Gestion de la collaboration et compétition dans le crowdsourcing : une approche avec prise en compte de fuites de données via les réseaux sociaux

Iheb Ben Amor

THESE DE DOCTORAT DE
L'UNIVERSITE PARIS DESCARTES

Spécialité

INFORMATIQUE
École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

M. **Iheb BEN AMOR**

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PARIS DESCARTES

Sujet de la thèse :

Gestion de la collaboration et compétition dans le crowdsourcing : une approche avec prise en compte de fuites de données via les réseaux sociaux

Soutenue le 27-11-2014

Devant le jury composé de :

Directeur de thèse : Professeur : Mme. Salima BENBERNOU (Université Paris Descartes)
Co-Directeur de thèse : Docteur : M. Mourad OUZIRI (Université Paris Descartes)

Rapporteurs :
Professeur : M. Lakhdar SAIS (Université d'Artois)
Docteur : M. Mustapha LEBBAH (Université Paris 13)

Examinateurs
Mme Sihem Amer Yehia (Directrice de recherche CNRS)

Gestion de la collaboration et compétition dans le crowdsourcing : une approche avec prise en compte de fuites de données via les réseaux sociaux

Résumé

Le crowdsourcing est une pratique permettant aux entreprises de faire appel à l'intelligence humaine à grande échelle afin d'apporter des solutions à des problématiques qu'elles souhaitent externaliser.

Les problématiques externalisées sont de plus en plus complexes et ne peuvent être résolues individuellement. Nous proposons dans cette thèse une approche appelée *SocialCrowd*, contribuant à améliorer la qualité des résultats de crowdsourcing. Elle consiste à faire collaborer les participants afin d'unir leur capacité de résolution et apporter des solutions aux problèmes externalisés de plus en plus complexes. Les groupes collaboratifs sont mis en compétition, via des rétributions attrayantes, afin d'obtenir de meilleures résolutions. Par ailleurs, il est nécessaire de protéger les données privées des groupes en compétition.

Nous utilisons les réseaux sociaux comme support de fuite de données. Nous proposons une approche basée sur l'algorithme Dijkstra pour estimer la probabilité de propagation de données privées d'un membre sur le réseau social. Ce calcul est complexe étant donné la taille des réseaux sociaux. Une parallélisation du calcul est proposée suivant le modèle MapReduce. Un algorithme de classification basé sur le calcul de propagation dans les réseaux sociaux est proposé permettant de regrouper les participants en groupes collaboratifs et compétitifs tout en minimisant les fuites de données d'un groupe vers l'autre.

Comme ce problème de classification est d'une complexité combinatoire, nous avons proposé un algorithme de classification basé sur les algorithmes d'optimisation combinatoires tels que le recuit simulé et les algorithmes génétiques. Etant donnée le nombre important de solutions possible, une approche basée sur le modèle du Soft Constraint Satisfaction Problem (SCSP) est proposée pour classer les différentes solutions.

Mots-clés : crowdsourcing, réseaux sociaux, fuite de données, classification, optimisation combinatoire.

Managing collaboration and competition in crowdsourcing: Approach that takes into account data leakage via social networks

Abstract

Crowdsourcing is the practice of allowing companies to use human intelligence scale to provide solutions to issues they want to outsource.

Outsourced issues are increasingly complex and cannot be resolved individually. We propose in this thesis an approach called SocialCrowd, helping to improve the quality of the results of crowdsourcing. It compromise to collaborate participants to unite solving ability and provide solutions to outsourced problems more and more complex. Collaborative groups are put in competition through attractive remuneration, in order to obtain better resolution. Furthermore, it is necessary to protect the private information of competing groups.

We use social media as a support for data leakage. We propose an approach based on Dijkstra algorithm to estimate the propagation probability of private data member in the social network. Given the size of social networks, this computation is complex. Parallelization of computing is proposed according to the MapReduce model. A classification algorithm based on the calculation of propagations in social networks is proposed for grouping participants in collaborative and competitive groups while minimizing data leaks from one group to another.

As this classification problem is a combinatorial complexity, we proposed a classification algorithm based on combinatorial optimization algorithms such as simulated annealing and genetic algorithms. Given the large number of feasible solutions, an approach based on the model of Soft Constraint Satisfaction Problem (SCSP) is proposed to classify the different solutions.

Keywords: crowdsourcing, social networking, data leakage, classification, combinatorial optimization.

REMERCIEMENTS

Au terme de ce travail, nous tenons à exprimer nos vifs remerciements à :

Mme Salima BENBERNOU

Pour sa disponibilité, sa patience, son assistance ainsi que pour tous les conseils judicieux qu'elle nous a prodigués tout au long de cette thèse.

Qu'elle en soit chaleureusement remerciée pour la qualité de son encadrement ainsi que son constant soutien.

Mr Mourad OUZIRI

Pour ses conseils et pour l'attention qu'il a bien apporté à notre travail à travers son encadrement.

Pour son effort épargné qui a guidé nos pas et enrichi nos travaux tout au long de cette thèse.

Je vous remercie d'avoir cru en mes capacités, pour le temps et la patience que vous m'avez accordés tout au long de ces années en me fournissant d'excellentes conditions logistiques.

Je garderai dans mon cœur votre générosité, votre compréhension et votre efficacité. Pour tout ce que vous m'avez donné, je vous remercie très sincèrement.

Nous tenons à exprimer nos vifs remerciements et notre grande gratitude pour tous ceux qui nous ont aidés de près ou de loin à réaliser ce modeste travail.

Un remerciement distingué aux membres de jury, pour nous avoir accordé cet honneur. Que ce travail soit l'expression de notre grande reconnaissance et notre respectueuse admiration.

DEDICACE

Je dédie cette thèse en signe de reconnaissance

A celui qui a lutté et sacrifié pour m'offrir les conditions propices à ma réussite : mon très cher père :

Abdeljabar BEN AMOR

A celle qui m'a témoigné de tendresse d'affection et qui a constitué la première école de mon existence : ma très précieuse, chaleureuse et aimable mère :

Hedia ELLEUCH

A celle qui m'a encouragé acceptant tout ce temps soustrait à ma présence auprès d'elle. Affable, honorable, aimable, elle représente pour moi le symbole de la bonté par excellence, la source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager: Ma très belle femme et merveilleuse copine

Ines MAKHLOUFI

A tous les trois, pour leur grande affection et leur sacrifice consentis tout au long de mon parcours.

A ma très chère sœur ***Olfa et ma nièce Khadija*** avec mes souhaits de bonheur et de réussite.

A mon très cher frère ***Oussama, son épouse Fatma, mon neveu Mohamed Yassine et ma nièce Yessmine*** avec mes souhaits de bonheur et de réussite.

A toute la famille **Ben amor**, **Elleuch et Makhloufi**

A mes précieux amis,

A mes chers encadreurs,

ET

A tous ceux qui m'aiment.

Iheb

Table des matières

Chapitre 1

Introduction

Le crowdsourcing est un terme générique pour une variété d'approches qui exploitent le potentiel des grandes foules de gens en émettant des appels à contribution pour des tâches particulières. Bien que les approches de crowdsourcing puissent prendre de nombreuses formes différentes, aujourd'hui il est effectué de plus en plus via le Web, qui permet l'interaction avec une pluralité de contributeurs du monde entier. Plusieurs approches de crowdsourcing incluent les plates-formes sur le Web. Par exemple, pour la résolution de problèmes on cite InnoCentive, pour l'agrégation des connaissances on a Wikipedia et TripAdvisor, pour le traitement de données on note ReCaptcha, pour la conception on identifie iStockphoto et Threadless, et d'autres contenus générés par les utilisateurs comme YouTube et App Store.

Tout en utilisant la technologie de l'information en tant que facilitateur, les organisations de crowdsourcing sont capables de traiter un grand nombre de tâches. Un processus de crowdsourcing comprend un certain nombre d'activités qui impliquent des ressources au-delà des frontières organisationnelles.

La recherche sur le crowdsourcing provient d'une variété de domaines tels que l'informatique, la gestion, la psychologie, et de nombreux autres domaines qui ont découvert le crowdsourcing comme une approche utile.

La collaboration dans un processus de crowdsourcing est un point essentiel à la bonne résolution des problèmes externalisés par l'entreprise. De nos jours, les réseaux sociaux constituent des plates-formes en ligne permettant à des membres d'interagir et de partager des informations. Ces réseaux sociaux permettent au crowdsourcing une meilleure résolution des problèmes existants.

Les réseaux sociaux, y compris Friendster.com, Tagged.com, Xanga.com, LiveJournal, Myspace, Facebook et LinkedIn, se sont développés au cours des dernières années. Ces réseaux ont réussis à attirer l'attention des utilisateurs. Selon ComScore Media Metrix, il y a plus d'utilisateurs qui visitent Myspace que Yahoo, MSN ou le site web de jeux vidéo Electronic Arts[81, 62]. Cependant, Facebook reste le réseau le plus visité selon les statistiques présentées par " complete.com " pour l'année 2014, avec 1,310,000,000 d'utilisateurs actifs par mois suivi par MySpace avec 810,153,536 de visites par mois. Ces chiffres sont en forte progression avec le développement rapide des réseaux sociaux [32, 66].

Ces derniers permettent des échanges à grande échelle et d'être plus social. Les utilisateurs peuvent définir un profil et le personnaliser comme ils souhaitent. Ils peuvent aussi s'engager avec d'autres utilisateurs pour des buts divers, comme les affaires professionnelles, le divertissement et le partage des connaissances. Le succès commercial de ces réseaux sociaux dépend essentiellement du nombre d'utilisateur. Ils attirent et encouragent ces membres pour ajouter plus d'utilisateurs à leurs réseaux et partager des données avec d'autres utilisateurs dans le même réseau.

Cependant, les utilisateurs ne sont pas souvent conscients du type d'auditoire ayant accès à leurs données et également au sens d'intimité partagée avec des amis numériques. Ceci, mène souvent aux révélations qui ne peuvent pas être appropriées à un forum public. Une telle disponibilité de données ouvertes et exposées provoque des risques sur la vie privée, qui évoluent jour après jour [64, 115, 62], d'où, une menace significative de la vie privée qui augmente avec la croissance du contenu médiatique posté par les utilisateurs dans leurs profils personnels.

A ce propos, des images numériques fournies par des utilisateurs, constituent une partie intégrale et extrêmement populaire des profils sur les réseaux sociaux. A titre d'exemple, à partir de 2006 Facebook héberge 70 billion de pièces partagées par les utilisateurs, [66].

Ces contenus partagés sont liées explicitement aux profils individuels ou implicitement (par la répétition), identifiant ainsi le détenteur de profil [62]. De ce fait, de tels contenus seront à la disposition d'autres utilisateurs du même réseau social, qui peuvent voir et ajouter des commentaires en utilisant des techniques d'annotations. Ils peuvent aussi ajouter des liens hypertextes pour identifier les utilisateurs qui apparaissent sur les éléments partagés.

En général, les données, sont contrôlées et gérées par les utilisateurs eux même qui ne sont pas les parties prenantes réelles ou uniques. Ainsi, de sérieuses préoccupations de la vie privée augmentent. Ces parties prenantes de ces données peuvent être inconscientes du fait que leurs données (ou les données qui sont rapprochées d'eux) sont gérées par d'autres. Même si deux utilisateurs se connaissent, leur relation sociale n'implique pas souvent qu'ils ont les mêmes préférences de la vie privée. Le nombre moyen des utilisateurs amis de Myspace est d'environ 115, ce qui indique que la relation d'amitié est généralisée pour couvrir un grand niveau d'intimité [23].

En effet, les utilisateurs qui partagent le même contenu peuvent avoir de différentes préférences de la vie privée et par conséquent, ils peuvent être en conflit sur un certain contenu de données qu'ils partagent.

L'accès à des informations personnelles détaillées, serait idéal pour lancer des attaques ciblées, souvent mentionnées comme phishing [66, 23]. En outre, les adresses électroniques rassemblées et les informations personnelles seraient inestimables par des spammeurs.

Dans cette thèse, on traite les problématiques de résolution des tâches dans un environnement de crowdsourcing. On s'occupe du calcul de la divulgation d'informations entre les différentes intelligence humaine via les réseaux sociaux. On propose également le regroupement en équipes sans fuite de données des différents membres inscrits pour l'appel au crowdsourcing. Enfin on classe les différentes solutions de groupement possibles par rapport aux préférences de demandeur.

1.1 Exemples de motivations

De nos jours, les réseaux sociaux évoluent fortement. En effet, des milliers d'enregistrements sont effectué en une minute. Compte tenu du volume important d'interaction et d'échange de données entre les utilisateurs, il est difficile d'analyser les traces.

De plus, les interactions entre les utilisateurs d'un même réseau social sont dociles. Les systèmes d'information sont en mesure d'analyser automatiquement les traces du passé pour déterminer les structures du réseau et les interactions des membres. Sur la base de ces hypothèses, notre travail vise à générer des équipes compétitives dans un réseau de collaboration pour

répondre à la demande de crowdsourcing tout en préservant la vie privée de chacun.

Pour ce faire, nous considérons que la fuite de données d'un utilisateur et de chaque membre de l'équipe est la composante principale de propagation des données privées d'une équipe à une autre. Ensuite, nous mettons en oeuvre une méthode de classification des différents membres participant à la résolution de la demande de crowdsourcing. Après, nous utilisons une technique de classement afin de choisir la meilleure solution en fonction de plusieurs préférences telles que la compétence et la valorisation du travail en équipe.

1.1.1 Exemple médical

1.1.1.1 Problème

Supposons qu'un hôpital doit masquer l'identité des patients et préserver leur vie privée alors qu'il décide d'externaliser l'analyse d'un patient au crowdsourcing où un ensemble de médecins experts traitent les requêtes désignées à la demande.

Considérons un patient p_x qui est atteint d'une maladie multifactorielle et hébergé dans un hôpital. Ce dernier doit masquer l'identité du patient p_x. Il se trouve dans l'obligation de préserver sa vie privée au cours du processus d'analyse.

Le patient doit avoir une analyse du sang, une analyse de l'imagerie des poumons, une analyse de la fonction respiratoire, et une étude de la compatibilité entre les médicaments. Or, les compétences des médecins disponibles à l'hôpital ne peuvent pas résoudre tous ces problèmes.

Compte tenu des maladies multifactorielles causées par une combinaison de facteurs génétiques, environnementaux et de mode de vie, dont la plupart n'ont pas encore été identifiés, et étant donné le nombre assez important de patients et la lourdeur de la procédure d'analyse, l'hôpital décide d'externaliser le problème du patient p_x au crowdsourcing tout en cachant son identité et préservant sa vie privée. L'hôpital a besoin d'un grand nombre de cardiologues, pneumologues, carcinologues et endocrinologue pour expliquer et analyser les différents problèmes du patient p_x.

1.1.1.2 Objectif

Le défi est de découvrir des équipes de collaboration tout en préservant l'intimité entre les équipes afin de maintenir la compétitivité entre eux.

L'examen des tâches peut être basé sur les récompenses monétaires. A cet effet, plusieurs équipes compétitives et collaboratives peuvent être construites pour résoudre le problème.

Pour ce faire, les médecins experts s'enregistrent sur le réseau médico-social afin de collaborer pour résoudre la demande de l'hôpital.

1.1.1.3 Solution

Le but est d'externaliser chaque problème du patient p_x à un groupe de médecins, et le défi consiste à interdire la reconstitution du profil du patient p_x en interdisant la propagation des données entre les équipes de médecin. Dans ce scénario, l'hôpital exige que les préférences devraient être autour de la $vie\ privee$, $\prec couverture\ des\ specialites \succ$ et $\prec favoriser\ le\ travail\ en\ equipe \succ$. L'hôpital doit définir et donner toutes les valeurs des préférences en utilisant un domaine fini afin de les appliquer pour choisir la meilleure solution répondant au mieux à sa problématique.

La solution est de générer des équipes compétitives et collaboratives répondant au préférences de l'hôpital tout en préservant la vie privée de chaque partie du processus de crowdsourcing.

1.1.1.4 Discussion

Dans un processus général, les médecins experts seront constitués par la compétence et par le domaine, mais la fuite de données entre les experts implique la validité des solutions de groupement (révélation de l'identité du patient).

Dans cet ordre, la répartition des médecins experts dépendra des fuites de données et de l'expertise des personnes. En particulier, la tâche la plus complexe est de savoir comment préserver les solutions tout en découvrant

des équipes de médecins experts.

Dans ce cas, les préférences de l'hôpital sont également importantes pour définir la découverte de la meilleure composition d'équipes et de donner une grande flexibilité de décision à l'hôpital pour choisir la meilleure solution.

1.1.2 Exemple de traduction

La traduction des livres fait toujours fantasmer les auteurs. En 2014, il existe entre 6000 et 7000 langues dans le monde entier.

1.1.2.1 Problème

Une traduction automatique dans une langue étrangère permet de comprendre globalement le contenu du texte, par contre, elle n'est pas précise ni fiable et en aucun cas il est possible de remplacer le traitement d'un traducteur.

Supposons qu'une maison d'édition souhaite traduire un document dans les différentes langues possibles. L'éditeur est dans l'obligation de ne pas dévoiler le contenu du livre avant sa date de publication. Actuellement il très difficile de trouver les différents traducteurs possibles pour les langues souhaitées. Il n'est pas possible de chercher des maisons de traduction dans tous les pays souhaités.

1.1.2.2 Objectif

L'objectif est de constituer des équipes compétitives et collaboratives permettant de traduire le document dans les différentes langues possibles tout en préservant la divulgation de son contenu.

1.1.2.3 Solution

La solution consiste à trouver un groupement possible des traducteurs avec les compétentes demandées. La fuite de données entre les différentes équipes compétitives permet de reconstituer le contenu du document demandé. Il nécessaire de minimiser la probabilité de fuite de données entre les équipes.

1.2 Problématique

La préservation des solutions lors de la découverte des équipes est basée sur des contraintes variées, incluant la vie privée comme un critère obligatoire du processus d'assemblage qu'un groupe d'utilisateurs doit satisfaire. Dans la figure 1.1, nous décrivons en détail les problèmes originaux et discutons ensuite l'approche que les utilisateurs pourront mettre en place.

Chaque membre est associé à une étiquette représentant ces compétences. Prenons six membres du réseau qui ont une relation directe entre eux. Par exemple, la fuite de données entre George et Alice est de 0,3 et entre Bob et George est de 0,7. Pour découvrir les équipes en fonction des compétences, une solution optimale est donnée par la combinaison des compétences et des liens pondérés entre les membres.

Ainsi, la solution dans ce cas est de regrouper les capacités des membres $\{David, Mickael, Alice\}$ $\{Bob, George, John\}$. Cependant, dans un immense réseau social et une plate-forme dynamique, la propagation de données privées est plus importante et influe sensiblement sur la capacité de regrouper un membre dans une équipe spécifique.

Étant donné que Alice et David ont une relation indirecte via Mickael, ceci nous amène à découvrir une fuite de données égale à 0,45 au lieu de la valeur 0 de la relation directe. Cette découverte nous permet de changer la composition des équipes afin de préserver la confidentialité de chaque équipe compétitive.

L' assemblage optimal des équipes de l'exemple donné est assez simple. Par contre la demande de formation d'équipes devient plus difficile dans un réseau social de taille immense.

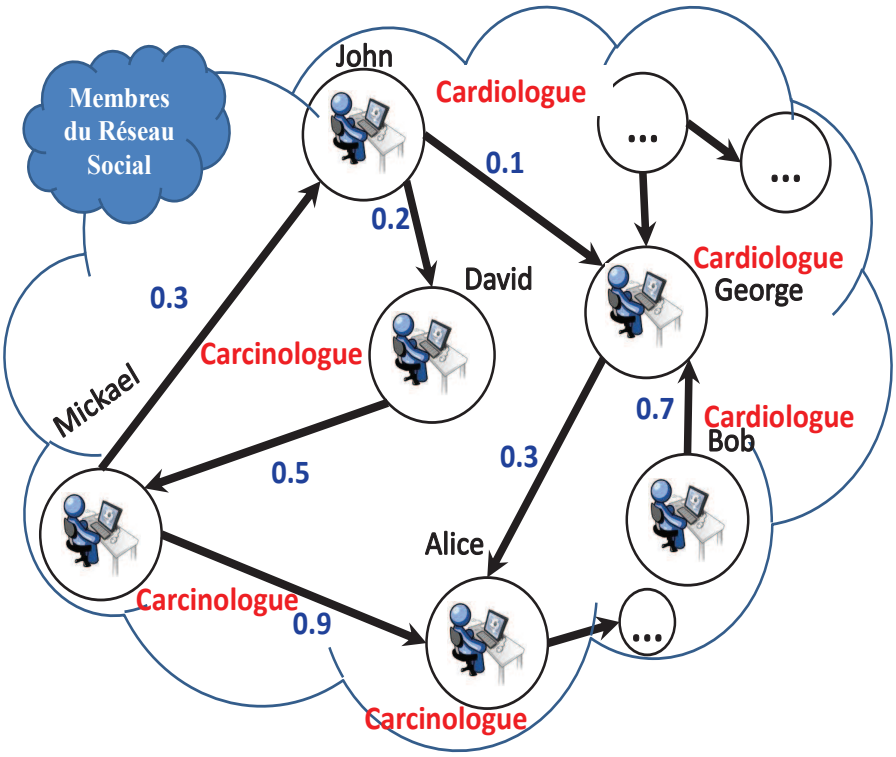

FIGURE 1.1 – Description des problèmes d'origines

Notre objectif est de créer des équipes compétitives et collaboratives sans fuites de données tout en tenant compte les préférences du demandeur.

Étant donné que le nombre de résultats est important et afin de découvrir la meilleure solution de composition des équipes, nous intégrons les préférences du demandeur en se basant sur une technique de classement.

Dans notre exemple, le demandeur requiert que :

- chaque équipe couvre toute les spécialités (Carcinologue et Cardiologues dans notre),

- les membres doivent collaborer au sein des équipes.

Les trois solutions suivantes sont composées d'équipes sans fuite de données :
(1) $\{Bob, George\}$ $\{Alice, David, Mickael\}$ $\{John\}$
(2) $\{Bob, George, John\}$ $\{Alice, David, Mickael\}$
(3) $\{Bob, George\}$ $\{Alice, John, David, Mickael\}$.

On constate que, uniquement, la solution (2) satisfait la première contrainte. La collaboration au sein des équipes d'une solution peut être mesurée par la moyenne des données partagées entre les membres des équipes. La collaboration au sein des équipes de la solution (2) est la suivante :

$$\frac{(0.5+0.9)+(0.7+0.1)}{4} = 0.55.$$

En augmentant les membres du réseau social et l'ensemble des relations, le nombre de combinaisons servant à vérifier les fuites de données croît de façon exponentielle. En même temps, la solution optimale est celle qui satisfait toutes les préférences du demandeur tout en préservant contre les fuites de données entre les équipes. Par conséquent, le choix de la meilleure solution de groupement possible serait en incluant les préférences des demandeurs pour classer les solutions,.

1.3 Contributions

Nous proposons une approche permettant d'une part de constituer des équipes compétitives et collaboratives répondant à des requêtes, et d'autre part d'éviter la divulgation des informations privées, tout en incluant les préférences des demandeurs. Quatre grandes étapes décrivent les progrès vers la découverte des équipes :

- Pour être au courant de la fuite des données entre les clusters, deux approches ont été proposées basée sur le modèle MapReduce ; la première en utilisant les propriétés des chaînes de Markov et la deuxième en utilisant l'algorithme de Dijkstra ; permettant de découvrir les relations implicites pouvant avoir un utilisateur avec les autres membres. Les relations directes entre les utilisateurs sont explicites. Cependant, il est obligatoire de découvrir les relations cachées (indirectes) entre les utilisateurs. Ainsi, la découverte de ces relations permettra un assemblage optimal des membres et évitera la fuite de données privées entre les clusters.
- Afin de découvrir des équipes compétitives et collaboratives, nous avons développé un algorithme basé sur une distance spécifique représentant la propagation maximale de regroupement des utilisateurs. L'algorithme regroupera les membres du réseau social tout en préservant la vie privée. Ceci permet d'éviter la fuite de données entre les équipes. Il permet

d'établir un arbre de résolution contenant toutes les solutions de groupement possibles.
- Afin de réduire la complexité de l'approche de découverte des équipes collaboratives et compétitives, nous avons développé un nouveau algorithme basé sur l'heuristique de recuit simulé. Cet algorithme permet de chercher dans l'espace de solutions possible et de trouver les meilleures solutions répondant à la requête de crouwdsourcing.
- Une approche de classement des différentes solutions de groupement possibles trouvées est développé en se basant sur les "'soft constraint satisfaction problem"' (SCSP). Les CSP ou Constraint Satisfaction Problem est une technique utilisée pour trouver les solutions satisfaisant la totalité des contraintes données. Or, il n'est pas toujours évident de trouver une solution permettant de satisfaire toutes les contraintes du demandeurs. D'où le recours aux SCSP qui permettent une relaxation des différentes contraintes pour satisfaire au maximum le demandeur de la requête. Les SCSP sont utilisées afin d'inclure les préférences du demandeur pour pouvoir classer les différentes solutions de groupement possible d'un travail spécifique externalisé au crowdsourcing.

1.4 Processus de crowdsourcing proposé

Nous concevons un processus de crowdsourcing pour découvrir les fuites de données entre les équipes compétitives. Ce processus intègre trois principaux composants représentés dans la figure 1.2 et qui seront abordés dans cette thèse.

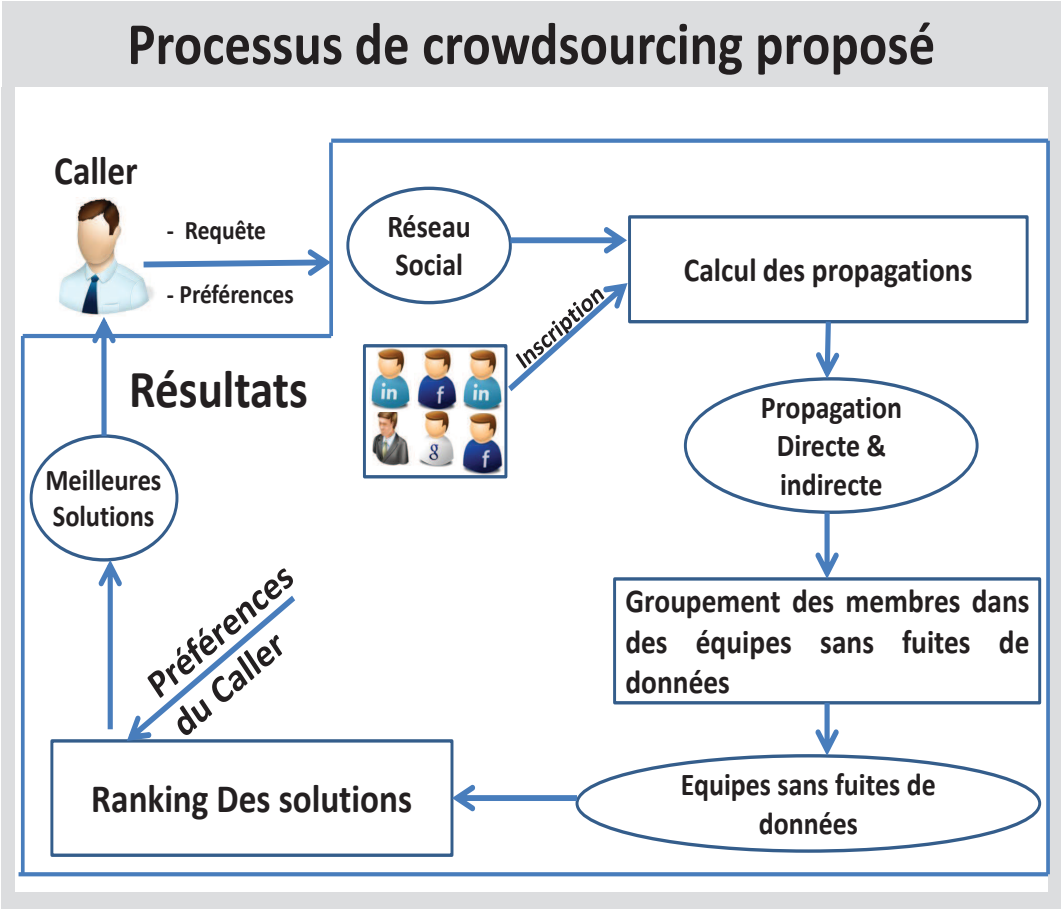

FIGURE 1.2 – Processus de crowdsourcing proposé

-Modèle de propagation de données :

Le système calcule les données sur la base des informations recueillies par le réseau social. Ceci permet d' interdire la fuite des données tout en découvrant les équipes. Dans la première étape, la demande est obtenue par le processus de propagation, identifiant ainsi les relations directes entre les membres du réseau social.

Ensuite, le processus découvre les interactions implicites dans le réseau social et maximise la propagation des données entre les membres. Les détails sont fournis dans la section suivante.

-Modèle de classification :

Dans une deuxième phase, un algorithme générera des solutions à partir de l'ensemble de la propagation des données calculées.

Le modèle de classification groupera les utilisateurs du crowdsourcing dans les mêmes groupes ayant une forte propagation et utilisant les préférences des demandeurs. Cela signifie que, plus la probabilité de propagation des données entre les utilisateurs augmente, plus ils doivent être dans le même groupe pour la collaboration et non dans les pôles de compétitivité.

Le modèle de classification découvre les compositions possibles d'équipes tout en interdisant la fuite de données entre les équipes.

- Modèle de classement de solutions :

Sur la base de la composition possible d'équipes fournie par le modèle de classification, le module de classement utilisera les préférences des demandeurs pour choisir la meilleure solution. Il combinera les préférences afin de trouver la solution qui satisfait la majorité d'entre eux. Ce modèle de classement des solutions est basé sur une approche SCSP. La solution retenue sera donc celle qui satisfait le maximum des préférences combinées. Les détails seront décrits dans la section 6.

1.5 Structure de la thèse

La dissertation sera structurée comme suit : après avoir introduire quelques définitions et la problématique dans le premier chapitre, on présentera le contexte et l'état de l'art dans un second chapitre,.

Ensuite, dans le chapitre 3, nous détaillerons nos approches proposées pour la découverte des relations cachées dans les réseaux sociaux. En particulier, nous présenterons une vue d'ensemble, puis, nous détaillerons les approches basées sur le modèle Markov, et le modèle de Dijkstra.

Le quatrième chapitre sera consacré à la découverte des équipes compétitives et collaboratives pour répondre au problème du demandeur. Pour cela, nous présenterons le modèle K-means,l'algorithme de classification hiérarchique, le modèle à solution unique, le modèle glouton puis le modèle parallèle. Les expériences des implémentations de ces approches seront traitées.

Dans le chapitre 5, on s'intéressera aux heuristiques qui permettent de réduire le temps de calcul pour la découverte des équipes. Nous présenterons l'utilisation des heuristiques ainsi que quelques différences de modèles. Ensuite, nous examinerons les résultats des implémentations de cette approche.

Le sixième chapitre concernera l'approche de classement des résultats retournés. Nous citerons quelques définitions des SCSP et nous nous concentrerons sur l'application de cette approche dans notre cas. Enfin, nous analyserons les résultats retournés par cette dernière.

Chapitre 2

Contexte et état de l'art

2.1 Préliminaires et définitions

La donnée privée :
La donnée privée est une catégorie d'informations sensibles qui est associé à un individu, comme un employé, un étudiant, un donateur ou un membre d'un réseau social. La donnée privée devrait être accessible uniquement sur une base stricte, manipulée et stockée avec soin.

La donnée privée est une information qui peut être utilisée pour identifier, contacter ou localiser une seule personne. Les renseignements personnels dépersonnalisés (maintenu d'une manière qui ne permet pas d'association avec une personne en particulier) ne sont pas considérés comme sensibles.

La confidentialité de données privées :
La confidentialité de données privées se réfère à la relation en évolution entre la «*technologie* et le *droit public de confidentialité de données privées* » dans la collection et le partage de données personnelles. Les préoccupations sur «*confidentialité de données privées* » existent partout la où il existe des données uniquement identifiables touchant à une personne ou plusieurs, rassemblées et stockées sous forme numérique ou autre.

Dans certains cas, ces préoccupations s'intéressent à la façon de rassembler, stocker et associer ces données. Dans d'autres cas, une question sur comment s'effectue l'accès aux informations et qui possède le droit de propriété et le droit de voir et de vérifier des informations qui appartiennent aux tiers , s'impose.

Pour des raisons diverses, les utilisateurs ne veulent pas que ces informations personnelles se rapportant soit à leur religion, l'orientation sexuelle, les affiliations politiques, ou les activités personnelles seront révélées. Cela peut éviter la discrimination, l'embarras personnel, ou les dégâts pouvant affecter la réputation professionnelle.

La confidentialité des données privées sur Internet est la capacité de contrôler toutes les informations personnelles et choisir ceux ou celles qui peuvent y accéder. Ces préoccupations s'intéressent également à la sécurité du stockage et la lecture du courrier électronique. On exige le consentement ou, dans le cas échéant, le suivie de la trace de navigation des internautes. Parmi ces préoccupations, on distingue aussi les sites Web qui rassemblent, stockent et partagent probablement des informations personnellement identifiables sur des utilisateurs.
La relation entre la confidentialité de données privées et le réseau social est à facettes multiples. En effet, dans certains cas nous publions des informations pour être connu uniquement par un petit cercle d'amis. Les étrangers seront donc totalement exclu. Dans d'autres cas, c'est l'inverse qui se produit.

La confidentialité des données privées associées au réseau social dépend essentiellement de l'identifiabilité de ces informations, ses destinataires et ses utilisations possibles. Même les sites Web des réseaux sociaux qui n'exposent pas ouvertement les identités de leurs utilisateurs, peuvent fournir assez d'informations pour identifier le propriétaire du profil. Ceci est dû , par exemple, à la ré-identification du visage [61].
Cette confidentialité peut être en danger, puisque les informations sont volontairement fournies. Différents facteurs mènent probablement à la révélation de ces informations dans des réseaux sociaux.

L'évaluation myope des risques de la vie privée [2] ; ou aussi la propre interface utilisateur du service, peuvent induire l'utilisateur à l'acceptation incontestée de préférences de vie privée perméables mise par défaut par le fournisseur.

Le crowdsourcing :
Le crowdsourcing est un terme utilisé pour décrire le processus d'externalisation d'une tâche traditionnellement faite en entreprise à un grand nombre d'internautes. Le concept de base derrière ce terme est d'utiliser un grand groupe de personnes pour leurs compétences, leurs idées et leurs participations pour générer du contenu ou pour faciliter la création de contenus ou produits. Le crowdsourcing est la distribution de la résolution de problèmes.

Si une entreprise a besoin de résoudre un problème, la foule est une ressource puissante et capable de générer d'énormes quantités d'informations.

Actuellement, Internet est un référentiel de contenu généré par les utilisateurs. La distinction entre le producteur et le consommateur n'est plus une telle distinction répandue. Aujourd'hui, tout le monde est équipé des outils nécessaires pour créer ainsi que pour consommer.

Du point de vue stratégie d'entreprise, en sollicitant l'avis du client n'est pas quelques choses de nouveau, et les logiciels open source ont prouvé la productivité évolutive grâce à un grand groupe de personnes. Bien que l'idée derrière crowdsourcing ne soit pas nouvelle, son utilisation active en ligne en tant que stratégie de construction d'affaires a seulement été depuis 2006. L'expression a été initialement inventée par Jeff Howe, où il décrit un monde dans lequel les gens extérieur à l'entreprise contribuent au travail pour le succès des projets. Les entreprises utilisent le crowdsourcing non seulement dans la recherche et développement, mais aussi pour obtenir des idées, des opinions, ou pour utiliser les capacités de raisonnement sur le plan qualitatif que la machine ne peut pas traiter.

Le crowdsourcing permet d'augmenter la productivité d'une entreprise tout en minimisant les frais de main-d'oeuvre, ce qui rend le crowdsourcing, un outil incroyablement efficace.

Anonymisation :
L'anonymisation est une pratique commune. L'adresse IP est l'attribut qui identifie l'utilisateur dans le réseau. Les traces du réseau apparaissent souvent après le chiffrage de l'adresse IP. Cette anonymisation respecte donc les buts utilitaires de l'administrateur de données étant donné que la plupart des analyses du réseau social peuvent se faire en l'absence de noms et d'identificateurs uniques.

De ce fait, l'anonymisation et la vie privée devraient être respectées. Par conséquent, les informations personnelles devraient être confidentielles. Et, dans certains cas, il peut être nécessaire de se décider s'il est approprié d'enregistrer certaines informations à caractère sensibles dans un réseau social. On devrait prévoir des menaces à la confidentialité et à l'anonymat. Les identités et les rapports entre participants, dans la recherche par exemple, devraient garder la confidentialité totale dans le cas d'un engagement antérieur dans ce sens. Des mesures appropriées devraient être prises en compte pour stocker les données d'une manière sécurisée.

Exemple 1 *Un internaute ouvre un compte sur un réseau social où il publie des informations sur un travail de recherche qui est en cours et au profit d'un*

laboratoire. Ces informations peuvent être utilisées ailleurs.

Les membres devraient respecter leurs obligations conformément à la Loi en vigueur. Il est judicieux d'utiliser les méthodes disponibles pour préserver la confidentialité de données privées quand cela est nécessaire et pratique. Ces méthodes peuvent inclure le déplacement d'identificateurs, l'utilisation de pseudonymes et d'autres moyens techniques pour couper la liaison entre les données et les individus identifiables comme 'broadbanding' ou la micro-accumulation.
Il est aussi nécessaire d'empêcher la publication des données sous une forme qui permettrait l'identification réelle ou potentielle d'un utilisateur.

Le contrôle d'accès :
Le contrôle d'accès consiste à vérifier si une entité (une personne, un ordinateur, ...) demandant l'accès à une ressource a le droit pour le faire. Il offre ainsi la possibilité d'accéder à des ressources physiques (un bâtiment, un local, un pays) ou logiques (un système d'exploitation ou une application informatique spécifique).

Ce contrôle d'accès comprend généralement 3 composantes :

- un mécanisme d'authentification de l'entité (un mot de passe, une carte, une clé, un élément biométrique,...) qui n'est pas utile en soi mais indispensable au fonctionnement des 2 suivants :
- un mécanisme d'autorisation (l'entité peut être authentifiée mais n'a pas le droit d'accéder à cette ressource en ce moment) ;
- un mécanisme de traçabilité : le mécanisme d'autorisation peut être insuffisant pour garantir que l'entité dispose du droit d'accès à cette ressource (respect d'une procédure ...). La traçabilité compense alors ce manque en introduisant une épée de Damoclès responsabilisant les entités. On peut également retrouver à posteriori le responsable d'une action.

Les utilisateurs contrôlent l'accès de leurs données partagées en cachant et en dressant la carte des informations choisies dans un third-party stockage. Tenant l'exemple, des images, qui pourraient être cachées dans un serveur de stockage comme Picasa6. Le besoin principal est de donner confiance à un third-party stockage approuvé pour les informations cachées.

Les auteurs ne considèrent pas que l'exécution du contrôle d'accès sélectif par les moyens cryptographiques alourdie le système d'identification à travers ce stockage thirdparty. Cette question, par exemple soulevée par le NOYB [118] qui chiffre des informations personnelles utilisant un chiffre de substi-

tution pseudo-aléatoire

Cependant, leurs oeuvres d'approche, qui chiffre seulement des données personnelles d'un domaine relativement petit ne permettent pas de chiffrer des entrées de texte libres comme fréquemment trouvé dans un réseau social. Dans une autre approche appelée «flyByNight» [88], Matthew Lucas et Nikita Borisov présentent une application Facebook pour protéger des données privées en les stockant sur Facebook sous forme chiffrée. Cette application permet à chaque personne de reporter la charge de gestion des clés de cryptage au réseau social qui possède à son tour des serveurs pour le faire. L'algorithme de décryptage est mis en oeuvre dans JavaScript et recouvré de Facebook.

Le navigateur est donc indépendant et supporte l'utilisation du terminal d'Internet arbitraires tant qu'ils supportent JavaScript, ce qui n'est pas sécurisé contre des attaques actives par le fournisseur du réseau social, Facebook. Le plus grand écart des auteurs est le manque de contrôle d'accès sélectif, tandis que NOYB assure une clé secrète partagée que l'on connaît au cercle d'utilisateur du réseau social d'amis et les amis d'amis.

Une autre approche développée par Ti Berners-lee & d'autres., permet de forcer le contrôle d'accès dans le web sémantique : l'obligation de vérifier l'autorisation de l'accès de l'utilisateur à un objet donné est à la charge du demandeur de la requête qui doit prouver au propriétaire des ressources satisfaisant les exigences de la politique du contrôle d'accès.[135]

On peut aussi citer l'approche introduite par Barabara Carminati, Elena Ferrai et Andrea Perego [25] . Elle permet le contrôle d'accès par certificat. Elle est basée sur le principe que lorsqu'un utilisateur demande des ressources d'un autre utilisateur, l'utilisateur des données reçoit un ensemble de règles de contrôle d'accès pour réglementer la publication des ressources du propriétaire.
Ces règles englobent le type de relation que doit exister entre le propriétaire des ressources et le demandeur exigé, la profondeur maximale de l'autorisation ainsi que le niveau de confiance permit.
Le demandeur des ressources doit obligatoirement fournir des justificatifs au propriétaire pour prouver l'existence du type de relation requis avec la profondeur exigée et le niveau de confiance minimum.
Ainsi, les ressources fournies avec ces justificatifs permettent de vérifier localement la publication des ressources. Pour plus de précision, dans leurs système, les règles de contrôle d'accès sont exprimées en utilisant "'N3"' et évaluées par le "'Cwn Reasoner"'.

L'adoption de N3 au lieu du langage des règles du web sémantique comme RuleMl [67] ou bien SWRL [65], est déterminé par le fait que N3 autorise une représentation compacte des éléments de "'RDF"' ce qui peut être plus facile à délivrer à travers "'HTTP headers"'.

