جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Université abderrahmane mira - bejaia
Faculté des Sciences Exactes
Département : Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de MASTER Académique en Informatique

Option : Système d'information avancés

Thème

Gestion de la confiance dans les reseaux sociaux

Présenté Par :

BENDELAL ABDERAZAK

FELLAH ABDERAHIM

Devant le jury :

Dr SADI MUSTAPHA	*Univérsité de bejaia*	*Président*
Mr BOUCHEBBAH FATAH	*Univérsité de bejaia*	*Examinateur*
Mr CHEKRID MOHAMED	*Univérsité de bejaia*	*Encadrant*

Date de soutenance : 04/07/2023

Remerciments

Nous tenons tout d'abord à exprimer notre gratitude envers Dieu pour nous avoir donné la force de mener à bien ce travail.

Nos sincères remerciements vont également à toutes les personnes qui nous ont apporté leur aide et ont contribué à la réalisation de ce mémoire, ainsi qu'à la réussite de cette année académique enrichissante.

Nous souhaitons remercier tout particulièrement Monsieur Chekrid Mohamed, pour sa générosité et la grande patience dont il a fait preuve malgré ses responsabilités professionnelles.

Nous exprimons également notre reconnaissance envers l'équipe pédagogique et administrative du master académique en Informatique, spécialité Systèmes d'Information Avancés, pour leur soutien et leurs conseils précieux.

Enfin, nous tenons à remercier chaleureusement tous nos proches et amis qui nous ont soutenus et encouragés tout au long de la réalisation de ce mémoire.

Dédicace

Je remercie Allah de m'avoir donné le courage pour accomplir ce modeste travail que je dédie :

À mes très chers parents,

Je souhaite exprimer ma profonde gratitude envers mes parents pour leur amour, leur soutien inconditionnel et leurs sacrifices constants. Votre encouragement et vos encouragements ont été une source d'inspiration pour moi.

A mes chers frères,

Idris , akram, aicha , fatiha

Je leur souhaite du succès dans leur vie personnelle et académique.

A Mes chers amis :

Je tiens également à remercier mes chers amis :rahim chinwi, Abderaouf, Nasri, Serhane, Youcef, Anouar,rabeh, benhamada,chemsou, hmida , khetat ,badis, amine bouira, taki et taki Aymen, Agheslane et Imad. Et ryma (marga rita) Votre présence, vos encouragements et vos moments de détente ont rendu cette aventure encore plus mémorable.

abderazak

Dédicace

Je remercie Allah de m'avoir donné le courage pour accomplir ce modeste travail que je dédie :

À mes très chers parents,

Je souhaite exprimer ma profonde gratitude envers mes parents pour leur amour, leur soutien inconditionnel et leurs sacrifices constants. Votre encouragement et vos encouragements ont été une source d'inspiration pour moi.

A mes chers frères,

Younes zinou nouha

Je leur souhaite du succès dans leur vie personnelle et académique.

La famille

Ma famille je vous aime et je vous souhaite que de bonheur et le paradis surtout djedi et ayou

A Mes chers amis :

Je tiens également à remercier mes chers amis :said, rass,nassim ,za9zo9,Youcef, serhane, Agheslane ,chemssou,Imad ,rabah Votre présence, vos encouragements et vos moments de détente ont rendu cette aventure encore plus mémorable.

abderahim

Tables des matières

Liste des figures

Liste des tableaux

Notations et symboles

RSL	Réseau Social en Ligne.
RS	Réseau Social.
RSBW	Réseaux Sociaux Basés sur le Web.
G = (V, E)	Graphe de confiance avec V sommets et E arcs.
n	Nombre de sommets du graphe de confiance G.
m	Nombre d'arcs arcs du graphe de confiance G.
vs	Sommet source du graphe de confiance G.
$\mathbf{v_t}$	Sommet cible du graphe de confiance G.
l	Niveau du sommet cible vt.
Seuil	Seuil de confiance.
EAM	Erreur Absolue Moyenne.
EQM	Erreur Quadratique Moyenne.
Prc	Précision.
Cvr	Couverture.
N	Nombre d'utilisateurs renvoyé par l'algorithme.
Nf	Nombre d'utilisateurs fiables renvoyé par l'algorithme.
$\mathbf{N_{nf}}$	Nombre d'utilisateurs non fiables renvoyé par l'algorithme.
$\mathbf{N_{tf}}$	Nombre total effectif d'utilisateurs fiables dans le système.

Introduction générale

Introduction générale

La communication humaine remonte aux premiers temps de l'humanité, lorsque les individus interagissaient les uns avec les autres au sein de diverses relations sociales telles que la famille, les voisins, le travail, l'amitié, etc. Ces interactions créent des groupes connus sous le nom de Réseaux Sociaux (RSs), qui sont formés en fonction des liens qui unissent les personnes [52].

Les avancées scientifiques dans le domaine des technologies de l'information et de la communication, notamment sur le Web, ont introduit de nouvelles technologies qui modifient profondément le mode de notre vie courante. Le Web 2.0 en est un exemple, marquant une évolution et une extension du Web 1.0, qui était principalement axé sur la relation entre "auteurs et lecteurs" [53]. Le Web 2.0 permet aux développeurs de concevoir des plateformes standardisées offrant aux utilisateurs des environnements dynamiques qui facilitent la communication et l'échange d'informations de manière rapide et accessible, favorisant ainsi la formation de communautés en ligne, telles que les réseaux sociaux en ligne (RSLs) [54]. Facebook, Twitter, MySpace, et d'autres plateformes populaires sont parmi les exemples les plus connus de réseaux sociaux en ligne. Les RSLs sont des versions numériques des réseaux sociaux qui permettent aux utilisateurs de former leurs propres réseaux en ligne en se basant sur leurs relations sociales [55].

Les réseaux sociaux en ligne (RSLs) sont constitués de millions d'utilisateurs qui génèrent et échangent d'importantes quantités d'informations via des interactions en ligne. La nature ouverte de ces réseaux et le partage de données personnelles posent des défis majeurs en matière de sécurité des données échangées [8].

De nombreux utilisateurs sont confrontés à la question de savoir à qui ils peuvent faire confiance pour partager leurs données privées. Par manque de connaissance, ils finissent souvent par les partager avec des individus malveillants, mettant ainsi en péril leur vie privée. Pour éviter de telles situations, il est crucial de trouver des mécanismes permettant aux utilisateurs d'identifier des personnes de confiance avec lesquels ils peuvent partager leurs données privées, ce qui constitue l'objectif de notre mémoire. La littérature propose plusieurs algorithmes de calcul de confiance, qui peuvent être fondamentalement classés en deux catégories : Global : exploitent tous le réseau, et Local : exploitent une partie du réseau. Parmi les algorithmes globaux les plus connus, on peut citer les deux algorithmes MoleTrust. [42] et TidalTrust[15]. L'idée fondamentale des algorithmes globaux est de propager la confiance à travers différents chemins de confiance, de la source à la cible, en utilisant des fonctions de

propagation et d'agrégation. En revanche, dans les algorithmes locaux, on exploite uniquement quelque les vois de la source et de la cible.

Les deux catégories d'algorithmes présentent des avantages et des inconvénients distincts. Les algorithmes de la première catégorie se caractérisent par leur simplicité et leur grande complexité qui est linéaire. Cependant, les fonctions de propagation et d'agrégation peuvent qu'ils utilisent présentent des lacunes dans certaines situations, ce qui peut affecter la qualité des résultats obtenus. En revanche, les algorithmes de la deuxième catégorie offrent souvent des résultats de bonne qualitéavec une complexité très faible qui est constante O(1). Pour évaluer notre approche, nous avons mis en implémenté notre algorithme local proposé *FastTrust*, ainsi que deux autres algorithmes *MoleTrust* et *TidalTrust* en utilisant le langage de programmation Python et l'environnement de développement intégré Spyder. Ensuite, nous avons comparé les performances des trois algorithmes selon quatre mesures d'évaluation : l'erreur absolue moyenne (EAM), l'erreur quadratique moyenne (EQM), la précision (Prc) et la couverture (Cvr) en utilisant un jeu de données de test réel appelé "Résidence hall". [38]. Les résultats obtenus montrent que l'algorithme proposé fournit des résultats nettement meilleurs que ceux de *MoleTrust* et *TidalTrust*.

Ce mémoire est structuré en cinq chapitres, dont voici une brève présentation :

Dans le premier chapitre, nous pressentons d'abord définitions et quelques concepts de base liés aux réseaux sociaux en ligne (*RSLs*), à savoir leurs architectures et leurs classifications. De plus, nous étudions les méthodes d'analyse et les problèmes de sécurité dans les *RSLs*.

Le deuxième chapitre est consacré à l'étude de la confiance dans les *RSLs* à savoir ses définitions, ses propriétés, sa présentation et son importance.

Dans le troisième chapitre, nous introduisons d'abord les concepts de base de la complexité algorithmique, puis nous présentons différents algorithmes de confiance existants dans la littérature, et enfin nous concluons le chapitre par une étude comparative de ces algorithmes selon plusieurs critères.

Dans le quatrième chapitre, nous présentons premièrement les inconvénients des méthodes glabales. Ensuite, nous décrivons en détail le fonctionnement de l'algorithme *FastTrust* en donnant son pseudo-code et sa complexité algorithmique.

Dans le cinquième chapitre, nous présentons d'abord le langage de programmation et l'environnement de développement utilisés, puis nous implémentons l'algorithme proposé *fastTrust* ainsi que les deux autres algorithmes (*MoleTrust*et *TidalTrust*). Ensuite, nous comparons les résultats des trois algorithmes en utilisant un jeu de données de test réel ("*Residence hall*") selon quatre mesures d'évaluation (EAM, EQM, Prc, Cvr).

Enfin, nous terminons le mémoire par une conclusion générale.

Problématique

La question qui se pose dans notre problématique est Comment calculer les valeurs de confiance entre les différents sommets d'un graph de confiance qui represente des relation direct entre des perssonnes de meme résau ?

Chapitre 01

1.1 Introduction

L'utilisation des réseaux sociaux en ligne a connu une croissance exponentielle au cours des dernières années, transformant la manière dont les individus communiquent, interagissent et partagent des informations. Ces plateformes ont évolué à partir des réseaux sociaux traditionnels pour inclure des fonctionnalités de partage de contenu, de messagerie, de création de profil et de découvertes de nouveaux utilisateurs, offrant ainsi une portée mondiale et une connectivité beaucoup plus grande que les réseaux sociaux traditionnels.

Dans ce chapitre, nous allons examiner de près les réseaux sociaux en ligne et leur architecture, en nous concentrant sur la manière dont les données sont stockées, gérées et présentées aux utilisateurs à travers différentes couches d'applications. Nous allons également explorer la taxonomie des RSLs, y compris les modèles de domaine, de données, de système et de réseau. Ensuite on passe aux concepts fondamentaux des méthodes d'analyse des Réseaux Sociaux en Ligne (RSLs). Nous abordons ensuite les problèmes de sécurité inhérents aux réseaux sociaux en ligne (RSLs), notamment la confidentialité des données.

1.2 Réseaux sociaux (SNs) et réseaux sociaux en ligne (OSNs)

Un réseau social est un ensemble d'acteurs et des relations qu'ils entretiennent entre eux [1]. Le concept de réseaux sociaux a été introduit pour la première fois par Barnes en 1954 [2], qui représente un réseau social par un graphe appelé "graphe social", où les sommets représentent les entités sociales et les arêtes représentent les relations entre ces entités. Les entités peuvent être des individus, des groupes, etc., tandis que les arêtes peuvent être des interactions verbales ou gestuelles, des transactions monétaires, des échanges de services, des transmissions d'informations, des invitations, des valeurs, etc.

L'expression "Réseau Social en Ligne" (RSL) désigne l'ensemble des applications informatiques liées à Internet qui permettent de relier des amis, des associés ou d'autres individus ayant des intérêts communs, tout en réduisant les contraintes de communication (temps et espace). L'émergence des réseaux sociaux basés sur le Web tels que Facebook, Myspace, Twitter, etc. a étendu la notion des réseaux sociaux à des échelles plus larges [3]. L'accès à ces réseaux (devenir membre) se fait par une simple inscription, généralement gratuite, où l'utilisateur est invité à fournir certaines informations le concernant (nom, âge, adresse e-mail, centres d'intérêts, etc.). Ces informations définissent sa page individuelle

"profil", qui peut inclure plusieurs détails tels que des photos et des vidéos, et à partir de laquelle il peut se connecter avec d'autres utilisateurs de la plate-forme, créant ainsi un réseau de connexions.

Au fil du temps, les réseaux sociaux en ligne ont évolué au-delà de la simple connexion entre les individus. Ces dernières années, les entreprises et les gouvernements ont commencé à reconnaître leur potentiel en tant que plateformes pour offrir et améliorer leurs services, exploitant ainsi les opportunités qu'ils offrent.

1.3 Architecture des RSLs

La Figure 1.1 présente l'architecture de référence d'un RSL, qui est constituée des différentes couches suivantes[4] :

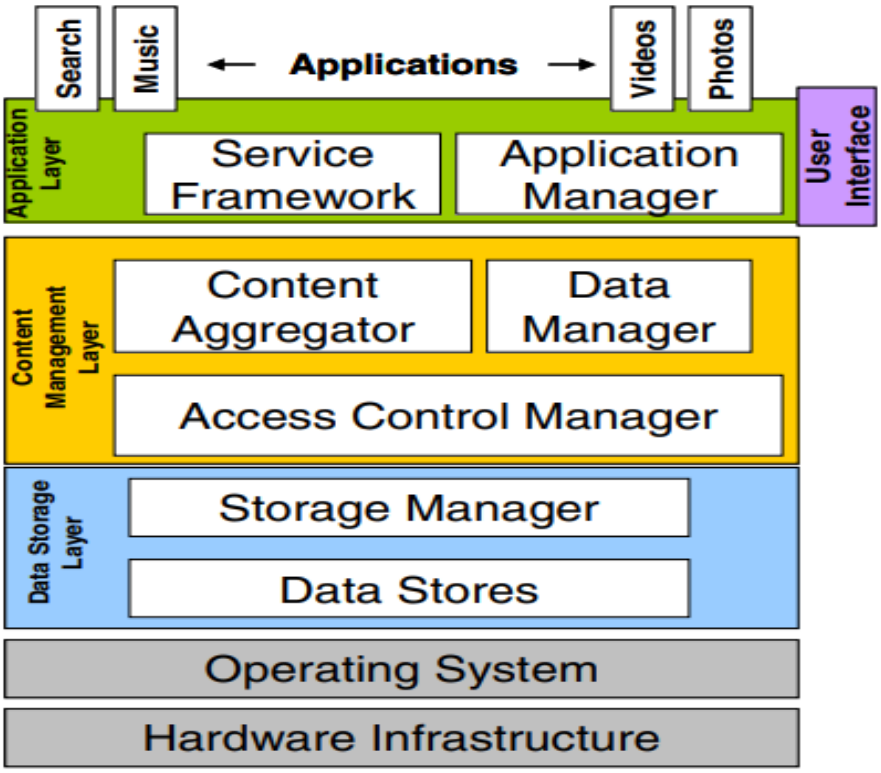


Figure1.1 Architecture de référence d'un RSL [4]

1.3.1 DataStorage layer

Cette couche se compose de deux composants : Le gestionnaire de stockage « **The StorageManager »** : qui est responsable de stocker efficacement les informations des graphes sociaux et de gérer les charges de base de données accrues. Cela est généralement réalisé en adoptant une mise en cache de mémoire distribuée. L'autre composant, appelé magasin de données **« Data Store »** : comprend les éléments de stockage qui stockent les éléments d'information d'un service de réseautage social. Les magasins de données peuvent être des bases de données multimédias, des bases de données de profils d'utilisateurs, etc.

1.3.2 Content Management layer

Cette couche est responsable de trois tâches principales. Tout d'abord, elle facilite l'incorporation d'informations sociales provenant de sites ***RSLs*** distants grâce à un agrégateur de contenu**«ContentAggregator »** : qui rassemble et organise le contenu des médias sociaux mais distribue également aux autres plateformes ***RSLs***. Deuxièmement, elle facilite la maintenance et la récupération du graphe de contenu social grâce au gestionnaire de données **« Data Manager ».** Troisièmement, elle contrôle l'accès des utilisateurs en créant et en maintenant un schéma de contrôle d'accès **« access control manager ».**

1.3.3 Application layer

Chaque site de réseau social en ligne prend en charge de nombreux services tels que la recherche, les fils d'actualités, l'accès mobile, etc. Les services communiquent avec le gestionnaire de données et le gestionnaire de contrôle d'accès afin d'analyser et de gérer le graphe de contenu social. Les applications sont fournies aux utilisateurs via un gestionnaire d'applications. Le gestionnaire d'applications **« application manager »** facilite l'interaction de l'utilisateur via un ensemble d'API. Cette composante comprend également un cadre de services **« service framework »** pour le développement de services évolutifs multi-langages. Un tel cadre permet aux utilisateurs de déployer des applications en abstrayant les parties de chaque langage qui ont tendance à nécessiter le plus de personnalisation dans une bibliothèque commune implémentée dans chaque langage de programmation.

1.4 Taxonomie des RSLs

Les Réseaux Sociaux en Ligne *(RSLs)* existants sur le Web sont de natures très variées. Selon leurs aspects clés, on peut les classer selon les quatre modèles suivants : Modèle de

domaine, Modèle de données, modèle de système et modèle de réseau [4]. CetteTaxonomie est donnée par la Figure 1.2

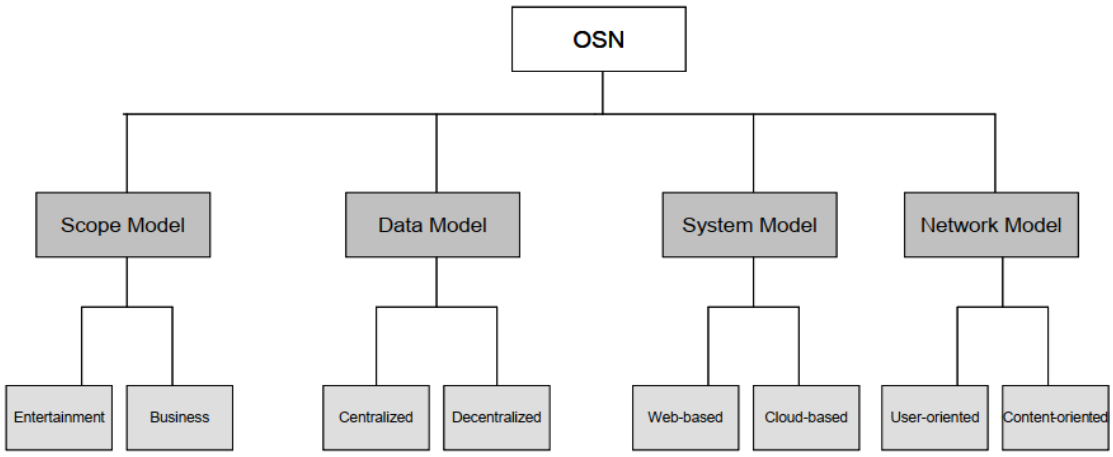


Figure1.2 Taxonomie des RSLs [4]

1.4.1 Modèle de domaine

Les RSLs peuvent être classés en deux catégories en fonction de leur modèle de domaine

- **Divertissement :** la plupart des RSLs sont dédiés au divertissement. Ils se concentrent sur la fourniture d'une expérience sociale en ligne amusante et interactive pour les utilisateurs inscrits. Les sites RSLs populaires qui sont principalement axés sur le divertissement sont Facebook, Myspace, Hi5 et Flickr.

