la distances de HAMMING est 4 (Le maximum dans le cas d'un Soundex). La différence de HAMMING est un algorithme simple et très performant car d'une complexité en $O(n)$, où n est la longueur de la plus longue de deux chaînes de caractères comparées.

4.3.2.2 Distance de Jaro

Introduite en 1978 dans [47], la distance de Jaro entre deux chaînes de caractères est nulle si ces chaînes de caractères sont totalement différentes. Elle vaut 1 pour deux chaînes de caractères parfaitement similaires (identiques). La distance de Jaro d_j entre deux chaînes C_1 et C_2 est définie par :

$$d_j = \frac{1}{3} \times (\frac{m}{|C_1|} + \frac{m}{|C_2|} + \frac{m-t}{m}),$$

où m est le nombre de caractère correspondants entre les deux chaînes de caractère et t est le nombre de transpositions.

Deux caractères identiques de C_1 et C_2 sont considérés comme similaires si leur éloignement (i.e. la différence entre leurs positions dans leurs chaînes respectives) ne dépasse pas :

$$\max(|C_1|, |C_2|/2) - 1.$$

Le nombre de transpositions est obtenu en comparant le i-ème caractère correspondant de C_1 avec le i-ème caractère correspondant de C_2. Le nombre de fois où ces caractères sont différents, divisé par deux, donne le nombre de transpositions.

4.3.2.3 Distance de Jaro Winkler

La distance de Jaro Winkler est une variante de la distance de Jaro qui accorde plus d'importance au début de deux chaînes de caractères. La méthode introduite par Winkler utilise un coefficient de préfixe p qui donne plus de poids au préfixe de longueur l (avec l<4). En considérant deux chaînes C_1 et C_2, leur distance de Jaro-Winkler d_w est donné par la formule suivante :

$$d_w = d_j + (l \times p \times (l - dj)),$$

où : dj est la distance de Jaro entre C_1 et C_2, l est la longueur du préfixe commun (maximum 4 caractères) et p est un coefficient qui permet de favoriser les chaînes avec un préfixe commun. Winkler propose pour ce coefficient une valeur p = 0,1

4.3.2.4 Distance de Levenshtein

Proposée par Vladimir Levenshtein en 1965 [53], la distance de Levenshtein est une distance mathématique qui donne une mesure de similarité entre deux chaînes de caractères. Elle est égale au nombre minimum d'opération qu'il faut effectuer sur une chaîne de caractères C_1 pour qu'elle devienne identique à une chaîne de caractères C_2. On entend par opération :

- Une substitution d'un caractère de C_1 en un caractère de C_2
- Un ajout dans C_1 d'un caractère de C_2
- Une suppression d'un caractère de C_1

Cette distance est aussi appelée *la distance d'édition* ou la *déformation dynamique temporelle*. Elle est utilisée dans plusieurs domaines et notamment en reconnaissance de forme et reconnaissance vocale.

La distance de Levenshtein peut être considérée comme une généralisation de la distance de Hamming. On peut montrer en particulier que la distance de Hamming est une borne supérieure de la distance de Levenshtein.

Un algorithme en programmation dynamique (solution de type du bas en haut) a été proposé pour calculer la distance de Levenshtein. Cet algorithme utilise une matrice de dimension $(n+1) \times (m+1)$ où n et m sont respectivement les longueurs des deux chaînes de caractères C_1 et C_2.

4.3.2.5 Algorithme de Needleman-Wunsch

L'algorithme de Needleman-Wunsch a été proposé en 1970 par Saul Needleman et Christian Wunsch [69]. Cet algorithme, qui est souvent utilisé en bio-informatique pour aligner des séquences de protéines ou de nucléotides, effectue un alignement global maximal de deux chaînes de caractères.

L'algorithme de Needleman-Wunsch est une évolution de l'algorithme Levenshtein qui intègre une pénalité pour les suppressions. En bio-informatique et notamment lors de l'alignement des séquences, cette pénalité de suppression se traduit par le fait de tenir compte des « trous » dans les sequences.

Tout comme Levenshtein, l'algorithme de Needleman-Wunsch est basé sur la programmation dynamique. L'algorithme de Needleman-Wunsch fut la première application de la programmation dynamique pour la comparaison de séquences biologiques.

4.3.2.6 Distance de Smith Waterman

Proposé par Temple F. Smith et Michael Waterman en 1981 [84], l'algorithme de Smith-Waterman correspond à la distance de Levenshtein, en ajoutant deux paramètres : un coût à la suppression, et un coût à la substitution. Cet algorithme est aussi basé sur la programmation dynamique.

4.3.2.7 Distance de Monge Elkan

Comme son nom l'indique, cette distance a été introduite par Monge et Elkan [63]. Pour comparer deux chaînes de caractères S_1 et S_2, Monge et Elkan proposent de couper chaque chaîne en un ensemble de sous-chaînes de caractères, $S_1 = a_1 a_2 ... a_K$, $K \in N$ et $S_2 = b_1 b_2 ... b_L$, $L \in \mathbb{N}$.

Le principe de cette mesure de similarité est le suivant. A chaque sous-chaîne de caractères a_i, $i = 1, ..., K$ de S_1, on associe une sous-chaîne de caratères b_j, $j = 1, ..., L$ de S_2 de façon à maximiser la similarité entre les sous-chaînes a_i et b_j. Cette similarité peut être calculée par l'une des distances présentées précedemment. Les scores de similarité maximale calculée pour chaque a_i, $i = 1, ..., K$ de S_1 sont par la suite additionnés et normalisés par rapport à K (nombre total de sous-chaînes de S_1). Par conséquent, la mesure de similarité de Monge-Elkan est donnée par la formule suivante :

$$sim(S_1, S_2) = \frac{1}{K} \sum_{i=1}^{K} \max_{i=1,...,L} sim'(a_i, b_j),$$

où sim' est l'une des mesures de similarité mentionnées précédemment. Monge et Elkan [64] ont considéré les sous-chaînes comme des *jetons* (tokens). Une telle mesure est appelée *distance à deux niveaux* (level two distance).

4.3.3 Algorithme d'encodage vectoriel et de classification

Ces algorithmes servent à classer plusieurs contenus textuels, par analyse globale du texte, souvent encodé de façon vectorielle. Ils peuvent servir à identifier des textes utilisant du vocabulaire proche.

4.3.3.1 Coefficient de Dice

Le coefficient de Dice calcule une valeur liée au nombre de termes communs. Soient C_1 et C_2 deux chaînes de caractères. On note par : T_C : le nombre de terme (mot) commun entre C_1 et C_2

T_{c_1} : le nombre de terme de la chaîne C_1

T_{c_2} : le nombre de terme de la chaîne C_2

Le coefficient de Dice C_D de (C_1,C_2) est défini par :

$$C_D = \frac{2 \times T_C}{T_{c_1} + T_{c_2}}$$

Ainsi, ce coefficient vaut 0 si aucun terme n'est en commun entre C_1 et C_2, et 1 si tous les termes sont communs.

Par contre, il pondère de la même façon les termes signaficatifs que les mots vides. Pour améliorer les résultats obtenus par le coefficient de Dice, il convient d'éliminer les mots vides de C_1 et C_2 avant de les comparer.

4.3.3.2 Distance euclidienne

Etant donné deux documents d_1 et d_2, la distance Euclidienne définit la similarité entre d_1 et d_2 comme étant la distance entre les deux représentations vectorielles de d_1 et d_2 ramenée à un seul point :

$$euc(d_1, d_2) = \sqrt{\sum_{i=1}^{n}(d_{1_i} - d_{2_i})^2},$$

Cette solution possède le défaut de considérer que deux vecteurs peuvent être éloignés, alors qu'ils peuvent être colinéaires. Dans le cas de certains encodages (par exemple les vecteurs des fréquences des termes), cette mesure trouve une distance élevée, alors qu'un texte n'est que la répétition de l'autre.

4.3.3.3 Cosinus

Le Cosinus compare l'angle des deux vecteurs et permet de vérifier que les termes des documents apparaissent selon des proportions similaires. Cette mesure est souvent préférée dans le cas d'encodages vectoriels, car elle limite l'influence de la longueur des textes.

4.3.3.4 Q-Gram

Le principe du q-gram consiste à extraire d'un texte toutes les sous-chaînes de longueur q [91, 90]. Si deux textes partagent de nombreux q-grams, il est probable qu'ils contiennent de nombreuses similitudes. Par exemple 'Sam Chapman', donne en 3-grams :

(1,' s'), (2,' sa'), (3,'sam'), (4,'am '), (5,'m c'), (6,' ch'), (7,'cha'), (8,'hap'), (9,'apm'), (10,'pma'), (11,'man'), (12,'an '), (13,'n ')

Cet encodage vectoriel peut donner des résultats pertinents, car si q est suffisamment petit il intègre naturellement la comparaison des termes en autorisant des écarts orthographiques. En effet, deux chaînes de caractères similaires, mais contenant quelques erreurs, possèdent un fort pourcentage de q-grams communs. De plus, cette solution a l'avantage de créer une base vectorielle (ensemble des q-grams possibles) contenant des mots de petite taille, ce qui peut accélérer les requêtes.

Il faudra par contre réaliser des tests afin d'évaluer q : s'il est trop petit, alors de nombreuses fausses similarités seront trouvées (au cas limite $q = 1$, cela revient à

faire une recherche des lettres communes), s'il est trop grand, alors cela revient à rechercher des mots complets, limitant l'efficacité de l'algorithme pour les erreurs de syntaxe.

4.3.3.5 Term Frequency/Inverse Document Frequency

Le principe du Term Frequency/Inverse Document Frequency *TF/IDF*consiste à calculer la fréquence d'un terme dans un texte, en la divisant par le nombre de textes où ce terme apparaît.

Ainsi un terme utilisé fréquemment dans un texte, mais absent des autres, aura un TF/IDF élevé, tandis qu'un mot courant présent dans tous les textes aura un TF/IDF faible. Ce paramètre correspond d'une certaine façon à la quantité d'information d'un terme. Par exemple, les requêtes sont améliorées, car des termes courant comme 'le','la','les' interviennent peu dans le calcul de similarité, contrairement à des termes spécifiques (l'acronyme d'une entreprise par exemple).

Le texte est alors encodé en utilisant les valeurs TF/IDF pour les termes composant le texte. Cet encodage est extrêmement pertinent pour classer des documents parlant de sujets similaires, dès lors qu'ils sont tous dans la même langue. Il nécessite par contre la construction d'une table de fréquences des termes sur l'ensemble des documents.

4.3.4 Logiciels de déduplication

Au cours des dernières années, une gamme d'outils destinés au nettoyage des bases de données est apparue sur le marché et au sein de certains groupes de recherche. Dans ce contexte, des progiciels de détection de doublons ont été mis à la disposition du public.

Développé dans le cadre d'une collaboration entre "Australian National University de Canberra" et "New South Wales Department of Health" de Sydney, FEBRL (Freely Extensible Biomedical Record Linkage) [23] est un outil qui avait pour objectif de nettoyer et standardiser les bases de données dans le domaine de la santé. FEBRL est actuellement un outil en open source [1] qui consiste en deux composantes : la première composante vise à standardiser les données et la seconde vise l'élimination des doublons.

TAILOR [35] (Record Linkage Toolbox) est une boîte à outils flexible qui permet à l'utilisateur de détecter les doublons de façon flexible. La flexibilité est dû au fait que l'outil propose plusieurs méthodes de déduplication des données permettant,

selon la base de données traitée, de choisir laquelle (ou lesquelles) des méthodes doit (ou doivent être) utilisée(s).

WHIRL [2, 24] (Word-based Information Representation Language) est un système de détection de doublons disponible gratuitement pour tout usage académique ou dans la recherche. Basé sur l'utilisation de la métrique TF/IDF pour la similarité, WHIRL est un outil qui permet l'intégration structurée de sources d'informations hétérogènes et différentes.

D'autres outils comme le projet Flamingo [3] et BigMatch [94] ont été aussi développé dans la même optique. Pour plus de détails sur les outils et systèmes traitant le problème de déduplication des données ainsi que d'autres logiciels, le lecteur peut se référer à [19, 31, 36].

Dans la section suivante, nous présentons la méthode de résolution que nous avons mis en place pour le problème PDD.

4.4 Méthode de résolution

La méthode de résolution proposée doit permettre de dé-dupliquer les enregistrements de la base de données BDD_DIR par un traitement dit *traitement d'initialisation*. L'objectif est d'avoir une base initiale "propre". Ensuite, cette méthode doit être implementée dans la chaîne d'intégration des données afin de permettre de répondre efficacement aux questions posées dans la Section 4.2.3 et garder ainsi une base de données de bonne qualité.

Dans cette section, nous décrivons la méthode de résolution proposée ainsi que les étapes qui ont permis d'élaborer cette méthode. Dans un premier temps, nous nous focalisons sur l'attribut « lieu de naissance » et expliquons les particularités de ce champ qui ont conduit à lui proposer un traitement différent de celui du nom et du prénom. Ce traitement sera appelé la *normalisation des lieux de naissance*. Puis nous présentons la méthode proposée pour pallier aux erreurs de saisie et aux fautes d'orthographe des noms et des prénoms. Ensuite, nous décrivons un traitement particulier qui a été appliqué aux personnes qui dirigent la même société en expliquant les motivations qui ont poussés à considérer ce traitement appelé *Assouplissement des règles*. Enfin nous présentons les résultats obtenus par l'application de l'approche proposée sur la base de données BDD_DIR d'Altares.

4.4.1 Normalisation de lieu de naissance

L'analyse de la qualité de la base de données BDD_DIR, a permis de constater que beaucoup de doublons, ont été générés à cause de lieux de naissance écrits de façon non normalisée et donc considérés comme différents alors qu'ils désignent bien le même lieu. Comme vu dans la partie description de la source des données, ce champ est un texte libre de longueur de 100 caractères dont le contenu est déclaratif. Par conséquent, les déclarants peuvent décrire de différentes façons le même lieu.

Pour signifier le même lieu, les combinaisons qu'on peut faire à partir du nom de la commune, du département, du code postal, de la région, de l'arrondissement et du pays sont très nombreuses. D'autre part, les différentes abréviations plus au moins courantes viennent augmenter encore plus le nombre de combinaisons possibles pour décrire le même lieu. Á cela, s'ajoute les anomalies classiques d'une chaîne de caractères, à savoir les fautes d'orthographe et les erreurs de saisie. Par exemple, nous avons identifié 3516 types d'écritures différentes pour PARIS dont 264 concerne Paris 14ième. Ci-après quelques exemples d'écriture pour Paris 14ième :

PARIS 14, PARIS (75014), PARIS 14 (75), Paris (75014), 75 PARIS 14, PARIS 14EME, PARIS-14EME ARRONDISSEMENT, PARIS-14E ARRONDISSEMENT, PARIS 14E, PARIS 75014, 75014 PARIS, 75 PARIS 14EME, 75 PARIS (14EME), PARIS 14 EME, PARIS14, 75014 PARIS (FRANCE), 75 PARIS 14 ARRONDISSEMENT

Il est évident que ces combinaisons ne tiennent pas compte des différences classiques souvent éliminées lors de la comparaison des chaînes de caractères comme les majuscule/minuscule, les caractères accentués et les caractères spéciaux.

Les lieux de naissance des dirigeants nés à l'étranger, posent davantage de problème. S'agissant de lieux moins connus pour l'opérateur qui saisit l'information (par rapport aux communes françaises), les fautes d'orthographe sont plus fréquentes. D'autre part, beaucoup parmi ces lieux, étaient initialement écrits dans des langues étrangères, ils peuvent alors accepter plusieurs orthographes en langue française. Cela augmente par la suite le nombre de combinaisons possibles pour signifier le même lieu situé à l'etranger. Par ailleurs, le pays doit être mentionné pour les lieux de naissance situé à l'étranger, ce qui est aussi une source supplémentaire des doublons des personnes et donc des dirigeants.

Notons que la probabilité que deux personnes différentes, qui ont le même nom, le même prénom, qui sont nées à l'étranger le même jour dans le même pays et se retrouve en France en train de diriger la même entreprise est très faible pour ne pas dire quasi-nulle. Par conséquence et afin d'éviter les doublons engendrés par les différences d'écriture des villes étrangères, nous avons décidé de rester au niveau du

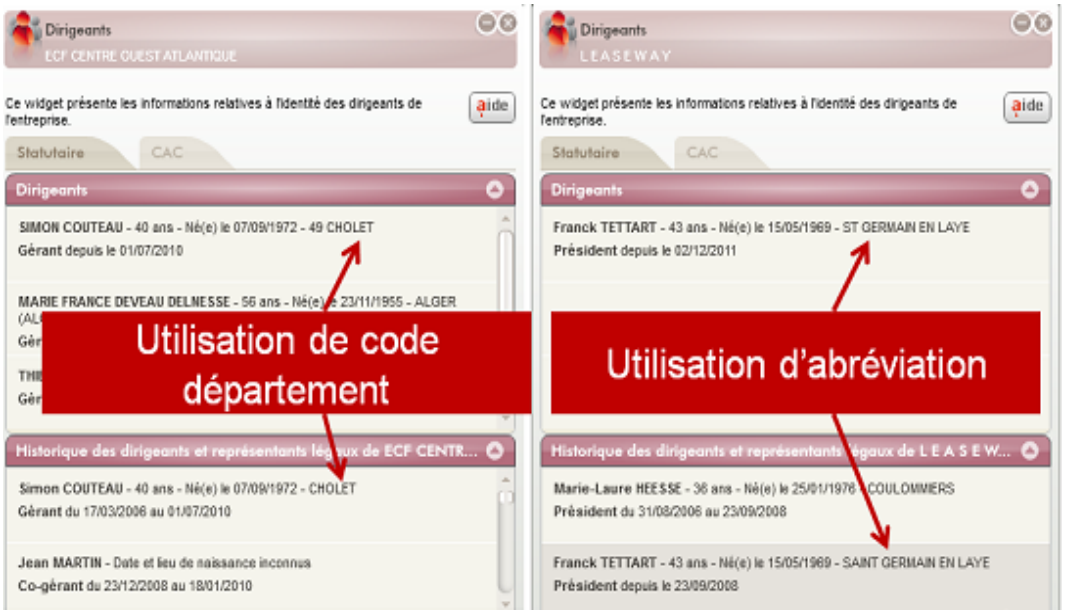

FIGURE 4.2 – Exemples doublons engendrés par le champ "lieu de naissance"

pays pour les lieux de naissance des dirigeants nés à l'étranger.

Deux lieux de naissance peuvent avoir deux orthographes très semblables sans qu'ils soient identiques ni même être géographiquement proches. C'est par exemple le cas de la commune de « Agen » dans le 34 et « Agel » dans le 47, de « Ajou » dans le 27 et « Ajoux » dans le 07, de « Allone » dans le 79 et « Allones » dans le 72, de « Dolaincourt » dans le 88 et « Dolancourt » dans le 10 et de « Palis » dans le 10 et « Paris » dans le 75. Les exemples de ce type sont très nombreux.

Nous ne pouvons donc pas comparer les lieux de naissance par les méthodes de similarité. En effet, qu'il s'agisse d'encodage phonétiques, des algorithmes de comparaison syntaxique, au vu des exemples cités plus haut, le risque de considérer comme identique, des lieux qui ne le sont pas, est très élevé.

Au départ de notre travail, la table EB_PERSONNE contient 650 000 lieux de naissance informatiquement différents. Nous avons mis ces lieux de naissance dans une table créée à cet effet appelée EB_LIEU_NAISSANCE. Le Tableau 4.5 donne la structure de cette table.

Ensuite nous avons créé les deux tables suivantes :

- ES_TABLE_VILLE : cette table contient toutes les communes françaises, soit 36 568 commune.
- ES_TABLE_PAYS : cette table contient la liste de tous les pays étrangers.

Champ	Type	Descriptif
ES_LIEU_NAISSANCE_NUM	NUMBER	La clé primaire de cette table
LIEU_NAISSANCE	VARCHAR2(255 CHAR)	Le lieu de naissance à normaliser. Ce champ est issu de la table EB_PERSONNE
NOMBRE_PERSONNE	NUMBER	Le nombre de lignes de la table EB_PERSONNE avec EB_PERSONNE.LIEU_NAISSANCE = ES_LIEU_NAISSANCE.LIEU_NAISSANCE Ce champ donne donc le nombre de personnes nées au lieu de naissance présent dans la même ligne.
X_LIEU_NAISSANCE	NUMBER	C'est ce champ qui permettra de faire le lien avec la table de référence des lieux de naissance.
TYPE_LIEU_NAISSANCE	VARCHAR2(10 CHAR)	Ce champ permettra de distinguer les lieux de naissance en France pour lesquels il sera positionné à 'VILLE' de ceux qui sont à l'étranger pour lesquels il sera positionné à 'PAYS'.
ID_LIEU_NAISSANCE	VARCHAR2(10 CHAR)	Ce champ est la concatenation du type de lieu de naissance et TYPE_LIEU_NAISSANCE et du X_LIEU_NAISSANCE.

TABLE 4.5 – La structure de la table ES_TABLE_VILLE

Champ	Type	Descriptif
ES_TABLE_PAYS_NUM	NUMBER	Identifiant unique par pays
PAYS_NOM	VARCHAR2(255 CHAR)	Le nom du Pays concerné
PAYS_NOM_MATCHING	VARCHAR2(255 CHAR)	C'est ce champ qui sera utilisé pour faire le rapprochement avec la table ES_LIEU_NAISSANCE. Ce champ est obtenu en faisant les opérations suivantes sur le nom du pays : – Passer en majuscule – Remplacer les caractères accentué et l'apostrophe
TRAITE	NUMBER(1)	C'est un champ qui permet de savoir si nous avons déjà traité cette ligne ou non. Nous verrons son utilité dans l'algorithme qui sera proposé plus loin

Table 4.6 – La structure de la table ES_TABLE_PAYS

Nous avons listé 242 pays.

Ces deux tables constituent un référentiel de lieux de naissance. La normalisation des lieux de naissance consistera donc à associer à chacun des lignes de la table ES_LIEU_NAISSANCE, soit une ligne de la table ES_TABLE_PAYS si le lieu de naissance est à l'étranger soit une ligne de la table ES_TABLE_VILLE si le lieu de naissance est en France. Par exemple les 3516 types d'écritures différentes de Paris doivent être associées à la ville de Paris présente une seule fois dans le référentiel des lieux de naissance.

4.4.1.1 Normalisation des lieux de naissance situés à l'étranger

Cette étape consiste à identifier toutes les lignes de la table ES_LIEU_NAISSANCE qui concerne des lieux de naissance situés à l'étranger et associer chacun de ces lieux au pays dans lequel il se trouve.

Le Tableau 4.6 donne la structure de la table ES_TABLE_PAYS. Cette table ne contient pas une ligne par pays. En effet, cette table contient, pour chaque pays, autant de ligne que de nom connu pour ce pays. Pour un pays qui a plusieurs noms, il sera présent dans cette table autant de fois qu'il a de nom avec à chaque fois un PAYS_NOM_MATCHING différent mais la même valeur pour le champ ES_TABLE_PAYS_NUM. Par exemple, les Etats-Unis sont connus par les noms Etats-Unis et USA, nous avons donc les deux lignes le concernant dans la table ES_TABLE_PAYS (voir le Tableau 4.7).

Nous avons proposé un algorithme qui permet de faire le rapprochement entre

ES_TABLE_PAYS_NUM	PAYS_NOM	PAYS_NOM_MATCHING
304	Etats-Unis	ETATS UNIS
304	Etats-Unis	USA

TABLE 4.7 – La contenu de la table ES_TABLE_PAYS pour le USA

les lignes de la table ES_LIEU_NAISSANCE et leurs correspondants dans la table ES_TABLE_PAYS. Les étapes principales de cet algorithme sont données dans l'Algorithme 3.

Algorithme 3 : L'Algorithme de normalisation des lieux de naissance situés à l'étranger en utilisant le nom de pays

Data : les deux tables ES_TABLE_PAYS et ES_LIEU_NAISSANCE
Result : la table ES_LIEU_NAISSANCE mise à jour

1. Positionner le champ TRAITE = 0 pour toutes les lignes de la table ES_TABLE_PAYS
2. Si le nombre des lignes de la table ES_TABLE_PAYS qui ont le champ TRAITE =0 est égale 0 fin de traitement.
3. L1 = Parmi les lignes de la table ES_TABLE_PAYS qui ont le champ TRAITE = 0, celle dont le champ PAYS_NOM_MATCHING est le plus long.
4. Chercher toutes les lignes de la table ES_LIEU_NAISSANCE dont le champ X_LIEU_NAISSANCE est vide et le champ LIEU_NAISSANCE contient le champ l1. PAYS_NOM_MATCHING.
5. Pour toutes ces lignes, positionner le champ X_LIEU_NAISSANCE à l1.ES_TABLE_PAYS_NUM et le champ TYPE_LIEU_NAISSANCE à la valeur 'PAYS'.
6. Positionner le champ l1.TRAITE = 1
7. Revenir à 2

Après une première exécution de cet algorithme, nous avons ré-exécuté une deuxième variante de cet algorithme basée sur les noms des pays en anglais. Cette deuxième variante permettra de faire le rapprochement avec les lieux de naissance situés à l'étranger écrits en anglais.

Champ	Type	Descriptif
ES_TABLE_VILLE_NUM	NUMBER	Identifiant unique par ville
VILLE_NOM	VARCHAR2(255 CHAR)	Le nom de la ville concerné
VILLE_NOM_MATCHING	VARCHAR2(255 CHAR)	C'est ce champ qui sera utilisé pour faire le rapprochement avec la table ES_LIEU_NAISSANCE. Ce champ est obtenu en faisant les opérations suivantes sur le nom du ville : – Passer en majuscule – Remplacer les caractères accentué et l'apostrophe
TRAITE	NUMBER(1)	Champ permettant de savoir si nous avons déjà traité ce champ ou nom.

TABLE 4.8 – La structure de la table ES_TABLE_VILLE

4.4.1.2 Normalisation des lieux de naissance situés en France

Nous nous intéressons à présent aux lieux de naissance situés en France. L'objectif est de faire le rapprochement entre la table ES_TABLE_VILLE et la table ES_LIEU_NAISSANCE en associant à chaque ligne de la table ES_LIEU _NAISSANCE dont le champ X_LIEU_NAISSANCE est vide, c'est-à-dire qu'elle n'a pas été rapprochée dans l'étape précédente, la ligne qui lui correspond dans la table ES_TABLE_VILLE.

La structure de la table ES_TABLE_VILLE est donnée par le Tableau 4.8

Notons que pour plusieurs villes françaises, il existe des villes étrangères qui ont le même nom. Par exemple, il existe une ville aux Etats-Unis qui s'appelle Paris. De la même façon, il existe une ville française qui s'appelle Bruxelles.

Il est très important de commencer par le rapprochement avec les pays étrangers avant le rapprochement avec les communes françaises. Nous considérons ainsi que le nom du pays est plus discriminant que celui de la commune. Cela permet par exemple de rapprocher le lieu de naissance « Bruxelles Belgique » à la « Belgique » et non pas à la commune française qui s'appelle « Bruxelles ».

Pour la normalisation des lieux de naissance situés en France, nous avons proposé un algorithme, similaire à celui qui a été proposé pour les lieux étrangers, qui permet de faire le rapprochement entre les lignes de la table ES_LIEU_NAISSANCE et leurs correspondants dans la table ES_TABLE_VILLE. Les étapes principales de cet algorithme sont données dans l'Algorithme 4.

Notons que plusieurs villes ont leur nom inclus dans le nom d'une autre ville. Ce

Algorithme 4 : L'Algorithme de normalisation des lieux de naissance situés en France

Data : les deux tables ES_TABLE_VILLE et ES_LIEU_NAISSANCE
Result : la table ES_LIEU_NAISSANCE mise à jour

1. Positionner le champ TRAITE = 0 pour toutes les lignes de la table ES_TABLE_VILLE
2. Si le nombre des lignes de la table ES_TABLE_VILLE qui ont le champ TRAITE =0 est égale 0 fin de traitement.
3. L1 = Parmi les lignes de la table ES_TABLE_VILLE qui ont le champ TRAITE = 0, celle dont le champ VILLE_NOM_MATCHING est le plus long.
4. Chercher toutes les lignes de la table ES_LIEU_NAISSANCE dont le champ X_LIEU_NAISSANCE est vide et le champ LIEU_NAISSANCE contient le champ l1. VILLE_NOM_MATCHING.
5. Pour toutes ces lignes, positionner le champ X_LIEU_NAISSANCE à l1.ES_TABLE_VILLE_NUM et le champ TYPE_LIEU_NAISSANCE à la valeur 'VILLE'.
6. Positionner le champ l1.TRAITE = 1
7. Revenir à 2

qui explique l'importance de commencer chaque itération (étape 2 de l'algorithme 4) par la ville dont le nom de matching est le plus grand. Prenons l'exemple des trois villes suivantes :

- La ville de *Farges Allichamps* dans le département *Cher (18)*, VILLE_NOM_MATCHING = FARGES ALLICHAMPS
- La ville de *Farges* dans le département *Ain (01)*, VILLE_NOM_MATCHING = FARGES
- La ville de *Allichamps* dans le département *Haute-Marne (52)*, VILLE_NOM_MATCHING = ALLICHAMPS

Si nous commençons par chercher dans la table ES_LIEU_NAISSANCE tous les lieux de naissance qui contiennent *FARGES* et les rapprocher avec la ville de *Farges* avant de chercher les lieux qui contiennent *FARGES ALLICHAMPS* et de les rapprocher avec la ville de *Farges-Allichamps*, nous allons considérer que le lieu de naissance normalisé de tous les dirigeants nés à *Farges-Allichamps* est la ville de *Farges*. Ce qui est faux car la ville *Farges-Allichamps* et la ville *Farges* sont bien des villes différentes. De même, si nous commençons par chercher dans la table ES_LIEU_NAISSANCE tous les lieux de naissance qui contiennent *ALLICHAMPS* et les rapprocher avec la ville d'*Allichamps* avant de chercher les lieux qui contiennent *FARGES ALLICHAMPS* et les rapprocher avec la ville de *Farges-Allichamps*, nous allons considérer que le lieu de naissance normalisé de tous les dirigeants nés à *Farges-Allichamps* est la ville de *Allichamps*. Ce qui est faux car la ville *Farges-Allichamps* et la ville *Allichamps* sont des villes différentes.

L'ordre définit dans l'étape 2 de l'algorithme 4 permet d'éviter cette erreur. En effet, parmi les trois villes de notre exemple, c'est bien la ville de Farges-Allichamps qui a le champ VILLE_NOM_MATCHING le plus long, nous commençons donc par celle-ci avant les deux autres. Nous cherchons dans la table ES_LIEU_NAISSANCE tous les lieux non rapprochés dans les étapes précédentes, c'est-à-dire X_LIEU_NAISSANCE est vide, et dont le champ LIEU_NAISSANCE contient *FARGES ALLICHAMPS* pour les rapprocher à la ville *Farges-Allichamps* en positionnant leur champ X_LIEU_NAISSANCE. Ensuite nous faisons les mêmes opérations pour la ville *ALLICHAMPS* puis pour la ville *FARGES* et à chaque fois on ne cherche dans la table ES_LIEU_NAISSANCE que les lignes non approchées dans les étapes précédentes.

Des villes, dont les noms sont inclus dans ceux des autres villes, sont très nombreuses (plus de 20000 villes ou communes). Nous pouvons en citer à titre d'exemple :

- La ville de Saint Quentin située dans le département Aisne (02) et la ville de Marcillac située dans le département Gironde (33) dont les noms sont inclus dans celui de la ville de Marcillac-Saint-Quentin située dans le département Dordogne (24).

- La ville de Angerville située dans le département Essonne (91) et la ville de Bailleul située dans le département Nord (59) dont les noms sont inclus dans celui de la ville de Angerville-Bailleul située dans le département de la Seine-Maritime (76).
- La ville de Paris dans le (75) et la ville de l'Hôpital dans le département Moselle (57) dont les noms sont inclus dans celui de la ville de Paris-l'Hôpital située dans le département Saône-et-Loire (71)

4.4.1.3 Utilisation des villes étrangères

L'application des algorithmes proposées dans les sections 4.4.1.1 et 4.4.1.2, a permis de normaliser les lieux de naissance de 97% des dirigeants. Ce résultat était considéré comme très positif, mais l'objectif, défini par les équipes métiers d'Altares, était d'aller jusqu'au 98% des lieux de naissance. Nous avons donc procédé à une analyse d'un échantillon des lieux de naissance non normalisés par le traitement proposé dans les sections 4.4.1.1 et 4.4.1.2. Cette analyse a permis de constater que les lieux de naissance, qui n'ont pas été normalisés, ne l'ont pas été à cause de l'une de deux raisons suivantes :

- Certains des lieux de naissance contiennent des erreurs de saisie.
- Certains des lieux de naissance situés à l'étranger ne contiennent pas le nom du pays et contiennent uniquement le nom de la ville. Par exemple Rabat, London, Tunis, Alger, Lisbonne.

Pour améliorer notre taux de normalisation, nous avons décidé de nous intéresser au deuxième problème car ce problème concernait plus de cas. Comme indiqué auparavant, Altares fait partie du réseau mondial Dun & Bradstreet appelée D&B. Nous avons donc sollicité D&B pour avoir un référentiel des villes du monde. Ce référentiel était sous forme d'une table de structure simple (Ville, Pays). Á partir de ce référentiel, nous avons construit une table ES_TABLE_VILLE_MONDE dont la structure est donnée par le Tableau 4.9.

L'idée est de se baser sur le nom de la ville pour retrouver le pays de naissance. Ceci sera possible uniquement pour les villes dont les noms n'existent que dans un seul pays. Par exemple, en *Mauritanie*, il y a une ville qui s'appelle *Rosso* et il existe au *Sénégal* une ville qui porte le même nom. De ce fait, si nous trouvons dans la table ES_LIEU_NAISSANCE un lieu qui vaut *Rosso*, on ne saura pas s'il s'agit de la ville de *Rosso* qui est en *Mauritanie* ou la ville de *Rosso* qui est au *Sénégal*. C'est pourquoi nous commençons par supprimer, de notre référentiel des villes étrangères, tous les noms de villes qui existent dans deux pays différents. Ceci permettra d'assurer qu'un nom de ville n'existe dans notre table ES_TABLE_VILLE_MONDE qu'une seule fois et donc dans un seul pays.

Champ	Type	Descriptif
NOM_VILLE	VARCHAR2(255 CHAR)	Le nom de la ville étrangère
NOM_PAYS	VARCHAR2(255 CHAR)	Le nom du pays étranger dans lequel se situe la ville
ES_TABLE_PAYS_NUM	NUMBER	L'identifiant du pays étranger dans lequel se situe la ville. Ce champ permettra de faire le lien avec la table ES_LIEU_NAISSANCE
VILLE_NOM_MATCHING	VARCHAR2(255 CHAR)	Champ utilisé pour faire le rapprochement avec la table ES_LIEU_NAISSANCE. Ce champ est obtenu en faisant les opérations suivantes sur le nom de la ville étrangère : – Passer en majuscule – Remplacer les caractères accentué et l'apostrophe
TRAITE	NUMBER(1)	Champ qui permettant de savoir si nous avons déjà traité ce champ ou nom.

TABLE 4.9 – La structure de la table ES_TABLE_VILLE_MONDE

Une variante de l'algorithme proposé dans la Section 4.4.1.1, mais qui se base sur le nom de la ville étrangère pour faire le rapprochement avec ES_LIEU_NAISSANCE, a été proposée et appliquée aux lieux de naissance non normalisé dans les sections 4.4.1.1 et 4.4.1.2. Cette variante est détaillé dans l'Algoritme 5.

La mise en place de cet algorithme a permis d'améliorer les résultats obtenus dans les sections 4.4.1.1 et 4.4.1.2 en passant le taux de normalisation de 97% à 99% permettant ainsi d'atteindre l'objectif de cette partie qui consistait à normaliser les lieux de naissance de 98% des dirigeants.

Notons que les traitements de normalisation des lieux de naissance ont été développés en PL/SQL.

4.4.2 Similarité sur le nom et le prénom

Les informations qui identifient une personne et notamment la date de naissance et le lieu de naissance ne sont pas toujours renseignées. Or deux personnes portant le même nom et le même prénom peuvent être différents. C'est le cas des homonymes. Il n'est donc pas très fiable de considérer deux personnes comme identique en se basant uniquement sur le nom et le prénom.

Algorithme 5 : L'algorithme de normalisation des lieux de naissance situés à l'étranger en utilisant les noms des villes

Data : les deux tables ES_TABLE_VILLE_MONDE et ES_LIEU_NAISSANCE

Result : la table ES_LIEU_NAISSANCE mise à jour

1. Positionner le champ TRAITE = 0 pour toutes les lignes de la table ES_TABLE_VILLE_MONDE
2. Si le nombre des lignes de la table ES_TABLE_VILLE_MONDE qui ont le champ TRAITE =0 est égal 0 alors fin de traitement.
3. L1 = Parmi les lignes de la table ES_TABLE_VILLE_MONDE qui ont le champ TRAITE = 0, celle dont le champ VILLE_NOM_MATCHING est le plus long.
4. Chercher toutes les lignes de la table ES_LIEU_NAISSANCE dont le champ X_LIEU_NAISSANCE est vide et le champ LIEU_NAISSANCE contient le champ l1.VILLE_NOM_MATCHING.
5. Pour toutes ces lignes, positionner le champ X_LIEU_NAISSANCE à l1.ES_TABLE_PAYS_NUM et le champ TYPE_LIEU_NAISSANCE à la valeur 'PAYS'.
6. Positionner le champ l1.TRAITE = 1
7. Revenir à 2

Dans cette section, nous nous intéressons aux attributs nom et prénom des dirigeants pour lesquels la date de naissance et le lieu de naissance sont connus. Les nom et prénom sont les plus concernés par les erreurs de saisie et les fautes d'orthographe. Nous souhaitons, lors de la comparaison des noms et prénoms de deux identités $p_1, p_2 \in P$, pouvoir tenir compte des anomalies que peuvent subir ces champs.

