Table des Matières

Liste des Tableaux

Liste des figures

Introduction générale

De récentes avancées dans le domaine des systèmes complexes ont fait ressortir le rôle central que jouent les graphes dans de nombreux phénomènes. Ces graphes permettent de modéliser les interactions entre les différents acteurs de ces phénomènes, qui interviennent dans de très nombreux domaines : sociologie, biologie, linguistique, physique, informatique, épidémiologie, …etc. Ces grands graphes d'interactions possèdent des propriétés structurelles communes et leur étude soulève de nombreuses difficultés algorithmiques qui deviennent des défis vues les tailles des graphes correspondants. Pour s'adapter à ces tailles, il serait utile de développer une algorithmique spécifique aux grands graphes d'interactions pouvant tirer parti de leurs propriétés communes.

Ainsi, le but de ce travail s'inscrit dans ce contexte interdisciplinaire, en se concentrant sur la question algorithmique de découverte de communautés d'intérêts (c'est-à-dire de groupes d'acteurs partageant des centres d'intérêts commun ou ayant des profils similaires) dans les réseaux d'interactions et, plus particulièrement, la découverte et le référencement des membres des différentes communautés dans les réseaux sociaux, issues des systèmes distribués mobiles tels que les réseaux ad hoc. Ces derniers sont constitués de mobiles possédant des caches limités ne permettant pas à chacun d'eux de référencer tous les membres de sa communauté, notamment dans le cas du passage à l'échelle.

Plusieurs méthodes de détection communauté ont été proposées et examinées dans la littérature. Ces méthodes adoptent généralement une vue statique et globale du réseau, cependant elles ne peuvent être directement appliquées dans des environnements mobiles. La découverte de communautés dans ces derniers est en plein expansion et présente un véritable challenge. C'est ainsi que notre objectif s'inscrit en droite ligne dans ce contexte.

Nous proposons dans ce rapport un nouvel algorithme distribué de découverte de communauté d'intérêt dans les réseaux mobiles qui supporte le passage à l'échelle et prend en considération la contrainte de la limitation des mémoires ou caches des mobiles.

L’idée consiste en l’adaptation de certains concepts de la théorie de graphe : le concept de l’ensemble indépendant maximal (MIS) et de l’arbre couvrant de degré borné (k-arbre) en vue de construire une topologie virtuelle dynamique adaptée à la découverte distribuée de communauté, combinant ainsi les avantages des deux concepts cités précédemment, à savoir, minimiser le nombre de nœuds participants dans la topologie ainsi que le nombre de liens utilisés.

L’ensemble indépendant maximal de nœuds assure la tâche de découverte et de référencement des membres des communautés, et l’arbre couvrant permet de réduire les degrés (nombre des entrées dans les caches) des nœuds du MIS afin de satisfaire la contrainte de caches limités.

Le reste de ce mémoire est structuré comme suit : Nous allons d'abord présenter les grands réseaux d’interaction et leurs applications dans les différents domaines concernés. Nous introduirons ensuite la problématique de détection de communautés, une synthèse des travaux sur ce sujet sera proposée, suivie d’un état de l’art sur les topologies virtuelles dans les réseaux mobiles ad hoc. Nous présenterons par la suite notre contribution, et donnerons un exemple de déroulement dans lequel nous illustrons le fonctionnement de notre algorithme. Nous terminerons ce mémoire par une conclusion et des perspectives.

Chapitre 1

Grands réseaux d'interactions

1.1 Introduction

Un graphe G = (V, E) permet de modéliser des interactions entre des objets (représentés par un ensemble V de n nœuds) reliés par des liens (représentés par un ensemble $E \subseteq V \times V$ de m paires de nœuds nommées arêtes). Ces graphes peuvent être orientés ou non-orientés, pondérés ou non-pondérés.

Les graphes permettent de modéliser de nombreux phénomènes provenant de domaines variés. Ils se caractérisent par le fait que des interactions simples et facilement compréhensibles entre nœuds à l'échelle microscopique produisent des propriétés topologiques globales et des comportements macroscopiques non-triviaux difficilement interprétables.

Nous allons présenter quelques domaines dans lesquels ces graphes jouent un rôle important. Nous allons ensuite présenter leurs propriétés communes qui permettent d'envisager des algorithmes génériques qui peuvent s'appliquer à ces domaines.

1.2 Domaines concernés

Nous présentons quelques exemples dans lesquels des graphes sont utilisés comme outils de modélisation de phénomènes complexes (Les acteurs du phénomène sont modélisés par les nœuds du graphe, et les interactions entre eux sont modélisées par des liens ou arêtes entre les nœuds). Ces exemples vont illustrer la diversité des domaines d'applications possibles.

- **Les réseaux sociaux :**

  Les réseaux sociaux forment un champ d'application important [1] dont les acteurs sont des individus ou bien des entités sociales (associations, entreprises, pays, etc). Les liens entre eux peuvent être de différentes natures. Nous pouvons, ainsi, distinguer plusieurs types de réseaux : les réseaux de connaissance (deux individus sont reliés s'ils se connaissent), les réseaux de collaboration (deux individus sont reliés s'ils ont travaillé ensemble, les réseaux d'appels téléphoniques [2] (deux individus ou numéros de téléphones sont reliés s'il y a eu un appel entre eux), les réseaux d'échanges (deux entités sont reliées si elles ont échangé un fichier [3] ou un courrier électronique [4] par exemple), etc.

- **Les réseaux biologiques :**

  Les réseaux biologiques sont assez divers parmi lesquels nous citons les réseaux métaboliques [5] (les nœuds sont des gènes ou des protéines qui sont liés par leurs interactions chimiques), ou les réseaux de neurones (chaque neurone est connecté à plusieurs autres neurones) ou les réseaux trophiques [WM00] (les espèces d'un écosystème sont reliées pour représenter les chaînes alimentaire). Un exemple d'une chaîne alimentaire [33] est illustré sur la figure 1.1.

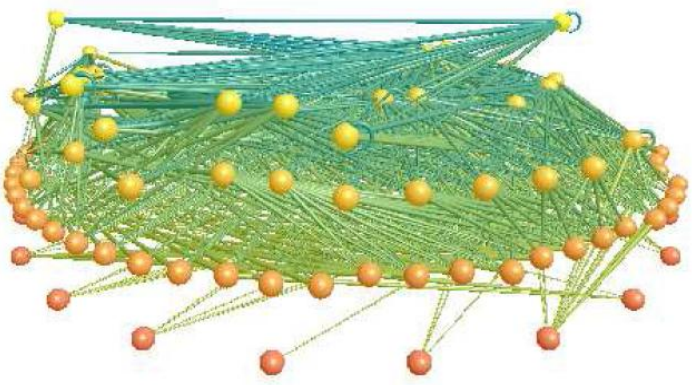

Fig.1.1. Une chaîne alimentaire des interactions de prédateur-proie entre les espèces dans un lac d'eau. [33]

- **Les réseaux d'infrastructure :**

  Ces réseaux représentent des connections matérielles entre objets distribués dans un espace géographique. Nous pouvons citer les réseaux de transport (routes entre villes ou liaisons aériennes entre aéroports), les réseaux de distribution électrique (câbles entre les lieux de production et de consommation) ou encore le réseau physique de l'Internet [6] (câbles entre ordinateurs).