Identification :
L'identification permet de connaître l'identité d'une entité, souvent à l'aide d'un identifiant tel qu'un nom d'utilisateur. Dans les réseaux sociaux un identifiant est l'ID unique de l'utilisateur choisi par ce dernier.
Un utilisateur est une personne qui utilise un ordinateur ou un service internet, qui peut avoir un "'user account"' ou aussi un "'screen name"'. Dans les systèmes d'informations, la correspondance d'identité est un processus qui concilie et valide la propriété du compte utilisateur qui réside sur des systèmes et des applications à travers une organisation et des liens permanents identifiant la propriété du compte.

Trust :
la Confiance est définie par (Mayer, Davis et Schoorman, 1995) comme étant la volonté d'une partie à être vulnérable aux actions d'une autre partie basée sur l'espérance que l'autre volonté exécute une action particulière importante, sans tenir compte de la capacité de contrôler cette autre partie ".

La confiance est un déterminant critique pour partager des informations et développer de nouvelles relations (Fukuyama, 1995, Lewis et Weigert, 1985). Elle est aussi importante pour des interactions en ligne réussites (Coppola, Hiltz et le Sale type, 2004, Jarvenpaa et Leidner, 1998, Meyerson, 1996, Piccoli et Ives, 2003). Des millions de gens rejoignent des sites des réseaux sociaux, ajoutant les profils qui révèlent des informations personnelles.

Les réputations des sites de réseau social ont été diminuées d'un certain nombre d'incidents rendus publics par les mass-media (Chiaramonte et Martinez, 2006, Hass, 2006, Mintz, 2005, Lu, 2006). On ne s'attend pas souvent à la confidentialité des données privées dans des sites d'un réseau social. Ces derniers enregistrent toutes les interactions et les conservent pour l'utilisation potentielle dans l'extraction de données sociales. En mode autonome, la plupart des transactions sociales ne partent derrière aucune trace.

Ce manque de rapport est un activateur passif de confidentialité des données privées. Ces sites ont donc besoin de politiques explicites et des mécanismes de protection de données pour livrer le même niveau de confidentialité des données privées trouvées en mode autonome [10].

Contrôle d'usage :
Le contrôle de l'usage est le fait de contrôler l'utilisation des données personnelles après la publication sur un réseau social. Les données personnelles de chaque utilisateur d'un réseau social sont privées ou publiques à tous les membres en fonction du choix de préférences fait par le propriétaire unique des données. Cependant, un problème se pose de nos jours. Il s'agit de trouver la manière de contrôler nos données privées dans un réseau social public si par exemple on veut partager des informations avec une partie des membres du réseau social ou de contrôler l'usage de l'utilisation des données privées dans l'arbre de relation de chaque utilisateur.

Une autre approche développée par S.Benbernou, H.Meziane, M.S.hacid. Cer derniers présentent un système de surveillance de l'accord de conformité d'usage des données privées[17]. Ils expliquent le problème de surveillance de la conformité de l'accord du contrôle de l'usage qui énonce les droits d'intimité d'un utilisateur et comment les informations personnelles d'un utilisateur doivent être manipulées par le fournisseur du service.
Ils proposent un modèle de machine qui décrit le "'PDUF"' vers la surveillance qui peut être utilisée par les analystes du contrôle de l'usage des données privées pour observer les flux et capturer la vulnérabilité de l'usage qui peut conduire à la non conformité.

2.2 Le crowdsourcing

2.2.1 La description du crowdsourcing

Le crowdsourcing est le processus d'obtenir un travail ou un financement, d'habitude en ligne, d'une foule. Il est composé du mot 'la foule' et 'l'externalisation'. L'idée est de prendre le travail et l'externaliser à une foule de travailleurs.

Un exemple célèbre du crowdsourcing qui est Wikipédia. Ce dernier, au lieu qu'il crée une encyclopédie tout seul, recrutant des auteurs et des rédacteurs, il a donné à une foule la capacité de créer les informations tout seul. Wikipédia est donc l'encyclopédie la plus complète.

Crowdsourcing et la qualité : le principe de crowdsourcing est que plusieurs têtes sont meilleures qu'une seule. En démarchant une grande foule

pour des idées, des compétences, ou la participation, la qualité du contenu et la génération d'idée sera supérieure.

Les Différents types de Crowdsourcing :

Crowdsource Design :

En vu de chercher une conception de logo, on peut expliquer à une foule de concepteurs le résultat voulu, combien on peut payer et le délai de livraison. Tous les concepteurs intéressés créeront un design fini spécifiquement à la requête. On recevra de différentes conceptions de logo finis et on gardera le meilleur. En faisant le design de cette façon (faisant du crowdsourcing) augmente en réalité la qualité et diminue le prix, comparé au travail en indépendant en ligne.

Le crowdsourcing peut faire des conceptions multiples comme les designs des meubles, la mode, des publicités, la vidéo ou le design produit.

Le Crowdfunding :

Le Crowdfunding implique la demande de financement d'un projet par la foule (généralement sous forme de don). Par exemple, si on veut avoir 10,000 $ pour payer un studio qui va enregistrer un nouveau CD, le crowdfunding peut nous aider à collecter cet argent.

Pour le faire, il suffit de chercher une plate-forme de crowdfunding, on renseigne le montant voulu, le délai et n'importe quelle récompense à offrir aux donateurs. On doit atteindre 100 % de notre but avant la date limite, ou toutes les donations seront rendues aux donateurs. Les délais sont typiquement moins de 60 jours.

Le Crowdfunding est utilisé par la majorité des artistes et des start-up afin de se procurer de l'argent pour des projets comme le tournage d'un documentaire, la fabrication d'une montre, la recherche sur le cancer, ou le capital de départ. Pour plus d'informations sur le crowdfunding, on peut se référer au site web suivant : http ://fr.vox.ulule.com/crowdfunding-1195.

Micro tâches :

La micro-gestion implique la cessation du travail dans des tâches minuscules et l'envoi du travail à une foule. Si on a 1,000 photos sur un site Web et que ces photos doivent être présentées avec des légendes, on peut deman-

der à 1,000 personnes d'ajouter une légende à chaque photo. On peut à tout moment, rompre le travail et décider de payer celui qui accompli sa tâche (typiquement 0.01 $ - 0.10 $ par tâche). Avec la micro-gestion, on peut s'attendre à voir des résultats après quelques minutes. La micro-gestion peut impliquer des tâches comme le balayage d'images, la correction des épreuves, la correction de base de données et la transcription des fichiers audio.

Le travail sera donc achevé plus rapidement, moins cher et avec moins d'erreurs, surtout quand les systèmes de validation sont en place. De plus, les micro-tâches peuvent souvent être exécutées par les gens dans des pays moins chanceux, y compris ceux qui utilisent les SMS, mais ne possèdent pas d'ordinateurs.

Innovation Ouverte :

En utilisant le crowdsourcing, on peut débuter un business et faire des designs pour des produits. En effet, le crowdsourcing peut aider par l'innovation ouverte. Cette dernière permet aux investisseurs, des concepteurs, des inventeurs et des gens du marketing de collaborer dans un bénéfice fonctionnel faisant la réalité. Ceci peut être fait par une plate-forme Web consacrée pour gagner la perspective extérieure, ou par des salariés internes.

L'innovation ouverte rassemble les gens de différents secteurs mondiaux pour construire ensemble un projet. Ceci est une collection efficace de domaines différents et de niveaux d'expertise qui ne serait pas autrement disponible pour chaque entrepreneur.

Professionnel et Duperies :

Le plus grand avantage du Crowdsourcing est la capacité de recevoir des résultats de meilleure qualité, étant donné que les travailleurs offrent leurs meilleures idées, leurs compétences et leurs supports. Le crowdsourcing permet alors de choisir un des plus meilleurs résultats. Ces derniers peuvent être livrés beaucoup plus rapidement que des méthodes traditionnelles, vu que le crowdsourcing est une forme de travail en indépendant. On peut donc obtenir une vidéo finie dans un mois, un design fini ou une idée dans une semaine et les micro-tâches apparaissent dans les minutes qui suivent la demande.

Des instructions claires sont essentielles dans le crowdsourcing. On peut potentiellement fouiller des milliers d'idées possibles, qui peuvent être minutieuses, ou même compliquées, si les instructions ne sont pas clairement

comprises. La qualité peut être difficilement jugée si les espérances appropriées ne sont pas clairement exposées.

2.2.2 L'utilisation du Crowd

Un modèle de réponse à des requêtes du crowdsourcing a été présenté par le groupe de Kossmann [53] . Ils présentent une solution initiale des problématiques de fouille de données et les comparer. Pour cela, ils proposent un simple schéma SQL et les extensions de requêtes qui permettent l'intégration de données du crowdsourcing et leurs traitements. Ils présentent aussi la conception du CrowdDB incluant des nouveaux opérateurs de requêtes pour le crowdsourcing et planifient des techniques de génération qui combinent des opérateurs de requêtes traditionnels et des opérateurs de requêtes du crowdsourcing. Ensuite, ils décrivent des méthodes pour générer automatiquement des interfaces utilisateur efficaces pour des tâches destinés au crowdsourcing. Ils présentent aussi les résultats du benchmarking des performances des opérateurs de requêtes faites pour le crowdsourcing sur la plate-forme d'ATM. Ils démontrent que le CrowdDB est capable de faire ce que les systèmes de gestion de base de données traditionnel ne peuvent pas le résoudre. Cette approche la façon de proposer les tâches (questions) dans le crowdsourcing, par contre, elle ne traite pas la résolution de problème complèxe nécessitant la collaboration de plusieurs travailleurs ni les problématiques de la vie privée.

Une nouvelle approche a été proposée par Florian et al. permettant d'intégrer des capacités humaines dans des flux de gestion de données [125]. Ils se sont concentrés sur l'application du crowdsourcing en étendant le modèle SOA traditionnel avec des services fournis à l'homme. Ils discutent une approche qui permet de savoir les utilisateurs du Crowd sourcing dont les interactions sont basées sur les compétences, la tâche effectuée et les préférences du réseau social. Ils clarifient l'architecture SOA en présentant le système de gestion de la confiance du réseau avec XHTML. Ils ont montré que leur approche peut être utilisée dans un scénario d'application du monde réel. Par contre, il ne traite en aucun cas la collaboration et la compétition entre les travailleurs du crowdsourcing. Leurs travail montre d'identifier un utilisateur de la foule pour résoudre la tâche.

Le groupe de Kumar [139] introduisent le problème des recherches d'images. En effet, pour des variations d'éclairage, de texture ou de qualité d'image leur

recherche s'est avérée précise et très peu de résultats sont erronés. Dans le but d'augmenter la précision des résultats de recherche, ils ont proposé une solution nommée CrowdSearch, qui permet de fournir un système précis de recherche d'image pour les appareils mobiles tout en combinant une recherche d'image automatisée, puis une validation humaine basée sur le crowdsourcing. La combinaison présente un ensemble complexe de compromis impliquant le délai, la précision et le coût. Leurs approche ne prend pas en compte la collaboration et ne gère pas la vie privée dans les traitements de recherche.

Le groupe de Gianluca [38] s'intéressent au problème de liaison des entités d'un texte au nuage de données ouverte. Ils rattachent les définitions aux mots dans un texte. Pour cela, ils proposent le ZenCrowd permettant de relier les mots avec la base de définitions. Ils combinent les algorithmes automatiques et le rattachement manuel en utilisant le crowdsourcing. Puis, ils estiment dynamiquement les crowders (les utilisateurs du crowd) à travers un framework de raisonnement probabiliste. Cette approche traite uniquement le rattachement de données par une seule personne. La collaboration pour l'identification du meilleur rattachement n'est pas présentée.

Le groupe de Kittur[73] traitent le problème de l'écriture d'article. Ils proposent un framework basé sur le modèle MapReduce pour les crowders nommé CrowdfForge. Ceci a pour but de partager les tâches entre les différents intervenants dans le monde du crowdsourcing. Ils divisent les tâches utilisant le MapReduce afin de proposer aux utilisateurs une tâche simple, puis assemblent toutes les réponses. Ce travail propose une division des tâches par utilisateurs et n'offre pas une collaboration et compétition dans la résolution des tâches assignées. Le travail ne traite pas également les problèmes de la vie privée sur la propriété intellectuelle

Le groupe de Hui [68] introduisent le crowdfunding, qui exploitent la puissance de la foule pour financer de petits projets. Par contre, il ne présente pas l'impact sur les données partagées dans le crowdfunding et les effets négatifs sur la vie privée.

Le groupe de Marcus [95] et [94] s'attaquent au problème d'utilisation des humains pour comparer, trier et assembler des types de données complexes telles que les images et les Blobs. Ils proposent une approche nommée «Qurk». Cette approche est un système de traitement de requête déclarative conçu pour délivrer des tâches extraites, afin de, filtrer et relier des images. Qurk utilise les fonctions définies par l'utilisateur en SQL pour spécifier les interfaces utilisateurs et donner des instructions sur les " Turkers " ou crowder

afin d'exécuter des tâches. Ils ont développé la jointure simple, qui crée des tâches pour chaque paire d'enregistrements. Pour chaque tâche, demander aux gens dans le monde entier de décider si oui ou non les deux documents se réfèrent tous les deux à une même entité. Ils ont développé le dosage naïf au lieu de placer plusieurs paires d'enregistrements en tâches simples. L'utilisateur doit vérifier individuellement chaque paire dans la tâche. Enfin ils ont développé le dosage intelligent qui place plusieurs paires d'enregistrements en une seule tâche, mais qui demande aux utilisateurs de trouver toutes les correspondances entre les enregistrements.Leurs approche ne traitent que les problèmes pouvant être prise en compte par un seul utilisateur du crowd. Or actuellement, les problèmes sont devenus de plus en plus complexe et ne pouvant plus se gérer qu'avec une collaboration. La question de la vie privée n'est pas traitée dans cette approche.

Walid G. Aref et al.[92] traitent le problème de la construction d'un arbre avec des clés (images ou vidéos) qui ne peuvent pas être indexés par l'ordinateur. Ils présentent une approche de construction d'index qui combine les propriétés subjectives avec quelques valeurs qualitatives, la construction d'index est basée sur les valeurs quantitatives. Cette approche ne traite pas la propriété intellectuelle des données lors de la construction des arbres d'images ou de vidéos, d'où il ne traite pas les problèmes liées à la vie privée. La collaboration n'est pas possible également parce que chaque requête est traitée par une seule personne.

Gianluca Demartini et al.[39] présentent la conduite hybride homme-machine, un système innovant utilisé pour répondre à des requêtes de mots clés complexes. Ils utilisent les crowders pour comprendre la requête et proposent un nouveau langage pour répondre aux requêtes. Le travail traite les requêtes complexe par contre, il ne donne pas la possibilité de travailler ensemble dans un système de crowdsourcing permettant d'avoir des résultats plus fiable et efficace. Les auteurs persente les requêtes complexex au travailleurs du crowdsourcing par contre, il ne traite comment garder la vie privée dans ce processus.

Yael Amsterdamer et al.[9] et [8] développent l'exploitation des capacités des utilisateurs extraites de la foule. Ils définissent une méthode pour accéder aux données personnelles en se basant sur des règles d'association. Les définitions de soutien et la confiance sont utilisées comme une mesure de l'importance d'une règle pour chaque utilisateur. Les auteurs, utilisent les données privées des utilisateurs mais ils ne traitent pas la protection de ces informations personnelles. Ils ne gèrent pas également la collaboration dans

un processus où les données privées sont des règles importantes d'associations.

Susan B. Davidson et al. [35] formalisent le problème de top-k et celui du group-by dans un contexte de crowdsourcing pour l'utilisation dans la classification. Ensuite, ils donnent un algorithme efficient qui garantie l'achèvement de bon résultats. Ils analysent leurs complexités et montrent que quelques questions sont nécessaires lorsque des valeurs et des types sont corrélés ou lorsque le modèle d'erreur est le même. Le travail effectué ne gère en aucun cas la protection de la vie privée dans un contexte de crowdsourcing.

[117] ils introduisent le problème de crowdsourcing interactif. Ils présentent le smartCrowd, un framework pour mobiliser le crowdsourcing dans le but d'approcher la réalité, tout en tenant compte de l'incertitude innée du comportement humain. Ils formulent l'attribution des tâches comme un problème d'optimisation, de manière à ce que le processus d'attribution de tâche soit optimisé à la fois qualitativement et aussi sur le temps de calcul. Ils présentent des analyses théoriques rigoureuses du problème d'optimisation. Le travail présenté n'offre pas une possibilité de collaboration entre les utilisateurs des terminaux mobiles. Sachant que les utilisateurs utilisent leurs propres informations privées, les chercheurs ne s'attaquent pas également au problèmes de vie privée

2.2.3 Le Crowd dans les réseaux sociaux

Nous vivons une "époque de réseaux sociaux". Facebook, twitter, ou Linkedin deviennent très populaires grâce à des services de partage de contenus. L'utilisateur partage des amitiés, des informations personnelles tels que l'âge, le sexe, le lieu d'habitation. Cette tendance a incité les organisations à s'éloigner des pratiques traditionnelles des affaires en se dirigeant vers le web qui offre plus de diffusion d'informations. En effet, les entreprises partout dans le monde sont interconnectées les unes avec les autres. En même temps, un nouveau type de travail d'équipe est apparu appelé le Crowdsouring. C'est l'action de l'externalisation des tâches, traditionnellement effectuées par un employé ou un entrepreneur, à un groupe indéterminé de personnes à travers un appel d'offres ouvert. Les applications de Crowdsourcing permettent de rechercher des personnes à la demande du lanceur de l'appel afin d'effectuer une tâche. Plusieurs utilisateurs vont montrer leurs compétences et leurs capacités pour répondre à cet appel. Étant donné un appel à différents réseaux

sociaux pour trouver une solution à un problème, les questions qui peuvent être prises en compte comprennent, comment mettre en place et découvrir les équipes qui vont répondre à la requête et la façon de préserver la divulgation partielle ou totale des solutions au problème lors de la constitution des équipes.

En fait, la compétitivité de chaque équipe est impactée par la divulgation des informations privées de la solution de l'équipe. On ne peut pas avoir des équipes compétitives alors que leurs données privées circulent dans le réseau social.

Les membres qui travaillent en collaboration sur une tâche peuvent partager des informations sensibles (une partie de la solution du problème par exemple) qui peuvent êtres propagées ou transmises aux membres des autres équipes dans le réseau social.

L'équipe de Gianluca Demartin [41] en se basant sur les réseaux sociaux, ils cherchent l'utilisateur convenable pour répondre aux tâches. Ils modélisent les utilisateurs à partir des informations du profil provenant du réseau social et les tâches précédemment complétées. Ils développent une application Facebook nommée Open Turk en implémentant les affectations des tâches de la plateforme Pick A Crowd. Cette approche ne gère en aucun cas le groupement de meilleurs profils permettant une meilleures collaboration et un meilleur résultat à la requête. Ils ne traitent pas aussi les problématiques liées à la vie privée en externalisant la résolution des tâches au crowdsourcing.

2.3 La découverte des relations implicites

2.3.1 Les données privées

Pour viser l'apprentissage de préférences de vie privée, Lujun Frang et Kristen LeFevre [51] ont conçu un outil qui permet de configurer automatiquement les préférences de vie privée d'un utilisateur avec un effort minimal, afin de classifier les utilisateurs et affecter automatiquement de nouveaux utilisateurs à ces classes. Cet outil, facile à utiliser, fournie de simples interactions aux utilisateurs. Ce qui leur permet d'activer automatiquement la gestion de la vie privée. L'outil demande à l'utilisateur, en mode interactif, d'assigner des étiquettes spécifiques, soigneusement choisi, de la vie privée des amis. Tant que l'utilisateur fournit plus de saisie, la qualité du classificateur s'améliore.Par contre, les auteurs ne présentent pas les propagations

des données personnelles malgré la protection de la vie priée.

[126], les auteurs ont présenté l'outil de protection de la vie privée qui mesure la quantité de fuite sensible de l'information dans un profil utilisateur et suggérer de régler la quantité de fuite. L'outil de protection détermine la probabilité des données sensibles dans le profil de l'utilisateur en se basant sur ses relations et sur les attribues de ses amis. Ils ne traitent pas le problème du contrôle de l'usage des données partagées.

[26] a conclu le besoin des mécanismes les plus flexibles pour la protection, laissant un utilisateur capable de se décider que certains participants du réseau sont autorisés à avoir accès à ses ressources et ses informations personnelles pour faire respecter les politiques de vie privée. D'où, ils ont concentré leur travail sur la protection des relations, en proposant une stratégie qui exploite des techniques cryptographiques pour faire respecter une propagation sélective des informations à travers les relations dans les réseaux sociaux. Le problème de propagation de données partagées dans le réseau n'est abordé, ils présentent une approche pour sécuriser l'accès à des données privées.

L'équipe de Wang [81] ont présenté une première tentative dans la modélisation de réseaux sociaux pour développer un paradigme d'évaluation statistique de risque et quantifier les menaces internes. Pour cela, ils ont modélisé l'accès d'un utilisateur aux informations privées et les garanties d'accès postérieur aux données privées.

Ils ont développé aussi les opérateurs d'évaluation de risque qui capturent de tels effets du réseau. Ils représentent aussi leur première tentative d'étude de l'impact des effets du réseau dans le contrôle d'accès à base de risque, un paradigme de sécurité émergent. Leur travail se limite au contrôle d'accès de la données dans le réseau et ne présente en aucun cas une approche de contrôle de propagation de ces données dans le réseau.

Dans une autre étude, Foin et al [63] ont évalué le degré de distribution d'un graphe puis ils ont développé un algorithme qui fournit une vie privée contenant des règles strictes et adaptable. La statistique supplémentaire peut être incorporée dans le cadre de la vie privée. Cet algorithme satisfait une norme de vie privée rigoureuse appelée différentielle de la vie privée. Par contre, ils ne gèrent en aucun cas les données partagées par des utilisateurs après l'application des règles de vie privées définies.

Bansal et al [11] ont proposés une solution qui préserve la vie privée pour la propagation dans les réseaux de neurones dans le cas de données arbi-

trairement partagées. Leur algorithme est basé sur des informations privées aléatoirement partagées entre deux partis. Leur solution se base sur le cryptage des données, par contre, ils ne gèrent pas l'utilisation des données privées décryptées. Ils sécurisent la propagation des données dans les réseaux de neurones mais ne contrôle pas l'usage de ces données.

2.3.2 l'analyse des relations

Les modèles de propagations de données sont généralement représentés en trois catégories distinctes. Les modèles de contagion, les modèles d'influence sociale et les modèles d'apprentissage social. Dans [140] et [98] les auteurs ont présentés les modèles de propagations de données basé sur l'apprentissage social. Ces modèles se basent sur l'utilité de l'information propagée pour les utilisateurs voisins. Par contre, les auteurs n'ont pas essayé de trouver la propagation maximale entre l'utilisateur propriétaire de l'information et le reste des membres dans le réseau social.

Les modèles de contagion se basent sur les utilisateurs actifs et leurs voisins qui s'activent en contact avec un membre actif. Ces modèles s'intéressent au passage d'un état sain à un état infecté ou à d'autres états. Différentes équipes [127],[102], et [21] ont développés plusieurs variantes des modèles de contagion. [57] ont proposé le modèle IC. Ce modèle est fondé sur le principe que dès qu'un noeud u est actif, il existe une probabilité $P_{u,v}$ d'activer ces noeuds voisins v. Ce travail n'a pas traité la propagation des données dans le réseau social et n'a pas pris en compte la différence des profils des utilisateurs.

Les modèles d'influence sociale reposent sur le principe qu'un individu s'active si le nombre des membres voisins déjà activés est supérieur à un seuil données.
Les chercheurs dans [121], [59] ont proposés les premiers travaux sur les modèles d'influence. Ils développent leurs approches en se basant sur le principe que la pression sociale détermine l'activation des noeuds.

Différents travaux, [114], [42], [87] et [20] ont proposés des approches se focalisant sur le fait qu'un noeud v est actif si la somme des poids des relations entrantes à ce membre est supérieure à un seuil propre à lui. Par contre, les travaux ne s'intéressent pas à la propagation des données d'un membre u vers tous les membres du réseau. Ils ne prennent pas en compte les similarités des profils pour valoriser la probabilité de propagation entre les individus.

Une modélisation de la force de relation dans des réseaux sociaux a été développée par Rogati et al [137]. Elle permet d'évaluer la force avec les interactions et la similitude des utilisateurs. Ce modèle peut représenter la totalité des forces de relation et proposer une méthode non surveillée pour les déduire. Cette force a directement un impact sur la nature et la fréquence d'interactions en ligne entre une paire d'utilisateurs, donc, impacte les préférences de vie privée de chaque utilisateur. Cette approche ne traite pas gérer l'usage des informations privées partagées mais plutôt le calcul de la force de relation pour mettre à jour la préférence de vie privée.

2.4 Anthologie

Historiquement, l'homme a essayé de décrire le monde autour de lui en trouvant des régularités dans ce qui est perceptible. En effet, les modèles d'interaction entre des entités apparaissent clairement, à l'échelle macroscopique et microscopique, dans la plupart des environnements naturels et artificiels. Citons par exemple, la relation proie -prédateur dans des écosystèmes naturels, le processus de propagation d'un virus ou les interactions entre deux ou plusieurs protéines.

2.4.1 La description des réseaux sociaux

En général, chaque système mettant en capsule une relation ou une interaction parmi certains de ses éléments constitutifs, admet un peu de représentation abstraite en termes d'un réseau, ou, d'une perspective mathématique, d'un graphe. Ce dernier est formé par un ensemble de sommets (ou des nœuds) qui identifie les éléments du système, avec un ensemble d'arêtes représentant la relation entre les sommets. Un tel cadre théorique très général peut être appliqué pour modeler une large variété de systèmes complexes. Le terme «complexité» a été défini par Barrat et al comme suit : *Complex systems consist of a large number of elements capable of interacting with each other and their environment in order to organize in specific emergent structures.* L'analyse de systèmes complexe est donc une discipline relativement jeune. Elle s'est concentrée sur l'étude empirique et la modélisation systématique de réseaux réalistes. Ceci a pour but d'extraire les informations cachées par les modèles complexes.

L'Analyse des Réseaux Sociaux (SNA) a été une des premières disciplines qui peuvent être classifiées sous l'étiquette "de l'analyse de systèmes

complexe". Initialement, l'SNA était une branche de sociologie (ou, plus précisément, une évolution de sociométrie) qui compte sur les notions de théorie des graphes pour comprendre la dynamique des accumulations sociales des gens. Revenant aux premières époques de l'SNA, les techniques de collecte de données (principalement, enquêtes écrites par des individus) ne permettent pas l'analyse des réseaux de grandes tailles. Cependant, même les très petits environnements, elles peuvent être traitées par des modèles complexes et par de nombreuses techniques. Ces derniers sont largement utilisés de nos jours dans l'analyse des ensembles de données de grandes tailles qui ont été étudiés et mis au point à partir de l'analyse des petites affaires du monde réel [132] .

Récemment, l'évolution rapide d'Internet, et particulièrement du Web dans sa facette collaborative, a fourni une source fraîche des données sociales pouvant être extraites pour décrire les modèles sous-jacents des systèmes sociaux en ligne ayant des tailles importantes et très dynamiques.

De telles données sont permises pour tester les théories classiques de l'SNA à plus grande échelle. Par exemple, les résultats de l'expérience de Milgram [97], sur les six degrés de séparation, le nombre des étapes qui sépare n'importe quels deux individus dans le graphe de connaissances sociales sont étonnamment petits (6 était le nombre évalué pour la population américaine). Cette information a été récemment vérifié sur le Messenger de Microsoft [83] et Facebook [128].

Un autre exemple est la théorie Dunbar qui est basée d'une part sur l'observation de la dynamique sociale d'un groupe de primates, et d'autre part sur l'utilisation d'une équation de régression sur leur taille corticale, Cette théorie revendique que "la taille moyenne de groupe" permettant des relations sociales stables et profitables pour des humains, se situ aux alentours de 150 [47, 46] ; la même limite cognitive théorique a été vérifiée dans le réseau Twitter [58].

En plus des études de socio-systèmes virtuels, l'analyse de réseau complexe a été efficacement appliquée pour aborder des problèmes sociaux et humanitaires comme la propagation de maladies sexuellement transmissibles [86] et la prolifération d'organisations terroristes [138].

En général, une étude axée sur les graphes appelés "réseaux sombres" est utile pour découvrir les faiblesses dans la structure des accumulations indésirables d'individus. Ces faiblesses permettent d'effectuer des attaques

ciblées capables de perturber l'efficacité du réseau avec un coût minimal. L'analyse structurelle a été un domaine super intéressant également sur des réseaux technologiques, Internet [50] ou le Web à titre d'exemple. La croissance rapide d'une telle échelle mondiale des systèmes technologiques a soulevé pour la première fois (bien avant la diffusion virale de réseaux sociaux en ligne) le problème de traitement de la dimension très élevée des données et la difficulté conséquente d'extraire des modèles significatifs.

Le travail fondamental sur de tels systèmes a été effectué, par exemple, dans l'évaluation des frontières et le diamètre du World Wide Web [6] et dans le dévoilement de sa forme pour être organisé dans trois composants principaux (Entrée, sortie, connexion forte) entouré par des vrilles [22].

Autour du millénaire, la disponibilité de l'ensemble de données ayant beaucoup de sources et de différents domaines, a déclenché un effort significatif dans l'enquête de toutes les sortes d'environnements de réseau. Ceci, en utilisant le paradigme de l'analyse de systèmes complexe. Une telle étude permettra de découvrir des régularités saisissantes dans de divers réseaux.

Plusieurs systèmes d'organisation technologique, biologiques ou sociaux [101, 7] se trouvent caractérisés par des petites propriétés [133], qui déterminent une croissance logarithmique de la distance moyenne entre les paires de nœuds. Ceci concerne le nombre total de nœuds.

Beaucoup de ces réseaux ont été retrouvés sans échelle, à savoir caractérisés par une distribution de connectivité de nœud qui se délabre comme une loi de puissance [12, 24]. La découverte de ceux-ci et beaucoup d'autres invariants, ont permis au domaine d'analyse de systèmes complexes de détecter et d'exploiter des modèles cachés et omniprésents dans des structures technologiques, ainsi que dans la vie biologiques.

Depuis lors, la perspective de l'étude fondée sur les données de systèmes basée sur les graphes a considérablement élargi les résultats de l'analyse exploratoire des grands réseaux [5, 69, 82]. Plus de richesse de données sur l'activité humaine, la mobilité et les interactions [119] ouvrent la voie pour concevoir les services qui peuvent profitablement combiner plusieurs sources de données de systèmes complexes.

En plus, de nouvelles recommandations et des outils de suggestion pour des réseaux sociaux en ligne apparaissent. Par exemple, on peut citer : le contrôle de diffusion de virus dans des réseaux de contact informatiques

ou humains [31, 110], le contrôle d'interactions face à face à l'intérieur des hôpitaux pour la minimisation des infections dans la structure [13], "la publicité extérieure" sur des téléphones portables [113] basé sur les modèles de mobilité des gens dans une ville et la diffusion d'informations entre des dispositifs mobiles basés sur les propriétés spontanées d'utilisateurs [122].

2.4.2 La représentation du contexte

Les sociologues ont étudié diverses propriétés des réseaux sociaux dont nous décrivons quelques une. Pour une vue d'ensemble plus complète sur les réseaux sociaux et les techniques d'analyse associées, on peut se référer au livre de Wasserman [132].
Milgram [97] a montré que la distance moyenne entre deux Américains est de six étapes et que des réseaux sociaux peuvent être classifiés comme étant un petit monde. Kochen et al. [37] ont expliqué l'influence de la propriété du petit monde des réseaux sociaux sur les contacts. Ceci a été aussi justifié par Granovetter [60] qui a montré qu'un réseau social peut être divisé sous des liens 'forts' et 'faibles'. Ces liens 'forts' sont fermement groupés, tandis que les liens 'faibles' représentent des relations de distance plus longue. Par contre, aucun travail ne traite la gestion de la vie privée lors de la classification des utilisateurs des réseaux sociaux.

Étant donné que les réseaux sociaux sont plus populaires, les chercheurs ont commencé à examiner leurs propriétés. Adamic et al. [3] ont étudié le premier réseau social à l'Université de Stanford et ont constaté que le réseau a des caractéristiques mondiales aussi bien qu'un coefficient de classification. Le groupe de Liben-Nowell [85] a trouvé une corrélation forte entre l'amitié et l'emplacement géographique des utilisateurs, en se basant sur des données figurant dans le "Life Journal". Une autre étude sur deux réseaux sociaux de Yahoo a été faite par Kumar et al. [76]. Ils ont constaté que ces deux réseaux possèdent des composantes fortement connexes et dominantes. Girvan et Newman ont observé que les utilisateurs dans des réseaux sociaux ont tendance à se former des groupes bien serrés [56], évidemment avec un coefficient de classification important. Les travaux cités traitent uniquement la classification basé sur les informations présentes sur le profil de l'utilisateur et ne s'attaquent en aucun cas à la données pruvées dans ces réseaux.

Plus récemment, le groupe d'Ahn [5] a analysé les données complètes du grand réseau social sud coréen Cyworld [33], avec les données de de MyS-

pace et Orkut. La comparaison avec des réseaux différents, d'autre part, est limitée par les échantillons de données de Myspace et Orkut.

D'autres groupes ont examiné le réseau d'activité, ou le modèle d'interactions entre utilisateurs et réseau social. Citons par exemple, le groupe de Wilson [136] qui a étudié le réseau d'activité en se basant sur des échantillons du réseau Facebook. Ils ont constaté que, par contraste avec le réseau social, le réseau d'activité est beaucoup plus rare. Chun et ses collègues [27] ont trouvé des propriétés semblables pour le réseau d'interaction dans CyWorld. Dans ces travaux, ils ne gèrent pas la classification des différents membres du réseau. Ils ne s'intéressent pas au données privées dans les réseaux d'activité.

Dans ce présent travail, nous nous concentrerons seulement sur le réseau social. Mais notre approche et méthodes pourraient être naturellement appliquées au réseau d'activité.

Plusieurs recherches examinent la structure de réseaux de l'information.Par exemple, les graphes de pages Web ou la topologie de routage d'internet. Une étude sur la structure du Web [22] a montré qu'il possède une forme de "nœud papillon", constitué de composante fortement connexe et de groupes de nœuds pouvant atteindre ces composantes ou être atteints par ces derniers. Par contre, il ne s'intéressent pas à la classification de ces nœud ni des données qu'ils partagent.

Faloutsos et al. [50] ont constaté que le degré de la distribution de la topologie de routage d'internet suit une loi de pouvoir. Dans une autre étude faite par Siganos et al [124], il a été démontré que la structure de haut niveau d'internet ressemble à "une méduse".

Kleinberg [74] a montré que les nœuds de degré haut peuvent être observés dans le Web. Ce dernier fonctionne soit en tant que centres (des pages contenant des références utiles sur un sujet) soit en tant que autorités (des pages contenant des informations pertinentes sur un sujet).

Étant donné un graphe de pages Web, Kleinberg a aussi présenté un algorithme [75], qui peut déduire la fonction de pages comme des centres et des autorités. Le célèbre algorithme PageRank [107] utilise la structure Web pour déterminer les pages que l'on considère honorable.

2.4.3 Les modèles des réseaux

Les macros propriétés statistiques d'un réseau permettent d'une part de les diviser dans des catégories et d'autre part de construire les modèles qui peuvent reproduire des réseaux artificiels avec les mêmes caractéristiques qualitatives. Dans ce manuscrit, nous présentons trois catégories du réseau avec leurs modèles correspondants, en se concentrant sur le diamètre, le coefficient de classification et le degré de distribution des graphes.

Le modèle de réseau en boucle de treillis :
Des modèles déterminés et simples de la topologie des graphes sont donnés au treillis, un réseau où les nœuds sont connectés selon des modèles semblables au réseau réguliers. Le réseau en boucle de treillis (ou le treillis unidimensionnel) appartient à cette classe de graphes. Il est produit en arrangeant logiquement des nœuds dans un anneau et connectant chaque nœud avec les " m " nœuds les plus proches dans l'anneau.

Les nœuds dans des réseaux de treillis sont fortement connectés en local et donc le coefficient de classification est haut. D'autre part, le chemin le plus court connectant deux nœuds génériques dans le réseau est en moyenne long. Par conséquent le diamètre est haut.

Le modèle de réseau aléatoire :
Un processus arbitraire de création d'arêtes produit un graphe aléatoire dont le degré de distribution finale dépend de la nature du processus stochastique. Parmi plusieurs processus aléatoires étudiés dans la modélisation des graphes de réseaux [19], le plus connu est celui à la base de l'Erdös Rényi (ER) des graphes [48, 49].

Un graphe ER peut être produit suivant un nombre de nœuds n et une probabilité d'attachement p. Le graphe est produit par un processus sans mémoire attachant chaque paire de nœuds avec la probabilité p.

La probabilité qu'un nœud ait le degré k est donnée par la distribution binomiale :

$$P(k) = \begin{pmatrix} n-1 \\ k \end{pmatrix} p^k (1-p)^{n-1-k}$$

Et comme n augmente, le degré de la distribution à tendance à une distribution de Poisson :

$$P(k) \approx e^{-n.p} \frac{n.p}{k!}$$

Ce qui signifie que la gamme de variabilité de degré de nœud est relativement faible. La connectivité globale du graphe dépend alors des valeurs de n

et de p, à la suite d'un comportement de seuil. Pour $n \cdot p < 1$, le graphe sera probablement débranché, avec une complexité de $O(log(n))$. Alors que si le seuil $n \cdot p = 1$ le graphe aura un composant fortement connecté d'une taille autour de $n^{\frac{2}{3}}$. Finalement, si $n \cdot p > 1$, le graphe aura un immense composant fortement connecté contenant la grande majorité des arêtes. Le composant immense du graphe ER a de très petits diamètres et un faible coefficient de classification. Cela signifie que le réseau peut être traverser dans quelques étapes mais son nœud aura tendance à ne pas former des groupes bien reconnaissables.

Le modèle small-world (Watts-Strogatz) :
Plusieurs réseaux présentent les caractéristiques qui sont entre des graphes aléatoires et réguliers. En particulier, ils exposent un petit diamètre et une classification importante. Par exemple dans des réseaux sociaux, des groupes sont fortement groupés mais deux individus génériques sont sur la moyenne séparée par un faible nombre d'étapes dans le réseau de connaissance.

Pour modéliser ce phénomène de "petit monde", Watts et Strogatz [133] ont proposé un modèle (WS) basé sur une stratégie de raccourci sur des réseaux en boucle de treillis.

Étant donné un graphe d'anneau treillis, chaque arête du nœud i est remplacée par une probabilité p avec une nouvelle arête qui est raccordé avec un critère différent choisi au hasard parmi les nœuds dans $V \backslash \Gamma(i)$. Comme la probabilité p augmente, les macros caractéristiques du graphe changent d'une façon axée sur le seuil. Approximativement, si $p \in [0, 10^{-3})$ le graphe préserve toujours les caractéristiques principales de l'anneau de treillis et si p est plus de 10^{-1} la forme du graphe devient semblable à la configuration ER. Dans l'intervalle $[10^{-3}, 10^{-1}]$ le graphe assume les petites propriétés désirées. Particulièrement, si on le compare avec un graphe aléatoire de même taille, ce petit graphe a une distance moyenne plus courte, mais un plus fort coefficient de classification.

Le modèle scale free (Barabási-Albert) :
Les graphes de watts-Strogatz reproduisent bien la distance moyenne et la classification de beaucoup de réseaux réels. Cependant, ils ne peuvent pas modeler leur distribution, qui est beaucoup plus large dans des réseaux réels.

Pour résoudre ce problème, Barabasi et Albert [12] ont proposé le modèle d'attachement préférentiel de la croissance d'un réseau.

Étant donné un réseau aléatoire initial avec des nœuds m_0 et c un degré supérieur à 1, le modèle d'attachement préférentiel enrichi le graphe en ajoutant un nouveau nœud et en le connectant au nœud $m < m_0$ avec une probabilité proportionnelle au degré des nœuds existants.

Plus spécifiquement, la probabilité de connexion avec un nœud existant i est :

$$p_i = \frac{k_i}{\sum_j k_j}$$

Le "riche devient plus riche", cette stratégie est semblable au processus de croissance des réseaux réels. Par exemple,le graphe de World Wide Web, où ces pages très populaires sont liées par de nouvelles pages avec une probabilité beaucoup plus importante que des pages impopulaires.

La distance moyenne dans le modèle de Barabasi-Albert (BA) croit logarithmiquement avec la taille du réseau :

$$\langle l \rangle = \frac{ln|V|}{ln\ ln|V|}$$

Le coefficient de classification est plus important que le graphe ER avec une taille équivalente. Le degré de distribution de BA suit une loi de puissance ayant la forme suivante :

$$p(k) = c \cdot k^{-\alpha}$$

Les deux modèles BA et WS échouent la modélisation de quelques macros propriétés statistiques des réseaux réels. Néanmoins, de tels modèles ont donné une contribution sans prix à l'étude et à la modélisation des réseaux complexes. Ils ont fourni des abstractions utiles pour la comparaison des propriétés des systèmes différents.

2.5 La théorie de graphe et l'analyse du réseau

2.5.1 Les mesures topologiques des graphes

L'analyse d'un réseau complexe est basée sur un noyau de mesures topologiques qui décrivent les principales propriétés structurelles du graphe [132, 104]. Dans ce qui suit, nous examinons certains d'entre eux.

Degré et la force :

Le nombre d'arêtes adjacents au nœud est le degré du nœud. Il est généralement désigné par la lettre k. Dans le cas des réseaux dirigés, nous pouvons distinguer le degré entrant k_{in} et le degré sortant k_{out}, par le nombre d'arêtes dont le nœud est un successeur ou prédécesseur. Pour les réseaux pondérés, la somme des poids des arêtes adjacentes sur un nœud est la force du nœud. La force entrante et la force sortante sont définies pour les réseaux pondérés.

$$D = \frac{2|E|}{|V|(|V|-1)}$$

Dans les réseaux dirigés, la densité est divisée par le facteur 2 vu que le nombre possible de connexions est le double que le nombre connexion dans les graphes non orientés. Notons que la densité est une mesure qui est strictement dépendante de la taille du réseau. Dans le cas du monde réel, le plus grand correspond au réseau, alors que la plus faible est la densité. Il n'existe pas un moyen standard pour évaluer si un graphe est dense ou épar. Un réseau est considéré comme épar lorsque son degré moyen est beaucoup plus petit que le nombre de nœuds. $\langle k \rangle << |V|$, ou quand les ordres de grandeur du nombre de nœuds et d'arêtes sont approximativement équivalent $|V| \approx |E|$.