Soit n le nombre de dirigeants (nombre d'éléments de l'ensemble P), une comparaison de ces dirigeants 2 à 2, a un coût de $\frac{n\times(n-1)}{2}$. Dans notre cas $n = 8798005$. Le coût de cette comparaison est donc très élevé. Par conséquence, nous avons eu recours à une méthode de *clustering* permettant de réduire considérablement ce coût.

4.4.2.1 Clustering des dirigeants

La méthode de clustering (regroupement) des données en clusters a été utilisée dans la littérature afin de regrouper les champs de grande similarité et améliorer la résolution du problème [11, 42]. Nous avons donc pensé à créer une partition et mettre les personnes comparables dans le même groupe afin de linéariser le nombre de comparaisons. Pour cela nous cherchons une partition $\mathcal{P} = (P_1, ..., P_k)$ des identités P tel que toutes les identités susceptibles de concerner la même personne se trouvent dans le même ensemble. Ainsi au lieu de comparer 2 à 2 tous les éléments de P, nous ne comparons que les éléments d'un même ensemble.

Soit $\mathcal{P} = (P_1, ..., P_k)$ une partition de P. Pour tout $i \in \{1, ..., k\}$, on note par n_i le nombre d'élément de l'ensemble P_i. Nous savons que $\sum_{i=1}^{k} n_i = n$. Ainsi le coût de la comparaison 2 à 2 de tous les éléments du même ensemble est

$$\sum_{i=1}^{k} \frac{n_i \times (n_i - 1)}{2}$$

Si $k = 1$, ce coût vaut $\frac{n\times(n-1)}{2}$ et plus k est grand, plus le nombre total de comparaison devient faible.

Nous avons partitionné P par date de naissance, lieu de naissance normalisé et par *Soundex*(prénom). Les motivations de ce partitionnement sont les suivantes :

- La date de naissance : Il est inutile de comparer une personne né le *12/02/1968* avec une autre personne né le *05/08/1975*, ces deux personnes ne seront jamais considérées comme identique. D'autre part, cette information est la plus fiable parmis les 4 inforations qui identifient une personne. En effet, vu qu'elle est sous format de date, elle n'est pas concernée par les anomalies que peuvent subir les chaînes de caractères.

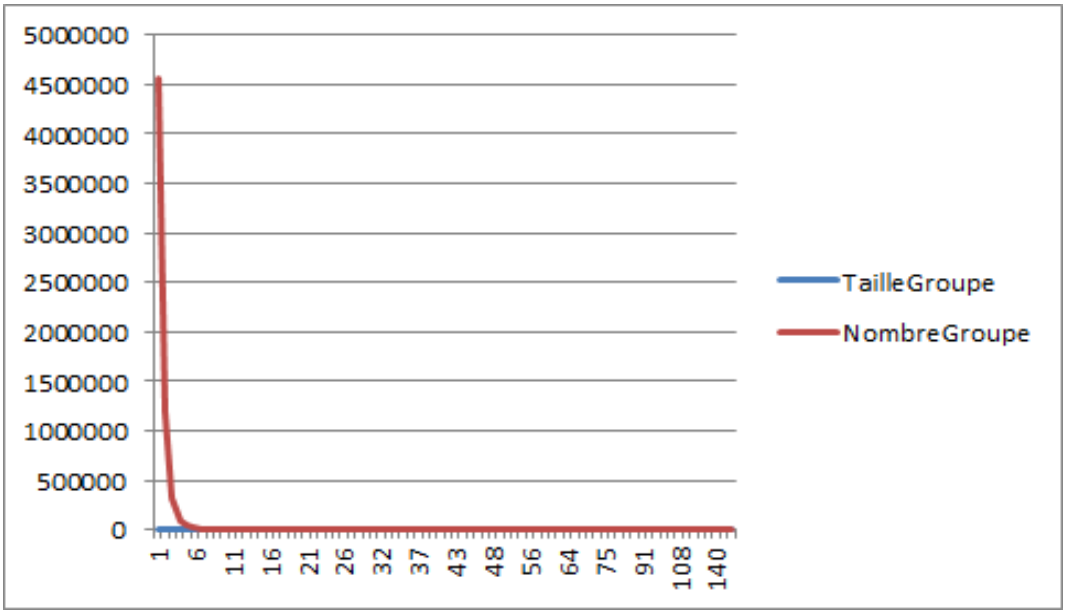

FIGURE 4.3 – Courbe donnant le nombre de groupes par taille

- Le lieu de naissance normalisé : Il est inutile de comparer une personne née à *Marseille* avec une autre personne née à *Lyon*, les deux personnes ne seront jamais considérées comme identiques. D'autre part, les travaux effectués dans la section 4.4.1 ont permis de normaliser les lieux de naissance de 99% des dirigeants.
- Le *Soundex*(prénom) : Cela nous évitera de comparer une personne dont le prénom est *François* avec une personne dont le prénom est *Nicolas*, les deux personnes se seront jamais considérées comme identique. D'autre part, comme vu dans l'état de l'art, le *Soundex* est un encodage très simple et très grossier qui permet de donner le même code à toutes les chaînes similaires. Sa simplicité et sa rapidité, dû à son coût linaire, sont des atouts très intéressant. Son seul inconvénient est le fait qu'il donne la même valeur à deux chaînes assez différentes, par exemple *Grégory* et *Grégoire* ont le même *Soundex*, *Ridha* et *Riad* ont le même *Soundex* aussi. Cet inconvénient n'est pas gênant pour cette utilisation. En effet, les personnes qui sont dans le même ensemble ne sont pas considérées comme identiques mais comme comparables. La méthode de comparaison de ces personnes est décrite dans la section suivante.

Chaque ensemble (groupe), contient toutes les identités qui ont la même date de naissance, même lieu de naissance normalisé et même *Soundex*(prénom). Plus formellement, nous avons créé $\mathcal{P} = (P_1, ..., P_k)$ une partition de P tel que pour tout $r \in \{1, ..., k\}$, p_i et $p_j \in P_r$ si et seulement si p_i et p_j ont la même date de naissance, même lieu de naissance normalisé et même *Soundex*(prénom).

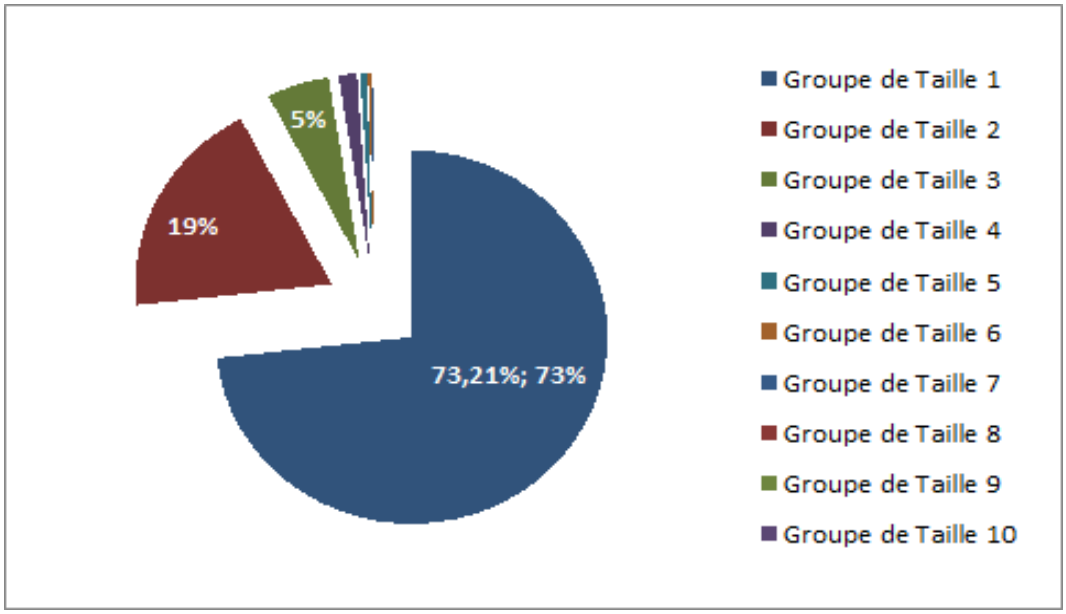

FIGURE 4.4 – Camembert donnant le nombre de groupes par taille

En groupant par les trois paramètres décrits plus haut, nous avons obtenu 6 231 692 groupes. La Figure 4.3 donne le nombre de groupes par taille de groupe. La Figure 4.4 donne cette répartition sous forme d'un camembert.

Observations : Le tableau de l'annexe 1, détaille les données des Figures 4.3 et 4.4 en donnant sous forme d'un tableau la répartition du nombre de groupes par taille. L'étude de cette répartition permet de faire les observations suivantes :

- 4 562 169 groupes (soit 73% des groupes) sont des groupes de taille 1 pour lesquels aucune comparaison n'est à faire.
- 1 190 507 groupes (soit 19% des groupes) sont des groupes de taille 2, c'est-à-dire contenant deux personnes. Pour chacun de ces groupes une seule comparaison est nécessaire. Il s'agit tout simplement de comparer les deux personnes du groupe pour savoir si elles sont identiques ou non. Notons que les groupes de taille 1 et les groupes de taille 2 représentent à eux seuls 92% des groupes.
- Le groupe qui contient le plus des personnes en contient 189.
- Le nombre total de comparaison nécessaire est 4046955. Ce nombre est 10^6 fois plus petit que $\frac{n\times(n-1)}{2}$ avec $n = 8798005$. Le calcul de ce nombre est détaillé dans l'annexe 1. La partition que nous avons proposée a donc permis de diviser par 10^6 le nombre de comparaisons nécessaires.

Les groupes de taille 1, c'est-à-dire contenant une seule personne, ne contiennent pas de doublon. Par conséquence, ces groupes ne seront pas considérés dans la suite de l'étude. Soit $\mathcal{P}' = \{P_i \in \mathcal{P} \mid |P_i| > 1\}$. Dans la suite de ce chapitre, nous nous limitons à l'étude de $\mathcal{P}'$.

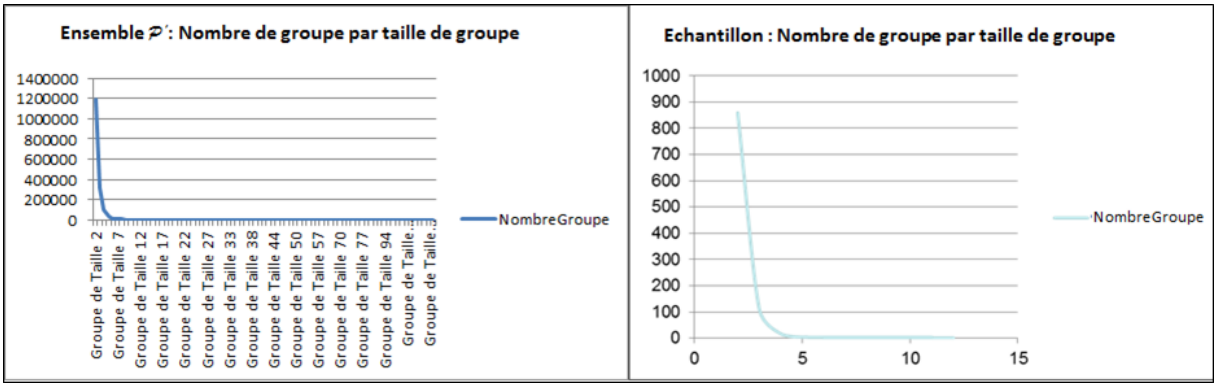

FIGURE 4.5 – étude de l'echantillon

4.4.2.2 Calcul de similarité Syntaxique

Nous avons pu normaliser les lieux de naissance pour que chaque lieu soit toujours écrit de la même façon (Section 4.4.1). Ensuite, nous avons construit une partition de $\mathcal{P}$ dont chaque ensemble, contient des personnes comparables qui ont la même date de naissance et même lieu de naissance. Puis nous avons réduit $\mathcal{P}$ à $\mathcal{P}'$ en ne gardant dans $\mathcal{P}'$ que les ensembles de $\mathcal{P}$ qui contiennent plus d'une personne. Il nous reste maintenant à savoir s'il existe parmi les personnes comparables de chaque ensemble $P_i \in \mathcal{P}'$ des personnes identiques (doublons). Pour cela, il faut comparer les noms et les prénoms des personnes de chaque ensemble $P_i \in \mathcal{P}'$ en tenant compte des anomalies que peuvent subir ces attributs de type chaîne des caractères.

Dans cette section, nous expliquons la méthode de comparaison qui a été mise en place ainsi que la démarche qui a abouti à l'élaboration de cette méthode.

Comme vu dans la Section 4.3, plusieurs méthodes de calcul de similarité syntaxique ont été proposées dans la littérature. Ces méthodes ont toutes leurs points forts et leurs points faibles et répondent chacune à un usage particulier. Nous avons souhaité mesurer l'efficacité de ces méthodes dans notre cas et sur nos propres données.

Par conséquence, nous avons identifié un échantillon représentatif de nos données. Cet échantillon est constitué de 1000 groupes. Nous pouvons constater dans La Figure 4.5 que la distribution de taille des groupes de cet échantillon est conforme à la distribution de taille de la population initiale ($\mathcal{P}'$). Nous avons étudié manuellement cet échantillon et avons identifié les doublons qu'il contient.

Ensuite, nous avons appliqué les principales mesures ou calcul de similarité syntaxique citées dans l'état de l'art sur cet échantillon pour mesurer leur efficacité

et pertinence dans notre cas. Le Tableau 4.10 donne le résultat obtenu avec ces algorithmes et contient les colonnes suivantes.

- Distance : La méthode de calcul de similarité.
- Taille échantillon : Le nombre de personnes dans notre échantillon. Cet échantillion composé de 1000 groupes contient 2500 personnes.
- Nb vrai doublons : le nombre des doublons que contient notre échantillion. Ces doublons ont été identifiés suite à l'analyse manuelle de l'échantillion par les experts métier d'Altares.
- Nb Matching : Le nombre de doublons identifiés par la mesure de similarité.
- Nb Matching Correct : Parmi les personnes considérées comme identiques par la mesure de similarité, le nombre des personnes qui sont vraiment identiques (doublons).
- Pr Matching Correct $= \frac{(\text{Nb Matching Correcte}) \times 100}{\text{NB Matching}}$
- Pr Doublons trouvé $= \frac{(\text{Nb Matching Correcte}) \times 100}{\text{NB vrai doublons}}$
- Nb faux Matching : Parmi les personnes considérées comme identiques par la mesure de similarité, le nombre des personnes qui ne sont pas vraiment identiques.
- Pr faux Matcging $= \frac{(\text{Nb faux Matcging}) \times 100}{\text{NB Matching}}$

Notons qu'il y a deux métriques intéressantes permettant d'évaluer les résultats obtenus par chaque calcul de similarité. Il s'agit du pourcentage *Pr Matching Correcte* et du *Pr Matching Faux*. Le *Pr Doublons trouvé* seul n'est pas suffisant pour apprécier la qualité de la mesure de similarité. Il faut aussi tenir compte du nombre de personnes considérées à tort comme identiques. Rappelons que, les données des dirigeants sont des données légales et sensibles. Nous avons auparavant expliqué l'impact négatif de considérer comme différentes deux personnes qui sont identiques. Mais d'autre part, considérer comme identique deux personnes qui sont différentes, revient à considérer qu'une personne dirige une entreprise qu'il ne dirige pas. Cela peut avoir des conséquences lourdes qui peuvent aller jusqu'à engager la responsabilité pénale d'Altares.

Comme nous pouvons le constater dans le Tableau 4.10 les deux mesures qui donnent les résultats le plus intéressants sont la *distance de Levenshtein* et le *3-gram*. Notons que la distance de *Jaro Winkler* permet dans notre cas d'identifier 97% des doublons, mais par ailleurs considère beaucoup de personnes comme identique alors qu'elles sont différentes (*Pr Matching Faux* = 41%).

Distance/mesure de similarité	Taille échantillon	Nb doublons	Nb Matching	Nb Matching Correcte	Pr Matching Correcte	Pr Doublons trouvé	NB faux Matching	Pr faux Matching
Hamming distance	2500	895	710	570	80,28%	63,69%	140	19,72%
Levenshtein distance	2500	895	755	618	81,85%	69,05%	137	18,15%
Needleman-Wunch distance	2500	895	727	582	80,06%	65,03%	145	19,94%
Smith-Waterman distance	2500	895	677	567	83,75%	63,35%	110	16,25%
Smith-Waterman-Gotoh distance	2500	895	895	640	71,51%	71,51%	255	28,49%
Monge Elkan distance	2500	895	827	605	73,16%	67,60%	222	26,84%
Jaro distance metric	2500	895	1330	833	62,63%	93,07%	497	37,37%
Q-gram (q=3)	2500	895	730	643	88,08%	71,84%	87	11,92%
Jaro Winkler	2500	895	1490	870	58,39%	97,21%	620	41,61%
Dice's Coefficient	2500	895	1212	787	64,93%	87,93%	425	35,07%
Jaccard Similarity	2500	895	1250	808	64,64%	90,28%	442	35,36%
Euclidean distance	2500	895	1000	775	77,50%	86,59%	225	22,50%
Cosine similarity	2500	895	1045	798	76,36%	89,16%	247	23,64%

TABLE 4.10 – Tableau de comparaison des différents métriques de similarité

Afin d'améliorer la détection des doublons, on a recours à la combinaison de plusieurs distances de calcul [60]. Nous avons souhaité étudier une méthode basée sur deux mesures de similarité. Pour cela, nous avons combiné *Levenshtein* et *3-gram*. Le Tableau 4.11 permet de comparer les résultats obtenus par la combinaison de ces deux mesures de similarité avec les résultats obtenus par chacune de ces deux mesures.

Nous constatons que la combinaison de *Levenshtein* et *3-gram* donne un résultat très satisfaisant. Notons que la marge d'erreur considérée comme acceptable dans le domaine est 90% de vrai et 10% de faux. Les résultats obtenus donne 97% de vrai et 3% de faux. Par conséquence, nous avons retenu cette méthode pour la comparaison des noms et prénoms.

4.4.3 Assouplissement des règles

Nous avons évoqué dans la Section 4.4.2 le problème de présence des informations de date de naissance et de lieu de naissance. La comparaison faite dans la section précédente ne s'applique qu'aux personnes dont on connait la date et lieu de naissance. Dans cette section nous nous intéressons aux personnes pour lesquelles on ne connait pas la date et/ou le lieu de naissance et qui dirigent la même société.

La probabilité que deux personnes ayant le même nom et le même prénom soient identiques est plus grande dans le cas où ces deux personnes dirigent la même société que le cas où elles n'ont aucune société en commun. Pour ces raisons, nous avons assouplie les règles de déduplication pour les personnes qui dirigent la même société.

L'assouplissement des règles pour les dirigeants de la même entreprise, s'est basé sur trois axes :

- Date et/ou lieu de naissance non connu(s) : si deux personnes portent le même nom et prénom, dirigent la même entreprise, et que pour l'une (ou les deux) nous n'avons pas d'information sur la date et/ou le lieu de naissance, nous considérons que ces deux personnes sont identiques. Par exemple, "*Mr DUPONT Jean né le 12/11/1967 à Paris dirigeant de la société Altares*" et "*Mr DUPONT Jean dont la date de naissance et le lieu de naissance* sont inconnus dirigeant de la société *Altares*" sont considérées comme identiques.
- Prénom composé : si deux personnes ont le même nom, même date de naissance et même lieu de naissance, dirigent la même entreprise, et que le prénom de l'une est inclus dans le prénom de l'autre, nous considérons que ces deux personnes sont identiques. Par exemple, *Mr DUPONT Jean né le 12/11/1967 à Paris* dirigeant de la société Altares et *Mr DUPONT Jean-pierre, né le*

Distance/mesure de similarité	Taille échantillon	Nb doublons	Nb Matching	Nb Matching Correcte	Pr Matching Correcte	Pr Doublons trouvé	NB faux Matching	Pr faux Matching
Levenshtein distance	2500	895	755	618	81,85%	69,05%	137	18,15%
Q-gram (q=3)	2500	895	730	643	88,08%	71,84%	87	11,92%
Levenshtein et 3-gram	2500	895	685	665	97,08%	74,30%	20	2,92%

TABLE 4.11 – Tableau de comparaison de la distance de Levenshtein, du 3-gram et d'une combinaison des deux

12/11/1967 à Paris dirigeant de la société *Altares* sont considérées comme identiques.

- Nom composé : si deux personnes ont le même prénom, la même date de naissance et même lieu de naissance, dirigent la même entreprise, et que le nom de l'une est inclus dans le nom de l'autre, nous considérons que ces deux personnes sont identiques. Par exemple, *Mr DUPONT LEGRAND Jean né le 12/11/1967 à Paris* dirigeant de la société Altares et *Mr DUPONT Jean, né le 12/11/1967 à Paris* dirigeant de la société *Altares* sont considérées comme identiques.

4.4.4 Résultats obtenus

La méthode de résolution décrite dans les sections précédentes a donnée des résultats satisfaisants sur l'échantillon qui a servi à l'étude. Ensuite, cette méthode a été appliquée sur la base complète des dirigeants.

Le Tableau 4.12 donne les outils de développement utilisés pour chaque partie de cette méthode en distinguant le traitement d'initialisation des données du traitement quotidien intégré dans le processus d'alimentation de la base de données BDD_DIR.

La méthode de résolution proposée a permis de :

- Normaliser les lieux de naissance de 99% des dirigeants.
- Identifier et supprimer 4 millions de doublons dans la base de données BDD _DIR. Ainsi cette base de données passe d'environ 10 millions à environ 6 millions d'entregistrements.
- Reconstituer les dates de début et fin de fonction des dirigeants.
- Identifier 3 millions d'entreprises qui ont au moins un dirigeant en commun.
- Construire la liste d'expérience de chacun des dirigeants de notre base de données.
- Par ailleurs, un référentiel des personnes proches a été créé. Ce référentiel contient les personnes identifiées comme similaire sans que cette similarité soit suffisamment forte pour les considérer comme identiques. Ce référentiel pourra répondre à des besoins particuliers comme par exemple la détection de la fraude.

Ensuite, cette méthode a été intégrée au processus d'intégration des données et de mise à jour de la base de données BDD_DIR.

Notons que les résultats obtenus ont été validés par les équipes métiers d'Altares selon trois axes :

Traitement	*Traitement intiale*	*Traitement de Mise à jour*
Normalisation Lieu de naissance	PL/SQL	
la création des groupes	PL/SQL	Java et SQL
Calcul de similarité	Java Utilisation la librairie SimMetrics (Cette librairie propose une implémentation des principales métriques de calcul de distance entre les chaîne de caractère) l'annexe 1 donne plus de détail sur cette librairie JAVA	Java utilisation de la librérie
Suppression des doublons	PL/SQL	(Néant on évite de créer des doublons)

TABLE 4.12 – Tableau des outils utilisées

- La recette interne : Une équipe de testeurs s'est occupée de spécifier un cahier de tests, de l'exécuter et de valider le fait que les résultats obtenus par la méthode proposée correspondent aux résultats attendus.
- Une validation par rapport au fichier *FIBEN* de la Banque de France : Ce fichier permet d'avoir la liste des entreprises que dirige une personne. En effet, certains clients nous ont fait remarquer des cas où le parcours du dirigeant est incomplet par rapport à celui du fichier FIBEN. Suite aux résultats de la déduplication des dirigeants, nos équipes ont donc pu reprendre ces cas et vérifier que le parcours proposé par Altares devient conforme à celui du fichier FIBEN.
- Les remontées clients : Plusieurs anomalies sur la date de début et fin de fonction, le parcours des dirigeants actuels d'une entreprise, le parcours des anciens dirigeants d'une entreprise, avaient été remontées à Altares par des clients utilisateurs de ces données. Après la déduplication des dirigeants, nos équipes ont repris ces cas pour valider que les anomalies signalées ont été résolues.

Conclusion

Du point de vue technique, le problème PDD a été résolu. La méthode de résolution a été appliquée sur la base BDD_DIR et intégré au processus quotidien de mise à jour. Du point de vue fonctionnel, les résultats obtenus ont été validés par les équipes métiers d'Altares. Une étude a été lancé par les équipes Marketing

d'Altares afin de comparer la qualité des informations des dirigeants affichées dans les solutions d'Altares avec la qualité de ces mêmes informations dans les solutions proposées par les concurrents d'Altares. Cette étude a conclu que suite à nos travaux, les informations des dirigeants proposées par Altares sont les meilleures sur le marché.

Par ailleurs, notre travail a été présenté à plusieurs de nos clients dans différents domaines dont nous citons à titre d'exemple : télécommunication (présentation chez *France-Telecom*), banques et assurances (présentation chez *HSBC*, puis une présentation à la *BPI* : Banque Publique d'Investissement), les institutions (Présentation à la *DGFIP* : Direction Générale de Finance Publique). Ces présentations ont été très bien accueillies. Les résultats obtenus ainsi que la démarche suivie par Altares ont été appréciés.

Suite aux résultats obtenus et la façon dont ils ont été accueillis par les clients, Altares a décidé de lancer sur le marché du BtoB une solution dédiée sur les données des dirigeants, leurs expériences actuelles et passées. Cette solution innovante sera la première de ce type sur ce marché. Elle est appelée *AltaDir*. AltaDir est encours de conception chez Altares. Un prototype a déjà été réalisé et présenté au sein de l'entreprise début décembre 2013. L'annexe 1 donne plus d'information sur AltaDir.

Chapitre 5

Liens Capitalistiques

Sommaire

5.1 Introduction

Les liens capitalistiques d'une entreprise font partie des informations importantes qui sont évaluées, appreciées et prises en compte par ses clients, ses fournisseurs et ses partenaires. Connaître les liens capitalistiques d'une entreprise permet d'avoir une vision plus complète sur celle-ci, sa solidité, sa pérennité et ses capacités à surmonter les difficultés. Par exemple, une entreprise en difficulté, qui a un actionnaire puissant, peut compter sur son aide pour surmonter ses difficultés. D'autre part, ces informations sont indispensables pour une entreprise qui veut se déployer dans un groupe ou mettre en place un contrat cadre. La composition des liens capitalistiques de chaque société est déclarée dans un document officiel nommé *statuts de l'entreprise*. Cependant, les informations des liens capitalistiques ne sont pas diffusées par les sources officielles. Des entreprises privées se sont spécialisées dans la collecte et la diffusion de ces informations. Les entreprises les plus connues sur ce marché sont DAFSA-LIENS et BvD.

De son côté, Altares a toujours proposé les informations des liens capitalistiques à ses clients dans le cadre de son offre globale de données d'entreprises. Dans le passé, les informations des liens capitalistiques diffusées par Altares provenaient de DAFSA-LIENS. En 2008, Altares a souhaité collecter ces informations importantes par ses propres moyens. Cette décision est motivée par deux raisons. La première raison est que DAFSA-LIENS est une filiale de Coface Service le premier concurrent d'Altares. D'un point de vue stratégique, Altares ne souhaitait aucune forme de dépendance à l'un de ses concurrents. La seconde raison est que Altares souhaitait maîtriser la qualité de ses liens capitalistiques et en faire un point de différenciation. En effet, la qualité et la fraîcheur des liens capitalistiques commercialisés par DAFSA-LIENS ne correspondaient plus aux attentes d'Altares. En collectant elle-même ces informations, Altares est certaine d'améliorer la qualité ainsi que la fraicheur de ces données.

Le projet de collecte des données des liens capitalistiques a abouti en 2010. Á partir de cette date, les informations des liens capitalistiques disponibles dans les produits d'Altares sont les informations collectées par ses équipes. Comme les informations qui provenaient de DAFSA-LIENS, les informations collectées par Altares donnent pour chaque entreprise les actionnaires et les participations directs. Cette collecte des données a permis à Altares de remplacer les liens DAFSA-LIENS qui alimentaient les bases de données Altares.

Suite à la réussite de ce projet ainsi qu'à la qualité et la fraîcheur de ces informations des liens capitalistiques, Altares souhaite aller plus loin et proposer une offre complète sur les données des liens capitalistiques. Pour cela, Altares doit commencer par identifier les liens indirects non collectés ainsi que les pourcentages détenus par les actionnaires indirects dans le capital social de leurs participations. Ce problème est appelé *Calcul des Liens Indirects*. Ensuite, Altares doit déterminer les groupes, c'est-à-dire les ensembles des entreprises contrôlées par le même actionnaire, et identifier pour chaque groupe l'entreprise actionnaire qui le contrôle appelée *Tête de groupe*. Ce problème est appelé *l'Indentification des têtes de groupe*.

Dans ce chapitre nous nous intéressons aux deux problèmes cités plus haut à savoir le problème de Calcul des Liens Indirects et le problème de l'Identification des têtes de groupe. La résolution de ces deux problèmes permettra à Altares de lancer une offre complète sur les données des liens capitalistiques. Par ailleurs, ces informations combinées avec les données des dirigeants, voir le Chapitre 6, donneront les réseaux d'influence que nous cherchons à caractériser dans le chapitre suivant. En effet, une personne physique ou morale, peut être influente dans une entreprise en étant un dirigeant, un actionnaire direct ou un actionnaire indirect.

La première section de ce chapitre est dédiée à la compréhension des problèmes étudiés et aux motivations qui ont poussées à les étudier. Dans un premier temps, et

afin de mieux comprendre les données des liens capitalistiques ainsi que les problèmes posés, nous commençons par présenter la méthode de collecte mise en place par Altares et les données collectées ainsi que leurs structures. Ensuite, nous présentons de façon plus formelle les deux problèmes posés et nous nous servons d'un exemple réel pour mieux comprendre les problématiques et les questions liées. Enfin, nous donnons les motivations de ces problèmes en présentant l'utilité des données que nous souhaitons calculer pour Altares, ses produits et ses clients.

La deuxième section de ce chapitre est consacrée au calcul des liens indirects. Dans un premier temps, nous présentons l'algorithme qui a été proposé et nous le déroulons sur un exemple simple afin de mieux le comprendre. Nous expliquons aussi les conditions de convergeance de cet algorithme. Ensuite, nous présentons les résultats obtenus par l'application de cet algorithme sur les données d'Altares.

La dernière section de ce chapitre est destinée à l'identification des groupes et des têtes de groupe. Nous présentons tout d'abord, la méthode de calcul des têtes de groupe, puis nous expliquons comment déterminer le groupe contrôlé par chaque entreprise tête de groupe en se servant d'un exemple simple.

5.2 Liens Capitalistiques

Un *lien capitalistique* caractérise la relation entre 2 entités juridiques. L'une appelée *actionnaire* détenant une partie ou la totalité du capital social de l'autre, appelée *participation*. Par conséquent, on dit que la première est actionnaire de la seconde ou que la seconde est une participation de la première. L'actionnaire peut être une personne physique ou morale. La participation est forcément une personne morale. L'actionnaire a le droit de recevoir des dividendes. Il a aussi le droit de voter à l'assemblée générale des actionnaires (sauf le cas particulier des actions sans droit de vote). En cas de liquidation de l'entreprise participation, l'actionnaire a droit à un boni de liquidation.

On appelle un *groupe* l'ensemble des entreprises appartenant directement ou indirectement au même actionnaire personne morale. Cet actionnaire est appelé *tête de groupe*. La tête de groupe contrôle l'ensemble des entreprises qui forme le groupe. Une *arborescence* est la liste des entreprises appartenant au même groupe avec les détails des actionnaires et le pourcentage de détention associé.

Dans cette section dédiée à la compréhension des problèmes et leurs motivations, nous commençons par décrire la collecte des liens capitalistiques. Ensuite, nous présentons les deux problèmes posés dans ce chapitre. Enfin, nous donnons les motivations de ces problèmes.

5.2.1 Collecte des liens capitalistiques

Depuis août 2008, Altares a une équipe d'experts dédiée à la collecte des liens capitalistiques entre entreprises françaises. Cette équipe utilise les publications des entreprises et procède également à des enquêtes téléphoniques. La collecte porte sur la totalité des arborescences. En effet, les experts d'Altares explorent les arborescences jusqu'à 0,1% de taux de détention.

Dans cette section nous commençons par présenter la méthode de collecte et ses principes qui garantissent une fraîcheur optimale des données. Ensuite, nous décrivons les données collectées ainsi et donnons la couverture de cette collecte. Enfin, nous expliquons la structure des données des liens capitalistiques.

5.2.1.1 Méthode de collecte

Altares collecte les liens capitalistiques grâce à une interface web appelée l'Atelier de Collecte. Cet atelier alimente la base de données référentielle d'Altares (Repository). Les liens capitalistiques sont ensuite consultables au travers d'un widget spécifique disponible dans les produits «on-line » d'Altares. Les informations des liens capitalistiques sont aussi utilisées dans le calcul de score de défaillance des entreprises qu'Altares met à la disposition de ses clients.

La collecte s'effectue selon quatre critères :

- L'exhaustivité : l'intégralité des liens d'un groupe est revue à chaque mise à jour. Toutes les participations sont recherchées, y compris les non consolidées
- La granularité : consiste à la collecte des liens de tous les niveaux de l'arborescence qui ont un pourcentage d'actionnariat supérieur à 0,1%. Pour chaque lien, les experts collectent toutes les informations disponibles.
- L'exactitude : si l'information vient d'une publication officielle, l'interview téléphonique de la société vient la compléter ou la rafraîchir. Si l'information vient d'autres sources (par exemple la presse) elle doit obligatoirement être recoupée avec d'autres (principalement la société elle-même) avant intégration dans nos bases
- La fraîcheur : l'actualisation s'effectue par une mise à jour récurrente des informations. Cette mise à jour est garantie grâce à une veille quotidienne assurée par l'équipe des experts des liens capitalistiques. La mise à jour s'effectue en mode dit réactif, c'est-à-dire que les liens sont mis à jour dès la réception d'une alerte presse, publication, etc pour une réactivité au jour le jour et une véracité accrue.

Pour chaque entité, les experts d'Altares collectent les données des actionnaires et les données des participations jusqu'à 0,1%. Les actionnaires peuvent être français et/ou étrangers, des personnes morales et/ou des personnes physiques. Les participations sont forcément des personnes morales qui peuvent être des entreprises françaises et/ou étrangères.

Les données sont collectées selon une méthode appelée la méthode «Top-Down». Cette méthode consiste, à partir d'une entité donnée, de collecter ses actionnaires directs et ses participations directes. Ensuite on fait la même opération pour chacune des participations françaises avec un pourcentage supérieur à 10%. Pour les participations étrangères seul le premier niveau est collecté. Ainsi les enquêteurs continuent à explorer l'arborescence jusqu'à avoir l'ensemble des liens qui la composent. Chaque entreprise de cette arborescence est ensuite mise sous alerte dans le système d'Altares. Cette mise sous alerte permet d'informer l'enquêteur de toute nouvelle information concernant cette entreprise (une nouvelle annonce légale, un nouveau bilan, un nouveau dirigeant etc). En fonction de ces alertes, l'enquêteur peut déclencher une mise à jour d'une partie ou de toute l'arborescence si besoin.

Le processus de collecte peut être résumé dans les quatre étapes suivantes :
- 1ère étape :
  - Collecte des documents de références.
  - Recherche via les sites institutionnels.
- 2ème étape
  - Analyse et compréhension de l'arborescence.
  - Vérification et enquêtes par les experts Altares.
- 3ème étape
  - Saisie dans l'atelier dédié.
  - Mise à disposition des données.
- 4ème étape
  - Mise sous surveillance annonces légales et publications.
  - Vérification par nos experts et si besoin déclenchement d'une mise à jour.

5.2.1.2 Données collectées et couverture

Naturellement, les données collectées sur les personnes morales (entreprises) sont différentes de celles collectées sur des personnes physiques, la Figure 5.1 donne un exemple d'une entreprise ayant un actionnaire personne physique et un actionnaire personne morale.

Pour les actionnaires personnes physiques, les équipes d'Altares collectent les informations de la personne à savoir son nom, son nom d'usage, son prénom, sa civilité

FIGURE 5.1 – Actionnaire personne physique et actionnaire personne morale

mais aussi sa date et son lieu de naissance quand ces informations sont communiquées. Ils collectent aussi les informations décrivant l'actionnariat à savoir le pourcentage détenu, la nature du lien (majoritaire/minoritaire), sa source (publication et ou enquête), sa date de création et sa date de mise à jour.

Pour les actionnaires personnes morales, les informations collectées sont celles de la personne morale à savoir : la raison sociale, le pays (une entreprise française peut avoir un actionnaire étranger), sa ville, son activité, son SIREN (si l'actionnaire est français) et son DUNS (le *DUNS* est un identifiant unique mondial pour les entreprises, c'est le seul identifiant pour les entreprises étrangères).

Altares dispose, dans sa base de données, des informations concernant 140 000 liens capitalistiques. Ces informations couvrent l'ensemble des liens capitalistiques des 500 groupes de l'Eurolist. Elles couvrent aussi les liens capitalistiques de 150 groupes cotés au second marché.

En parallèle Altares étend de façon continue la couverture de ces liens capitalistiques en collectant des nouveaux groupes selon les critères d'exhaustivité, de fraîcheur, d'exactitude et de granularité, vu précédemment.

Les liens capitalistiques d'Altares couvrent déjà la totalité des groupes cotés et grands groupes français. La priorité se situe aujourd'hui sur les ETI(Entreprises de Taille Intermédiaire) et PME(Petites et moyennes entreprises) françaises. Cette couverture est en amélioration constante. Chaque jour, les équipes d'Altares collectent de nouveaux liens tout en garantissant une mise à jour efficace des liens déjà collectés.