- **Business :** dans cette catégorie, l'objectif des RSLs est de connecter les professionnels du monde entier pour les rendre plus productifs et plus performants. Grâce aux RSLs professionnels, les utilisateurs enregistrés créent des profils qui résument leur expertise et leurs réalisations professionnelles. Les sites RSLs représentatifs de cette catégorie sont LinkedIn et Xing.

1.4.2 Modèle de données

Cette dimension concerne la manière dont les données sont stockées dans les RSLs. On peut distinguer deux catégories de RSLs en fonction de leur modèle de données :

- **Les RSLs centralisés :** les données sont stockées dans un seul endroit physique (cluster ou data center) sous un seul domaine administratif. La plupart des RSLs fonctionnent aujourd'hui selon ce modèle, mais cela soulève des questions concernant la protection de la vie privée et la capacité de scalabilité face à l'augmentation du nombre d'utilisateurs et d'applications.

- **Les RSLs décentralisés :** les données sont stockées dans plusieurs endroits à travers plusieurs domaines administratifs. Les applications s'exécutent sur les machines des utilisateurs, ce qui permet une meilleure protection de la vie privée. Toutefois, cette catégorie présente l'inconvénient majeur de ne pas garantir la disponibilité permanente des utilisateurs en raison des pannes, des machines hors ligne et des déconnexions du réseau.

1.4.3 Modèle de système

Le modèle de système des RSLs peut être classé en deux catégories :

- **Les schémas basés sur le Web** : les serveurs d'application sont hébergés par des sites Web qui proposent des services et des API. Dans ce type de schéma, l'équilibreur de charge répartit la charge des requêtes, gère les pannes et transfère les requêtes vers les serveurs d'application appropriés. Dans les schémas basés sur le Web, la plupart des services des RSLs sont gratuits pour les utilisateurs.

- **Les schémas basés sur le Cloud :** les serveurs d'application sont hébergés par une infrastructure informatique telle que Amazon ElasticCompute Cloud (EC2). Dans ce type de schéma, chaque utilisateur stocke ses propres données sur une instance de machine virtuelle personnelle appelée Virtual Server. Les principaux avantages de ce schéma sont sa disponibilité élevée et son amélioration de la vie privée, car chaque utilisateur conserve ses données personnelles dans un Virtual Server résidant dans un environnement informatique en nuage. Les schémas basés sur le Cloud sont généralement intégrés à des infrastructures CDN (Content Delivery Network) (par exemple, Amazon Cloud Front est intégré à Amazon EC2), ce qui permet de distribuer le contenu aux utilisateurs finaux avec une latence faible et une vitesse de transfert de données élevée. Cependant, l'hébergement de données dans un Cloud augmente les coûts en raison de l'utilisation d'une infrastructure commerciale.

1.4.4 Modèle de Réseau

Le modèle de réseau des RSLs qui peut être classé en deux catégories distinctes :

- ***RSLs* orienté utilisateur :** Les RSLs orientés utilisateur mettent l'accent sur les relations sociales entre les membres et le partage de contenu est principalement entre les utilisateurs appartenant à la même communauté. Les exemples de RSLs orientés utilisateur incluent les réseaux sociaux tels que Facebook, LinkedIn et MySpace, les microblogs comme Twitter et les réseaux de rencontres tels que Meetic.

- ***RSLs* oriente contenu :** Le réseau des utilisateurs n'est pas déterminé par les relations sociales sous-jacentes, mais par leurs centres d'intérêts communs. Des exemples d'RSLs de cette catégorie sont les réseaux de blogs, les réseaux de questions-réponses et les réseaux de vidéos tels que YouTube.

1.5 Analyse des OSNs

L'analyse de réseau est liée à la formulation et à la résolution de problèmes qui présentent une structure en réseau. Cette structure est souvent représentée sous forme de graphe. Pour analyser ce graphe, il est nécessaire de se baser sur la théorie des graphes, qui offre un ensemble de concepts abstraits et de méthodes. De plus, ces méthodes peuvent être combinées avec d'autres outils analytiques et des méthodes de visualisation et d'analyse de graphes (à développer) [5]. L'analyse des réseaux sociaux permet de comprendre en détail le fonctionnement d'une société en se concentrant sur les relations entre les personnes, les groupes, etc., plutôt que sur les individus eux-mêmes et leurs caractéristiques [6]. Elle considère les personnes comme des entités imbriquées dans un graphe et cherche à comprendre les raisons spécifiques derrière certains comportements. Elle nous offre un moyen d'exprimer la confiance dans les environnements des réseaux sociaux. Les chercheurs exploitent souvent l'analyse des réseaux sociaux dans les cas suivants :

- Lorsqu'ils recherchent des informations, des explications ou des indications pour améliorer l'efficacité d'un réseau en fonction de ses objectifs.
- Lorsqu'ils veulent visualiser ou quantifier les données afin de découvrir des structures spécifiques entre les interactions des personnes.
- Lorsqu'ils veulent découvrir les chemins empruntés par les informations (analyse des flux d'informations).

- Lorsqu'ils veulent identifier les différents acteurs d'un réseau spécifique, ce qui permet d'extraire des informations utiles sur son fonctionnement.
- Lorsqu'ils veulent identifier les causes d'un mauvais comportement d'un réseau [5].

1.6 Sécurité et vie privée dans les RSLs

Les réseaux sociaux en ligne (RSLs) sont devenus extrêmement populaires en tant que principale plateforme pour l'échange et le partage d'informations. Les utilisateurs peuvent créer des profils, se connecter avec des amis et explorer les connexions d'autres utilisateurs dans le système [7]. Cependant, la nature ouverte des RSLs et l'accès facile aux données privées soulèvent des préoccupations majeures en matière de sécurité et de confidentialité. Une étude menée par Gao et al. [8] a identifié quatre catégories principales de problèmes de sécurité dans les RSLs : violation de la vie privée, spam et attaques de phishing, attaques Sybil et attaques de logiciels malveillants.

Les utilisateurs partagent une grande quantité d'informations dans les RSLs en utilisant différents services. Cela rend la violation de la vie privée possible, que ce soit par les fournisseurs de RSLs, les applications tierces ou d'autres utilisateurs. Les fournisseurs de RSLs peuvent exploiter les données des utilisateurs à des fins commerciales, telles que la publicité et l'amélioration des services. De plus, de nombreuses RSLs permettent l'exécution d'applications tierces sur leurs plateformes, ce qui donne accès aux données des utilisateurs sans leur pleine conscience.

La principale menace pour la protection de la vie privée provient des autres utilisateurs dans les RSLs, car de nombreux utilisateurs ne savent pas avec qui partager leurs données en toute confiance. Cela les conduit souvent à partager leurs informations avec des utilisateurs malveillants, mettant ainsi leur vie privée en danger. Afin d'éviter de telles situations, il est essentiel de développer des mécanismes permettant aux utilisateurs d'identifier les utilisateurs de confiance avec lesquels ils peuvent partager leurs données, tout en identifiant également les utilisateurs peu fiables à éviter. Cela permet de créer des cercles de confiance au sein desquels les membres peuvent partager et échanger des informations en toute sécurité et de manière transparente, ce qui constitue l'objectif de notre étude.

1.7 Conclusion

Les réseaux sociaux en ligne ont connu une croissance phénoménale au cours des dernières années, transformant la manière dont les individus communiquent, interagissent et partagent des informations. Ces plateformes ont évolué à partir des réseaux sociaux traditionnels pour inclure des fonctionnalités de partage de contenu, de messagerie, de création de profil et de découvertes de nouveaux utilisateurs, offrant ainsi une portée mondiale et une connectivité beaucoup plus grande que les réseaux sociaux traditionnels.

Dans ce chapitre, nous avons examiné de près les réseaux sociaux en ligne et leur architecture, en nous concentrant sur la manière dont les données sont stockées, gérées et présentées aux utilisateurs à travers différentes couches d'applications. Nous avons également exploré la taxonomie des RSLs, y compris les modèles de domaine, de données, de système et de réseau.

Enfin, nous avons abordé les concepts fondamentaux des méthodes d'analyse des Réseaux Sociaux en Ligne (RSLs) ainsi que les problèmes de sécurité inhérents aux réseaux sociaux en ligne (RSLs), notamment la confidentialité des données. Dans le prochain chapitre, nous examinerons de plus près la question de la confiance dans les réseaux sociaux en ligne (RSLs).

Chapitre 02

2.1 Introduction

La confiance joue un rôle essentiel dans de nombreux aspects de notre vie quotidienne, qu'il s'agisse de la psychologie, de l'économie, de la politique, de l'histoire ou de l'informatique. Elle est considérée comme un élément clé pour assurer le succès de nos transactions, où nos actions dépendent des actions des autres.

Cependant, la façon dont la confiance est interprétée en informatique varie selon les sous-domaines. Par exemple, dans les réseaux peer-to-peer, la confiance d'un nœud est définie par sa fiabilité dans le réseau selon une norme commune à tous les nœuds : la confiance d'un nœud correspond à sa réputation dans le réseau. En revanche, dans les Réseaux Sociaux en Ligne (RSLs), qui sont l'objet de notre étude, la confiance est subjective et dépend de plusieurs facteurs tels que l'expérience, les compétences, l'environnement, la situation, etc. De plus, la perception de la confiance peut varier d'une personne à une autre pour un même sujet.

Le chapitre est structuré de la manière suivante : tout d'abord, nous abordons la définition de la confiance dans diverses disciplines en mettant l'accent sur les RSLs, qui sont l'objet central de notre étude. Ensuite, nous examinons en détail les différentes caractéristiques de la confiance spécifiquement dans le contexte des RSLs. Enfin, nous explorons la présentation et l'importance de la confiance dans les RSLs, en mettant en évidence son rôle crucial dans ces environnements.

2.2 Définition de la confiance

Selon McKnight et Chervany [9], il existe une perception différente de la confiance dans diverses disciplines, ce qui rend difficile la recherche d'une définition unique. Cette diversité découle principalement de deux raisons. Tout d'abord, la confiance est un concept abstrait qui est souvent utilisé de manière interchangeable avec d'autres concepts connexes tels que la confiance en soi, la fiabilité et la réputation. Deuxièmement, la confiance est un concept multidimensionnel qui englobe généralement des aspects émotionnels, comportementaux, cognitifs, et bien d'autres encore. [10]. La définition de la confiance varie d'une discipline à l'autre, reflétant différentes perspectives. En général, la confiance est considérée comme une évaluation de la probabilité qu'une entité se comporte d'une certaine manière.

Deutsch propose en psychologie une définition de la confiance comme étant le choix individuel et irrationnel d'une personne face à un événement incertain, où les pertes anticipées sont supérieures aux gains espérés [11].

En sociologie et en histoire, Luhmann adopte une perspective différente de Deutsch en situant la confiance dans un contexte social multidimensionnel, où les individus interagissent les uns avec les autres. Il considère la confiance comme un élément fondamental de la vie humaine permettant de réduire la complexité du monde social [12].

Dans le domaine de l'économie, le Centre Commun de Recherche de la Commission européenne définit la confiance comme une propriété d'une relation commerciale, englobant à la fois la confiance envers les partenaires et les transactions réalisées.

Dans les sciences sociales et politiques, Gambetta définit la confiance comme un niveau de probabilité subjective avec lequel un agent accomplira une action spécifique dans un contexte donné. Selon lui, ce niveau de confiance particulier dépend à la fois de l'agent qui l'évalue et des circonstances, ce qui rend la confiance spécifique au contexte plutôt que généralisable [13].

Sztompka propose une définition générale et simple de la confiance en la décrivant comme un pari sur les actions futures éventuelles des autres.Il y a deux composants principaux dans cette définition : la croyance et l'engagement [14].

La définition de la confiance comprend deux composantes principales : la croyance et l'engagement. Tout d'abord, la personne qui accorde sa confiance (le confiant) croit que l'autre personne en qui elle place sa confiance (le confiable) agira d'une certaine manière. Cependant, la simple croyance ne suffit pas à établir la confiance. La confiance se manifeste lorsque cette croyance se traduit par un engagement envers une action spécifique. Par exemple, on peut dire que John fait confiance à Bob pour les e-mails s'il choisit de lire les messages que Bob lui envoie, ce qui repose sur la croyance que Bob ne gaspillera pas son temps.

Dans le domaine de l'informatique, Golbeck [15] donne une définition de la confiance comme la croyance en un déroulement favorable des actions futures d'une autre entité. Selon Grandison [16], l'acte de faire confiance se produit dans un contexte spécifique et est défini par une croyance quantifiée concernant les compétences de l'entité à qui l'on accorde sa confiance.

Cette quantification peut être exprimée à l'aide d'une échelle de valeurs ou d'une simple classification.

Dans le contexte des réseaux sociaux en ligne, qui sont le sujet de notre étude, Golbeck et al. [17] proposent la définition suivante de la confiance : "Faire confiance à une personne implique un engagement envers une action, basé sur la croyance que les actions futures de cette personne conduiront à des résultats positifs". Cette définition rejoint celles précédemment présentées par Deutsch et Sztompka, mettant en évidence la croyance et l'engagement comme deux composantes essentielles de la confiance. Dans les réseaux sociaux, les utilisateurs attribuent des évaluations confidentielles (des descriptions simples) à leurs connexions avec d'autres utilisateurs, en prenant en compte diverses informations (historique, comportement, etc.) [18]. Malgré leurs différences, toutes les définitions de la confiance évoquées ci-dessus partagent trois éléments communs :

- Un individu capable de raisonner qui accorde sa confiance (le confiant).
- Une personne ou une entité en qui la confiance est accordée (le bénéficiaire de la confiance).
- Une situation ou des circonstances dans lesquelles la confiance est définie.

2.3 Propriétés de la confiance

Dans cette section, nous examinons les caractéristiques de la confiance spécifiquement liées aux réseaux sociaux en ligne (RSLs). Ces caractéristiques, telles que la transitivité, l'asymétrie et la personnalisation, sont dérivées de la définition de la confiance et servent de fondement au développement d'algorithmes de calcul de la confiance [19]. Il est essentiel de prendre en compte ces caractéristiques lors de la conception des algorithmes, car si l'une d'entre elles diffère dans le contexte des RSLs, des ajustements doivent être apportés aux algorithmes existants.

2.3.1 Transitivité

La transitivité est une propriété fondamentale de la confiance, qui permet à celle-ci de se propager le long des chemins de confiance pour atteindre d'autres utilisateurs [19]. Par exemple, si John fait confiance à Bob et que Bob fait confiance à Eric, alors John aura confiance en Eric [20]. Cependant, il est important de noter que la confiance n'est pas parfaitement

transitive au sens mathématique. Même si Bob accorde une grande confiance à Eric, cela ne signifie pas nécessairement que John accordera une grande confiance à Eric. Néanmoins, il existe une notion de propagation de la confiance entre les individus [21]. Dans les applications, la transitivité de la confiance se manifeste de deux manières : la confiance en une personne spécifique et la confiance dans les recommandations de cette personne pour accorder sa confiance à d'autres individus. Par exemple, John peut faire confiance à Bob en matière de films, mais cela ne signifie pas qu'il fera confiance à toutes les recommandations de Bob concernant d'autres personnes ayant une expertise en cinéma. Malgré cette dichotomie, il est courant dans les réseaux sociaux d'utiliser une seule valeur pour représenter les deux situations. Ainsi, si l'on dit que John fait confiance à Bob en matière de films, cela signifie qu'il accorde sa confiance aux films recommandés par Bob ainsi qu'aux personnes pour lesquelles Bob affirme avoir des connaissances approfondies dans le domaine cinématographique.

Il est crucial de faire une distinction entre deux types de confiance : la confiance de référence (referral trust) et la confiance fonctionnelle (functional trust). La confiance de référence concerne la confiance que l'on accorde aux recommandations d'une personne, tandis que la confiance fonctionnelle concerne la confiance en une personne dans un domaine spécifique. Lorsqu'il existe une relation de confiance directe entre deux personnes, où une personne accorde sa confiance à une autre personne, on parle de confiance directe. En revanche, lorsqu'il y a une relation de confiance entre deux personnes sur la recommandation d'une troisième personne, on parle de confiance indirecte. Vous pouvez consulter la Figure 2.1 pour visualiser ces relations.

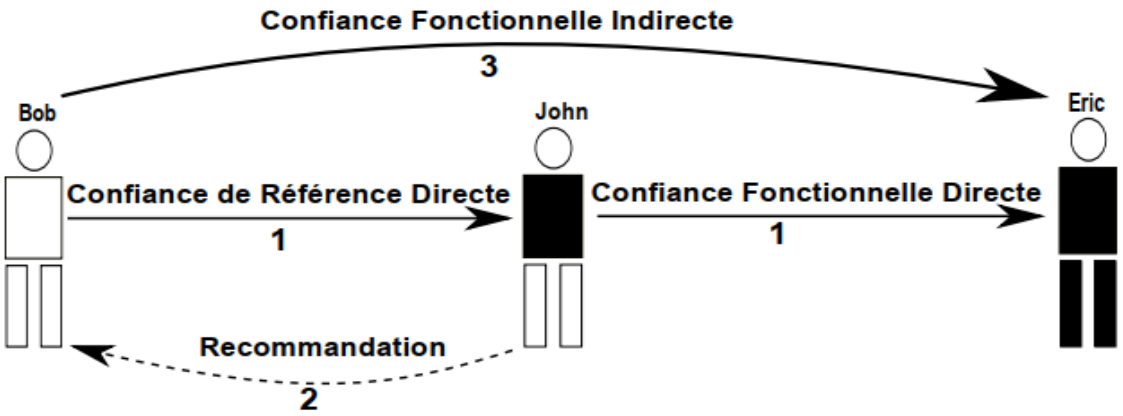


Figure 2.3 Principe de transitivité de la confiance [22]

Lorsque la relation de confiance entre deux personnes se fait par l'intermédiaire de plusieurs personnes (plus de deux recommandations), on parle de la confiance indirecte d´dérivée par transitivité ou tout simplement de la confiance dérivée par transitivité. Voir la Figure 2.2.