- **Les réseaux d'information :**

  Ils représentent des liens abstraits de référencement entre des supports d'information. Parmi eux, les réseaux de citation d'articles (les nœuds sont des articles et un lien orienté de l'article A vers l'article B indique que A cite B) ou les graphes du Web [7] (les nœuds sont des pages Web liées par des liens hypertextes).

Fig.1.2. Réseau des citations.

- **Les réseaux linguistiques :**

  Ces réseaux relient les mots d'un langage donné et regroupent entre autres les réseaux des synonymies (deux mots sont reliés s'ils sont synonymes), les réseaux de cooccurrences [8] (deux mots sont reliés s'ils apparaissent dans une même phrase d'un ouvrage), ou encore les réseaux de dictionnaires (deux mots sont liés si l'un est utilisé dans la définition de l'autre).

Les problématiques d'étude sont variées selon les disciplines. Par exemple les réseaux métaboliques sont utilisés pour comprendre le fonctionnement de la cellule [5]. L'analyse des réseaux d'infrastructure permet de détecter leurs points faibles susceptibles d'être la cible d'attaques ou de pannes ayant des conséquences majeures [9]. Les moteurs de recherche du Web se basent sur des graphes du Web pour référencer de manière pertinente les pages Web. Un opérateur de télécommunications peut vouloir établir les profils de ses clients en analysant le réseau des appels téléphoniques, etc.

Ces objets provenant de domaines différents ont des propriétés communes et soulèvent des problématiques similaires, ce qui permet d'envisager des méthodes algorithmiques générales.

1.3 Propriétés communes

Plusieurs études ont montrés que les graphes complexes ont des caractéristiques structurelles communes et non triviales [34, 35]. Les plus étudiées sont :

- **Effet 'petit monde' «**The small-world effect ».

  L'effet petits-mondes (small-word de l'expérience de Milgram [10]) traduit le fait qu'il existe des très courts chemins pour relier chaque paire de nœuds. La notion petit monde est définie, dans certains articles [11] comme la combinaison d'un fort coefficient de clustering et d'un petit diamètre. Cette propriété étudiée est vérifiée par le modèle de graphes aléatoires d'*Erdös-Rényi* [36].

- **Clustering** «Transitivity or clustering».

  Une des propriétés essentielles des réseaux complexe est l'existence d'une forte densité locale qui s'oppose à la faible densité globale du graphe. Cette densité est souvent capturée par le coefficient de clustering [11] qui compte la probabilité que deux voisins d'un même nœud soient eux-mêmes voisins. Elle s'explique par la tendance des acteurs à se regrouper en modules ou communautés.

- **Distribution des degrés** «Degree distributions ».

  Il existe dans les grands réseaux un nombre non négligeable de nœuds possédant un très fort degré par rapport à une majorité de nœuds possédant un très faible degré. Cette distribution est souvent bien approximée par une loi de puissance pour laquelle le nombre P_k de nœuds de degré k est proportionnelle à $k^{-\alpha}$, pour une constante $\alpha > 0$ sur un intervalle de plusieurs ordres de grandeur (par exemple entre $k = 10$ et $k = 10^6$). En 1999, *Faloutsos* et *al* [37] ont observé que le réseau Internet présentait cette propriété. Par la suite, elle a également été observée dans des réseaux de pages web, et des réseaux de distribution d'électricité [35].

- **Résilience des réseaux** «Network resilience»

  La propriété de résilience des réseaux est liée à la distribution de degrés. Quand il y a une suppression de nœuds, la longueur typique des chemins augmentera, et des paires de

nœuds seront déconnectées ainsi la communication entre eux deviendra impossible. Le niveau de résilience du réseau se varie selon la déconnexion des nœuds.

- **Mixing patterns**

  Dans la plupart des réseaux, il existe quelques types de nœuds différents et la probabilité qu'il existe un lien entre une paire de nœuds différente dépend souvent du type en question. Cette propriété surprenante dans un réseau complexe explique la différence des types de ses nœuds. *Maslov* et al [38] ont étudié l'existence de trois types de nœuds dans le réseau Internet : les fournisseurs qui ont une forte connectivité, les consommateurs qui sont les utilisateurs finaux, et les providers de services Internet qui jouent le rôle de relais entre les deux types de nœuds précédents.

- **Corrélation entre degré** «Degree correlations ».

  Comme le degré est lui-même une propriété qui représente la topologie du réseau, la corrélation de degré peut fournir des détails intéressants sur l'effet de la structure du réseau. En utilisant la propriété de corrélation de degré on veut savoir si les nœuds de fort degré sont associés préférentiellement avec les nœuds de haut degré ou bien avec ceux de faible degré. Plusieurs études ont été proposées pour quantifier la corrélation de degré à l'exemple des travaux de *Maslov* et al [38, 39].

- **Navigabilité** «Network navigation»

  Kleinberg en 2000 [40] a introduit la notion de navigabilité. Il s'agit de caractériser non seulement la longueur des chemins, mais aussi la façon dont ils sont découverts. Dans l'expérience de *Milgram*, les individus n'utilisent que leurs contacts locaux pour acheminer la lettre, il s'agit donc d'un routage décentralisé, en ce sens que l'on n'utilise qu'une vue locale du réseau pour transmettre le message. C'était aussi le cas de la navigation à travers le réseau des pages web il y a quelques années, qui se faisait d'une page à l'autre sans connaître la carte globale du réseau [41]. Par ailleurs, la découverte des chemins de façon décentralisée est une nécessité pour les réseaux d'interactions réels qui comportent un très grand nombre de sommets et où une recherche classique des plus courts chemins n'est pas possible, car très coûteuse en temps .

1.4 Conclusion

On a vue dans ce chapitre que les grands graphes d'interactions possèdent des caractéristiques communes non-triviales, à savoir, le faible degré moyen, la forte hétérogénéité des degrés, les chemins courts entre tous les nœuds et la faible densité globale couplée à la forte densité locale. Cette dernière caractéristique, souvent mesurée par le coefficient de clustering, traduit l'existence de zones denses faiblement interconnectées appelées communautés qui représentent le mot clé de la problématique traité dans ce travail.

Dans le chapitre suivant, nous allons présenter les déférentes définitions de communautés, leurs intérêts et applications et nous donnerons, par la suite, un panorama des différentes approches qui ont été envisagées pour extraire la structure de communauté d'un graphe.