Le Diamètre :

La distance $l(i, j)$ entre deux nœuds i, j dans un graphe est donnée par le plus court chemin entre eux. La distance maximale entre toutes les paires possibles de nœuds est le diamètre du graphe. Ce Diamètre est utilisé pour évaluer la largeur du réseau, mais il est très sensible aux valeurs aberrantes. En effet, seulement le plus long des courts chemins peut déterminer un diamètre élevé. Afin d'éviter ce problème, la mesure effective du diamètre d'un graphe est la distance minimale qui inclut les 90% des paires de nœuds [109].

Coefficient de classification :

La structure du voisinage d'un nœud dépend de la nature de la classification des nœuds locaux. La classification, ou transitivité dans les sciences sociales, mesure la tendance des voisins d'un nœud à être relié à l'autre, formant ainsi un réseau de triangles dense. Dans des réseaux sociaux humains, la tendance à la classification importante peut être résumée par la phrase populaire suivante :"tous les amis de mes amis sont mes amis aussi". Quantitativement, la classification d'un nœud i est mesurée par un coefficient $C(i)$ [133] calculé par le rapport entre le nombre de connexions entre i et ses voisins, et le nombre maximal de tels liens. Soit k_i le degré du nœud i et e_i le nombre d'arêtes entre ses voisins. Le coefficient de classification est alors définie comme suit :

$$C(i) = \frac{e_i}{k_i(k_i-1)/2}$$

Cette mesure n'est valable que pour $k_i > 1$. Si $k_i = 1$, on peut considérer que $C(i) = 0$. Le coefficient de classification du réseau, qui mesure le degré global de regroupement du graphe, est défini simplement par la moyenne de classification :

$$\langle C(i) \rangle = \frac{1}{N} \sum_i C(i)$$

Centralité et la centralisation :

Le rôle des nœuds et des arêtes spécifiques à l'égard de la topologie globale du

graphe est l'une des principales réflexions pour caractériser le réseau. L'importance d'un nœud ou une arête est évaluée par des mesures de centralité [54]. Ces derniers peuvent être définies sur plusieurs caractéristiques structurelles du graphe, comme sa connectivité ou sa position par rapport aux autres nœuds. La mesure de la position centrale la plus couramment utilisée est la position centrale de mesure, qui coïncide avec le degré du nœud.

$$C_D(i) = k_i$$

La centralité de proximité se concentre plutôt sur la distance du nœud cible à tous les autres sommets. La centralité d'un nœud i est définie comme suit :

$$C_C(i) = \frac{1}{\sum\limits_{j \neq i} l(i,j)}$$

La centralité de l'intermédiarité est utilisée, pour tenir compte de l'importance des nœuds qui peuvent ne pas être bien relié au reste du réseau, mais agir comme des ponts entre deux ou plusieurs composants qui sont vaguement reliés les uns aux autres. :

$$C_B(i) = \sum_{h \neq j \neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}}$$

Où σ_{hj} est le nombre total des chemins les plus courts du nœud h au nœud j et $\sigma_{hj}(i)$ est le nombre de ces plus courts chemins passant par i. La variation de la valeur des mesures de la centralité sur l'ensemble des nœuds du réseau (et avec des facteurs de normalisation appropriés) est appelée centralisation. Elle permet de mesurer la quantité de l'ensemble du réseau centralisée. Par exemple, la centralisation de degré sera maximale dans un graphe en étoile, où tous les nœuds sont reliés à des arêtes d'un nœud central (et donc la variation relative sur le degré sortant est maximale), et au moins dans un graphe en anneau, où chaque nœud est relié à deux voisins dans un cercle. Le degré de centralisations, la proximité, et l'intermédiarité sont définis respectivement comme :

$$C_D = \frac{\sum_i [C_D^{max} - C_D(i)]}{(|V|-1)(|V|-2)}$$
$$C_C = \frac{\sum_i [C_C^{max} - C_C(i)]}{[(|V|-2)(|V|-1)]/(2|V|-3)}$$
$$C_B = \frac{2\sum_i [C_B^{max} - C_B(i)]}{(|N|-1)^2(|N|-2)}$$

2.5.2 Les outils d'analyse et de virtualisation

La visualisation des réseaux est une phase importante dans l'analyse des systèmes complexes. Une bonne représentation illustrée d'un graphe, peut

mettre en évidence : ses composants structurels les plus importants, diviser logiquement ses régions différentes et indiquer les nœuds les plus centraux et les arêtes sur lequel les flux d'information sont plus fréquent.

Les valeurs de la plupart de la métrique que nous avons définie précédemment peuvent être d'une façon ou d'une autre représentées utilisant des nœuds, des couleurs d'arêtes, des tailles et des dispositions différentes.

Beaucoup d'outils gratuits pour la visualisation de graphe ont été développés au cours de la dernière décennie. Citons par exemple :

1. Pajek [36] (pajek.imfm.si) : Un des premiers outils exploratoires visuels pour la visualisation et l'analyse de petit graphes.

2. Networkbench (nwb.cns.iu.edu) : Un outil pour poser et visualiser des ensembles de données en réseau de domaines différents, à l'appui de la recherche interdisciplinaire.

3. Walrus [99] (www.caida.org/tools/visualization/walrus) : Il permet de visualiser des graphes en se basant sur leur représentation d'arbre.

4. Gephi [14] : Sponsorisé comme "photoshop for graphs" (gephi.org). Il fournit une interface d'utilisateur avancée pour la manipulation visuelle et une API utile pour la visualisation en continu de graphes dynamiques.

5. GUESS [4] (graphexploration.cond.org) :Un outil d'analyse basé sur Gython, un langage spécifique à un domaine qui supporte les opérateurs qui peuvent avoir affaire directement avec des structures de graphe d'une façon efficace et intuitive.

6. GleamViz (www.gleamviz.org) : Spécifiquement conçu pour simuler et visualiser la diffusion de maladies infectieuses à travers le monde.

Plusieurs paquets d'analyse de graphe comme iGraph (igraph.sourceforge.net), networkx networkx.lanl.gov), et R (www.r-project.org) fournissent des outils de visualisation de réseau ou des plug-ins.

2.6 La classification dans les réseaux sociaux

Une communauté est un sous-ensemble d'utilisateurs dans un réseau social qui est plus fermement connecté que le réseau global [103]. Le travail décrit dans cette section essaye de détecter les groupements d'équipes dans un graphe. À un haut niveau, les approches peuvent être divisées en approches globales qui assument la connaissance du graphe entier, ou en approches locales, qui assument uniquement la connaissance détaillée d'une région du réseau. Nous décrivons les études empiriques des réseaux sociaux qui ont cherché la présence de communautés.

2.6.1 la découverte des communautés classiques

La détection communautaire classique a pris l'approche de diviser les sommets d'un réseau social dans des communautés différentes en minimisant le nombre de relation entre des communautés. Cette approche se caractérise par deux algorithmes principaux :

Se partage en deux spectral [111] et l'algorithme Kernighan-Lin [72]. Ces deux algorithmes divisent le graphe en deux communautés possible, ensuite ils les subdivisent jusqu'au atteindre le nombre d'utilisateur de communautés indiqué. Cependant, les deux algorithmes exigent initialement que l'utilisateur spécifie les tailles des deux communautés, aussi bien que le nombre final de communautés désirées. Cette découverte de communautés ne traitent pas la propagation de données entre les membres de chaque communauté.

2.6.2 la découverte des communautés globale

Les premiers algorithmes de détection communautaires ont été proposé par Girvan et Newman [103] . Ils calculent le "plus important" lien dans le réseau, puis ils l'enlèvent. Cette étape se répète jusqu'à ce que le graphe du réseau social devienne divisé. A partir de ce moment, on peut considérer les divisions diverses comme des communautés. En continuant d'exécuter l'algorithme sur les divisions différentes, on produira des communautés encore meilleures, jusqu'à ce que tous les liens soient enlevés du réseau. Ce travail ne traite pas la diffusion des données privées dans le réseau social et ne gère pas la collaboration entre les différentes communautés.

Il est donc clair que la sélection du lien le plus important est intégrale au fonctionnement de l'algorithme. Un bon indicateur de l'importance peut partitionner rapidement le graphe dans ses différentes communautés, tandis qu'une mauvaise métrique peut débrancher les nœuds un par un et produire des partitions dégénérées.

Ces mêmes chercheurs ont suggéré l'utilisation de la métrique de " betweenness centrality ". Par exemple, un réseau social est divisé en communautés fortement connectées, la métrique " betweenness centrality " cherchera des liens qui connectent les communautés. Étant donné que les communautés sont plus denses que le graphe dans l'ensemble, ces liens de rapprochement auront naturellement une centralité plus importante. Une fois qu'ils sont enlevés du graphe, la structure de la communauté sous-jacente apparaît. Par contre, ils ne s'intéressent pas à la propagation des données entre les communautés ni à la vie privée des membres de la communauté.

[105] Une approche alternée, basée sur l'optimisation de modularité a été proposé par la suite. L'algorithme commence par chaque sommet dans une communauté séparée et fusionne les paires de communautés, choisissant à chaque étape la paire qui rapporterait l'augmentation la plus haute ou la diminution la plus petite dans la modularité. L'approche proposée ne gèrent pas l'usage des données privées entre les différentes communautés et ne traite pas la propagation des informations dans le réseau.

Clauset et al. [30] ont présenté une variante plus rapide de cet algorithme par la nouvelle optimisation des opérations. Pour cela, ils ont utilisé des structures de données plus efficaces. Ces améliorations de vitesse de calcul sont importantes, étant donné que la durée de l'algorithme original interdit son utilisation sur des graphes avec plus de quelques milliers de liens. Par contre, ils n'ont pas proposé une approche combinant la recherche de communauté avec la propagation des données privées et la collaboration entre les différent groupe.

D'autres approches ont exploré la découverte multiple et le chevauchement de structures communautaires d'une perspective globale. Ceci est par contraste avec les approches précédemment discutées, qui ont été seulement concernées par la découverte de la meilleure façon de diviser les nœuds. Les même approches ont été proposé par le groupe de Palla en 2005 [108]. Ils ont utilisé des k-cliques pour trouver des communautés de chevauchement aux échelles différentes. Un autre chercheur [15, 16] a proposé une approche semblable pour trouver des communautés de chevauchement par la recherche

des nœuds fortement connectés du graphe. Ces travaux ne gèrent pas la probabilité de propagation de données entre les équipes qui génèrent des fuites de données dans un processus de crowdsourcing.

Du et al. [44] ont présenté un algorithme pour détecter des communautés dans des réseaux sociaux à grande échelle en considérant la nature de chevauchement de communautés. Une autre approche séparée a été proposé par l'équipe de Li [84]. Cette approche permet de détecter des communautés imbriquées et de les classifier en se basant sur la similarité du texte. Par contre, ils ne s'intéressent pas à la propagation des données privées dans les réseaux sociaux. Ils ne traitent pas les problèmes de vie privée dans les communautés pour la collaboration, ni entre les communautés pour mesurer la fuite de données.

2.6.3 La découverte des communautés locales

Un inconvénient potentiel des approches globales à la détection communautaire est l'obligation de connaître la structure du graphe entier ; ceci est souvent a un coût prohibitif (tant que les graphes réalistes sont extrêmement grands) ou difficilement obtenu (par exemple, le graphe de Facebook). Un certain nombre de chercheurs ont observé des approches locales à la détection de communautés, qui utilisent seulement la connaissance locale pour construire une communauté autour d'un ensemble de nœuds source.

Par contraste avec les approches globales, des approches locales ont le potentiel pour être significativement plus évolutives et applicables à de plus grands graphes, aussi bien que les graphes qui ne sont pas complètement visibles en raison des restrictions de vie privée. De plus, Ces approches locales à la détection communautaire tiennent aussi à détecter des structures communautaires multiples. Ils tiennent compte de la décentralisation naturelle, vu que le calcul peut être trivialement partagé et parallèle.

Clauset [29] a proposé une des premières approches locales à la détection communautaire. Elle est basée sur la construction d'une communauté autour d'un nœud source. L'algorithme crée une communauté en ajoutant des sommets un à un, choisissant le sommet, à chaque étape, qui maximise le ratio des arrêtes inter communautaires aux arrêtes intra communautaires pour les nœuds de la communauté. Ainsi, cet algorithme essaye de créer une communauté forte en choisissant les nœuds qui ont beaucoup de liens à l'intérieur de la communauté. Récemment, Wakita et al. [130] ont proposé une modifica-

tion de l'algorithme de Clauset, qui devient capable d'identifier les communautés dans des réseaux sociaux comptant jusqu'à 5 millions d'utilisateurs. Cependant, leur travail n'a pas validé la structure communautaire déduite du réseau. Par contre, ces travaux ne s'intéressent pas à la propagation de données entre les communautés.

Le groupe de Luo [89] ont proposé un algorithme semblable à celui de Clauset en ajoutant et enlevant des nœuds, jusqu'à l'obtention d'une mauvaise répartition.

Il est important de noter qu'aucun de ces algorithmes n'a été validé sur un réseau social à grande échelle. Ceci est du principalement au manque de disponibilité de données. Typiquement, les algorithmes sont évalués sur des graphes synthétiquement produits, des réseaux de recommandation de produit, ou de très petits réseaux sociaux comme le club de karaté de Zachary [142], contenant 34 membres. Ainsi, qu'on n'a pas la connaissance s'ils peuvent détecter des communautés dans des réseaux sociaux en ligne ou non.

2.6.4 Observations sur les communautés

Des études ont été consacrées aux structures communautaires qui existent dans des réseaux sociaux en ligne. Citons parmi eux, les travaux de Nazir et al. [100]. Ils ont présenté une étude de mesure à grande échelle des caractéristiques d'utilisation des applications de Facebook. Ils ont lancé quelques applications Facebook et ont utilisé les données rassemblées de ces applications. Ils ont caractérisé la charge de travail, la structure d'interactions de l'utilisateur et l'existence de communautés dans le réseau. Leurs résultats, cependant, ne montre pas une image complète du réseau Facebook comme ils peuvent seulement rassembler des données sur les utilisateurs qui ont installé une de leurs applications. Ils ne traitent pas la collaboration entre les membres du réseau social. Ils ne s'intéressent pas à la classification des utilisateurs ni aux fuites de données entre les différentes équipes.

2.7 Les heuristiques dans les réseaux sociaux

L'équipe de Schahram Dustdar [43] présente une approche pour la composition d'équipe utilisant deux heuristiques, les algorithmes génétiques et le recuit simulé, sur les environnements des réseaux en général, ils utilisent les recommandations implicites de collaborateurs partenaires pour supporter les

réseaux connectés. Ils identifient l'assemblage d'équipe basé sur l'expertise des utilisateurs dans le réseau. Ce travail, ne gère pas les fuites de données entre les équipes compétitives dans le réseaux social. Ils ne traitent pas la propagation des données entre les différents membres du réseau social pour mesurer la fuite. Leur système propose uniquement une équipe permettant la résolution d'une requête en se basant uniquement sur les compétences. Les problématiques de la vie privée ne sont pas mentionnées dans le papier.

Wang et al. [131] s'intéressent au problème d'enregistrements dupliqués. Ils proposent un système hybride homme-machine pour la résolution de ce problème. Ils utilisent des techniques à base de machines dans le but de rejeter les enregistrements qui semblent très différents et interrogent les êtres humains pour vérifier uniquement les paires restantes. Ils combinent l'efficacité des machines en fonction des approches avec la qualité de la réponse obtenue. Ils utilisent l'approche heuristique pour lutter contre le problème du temps de traitement. Leur travail ne permet pas un travail collaboratif mais uniquement un traitement individuel de la requête. La question de la vie privée n'est abordée également.

2.8 Les Soft Constraint Satisfaction Problem

Le problème de satisfaction de contraintes (CSP) implique un ensemble fini de variables, chaque variable prend une valeur dans un domaine fini. Les valeurs sont liées par des contraintes qui imposent des restrictions sur les combinaisons de valeurs. Or, la majorité des problèmes de la vie actuelle ne permettent pas la satisfaction de toutes les contraintes imposées. Les SCSP (Soft constraint satisfaction problem) est une technique permettant de relaxer les contraintes CSP afin d'en satisfaire un maximum. Cette technique nous permet de retrouver des solutions même en l'absence de la satisfaction de la totalité des contraintes.

Stefano Bistarelli et al. [18] rappellent l'idée de résolution de CSP basée sur le semiring. Ils ont présenté un Framework de recherche locale pour aborder des problèmes SCSP. Leur Framework est basé sur la structure de semiring et le schéma de recherche locale pour SCSP qui consiste en trois composants principaux : amortissement, contraction et moteur de recherche locale.

Leur solution de recherche de processus peut être exprimée par le problème de minimisation suivant :

$$minL(n, p, g) = \sum_{i=1}^{m} p_i g_i$$

Où " n " est une attribution, " p " est la pénalité et " g " est une contrainte.

Ils implémentent leur Framework en se basant sur l'E-Genet, qui permet une transformation naturelle du problème avec moins de surcharge.

Alexander Topchy et al. [78]présentent le concept de classification en utilisant des groupes de contraintes basées sur des paramètres données par l'utilisateur et la création d'un modèle de paramétrage qui peut être évalués en maximisant la probabilité de la fonction de similarité des données.

Chaque donnée y_i est assigné au groupe q w_l basé sur quelques critères mentionnés par l'utilisateur et d'autres évaluées par L'algorithme espérance-maximisation.
Le modèle de y_i est spécifié comme suit :

$$P(w_l = j) = \alpha_j, l \in \{1, ..., L\}, j \in \{1, ..., K\}$$

$$P(v_i = j) = \lambda_{il}, i \in \{1, ..., N\}, j \in \{1, ..., L\}$$

$$P(z_i = j | v_i, w_l, ...w_L) =$$

$$\begin{cases} \alpha_j & if \quad v_i = 0 \\ \delta_{w_l,j} & if \quad v_i = l \end{cases}$$

D'abord, l'étiquette de groupe L w_l est produite selon les probabilités des composantes α_j. Pour chaque $i \in \{1, ..., N\}$, ils produisent v_i avec les probabilités λ_{il}. Le résultat détermine comment z_i obtient sa valeur : si v_i est entre 1 et L, z_i est mis à w_l ; autrement, z_i est produit indépendamment selon α_j.

Prodip Hore et al. [112] présentent leur Framework de classification de grand nombre de données basé sur l'algorithme k-mean de classification utilisant la distance euclidien pour le calcul de mesures.

L'idée de la classification d'un grand nombre de données est l'utilisation de l'algorithme de K-means pour assigner des données aux différents groupes et ensuite l'utilisation de la distance euclidienne pour calculer la distance entre chaque groupe et fusionner les solutions de classification parallèles.

Ils utilisent la formulation suivante pour calculer la distance entre les centroid :

$$D^n(i) = 1/2 \sum_{j=1}^{r} d(c_i^n, c_j^n)_{i!=j}$$

La fonction objective doit globalement optimiser l'assignent ψ^* de toutes les familles possibles f d'attributions de centroid aux chaînes de consensus k.
Le coût de chaîne de consensus est exprimé comme suit :

$$cost(consensus_chain(n)) = \sum_{i=1}^{r} D^n(i)$$

La somme du coût de chaîne de consensus est reformulée comme suit :

$$\psi = \sum_{n=1}^{k} cost(consensus_chain(n)) =$$

La fonction objective est représentée comme suit :

$$\psi^* = arg_{\psi \subset f} min \{\psi\}$$

Francesca Rossi et son équipe [116] passent en revue différentes approches de Soft Constraint et énumèrent un certain concept pour modéliser les préférences quantitatives.

Ils expliquent la programmation de contrainte, les contraintes et le problème de satisfaction de contrainte.

Dans un deuxième temps, ils passent en revue une certaine méthode modélisant des préférences quantitatives via des Soft Constraints comme les contraintes floues qui emploient une approche pessimiste qui peut être utile ou même nécessaire dans des applications critiques. Un autre formalisme à base de contrainte générale pour modéliser des préférences est le formalisme basé sur le Semiring, qui englobe la plupart des extensions précédentes dans le but de fournir un environnement simple où les propriétés peuvent être prouvées une seule fois et ensuite héritées par toute l'instance.

SCSPs, où les contraintes ont des niveaux différents de satisfaisabilité, qui sont, totalement ou partiellement, classées selon la structure du Semiring.

Ils passent en revue de différentes sortes de préférences comme les préférences Bipolaires indiquant la possibilité de déclaration des deux degrés de satisfaction et de rejet, ou les préférences qualitatives qui sont représentées par les réseaux de préférences Conditionnels.

Ils expliquent les préférences temporelles en détaillant les variables représentent des événements instantanés, tel que "‘ quand un avion décolle" ' ou des intervalles temporels, comme "‘ la durée du vol" '. Les contraintes temporelles permettent de mettre des restrictions temporelles quand un événement devrait se produire.

Finalement, ils passent en revue la manière de surmonter la complexité des soft constraints utilisant l'abstraction, la génération d'explication, apprentissage et l'élicitation.

Louise Leenen et al. [80] expliquent le SCSP et sa possibilité de relaxation en le détaillant.
Ils définissent la relaxion d'un problème de contraintes utilisant une contrainte affaiblie.

$c_j = \langle def_j^p, conn_j \rangle$ appelé c_i-weakened constraint de $c_i = \langle def_i^p, conn_i \rangle$
Si :
$con_i = con_j$
pour chaque tuples t, $def_i^p(t) \leq sdef_j^p(t)$
pour chaque deux tuples t1 et t2, si $def_i^p(t1) \leq s_p def_i^p(t2)$, alors $def_j^p(t1) \leq s_p def_j^p(t2)$

W_c est l'ensemble des c-weakened contraintes. $wdef_c^d$ est une fonction qui assigne la valeur du c-semiring de l'ensemble c-semiring S_d pour chaque contrainte c-weakened.

Louise Leenen et al. [79] expliquent les SCSP basés sur le c-semiring et définis par le tuple $S = \langle A, +, x, 0, 1 \rangle$, $CS = \langle S_p, D, V \rangle$.

Ils expliquent le MAX-SAT et le MAX-SAT pondéré basé sur la fonction objective définie comme $f(x) = \sum_{i=1}^{m} w_i.I(C_i)$ où I est un ensemble de variables, w_i est le poids assigné à chaque clause et $I(C_i)$ égale à 1 si et seulement si la clause C_i est satisfaite, 0 sinon.

Ils présentent le semiring pondéré MAX-SAT en indiquant un nouveau théorème utilisant une attribution de vérité α. Ils présentent l'algorithme Gsat qui essaye de maximiser le nombre de clauses satisfaites et l'algorithme BnB pour WS-MAX-SAT.

Chapitre 3

Calcul de propagation de données privées dans les réseaux sociaux

3.1 Introduction

Les réseaux sociaux sont devenus récemment un moyen populaire de partage et de diffusion des informations. L'utilisation des sites comme MySpace , Flickr , Facebook crée des réseaux d'amis . Les utilisateurs peuvent partager et diffuser des contenus public et privé à une échelle massive. Chaque minute, dix heures de vidéo sont téléchargées sur YouTube [141] ; Flickr contient plus de deux milliards de photos [52]. En raison de leur grande popularité, ces sites ont été exploités en tant que plate-forme d'espionnage d'informations privées. Par exemple, les grandes sociétés de recrutement cherchent des informations sur les candidats sur Facebook. La police utilisent aussi ces réseaux sociaux pour avoir un maximum d'informations sur les suspects. Pareil pour les pirates, ils exploitent ces informations publiques afin de préparer des attaques ciblées, tout en espérant atteindre des millions d'internautes. Les caractéristiques de la propagation des informations dans les réseaux sociaux ainsi que les mécanismes par lesquels les informations sont échangées, ne sont pas bien comprises. L'un des traits distinctifs de ces réseaux est le potentiel de diffusion de l'information sur les liens sociaux, à savoir, des propagations

informations entre amis. Il est largement admis que les échanges, connu par les échanges " bouche-à-oreille", peuvent étendre rapidement des données privées et des informations publiques dans le réseau.

3.2 Vue d'ensemble

Un réseau social est un ensemble de relations sociales entre les membres. A travers ces relations, les membres transmettent des données privées dans le réseau social. Ces relations nous permettent de calculer la probabilité de propagation des données entre les amis directement connecté, et ainsi, la probabilité des fuites de données.Il n'est pas suffissent de connaître les relations directes entre les membres, mais aussi les propagations indirections entre les différents membres du réseau. Toutefois, pour découvrir des équipes compétitives sans fuite de données, il est nécessaire de connaître toutes propagations de données possibles de "ami - à - ami" dans l'ensemble du réseau. Par exemple, dans la figure 3.1 David partage 50% de ses données avec Mickael et ce dernier partage 90 % de ses données avec Alice. Ces taux sont considérés comme la probabilité de propagations des données entre amis directement connectés notées par $p_{(David;Mickael)} = 0.9$ et $p_{(Mickael;Alice)} = 0.5$, respectivement.

Cependant, les propagations des données à des amis indirects sont implicites et dépendent de la probabilité de propagation des données entre amis directs. Dans la figure 3.1, la probabilité de propagation de données à partir de David au membre indirect Alice est calculée par l' expression suivante :

$$p_{(David;Alice)} = p_{(David;Mickael)} \times p_{(Mickael;Alice)} = 0.5 \times 0.9 = 0.45.$$

Tous les chemins possibles dans le réseau social doivent être explorés pour calculer la propagation des données entre deux membres. Il s'agit d'un calcul assez complexe dans les réseaux sociaux réels, représenté par un graphe.

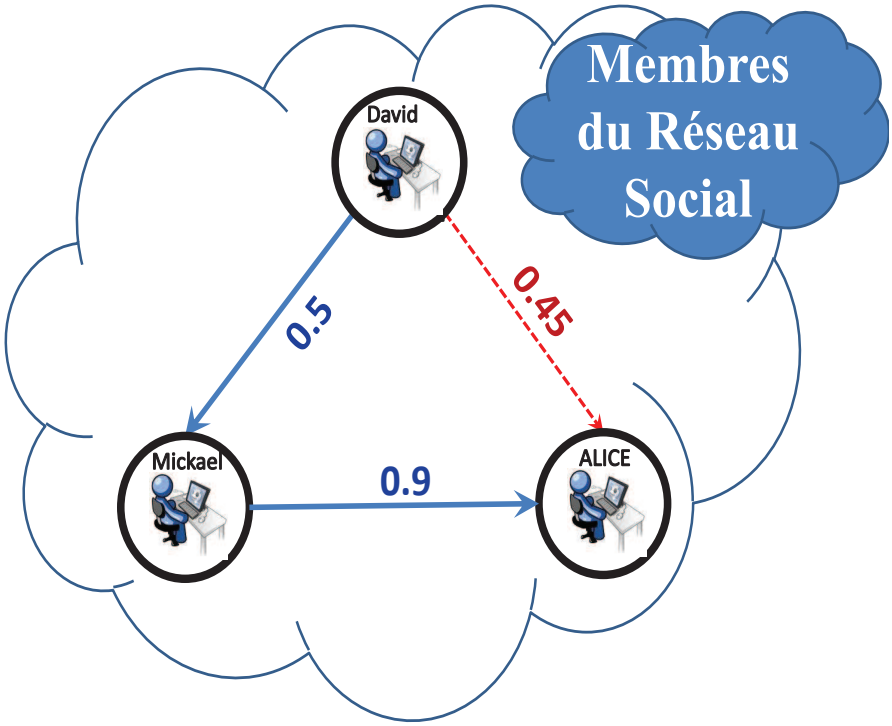

FIGURE 3.1 – Découverte des propagations cachées

3.3 La définition d'un graphe et les termes de base

La théorie des graphes est une branche des mathématiques dont l'origine peut être retracée à Euler. Elle vise à étudier des structures formées par les relations entre les objets. Selon la définition donnée par Bondy et Murthy [40], un graphe est un triplet $G = (V, E, \varphi)$ composé par un ensemble de sommets (ou nœuds) V, un ensemble d'arêtes (ou liens) E, et une fonction d'incidence φ qui associe une paire de nœuds avec chaque arête. Deux nœuds mis en relation par une arête sont communément appelés voisins. L'ensemble des voisins d'un nœud spécifique i peuvent être désignées par $\Gamma(i)$.

Les graphes peuvent être très intuitivement établis avec des cercles pour représenter les sommets et des lignes, reliant les cercles, pour représenter les arêtes. Si les paires de nœuds dans le domaine de φ sont non ordonnés ;

$\varphi(e) = (i, j) \leftrightarrow \varphi(e) = (j, i)$), le graphe est donc soit non orienté, ou symétrique, soit dirigé. Dans le cas des graphes orienté, les arêtes peuvent être, également, appelées arcs. Le nœud à partir duquel provient l'arc, est appelé prédécesseur et successeur de l'autre.

Une arête peut connecter les nœuds qui ne sont pas nécessairement distincts. D'où, elle peut être une boucle qui commence et se termine par le même nœud. Dans le cas où plusieurs carte d'arête sont découvert sur le même paire de nœuds, il s'agit donc d' un multi-graphe. Au contraire, un simple graphe ne contient pas de boucles et d'arêtes multiples. En se concentrant sur les graphes sans arêtes multiples, la définition de graphe peut être simplifiée à une paire $G = (V, E)$.

Les nœuds et les arêtes peuvent être annotées avec des attributs. En particulier, un graphe pondéré a des poids numériques associées aux arêtes ou, plus précisément, il existe une fonction $W : E \to R$).

Quand l'ensemble des prédécesseurs dans la fonction d'incidence a une intersection vide avec l'ensemble des successeurs, c'est à dire,$\varphi(e) = (i, j) \to i \in V_1, j \in V_2, V_1 \cap V_2 = \varnothing \wedge V_1 \cup V_2 = V, \forall e \in E$, le graphe est donc une bipartite. On cite d'autres exemples de structures bien connues comme, le graphe vide ($V = \varnothing, E = \varnothing$), le graphe trivial (avec un seul nœud), et le graphe complet où chaque paire de nœuds possible est reliée par une arête.

Semblablement à des ensembles, un graphe comprend des sous-graphes. On définit, un sous-graphe $H = (E_H, V_H, \varphi_H)$ du graphe $G = (E_G, V_G, \varphi_G)$ par les relations de sous-ensembles $V_H \subseteq V_G, E_H \subseteq E_G$. Ce sous graphe est provoqué par un ensemble de nœud $V' \subseteq V$ s'il contient tous les nœuds de V' et toutes les arêtes de G entre les nœuds de V'.

Les composants connectés dans un graphe orienté sont appelés composantes fortement connexes, alors que ceux faiblement connectés sont les éléments qui ne respectent pas la direction des arcs. D'un point de vue mathématique, un graphe peut être défini par une matrice d'adjacence $|V| \times |V|$, où chaque entrée (i, j) contient une valeur binaire indiquant la présence d'un lien entre les nœuds i et j ou le poids de la connexion en cas de graphes pondérés. La matrice d'adjacence est symétrique pour les arêtes non orientés. La diagonale est indéfinie pour les graphes simples.

3.3.1 Un modèle de propagation de données

La matrice de propagation donne uniquement la probabilité de propagation de données entre les amis directs. Par contre, il est insuffisant de calculer la probabilité de propagation des données entre les amis directs pour les raisons suivantes :

1. La probabilité de propagation entre deux amis direct n'est pas optimale : suivant des différents chemins dans le graphe du réseau social, nous pouvons calculer différentes probabilités de propagation des données entre deux membres. Dans la figure 3.2, la probabilité de propagation directe de *Alice* à *David* est de ($p_{Alice,David} = 0$). Par contre, passant par $Mickael$, les données de *Alice* peuvent être propagées à *David* avec une probabilité de $0.9 \times 0,5 = 0.4$.
2. La propagation aux amis indirects n'est pas donnée dans la matrice de propagation : La probabilité de partage de données avec les amis indirects est de zéro dans la matrice de propagation. Ceci est du au partage des membres de leurs données qu' avec les amis directs. Par contre, les données peuvent se propager d'un ami direct à un ami indirect. Dans la matrice de propagation de la figure 3.2, la probabilité que les données du membre *George* seront propagé à son ami indirect *David* est de zéro. *George* n'a aucune relation d'amitié avec eux. Par contre, les données peuvent se propager de *George* à *David* à travers *Bob* avec une probabilité de $0.7 \times 0.1 = 0.07$.
3. La propagation aux amis indirects est difficile à calculer : par exemple, quelle est la probabilité que les données de *Alice* peuvent se propager à *David* (la probabilité de la flèche rouge en pointillés dans la figure 3.2) ? Pour calculer cette probabilité, on a besoin d'explorer tous les chemins permettant la propagation de *Alice* vers *David*. Chacun permet de calculer une probabilité de propagation. Cette dernière correspond à la probabilité de propagation de la données par l'ensemble des amis directs. Le chemin ($Alice, Mickael, David$) indiqué avec des flèches vertes en pointillés sur la figure 3.2 permet de noter la propagation des données de *Alice* à *David* avec un maximum de probabilité $0.9 \times 0,5 = 0.45$. Cependant, il est difficile de calculer cette probabilité vu que les réseaux sociaux réels sont complexes.

La matrice de propagation du réseau social de la figure 3.2 est donnée comme suit :

$$
\begin{array}{c}
 \\
Bob \\
Alice \\
John \\
David \\
Mickael \\
George
\end{array}
\begin{array}{c}
\begin{array}{cccccc}
Bob & Alice & John & David & Mickael & George
\end{array} \\
\left(\begin{array}{cccccc}
1 & 0 & 0 & 0.1 & 0.1 & 0.7 \\
0 & 1 & 0 & 0 & 0.9 & 0 \\
0 & 0 & 1 & 0.2 & 0 & 0 \\
0 & 0 & 0 & 1 & 0,5 & 0 \\
0 & 0.9 & 0.3 & 0.5 & 1 & 0.2 \\
0.7 & 0.3 & 0.1 & 0 & 0 & 1
\end{array}\right)
\end{array}
$$

La fonction d'énergie utilisée pour le calcul de la probabilité maximale est définie comme suit :

Définition 1 *la fonction d'énergie p_i d'un membre m_i est la probabilité que les données se propagent au membre m_i. Dans notre modèle, les données sont propagées suivant la propriété de Markov :*

$$p_i = 1 - (\prod_{m_k \in N_{m_i}} (p_k \times (1 - p_{ki})))$$

où,

- N_{m_i} est l'ensemble des amis directs de m_i.
- p_k est la fonction d'énergie de m_k.
- p_{ki} est la probabilité de propagation des données de m_k à m_i.

La fonction d'énergie se traduit comme suit :

La probabilité qu'un membre m_1 reçoit les données privées de ses amis directs m_2 ou m_3 ou ... ou m_n est la probabilité que les données ne se propagent pas de m_2 et m_3 et ... et m_n.

3.3.2 Un modèle de partage de données à base de graphes

Nous modélisons les relations dans le réseau social sans se focaliser sur leur nature. Nous avons seulement besoin de raisonner sur la quantité des données propagées entre les membres du réseau social. Ainsi, nous notons toutes sortes de relations par la relation d'amitié. Nous modélisons le réseau comme un graphe orienté étiqueté $G\langle M, A, P\rangle$ où,

- $M = \{m_i\}$: est un ensemble de noeuds où chaque noeud représente un membre du réseau social.

– $A = \{a_{ij} = (m_i, m_j)/(m_i, m_j) \in M\}$: est un ensemble d'arêtes où chaque arête représente une relation entre deux membres.
– $P = \{p_{ij}/\forall i, j\ p_{ij} \in [0, 1]\}$: est un ensemble d'étiquettes où chaque étiquette p_{ij} de l'arête a_{ij} représente le taux (probabilité) de données partagées par le membre $m_i \in M$ vers $m_j \in M$.

Définition 2 *(Matrice de propagation) La matrice de propagation d'un réseau social* $G\langle M, A, P\rangle$ *est une matrice qui donne la probabilité* p_{ij} *de propagation de données à chaque couple de membre ayant une relation d'amitié directe* (m_i, m_j) *:*

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ \dots \\ m_n \end{array} \begin{array}{c} \begin{array}{ccccc} m_1 & m_2 & m_3 & \dots & m_n \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} & \dots & p_{1n} \\ p_{21} & p_{22} & p_{23} & \dots & p_{2n} \\ p_{31} & p_{32} & p_{33} & \dots & p_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ p_{n1} & p_{n2} & p_{n3} & \dots & p_{nn} \end{pmatrix} \end{array}$$

où

$$p_{ij} = \begin{cases} \frac{\sum q^n * (1+sim(m_j, q^n))}{MAX(\sum q^n_{m_i}, \sum q^n * (1+sim(m_j, q^n)))} & si\ (m_i, m_j) \in A \\ 1 & pour\ i = j \\ 0 & sinon \end{cases}$$

où

q^n *= les données de* m_i *partagées avec* m_j *de type* n*.*

$sim(m_j, q^n)$ *= la similarité entre le profil de l'utilisateur* m_j *et la description des données* q^n *avec* $Sim(m_j, q^n) \in [0, 1]$*.*

$q^n_{m_i}$ *= les données de* m_i *de type* n*.*

Chaque membre dans le réseau social détient la totalité de ses données. La probabilité de propagation des informations du membre qui les détient vers lui même est donc de 1.

L'absence de relation directe entre deux membre dans le réseau social se traduit par une propagation de 0.

Dans le modèle de graphe donné, la relation *ami* est représentée en utilisant l'arête A étiqueté avec la probabilité réelle de partage de données P. On considère l'exemple de réseau social présenté dans la figure 3.2, où les membres du crowdsourcing peuvent partager leurs données avec les probabilités suivantes :

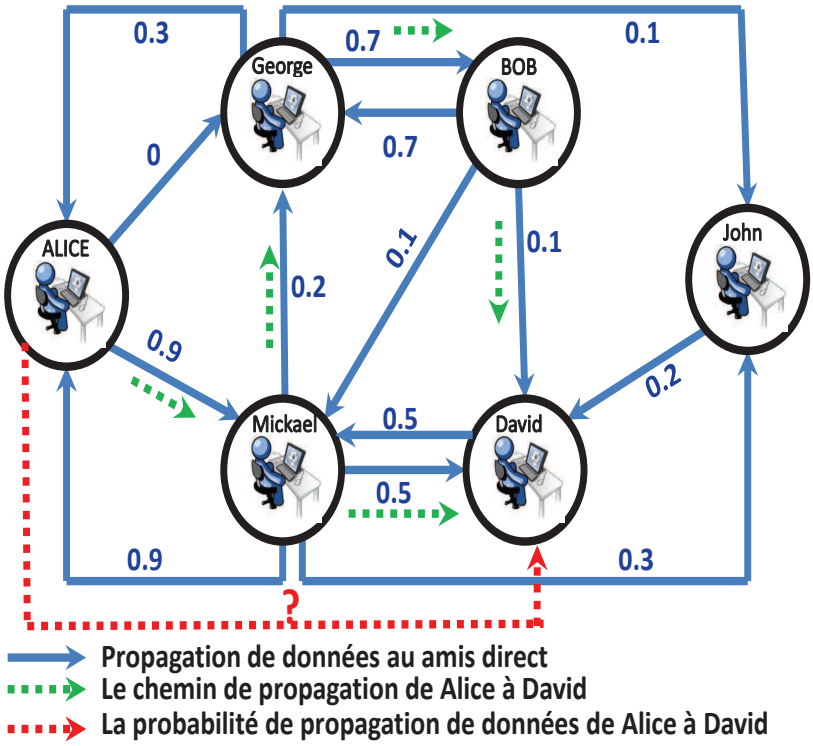

FIGURE 3.2 – Les interactions des membres du crowd dans un réseau social

Nous présentons notre approche de la propagation de données entre les membres.

Étant donné un ensemble de données appartenant à un membre, quelle est la possibilité de propagation de ces données dans le réseau social ?

Dans les réseaux sociaux, les données se propagent d'un ami à un ami. Cela veut dire que, les crowder partagent les données avec leurs amis et, ensuite, chaque ami partage les données avec ses amis et ainsi de suite.

La matrice de propagation possède les propriétés suivantes :
- $p_{ii} = 1$, qui signifie que le membre m_i ne perd pas la propriété des données à cause du partage.
- $\sum_{k\in[1,n]}(p_{ik}) \neq 1$, les données peuvent être propagées à plusieurs membres en même temps.
- $p_{ij} \neq p_{ji}$, signifie que le membre m_i peut partager les donnes avec son ami m_j avec une quantité différente .
- $(m_i, m_j) \in A \nRightarrow p_{ij} \neq 0$, cela signifie que les membres ne partagent pas nécessairement leurs données avec leurs amis.
- $(m_i, m_j) \notin A \Rightarrow p_{ij} = 0$, pour exprimer que la propagation des données aux amis indirect est inconnue.

3.4 Calcul des propagations basée sur le modèle de Markov

3.4.1 Définitions et propriétés

Une chaîne de Markov est constituée d'un ensemble dénombrable S éventuellement fini, appelé l'espace d'état, avec une famille dénombrable de variables aléatoires $X_0, X_1, X_2, \cdots$ avec des valeurs de S tels que :

Définition 3 *(Les chaînes de Markov)*

$P[X_{l+1} = s|X_l = s_l, X_{l-1} = s_{l-1}, ...X_0 = s0] = P[Xl + 1 = s|X_l = s_1]$.

Nous nous référons à cette équation fondamentale comme la propriété de Markov. Les variables aléatoires $X_0, X_1, X_2, \cdots$ sont dépendantes. Les chaînes de Markov sont parmi les quelques séquences de variables aléatoires dépendantes qui sont de caractère général et ont été étudiés avec succès avec des résultats profonds sur leur comportement.