5.2.1.3 Structure des données

La base de données référentielle d'Altares est conçue pour pouvoir alimenter plusieurs Datamart. Chaque Datamart est dédié à un métier particulier. Aujourd'hui, cette base de données référentielle alimente principalement un Datamart dédié aux risques de défaillance et à la solvabilité des entreprises. Ce Datamart est appelé DTMRISK. Les données des liens capitalistiques sont disponibles dans DTMRISK. Dans la suite de ce chapitre, nous désignons par BDD_LIENS les objets de DTMRISK dédiés aux liens capitalistiques. Ces objets sont composés de tables, de triggers, d'index, de clés étrangères et primaires. Comme pour la BDD_DIR, nous nous limiterons dans cette section à la description des objets de BDD_LIENS dont la compréhension est nécessaire pour la suite de ce chapitre. Il s'agit des tables suivantes :

- La table EB_ACTIONNAIRE : cette table contient toutes les informations qui caractérisent un lien d'actionnariat. Notons qu'il s'agit d'un actionnaire

personne physique ou un actionnaire personne morale, les informations qui caractèrisent un lien d'actionnariat sont les mêmes. Ces informations sont : le pourcentage de l'actionnariat, sa date de création, sa date de dernière mise à jour, sa nature (majoritaire ou minoritaire) et sa source.

- La table EX_ACTIONNAIRE_PP : Cette table permet de faire le lien entre l'entreprise, l'actionnaire personne physique et les informations qui caractérisent le lien d'actionnariat qui sont stockées dans la table EB_ ACTIONNAIRE. Cette table contient donc, une référence vers l'entreprise, une référence vers la personne physique actionnaire et une référence vers les informations qui caractérisent l'actionnariat. Notons que les informations de l'actionnaire personne physique sont stockées dans la table EB_PERSONNE présentée dans le chapitre précédent. Ceci permet de faire le lien entre les personnes physiques dirigeantes et les personnes physiques actionnaires.

- La table EX_ACTIONNAIRE_PM : Cette table permet de faire le lien entre l'entreprise, l'actionnaire personne morale et les informations qui caractérisent le lien d'actionnariat. Cette table contient donc, une référence vers l'entreprise filiale, une référence de l'entreprise actionnaire et une référence vers les informations qui caractérisent l'actionnariat.

En plus des informations citées plus haut, chaque table contient aussi des informations techniques permettant une exploitation optimale des données.

Notons qu'il n'y a pas de table particulière pour stocker les participations. Comme indiqué plus haut, un lien capitalistique peut être un lien d'actionnariat ou un lien de participation. Cependant nous avons choisi de ne stocker que les informations de liens d'actionnariat. En effet, si les experts d'Altares saisissent une information indiquant que l'entreprise A est une participation de l'entreprise B, cela veut dire que B est actionnaire de A. Nous transformons le lien de participation en lien d'actionnariat et nous ne stockons en base que des informations relatives aux liens d'actionnariat. La Figure 5.2 donne une présentation graphique de la structure des données de la base BDD_LIENS.

5.2.2 Problèmes Posés

Dans cette section, nous présentons les deux problèmes traités dans ce chapitre à savoir le calcul des liens indirects et l'identification des têtes de groupe. Nous nous appuyons sur un exemple pour expliquer ces problèmes.

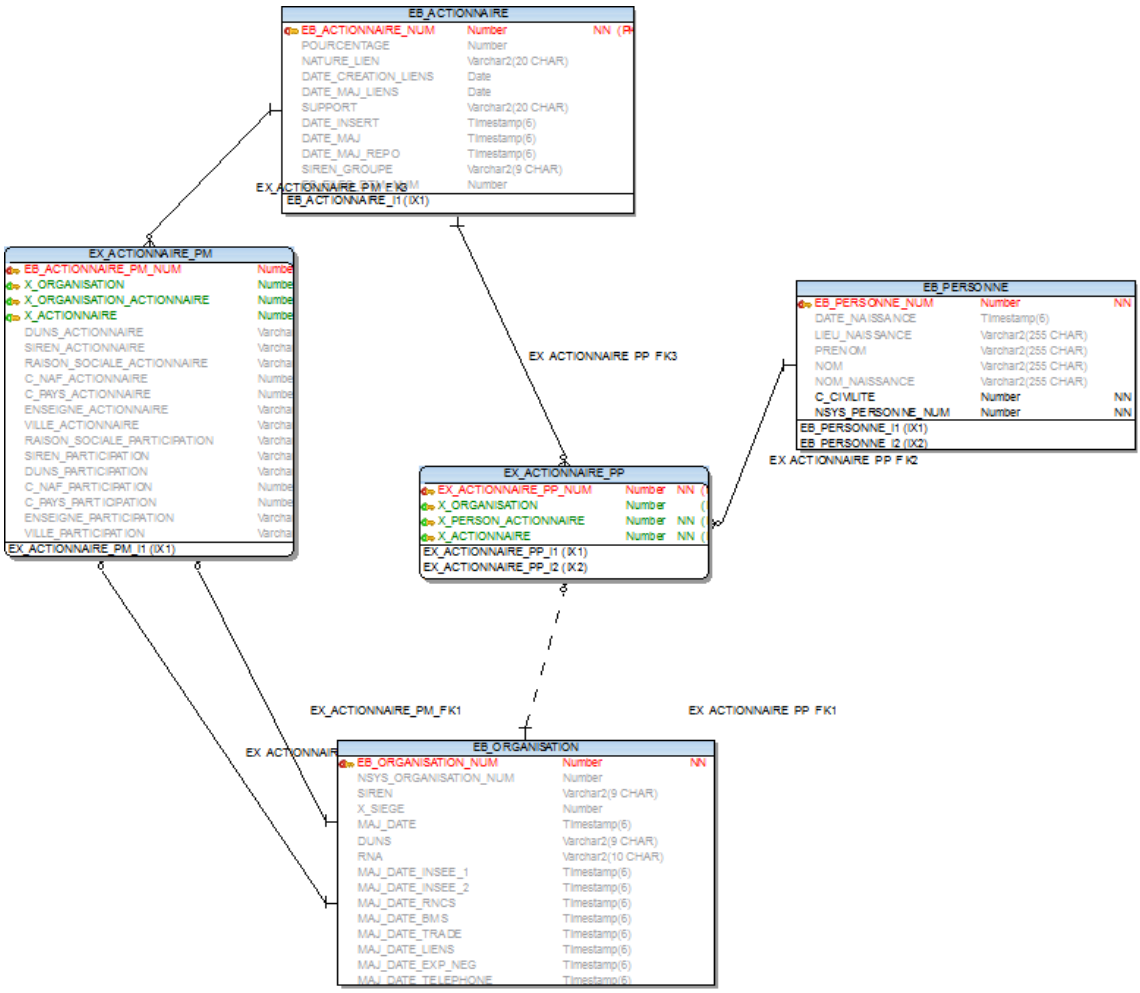

FIGURE 5.2 – Structure de la base de données BDD_LIENS

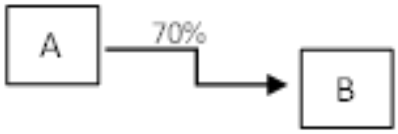

FIGURE 5.3 – Lien direct : A est actionnaire de B avec un pourcentage de 70%

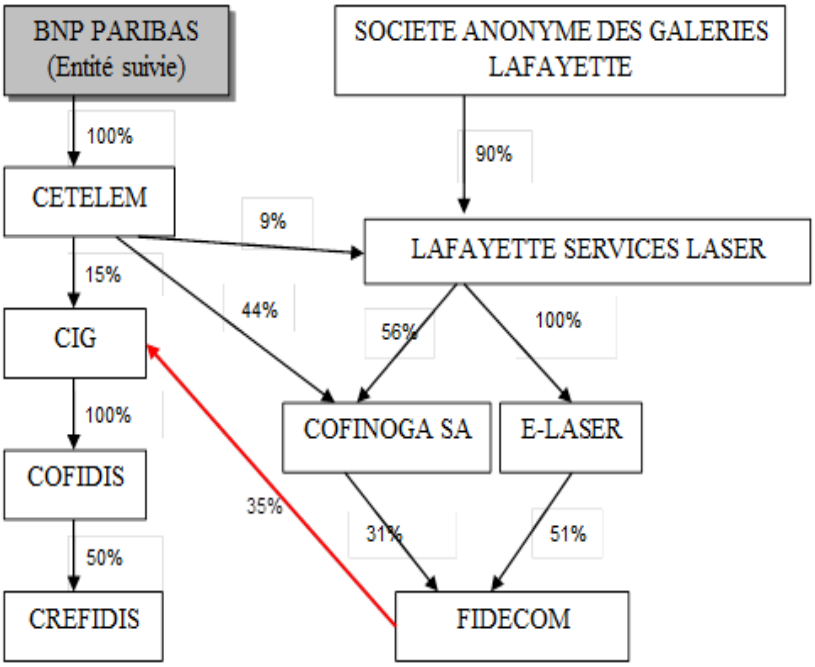

FIGURE 5.4 – Exemple de liens capitalistiques

5.2.2.1 Liens Indirects

La collecte permet d'identifier et de caractériser les liens capitalistiques directs. La Figure 5.3 donne un exemple de lien capitalistique direct entre deux entités.

Une entreprise peut être actionnaire d'une autre de façon indirecte. Prenons l'exemple donné par la Figure 5.4, l'entreprise SOCIETE ANONYME DES GALERIES LAFAYETTE est actionnaire de l'entreprise LAFAYETTE SERVICE LASER. Cette dernière est actionnaire de l'entreprise COFINOGA SA. En conséquence, SOCIETE ANONYME DES GALERIES LAFAYETTE est actionnaire indirect de l'entreprise COFINOGA SA. Dans cet exemple, CETELEM détient directement 15% du capital social de l'entreprise CIG. CETELEM détient aussi indirectement un autre pourcentage du capital social de l'entreprise CIG. Pour calculer ce pourcentage, il faut identifier tous les chemins qui relient CETELEM et CIG puis calculer le pourcentage détenu sur chaque chemin et ensuite nous effectuons la somme de

ces pourcentages pour avoir le pourcentage que détient indirectement CETELEM dans le capital social de CIG. Ce pourcentage détenu indirectement vient s'ajouter au pourcentage direct pour avoir le pourcentage total que détient CETELEM dans le capital social de CIG.

Comme nous pouvons le constater sur cet exemple : CETELEM est actionnaire à 9% de LAFAYETTE SERVICE LASER. Ce dernier est actionnaire à 58% de CONFINOGA SA. De sa part, COFINOGA SA est actionnaire de FIDECOM à 31%. Enfin, FIDECOM est actionnaire à 35% de CIG. Par cet enchainement, appelée dans la suite de ce chapitre *chemin*, CETELEM détient indirectement $9 \times \frac{58}{100} \times \frac{31}{100} \times \frac{35}{100} = 0,57\%$ du capital social de l'entreprise CIG.

Il existe deux autres chemins indirects entre CETELEM et CIG. Le premier passe dans l'ordre par LAFAYETTE SERVICE LASER, E-LASER et FIDECOM. Par ce chemin CETELEM détient indirectement $9 \times \frac{100}{100} \times \frac{51}{100} \times \frac{35}{100} = 1,6\%$ du capital social de l'entreprise CIG. Le deuxième passe par COFINOGA SA puis FIDECOM. Par ce deuxième chemin, CETELEM détient indirectement $44 \times \frac{31}{100} \times \frac{35}{100} = 4,78\%$ du capital social de l'entreprise CIG. En conséquence, CETELEM détient indirectement 0,57 + 1,6 + 4,78 = 6,95 % du capital social de CIG. Finalement le pourcentage total que détient directement et indirectement CETELEM dans le capital social de CIG est égal 6,94 + 15 = 21,94%.

Le calcul des liens indirects consiste à identifier pour chaque entreprise de la base BDD_LIENS l'ensemble des actionnaires indirects. Puis pour chaque actionnaire, direct et/ou indirect, calculer le pourcentage total d'actionnariat.

Notons que les liens indirects vont nous permettre de déterminer les têtes de groupe et les entreprises qui composent chaque groupe.

5.2.2.2 Têtes de groupe

Une société tête de groupe est une société non contrôlée et qui contrôle elle même au moins une autre société. Cette définition pose la question de savoir à partir de quel pourcentage une société actionnaire d'une autre la contrôle. En l'absence d'une réponse juridique à cette question ce pourcentage, appelée *seuil de contrôle*, est défini en fonction du contexte et du besoin. Les valeurs admises pour ce seuil sont généralement 50%, 40%, 33.33% et 25%. Dans le cadre de cette étude, les experts d'Altares ont retenus deux seuils de contrôle à savoir 25% et 50%. En conséquence, deux types de têtes de groupe ont été définis :

- Tête de groupe 50% : Une tête de groupe est une entité française, non détenue

en consolidé à hauteur de 50% ou plus par une autre entité, et qui détient au moins une filiale en consolidé à hauteur de 50% ou plus.

- Tête de groupe 25% : Une tête de groupe est une entité française, non détenue en consolidé à hauteur de 25% ou plus par une autre entité, et qui détient au moins une filiale en consolidé à hauteur de 25% ou plus.

Dans ces définitions, le mot *consolidé* veut dire en tenant compte de tous les liens directs et indirects.

Le besoin est d'identifier les entreprises têtes des groupes 25% et les entreprises têtes des groupes 50%. Ensuite, pour chacune des entreprises tête de groupe 25% et chacune des entreprises tête de groupe 50%, il faut identifier la liste des entreprises qu'elle contrôle respectivement à 25% et 50%.

Savoir si S_1 est une tête de groupe 25%, necéssite d'identifier tous les actionnaires et participations directs et indirects consolidés de l'entreprise S_1. Cela implique que la résolution du problème de Calcul de Liens Indirects présenté dans la Section précédente est un pré-requis à l'identification des têtes de groupe.

5.2.3 Motivations des problèmes

Le traitement de ces deux problèmes est principalement motivé par deux raisons. La première est le souhait d'Altares de lancer une offre spéciale sur les données des liens capitalistiques appelée *Link-File*. La seconde résulte de l'initiative du gouvernement français d'aider les Petites et Moyennes Entreprises (PME) à se développer. Cette initiative est appelée *PactePME*.

Dans cette section nous présentons l'offre « Link File» qu'Altares souhaite lancer ainsi que l'initiative PactePME en montrant à chaque fois l'intérêt de calcul des liens indirects et l'identification des têtes de groupe.

5.2.3.1 Link-File

Dans le cadre de ses offres, Altares affiche les données des liens capitalistiques en donnant pour chaque entreprise, couverte par la collecte des données liens capitalistiques, l'ensemble des actionnaires et participations directs, mais, Altares souhaiterait aller plus loin et lancer une offre des liens en fichier capable de rivaliser avec les solutions fichiers proposées par d'autres acteurs du marché comme DAFSA-liens et BvD. Cette offre permettra de répondre aux demandes de comptage et d'extraction des données pour les clients, mais aussi aux demandes d'enrichissement des fichiers ou bases de données des cliens à partir des données de la BDD_LIENS. L'offre Link

File doit être commercialisable et permettre de générer du chiffre d'affaires pour Altares. Elle permettra aussi à Altares d'améliorer sa position dans le domaine des connaissances inter-entreprises.

L'offre Link File doit permettre de répondre aux besoins suivants :

- Se conformer à la législation : connaître les clients et leurs actionnaires ultimes. Il s'agit de se conformer à la 3ème directive (cf ordonnance du 30 janvier 2009 relative à la prévention de l'utilisation du système financier aux fins de blanchiment de capitaux et de financement du terrorisme transposant la 3ème directive anti-blanchiment du 26 octobre 2005). Dans ces conditions, le client souhaite partir d'une entité (souvent de taille moyenne ou petite) et remonter l'arborescence pour connaître l'ultime actionnaire. Les banques souhaitent en général pouvoir remonter les liens supérieurs à 25%.

  Ce besoin est très présent dans les banques et les assurances mais aussi chez les avocats et les notaires.
- Anticiper et consolider les risques financiers : les liens servent ici à consolider les risques sur une tête de groupe. Á partir d'une entité, l'objectif est de connaître la tête de groupe. La notion d'appartenance à un groupe est donc cruciale car elle sert à affiner la couverture des risques financiers. Dans cette optique, les liens apportant une vision consolidée des encours et créances.

  Comme le précédent, ce besoin est assez présent dans les banques et les assurances. Il est aussi présent dans les directions financières et dans les directions générales.
- Evaluer rapidement la solvabilité d'une entreprise : l'actionnariat fait partie des éléments consultés au moment d'une prise de décision rapide concernant l'octroi d'un crédit client. Les éléments premiers restent cependant : la notation, l'encours, les bilans, les annonces légales et les dirigeants éventuellement. Savoir qui est l'actionnaire, est considéré comme un avantage important.

  Ces sont les credit managers qui expriment en général ce besoin.
- Optimiser sa politique d'achat : le besoin consiste à rationaliser les achats. En identifiant les fournisseurs appartenant à un même groupe, on peut améliorer son pouvoir de négociation. Par ailleurs, avoir une vision consolidée des achats permet d'évaluer le degré de dépendance et le risque de défaillance des fournisseurs. C'est par l'identification d'une tête de groupe et le regroupement autour de cette tête que les clients cherchent à effectuer la consolidation.

  Ce besoin est exprimé par les directions des achats et les directions générales des grandes entreprises.
- Piloter le développement et la stratégie de l'entreprise : les liens permettent d'identifier des axes qui sont :
  - Croissance interne : optimiser la couverture commerciale et ajuster la politique commerciale (mise en place d'accord cadre, politique grands comptes),

évaluer les potentiels du marché (identification de prospects dans les filiales).
- Croissance externe : évaluer les opportunités d'acquisition d'entreprise, mesurer les conséquences de fusion/absorption.

Ce besoin est présent dans les directions marketing, commerciales et dans les directions de la stratégie.

Pour répondre à ces besoins, il est donc impératif de calculer les liens indirects et d'identifier les têtes de groupes.

5.2.3.2 PactePME

Les analyses des économistes montrent que la différence entre l'économie française, qui globalement ne va pas très bien, et l'économie Allemande, qui s'en sort mieux, réside dans la capacité des petites et moyennes entreprises à se développer, à générer de l'emploi et à exporter. En conséquence, le gouvernement français a lancé une initiative en faveur des petites et moyennes entreprises françaises appelée PactePME.

Les pouvoirs publics cherchent à agir en faveur de l'émergence des entreprises, PME(Petites et Moyennes Entreprises) et ETI(Entreprises de Taille Intermédiaire) leaders dans leur domaine, afin de corriger la faiblesse principale du tissu économique français et ses résultats globaux insuffisants. Les grands comptes, qui sont les acteurs les plus puissants, ont été mobilisés en faveur de cette cause nationale. Ils ont signé avec le gouvernement un engagement de développer une politique d'achat orientée vers les PME. Ainsi chacun des grands comptes a un objectif d'achat à réaliser avec des PME.

Cela a créé un nouveau besoin chez ces grands comptes. Ce besoin consiste à identifier les PME. La question de savoir si une entreprise est une PME ou non, peut paraître simple mais elle ne l'est pas, surtout pour les entreprises qui ont des actionnaires. En effet, il y a des conditions nécessaires relatives à la taille pour qu'une entreprise soit une PME. Or ces conditions relatives à la taille ne sont pas suffisantes si l'entreprise a un actionnaire. Par exemple, une entreprise de taille petite et qui est filiale directe ou indirecte d'une grande entreprise n'est pas considérée comme une PME. Elle n'est donc pas concernée par l'initiative PactePME.

Altares souhaite fournir aux grands comptes la catégorisation des PME et les accompagner à mettre en place une politique d'achat orientée vers les PME. Or, elle doit calculer les liens indirects et identifier les têtes de groupe afin d'être en mesure de caractériser les PME et répondre au besoin des grands comptes.

5.3 Liens Indirects

Le calcul des liens indirects est indispensable pour Altares pour les raisons expliquées dans la section précédente. Par ailleurs, ce calcul est une étape nécessaire à l'identification des têtes de groupe.

Dans cette section, nous présentons et expliquons l'algorithme qui a été proposé pour calculer les liens indirects. Nous utilisons un exemple pour illustrer le déroulement de cet algorithme. Enfin, nous donnons les résultats obtenus sur la base BDD_LIENS.

5.3.1 Modélisation en termes de graphe

Les données des liens capitalistiques peuvent être modélisées par un graphe orienté et pondéré. Les sommets de ce graphe représentent les entreprises de la base BDD_LIENS. Les arcs représentent les relations d'actionnariat entre ces entreprises et le poids des arcs correspond au pourcentage d'actionnariat.

Soit le graphe orienté $D = (V, A)$ où chaque sommet de V correspond à une entreprise et il existe un arc $(u, v) \in A$ entre deux sommets $u, v \in V$ si et seulement si l'entreprise représentée par le sommet u détient une part du capital social de l'entreprise représentée par le sommet v. On note par $w(u, v, 1) \rightarrow \mathbb{R}^+$ la fonction de pondération qui associe à chaque arc $(u, v) \in A$ le pourcentage que détient, directement (lien de niveau 1), l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v.

Au début de notre travail, le graphe D, qui représente les données des liens capitalistiques, contenait environ 65000 arcs et 44000 sommets. Un projet d'extension de couverture lancé début 2013 a permis d'augmenter considérablement la taille de notre graphe. En décembre 2013, la taille de ce graphe était de 174000 arcs et 78 000 sommets.

Étant donné deux sociétés S_1 et S_2, le pourcentage total que détient l'entreprise S_1 dans le capital social de l'entreprise S_2 correspond au pourcentage que détient directement et indirectement l'entreprise S_1 dans le capital social de l'entreprise S_2. Le calcul de ce pourcentage doit tenir compte de tous les chemins entre l'entreprise S_1 et l'entreprise S_2. Autrement dit, tous les chemins qui relient le sommet u et le sommet v dans le graphe orienté D, u étant le sommet associé à l'entreprise S_1 et v le sommet associé à l'entreprise S_2.

Plus formellement, soit D le graphe correspondant aux données des liens capitalistiques et u, v deux sommets de V. Soit $\mathcal{P}_{uv}$ l'ensemble des chemins élémentaires

entre u et v. On associe à chaque chemin $P = ((u_1, v_1), (u_2, v_2), ..., (u_k, v_k))$ avec $u_1 = u$, $v_k = v$, $vi = u_{i+1}$ pour tout $i = 1, 2, ..., k-1$ et $(ui, vi) \in A$, le poids $w(P)$ définit par

$$w(P) = 100 \times \prod_{(u_i,v_i)\in P} \frac{w(u_i, v_i, 1)}{100}. \tag{5.1}$$

$w(P)$ correspond au pourcentage que détient l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v par le lien indirect représenté par le chemin P. Le pourcentage total que détient l'entreprise S_1 dans le capital social de l'entreprise S_2, (S_1 étant l'entreprise réprésentée par le sommet u et S_2 l'entreprise réprésentée par le sommet v), est notée $w(u, v)$. Ce pourcentage, appelé aussi *taux de détention consolidé*, est donné par l'égalité suivante :

$$w(u, v) = \sum_{P\in\mathcal{P}_{uv}} w(P). \tag{5.2}$$

Cette égalité tient aussi compte de lien direct éventuel entre l'entreprise S_1 et l'entreprise S_2. En effet, si S_1 est actionnaire direct de S_2, le chemin $((u, v))$ appartient à P.

Le pourcentage que détient indirectement l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v est donné par l'égalité suivante :

$$w'(u, v) = \sum_{P\in\mathcal{P}_{uv}\subset\{(u,v)\}} w(P). \tag{5.3}$$

L'objectif est de calculer $w(u, v)$ et $w'(u, v)$ pour tout couple de sommet u et v appartenant à V tel que $u \neq v$.

La Figure 5.5 donne un exemple simple de ce graphe. Cet exemple appelé *GLCD* (Graphe des Liens Capitalistiques Directs), sera utilisé dans la suite de ce chapitre pour illustrer le déroulement de l'algorithme proposé pour le calcul des liens capitalistiques indirects.

5.3.2 Méthode de résolution

Dans cette section, nous présentons l'algorithme qui a été proposé pour calculer $w(u, v)$ et $w'(u, v)$ pour tout couple de sommet u et v appartenant à V tel que $u \neq v$. Afin de mieux comprendre cet algorithme, nous illustrons son déroulement sur un exemple simple.

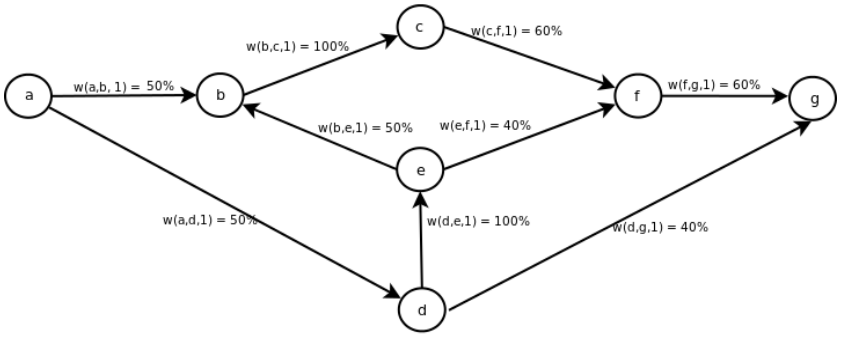

FIGURE 5.5 – L'exemple GLCD (Graphe des Liens Capitalistiques Directs)

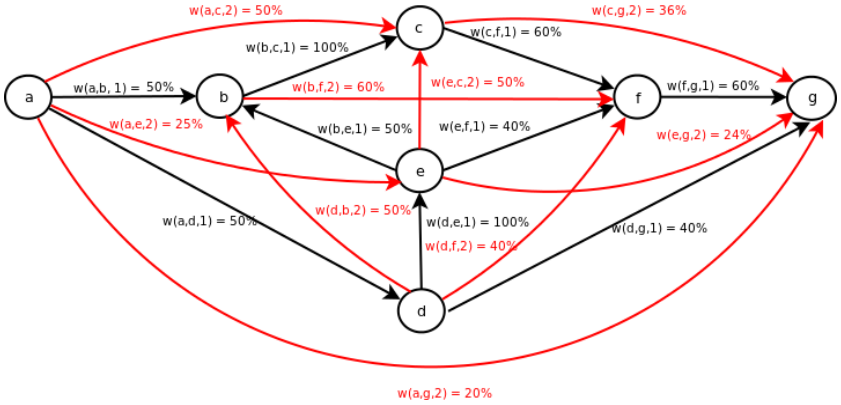

FIGURE 5.6 – La première itération de l'Algorithme **??** sur l'exemple GLCD

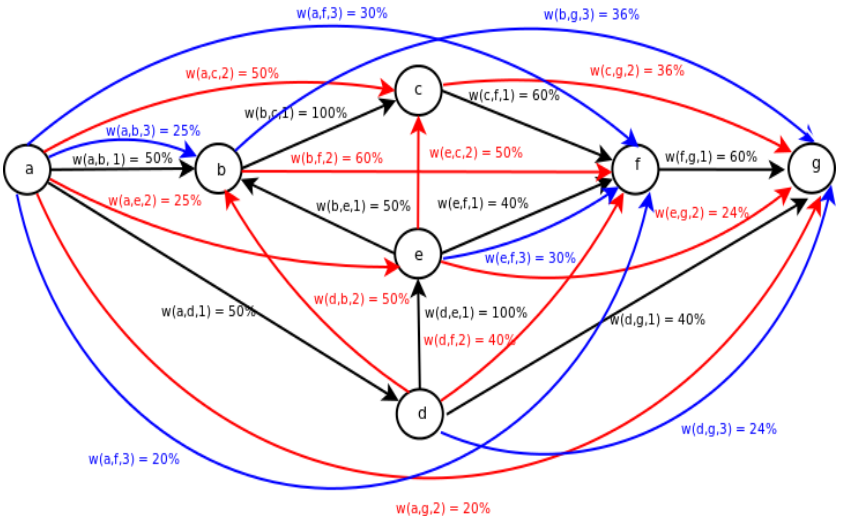

FIGURE 5.7 – La deuxième itération de l'Algorithme ?? sur l'exemple GLCD

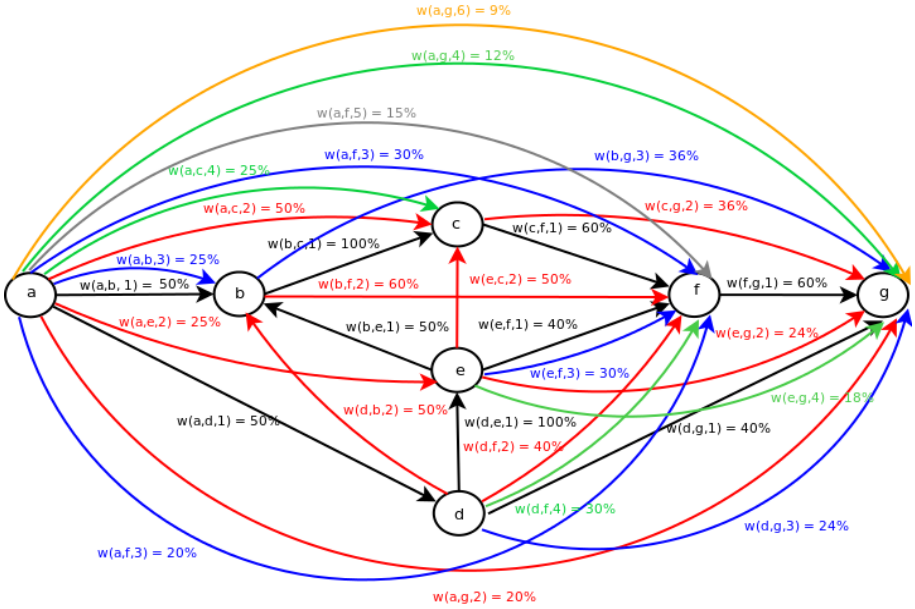

FIGURE 5.8 – La cinquième et dernière itération de l'Algorithme ?? sur l'exemple GLCD

Soit D le graphe orienté associé à une instance du problème de calcul des liens capitalistiques. Pour tout $u, v \in V$ tel que $u \neq v$, $w(u, v, 1)$ correspond au pourcentage que detient directement l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v. Soit $\mathcal{P}_{uv}$ l'ensemble de chemins entre u et v.

Soit i un nombre entier, on note par $\mathcal{P}^i_{uv}$ l'ensemble des chemins de cardinalité i réliant u et v, $\mathcal{P}^i_{uv} = \{P \in \mathcal{P}_{uv} | \, |P| = i\}$. Soit $n_{uv} = \max_{P \in \mathcal{P}_{uv}} |P|$ la cardinalité du chemin élémentaire le plus long entre u et v. On peut écrire $\mathcal{P}_{uv} = \{\mathcal{P}^1_{uv}, \mathcal{P}^2_{uv}, \ldots, \mathcal{P}^{n_{uv}}_{uv}\}$.

Pour tout $i \in \mathbb{N}$ tel que $i < n_{uv}$, $w(u, v, i)$ est défini par l'égalité suivante :

$$w(u, v, i) = \sum_{P \in \mathcal{P}^i_{uv}} (100 \times \prod_{(u_i, v_i) \in P} \frac{w(u_i, v_i)}{100}). \tag{5.4}$$

$w(u, v, i)$ est appelé le pourcentage du lien indirect de niveau i entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v. Le pourcentage total que detient directement et indirectement l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v est notée $w(u, v)$. Ce pourcentage est donné par :

$$w(u, v) = \sum_{i=1}^{n_{uv}} w(u, v, i). \tag{5.5}$$

Si $w(u, v) = 0$, cela veut dire qu'il n'existe aucun lien direct ni indirect entre u et v. Autrement dit, l'entreprise représentée par le sommet u ne detient pas directement ou indirectement un pourcentage du capital social de l'entreprise représentée par le sommet v.

Le pourcentage que detient indirectement l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v est noté $w'(u, v)$. Ce pourcentage est donné par :

$$w'(u, v) = \sum_{i=2}^{n_{uv}} w(u, v, i). \tag{5.6}$$

Dans le cas où $w'(u, v) = 0$, cela signifie que l'entreprise représentée par le sommet u n'est pas un actionnaire indirect de l'entreprise représentée par le sommet v.

De façon plus générale, on note par $\mathcal{P}$ l'ensemble des chemins élémentaires dans le graphe D et par $\mathcal{P}^i$ l'ensemble de chemins élémentaires de cardinalité i, $\mathcal{P}^i =$

$\{P \in \mathcal{P} | \, |P| = i\}$. Soit n la cardinalité du plus long chemin élémentaire dans D, $n = max_{P \in \mathcal{P}} |P|$. Alors, l'ensemble $\{\mathcal{P}^1, \mathcal{P}^2 \ldots \mathcal{P}^n\}$ est une partition de $\mathcal{P}$.

Pour tout chemin $P \in \mathcal{P}$, on note par $Terminal(P)$ la fonction qui retourne le sommet terminal de P, par $Initial(P)$ la fonction qui retourne le sommet initial de P et par $Sommet(P)$ la fonction qui retourne l'ensemble des sommets par lesquels passe le chemin P.

L'idée de l'algorithme que nous proposons est de générer un arc par couple (u, v) de sommets ayant au moins un chemin de cardinalité supérieur 1 et de lui associer un poids. Ce poids correspond au pourcentage que détient l'entreprise représentée par le sommet u dans le capital social de l'entreprise représentée par le sommet v. Pour cela, on commence par générer les couples (u, v) possédant un chemin de cardinalité 2 qui les connecte. Pour cela on associe à tous les couples directs (représentés par un arc) un autre arcs. Puis, on génère les couples ayant au moins un chemin de cardinalité 3 les reliant en utilisant les couples de l'itération précédentes. Lors de l'itération i on génère les couples ayant au moins un chemin de cardinalité i les connectant en utilisant les couples de l'itération $i-1$ et un ensemble d'arcs.

On appelle couple ou lien de niveau i les couples de sommets ayant un chemin de longueur i les connectant. Remarquons qu'un couple (u, v) de niveau i regroupe directement tous les chemins de taille i entre u et v. En effet, il correpond au lien indirect de niveau i entre l'entreprise associée au sommet u et l'entreprise associée au sommet v.

Pour calculer les liens indirects et les liens totaux(directs et indirects) nous avons proposé l'Algorithme **??**. C'est un algorithme itératif qui peut être décrit comme suit :

- La première itération est une itération d'initialisation. Rappelons que A est l'ensemble des couples directs. Nous notons les couples de niveau 1 (directs) par A_1.
  On pose $V_1^{in} = \{v \in V \mid \exists \, u \in V \text{ où } (u, v) \in A\}$. V_1^{in} est l'ensemble des sommets qui ont un arc entrant. Nous constatons que $V_1 = \{v_1 \in V \mid \exists \, u \in V \text{ où } w(u, v_1, 1) \neq 0\}$
- La deuxième itération consiste à calculer les liens indirects de niveau 2. C'est-à-dire les liens qui font intervenir une seule entreprise entre l'actionnaire et la participation. Pour chaque sommet $v \in V$ où il existe un arc (u', v) avec $u' \in V_1^{in}$ et pour chaque sommet u tel que $(u, u') \in A_1$ on considère un couple de niveau 2 (u, v). On note les couples de niveau 2 par l'ensemble A_2. Le pourcentage de ces couples (liens indirects) est donné par

$$w(u, v, 2) = \sum_{u' \in V_1^{in}} \frac{w(u, u', 1) \times w(u', v, 1)}{100}. \tag{5.7}$$

Si $w(u, v, 2)$ est différent de 0, cela veut dire qu'il existe un lien indirect de longueur 2 entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v. Dans ce cas, on crée un nouvel arc (u, v) auquel on associe le poids $w(u, v, 2)$.
La Figure 5.6 donne le résultat de l'application de cette itération à l'exemple de la Figure 5.5 (le graphe GLCD). Les arcs créés dans cette étape sont les arcs de couleur rouge. Par exemple, on peut constater que $f \in V_1^{in}$ car $(c, f) \in A$. On sait aussi que $(f, g) \in A$. On peut en déduire qu'il existe un lien indirect de niveau 2 entre l'entreprise représentée par le sommet c et l'entreprise représentée par le sommet g. Le pourcentage de ce lien $w(c, g, 2)$ est obtenu par :
$w(c, g, 2) = 100 \times \frac{w(c,f,1)}{100} \times \frac{w(f,g,1)}{100} = 100 \times \frac{60}{100} \times \frac{60}{100} = 36$
Cette itération permet d'obtenir, pour tout couple de sommets distincts $u, v \in V$, l'ensemble des couples de niveau 2 ainsi que le pourcentage $w(u, v, 2)$ associé. On pose $V_2^{in} = \{v \in V \mid \exists u \in V \text{ où } w(u, v, 2) \neq 0\}$. V_2^{in} est l'ensemble de sommets terminaux des couples de niveau 2.

– La troisième itération consiste à calculer les liens indirects de niveau 3, c'est-à-dire les liens qui font intervenir deux entreprises entre l'actionnaire et la participation. Un lien indirect de niveau 3 est forcément composé d'un lien indirect de niveau 2 et d'un lien direct. Pour chaque sommet $v \in V$ où il existe un arc (u', v) avec $u' \in V_2^{in}$ et pour chaque sommet u tel que $(u, u') \in A_2$ on considère un couple de niveau 3 (u, v). On note les couples de niveau 3 par l'ensemble A_3. Le pourcentage de ces couples (liens indirects) est donné par

$$w(u, v, 3) = \sum_{v_2 \in V_2} \frac{w(u, v_2, 2) \times w(v_2, v, 1)}{100}. \tag{5.8}$$

De la même façon que lors de l'itération précédente, si $w(u, v, 3)$ est différent de 0, cela veut dire qu'il existe un lien indirect de longueur 3 entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v. Dans ce cas, on crée un nouvel arc (u, v) auquel on associe le poids $w(u, v, 3)$. Le graphe de la Figure 5.7 est obtenu par l'application de cette itération à notre exemple (le graphe GLCD). Les liens créés dans cette étape sont représentés par les arcs de couleur bleu.
Cette itération permet d'avoir, pour tout couple de sommet $u, v \in V$ tel que $u \neq v$, l'ensemble des couples de niveau 3 entre u et v ainsi que le pourcentage $w(u, v, 3)$ associé. On pose $V_3^{in} = \{v \in V \mid \exists u \in V \text{ où } w(u, v, 3) \neq 0\}$. V_3^{in} est l'ensemble de sommets terminaux des couples de niveau 3.