Chapitre 2

Découverte de communautés

2.1 Introduction

L'existence dans les réseaux d'interactions de zones plus denses que d'autres découle souvent de la présence dans le graphe d'une structure de communautés. Les communautés sont des groupes de nœuds qui partagent probablement des propriétés communes et/ou rôles semblables dans le réseau en question.

Ce type de structure intervient dans de nombreux réseaux d'interactions et apporte de l'information sur leur organisation. Les communautés peuvent avoir des interprétations différentes suivant le type de réseau considéré. Ainsi, les communautés peuvent correspondre par exemple à des fonctions biologiques de la cellule pour les réseaux métaboliques, [12], aux subdivisions dans les chaînes alimentaires [13], à des thématiques, par exemple , les groupes de pages Web traitant le même sujet dans les réseaux d'information [14], et à des ensembles d'individus possédant des points communs et dont les liens sociaux sont naturellement plus forts dans les réseaux sociaux [15], etc .

La détection de communautés peut donc être vue comme un outil générique qui peut s'appliquer à tous les grands réseaux d'interactions. Les intérêts principaux qui ont motivés les études portées sur la découverte de communautés sont :

- La détection de communautés est un outil important pour la compréhension des structures et des fonctionnements des systèmes complexes.
- Les communautés permettent de donner un point de vue macroscopique sur la structure des graphes. Elles permettent de regrouper et d'identifier les nœuds qui jouent potentiellement des rôles similaires.
- La détection de communautés peut aussi être utilisée pour la visualisation des graphes complexes [16].
- Les communautés sont utilisées pour diviser le graphe afin d'effectuer des calculs séparés moins coûteux sur chaque communauté. Ce procédé de parallélisme de calcul permet d'envisager des gains de complexité pour les algorithmes qui s'appliquent sur certains grands graphes.
- La détection de communautés permet une classification des nœuds, selon leur position topologique dans les groupes. Ainsi, les nœuds possédant une position centrale dans leurs clusters partagent un grand nombre de liens avec les autres groupes, ils peuvent avoir une importante fonction de contrôle et de stabilité au sein du groupe, quand aux nœuds de frontières, ils jouent un rôle important de relais entre les différentes communautés. Une telle classification est très utile dans les réseaux sociaux et les réseaux métaboliques [42].
- Les communautés peuvent être utilisées pour améliorer les méthodes de compression de graphes à l'exemple du travail [43].

2.2 Définitions de communauté

La notion de communauté dans un graphe est difficile à définir formellement. C'est pourquoi toutes les approches récentes ont utilisé une notion intuitive des communautés.

Une communauté [1] est vue comme un ensemble de nœuds dont la densité de connexions internes est plus forte que la densité de connexions vers l'extérieur, comme illustré dans la figure 2.1.

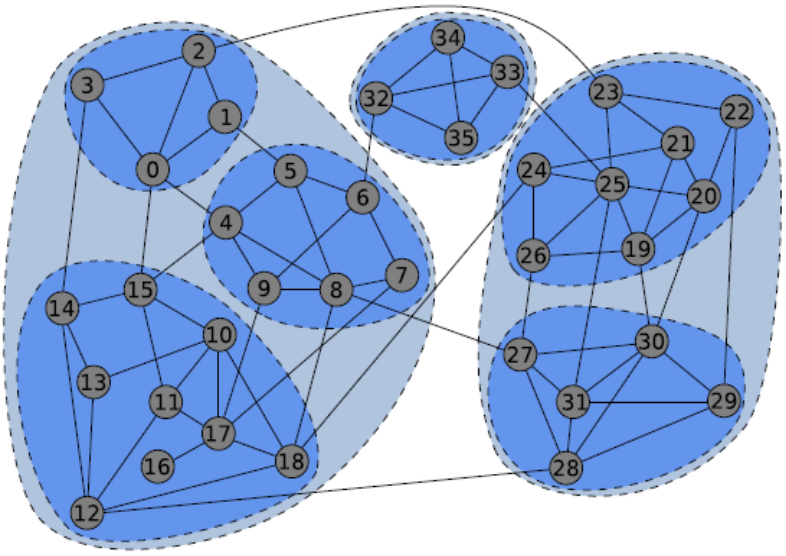

Fig. 2.1. Exemple de structures de communautés dans un graphe

Dans [17], les auteurs, donnent deux notions de communautés, à savoir, communautés au sens fort et communautés au sens faible:

Soit G un graphe, on considère k_i le degré d'un nœud $i \in G$, en termes de la matrice d'adjacence A_{ij}, on a : $k_i = \sum_j A_{i,j}$. Soit un sous graphe $V \subset G$ et $i \in V$, le degré total est donné par l'expression suivante :

$$k_i(V) = k_i^{in}(V) + k_i^{out}(V)$$

Tel que : $k_i^{in}(V) = \sum_{j \in V} A_{i,j}$ est le nombre de liens reliant le nœud i à d'autres nœuds appartenant $à$ V, et $k_i^{out}(V) = \sum_{j \notin V} A_{i,j}$ est le nombre de liens vers les nœuds qui n'appartiennent pas à V (le reste du réseau).

- ***Définition d'une communauté au sens fort:***

  Le sous-graphe V est une communauté au sens fort si :

$$k_i^{in}(V) > k_i^{out}(V), \forall i \in V$$

  Une communauté est définie en tant qu'un ensemble de nœuds dans lequel chaque nœud a plus de connexions au sein de cette communauté qu'avec le reste du réseau.

- ***Définition d'une communauté au sens faible :***

  Le sous graphe V est une communauté au sens faible si :

$$\sum_{i \in V} k_i^{in}(V) > \sum_{i \in V} k_i^{out}(V)$$

Une communauté est définie comme un ensemble de nœuds dont le nombre total de liens internes est supérieur au nombre total des liens vers l'extérieur.

Une définition basic d'une communauté est une clique. Les triangles sont les cliques les plus simples, et sont fréquents dans les réseaux du monde réel mais les plus grandes cliques sont rares, ainsi elles ne sont pas de bons modèles de communautés.

Une communauté est un sous graphe indivisible [18].

Les communautés sont des groupes de nœuds qui sont similaires les uns aux autres. Un critère est choisi pour l'évaluation de la similarité.

2.3 Représentation graphique des communautés

La théorie des graphes est employée pour représenter le réseau et même les communautés dans le réseau en question. Les Dendrogrammes sont aussi souvent utilisés pour illustrer la progression entière de l'algorithme de découverte de communautés et le regroupement des nœuds depuis le graphe initial au graphe résultant partitionné en communauté comme le montre la figure 2.2. Les coupes horizontales à travers l'arbre hiérarchique représentent clairement toutes les divisions possibles en communautés à chaque niveau.