On pense souvent que l'indice l de la variable aléatoire X_l représente le temps (discrètement), et les variables aléatoires représentent l'évolution d'un système dont le comportement est connu seulement de manière probabiliste. La propriété de Markov exprimant l'hypothèse de la connaissance du présent ($i.e., X_l = s_l$) se rapporte à des prévisions sur l'avenir du système. Cependant, toute information supplémentaire sur le passé ($X_j = s_j, j \leq l - 1$) est imprévisible.

Étant donné que l'espace d'état est dénombrable, ou même fini, il est ordinaire dans certain cas d'utiliser : les entiers $\mathbb{Z}$ ou un sous-ensemble tel que $\mathbb{Z}_+$ (entiers non négatifs), les nombres naturels $N = 1, 2, 3, \cdots$ ou $0, 1, 2, \cdots, m$ comme l'espace d'état. La chaîne spécifique de Markov détermine, souvent, la notation naturelle pour l'espace d'état. Dans le cas général où aucune chaîne de Markov spécifique n'est choisie, nous utilisons souvent N ou $\mathbb{Z}_+$ comme l'espace d'état. L'équation que nous proposons est la suivante :

$$P_{ij}^{l,l+1} = P[X_{l+1} = j | X_l = i]$$

Quand l est fixe (éventuellement infini) la matrice $P_l = (P_{ij}^{l,l+1})$ est appelée la matrice de transition (à l'instant l). Sauf indication contraire, toutes les chaînes de Markov prise en compte dans cette thèse sont homogènes dans le temps. L'indice l est donc omis. Nous représentons simplement la matrice de transition comme $P = (P_{ij})$. P est appelée la matrice de transition.

Partant de cette définition formelle, la probabilité qu'un membre donné obtient des données, dépend de la probabilité de l'obtention uniquement à partir de ses amis directs.

3.4.2 Algorithme de découverte des relations cachées (HDPD) utilisant MapReduce

MapReduce est un framework qui permet d'écrire des programmes qui traitent de grandes quantités de données non structurées d'une façon parallèle et distribués sur un groupe de processeurs ou d'ordinateurs autonomes.

Le framework est divisé en deux parties :
- Map, une fonction qui traite les travaux sur différents noeuds du cluster distribué.
- Reduce, une autre fonction qui rassemble le travail et résout les résultats en une seule valeur.

Dans le framework MapReduce chaque noeud du cluster devrait remonter les mises à jour des tâches ainsi que leurs états. Si un noeud reste silencieux pendant un intervalle donné, un noeud maître ré-attribue la tâche à d'autres noeuds.

Étant donné que l'objectif de l'approche est d'estimer le risque maximal de propagation de données, la fonction qui nous estime la probabilité maximale est donc celle la plus appropriée. L'algorithme proposé nommé HDPD

permet de calculer la probabilité de propagation des données p_{ij} de la matrice de propagation. Cela se fait en utilisant l'approche de MapReduce pour réduire la complexité. L'algorithme HDPD calcule en parallèle les lignes de la matrice de propagation. Son traitement se compose en deux étapes :

1. Etape 1 : La fonction Map
   La Map récupère en entrée une liste de membres, pour chaque membre, elle calcule la probabilité de propagation des données d'un utilisateur m_i vers tous les autres en utilisant la matrice du réseau social à partir du contexte. Cela veut dire qu'elle calcule les probabilités $p_{ik}, (i \neq k)$ de la ligne i (membre m_i) dans la matrice de propagation en utilisant la fonction d'énergie définie. Par conséquent, l'algorithme peut utiliser autant de Map que de membres.
2. Etape 2 : La fonction Reduce
   Le reducer génère la matrice de propagation par l'agrégation des résultats des Maps.

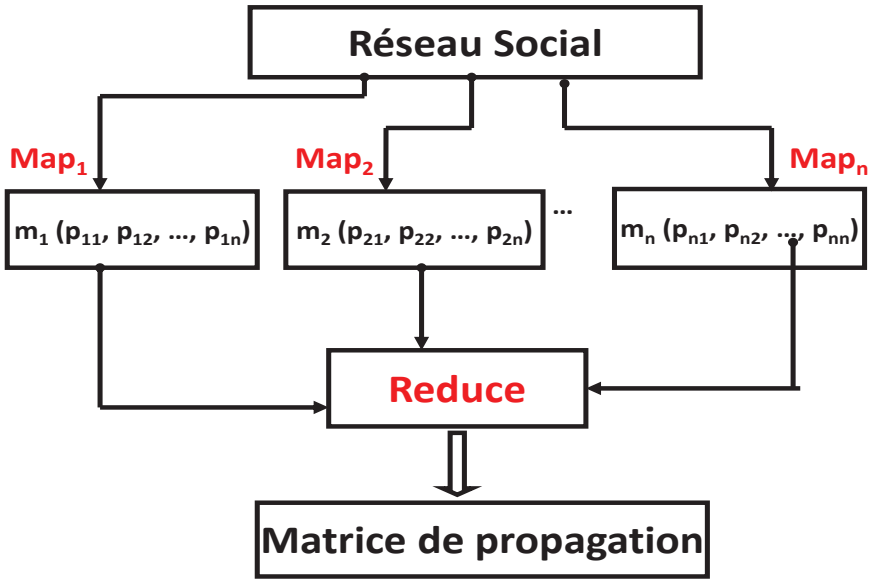

FIGURE 3.3 – Algorithme HDPD utilisant MapReduce

L'algorithme de propagation HDPD proposé (Algorithme 1) calcule la fonction d'énergie optimale $P_{o*} = (p_{o1}, p_{o2}, ..., p_{on})$ qui représente la probabilité de propagation des données du membre m_o au membres $\{m_i\}_{i\in[1,n]}$

La fonction Mapper se présente comme suit :

Algorithm 1 Mapper function of the High Data Propagation Discovery (HDPD)

Require: $InputMap$ – The list of members to be computed

m_o – owner of the data

m_r – recipient member of data that we want calculate the propagation probability

Ensure: PS – all the implicit data propagation from m_o

1: $\mathcal{PS} = (ps_1, \ldots, ps_o, \ldots, ps_n)$ – Energy function.

2: $G\langle M, A, PM\rangle \leftarrow$ Social Network Recovery Function() – labeled directed graph of the social network where PM is the propagation matrix

3: **for** each member m_i in InputMap **do**

4: $p_o = 1$ and $\forall i \neq o, p_i = 0$ {I}terations

5: **for** each member $m_i \neq m_o$ **do**

6: $ps_i = 1 - (\prod_{m_k \in N_{m_i}} (p_k \times (1 - p_{ki})))$

7: **end for**

8: Collect (PS)

9: **end for**

Et la fonction Reducer se présente comme suit :

Algorithm 2 Reducer function of the High Data Propagation Discovery (HDPD)

Require: $InputMap$ – List of propagations

1: **while** $InputMap$ has more value **do**

2: p –all the implicit data propagation from m_o

3: Collect(m_o,p) – Collect propagation of the member m_o

4: **end while**

L'application de notre algorithme sur la matrice de propagation nous

donne le résultat suivant :

	Bob	*Alice*	*John*	*David*	*Mickael*	*George*
Bob	1	0.28	0, 09	0.16	0.38	0.7
Alice	0.12	1	0.28	0.48	0.9	0.18
John	0.01	0.09	1	0.2	0.1	0.02
David	0.07	0.46	0.15	1	0, 5	0.1
Mickael	0.14	0.9	0.31	0.5	1	0.2
George	0.7	0.3	0.18	0.19	0.32	1

Dans la figure 3.4, on note que l'algorithme HDPD découvre les relations cachées entre Alice et David et met à jour la matrice de propagation en modifiant la valeur de propagation de zéro avec une propagation au-dessus du seuil. Pour des raisons de visibilité la figure contient uniquement les mises à jour pour Alice, Bob et John.

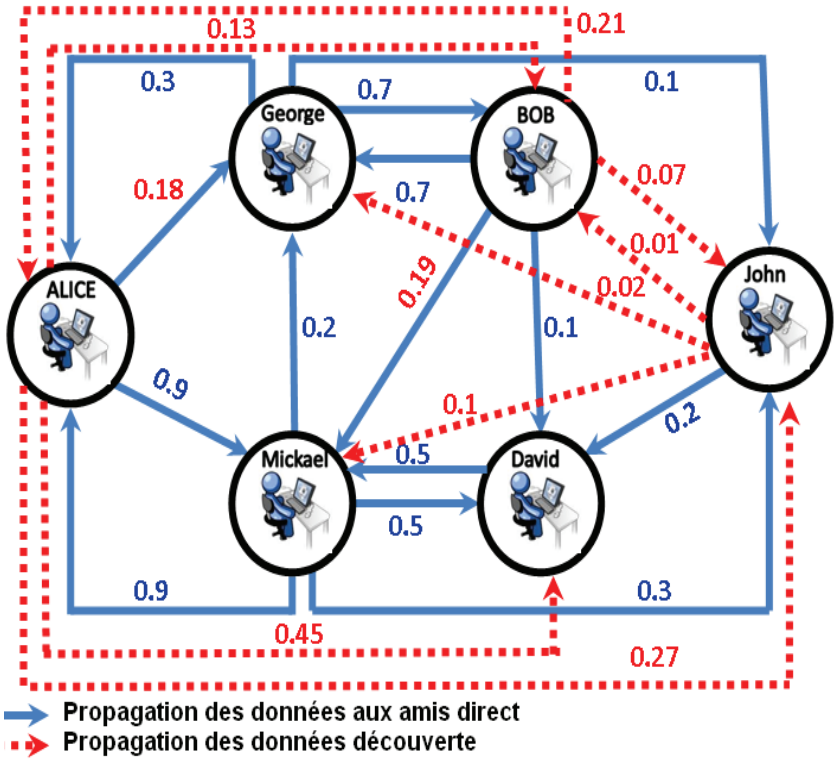

FIGURE 3.4 – Résultats du HDPD

Étant donnée n le nombre de membres enregistrés, m le nombre de relations et $f(n)$ la complexité de la fonction de propagation. Cette fonction est

composée de $n * m$ appel récursif. Ensuite, la complexité de cette fonction est : $O(n * m)$. Dans le pire des cas, le nombre relation est égal au nombre de membres.

Si on considère le cas théorique extrême, alors, la complexité de l'algorithme est de $O(n^2)$.

3.5 Calcul des propagations avec le modèle de Dijkstra basée sur MapReduce

3.5.1 Définition et propriétés

L'algorithme de Dijkstra est appelé la source du chemin le plus court. Il est également connu par le problème de la plus courte trajectoire de la source. Il calcule la distance la plus courte de la source vers les autres sommets du graphe.

Le problème de la trajectoire la plus courte de la source peut être décrit comme suit :
Soit $G = V, E$ un graphe orienté réalisé avec V sommets et E arêtes. Tous les poids dans le graphe doivent être positifs. L'algorithme de Dijkstra est principalement un algorithme glouton. Ce dernier utilise les méthodes de résolution en fonction des actions pour voir s'il y a une meilleure stratégie à long terme. L'algorithme de Dijkstra utilise l'approche gloutonne pour résoudre le problème du plus court chemin.
L'algorithme fonctionne en gardant pour chaque sommet v le coût $d[v]$ du plus court chemin trouvé jusqu'ici entre s (Source) et v. Initialement, cette valeur est de 0 pour la source ($d[s] = 0$), et l'infini pour tous les autres sommets sachant que les chemins menant à ces sommets ne sont pas tous connus. $d[v] =?$ pour tout v dans V , à l'exception de s .
Lorsque l'algorithme se termine, $d[v]$ sera le coût du plus court chemin de s à v - ou l'infini , si un tel chemin existe le fonctionnement de base de l'algorithme de Dijkstra est la relaxation des arêtes : s'il y a une arête entre u et v , alors le plus court chemin connu de s à u ($d[u]$) peut être étendue à un chemin de s à v par addition d'arête (u, v) à la fin . Ce chemin va avoir une longueur $d[u]+w(u, v)$.S'il est inférieur à $d[v]$, nous pouvons remplacer la valeur actuelle de $d[v]$ avec la nouvelle valeur de relaxation. La relaxation des arêtes est appliquée jusqu'à ce que toutes les valeurs $d[v]$ représente le coût du plus court chemin de s à v. L'algorithme est organisé pour que chaque

arête (u, v) est relaxée qu'une seule fois, lorsque $d[u]$ a atteint sa valeur finale.

L'algorithme maintient deux ensembles de sommets S et Q. L'ensemble S contient tous les sommets pour lesquels la valeur $d[v]$ est déjà le coût du plus court chemin et Q contient tous les autres sommets. L'ensemble S commence à vide, et à chaque étape un sommet est déplacé à partir de Q à S. Ce sommet est choisi étant donné qu' il a la plus faible valeur de $d[u]$. Lorsqu'un sommet u est déplacé à S, l'algorithme relaxe chaque arête sortante (u, v).

Le pseudo code de l'algorithme de Dijkstra est le suivant :

DIJKSTRA$(G = (V, E), u)$

```
1  N ← {u}
2  for all v ∈ V
3        do if v ∈ neighbors(u)
4              then D[v] ← c(u, v)
5                   p[v] ← u
6              else  D[v] ← ∞
7  while N ≠ V
8        do find w ∉ N such that D[w] is minimum
9           N ← N ∪ {w}
10          for all v ∈ neighbors(w) \ N
11                do if D[w] + c(w, v) < D[v]
12                      then D[v] ← D[w] + c(w, v)
13                           p[v] ← w
```

3.5.2 Algorithme de calcul de la probabilité de propagation des données (PPCA)

L'algorithme PPCA est un algorithme basé sur l'algorithme Dijkstra. Pour calculer la fonction d'énergie p_i pour tous les membres m_i du réseau social, nous devons utiliser un algorithme itératif [134]. Notre approche est basée sur le modèle de programmation MapRedure pour le traitement de grands ensembles de données avec un algorithme parallèle. Nous concevons notre algorithme (3) basé sur Dijkstra.

L'algorithme procède comme suit :

1. initialisation : $p_{ow} = 1$, $\forall i \neq ow$ $p_i = 0$. C'est à dire, uniquement le propriétaire des données m_{ow} qui possède ses données.

2. Itérations : A chaque itération, l'algorithme calcule p_i pour $m_i \in N$ utilisant la formule (1). La technique MapReduce est utilisée pour trouver la propagation maximale entre un utilisateurs et le reste du réseau social à l'aide du graphe de réseau social. Ainsi, nous parallélisons la phase de recherche sur les différentes Map pour trouver les propagations de données de chaque membre avec le reste des Crowders. Le reducer permet de regrouper les propagations par membre.

   • La fonction Map est appliquée en parallèle à chaque paire dans l'ensemble de données d'entrée. Chaque Map reçoit en entrée une liste de membres. Pour chaque membre elle calcule la liste des propagations de données avec le reste des membres du réseau social en se basant sur la matrice du réseau récupérée à partir du contexte. Cela produit une liste de paires pour chaque appel. Le motif de MapReduce recueille toutes les paires avec la même clé de toutes les listes et les regroupe, en créant un groupe pour chaque clé. La $inputMap_k < key, value >$ devient dans notre cas
   $< User, user_i >$, où i représente le numéro de l'utilisateur.

   La Map générée
   $OutputMap_k < key, List(key, value) >$ sera comme suit :$< UserNumber, List(< HashCode_k,$
   $Propagation >) >$ où, Hash code est le code unique identifiant l'utilisateur $user_k$ tout en prenant en compte sa structure. La $Propagation$ représente la probabilité de propagation du $user_i$ aux reste des utilisateurs.

   Le but du script de map est de modéliser les données en $< key, value >$ paires et au reducer de regrouper les paires émises $< Key, value >$. Ce qui signifie essentiellement que, les paires avec la même clé sont regroupées et transmises à une seule machine, qui ensuite exécutera le script reducer sur eux.

   • La fonction Reducer combine les valeurs d'une clé. Dans notre cas, elle permet de grouper les propagations par utilisateur .

   La figure 3.5(a) décrit l'application du MapReduce et la figure 3.5(b) décrit sont implémentation.

3. Arrêt : L'algorithme s'arrête lorsqu'il n'existe plus de propagation à

calculer.

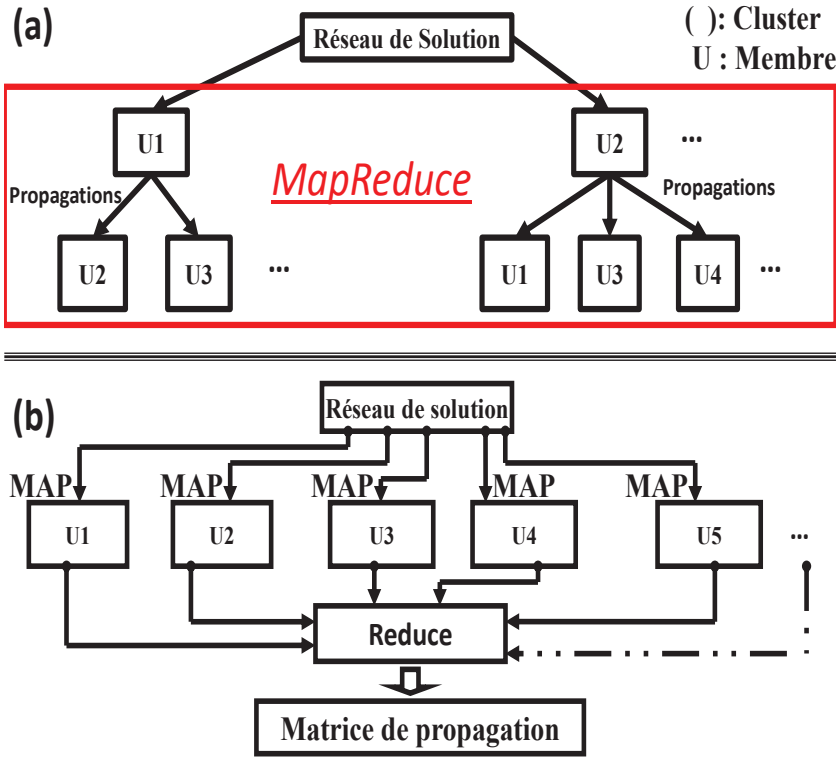

FIGURE 3.5 – Architecture du MapReduce et son application

En appliquant notre algorithme à la matrice de propagation citée ci dessus, on obtient la matrice de propagation suivante :

$$\begin{array}{c} \\ Bob \\ Alice \\ John \\ David \\ Mickael \\ George \end{array} \begin{array}{c} \begin{array}{cccccc} Bob & Alice & John & David & Mickael & George \end{array} \\ \begin{pmatrix} 1 & 0.28 & 0,09 & 0.16 & 0.38 & 0.7 \\ 0.12 & 1 & 0.28 & 0.48 & 0.9 & 0.18 \\ 0.01 & 0.09 & 1 & 0.2 & 0.1 & 0.02 \\ 0.07 & 0.46 & 0.15 & 1 & 0,5 & 0.1 \\ 0.14 & 0.9 & 0.31 & 0.5 & 1 & 0.2 \\ 0.7 & 0.3 & 0.18 & 0.19 & 0.32 & 1 \end{pmatrix} \end{array}$$

La fonction Mapper se présente comme suit :

Algorithm 3 MAPPER FUNCTION OF THE PROPAGATION PROBABILITY COMPUTING ALGORITHM (PPCA)

Require: $InputMap$ – The list of members to be computed

u – The member to be computed u – owner of the data

Ensure: $pred$ – longest path tree from u

1: $G\langle M, E\rangle \leftarrow$ Social Network Recovery Function() – labeled directed graph of the social network where M in the Members and E are the Edges
2: **for** each member u in InputMap **do**
3: **for** u in V **do**
4: $P(u) = 0$
5: $mark(u)$ = undiscovered
6: $pred(u)$ = null
7: **end for**
8: P(s) = 1 {distance to source is 1}
9: Q = a priority queue of all vertices u sorted by P(u) {until all vertices processed}
10: **while** Q is nonEmpty **do**
11: u = extract vertex with maximum P(u) from Q
12: **for** v $\in$ Adj(u) **do**
13: **if** $P(u) * w(u, v) > P(v)$ **then**
14: $P(v) = 1 - (\prod_{u \in Adj(u)} (P(u) \times (1 - w(u, v))))$ {relax(u,v)}
15: decrease $v's$ key in Q to P(v)
16: $pred(v) = u$
17: **end if**
18: **end for**
19: $mark(u)$ = finished
20: **end while**{[The pred define the longest path tree]}
21: Collect ($pred$)
22: **end for**

La fonction Reducer se présente comme suit :

Algorithm 4 Reducer function of the Propagation probability computing algorithm (PPCA)

Require: $InputMap$ – List of propagations

1: **while** $InputMap$ has more value **do**

2: $pred$ – The longest path tree of the member m_o

3: Collect(m_o,$pred$) – Collect propagation of the member m_o

4: **end while**

Pour analyser l'algorithme de calcul de la probabilité de propagation, on rappelle que n = —utilisateurs — et m = —relations directes—. Nous comptabilisons le temps consacré à chaque utilisateur après son extraction de la file d'attente prioritaire. Il faut O(log n) pour extraire l'utilisateur de la file . Pour chaque relation directe, nous passons potentiellement O(log n) de temps si nous avons besoin de diminuer la clé des utilisateurs voisins. Ainsi le temps est de O(log n + deg(u)· log n).

Les autres étapes consistant à exécuter la mise à jour en temps constant. En rappelant que la somme des degrés des sommets dans un graphe est de O(m). Le temps global de fonctionnement est donné par T (n, m), où $O(n, m)$ = $\sum_{n\in V}$(log n + deg(u) · log n) = $\sum_{n\in V}$ (1 + deg(u)) log n = log n $\sum_{n\in V}$ (1 + deg(u)) = (log n)(n + 2m) = O((n + m) log n).

Comme G est connecté, n n'est pas asymptotiquement supérieur à m, donc c'est O(m log n). Pour calculer les propagations cachées pour chaque utilisateur, la complexité de calcul doit être de n * O(m log n). Par contre, nous somme entrain d'utiliser le modèle de MapReduce dans notre approche, donc on peut paralléliser le traitement en n Maps. Ceci nous amène à une complexité de $(n * O(m\ log\ n)) \div n \Rightarrow O(m\ log\ n)$.

Dans la figure 3.6, on note que l'algorithme PPCA découvre les relations indirectes cachées entre Alice et David. Il met également à jours la matrice de propagation en modifiant les valeurs de propagation zéro en des vraies valeurs de propagation cachée. Pour une raison de visibilité la figure contient seulement la propagation à jour pour Alice, Bob et John.

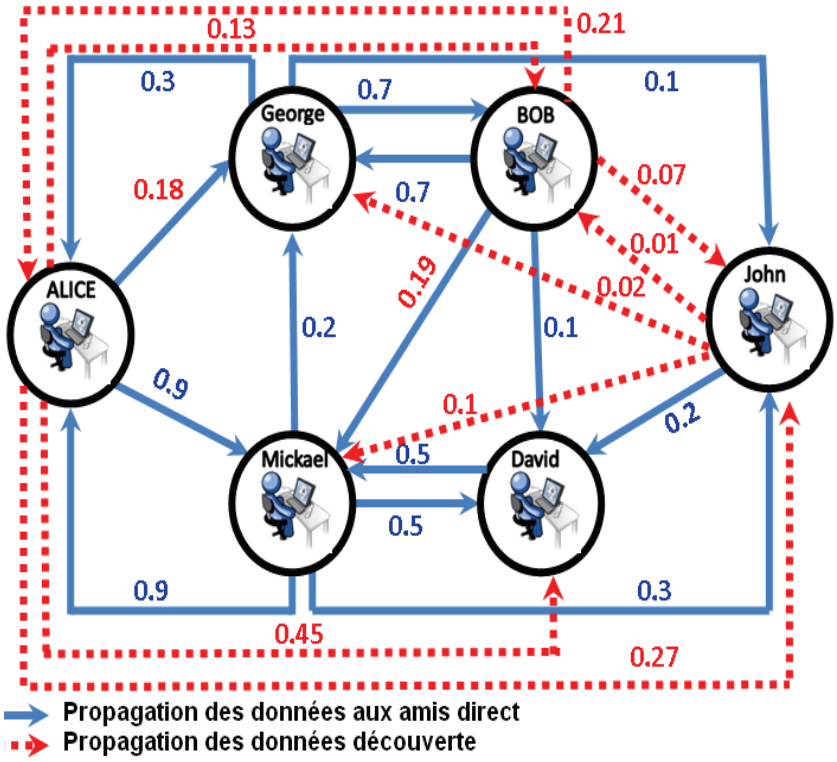

FIGURE 3.6 – Résultats du PPCA

3.6 Discussion

Dans ce chapitre, nous avons présenté deux méthodes permettant d'estimer la propagation maximale des relations cachées entre les différents utilisateurs. La première méthode se repose sur le modèle des chaînes de Markov alors que la deuxième se repose sur le modèle de Dijkstra.

Les deux algorithmes nous amènent aux mêmes résultats de découverte des propagations maximales des données entre tous les utilisateurs du réseau.

Le premier algorithme de découverte des relations cachées HDPD, est plus consommateur en temps de calcul que celui de l'algorithme PPCA présenté en deuxième lieu.

L'algorithme PPCA améliore nettement le temps de calcul avec des résultats similaire à l'algorithme HDPD, du fait qu'il utilise un système de parcours

efficace permettant de déterminer rapidement la valeur de propagation maximale entre les utilisateurs.

Dans ce qui suit, nous allons utiliser l'algorithme PPCA pour découvrir les relations implicites entre les utilisateurs du crowdsourcing.

Chapitre 4

Découverte des équipes en préservant la vie privée

4.1 Introduction

La préservation de la vie privée se base essentiellement sur la non propagation des données privées d'une équipe à une autre équipe concurrente.

L'objectif est de créer des équipes collaboratives et compétitives, d'où l'intérêt de maximiser la propagation des données dans la même équipes et minimiser la propagation des données entre les différentes équipes.

4.2 Approches de découverte d'une solution unique

Nous allons maintenant examiner les méthodes d'apprentissage non supervisé, où il n'est pas clair dans quelles catégories, le cas échéant, les données seront classifié.

4.2.1 Méthodes spectrales

Le clustering est une des techniques les plus utilisées pour l'analyse exploratoire des données. Pratiquement, dans tous les domaines scientifiques portant sur des données empiriques, les utilisateurs cherchent à faire une première impression sur leurs données en essayant d'identifier les groupes de "comportement similaire". Par rapport aux algorithmes "'traditionnels"' tels que k-means. La classification spectrale présente de nombreux avantages fondamentaux.

Au cours des dernières années, la classification spectrale est devenue l'un des algorithmes les plus populaires de clustering modernes. Cet algorithme se caractérise par sa simplicité de mise en oeuvre, il surpasse souvent les algorithmes de clustering traditionnels tels que l'algorithme K-means. Au premier coup d'oeil la classification spectrale apparaît un peu mystérieuse.

La méthode spectrale se base sur la mesure d'affinité entre tous les couples de points de données, généralement, cette méthode ne s'intéresse pas à l'étude des formes des classes (ou cluster). Le paramètre de l'affinité à un rôle important dans le regroupement des membres et il n'existe pas a priori de solution pour trouver un paramètre optimal.

La méthode de clustering spectral [34] consiste à extraire les vecteurs propres associés aux plus grandes valeurs propres d'une matrice affinité normalisée, issue d'un noyau de Mercer [55] .

Les vecteurs propres constituent un espace de dimension réduit dans lequel les données transformées seront linéairement séparables.
Deux grandes principales classes d'algorithmes de classification spectrale ont été proposés à partir de la division de graphes [129] .

La première classe d'algorithme est fondée sur un partitionnement bipartite récursif en se basant sur le vecteur propre associé à la seconde plus grande valeur propre du graphe du Laplacien normalisé [71, 123], ou vecteur de Fiedler [28] dans le cas non-normalisé.

La deuxième projette les données originales dans un espace défini par les k plus grands vecteurs propres d'une matrice d'adjacence normalisée (ou matrice similaire à celle-ci), et applique un algorithme de classification ainsi que le k-means sur ces nouvelles points calculés [106, 96] .

Nous allons détailler uniquement la dernière classe dans un souci de simplicité algorithmique. Y.Weiss et al [106] présentent cette dernière classe d'algorithmes 5 pour partitionner un ensemble de points $S = x_1, \ldots, x_N \subset \Re^p$ en k clusters où k est fixé.

Par contre, généralement où les clusters ne sont pas nettement séparés par une distance importante, matrice affinité est plus ou moins compromise et peut être considérablement altérée par le choix de la valeur du paramètre de l'affinité (σ).

Ce paramètre, en facteur de la norme entre chaque couple de points, sert de pondérateur. Il peut donc, suivant sa valeur, diminuer l'affinité intra-cluster et augmenter celle entre les clusters.

Algorithm 5 ALGORITHME DE PARTITIONNEMENT SPECTRAL

Require: Ensemble des données S, Nombre de clusters k

1: Construction de la matrice affinité $A \in \Re^{N \times N}$ définie par :

$$A_{ij} = \begin{cases} exp(-\left\|x_i - x_j\right\|^2 / 2\sigma^2 \text{ si } i = j \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad \text{sinon} \end{cases}$$

2: Construction de la matrice normalisée $L = D^{-1/2} A D^{-1/2}$ où D matrice diagonale définie par

$$D_{i,i} = \sum_{j=1}^{N} A_{ij}.$$

3: Construction de la matrice $X = [X_1, X_2, \ldots, X_k] \in \Re^{N \times N}$ formée à partir de k plus grands vecteurs propres $x_i, i = \{1, \ldots, k\}$ de L.

4: Construction de la matrice Y formée en normalisant les lignes de X :

$$Y_{ij} = \frac{X_{ij}}{\left(\sum j X_{ij}^2\right)^{1/2}}$$

5: Traiter chaque ligne de Y comme un point de $\Re^k$ et les classer en k clusters via la méthode K-means.

6: Assigner le point original x_i au cluster j si et seulement si la ligne i de la matrice Y est assignée au cluster j.

Parmi les différents noyaux de Mercer [55], généralement, le noyau Gaussien est utilisé. L'affinité entre deux points de données distincts x_i et x_j de $\Re^p$ est alors définie par :

$$A_{ij} = \begin{cases} exp(-\left\|x_i - x_j\right\|^2 / 2\sigma^2 \text{ si } i = j \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad \text{sinon} \end{cases}$$

où σ est un paramètre et $\|.\|_2$ est la norme euclidienne habituelle.

Le principe de la classification spectrale se base sur la mesure d'affinité. Or, la fonction de l'affinité gaussienne dépend du paramètre σ. Cette donnée influe directement sur la méthode.En effet, le paramètre σ influe sur la séparabilité des points dans l'espace de projection spectrale.

Dans notre cas, il sera difficile de représenter les membres dans un espace spectrale. La représentation de la matrice de probabilité de propagations de données entre les différents membres ne permet pas un partionnement adéquat en équipes compétitives et collaboratives tout en minimisant la fuite de données entre les différentes équipes concurrentes.

Cette approche permet de résoudre le problème en proposant une solution unique et approximative en terme de fuite de données. Cette solution ne pourra pas être validé du fait qu'elle ne respecte pas la contrainte de la fuite de données.

Une approximation ne garantit en aucun cas le bon déroulement du processus de crowdsourcing dans notre cas. Concernant le "Caller" cela représente une fuite de données qui est susceptible de fragiliser l'intêret de ce dernier au passage au crowdsourcing. Et concernant l'intélligence humaine, cela permet de maximiser le risque de perte de la récompense attachée à l'appel du "Caller".

4.2.2 Algorithme de classification K-means

L'algorithme K-means est utilisé pour des fins utiles de comparaisons et il n'est en aucun cas pris en compte comme solution pour le problème de découverte d'équipes sans fuites de données.

La classification est utilisée pour segmenter les données en groupes d'éléments similaires. Il est utile pour organiser automatiquement des données, de trouver une structure masquée dans les données, et une forme de compression (A

titre exemple, la représentation de 3000 points de données dans un ensemble appelé bac 1).

Par exemple, prévoir les habitudes d'achat des clients, trouver des modèles / des groupes de gènes afin d'apprendre la structure des données à un niveau supérieur ou le regroupement des utilisateurs de MySpace en fonction de différents intérêts.

La classification des données comme les emails, les profils d'expression des gènes, ou l'historique des achats, sont représentés comme suit : $D = X_1, \ldots, X_N$.

Puisque les données sont p-dimensionnel, nous les représentons comme suit : $X_n = x_{n,1}, \ldots, x_{n,p}$.

La fonction de distance est : $d(X_n, X_m)$ entre deux points de données. L'ensemble de K groupes où on souhaite diviser les données sont $z_1, \ldots, z_N$ où $x \in 1, \ldots, K$. Donc on souhaite assigner un label à chaque point x. On note que nous pouvons les assigner arbitrairement.

Considérant la figure de données fig 4.2.2. Une bonne fonction de distance est la distance euclidienne $d(X_n, X_m) = \sum_{i=1}^{p}(x_{n,i} - x_{m,i})^2 = ||x_n - x_m||^2$. Par contre, actuellement nous cherchons à segmenter les données en k groupes. On choisie $k = 4$ pour l'instant, parce que trouver k est compliqué. Considérons les étapes intuitives que l'algorithme k-means prendra, comme le montre la figure 4.2.2. On recalcule les centres des points de données. Ensuite, nous réaffectons les points de données à de nouveaux groupes. L'algorithme détaillé du K-means de base est le suivant :

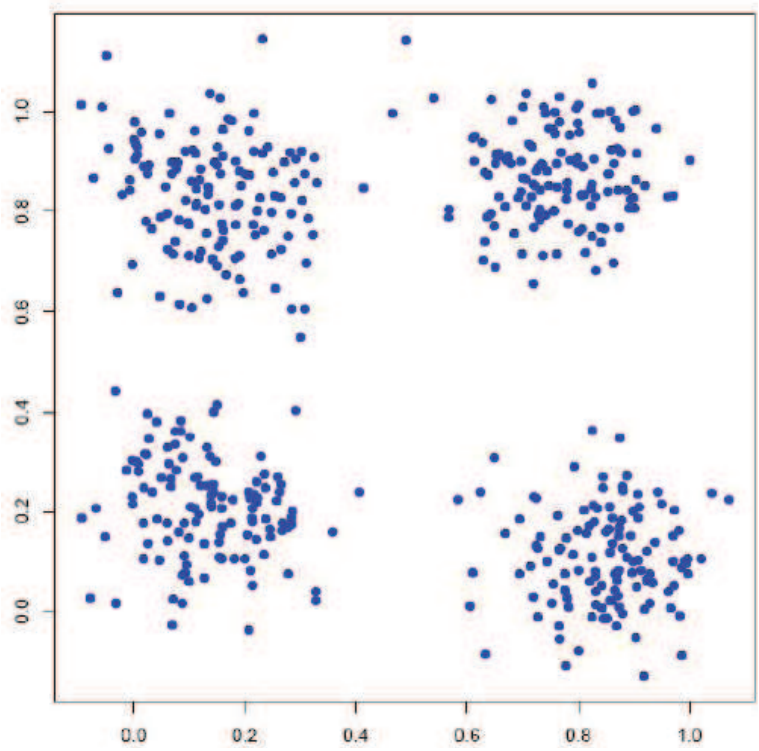

FIGURE 4.1 – représentation de 500 2-dimensionnel point de donnée $X_n = \langle X_{n,1}, X_{n,2} \rangle$

1. Initialisation
   (a) Les données sont $X_{1:N}$
   (b) Choisir arbitrairement un cluster initial $m_{1:k}$
2. Répéter
   (a) Assigner chaque point de donnée au cluster le plus proche

$$Z_n = argmin_{i \in 1...k} d(X_n, m_i)$$

   (b) Calculer la distance moyenne entre toutes les coordonnées attribuées au nouveau cluster et la moyenne du cluster.

$$m_k = \frac{1}{N_k} \sum_{n:Z_n=k} X_n$$

3. Jusqu'à ce que l'affectation de $Z_{1:N}$ ne change pas

La fonction objective :

Nous mesurons la faisabilité de l'algorithme en utilisant les distances entre la somme des carrés de chaque point et sa moyenne respective, $F(z_{1:N}, m_{1:k}) =$

$\frac{1}{2}\sum_{n=1}^{N}||X_n - m_{Z_n}||^2$.

Rappelons que x_n est un point de données et m_{Z_n} est le groupe de ce point de données. Nous trouvons la convergence en regardant le changement relatif entre les cycles successifs et nous nous arrêterons lorsque on arrive à ce que nous considérons une différence négligeable.

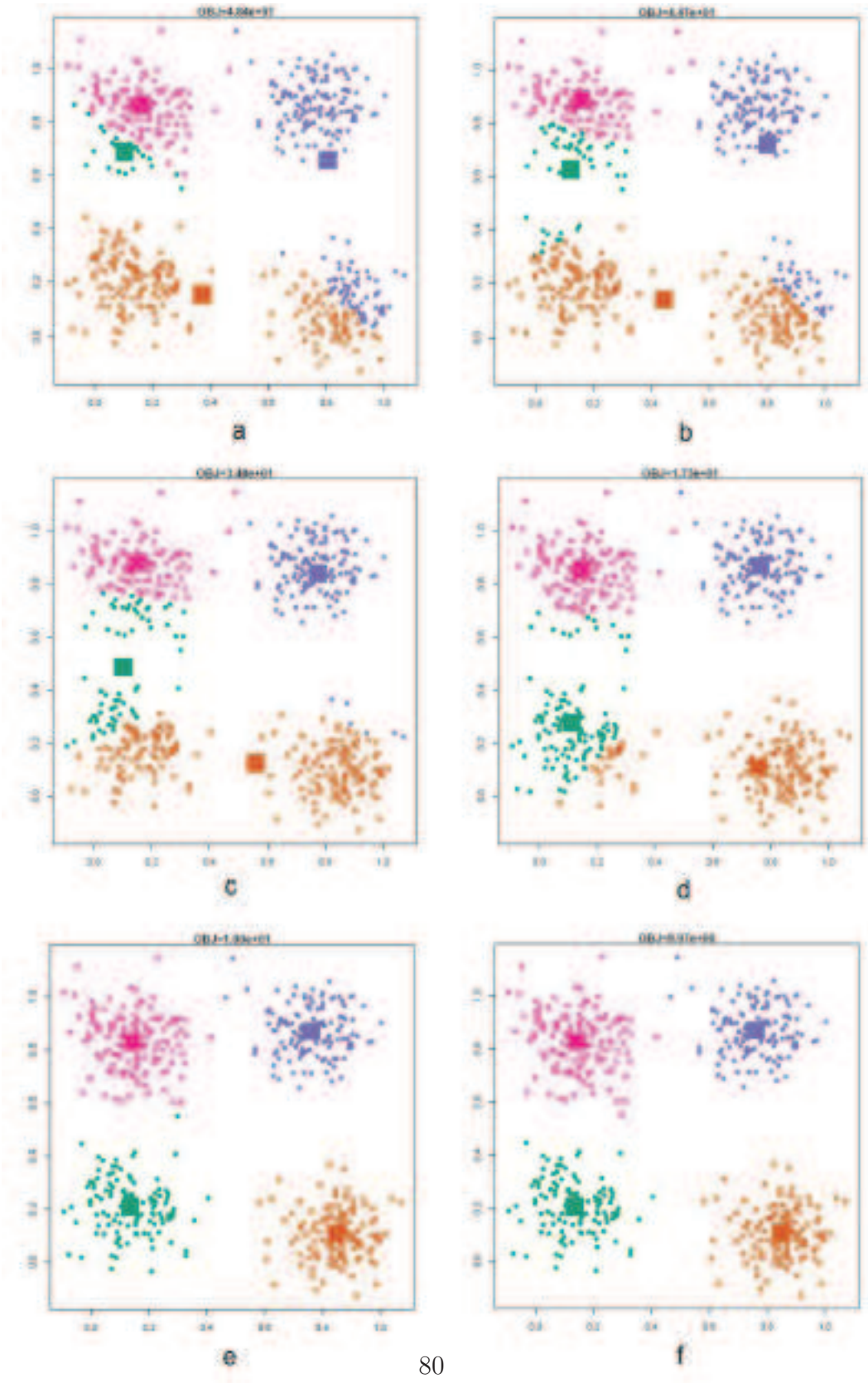

FIGURE 4.2 – Progression de l'algorithme de classification K-means sur les données de la figure 4.2.2

Algorithme à directions de descente :
Le K-means est un algorithme à direction de descente, tout d'abord, il attribue chaque point au groupe le plus proche, minimisant ainsi F en respectant $z_{1:N}$. Deuxièmement, il calcule les nouvelles moyennes de chaque groupe tout en fixant les assignations, minimisant ainsi F par rapport à $m1 : k$.

Cependant on pourra avoir des minimums locaux, il est donc essentiel de faire fonctionner l'algorithme plusieurs fois.

4.2.3 Algorithme de classification D-Max discovering teams

En se basant sur la propagation des données calculées dans le chapitre précédent, nous proposons un processus de classification qui regroupe les membres du crowdsourcing dans les clusters sans de fuite de données.
Afin de mieux comprendre le déroulement de l'algorithme de classification, nous préférons donner quelques définitions.

Définition 4 *Un cluster C est un ensemble d'utilisateurs du crowdsourcing ayant ou pas une forte propagation(en dessus du seuil maximal autorisé) :*

$$C = \{m_i\} \text{ de telle sorte que } \forall m_i, m_j \in C, p_{ij} \in [0,1], p_{ji} \in [0,1]$$

Définition 5 *Deux clusters C_k et C_s sont sans fuite de données si et seulement si :*

$$\forall m_i \in C_k, \forall m_j \in C_s, p_{ij} \leq \eta \wedge p_{ji} \leq \eta$$

De la définition 5, nous considérons qu'il existe un risque de fuite de données entre deux clusters C_K et C_S si le taux de propagation entre tous les membres des deux groupes est supérieur à un seuil η.

Ce dernier est proposé comme une valeur pour laquelle la propagation de données d'un membre du crowdsourcing est acceptable dans un réseau social. La dynamique des profils des membres du réseau social influe la valeur de η mais il est hors de la portée de cette thèse.