– L'itération i consiste à calculer les liens indirects de longueur i. Un lien indirect de longueur i est un lien qui fait intervenir $i-1$ entreprises entre l'actionnaire et la participation. Un lien de longueur i est forcément composé d'un lien indirect de longueur $i-1$ et d'un lien direct. Pour chaque sommet $v \in V$ où il existe un arc (u', v) avec $u' \in V_{i-1}^{in}$ et pour chaque sommet u tel que $(u, u') \in A_{i-1}$ on considère un couple de niveau i (u, v). On note les couples de niveau i par

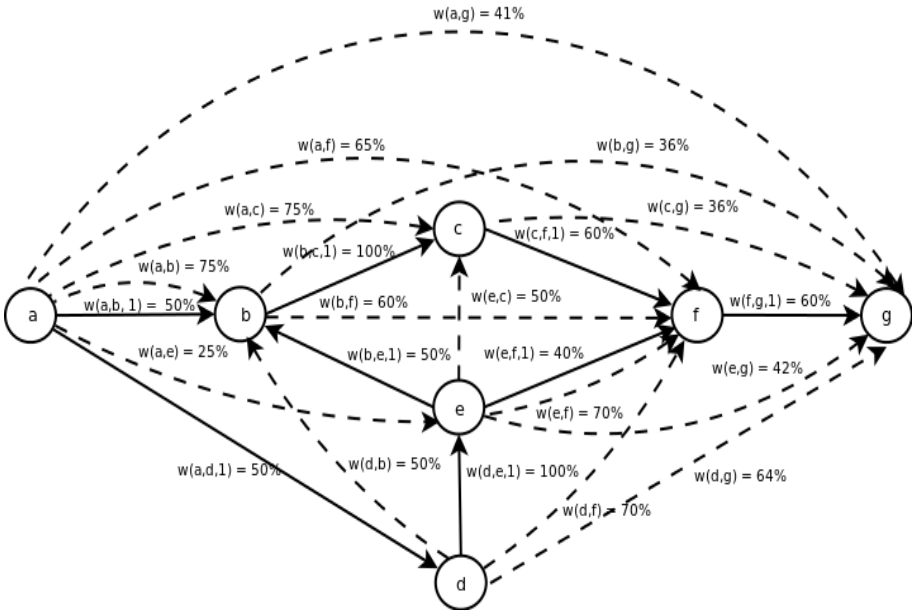

FIGURE 5.9 – Le graphe résultant de l'application de l'Algorithme **??** sur l'exemple GLCD

l'ensemble A_i. Le pourcentage de ces couples (lien indirect) est donné par

$$w(u,v,i) = \sum_{v_{i-1} \in V_{i-1}} \frac{w(u,v_{i-1},i-1) \times w(v_{i-1},v,1)}{100} \ . \tag{5.9}$$

Si $w(u,v,i)$ est différent de 0, cela veut dire qu'il existe un lien indirect de longueur i entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v. Dans ce cas, on crée un nouvel arc (u,v) auquel on associe le poids $w(u,v,i)$.
Cette itération permet d'avoir, pour tout couple de sommet $u,v \in V$ où u et v sont différents, l'ensemble des couples de niveau i entre u et v ainsi que le pourcentage $w(u,v,i)$ associé. On pose $V_i^{in} = \{v \in V \mid \exists u \in V \text{ où } w(u,v,i) \neq 0\}$. V_i^{in} est l'ensemble de sommets terminaux des couples de niveau i.
Si $\sum_{u \neq v \in V} w(u,v,i) > 0$, on passe à l'itération suivante. Sinon, on arrête le traitement itératif. En effet, on ne peut pas avoir un lien de niveau $i+1$ si nous n'avons aucun lien de niveau i. Le pourcentage que détient l'entreprise représentée par le sommet u dans le capital de l'entreprise représentée par le sommet v, est donné par la somme des pourcentages obtenus sur chaque niveau. $w(u,v) = \sum_{i=1}^{n} w(u,v,i)$. D'autre part, le niveau du lien le plus court entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v est donné par $nMin_{uv} = \min\{r | r \in \{1,...,n\} \text{ et } w(u,v,r) > 0\}$ Cette longueur donne le niveau du lien le plus court entre l'entreprise représentée par le sommet u et l'entreprise représentée par le sommet v.

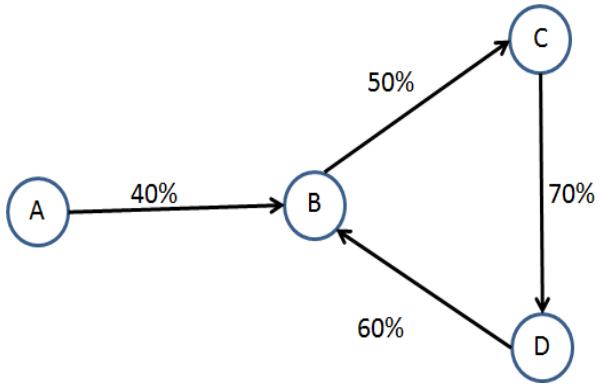

FIGURE 5.10 – Conditions de convergeance : exemple avec un cycle

La Figure 5.8 donne le résultat des itérations 3, 4 et 5 sur l'exemple de la Figure 5.5. Les arcs de couleur verte, les arcs de couleur grise et les arcs de couleur orange réprésentent respectivement les liens de niveau 4, 5 et 6. Nous pouvons constater qu'on ne peut plus créer de liens de niveau 7 dans notre exemple. En effet, il existe un seul chemin de cardinalité 6 (arcs oranges). Le sommet terminal de ce chemin est g, $V_6 = \{g\}$ et n'existe aucun sommet v dans V tel que $(g, v) \in A$. Par conséquent il n'existe pas de chemin de cardinalité 7 dans l'exemple GLCD. Le résultat final du calcul des liens indirects, pour cet exemple, est présenté dans la Figure 6.1. Les liens directs sont représentés par les arcs continuent. Les liens indirects (résultat de notre calcul) sont représentés par les arcs discontinuent.

La condition $u \neq v$ s'explique fonctionnellement par le fait que l'actionnaire doit être différent de la participation. D'autre part, cette condition est nécessaire pour la convergence de l'algorithme, mais elle n'est pas suffisante. Prenons l'exemple simple donné par la Figure 5.10. A est actionnaire de B, B est actionnaire de C, C est actionnaire de D et D est actionnaire de B.

La première itération donne les liens indirects de niveau 2 : A est actionnaire de C, B est actionnaire de D, C est actionnaire de B et D est actionnaire de C. La deuxième itération donne un seul lien indirect de niveau 3 : A est actionnaire de D. En imposant que l'actionnaire soit différent de la participation, on évite ici de créer des liens indirects de niveau 3 entre B et B, entre C et C et entre D et D. La troisième itération donne un lien indirect de niveau 4 entre A et B. Ce lien est obtenu par le chemin (A, B, C, D, B). Notons que ce chemin contient bien un cycle. Si on accepte ce lien contenant le cycle (B, C, D, B), l'algorithme ne convergera pas.

Si on interdit les cycles, la cardinalité du chemin le plus long dans un graphe D est majorée par le nombre de sommets. Dans notre algorithme, on interdit les cycles en ne considérant que les chemins élémentaires. Nous savons donc que le nombre d'itération de notre algorithme est majoré par le nombre de sommets. En pratique,

le graphe D qui représente les données des liens capitalistiques n'est pas connexe et la longueur du chemin le plus long est très petit par rapport au nombre de sommets de V. Lors de l'implémentation on le décompose en plusieurs sous-graphes connexes et nous utilisons le parallélisme d'Oracle afin de traiter les sous graphes en parallèle et d'accélérer ainsi le traitement.

5.3.3 Résultats Expérimentaux

L'Algorithme **??** a été implémenté en PL/SQL avec un traitement ensembliste. Pour cela nous avons créé les tables EB_LIENS_DIRECT, EB_LIENS_INDIRECT_N, EB_LIENS_INDIRECT_N_PLUS_1, EB_LIENS_INDIRECT_ALL_CHEMIN, EB_LIENS_FINALE dont on peut voir la structure dans la Figure 5.11.

Comme leurs noms l'indiquent, la table EB_LIENS_DIRECT est dédiée aux liens directs et la table EB_LIENS_INDIRECT_N aux liens indirects de niveau n. La table EB_LIENS_INDIRECT_N_PLUS_1 est destinée aux liens indirects de niveau $n+1$. Les liens indirects de niveau $n+1$ sont calculés à partir des liens indirects de niveau n et des liens directs. Les liens calculés à chaque itération de notre algorithme sont ensuite insérés dans la table EB_LIENS_INDIRECT_ALL_CHEMIN. La table EB_LIENS_FINALE contiendra les résultats finaux du calcul des liens indirects.

Nous commençons par une étape d'initialisation qui consiste à insérer les liens directs dans les tables EB_LIENS_DIRECT, EB_LIENS_INDIRECT_N et EB_LIE NS_INDIRECT_ALL_CHEMIN. À cette étape n vaut 1.

Ensuite, tant que la table EB_LIENS_INDIRECT_N n'est pas vide, nous effectuons les opérations suivantes :

- Nous calculons les liens indirects de niveau $n+1$ à partir des liens indirects de niveau n (contenu de la table EB_LIENS_INDIRECT_N) et les liens directs (contenu de la table EB_LIENS_DIRECT). Le résultat de ce calcul est stocké dans la table EB_LIENS_INDIRECT_N_PLUS_1.
- Nous insérons les liens que nous venons de calculer (le contenu de la table EB_LIENS_INDIRECT_N_PLUS_1) dans la table EB_LIENS_INDIREC T_ALL_CHEMIN.
- Nous vidons le contenu de la table EB_LIENS_INDIRECT_N et nous y insérons le contenu de la table EB_LIENS_INDIRECT_N_PLUS_1. n devient $n+1$. Cela permet de passer à l'itération suivante.
- Nous vidons la table EB_LIENS_INDIRECT_N_PLUS_1.

Enfin, le contenu de la table EB_LIENS_FINALE, qui donnera le pourcentage total que détient directement et indirectement une entreprise dans une autre, est

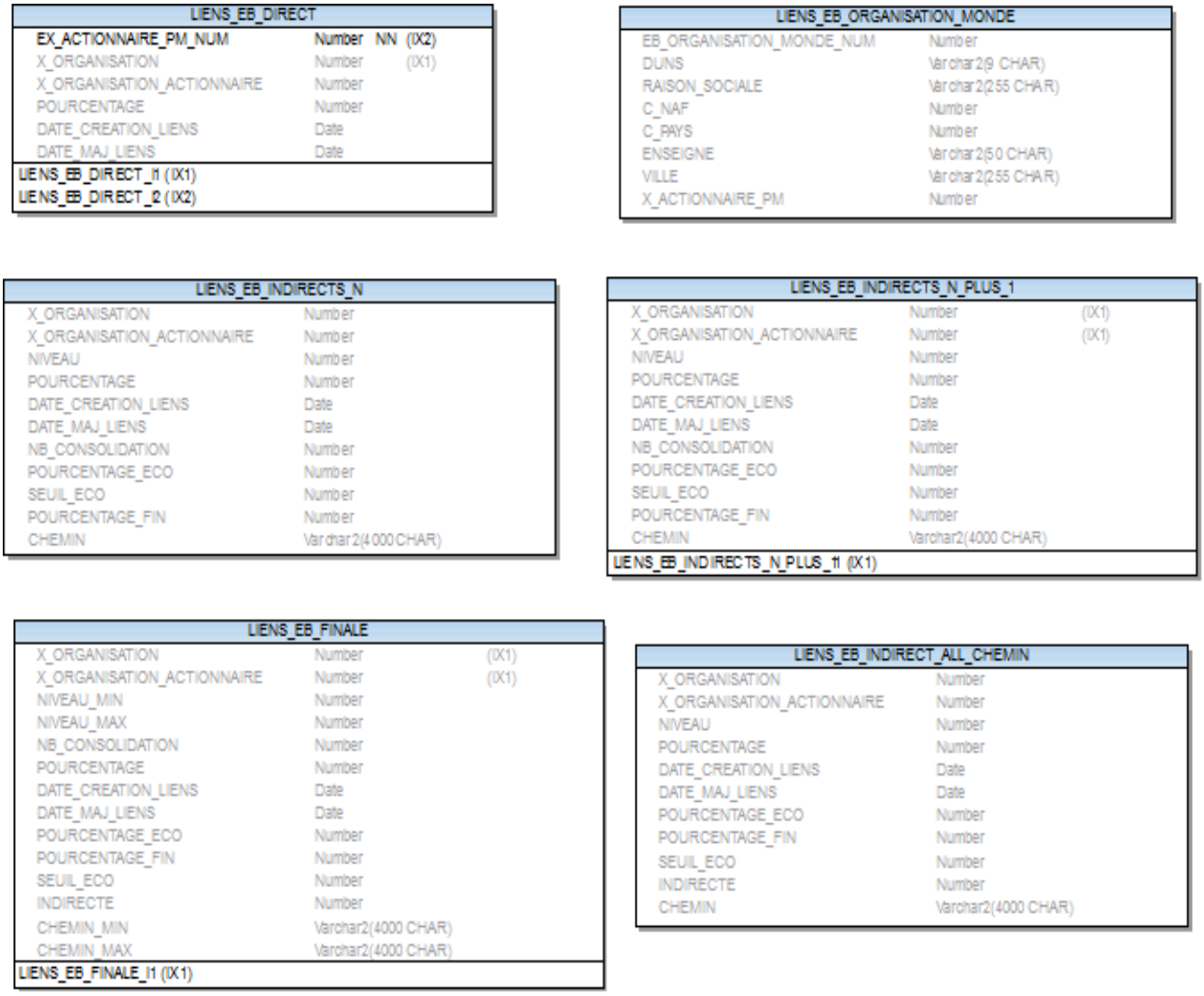

FIGURE 5.11 – Les structures des tables créées pour le calcul des liens indirects

Taille de l'instance				Résultats de calcul		Performances	
Nombre arcs	**Nombre Sommets**	**Niveau max**	**Niveau inversion**	**Nb Liens générés**	**Nb liens totaux**	**Temps de calcul (min)**	**Espace utilisé GO**
65000	44056	32	13	32105462	8096897	43	12,91
80134	56813	32	13	33841540	8129841	44	13,48
91546	61748	32	13	35646892	8197872	46	14,08
108456	75009	33	14	38452240	8273350	47	15,01
120576	86125	33	14	40761280	8398712	47	15,79
130000	90909	33	14	41367524	8504356	48	16,02
145800	98513	33	14	42782803	8654611	49	16,52
154879	103945	34	14	43803911	8778543	50	16,89
175145	115990	34	14	44117375	9190752	52	17,12

TABLE 5.1 – Tableau de résultat

calculé à partir des données de la table EB_LIENS_INDIRECT_ALL_CHEMIN comme expliqué dans l'Algorithme **??**.

Le Tableau 5.1 donne les résultats obtenus par l'application de l'Algorithme **??** sur les données des liens capitalistiques. Chaque ligne correspond à un graphe D qui représente une instance de taille différente du problème de calcul des liens indirects. La dernière ligne correspond au graphe qui représente la totalité des données des liens capitalistiques. Les colonnes du Tableau 5.1 sont les suivantes :

- *Taille de l'instance* : la taille de l'instance est donnée par
  - *Nombre arcs* : le nombre d'arcs de notre graphe D. Ces arcs représentent les liens directs.
  - *Nombre sommets* : le nombre de sommets du graphe D.

  Comme nous pouvons le constater, le graphe qui représente les données des liens capitalistiques n'est pas dense. En effet, le ratio donné par $\frac{\text{Le nombre d'arcs}}{\text{le nombre de sommets}}$ est autour de 1.5.
- *Résultat de calcul* : les résultats du calcul sont les suivants
  - *Niveau max* : correspond au chemin le plus long dans le graphe D. Autrement dit, le lien indirect de niveau le plus élevé (qui fait intervenir le nombre maximum d'entreprises entre l'entreprise actionnaire et l'entreprise participation). Ce nombre est égal au nombre d'itérations nécessaires pour calculer l'ensemble des liens indirects.
  - *Niveau inversion* : Dans le graphe D, il y a plus de chemin de cardinalité 2

qu'il y en a de cardinalité 1, plus de chemin de cardinalité 3 qu'il y en a de cardinalité 2. En effet, la fonction qui donne le nombre de chemins en fonction de la cardinalité commence à croître dès le lancement du traitement. Ensuite, arrivé à un niveau que nous appelons le *niveau inversion*, cette fonction décroît pour tendre vers zéro.
  - *Nb Liens générés* : correspond au nombre total des liens générés. Ce nombre est égal à la somme du nombre des liens générés dans chaque itération.
  - *Nb Liens totaux* : ce nombre correspond au nombre total des liens directs et indirects. Pour l'exemple GLCD, ce nombre correspond au nombre d'arcs dans la Figure 6.1. Ce sont ces liens qui sont stockés dans la table EB_LIENS_FINALE.
- *Performances* : la performance de l'algorithme est définie par deux métriques qui sont
  - *Temps calcul (min)* : le temps de calcul en minute. Nous pouvons constater que l'ensemble des liens indirects sont calculés en 52 minutes
  - *Espace utilisé (GO)* : l'espace de stockage utilisé durant l'algorithme pour stocker les objets utilisés pour le calcul (les tables présentées en début de cette section)

Comme nous pouvons le remarquer dans le tableau des résultats, le temps de calcul ainsi que l'espace utilisé croit de façon logarithmique en fonction de la taille de l'instance. La taille de l'instance représentée par la dernière ligne est pratiquement trois fois plus que la taille de celle représentée par la première ligne. Cependant, ni le temps de calcul ni l'espace utilisé n'ont été multipliés par trois. D'autre part, le niveau max et le niveau inversion qui jouent un rôle très important dans la convergence de l'Algorithme **??** reste sensiblement le même entre la première et la dernière instance.

Ces résultats ont été validés par les équipes d'Altares. Du point de vue technique, le temps de calcul ainsi que l'espace de stockage consommés sont convenables. De point de vue fonctionnel, les équipes métiers ont validés que les résultats obtenus (les liens indirects et les liens totaux) sont corrects.

5.4 Têtes de groupe

L'objectif de cette question est de déterminer les têtes des groupes ainsi que les entreprises qui composent chaque groupe. Rappelons que la définition de tête de groupe dépend du seuil qui définit le contrôle. Comme expliqué ce seuil peut prendre plusieurs valeurs en fonction de l'usage. Les valeurs les plus courantes sont 25%, 33%, 40% et 50%. Les équipes métiers d'Altares ont fait le choix de calculer les têtes de groupe selon deux seuils 25% et 50%.

5.4.1 Méthode de Calcul des têtes de groupe

Dans cette section, nous présentons la méthode simple qui a été mise en place pour calculer les têtes de groupe 25% et les têtes de groupe 50%. Nous verrons que le seuil (25% ou 50%) est tout simplement considéré comme un paramètre d'entrée. Sa valeur n'augmente pas la complexité ni le temps de calcul observé. En conséquence, nous décrirons la méthode de résolution uniquement pour le calcul des têtes des groupes 25%. Les têtes des groupes 50% sont obtenues de la même façon en changeant le seuil 25% par 50%.

Nous pouvons remarquer que selon la définition, une entreprise qui n'a pas d'actionnaire et qui a une seule participation de 20%, n'appartient à aucun groupe et n'est pas elle-même une tête de groupe.

Notons qu'une entreprise peut avoir deux actionnaires principaux qui détiennent le même pourcentage d'actionnariat et qui est supérieur au seuil de contrôle. Ce cas s'appelle *Join Venture*. Par exemple, deux groupes qui ont un intérêt commun dans une entreprise et qui décident de la contrôler conjointement, c'est-à-dire en ayant le même pouvoir de contrôle. Ces groupes peuvent être les seuls actionnaires de cette entreprise (en détenant chacun 50%) et peuvent s'associer avec un ou plusieurs actionnaires qui détiendraient des parts moins importantes. C'est le cas par exemple de deux groupes qui détiennent chacun 40% d'une entreprise et les 20% restant sont détenus par un ou plusieurs actionnaires. Le contrôle d'une entreprise Join Venture, n'appartient pas exclusivement à l'un des deux groupes qui la contrôlent. De ce fait, cette entreprise ne fait partie d'aucun de ces deux groupes. Si elle est actionnaire principal d'au moins une entreprise, elle est considérée comme tête de groupe.

Comme vu dans la section précédente, l'ensemble des liens directs et indirects sont stockés dans la table EB_LIENS_FINALE. Le champ X_ORGANISATION de cette table donne l'identifiant de l'entreprise participation, le champ X_ORGANISAT ION_ACTIONNAIRE donne l'identifiant actionnaire et le champ pourcentage correspond au pourcentage que détient directement et indirectement l'entreprise X_ORG ANISATION_ACTIONNAIRE dans le capital social de l'entreprise X_ORGANISATION.

La méthode de calcul des têtes de groupe à 25% est la suivante :

- Pour chaque entreprise qui a au moins un actionnaire, nous commençons par identifier le pourcentage de son actionnaire principal. Cette étape est importante. En effet une entreprise qui posséde un actionnaire détenant un pourcentage supérieur au seuil de contrôle n'est pas autonome. Elle ne peut pas être une tête de groupe. Les pourcentages de l'actionnaire principal de chaque entreprise qui a un actionnaire, sont insérés dans la table EB_LIENS_PR_ACT _PRINCIPAL créés à cet effet.

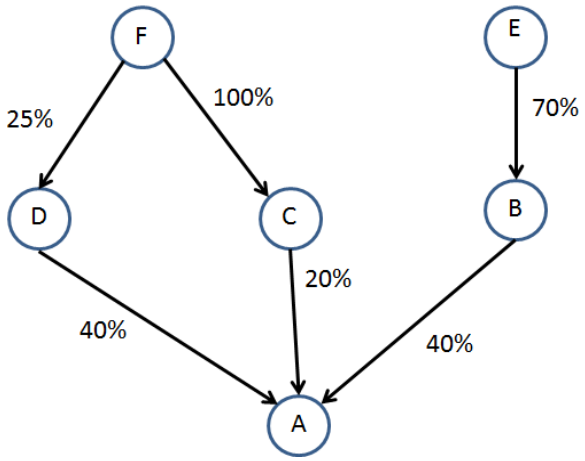

FIGURE 5.12 – Exemple d'appartenance à un groupe

- En utilisant la table EB_LIENS_PR_ACT_PRINCIPAL qui donne pour chaque entreprise qui a un actionnaire, l'actionnaire principal et le pourcentage d'actionnariat, nous mettons dans la table des têtes des groupes toute entreprise qui n'a pas d'actionnaire principal qui détient plus de 25% et qui est actionnaire principal de plus de 25% d'au moins une entreprise. Cette étape permet d'identifier les têtes de groupe qui ont un actionnaire.
- Nous ajoutons dans la table des têtes de groupe toute entreprise qui n'a pas d'actionnaire et qui est actionnaire principal de plus de 25% d'au moins une entreprise.
- Nous identifions les têtes de groupe Join Venture. Pour cela, en partant de la table EB_LIENS_PR_ACT_PRINCIPAL et de la table EB_LIENS_FINALE, nous cherchons les entreprises pour lesquelles
  – le pourcentage de l'actionnaire principal est supérieur à 25%
  – ce pourcentage est détenu par deux actionnaires
  Ces entreprises sont considérées comme têtes de groupe. Elles sont donc insérées dans la table des têtes de groupe.

5.4.2 Identification des groupes

Suite au calcul expliqué dans la section précédente, la table des têtes des groupes contient toutes les têtes de groupe 25%. Nous cherchons à présent à identifier pour chacune de ces entreprises têtes de groupe, l'ensemble des entreprises qui composent son groupe. Autrement dit, l'ensemble des entreprises qu'elle contrôle.

Deux groupes différents peuvent détenir simultanément des parts d'une entreprise donnée. Cependant, cette entreprise fait partie du groupe qui en détient le

pourcentage le plus élevé. Il est donc nécessaire d'identifier pour chaque entreprise l'actionnaire tête de groupe qui détient le plus de parts parmi tous ces actionnaires têtes de groupe.

Prenons l'exemple donné par la Figure 5.12. Nous constatons que :
- B est un actionnaire direct de A à 40%.
- E est un actionnaire direct de B à 70%
- C est un actionnaire direct de A à 20%
- D est un actionnaire direct de A à 40%
- F est un actionnaire direct de D à 25%
- F est un actionnaire direct de C à 100%

Le calcul des liens indirects nous permet de conclure que :
- E est actionnaire indirect A à $\frac{70\times40}{100} = 28\%$
- F est actionnaire indirect A à $\frac{100\times20}{100} + \frac{25\times40}{100} = 30\%$

Le calcul de tête de groupe permettra d'identifier que E et F sont les têtes de groupe. C et D appartiennent au groupe de F et B appartient au groupe de E. Les deux groupes détiennent des parts de la société A. Le groupe de F détient 30% de la société A et le groupe de E détient 28% de la société A. Par conséquence, A appartient au groupe de F même si son actionnaire principal B appartient au groupe de E.

Conclusion

Les problèmes de calcul des liens indirects et d'identification des têtes de groupe ont été résolues. Le temps de calcul ainsi que l'espace utilisé sont très satisfaisants. Lors du lancement de ce projet, les équipes d'Altares envisageaient de lancer le calcul des liens indirects et l'identification des têtes de groupe une fois par mois. En effet, cette fréquence était considérée comme acceptable d'un point du vue métier et d'autre part, le plan production d'Altares est très chargé vu la quantité des données à traiter quotidiennement.

Aujourd'hui, vu les performances obtenues, les équipes d'Altares prévoit de lancer les liens indirects et les têtes de groupe chaque semaine.

Par ailleurs, les résultats obtenus ont été validés par les équipes métiers d'Altares. Ces résulats sont aujourd'hui utilisés dans l'identification des PME. Une première version de Link-File, qui utilisent les liens indirects et les têtes de groupe, a été mise en place par Altares.

Chapitre 6

Réseaux D'influence

Sommaire

6.1 Introduction

Dans ce chapitre, nous nous intéressons à la caractérisation des réseaux d'influence des personnes dirigeantes ou actionnaires. En effet, le travail présenté dans le cha-

pitre 4 a permis de constituer l'expérience des dirigeants. Désormais, pour chacun d'eux, on sait identifier l'ensemble des entreprises qu'il dirige, celles qu'il a dirigées ainsi que les personnes avec qui il dirige actuellement ou il a dirigé dans le passé. Ces expériences que ces dirigeants partagent, ont certainement créé entre eux des liens plus au moins forts. Ainsi, on peut imaginer qu'un investisseur soit mieux informé que les autres investisseurs, des opportunités d'investissement dans une entreprise gérée par une personne avec laquelle cet investisseur dirige une autre entreprise.

D'autre part, le travail présenté dans le chapitre précédant, nous a permis de calculer l'ensemble des liens capitalistiques indirects. Désormais, pour un actionnaire donné, on sait identifier l'ensemble des entreprises dans lesquelles il est actionnaire direct ou indirect. Ainsi, on peut imaginer qu'un actionnaire peut avoir de l'influence, non seulement dans les entreprises dans lesquelles il est actionnaire direct, mais aussi dans celles où il est actionnaire indirect et même éventuellement dans les entreprises, dirigées par une personne qui dirige une entreprise dans laquelle notre actionnaire est actionnaire direct ou indirect.

Toutes ces personnes dirigeantes ou actionnaires, forment ou peuvent former, des communautés plus au moins fortes. Dans ce chapitre, nous cherchons à identifier les réseaux d'influence que forment les dirigeants et actionnaires des entreprises françaises. Bien évidemment, en plus de ces réseaux qu'on peut appeler «professionnels », un dirigeant ou un actionnaire peut mobiliser d'autres types de réseau dont par exemple, les réseaux personnels, les réseaux politiques, les réseaux associatifs. Les données sur lesquelles nous travaillons dans ce chapitre, sont uniquement les données des dirigeants et les données des liens capitalistiques. Cependant, les modèles et les résultats théoriques obtenus, peuvent être appliqués pour caractériser d'autres types de réseau dont ceux cités plus haut.

Nous proposons dans ce chapitre de modéliser les liens entre les dirigeants et les actionnaires par un graphe et de ramener le problème d'identification des réseaux d'influence à un problème de partitionnement de graphe appelé Problème de Coupe Min-Max (PCMM).

Ce problème permet de partitionner l'ensemble des personnes dirigeantes ou actionnaires en k groupes d'influence. Cela représente un partitionnement du paysage entrepreneurial français. En changeant la valeur de k il est possible de déterminer différentes répartitions.

La première section de ce chapitre est consacrée à la présentation du problème PCMM, sa modélisation en termes de graphe et sa formulation à l'aide d'un programme linéaire en nombres entiers. Pour cela, nous commençons par présenter le graphe d'influence et le problème PCMM. Ensuite, nous présentons quelques travaux sur le partitionnement de graphe. Enfin, nous donnons une formulation en

programme linéaire en nombres entiers pour le problème PCMM.

La deuxième section est destinée à l'étude du polytope $P(G, k, q)$ des solutions du programme mathématique associé au PCMM. Nous commençons par caractériser la dimension du polytope et nous étudions l'aspect faciale des contraintes de la formulation. Enfin, nous introduisons quelques inégalités valides pour notre formulation.

La troisième et dernière section est dédiée aux résultats expérimentaux. Dans cette section, nous commençons par donner le contexte informatique. Puis, nous présentons notre algorithme de coupes et branchements ainsi que les algorithmes de séparations utilisés. Ensuite, nous présentons une heuristique que nous avons introduite pour résoudre le PCMM afin de donner une borne supérieure à notre algorithme de coupe et branchement. Enfin, nous présentons les instances testées et donnons quelques tableaux de résultats.

6.2 Les problèmes de partitionnement

Dans cette section, nous commençons par donner quelques définitions qui seront utile tout au long de ce chapitre.

6.2.1 Définitions

Définition 5 *Soit un graphe $G = (V, E)$ et $\mathcal{V} = \{V_1, ..., V_k\}$ k sous-ensembles de V. $\mathcal{V}$ est dite une partition de G si :*
- *aucun élément de $\mathcal{V}$ n'est vide, c'est-à-dire $V_i \neq \emptyset$ pour tout $i \in \{1, ..., k\}$,*
- *les éléments de $\mathcal{V}$ sont deux à deux disjoints, c'est-à-dire $V_i \cap V_j = \emptyset$, pour tout $i, j \in \{1, ..., k\}$ tels que $i \neq j$,*
- *l'union de tous les éléments de $\mathcal{V}$ est égale à V, $\bigcup_{V_i \in \mathcal{V}} V_i = V$.*

Le nombre k est appelé cardinalité de la partition $\mathcal{V}$, et les $V_i, i = 1, ..., k$ sont appelés les parties de $\mathcal{V}$.

Dans la littérature, on distingue deux types de partitionnement de graphe : *le partitionnement contraint* et le *partitionnement non contraint*, deux types de problèmes montrés NP-difficile [13].

6.2.2 Partitionnement de graphe

Le partionnement de graphe est un problème transverse à plusieurs domaines d'ingénierie et de recherche qui a été largement étudié depuis le début des années 70. Plu-

sieurs problèmes en relation avec la conception de circuits intégrés électriques [93], les télécommunications [74], le contrôle de trafic aérien [12] ainsi que la segmentation d'image [66] se réduisent à des problèmes de partitionnement de graphe. Le problème est aussi très utile dans le domaine du data mining et y est plus connu sous le nom de *regroupement de données* (*data clustering*). Pour plus de détails sur le problème de partitionnement de graphe et ses applications, le lecteur peut se référer à [14, 13, 81, 45].

6.2.2.1 Approches de résolution

Le problème de partitionnement de graphe a été largement étudié dans la littérature et différentes approches de résolution ont été proposées. Certaines de ces approches adoptent explicitement dans leurs formulations les concepts de la théorie des graphes. D'autres, utilisent les graphes comme support pour leurs structures de données [67]. En fonction du problème de partitionnement à résoudre, de nombreux algorithmes utilisant des méthodes inertielles, la classification hiérarchique, la méthode spectrale, les heuristiques et les méthaheuristiques ont été proposées. En particulier, Kernighan et Lin [48] présente une heuristique linéaire et efficace pour le problème de partitionnement de graphe en k parties avec coût minimum. Dans [14, 13, 37], les auteurs énumèrent plusieurs autres travaux ayant optés pour différentes approches de résolution dépendant de la structure du problème traité.

6.2.2.2 Exemples d'applications

Comme nous l'avons mentionné auparavant, le problème de partitionnement de graphe a diverses applications. Dans cette section, nous présentons quelques exemples de problèmes dont la résolution se ramène à celle du problème de partitionnement de graphe. Pour plus de détails, le lecteur peut se référer à [14, 13]

Classification de documents

Le problème de la classification de documents consiste à classer une collection de documents en différentes catégories partageant des sujets communs. Suivant l'approche classique, la classification des documents est basée sur la distribution des mots (appelés aussi tokens) dans les documents. Une distance basée sur la similarité entre les tokens utilisés est calculée entre chaque paire de documents. Puis les documents sont classés au sens de cette similarité. Ce problème peut être vu comme un problème de partitionnement dans un graphe biparti. Il s'agit de classer les documents et les tokens en même temps, sans même calculer des distances entre les

documents. Le principe de partitionnement se base sur le fait que deux documents qui partagent les mêmes tokens sont classés ensemble.

Le problème de partitionnement min-max

Une application importante du partitionnement de graphe est le regroupement de données ou data clustering. À la différence des problèmes de partitionnement classiques, l'un des problèmes étudiés dans le data clustering est *le problème de partitionnement min-max* (min-max clustering problem). Le problème a été proposé par Ding et al. [28, 30]. Il consiste à chercher une partition du graphe de façon à minimiser les similarités externes (inter-cluster similarities), tout en maximisant les similarités internes (intra-cluster similarities).

6.3 le problème de coupe Min-Max

Nous présentons une modélisation en termes de graphe pour le problème de réseaux d'influence et illustrons cette modélisation à l'aide d'un exemple. Puis, nous présentons le problème de partitionnement de graphe. Enfin, nous donnons une modélisation à l'aide d'un Programme Linéaire en Nombres Entiers (*PLNE*) pour le problème étudié dans ce chapitre.

6.3.1 Formulation en termes de graphes

Considérons le graphe $G = (V, E)$ où les sommets V représentent les personnes (dirigeants ou actionnaires) et il existe une arête $uv \in E$ si et seulement si la personne représentée par le sommet u et la personne représentée par le sommet v dirigent la même entreprise ou s'il existe un lien capitalistique (direct ou indirect) entre l'entreprise dirigée par la personne représentée par le sommet u et l'entreprise dirigée par la personne représentée par le sommet v. On note $w(uv) \to \mathbb{R}^+$ la fonction de pondération qui associe à chaque arête $uv \in E$

- Si la personne représentée par le sommet u et la personne représentée par le sommet v dirigent la même entreprise, alors $w(uv) = 100$.
- Sinon $w(uv)$ est égale au pourcentage de lien (direct ou indirect) entre l'entreprise dirigée par la personne représentée par le sommet u et l'entreprise dirigée par la personne représentée par le sommet v.

Afin d'illustrer cette modélisation, considérons le graphe donné par la Figure 6.1 où les sommets de couleur noire représentent des entreprises. Les arcs de couleur noire représentent les liens capitalistiques directs entre ces entreprises. Les sommets

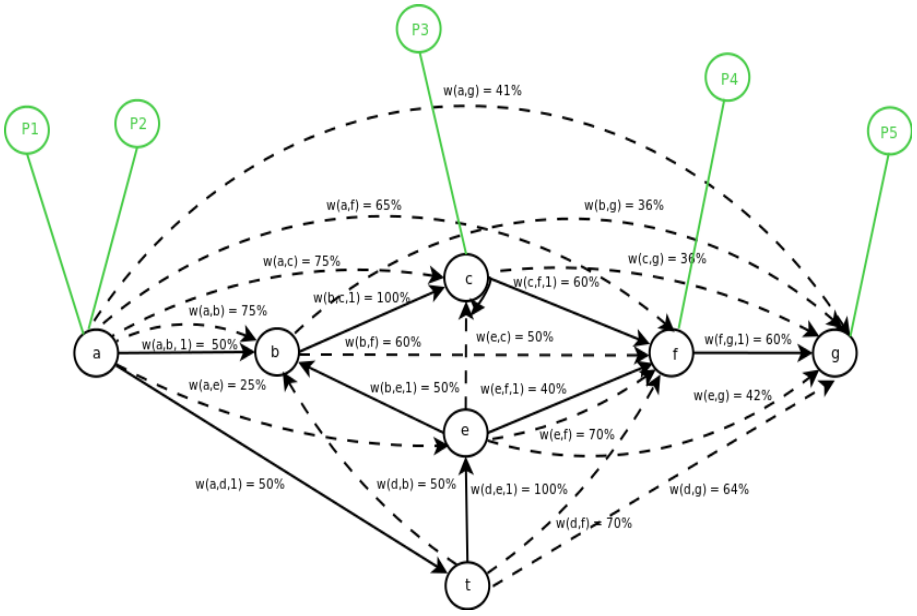

FIGURE 6.1 – Graphe contenant les informations des dirigeants et les informations des liens capitalistiques

de couleur verte, représentent des personnes physiques. Les arcs de couleur verte qui relient un sommet de couleur verte et un sommet de couleur noire, indiquent que la personne représentée par le sommet vert est dirigeant de l'entreprise représentée par le sommet de couleur noire. Les arcs discontinuent de couleur noire, représentent les liens indirects entre les entreprises.