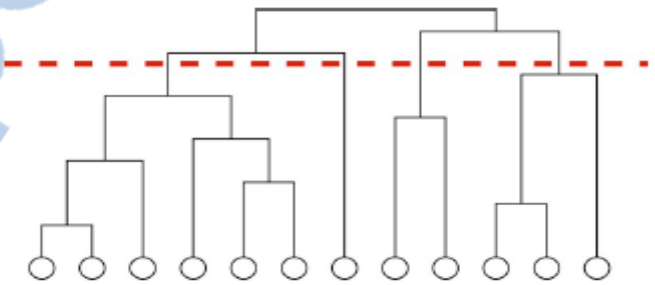

Fig.2.2. Dendrogramme illustrant le résultat d'un algorithme de détection de communautés.

2.4 Mesures de la qualité de partition d'un réseau en communauté

Comment savoir si les communautés détectées sont bonnes ou non et comment évaluer une telle partition? Quelle est la meilleure partition pour le réseau en question ?

Quand est ce qu'on coupe le dendrogramme pour obtenir le niveau de division adéquat du réseau ou bien le nombre de communautés appropriées?

Pour répondre à ces questions, *Newman* et *al* [21] ont introduit une mesure de la qualité d'une division particulière du réseau appelé «*modularité*». Cette mesure est basée sur la mesure d'*assortative mixing* proposée déjà par *Newman* dans [N03].

Imaginer une partition particulière d'un réseau en *k* communautés. Soit *e* une matrice symétrique $k \times k$, ses éléments e_{ij} représente la fraction de tous les liens dans le réseau qui relient les sommets de la communauté i aux sommets de la communauté j. (les auteurs considèrent tous les liens dans le réseau original). La trace de la matrice e : $Tr\, e = \sum_i e_{ii}$ représente la fraction de tous les liens qui connectent les sommets dans les mêmes communautés. Une valeur élevée de la trace indique une bonne division en communautés. La somme de n'importe quelle ligne (ou colonne) de e : $a_i = \sum_j e_{ij}$ correspond à la fraction de tous les liens reliés aux sommets de la communauté i.

Si le réseau ne possède pas la propriété de structure de communauté, la valeur prévue des fractions des liens dans une partition peut être estimée. C'est la probabilité qu'un sommet d'extrémité d'un lien soit dans la communauté *i*, donc a_i, multiplié par la fraction des liens qui se termine par un sommet dans la communauté *i, donc* a_i. Alors, on peut écrire: $e_{ij} = a_i a_j$, est le nombre de liens intra-communauté qui sont prévus.

Ainsi, la mesure de modularité est définit comme suit :

$$Q = \sum_i (e_{ii} - a_i^2) = Tr\ e - \left\| e^2 \right\|$$

Cette quantité mesure la fraction des liens reliant des sommets de la même communauté moin la valeur prévue de la même quantité aux mêmes partitions de communauté mais en employant des connexions aléatoires entre les sommets.

Si une partition particulière ne possède plus de liens intra-communautés (une seule communauté) comme il a été prévu dans le modèle aléatoire, la modularité est Q = 0. Les valeurs autres que 0 indiquent des déviations de l'aspect aléatoire, et les valeurs supérieures à 0. 3 semblent indiquer l'existence d'une structure de communauté significative. Pratiquement, les valeurs de modularité pour de tels réseaux sont souvent entre 0.3 à 0.7, alors que les valeurs les plus élevées sont rares. Cependant, il est possible que les partitions de meilleures modularités ne correspondent pas aux partitions en communautés les plus pertinentes [44].

2.7 Méthodes de découverte de communauté

Nous allons présenter quelques approches de découverte de communautés qui ont été proposées. Nous présenterons quelques une qui ont reçu le plus d'attention de la part de la communauté scientifique. Notre but est de donner une vue d'ensemble des méthodes proposées.

2.7.1 Approches classiques

La détection de communautés s'approche des deux thématiques classiques en informatique, à savoir, le partitionnement de graphe et le clustering de données.

La première, initialement introduite pour la parallélisation de processus, cherche à répartir des tâches, représentées par les sommets d'un graphe, tout en minimisant les échanges, représentés par les arêtes.

La seconde thématique de clustering de données est une thématique plus vaste dans laquelle on cherche à regrouper des données possédant des caractères communs. Cette problématique possède des applications dans de très nombreux domaines dont la détection de communautés est considérée comme une de ses applications pour les grands graphes. Le problème de détection de communautés peut être vu comme un problème de clustering de données pour lequel il faut choisir une distance adéquate.

- **Partitionnement de graphe.**

  Le but du partitionnement de graphe est de grouper les nœuds d'un graphe en un nombre prédéterminé de parties tout en minimisant le nombre d'arêtes entre les différents groupes. Ces approches ne conviennent pas complètement au problème de détection de communautés car elles nécessitent de connaitre préalablement le nombre de communautés recherchées ainsi que leur taille. Nous citons les méthodes ayant eu le plus de succès :

- **La méthode de bissection spectrale [19] :**

  Cette méthode consiste à calculer le vecteur propre correspondant à la plus petite valeur propre non nulle de la matrice Laplacienne du graphe L = D − A (avec D la matrice diagonale des degrés des différents nœuds du graphe et A sa matrice d'adjacence). Le graphe est alors séparé en deux parties en fonction du signe de leur composante selon ce vecteur propre. La complexité de cette méthode est $O(n^3)$ et

obtient de bons résultats lorsque le graphe possède deux grandes communautés de tailles similaires, ce qui n'est qu'un cas particulier dans le problème de détection de communautés.

- **La méthode de Kernighan et Lin [45] :**

  C'est un algorithme de bissection visant à trouver la coupe du graphe minimisant le nombre d'arêtes tombant entre les deux groupes. L'algorithme nécessite comme paramètre la taille des communautés à détecter. Une coupe de la bonne taille est choisie arbitrairement comme point de départ en utilisant une heuristique gloutonne. La bissection est améliorée itérativement en faisant des échanges de sommets entre les communautés. A chaque étape on échange les deux sommets procurant la meilleure réduction du nombre d'arêtes externes, avec la condition imposée de ne jamais changer deux fois un même sommet. La meilleure partition rencontrée au cours du déroulement de l'algorithme est retenue. La complexité du pire cas est en $O(n^3)$.

- **Méthode de clustering hiérarchique :**

  Cette méthode a pour but de grouper les nœuds en sous-ensembles (qui représenteront les communautés) de telle sorte que chaque nœud soit groupé avec d'autres nœuds similaires. Pour cela, une mesure de similarité entre chaque paire de nœuds nécessaire d'être introduite. L'algorithme commence d'une structure dans laquelle chaque nœud est identifié à une communauté. On itère alors les étapes suivantes : on calcule des distances entre communautés et on fusionne les deux communautés les plus proches en une nouvelle communauté. Le nombre de communautés diminue de un à chaque étape, le processus s'arrête lorsqu'il n'y a plus qu'une seule communauté correspondant au graphe entier. On obtient ainsi une structure hiérarchique de communautés qui peut être représentée sous une forme arborescente appelée dendrogramme : les feuilles sont les nœuds du graphe tandis que les nœuds représentent les communautés créées et sont reliés en fonction des fusions de communautés. La racine de la structure correspond au graphe entier.