Le but de ce travail est de minimiser la propagation des données entre les équipes. Nous souhaitons donc garder la vie privée de chaque utilisateur du crowdsourcing. Afin de minimiser les risques de propagation, il nous faut détecter le risque maximum de propagation, la matrice de propagation calculée par l'algorithme 3 est mise à jour comme suit :

$$\forall i, j, p_{ij} = Max(p_{ij}, p_{ji})$$

La matrice de propagation calculée dans le chapitre précédent est donc mise à jour comme suit :

$$\begin{array}{c} \\ Bob(B) \\ Alice(A) \\ John(J) \\ David(D) \\ Mickael(M) \\ George(G) \end{array} \begin{array}{c} \begin{array}{cccccc} B & A & J & D & M & G \end{array} \\ \begin{pmatrix} 1 & 0.21 & 0.07 & 0.1 & 0.19 & 0.7 \\ 0.21 & 1 & 0.27 & 0.45 & 0.9 & 0.3 \\ 0.07 & 0 & 1 & 0.2 & 0.3 & 0.1 \\ 0.1 & 0.45 & 0.2 & 1 & 0,5 & 0.13 \\ 0.19 & 0.9 & 0.3 & 0.5 & 1 & 0.27 \\ 0.7 & 0.3 & 0.1 & 0.13 & 0.27 & 1 \end{pmatrix} \end{array}$$

Sur la base de cette matrice de propagation mise à jour, les membres sont classés en groupes sans fuite de données privées en utilisant un algorithme de groupement.

Notre algorithme de classification est un algorithme inspiré de l'algorithme K-means [91] pour lequel nous avons défini une fonction une distance spécifique appelée D_{max} :

$$D_{max}(C_k, m_j) = \underset{m_i \in C_k}{Max}\ p_{ij}$$

où Max est la fonction maximum, C_k est un cluster à classifier, m_j est un membre susceptible à être classifier dans le cluster C_k, et m_i est un membre du crowdsourcing dans C_k, p_{ij} est la valeur de propagation entre m_i et m_j donnée dans la matrice de propagation.

Le processus de l'algorithme de classification est le suivant :

- entrées : le seuil de divulgation de données η, le nombre de cluster.
- Initialisation : L'initialisation des clusters est effectuée par l'attribution arbitraire d'un membre à chaque cluster.
- Itérations : Pour chaque membre candidat m_i et un cluster C_j, si $D_{max}(C_j, m_i) \geq \eta$ alors m_i est ajouté à C_k
- Arrêt : L'algorithme est réputé avoir convergé lorsque tous les membres sont assignés à des équipes.

En outre, dans le cas où un membre candidat a une propagation forte avec plus d'un groupe de la solution, nous fusionnons les groupes avec lesquels le membre candidat a une propagation de données élevée et l'affecter au nouveau cluster fusionné, parce qu'il peut probablement divulguer les données d'un cluster aux autres clusters. Par exemple, si $d(C_1, m_k) = 0,8$ et $d(C_2, m_k) = 0,7$, alors nous fusionnons le groupe C_1 et C_2 et m_k intégrera le cluster C_{12}le résultat de la fusion du cluster C_1 et C_2. L'algorithme est présenté comme :

Algorithm 6 D-MAX DISCOVERING TEAMS ALGORITHM

print $Clusters = (Clust_1, Clust_2, ..., Clust_{Cluster})$: Teams constitution

{I}nitializations

$Distances, Centroid, MaxDistancesMax = 0$

$ClustNB = -1$

$Threshold = \eta$

for each $ClustersinCluster$ **do**

  $Clusters_i = member_{m_i}$

  $Distances_i = 0$

  $Centroid_i = 0$

end for

{I}terations

for each members m_i in G **do**

  for each members m_i in $Clusters$ **do**

    $Distances_i \leftarrow P_{member,member_i}$

    if $Distances_i \succ Centroid_i$ **then**

      $Centroid_i \leftarrow Distances_i$

    end if

  end for

  for each $Centroid_i$ **do**

    if $Centroid_i \succ MaxDistancesMax_i$ **then**

      $MaxDistancesMax_i \leftarrow Centroid_i$

      if $MaxDistancesMax_i \succ \eta$ AND $ClustNB \neq -1$ **then**

        $Clusters_i = FUSION(Clusters_i, Clusters_{NB})$

      end if

      $ClustNB = i$

      $Clusters_{ClustNB} \leftarrow member_i$

    end if

  end for

end for

return $Clusters$

4.2.4 Expérimentations

Nos expériences évaluent notre modèle proposé sur un réseau social similaire à l'architecture du réseau Facebook (www.facebook.com). Facebook est un site populaire de réseau social en ligne avec plus de 1.11 Billion d'utilisateurs à travers le monde, plus de 4.75 billion de données se partagent par jour (des pages, des groupes, des événements et des pages de communauté), l'utilisateur moyen est connecté à 150 pages communautaires, groupes et événements.

Configuration des expériences

Toutes nos expériences sont codées avec la technologie JAVA et s'exécutent sous le système d'exploitation Mac OS X 10.5.8, 2.13 Ghz Intel Core 2 Duo avec 2 Go de mémoire.

Pour ces expériences, nous avons choisie arbitrairement 5000 utilisateurs ; le nombre maximum d'amitié autorisé sur la plate-forme Facebook.

Ensuite, notre échantillon contient un minimum de 25 000 000 relations. A partir de cet échantillon, nous avons construit un ensemble avec tous les utilisateurs afin de préserver la divulgation des données au cours du processus de découverte des équipes. Le modèle de découverte des relations implicites (basé sur Dijkstra) est utilisé sur l'échantillon pour découvrir les valeurs de propagation des données cachées.

Expériences de la vie privée

A. La distance Classique et la distance $D - max$ pour la classification :

Nous avons testé notre échantillon en utilisant deux modèles de classifications différents :

- Pour la première expérience on a utilisé le modèle K-means avec une distance euclidienne, qui tient compte de la différence entre deux utilisateurs, directement, sur la base de l'amplitude des variations dans les niveaux de l'échantillon. Ce type de distance est habituellement utilisé pour les ensembles de données qui sont convenablement normalisée ou sans problème de distribution spéciale, nous avons utilisé la technique MDS (Multidimensional scaling) pour transformer chaque propagation des utilisateurs à un point multidimensionnel.

  La distance euclidienne entre deux points p et q est la longueur du

segment de droite reliant (pq). En coordonnées cartésiennes, si $p = (p_1, p_2, ..., p_n)$ et $q = (q_1, q_2, ..., q_n)$ sont deux points dans l'espace euclidien, alors la distance de p à q, ou de q à p est donnée par :

$$d1(p, q) = d1(q, p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Pour cette expérience nous avons utilisé la projection MDS (Multidimensional Scaling) [90] de notre matrice de dissimilarité.

– La deuxième expérience de classification est faite avec l'algorithme D-Max discovering teams en se basant sur une distance qu'on propose pour rattacher un membre à un cluster, qui prend en compte la valeur maximale de propagation des données entre les équipes compétitives.

La figure 4.3(a) et 4.3(b) représente le résultat de l'analyse MDS de la matrice de 500 et 2500 utilisateurs respectivement.

Nous avons obtenu une représentation bidimensionnelle des emplacements des utilisateurs du crowdsourcing.

La représentation en nuages est le résultat de l'importante dissemblance entre les utilisateurs du réseau, basée sur les différentes probabilités de propagation entre les utilisateurs connectés. Donc, nous pensons que l'opération de regroupement pourrait être faite sur notre échantillon.

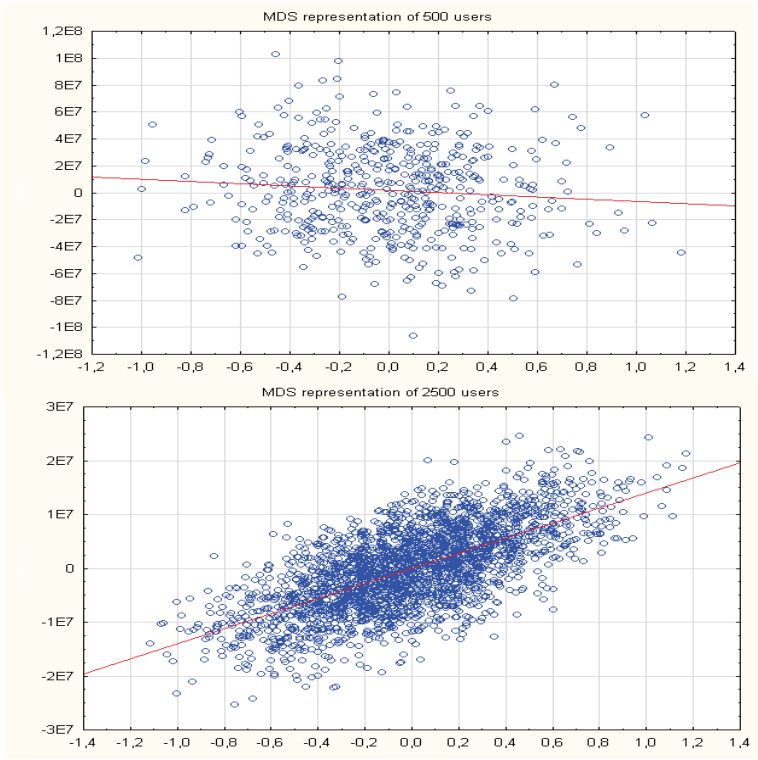

FIGURE 4.3 – Projection MDS sur 500 et 2500 utilisateurs du crowdsourcing

Ensuite, nous avons analysé les groupes générés à l'aide de la distance D_{max} et la distance euclidienne et nous remarquons que, pour 10 groupes, de 10 à 5000 utilisateurs l'algorithme D-Max discovering teams utilisant la distance D_{max} n'enregistre aucune fuite de données entre les clusters.

Cependant, avec la distance euclidienne (figure 4.4 (a)) nous percevons des fuites de données variables (a) entre 5 et 6 % en générant 10 clusters, (b) entre 2 et 3 % en générant 20 clusters (c) environ 2% en générant 30 clusters . Pour les 10 équipes générés avec l'algorithme k-means utilisant la distance euclidienne, on note 54005 fuite de données. Ensuite, pour 1 000 000 relations nous enregistrons une fuite de données égale à 5.4 %. Ce calcule est basé sur une constance égale à 0.4 représentant le seuil maximum de propagation de données autorisé entre deux utilisateurs du crowdsourcing.

Par la suite , nous avons calculé les fuites de données de 1000 utilisateurs ayant entre 10 et 1000 relations dans le réseau social en utilisant le modèle de classification k-means avec la distance euclidienne, figure 4.4 (b). On note que la fuite de données pour 10 clusters générés est environ de 5 %, la fuite

de données pour 20 clusters est entre 2.78 % et 4.4 % et pour 30 clusters, on obtient une fuite de données de 1.89 % à 3.

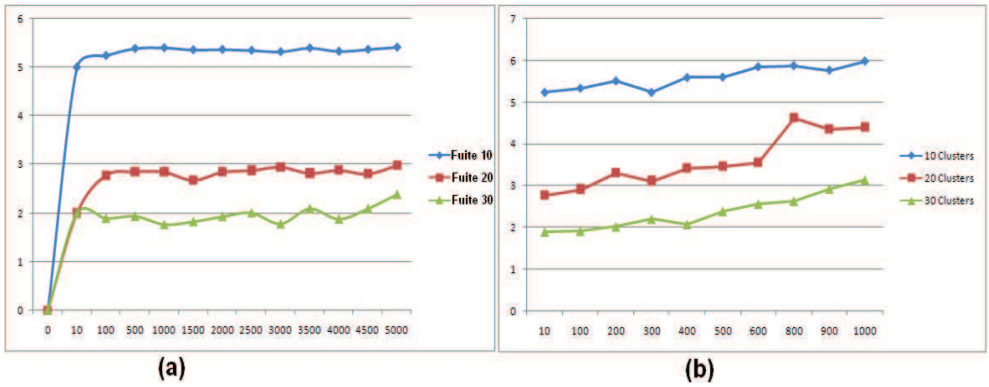

FIGURE 4.4 – Présentation des fuites de données

Ces expériences nous montrent que l'algorithme D-Max discovering teams avec la distance D_{max} est plus fiable que le K-means avec la distance euclidienne. La distance D_{max} permet d'interdire complètement les fuites de données entre les équipes compétitives.

B. La distance Chebyshev et la distance D-max pour l'algorithme k-means :

Dans une deuxième expérience, nous avons modifié la distance de calcul de l'algorithme k-means. Nous avons utilisé la distance *Chebyshev* que l'on appelle aussi la distance de la valeur maximale. Elle examine la valeur absolue de la différence entre les coordonnées d'une paire d'objets, dans notre cas, ils seront les membres du crowdsourcing et les équipes. Cette distance peut être utilisée à la fois pour les variables quantitatives et ordinaires. La distance de Chebyshev est représentée comme suit :

$$d(i,j) = max|x_i - x_j|$$

La distance entre deux utilisateurs du crowdsourcing est la probabilité de propagation de données privées entre eux, or pour assigner un utilisateur du crowdsourcing à un cluster, la distance entre une équipe et le nouveau membre doit être en dessous de 0.4. Lorsqu'on calcule le centroïde du cluster, on observe que le centre du cluster ne reflète pas la réalité, parce que, ce centroïde est calculé en se basant sur la moyenne des valeurs maximales des probabilités de propagations entre les membres des clusters, or le centroïde correcte est la plus forte valeur de probabilité de propagation de données

privées entre chaque membre du cluster et le nouveau membre à classifier.

Expériences de temps d'exécution

On a analysé le temps d'exécution de l'algorithme K-means avec la distance euclidienne et l'algorithme D-Max discovering teams avec la distance D_{max}, pour 10,20,30 et 40 clusters en utilisant entre 10 et 5000 utilisateurs. Dans la figure 4.5 (a) pour 10 clusters et utilisant le K-means avec la distance euclidienne, le temps d'exécution est entre 2 et 2346 millisecondes,

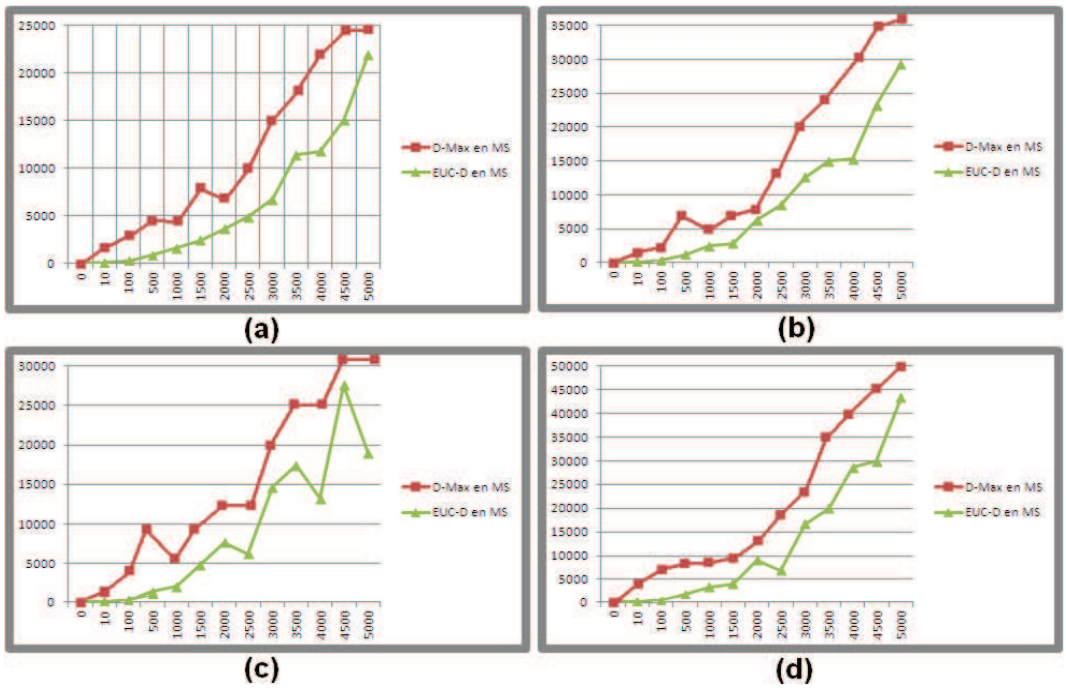

FIGURE 4.5 – Représentation des temps d'exécution

il nettement meilleur pour le K-means avec la distance euclidienne que pour le D-Max discovering teams avec la distance D_{max}. La même conclusion est établie pour 20,30 et 40 clusters. Les résultats sont représentés sur la figure 4.5 (b,c,d).

L'algorithme D-Max discovering teams avec la distance D_{max} est plus fiable sur la préservation de la fuite de données, par contre, cette distance ralentit le temps d'exécution de l'algorithme.

4.2.5 Discussion

Dans cette section nous avons détaillé le modèle k-means avec sa distance euclidienne et nous avons proposé notre algorithme D-Max discovering teams avec une distance spécifique de calcul D_{max} qui se repose sur la distance maximale de propagation des données privées entre les utilisateurs.

Les expériences montrent que l'algorithme K-means avec la distance euclidienne ne donne pas des solutions fiable de découverte d'équipes compétitives, par contre, en utilisant l'algorithme D-Max discovering teams avec la distance D_{max} on arrive à découvrir une seule solution avec des équipes compétitives mais avec un temps de calcul plus long.

L'approche proposée nous permet, néanmoins, d'avoir une solution sans fuite de données entres les différents clusters.

Malheureusement, l'algorithme nous retourne une solution unique, et ne nous permet pas d'explorer la majorité des solutions possibles de groupement des membres du crowdsourcing.

En plus, notre approche ralentit le temps d'exécution par rapport à l'algorithme de K-means classique puisque elle cherche toutes les relations implicites entre les utilisateurs lors de la phase de classification pour interdire les fuites de données entre les clusters.

Dans la section suivante nous développons une nouvelle approche permettant de découvrir toutes les solutions de groupement possibles de travailleurs dans le crowdsourcing.

4.3 L'algorithme de classification hiérarchique

4.3.1 Définitions et propriétés

La classification hiérarchique [70] permet de classer des éléments dans une hiérarchie de structure arborescente sur la base de la distance ou de la similarité entre ces éléments. La représentation graphique de la hiérarchie qui en résulte est un graphique arborescent appelé un dendrogramme. L'Algorithme de classification hiérarchique existe en deux types :

i) l'algorithme de classification hiérarchique ascendante

ii) l'algorithme de classification hiérarchique séparatif.

Les deux algorithmes sont exactement l'inverse l'un de l'autre. Donc, nous allons présenter uniquement l'algorithme de classification ascendante hiérarchique en détail.

L'algorithme de classification hiérarchique ascendante regroupe les données une par une en se basant sur la distance le plus proche parmi l'ensemble des pairs de distance entre les points. Ensuite, la distance entre les points est recalculée. Il existe de nombreuses méthodes disponibles pour le calcul de la distance. On cite par exemple :

1) La distance la plus proche ou le lien unique.

2) La distance du chemin complet le plus long.

3) La distance de la moyenne.

4) La distance du centre de gravité.

5) La distance de Ward : la minimisation de la somme de la distance euclidienne au carré.

Les regroupements de données seront effectués jusqu'à la formation d'un cluster. Ensuite, en se basant sur le graphe du dendrogramme nous pouvons identifier le nombre de cluster qui doit être effectivement présent.

Les étapes algorithmiques de la classification hiérarchique ascendante [45] sont les suivants :

Soit X = x_1,x_2,x_3, ..., x_n l'ensemble de points des données.

Étape 1 : Choisir d'une façon aléatoire A individus comme centres initiaux des classes.

Étape 2 : Attribuer chaque point de données à la classe la plus proche, ce qui définit A classes

Étape 3 : Recalculer les centroïdes de chaque classe.

Étape 4 : Redistribuer les points de données dans la classe qui leur est la plus proche en se basant sur les nouveaux centres de classe calculés à l'étape précédente.

Étape 5 : Refaire l'étape numéro 3 jusqu'à ce qu'il n'y ai plus aucun individu à changer de classe.

La classification hiérarchique de séparation est juste l'inverse de l'approche Hiérarchique Ascendante.

Parmi les avantages de la classification hiérarchique ascendante on note :

- Aucune information a priori sur le nombre de classes nécessaires.
- Facile à mettre en oeuvre et donne le meilleur résultat dans certains cas.

Les désavantages se présente comme suit :

- L'algorithme ne peut jamais défaire ce qui a été fait précédemment.

- Une complexité d'au moins O(n2 log n) est nécessaire, où n est le nombre de points de données.

- Selon le type de matrice de distance choisie pour la fusion, différents algorithmes peuvent souffrir d'un ou de plusieurs des éléments suivants :

  - La sensibilité au bruit et les valeurs aberrantes.
  - La séparation de grands clusters.
  - La difficulté de manipulation des clusters de différentes tailles.

- Il n'existe pas de fonction objective à minimiser directement.

- Il est parfois difficile de déterminer le nombre exact de groupes par le dendrogramme.

L'algorithme de classification hiérarchique ascendante est décrit comme suit :

Algorithm 7 HIERARCHIAL CLUSTERING ALGORITHM

Input : Individuals : individual list, ClassesNB : number of classes we want finally get

Output : Classes : list of classes initially empty, a class is seen as a list of individuals

for i=1 to individuals.length **do**

classes.add(new class($individual[i]$)) ;

end for

while classes.length < ClassesNB **do**

DissmMatrix = new matrix(classes.width, classes.length)

for i=1 to classes.length **do**

for j=i+1 to classes.length **do**

$DissmMatrix[i][j] = dissim(classes[i], classes[j])$

end for

end for

Let (i,j) as $DissmMatrix[i][j] = min(DissmMatrix[k][l])$ with $1 <= k <= classes.length$ and $k + 1 <= l <= classes.length$

for each element in $classes[j]$ **do**

classes[i].add(element) ;

end for

Delete($classes[j]$) ;

end while

4.3.2 Expérimentations

Nos expériences évaluent l'algorithme de classification hiérarchique ascendante sur un réseau social générée.

Configuration des expériences

Toutes nos expériences sont codées avec la technologie JAVA et s'exécutent sous le système d'exploitation Mac OS X 10.5.8, 2.13 Ghz Intel Core 2 Duo avec 2 Go de mémoire.

Pour ces expériences, nous avons choisie arbitrairement 5000 utilisateurs.

Expériences de la vie privée

Nous avons comparé notre échantillon en utilisant l'algorithme k-means, l'algorithme D-max discovering teams et l'algorithme de classification ascendante.

La distance utilisée pour les différents algorithmes est la distance proposée D_{max}.

Nous avons analysé les groupes générés à l'aide de la distance D_{max} et nous remarquons que de 10 à 5000 utilisateurs l'algorithme D-Max discovering teams n'enregistre aucune fuite de données entre les clusters.

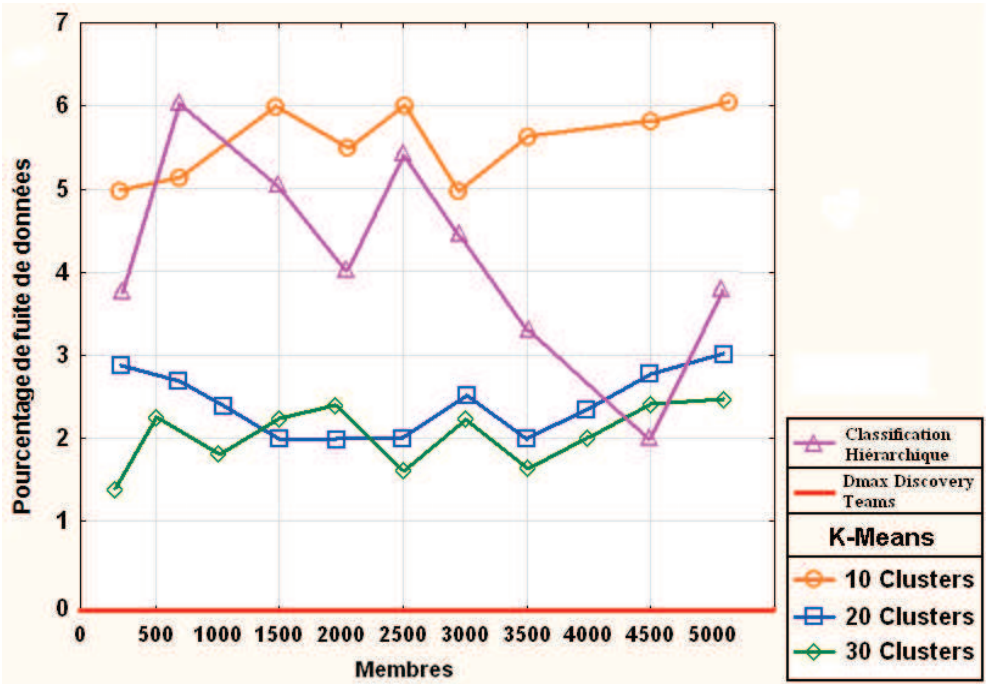

FIGURE 4.6 – Comparaison des fuites de données

Cependant, avec l'algorithme k-means nous percevons des fuites de données variables (a) entre 5 et 6 % en générant 10 clusters, (b) entre 2 et 3 % en générant 20 clusters (c) environ 2% en générant 30 clusters . Et nous enre-

gistrons des fuites de données avec l'algorithme de classification hiérarchique ascendante entre 2 et 7 %.

Ce calcule est basé sur une constance égale à 0.4 représentant le seuil maximum de propagation de données autorisé entre deux utilisateurs du crowdsourcing.

Expériences de temps d'exécution

On a analysé le temps d'exécution de l'algorithme K-means, l'algorithme de classification hiérarchique et l'algorithme D-Max Discovering teams appliquant la distance D_{max} et utilisant entre 10 et 5000 utilisateurs.
Dans la figure 4.7 pour 10 clusters le K-means affiche un temps d'exécution entre 2 et 1702 millisecondes, l'algorithme de classification hiérarchique ascendante indique un temps d'exécution entre 2 et 1608 millisecondes et l'algorithme D-Max Doscovering teams affiche un temps de calcul entre 2 et 2500 millisecondes.

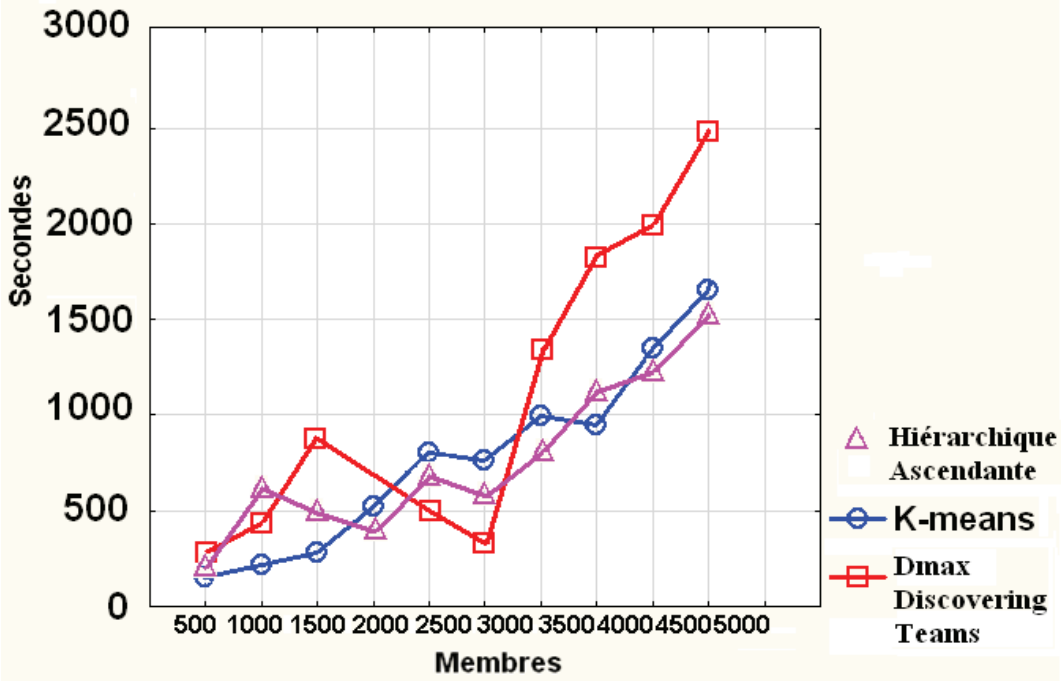

FIGURE 4.7 – Comparaison des différents temps d'exécution

Le temps de calcul de l'algorithme D-Max Discovering teams est plus long avec la distance D_{max} suite à son caractère de fusion des équipes ayant des

probabilités de propagation en dessus du seuil limite autorisé.

L'algorithme D-Max discovering teams avec la distance D_{max} est plus fiable sur la préservation contre la fuite de données par rapport à l'algorithme k-means et l'algorithme de classification hiérarchique ascendante.

4.3.3 Discussion

Compte tenu de toutes les informations présentées sur l'algorithme de classification hiérarchique ascendante, aucune des distances généralement utilisées n'est applicable à notre problème. Même en utilisant la distance maximale entre les membres et les différents clusters, la classification des différents membres ne pourra pas générer des équipes sans fuites de données.

Le fait qu'un membre doit être classifier avec le groupe le plus proche, cela signifie que lorsqu'un membre à plus d'une propagation de données supérieure au seuil maximal autorisé avec certains clusters, il existera une fuite de données entre ces équipes. En plus, l'application de cette approche donne une solution unique qui ne permet pas de répondre à la problématique de base.

4.4 Approche basée sur l'algorithme glouton

Dans cette section, nous fournissons des détails sur le processus de classification qui groupe les utilisateurs du crowdsourcing en clusters sans fuite de données, basés sur la technique de propagation de données définie dans le chapitre précédent, cette approche nous permet d'explorer la totalité des solutions de groupement possibles.

4.4.1 Définitions

Contrairement aux algorithmes existants de classification (comme k-means [MacQueen et al. 1967]) qui génèrent une solution de classification unique, notre approche génère toutes les solutions possibles de regroupement tout en préservant leurs données respectives. Avant de présenter les détails de l'algorithme, nous donnons les définitions suivantes :

Définition 6 *(Cluster) Un cluster C est un ensemble d'utilisateur du crowdsourcing (ayant ou non une forte propagation)*

$$C = m_i \text{ de telle sorte que } \forall m_i, m_j \in C, P_{ij} \in [0,1], p_{ij} \in [0,1]$$

Définition 7 *(Cluster sans fuite de données) Deux clusters C_k et C_s n'ont pas de fuite de données si et seulement si :*

$$\forall m_i \in C_k, \forall m_j \in C_s, p_{ij} \leq \eta_{ji}^{p} \leq \eta$$

Dans la définition 8, on considère qu'il existe un risque de fuite de données entre deux clusters C_k et C_s si le taux de propagation entre tous les utilisateurs des deux clusters est inférieur au seuil n. L'algorithme de classification utilise cette définition de la distance pour générer des clusters sans fuite de données.

Définition 8 *(Cluster sans fuite de données) Nous considérons qu'il existe un risque de divulgation de données si le taux de propagation entre un candidat membre à un cluster et les tous les membres de ce cluster est supérieur à un seuil (fixé par le réseau social). La valeur maximale de propagation entre un cluster et un membre à ajouter dans le cluster est décrite par la valeur Ω qui peut être calculée entre le membre et tous les membres existants du cluster :*

$$\Omega(C_k, m_j) = \underset{m_i \in C_k}{Max}\ p_{ij}$$

où Max la valeur maximale, C un cluster à construire, K est le K-ème membre qui peut être classifié dans le cluster C et p_{ij} est la valeur de propagation entre les membres i et j. Il résulte des définitions 7 et 8 qu'il existe une fuite de données à partir d'un cluster C_k avec un membre m_i si : $\Omega(C_k, m_i) > \eta$

Définition 9 *(la solution de classification) Une solution $S = C_1, \ldots, C_{nn>=1}$ est un ensemble de clusters de telle sorte que :*

$$\forall i, j \in [1, n] C_i \text{ et } C_j \text{ sans fuite de données.}$$

4.4.2 Processus de classification

Notez qu'une solution est incomplète si elle ne contient pas tous les membres, et complète sinon. En utilisant la définition 8, on peut transformer la matrice de propagation (à partir de l'algorithme 2) en une matrice symétrique comme suit :

$$\forall i, j, p_{ij} = Max(p_{ij}, p_{ji})$$

Pour illustrer, la matrice symétrique obtenue de la matrice de propagation calculée dans le chapitre précédent est la suivante :

	Bob	*Alice*	*John*	*David*	*Mickael*	*George*
Bob	1	0.21	0.07	0.1	0.19	0.7
Alice	0.21	1	0.27	0.45	0.9	0.3
John	0.07	0	1	0.2	0.3	0.1
David	0.1	0.45	0.2	1	0, 5	0.13
Mickael	0.19	0.9	0.3	0.5	1	0.27
George	0.7	0.3	0.1	0.13	0.27	1

Comme mentionné précédemment, l'algorithme de classification proposée génère toutes les solutions possibles de regroupement sur la base de la matrice symétrique définie. Les procédés sont les suivants :
Initialisation :
- Un membre m_0 est aléatoirement sélectionné, et fait le membre d'un nouveau cluster C_1. Autrement dit, $C_1 = m_0$.
- La solution partielle initiale est créée ainsi, $S_1 = C_1$.
Utilisant la (les) solution(s) partielle (s) $S_i = C_1, \ldots, C_{nn>=1}$ (i est le nombre d'utilisateurs classifiés dans S_i). L'algorithme classifie un nouveau membre m_i, selon les règles suivantes :

Règles 1 : Si m_i a une fuite de données avec un seul cluster uniquement C_k dans S_i. (C'est-à-dire, $\exists C_k \in S_i \Omega(C_k, m_i) > \eta$ et $\forall C_{j \neq k} \in S_i, \Omega(C_j, m_i \leq \eta$ alors le membre m_i est classifié dans le cluster C_k. La solution partielle S_i est majorée à S_{i+1} de telles sorte que $S_{i+1} = S_i$ où $C_k \cup m_i$.

Règles 2 : si m_i n'a aucune fuite de données avec aucun cluster dans S_i (C'est-à-dire, $\forall C_j \in S_i, \Omega(C_j, m_i) > \eta$), alors le membre est classifié dans

chaque cluster en créant une nouvelle solution pour chaque instance. Autrement dit, S_i augmente à n solutions partielles possibles S_{i+1}^{α} :
$S_{i+1}^{\alpha} = S_i \cup C_\alpha$ où, $C_\alpha \cup m_i$ pour $\alpha \in [1, n]$.

Règles 3 : si m_i à une fuite de données avec deux clusters ou plus dans S_i, c'est-à-dire pour SC_i dans S_i, $SC_i = C_1, \ldots, C_{k2 \leq k \leq n}$ de telle sorte que, $\forall C_j \in SC_i, \Omega(C_j, m_i) > \eta$, alors il existe deux solutions possibles $S_{i+1}^1 et S_{i+1}^2$:

Règles 3.1 Fusionner les clusters $C_1, \ldots, C_n$ et m_i dans un seul cluster $C_g = C_1 \cup \ldots \cup C_k \cup m_i$.

- La solution partielle S_i est majorée S_{i+1}^1 où $S_{i+1}^1 = S_i - C_1, \ldots, C_m \cup C_g$.
Règles 3.2 Calculer le sous-ensemble $L \cup_{C_i \in SC_i} C_i$ de telle sorte que les membres de $L \cup m_i$ ont une fuite directe ou indirecte entre eux. C'est-à-dire $\forall m_x \in L \cup m_i : p_{ix} > \eta$ nous rappelons que $p_{ix} = p_{xi}$ ou, $\exists m_y \in L \cup m_i$ de telle sorte que $p_{iy} > \eta$ et $p_{yx} > \eta$.

- Création d'un nouveau cluster $C_g = L \cup m_i$ et suppression des membres L de leurs cluster respectif. Suppression des clusters vides.

- La solution partielle S_i sera majorée S_{i+1}^2, $S_{i+1}^2 = S_i \cup C_g$.

L'algorithme de classification applique les règles définies ci-dessus jusqu'à ce que tous les membres soient regroupés dans toutes les solutions possibles. L'algorithme est tracé comme suit :

Algorithm 8 Data Leak Free Team Discovery

Input : The symmetric propagation matrix, M : set of all crowd members to be clustered. **Output :** All possible solutions S_i of data free clustering.

{I}nitializations

randomly select m_0 from M ;

create the cluster $C_1 = m_0$;

$S_1 = C_1$ **call** FreeLeakClustering $(S_1, M - m_0)$; *Clustering funtion* **Function** *FreeLeakClustering*(Solution S_i, Members M) ;

if M $= \emptyset$ **then**

Return S_i ;

$\backslash S_i$ is a final solution as there are no remaining members to cluster

end if

if Rule1 : m_i has data leak with only one cluster C_k in S_i **then**

$C_k = C_k \cup m_i$

$S_{i+1}^j = S_i$

call *FreeLeakClustering*$(S_{i+1}, M - m_i)$;

end if

if Rule2 : m_i has no data leak with any cluster in S_i **then**

for each cluster C_j in S_i **do**

$C_j = C_j \cup m_i$

$S_{i+1}^j = S_i$

call *FreeLeakClustering*$(S_{i+1}^j, M - m_i)$;

end for

end if

if Rule3 : m_i has data leak with two or more cluster SC_i in S_i, $SC_i = C_1, \ldots, C_{k2 \leq k \leq n} \subseteq S_i$ **then**

\ First alternative solution Merge all the clusters of SC_i with m_i into the new cluster C_g. that is,

$C_g = C_1 \cup \ldots \cup C_k \cup m_i$;

call *FreeLeakClustering*$(S^1_{i+1}, M - m_i)$;

\ Second alternantive solution compute the subset $L \subseteq \bigcup_{C_i} \in SC_i\ C_i$ of members having direct or indirect data leak between them 'see Rule3.2) ;

Create a new cluster $C_g = L \bigcup m_i$;

for m_j in L **do**

Delete m_j from its cluster in S_i ;

end for

$S^2_{i+1} = S_i \bigcup C_g$

call *FreeLeakClustering*$(S^2_{i+1}, M - m_i)$;

end if

La figure 4.4.1 montre le graphe de résolution généré par l'algorithme de classification DLTD lors de son exécution sur le graphe de la figure 3.2 . Les noeuds internes $(S^*_i, i \leq 5)$ sont des solutions partielles et les noeuds feuilles (S^*_6) sont des solutions définitives quand tous les membres sont regroupés (il existe 6 membres à regrouper pour cet exemple).

Le graphe montre cinq solutions de classification. Dans chaque solution, les membres du réseau social sont regroupés en clusters sans fuite de données, qui peuvent être facilement vérifiés en utilisant la matrice symétrique de propagation de données.

Dans ce qui suit, nous présentons quelques propriétés algorithmiques et leurs preuves intuitives relatives à notre algorithme de classification :

1. *Solidité et correction* : Dans toutes les solutions générées, les clusters sont sans fuite de données. La preuve intuitive découle du principe de base des règles de regroupement, c'est à dire, chaque paire d'éléments avec une propagation de données élevés sont regroupés dans le même

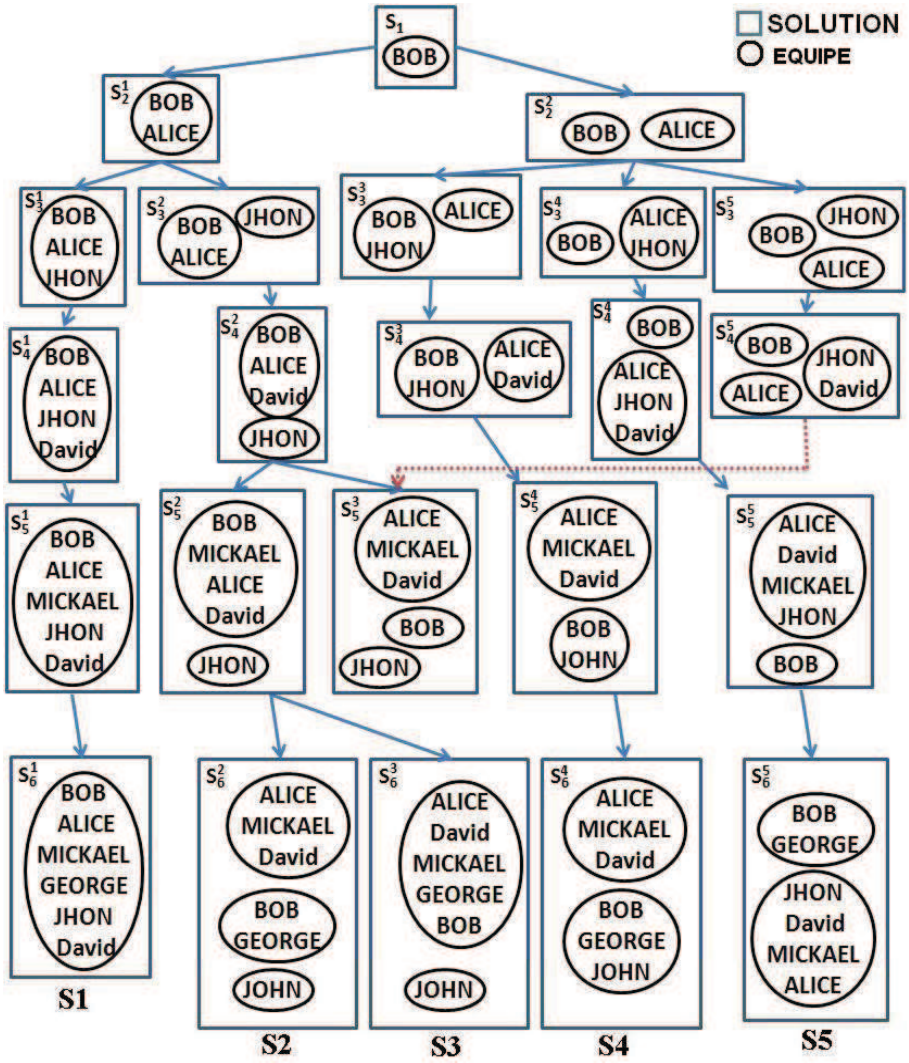

FIGURE 4.8 – Arbre de résolution de solutions

cluster.