À partir du graphe donné par la Figure 6.1, nous construisons le graphe donné par la Figure 6.2 que nous appelons le graphe d'influence. Le graphe d'influence est un graphe non orienté qui modélise les relations entre les personnes comme expliqué plus haut. Le poids 100 de l'arête qui relie P_1 et P_2 indique que les deux personnes représentées par les sommets P_1 et P_2 dirige la même entreprise. Comme on peut le constater dans la Figure 6.1, les personnes représentées par P_1 et P_2, dirigent toutes les deux l'entreprise représentée par le sommet a. L'arête qui relie le sommet P_1 et P_5 dans le graphe donné par la Figure 6.2, résulte du fait que l'entreprise, représentée par le sommet a et dirigée par la personne représentée par le sommet P_1, est actionnaire de l'entreprise représentée par le sommet g et qui est dirigée par la personne représentée par le sommet P_5. De plus, le poids de l'arête qui relie P_1 et P_5 correspond au pourcentage que détient (directement et indirectement) l'entreprise représentée par le sommet a dans le capital sociale de l'entreprise représentée par le sommet g.

Soit $E' \subseteq E$ un sous-ensemble d'arêtes, on note $w(E')$ la somme des poids associés aux arêtes appartenant à E', c'est-à-dire $w(E') = \sum_{e \in E'} w(e)$. Soit $V_i, V_j \subseteq V$ deux

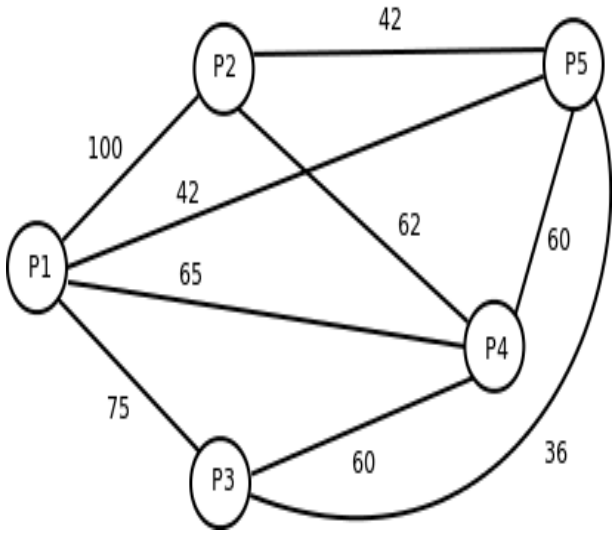

FIGURE 6.2 – Le graphe d'influence

sous-ensembles de sommets, telque $V_i \cap V_j = \emptyset$. On note par $[V_i, V_j]$ l'ensemble des arêtes entre V_i et V_j, c'est-à-dire $[V_i, V_j] = \{uv \in E | u \in V_i, v \in V_j\}$.

L'objectif est de trouver une partition $\mathcal{V} = \{V_1, ..., V_k\}$ des sommets V telle que la plus grande coupe entre deux ensembles V_i et V_j de $\mathcal{V}$, où $V_i \neq V_j$, soit minimum, c'est-à-dire $\min\{z | w([V_i, V_j]) \leq z : V_i, V_j \in \mathcal{V}\}$). De plus, la différence entre la cardinalité de chaque deux ensembles de cette partition ne doit pas dépasser q. En d'autres termes, nous cherchons à minimiser la plus grande coupe entre deux ensembles de la partition et respectant un certain équilibrage donné par une constante q. Dans la suite de ce document, ce problème sera appelé Problème de Coupe-Min-Max(PCMM)

Ce problème est NP-difficile car il est une généralisation du problème de bisection [7]. Le problème de bisection consiste à partitionner l'ensemble des sommets V d'un graphe en 2 ensembles V_1 et V_2 tel que $|V_1| - |V_2| \leq 1$, $|V_2| - |V_1| \leq 1$ et la coupe est de poids minimum, c'est-à-dire minimiser $w([V_1, V_2])$. Il est claire que si $q = 1$ et $k = 2$ notre problème se ramène au problème de bisection.

6.3.2 Modélisation par un programme linéaire en nombres entiers

Associons à chaque sommet $u \in V$ et à chaque $t \in \{1, ..., k\}$, une variable binaire $x_{u,t}$ valant 1 si le sommet u appartient à V_t et 0 sinon.
Associons aussi à chaque arête $uv \in E$ et à chaque $r, t \in \{1, ..., k\}$ tel que $r \neq t$, une

variable binaire $y_{uv,r,t}$ valant 1 si l'arête uv appartient à la coupe $[V_r, V_t]$ et 0 sinon. Enfin, nous définissons une variable entière z qui correspond à la plus grande coupe entre n'importe quels deux ensembles de la partition.

Le problème PCMM est équivalent au programme linéaire en nombres entiers (P) donné par :

$$\min z$$

$$\sum_{uv \in A} y_{uv,r,t} w(uv) \leq z, \quad \text{pour tout } t, r \in \{1, ..., k\} \text{ tel que } t \neq r, \tag{6.1}$$

$$\sum_{t=1}^{k} x_{u,t} = 1, \quad \text{pour tout } u \in V, \tag{6.2}$$

$$\sum_{v \in V} x_{v,r} - \sum_{v \in V} x_{v,t} \leq q, \quad \text{pour tout } r \neq t \in \{1, ..., k\} \tag{6.3}$$

$$x_{u,r} + x_{v,t} \leq 1 + y_{uv,r,t}, \quad \text{pour tout } t \neq r \in \{1, ..., k\} \text{ et pour tout } uv \in E, \tag{6.4}$$

$$x_{u,t} \leq 1, \quad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.5}$$

$$x_{u,t} \geq 0, \quad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.6}$$

$$y_{uv,r,t} \leq 1, \quad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\}, \tag{6.7}$$

$$y_{uv,r,t} \geq 0, \quad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\}, \tag{6.8}$$

$$x_{u,t} \in \{0, 1\}, \quad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.9}$$

$$y_{uv,r,t} \in \{0, 1\}, \quad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\}, \tag{6.10}$$

$$z \in \mathbb{R}. \tag{6.11}$$

Nous appelons dans la suite *inégalités de partition* les contraintes (6.2). Les contraintes (6.3) sont appelées *inégalités d'équilibre*. Les inégalités (6.4) sont nommées *inégalités d'arête*.

Les inégalités (6.1) bornent la plus grande coupe entre deux sous-ensembles par la variable z. Les inégalités de partition (6.2) impliquent qu'un sommet apparait uniquement dans un ensemble et vérifie donc que nous obtenons une partition. Les inégalités d'équilibre (6.3) assurent l'équilibrage entre les ensembles de la partition. Les inégalités d'arête (6.4) permettent d'assurer que la variable associée à une arête est égale à 1 si ces deux extrémités sont dans deux ensembles différents.

Remarquons qu'il est possible qu'une variable associée à une arête soit égale à 1 alors que ses deux extrémités sont dans le même ensemble. Comme nous minimisons la coupe maximum entre n'importe quels deux ensembles il n'y a pas d'intérêt à activer cette variable. Dans ce cas, si elle est égale à 1 et que nous sommes à l'optimum

alors il est possible de la mettre à 0 tel que la solution reste valide et que la valeur de la fonction objectif reste inchangée. On en déduit que le programme linéaire en nombres entiers ci-dessus permet de résoudre notre problème.

Dans la suite de ce chapitre, notons le nombre de sommets $|V|$ par n, le nombre d'arêtes $|E|$ par m. Notons aussi par A la matrice qui représente le système d'equations (6.2) du programme (P). Ces égalités peuvent être écrites sous la forme $Ax = b$.

Soit $P(G,k,q)$ l'enveloppe convexe des solutions du programme (P), c'est-à-dire,

$$P(G,k,q) = conv(\{x \in \{0,1\}^{n\times k}, y \in \{0,1\}^{m\times \frac{|k|\times(|k|-1)}{2}}, z \in \mathbb{R}^{+} | x,y,z \text{ satisfont } (6.1),(6.4)\}).$$

6.4 Étude polyédrale de $P(G,k,q)$

Dans cette section, on étudie l'enveloppe convexe des solutions du problème PCMM. Dans un premier temps, nous introduisons des propositions que nous allons utilisés tout au long de ce chapitre. Puis, nous caractérisons la dimension du polytope $P(G,k,q)$. Ensuite, nous donnons des conditions nécessaires et suffisantes pour que les contraintes de la formulation initiale définissent des facettes. Enfin, nous proposons de nouvelles inégalitées valides.

Tout au long de cette étude, nous supposons, que n n'est pas un multiplicateur de k.

Définition 6 *Soient $u \in V_r$ et $v \in V_t$. Une permutation entre u et v consiste à inverser leurs affectations dans la partition c'est-à-dire $V_r = V_r \setminus \{u\} \cup \{v\}$ et $V_t = V_t \setminus \{v\} \cup \{u\}$.*

Vecteur d'incidence

Soient $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ une partition de V. $\mathcal{V}$ est une solution de PCMM si et seulement si pour tout $i,j \in \{1,2,\ldots,k\}$, $|V_i| - |V_j| \leq q$. On définit $(x^{\mathcal{V}}, y^{\mathcal{V}}, z^{\mathcal{V}})$ le vecteur d'incidence associé à $\mathcal{V}$ tel que :

$$x^{\mathcal{V}}_{u,t} = \begin{cases} 1 & \text{si } u \in V_t, \\ 0 & \text{sinon,} \end{cases} \qquad \text{pour tout } u \in V \text{ et pour tout } V_t \in \mathcal{V}.$$

$$y^{\mathcal{V}}_{uv,r,t} = \begin{cases} 1 & \text{si } uv \in [V_r, V_t], \\ 0 & \text{sinon,} \end{cases} \qquad \begin{array}{l} \text{pour tout } uv \in E \text{ et pour tout} \\ r,t \in \{1,2,\ldots,k\} \text{ tel que } r \neq t. \end{array}$$

$z^{\mathcal{V}} = \sum_{uv \in E} \sum_{r=2}^{k} \sum_{t=1}^{r-1} w(uv)$

Remarquons que $z^{\mathcal{V}}$ correspond à une borne supérieure triviale.

Solution de référence

Numérotons les sommets de V de la manière suivante $V = \{v_1, v_2, \ldots, v_n\}$. On pose $p = (n \mod k)$ et $d = \lfloor \frac{n}{k} \rfloor$. Soit $\mathcal{V}^0 = \{V_1, V_2, \ldots, V_k\}$ une partition de V définie de la manière suivante. Pour tout $i \in \{1, 2, \ldots, k\}$,

- si $i \leq p$ alors $V_i = \{v_f, \ldots, v_g\}$ avec $f = (i-1)(d+1)+1$ et $g = i(d+1)$,
- sinon $V_i = \{v_f, \ldots, v_g\}$ avec $f = p(d+1) + ((i-1)-p)d + 1$ et $g = p(d+1) + (i-p)d$.

Remarquons que pour tout $i \in \{1, \ldots, p\}$ les ensembles V_i contiennent $d+1$ sommets ($|V_i| = d+1$ pour tout $i \in \{1, \ldots, p\}$) et pour tout $i \in \{p, \ldots, k\}$ les ensembles V_i contiennent d sommets ($|V_i| = d$ pour tout $i \in \{p, \ldots, k\}$). Il est clair donc que $\mathcal{V}^0$ est une solution de PCMM. Cette solution, appelée solution de référence, sera utilisée par la suite.

Proposition 1 *Soient $u, v \in V$ tel que $u \neq v$ et $r \in \{1, 2, \ldots, k\}$. Il existe $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$, une partition de V et solution de PCMM tel que u et v appartiennent au même ensemble V_r*

Preuve. Soit $\mathcal{V}^0 = \{V_1^0, V_2^0, \ldots, V_k^0\}$ la solution de référence. Nous allons montrer qu'il existe une solution $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$ vérifiant la proposition. Posons $\mathcal{V}^1 = \mathcal{V}^0$. Supposons que u et v n'appartiennent pas au même ensemble. En effet, si u et v sont dans le même ensemble V_i^0 alors en inversant les ensembles V_i et V_r ($V_i^1 = V_r^0$, $V_r^1 = V_i^0$) nous obtenons une solution vérifiant la proposition. Sans perte de généralité considérons que $u \in V_r$ et $v \in V_t$. Si $u \in V_i^0$ (resp. $v \in V_i^0$) alors on pose $V_i^1 = V_r^0$, $V_r^1 = V_i^0$ (resp. $V_i^1 = V_t^0$, $V_t^1 = V_i^0$). Considérons un sommet $w \in V_r \setminus \{u\}$. Posons desormais $V_r^1 = V_r^1 \setminus \{w\} \cup \{v\}$ et $V_t^1 = V_t^1 \setminus \{v\} \cup \{w\}$.

Il est clair que $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$ est une solution de PCMM tel que u et v sont dans le même ensemble □

Proposition 2 *Soient $r, s \in \{1, 2, \ldots, k\}$ tel que $r \neq s$. Il existe $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$, une partition de V et solution de PCMM tel que $|V_s| < |V_r|$*

Preuve. Soit $\mathcal{V}^0 = \{V_1^0, \ldots, V_k^0\}$ la solution de référence. Nous allons montrer qu'il existe une solution $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$ vérifiant la proposition.

Posons $\mathcal{V}^1$ tel que :

$$\begin{cases} V_r^1 = V_1^0, \\ V_0^1 = V_r^0, \\ V_s^1 = V_k^0, \\ V_k^1 = V_s^0, \\ V_t^1 = V_t^0 \quad ,\forall t \in \{1,2,\ldots,k\} \setminus \{1,r,s,k\}, \end{cases}$$

Par construction de la sotution de référence, on a $|V_s| < |V_r|$. Il est clair que $\mathcal{V}^1 = \{V_1^1,\ldots,V_k^1\}$ est une solution de PCMM et que $|V_s^1| < |V_r^1|$. □

Proposition 3 *Soient $u \in V$ et $r,s \in \{1,2,\ldots,k\}$ tel que $r \neq s$. Il existe $\mathcal{V} = \{V_1,V_2,\ldots,V_k\}$, une partition de V et solution de (P) tel que u appartient à V_r et $|V_s| < |V_r|$*

Preuve. Soit $\mathcal{V}^1 = \{V_1^1,V_2^1,\ldots,V_k^1\}$ une solution de PCMM tel que $|V_s^1| < |V_r^1|$ (d'après la proposition 2, on sait qu'une telle solution existe). Si u n'appartient pas à V_r^1, il suffit de permuter u avec n'importe quel sommet de V_r^1 pour avoir une solution qui respecte les conditions de la proposition 3. □

Proposition 4 *Soient u,v et w trois sommets distincts de V et $t \in \{1,2,\ldots,k\}$. Il existe $\mathcal{V} = \{V_1,V_2,\ldots,V_k\}$, une partition de V est solution de PCMM tel que u et v appartiennent au même ensemble de $\mathcal{V}$, c'est-à-dire qu'il existe $r \in \{1,2,\ldots,k\}$ tel que $u,v \in V_r$ et w n'appartient pas à l'ensemble V_t*

Preuve. $\mathcal{V}^1 = \{V_1^1,V_2^1,\ldots,V_k^1\}$ une de solution de PCMM tel que u et v appartient au même ensemble (d'après la proposition 1 on sait qu'une telle solution existe). Si $w \in V_t^1$, soit $s,t \in \{1,2,\ldots,k\}$ tel que $s \neq t$. Posons $\mathcal{V}^2 = \{V_1^2,V_2^2,\ldots,V_k^2\}$ tel que $V_s^2 = V_t^1$, $V_t^2 = V_s^1$ et $\forall r \in \{1,2,\ldots,k\} \setminus \{s,t\}$, $V_r^2 = V_r^1$. $\mathcal{V}^2$ est une solution de PCMM qui vérifie les conditions de la proposition 4. □

Proposition 5 *Soient $u \neq v \in V$ et $t,r,s \in \{1,2,\ldots,k\}$ tel que $r \neq s$. Il existe $\mathcal{V} = \{V_1,V_2,\ldots,V_k\}$, partition de V et solution de P tel que u n'appartient à l'ensemble V_t, v appartient à V_r et $|V_s| < |V_r|$*

Preuve. Soit $\mathcal{V}^1 = \{V_1^1,V_2^1,\ldots,V_k^1\}$ une de solution de PCMM tel que $v \in V_r^1$ et $|V_s^1| < |V_r^1|$ (d'après la proposition 3, on sait que $\mathcal{V}^1$ existe). Si $u \in V_t^1$, il suffit de permuter u avec n'importe quel sommet de $V \setminus \{V_t^1 \cup \{v\}\}$ pour avoir une solution qui vérifie les conditions de la proposition 5. □

Dans la suite, notons par M en entier tel que $M = \sum_{e \in E} w(e)$

6.4.1 Dimension du $P(G,k,q)$

Dans cette section, nous caractérisons la dimension de $P(G,k,q)$. Pour cela, nous montrons d'abord que toute équation de $P(G,k,q)$ est une combinaison linéaire des contraintes (6.2).

Proposition 6 *Toute équation $a^Tx + b^Ty + cz = d$ est une combinaison linéaire des équations (6.2)*

Preuve. Soit $a^Tx + b^Ty + cz = d$ une equation de $P(G,k,q)$. On montre qu'il existe $\lambda = (\lambda_u \mid u \in V) \in \mathbb{R}^n$ tel que $a = \lambda A$ et où $b = 0$ et $c = 0$.

Dans un premier temps nous commençons par montrer que $c = 0$. Soit $\mathcal{V}_0$ la solution de référence et (x^0, y^0, z^0) est le vecteur d'incidence associé à $\mathcal{V}_0$. On pose $(x^{0'}, y^{0'}, z^{0'}) = (x^0, y^0, z^0 + 1)$. Il est clair que $(x^{0'}, y^{0'}, z^{0'})$ est une solution de notre programme linéaire. Cela implique que les vecteurs d'incidence (x^0, y^0, z^0) et $(x^{0'}, y^{0'}, z^{0'})$ vérifient l'équation $a^Tx+b^Ty+cz = d$. On en déduit que $a^Tx^0+b^Ty^0+cz^0 = a^Tx^{0'} + b^Ty^{0'} + cz^{0'}$. Il s'ensuit que $c = 0$.

Désormais montrons que $b_{e,r,t} = 0$ pour tout $e \in E$ et $r, t \in \{1, ..., k\}$ tel que $r \neq t$. Soient $uv \in E$ une arête quelconque et r, t deux indices quelconques de $\{1, ..., k\}$ tel que $r \neq t$. Par la proposition 1, considérons $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$, une solution de PCMM tel que u et v appartiennent au même ensemble et (x^1, y^1, z^1) le vecteur d'incidence de $\mathcal{V}^1$. Soit $(x^{1'}, y^{1'}, z^{1'})$ le vecteur obtenu à partir de (x^1, y^1, z^1) en posant $y^{1'}_{uv,r,t} = 1$ et $z^{1'} = M$. Il est clair que $(x^{1'}, y^{1'}, z^{1'})$ est une solution de (P) Le vecteur $(x^{1'}, y^{1'}, z^{1'})$ est bien une solution de notre programme linéaire. Cela implique que les deux vecteurs d'incidence (x^1, y^1, z^1) et $(x^{1'}, y^{1'}, z^{1'})$ vérifient l'équation $a^Tx + b^Ty + cz = d$. Étant donné que $c = 0$ on en déduit que $b_{e,r,t} = 0$. Comme uv est une arête quelconque et que les indices r et t sont eux aussi quelconques, il s'ensuit que $b = 0$.

Nous allons à présent montrer que pour tout sommet $u \in V$ et pour tout couple $r, t \in \{1, \ldots, k\}$ tel que $r \neq t$, nous avons $a_{u,r} = a_{u,t}$. Soient r, s deux indices quelconques de $\in \{1, \ldots, k\}$ tel que $r \neq s$ et u un sommet quelconque de V. Par la proposition 3, soit $\mathcal{V}^2 = \{V_1^2, \ldots, V_n^2\}$ une solution de V tel que $u \in V_r^2$ et $|V_s^2| < |V_r^2|$. Soit (x^2, y^2, z^2) le vecteur d'incidence associé à $\mathcal{V}^2$. Considérons aussi la solution $\mathcal{V}^{2'}$ construit de la manière suivante :

$$\begin{cases} V_r^{2'} = V_r^2 \setminus \{u\}, \\ V_s^{2'} = V_s^2 \cup \{u\}, \\ V_t^{2'} = V_t^2, \qquad \forall t \in \{1, 2, \ldots, k\} \setminus \{r, t\}, \end{cases}$$

et soit $(x^{2'}, y^{2'}, z^{2'})$ son vecteur d'incidence. Remarquons que $\mathcal{V}^{2'} = \{V_1^{2'}, \ldots, V_k^{2'}\}$ est une solution de PCMM. Cela implique que les vecteurs d'incidence de ces solutions vérifient l'équation $a^T x + b^T y + cz = d$. Comme $c = 0$ et $b_{e,r,t} = 0$ pour tout $e \in E$ et $r \neq t \in \{1, ..., k\}$ et que u, r et s sont quelconques il s'ensuit que $a_{u,r} = a_{u,s}$. On pose $\lambda_u = a_{u,t}$ pour tout $t \in \{1, 2, \ldots, k\}$. On en déduit que $a = \lambda A$, $b = 0$ et $c = 0$. La preuve est terminée. □

Théoreme 1 $dim(P(G,k,q)) = n.k + m.\frac{k(k-1)}{2} + 1 - n$

Preuve. Par la proposition 6, les seules équations du polytope $P(G,k,q)$ sont les contraintes de partionnement (6.2). Il en résulte que $dim(P(G,k,q)) = N - rang(A)$ où $N = n.k + m.\frac{k(k-1)}{2} + 1$ est le nombre de variables.

La matrice A est une matrice de n lignes et nk colonnes qui pourra être écrite comme suit :

$$A = \begin{pmatrix} A^1 & & & \\ & A^2 & & \\ & & \ddots & \\ & & & A^n \end{pmatrix}$$

Avec pour tout i de $\{1, 2, \ldots, n\}$ la matrice A^i est une matrice d'une seule ligne et de k colonnes tel que :

$$A_i = \begin{pmatrix} 1 & 1 & . & . & . & 1 \end{pmatrix}$$

Notons par $(A_1, A_2, \ldots, A_{nm})$ les colonnes de la matrice A. Il est clair que les colonnes $A_1, A_{k+1}, \ldots, A_{ik+1}, \ldots, A_{(n-1)k+1}$ forment la matrice d'identité $n \times n$ et sont donc linéaiment indépendantes. Nous en déduisons que $rang(A) = n$ et que

$$dim(P(G,k,q)) = n \times k + m \times \frac{k(k-1)}{2} + 1 - n.$$

□

6.4.2 Étude faciale du polytope du $P(G,k,q)$

6.4.2.1 Inégalités triviales

Théoreme 2 *Les inégalités $x_{u,t} \geq 0$, où $u \in V$ et $t \in \{1, 2, \ldots, k\}$, définissent des facettes pour $P(G,k,q)$.*

Preuve. Soient u^* un sommet quelconque de V, et $t^* \in \{1, 2, \ldots, k\}$. On montre que la contrainte $x_{u^*,t^*} \geq 0$ définit une facette du polytope du $P(G, k, q)$. Pour cela, notons $x_{u^*,t^*} \geq 0$ par $a^T x + b^T y + cz \geq \alpha$ et soit $F_{u^*}^{t^*}$ la face induite par cette inégalité, c'est-à-dire

$$F_{u^*}^{t^*} = \{(x, y, z) \in P(G, k, q)) \mid a^T x + b^T y + cz = \alpha\}$$

Soit $a'^T x + b'^T y + c'z \geq \beta$ une inégalité valide définissant une facette F de $P(G, k, q)$ telle que $F_{u^*}^{t^*} \subseteq F$. On montre qu'il existe $\gamma \in \mathbb{R}$, $\lambda = \{\lambda_u^t \mid u \in V \text{ et } t \in \{1, 2, \ldots, k\}\} \in \mathbb{R}^{|V|}$ tels que

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

(rappellons que A est la matrice qui représente le système d'equations (6.2) du programme (P)).

Pour cela, nous commençons par montrer que $c' = b' = 0$. Soient u et v deux sommets quelconques de V. Par la proposition 4, il existe une solution $\mathcal{V}^0 = \{V_1^0, \ldots, V_k^0\}$ du PCMM telle que les sommets u et v appartiennent au même ensemble et u^* n'appartient pas à $V_{t^*}^0$. Soit (x^0, y^0, z^0) le vecteur d'incidence de $\mathcal{V}^0$. On a donc $x_{u^*,t^*} = 0$. Soit la solution $(x^{0'}, y^{0'}, z^{0'}) = (x^0, y^0, z^0 + 1)$. Il est clair que $(x^{0'}, y^{0'}, z^{0'})$ est aussi une solution de (P) avec $x_{u^*,t^*}^{0'} = 0$. Cela implique que (x^0, y^0, z^0) et $(x^{0'}, y^{0'}, z^{0'})$ vérifient l'équation $a'x + b'y + c'z = \beta$ et par conséquent $c' = 0$.

Considérons de nouveau la solution $\mathcal{V}^0$. Rappelons que les deux sommets u et v appartiennent au même élément de la partition. Soient $r, s \in \{1, 2, \ldots, k\}$ tels que $r \neq s$. On a $y_{uv,r,s}^0 = 0$. Soit $(x^{0''}, y^{0''}, z^{0''})$ la solution de (P) obtenu de (x^0, y^0, z^0) en posant $y_{uv,r,s}^{0''} = 1$ et $z^{0''} = M$. Il est clair que $(x^{0''}, y^{0''}, z^{0''}) \in F_{u^*}^{t^*}$. Il en résulte que les deux vecteurs (x^0, y^0, z^0) et $(x^{0''}, y^{0''}, z^{0''})$ vérifient l'égalité $a'x + b'y + c'z = \beta$. Comme $c = 0$, on en déduit que $b'_{uv,r,s} = 0$. Comme uv est une arête quelconque et r et s sont deux indices quelconques, il en résulte $b' = 0$

Nous allons à présent montrer que pour tout sommet $v \in V \backslash \{u^*\}$, et pour tout $r, s \in \{1, 2, \ldots, k\}$ tel que $r \neq s, a'_{v,r} = a'_{v,s}$. D'après la proposition 5, il existe une solution $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$ du PCMM telle que $u^* \notin V_{t^*}$, $v \in V_r$ et $|V_s| < |V_r|$. Notons que t^* peut être égal à r ou s. Soit (x^1, y^1, z^1) le vecteur d'incidence associé à $\mathcal{V}^1$. On a $x_{u^*,t^*}^1 = 0$. Considérons aussi la solution $\mathcal{V}^{1'} = \{V_1^{1'}, V_2^{1'}, \ldots, V_k^{1'}\}$ du PCMM telle que $V_r^{1'} = V_r^1 \setminus \{v\}$, $V_s^{1'} = V_s^1 \cup \{v\}$ et $V_t^{1'} = V_t^1$, pour tout $t \in \{1, 2, \ldots, k\} \setminus \{r, t\}$. Soit $(x^{1'}, y^{1'}, z^{1'})$ le vecteur d'incidence de $\mathcal{V}^{1'}$. On a $x_{u^*,t^*}^{1'} = 0$. Il en résulte que (x^1, y^1, z^1) et $(x^{1'}, y^{1'}, z^{1'})$ vérifient l'égalité $a'x + b'y + c'z = \beta$. Comme $c' = 0$ et $b' = 0$, on peut en déduire que $a'_{v,r} = a'_{v,s}$.

Nous allons montrer que pour tout $r,s \in \{1,2,\ldots,k\} \setminus \{t^*\}$, $a'_{u^*,r} = a'_{u^*,s}$. Dans le cas où $k=2$, on a $r=s$ et l'égalité précédente est triviale. Supposons que $k \geq 3$. D'après la proposition 3, il existe une solution $\mathcal{V}^2 = \{V_1^2, V_2^2, \ldots, V_k^2\}$ du PCMM telle que $u^* \in V_r$ et $|V_s| < |V_r|$. Soit (x^2, y^2, z^2) le vecteur d'incidence associé à $\mathcal{V}^2$. On a $x^2_{u^*,t^*} = 0$. Soit la solution $\mathcal{V}^{2'} = \{V_1^{2'}, V_2^{2'}, \ldots, V_k^{2'}\}$ construite comme suit :

$$\begin{cases} V_r^{2'} = V_r^2 \setminus \{u^*\}, \\ V_s^{2'} = V_s^2 \cup \{u^*\}, \\ V_t^{2'} = V_t^2, & \text{pour tout } t \in \{1,2,\ldots,k\} \setminus \{r,t\} \end{cases}$$

Soit $(x^{2'}, y^{2'}, z^{2'})$ son vecteur d'incidence. Il est clair que (x^2, y^2, z^2) et $(x^{2'}, y^{2'}, z^{2'})$ vérifient tous les deux l'égalité $x_{u^*,t^*} = 0$. Il en résulte que (x^2, y^2, z^2) et $(x^{2'}, y^{2'}, z^{2'})$ satisfont aussi l'égalité $a'x + b'y + c'z = \beta$. Étant donné que $c' = 0$ et $b' = 0$, on en déduit que $a'_{u^*,r} = a'_{u^*,s}$.

De ce qui précéde, nous obtenons que :

$$\begin{cases} c' = 0, \\ b' = 0, \\ a'_{v,t} = a'_{v,r} = \lambda_v, & \text{pour tout } v \neq u^* \in V \text{ et pour tout } r,t \in \{1,2,\ldots,k\} \\ a'_{u^*,t} = a'_{u^*,r} = \lambda_{u^*}, & \text{pour tout } r,t \in \{1,2,\ldots,k\} \setminus \{t^*\} \end{cases}$$

On pose $\gamma = a'_{u^*,t^*} - \lambda_{u^*}$

Nous avons alors :

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Ce qui montre que les inégalités $x_{u,t} \geq 0, u \in V$ et $t \in \{1,2,\ldots,k\}$ définissent des facettes pour $P(G,k,q)$. □

Théoreme 3 *Les inégalités $y_{uv,r,t} \leq 1$, où $uv \in E$ et $r \neq t \in \{1,2,\ldots,k\}$ définissent des facettes pour $P(G,k,q)$.*

Preuve. Soient u^*v^* une arête quelconque de E et r^* et s^* deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r^* \neq s^*$. On montre que la contrainte $y_{u^*v^*,r^*,s^*} \leq 1$ définit une facette du polytope $P(G,k,q)$. Pour cela, notons $y_{u^*v^*,r^*,s^*} \leq 1$ par $ax + by + cz \geq \alpha$ et $F^{r^*s^*}_{u^*v^*}$ la face induite par cette inégalité, c'est-à-dire que

$$F^{r^*s^*}_{u^*v^*} = \{(x,y,z) \in P(G,k,q)) \mid ax + by + cz = \alpha\}$$

Soient $a'x+b'y+c'z \leq \beta$ une inégalité valide définissant une facette F de $P(G,k,q)$ telle que $F^{r^*s^*}_{u^*v^*} \subseteq F$. On montre qu'il existe $\gamma \in \mathbb{R}$, $\lambda = \{\lambda^t_u \,|u \in V \text{ et } t \in \{1,2,\ldots,k\}\} \in \mathbb{R}^{|V|}$ tels que

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Nous commençons par montrer que $c' = 0$ et que, pour toute arête $uv \in E$ et pour tout $r,s \in \{1,2,\ldots,k\}$ tels que $r \neq s$ et $(uv,r,s) \neq (u^*v^*,r^*,s^*)$, $b'_{uv,r,s} = 0$. Soit uv une arête quelconque de $E\backslash\{u^*v^*\}$. Considérons $\mathcal{V}^0 = \{V^0_1, V^0_2, \ldots, V^0_k\}$ une solution de PCMM tel que u et v appartiennent au même ensemble et (x^0,y^0,z^0) son vecteur d'incidence. Soit $(x^{0'},y^{0'},z^{0'})$ la solution obtenue à partir de (x^0,y^0,z^0) en posant $y^{0'}_{u^*v^*,r^*,s^*} = 1$ et $z^{0'} = M$, et soit $(x^{0''},y^{0''},z^{0''})$ la solution obtenue à partir de $(x^{0'},y^{0'},z^{0'})$ en posant $z^{0''} = z^{0'} + 1$. Il est clair que $(x^{0'},y^{0'},z^{0'})$ et $(x^{0''},y^{0''},z^{0''})$ sont deux solutions du problème PCMM qui appartiennent à la face $F^{r^*s^*}_{u^*v^*}$. Cela implique que $(x^{0'},y^{0'},z^{0'})$ et $(x^{0''},y^{0''},z^{0''})$ satisfont l'égalité $a'x + b'y + c'z = \beta$. Il en résulte que $c' = 0$.

Soient r et s deux indices quelconques de $\{1,2,\ldots,k\}$ et soit $(\bar{x}^{0'},\bar{y}^{0'},\bar{z}^{0'})$ la solution obtenue à partir de $(x^{0'},y^{0'},z^{0'})$ en fixant $\bar{y}^{0'}_{uv,r,s} = 1$ et $\bar{z}^{0'} = M$. Remarquons que $(\bar{x}^{0'},\bar{y}^{0'},\bar{z}^{0'})$ est une solution de PCMM qui appartient à la face $F^{r^*s^*}_{u^*v^*}$. Les deux vecteurs $(\bar{x}^{0'},\bar{y}^{0'},\bar{z}^{0'})$ et $(x^{0'},y^{0'},z^{0'})$ satisfont l'équation $a'x + b'y + c'z = \beta$. Étant donné que $c' = 0$, on en déduit que $b'_{uv,r,s} = 0$.

Considérons maintenant $\mathcal{V}^1 = \{V^1_1, V^1_2, \ldots, V^1_k\}$ une solution de PCMM tel que u^* et v^* appartiennent au même ensemble. Notons par (x^1,y^1,z^1) le vecteur d'incidence de $\mathcal{V}^1$. Soient $r,s \in \{1,2,\ldots,k\}$ tels que $r \neq s$ et $(r,s) \neq (r^*,s^*)$. Soit $(x^{1'},y^{1'},z^{1'})$ la solution obtenue à partir de (x^1,y^1,z^1) en posant $y^{1'}_{u^*v^*,r^*,s^*} = 1$ et $z^{1'} = M$, et soit $(x^{1''},y^{1''},z^{1''})$ la solution obtenue à partir de $(x^{1'},y^{1'},z^{1'})$ en posant $y^{1''}_{u^*v^*,r,s} = 1$. Observons que $(x^{1'},y^{1'},z^{1'})$ et $(x^{1''},y^{1''},z^{1''})$ sont deux solutions du problème PCMM qui appartiennent à la face $F^{r^*s^*}_{u^*v^*}$ et, par conséquence, satisfont l'égalité $a'x + b'y + c'z = \beta$. Sachant que $c' = 0$, nous en déduisons que $b'_{u^*v^*,r,s} = 0$.

Nous allons à présent montrer que pour tout sommet $u \in V$ et pour tout couple $r,t \in \{1,\ldots,k\}$ tel que $r \neq t$, nous avons $a'_{u,r} = a'_{u,t}$. Soient $r,s \in \{1,\ldots,k\}$ tel que $r \neq s$ et $u \in V$. Soient $\mathcal{V}^2 = \{V^2_1,\ldots,V^2_n\}$ une solution de V tel que $u \in V^2_r$ et $|V^2_s| < |V^2_r|$ et (x^2,y^2,z^2) son vecteur d'incidence. Soient $(\bar{x}^2,\bar{y}^2,\bar{z}^2)$ la solution obtenue à partir de (x^2,y^2,z^2) en posant $\bar{y}^2_{u^*v^*,r^*,s^*} = 1$ et $\bar{z}^2 = M$, et soit $(x^{1''},y^{1''},z^{1''})$ la solution obtenue à partir de $(x^{1'},y^{1'},z^{1'})$ en posant $y^{1''}_{u^*v^*,r,s} = 1$.