2.7.2 Approches séparatives

L'idée commune à toutes ces méthodes est de diviser le graphe en plusieurs communautés en retirant progressivement les arêtes reliant deux communautés distinctes. Les arêtes sont retirées une à une et, à chaque étape, les composantes connexes du graphe obtenu sont identifiées à des communautés. Le processus est répété jusqu'au retrait de toutes les

arêtes. On obtient alors une structure hiérarchique de communautés (dendrogramme), Les méthodes existantes diffèrent par la façon de choisir les arêtes à retirer.

- **L'algorithme de Girvan et Newman basé sur la centralité d'intermédiarité [20, 21, 22, 23]** fait partie de cette famille d'approches : il consiste en la suppression d'arêtes de plus forte centralité d'intermédiarité. Cette centralité est définie pour une arête comme étant le nombre de plus courts chemins passant par cette arête. Il existe en effet peu d'arêtes reliant les différentes communautés et les plus courts chemins entre deux nœuds de deux communautés différentes ont de grandes chances de passer par ces arêtes. Un algorithme calculant la centralité de toutes les arêtes en O(mn) est proposé. Ce calcul est effectué à chaque étape sur le graphe obtenu après retrait des arêtes. La complexité de l'algorithme est $O(m^2n)$.
- **Les approches de Radicchi et al [17] et d'Auber et al [16] basées sur le clustering d'arêtes :**

  La détection des arêtes intercommunautaires est dans ces méthodes basée sur le fait que de telles arêtes sont dans des zones peu clustérisées. Radicchi et al [68] proposent un coefficient de clustering (d'ordre g) d'arêtes. Il est défini comme étant le nombre de cycles de longueur g passant par l'arête divisé par le nombre total de tels cycles possibles (étant donné les degrés des extrémités de l'arête). Cet algorithme retire à chaque étape l'arête de plus faible clustering (d'un ordre donné, 3 ou 4, …). Chaque suppression d'arête ne demande alors qu'une mise à jour locale des coefficients de clustering, ce qui lui permet d'être plus rapide que le précédent algorithme. La complexité totale est en $O(m^2)$. L'approche d'Auber et al [6] utilise un clustering d'arêtes d'ordre 3 pour proposer un outil de visualisation de graphes.
- **L'algorithme de Fortunato et al basé sur la centralité d'information [46] :**

  C'est une variante de l'algorithme de Girvan et Newman basé sur une autre notion de centralité : la centralité d'information. Les auteurs définissent l'efficacité de communication ε_{ij} entre deux sommets i et j du graphe comme étant l'inverse de leur distance (en terme de plus courts chemins). L'efficacité E du réseau est définie comme la moyenne des ε_{ij} pour tous les couples de sommets. Enfin, la centralité d'information d'une arête est définie comme étant la diminution relative de l'efficacité du réseau lorsque l'on retire cette arête du

graphe. Cette approche donne de meilleurs résultats que l'approche de Girvan et Newman mais a une complexité en $O(m^3n)$.

2.7.3 Approches agglomératives

L'idée commune de toutes ces méthodes est d'utiliser une approche, s'apparentant à celle du clustering hiérarchique, dans laquelle les nœuds sont regroupés itérativement en communautés en partant d'une partition de n communautés composées d'un seul nœud. Les regroupements de communautés sont poursuivis jusqu'à obtenir une seule communauté regroupant tous les nœuds, et une structure hiérarchique de communautés (dendrogramme) est ainsi construite. Ces approches sont similaires aux approches séparatives à la différence qu'elles commencent de bas en haut dans la hiérarchie des communautés au lieu de commencer de haut en bas.

- **L'algorithme d'optimisation de la modularité proposé par Newman [4, 24]:**

  Newman introduit une notion de **modularité** ; il s'agit d'une valeur quantifiant la qualité d'une partition du graphe en communautés. L'algorithme fusionne alors à chaque étape les communautés permettant d'avoir la plus grande augmentation de la modularité. Etant donnée une partition du graphe en communautés, on définit la fraction *eij* d'arêtes du graphe joignant la communauté *i* à la communauté *j* (une arête reliant deux communautés distinctes est comptabilisée pour moitié dans *eij* et pour moitié dans *eji*). Notons $ai = \sum j\ eij$ la fraction d'arêtes ayant une extrémité dans la communauté, la modularité est alors définie par : $\sum_i (e_{ii} - a_i^2)$ .Cette valeur représente la proportion d'arêtes qui sont à l'intérieur d'une communauté moins cette même proportion dans le cas où les arêtes auraient été placées au hasard entre les communautés (en respectant les fractions d'arêtes *ai* intervenant dans chaque communauté i). La complexité globale de l'algorithme est O(mn) qui est améliorée par Clauset [47] en utilisant une structure de données adaptée. D'autres optimisations de cette méthode ont récemment été proposées [48] afin de traiter des graphes de taille encore plus importante.

- **Une autre généralisation de la modularité suggérée par Arenas et al. [52].** Ils ont fait remarqué que l'élément fondamentale pour définir la modularité est l'arête, mais que la forte densité d'arêtes à l'intérieur des communautés impliquent la présence d'un grand nombre de chemins et cycles entre les nœuds

(motifs i. e. graphes connexe non orientés) [53], ainsi, ils ont déduit que les chemins courts, ou les **motifs**, d'un réseau, pourrait être utilisé pour définir et identifier des communautés. Ils ont utilisé une extension de la modularité où ses éléments sont des motifs, et non seulement des arêtes. La modularité à base de motifs peut être généralisée simplement en comparant la densité de motifs à l'intérieur des communautés par rapport à la densité attendue dans un graphe aléatoire.

- **L'algorithme de Donetti et Muñoz basé sur les propriétés spectrales de la matrice Laplacienne du graphe [49, 50]:**

  Les propriétés spectrales de la matrice Laplacienne du graphe, L = D−A sont utilisées pour détecter des communautés. En effet les coordonnées i et j des vecteurs propres correspondant aux plus petites valeurs propres non nulles sont corrélées lorsque les sommets i et j sont dans la même communauté. Une distance entre sommets est alors calculée à partir de ces vecteurs propres, cette distance étant ensuite utilisée dans un algorithme de clustering hiérarchique. Le nombre de vecteurs propres à considérer est à priori inconnu. Plusieurs calculs sont successivement effectués en prenant en compte différents nombres de vecteurs propres, et le meilleur résultat est retenu. Les performances de l'algorithme sont limitées par les calculs des vecteurs propres qui se fait en $O(n^3)$ pour une matrice creuse. Une amélioration de cette approche a été proposée et utilise une version normalisée de la matrice Laplacienne [20].