2. *Résiliation* : L'algorithme se termine lorsque tous les membres sont regroupés (et aucun membre ne reste sans cluster). A chaque itération, l'algorithme classifie un membre et le supprime de l'ensemble des membres à classifier. Comme l'ensemble des membres à classifier est fini, l'algorithme se termine.
3. *Exhaustivité* : L'algorithme calcule toutes les solutions existantes. La preuve intuitive est que l'algorithme explore toutes les fuites de données des clusters dans les solutions pour chaque membre.
4. *Complexité* : L'algorithme est récursif et composé de trois règles principales : règle 1, règle 2 et règle 3. La complexité de l'algorithme correspond à l'appel principal. FreeleakClustering $(S_0 = \emptyset, M)$. Considérant, $n = |M|$ le nombre d'utilisateur à classifier, et $f(n)$ la complexité de l'appel à la fonction principale de classification. Ensuite, $f(n) = Max(O(Rule1(n)), O(Rules2(n)), O(Rule3(n))$.
   (a) la complexité de la règle 1 : Cette règle est composée d'un test de complexité $O(n)$, et un appel récursif. $FreeLeakClustering(S_{i+1}, M - M_i)$. Ensuite, la complexité de cette règle est : $O(n)+O(f(n-1)) = n + (n-1) + O(f(n-2)) = n + (n-1) + \ldots + O(1) = O(n^2)$.
   (b) la complexité de la règle 2 : Cette règle est la plus complexe car elle fait k appels récursifs à $FreeLeakClustering(S^j_{i+1}, M - m_i)$ où k est le nombre de clusters dans une solution. Ensuite, la complexité de la règle est la suivante : $O(n) + k \times O(f(n-1)) = n + k \times (k-1) \times \ldots \times 1 = O(n + k!)$. Si l'on considère pire des cas théorique $k = n$, alors la complexité de la règle 2 est $O(n!)$.
   (c) La complexité de la règle 3 : Cette règle est composée d'un test de complexité O (n), et deux appels récursifs. Ensuite, la complexité de cette règle est : $O(n)+2\times O(f(n-1)) = n+2(n-1)+O(f(n-2)) = n + 2(n-1) + \ldots + 2 = O(n^2)$.

Par conséquent, la complexité globale de l'algorithme est O(n !). C'est une complexité théorique dans le pire des cas. Dans la pratique, le nombre de cluster k est généralement beaucoup plus petit que n, donc, $O(n^2) < f(n) < O(k!)$.

4.4.3 Expérimentations

Configuration des expériences

Toutes nos expériences sont codées avec la technologie JAVA et s'exécutent sous le système d'exploitation Mac OS X 10.5.8, 2.13 Ghz Intel Core 2 Duo avec 2 Go de mémoire.

Pour ces expériences, notre réseau social est généré sur la base de la loi de Metcalfe. La loi de Metcalfe caractérise la plupart des effets de réseau de technologies et réseaux de communication comme l'Internet, les réseaux sociaux, et le World Wide Web.

L'ancien président de la Commission fédérale des communications des États-Unis, Reed Hundt, a déclaré que cette loi donne plus de compréhension pour le fonctionnement de l'Internet.

La loi de Metcalfe est liée au fait que le nombre de connexions uniques dans un réseau de noeuds (n) peut être exprimé mathématiquement comme le nombre triangulaire n (n - 1) / 2, qui est proportionnel à n2 asymptotiquement. La loi est abondante et inexistante en raison de la capacité des utilisateurs d'Internet à se relier entre eux. Sites et blogs tels que Twitter, Facebook, et Myspace sont au centre de cette prise de loi en vigueur.

Nous employons de façon aléatoire jusqu'à 5000 utilisateurs. De cet échantillon, nous avons construit un ensemble d'utilisateurs afin de préserver la divulgation de données au cours du processus de découverte de l'équipe. L'approche PPCA proposée est utilisée sur l'échantillon pour découvrir les valeurs cachées de propagation de données.

Expérimentations sur la vie privée

A. Expérimentation de classification utilisant la distance euclidienne et la distance Ω avec respectivement le modèle k-means et DLTD :

Nous avons calculé notre échantillon en utilisant deux modèles de classification différents : Les premières expériences étaient avec le modèle K-means en utilisant la distance euclidienne comme c'était décrit dans la section précédente.

La deuxième expérience de classification est avec l'algorithme DLTD basé sur la distance Ω.

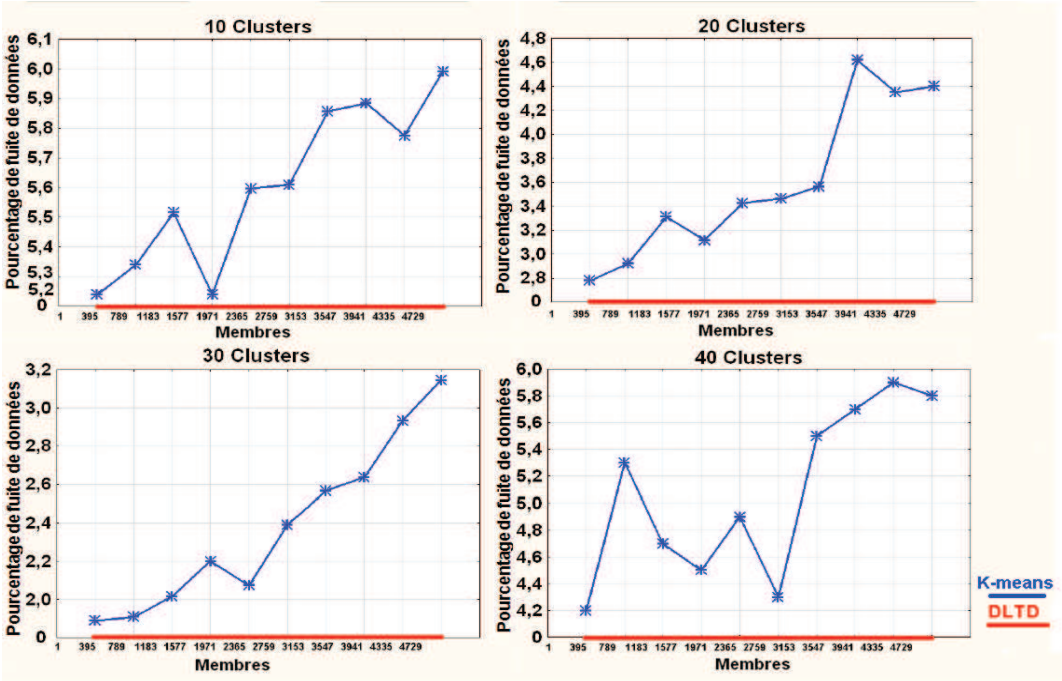

FIGURE 4.9 – Présentation des fuites de données

Après, nous avons analysé les clusters générés par l'algorithme DLTD utilisant la distance Ω et ceux générés par l'algorithme K-means utilisant la distance euclidienne. On note que, pour 10 clusters classifié en utilisant l'algorithme k-means, on s'aperçoit qu'il existe des fuites de données entre 5 et 6 %, entre 2 et 3 % en générant 20 clusters et environ 2% en générant 30 clusters dans le figure 4.9. Ce calcul est basé sur une constante égale à 0.4 le seuil de propagation maximum toléré entre les utilisateurs du crowdsourcing. Par contre, l'algorithme DLTD utilisant la distance Ω n'enregistre aucune fuite de données entre les différents clusters.

Nous avons énuméré le nombre de solutions retourné en utilisant l'algorithme de DLTD et nous avons testé la fuite de données entre chaque composition d'équipe. Lorsque nous conservons la même structure de réseau social et nous calculons une deuxième fois, nous recevons un ensemble de solutions identiques à celui du première calcul, qui démontrent que notre approche est stable et il n'est pas un calcul au hasard.

B. La classification en utilisant k-means et DLTD basés sur la distance Ω

On a comparé les fuites de données entre les clusters générés utilisant l'algorithme k-means et l'algorithme DLTD utilisant la distance Ω. Le seuil de fuite de données est fixé à 40%. Avec la distance Ω l'algorithme DLTD n'enregistre aucune fuites de données entre les clusters compétitifs même lorsque le nombre d'utilisateur dans le réseau social augmente.

Ces résultats sont solides parce que les membres avec une propagation de données élevée (supérieur à 40%) sont groupés dans le même cluster. Au contraire, avec le k-means utilisant la distance Ω, on remarque des fuites de données entre 3% et 4% pour 20 clusters et entre 4% et 6% pour 30 clusters (Figure 4.10), parce que le processus de k-means ajoute le membre dans l'équipe avec la quelle il justifie un maximum de propagation de données, même s'il existe une propagation de données supérieure au seuil avec d'autres équipes. Par exemple, supposons qu'on a 3 équipes au moment de la classification du membre U_x, ce membre U_x à une propagation de données de 45%, 46%, 47% respectivement avec $equipe_1$, $equipe_2$, $equipe_3$, Le processus de K-means classifie l'utilisateur U_x avec $equipe_3$ alors qu'il existe une fuite de données avec $equipe_1$ et $equipe_2$ en ajoutant U_x à $equipe_3$ la propagation de données entre U_x et $equipe_1 >$ seuil et U_x et $equipe_2 >$ seuil.

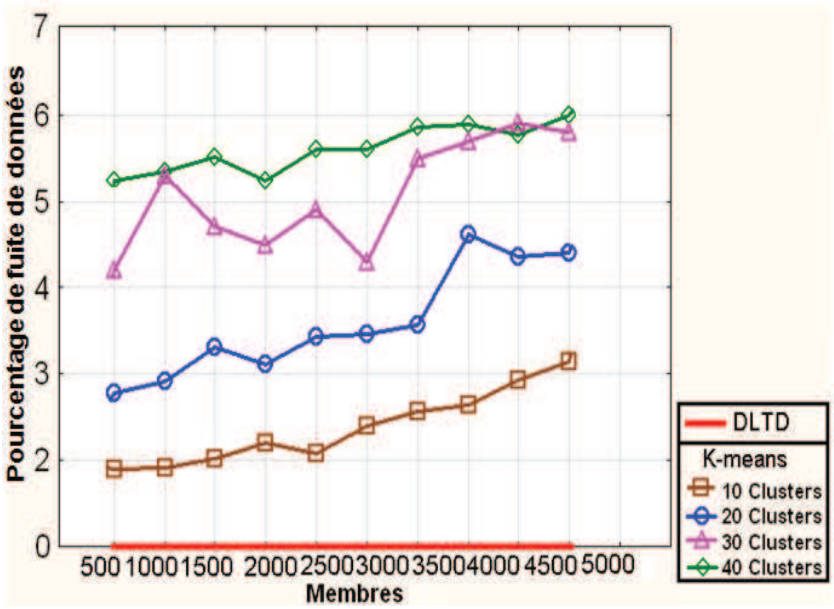

FIGURE 4.10 – Comparaison des fuites

C. Le résultat de la génération de solutions en variant le seuil et le nombre d'enregistrement

Dans cette expérience, on a fixé le seuil de fuite de donnés à 35% et on a varié le nombre d'utilisateurs enregistrés. Avec 1000 membre enregistrés pour collaborer afin de résoudre la requête on obtient 41 solutions sans fuite de données, et pour 5000 enregistrement, on obtient 146 solutions possibles (figure 4.11-A).

Après, on a fixé le nombre d'enregistrement à un appel à 1000 et on a varié le seuil de fuite de données afin de découvrir le nombre de solutions possible sans fuite de données en utilisant la même infrastructure du réseau social.

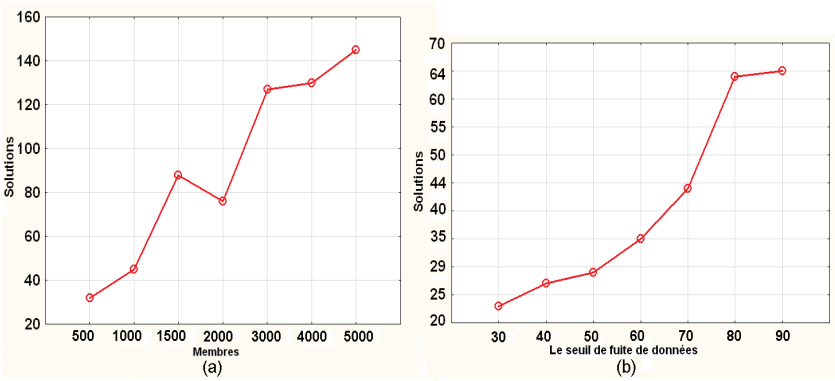

FIGURE 4.11 – Les résultats du DLTD

Par exemple, quand le seuil est fixé à 30% l'algorithme DLTD retourne 24 solutions possible sans fuite de données, en utilisant le même réseau social, les mêmes membres et les mêmes propagations. En variant le seuil de fuite de données à 70%, on obtient 43 solutions possible sans fuite de données (figure 4.11-B).

Ces expériences nous montrent que le nombre solutions générées dépend toujours des paramètres de classification comme le nombre d'utilisateurs à classifier et/ou le seuil de propagation autorisé entre les utilisateurs.

Étant donnée le grand nombre de solutions possible, il faut utiliser les préférences du demandeur pour choisir la meilleure solution à retourner.

Expérimentation sur le temps d'exécution

On a analysé le temps d'exécution de l'algorithme DLTD et K-means utilisant la distance Ω de 1000 à 5000 utilisateurs.

On a calculé le temps d'exécution de 12497500 relations dans un réseau social, ce qui signifie environ 5000 membres, basé sur la loi Metcalfe, nous observant que notre algorithme n'enregistre aucune fuite de données par contre le temps d'exécution est supérieur à celui du k-means.

Dans la figure 4.12, le temps d'exécution est entre 153 et 1653 secondes pour le K-means et entre 172 et 2631 pour le DLTD, ensuite, il est meilleur pour le k-means que pour le DLTD.

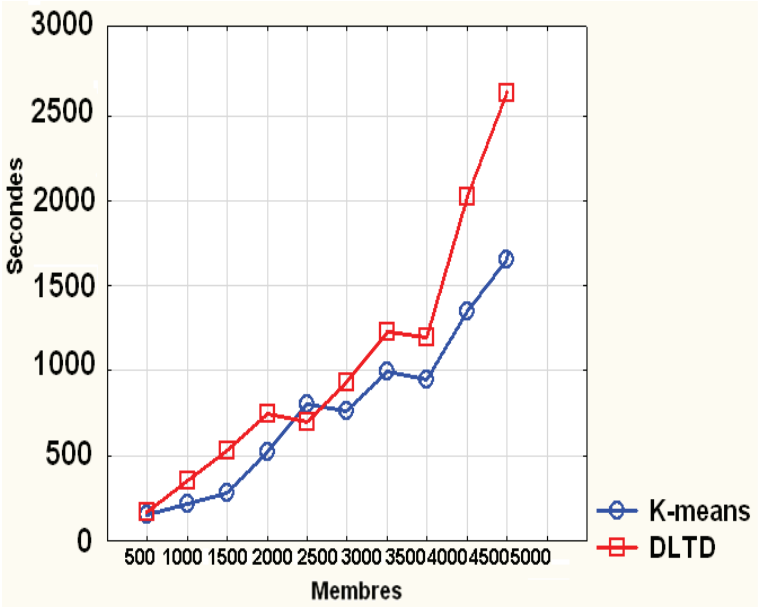

FIGURE 4.12 – Temps d'exécution

Le temps d'exécution de l'algorithme DLTD est plus important que le k-means parce que le DLTD découvre toutes les solutions possible alors que le K-means produit une solution unique.

En outre, les résultats de K-means présentent des fuites de données importantes entre les différentes équipes en utilisant la distance Ω, en raison, du mécanisme de classification qui classent le membre dans l'équipe la plus proche, ce qui veut dire, l'équipe avec laquelle le membre a la valeur de la propagation la plus forte, même s'il existe d'autres propagations (avec d'autres équipes) dépassant le seuil fixé.

Le temps d'exécution de l'algorithme k-means est causé par le processus de convergence. Le k-means répète la même phase de classification jusqu'au convergence. Le nombre de répétition est variable, dans notre cas, il dépend du nombre des membres à classifier et leurs propagations de données.

La convergence du k-means est atteinte lorsque la comparaison des deux dernières itérations révèle que les membres ne se déplacent plus entre les équipes, ainsi, le calcul du k-means atteint sa stabilité et aucune itération n'est nécessaire. De plus, la convergence à un minimum local peut produire un mauvais résultat. Le critère de convergence de l'algorithme DLTD est atteint quand il n'existe pas de membre à classifier.

4.4.4 Discussion

Dans cette section nous avons proposé une nouvelle approche afin d'énumérer toutes les solutions de classifications possibles dans le but de choisir la meilleure solutions de groupement à retourner au demandeur de la requête. (cf. section 1.1)

Notre approche est nettement meilleure sur l'exploration des solutions de classification possible par rapport au K-means qui n'est pas adapté à notre problème et nous retourne une seule solution.

Les résultats montrent que les solutions retournées par l'algorithme DLTD n'enregistrent aucune fuite de données entre les équipes compétitives et collaboratives.

Cependant, l'algorithme DLTD enregistre un temps de calcul théorique très important, dans la pratique, ce temps est beaucoup plus petit.

L'avantage de cette approche est d'explorer toutes les solutions de classifications possibles pour retourner la meilleure au demandeur de la requête.

En plus, cette approche ne présente aucune fuite de données, donc assure la compétitivité entre les équipes.

Le problème de cette approche est qu'elle n'est pas exploitable sur un réseau social de grande taille puisqu'elle explore toutes les solutions de classification possibles.

Un travail doit être effectué pour réduire le temps de calcul de cette approche. Dans la prochaine section, nous proposons une nouvelle approche de classification en se basant sur un modèle parallèle.

4.5 Classification basée sur le modèle parallèle

4.5.1 Le principe

Commençant à partir des probabilités de propagation de données calculées dans la section précédente, le processus de classification va grouper les utilisateurs du crowdsourcing ayant une forte propagation dans le même cluster en utilisant la technique de parallélisation. Cela signifie que, plus la probabilité de propagation de données entre deux membres du crowdsourcing est plus élevée, plus ils doivent être dans le même groupe pour la collaboration et non pas dans des groupes de compétitivité.

4.5.2 Définitions

La phase de découverte des l'équipes sera traitée en utilisant l'algorithme DLTD parallèle, un algorithme de classification avec la même distance Ω utilisée précédemment afin d'éviter les fuites de données.

L'algorithme DLTD parallèle procède comme suit :

1- Initialisation : Le seuil de divulgation de données est fourni au début. L'initialisation du premier cluster est faite en attribuant arbitrairement un membre à ce groupe.
2- Itérations : Pour chaque nouveau membre, la distance qui le sépare à chaque cluster est calculée.
Cette distance représente la valeur maximale de propagation avec les membres du cluster. A chaque itération, il est possible de fusionner, supprimer ou créer un nouveau cluster pour gérer le grand nombre de données en utilisant des processus parallèle. L'idée de base derrière le système parallèle est le mappage de l'ensemble de données en une collection de paires de données $< key, value >$, puis la réduction sur toutes les paires avec la même clé.

La système parallèle est utilisée pour classifier un nouveau membre à l'aide des solutions partielles en entrée. Avec cela, nous parallélisons la phase

de regroupement dans des "'Processus"' différents à chaque classification de membre, il est possible que des "'Processus"' différents produisent les mêmes solutions partielles, par conséquent, l'objectif du "'Collecteur"' est de garder une occurrence de chaque solution partielle.

• Un processus est lancé en parallèle à chaque paire dans le jeu de données d'entrée. Cela produit une liste de paires pour chaque appel. Après cela, le collecteur recueille toutes les paires avec la même clé de toutes les listes et les regroupe, en créant une clé pour chaque groupe.

La $inputMap_k < key, value >$ devient dans notre cas $< UserNumber$, $PartialSolutions_k >$,k est le k^{eme} itération de classification de membre, la Map Générée $OutputMap_k < key, List(key, value) >$ va être définit comme suit :$< UserNumber, List(< HashCode_k, PartialSolution_k >) >$ où, HashCode est le code unique qui identifie le résultat de partialSolution à la k^{eme} itération tout en tenant compte de sa structure. $PartialSolution_k$ représente le résultat de classification du membre désigné par "'clé"'.

Le but du processus parallèle est de modéliser les données en paires de $< key, value >$ pour que le "'Collecteur"' puisse les agréger. Les paires $< Key, value >$ produites sont alors "mélangées", ce qui signifie essentiellement que les paires avec la même clé sont regroupées et transmises à une seule machine, qui ensuite exécutera le script "'Collecteur"' sur eux.

• Le "'Collecteur"' combine les valeurs d'une clé. Dans notre cas, il ne garder qu'une seule occurrence de chaque solution (ou des solutions partielles) et retourne uniquement les solutions partielles distinctes pour chaque itération d'une nouvelle classification d'un membre.

La figure 4.13(a) décrit l'application du système parallèle et 4.13(b) décrit sont implémentation.

Le processus commence avec une solution partielle contenant le premier membre à classifier et se termine avec le dernier membre classifié.

3- Stop : L'algorithme est conçu pour converger quand nous n'avons aucun membre à regrouper.

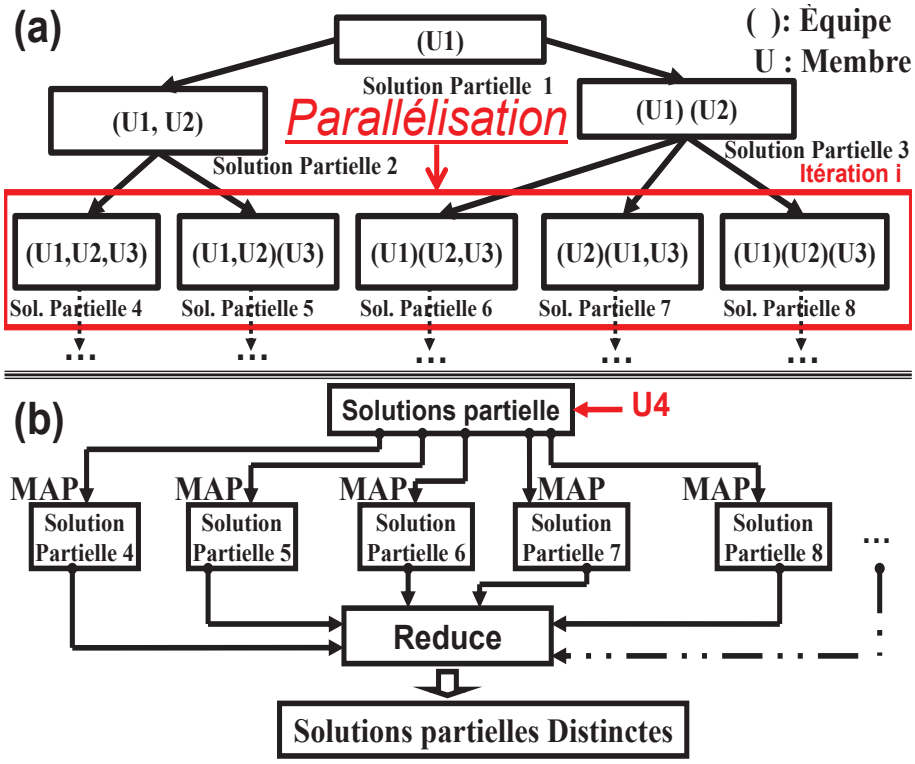

FIGURE 4.13 – Classification parallèle des membres

4.5.3 Les expérimentations

Pour ces expériences, nous avons utilisé un réseau social généré sur la base de la loi de Metcalfe avec 12497500 relations pour 5000 utilisateurs. Nous avons construit un ensemble de solution avec tous les utilisateurs afin de préserver la divulgation de données lors du processus de découverte des équipes. L'approche PPCA proposée est utilisée sur l'échantillon pour découvrir les valeurs cachées de propagation de données.

- k-means et DLTD parallèle basé sur Ω :

On a analysé les différents résultats générés par l'algorithme k-means utilisant la distance Ω et les productions de l'algorithme DLTD parallèle utilisant la même distance Ω. Notre analyse est focalisée sur les fuites de données privées.

Le seuil de fuite de données est toujours fixé à 40%. Avec la distance

Ω l'algorithme DLTD parallèle n'enregistre aucune fuite de données entre les clusters générés même quand la taille du réseau social augmente. L'inexistence de fuite de données s'explique par le fait que les membres qui se partagent les données privées avec une propagation supérieure au seuil fixé par l'algorithme seront groupés dans le même cluster. Par contre, avec le K-means utilisant la distance Ω, nous percevons des fuites de données privées entre 1% et 4% pour 20 clusters et de 2% à 5% pour 30 clusters (voir la figure 4.14).

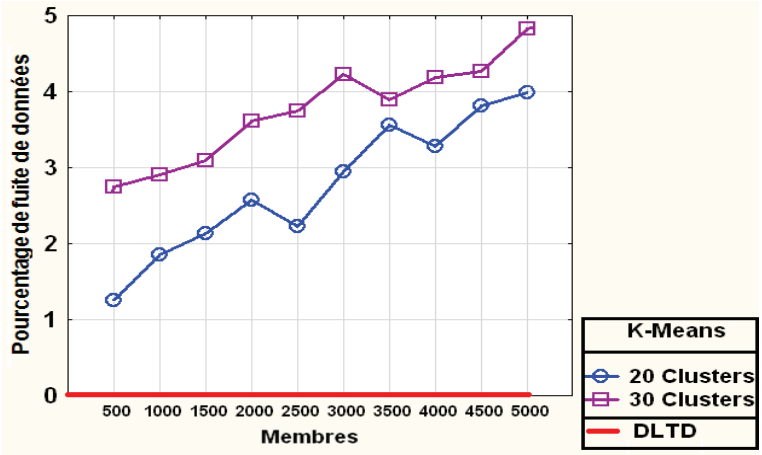

FIGURE 4.14 – Comparaison des fuites de données

- Les solutions résultat en variant le seuil et le nombre d'enregistrement :

Dans une première expérience, on a fixé le nombre d'utilisateur du crowdsourcing enregistrés pour répondre à la requête, le seuil est fixé à 68% et on fait varier le nombre de membres enregistrés. Avec 1000 enregistrement, on obtient 158 solutions sans fuite de données, et pour 5000 enregistrement, on obtient 241 solutions possibles (voir figure 4.15-a).

Dans une deuxième expérience, on a fixé le nombre d'enregistrement pour la réponse à la requête à 1000 et on a fait varier le seuil maximum de propagation autorisé dans le but de découvrir le nombre de solutions possible sans fuite de données en utilisant le même réseau social.

Par exemple, quand le seuil est fixé à 10% l'algorithme DLTD parallèle retourne 13 solutions possibles sans fuite de données, en utilisant la même

structure du réseau et en variant le seuil à 45%, on obtient 88 solutions possibles sans fuite de données (Voir figure 4.15-b).

Ces expériences nous permet de remarquer que les solutions générées sont dépendantes du nombre d'utilisateurs et du seuil de propagation des données ainsi que les valeurs de propagations des données.

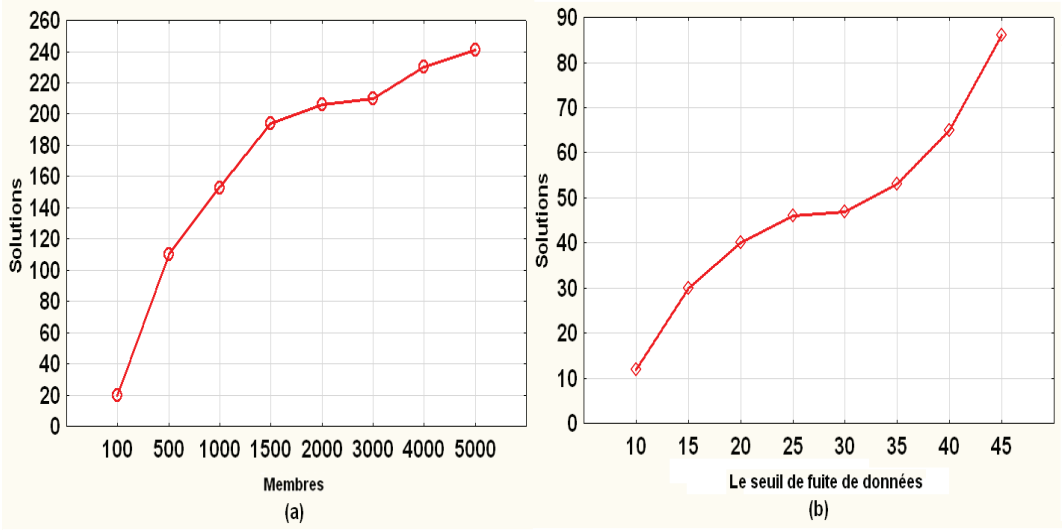

FIGURE 4.15 – Les résultats du DLTD parallèle

Ces expériences nous montrent que l'algorithme DLTD parallèle n'enregistre aucune fuite de données entre les clusters puisque sont mécanisme se base sur le groupement des utilisateurs du crowdsourcing ayant une forte propagation de données entre eux, qui interdit la dispersion de ces utilisateurs dans différents clusters.

Les résultats du DLTD parallèle montré dans la figure 4.15 confirme que les solutions générées par cet algorithme dépendent du nombre d'utilisateur à classifier et du paramétrage du seuil limite de propagation autorisé sans pouvoir l'identifié comme fuite de données.

L'algorithme DLTD parallèle enregistre une nette amélioration du temps de calcul par rapport au DLTD classique, suite à l'utilisation du système parallèle le traitement en n processus différents puis rassemble les résultats par clusters.

Ce traitement nous permet de gagner en temps d'exécution.

Par contre, le temps de calcul reste toujours important et n'est pas facilement applicable sur un réseau social de grande taille (exemple Facebook contenant des milliards de relations).

Même en utilisant le modèle parallèle, cela ne permet pas d'alléger ce temps de calcul en une valeur acceptable pour un réseau social important.

L'application de cette approche reste théorique, parce qu'il impossible d'allouer une ressource matérielle pour chaque processus du modèle parallèle (un très grand nombre de machine à prévoir), pour arriver à un temps d'exécution acceptable.

4.6 Discussion

Dans ce chapitre nous avons présenté différentes techniques pour chercher les groupement possibles des utilisateurs du crowdsourcing pour répondre à un appel d'une personne physique ou morale.

L'algorithme K-means ne permet pas d'arriver à produire des résultats fiables sans fuite de données entre les clusters. L'adaptation de l'algorithme K-means nous a permit d'arriver à produire des résultats respectant la vie privée entre les clusters, par contre, cela ne permet pas d'explorer les meilleures solutions possibles de classement.

L'algorithme de classification hiérarchique n'offre pas la possibilité des générer des équipes sans fuites de données même en utilisant la distance maximale de propagation. De même il permet de découvrir uniquement une solution de groupement possible et ne permet pas l'exploration des différentes possibilités de classification.

L'approche DLTD nous a permit d'explorer toutes les solutions possibles de classification, sauf, que cette approche basée sur le modèle glouton, est très lente en temps de calcul, qui nous amène à utiliser une technique de parallélisation dans l'approche du DLTD parallèle afin de réduire le temps d'exécution de l'algorithme.

Les approches proposées ne nous permettent pas de clôturer le sujet du classement des utilisateurs du crowdsourcing dans des équipes compétitives et collaboratives, puisque, on n'est pas encore arrivé à avoir des solutions sans fuites de données en explorant un ensemble important de solutions possibles dans un temps de calcul raisonnable.

L'utilisation des heuristiques s'imposent dans notre cas pour réduire le temps de calcul de l'algorithme et l'exportation de différentes solutions de groupement.

Chapitre 5

La découverte des équipes en préservant la vie privée basée sur les heuristiques

Dans ce chapitre, nous nous intéressons au domaine de simulation. Compte tenu du grand nombre de possibilités de classification, il n'est pas possible d'explorer toutes les possibilités de groupement des membres et garder ceux satisfaisant les critères de fuite de données. Le but de cette section est d'expliquer le besoin d'utiliser des heuristiques pour estimer les solutions optimales et de proposer un modèle de découverte d'équipes compétitives et collaboratives.

5.1 Le recours aux heuristiques

Nous démontrons que le problème de formation d'équipes compétitives et collaboratives est NP complet en le réduisant d'abord à un problème d'objectif simple. Nous supposons que pour une compétence particulière "'s"' tous les experts avec un niveau d'expertise en dessus d'un certain seuil sont également bien convenu. Ainsi, le problème est réduit à la découverte de la structure d'équipe avec l'intermédiaire de liens plus fort entre tous les membres.

Ce problème est lié à la détermination d'une clique dans un graphe pondéré. Cependant, il est peu clair combien d'experts font partie de la meilleure équipe.

Aussi il est peu probable que cette équipe expose en réalité un sous-graphe entièrement connecté. Par conséquent, nous ne pouvons pas encore chercher directement la meilleure clique. Nous modélisons les compétences et la structure d'interaction comme un graphe orienté.

Un graphe orienté est un graphe dont les arêtes sont définies par leur origine et leur extrémité, c'est-à-dire dont les arêtes sont orientées, munies d'un sens. Une arête d'un graphe orienté est définie par la donnée d'un couple de sommets.

Pour notre problème, l'ensemble de compétences exigées correspond aux ensembles de k clusters à découvrir. En fin de compte, chaque ensemble consiste en un tuple qui contient la ou les compétences "'s"' exigées. Tous les experts fournissent cette ou ces compétences tout en préservant la vie privée de chaque équipe.

Une sélection d'un cluster à partir d'une solution de classification constitue une équipe valable où chaque membre peut être connecté ou non à chaque membre dans la même équipe, et doit avoir une propagation inférieure au seuil maximum de propagation autorisé avec les membres des autres équipes. En aplanissant le graphe orienté dans un graphe régulier : (i) n'importe quelle équipe valable sera une clique et (ii) la taille de clique maximale est k. La meilleure équipe est alors une question de découverte de la clique minimale-pondérée en fonction de la propagation des données entre les différents membres. Ceci est trivialement transformé dans un problème de clique maximal en inversant les poids des arêtes. Il est généralement connu que le problème de clique maximal est NP-complet. Ainsi, nous pouvons déduire aussi que le problème de découverte d'équipe observé est NP-complet.

5.2 Approche proposée basée sur les heuristiques

Avant de commencer à expliquer l'approche proposée nous choisissons de faire une comparaison entre les algorithmes génétiques et le Recuit Simulé,

les deux méta-heuristiques les plus utilisées dans ce type de problème.

5.2.1 Le recuit simulé et l'algorithme génétique

Le recuit simulé est souvent présenté comme la plus ancienne des méta-heuristiques, en tout cas, la première à mettre spécifiquement en oeuvre une stratégie afin d'éviter des minima locaux (Kirkpatrick, Gelatt, Vecchi, 1983). Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnistes. Leur but est d'obtenir une solution approchée à un problème d'optimisation, lorsqu'il n'existe pas de méthode exacte pour le résoudre en un temps raisonnable.

Théoriquement , SA (Recuit Simulé) et GA (Algorithme Génétique) sont assez proches, et une grande partie de leur différences est superficielle . Les deux approches sont généralement formulées de manière très différente ,en utilisant une terminologie très différente.

Avec SA , on parle généralement de solutions , leurs coûts et leurs voisins et des déplacements ; tandis que avec GA , on parle de personnes (ou des chromosomes) , leur conditions physique, la sélection , le croisement et la mutation .

Cette différence dans la terminologie reflète les différences d'accent, mais sert aussi à masquer les similitudes et les différences réelles entre SA et GA .

Fondamentalement , SA peut être considéré comme GA où la taille de la population est fixe. La solution en cours est le seul individu de la population. Comme il n'y a qu'une seule personne , il n'y a pas de croisements , mais uniquement des mutations.

La principale différence entre SA et GA est que SA crée une nouvelle solution en modifiant une seule solution avec un déménagement local ,or que, GA crée également des solutions en combinant deux solutions différentes. Si cela fait effectivement le meilleur algorithme ou le pire, cela dépend du problème et de la représentation.

Il convient de noter que les deux algorithme SA et GA partent de l'hypothèse fondamentale que les bonnes solutions sont probablement trouvé " près " des bonnes solutions déjà connues par la sélection au hasard de l'ensemble de l'espace de solutions .

Le GA diffère par le traitement des combinaisons de deux solutions existantes comme étant "proche" , en faisant l'hypothèse que ces combinaisons (enfants) partagent les propriétés de leurs parents , de telle sorte qu'un enfant de deux bonnes solutions est probablement meilleur qu'une solution aléatoire .

Cela dépend évidemment de l'opérateur de croisement. S'il est mal choisi par rapport au problème et sa représentation , une recombinaison sera effectivement une solution aléatoire. Comme indiqué dans [120] , ce type de croisement entraîne souvent une destruction des problèmes combinatoires si un chromosome exprime directement une solution.

En outre, il convient de noter que le poids relatif accordé à la mutation et la recombinaison est un paramètre crucial qui affecte ce que l'algorithme génétique fait en réalité.

Du point de vue pratique, il convient de noter que, pour certains problèmes , l'évaluation des solutions qui sont à proximité d'une solution existante peut être très efficace , ce qui peut donner un grand avantage de performance de SA , par rapport à l'AG , si l'évaluation des solutions recombinés n'est pas si efficace.

Il est surprenant de voir que le temps d'exécution est souvent négligé dans les comparaisons empiriques des algorithmes d'optimisation pour les problèmes combinatoires . Il devrait être un élément clé d'une telle comparaison .

S'il n'y avait pas de limites sur le temps d'exécution , on peut toujours effectuer une recherche complète , et obtenir la meilleure solution possible .

La plupart des algorithmes stochastiques peuvent faire la même chose , sans limite de temps donnée. Dans la pratique , il y a toujours des limites sur le temps d'exécution .

En outre, une propriété clé des algorithmes stochastiques tels que les SA et GA est le temps , ils fournissent habituellement des meilleures solutions en leurs allouant plus de temps, au moins jusqu'à une certaine limite. Si , dans une comparaison empirique , l'algorithme A est autorisé à utiliser plus de temps que l'algorithme B , leurs qualités de solution ne sont plus comparables , car il n'y a aucune indication sur la façon dont une bonne solution

de l'algorithme A était produite en même temps que l'algorithme B.

Il y a quelques conseils dans la littérature pour SA et GA, des comparaisons expriment que SA est un " starter rapide " qui obtient de bonnes solutions dans un court laps de temps , mais n'est pas en mesure d'améliorer ses solutions en lui allouant plus de temps , alors que GA est un "starter lent " qui est capable d'améliorer la solution constamment lorsqu'il a plus de temps.

Mann et le Forgeron [93] comparent SA et GA pour un problème de routage. Ils rapportent les temps d'exécution, la comparaison se concentre principalement sur des coûts de solution. Les temps d'exécution du GA étaient de 10 à 24 fois plus longtemps que ceux du SA. Ils rapportent que GA a donné des solutions légèrement meilleures que SA, mais ils notent aussi que le SA a réalisé ses solutions beaucoup plus rapides. Cependant, ils ne rapportent pas ce qui arrive si on donne le même temps aux algorithmes. Dans le papier [77], il y a une bonne discussion sur la façon de faire une comparaison empirique significative. À titre d'exemple, ils considèrent un problème de minimisation de coût d'arbre. Ils comparent plusieurs algorithmes incluant SA et GA et normalisent soigneusement le temps d'exécution donné aux différents algorithmes. Leurs résultats indiquent que, étant donné le même temps, systématiquement, SA a donné de meilleures solutions que GA.

Sachant que notre problème est d'avoir des meilleures solutions dans un temps relativement limité nous avons choisie de se baser sur le modèle de recuit simulé.

5.2.2 Approche de classification basée sur les heuristiques

Dans cette section, nous présentons l'algorithme **PMTD** basé sur une distance spécifique que nous avons fourni appelée Ω et le modèle de recuit simulé, afin de préserver la divulgation de données au cours du processus de découverte des équipes.

Nous proposons un modèle dynamique qui traite non seulement la relation statique entre les membres du crowdsourcing mais aussi les taux effectifs de données partagées. Notre algorithme est présenté comme suit :

Algorithm 9 PROBABILISTIC METAHEURISTIC TEAM DISCOVERING (PMTD)

Require: t – temperature
f – Objective function
SL – Solutions List {I}nitializations
$t = T_0$
$x = RSGresult$
$x_{best} = x$
$SL = null$
while stopping criterion is not met **do**
  for $iter := 1$ **to** max **do**
    s $\longleftarrow$ Generate_Neighbor_Solution(x)
    if $f(s) \prec f(x)$ **then**
      $x \equiv s$
      $SL \longleftarrow add(s)$
      if $f(s) \prec f(x_{best})$ **then**
        $x_{best} \equiv s$
      end if
    else
      if $random \prec \exp\{-(f(s) - f(x)) \div t\}$ **then**
        $x \equiv s$
        $SL \longleftarrow add(s)$
      end if
    end if
  end for
  $t \longleftarrow \alpha \times t$
end while

L'algorithme de classification proposé comporte trois phases :

1. La matrice de propagation fournie par l'algorithme **PPCA** sera modifiée pour considérer maintenant un graphe non orienté, mettre à jour les relations entre deux utilisateurs avec la valeur élevée de leur propagation. Par exemple, la probabilité entre $(Bob, Alice = 0,2)$ et $(Alice, Bob = 0)$ et leur max est de 0,2. Le maximum sera le critère utilisé pour éviter les fuites de données entre les clusters.