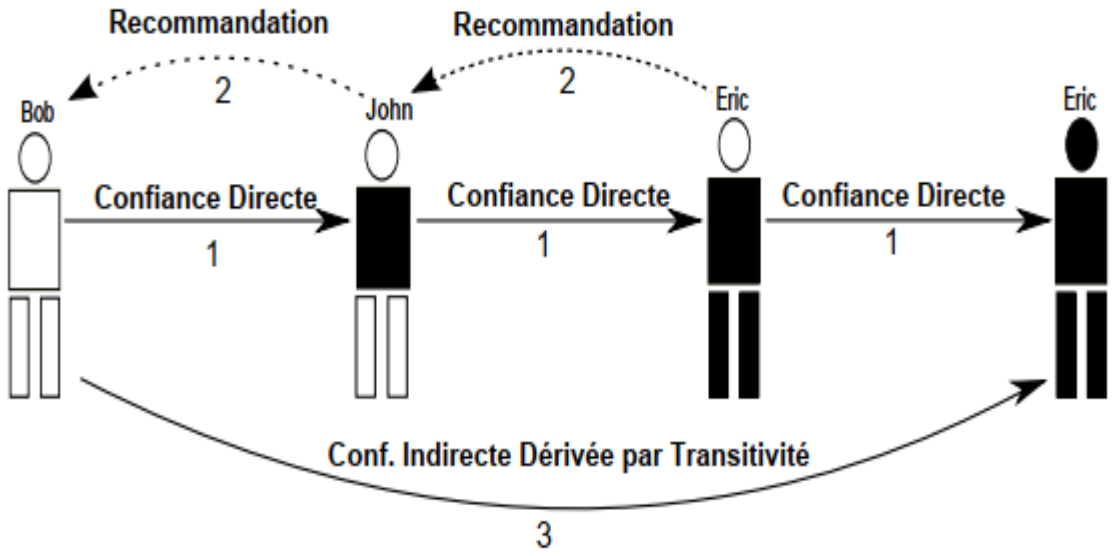


Figure2.4 Confiance dérivée par transitivité [22]

2.3.1.1 Critères de dérivation de la confiance fonctionnelle

Pour dériver la confiance fonctionnelle à partir de la confiance de référence, il est nécessaire que le dernier lien de confiance soit une confiance fonctionnelle, tandis que tous les liens de confiance précédents sont des confiances de référence [22]. Dans des situations pratiques, le domaine de confiance peut être général ou spécifique. Par exemple, avoir des connaissances en réparation d'une pompe d'injection d'une voiture est plus spécifique que d'être simplement mécanicien, car le premier domaine est un sous-ensemble du deuxième. Lorsque le domaine de confiance fonctionnelle est un sous-ensemble ou égal au domaine de confiance de référence, il est possible d'établir un chemin transitif de confiance.

2.3.1.2 Critères de consistance du domaine de confiance

Pour qu'un chemin de confiance transitif soit valide, il est nécessaire que le domaine de confiance du dernier lien (confiance fonctionnelle) soit un sous-ensemble de tous les domaines des liens précédents (confiances de référence) [22]. Un chemin de confiance transitif s'achève lorsqu'il rencontre le premier lien de confiance fonctionnelle. Il est possible qu'une personne ait à la fois une confiance fonctionnelle et une confiance de référence envers une autre personne,

à condition que ces confiances soient exprimées par des liens de confiance distincts. L'existence de ces deux liens entre Bob et John signifie que Bob fait confiance à John ainsi qu'à ses recommandations.

Il est important de noter que la présence d'une confiance négative dans une chaîne transitive peut avoir un effet paradoxal sur la confiance dérivée. Par exemple, si Bob ne fait pas confiance à John et que John ne fait pas confiance à Eric, il est raisonnable que Bob fasse confiance à Eric (par dérivation) puisqu'il croit que John essaie de le tromper et qu'il ne peut donc pas compter sur lui. En utilisant le principe "l'ennemi de mon ennemi est mon ami", le fait que John recommande à Bob de ne pas faire confiance à Eric constitue un argument en faveur de la confiance de Bob envers Eric.

Pendant la transitivité, la confiance diminue à mesure que le nombre de sauts de transition le long du chemin de confiance augmente [20]. Cette diminution de confiance globale est non linéaire [23, 24] et peut être divisée en trois phases, comme illustré dans la figure suivante.

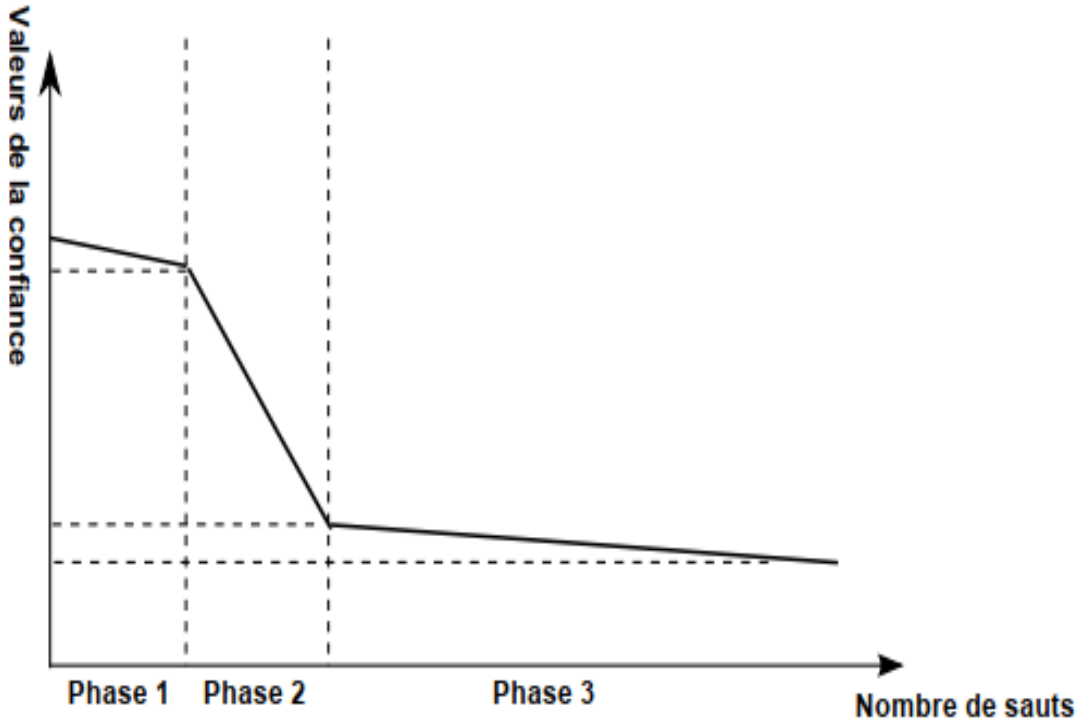


Figure 2.5 Diminution de la confiance par transitivité

- Dans la phase 1 de la transitivité (diminution lente), la confiance diminue progressivement le long du chemin reliant la source à la cible, jusqu'à un certain nombre de sauts. La source prend en compte la proximité de la cible, en se limitant à un certain nombre de sauts de transition.

- Dans la phase 2 de la transitivité (diminution rapide), au-delà d'un certain nombre de sauts de transition, la confiance diminue rapidement jusqu'à atteindre une valeur minimale. À ce stade, la cible devient étrangère à la source.
- Dans la phase 3, lorsque la confiance entre la source et la cible approche de sa valeur minimale et dépasse un certain nombre de sauts, la vitesse de diminution de la confiance passe de rapide à lente. La cible devient alors complètement étrangère à la source.

2.3.2 Asymétrie

La confiance est une relation personnelle et subjective entre les utilisateurs, créant des relations unidirectionnelles dans les réseaux sociaux [19]. Cela signifie que la confiance n'est pas nécessairement réciproque entre deux personnes impliquées dans une relation de confiance. Les individus accordent leur confiance à des degrés variables en raison de leurs expériences, de leur histoire et de leurs compétences psychologiques. Par exemple, les parents et les enfants ont des niveaux de confiance différents les uns envers les autres.

En général, dans les organisations hiérarchiques, les subordonnés accordent davantage leur confiance aux responsables [25]. Même en dehors des structures hiérarchiques, la confiance conserve sa propriété d'asymétrie. Dans certaines situations et sous certaines conditions extrêmes, une personne peut accorder sa confiance à une autre sans réciprocité. Ce phénomène est appelé "one-way trust" (confiance unilatérale) [26, 27]. Lors de l'évaluation de la confiance, il est important de prendre en compte cette caractéristique. Pour représenter la direction de la confiance, une flèche est utilisée, pointant de la personne qui accorde sa confiance vers celle en qui la confiance est placée.

2.3.3 Personnalisation

La confiance est un avis personnel. Le degré de confiance qu'une personne accorde à une autre peut varier d'une personne à une autre. Cette propriété est utilisée pour définir et formuler la confiance locale [19]. Il est courant que deux individus aient des opinions

divergentes concernant une même personne. Par exemple, supposons que John et Bob soient deux acheteurs et qu'Eric soit un agent de transit. Il est possible que John fasse confiance à Eric tandis que Bob ne lui accorde pas sa confiance, en se basant sur leurs préférences personnelles concernant les services fournis par Eric et leurs critères d'évaluation de la confiance.

Les personnes peuvent avoir des conflits d'intérêt, de priorité, d'opinions et des critères différents en ce qui concerne la confiance dans différents domaines [17]. La confiance varie donc d'un individu à un autre. Lors du calcul de la confiance, il est essentiel de prendre en compte la personnalisation (la perspective) des individus afin de refléter leurs intérêts et leurs opinions.

2.3.4 Composabilité

La composabilité décrit la manière de combiner plusieurs valeurs de confiance, issues de différents chemins, concernant une personne [19]. Lorsque plusieurs individus fournissent des recommandations ou des informations sur une personne spécifique, il est nécessaire de traiter ces informations afin de décider si l'on doit accorder ou non sa confiance à cette personne. La composabilité devient pertinente lorsque ces recommandations sont considérées comme des éléments de croyance, permettant ainsi de déduire une valeur de confiance spécifique. Les recommandations des contacts ou des voisins concernant une personne donnée sont utilisées comme entrées dans une fonction de composition, qui peut varier selon l'algorithme utilisé [Richardson et al., 2003].

2.3.5 Relativité

La confiance est contextualisée, ce qui signifie que faire confiance à quelqu'un dans un domaine spécifique ne garantit pas nécessairement de lui faire confiance dans un autre domaine [19]. Par exemple, John peut accorder sa confiance à Bob pour conduire une voiture, mais pas pour piloter un avion.

2.3.6 Temporalité et évolutivité

La confiance est sujette à l'évolution et n'est pas immuable dans le temps. Le fait que John ait accordé sa confiance à Bob par le passé ne garantit pas automatiquement qu'il lui fera toujours confiance à l'avenir. Des informations nouvelles et pertinentes, ainsi que l'évolution des circonstances, peuvent amener John à réévaluer sa confiance envers Bob.

2.3.7 Non distributivité

La confiance envers un groupe de personnes n'implique pas automatiquement la confiance envers chacun de ses membres. Par exemple, si John accorde sa confiance à un groupe de personnes pour un projet spécifique, cela ne signifie pas qu'il accorde sa confiance à tous les membres individuellement pour ce même projet.

2.3.8 Réflexivité

Il est également possible qu'une personne ait confiance en elle-même dans un domaine particulier. Par exemple, John peut avoir confiance en sa propre capacité à conduire une voiture.

2.4 Valeurs de la confiance

La confiance est une information qui représente une relation sociale dans les réseaux sociaux. Elle peut être exprimée sous différentes formes, telles qu'un label ou une valeur [18]. Les utilisateurs ont différentes façons d'exprimer la confiance dans les réseaux sociaux. Par exemple, sur Ecademy, les utilisateurs peuvent choisir de ne faire aucune déclaration ou de déclarer un ami en qui ils ont confiance, sans pour autant classer les personnes selon un niveau de confiance spécifique.

Cependant, la confiance n'est pas un concept binaire et peut prendre des valeurs dans un intervalle défini [Gambetta 1990, Marsh 1992, Marsh 1994]. Au-dessus d'un seuil spécifique, une personne est considérée comme digne de confiance, tandis qu'en dessous de ce seuil, elle ne l'est pas. Ce seuil peut varier en fonction des circonstances, selon les arguments avancés par Gambetta. La confiance peut également être représentée par des valeurs multiples, comme c'est le cas sur Overstock.com.

Certains systèmes utilisent des échelles ou des niveaux pour représenter la confiance, en utilisant des valeurs discrètes (par exemple, {0,...,10}) ou continues (par exemple, [0,1]), ou en utilisant des labels tels que "petit", "moyen" ou "grand" [28]. Ces valeurs de confiance jouent un rôle important dans le calcul de la confiance. L'utilisation de niveaux avec des valeurs offre une plus grande praticité que l'utilisation de labels, car les valeurs peuvent être manipulées mathématiquement [29].

2.5 Importance de la confiance dans les réseaux sociaux

Les Réseaux Sociaux en Ligne (RSLs) ont connu une croissance importante depuis les années 1990 et ont joué un rôle majeur dans le développement des communautés en ligne. Ils

facilitent la connexion et les échanges d'idées et d'expériences entre leurs membres, et sont devenus une plateforme de communication essentielle pour les individus et les organisations [3].

Dans de nombreux réseaux sociaux, le concept de FOAF (Friend of a Friend - l'ami de mon ami est mon ami) est largement répandu [30]. L'idée sous-jacente à ce concept est que la relation d'amitié, qui est à la base de chaque relation, est transitive. Ainsi, les relations FOAF suggèrent indirectement que la confiance est également transitive dans les réseaux sociaux. Cependant, comme nous l'avons vu précédemment, la confiance se propage mais n'est pas nécessairement transitive : on peut faire confiance à une personne sans être certain de faire confiance à ses amis. Par conséquent, il existe un risque inhérent aux informations confidentielles des membres dans de tels réseaux sociaux en raison de l'hypothèse sous-jacente à la confiance implicite dans les relations FOAF. La confiance entre les utilisateurs est une véritable source de puissance et de développement pour chaque communauté.

Un autre concept étudié dans les réseaux sociaux en relation avec la confiance est la force des liens entre les personnes. Granovetter a introduit le concept de "liens forts" en 1973. Les liens forts sont caractérisés par la durée, l'intensité, l'émotion, l'intimité et la réciprocité [31]. Ils représentent un cercle de confiance où chaque personne fait confiance aux autres. En revanche, les "liens faibles" correspondent à de simples connaissances entre les individus. L'analyse des liens forts et des liens faibles, réalisée par Gilbert et Karahalios sur les données de Facebook en utilisant une approche binaire, est importante pour comprendre la confiance dans les réseaux sociaux. Un lien fort établit implicitement une relation de confiance.

2.6 Conclusion

Dans ce chapitre, nous avons exploré le concept de confiance dans les Réseaux Sociaux en Ligne (RSLs). Nous avons commencé par définir la confiance dans différents domaines, en mettant l'accent sur son application spécifique dans le contexte des RSLs, qui constitue le sujet central de notre étude. Nous avons examiné en détail les caractéristiques clés de la confiance dans les RSLs, telles que la transitivité, l'asymétrie et la personnalisation. De plus, nous avons introduit des concepts fondamentaux liés à la confiance, tels que la confiance fonctionnelle, la confiance de référence et la confiance directe. Enfin, nous avons discuté des différentes formes de représentation de la confiance dans les RSLs, qu'il s'agisse de valeurs binaires, multiples, discrètes ou continues.

Ayant ainsi présenté la notion de confiance dans les RSLs dans ce chapitre, nous nous concentrerons dans le chapitre suivant sur l'étude des algorithmes permettant de calculer la confiance dans les RSLs.

Chapitre 03

Chapitre 03 :Etat de l'art sur les algorithmes de confiance

3.1 Introduction

L'objectif principal de concevoir et mettre en œuvre des algorithmes de confiance dans les Réseaux Sociaux en Ligne (RSLs) est de former des groupes d'utilisateurs de confiance afin de faciliter le partage sécurisé de données privées, tout en empêchant l'accès de personnes malveillantes à ces données. Ces algorithmes se divisent en deux catégories : ceux qui exploitent directement le réseau de confiance et ceux qui transforment le réseau de confiance en un autre type de réseau, tel qu'un réseau de flots ou un réseau de résistances. Dans la première catégorie, l'idée de base consiste à propager la confiance d'un nœud source vers un nœud cible en utilisant différents chemins et en définissant des fonctions de propagation et d'agrégation. Dans la deuxième catégorie, chaque algorithme exploite les caractéristiques et les outils propres au type de réseau obtenu après transformation. Dans les RSLs, un bon algorithme est celui qui fournit des résultats de qualité tout en étant le moins complexe possible, car un algorithme présentant une complexité élevée (exponentielle) n'est plus utilisé en pratique au-delà d'une certaine quantité de données (un million par exemple). La structure du graphe reflète clairement le niveau de confiance dans le réseau social, et une forte densité du graphe (c'est-à-dire de nombreuses interconnexions entre les membres) peut générer un niveau élevé de confiance dans le réseau. [32].

3.2 Complexité algorithmique

Avant d'entamer la phase de développement, l'évaluation de la complexité (temps d'exécution) d'un algorithme revêt une grande importance. La complexité du calcul est indépendante de divers facteurs tels que le matériel, le langage de programmation et le compilateur utilisés. Elle dépend uniquement de la taille des données d'entrée et du nombre d'opérations élémentaires exécutées par l'algorithme. La taille des données d'entrée, notée n, représente la quantité d'informations manipulées par l'algorithme. Par exemple, pour les tableaux, n correspond au nombre d'éléments du tableau, et pour les graphes, n correspond au nombre de sommets plus le nombre d'arcs. Les opérations élémentaires englobent les opérations arithmétiques (addition, soustraction, multiplication, division), logiques (comparaisons), de branchement et d'affectation. En général, l'évaluation de la complexité peut être effectuée en se basant sur deux approches distinctes : le pire des cas et la moyenne. L'approche moyenne consiste à évaluer la durée moyenne d'exécution de l'algorithme, tandis que l'approche du pire des cas vise à évaluer la durée maximale d'exécution de l'algorithme.

3.2.1 Calcul de la complexité

En pratique, la complexité d'un algorithme est une fonction C(n) qui donne le nombre d'opérations élémentaires exécutées par l'algorithme dans le pire des cas en fonction de la taille n. Généralement, on ne calcule pas exactement la complexité C(n) car cela peut être long et fastidieux, et souvent la précision requise est inutile [33] (calcul exact pour une valeur fixée de n). On se contente souvent d'une approximation par une analyse asymptotique (que se passe-t-il lorsque n tend vers l'infini ?). La complexité est exprimée en utilisant la notation "grand O" (notation de Landau), où l'on écrit C(n) = O(g(n)). Cela signifie que C(n) est borné par c × g(n) (C(n) est asymptotiquement dominé par g(n)), où c'est une constante, lorsque n tend vers l'infini. Les instructions de base (opérations élémentaires) mentionnées précédemment ont un temps constant noté O(1).