Considérons aussi la solution $\mathcal{V}^{2'}$ construite de la manière suivante :

$$\begin{cases} V_r^{2'} = V_r^2 \setminus \{u\}, \\ V_s^{2'} = V_s^2 \cup \{u\}, \\ V_t^{2'} = V_t^2, \qquad \text{pour tout } t \in \{1,2,\ldots,k\} \setminus \{r,s\} \end{cases}$$

Soit $(x^{2'}, y^{2'}, z^{2'})$ le vecteur d'incidence de $\mathcal{V}^{2'}$ et $(\bar{x}^{2'}, \bar{y}^{2'}, \bar{z}^{2'})$. Remarquons que $(\bar{x}^2, \bar{y}^2, \bar{z}^2)$ et $(\bar{x}^{2'}, \bar{y}^{2'}, \bar{z}^{2'})$ sont deux solutions du PCMM qui appartiennent à la face $F_{u^*v^*}^{r^*s^*}$. Ils vérifient donc l'égalité $a'x + b'y + c'z = \beta$. Étant donné que $c' = 0$ et que, pour toute arête $uv \in E$ et pour tout couple d'indice $r, s \in \{1,2,\ldots,k\}$ tels que $r \neq s$ et $(uv,r,s) \neq (u^*v^*, r^*, s^*)$, $b'_{uv,r,s} = 0$, on en déduit que $a'_{u,r} = a'_{u,s}$

De ce qui précéde, nous avons :

- $c' = 0$
- $b'_{uv,r,s} = 0$ pour toute arête uv de E et tout couple d'indice r, s de $\{1,2,\ldots,k\}$ tels que $r \neq s$ et $(uv,r,s) \neq (u^*v^*, r^*, s^*)$
- $a'_{u,t} = a'_{u,r} = \lambda_u$ pour tout $u \in V$ et pour tout $r, t \in \{1,2,\ldots,k\}$

En posant $\gamma = b'_{u^*v^*,r^*,s^*}$, nous obtenons

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Cela montre que les inégalités $x_{u,t} \geq 0, u \in V$ et $t \in \{1,2,\ldots,k\}$ définissent des facettes pour $P(G,k,q)$. □

Théoreme 4 *Les inégalités $y_{uv,r,t} \geq 0$, où $uv \in E$ et $r, t \in \{1,2,\ldots,k\}$ tel que $r \neq t$ définissent des facettes pour $P(G,k,q)$ si et seulement si $k \geq 3$.*

Preuve. ($\Rightarrow$) Supposons d'abord que $k = 2$. En d'autre termes, nous cherchons une partition qui comporte deux ensembles. Soient V_r et V_t les deux ensembles de la partition et uv une arête quelconque de E. La contrainte (6.4) peut s'écrire :

$$x_{u,r} + x_{v,t} \leq y_{uv,r,t} + 1 \tag{6.12}$$

$$x_{v,r} + x_{u,t} \leq y_{uv,r,t} + 1 \tag{6.13}$$

D'autre part les contraintes de partionnement (6.2) permettent d'écrire

$$-x_{u,r} - x_{u,t} \leq -1 \tag{6.14}$$

$$-x_{u,r} - x_{u,t} \leq -1 \tag{6.15}$$

En sommant les inégalités (6.12), (6.13) et les égalités (6.14) et (6.15), on obtient $2y_{uv,r,t} \geq 0$. Comme uv est une arête quelconque, on en déduit que si $k = 2$, alors les contraintes $y_{uv,r,t} \geq 0$ sont une combinaison linéaire des contraintes (6.2) et (6.4) et par conséquence, elles ne peuvent pas définir une facette pour $P(G, k, q)$.

($\Leftarrow$) Supposons maintenant que $k \geq 3$ et montrons que les inégalités $y_{uv,r,t} \geq 0, uv \in E$ et $r, t \in \{1, 2, \ldots, k\}$ tel que $r \neq t$, définissent des facettes pour $P(G, k, q)$.

Soient u^*v^* une arête quelconque de E, et r^* et s^* deux indices quelconques de $\{1, 2, \ldots, k\}$ tel que $r^* \neq s^*$. On montre que la contrainte $y_{u^*v^*,r^*,s^*} \geq 0$ définit une facette du polytope $P(G, k, q)$. Pour cela, notons $y_{u^*v^*,r^*,s^*} \geq 0$ par $ax+by+cz \geq \alpha$ et $F^{r^*s^*}_{u^*v^*}$ la face induite par cette inégalité, c'est-à-dire

$$F^{r^*s^*}_{u^*v^*} = \{(x, y, z) \in P(G, k, q)) \mid ax + by + cz = \alpha\}$$

Soit $a'x+b'y+c'z \leq \beta$ une inégalité valide définissant une facette F de $P(G, k, q)$ telle que $F^{r^*s^*}_{u^*v^*} \subseteq F$. On montre qu'il existe $\gamma \in \mathbb{R}$, $\lambda = \{\lambda^t_u \mid u \in V$ et $t \in \{1, 2, \ldots, k\} \in \mathbb{R}^{|V|}$ tels que

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Nous allons d'abord montrer que $c' = 0$ et que, pour toute arête $uv \in E$ et pour tout $r, s \in \{1, 2, \ldots, k\}$ tels que $r \neq s$ et $(uv, r, s) \neq (u^*v^*, r^*, s^*)$, $b'_{uv,r,s} = 0$. Soit uv une arête quelconque de $E \setminus \{u^*v^*\}$. Considérons $\mathcal{V}^0 = \{V^0_1, V^0_2, \ldots, V^0_k\}$ une solution de PCMM tel que u^* et v^* appartiennent au même ensemble et que u et v appartiennent au même ensemble. Soient (x^0, y^0, z^0) le vecteur d'incidence de $\mathcal{V}^0$ et $(x^{0'}, y^{0'}, z^{0'})$ la solution obtenue à partir de (x^0, y^0, z^0) en posant $z^{0'} = z^0 + 1$. Remarquons que les deux vecteurs (x^0, y^0, z^0) et $(x^{0'}, y^{0'}, z^{0'})$ appatiennent à la face $F^{r^*s^*}_{u^*v^*}$. Cela implique que (x^0, y^0, z^0) et $(x^{0'}, y^{0'}, z^{0'})$ vérifient l'égalité $a'x+b'y+c'z = \beta$. Nous en déduisons que $c' = 0$.

Soient r et s deux indices quelconques de $\{1, 2, \ldots, k\}$ et $(x^{0''}, y^{0''}, z^{0''})$ le vecteur obtenu à partir de $(x^{0'}, y^{0'}, z^{0'})$ en fixant $y^{0''}_{uv,r,s} = 1$ et $z^{0''} = M$. Il est clair que $(x^{0''}, y^{0''}, z^{0''})$ est une solution de P qui vérifie l'égalité $y_{u^*v^*,r^*,s^*} = 0$. Étant donné que les deux vecteurs (x^0, y^0, z^0) et $(x^{0''}, y^{0''}, z^{0''})$ satisfont l'égalité $a'x+b'y+c'z = \beta$. et que $c = 0$, on en déduit que $b'_{uv,r,s} = 0$.

Considérons à présent $\mathcal{V}^1 = \{V^1_1, V^1_2, \ldots, V^1_k\}$ une solution de PCMM tel que u^* et v^* appartiennent au même ensemble et soit (x^1, y^1, z^1) le vecteur d'incidence de $\mathcal{V}^1$.

Soient r,s deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r \neq s$ et $(r,s) \neq (r^*,s^*)$. Soit $(x^{1'},y^{1'},z^{1'})$ le vecteur obtenu à partir de (x^1,y^1,z^1) en posant $y^{1'}_{u^*v^*,r,s}=1$ et $z^{1'}=M$. Notons que (x^1,y^1,z^1) et $(x^{1'},y^{1'},z^{1'})$ appartiennent à la face $F^{r^*s^*}_{u^*v^*}$. Par conséquence, ils satisfont l'égalité $a'x+b'y+c'z=\beta$. Étant donné que $c'=0$, il en suit que $b'_{u^*v^*,r,s}=0$.

Soient v un sommet quelconque de $V\setminus\{u^*,v^*\}$ et r,s deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r\neq s$. Nous allons montrer que $a'_{v,r}=a'_{v,s}$. Pour cela, considérons $\mathcal{V}^2=\{V^2_1,V^2_2,\ldots,V^2_k\}$ une solution de PCMM tels que u^* et v^* appartiennent au même ensemble de $\mathcal{V}^2$, $v\in V^2_r$ et $|V^2_s|\leq|V^2_r|$. Notons par (x^2,y^2,z^2) le vecteur d'incidence de $\mathcal{V}^2$. Soient $\mathcal{V}^{2'}=\{V^{2'}_1,V^{2'}_2,\ldots,V^{2'}_k\}$ la solution construite de la manière suivante :

$$\begin{cases} V^{2'}_r = V^2_r\setminus\{v\}, \\ V^{2'}_s = V^2_s\cup\{v\}, \\ V^{2'}_t = V^2_t, & \text{pour tout } t\in\{1,2,\ldots,k\}\setminus\{r,s\} \end{cases}$$

Notons par $(x^{2'},y^{2'},z^{2'})$ le vecteur d'incidence de $\mathcal{V}^{2'}$. Les deux vecteurs (x^2,y^2,z^2) et $(x^{2'},y^{2'},z^{2'})$ sont deux solutions de P qui satisfont l'égalité $y_{u^*v^*,r^*,s^*}=0$. Par conséquence, (x^2,y^2,z^2) et $(x^{2'},y^{2'},z^{2'})$ satisfont aussi l'égalité $a'x+b'y+c'z=\beta$. Étant donné que $c'=0$ et que, pour toute arête $uv\in E$ et pour tout couple d'indice $r,s\in\{1,2,\ldots,k\}$ tel que $r\neq s$ et $(uv,r,s)\neq(u^*v^*,r^*,s^*)$, $b'_{uv,r,s}=0$, nous déduisons que $a'_{v,r}=a'_{v,s}$.

Étant donné que $k\geq 3$, l'ensemble $\{1,2,\ldots,k\}\setminus\{r^*,s^*\}$ n'est pas vide. Soient r un indice quelconque de $\{1,2,\ldots,k\}\setminus\{r^*,s^*\}$, s un indice quelconque de $\{1,2,\ldots,k\}$ et $\mathcal{V}^3=\{V^3_1,V^3_2,\ldots,V^3_k\}$ une solution de PCMM tels que u^* et v^* appartiennent au même ensemble V^3_r et $|V^3_s|\leq|V^3_r|$. Notons par (x^3,y^3,z^3) le vecteur d'incidence de $\mathcal{V}^3$ et soi $v\in\{u^*,v^*\}$. Considérons la solution $\mathcal{V}^{3'}=\{V^{3'}_1,V^{3'}_2,\ldots,V^{3'}_k\}$ construite de la manière suivante :

$$\begin{cases} V^{3'}_r = V^3_r\setminus\{v\}, \\ V^{3'}_s = V^3_s\cup\{v\}, \\ V^{3'}_t = V^3_t, & \text{pour tout } t\in\{1,2,\ldots,k\}\setminus\{r,s\} \end{cases}$$

Soit $(x^{3'},y^{3'},z^{3'})$ le vecteur d'incidence de $\mathcal{V}^{3'}$. Il est clair que les deux vecteurs (x^3,y^3,z^3) et $(x^{3'},y^{3'},z^{3'})$ appartiennent à $P(G,k,q)$ et à la face $F^{r^*s^*}_{u^*v^*}$. Par conséquence, ces deux vecteurs satisfont l'égalité $a'x+b'y+c'z=\beta$. Étant donné que $c=0$ et que, pour toute arête $uv\in E$ et pour tout couple d'indice $r,s\in\{1,2,\ldots,k\}$ tels que $r\neq s$ et $(uv,r,s)\neq(u^*v^*,r^*,s^*)$, $b'_{uv,r,s}=0$, on en déduit que $a'_{v,r}=a'_{v,s}$. Comme s est un indice quelconque de $\{1,2,\ldots,k\}$, il en résulte que pour tout $r\neq t\in\{1,2,\ldots,k\}$ et pour tout $v\in\{u^*,v^*\}$, $a'_{v,r}=a'_{v,s}$.

De ce qui précéde, nous avons :

- $c' = 0$
- $b'_{uv,r,s} = 0$ pour toute arête uv de E et tout couple d'indice $r \neq s$ de $\{1, 2, \ldots, k\}$ tels que $uv \neq u^*v^*$ ou le couple $(r, s) \neq (r^*, s^*)$
- $a'_{u,t} = a'_{u,r} = \lambda_u$ pour tout $u \in V$ et pour tout $r, t \in \{1, 2, \ldots, k\}$

En posant $\gamma = b'_{u^*v^*,r^*,s^*}$, nous obtenons :

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Cela montre, dans le cas où $k \geq 3$, les inégalités $y_{uv,r,t} \geq 0, uv \in E$ et $r, t \in \{1, 2, \ldots, k\}$ tel que $r \neq t$ définissent des facettes pour $P(G, k, q)$. □

6.4.2.2 Inégalités d'arête

Avant d'étudier l'aspect faciale des inégalités d'arête (6.4), nous commençons par introduire une propriété qui sera utilisée dans l'étude faciale de ces contraintes.

Proposition 7 *Soient $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ une solution de PCMM et $(\bar{x}, \bar{y}, \bar{z})$ le vecteur d'incidence de $\mathcal{V}$. Soient uv une arête quelconque de E et r, s deux indices quelconques de $\{1, 2, \ldots, k\}$ tel que $r \neq s$. Le vecteur $(\bar{x}, \bar{y}, \bar{z})$ vérifie l'égalité*

$$x_{u,r} + x_{v,s} = 1 + y_{uv,r,s} \tag{6.16}$$

si et seulement si $u \in V_r$ ou $v \in V_s$

Preuve. $\Rightarrow$ Supposons que $u \notin V_r$ et $v \notin V_s$. Il est clair que $\bar{x}_{u,r} = \bar{x}_{v,s} = 0$ et par conséquent le vecteur $(\bar{x}, \bar{y}, \bar{z})$ ne peut pas satisfaire l'égalité 6.16.

$\Leftarrow$ Si $u \in V_r$ et $v \notin V_s$, alors $\bar{x}_{u,r} = 1$, $\bar{x}_{v,s} = 0$ et $\bar{y}_{uv,r,s} = 0$. Le vecteur $(\bar{x}, \bar{y}, \bar{z})$ satisfait l'égalité 6.16.

Si $u \notin V_r$ et $v \in V_s$, alors $\bar{x}_{u,r} = 0$, $\bar{x}_{v,s} = 1$ et $\bar{y}_{uv,r,s} = 0$. Le vecteur $(\bar{x}, \bar{y}, \bar{z})$ satisfait l'égalité 6.16.

Si $u \in V_r$ et $v \in V_s$, alors

$$\bar{x}_{u,r} = \bar{x}_{v,s} = \bar{y}_{uv,r,s} = 1.$$

Le vecteur $(\bar{x}, \bar{y}, \bar{z})$ satisfait l'égalité 6.16. □

Théoreme 5 *Les inégalités $x_{u,r}+x_{v,s} \leq 1+y_{uv,r,s}$, où $uv \in E$ et $r,s \in \{1,2,\ldots,k\}$ tel que $r \neq t$, définissent des facettes pour $P(G,k,q)$.*

Preuve. Soient u^*v^* une arête quelconque de E, et r^* et s^* deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r^* \neq s^*$. On montre que la contrainte $x_{u,r}+x_{v,s} \leq 1+y_{uv,r,s}$ définit une facette du polytope $P(G,k,q)$. Pour cela, notons $x_{u,r}+x_{v,s} \leq 1+y_{uv,r,s}$ par $ax+by+cz \geq \alpha$ et $F_{u^*v^*}^{r^*s^*}$ la face induite par cette inégalité. On a

$$F_{u^*v^*}^{r^*s^*} = \{(x,y,z) \in P(G,k,q)) \mid ax+by+cz = \alpha\}$$

Soit $a'x+b'y+c'z \leq \beta$ une inégalité valide définissant une facette F de $P(G,k,q)$ telle que $F_{u^*v^*}^{r^*s^*} \subseteq F$. On montre qu'il existe $\gamma \in \mathbb{R}$, $\lambda = \{\lambda_u^t \mid u \in V$ et $t \in \{1,2,\ldots,k\} \in \mathbb{R}^{|V|}$ tels que

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Commençons par montrer que $c'=0$. Soit $\mathcal{V}^0 = \{V_1^0, V_2^0, \ldots, V_k^0\}$ une solution de PCMM tel que le sommet u^* appartient à l'ensemble $V_{r^*}^0$. Notons par (x^0,y^0,z^0) le vecteur d'incidence de $\mathcal{V}^0$. D'après la proposition 7 $(x^0,y^0,z^0) \in F_{u^*v^*}^{r^*s^*}$. Soit $(x^{0'},y^{0'},z^{0'})$ le vecteur obtenu à partir de (x^0,y^0,z^0) en posant $z^{0'} = z^0+1$. Il est clair que $(x^{0'},y^{0'},z^{0'})$ est une solution de (P). Étant donné que $(x^0,y^0,z^0) \in F_{u^*v^*}^{r^*s^*}$, on a $(x^{0'},y^{0'},z^{0'}) \in F_{u^*v^*}^{r^*s^*}$. Par conséquence, les deux vecteurs (x^0,y^0,z^0) et $(x^{0'},y^{0'},z^{0'})$ satisfont l'égalité $a'x+b'y+c'z=\beta$. Il en résulte que $c'=0$.

Nous allons montrer que $b'_{uv,r,s}=0$ pour toute arête $uv \in E$ et pour tout couple d'indice $r,s \in \{1,2,\ldots,k\}$ tels que $r \neq s$ et $(uv,r,s) \neq (r^*v^*,r^*,s^*)$. Soient uv une arête quelconque de $E \setminus \{(u^*v^*)\}$ et $\mathcal{V}^1 = \{V_1^1, V_2^1, \ldots, V_k^1\}$ une solution de PCMM tel que $u^* \in V_{r^*}^1$ et les deux sommets u et v appartiennent au même ensemble de $\mathcal{V}^1$. Notons par (x^1,y^1,z^1) le vecteur d'incidence de $\mathcal{V}^1$. D'après la proposition 7 et vu que $u^* \in V_{r^*}^1$, (x^1,y^1,z^1) appartient à la face $F_{u^*v^*}^{r^*s^*}$. Soient r,s deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r \neq s$ et soit $(x^{1'},y^{1'},z^{1'})$ le vecteur obtenu à partir de (x^1,y^1,z^1) en posant $y_{uv,r,s}^{1'}=1$ et $z^{1'}=M$. Remarquons que $(x^{1'},y^{1'},z^{1'})$ est une solution de (P). Étant donné que $(x^1,y^1,z^1) \in F_{u^*v^*}^{r^*s^*}$ et que $uv \neq u^*v^*$, on a $(x^{1'},y^{1'},z^{1'})$ appartient à $F_{u^*v^*}^{r^*s^*}$. Il en résulte que les deux vecteurs (x^1,y^1,z^1) et $(x^{1'},y^{1'},z^{1'})$ satisfont l'égalité $a'x+b'y+c'z=\beta$. Étant donné que $c'=0$, il en suit que $b'_{uv,r,s}=0$.

Considérons à présent $\mathcal{V}^2 = \{V_1^2, V_2^2, \ldots, V_k^2\}$ une solution de PCMM tel que les deux sommets u et v appartiennent à l'ensemble V_r^2 et notons par (x^2,y^2,z^2) le vecteur d'incidence de $\mathcal{V}^2$. D'après la propostion 7 et vu que $u^* \in V_{r^*}^1$, (x^1,y^1,z^1) appartient à la face $F_{u^*v^*}^{r^*s^*}$. Soient r et s deux indices quelconques de $\{1,2,\ldots,k\}$

tels que $r \neq s$ et $(r,s) \neq (r^*,s^*)$. Soit $(x^{2'},y^{2'},z^{2'})$ le vecteur obtenu à partir de (x^2,y^2,z^2) en posant $y^{2'}_{uv,r,s}=1$ et $z^{2'}=M$. Par construction, $(x^{2'},y^{2'},z^{2'})$ est une solution de (P). Étant donné que $(x^2,y^2,z^2) \in F^{r^*s^*}_{u^*v^*}$ et que $(r,s) \neq (r^*,s^*)$, on a $(x^{2'},y^{2'},z^{2'}) \in F^{r^*s^*}_{u^*v^*}$. Les deux vecteurs (x^2,y^2,z^2) et $(x^{2'},y^{2'},z^{2'})$ satisfont donc l'égalité $a'x+b'y+c'z=\beta$. Comme $c'=0$, on déduit que $b'_{u^*v^*,r,s}=0$.

Nous allons à présent montrer que pour tout sommet u de $V\setminus\{u^*,v^*\}$ et pour tout couple d'indice r et s de $\{1,2,\ldots,k\}$ tel que $r\neq s$, $a'_{u,r}=a'_{u,s}$. Soit u un sommet quelconque de $V\setminus\{u^*,v^*\}$ et r,s deux indices quelconques de $\{1,2,\ldots,k\}$ tel que $r\neq s$. Considérons $\mathcal{V}^3=\{V^3_1,V^3_2,\ldots,V^3_k\}$ une solution de PCMM tel que $u^*\in V^3_{r^*}$, $u\in V^3_r$ et $|V^3_s|<|V^3_r|$. Notons que chacun de deux indices (r,s) peut appartenir à $\{r^*,s^*\}$. Notons par (x^3,y^3,z^3) le vecteur d'incidence de $\mathcal{V}^3$. Soit la solution $\mathcal{V}^4$ la solution construite comme suit :

- $V^4_r=V^3_r\setminus\{u\}$,
- $V^4_s=V^3_s\cup\{u\}$
- $V^4_t=V^4_t$, pour tout $t\in\{1,2,\ldots,k\}\setminus\{r,s\}$

Soit (x^4,y^4,z^4) le vecteur d'incidence de $\mathcal{V}^4$. Remarquons que, comme $u^*\in V^3_{r^*}$ et $u^*\in V^4_{r^*}$, d'après la proposition 7, les deux vecteurs (x^3,y^3,z^3) et (x^4,y^4,z^4) appartiennent à $F^{r^*s^*}_{u^*v^*}$ et satisfont donc l'égalité $a'x+b'y+c'z=\beta$. Étant donné que $c'=0$ et que pour toute arête uv de E et tout couple d'indice r,s de $\{1,2,\ldots,k\}$ tels que $r\neq s$ et $(uv,r,s)\neq(r^*v^*,r^*,s^*)$, $b'_{uv,r,s}=0$, on déduit que $a'_{u,r}=a'_{u,s}$.

Enfin, nous allons montrer que

- $a'_{u^*,r}=a'_{u^*,s}$ pour tout $r,s\in\{1,2,\ldots,k\}\setminus\{r^*\}$ tel que $r\neq s$
- $a'_{v^*,r}=a'_{u^*,s}$ pour tout $r,s\in\{1,2,\ldots,k\}\setminus\{s^*\}$ tel que $r\neq s$

Remarquons que si $k=2$, les deux égalités précédentes sont triviales. Supposons que $k\geq 3$ et soient $t\in\{1,2,\ldots,k\}\{r^*,s^*\}$ et $\mathcal{V}^5=\{V^5_1,V^5_2,\ldots,V^5_k\}$ une solution de PCMM tels que les deux sommets u^* et v^* appartiennent à l'ensemble $V^5_{s^*}$ et $|V^5_t|<|V^5_{s^*}|$. Notons par (x^5,y^5,z^5) le vecteur d'incidence de $\mathcal{V}^5$. Soit $\mathcal{V}^6=\{V^6_1,V^6_2,\ldots,V^6_k\}$ la solution construite comme suit :

- $V^6_{s^*}=V^5_{s^*}\setminus\{u^*\}$,
- $V^6_t=V^5_t\cup\{u^*\}$
- $V^6_r=V^5_r$, pour tout $r\in\{1,2,\ldots,k\}\setminus\{t,s^*\}$

Soit (x^6,y^6,z^6) le vecteur d'incidence de $\mathcal{V}^6$. Notons que, d'après la proposition 7 et vu que $v^*\in V^5_{s^*}$ et $v^*\in V^6_{s^*}$, les deux vecteurs (x^5,y^5,z^5) et (x^6,y^6,z^6) appartiennent à $F^{r^*s^*}_{u^*v^*}$. Étant donné que $c'=0$, que pour toute arête uv de E et tout couple d'indice r,s de $\{1,2,\ldots,k\}$ tels que $r\neq s$ et $(uv,r,s)\neq(r^*v^*,r^*,s^*)$, $b'_{uv,r,s}=0$ et que (x^5,y^5,z^5) et (x^6,y^6,z^6) satisfont l'égalité $a'x+b'y+c'z=\beta$, on en déduit que $a'_{u^*,t}=a'_{u^*,s^*}$. Cela implique que pour tout couple d'indice r et t de $t\in\{1,2,\ldots,k\}\setminus\{r^*\}$ tel que $r\neq s$, $a'_{u^*,t}=a'_{u^*,r}$. Par symétrie entre u et v, on déduit aussi que pour tout couple d'indice r et t de $t\in\{1,2,\ldots,k\}\{s^*\}$ tel que $r\neq s$, $a'_{v^*,t}=a'_{v^*,r}$.

Soit $\mathcal{V}^7 = \{V_1^7, V_2^7, \ldots, V_k^7\}$ une solution de PCMM tels que les deux sommets u^* et v^* appartiennent à l'ensemble $V_{s^*}^7$ et $|v_{r^*}^7| < |V_{s^*}^7|$ et soit (x^7, y^7, z^7) le vecteur d'incidence de $\mathcal{V}^7$. Considérons la solution $\mathcal{V}^8 = \{V_1^8, V_2^8, \ldots, V_k^8\}$ construit comme suit :

- $V_{s^*}^8 = V_{s^*}^7 \setminus \{u^*\}$,
- $V_{r^*}^8 = V_{r^*}^7 \cup \{u^*\}$
- $V_r^8 = V_r^7$, pour tout $r \in \{1, 2, \ldots, k\} \setminus \{r^*, s^*\}$

Notons par (x^8, y^8, z^8) le vecteur d'incidence de $\mathcal{V}^8$. D'après la proposition 7 et vu que $v^* \in V_{s^*}^7$ et $v^* \in V_{s^*}^8$, les deux vecteurs $(x^7, y^7, z^7) \in F_{u^*v^*}^{r^*s^*}$ et $(x^8, y^8, z^8) \in F_{u^*v^*}^{r^*s^*}$. Il en résulte que (x^7, y^7, z^7) et (x^8, y^8, z^8) satisfont l'égalité $a'x + b'y + c'z = \beta$. Étant donné que $c' = 0$, on déduit que

$$a'_{u^*,s^*} = a'_{u^*,r^*} + b'_{u^*v^*,r^*,s^*}$$

Cela implique que

$$a'_{u^*,r^*} = a'_{u^*,s^*} - b'_{u^*v^*,r^*,s^*}$$

Par symétrie, on peut montrer aussi que

$$a'_{v^*,s^*} = a'_{v^*,r^*} - b'_{u^*v^*,r^*,s^*}$$

De ce qui précéde, nous avons que :

- $c' = 0$,
- $b'_{uv,r,s} = 0$ pour toute arête uv de E et tout couple d'indice r, s de $\{1, 2, \ldots, k\}$ tels que $r \neq s$ et $(uv, r, s) \neq (u^*v^*, r^*, s^*)$,
- $a'_{u,t} = a'_{u,r} = \lambda_u$ pour tout $u \in V \setminus \{u^*, v^*\}$ et pour tout $r, t \in \{1, 2, \ldots, k\}$,
- $a'_{u^*,r} = a'_{u^*,s} = \lambda_{u^*}$ pour tout $r, s \in \{1, 2, \ldots, k\} \setminus \{r^*\}$ tel que $r \neq s$,
- $a'_{v^*,r} = a'_{u^*,s} = \lambda_{v^*}$ pour tout $r, s \in \{1, 2, \ldots, k\} \setminus \{s^*\}$ tel que $r \neq s$,
- $a'_{u^*,r^*} = \lambda_{u^*} - b'_{u^*v^*,r^*,s^*}$,
- $a'_{v^*,s^*} = \lambda_{v^*} - b'_{u^*v^*,r^*,s^*}$

En posant $\gamma = -b'_{u^*v^*,r^*,s^*}$, nous obtenons

$$\begin{cases} a' = \gamma a + \lambda A, \\ b' = \gamma b, \\ c' = \gamma c. \end{cases}$$

Cela montre que les inégalités $x_{u,r} + x_{v,s} \leq 1 + y_{uv,r,s}, uv \in E$ et $r, s \in \{1, 2, \ldots, k\}$ tel que $r \neq t$ définissent des facettes pour $P(G,k,q)$. □

6.5 Inégalités valides

Rappelons que pour qu'une partition $\mathcal{V} = \{V_1, ..., V_k\}$ de V définisse une solution, elle doit satisfaire la contrainte d'équilibrage (6.3). Ceci implique des bornes inférieures et supérieures sur la cardinalité de chaque ensemble de la partition $\mathcal{V}$. Notons par N_{min} et N_{max} respectivement la borne inférieure et supérieure pour chaque ensemble de la partition. Les valeurs de ces bornes dépendent des paramètres n, k et q.

Proposition 8

$$N_{min} \geq \left\lceil \frac{n - q(k-1)}{k} \right\rceil$$

Preuve. Soit la partition $\mathcal{V} = (V_1, ..., V_k)$ de V définissant une solution du problème PCMM tel qu'il existe $V_i \in \mathcal{V}$ vérifiant $|V_i| = N_{min}$. Remarquons que pour tout $t \in \{1, ..., k\} \setminus \{i\}$, $N_{min} \leq V_t \leq N_{min} + q$. Étant donné que $\sum_{t=1}^{k} |V_t| = n$, nous pouvons déduire que

$$\begin{aligned} N_{min} = |V_i| &= n - \sum_{t \neq i} |V_t| \\ &\geq n - \sum_{t \neq i} (N_{min} + q) \\ &\geq n - (k-1)(N_{min} + q) \end{aligned}$$

Ceci implique que

$$kN_{min} \geq n - (k-1)q$$

et par conséquent

$$N_{min} \geq \left\lceil \frac{n - q(k-1)}{k} \right\rceil$$

□

Proposition 9

$$N_{max} \leq \frac{n + q(k-1)}{k}.$$

Preuve. Soit la partition $\mathcal{V} = (V_1, ..., V_k)$ de V définissant une solution du problème PCMM tel qu'il existe $V_i \in \mathcal{V}$ vérifiant $|V_i| = N_{max}$. Remarquons que pour tout $t \in \{1, ..., k\} \setminus \{i\}$, $N_{max} - q \leq V_t \leq N_{max}$. Étant donné que $\sum_{t=1}^{k} |V_t| = n$, nous

pouvons déduire que

$$\begin{aligned} N_{max} = |V_i| &= n - \sum_{t \neq i} |V_t| \\ &\leq n - \sum_{t \neq i} (N_{max} - q) \\ &\leq n - (k-1)(N_{max} - q) \end{aligned}$$

Ceci implique que

$$kN_{max} \leq n + (k-1)q$$

et par conséquent

$$N_{max} \leq \frac{n + q(k-1)}{k}$$

□

La Figure 6.3 illustre une solution réalisable pour le problème PCMM où $n = 7$, $q = 2$ et $k = 3$. Pour cette exemple, nous avons $N_{min} = 1$ et $N_{max} = 3$.

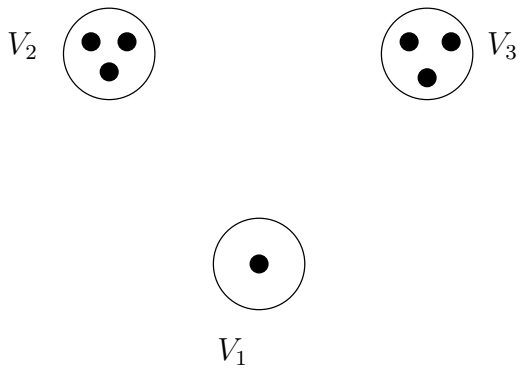

FIGURE 6.3 – Contre exemple pour la valeur de N_{min}

6.5.1 Inégalités de bornes

Considérons les définitions de bornes précédemment données. Il est clair que toute solution du problème PCMM satisfait les inégalités suivantes :

$$\sum_{u \in V}^{k} x_{u,t} \geq N_{min}, \quad \text{pour tout } t \in \{1, ..., k\}, \tag{6.17}$$

et

$$\sum_{u \in V}^{k} x_{u,t} \leq N_{max}, \quad \text{pour tout } t \in \{1, ..., k\}. \tag{6.18}$$

Nous allons désormais introduire une fonction nous permettant de définir plusieurs inégalités valides basées sur cette notion de cardinalité minimum et maximum de chaque ensemble.

Définition 7 *Soit la fonction* $f : \mathcal{W} \mapsto \mathbb{N}$ *qui à tout sous-ensemble de sommets* $W \subseteq V$ *associe le nombre* $f(W)$ *correspondant aux nombres minimum de sous-ensembles de la partition* $\mathcal{V}$ *devant contenir au moins un sommet du sous-ensemble* W. $f(W)$

La valeur de $f(W)$ nous utilisons la fonction récursive $g(|W|, n, k, q)$ suivante :
- calculer $N_{max} = \left\lfloor \frac{n-q(k-1)}{k} \right\rfloor$
- si $|W| \leq N_{max}$, alors $g(|W|, n, k, q) = 1$
- sinon, $g(|W|, n, k, q) = 1 + g(|W| - N_{max}, n - N_{max}, k - 1, q)$

6.5.2 Inégalités de coupe

Considérons un sous-ensemble de sommets $W \subset V$ tel que, le sous graphe induit par W est connexe et $f(W) = 2$. Remarquons que si $f(W) = 2$ cela implique que $N_{max} < |W| < 2N_{max}$. Par définition de $f(W)$, il est évident qu'il n'existe pas de solution $\mathcal{V}$ de PCMM tel que tous les sommets de W appartiennent au même ensemble de la partition $\mathcal{V}$. Soit $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ une solution quelconque de PCMM. Il est clair que les sommets de W appartiennent au moins à deux ensembles. Comme $G[W]$ est connexe, il doit y avoir au moins une arête entre deux ensembles de la partition $\mathcal{V}$. Ceci peut être traduit par la contrainte suivante

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E[W]} y_{uv,r,t} \geq 1 \qquad \begin{array}{l} \text{pour tout } W \subset V \text{ tel que} \\ G[W] \text{ est connexe et } |W| > N_{max} \end{array} \tag{6.19}$$

Il est possible de généraliser les inégalités (6.19) de la manière suivante :

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E[W]} y_{uv,r,t} \geq f(W) - 1 \qquad \begin{array}{l} \text{pour tout } W \subset V \text{ tel que} \\ G[W] \text{ est connexe} \end{array} \tag{6.20}$$

Nous notons les contraintes (6.20) les inégalités de coupes. Si $f(W) = 1$ alors la contrainte est une combinaison linéaire des contraintes triviales. Par contre, si $f(W) \geq 2$ alors il existe au moins $f(W) - 1$ arêtes entre les ensembles de la partition. En effet, il y aura au moins un sommet dans $f(W)$ ensembles de $\mathcal{V}$ et comme $G[W]$ est connexe on en déduit que ces ensembles sont aussi connectés.

Les inégalités (6.20) sont donc valides pour $P(G, k, q)$.

Proposition 10 *L'inégalité de coupe (6.20) associée à $W \subset V$ définit une facette de $P(G, k, q)$ seulement s'il n'existe pas un ensemble de sommets $W' \subset W$ tel que $f(W) = f(W')$.*

Preuve. S'il existe un ensemble de sommets $W' \subset W$ tel que $f(W) = f(W')$ alors la contrainte de coupes associée à W' domine la contrainte de coupes associée W. □

Dans la suite nous donnons d'autres familles d'inégalités valides utilisant des idées similaires.

6.5.3 Inégalités de cycles

Dans cette section, nous proposons des inégalités valides basées sur la structure de cycles. Nous commençons par montrer sur un exemple comment sont définies ces inégalités de cycles Considérons la Figure 6.4. La figure représente un graphe contenant $n = 7$ sommets. Supposons que l'on cherche à partitionner le graphe de la Figure 6.4 en $k = 3$ ensembles avec un coefficient d'équilibrage $q = 2$. Par la Proposition 9, on sait que toute partie $V_j, j \in \{1, ..., k\}$ de la partition $\mathcal{V}$ doit contenir au moins $N_{min} = 1$ sommets, et ne peut pas contenir plus que $N_{max} = 3$ sommets.

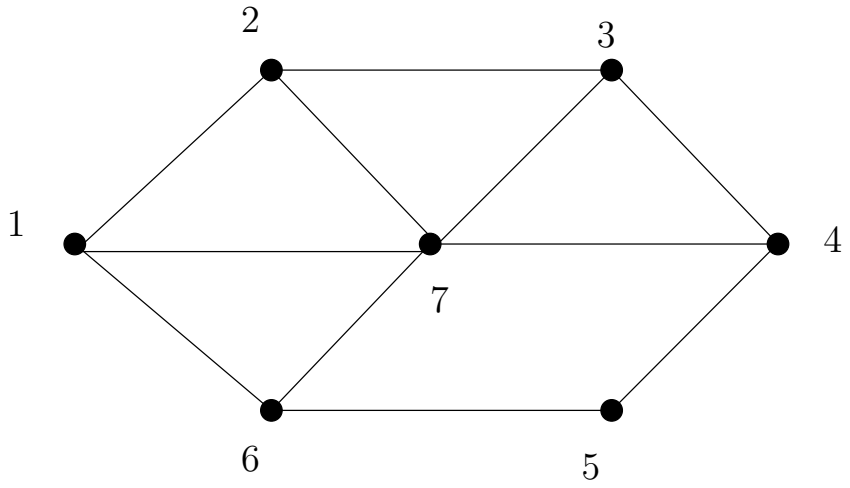

FIGURE 6.4 – Exempe des inégalités de Cycle

Considérons maintenant le cycle $C = (4, 5, 6, 7)$. Remarquons que ce cycle contient 4 sommets et nous avons donc $|C| > N_{max}$. Par conséquent, les sommets du cycle C

ne peuvent jamais appartenir à un seul ensemble de la partition $\mathcal{V}$. Ceci, peut être traduit par l'inégalité suivante

$$\sum_{t=1}^{k-1}\sum_{r=t+1}^{k}\sum_{u,v\in C} y_{uv,r,t} \geq 2.$$

Remarquons, en particulier, que la partie droite de cette inégalité correspond aux nombres minimums de sous-ensembles de la partition $\mathcal{V}$, nécessaires pour couvrir les sommets du cycle C.

Nous notons par $\mathcal{C}$ l'ensemble des cycles du graphe G.

Nous considérons les inégalités suivantes appelés *inégalités de cycles*

$$\sum_{t=1}^{k-1}\sum_{r=t+1}^{k}\sum_{u,v\in C} y_{uv,r,t} \geq f(C) \quad \text{pour tout } C \in \mathcal{C}. \tag{6.21}$$

Les inégalités (6.21) sont valides pour $P(G,k,q)$. En effet, les sommets du cycle C doivent appartenir à au moins $f(C)$ ensembles $V_C \subseteq \mathcal{V}$. Les ensembles V_C sont reliés par au moins un cycle, il est donc nécessaire de prendre au moins $f(C)$ arêtes.