2.7.4 Autres approches

- **Une méthode d'optimisation par recuit simulé proposée par Guimerà et Amaral [51].**

  Cette méthode utilise une approche directe d'optimisation de la modularité par recuit simulé. Elle envisage des modifications aléatoires d'une partition, les probabilités de transition étant définies en fonction de l'amélioration de la modularité. Les transformations de partitions envisagées sont des fusions et divisions de communautés ainsi que des déplacements de sommets d'une communauté vers une autre.

- **Une méthode d'optimisation proposée par Duch et Arenas [54].**

  Cette méthode consiste à diviser le graphe en deux communautés et à diviser récursivement les deux communautés ainsi trouvées. Chaque étape part d'une

division arbitraire et des nœuds sont ensuite changés de communauté de façon à améliorer la modularité (mesurée comme une somme de participations élémentaires pour chaque nœud). Le nœud à déplacer est choisi aléatoirement (selon des probabilités adéquates) parmi les nœuds ayant les moins bonnes contributions à la modularité globale de la coupe. La coupe obtenant la meilleure modularité est retenue tout au long du processus.

- **Méthodes utilisant des marches aléatoires**

  Les marches aléatoires dans les graphes sont des processus aléatoires dans lesquels un marcheur est positionné sur un sommet du graphe et peut à chaque étape se déplacer vers un des sommets voisins. Le comportement des marches aléatoires est étroitement lié à la structure du graphe, ainsi plusieurs approches de détection de communautés se basent sur ces comportements.

- **Markov Cluster Algorithm de van Dongen [55].**

  Cette approche calcule des probabilités de transition entre tous les sommets du graphe en partant de la matrice de transition des marches aléatoires. Deux opérations matricielles sont successivement itérées. La première calcule les probabilités de transition par des marches aléatoires de longueur fixée **r** et correspond à une élévation de la matrice à la puissance r. La seconde consiste à amplifier les différences en augmentant les transitions les plus probables et en diminuant les transitions les moins probables. Les transitions entre sommets d'une même communauté sont alors favorisées et les itérations successives des deux opérations conduisent à une situation limite dans laquelle seules les transitions entre sommets d'une même communauté sont possibles. La complexité de l'algorithme est en O(n3).

- **Une méthode de Clique Percolation (CPM) proposée par Palla et al [25] :**

  Cette méthode consiste à détecter des communautés qui se chevauchent dans les réseaux (un nœud peut être un membre de plusieurs différentes communautés en même temps). CPM utilise l'information locale qui est la densité de liens. Les auteurs se sont basés sur l'observation qu'une communauté peut être interprétée comme union de plus petits sous graphes complets qui partagent des nœuds entre eux. De tels sous graphes complets dans un réseau s'appellent les k-cliques, où k représente le nombre de nœuds dans le sous graphe. Deux k-cliques seraient adjacentes si elles partagent (k-1) nœuds, et une

communauté est définie en tant que l'union de toutes les k-cliques qui peuvent être atteintes par une série de k-cliques adjacentes.

2.8 Discussion et conclusion

Toutes les approches de découverte de communauté qu'on vient de citer adoptent généralement une vue statique et globale du réseau. Or, dans un contexte distribué, il n'est pas envisageable de considérer une information globale sur tout le graphe d'autant plus qu'il devient de plus en plus nécessaire de capturer la dynamique du réseau. La plus part des réseaux sociaux sont dynamiques. Peu de travaux ont été effectués pour la découverte de communauté d'intérêt dans les réseaux mobiles.

On peut citer le travail **de Shu-Yan et Al**.[27] qui ont proposé un algorithme de détection de communautés dans ce contexte. Ils ont définit une communauté comme étant une série de k-cliques adjacentes où chaque nœud de la k-clique a une interaction forte avec les k-1 autres nœuds. Les interactions entres nœuds sont mesurés par leurs durée de contactes, ainsi, une interaction entre deux nœuds est dite forte si leurs durée de contacte dépasse un certain seuil λ. Pour étudier la dynamique des communautés sur une longue période de temps, ils partitionnent la trace de mobilités en des intervalles de temps plus petits, puis appliquent l'algorithme de détection de k-clique communautés (CPM de Palla présenté en haut) pour les données de chaque intervalle. A tout moment, les utilisateurs spécifient k et λ (seuil de durée contacte) pour calculer les structures de communautés actuelles.

Nous allons nous intéresser dans notre travail au contexte des réseaux **Ad-Hoc** sans fil (MANETS, pour Mobile Ad-Hoc Networks). Ainsi, le but de notre travail porte sur la découverte de communauté d'intérêts dans ces réseaux.

Dans le chapitre suivant, nous introduisons des notions sur les réseaux ad hoc et nous présentons quelques structures connues dans les graphes, qui nous seront utile par la suite.

Chapitre 3

Topologies virtuelles dans les réseaux mobiles Ad hoc

3.1 Introduction

Avec le déploiement de la norme IEEE 802.11 ces dernières années, les réseaux locaux sans fil connaissent un grand succès auprès des institutions et les réseaux ad hoc suscitent un réel intérêt auprès de la communauté de Recherche et Développement.

Plusieurs systèmes utilisent déjà le modèle cellulaire de réseaux sans fil, et connaissent une très forte expansion à l'heure actuelle. On peut par exemple citer les réseaux GSM, sans omettre l'exemple du projet hollandais NAFIN (Netherlands Armed Forces Integrated Network), qui a visé à améliorer les performances des forces militaires, en y intégrant la technologie des réseaux sans fil. L'inconvénient majeur du modèle cellulaire est qu'il requière une importante infrastructure logistique et matérielle fixe dont la réalisation peut se révéler coûteuse, voire impossible.

La contrepartie des réseaux cellulaires sont les réseaux mobiles ad hoc. Un réseau ad hoc est une collection d'entités mobiles interconnectées par une technologie sans fil formant un

réseau temporaire. Les entités constituant un réseau ad hoc, appelées également nœuds, s'organisent seules et sans l'aide de toute administration ou de tout support fixe. Elles forment ainsi un réseau autonome et dynamique ne reposant sur aucune infrastructure. En réalité, ce sont les entités mobiles elles-mêmes qui forment, d'une manière ad hoc, une infrastructure du réseau.