Domination asymptotique : Soient deux fonctions f et g, on dit que g est d'ordre supérieur ou égal à f (f est asymptotiquement dominée par g) ou d'ordre au moins f, si l'on peut trouver un entier n0 et un réel positif c tels que $\forall n \geq n0$, $f(n) \leq c \times g(n)$. On écrit f = O(g), que l'on prononce "f est en grand O de g" [33]. L'allure des deux fonctions f et c × g est donnée par la figure suivante.

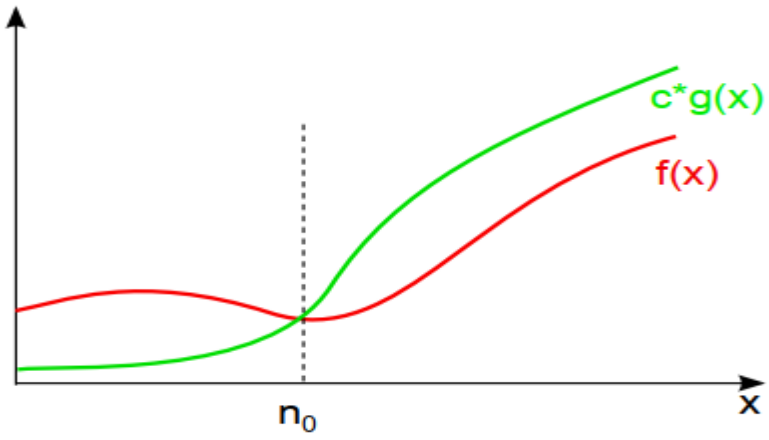


Figure 3.6 Domination asymptotique

3.2.2 Exemple de calcul de la complexité

Soit l'algorithme 1 suivant

Algorithm 1 Exemple de calcul de la complexité

```
s ← 0
for i ← 1 to n do
 for j ← 1 to 2n do
        end for

s ← s + 1
end for
```

Pour calculer l'ordre de complexité (temps de calcul asymptotique) de cet algorithme C(n), on cherche d'abord le nombre d'opérations élémentaires exécutées par l'algorithme, C(n). Cet algorithme comporte deux instructions : une instruction d'affectation élémentaire à l'entrée de la boucle et une autre instruction d'affectation à l'intérieur de la boucle imbriquée, qui est composée de deux opérations élémentaires (addition et affectation). La première instruction est exécutée une seule fois, tandis que la deuxième instruction est exécutée $2n^2$ fois. Ainsi, le nombre total d'opérations élémentaires exécutées par l'algorithme C(n) est donné par $C(n) = 1 + 2n^2 \times 2 = 1 + 4n^2$. Il reste à trouver la fonction d'ordre supérieur ou égale à C(n), g(n). Il est facile de voir que C(n) est inférieure ou égale à $5n^2$ pour tout $n \geq 1$:

$C(n) = 1 + 4n^2 \leq 5n^2, \forall n \geq 1$

Donc il existe des valeurs $(n_0, c) = (1, 5)$ telles que $C(n) \leq c(g(n))$, où $g(n) = n^2$. Par conséquent, $C(n) = O(n^2)$.

La complexité de cet algorithme est donc quadratique, c'est-à-dire que le temps de calcul varie asymptotiquement comme n^2.

3.2.3 Classes de complexité

Il existe plusieurs classes de complexité prédéfinies, les plus importantes sont (par ordre croissant en termes de O) [34]:

- **O(1):** Complexité constante - pas d'augmentation du temps d'exécution lorsque le paramètre n croît.
- **O(log(n)):** Complexité logarithmique - augmentation très faible du temps d'exécution lorsque le paramètre n croît.

- **O(n):** Complexité linéaire - augmentation linéaire du temps d'exécution lorsque le paramètre n croît.
- **nO($n^{log(n)}$):** Complexité quasi-linéaire - augmentation légèrement supérieure à O(n).
- **O(n^2):** Complexité quadratique - lorsque le paramètre n est doublé, le temps d'exécution est multiplié par quatre.
- **O(n^3):** Complexité cubique - lorsque le paramètre n est doublé, le temps d'exécution est multiplié par huit.
- **O(n^j):** Complexité polynomiale - lorsque le paramètre n est doublé, le temps d'exécution est multiplié par 2^j.
- **O($n^{log(n)}$):** Complexité quasipolynomiale.
- **O(2^n):** Complexité exponentielle - lorsque le paramètre n est doublé, le temps d'exécution est élevé à la puissance 2.
- **O(n!):** Complexité factorielle - asymptotiquement équivalente à n^n.

3.2.4 Exemple de complexité et temps d'exécution

Dans la plupart des cas, ces complexités ne sont pas faciles à appréhender. Le Tableaux 3.1 présente quelques-unes de celles-ci avec les temps d'exécution correspondants sur des données de taille un million, la première colonne représente la puissance du processeur d'exécution. On remarque sur ce tableau que les algorithmes exponentiels au-delà d'une certaine taille de données ne sont pas utilisables en pratique.

Tableau 3.1 Exemple de complexité et temps d'exécution [34]

Complexité / Flops	ln(n)	n	n^2	2^n
10^6	0.013ms	1s	278 heures	10000 ans
10^9	0.013μs	1ms	15 minutes	10 ans
10^{12}	0.013ns	1μs	1s	1 semaine

3.2.5 Classes des problèmes informatiques

Selon leurs degrés de difficulté formelle, les problèmes informatiques peuvent être classés en trois grandes classes :

- **La classe P :** regroupe l'ensemble des problèmes qu'on peut résoudre avec un algorithme de complexité polynomiale. Cette classe comporte tous les problèmes faciles tels que le calcul de l'itinéraire le plus court entre deux sommets d'un graphe, le calcul du minimum d'un ensemble de valeurs, le tri d'une base de données [33].

- **La classe NP :** regroupe l'ensemble des problèmes dont on peut vérifier une solution avec un algorithme de complexité polynomiale. Elle comporte tous les problèmes difficiles tels que le problème de partition (séparation d'un ensemble S en deux sous-ensembles A et B de telle sorte que la somme des éléments de A égale celle de B [33]).

- **La classe NP-Complet :** regroupe l'ensemble des problèmes les plus difficiles de la classe NP (au moins aussi difficile que tout problème de NP) tels que le problème du marchand itinérant (TSP, traveling salesmanproblem) (existence d'un circuit traversant un ensemble de nœuds et de longueur inférieure à d).

Un problème P de NP est NP-complet si tout problème de NP peut être transformé en P en un temps polynomial [35].

3.3 Algorithmes de confiance

La littérature propose de nombreux algorithmes de confiance dans les réseaux sociaux, qui peuvent être fondamentalement divisés en deux catégories : les algorithmes globaux et les algorithmes locaux. Les algorithmes globaux considèrent toutes les relations de confiance entre les utilisateurs et calculent une valeur de confiance unique (appelée réputation) pour chaque utilisateur, en tenant compte des autres utilisateurs du système. Si l'utilisateur A a une confiance x envers l'utilisateur B, alors l'utilisateur C aura également une confiance x envers B. Ce type d'algorithme convient davantage aux applications où la confiance reflète un comportement universellement jugé bon ou mauvais. En revanche, les algorithmes locaux calculent plusieurs valeurs de confiance pour chaque utilisateur cible en tenant compte des tendances et des opinions personnelles de chaque utilisateur source. Ils se basent sur la confiance entre pairs utilisateurs (source, cible) : en prenant un nœud source et un nœud cible en entrée, ils calculent la valeur de confiance que la source accorde à la cible. Ces algorithmes conviennent mieux aux

applications basées sur l'opinion, où il n'existe pas d'opinions de jugement communes et où les utilisateurs ont des opinions différentes.

3.3.1 Advogato

3.3.1.1 Description de l'algorithme

L'algorithme Advogato, développé par Levien [36], est largement reconnu dans le domaine du Web. Il est utilisé sur le site Advogato.com, une plateforme dédiée au développement de logiciels libres, dans le but de prévenir les attaques malveillantes en utilisant une méthode d'évaluation entre pairs. Les utilisateurs du site se certifient mutuellement en attribuant trois niveaux de confiance possibles : apprenti, aventurier et maître. Le réseau des membres est représenté par un graphe de confiance (G), où les membres sont les nœuds et les relations de confiance sont les arêtes. L'algorithme Advogato utilise des modèles de réseaux de flots pour calculer la confiance des membres par rapport à un groupe de membres fiables appelé "grains" (seednodes). L'objectif principal de cet algorithme est de diviser les membres du graphe en deux groupes : les membres fiables et les membres non fiables, en fonction d'un grain donné, afin d'évaluer la réputation de chaque nœud du graphe.

Les paramètres d'entrée de l'algorithme Advogato sont le graphe de confiance (G) et un entier (n) représentant la capacité du grain, c'est-à-dire le nombre de personnes que le grain examinera. En sortie, l'algorithme fournit une liste de personnes fiables.

Le fonctionnement de l'algorithme Advogato repose sur trois étapes clés : l'assignation des capacités, la conversion du graphe et le calcul du flot maximal.

3.3.1.2 Assignation de capacités

L'assignation de capacités à chaque sommet du graphe repose sur le calcul du plus court chemin depuis la source (seednode) jusqu'à chaque sommet. Les plus courts chemins sont obtenus à l'aide de l'algorithme de parcours en largeur (BFS). La capacité du seednode, notée n, est fournie en tant que paramètre d'entrée, comme expliqué précédemment. La capacité de chaque niveau successif est égale à la capacité du niveau précédent divisée par le nombre moyen d'arcs sortants de ce niveau. Ainsi, tous les sommets d'un même niveau ont la même capacité. La figure suivante illustre un exemple d'assignation de capacités.

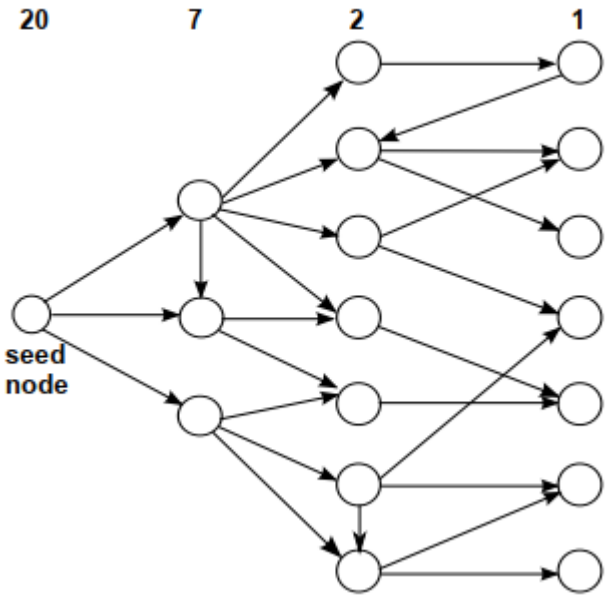


Figure 3.7 Assignation de capacités [66]

La capacité du seednode (niveau 0) est égale à 20 qui est représenté le nombre total des sommets à examiner, par contre la capacité du niveau suivant (niveau 1) est égale à 7 qui représente la capacité du niveau supérieur (niveau 0) divisée par le nombre moyen de nœuds sortants (3). Le calcul des capacités des niveaux inférieurs se fait de la même manière.

3.3.1.3 Conversion du graphe

Pour appliquer l'algorithme de calcul du flot maximum de Ford-Fulkerson, le graphe obtenu à l'étape précédente, comprenant une source et plusieurs cibles, doit être converti en un graphe spécifique comportant une seule cible. Cette cible spéciale est représentée par un nœud appelé "Supersink" ($). La conversion du graphe est réalisée de la manière suivante :

Chaque sommet v du graphe, avec une capacité c(v), est divisé en deux sommets, v- et v+, et deux arcs sont ajoutés. Le premier arc relie v- à v+ avec une capacité de c(v)-1, et le deuxième arc relie v- à $ avec une capacité de 1. Tous les arcs entrants vers v deviennent des arcs entrants vers v-, et tous les arcs sortants de v deviennent des arcs sortants de v+ avec une capacité infinie.

L'exemple présenté dans la figure suivante illustre cette conversion de graphe.

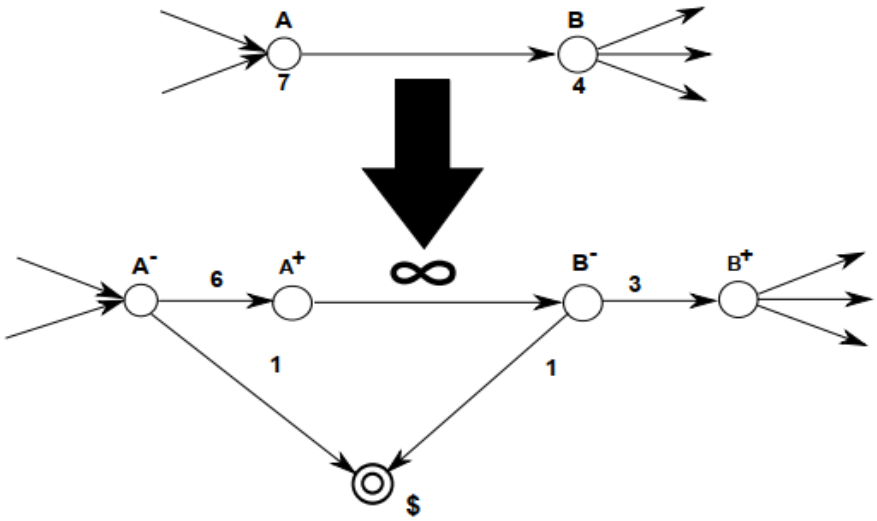


Figure 3.8 Conversion du graphe

3.3.1.4 Calcul du flot maximum

Une fois le graphe transformé, on procède au calcul du flot maximum entre le seednode s et le supersink $ en utilisant l'algorithme de Ford-Fulkerson. Les capacités des arcs sont prises en compte lors de ce calcul. Les sommets renvoyés par l'algorithme correspondent à ceux qui présentent un flux négatif du sommet vers le supersink. Ces sommets représentent les utilisateurs considérés comme fiables.

3.3.1.5 Exemple

Considérons le graphe de la figure suivante.

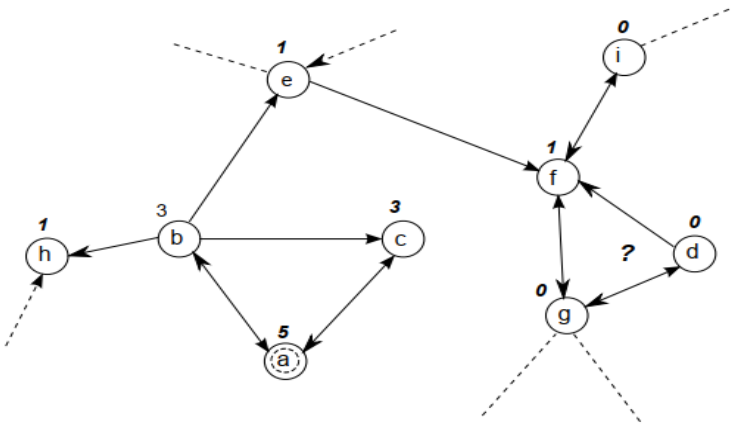


Figure3.9 Graphe de confiance avant conversion [37]

Supposons que la capacité initiale du seednode a soit de 5. Pour calculer la capacité des sommets du niveau 1, il suffit de diviser la capacité du niveau 0 par le nombre d'arcs sortants de ce niveau. Ainsi, nous obtenons c(b) = c(c) = 5/2 = 3. En poursuivant de la même manière, nous trouvons la capacité des sommets du niveau 3, c(e) = c(h) = 2/3.5 = 1, et la capacité du sommet f du niveau 4, c(f) = 1/1 = 1.

Après avoir assigné les capacités, nous passons à la deuxième étape, c'est-à-dire la conversion du graphe. En suivant la même procédure que l'exemple précédent, nous obtenons le graphe converti représenté dans la figure suivante.

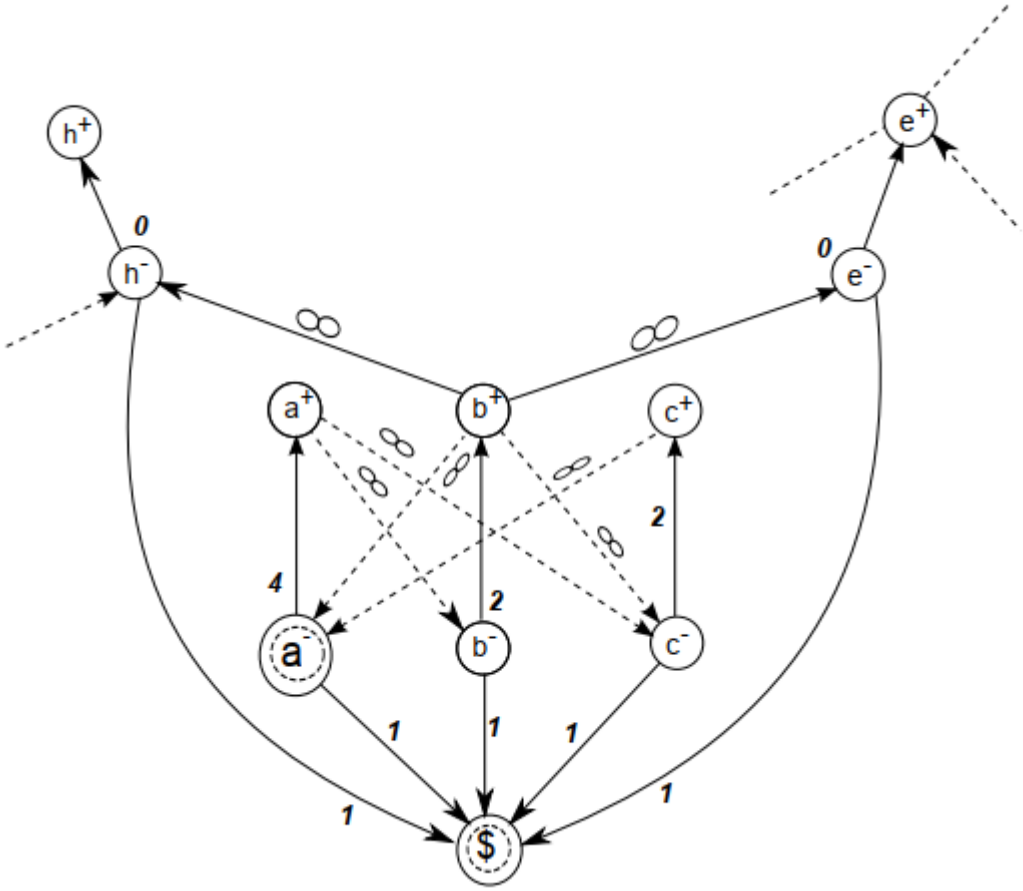


Figure3.10 Graphe de confiance après conversion [37]

Enfin, en appliquant l'algorithme de calcul du flot maximum de Ford-Fulkerson du seednode*a* au supesink $, l'algorithme renverra les sommets *b, c, e* et *h* comme pairs de fiables.