Proposition 11 *L'inégalité de cycles* (6.21) *associée à un cycle C définit une facettes de $P(G,k,q)$ seulement si*
- il n'existe pas un cycle $C' \subset C$ tel que $f(C) = f(C')$,
- il n'existe pas deux cycles $C_1, C_2 \subset C$ tels que $C_1 \cap C_2 = \emptyset$ et $f(C_1)+f(C_2) > f(C)$.

Preuve. S'il existe un cycle $C' \subset C$ tel que $f(C) = f(C')$ alors la contrainte de cycles associée à C' domine la contrainte de cycles associée à C. De plus, S'il existe deux cycles $C_1, C_2 \subset C$ tels que $C_1 \cap C_2 = \emptyset$ et $f(C_1) + f(C_2) \geq f(C)$ alors la contrainte

$$\sum_{t=1}^{k-1}\sum_{r=t+1}^{k}\sum_{u,v\in C_1} y_{uv,r,t} + \sum_{t=1}^{k-1}\sum_{r=t+1}^{k}\sum_{u,v\in C_2} y_{uv,r,t} \geq f(C_1) + f(C_2), \tag{6.22}$$

obtenue en sommant les contraintes de cycles (6.21) associées à C_1 et C_2, domine la contrainte de cycles associée à C. □

6.5.4 Inégalités de coupe étoile

Nous appelons une structure d'étoile associée à $u \in V$ est un ensemble de sommets $S(u)$ composé du sommet $u \in U$ et de son voisinage (Voir l'exemple donné par

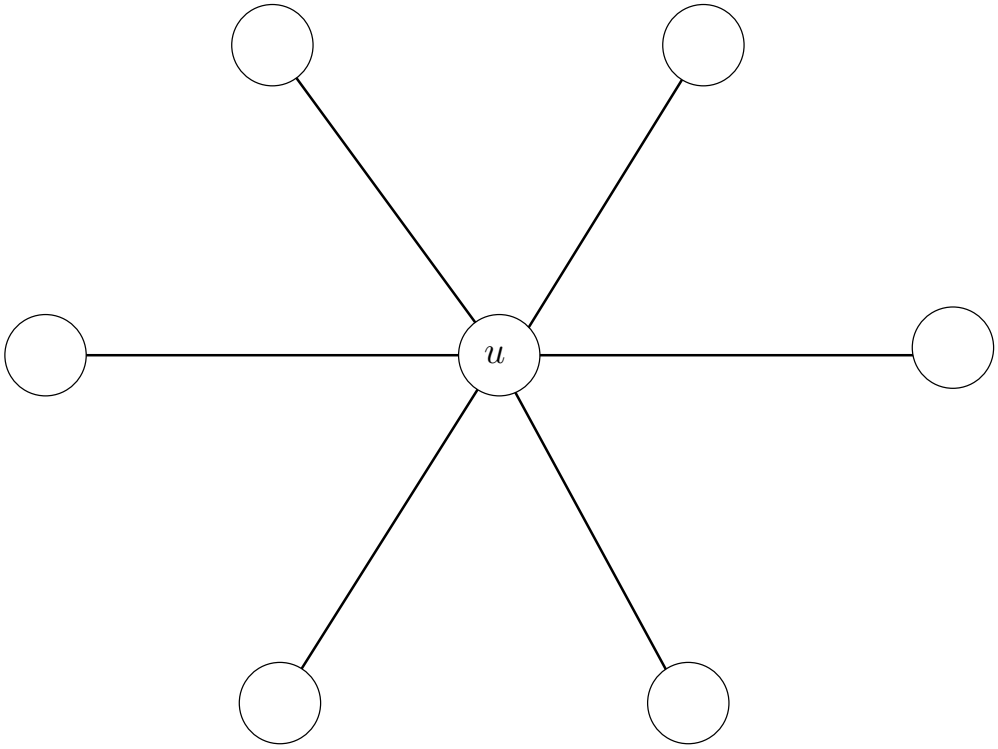

FIGURE 6.5 – Exemple des inégalités de coupe étoile

la Figure 6.5). Dans la suite, nous considérons que la structure d'étoile peut être uniquement associé à un sommet $u \in V$ si et seulement si $|\delta(u)| \geq 3$.

Considérons les contraintes suivantes

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in \delta(u)} y_{uv,r,t} \geq \left\lceil \frac{|\delta(u)|)}{N_{max}} \right\rceil \qquad \text{pour tout } u \in V \text{ tel que } |\delta(u)| > N_{max}. \tag{6.23}$$

Ces inégalités sont appelées *Inégalités de coupe étoile.*

Proposition 12 *Les inégalités de coupe étoile* (6.23) *sont valides pour* $P(G, k, q)$.

Preuve. La preuve utilise la procédure de Chvátal-Gomory.
Soit un sommet $u \in V$ tel que $|\delta(u)| > N_{max}$. Notons $V(u) = \{w_1, w_2, ..., w_{|\delta(u)|}\}$ l'ensemble des sommets adjacents à u et soit $W_0 \subseteq V(u)$ tel que $|W_0| = N_{max}$. Notons $W_1 = W_0 \cup \{u\}$. Soit $G'[W_1] = (V'_1, E'_1) = (W_1 \cap \delta(u), E(W_1 \cap \delta(u)))$. Puisque $|W_1| > N_{max}$ et $G'[W_1]$ est connexe, par la contrainte (6.20), nous avons

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E'_1} y_{uv,r,t} \geq 1.$$

Sachant que les sommets de W_0 sont quelconques, il existe $C^{N_{max}}_{|\delta(u)|}$ ensembles W_0 possibles et par conséquent nous pouvons écrire

$$\begin{aligned}
&\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E'_1} y_{uv,r,t} && \geq 1 \\
&\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E'_2} y_{uv,r,t} && \geq 1 \\
&. \\
&. \\
&. \\
&\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in E'_{N_{max}}} y_{uv,r,t} && \geq 1
\end{aligned}$$

En faisant la somme des inégalités précédentes, nous obtenons

$$C^{N_{max}-1}_{|\delta(u)|-1} \sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in \delta(u)} y_{uv,r,t} \geq C^{N_{max}}_{|\delta(u)|}.$$

Ceci implique que

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{uv \in \delta(u)} y_{uv,r,t} \geq \frac{C^{N_{max}}_{|\delta(u)|}}{C^{N_{max}-1}_{|\delta(u)|-1}},$$

ce qui donne après simplification

$$\sum_{t=1}^{k-1}\sum_{r=t+1}^{k}\sum_{uv\in\delta(u)} y_{uv,r,t} \geq \left\lceil \frac{|\delta(u)|)}{N_{max}} \right\rceil \qquad \text{pour tout } u \in V \text{ tel que } |\delta(u)| > N_{max}.$$

□

Il est clair que si $\frac{|\delta(u)|)}{N_{max}}$ est entier alors la contrainte (6.23) est une combinaison linéaire des contraintes (6.20).

Pour Illustrer l'intérêt des inégalités de coupe étoile (6.23), considérons l'exemple donner par la Figure 6.5 avec $n = 6$, $k = 2$ et $q = 2$. On a $N_{max} = 4$. Remarquons que le point fractionnaire $6/4$ où à chaque arête est associé la valeur $1/4$ satisfait les contraintes de coupe. Ce point est coupé par la contrainte de coupe étoile :

$$\sum_{uv\in\delta(u)} y_{uv,1,2} \geq 2$$

6.5.5 Inégalités de liens

Dans cette section, nous proposons de nouvelles inégalités valides faisant le lien entre les variables x et les variables y.

Proposition 13 *Les inégalités*

$$x_{u,r} - x_{v,r} \leq \sum_{t\in\{1,\ldots,k\}\setminus\{r\}} y_{uv,r,t}, \quad \textit{pour tout } r \in \{1,\ldots,k\} \textit{ et pour tout } uv \in E, \tag{6.24}$$

sont valides pour $P(G,k,q)$.

Preuve. Soit uv une arête quelconque de E et r un indice quelconque de $\{1,\ldots,k\}$. Nous allons montrer que

$$x_{u,r} - x_{v,r} \leq \sum_{t\in\{1,\ldots,k\}\setminus\{r\}} y_{uv,r,t}$$

Soit $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ une solution de PCMM. Si u et v appartiennent au même ensemble ou si u n'appartient pas à V_r, alors $x_{u,r} - x_{v,r} \leq 0$. Comme les variables $y_{uv,r,t}$ sont positives, il est clair que l'inégalité (6.24) est valide. Si u appartient à V_r et v n'appartient pas à V_r, alors uv appartient à la coupe $[V_r, V\setminus V_r]$. Par conséquent, $\sum_{t\in\{1,\ldots,k\}\setminus\{r\}} y_{uv,r,t} = 1$. Étant donné que $x_{u,r} - x_{v,r} = 1$, il est clair que l'inégalité (6.24) est valide. □

6.6 Algorithme de coupes et branchements

Dans cette section, nous présentons un algorithme de coupes et branchements pour résoudre le problème de partitionnement de graphe d'influence. Cet algorithme est basé sur les résultats polyédraux obtenus dans la section précédente. Les résultats expérimentaux obtenus à l'aide de cet algorithme montrent l'efficacité de la formulation introduite et l'intérêt des inégalités valides ajoutées.

6.6.1 Description de l'algorithme

L'algorithme de coupes et branchements que nous proposons calcule une solution optimale pour le PCMM formulé par le programme linéaire en nombres entiers (P). D'autre part, la formulation (P) contient une seule famille de contraintes en nombre très important à savoir la contrainte d'arête (6.4). Contrairement aux autres contraintes de la formulation (P), cette contrainte sera ajoutée dynamiquement au programme linéaire courant. Nous avons uniquement k contraintes (6.17) et (6.18) et $\frac{k\times(k-1)}{2}$ contraintes (6.3). Elles ne nécessitent donc pas d'être séparé et sont ajoutées au programme initial. Les contraintes de cycle et les contraintes de coupe ne sont utiles que quand elles satisfont respectivement les conditions, données par la proposition 10 et la proposition 11, nécessaires pour qu'elles définissent des facettes de $P(G,k,q)$. Ces contraintes sont en nombres exponentiels et sont ajoutées dynamiquement au programme courant. Les contraintes de coupe étoile (6.23) sont en nombre polynomial mais elles sont utiles uniquement quand $\frac{|\delta(u)|)}{N_{max}}$ n'est pas entier. Dans ce cas, ces contraintes sont ajoutées au programme initial.

Notre algorithme commence par résoudre le programme suivant noté (P') :

$$\min z$$

$$\sum_{uv \in A} y_{uv,r,t} w(uv) \leq z, \qquad \text{pour tout } t, r \in \{1, ..., k\} \text{ tel que } t \neq r, \tag{6.25}$$

$$\sum_{t=1}^{k} x_{u,t} = 1, \qquad \text{pour tout } u \in V, \tag{6.26}$$

$$\sum_{v \in V} x_{v,r} - \sum_{v \in V} x_{v,t} \leq q, \qquad \text{pour tout } r \neq t \in \{1, ..., k\} \tag{6.27}$$

$$\sum_{u \in V}^{k} x_{u,t} \geq N_{min} \qquad \text{pour tout } t \in \{1, ..., k\}, \tag{6.28}$$

$$\sum_{u \in V}^{k} x_{u,t} \leq N_{max}, \qquad \text{pour tout } t \in \{1, ..., k\}. \tag{6.29}$$

$$x_{u,t} \geq 0, \qquad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.30}$$

$$y_{uv,r,t} \leq 1, \qquad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\}, \tag{6.31}$$

$$y_{uv,r,t} \geq 0, \qquad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\} \tag{6.32}$$

La solution optimale $(x^*, y^*, z^*) \in \mathbb{R}^{nk+mk(k-1)+1}$ de cette relaxation est réalisable si et seulement si (x^*, y^*, z^*) est un vecteur qui satisfait les contraintes d'arêtes (6.4). En général (x^*, y^*, z^*) n'est pas réalisable. Dans ce cas, à chaque itération, l'algorithme génère des inégalités valides pour le problème mais pas pour la solution optimale courante. Les inégalités ainsi trouvées sont ajoutées au programme courant et le nouveau programme est résolu. Ce processus est repris tant qu'il y a des inégalités violées. Si la solution n'est pas encore réalisable l'algorithme procède alors à un branchement. L'algorithme 6 ci-dessous illustre les principales phases de notre algorithme de coupe et branchement.

On peut remarquer que toutes les inégalités sont globales, c'est-à-dire valides dans tout l'arbre de branchement. Par ailleurs, plusieurs contraintes peuvent être ajoutées à chaque itération.

6.6.2 Séparation des contraintes d'arête

Nous présentons un algorithme simple afin de séparer les contraintes d'arête. Le problème de séparation peut être défini de la manière suivante. Rappelons que (x^*, y^*, z^*) est la solution courante à séparer. Existe-t-il une arête $uv \in E$ tel que la contrainte d'arête associée à uv est violée par (x^*, y^*, z^*) ? La contrainte (6.4) associé à une arête uv de E est violée par la solution (x^*, y^*, z^*) si et seulement si,

Algorithme 6 : Algorithme de coupes et branchements

Data : Un graphe d'influence $G = (V, E)$, un entier k, un entier q et le vecteur $w \in \mathbb{R}^{+|E|}$

Result : Solution optimale de (P)

1 : PL $\leftarrow$ Programme linéaire initial

2 : Résoudre le programme linéaire (P').

Soit (x, y, z) la solution optimale de (P').

3 : **if** (x, y, z) *est réalisable pour* $P(G, k, q)$ **then**

| (x, y, z) est une solution optimale. STOP

end

4 : **if** *des contraintes violées par* (x, y, z) *sont trouvées* **then**

| Les ajouter à (P').

| Aller en 2.

end

5 : **else**

| Brancher sur une variable fractionnaire.

end

6 : Prendre la meilleure solution de tous les sous-problèmes.

il existe deux indices r, t de $\{1, \ldots, k\}$ tel que $r \neq t$ et

$$y^*_{uv,r,t} < x^*_{u,r} + x^*_{v,t} - 1$$

L'algorithme que nous proposons consiste à parcourir l'ensemble d'arêtes E et à regarder pour chaque arête uv de E s'il existe un couple d'indice r, t de $\{1, \ldots, k\}$ tel que $r \neq t$ et

$$y^*_{uv,r,t} < x^*_{u,r} + x^*_{v,t} - 1.$$

6.6.3 Séparation des contraintes de Cycle et Coupe

Nous présentons dans cette section une heuristique de séparation pour les contraintes de cycle et de coupe. En effet, les complexités des séparations exactes des contraintes de cycle et de coupe sont des questions ouvertes.

Nous allons proposer une heuristique permettant de générer à la fois des contraintes de cycle et de coupe. Cette heuristique consiste à chercher un ensemble de cycle $\mathcal{C}$ (resp. composante connexe $\mathcal{W}$) tels que leurs cardinalités soient supérieures à N_{max} et les contraintes de cycle (resp. de coupe) associées sont violées par la solution courante (x^*, y^*, z^*), c'est-à-dire :

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{u,v \in C} y^*_{uv,r,t} < f(C) \text{ pout tout } C \in \mathcal{C}$$

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{u,v \in W} y^*_{uv,r,t} < f(W) \text{ pout tout } W \in \mathcal{W}$$

Tout d'abord, nous commençons par associer pour tout arête uv de E, une variable y^*_{uv} tel que :

$$y^*_{uv} = \sum_{t=1}^{k-1} \sum_{r=t+1}^{k} y^*_{uv,r,t}$$

Il est clair que les inégalités de cycle (resp. coupe) sont violées par la solution courante si et seulement si :

$$\sum_{u,v \in C} y^*_{uv} < f(C)$$

$$\sum_{u,v \in W} y^*_{uv} < f(W) - 1$$

L'idée de l'heuristique est de parcourir le graphe en profondeur pour chaque sommet et de définir les cycles et les composantes connexes tel que les contraintes de cycle (6.21) et de coupe (6.20) associées soient violées par la solution courante.

Pour tout sommet u de V, notons par $N(u)$ l'ensemble de sommets adjacents à u, $N(u) = \{v \in V | uv \in E\}$.

Considérons le vecteur de poids $w \in \mathbb{R}_+^{|E|}$ associé aux arêtes du graphe G défini par $w(e) = y^*_e$. Pour chaque sommet u de V, nous effectuons un parcours de proche en proche pour construire le ou les cycles (resp. composantes connexes) qui contiennent le sommets u en ajoutant à chaque étape, l'arête e qui à la pondération la plus petite. Les principales étapes de cette heuristique sont données dans l'Algorithme 7 et peuvent être expliqué de la manière suivante :

1. On appelle v_c le sommet courant. Au début de notre algorithme, on initialise v_c par n'importe quel sommet de V.
2. Nous commençons par initialisé l'ensemble V^{NT} par l'ensemble $V \setminus \{v_c\}$. L'ensemble V^{NT} contient à chaque étape l'ensemble des sommets Non Traité (NT). Autrement dit, l'ensemble des sommets par lesquels on n'est pas encore passé lors de notre parcours de graphe G.
3. L'ensemble V^T contient à chaque étape l'ensemble des sommets Traité (T). Autrement dit, l'ensemble des sommets par lesquels on est déjà passé lors de notre parcours de graphe G. Au début, l'ensemble V^T est vide. Remarquons pendant tout le traitement, on a toujours $V = V^{NT} \cup V^T \cup \{v_c\}$

4. Á chaque itération, nous parcourons le graphe G en partant du sommet courant v_c comme suit
   - La première étape de parcours, soit u le sommet appartenant à $V(v_c) \cap V^{NT}$ tel que $w(v_c u)$ est le plus petit, $u = \{v \in V(v_c) \cap V^{NT} \mid w(v_c u) = min_{v \in V(v_c) \cap V^{NT}} w(v_c v)\}$. On pose $P = \{(v_c u)\}$. On ajoute le sommet u à l'ensemble V^T et on le supprime de l'ensemble V^{NT}, $V^T = V^T \cup \{u\}$ et $V^{NT} = V^{NT} \setminus \{u\}$.
   - Tant que $V(Terminal(P)) \cap V^{NT}$ n'est pas vide, on pose $w = Terminal(P)$ et on ajoute au chemin P l'arête wu tel que $u = \{v \in V(w) \cap V^{NT} \mid w(wv) = min_{v \in V(w) \cap V^{NT}} w(wv)\}$. On ajoute le sommet u à l'ensemble V^T et on le supprime de l'ensemble V^{NT}, $V^T = V^T \cup \{u\}$ et $V^{NT} = V^{NT} \setminus \{u\}$. Si $w = v_c$, le chemin (P) est un cycle. Nous l'ajoutons à l'ensemble des cycles $\mathcal{C}$, $\mathcal{C} = \mathcal{C} \cup \{P\}$
   - Si $V(Terminal(P)) \cap V^{NT}$ est vide, alors ajoute P à l'ensemble des chemins $\mathcal{P}$, $\mathcal{P} = \mathcal{P} \cup \{P\}$ et on ajoute le sommet v_c à l'ensemble V^T et on le supprime de l'ensemble V^{NT}, $V^T = V^T \cup \{v_c\}$ et $V^{NT} = V^{NT} \setminus \{v_c\}$.
5. Si V^{NT} n'est pas vide, on met dans v_c un sommet quelconque de V^{NT} et on recommence le parcours à partir de v_c en allant à l'étape 4.
6. Si V^{NT} est vide
   - Pour chaque cycle C_1 de $\mathcal{C}$ tel que C_1 ne satisfait pas la contrainte de cycle (6.21) et il n'existe pas un autre cycle $C_2 \in \mathcal{C}$ vérifiant $C_2 \subset C_1$ et $f(C_2) = f(C_1)$, on ajoute l'inégalité

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{u,v \in C_1} \bar{y}_{uv,r,t} < f(C_1)$$

     au programme (P')
   - Pour chaque chemin P_1 de $\mathcal{P}$ tel que P_1 ne satisfait pas la contrainte de coupe (6.20) et il n'existe pas un autre cycle $P_2 \in \mathcal{C}$ vérifiant $P_2 \subset P_1$ et $f(P_2) = f(P_1)$, on ajoute l'inégalité

$$\sum_{t=1}^{k-1} \sum_{r=t+1}^{k} \sum_{u,v \in P_1} \bar{y}_{uv,r,t} < f(P_1) - 1$$

     au programme (P')

6.6.4 Heuristique

Nous donnons dans cette partie une heuristique qui permet d'obtenir une borne supérieure à notre algorithme de coupes et branchements. Tout d'abord, remarquons

Algorithme 7 : Heuristique de séparation des contraintes de cycles et les contraintes de coupe

Data : Le graphe $G(V, E)$, un entier k, un entier q, le vecteur $(\bar{x}, \bar{y}, \bar{z})$ et le vecteur de pondération $w \in \mathbb{R}^{|E|}$

Result : Un ensemble de contrainte à ajouter au programme (P')

$V^{NT} \leftarrow V$ **forall the** $uv \in E$ **do**
  for $t = 1$; $t \leq k-1$;$t++$ **do**
    for $r = t+1$; $r \leq k$; $r++$ **do**
      $\bar{y}_{uv} \leftarrow \bar{y}_{uv} + \bar{y}_{uv,r,t}$;
    end
  end
end
forall the $v_c \in V^{NT}$ **do**
  $v \leftarrow v_c$;
  $P \leftarrow \emptyset$;
  $y^P \leftarrow 0$;
  while $V(w) \cap V^T \neq \emptyset$ **do**
    $y^{min} \leftarrow M$;
    forall the $u \in V(w) \cap V^T$ **do**
      if $y_{vu} < y^{min}$ **then**
        $y^{min} \leftarrow y_{vu}$;
        $u^{min} \leftarrow u$;
      end
    end
    $V^{NT} \leftarrow V^{NT} \setminus \{u^{min}\}$;
    $V^T \leftarrow V^T \cup \{u^{min}\}$;
    if $u^{min} = v_c$ *et* $y^P + y^{min} < f(P \cup \{vu^{min}\})$ **then**
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{P \cup \{vu^{min}\}\}$;
    else
      $P \leftarrow P \cup \{vu^{min}\}$;
      $y^P \leftarrow y^P + y^{min}$;
      $u_T \leftarrow u^{min}$;
    end
  end
  if $y^P + y^{min} < f(P \cup \{vu^{min}\}) - 1$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{P \cup \{vu^{min}\}\}$;
  end
end

que k joue un rôle très important dans la complexité du problème PCMM. En effet, plus que k est petit plus que le nombre de variables et le nombre de contraintes du programme (P) est petit et plus ce dernier est simple à résoudre. L'idée de notre heuristique est de résoudre plusieurs fois un problème de partitionnement en deux ensembles. Notons que pour qu'une partition quelconque de V soit une solution de notre problème, elle doit comporter k ensembles et ces ensembles doivent satisfaire la condition d'équilibre donnée par la contrainte (6.3) ou les contraintes (6.17) et (6.18).

Considérons le problème qui consiste à trouver une partition $\mathcal{V} = \{V_1, V_2\}$ de V tel que $N_{min} \leq V_1 \leq N_{max}$ et la coupe $[V_1, V_2]$ est minimale. Ce problème est équivalant au programme linéaire en nombres entiers suivant noté (P^1) :

$$\min z$$

$$\sum_{uv \in A} y_{uv,1,2} w(uv) \leq z, \qquad \text{pour tout} \tag{6.33}$$

$$\sum_{t=1}^{2} x_{u,t} = 1, \qquad \text{pour tout } u \in V, \tag{6.34}$$

$$\sum_{u \in V}^{k} x_{u,1} \geq N_{min} \qquad \text{pour tout } t \in \{1, ..., k\}, \tag{6.35}$$

$$\sum_{u \in V}^{k} x_{u,1} \leq N_{max}, \qquad \text{pour tout } t \in \{1, ..., k\}. \tag{6.36}$$

$$x_{u,1} + x_{v,2} \leq 1 + y_{uv,1,2}, \qquad \text{pour tout } uv \in E, \tag{6.37}$$

$$x_{u,t} \geq 0, \qquad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.38}$$

$$y_{uv,r,t} \leq 1, \qquad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\} \tag{6.39}$$

Considérons aussi le problème de graphe présenté au début de ce chapitre. Remarquons que ce problème est équivalent au programme linéaire en nombres entiers

suivant noté (P^2)

$$\min z$$

$$\sum_{uv \in A} y_{uv,1,2} w(uv) \leq z, \qquad \text{pour tout} \tag{6.40}$$

$$\sum_{t=1}^{2} x_{u,t} = 1, \qquad \text{pour tout } u \in V, \tag{6.41}$$

$$\sum_{v \in V} x_{v,r} - \sum_{v \in V} x_{v,t} \leq 1, \qquad \text{pour tout } r \neq t \in \{1, ..., k\} \tag{6.42}$$

$$x_{u,1} + x_{v,2} \leq 1 + y_{uv,1,2}, \qquad \text{pour tout } uv \in E, \tag{6.43}$$

$$x_{u,t} \geq 0, \qquad \text{pour tout } u \in V,\ t \in \{1, ..., k\}, \tag{6.44}$$

$$y_{uv,r,t} \leq 1, \qquad \text{pour tout } uv \in E,\ r \neq t \in \{1, ..., k\} \tag{6.45}$$

Soit (G, k, q) une instance quelconque du problème PCMM. Soit r le plus grand entier tel que $2^r \leq k, r = \max_{j \in \mathbb{N} | 2^j \leq k} j$. Soit $i = k - 2^r$. Nous commençons par lancer le programme (P^1) i fois. Á chaque itération, nous ajoutons l'ensemble le plus petit parmi les deux ensembles trouvés et nous partitionnons l'ensemble le plus grand. Á l'issu de cette opération, nous avons $\mathcal{V} = \{V_1, \ldots, V_i\}$ et chaque ensemble V_j de $\mathcal{V}$ satisfait les conditions d'équilibre données par les contraintes (6.17) et (6.18).

Ensuite, on partitionne l'ensemble $V \setminus \{V_1 \cup \ldots V_i\}$ en deux ensembles par le programme (P^2). Puis, toujours en utilisant le programme (P^2), nous partionnons chacun de ces deux ensembles en deux ensembles. Nous continuons cette procédure jusqu'à avoir des ensembles qui respecte la condition d'équilibre. Les ensembles ainsi obtenu complété par les ensembles $\{V_1 \cup \ldots V_i\}$ forment une partition de V qui est solution de PCMM.

Les principales étapes de cette heuristique sont données par l'algorithme 8

Cette heuristique ne se résout pas en temps polynomial. En effet, dans le cas $k = 2$ nous devons résoudre le problème de bissection. Cette procédure heuristique s'est avérée éfficace sur les instances réelles mais beaucoup moins performante en terme de temps de calcul sur les instances aléatoires.

6.7 Résultats expérimentaux

Dans un premier temps, nous donnons le contexte informatique de notre environnement de test. Puis, nous décrivons les différents types d'instances testés. Enfin, nous détaillons les résultats obtenus.

Algorithme 8 : Heuristique pour trouver une solution de départ

Data : Le graphe $G(V, E)$, un entier k, un entier q, et le vecteur de pondération $w \in \mathbb{R}^{|E|}$

Result : Une solution $\mathcal{V}$ du problème PCMM

$r \leftarrow 0$;
$V^1 \leftarrow V$;
while $2^r \leq k$ **do**
  $r \leftarrow r + 1$;
end
for $i = 1$; $i \leq k - 2^r$;$i++$ **do**
  Partionner V^2 en utilisant le programme (P^1) et ajouter l'ensemble le plus petit parmi les deux ensembles de la partition à $\mathcal{V}^1$ et affecter l'ensemble le plus grand à V^1.
end
$V^0 \leftarrow V \setminus \{V_1 \cup \dots V_i\}$ $\mathcal{V}^0 \leftarrow \{V^0\}$ **for** $i = 1$; $i \leq r$;$i++$ **do**
  forall the $V^{i-1} \in \mathcal{V}^{i-1}$ **do**
    En utilisant le programme (P^2), partionner l'ensemble V^{i-1} et ajouter les deux ensembles résultants à $\mathcal{V}^i$;
  end
end
Retourner les ensembles de $\mathcal{V}^r$ completés par les ensemble $\{V_1 \cup \dots V_i\}$

6.7.1 Contexte informatique

Avant de présenter les différents résultats expérimentaux que nous avons obtenus, nous donnons un bref descriptif des logiciels et du matériel informatique que nous avons utilisés.

L'algorithme de coupes et branchements décrit précédemment a été programmé en Java en utilisant la librairie Concert technology de CPLEX.

Notre algorithme de coupes et branchements a été testé sur machine core 2 duo 2,66 Ghz en limitant l'espace mémoire dédié à 4Go de mémoire vive, sous le système d'exploitation Windows. Nous avons fixé un temps maximum d'exécution à 5 heures.

6.7.2 Description des instances testées

6.7.2.1 Instances réelles

Les instances réelles sont issues de la base de données d'Altares et du graphe d'influence obtenu en combinant les informations des dirigeants et les informations des liens capitalistiques et en se basant sur les résultats de deux chapitres précédents. Bien que le graphe d'influence est de taille très importante, ce graphe n'est pas connexe mais composé de plusieurs composantes connexes. Chacunes de ces composantes connexes peuvent être considéré comme une instance indépendante. Pour partitionner le graphe d'influence, nous travaillons sur chacune de ces composantes connexes. La plus grande composante connexe du graphe d'influence, contient 836 sommets. Dans la suite de ce chapitre, nous désignons par une instance réelle, une composante connexe du graphe d'influence auquel nous avons associé des entiers k et q.

On appelle densité d'un graphe le ratio entre le nombre d'arêtes du graphe et le nombre total d'arêtes qu'il pourrait contenir.

Les graphes des instances réelles sont des graphes ayant une très faible densité, inférieure à 0,3%.

6.7.2.2 Instances aléatoires

Afin de compléter les testes de notre algorithme de coupes et branchements, obtenus avec les instances réelles, nous avons générées plusieurs instances aléatoires. Ces

instances aléatoires du PCMM sont définies par le triplet (G, k, q) où $G = (V, E)$ est un graphe connexe, k et q sont deux entiers.

La taille d'une instance aléatoire est définie par le nombre de sommets de V ainsi que sa densité du nombre d'arêtes. Les graphes aléatoirements générés contiennent entre 100 et 1000 sommets. À partir d'un nombre de sommets donné n, nous créons trois graphes de n sommets.

1. Instances aléatoires de type 1 : ces instances sont basées sur des graphe de densité 0.5%. Cette densité est proche de la densité des graphe réels.
2. Instances aléatoires de type 2 : ces instances sont basées sur des graphe de densité 2%. Cette densité est relativement grande pas rapport à la densité des graphe réels.
3. Instances aléatoires de type 3 : ces instances sont basées sur des graphe de densité 10%. Cette densité est très grande pas rapport à la densité des graphe réels.

À partir de chacun de ces trois graphes, nous créons trois instances pour le problème PCMM en variable le valeur de k en s'inspirant des instances réelles pour lequelles k est connu.

Pour chaque $G = (V, E)$ généré aléatoirement, nous créons un chemin qui passe par tous les sommets de V et donc assure la connexité de G. Puis nous ajoutons de manière aléatoire des arêtes afin d'obtenir la densité souhaitée. L'Algorithme 9 donne un aperçu global de notre générateur d'instances aléatoires.

6.7.3 Résultats obtenus

Nous avons testé 3 versions différentes de notre algorithme de coupes et branchements. La première version utilise les contraintes de base de la formulation et en ajoutant les contraintes d'arête. En effet, nous avons remarqué que la séparation des contraintes d'arête est plus éfficace que l'ajout des le début de la totalité de ces contraintes. Dans la deuxième version, nous avons ajouté les contraintes de borne et les contraintes de coupe étoile. Dans la dernière version nous avons simplement ajouté les contraintes de cycle.

Nos résultats expérimentaux sont reportés dans les tables suivantes. Les différentes colonnes de ces tables représentent :

Algorithme 9 : Générateur des instances aléatoires

Data : k_{faible},k_{moyen} et k_{grand}
Result : Liste des instances aléatoires du problème PCMM
for $n = 0; i \leq 1000$ **do**
  $i \leftarrow i + 100$;
  $d_{faible} \leftarrow \frac{0,5}{100} n(n-1)$;
  $d_{moyen} \leftarrow \frac{2}{100} n(n-1)$;
  $d_{grand} \leftarrow \frac{10}{100} n(n-1)$;
  $q \leftarrow \frac{5}{100} n$;
  for $i = 1; i \leq n-1; i++$ **do**
    $u \leftarrow i$; $u \leftarrow i+1$;
    $E_{faible} \leftarrow E_{faible} \cup \{(uv)\}$; $V_{faible} \leftarrow V_{faible} \cup \{u\}$;
    $E_{moyen} \leftarrow E_{moyen} \cup \{(uv)\}$; $V_{moyen} \leftarrow V_{moyen} \cup \{u\}$;
    $E_{grand} \leftarrow E_{grand} \cup \{(uv)\}$; $V_{grand} \leftarrow V_{grand} \cup \{u\}$;
  end
  $V_{faible} \leftarrow V_{faible} \cup \{n\}$; $V_{moyen} \leftarrow V_{moyen} \cup \{n\}$; $V_{grand} \leftarrow V_{grand} \cup \{n\}$;
  for $i = 1$; $i \leq d_{faible}$;$i++$ **do**
    $u \leftarrow$ un sommet tiré aléatoirement de V_{faible};
    $v \leftarrow$ un sommet tiré aléatoirement de $V_{faible} \setminus \{u\}$;
    $E_{faible} \leftarrow E_{faible} \cup \{(uv)\}$;
  end
  for $i = 1$; $i \leq d_{moyen}$;$i++$ **do**
    $u \leftarrow$ un sommet tiré aléatoirement de V_{moyen};
    $v \leftarrow$ un sommet tiré aléatoirement de $V_{moyen} \setminus \{u\}$;
    $E_{moyen} \leftarrow E_{moyen} \cup \{(uv)\}$;
  end
  for $i = 1$; $i \leq d_{grand}$;$i++$ **do**
    $u \leftarrow$ un sommet tiré aléatoirement de V_{grand};
    $v \leftarrow$ un sommet tiré aléatoirement de $V_{grand} \setminus \{u\}$;
    $E_{grand} \leftarrow E_{grand} \cup \{(uv)\}$;
  end
  $G_{d_{faible}} \leftarrow (V_{faible}, E_{faible})$;
  $G_{d_{moyen}} \leftarrow (V_{moyen}, E_{moyen})$;
  $G_{d_{grand}} \leftarrow (V_{grand}, E_{grand})$;
  Ajouter les trois instances $(G_{d_{faible}}, k_{faible}, q)$, $(G_{d_{faible}}, k_{moyen}, q)$ et $(G_{d_{faible}}, k_{grand}, q)$ à la liste des instances aléatoire;
  Ajouter les trois instances $(G_{d_{moyen}}, k_{faible}, q)$, $(G_{d_{moyen}}, k_{moyen}, q)$ et $(G_{d_{moyen}}, k_{grand}, q)$ à la liste des instances aléatoire;
  Ajouter les trois instances $(G_{d_{grand}}, k_{faible}, q)$, $(G_{d_{grand}}, k_{moyen}, q)$ et $(G_{d_{grand}}, k_{grand}, q)$ à la liste des instances aléatoire;
  $E_{faible} \leftarrow \emptyset$; $V_{faible} \leftarrow \emptyset$; $E_{moyen} \leftarrow \emptyset$; $V_{moyen} \leftarrow \emptyset$; $E_{grand} \leftarrow \emptyset$;
  $V_{grand} \leftarrow \emptyset$;
end

n	:	nombre de sommets de l'instance,
m	:	nombre d'arêtes de l'instance,
q	:	le paramètre d'équilibre,
k	:	le nombre d'ensembles de la partition recherché,
N_C	:	nombre de contraintes de coupes générées,
N_{Cy}	:	nombre de contraintes de cycles générées,
T_{sep}	:	temps de séparation des contraintes en secondes,
No	:	nombre de nœuds dans l'arbre de branchements,
o/p	:	le nombre de problèmes résolus à l'optimum sur le nombre d'instances testées,
CPU	:	temps de résolution en secondes.

Notre première série d'expérimentations concerne la première variante sur les instances aléatoires de type 1. Nous considérons des instances contenant entre 100 et 1000 sommets. Nous remarquons que nous arrivons à résoudre des instances ayant jusqu'à 400 sommets. En effet, la première variante n'arrive pas à résoudre des instances contenant plus de 500 sommets. Par contre nous avons testé notre formulation sous CPLEX en ajoutant toutes les contraintes d'arête dès le début. Dans cette version CPLEX n'arrive pas à résoudre des instances ayant plus de 150 sommets. En effet, dès qu'il commence à brancher nous obtenons un dépassement mémoire. Remarquons que la première variante permet de trouver la solution optimale des instances possèdant moins de 400 sommets en ajoutant peu de contraintes d'arête. Il génère en moyenne 20% des contraintes d'arête pour les instances possédant entre 100 et 300 sommets. Enfin, le temp de séparation des contraintes d'arête est très faible, moins de 4% en moyenne. Par contre, la taille de l'arbre de branchement est assez importante plus de 4000 noeuds en moyenne.