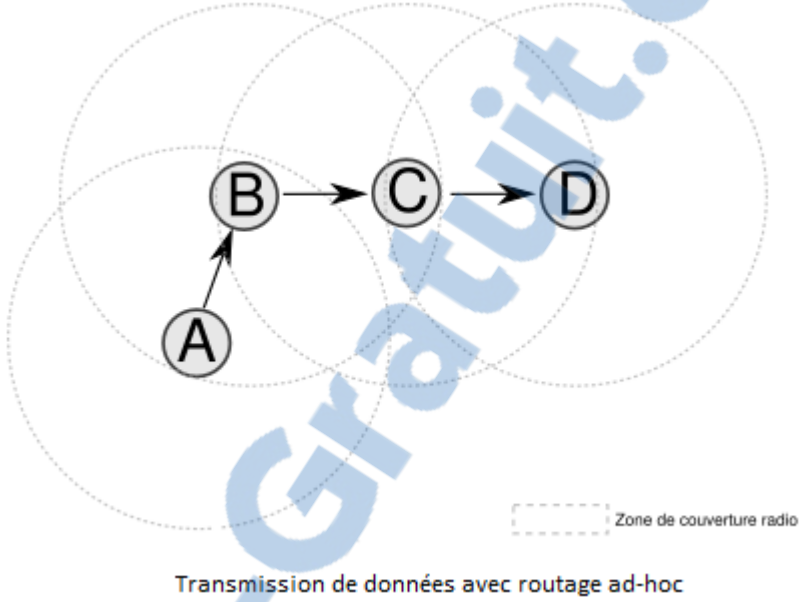

Fig.3.1 Réseau ad hoc

Les réseaux ad hoc sont par conséquent des réseaux sans fil capables de s'organiser sans infrastructure préalablement définie. Nous nous sommes intéressés dans ce travail aux réseaux ad hoc dans leur configuration mobile, plus connus sous le nom de MANET (pour *Mobile Ad-hoc NETworks*). L'intérêt de ce type de réseaux sans fil est mis en évidence lorsqu'ils permettent d'éviter le déploiement d'infrastructures dans des situations d'urgence ou temporaires.

Malgré l'absence d'infrastructure, les entités peuvent communiquer entre elles. En fait, chaque nœud communique directement avec son voisin. Pour communiquer avec d'autres nœuds, qui ne se trouvent pas dans sa portée, il lui est nécessaire de faire passer ses données à travers d'autres nœuds qui se chargeront de les acheminer (voir Fig.3.1). Pour cela, il est d'abord primordial que les entités se situent les unes par rapport aux autres, et soient capables de construire des routes entre elles : c'est le rôle du protocole de routage.

3.2 Définition

La RFC 2501 [3] définit un MANET comme une collection de plates-formes mobiles, appelés des nœuds, qui sont libres de se déplacer arbitrairement. Ces nœuds peuvent se trouver dans des lieux tels que les avions, les bateaux, les voitures, voire même sur des personnes.

Un MANET est un système autonome de nœuds mobiles. Ce système peut être isolé, mais il peut aussi avoir des passerelles ou des interfaces le reliant à un réseau fixe. Les nœuds des MANET sont équipés d'émetteurs et de récepteurs sans fil. A un instant donné, en fonction de la position des nœuds, de la configuration de leurs émetteurs-récepteurs, des niveaux de puissances de transmission et d'interférence entre les canaux, il peut exister une connexion sans fil entre les nœuds. Cette topologie ad hoc peut changer avec le temps en fonction du mouvement des nœuds ou de l'ajustement de leurs paramètres d'émission-réception.

3.3 Modélisation d'un réseau ad hoc

Un réseau ad hoc peut être modélisé par un graphe G(t) (*Vt*, *Et*), où :

- *Vt* représente l'ensemble des nœuds du réseau à l'instant *t ;*
- *Et* modélise l'ensemble des arêtes (connexions entre les nœuds), de sorte que s'il existe une arête, cela veut dire que les nœuds *u* et *v* sont en mesure de communiquer directement à l'instant *t.*

Le réseau de la figure 3.1 aura donc la modélisation suivante :

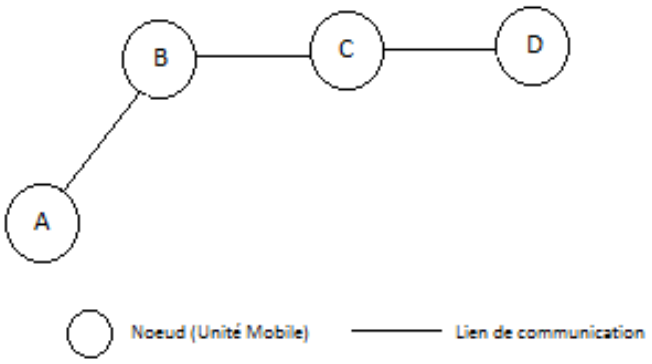

Fig.3.2. Modélisation d'un réseau ad hoc

Les nœuds sont libres de se déplacer aléatoirement et s'organisent arbitrairement. Par conséquent, la topologie du réseau peut varier de façon rapide et surtout imprévisible comme le montre la figure 3.3.

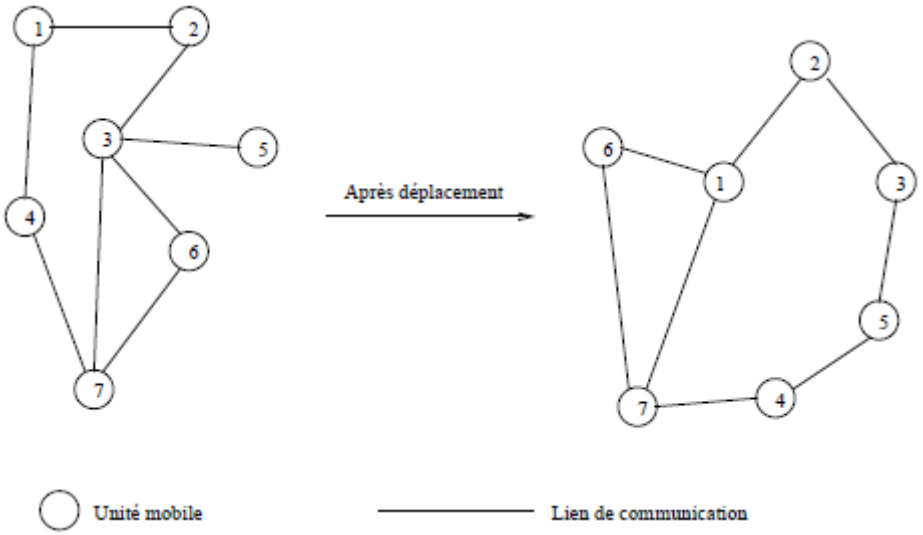

Fig. 3.3. Topologie dynamique d'un réseau ad hoc

3.4 Domaines d'application des réseaux ad hoc

Les applications des réseaux ad hoc sont nombreuses, l'exemple classique de leur application est dans le domaine militaire et les autres applications de tactique comme les opérations de secours et les missions d'exploration.

Cependant, avec l'avancement des recherches dans le domaine des réseaux et l'émergence des technologies sans fil, d'autres applications civiles sont apparues. On peut distinguer :

- **Les services d'urgence** : opération de recherche et de secours des personnes, tremblement de terre, feux, inondation, dans le but de remplacer l'infrastructure filaire.
- **Le travail collaboratif et les communications dans des entreprises ou bâtiments** : dans le cadre d'une réunion ou d'une conférence par exemple.
- **Home Network** : partage d'applications et communications des équipements mobiles.
- **Applications commerciales** : pour un paiement électronique distant (taxi) ou pour l'accès mobile à l'Internet, où service de guide en fonction de la position de l'utilisateur (GPS sur téléphone mobile) .