3.3.1.6 Analyse de l'algorithme

Avantages : L'algorithme Advogato présente plusieurs avantages, tels que sa simplicité et sa faible complexité (O(V * E)), qui correspond à la complexité de l'algorithme de recherche du flot maximum de Ford-Fulkerson. À titre de rappel, la complexité de ce dernier est de l'ordre de O(f * E) [38], où E représente le nombre d'arcs du graphe et f est le flot maximum. Dans notre cas, f représente simplement le nombre de sommets du graphe, qui est équivalent à V.

Inconvénients : Cependant, cet algorithme présente quelques inconvénients. Tout d'abord, il ne prend pas en compte la force des liens (le degré de confiance entre les utilisateurs) dans les calculs, se basant uniquement sur les plus courts chemins. De plus, l'assignation des capacités pose un autre inconvénient. En effet, la confiance est normalisée selon la capacité [15]. Prenons l'exemple d'une source ayant une capacité de 8 et considérons deux cas : le premier cas où la source a 2 voisins, et le deuxième cas où la source a 4 voisins. Dans le premier cas, les voisins auront une capacité de 4, tandis que dans le deuxième cas, ils auront une capacité de 2. Dans cette situation, la capacité est propagée différemment des valeurs de confiance attendues.

3.3.2 TidalTrust

3.3.2.1 Description de l'algorithme

L'algorithme TidalTrust, introduit par Golbeck en 2005[15]. Il est utilisé dans le réseau social FilmTrust, qui propose un système de recommandation de films avec une grande quantité d'utilisateurs. Dans ce système, chaque utilisateur peut évaluer sa confiance envers d'autres utilisateurs en utilisant des valeurs discrètes dans l'intervalle [0..10][39]. L'idée de représenter la confiance par des valeurs discrètes est considérée comme plus intuitive selon Golbeck. Les relations entre les utilisateurs sont représentées par un graphe orienté G = (V, E), où chaque arc est associé à une valeur de confiance entre 0 et 10. Une valeur de 10 représente une confiance totale, tandis que 0 indique une absence d'information. Avant de commencer, chaque utilisateur attribue des valeurs de confiance à un ensemble de personnes, ses voisins directs en qui il a confiance.

Pour évaluer la confiance d'un sommet source s envers un sommet cible t, la source s interroge tous ses voisins pour connaître leurs valeurs de confiance concernant t. Si un voisin a une relation directe avec la cible, il renvoie sa valeur de confiance. Sinon, il interroge à son tour

ses propres voisins pour obtenir leurs valeurs de confiance concernant t. Ce processus est répété par chaque sommet jusqu'à atteindre un sommet en relation directe avec la cible t.Une fois qu'un chemin entre la source s et la cible t est trouvé, la profondeur maximale est fixée à la profondeur du chemin trouvé. Étant donné que la recherche est effectuée en utilisant la méthode Breadth First Search, le premier chemin trouvé est celui de la profondeur minimale. La recherche se poursuit pour trouver tous les chemins de profondeur minimale.

Pendant la recherche de la cible, chaque sommet conserve la force maximale des chemins de profondeur minimale qui le relient à la source s, ainsi que la profondeur actuelle. La force d'un chemin est égale à la valeur de confiance minimale entre les utilisateurs le long de ce chemin. Les sommets voisins de la source s conservent les valeurs de confiance que la source a envers eux. Une fois la recherche terminée, le seuil de confiance (variable max dans la formule 3.1) est déterminé en prenant la force maximale des chemins de confiance menant à la cible t. Ensuite, les valeurs de confiance des sommets en relation directe avec la cible t sont renvoyées en utilisant les chemins inverses jusqu'à atteindre la source s. Pendant le cheminement inverse, si un sommet a une valeur supérieure ou égale au seuil et reçoit une ou plusieurs valeurs de confiance à travers des arcs ayant des poids supérieurs ou égaux au seuil, il combine ces valeurs de confiance en utilisant la formule (3.2). Les valeurs combinées sont ensuite renvoyées par les chemins inverses. Ce processus se poursuit jusqu'à ce que la source s soit atteinte. Une fois que la source s reçoit les valeurs de confiance à travers des arcs ayant un poids supérieur ou égal au seuil, elle applique la même formule (3.2) pour calculer sa confiance envers la cible t.

$$tst = \frac{\sum_{j \in adj(s) \mid tsj > max} tsjtjt}{\sum_{j \in adj(s) \mid tsj > max} tsj} \qquad (3.1)$$

Où
adj(*s*) : sommets voisins de la source *s* dans le graphe ;
max: Seuil de confiance ; valeur de confiance que la source *s* a
tsj: envers le sommet *j*; valeur de confiance que le sommet *j* a
tjt: envers la cible *t*.

Le pseudo-code de l'algorithme *TidalTrsut*est donnée par l'algorithme suivant.

Algorithm 3 Algorithme*TidalTrust*[15]

TidalTrust(*source, sink*)
for *each n in G* **do**
color(*n*) = *white*
end for
q = empty, *push*(*q, source*), *depth* = 1, *maxdepth= infinity*
while *q not empty and depth* ≤ *maxdepth***do**
n = pop(*q*), *push*(*d*(*depth*), *n*)
if*sink in adj*(*source*)**then**
cached rating(*n, sink*) = *rating*(*n, sink*)
maxdepth= depth
flow = min(*path flow*(*n*), *rating*(*n, sink*))
path flow(*sink*) = *max*(*path flow*(*sink*), *flow*) *push*(*children*(*n*), *sink*)
else
for *each n*2 *in adj*(*n*) **do**
if *color*(*n*2) = *gray* **then**
color(*n*2) = *gray*, *push*(*temp q, n*2)
end if
if *n*2 *in temp q* **then**
flow = min(*path flow*(*n*), *rating*(*n, n*2))
path flow(*n*2) = *max*(*path flow*(*n*2), *flow*)
push(*children*(*n*), *n*2)
end if
end for
end if
if *q empty* **then**
q = temp q, depth = depth + 1, *temp q = empty*
end if
end while
max = path flow(*sink*), *depth = depth* –1
while *depth* >0 **do**
while *d*(*depth*) *not empty* **do**
n = pop(*d*(*depth*))
for *each n*2 *in children*(*n*) **do**
if *rating*(*n, n*2) ≥ *max and cached rating*(*n*2, *sink*) = 0 **then**
numerator = numerator + rating(*n, n*2) ∗ *cached rating*(*n*2, *sink*)
denominator = denominator + rating(*n, n*2)
end if
end for
if *denominator* >0 **then**
$cached_r ating$(*n, sink*) = *numerator/denominator*
else
$cached_r ating$(*n, sink*) = -1
end if
end while
depth= depth–1
endwhile

3.3.2.2 Exemple

Considérons le graphe de la figure suivante.

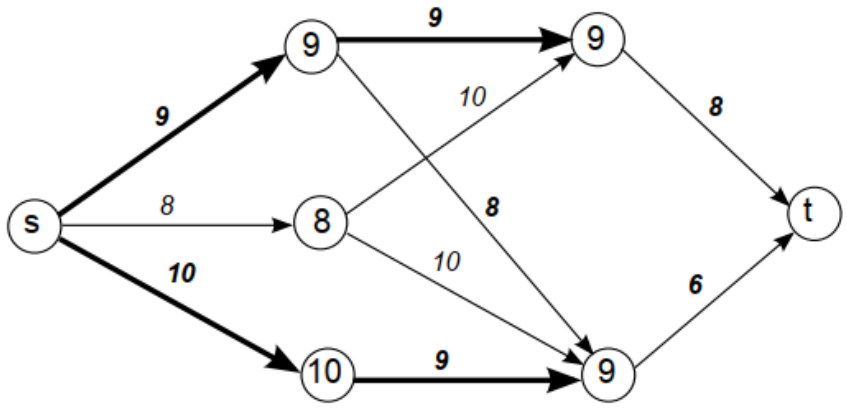


Figure3.11 TidalTrust : Exemple de calcul de la confiance [15]

Dans ce contexte, chaque arc est associé à un label représentant la confiance qu'un sommet initial a envers un autre sommet final. De plus, chaque sommet possède un label représentant la force maximale des chemins menant de la source s à ce sommet. La profondeur minimale est de 3, correspondant à la longueur du plus court chemin, et le seuil maximum est de 9, représentant la force maximale des plus courts chemins. Les sommets adjacents à la source s prennent les valeurs de confiance que la source a envers eux (9, 8 et 10), tandis que les sommets adjacents à la cible t prennent la force maximale des chemins menant à eux (9). Les arcs en gras indiquent les chemins qui seront utilisés dans les calculs lors du chemin inverse, car ils ont une profondeur minimale (3) et une force égale au seuil maximum de 9. En utilisant la formule (3.1), nous obtenons :

$tst = 9 \times 8 + 10 / (9 + 10 \times 6) = 6.95.$

3.3.2.3 Analyse de l'algorithme

Avantages : Les points forts de l'algorithme de Godbeck sont sa simplicité et sa faible complexité qui est égale à $O(n+m)$ [40]. L'algorithme *TidalTrust* est connu comme l'algorithme le plus célèbre et fortement cité pour le calcul de la confiance [41] ; c'est pour cette raison que cet algorithme est choisi, souvent, comme une référence pour la comparaison des résultats.

Inconvénients :L'algorithme TidalTrust présente deux inconvénients. Premièrement, la restriction imposée sur les plus courts chemins peut parfois entraîner l'omission de certaines

informations utiles. Deuxièmement, un problème majeur se pose lorsqu'il y a un chemin unique entre la source et la cible. Supposons qu'il existe une longue chaîne unique où chaque sommet a une valeur de confiance égale à 9 envers son voisin [40]. Supposons également qu'il existe une autre chaîne de même distance où chaque sommet a une valeur de confiance égale à 1 envers son voisin, à l'exception de l'avant-dernier sommet qui a une valeur de confiance égale à 9 envers la cible. Avec l'algorithme TidalTrust, dans les deux cas, la valeur de confiance calculée entre s et t est égale à 9. Cependant, il est évident que la valeur de confiance calculée dans le premier cas devrait être supérieure à celle calculée dans le deuxième cas.

3.3.3 MoleTrust

3.3.3.1 Description de l'algorithme

L'algorithme MoleTrust est proposé par Massa [42]. Il est utilisé dans le site Epinions.com, qui est un réseau social dédié à la vente de produits commerciaux, où chaque utilisateur peut exprimer sa confiance envers d'autres utilisateurs en utilisant des valeurs continues appartenant à l'intervalle [0,1]. Cet algorithme permet de calculer la confiance qu'un utilisateur (la source) a envers un autre utilisateur (la cible) en parcourant le graphe de confiance G, généré à partir du réseau social, depuis la source jusqu'à la cible et en propageant la confiance le long des arcs [43]. La confiance d'une cible donnée dépend des confiances que les autres utilisateurs ont envers elle et des confiances de ces derniers. Principalement, le calcul de la confiance entre une source s et une cible t se fait en deux étapes. La première étape consiste à enlever les circuits du graphe de confiance G pour obtenir un graphe orienté sans circuit, ayant la source s comme racine. La deuxième étape consiste à la propagation de la confiance depuis la source s jusqu'au sommet cible t dans le but de calculer la confiance que la source a envers la cible t, ainsi que celle de chaque sommet visité. Le calcul de la confiance d'un sommet visité est donné par l'équation suivante...

$$\mathrm{b(xj)} = \frac{\sum_{k \in p(j)} b(xk) T\,(xk, xj)}{\sum_{k \in p(j)} b(xk)}, \qquad (3.2)$$

Où :

$b(xj)$: Confiance du sommet xj;

$T\,(xk, xj)$: valeur de confiance qu'un sommet xka envers le sommet xj;

$p(j)$: L'ensemble des prédécesseurs du sommet xj.

MoleTrust applique deux restrictions lors du calcul de la confiance entre la source s et la cible t. Premièrement, seuls les sommets ayant des valeurs de confiance supérieures ou égales à un certain seuil T sont pris en considération. En général, le seuil de confiance T est fixé à 0,5. Deuxièmement, la longueur maximale (profondeur) d'un chemin de confiance entre la source s et la cible t est fixée à 5 : seules les valeurs de confiance des sommets situés à une profondeur inférieure ou égale à 5 peuvent être calculées. Le pseudo-code de l'algorithme MoleTrust est donné par l'algorithme suivant :

Algorithm 4 Algorithme*MoleTrsut*[44]

Input Trust network G, Source of measure u_x, Trust degree threshold TT, Max depth $maxdep$
Output Trust degrees and corresponding depth of all the reachable nodes of $maxdep$ REC
$set\ i \leftarrow 1,\ REC \leftarrow \{\}$
$\forall u_y \in V_x : set\ N_1 \leftarrow N_1 \cup \{u_y\},\ REC \leftarrow REC \cup \{(t_{xy}, depth)\}$
while $N_i \neq \{\}$ and $i < maxdep$ **do**
$set\ TEMP \leftarrow \{\}$
for all $u_w \in N_i$ **do**
if $t_{xw} > TT$ **then**
for all $u_z \in G_w$ **do**
if $(t_{xz}, depth) \notin REC$ **then**
if $(t_{xz}, w_{xz}) \in TEMP$ **then**
set $TEMP(z) \leftarrow (t_{xz} + t_{xw}.t_{wz},\ w_{xz} + t_{xw})$
else
set $TEMP(z) \leftarrow (t_{xw}.t_{wz},\ t_{xw})$
end if
end if
end for
end if
end for
$\forall (t_{xz}, w_{xz}) \in TEMP : set\ REC(z) \leftarrow (t_{xz}/w_{xz},\ i)$
$i \leftarrow i + 1$
end while
return REC

3.3.3.2 Exemple

Considérons l'exemple de la figure suivante.

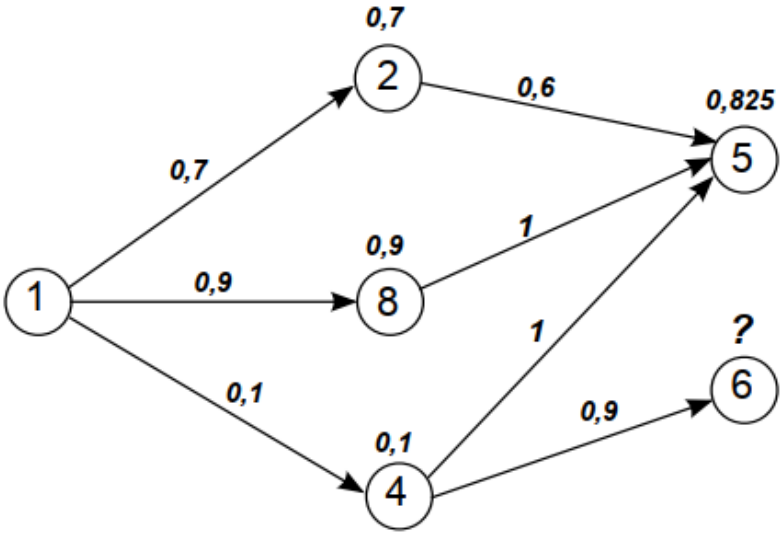


Figure3.12MoleTrust : Exemple de calcul de la confiance [43]

La confiance attribuée à chaque sommet adjacent à la source est équivalente à la confiance que la source accorde à ce sommet. Lors du calcul de la confiance de la source (sommet 1) envers la cible (sommet 5), l'algorithme MoleTrust ne prend en compte que les sommets 2 et 3, tandis que le sommet 4 est ignoré car sa confiance est inférieure au seuil défini (par exemple, 0.5). Ainsi, la confiance calculée par la source à l'égard de la cible est évaluée à$\frac{(0.7*0.6+0.9*0.1)}{(0.7+0.9)} = 0.825.$

3.3.3.3 Analyse de l'algorithme

Avantages : Les points forts de cet algorithme sont sa simplicité et sa faible complexité qui est égale à *O(n+m)* [45].

Inconvénients : Cependant, cet algorithme présente plusieurs inconvénients. Premièrement, la restriction appliquée aux plus courts chemins peut entraîner l'ignorance de certaines informations utiles dans certains cas. Deuxièmement, la limitation de la longueur des chemins peut avoir un impact sur la couverture de l'algorithme, empêchant le calcul de la confiance pour les cibles situées à une profondeur supérieure à la profondeur maximale définie. Troisièmement, un problème majeur survient lorsque seul un chemin unique relie la source à la cible. Supposons qu'il existe une longue chaîne unique entre la source et la cible, où chaque valeur de confiance entre deux sommets est égale à 1, sauf pour la valeur de confiance du dernier sommet envers la cible, qui est de 9. Dans cette situation, la valeur de confiance calculée

entre la source et la cible serait de 9, ce qui est anormal et illogique, car cette valeur de confiance est beaucoup plus élevée que celle attendue.

3.4 Conclusion

Au cours de ce chapitre, nous avons examiné différents algorithmes de calcul de confiance qui ont été proposés dans la littérature. Nous avons commencé par introduire les concepts de base de la complexité algorithmique, tels que la définition et le calcul de la complexité, les classes de complexité et les classes de problèmes informatiques.

Ensuite, nous avons présenté plusieurs algorithmes de calcul de confiance en suivant une structure cohérente. Pour chaque algorithme, nous avons fourni une description détaillée, illustré son application à l'aide d'un exemple concret, et discuté de ses avantages et de ses limites.

Après avoir étudié ces algorithmes existants, le prochain chapitre sera consacré à la présentation de notre propre algorithme de calcul de confiance, nommé FASTTRUST

Chapitre 04

Chapitre 04 : Proposition d'un nouveau algorithme de confiance

4.1 Introduction

Le calcul de la confiance est une tâche essentielle dans de nombreux domaines : les réseaux sociaux, les systèmes de recommandation, la gestion des informations et la prise de décision. L'approche la plus utilisée est la méthode des graphes, qui consiste à modéliser les relations entre les entités sous forme de graphes et à calculer la confiance en se basant sur ces relations. Néanmoins, cette méthode présente certaines contraintes qui peuvent affecter sa performance et sa précision.