   Cela signifie que, plus la probabilité de propagation est élevée, plus les utilisateurs doivent être dans le même groupe pour la collaboration et non dans des groupes différents. La matrice est mise à jour comme suit :

$$\begin{array}{c} \\ Bob \\ Alice \\ John \\ David \\ Mickael \\ George \end{array} \begin{array}{c} \begin{array}{cccccc} Bob & Alice & John & David & Mickael & George \end{array} \\ \begin{pmatrix} 1 & 0.21 & 0.07 & 0.1 & 0.19 & 0.7 \\ 0.21 & 1 & 0.27 & 0.45 & 0.9 & 0.3 \\ 0.07 & 0 & 1 & 0.2 & 0.3 & 0.1 \\ 0.1 & 0.45 & 0.2 & 1 & 0,5 & 0.13 \\ 0.19 & 0.9 & 0.3 & 0.5 & 1 & 0.27 \\ 0.7 & 0.3 & 0.1 & 0.13 & 0.27 & 1 \end{pmatrix} \end{array}$$

2. Générer une solution aléatoire et unique tout en préservant la "'Privacy"' entre les équipes. L'algorithme de génération de solution aléatoire (RSG) va découvrir une solution unique, tout en préservant la vie privée.

   L'algorithme RSG groupe les utilisateurs du crowdsourcing fondamentalement sur la propagation de données tout en se basant sur la distance Ω, les utilisateurs qui ont de forte propagation de données (supérieur au seuil fixé par le RSG) seront classés dans la même équipe.

   L'algorithme RSG attribue un utilisateur du crowdsourcing à une équipe spécifique en calculant sa propagation de données avec toutes les équipes existantes de la solution, il peut créer de nouvelles équipes, fusionner ou supprimer d'autres équipes.

   Lorsqu'un utilisateur a une propagation faible (en dessous du seuil fixé) l'algorithme RSG attribue au hasard l'utilisateur dans un nouveau groupe ou dans une équipe existante.

3. La découverte de différentes solutions, basée sur l'heuristique recuit simulé.

Notre algorithme est initialisé avec la solution générée par RSG. Ensuite, il découvre différentes solutions qui préservent la "'Privacy"'.

Le processus du PMTD permet de créer de nouvelles solutions avec différents assemblages permettant de préserver la vie privée. Il retourne les différentes solutions conservant l'intimité de chaque équipe.

Nous donnons les principales étapes de l'algorithme proposé sur la base de recuit simulé pour générer les meilleures solutions de regroupement de membres :

- Initialisation : l'initialisation est définie comme suit :
  1. Le PMTD commence à une température élevée T_0 et la température est légèrement réduite en vertu d'un certain mécanisme appelé le cycle de refroidissement lorsque la procédure se poursuit.

     Nous utilisons le cycle de refroidissement exponentiel, $t_i = \alpha \times t_{i-1}$, où $\alpha \in (0, 1)$ est le taux de décroissance de la température. L'idée fondamentale est de générer une nouvelle séquence voisine λ en se basant sur des règles aléatoires et la séquence principale S_0. La nouvelle solution λ est évaluée par un mécanisme appelé critère d'acceptation pour décider soit l'acceptation de λ soit son rejet.

     De toute évidence, si λ améliore S_0, elle est acceptée. En outre, la solution pire peut être acceptée avec une probabilité en fonction de la différence entre la qualité des deux solutions et de la température actuelle t_i. Par conséquent, il y a une chance plus élevée d'accepter la solution pire lorsque les températures sont plus élevées.
  2. La solution de début : l'algorithme génère une solution de début aléatoire pour explorer S_0.
- Explorer des solutions aléatoires : l'algorithme génère aléatoirement des solutions à explorer. A chaque itération, il dérive une nouvelle solution S_i de la solution S_{i-1} de l'itération précédente en déplaçant aléatoirement un membre du crowdsourcing entre deux clusters.
- Acceptabilité de la solution : L'acceptabilité de la solution générée S_i est mesurée par le biais de la fonction objective $f(S_i)$. La fonction ob-

jective utilisée combine plusieurs critères, à savoir la fuite de données, le recouvrement des spécialités et la collaboration dans l'équipe en une valeur de notation. Elle est donnée comme suit :

$$f(S_i) = \begin{cases} \infty & \text{Si } \exists\ Fuite\ De\ Donneés \\ \text{Moyenne (Fuite De Données)} + & \\ \text{Moyenne (Collaboration dans l'équipe)} & \\ + \text{ Taux De Couverture (spécialités)} & \text{Sinon} \end{cases}$$

où $Moyenne(Collaboration\ dans\ l'equipe)$ mesure la collaboration dans les équipes de S_i, $Moyenne(Fuite\ De\ Donnees)$ mesure la fuite de données entre les équipes compétitives et, $Taux\ De\ Couverture(specialites)$ conçu par :

$$\frac{nombre\ de\ specialites\ dans\ la\ solution}{\sum\ des\ specialites}$$

5.2.3 Algorithme de génération de solution aléatoire (RSG)

Une matrice D contenant les paires de distances $\Omega(x_n, x_m)$ entre tous les membres du crowdsourcing est calculée ; la distance proposée Ω représente la valeur de propagation maximale comprise entre les membres du crowdsourcing. Le critère de divulgation est fondé sur la *propagation de données*.

Ainsi, la distance entre deux membres du réseau social est la valeur de propagation entre eux : $d(m_i, m_j) = P_{m_i,m_j}$.

où P est la valeur de propagation entre m_i et m_j calculée dans la première phase et d est la distance séparant deux membres du crowdsourcing qui signifie la valeur de la propagation des données entre eux.

La phase de regroupement sera traitée sur la base des étapes suivantes :

(1) La fixation du seuil initial de la divulgation de données (c'est à dire, nous considérons qu'il existe un risque de divulgation de données si le taux de propagation entre un candidat membre à un cluster et le reste des membres est plus que le seuil de 0,4) et la création d'une équipe vide.

(2) L'initialisation des clusters se fait en attribuant arbitrairement un

membre au cluster. Après cela, nous assignons le premier membre à regroupés à cette équipe.
(3) Pour chaque nouveau membre (qui peut rejoindre le cluster), la distance qui le sépare de chaque membre dans les équipes est calculée.
La distance séparant le nouveau membre du crowdsourcing au cluster est la valeur maximale de la distance qui le sépare de chaque membre du cluster notée $\Omega(C, K)$.

(4) Chaque situation est différente lors du regroupement d'un nouveau membre du crowdsourcing, l'algorithme RSG se réserve les droits de création de nouvelles compositions d'équipes, de suppression de certains groupes et de fusion d'autres, nous notons que la solution proposée préserve la vie privée de chaque équipe.

En outre, dans le cas où un membre a une forte communication avec plus de deux clusters , différentes solutions existent comme la fusion des deux clusters , en déplaçant dans un seul cluster la totalité des membres du crowdsourcing reliés avec une distance de propagation supérieure au seuil fixé, une seule situation est choisie de façon aléatoire par l'algorithme RSG .

Un classement est effectué, pour ce faire, trois cas se présentent comme suit :

- le membre à regrouper, a une propagation en dessus du seuil, avec une seule équipe de la solution,donc, le membre est affecté à cette équipe.
- Le membre n'a pas de fuite de données avec aucune équipe dans la solution, donc, nous attribuons le membre arbitraire à une équipe ou à une équipe nouvellement créée.
- Le membre a une fuite de données avec deux ou plusieurs équipe dans la solution, pour cela, nous recherchons tous les membres connectés qui ont une propagation en dessus du seuil, ensuite, nous déplaçons tous les membres connectés soit dans une équipe arbitrairement choisie soit dans équipe nouvellement créé.

La dernière étape, est d'éliminer toutes les équipes vides du processus de classification. Par exemple
si $\Omega(C_1, K) = 0,8$ et $\Omega(C_2, K) = 0,7$, alors, l'algorithme RSG va choisir aléatoirement de déplacer les membres de C_1 (et les membres fortement connectés avec K, $\Omega \succ Seuil$) au cluster C_2 ou vice versa , une troisième situation est concevable est de choisir la création d'une nouvelle équipe C_3 et de déplacement de tous les membres connectés de C_1 et C_2 avec K dans C_3 , puis supprimer les équipes vides .

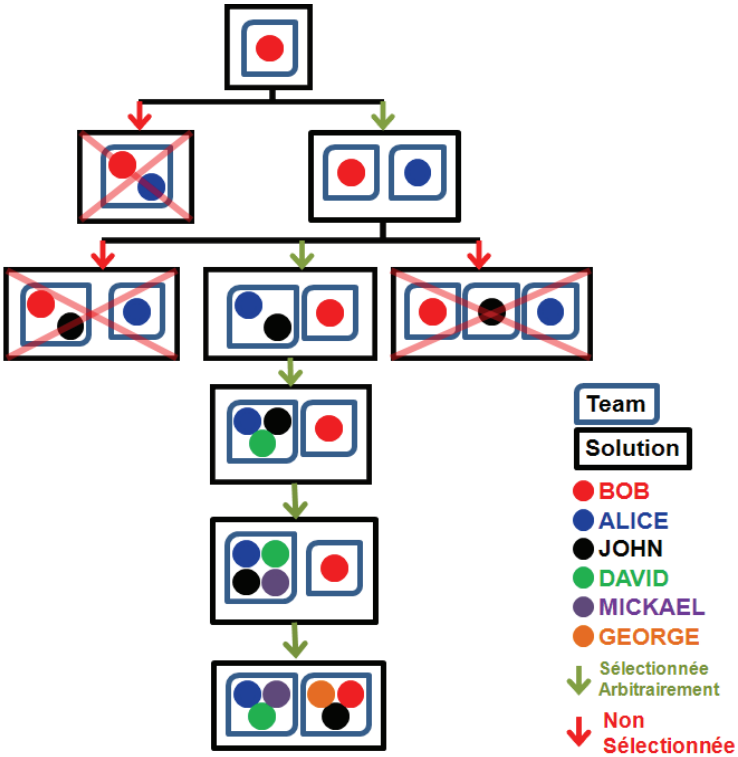

FIGURE 5.1 – RSG

L'algorithme retourne une constitution d'équipes possible tout en préservant la fuite de données.

La figure 5.1 présente une constitution aléatoire de solutions.

L'algorithme RSG retourne une solution aléatoire préservant la fuite de données entre les équipes, en général, cet algorithme en fonction de la structure du réseau social peut renvoyer une des nombreuses solutions possibles, tout en découvrant les équipes.

La fuite de données est calculée sur la base de la valeur de propagation maximale en dessus du seuil.

Notre algorithme de génération de solutions aléatoires est tracé comme suit :

Algorithm 10 Random solution generator algorithm (RSG)

m_i – Member of a social network

$HPTL$ – High Propagation Team List

$FHPM$ – Find High Propagation Member $\succ Threshold$

print $Teams = (team_1, team_2, ..., team_{Cluster})$: Teams constitution {I}nitializations

$Threshold$

$Team \leftarrow M_0$

$Solution \leftarrow Team$

for each Member m_i in G **do**

  Clustering(m_i)

end for

return $Solution$

Function $Clustering\,(Member\ mi)$

for each Team in Solution **do**

  for each member m_j in Team **do**

    $Distance_i \leftarrow \Omega_{Team_i, m_j}$

  end for

end for

HPTL ← High Propagation Team List

if size(HPTL == 1) **then**

  AddMember(HPTL[0],m_i)

end if

if size (HPTL $\succ$ 1) **then**

  ConnectedMembers ← FHPM(m_i)

  Team ← Random select team from HPTL

  AddMemberToTeam(Team,m_i, ConnectedMembers) % May move members

end if

if size (HPTL $\prec$ 1) **then**

Team $\leftarrow$ Random select team from solution % null if we create new team

AddMemberToTeam(Team,m_i)

end if

for each Team *in* Solution **do**

if Empty(Team) **then**

Delete(Team)

end if

end for

EndFunction

L'algorithme RSG permet de classifier tous les membres aléatoirement dans des équipes sans fuites de données. L'algorithme récupère tous les membres (n) et assigne chacun à une équipe sans fuite de données. L'algorithme est composé de trois règles principales, chaque règle à une complexité $O(n)$. La complexité théorique maximale est donc de $O(n)$.

5.2.4 Modèle probabiliste et méta heuristique de découverte d'équipe (PMTD)

Le PMTD est un algorithme basé sur le recuit simulé (RS), il appartient à la classe des algorithmes stochastiques de recherche appelés méta-heuristiques. Il s'agit d'un algorithme basé sur la recherche locale rapide conçu pour fournir de bonnes solutions optimales ou quasi-optimales dans un temps de calcul raisonnable (Kirkpatrick, 1983).

Recuit, désigne le processus qui se produit lorsque des substances physiques, tels que les métaux, sont portées à un haut niveau d'énergie (fondu), puis refroidi progressivement jusqu'à ce que un état solide sera atteint. Le but de ce processus est de parvenir à l'état de plus basse énergie. Dans ce processus, les substances physiques quittent généralement les états d'énergie supérieurs à ceux inférieurs si le processus de refroidissement est suffisamment lent, donc la minimisation se produit naturellement.

En raison de la variabilité naturelle, cependant, il existe une certaine probabilité à chaque étape du processus de refroidissement que la transition vers un état d'énergie supérieur se produit. Comme l'état d'énergie diminue naturellement, la probabilité de passer à un état d'énergie supérieur diminue. Une description détaillée de l'algorithme du recuit simulé et son utilisation pour l'optimisation peut être trouvé dans [1] .

Essentiellement, le recuit simulé dessine un point aléatoire initial pour commencer sa recherche. De ce point, l'algorithme fait un pas dans une plage prédéterminée par l'utilisateur. La valeur de la fonction objective de ce nouveau point est ensuite comparée à la valeur du point initial afin de déterminer si la nouvelle valeur est inférieure. Pour le cas de la réduction, si la valeur de la fonction objective diminue, il est automatiquement accepté et il devient le point à partir duquel la recherche se poursuivra. L'algorithme va alors procéder à une autre étape. Des valeurs plus élevées de la fonction objective peuvent également être acceptées avec une probabilité déterminée par les critères de Metropolis (voir [1]) .

En acceptant de temps en temps des points avec des valeurs plus élevées de la fonction objective, l'algorithme SA est en mesure d'échapper à des optima locaux. Comme l'algorithme progresse, la longueur des étapes diminue, en se fermant sur la solution finale.

L'algorithme de PMTD enregistre toutes les solutions acceptées afin de classer les résultats plus tard.

La fonction Generate_Neighbor_Solution(x) fournit de nouvelles solutions voisines à la dernière solution acceptée. La fonction Generate_Neighbor_Solution(x) est définie comme suit :

Algorithm 11 GENERATE_NEIGHBOR_SOLUTION

Function{Generate_Neighbor_Solution}{Solution Φ}

HardPreferencesValidities :=False % check the mandatory caller preferences

User $\Psi \longleftarrow$ Random selected user from Φ

Switch (Random selected user)

Case Swap :

User $\Theta \longleftarrow$ Random Selected user from Φ

Random Relocate Θ and Ψ

Case Single Relocation :

Random relocate user Ψ from a team to another

Case new team creation :

Create new team and assign user Ψ to it

EndFunction

Déplacement des utilisateurs du crowd :

t_i est un opérateur employé pour produire une solution λ voisine à la solution actuelle x en faisant un changement léger. Cet opérateur fonctionne afin d'éviter de produire des solutions infaisables. Nous prenons en considération trois mouvement d'opérateurs différents :

1) *Échange* : les positions de deux utilisateurs aléatoirement choisis sont échangées. Par exemple, considérons un problème avec n membres = 5 et quelques permutation $\{\{3,5\}\{2,4,1\}\}$. Supposons que les deux utilisateurs aléatoirement choisis sont des utilisateurs 5 et 4. Les positions correspondantes sont échangées ; donc, nous avons $\{\{3,4\}\{2,5,1\}\}$.

2) *Transfert seul* : un utilisateur aléatoirement choisi est aléatoirement transféré. Considérons la permutation précédente $\{\{3,5\}\{2,4,1\}\}$. Supposons que l'utilisateur choisi est l'utilisateur 4 et la nouvelle équipe aléatoirement choisie devient 1. Donc, la nouvelle solution devient $\{\{3,4,5\}\{2,1\}\}$.

3) *Nouvelle création d'équipe* : nous choisissons aléatoirement un utili-

sateur pour relocalisation. Nous créons une nouvelle équipe et y assignons l'utilisateur. Considérons l'exemple précédent, si l'utilisateur aléatoirement choisi est 3, la nouvelle solution sera $\{\{3\}\{5\}\{2,4,1\}\}$.

Critère d'acceptation

Comme mentionné précédemment, la variation est calculé $\Delta C = f(\lambda) - f(x)$. Si $\Delta C \leq 0$, la solution λ est acceptée. Sinon, la solution λ est acceptée avec une probabilité égale à $P_r = \exp\{-\Delta C \div t_i\}$.

Le schéma suivant illustre le processus :

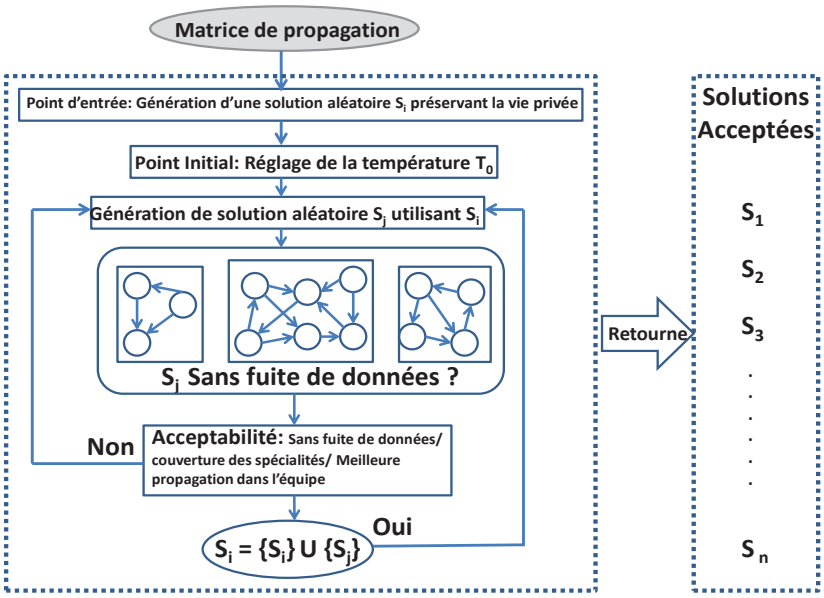

FIGURE 5.2 – Processus du PMTD

5.2.5 Expérimentations

Configuration des expériences :
Toutes nos expériences sont codées utilisant des technologies JAVA et s'exécutant sur un Mac OS X 10.5.8, Intel 2.13 GHz 2 Duo avec 2 Go de mémoire vive.

Pour ces expériences, notre réseau social est basé sur la loi de Metcalfe. La loi de Metcalfe caractérise beaucoup d'effets de réseau de technologies de

communication et des réseaux comme Internet, le réseau social et le World Wide Web.

La Loi de Metcalfe est liée au fait que le nombre de connexions uniques dans un réseau d'un certain nombre de noeuds (n) peut être exprimé mathématiquement comme le nombre triangulaire n (n - 1)/2, qui est proportionnel à n2 asymptotiquement.

La loi est abondante et existante en raison de la capacité des internautes à se lier ensemble. Les sites Web et des blogs comme Twitter, Facebook et Myspace sont le centre de cette entrée en vigueur légale.

Nous utilisons aléatoirement jusqu'à 5000 utilisateurs. De cet échantillon, nous avons construit un ensemble de formation de tous les utilisateurs pour préserver la divulgation de données pendant le processus de découverte d'équipe. L'approche PPCA proposée est utilisée sur l'échantillon pour découvrir les valeurs cachées de propagation de données.

Expérimentations sur la vie privée :

1) k-means et PMTD basés sur Ω :

Nous comparons la fuite de données entre des groupes produits utilisant l'algorithme de k-means avec la distance Ω et notre algorithme PMTD basé sur la même distance. Le seuil de fuite est mis à 40%. Avec la distance Ω dans l'algorithme PMTD, il n'y a aucune fuite de données entre des groupes produits même si la taille du réseau social augmente.

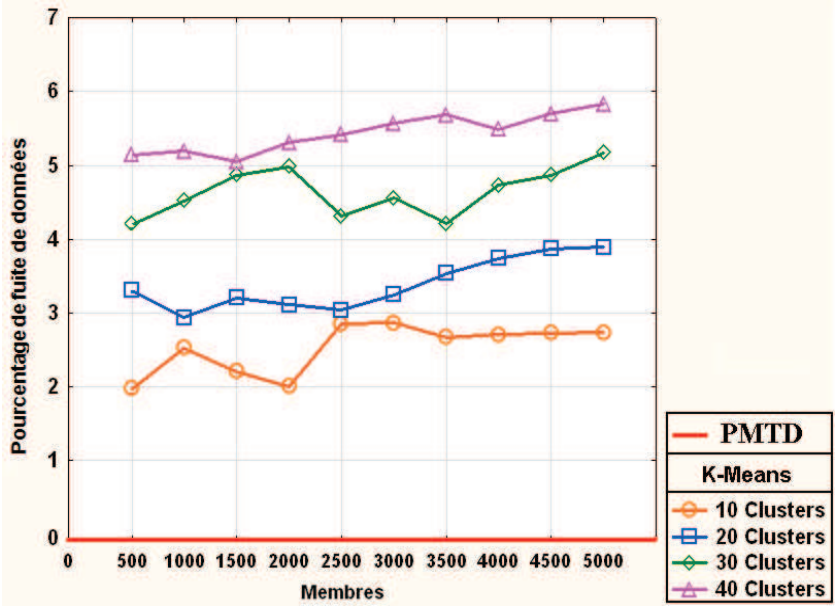

FIGURE 5.3 – Comparaison des fuite de données

2) Les solutions résultats en baissant la température de Recuit Simulée :

Dans cette expérience nous interceptons l'écart-type des propagations dans les équipes en baissant la température de l'algorithme de recuit simulé.

Au commencement (la température est égale à 9000) l'écart-type des propagations dans des équipes est autour de 56%. Nous obtenons un taux autour de 40% d'écart-type dans les équipes quand la température est autour de 5000 et nous observons une augmentation de ce taux pour atteindre 49% quand la température est de 0. (Voir figure 5.4-A).

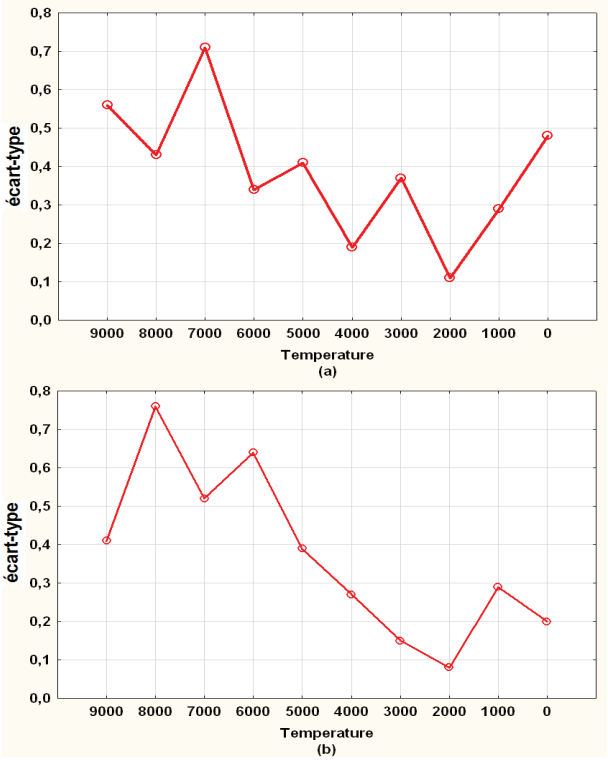

FIGURE 5.4 – Les résultats du PMTD

En même temps, nous interceptons l'écart-type des propagations en dehors des équipes en baissant la température de recuit simulé (Voir figure 5.4-B). D'abord, quand la température est autour de 9000, l'écart-type des propagations en dehors des équipes est autour de 41%. Ensuite, quand la température atteint la frontière de 8000, l'écart-type des propagations en dehors des équipes s'élève à 77% et quand la température est autour de 3000, l'écart-type des propagations en dehors des équipes atteint 15%.

Ces expérimentations confirment que les solutions générées sont aléatoires et permettent d'atteindre la meilleure solution rapidement en explorant probablement différentes solutions de groupement.

3) Analyse des solutions :

Pour cette expérience nous avons choisie les algorithmes génétiques et la méthode de recherche locale pour comparer l'efficacité de notre approche. On a décidé de fixer le temps de calcul au temps maximum de notre algorithme.

Le choix des algorithmes génétiques se repose sur le fait que c'est une

approche évolutionnaire, qui manipulent un ensemble de plusieurs solutions simultanément. Cette méta heuristique est la plus connue dans cette branche, inspiré du concept de sélection naturelle élaboré par Darwin.

Et le choix de la méthode de recherche locale (RL) est basé sur la simplicité de mise en oeuvre et sur son temps de calcul très faible. Le principe de la méthode de recherche locale (dite aussi basic local search) consiste à partir d'une solution s à choisir une solution s' dans un voisinage de s, telle que s' améliore la recherche.

Pour ces expériences, on a gardé la même fonction d'énergie pour les différents algorithmes pour pouvoir effectuer une meilleure comparaison.

Le critère de sélection de l'algorithme génétique (AG) est de choisir les équipes qui défavorisent le plus le travail en équipe (le taux de propagation de données dans la même équipe est faible). Ensuite, on effectue un croisement aléatoire ainsi, qu'une mutation stochastique.Finalement, l'évaluation se fait avec la fonction d'énergie.

Pour la méthode de recherche locale, nous avons décidé de choisir la première solution du voisinage pour des fins de rapidité de calcul. Cette approche ne fait que calculer $f(s+i)-f(s)$, où i correspond à un déplacement élémentaire.

La figure 5.5 nous montre que, au bout de 4500 secondes, le PMTD découvre une solution qui permet une propagation dans l'équipe de 0.7% et une propagation inter-équipes équivalente à 0.2%. Tandis que, l'algorithme génétique est arrivé à découvrir une solution avec 0.42% et 0.27% respectivement. La méthode de recherche locale à aussi réussi à découvrir une solution avec 0.33% et 0.24% respectivement.

Les résultats de la figure 5.5 montre qu'en fixant le temps de calcul, l'algorithme PMTD donne de meilleur résultats que l'algorithme génétique et la méthode de recherche locale. On observe que le PMTD obtient de bonnes solutions dans un court laps de temps mais il n'est pas en mesure de les améliorer en lui donnant plus de temps. Par contre, l'algorithme génétique obtient de bonnes solutions en lui donnant plus de temps. Il est capable de les améliorer constamment dans le temps.
L'efficacité des méthodes de recherche locale est très peu satisfaisante. D'abord, par définition, la recherche s'arrête au premier minimum local rencontré, c'est là leur principal défaut.

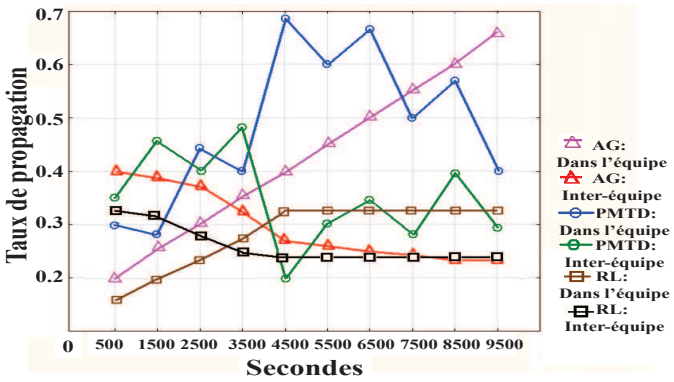

FIGURE 5.5 – Analyse des solutions

Cette expérience nous montre l'aspect aléatoire du PMTD qui permet d'exploiter un espace de solutions beaucoup plus large que celui de l'algorithme génétique ou de la méthode de recherche locale avec le même temps de calcul.

4) Expérimentations sur le temps d'exécution :

Nous avons analysé le temps d'exécution de PMTD, DLTD parallèle et l'utilisation de K-means avec la distance Ω de 500 à 5000 utilisateurs. Nous avons calculé le temps d'exécution de 12497500 relations à l'intérieur d'un réseau social ce qui signifie autour de 5000 membres, basés sur la loi de Metcalfe, nous observons que notre algorithme n'a aucune fuite de données et il a moins de temps de calcul que l'algorithme de k-means et l'algorithme DLTD parallèle.

Dans la figure 5.6, le temps d'exécution est entre 200 et 700 pour l'algorithme génétique, entre 200 et 1200 secondes pour l'algorithme DLTD parallèle, entre 251 et 1141 secondes pour des K-means et entre 167 et 403 secondes pour PMTD, alors, c'est meilleur pour l'algorithme PMTD utilisant SA que le K-means ou le DLTD parallèle.

On observe que l'algorithme génétique est plus long que l'algorithme PMTD parce qu'il améliore la solution trouvé tout au long du processus de découverte. Et on observe aussi que la méthode de recherche locale est la plus rapide mais on ne néglige pas que cette approche s'arrête au premier minimum local rencontré.

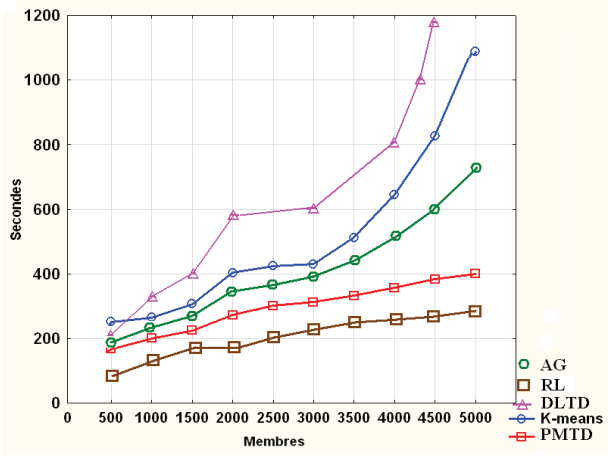

FIGURE 5.6 – Temps d'exécution

Les temps d'exécution de l'algorithme PMTD sont moins que l'algorithme K-means et moins que notre algorithme DLTD parallèle parce que le PMTD utilise le modèle de recuit simulé pour découvrir et fouiller dans des solutions voisines.

Les temps d'exécution de l'algorithme DLTD parallèle sont plus importants que ceux de K-means et du PMTD parce que le DLTD parallèle découvre toutes les solutions possibles et le k-means produit seulement une, en plus le PMTD utilise le modèle heuristique.

Nous notons que le critère de convergence de l'algorithme DLTD parallèle est atteint quand il n'existe aucun membre à classifier. Le critère de convergence de l'algorithme PMTD est atteint quand la température du SA est de 0.

La convergence de k-means est atteinte quand la comparaison des deux dernières itérations révèle que les membres ne déplacent pas d'équipes, ainsi, le calcul du k-means a atteint sa stabilité et aucune itération n'est nécessaire. En complément, la convergence à un minimum local peut produire de mauvais résultats. Les temps d'exécution importants de l'algorithme de k-means sont causés par le processus de convergence.

De plus, le résultat de k-means présente une fuite importante entre les différentes équipes en utilisant la distance Ω, en raison, du mécanisme de classification qui classifie les membres dans l'équipe la plus proche, c'est à dire, l'équipe avec qui le membre a la valeur de propagation la plus forte, même s'il existe d'autres propagation l'excédant le seuil de propagation avec

d'autres équipes.

La méthode de recherche locale est la plus rapide mais celle ci ne donne pas de solutions optimale.

5.3 Discussion

Comme discuté dans la figure 5.6 on a utilisé entre 500 et 5000 membres, le temps d'exécution de l'algorithme PMTD est nettement inférieur à celui de K-means ou du DLTD parallèle, du fait que, le PMTD est basé sur le recuit simulé pour découvrir et rechercher les solutions voisines.

Contrairement, les autres algorithmes sont itératifs. Le temps d'exécution de l'algorithme DLTD parallèle est plus important que le K-means et le PMTD, parce que, le DLTD permet de découvrir toutes les solutions possibles alors que le K-means produit uniquement une seule. Le PMTD enregistre un temps d'exécution moindre parce qu'il utilise les heuristiques.

On note aussi que, la recherche des solutions voisines peut être faite avec une méthode dirigée (en s'inspirant du modèle de l'algorithme génétique lors de la recherche d'une solution fille), par contre, cela va restreindre l'espace de recherche de solutions parce que le temps d'exécution reste toujours le même.
La méthode de recherche locale à permis de découvrir des solutions sans fuite de données entre les équipes en un temps record, par contre, on n'a aucune certitude que la solution est optimale.

Les algorithmes génétiques ont donné des solutions moins bonnes que celles retournées par le PMTD.

L'algorithme PMTD à réussi à découvrir des équipes compétitives et collaboratives en un temps réduit, tout en interdisant les fuites de données entres les équipes et en maximisant le travail au sein de chaque équipe pour plus de collaboration.

Notre approche permet aussi de découvrir plusieurs possibilités de classifications en fonctions des valeurs de propagations entre les différents membres du crowdsourcing.

Une approche pour classer ces différentes solutions est discutée dans le chapitre suivant pour ordonner les solutions en fonction des préférences du demandeur de la requête à savoir les compétences, les niveaux d'expertises, ...

5.4 Conclusion

Dans cette section nous avons présenté notre approche de classification des utilisateurs du crowdsourcing pour résoudre une requête d'un demandeur.

Cette approche est nettement meilleure que les approches citées précédemment. La problématique de temps d'exécution est levée en utilisant le recuit simulé comme heuristique pour l'exploration.

Au niveau de la vie privée, l'approche PMTD assure la découverte des meilleures solutions des équipes sans fuites de données. Sur le plan d'exploration de l'espace des solutions valables, l'algorithme PMTD à réussi à explorer aléatoirement différentes solutions dans tous l'espace des solutions possibles.

Chapitre 6

Approche de classement des différentes solutions de classification : Les SCSP

6.1 Introduction

Dans ce chapitre nous nous intéressons au domaine de programmation par contrainte. Suite au grand nombre de solutions retournées par l'algorithme DLTD et PMTD, dans cette section, nous démontrons la capacité du modèle semiring défini en « Soft Constraint Satisfaction Problem » dans le domaine de la programmation [Bistarelli 2004 ; Law 2004] pour classer les solutions de composition d'équipe proposées en utilisant les préférences des demandeurs du crowdsourcing.

Beaucoup de problèmes dans la vie réelle ne peuvent pas s'attendre à avoir une solution exacte. La plupart des problèmes qui doivent être résolus ont beaucoup de contraintes et donc la plupart du temps ne conduisent pas à une solution.

Dans de tels cas, nous devons assouplir les restrictions pour que nous puissions au moins trouver une solution qui soit aussi proche que possible de la solution attendue. C'est certainement beaucoup mieux que de ne pas

trouver une solution satisfaisant les différentes contraintes.

Ainsi, le concept de « Soft constraint » qui va être utilisé dans ce chapitre est que les contraintes sont classées en fonction de leur importances et la forme qui satisfait au maximum les contraintes de haut rang est considérée comme optimale. En ce qui suit, nous présentons en résumé l'approche de Soft constraint et montrons comment ce modèle peut être utilisé pour classer les solutions fournies.

6.2 Principes, propriétés et définitions

Un CSP définit un ensemble de variables dont les gammes sont prises à partir d'un domaine fini et un ensemble de contraintes qui limitent les valeurs que peuvent prendre ces variables. Une solution pour un CSP est une affectation d'une valeur à chaque variable, toutes les contraintes sont satisfaites simultanément.

Les « Soft constraints » [Bistarelli et al. 1997 ; 1995] généralise les CSP classiques en ajoutant un niveau de préférence pour chaque tuple dans le domaine des variables de contrainte.

Le niveau de préférence peut être utilisé pour obtenir une solution appropriée qui peut ne pas remplir toutes les contraintes, mais optimise certains paramètres (et qui dans notre cas seront naturellement appliquées aux besoins des utilisateurs).

Les opérations de base sur les « Sof constraints » (par exemple, la construction de conjonctions de contraintes et la projection sur les variables) doivent gérer les préférences d'une manière homogène.

Ceci exige un changement de la structure mathématique sous-jacente des CSP classiques pour passer d'une algèbre cylindrique à une algèbre « Semiring » , enrichi avec des propriétés supplémentaires, et appelée un C-semiring.

Définition 10 *un C-Semiring est un tuple* $\langle A, +, \times, 0, 1\rangle$ *- A est un ensemble et 0* $\in A$*, 1* $\in A$*. -* $\sum$ *est définie sur un sous-ensembles de la manière suivante :*

$*$) $+$ *est commutatif* $(a + b = b + a)$, *associatif* $(a + (b + c) = (a + b) + c)$, *avec un élément élémentaire 0* $(a + 0 = a)$ *et un élément absorbant 1* $(a + 1 = 1)$.

$*$) $\sum \emptyset = 0$ *et pour tout* $a \in A$, $\sum \{a\} = a$.

$*$) *Compte tenu de tout ensemble d'indices* S. $\sum_{i \in S} (\bigcup A_i) = \sum \left(\left\{ \sum_{i \in S} A_i \right\} \right)$ *(aplatissement)*

A représente le niveau de préférence de la solution qui peut être démontré qu'il est un treillis avec un ordre partiel $a \leq b$ si $a + b = b$, minimum 0, et maximum 1. La résolution est fondée sur la gestion des différentes préférences à l'aide de l'opérateur $+$ et $\times$. Il n'y a pas de supposition de la signification ou la manipulation des préférences, $+$ et $\times$ sont des espaces réservés pour des définitions concrètes utilisées par de nombreux systèmes de contraintes comme les contraintes floues, les contraintes traditionnelles, etc.

Définition 11 *(Contrainte).*

Étant donnée un c-semiring $\langle A, +, \times, 0, 1 \rangle$, *un ensemble de variable V, et un ensemble de domaine D, un pour chaque variable dans V, une contrainte est la paire* $\langle def, con \rangle$ *où* $con \subseteq V$ *et* $def : D^{|con|} \longrightarrow A$.

Définition 12 *(Soft Constraint Satisfaction Problem SCSP).*

Un SCSP est une paire $\langle def, con \rangle$ *où* $con \subseteq V$ *et C est un ensemble de contraintes. C peut contenir des variables qui ne sont pas dans con, autrement dit, ils ne sont pas intéressants pour le résultat final. Dans ce cas, les contraintes de C doivent être projetées sur les variables de con.*

Définition 13 *(Combinaison des contraintes).*

Deux contraintes $c_1 \langle def_1, con_1 \rangle$ *et* $c_2 \langle def_2, con_2 \rangle$ *peuvent être combiné dans* $C_1 \otimes C2 = \langle def, con \rangle$ *en prenant toutes les variables dans la contrainte d'origine* $(con = con_1 \bigcup con_2)$ *et en attribuant à chaque n-uplet dans la nouvelle contrainte une valeur de préférence qui provient la combinaison des valeurs*

des contraintes d'origine : $def(t) = def_1(t \downarrow \begin{matrix} con \\ con1 \end{matrix}) x def_2(t \downarrow \begin{matrix} con \\ con2 \end{matrix})$ *avec* $t \downarrow \begin{matrix} X \\ Y \end{matrix}$ *désignant la projection du tuple t, qui est défini sur l'ensemble des variables X, sur l'ensemble des variables* $Y \subseteq X$.

Définition 14 *(Projection).*

Compte tenu une "'Soft constraint"' c $= \langle def, con \rangle$ *et un ensemble de variable* $I \subseteq V$*, la projection de c sur I, notée* $c \Downarrow_I$ *est la contrainte* $\langle def', con' \rangle$ *où* $con' = con \bigcap I$ *et* $def'(t') = \sum_{\left\{ t|t\downarrow \begin{matrix} con \\ con \cap I \end{matrix} =t' \right\}} def(t)$.

Définition 15 *(Solution).*

Une solution d'un SCSP $\langle C, con \rangle$ *est la contrainte* $(\otimes C) \Downarrow_{con}$ *c'est à dire, la combinaison (conjonction) de toutes les contraintes de C projetée sur toutes les variables con d'intérêt.*

Les définitions 11,12,13,14,15,16 liées aux "'Soft constraints"' illustre une certaine fondamentale.

Il est possible de combiner les contraintes au sein d'autres contraintes (avec $\otimes$, similaire à la conjonction), et de projeter des contraintes sur un tuple de variables $\downarrow \begin{matrix} X \\ Y \end{matrix}$.

La valeur de la préférence de chaque tuple dans la contrainte de conjonction est extraite par les quêtes de la valeur de la préférence du tuple dans chaque contrainte individuelle. En éliminant les « colonnes » des tuples et en maintenant uniquement les composants qu'on ne peut pas enlever, nous utilisons les projections.

Bien que certaines "' lignes "" répétées pourraient apparaître, nous n'aurons qu'un seul reste, c'est la préférence calculée en appliquant + pour les préférences des tuple répétés. Sachant que la solution est simplement une projection sur une sélection de variables, les préférences de la solution sont calculées à l'aide de l'opération de projection.

Plus probablement, le tuple qui contient la valeur la plus élevée de préférence est élu solution "'optimale"'. Les définitions 17,18 et 19 montre le contexte du cadre SCSP.