Notre deuxième série d'expérimentations concerne la deuxième variante sur les mêmes instances aléatoires que notre première série d'expérimentation. Nous remarquons que nous arrivons à résoudre toutes les instances en quasiment une heure. Ici aussi nous générons peu de contraintes d'arête parcontre le temps de séparation est beaucoup moins important que dans la première série d'expérimentations. Remarquons aussi que la taille de l'arbre de branchement est relativement faible, en moyenne une centaine de noeuds dans l'arbre de branchement. Notons que pour les instances possédant 1000 sommets avec $q = 12$ et $k = 15$ malgré le nombre important de contrainte d'arête ajouté nous obtenons la solution optimale en un peu plus d'une heure.

n	m	q	k	N_C	T_{sep}	No	o/p	CPU
100	49	3	3	58	8	521	5/5	206
100	49	3	4	117	34	987	5/5	866
100	49	3	6	294	76	1467	5/5	1901
200	199	4	3	238	37	998	5/5	1753
200	199	4	5	796	164	1287	5/5	5023
200	199	4	7	1671	406	2537	3/5	14161
300	448	5	3	537	147	3276	5/5	3683
300	448	5	5	1792	441	4678	4/5	16044
300	448	5	8	5017	720	5342	0/5	18000
400	798	6	3	957	252	3451	5/5	6323
400	798	6	6	4788	766	6539	0/5	18000
400	798	6	9	11491	774	5917	0/5	18000
500	1247	7	4	2992	800	6714	0/5	18000
500	1247	7	7	10474	758	5772	0/5	18000
500	1247	7	10	22446	784	5231	0/5	18000
600	1797	8	4	4312	774	6929	0/5	18000
600	1797	8	7	15094	786	6448	0/5	18000
600	1797	8	11	39534	766	6605	0/5	18000
700	2446	9	4	5870	772	5770	0/5	18000
700	2446	9	8	27395	782	5588	0/5	18000
700	2446	9	12	64574	752	5866	0/5	18000
800	3196	10	4	7670	796	6967	0/5	18000
800	3196	10	8	35795	750	6868	0/5	18000
800	3196	10	13	99715	771	5738	0/5	18000
900	4045	11	4	9708	767	5442	0/5	18000
900	4045	11	9	58248	761	6163	0/5	18000
900	4045	11	14	147238	765	5833	0/5	18000
1000	4995	12	5	19980	770	6797	0/5	18000
1000	4995	12	10	89910	778	6741	0/5	18000
1000	4995	12	15	209790	798	6235	0/5	18000

TABLE 6.1 – résultats de la première variante sur des instances aléatoires de type 1

n	m	q	k	N_C	T_{sep}	No	o/p	CPU
100	49	3	3	36	0	13	5/5	16
100	49	3	4	41	0	16	5/5	17
100	49	3	6	43	1	25	5/5	20
200	199	4	3	47	1	29	5/5	34
200	199	4	5	51	2	35	5/5	43
200	199	4	7	53	2	40	5/5	59
300	448	5	3	63	2	46	5/5	54
300	448	5	5	76	3	52	5/5	76
300	448	5	8	83	6	57	5/5	132
400	798	6	3	95	3	63	5/5	76
400	798	6	6	478	7	69	5/5	143
400	798	6	9	1149	13	74	5/5	261
500	1247	7	4	299	6	80	5/5	127
500	1247	7	7	1047	12	86	5/5	258
500	1247	7	10	2244	23	92	5/5	467
600	1797	8	4	431	8	97	5/5	165
600	1797	8	7	1509	17	103	5/5	354
600	1797	8	11	3953	39	109	5/5	781
700	2446	9	4	587	10	114	5/5	207
700	2446	9	8	2739	29	120	5/5	584
700	2446	9	12	6457	61	126	5/5	1235
800	3196	10	4	767	12	131	5/5	254
800	3196	10	8	3579	37	137	5/5	746
800	3196	10	13	9971	93	143	5/5	1865
900	4045	11	4	970	15	149	5/5	304
900	4045	11	9	5824	57	154	5/5	1154
900	4045	11	14	14723	135	160	5/5	2711
1000	4995	12	5	1998	24	166	5/5	499
1000	4995	12	10	8991	86	171	5/5	1723
1000	4995	12	15	20979	191	177	5/5	3821

TABLE 6.2 – résultats de la deuxième variante sur des instances aléatoires de type 1

La troisième série d'expérimentations concerne la deuxième variante sur les instances aléatoires de type 2. En effet, nous avons testé la troisième variante sur les instances aléatoires de type 2. Dans ce cas seul les instances possédant moins de 200 sommets ont pu être résolu en moins de 5h. De plus, la première variante sur les instances de densité moyenne ont pu résoudre uniquement 2 instances possédant 100 sommets. C'est pour cette raison que nous nous intéressons directement à cette troisième série d'expérimentations. Nous pouvons remarquer que la densité augmente le temps de calcul de notre algorithme. En effet, notre algorithme met en moyenne deux fois plus de temps pour trouver la solution optimale que pour les instances aléatoires de type 1. Nous remarquons aussi que le nombre de contraintes d'arête généré est beaucoup plus important mais que le temps de séparation reste acceptable.

Nous avons testé la troisième variante sur les instances de densité moyenne. Dans ce cas notre algorithme résout jusqu'à 700 sommets. Comme la deuxième variante reste plus efficace sur ces instances nous nous sommes intéressés aux instances aléatoires de type 2 qui ont une densité relativement imporantes par rapport aux instances réelles, dans notre quatrième série d'expérimentations nous avons souhaité testé cette variante sur les instances aléatoires de type 3. Nous pouvons remarquer que la densité engendre des difficultées de plus en plus importantes. En effet, Nous n'arrivons plus à résoudre toutes les instances. De plus, le nombre de contraintes d'arête devient très important.

La dernière série d'expérimentations sur les instances aléatoires concerne la troisième variante sur les instances aléatoires de type 3. Nous remarquons que cette variante devient de plus en plus efficace quand la densité augmente. En effet, plus le graphe est dense plus les contraintes de cycle joue un rôle important. De plus, on remarque que le temps de séparation globale reste faible bien que nous ajoutons énormément de contraintes. Le nombre de nœuds dans l'arbre de branchemennt est souvent plus petit que le nombre obtenu lors de la série précédente.

Enfin, nous avons testé les trois variantes sur les instances réelles d'Altares. Remarquons que la première et la troisième variantes sont très mauvaises et n'arrivent

n	m	q	k	N_C	T_{sep}	No	o/p	CPU
100	198	3	3	118	2	16	5/5	17
100	198	3	4	237	2	16	5/5	19
100	198	3	5	396	3	25	5/5	22
200	796	4	3	477	5	29	5/5	39
200	796	4	5	1592	8	35	5/5	61
200	796	4	6	2388	10	41	5/5	77
300	1794	5	3	1076	9	46	5/5	66
300	1794	5	5	3588	16	53	5/5	116
300	1794	5	7	7534	27	61	5/5	195
400	3192	6	3	1915	13	64	5/5	98
400	3192	6	6	9576	35	73	5/5	251
400	3192	6	8	17875	58	83	5/5	417
500	4990	7	4	5988	27	83	5/5	194
500	4990	7	7	20958	69	96	5/5	494
500	4990	7	9	35928	111	109	5/5	793
600	7188	8	4	8625	36	102	5/5	262
600	7188	8	7	30189	97	118	5/5	693
600	7188	8	10	64692	193	141	5/5	1383
700	9786	9	4	11743	47	120	5/5	339
700	9786	9	8	54801	168	147	5/5	1201
700	9786	9	11	107646	315	180	5/5	2257
800	12784	10	4	15340	59	139	5/5	426
800	12784	10	8	71590	217	173	5/5	1551
800	12784	10	12	168748	489	227	5/5	3494
900	16182	11	4	19418	73	158	0/5	523
900	16182	11	9	116510	345	212	5/5	2465
900	16182	11	13	252439	725	286	5/5	5183
1000	19980	12	5	39960	132	186	5/5	949
1000	19980	12	10	179820	524	261	5/5	3746
1000	19980	12	14	363636	1039	359	5/5	7422

TABLE 6.3 – résultats de la deuxième variante sur des instances aléatoires de type 2

n	m	q	k	N_C	T_{sep}	No	o/p	CPU
100	495	3	3	297	1	16	5/5	36
100	495	3	4	594	2	16	5/5	51
100	495	3	6	1485	3	25	5/5	95
200	1990	4	3	1194	4	29	5/5	102
200	1990	4	5	3980	9	36	5/5	241
200	1990	4	7	8358	18	44	5/5	460
300	4485	5	3	2691	7	47	5/5	198
300	4485	5	5	8970	20	56	5/5	512
300	4485	5	8	25116	52	70	5/5	1320
400	7980	6	3	4788	13	65	5/5	325
400	7980	6	6	23940	51	81	5/5	1282
400	7980	6	9	57456	118	103	5/5	2958
500	12475	7	4	14970	34	88	5/5	855
500	12475	7	7	52395	109	112	5/5	2726
500	12475	7	10	112275	228	148	5/5	5720
600	17970	8	4	21564	48	108	5/5	1206
600	17970	8	7	75474	156	141	5/5	3902
600	17970	8	11	197670	400	207	5/5	10012
700	24465	9	4	29358	64	129	5/5	1617
700	24465	9	8	137004	280	189	5/5	7000
700	24465	9	12	322938	651	287	2/5	16296
800	31960	10	4	38352	83	151	5/5	2089
800	31960	10	8	178976	364	227	5/5	9120
800	31960	10	13	498576	720	392	0/5	18000
900	40455	11	4	48546	104	173	5/5	2620
900	40455	11	9	291276	590	300	4/5	14756
900	40455	11	14	736281	720	528	0/5	18000
1000	49950	12	5	99900	208	216	5/5	5209
1000	49950	12	10	449550	720	396	0/5	18000
1000	49950	12	15	1048950	720	701	0/5	18000

TABLE 6.4 – résultats de la deuxième variante sur des instances aléatoires de type 3

n	m	q	k	N_C	N_{Cy}	T_{sep}	No	o/p	CPU
100	495	3	3	297	14	5	6	5/5	41
100	495	3	4	594	29	7	10	5/5	53
100	495	3	6	1485	74	12	15	5/5	89
200	1990	4	3	1194	59	14	23	5/5	107
200	1990	4	5	3980	199	30	35	5/5	219
200	1990	4	7	8358	417	55	34	5/5	394
300	4485	5	3	2691	134	22	42	5/5	164
300	4485	5	5	8970	448	52	46	5/5	374
300	4485	5	8	25116	1255	127	66	5/5	912
400	7980	6	3	4788	239	36	57	5/5	259
400	7980	6	6	23940	1197	125	77	5/5	898
400	7980	6	9	57456	2872	282	55	5/5	2015
500	12475	7	4	14970	748	74	4	5/5	534
500	12475	7	7	52395	2619	224	14	5/5	1604
500	12475	7	10	112275	5613	464	93	5/5	3315
600	17970	8	4	21564	1078	104	19	5/5	744
600	17970	8	7	75474	3773	248	87	5/5	1777
600	17970	8	11	197670	9883	628	182	5/5	4492
700	24465	9	4	29358	1467	107	37	5/5	769
700	24465	9	8	137004	6850	442	179	5/5	3161
700	24465	9	12	322938	16146	918	242	5/5	6563
800	31960	10	4	38352	1917	124	84	5/5	887
800	31960	10	8	178976	8948	517	61	5/5	3699
800	31960	10	13	498576	24928	1412	273	5/5	10091
900	40455	11	4	48546	2427	154	153	5/5	1105
900	40455	11	9	291276	14563	834	174	5/5	5960
900	40455	11	14	736281	36814	2080	418	4/5	14860
1000	49950	12	5	99900	4995	300	42	5/5	2148
1000	49950	12	10	449550	22477	1279	238	5/5	9141
1000	49950	12	15	1048950	52447	2520	556	0/5	18000

TABLE 6.5 – résultats de la troisième variante sur des instances aléatoires de type 3

n	m	q	k	$CPU_{variante1}$	$CPU_{variante2}$	$CPU_{variante3}$
120	141	3	4	1723	34	2341
215	284	4	4	3893	45	5624
423	690	8	5	8012	148	NR
534	960	11	5	NR	312	NR
672	1348	15	5	NR	489	NR
765	1641	18	6	NR	1067	NR
834	1876	20	6	NR	1354	NR

TABLE 6.6 – comparaison des trois variantes sur des instances réelles

pas à résoudre des instances ayant plus de 423 sommets. Par contre la deuxième variante est très efficace et permet de résoudre toutes les instances en moins de 30 minutes.

Nous avons proposé dans ce chapitre plusieurs variantes de notre algorithme de coupes et branchements. Nous avons observé que la deuxième variante s'avère très éfficace pour résoudre des instances de faible densité. Alors que la troisième variante était bien plus adaptée aux instances ayant une densité plus importante.

Conclusion

Dans cette thèse, nous avons étudié le problème de déduplications des dirigeants pour les bases de données d'Altares ainsi que les problèmes de calcul des liens indirects et d'identification des réseaux d'influence.

Dans un premier temps, nous nous sommes intéressés à la qualité de la base de données des dirigeants. Comme cette dernière est très volumineuse, nous avons commencé par séparer les dirigeants en plusieurs groupes, en mettant dans chaque groupe les enregistrements qui sont susceptibles de concerner la même personne. Ensuite, nous avons proposé une méthode de détection et suppression des doublons. Cette méthode est basée sur l'utilisation des algorithmes de calcul de similarité syntaxique et sur la normalisation des données. Notre méthode a permis de nettoyer les bases de données d'Altares en supprimant 4 millions de doublons et en normalisant les lieux de naissance de 99% des dirigeants. Les résultats de ce travail, sont aujourd'hui utilisés par Altares et ses clients. Un produit basé sur ces résultats est un cours de réalisation chez Altares et sera prochainement proposé sur le marché.

Nous nous sommes ensuite intéressés au problème de calcul des liens indirects et l'identification des têtes des groupes. Nous avons tout d'abord proposé une modélisation à l'aide d'un graphe. Puis, nous avons développé un algorithme basé sur les chemins afin déterminer la totalité des liens indirectes entre chaque couple d'entreprises. Ensuite, nous nous sommes basés sur le résultat de cet algorithme pour identifier les différentes têtes de groupe. Les résultats obtenus ont été validé par les experts métier d'Altares et sont aujourd'hui proposés aux clients d'Altares par les différents produits que cette dernière commercialise.

Enfin, dans le dernier chapitre, nous avons considéré le problème d'identification des réseaux d'influence. Ce problème se ramène au problème de la coupe Min-Max. Nous avons proposé une formulation en nombres entiers. Puis, nous avons étudié la structure faciale de cette formulation et donné les conditions nécessaires et suffisantes pour que les contraintes de base définissent des facettes. Nous avons proposé de nouvelles inégalités valides pour le problème et nous avons donné des conditions nécessaires pour qu'elles définissent des facettes. Nous avons implémenté un algorithme de coupes et branchements avec plusieurs variantes afin de définir les contraintes les plus efficaces. Ces différentes variantes ont été testées sur des instances réelles et des instances aléatoires. Nous avons montré que deux de nos variantes se sont révélées efficaces l'une sur les instances réelles et l'autre sur les instances aléatoires ayant une densité plus grande.

Ce travail de recherche ouvre plusieurs perspectives liées à la fois à l'exploration

théorique et à l'expérimentation numérique. Tout d'abord sur le problème de déduplication des dirigeants, il serait intéressant de proposer des règles plus avancées pour la comparaison des personnes qui dirigent les mêmes entreprises.

D'autre part pour améliorer le temps de résolution du problème de la coupe Min-Max, il serait intéressent de définir les conditions nécessaires et suffisantes pour que les contraintes valides que nous avons ajouté définissent des facettes. Ces conditions nous permettraient d'améliorer nos algorithmes de séparation et donc d'accélérer le temps de calcul. Une autre piste intéressante serait de proposer une nouvelle famille d'inégalités valides qui généralise les contraintes de coupe et de cycle. De plus, il serait intéressant de trouver des procédures permettant de briser la structure symétrique du problème soit en ajoutant des familles d'inégalités valides soit en proposant une procédure de branchement plus adaptée. Par ailleurs, il serait aussi intéressant de proposer et étudier une formulation étendue afin de développer algorithme de coupes, génération de colonnes et branchements. Enfin, il serait intéressant de considérer l'aspect temporel des réseaux d'influence afin de permettre d'analyser plus finement la composition et l'évolution de ces réseaux.

Annexe A

Annexe1

A.1 Étude de groupe

Le Tableau A.1 donne le nombre de groupe par taille de groupe.

En se basant sur les données de ce tableau, nous pouvons constater que nous disposons d'une partition de 78 ensembles. Par conséquence le nombre total de comparaisons donné par la formule

$$\sum_{i=1}^{k} \frac{n_i \times (n_i - 1)}{2}.$$

est égal à 4 046 955

A.2 Résultats obtenus

Nous donnons ici un exemple pour chacune de trois composantes de la méthode qui a été proposée pour la déduplication des dirigeants.

A.2.1 Normalisation des lieux de naissance

La Figure **??** donne un exemple de personnes qui ont été considérées comme différentes parce que leurs lieux de naissances (identiques) étaient écrites de deux façons différentes. La partie à gauche de cette image donne l'information qui était affichée par Altares avant la normalisation de lieux de naissance. La partie à droite donne l'information affichée après la normalisation.

Taille De Groupe	*Nombre De Groupe*	*Taille De Groupe*	*Nombre De Groupe*
1	4562169	41	3
2	1190507	43	2
3	320437	44	3
4	94877	45	2
5	33932	46	3
6	14195	47	1
7	6718	48	3
8	3378	50	4
9	1976	51	4
10	1165	53	3
11	688	55	1
12	432	56	2
13	288	57	3
14	222	59	1
15	149	60	1
16	92	62	2
17	80	64	2
18	73	70	1
19	49	71	1
20	37	73	1
21	24	74	2
22	23	75	1
23	21	77	1
24	16	81	1
25	9	82	1
26	14	88	1
27	4	91	1
28	8	94	1
29	3	97	1
30	11	98	1
32	10	100	1
33	6	108	1
34	4	116	1
35	3	129	1
36	1	136	1
37	3	138	1
38	1	140	1
39	3	173	1
40	1	189	1

TABLE A.1 – études des groupes (Nombre de groupe par taille de groupe)

A.2.2 Similarité Nom et Prénom

La Figure ?? donne un exemple de doublons des personnes identifiées grâce au calcul de la similarité syntaxique des noms. Cette figure montre à gauche l'information avant la déduplication des dirigeants et à droite l'information après la déduplication des dirigeants. Dans cet exemple, cette personne était créée deux fois dans le système. Une fois avec le nom *Tromboli* et une autre fois avec le nom *Trimboli*. Les autres informations sont identiques dans les deux cas. En regardant la position de la lettre *"i"* et la lettre *"o"* dans le clavier français (azerty), on voit bien qu'il s'agit d'une erreur de saisie. On considère que l'orthographe correcte est unique et doit être égale à celle qui nous a été envoyée le plus souvent par la source. Ici dans notre exemple, cette personne s'appelle *Tromboli*.

A.2.3 Assouplissement des règles

Dans l'exemple de la Figure ?? le dirigeant Pascale COCHAN était considéré comme dirigeant du « Cabinet Infirmier de la place » depuis 26/10/2006 avec une date de naissance et un lieu de naissance inconnus. Dans l'historique des dirigeants de ce cabinet, une autre Pascale COCHAN né le 02/05/1967 à Royan était présent. Les deux personnes (celle qui dirige actuellement et celle qui est dans l'historique des dirigeants) sont identiques. Notre fournisseur de données nous a envoyé les informations de cette personne une fois avec un lieu de naissance et une date de naissance connus et une autre fois sans information de date ni de lieu de naissance. La partie droite de la Figure donne le résultat obtenu après la déduplication des dirigeants.

A.3 Description de la bibliothèque JAVA SimMetrics

SimMetrics est une librairie open-source (licence GPLv2) de calcul de distance développée par l'université de Sheffields (UK) en 2004. Cette librairie existe dans les langages de programmation suivants : Java, C#, VB.NET. La version de librairie, que nous avons utilité, est la version Java 1.6.2_d07_02_07 mise à jour en février 2007. Les fonctions de distance prennent en entrée deux chaînes de caractères et retourne deux types de résultat qui sont la distance non normalisée ou brute et la distance normalisée qui correspond à la distance non normalisée rapportée à la longueur des chaines traitées. La librairie SimMetrics est utilisée dans de nombreux domaines dont par exemple la détection de fraude, l'analyse d'ADN, l'analyse d'image et déduplication de base de données.

A.4 AltaDir

Le principe d'AltaDir est d'être une solution qui met le dirigeant au centre et calcule un ensemble d'indicateurs permettant d'apprécier son efficacité. Mais aussi d'apprécier la solvabilité, le retard de paiement d'une entreprise sur laquelle nous n'avons pas beaucoup de données à partir de la solvabilité et du retard de paiement des autres entreprises dirigées par le même dirigeant.

Cette solution propose une carthographie qui permet de naviguer et de visualiser les relations entre les entreprises voir la Figure **??**. Elle permet aussi d'avoir un tableau de bord de dirigeant en donnant la liste de ces expériences, le score moyen des entreprises qu'il dirige, la moyenne de leurs retard de paiement comme dans la Figure **??**.

Bibliographie

[1] https ://sourceforge.net/projects/febrl/. (Cité en page 56.)

[2] http ://www.cs.cmu.edu/ wcohen/whirl/. (Cité en page 57.)

[3] http ://www.ics.uci.edu/ flamingo/. (Cité en page 57.)

[4] Ioannis P Androulakis and Wanpracha Art Chaovalitwongse. Mathematical programming for data mining., 2009. (Cité en page 30.)

[5] Kamel Aouiche. *Techniques de fouille de données pour l'?optimisation automatique des performances des entrepôts de données.* PhD thesis, Université Lumière ? Lyon 2, 2005. (Cité en page 27.)

[6] Kamel Aouiche, Jérôme Darmont, Omar Boussaid, et al. Sélection automatique d'index dans les entrepôts de données. *1er atelier Fouille de Données Complexes dans un processus d'extraction des connaissances, EGC 04*, pages 91–102, 2004. (Cité en page 27.)

[7] Michael Armbruster, Marzena Fügenschuh, Christoph Helmberg, and Alexander Martin. A comparative study of linear and semidefinite branch-and-cut methods for solving the minimum graph bisection problem. In *Integer Programming and Combinatorial Optimization*, pages 112–124. Springer, 2008. (Cité en page 117.)

[8] Hugo Barbalho, Isabel Rosseti, Simone L Martins, and Alexandre Plastino. A hybrid data mining grasp with path-relinking. *Computers & Operations Research*, 2012. (Cité en page 31.)

[9] Kristin P Bennett, Michael C Ferris, and Yannis E Ioannidis. *A genetic algorithm for database query optimization.* Computer Sciences Department, University of Wisconsin, Center for Parallel Optimization, 1991. (Cité en page 26.)

[10] Pavel Berkhin. A survey of clustering data mining techniques. In *Grouping multidimensional data*, pages 25–71. Springer, 2006. (Cité en page 29.)

[11] Indrajit Bhattacharya and Lise Getoor. Relational clustering for multi-type entity resolution. In *Proceedings of the 4th international workshop on Multi-relational mining*, pages 3–12. ACM, 2005. (Cité en page 69.)

[12] Charles-Edmond Bichot and Nicolas Durand. Air traffic control graph partitioning application. *Graph Partitioning*, pages 225–248. (Cité en page 114.)

[13] Charles-Edmond Bichot and Patrick Siarry. Partitionnement de graphe : optimisation et applications. *Hermes Science*, 2010. (Cité en pages 113 et 114.)

[14] Charles-Edmond Bichot and Patrick Siarry. *Graph partitioning.* John Wiley & Sons, 2013. (Cité en page 114.)

[15] Mikhail Bilenko, Raymond Mooney, William Cohen, Pradeep Ravikumar, and Stephen Fienberg. Adaptive name matching in information integration. *Intelligent Systems, IEEE*, 18(5) :16–23, 2003. (Cité en page 46.)

[16] Vladimir Boginski, Panos M Pardalos, and Alkis Vazacopoulos. Network-based models and algorithms in data mining and knowledge discovery. In *Handbook of Combinatorial Optimization*, pages 217–258. Springer, 2005. (Cité en page 28.)

[17] Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8) :1157–1166, 1997. (Cité en page 45.)

[18] Stanislav Busygin, Oleg Prokopyev, and Panos M Pardalos. Biclustering in data mining. *Computers & Operations Research*, 35(9) :2964–2987, 2008. (Cité en page 29.)

[19] Kevin M Campbell, Dennis Deck, and Antoinette Krupski. Record linkage software in the public domain : a comparison of link plus, the link king, and abasic'deterministic algorithm. *Health Informatics Journal*, 14(1) :5–15, 2008. (Cité en page 57.)

[20] Junghoo Cho, Narayanan Shivakumar, and Hector Garcia-Molina. Finding replicated web collections. In *ACM SIGMOD Record*, volume 29, pages 355–366. ACM, 2000. (Cité en page 45.)

[21] Sunil Choenni, Henk Blanken, and Thiel Chang. Index selection in relational databases. In *Computing and Information, 1993. Proceedings ICCI'93., Fifth International Conference on*, pages 491–496. IEEE, 1993. (Cité en page 27.)

[22] Peter Christen. *Data matching : concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer, 2012. (Cité en page 47.)

[23] Peter Christen. A survey of indexing techniques for scalable record linkage and deduplication. *Knowledge and Data Engineering, IEEE Transactions on*, 24(9) :1537–1555, 2012. (Cité en pages 47 et 56.)

[24] William W Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems (TOIS)*, 18(3) :288–321, 2000. (Cité en page 57.)

[25] Douglas Comer. The difficulty of optimum index selection. *ACM Transactions on Database Systems (TODS)*, 3(4) :440–445, 1978. (Cité en page 27.)

[26] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, New York, 1971. Association for Computing Machinery. (Cité en pages 6 et 7.)

[27] Carlos Coronel and Steven A Morris. *Database Systems : Design, Implementation, and Management [With Access Code]*. CengageBrain. com, 2012. (Cité en page 15.)

[28] Chris Ding, Xiaofeng He, Richard F Meraz, and Stephen R Holbrook. A unified representation of multiprotein complex data for modeling interaction networks. *Proteins : Structure, Function, and Bioinformatics*, 57(1) :99–108, 2004. (Cité en page 115.)

[29] Chris Ding, Tao Li, and Michael I Jordan. Nonnegative matrix factorization for combinatorial optimization : Spectral clustering, graph matching, and clique finding. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 183–192. IEEE, 2008. (Cité en page 30.)

[30] Chris HQ Ding, Xiaofeng He, Hongyuan Zha, Ming Gu, and Horst D Simon. A min-max cut algorithm for graph partitioning and data clustering. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 107–114. IEEE, 2001. (Cité en page 115.)

[31] Uwe Draisbach and Felix Naumann. Dude : The duplicate detection toolkit. In *Proceedings of the International Workshop on Quality in Databases (QDB)*, volume 100000, page 10000000, 2010. (Cité en page 57.)

[32] Orlando Durán, Nibaldo Rodriguez, and Luiz Airton Consalter. Collaborative particle swarm optimization with a data mining technique for manufacturing cell design. *Expert Systems with Applications*, 37(2) :1563–1567, 2010. (Cité en page 31.)

[33] J. Edmonds. Covers and packings in a family of sets. *Bull. American Mathematical Society 68*, pages 494–499, 1962. (Cité en page 6.)

[34] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards (B) 69*, pages 9–14, 1965. (Cité en page 7.)

[35] Mohamed G Elfeky, Vassilios S Verykios, and Ahmed K Elmagarmid. Tailor : A record linkage toolbox. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 17–28. IEEE, 2002. (Cité en page 56.)

[36] Ahmed K Elmagarmid, Panagiotis G Ipeirotis, and Vassilios S Verykios. Duplicate record detection : A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 19(1) :1–16, 2007. (Cité en pages 47 et 57.)

[37] Per-Olof Fjällström. Algorithms for graph partitioning : A survey. *Linköping electronic articles in computer and information science*, 3(10), 1998. (Cité en page 114.)

[38] Georges Gardarin. *Bases de données*. Editions Eyrolles, 2003. (Cité en pages 15, 19 et 20.)

[39] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-completeness*. Freeman, San Francisco, 1979. (Cité en page 6.)

[40] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimisation. *Combinatorica 1*, pages 70–89, 1981. (Cité en page 11.)

[41] Jean-Luc Hainaut. *Bases de données : Concepts, utilisation et développement.* Dunod, 2009. (Cité en page 15.)

[42] Oktie Hassanzadeh, Fei Chiang, Hyun Chul Lee, and Renée J Miller. Framework for evaluating clustering algorithms in duplicate detection. *Proceedings of the VLDB Endowment*, 2(1) :1282–1293, 2009. (Cité en page 69.)

[43] William H Inmon and Richard D Hackathorn. *Using the data warehouse.* Wiley-QED Publishing, 1994. (Cité en page 19.)

[44] Yannis E Ioannidis and Younkyung Kang. Randomized algorithms for optimizing large join queries. In *ACM Sigmod Record*, volume 19, pages 312–321. ACM, 1990. (Cité en page 26.)

[45] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering : a review. *ACM computing surveys (CSUR)*, 31(3) :264–323, 1999. (Cité en page 114.)

[46] Matthias Jarke and Jurgen Koch. Query optimization in database systems. *ACM Computing surveys (CsUR)*, 16(2) :111–152, 1984. (Cité en page 26.)

[47] Matthew A Jaro. *Unimatch : A record linkage system : Users manual.* Bureau of the Census, 1978. (Cité en page 51.)

[48] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *Bell system technical journal*, 49(2) :291–307, 1970. (Cité en page 114.)

[49] Willi Klösgen and Jan M Żytkow. *Handbook of data mining and knowledge discovery*, volume 271. Oxford University Press New York, 2002. (Cité en page 15.)

[50] Phokion G Kolaitis, Enela Pema, and Wang-Chiew Tan. Efficient querying of inconsistent databases with binary integer programming. *Proceedings of the VLDB Endowment*, 6(6) :397–408, 2013. (Cité en page 27.)

[51] Jozef Kratica, Ivana Ljubić, and Dušan Tošić. *A genetic algorithm for the index selection problem.* Springer, 2003. (Cité en page 27.)

[52] Chi-Hoon Lee, Osmar R Zaïane, Ho-Hyun Park, Jiayuan Huang, and Russell Greiner. Clustering high dimensional data : A graph-based relaxed optimization approach. *Information Sciences*, 178(23) :4501–4511, 2008. (Cité en page 29.)

[53] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707, 1966. (Cité en page 52.)

[54] Xavier Llora, Josep M Garrell, et al. Knowledge-independent data mining with fine-grained parallel evolutionary algorithms. In *Proceedings of the Third Genetic and Evolutionary Computation Conference*, pages 461–468. Citeseer, 2001. (Cité en page 29.)

[55] Shen Lu and Richard S Segall. Linkage in medical records and bioinformatics data. *International Journal of Information and Decision Sciences*, 5(2) :169–187, 2013. (Cité en page 46.)

[56] A. R. Mahjoub. Approches Polyédrales. In V. Th. Paschos, editor, *Optimisation combinatoire : Concepts fondamentaux*. chapitre 9. Hermès, Paris, 2005. (Cité en page 7.)

[57] Olvi L Mangasarian. Mathematical programming in data mining. *Data mining and knowledge discovery*, 1(2) :183–201, 1997. (Cité en page 30.)

[58] Allison B McCoy, Adam Wright, Michael G Kahn, Jason S Shapiro, Elmer Victor Bernstam, and Dean F Sittig. Matching identifiers in electronic health records : implications for duplicate records and patient safety. *BMJ quality & safety*, 22(3) :219–224, 2013. (Cité en page 46.)

[59] Ruslan Mitkov. *Anaphora resolution*, volume 134. Longman London, 2002. (Cité en page 45.)

[60] Zied Moalla, Lina F Soualmia, Élise Prieur-Gaston, Thierry Lecroq, and Stéfan J Darmoni. Spell-checking queries by combining levenshtein and stoilos distances. *Proceedings of Network Tools and Applications in Biology*, 2011. (Cité en page 75.)

[61] Hamid Mohamadi, Jafar Habibi, Mohammad Saniee Abadeh, and Hamid Saadi. Data mining with a simulated annealing based fuzzy classification system. *Pattern Recognition*, 41(5) :1824–1833, 2008. (Cité en page 29.)

[62] Cristian Molinaro and Sergio Greco. Polynomial time queries over inconsistent databases with functional dependencies and foreign keys. *Data & Knowledge Engineering*, 69(7) :709–722, 2010. (Cité en page 27.)

[63] Alvaro E Monge, Charles Elkan, et al. The field matching problem : Algorithms and applications. In *KDD*, pages 267–270, 1996. (Cité en page 53.)

[64] Alvaro E Monge and Charles P Elkan. Efficient domain-independent detection of approximately duplicate database records. In *Proc. of the ACM-SIGMOD Workshop on Research Issues in on Knowledge Discovery and Data Mining*, 1997. (Cité en page 54.)

[65] Hiroyuki Mori. State-of-the-art overview on data mining in power systems. In *Power Systems Conference and Exposition, 2006. PSCE'06. 2006 IEEE PES*, pages 33–34. IEEE, 2006. (Cité en page 31.)

[66] Amir Nakib, Laurent Najman, Hugues Talbot, and Patrick Siarry. Application of graph partitioning to image segmentation. *Graph Partitioning*, pages 249–274. (Cité en page 114.)

[67] Mariá CV Nascimento and Andre CPLF De Carvalho. Spectral methods for graph clustering–a survey. *European Journal of Operational Research*, 211(2) :221–231, 2011. (Cité en page 114.)

[68] Felix Naumann and Melanie Herschel. An introduction to duplicate detection. *Synthesis Lectures on Data Management*, 2(1) :1–87, 2010. (Cité en page 47.)

[69] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3) :443–453, 1970. (Cité en page 53.)

[70] HB Newcombe, JM Kennedy, SJ Axford, and AP James. *Automatic linkage of vital records*. 1959. (Cité en page 46.)

[71] Howard B Newcombe. *Handbook of record linkage : methods for health and statistical studies, administration, and business*. Oxford University Press, Inc., 1988. (Cité en page 46.)

[72] Howard B Newcombe and James M Kennedy. Record linkage : making maximum use of the discriminating power of identifying information. *Communications of the ACM*, 5(11) :563–566, 1962. (Cité en page 46.)

[73] Sigurdur Olafsson, Xiaonan Li, and Shuning Wu. Operations research and data mining. *European Journal of Operational Research*, 187(3) :1429–1448, 2008. (Cité en page 28.)

[74] Mustapha Oughdi, Sid Lamrous, and Alexandre Caminada. Partitioning mobile networks into tariff zones. *Graph Partitioning*, pages 201–223. (Cité en page 114.)

[75] Stratos Papadomanolakis and Anastassia Ailamaki. An integer linear programming approach to database design. In *Data Engineering Workshop, 2007 IEEE 23rd International Conference on*, pages 442–449. IEEE, 2007. (Cité en page 28.)

[76] U.S. Patent R.C. Russell Index, April 1918. (Cité en page 47.)

[77] U.S. Patent R.C. Russell Index, November 1922. (Cité en page 47.)

[78] Tae-Wan Ryu and Christoph F Eick. A database clustering methodology and tool. *Information Sciences*, 171(1) :29–59, 2005. (Cité en page 29.)

[79] Haroldo G Santos, Luiz Satoru Ochi, Euler Horta Marinho, and Lúcia Maria de A Drummond. Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem. *Neurocomputing*, 70(1) :70–77, 2006. (Cité en page 31.)

[80] Sunita Sarawagi and Anuradha Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–278. ACM, 2002. (Cité en page 46.)

[81] Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1) :27–64, 2007. (Cité en page 114.)

[82] A. Schrijver. *Combinatorial Optimization : Polyhedra and Efficiency*. Algorithms and Combinatorics 24. Springer-Verlag, 2003. (Cité en page 7.)

[83] DY Sha and C-H Liu. Using data mining for due date assignment in a dynamic job shop environment. *The International Journal of Advanced Manufacturing Technology*, 25(11-12) :1164–1174, 2005. (Cité en page 31.)

[84] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1) :195–197, 1981. (Cité en page 53.)

[85] Michael Stillger and Myra Spiliopoulou. Genetic programming in database query optimization. In *Proceedings of the First Annual Conference on Genetic Programming*, pages 388–393. MIT Press, 1996. (Cité en page 26.)

[86] Arun Swami. Optimization of large join queries : combining heuristics and combinatorial techniques. In *ACM SIGMOD Record*, volume 18, pages 367–376. ACM, 1989. (Cité en page 26.)

[87] Arun Swami and Anoop Gupta. *Optimization of large join queries*, volume 17. ACM, 1988. (Cité en page 26.)

[88] Benjamin J Tepping. A model for optimum linkage of records. *Journal of the American Statistical Association*, 63(324) :1321–1332, 1968. (Cité en page 46.)

[89] Cheng-Fa Tsai, Chun-Wei Tsai, Han-Chang Wu, and Tzer Yang. Acodf : a novel data clustering approach for data mining in large databases. *Journal of Systems and Software*, 73(1) :133–145, 2004. (Cité en page 29.)

[90] Esko Ukkonen. Approximate string-matching with< i> q</i>-grams and maximal matches. *Theoretical computer science*, 92(1) :191–211, 1992. (Cité en page 55.)

[91] Julian R. Ullmann. A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words. *The Computer Journal*, 20(2) :141–147, 1977. (Cité en page 55.)

[92] JR Wang and Stuart E Madnick. The inter-database instance identification problem in integrating autonomous systems. In *Data Engineering, 1989. Proceedings. Fifth International Conference on*, pages 46–55. IEEE, 1989. (Cité en page 46.)

[93] Jin-Tai Yan and Pei-Yung Hsiao. A new fuzzy-clustering-based approach for two-way circuit partitioning. In *VLSI Design, 1995., Proceedings of the 8th International Conference on*, pages 359–364. IEEE, 1995. (Cité en page 114.)

[94] William E Yancey. Bigmatch : A program for extracting probable matches from a large file for record linkage. *Computing*, 1 :1–8, 2002. (Cité en page 57.)

[95] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch : an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996. (Cité en page 29.)