Dans ce chapitre, nous citons quelques inconvénients de la méthode des graphes pour le calcul de la confiance, notamment sa complexité et sa dépendance aux informations disponibles dans le graphe. Nous présenterons également l'approche alternative proposée « FastTrust » qui exploite uniquement les voisins des entités concrernés par le calcul de la confiance au lieu de tout les graphes. Nous présentons le détail de chaque étape de cette méthode avec des exemples pour une bonne compréhensions.

4.2 Inconvénients des méthodes basées sur les graphes

La méthode des graphes pour le calcul de la confiance présente certains inconvénients. Tout d'abord, la complexité peut devenir un défi, surtout lorsque les graphes deviennent grands et complexes. Le calcul de la confiance peut nécessiter des ressources de calcul importantes, ce qui engendre un temp d'exécution plus long et rend la scalabilité problématique car la performance de l'algorithme peut diminuer avec l'augmentation du nombre d'entités et de relations dans le graphe. Les méthodes basées sur les graphes dépendent fortement des informations disponibles dans le graphe, ce qui signifie que des données manquantes, incorrectes ou non fiables peuvent affecter la précision des calculs de confiance. Alors il est important de prendre en compte ces inconvénients lors de l'utilisation de la méthode basées sur les graphes pour le calcul de la confiance.

4.2.1 Temps de calcul

Le temps de calcul est l'un des inconvénients majeurs dee methodes basées sur les graphes pour le calcul de la confiance. En raison de la complexité associée à l'analyse de graphes, le temps nécessaire pour effectuer les calculs peut être considérable, surtout pour les graphes de grande taille. Cette augmentation du temps de calcul peut rendre l'utilisation de la méthode des graphes moins pratique, en particulier lorsqu'il est nécessaire de traiter des

volumes importants de données en temps réel ou de manière très réactive. La gestion efficace du temps de calcul devient donc un défi critique pour garantir des performances acceptables de l'algorithme de confiance basé sur les graphes. Les chercheurs travaillent continuellement sur des techniques d'optimisation et des algorithmes plus efficaces pour atténuer cet inconvénient et améliorer les performances de la méthode des graphes dans le calcul de la confiance.

Le temps de calcul est un paramèttre important à prendre en compte lors de l'utilisation de la méthode des graphes pour le calcul de la confiance. Dans ce qui suit, on cite quelques points à considérer concernant le temps de calcul :

- **Complexité du calcul :** Le temps de calcul dans la méthode des graphes dépend de la complexité de l'algorithme utilisé. Certains algorithmes peuvent avoir une complexité élevée, ce qui signifie que le temps nécessaire pour effectuer les calculs augmente considérablement avec la taille du graphe.

- **Taille du graphe :** Le temps de calcul est directement influencé par la taille du graphe. Plus le graphe est grand, avec un grand nombre de nœuds et d'arêtes, plus le temps de calcul sera long. Le traitement de graphes de grande taille peut nécessiter des ressources informatiques plus importantes pour effectuer les calculs dans un délai raisonnable.

- **Connectivité du graphe :** La connectivité du graphe peut également affecter le temps de calcul. Si le graphe est fortement connecté, cela peut entraîner des calculs plus longs. La propagation de la confiance à travers un graphe densément connecté peut nécessiter davantage d'itérations et de calculs.

- **Optimisation des algorithmes :** Des techniques d'optimisation peuvent être appliquées pour réduire le temps de calcul. Cela peut inclure l'utilisation d'algorithmes plus efficaces, la parallélisation des calculs sur plusieurs processeurs ou l'utilisation de techniques de réduction de la dimensionnalité pour traiter des graphes de grande taille de manière plus efficiente.

- **Trade-off entre précision et temps de calcul** : Il est important de noter que le temps de calcul peut être un compromis avec la précision des résultats. Parfois, des approximations ou des simplifications sont utilisées pour réduire le temps de calcul, mais cela peut entraîner une diminution de la précision des résultats obtenus.

En conclusion, le temps de calcul est un inconvénient potentiel de la méthode des graphes dans le calcul de la confiance, en raison de la complexité et de la taille des graphes à traiter. Cependant, des techniques d'optimisation et des algorithmes efficaces peuvent contribuer à atténuer cet inconvénient et à améliorer les performances de la méthode des graphes.

4.2.2 Perte d'information

La perte d'information est un concept général qui se produit dans de nombreux domaines et processus où des données ou des informations sont manipulées, analysées ou transmises. Elle fait référence à la diminution de la quantité ou de la qualité des informations lorsqu'elles sont traitées ou converties d'une forme à une autre.

Parmi les inconvénients de la méthode des graphes, on trouve la perte d'information. Dans ce qui suit quelques points à considérer concernant ce problème :

- **Agrégation des informations :** Dans la méthode des graphes, il est courant d'agréger les informations des nœuds voisins pour calculer la confiance d'un nœud donné. Cette agrégation peut entraîner une perte d'information, car les détails spécifiques de chaque nœud voisin peuvent ne pas être pris en compte ; ceci qui peut conduire à une perte de précision dans les estimations de confiance.

- **Simplification des relations :** Dans certains situations, la méthode des graphes peut nécessiter la simplification des relations entre les nœuds. Par exemple, des mesures de similarité peuvent être utilisées pour déterminer la confiance entre deux nœuds, ce qui peut conduire à une approximation des relations réelles. Cette simplification peut entraîner une perte d'information, car des nuances et des variations subtiles peuvent être ignorées dans le processus de simplification.

- **Ignorance des informations contextuelles** : La méthode des graphes peut ne pas prendre en compte les informations contextuelles lors du calcul de la confiance. Par exemple, des informations temporelles ou des informations spécifiques au domaine peuvent ne pas être prises en considération dans les calculs. Cela peut conduire à une perte d'information importante, car des facteurs importants pourraient être omis ou négligés, ce qui pourrait affecter la précision des estimations de confiance.

Erreurs de mesure ou de collecte de données : La méthode des graphes est sensible aux erreurs de mesure ou de collecte de données. Si les mesures ou les données utilisées pour

construire le graphe sont erronées, incomplètes ou biaisées, cela peut engendrer une perte d'information significative. Les erreurs ou les biais dans les données peuvent se propager à travers le graphe et influencer les estimations de confiance de manière négative.

En conclusion, la perte d'information est un inconvénient potentiel de la méthode des graphes dans le calcul de la confiance. Elle peut résulter de l'agrégation des informations, de la simplification des relations, de l'ignorance des informations contextuelles et des erreurs de mesure ou de collecte de données. Il est important de prendre en compte ces limitations et d'évaluer attentivement la précision des résultats obtenus à partir de la méthode des graphes.

4.3 Proposition d'une nouvelle méthode pour le calcul de la confiance

La méthode proposée (FastTrust) dans ce chapitre est basée sur l'utilisation d'un graphe de confiance pour calculer les valeurs de confiance entre les entités non connectées directement : les personnes. Ce graphe contient des valeurs de confiance directs entre des personnes qui sont calculées à partir d'un ensemble de données spécifique.

4.3.1 Principe de la méthode

Le principe de la méthode proposée est de calculer la confiance entre deux sommets, la source et la cible, dans une graphe, en se basant uniquement sur leurs prédécesseurs et leurs successeurs de la cible et de la sources respectivement, sans tenir compte des autres reltations et sommets du graphe.

La confiance entre deux sommets la source s et la cible c est alors calculée en utilisant les valeurs de confiance de la source s vers ses successeurs : conf(s,i), et les valeurs de confiance des prédécesseurs de la cible c : conf(j,c).

Notre méthode est basée sur les graphes et explore uniquement les voisins de la source et de la cible à savoir les successeurs de la source et les prédécesseurs de la cible.

-L'algorithme FastTrust : calcule de la confiance entre deux sommets.

-**Entrée** : un graphe de confiance G, une source s et une cible c.

- **Sortie** : la confiance de la source s envers la cible c : conf(s,c) .

- le calcule de la confiance :

- **conf_s** : Si le sommet source s a des successeurs, alors la confiance conf_s est égale à la somme des confiances de s envers ses successeurs divisé par le nombre de ces derniers n, sinon cette confiance est égale à zéro :

$$(\text{conf\_s}) = \sum_{i=1}^{n} \frac{conf(s,i)}{n}$$

Avec i est un successeur de source s.

- **conf_c :** Si le sommet c a des prédécesseurs, alors la confiance conf_c est égale à la somme des confiances des prédécesseurs de c divisé par le nombre de ces derniers m, sinon la confiance du sommet c est égale à zéro :

$$(\text{conf\_c}) = \sum_{j=1}^{n} \frac{conf(j,c)}{n}$$

avec j est un prédécesseur de c.

- **Conf (s, c)** : la confiance de s vers c : $conf(s\_c) = \frac{(\text{conf\_s})+(\text{conf c})}{2}$

4.3.2 Les étapes de FastTrust

Pour calculer la confiance d'un sommet source s vers un sommet un sommet cible c FastTrust effectue les étapes suivants :

a) Lecture du graph à partir d'un fichier texte scores (dataset)

Le fichier (dataset) score contient les valeurs de confiance entre chaque deux sommets i, j de la forme : i, j, val.

b) Parcours des voisins de la source set de la cible c pour le calcul de a confiance de s vers c.

c) Sauvegarde des valeurs calculées afin d'évaluer les performances de FastTrust on les comparant avec d'autres algorithmes dédiés pour la calcul de la confiance.

d) Exportation des résultats dans un fichier afin de les visionner.

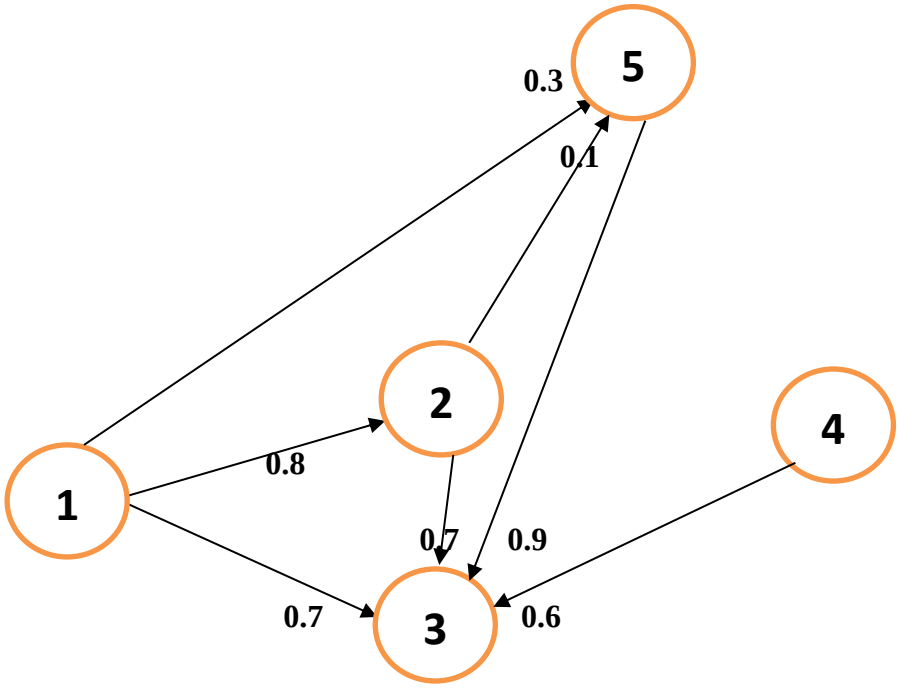
4.4 Exemple complet

Format de DATASET utilisé par FastTrust dans l'exemple.

Fichier score :

1	**2**	**0.8**
1	**3**	**0.7**
1	**5**	**0.3**
2	**3**	**0.7**
2	**5**	**0.1**
4	**3**	**0.6**
5	**3**	**0.9**

On va d'abord lire le graphe G à partir du fichier DATASET score pour que l'algorithme FastTrust peut l'utiliser.



Pour chaque arc (i, j) entre deux sommets i et j, on retire cet arce du graphe, ensuite on utilise l'algorithme FastTrust pour calculer les nouvelle valeur de confiance entre i et j conf(i, j).

i	j	Valeur réelle	valeur calculée conf(i, j)
1	2	0.8	0.7
1	3	0.7	0.6625
1	5	0.3	0.4
2	3	0.7	0.5625
2	5	0.1	0.3
4	3	0.6	0.6625
5	3	0.9	0.8125

On remarque dans cet exemple que les valeurs de confiance calculées (prédites) sont très proches des valeurs réelles, ce que explique l'efficacité de l'algorithme proposé. Une comparaisons complète fera l'objet du prochain chapitre.

4.5 Complixite

La complexité de calcul dans un graphe dépend du nombre de sommets et le nombre d'arcs dans le graphe. Dans notre cas, nous explorons uniquement les successeurs et les prédécesseurs de deux sommets à savoir la source et la cible respectivement. En conséquence le nombre d'opérations effectués par l'algorithme proposés FastTrust égale aux nombres de successeurs de la source s et de prédécesseurs de la cible c, et ceci, conduit à une complexité linéaire O(n).

L'algorithme FastTrust comporte deux fonctions principales : "lire_graphe_de_fichier" et "conf". Voici la complexité de chaque fonction :

Fonction "lire_graphe_de_fichier":

La fonction lit le graphe à partir d'un fichier text score.

Fonction "conf":

La fonction calcule la confiance entre deux sommets dans le graphe.

Elle récupère les successeurs de sommet i et les prédécesseurs du sommet j. Ensuite, il calcule la moyenne des confiances des successeurs du sommet i et des prédécesseurs du sommet j.

La complexité de cette fonction dépend du nombre de successeurs et de prédécesseurs, respectivement.

Dans le pire des cas, si chaque sommet a un successeur ou prédécesseurs, la complexité serait de l'ordre de O(n+m)=O(z) avec n et m et le nombre de successeurs et de prédécesseurs s et c respectivement.

La complexité réelle dépendra de la structure du graphe et du nombre de successeurs et de prédécesseurs.

En résumé, la complexité de l'algorithme proposé, FastTrust, dépend du nombre de successeurs n et de prédécesseurs m de la source et de la cible respectivement. Dans le pire des cas, la complexité est linéaire O(n+m)=O(z) . Ce qui permet une grande scalabilité de l'algorithme proposé FastTrust.

4.6 Conclusion

Le calcul de la confiance est une problématique cruciale dans de nombreux domaines, et il existe différentes approches pour aborder cette tâche. Nous avons examiné la méthode des graphes, largement utilisée, mais qui présente des limitations en termes de complexité computationnelle et de dépendance aux informations disponibles.

Nous avons également exploré un nouveau algorithme "FastTrust", qui utilise un graphe de confiance pour estimer les valeurs de confiance entre les entités de ce graph. Cette méthode offre des avantages tels qu'une meilleure scalabilité, une résilience accrue aux attaques et une flexibilité quant aux informations nécessaires pour calculer la confiance.

Cependant, il convient de noter que chaque approche présente ses propres forces et faiblesses, et il est important de les évaluer en fonction des besoins spécifiques du domaine d'application. De plus, la recherche dans le domaine du calcul de la confiance est en constante évolution, et de nouvelles approches et techniques continuent d'émerger.

Chapitre 04 : Proposition d'un nouveau algorithme de confiance

En fin de compte, en comprenant les limites de la méthode des graphes en terme de temps d'exécution, nous avons proposé un nouveau algorithme plus efficace "FastTrust pour évaluer les relations et la fiabilité entre les entités, cela contribuera à améliorer la confiance dans les systèmes de décisions basées sur ces évaluations, ouvrant ainsi la voie à de nouvelles opportunités dans divers domaines, tels que l'internet des objets, les réseaux pair-to-pair,…

Chapitre 05

Chapitre 05 : Implémentation et évaluation

5.1 Introduction

Après avoir vu le fonctionnement de l'algorithme proposé "*fastTrust*" dans le Chapitre 4, dans ce chapitre nous allons voir tout ce qui concerne l'implémentation et l′évaluation de l'algorithme proposé.

Ce chapitre est organisé comme suit : premièrement, nous décrivons le langage de programmation et l'environnement de développement utilisés pour l'implémentation de l'algorithme proposé, à savoir le langage python. Ensuite, nous présentons l'architecture logicielle utilisée pour l'implémentation de FastTrust. En dernier, nous allons évaluer les performances de l'algorithme proposé "*fastTrust*" en le comparant à deux algorithmes *MoleTrust*et *TidalTrust*en en se basant sur le jeu de données (Dataset) "*Residence hall*" en s'intéressant à quatre mesures à savoir : l'Erreur Absolue Moyenne (EAM), l'Erreur Quadratique Moyenne (EQM), la Précision (Prc), et la Couverture (Cvr).

5.2 Implémentation

Nous avons choisi de mettre en œuvre notre proposition en utilisant le langage de programmation Python, un langage populaire et largement utilisé en programmation. Pour faciliter le développement, nous avons opté pour l'environnement de développement Anaconda avec Spyder. Anaconda est une distribution Python qui inclut de nombreuses bibliothèques et outils essentiels pour le développement scientifique et de données. Spyder est l'IDE (Integrated DevelopmentEnvironment) inclus dans Anaconda, offrant un éditeur de code convivial, un débogueur intégré, un explorateur de variables et d'autres fonctionnalités utiles pour accélérer le processus de développement.

5.2.1 Langage de programmation

Python est un langage de programmation interprété, polyvalent et convivial. Il a été créé par Guido van Rossum dans les années 1990 et tire son nom de la série télévisée britannique "Monty Python'sFlying Circus". Initialement conçu comme un langage de script, Python est devenu l'un des langages les plus populaires dans le domaine du développement logiciel, de l'analyse de données, de l'apprentissage automatique et de l'intelligence artificielle [46].

L'une des principales caractéristiques de Python est sa syntaxe claire et lisible, ce qui en fait un langage très apprécié des débutants. Il favorise également une approche modulaire grâce à l'utilisation de modules et de packages, ce qui facilite la réutilisation du code et la collaboration entre développeurs.

Python bénéficie d'une large communauté de développeurs qui contribuent activement à son développement et créent une multitude de bibliothèques et de frameworks pour différentes applications. Cette vaste bibliothèque standard et l'écosystème Python riche en ressources en font un choix idéal pour divers projets.

Parmi les avantages de Python, on compte sa portabilité, car il est compatible avec de nombreux systèmes d'exploitation. De plus, sa syntaxe simple et expressive permet de rédiger du code concis et lisible, ce qui facilite la maintenance et le débogage. Python offre également une grande flexibilité et une facilité d'intégration avec d'autres langages, ce qui permet d'utiliser des bibliothèques écrites dans d'autres langages.

En ce qui concerne les inconvénients, Python peut être relativement lent par rapport à des langages tels que C++ ou Java en raison de son interprétation. Cependant, cela peut être atténué en utilisant des bibliothèques externes écrites en langages compilés. De plus, Python peut consommer plus de ressources système que certains autres langages, bien que cela soit généralement négligeable pour la plupart des applications.