Définition 16 *(Système de contrainte).*
un système de contrainte est un tuple $CS = \langle S, D, V \rangle$*, où S est c-semiring, D représente un ensemble de domaine, et V est l'ensemble ordonné de variables.*

Définition 17 *(Contrainte).*
Étant donnée un système de contrainte $CS = \langle S, D, V \rangle$*, et un problème* $P = \langle def, con \rangle$*, une contrainte est un tuple* $c = \langle def_c, type \rangle$ *où type représente le type de la contrainte et* def_c *est la fonction de définition de la contrainte.*

Définition 18 *(SCSP).*
Étant donnée un système de contrainte $CS = \langle S, D, V \rangle$*, un SCSP surchargé de CS est une paire* $P = \langle def, con \rangle$*, où con, appelé ensemble de variables d'intérêt pour C, un sous-ensemble de V et C est un ensemble fini de contraintes, qui peut contenir certaines contraintes définies sur les variables n'appartenant pas à con.*

6.3 Application des Soft Constraint Satisfaction Problem

Afin de répondre à une demande de crowdsourcing, les opérations de découverte d'équipes tout en préservant la vie privée peuvent donner beaucoup de solutions qui répondent aux conditions de la vie privée. Dans ce cas, nous intégrons les préférences du demandeur (Celui qui a lancé l'appel au crowdsourcing) pour choisir la solution la plus adaptée fondée sur le critère de la vie privée et les préférences. Soit CS = $\langle S, D, V \rangle$ un système de contrainte et P = $\langle def, con \rangle$ un problème à résoudre, où V = con = {TeamPropagationThreshold *(tpt)* : le seuil de propagation au sein d'une équipe, TeamSnPropagation *(tsp)* : Le seuil de propagation en dehors de

l'équipe, ExpertLevel *(el)* : niveau d'expertise, Competency *(com)* : compétence},
D = $\{\{tpt_1, tpt_2, tpt_3, tpt_4\}, \{tsp_1, tsp_2, tsp_3, tsp_4\}, \{el1, el2, el3, el4\}, \{com1, com2, com3\}\}$,
$S_p = \langle [0,1], max, min, 0, 1\rangle$, $C = \langle c_1, c_2, c_3, c_4\}$. Pour plus de simplicité, les variables et leurs domaines ont été écrits dans le même ordre. Les valeurs ci-dessus représente des domaines variables provenant d'une discrétisation tel que TPT $\in [0,1]$, TSP $\in [0,1]$, EL $\in [0,1]$, COMPETENCE $\in (cardiologue, pneumologue, carcinologue, Endocrinologue)$. Considérons les contraintes suivantes :
$c_1 = \langle def_{c1}, \{tpt, tsp\}\rangle$,
$c_2 = \langle def_{c2}, \{tsp, el\}\rangle$,
$c_3 = \langle def_{c3}, \{tpt, el\}\rangle$,
$c_4 = \langle def_{c4}, \{com\}\rangle$,
où les valeurs des préférences sont dans le tableau 1 et 2 .

t	def_{c1}	def_{c2}	def_{c3}
$\langle[0.00; 0.25[, [0.00; 0.25[\rangle$	0.25	0.00	0.00
$\langle[0.00; 0.25[, [0.25; 0.50[\rangle$	0.50	0.00	0.00
$\langle[0.00; 0.25[, [0.50; 0.75[\rangle$	0.75	0.00	0.75
$\langle[0.00; 0.25[, [0.75; 1.00]\rangle$	1.00	0.75	0.00
$\langle[0.25; 0.50[, [0.00; 0.25[\rangle$	0.50	0.00	0.00
$\langle[0.25; 0.50[, [0.25; 0.50[\rangle$	0.50	0.00	0.50
$\langle[0.25; 0.50[, [0.50; 0.75[\rangle$	0.75	0.25	0.00
$\langle[0.25; 0.50[, [0.75; 1.00]\rangle$	0.00	0.50	0.00
$\langle[0.50; 0.75[, [0.00; 0.25[\rangle$	0.75	0.00	0.75
$\langle[0.50; 0.75[, [0.25; 0.50[\rangle$	0.75	0.25	0.00
$\langle[0.50; 0.75[, [0.50; 0.75[\rangle$	0.00	0.75	0.00
$\langle[0.75; 1.00], [0.00; 0.25[\rangle$	0.00	0.50	0.00
$\langle[0.75; 1.00], [0.25; 0.50[\rangle$	1.00	0.75	0.00
$\langle[0.75; 1.00], [0.50; 0.75[\rangle$	0.00	0.50	0.00
$\langle[0.75; 1.00], [0.75; 1.00]\rangle$	0.00	0.50	0.00

TABLE 6.1 – Définition des contrainte "'tpt"', "'tsp"' et "'el"'

t	def_{c4}
$\langle Cardiologue\rangle$	1.00
$\langle pneumologue\rangle$	0.50
$\langle carcinologue\rangle$	0.75
$\langle Endocrinologue\rangle$	0.25

TABLE 6.2 – Définition des contraintes de compétence

Notez que celui qui a lancé l'appel peut assigner n'importe quelle valeur dans l'ensemble de la c-semiring à un tuple. Son choix de valeur représente

l'opportunité de ce tuple particulier. Considérons l'entrée def_{c1} ($\langle[0; 0.25[, [0.5; 0.75[= 0.69\rangle$). Le tuple $[0; 0.25[, [0.5; 0.75[$ est un tuple de la contrainte c1 qui représente le cas où TPT est entre 0 et 0.25 tandis que TSP est entre 0.5 et 0.75. La résolution est représentée comme suit :

- La première étape est de combiner les deux contraintes C_1 et C_2. $C_{1'} = C_1 \otimes C_2$
- La seconde étape est de combiner la contrainte $C_{1'}$ avec C_3. $C_{2'} = C_{1'} \otimes C_3$
- La troisième étape est de combiner la contrainte $C_{2'}$ avec C_4. $C_{3'} = C_{2'} \otimes C_4$

La première et la seconde étape de la résolution sont présentées dans le tableau 3. L'ensemble des solutions est classé par les préférences (voir le tableau 4). La solution de rang le plus élevé est choisi en premier.

t	$def_{c2'}$
$\langle[0.00; 0.25[, [0.50; 0.75[, [0.50; 0.75[, \rangle$	0.75
$\langle[0.00; 0.25[, [0.75; 1.00], [0.50; 0.75[, \rangle$	0.50
$\langle[0.00; 0.25[, [0.25; 0.50[, [0.50; 0.75[, \rangle$	0.25
$\langle[0.25; 0.50[, [0.25; 0.50[, [0.25; 0.50[, \rangle$	0.25
$\langle Alltherest \rangle$	0.00

TABLE 6.3 – Contrainte ordonnée avec 3 préférences

t	$def_{c3'}$
$\langle[0.00; 0.25[, [0.50; 0.75[, [0.50; 0.75[, Cardiologue\rangle$	0.75
$\langle[0.00; 0.25[, [0.50; 0.75[, [0.50; 0.75[, carcinologue\rangle$	0.75
$\langle[0.00; 0.25[, [0.75; 1.00], [0.50; 0.75[, Cardiologue\rangle$	0.50
$\langle[0.00; 0.25[, [0.50; 0.75[, [0.50; 0.75[, pneumologue\rangle$	0.50
$\langle[0.00; 0.25[, [0.75; 1.00], [0.50; 0.75[, pneumologue\rangle$	0.50
$\langle[0.00; 0.25[, [0.75; 1.00], [0.50; 0.75[, carcinologie\rangle$	0.50
$\langle[0.00; 0.25[, [0.25; 0.50[, [0.50; 0.75[, Cardiologue\rangle$	0.25
$\langle[0.25; 0.50[, [0.25; 0.50[, [0.25; 0.50[, Cardiologue\rangle$	0.25
$\langle[0.00; 0.25[, [0.25; 0.50[, [0.50; 0.75[, pneumologue\rangle$	0.25
$\langle[0.25; 0.50[, [0.25; 0.50[, [0.25; 0.50[, pneumologue\rangle$	0.25
$\langle[0.00; 0.25[, [0.25; 0.50[, [0.50; 0.75[, carcinologue\rangle$	0.25
$\langle[0.25; 0.50[, [0.25; 0.50[, [0.25; 0.50[, carcinologue\rangle$	0.25
$\langle[0.00; 0.25[, [0.50; 0.75[, [0.50; 0.75[, Endocrinologue\rangle$	0.25
$\langle[0.00; 0.25[, [0.75; 1.00], [0.50; 0.75[, Endocrinologue\rangle$	0.25
$\langle[0.00; 0.25[, [0.25; 0.50[, [0.50; 0.75[, Endocrinologue\rangle$	0.25
$\langle[0.25; 0.50[, [0.25; 0.50[, [0.25; 0.50[, Endocrinologue\rangle$	0.25
$\langle Alltherest\rangle$	0.00

TABLE 6.4 – Contrainte ordonnée avec 4 préférences

6.4 Expérimentations

Dans ces expériences, on à choisie quelques solutions retournées par l'algorithme PMTD, nous démontrons le fait de découvrir la meilleure solution en se basant sur les préférences du demandeur.
On a calculé le TeamPropagationThreshold (le seuil de propagation fixé entre les membres du crowd au sein de la même équipe), TeamSnPropagation (le seuil de propagation fixé entre les membres du crowd dans une équipe et le reste des membres dans les autres équipes) , ExpertLevel et Competency

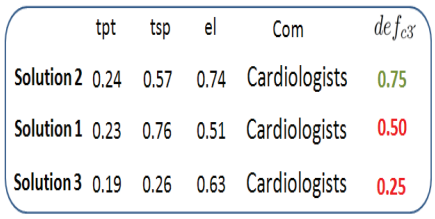

FIGURE 6.1 – SCSP Ranking

l'ensemble des préférences définies par le demandeur.

par exemple, pour TeamPropagationThreshold pour l'équipe composant la solution numéro 2 est de 0.24 et respectivement 0.23 pour la solution numéro 1, le TeamSnPropagation est de 0.57 et de 0.76 respectivement, le expertLevel est de 0.74 et de 0.51 respectivement et finalement le competency est Cardiologiste pour la solution 2 et 1.

En se basant sur les préférences du demandeur, on montre dans la figure 6.1 un exemple de classement de SCSP est on conclu que la solution numéro 2 est la meilleure parce qu'elle satisfait la majorité des préférences du demandeur.

On note qu'on n'a pas enregistré de fuite de donnée dans la meilleure solution et toutes les propagations en dehors des équipes sont inférieures au seuil.
Ces expériences confirme d'avantage que notre modèle est capable de choisir la meilleure solution lors du processus de découverte de solution de groupement en utilisant l'approche SCSP de classement basée sur les préférences du demandeur.

6.5 Discussion

Dans cette section nous avons montré la méthode de classement des solutions retournées par l'algorithme PMTD afin de les ordonner pour le demandeur de la requête.

Cette méthode se repose sur le modèle SCSP qui permet de combiner les préférences et d'ordonner les solutions en fonction des valeurs assignées par le demandeur.

L'avantage des SCSP c'est qu'on peut proposer plusieurs compositions d'équipes possible classées en fonction des préférences et c'est au demandeur de choisir la quelle des solutions à valider.

Cette méthode permet une souplesse de choix au demandeur, sachant que, ce n'est pas toujours évident de trouver des solutions aux besoins demandés.

6.6 Conclusion

Dans cette section nous avons détaillé la méthode SCSP en donnant toutes les définitions utiles au bon déroulement du classement.

Nous avons donné un exemple d'application pour les SCSP sur notre problème de découverte d'équipes compétitives et collaboratives.

Nous avons montré la faisabilité de classer les solutions générées par l'algorithme PMTD afin de présenter la meilleure au demandeur.

Chapitre 7

Conclusion et Perspective future

7.1 Conclusion

Le problème traité dans cette thèse est la constitution d'équipes compétitives et collaboratives tout en préservant la vie privée dans un contexte de crowdsourcing. Les données privées utilisées sont uniquement les données partagées avec l'entourage dans le réseau social.

Le challenge était de découvrir les relations cachées dans le réseau social, puis constituer des équipes sans fuites de données permettant de résoudre l'appel du demandeur.

Dans cette dissertation, nous avons proposé une approche pour la découverte des équipes compétitives et collaboratives dans le domaine du crowdsourcing. En particulier, nos principaux contributions à la recherche sont les suivantes :

- **Modèle de propagation de données :** Ce modèle permet de dévoiler les relations cachées dans le réseau social. En se basant sur les relations directes existantes, il permet de parcourir le réseau social pour mettre à jour les relations cachées (indirecte) et de prendre en compte les propagations maximales entre les utilisateurs.

  Afin de minimiser les fuites de données entre les différentes équipes, le

modèle fournit une matrice de propagation maximale entre les utilisateurs.

- **Modèle de classification glouton :** Ce modèle permet la classification des utilisateurs du crowdsourcing dans des équipes sans fuite de données. Pour ce faire, il doit regrouper les utilisateurs en fonction du risque de propagation de leurs données privées dans le réseau social et en se basant sur un seuil limite de propagation autorisée. Ce modèle permet de découvrir toutes les solutions de groupement possibles sans fuites de données des travailleurs dans le crowdsourcing.

- **Modèle de classification avec les heuristiques :** Ce modèle permet de chercher dans l'espace des différentes solutions possibles, les meilleures solutions de groupement des travailleurs dans le crowdsourcing.

  Un algorithme à base de méta heuristique à été proposé pour résoudre le problème NP-Complet identifié pour notre cas de recherche.

  Ce modèle permet de retourner les meilleures solutions de classification possibles sans fuites de données.

- **Modèle de classement de solutions :** Ce modèle permet de classer les solutions générées par le modèle de classification. Il se base sur les Soft Constraint Satisfaction Problem pour combiner les préférences du demandeur de la requête.

  Ce modèle permet de relaxer les préférences du demandeur et d'ordonner les solutions générées en fonction de leurs satisfaction des préférences. Ainsi, il permet au demandeur de choisir la solution la plus adéquate pour résoudre sa requête.

7.2 Perspective future

Le travail effectué dans cette thèse est susceptible d'être amélioré suivant plusieurs axes. Nous détaillons en particulier quelques travaux :

- **Gestion de l'influence sociale :** L'influence sociale est un terme général qui se rapporte à de nombreux phénomènes différents de psy-

chologie sociale. Les principaux types d'influence sociale qui font l'objet de recherches dans le domaine de la psychologie sociale sont les suivants :

1) La conformité : est l'acte de répondre favorablement à une demande explicite ou implicite. Techniquement, la conformité est un changement de comportement, mais ce n'est pas nécessairement de l'attitude, on peut se conformer en raison de la simple obéissance, ou par ailleurs opter pour retenir ses pensées privées en raison de pressions sociales. La satisfaction tirée de la conformité est en raison de l'impact social de l'influence acceptée (c'est à dire les individus se conforment pour avoir une récompense).

2) L'identification : est le changement d'attitudes ou de comportements en raison de l'influence d'une personne aimée. Actuellement les publicités s'appuient sur des célébrités pour commercialiser leurs produits en profitant de ce phénomène.

3) L'internalisation : est le processus d'acceptation d'un ensemble de normes établies par des personnes ou des groupes qui ont de l'influence sur l'individu. La personne accepte l'influence et le contenu de l'influence. Il est en harmonie avec le système des valeurs de l'individu.

4) La conformité : est un type d'influence sociale impliquant un changement de comportement, de croyance ou de penser à aligner avec ceux des autres ou de s'aligner sur des standards normatifs. C'est la forme la plus courante et omniprésente de l'influence sociale. Dans le cas de la pression, une personne est convaincue de faire quelque chose dont il pourrait ne pas vouloir faire, mais qu'il perçoit une récompense pour maintenir une relation positive avec d'autres personnes, comme leurs amis. La conformité par la pression résulte généralement de l'identification ou du respect de certains membres pour apaiser d'autres.

5) Influence par la minorité : est le fait que la majorité est influencée à accepter les croyances ou les comportements d'une minorité. Influence par la minorité peut être affectée par la taille des groupes majoritaires et minoritaires, le niveau de la cohérence du groupe minoritaire et des facteurs conjoncturels (comme la richesse de l'importance sociale de la minorité). L'influence minoritaire fonctionne le plus souvent grâce à l'influence sociale d'information parce que la majorité peut être indifférente au goût de la minorité.

6) Une prophétie auto-réalisatrice : est la prédiction qui cause directement ou indirectement la personne elle-même pour devenir vrai, en raison d'une rétroaction positive entre la croyance et le comportement. Une prophétie déclarée comme une vérité peut influencer suffisamment les différents individus, que ce soit par peur ou par confusion logique, de telle sorte que leurs réactions soient finalement remplient la fois par la fausse prophétie.

7) Réactance : est l'adoption d'un point de vue contraire à l'idée perçut par la pression. Ce comportement a également été appelé contre la conformité. Bien que les résultats soient à l'opposé de ce que voulait l'influenceur, ce comportement réactif est le résultat de la pression sociale.

Il nécessaire d'estimer la probabilité pour chaque type d'influence afin de pouvoir anticiper les réactions des travailleurs dans le Crowdsourcing. Il est aussi nécessaire de trouver une modélisation de l'influence des utilisateurs dans réseaux sociaux pour pouvoir effectuer une estimation de la probabilité de propagation et de son évolution dans le temps.

– **Le crowdsourcing en mobilité :** Le crowdsourcing mobile est un terme qui décrit les activités de crowdsourcing qui sont traitées sur les Smartphones et autres appareils mobiles. Les Smartphones sont de plus en plus performants et les utilisateurs de téléphones mobiles peuvent travailler sur des tâches de crowdsourcing sans difficultés. Aujourd'hui, les tâches sont beaucoup plus qu'une description de sites simples. Le Crowdsourcing mobile peut être utilisé pour collecter des données. Les utilisateurs des Smartphones équipés de GPS peuvent être localisés via des applications pour créer des profils de mouvement. Les utilisateurs de Smartphones peuvent également télécharger des données ainsi que des photos de restaurants, des adresses d'entreprises ou des informations sur des menus. Le crowdsourcing mobile peut également fournir une aide aux victimes de catastrophes par la coordination des programmes de secours en temps réel ou encore de documenter les dommages.

  On peut étendre notre approche en intégrant les informations des travailleurs du Crowdsourcing en prenant en compte par exemple la localisation, la situation de la région du travailleur, la situation politique du pays où habite le travailleur ...

– **La découverte des meilleurs profils par rapport à la requête :** La représentation du profil d'un utilisateur dans le Crowdsourcing dépend toujours de plusieurs critères. Du coup le choix d'un meilleur profil n'est pas toujours facile à définir par rapport aux différents contextes de la donnée. La collaboration de plusieurs utilisateurs du Crowdsourcing n'est pas toujours simple à mettre en place. Un axe de développement de notre approche repose sur la façon de rapprocher les différents profils d'un réseau social pour une meilleure collaboration.

  Le choix de la meilleure équipe pour réponde à l'appel du demandeur dépend de plusieurs facteurs, dans cette thèse on a utilisé les préférences du demandeur pour mieux orienter le classement des différentes solutions de groupement d'équipe. D'autres paramètres sont à prendre en compte tel que la localisation géographique, l'âge, le niveau intellectuel,...

– **La gestion des comportements des utilisateurs :** Le comportement est un phénomène qui change en fonction de plusieurs critères intérieur et extérieur à l'individu. Il désigne les actions d'un être vivant. Dans cette thèse, nous avons pris en compte le comportement sur le réseau social mais non pas son changement. Notre approche est susceptible d'évoluer en étudiant le changement de comportement du travailleur du Crowdsourcing et en prenant en compte différents changement de comportement dans la propagation des données privées entre les différentes équipes compétitives.

– **L'éducation des utilisateurs :** Dans la plupart des sites de réseaux sociaux, les utilisateurs doivent accepter les termes d'utilisation politique avant qu'ils puissent utiliser leurs services. Controversée, ces conditions d'utilisation déclarent que les utilisateurs doivent accepter souvent des clauses permettant aux opérateurs de réseaux sociaux de stocker des données sur les utilisateurs, ou même les partager avec des tiers.

  1) Le vol d'identité : En raison du volume élevé des renseignements personnels souvent affiché sur les réseaux sociaux, il est possible de faire d'autres estimations sur un utilisateur, telles que le numéro de sécurité sociale de la personne, qui peut ensuite être utilisé dans le cadre d'un vol d'identité. En 2009, les chercheurs de la Carnegie Mellon University ont publié une étude montrant qu'il est possible de trouver le numéro

de sécurité sociale d'un individu à l'aide des informations publiques sur les réseaux sociaux et sur les bases de données en ligne.

2) Traquer : Sur les sites de réseaux sociaux, il est devenu facile de construire un réseau d'amis et de connaissances, et partager avec eux des photos, des localisations, des coordonnées, et des intérêts sans jamais avoir la chance de les rencontrer effectivement. En utilisant les informations que les utilisateurs postent sur eux-mêmes en ligne, il est facile pour ces utilisateurs de devenir une victime d'harcèlement sans même en être conscients du risque.

3) Réputation involontaire : La confidentialité est une préoccupation des utilisateurs des réseaux sociaux. Plusieurs incidents de partage de vidéos, d'informations et de données privées ont été soulevés. Ces incidents constituent une source de nuisance pour les utilisateurs.

4) Surveillance : Bien que le concept d'un réseau social semble suivre le modèle de l'espace ouvert, les forces du marché contrôlent l'accès à une telle ressource. Récemment, une enquête par le Wall Street Journal a révélé que de nombreuses applications populaires sur Facebook transmettaient des informations d'identification sur les utilisateurs et leurs amis pour les annonceurs et les sociétés de suivi de l'internet, cette démarche constitue une violation de la politique de confidentialité des données privées de Facebook.

Chaque jour, les problèmes de la vie privée se multiplient de plus en plus. Notre approche peut s'étendre en intégrant un traitement d'éducation des utilisateurs pour éviter les nuisances de vol d'identité, de surveillance ou de la réputation involontaire. Ce traitement doit analyser en premier lieu la psychologie de l'utilisateur du crowdsourcing et puis proposer une éducation lui permettant de bien partager ces données privées dans le réseau social.

Table des figures

Liste des tableaux

Bibliographie

[1] Emile Aarts and Jan Korst. *Simulated Annealing and Boltzmann Machines : A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley & Sons, Inc., New York, NY, USA, 1989.

[2] A. Acquisti. Privacy in electronic commerce and the economics of immediate gratification. 2004.

[3] Lada Adamic, Orkut Buyukkokten, and Eytan Adar. A social network caught in the web. *First Monday*, 8(6), 2003.

[4] Eytan Adar. Guess : A language and interface for graph exploration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 791–800, New York, NY, USA, 2006. ACM.

[5] Yong-Yeol Ahn, Seungyeop Han, Haewoon Kwak, Sue Moon, and Hawoong Jeong. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 835–844, New York, NY, USA, 2007. ACM.

[6] R. Albert, H. Jeong, and A.L. Barabasi. The Diameter of the World Wide Web. *Nature*, 401 :130–131, 1999.

[7] L.A.N. Amaral, A. Scala, M. Barthélémy, and H.E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21) :11149, 2000.

[8] Yael Amsterdamer, Yael Grossman, Tova Milo, and Pierre Senellart. Crowd mining. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 241–252, New York, NY, USA, 2013. ACM.

[9] Yael Amsterdamer, Yael Grossman, Tova Milo, and Pierre Senellart. Crowdminer : Mining association rules from the crowd. *PVLDB*, 6(12) :1250–1253, 2013.

[10] Squicciarini Anna C, Mohamed Shehab, and Federica Paci. Collective privacy page 525/526/527 management in social networks security and privacy. 2009.

[11] Ankur Bansal, Tingting Chen, and Sheng Zhong. Privacy preserving back-propagation neural network learning over arbitrarily partitioned data. *Neural Computing and Applications*, 20(1) :143–150, 2011.

[12] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286 :509–512, 1999.

[13] Alain Barrat, Ciro Cattuto, Vittoria Colizza, Lorenzo Isella, Caterina Rizzo, Alberto E. Tozzi, and Wouter Van den Broeck. Wearable sensor networks for measuring face-to-face contact patterns in healthcare settings. In Martin Szomszor and Patty Kostkova, editors, *Proceedings of the 3rd International ICST Conference on Electronic Healthcare for the 21st century (eHealth 2010)*, 2011.

[14] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi : An open source software for exploring and manipulating networks, 2009.

[15] Jeffrey Baumes, Mark K. Goldberg, Mukkai S. Krishnamoorthy, Malik Magdon-Ismail, and Nathan Preston. Finding communities by clustering a graph into overlapping subgraphs. In Nuno GuimarÃ£es and Pedro T. IsaÃas, editors, *IADIS AC*, pages 97–104. IADIS, 2005.

[16] Jeffrey Baumes, Mark K. Goldberg, and Malik Magdon-Ismail. Efficient identification of overlapping communities. In Paul B. Kantor, Gheorghe Muresan, Fred S. Roberts, Daniel Dajun Zeng, Fei-Yue Wang, Hsinchun Chen, and Ralph C. Merkle, editors, *ISI*, volume 3495 of *Lecture Notes in Computer Science*, pages 27–36. Springer, 2005.

[17] Salima Benbernou, Hassina Meziane, and Mohand-Said Hacid. Runtime monitoring for privacy-agreement compliance. In *ICSOC*, pages 353–364, 2007.

[18] FUNG S. K. L. LEE J. H. M. BISTARELLI, S. and H. F. LEUNG. A local search framework for semiring-based constraint satisfaction problems. In *the 5th International Workshop on Soft Constraints*, 2003.

[19] B. Bollobas. *Random Graphs*. Cambridge University Press, 2001.

[20] Allan Borodin, Yuval Filmus, and Joel Oren. Threshold models for competitive influence in social networks. In Amin Saberi, editor, *WINE*, volume 6484 of *Lecture Notes in Computer Science*, pages 539–550. Springer, 2010.

[21] F. Brauer and C. Castillo-Chavez. *Mathematical models in population biology and epidemiology*, volume 40. Springer Verlag, New York, 2001.

[22] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. In *Proceedings of the 9th International World Wide Web Conference on Computer Networks : The International Journal of Computer and Telecommunications Netowrking*, pages 309–320, Amsterdam, The Netherlands, The Netherlands, 2000. North-Holland Publishing Co.

[23] C. Kruegel C. Karlberger, G. Bayler and E. Kirda. Exploiting redundancy in natural language to penetrate bayesian spam filters. 2007.

[24] Guido Caldarelli. *Scale-free networks : complex webs in nature and technology.* Oxford University Press, Oxford, 2007. Informations sur l'Ã©diteur http ://www.loc.gov/catdir/enhancements/fy0737/2007408536-d.html.

[25] Barbara Carminati, Elena Ferrari, and Andrea Perego. Rule-based access control for social networks. In *OTM Workshops (2)*, pages 1734–1744, 2006.

[26] Barbara Carminati, Elena Ferrari, and Andrea Perego. Enforcing access control in web-based social networks. *ACM Trans. Inf. Syst. Secur.*, 13(1), 2009.

[27] Hyunwoo Chun, Haewoon Kwak, Young-Ho Eom, Yong-Yeol Ahn, Sue Moon, and Hawoong Jeong. Comparison of online social relations in volume vs interaction : A case study of cyworld. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 57–70, New York, NY, USA, 2008. ACM.

[28] F. R. K. Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

[29] Aaron Clauset. Finding local community structure in networks. *Phys. Rev. E*, 72 :026132, August 2005.

[30] Aaron Clauset, M. E. J. Newman, , and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, pages 1– 6, 2004.

[31] V. Colizza, A. Barrat, M. BarthÃ©lemy, and A. Vespignani. The role of the airline transportation network in the prediction and predictability of global epidemics. *Proc. Natl. Acad. Sci. USA*, 103 :2015, 2006.

[32] compete.com. http ://blog.compete.com/2009/02/09/facebook-myspace-twitter-social-network/. janvier 2010.

[33] Cyworld. Cyworld, June 2005.

[34] Quoc Viet Dang, Sandrine Mouysset, and GÃ©raldine Morin. DÃ©tection de SimilaritÃ©s de Surfaces ParamÃ©triques. *Revue Electronique Francophone d'Informatique Graphique*, 6(2) :50–58, 2012.

[35] Susan B. Davidson, Sanjeev Khanna, Tova Milo, and Sudeepa Roy. Using the crowd for top-k and group-by queries. In *ICDT*, pages 225–236, 2013.

[36] Wouter de Nooy, Andrej Mrvar, and Vladimir Batagelj. *Exploratory Social Network Analysis with Pajek (Structural Analysis in the Social Sciences)*. Cambridge University Press, January 2005.

[37] Ithiel de Sola Pool and Manfred Kochen. Contacts and influence. *Social Networks*, 1 :5–51, 1978.

[38] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd : leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *WWW*, pages 469–478, 2012.

[39] Gianluca Demartini, Beth Trushkowsky, Tim Kraska, and Michael J. Franklin. Crowdq : Crowdsourced query understanding. In *CIDR*, 2013.

[40] Reinhard Diestel. *Graph Theory (Graduate Texts in Mathematics)*. Springer, August 2005.

[41] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. Pick-a-crowd : tell me what you like, and i'll tell you what to do. In *WWW*, pages 367–374, 2013.

[42] Peter S. Dodds and Duncan J. Watts. Universal behavior in a generalized model of contagion. *Physical Review Letters*, 2004.

[43] Christoph Dorn, Florian Skopik, Daniel Schall, and Schahram Dustdar. Interaction mining and skill-dependent recommendations for multi-objective team composition. *Data Knowl. Eng.*, 70(10) :866–891, 2011.

[44] Nan Du, Bin Wu, Xin Pei, Bai Wang, and Liutong Xu. Community detection in large-scale social networks. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, WebKDD/SNA-KDD '07, pages 16–25, New York, NY, USA, 2007. ACM.

[45] Richard C. Dubes and Anil K. Jain. Clustering techniques : The user's dilemma. *Pattern Recognition*, 8(4) :247–260, 1976.

[46] R.I.M. Dunbar. Neocortex size as a constraint on group size in primates. *Journal of Human Evolution*, 22(6) :469 – 493, 1992.

[47] R.I.M. Dunbar. The social brain hypothesis. *brain*, 9 :10, 1998.

[48] P. Erdös and A. Rényi. On random graphs i. *Publicationes Mathematicae Debrecen*, 6 :290, 1959.

[49] Paul Erdos and Alfred Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hungary. Acad. Sci.*, 5 :17–61, 1960.

[50] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '99, pages 251–262, New York, NY, USA, 1999. ACM.

[51] Lujun Fang, Heedo Kim, Kristen LeFevre, and Aaron Tami. A privacy recommendation wizard for users of social networking sites. In *ACM Conference on Computer and Communications Security*, pages 630–632, 2010.

[52] Flickr. Techcrunch, 2 billion photos on flickr. 2012.

[53] Michael J. Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. Crowddb : answering queries with crowdsourcing. In *SIGMOD Conference*, pages 61–72, 2011.

[54] L.C. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40 :35–41, 1977.

[55] M. Girolami. Mercer kernel-based clustering in feature space. *Trans. Neur. Netw.*, 13(3) :780–784, May 2002.

[56] Michelle Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Science*, 99(12) :7821–7826, 2002.

[57] J. Goldenberg, B. Libai, and E. Muller. Talk of the Network : A Complex Systems Look at the Underlying Process of Word-of-Mouth. *Marketing Letters*, pages 211–223, August 2001.

[58] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1019–1028, New York, NY, USA, 2010. ACM.

[59] M. Granovetter. Threshold models of collective behavior. *The American Journal of Sociology*, 83(6) :1420–1443, 1978.

[60] Mark Granovetter. The strength of weak ties. *The American Journal of Sociology*, 78(6) :1360–1380, May 1973.

[61] R. Gross. Re-identifying facial images. 2005.

[62] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. in workshop on privacy in the electronic society, and modeling and preventing phishing attacks.http ://www.informatics.indiana.edu/markus/ papers/phishing_jakobsson.pdf. 2005.

[63] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM*, pages 169–178, 2009.

[64] G. Hobgen. Security issues and recommendations for online social networks. 2007.

[65] Patel-Schneider P.F. Boley H. Tabet S. Grosof B. Dean M. Horrocks, I. Swrl : A semantic web rule language combining owl and ruleml. 2004.

[66] http ://www.microsoft.com/protect/yourself/ phishing/spear.mspx. Spear phishing : Highly targeted phishing scams. 2014.

[67] http ://www.ruleml.org. Rei : The rule markup initiative. 2006.

[68] Julie Hui, Michael D. Greenberg, and Elizabeth Gerber. Understanding the role of community in crowdfunding work. In *CSCW*, pages 62–74, 2014.

[69] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter : Understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, WebKDD/SNA-KDD '07, pages 56–65, New York, NY, USA, 2007. ACM.

[70] Stephen Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3) :241–254, 1967.

[71] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings : Good, bad and spectral. *J. ACM*, 51(3) :497–515, May 2004.

[72] B.W. Kernighan and S. Lin. An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell Systems Technical Journal*, 49(2), 1970.

[73] Aniket Kittur, Boris Smus, and Robert Kraut. Crowdforge : crowdsourcing complex work. In *CHI Extended Abstracts*, pages 1801–1806, 2011.

[74] Jon Kleinberg and Steve Lawrence. The structure of the web. *Science*, 294 :1849–1850, 2001.

[75] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5) :604–632, September 1999.

[76] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 611–617, New York, NY, USA, 2006. ACM.

[77] Jussi Lahtinen, Petri Myllymaki, Tomi Silander, and Henry Tirri. Empirical comparison of stochastic algorithms, 1996.

[78] Martin H. C. Law, Alexander P. Topchy, and Anil K. Jain. Clustering with soft and group constraints. In *SSPR/SPR*, pages 662–670, 2004.

[79] Louise Leenen, Anbulagan, Thomas Meyer, and Aditya K. Ghose. Modeling and solving semiring constraint satisfaction problems by transformation to weighted semiring max-sat. In *Australian Conference on Artificial Intelligence*, pages 202–212, 2007.

[80] Louise Leenen, Thomas Andreas Meyer, and Aditya Ghose. Relaxations of semiring constraint satisfaction problems. *Inf. Process. Lett.*, 103(5) :177–182, 2007.

[81] A. Lenhart and M. Madden. Teens. Teens, privacy & online social networks. 18 April 2007.

[82] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 462–470, New York, NY, USA, 2008. ACM.

[83] Jure Leskovec and Eric Horvitz. Planetary-scale views on a large instant-messaging network. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 915–924, New York, NY, USA, 2008. ACM.

[84] Xin Li, Bing Liu 0001, and Philip S. Yu. Discovering overlapping communities of named entities. In Johannes FÃ$\frac{1}{4}$rnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *PKDD*, volume 4213 of *Lecture Notes in Computer Science*, pages 593–600. Springer, 2006.

[85] David Liben-Nowell, Jasmine Novak, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences*, 102(33) :11623–11628, August 2005.

[86] Fredrik Liljeros, Christofer R. Edling, Luis A. Amaral, H. Eugene Stanley, and Yvonne Aberg. The web of human sexual contacts. *Nature*, 411(6840) :907–908, 2001.

[87] Dunia López-Pintado and Duncan J Watts. Social influence, binary decisions and collective dynamics. March 2006.

[88] Matthew M. Lucas and Nikita Borisov. Privacy in electronic commerce and the economics of immediate gratification. 2008.

[89] Feng Luo, James Z. Wang, and Eric Promislow. Exploring local community structures in large networks. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '06, pages 233–239, Washington, DC, USA, 2006. IEEE Computer Society.

[90] David B. MacKay and Joseph L. Zinnes. Replyï¿½considerations in the use of probabilistic multidimensional scaling models. *Marketing Science*, 5(4) :348–349, 1986.

[91] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA, 1967.

[92] Ahmed R. Mahmood, Walid G. Aref, Eduard C. Dragut, and Saleh Basalamah. The palm-tree index : Indexing with the crowd. In *DBCrowd*, pages 26–31, 2013.

[93] J. W. Mann and G. D. Smith. A comparison of heuristics for telecommunications traffic routing. In V. J. Rayward.smith, I. H. Osman, C. R. Reeves, and G. D. Smith, editors, *Modern Heuristic Search Methods*, pages 235–254. Wiley, 1996.

[94] Adam Marcus, Eugene Wu, David R. Karger, Samuel Madden, and Robert C. Miller. Human-powered sorts and joins. *PVLDB*, 5(1) :13–24, 2011.

[95] Adam Marcus, Eugene Wu, Samuel Madden, and Robert C. Miller. Crowdsourced databases : Query processing with people. In *CIDR*, pages 211–214, 2011.

[96] Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. 2001.

[97] Stanley Milgram. The small world problem. *Psychology Today*, 1(1) :61–67, 1967.

[98] Kaivan Munshi. Social learning in a heterogeneous population : Technology diffusion in the indian green revolution.

[99] Tamara Macushla Munzner. *Interactive Visualization of Large Graphs and Networks*. PhD thesis, Stanford, CA, USA, 2000. AAI9995264.

[100] Atif Nazir, Saqib Raza, and Chen-Nee Chuah. Unveiling facebook : A measurement study of social network based applications. In *Proceedings*

of the 8th ACM SIGCOMM Conference on Internet Measurement, IMC '08, pages 43–56, New York, NY, USA, 2008. ACM.

[101] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America*, 98(2) :404–409, January 2001.

[102] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45 :167–256, 2003.

[103] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review*, E 69(026113), 2004.

[104] Mark Newman, Albert-Laszlo Barabasi, and Duncan J. Watts. *The Structure and Dynamics of Networks : (Princeton Studies in Complexity)*. Princeton University Press, Princeton, NJ, USA, 2006.

[105] M.E.J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69, September 2003.

[106] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering : Analysis and an algorithm. 2001.

[107] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking : Bringing order to the web. Technical report ;, Stanford University, 1998.

[108] Gergely Palla, Imre DerÃ©nyi, IllÃ©s Farkas, and TamÃ¡s Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043) :814–818, June 2005.

[109] Christopher R. Palmer, Phillip B. Gibbons, and Christos Faloutsos. Anf : A fast and scalable tool for data mining in massive graphs. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 81–90, New York, NY, USA, 2002. ACM.

[110] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Phys. Rev. Lett.*, 86(14) :3200–3203, 2001.

[111] Alex Pothen, Horst D. Simon, and Kan-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.*, 11(3) :430–452, May 1990.

[112] Lawrence Hall Prodip Hore and Dmitry Goldgof. A clustering ensemble framework for large data sets. In *Systems, Man and Cybernetics, 2006. SMC '06. IEEE International Conference on*, 2006.

[113] Daniele Quercia, Giusy Di Lorenzo, Francesco Calabrese, and Carlo Ratti. Mobile phones and outdoor advertising : Measurable advertising. *IEEE Pervasive Computing*, 10(2) :28–36, 2011.

[114] Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 61–70, New York, NY, USA, 2002. ACM.

[115] D. Rosenblum. What anyone can know : The privacy risks of social networking sites. 2007.

[116] Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Preferences in constraint satisfaction and optimization. *AI Magazine*, 29(4) :58–68, 2008.

[117] Senjuti Basu Roy, Ioanna Lykourentzou, Saravanan Thirumuruganathan, Sihem Amer-Yahia, and Gautam Das. Optimization in knowledge-intensive crowdsourcing. *CoRR*, abs/1401.1302, 2014.

[118] Kevin Tang Saikat Guha and Paul Francis. Privacy in electronic commerce and the economics of immediate gratification. 2008.

[119] Salvatore Scellato. "beyond the social web : The geo-social revolution" by salvatore scellato with ching-man au yeung as coordinator. *SIGWEB Newsl.*, (Autumn) :5 :1–5 :5, September 2011.

[120] D. Schaffer and L. Eshelman. Combinatorial optimization by genetic algorithms : The value of the genotype/phenotype distinction. In *Proceedings of the Conference on Applied Decision Technologies (ADT'95). Volume 1 : Computational Learning and Probabilistic Reasoning*, pages 29–40, Uxbridge, UK, April 1995. Unicom Seminars.

[121] T. Schelling. Dynamic models of segregation. *Journal of Mathematical Sociology*, 1(1) :143–186, 1971a.

[122] Rossano Schifanella, André Panisson, Cristina Gena, and Giancarlo Ruffo. Mobhinter : Epidemic collaborative filtering and self-organization in mobile ad-hoc networks. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 27–34, New York, NY, USA, 2008. ACM.

[123] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8) :888–905, August 2000.

[124] Georgos Siganos, Sudhir Leslie Tauro, and Michalis Faloutsos. Jellyfish : A conceptual model for the as internet topology. *Journal of Communications and Networks*, 8(3) :339–350, 2006.

[125] Florian Skopik, Daniel Schall, Harald Psaier, Martin Treiber, and Schahram Dustdar. Towards social crowd environments using service-oriented architectures. *it - Information Technology*, 53(3) :108–116, 2011.

[126] Nilothpal Talukder, Mourad Ouzzani, Ahmed K. Elmagarmid, Hazem Elmeleegy, and Mohamed Yakout. Privometer : Privacy protection in social networks. In *ICDE Workshops*, pages 266–269, 2010.

[127] Helen Trottier and Pierre Philippe. Deterministic Modeling Of Infectious Diseases : Theory And Methods. *The Internet Journal of Infectious Diseases*, 1, 2001.

[128] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. The anatomy of the facebook social graph. *CoRR*, abs/1111.4503, 2011.

[129] Deepak Verma and Marina Meila. A comparison of spectral clustering algorithms. Technical report, 2003.

[130] Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks : [extended abstract]. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 1275–1276, New York, NY, USA, 2007. ACM.

[131] Jiannan Wang, Tim Kraska, Michael J. Franklin, and Jianhua Feng. Crowder : Crowdsourcing entity resolution. *Proc. VLDB Endow.*, 5(11) :1483–1494, July 2012.

[132] Stanley Wasserman and Katherine Faust. *Social network analysis : methods and applications*. Cambridge University Press, Cambridge ; New York, 1994.

[133] D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, (393) :440–442, 1998.

[134] Thomas Weise. *Global Optimization Algorithms - Theory and Application*. Self-Published, second edition, June 2009. Online available at http ://www.it-weise.de/.

[135] Hendler J. Berners-lee T. Connolly D. Weitzner, D.J. Creating a policy-aware web ;discretionay, rule-based avvess for the word wid web. 2006.

[136] Christo Wilson, Bryce Boe, Alessandra Sala, Krishna P.N. Puttaswamy, and Ben Y. Zhao. User interactions in social networks and their implications. In *Proceedings of the 4th ACM European Conference on Computer Systems*, EuroSys '09, pages 205–218, New York, NY, USA, 2009. ACM.

[137] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *WWW*, pages 981–990, 2010.

[138] Jennifer Xu and Hsinchun Chen. The topology of dark networks. *Commun. ACM*, 51(10) :58–65, October 2008.

[139] Tingxin Yan, Vikas Kumar, and Deepak Ganesan. Crowdsearch : Exploiting crowds for accurate real-time image search on mobile phones. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 77–90, New York, NY, USA, 2010. ACM.

[140] H. Peyton Young, Joshua Epstein, Edoardo Gallo, James Heckman, and Josef Hofbauer. Innovation diffusion in heterogeneous populations : Contagion, social influence, and social learning.

[141] YouTube. Youtube fact sheet. 2013.

[142] W.W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33 :452–473, 1977.