Dans l'ensemble, Python est un langage puissant, polyvalent et largement utilisé qui convient à une variété de domaines d'application. Sa simplicité, sa large communauté de développeurs et son écosystème dynamique en font un choix populaire pour les projets de développement de logiciels, d'analyse de données, d'apprentissage automatique et d'intelligence artificielle.

5.2.2 Environnement de développement

L'environnement de développement intégré (IDE) Anaconda avec Spyder est l'outil que nous utilisons pour le développement de notre système. Anaconda est une distribution Python populaire qui est largement utilisée dans le domaine de l'analyse de données, de l'apprentissage automatique et de l'intelligence artificielle. Il est conçu pour faciliter l'installation et la gestion de packages Python, ainsi que pour offrir un ensemble complet d'outils et de bibliothèques préinstallés pour le développement [47].

Spyder, quant à lui, est l'IDE inclus dans l'environnement Anaconda. Il est spécialement conçu pour les tâches liées à l'analyse de données et à la programmation scientifique en utilisant Python. Spyder offre une interface conviviale avec des fonctionnalités avancées telles que l'éditeur de code, le débogueur, l'explorateur de variables et l'intégration de la documentation.

Il permet également une exécution interactive du code et prend en charge les graphiques en direct, ce qui est particulièrement utile lors de l'exploration de données [48].

L'utilisation d'Anaconda avec Spyder présente plusieurs avantages. Tout d'abord, Anaconda simplifie l'installation et la gestion des packages Python, offrant ainsi un environnement cohérent et fiable pour le développement. Il inclut également de nombreuses bibliothèques populaires telles que NumPy, Pandas, Matplotlib et Scikit-learn, ce qui facilite le traitement des données et l'implémentation d'algorithmes d'apprentissage automatique.

De plus, Spyder est conçu spécifiquement pour les tâches liées à l'analyse de données, offrant une interface intuitive et conviviale adaptée aux scientifiques des données. Son intégration étroite avec les bibliothèques de données couramment utilisées permet un flux de travail fluide et efficace.

En résumé, l'utilisation de l'environnement de développement Anaconda avec Spyder offre un ensemble d'outils puissants et pratiques pour le développement de notre système. Il facilite l'installation des packages, fournit des fonctionnalités avancées pour le développement et l'analyse de données, et offre un environnement de développement cohérent et fiable pour nos besoins spécifiques.

5.3 Evaluation

Pour l'évaluation de notre approche, nous implémentons d'abord l'algorithme proposé *FastTrust* ainsi que les deux autres algorithmes *MoleTrsut* et *TidalTrust*, ensuite nous comparons leurs résultats en utilisant un jeu de données de test réel, à savoir le jeu de données "*Residence hall*" [49]. La comparaison se fera selon les quatre mesures suivantes : l'Erreur Absolue Moyenne (EAM), l'Erreur Quadratique Moyenne (EQM), la Précision (Prc), et la Couverture (Cvr).

5.3.1 Jeu de données (Data Set)

Pour l'évaluation de l'algorithme proposé *FastTrust*, nous utilisons un jeu de données (Dataset) de test réel contenant les valeurs de confiance explicites entre les utilisateurs. A cet effet, nous utilisons le jeu de données "*Residence hall*" qui contient les relations de confiance entre des résidents vivant dans une résidence du campus de l'université nationale australienne (ANU) [38]. Dans ce jeu de données, qui est anonymisé, chaque personne assigne des valeurs

de confiance à d'autres personnes en utilisant des valeurs entières de 1 à 5.La distribution des valeurs de confiance est donnée par le Tableau 5.1 suivant

Tableau 5.2 Distribution des valeurs de confiance dans "Residence hall"

Val. de conf.	**1**	**2**	**3**	**4**	**5**	**Total**	
Effectif	38	110	1624	602	298	2672	
Pourcentage	1.4 %	4.12 %	60.78 %	22.53 %	11.15 %	100 %	
Cumulé	1.42 %	5.54 %	66.32 %	88.85 %	100 %	-	

On voit sur ce tableau que la distribution n'est pas uniforme, et que plus de la moitié des valeurs de confiance sont égales à 3.

5.3.2 Démarche et mesures d'évaluation

L'objectif des algorithmes de confiance est de prédire les valeurs de confiance entre utilisateurs le plus précisément possible, et ce pour le maximum possible d'utilisateurs. Afin d'évaluer l'algorithme proposé fast*Trust*, nous comparons ses résultats avec ceux des deux algorithmes *MoleTrust*et *TidalTrust*en utilisant la méthode "*leave-one-out*" qui est largement utilisée dans les systèmes de recommandation. Cette dernière consiste à donner à l'algorithme toutes les valeurs de confiance sauf une, ensuite l'exécuter afin de prédire (calculer) la valeur manquante à partir des autres valeurs, et ainsi de suite pour chaque valeur de confiance. A partir de ces prédictions, nous comparons les trois algorithmes en se basant sur les quatre mesures de précisions utilisées dans le domaine des systèmes de recommandation [50], à savoir comme données ci-dessus : EAM, EQM, Prc, et Cvr. Cette méthode d'évaluation est entièrement reproductible, car il n'y a aucune sélection aléatoire. Dans ce qui suit, nous présentons en détail les quatre mesures d'évaluation utilisées.

5.3.2.1 Erreur Absolue Moyenne (EAM)

Elle mesure la d´déviation absolue moyenne entre les valeurs de confiance prédites et les valeurs de confiance réelles. Cette mesure est un indicateur de précision de confiance et elle est inversement proportionnelle à la précision, c'est-à-dire : la précision est suffisamment grande si EAM est suffisamment petite, et inversement. La formule de EAM est donnée par l'équation suivante [51].

$$EAM = \frac{\sum_{i=1}^{N} |pi - ri|}{N}$$

Où

N désigne le nombre total de valeurs prédites par l'algorithme ;

Pi désigne la valeur de confiance prédite (calculée) ;

ri désigne la valeur de confiance réelle.

5.3.2.2 Erreur Quadratique Moyenne (EQM)

Elle mesure la déviation moyenne des carrés des écarts entre les valeurs de confiance prédites et les valeurs de confiance réelles. Cette mesure est plus utilisée pour l'évaluation de la précision par rapport à l'Erreur Absolue Moyenne, car elle accorde plus d'importance aux erreurs élevées. L'EQM est donnée par l'équation suivante [51].

$$EQM = \sqrt{\frac{\sum_{i=1}^{N} (pi - ri)^2}{N}}$$

5.3.2.3 Précision (Prc)

Elle mesure le pourcentage d'utilisateurs fiables recommandés par l'algorithme. Soit *Nf* (resp. *Nnf*) le nombre d'utilisateurs fiables (resp. non fiables) recommandés par l'algorithme. La précision *Prc* est donnée par l'équation suivante [51].

$$Prc = \frac{Nf}{Nf + Nnf}$$

5.3.2.4 Couverture (Cvr)

Elle mesure le pourcentage d'utilisateurs fiables pour lesquels l'algorithme est capable de générer des recommandations. Soient *Nf* le nombre d'utilisateurs fiables recommandés par l'algorithme et *Ntf* le nombre total effectif d'utilisateurs fiables dans le système. La couverture *Cvr* est donnée par l'équation suivante [51].

$$Cvr = \frac{Nf}{Ntf}$$

Cette mesure est étroitement liée à la précision. Généralement quand un algorithme fournit des recommandations très précises, sa couverture a tendance à être réduite, et inversement.

5.3.3 Résultats : comparaison et interprétation

Pour l'évaluation de l'algorithme *FastTrust*, nous considérons cinq sous-jeux contenants respectivement 20%, 40%, 60%, 80% et 100% des valeurs de confiance du jeu de données initial "*Résidence hall*". Nous présentons dans cette sous-section une évaluation comparative des résultats obtenus avec l'algorithme *FastTrust* ainsi que les deux autres algorithmes *TidalTrust*et *MoleTrust*sur sur chacun des cinq sous-jeux de données selon les quatre mesures d'évaluation suscitées *(EAM, EQM, Prc, Cvr)*. A signaler que ces quatre catégories de mesures montrent les aspects différents auxquels un algorithme de confiance doit répondre.

5.3.3.1 Erreurs des prédictions *(EAM, EQM)*

La Figure 5.1 et 5.2 montre respectivement l'erreur absolue moyenne *(EAM)* et l'erreur quadratique moyenne *(EQM)* des trois algorithmes FastTrust, *MoleTrust*et *TidalTrust* en fonction du pourcentage des sous-jeux de données (20%, 40%, 60%, 80 et 100%).

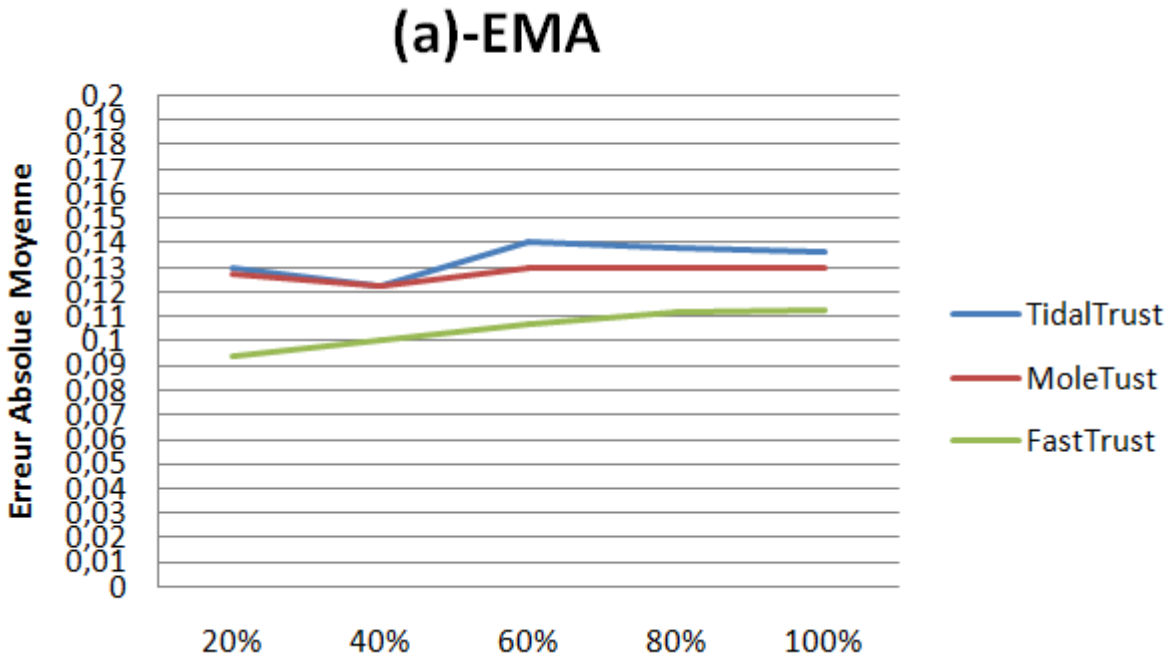


Figure 5.13 Erreurs Absolue Moyenne en fonction du pourcentage du jeu de données

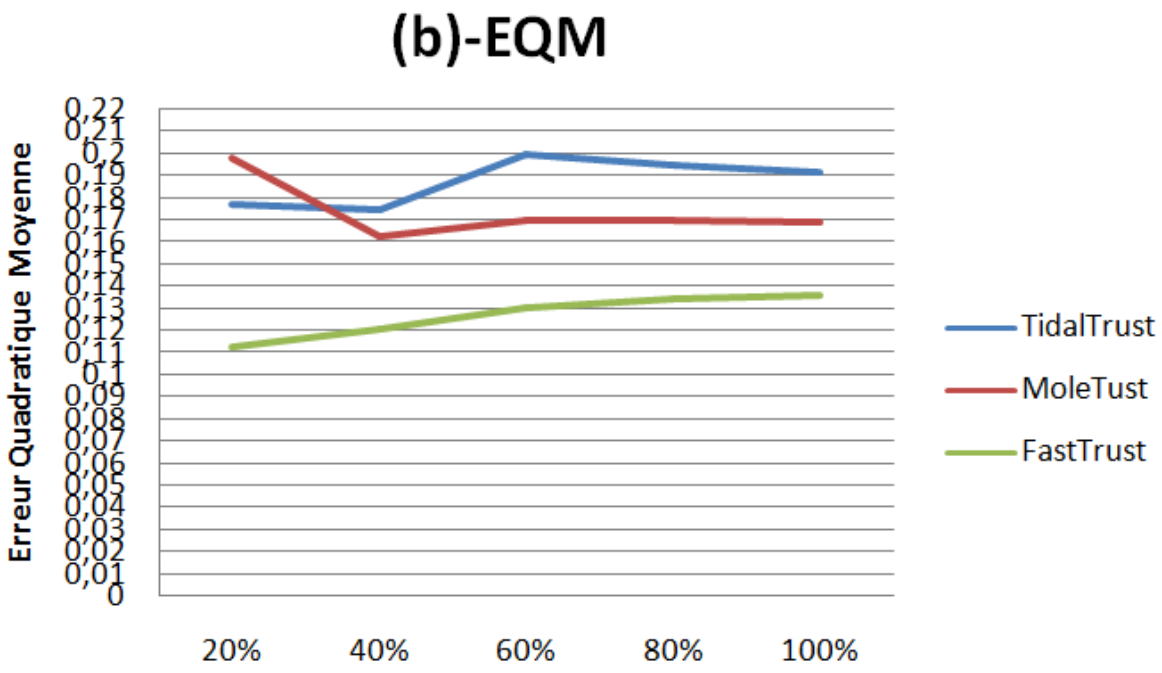


Figure 5.14 Erreur Quadratique Moyenne en fonction du pourcentage du jeu de données

On voit sur les graphes de la Figure 5.1 et 5.2 que l'algorithme proposé *FastTrust* est toujours plus précis que les deux autres algorithmes *TidalTrust*et *MoleTrust* en renvoyant des erreurs plus faibles. On constate aussi que *FastTrust* est moins influencé par les changements de la taille du jeu de données (presque même précision pour les cinq jeux de données (20%, 40%, 60%, 80% et 100%)). Ceci est dû essentiellement à la manière dont *FastTrust* explore les sommets dans le graphe. En effet, *FastTrust* fournit des résultats logiques et raisonnables (faibles écarts) dans toutes les situations, par contre pour *MoleTrust*et *TidalTrust*, dans certaines situations, elles fournissent des résultats illogiques et anormales (grands écarts), On remarque également que *MoleTrust* fournit des résultats meilleurs que *TidalTrust* car ce dernier prend en compte uniquement les personnes fiables lors du calcul de la confiance.

5.3.3.2 **Précision *(Prc)***

La Figure 5.3 montre la précision *(Prc)* des trois algorithmes *fastTrust*, *MoleTrust*et *TidalTrust*en fonction du pourcentage du jeu de données (20%, 40%, 60%, 80% et 100%).

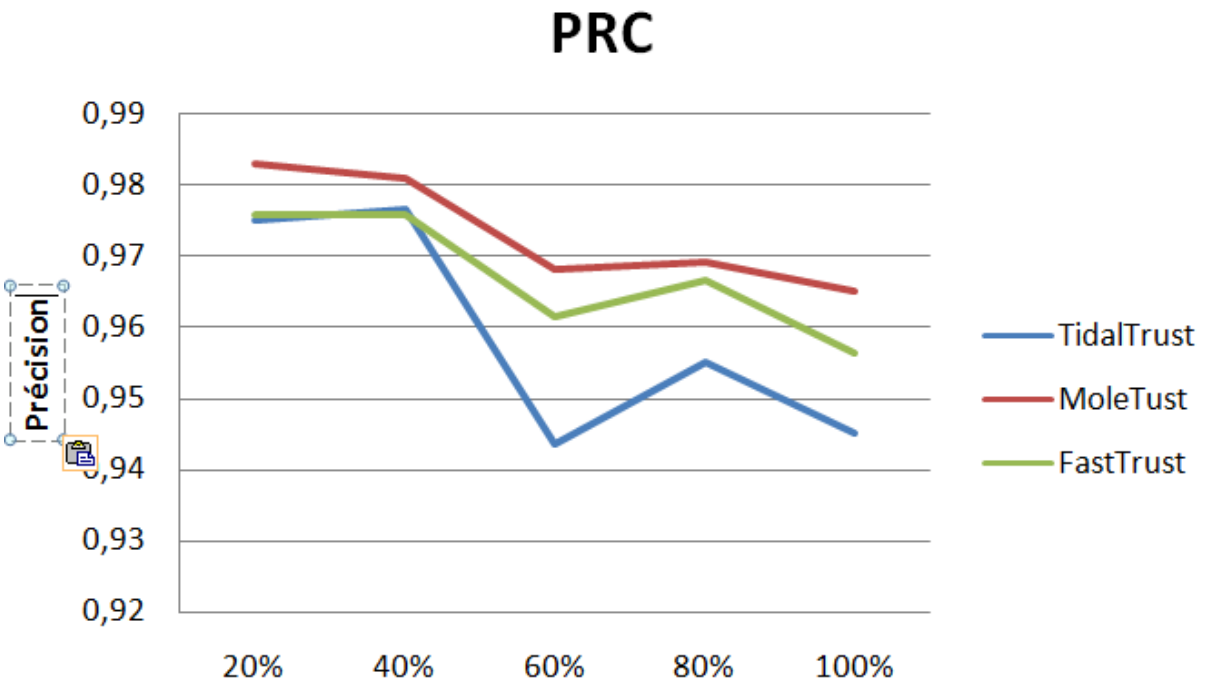


Figure 5.15 Précision des prédictions en fonction du pourcentage du jeu de données

On voit sur le graphe de la Figure 5.3 que l'algorithme proposé *FastTrust* fournit une précision toujours presque égale à celle de *MoleTrust*et supérieure à celle de *TidalTrust*. La précision est inversement proportionnelle à la longueur du chemin de confiance exploité, car plus le chemin de confiance est grand, plus le nombre de fonctions de propagation et de fonctions d'agrégation est plus grand (plus de calculs), et ceci influe négativement sur la qualité de la précision. FastTrust fournit des résultats presque similaires à ceux de *MoleTrust* car ce dernier exploite uniquement les plus courts chemins ce qui démunie considérablement sa couverture. On voit également que *TidalTrust* fournit des résultats inférieurs par rapport aux deux autres algorithmes et ceci est essentiellement dû à la qualité des fonctions de propagation et des fonctions d'agrégation qu'il utilise dans les calculs qui sont moins bonnes par rapport à *MoleTrust*.

5.3.3.3 Couverture *(Cvr)*

La Figure 5.4 montre la couverture *(Cvr)* des trois algorithmes fastTrust, *MoleTrust*et *TidalTrust*en fonction du pourcentage du jeu de données (20%, 40%, 60%, 80% et 100%).

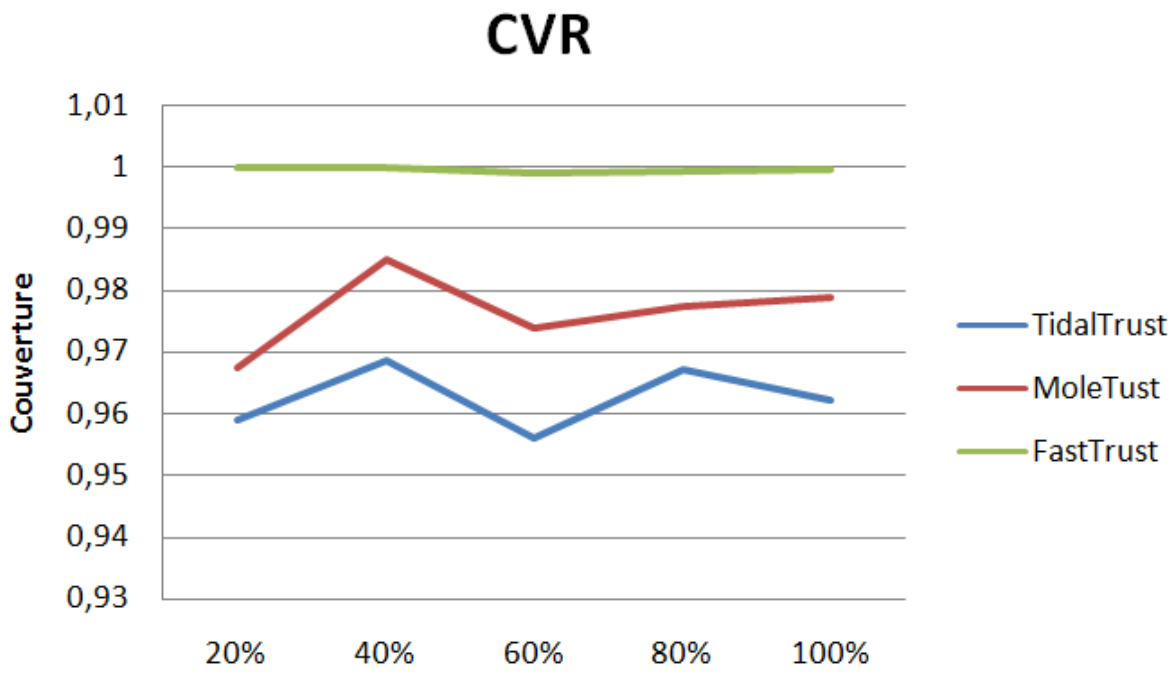


Figure 5.16 Couverture des prédictions en fonction du pourcentage du jeu de données

On voit sur le graphe de la Figure 5.4 que l'algorithme proposé *FastTrust* fournit une couverture toujours plus grande que celles fournies par *MoleTrust*et *TidalTrust* pour chacun des cinq sous-jeux de données. Ceci s'explique essentiellement par la manière dont les deux algorithmes *MoleTrust*et *TidalTrust* exploitent les chemins de confiance. En effet, *MoleTrust*exploite uniquement les plus courts chemins, *TidalTrust* exploite uniquement les plus courts chemins de poids maximum, alors que *fastTrust* exploite, uniquement les voisins de la source et de la cible. En effet, lorsqu'on exploite uniquement les plus courts chemins, on peut des fois ignorer des personnes fiables situées sur des chemins plus longs et ne pas les prendre en considération, et ceci influe négativement sur la couverture.

5.4 Conclusion

Dans ce chapitre, nous avons d'abord implémenté l'algorithme proposé *FastTrust* ainsi que les deux autres algorithmes *TidalTrust* et *MoleTrust* en utilisant le langage de programmation python et l'environnement de développement intégré (IDE) anaconda avec spyder, puis nous avons évalué les trois algorithmes sur un jeu de données (Dataset) de test réel ("Résidence hall"). Pour l'évaluation, nous avons d'abord séparé le jeu de donnée initiale en cinq sous-jeux de données contenant respectivement 20%, 40%, 60%, 80% et 100% des valeurs de confiance, puis nous avons évalué les trois algorithmes sur chacun de ces jeux de données

selon les quatre mesures suivante : l'erreur absolue moyenne (EMA), l'erreur quadratique moyenne (EQM), la précision (Prc), et la couverture (Cvr).

Les résultats obtenus montrent que l'algorithme proposé *FastTrust* fournit des résultats meilleurs par rapport à ceux des deux autres algorithmes TidalTrust et *MoleTrust* pour les trois mesures d´évaluation : l'erreur absolue moyenne (EMA), l'erreur quadratique moyenne (EQM) et la couverture (Cvr) concernant tous les sous-jeux de données (20 %, 40 %, 60 %, 80 % et 100 %). Pour la mesure d'évaluation précision (Prc), *FastTrust* fournit des résultats presque similaires à ceux de *MoleTrust*et meilleurs à ceux de *TidalTrust*. Comme attendu, les résultats obtenus par l'algorithme proposé *FasTrust*sont meilleurs par rapport à ceux des deux autres algorithmes *TidalTrust*et *MoleTrust*.

Conclusion générale

Conclusion générale

La gestion de la confiance dans les Réseaux Sociaux en Ligne (*RSLs*) constitue un axe de recherche très intéressant qui a connu ces dernières années beaucoup de progrès. Cependant beaucoup de problèmes de sécurité restent à résoudre dans ce domaine. L'un des problèmes les plus importants est lié à la confidentialité des données (protection de la vie privée) [8]. En effet, plusieurs solutions ont été proposées dans la littérature, basées sur un ensemble de théories, mais chacune d'elles répondu seulement à certaines exigences.

Nous avons abordé dans ce mémoire le problème de calcule des valeurs de confiance entre des personnes dans un réseau social en ligne. Vu l'expansion rapide et la nature ouverte de ces réseaux, les personnes cherchent toujours à protéger leurs données privées des pairs malveillants. Pour cela, il est indispensable de construire des mécanismes qui permettent d'identifier les pairs fiables avec lesquels partager les données et les pairs non fiables afin de les bloquer, ce qui est l'objet du présent mémoire.

Dans ce mémoire, nous avons d'abord introduit les concepts liés aux *RSLs* et à la confiance dans les *RSLs*. Ensuite, nous avons présenté différents algorithmes de calcul de confiance existants dans la littérature en mettant l'accent sur les avantages et les inconvénients de chacun d'eux.

Dans le cadre de notre contribution, premièrement nous avons proposé un nouveau algorithme de calcul de confiance, appelé *FastTrust*, qui prend en charge des inconvénients de deux algorithmes *TidalTrust* et *MoleTrsut* [15, 42]. *FastTrust* est un algorithme simple, efficace, et de complexité constante O(1), qui permet d'évaluer la confiance qu'une source *s* a envers une cible *c* en se basant sur leurs voisins dans un graphe de confiance.

Pour l'évaluation de notre approche, nous avons d'abord implémenté l'algorithme proposé *FastTrust* ainsi que deux autres algorithmes *MoleTrsut* et *TidalTrust*, ensuite nous avons comparé leurs résultats en utilisant un jeu de données de test réel "*Residence hall*" [38] selon quatre mesures d'évaluation (EAM, EQM, Prc, Cvr). Les résultats obtenus montrent que *FastTrust* fournit des résultats meilleurs que ceux des deux autres algorithmes *MoleTrsut* et *TidalTrust*.

Dans les travaux futurs, nous envisageons de comparer notre algorithme *FastTrust* à d'autres algorithmes plus récents et en utilisant des jeux de données plus pertinents pour une éventuelle publication dans un journal de renommé internationale. Nous envisageons aussi

d’adapter notre approche pour d’autre domaines tels que les internet des objets, les réseaux pair-to-pair, etc.

Références bibliographique

[1]K. Roudaut and C. Bothorel. Explorer et comprendre les réseaux sociaux.Département LUSSI, Brest, 2012.

[2] J.A. Branes. Class and committees in of a Norwegian island parish. *HumanRelations, 7 (1), pp. 39-54*, (1954).

[3] J. A. Golbeck. The dynamics of web-based social networks: Membership,relationships, and change. First Monday, 12 (11), (2007).

[4] G. Pallis, D. Zeinalipour-Yazti, and M. D. Dikaiakos. *New Directions in WebData Management, Online Social Networks : Status and Trends*. Springer, Heidelberg, Germany, 2011.

[5]G. Erto.Semantic social network analysis, 2011.

[6] R. Albert, H. Jeong., and A. L. Barabasi. Internet: Diameter of the world-wideweb. *Nature, 401, pp. 130-131*, (1999).

[7] D. Boyd and N. B. Ellison. Social network sites : Definition, history, andscholarship. *Journal of Computer-Mediated Communication, 13 (1), pp. 210{230*,(2007).

[8] H. Gao, J. Hu, T. Huang, J. Wang, and Y. Chen. Security issues in online socialnetworks. *Internet Computing, IEEE, 15 (4), pp. 56{63*, (2011).

[9] D. McKnight and N. Chervany. What trust means in e-commerce customerrelationships: An interdisciplinary conceptual typology. *International Journal of Electronic Commerce, 6 (2), pp. 35-59*, (2001).

[10] Y.D. Wang and H.H. Emurian. An overview of online trust: Concepts, elements,and implications. *Computers in Human Behavior 21 (1), pp. 105-125*, (2005).

[11] M. Deutsch. Cooperation and trust: some theoretical notes. *In Nebraska Symposium on Motivation. Nebraska University Press, USA*, (1962).

[12] N. Luhmann. *Trust and power: two works*. Wiley, USA, 1979.

[13] D. Gambetta. *Can we trust trust. Trust Making and Breaking CooperativeRelations*. Basil Blackwell, New York, 2000.

[14] P. Sztompka. *Trust: A Sociological Theory*. Cambridge University Press, England,1999.

[15] J. A. Golbeck. *Computing and applying Trust in Web Based Social Networks*.Doctor of philosophy, Department of Computer Science, University of Maryland, USA, 2005.

[16] T.W.A. Grandison. *Trust management for internet applications*. Phd thesis,Department of Computer Science, University of London, UK, 2003.

[17] J. A. Golbeck and J. Hendler. Inferring trust relationships in web-based socialnetworks. *ACM Transactions on Internet Technology, 6 (4), pp.497-529*, (2006).

[18] D.M. Fattaneh. Computational algorithms in social network trust. *Global Journalof Science, Engineering and Technology, 201 (2), pp. 21-26*, (2012).

[19] M. A. Abbasi, J. Tang, and H. Liu. *Trust-Aware Recommender Systems*. ComputerScience and Engineering, Arizona State University, 2014.

[20] B. Christianson and W.S. Harbison. Computational algorithms in social networktrust. In *In Proceedings of the International Workshop on Security Protocols,pp. 171-176, London, UK, Springer-Verlag*, (1996).

[21] F.D. Malayeri and A.D. Malayeri. Computational algorithms in social network trust. *Global Journal of Science, Engineering and Technology, 201 (2), pp. 21- 26*, (2012).

[22] A. Aldini and R. Gorrieri. Trust and reputation systems. *Foundations of Security Analysis and Design*, (2007).

[23] A. Josang, E. Gary, and M. Kinateder. Analysing topologies of transitive trust.In *In Proceedings of the First International Workshop on Formal Aspects inSecurity and Trust (FAST'03), Italy*, (2003).

[24] R. Mansell and B. Collins. *Trust and crime in information societies*. EdwardElgar Publishing, London, 2005.

[25] I. Yaniv and E. Kleinberger. Advice taking in decision making : Egocentricdiscounting and reputation formation. *Organizational Behavior and HumanDecision Processes, 83 (2), pp. 260-281*, (2000).

[26] K. Cook. *Trust in Society*. Russell Sage Foundation, New York, 2001.

[27] R. Hardin. *Trust and Trustworthiness*. Russell Sage Foundation, New York,2002.

[28] M. Richardson and R. Agrawal and P. Domingos. Trus management for the semantic web. In *Proceedings of the Second International Semantic Web Conference, pp. 351-368*, (2003).

[29] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust anddistrust. In *In Proceedings of the World Wide Web Conference, pp. 403-412*,(2004).

[30] P. Mika. *Social Networks and the Semantic Web*. Springer, New York, 2007.

[31] W. Sherchan and al. A survey of trust in social networks. *ACM ComputingSurveys, 45 (4), Article 47*, (2013).

[32] V. Buskens. The social structure of trust. *Social Networks, 20 (3), pp. 265-289*,(1998).

[33] A. Labarre. *Structures de données et algorithmes fondamentaux*. UniversitéParis-Est, Marne-la-Vallée, Octobre 2014.

[34] F. Fürst. *Algorithmique et Programmation*.département d'Informatique, Université de Picardie Jules Verne, 2014.

[35] S. L. Digabel. *Optimisation combinatoire*. D´département de mathématiques etgénie industriel, Ecole Polytechnique de Montréal, 2014.

[36] R. Levien. *Attack resistant trust metrics*. Phd thesis, Department of ComputerScience, University of California, Berkeley, USA, 2002.

[37] C. N. Ziegler and G. Lausen. Propagation models for trust and distrust in socialnetworks. *Springer Science and Business Media, 7 (4/5)*, (2005).

[38] J. Ruderman. A comparison of two trust metrics. *Computer Science andEngineering, University of California San Diego*, 2004.

[39] J. A. Golbeck and J. Hendler. Movie recommendations using trust in web-basedsocial networks. In *In Proceedings of the IEEE Consumer communications and networking conference, Las Vegas*, (2006).

[40] E. Wang. A survey of web-based social network trust. Technical report, ITEC810Information Technology Project Unit, 2009.

[41] M. Taherian, M. Amini, and R. Jalili. Trust inference in web-based social networks using resistive networks. *The Third International Conference on Internet and Web Applications and Services, Athens*, (2008).

[42] P. Massa and P. Avesani. Trust-aware recommender systems. In *In Proceedingsof the 2007 ACM conference on Recommender systems, RecSys '07, pp. 17{24*, (2007).

[43] H. Mase, K. Kanamori, and H. Ohwada. Trust-aware recommender systemincorporating review contents. *International Journal of Machine Learning and Computing, 4 (2), pp. 127-132*, (2014).

[44] T. Luo. *Trust-Based Collective View Prediction*. Springer Science+Business Media, New York, 2013.

[45] U. Kuter and J. A. Golbeck. Using probabilistic confidence models for trust inference in web-based social networks. *ACM Transactions on Internet Technology, 10 (2), Article 8,* (2010).

[46]Sweigart, Al. "The History of Python Programming Language." 2017.

[47]A.D. Hanke, C. Halchenko, M. Hagen, et al. "Anaconda: A software package forconducting reproducible research in large-scale neuroimaging studies." Journal of Neuroinformatics, vol. 14, no. 6, pp. 717-727, 20.

[48]J. Pérez, F. Granger, "IPython: A System for Interactive Scientific Computing",

Computing in Science & Engineering, vol. 9, no. 3, pp. 21-29, May/June 2007.

[49] J. Kunegis. Konect - the koblenz network collection. In In Proceedings International Conference on World Wide Web Companion, pp. 1343-1350, (2013).

[50] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22 (1), pp. 5{53, (2004).

[51] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. Foundations and Trends in Human-Computer Interaction, 4 (2), pp. 81{173, (2011).

[52] A. Lefebvre. *Les réseaux sociaux : pivot de l'Internet 2.0*. MM2 Editions, Paris,2005.

[53] A. Girard. *Réseaux Sociaux Numériques : revue de littérature et perspectives derecherche*. Université Montpellier II, France, 2009.

[54] Y. Cheng. *ACCESS CONTROL FOR ONLINE SOCIAL NETWORKS USING RELATIONSHIP TYPE PATTERNS*. Doctor of philosophy in computer science, The University of Texas at San Antonio, USA, 2014.

[55] S. Ten Kate. *Trustworthiness within social networking sites : A study on the intersection of HCI and sociology*. University of Amsterdam, Amsterdam, 2009.

Résumé

Ce mémoire se concentre sur la problématique de la confiance dans les réseaux sociaux en ligne (RSLs) et propose une approche visant à renforcer cette confiance. Les RSLs jouent un rôle central dans nos interactions en ligne, mais ils sont souvent confrontés à des problèmes tels que la fiabilité des informations partagées et la présence d'utilisateurs malveillants.

Dans ce contexte, nous avons étudié différents algorithmes de calcul de confiance qui ont été proposés dans la littérature. Nous avons analysé leurs concepts fondamentaux, leurs méthodes de propagation de la confiance et leurs performances. Cette étude nous a permis de mettre en évidence les forces et les limites de ces algorithmes existants.

Ensuite, nous avons développé un nouvel algorithme de calcul de confiance spécifiquement conçu pour les RSLs. Notre algorithme, appelé FastTrust ,repose sur une approche novatrice de propagation de la confiance au sein du réseau social. Nous avons réalisé des expérimentations approfondies pour évaluer son efficacité et sa performance, en comparaison avec les algorithmes existants. Les résultats obtenus ont démontré une amélioration significative, tant en termes de qualité des résultats que de complexité de calcul.

En conclusion, ce mémoire contribue à l'avancement des connaissances dans le domaine de la confiance dans les RSLs. Notre nouvel algorithme offre une solution pratique pour évaluer la confiance des utilisateurs et prendre des décisions plus éclairées en matière de partage de données privées. Il représente une avancée importante pour renforcer la confiance et la sécurité dans les réseaux sociaux en ligne.

Mots-clés : Réseaux sociaux en ligne, Confiance, Algorithmes de calcul de confiance, Nouvel algorithme, Renforcement de la confiance, Sécurité des données.

Abstract

This thesis focuses on the issue of trust in online social networks (OSNs) and proposes an approach to strengthen trust within these networks. OSNs play a central role in our online interactions, but they often face challenges such as the reliability of shared information and the presence of malicious users. In this context, we have studied various global trust algorithms proposed in the literature and analyzed their fundamental concepts and performance. This study has revealed the strengths and limitations of these existing global algorithms.

Subsequently, we have developed a new local trust algorithm specifically designed for OSNs, named *FastTrust*, which takes into account only the neighbors of the source and the targer in calculating trust between users within the social network. We conducted comprehensive experiments to evaluate the effectiveness of our approach by making comparison with two others algorithms namely *MoleTrust* and *TidalTrust*. The results obtained show that our approach provides better results and computational complexity than the other approach.

In conclusion, this thesis contributes to advancing our knowledge in the field of trust in OSNs. Our algorithm *FastTrust* provides a practical solution to assess user trust in OSNs and make decisions regarding the sharing of private data and represents a significant advancement in reinforcing trust and security in online social networks.

Keywords: Online social networks, Trust, Propagation, Aggregation, Local trust algorithm, Global trust algorithm, Acuuracy.