- **Réseaux de senseurs** (appareils dotés de mécanismes pour relever des informations sur son environnement): pour des applications environnementales (climat, activité de la terre, suivi des mouvements des animaux, etc.) ou domestiques (contrôle des équipements à distance).
- **Réseaux en mouvement** : informatique embarquée et véhicules communicants.
- **Réseaux Mesh ou Réseaux maillés** : c'est une technologie émergente qui permet d'étendre la portée d'un réseau ou de le densifier afin d'accroître la tolérance à d'éventuelles pannes ou interférences.

3.5 Caractéristiques et contraintes

Malgré leurs apports, les réseaux ad hoc héritent des mêmes propriétés et problèmes liés aux réseaux sans fil. En effet, le canal radio est limité en termes de capacité, plus exposé aux pertes (comparé au médium filaire), et sujet à des variations dans le temps : la superposition de toutes ces contraintes peut se révéler critique.

On peut tout d'abord citer les problématiques les plus courantes et classiques dont héritent les réseaux ad hoc des réseaux sans fil :

- Bande passante limitée : le partage du médium de communication fait que la bande passante allouée à un nœud est faible.
- Perte de données et de routes pouvant être générées, entre autres, par les interférences, les collisions et/ou la désactivation/mort d'un nœud se trouvant sur le chemin.
- Sécurité limitée : écoute du réseau et en particulier pour récupérer à la volée des informations sensibles lorsqu'elles ne sont pas chiffrées.
- Erreurs de transmission ou données erronées dues à rapport signal sur bruit excessivement faible.

D'autres caractéristiques spécifiques aux réseaux ad hoc conduisent à ajouter une complexité et des contraintes supplémentaires qui doivent être prises en compte lors de la conception des algorithmes et des protocoles de communication, à savoir :

- Absence d'infrastructure : Les nœuds mobiles sont responsables d'établir et de maintenir la connexité du réseau d'une manière continue.

- Interférences : Les liens radios ne sont pas isolés, deux transmissions simultanées sur une même fréquence ou sur des fréquences proches peuvent interférer.
- Topologie dynamique : les unités mobiles du réseau se déplacent de manière aléatoire et arbitraire. De ce fait, la topologie du réseau peut changer d'un instant à l'autre de manière imprévisible. Il faut donc définir des protocoles de construction et de maintien de topologies dans le temps.
- Des contraintes d'énergie : Les unités mobiles sont alimentées par des sources d'énergie autonomes comme des batteries dont la capacité est limitée. Par conséquent, ces équipements disposent d'une durée de traitement réduite. Il faut alors considérer avec le plus grand soin toute activité faite par l'hôte (telles que l'émission ou la réception de données, écoute du canal…) tant la durée de vie du réseau repose directement sur la durée de vie des entités qui le composent.

3.6 Topologies virtuelles dans les réseaux Ad hoc

Différents protocoles de communication, tels que le routage, le partage de ressources, découverte de services, de prédictions … utilisent l'inondation simple pour diffuser ou rassembler de l'information. Pour cette raison, des récents travaux tentent d'optimiser ces échanges de flux de données. Plusieurs études ont proposé de construire et de maintenir des topologies virtuelles dynamiques sur le réseau. Ceci permet d'obtenir une meilleure organisation du réseau. Les avantages d'utilisation de ces structures sont [28] :

- Diminution de l'impact de la mobilité. En effet, si une couche intermédiaire (une topologie virtuelle) peut offrir une certaine organisation du réseau, la conception des applications pourrait devenir plus facile(en considérant la mobilité).
- Optimisation de l'inondation. L'élection de quelques nœuds pour rediffuser les messages, diminue largement le cout de cette dernière.
- Amélioration du passage à l'échelle : une conséquence directe des deux points précédents : Une bonne organisation du réseau ne rend son passage à l'échelle que plus efficace.
- Réduction du temps de réponse.
- Prise en considération des mécanismes d'équilibrage de charge.

Nous présentons dans ce qui suit, les topologies virtuelles (notions de théorie de graphes) les plus utilisées dans les réseaux ad hoc et qui nous seront utiles pour résoudre notre problématique.

3.7 Classification des topologies virtuelles dynamiques dans les réseaux ad hoc

Les topologies virtuelles sont classées en deux grandes familles à savoir : ensembles de nœuds couvrants et structure basés liens. Cette classification à été proposée dans [32].

3.7.1 Ensemble de nœuds couvrants

Les topologies virtuelles de cette classe choisissent, généralement, un sous ensembles de nœuds qui vont êtres considérés comme backbone (ou noyau) du réseau. En d'autres termes, il s'agit de construire des ensembles de nœuds "privilégiés" pour assurer des tâches particulières selon le problème traité. Dans le cas de la découverte de communauté, ces nœuds doivent assurer les tâches de référencement (découverte) des nœuds de la communauté et acheminement des requêtes. Les nœuds ne faisant pas partie du backbone (ou nœuds normaux) interrogent les nœuds privilégiés afin de retrouver les membres de sa communauté. Les nœuds du backbone doivent alors coopérer entre eux pour satisfaire la requête. La détermination de ces ensembles de nœuds couvrant est la plus part du temps NP-difficile. Les topologies virtuelles les plus connus et se basant sur la notion de nœuds couvrant sont :

- **Ensemble multipoints relais (MultiPoint Relays, MPR)**

  Un ensemble MPR d'un nœud est l'ensemble minimum de ses voisins qui couvre tous les nœuds qui lui sont à distance 2.

- **Ensembles dominants(DS)**

  Un DS d'un graphe G = (V, E) est un sous ensemble V' de V tel que chaque nœud de V est soit dans V', soit adjacent à au moins un nœud de V'.

- **Ensemble Indépendant (Independent Set, IS)**

  Un IS d'un graphe est un ensemble de nœuds du graphe tel qu'aucune paire de nœuds de l'IS ne se trouvent adjacents dans le graphe. Soit IS un ensemble indépendant. Il est défini formellement comme suit : $\forall u \in IS, \neg (\exists v \in IS / v \in N(u))$.

- **Ensemble indépendant maximal (Maximal Independent Set, MIS)**

  Un MIS d'un graphe est un ensemble indépendant présentant une cardinalité maximale. Par ailleurs, un MIS ne possède pas dans le cas général la propriété d'unicité. Un MIS est aussi un ensemble dominant (DS) car tout nœud du graphe est soit dans l'ensemble MIS soit adjacent un aux moins un nœud du MIS. (La distance maximale entre un nœud dominant et son voisin dominant le plus proche est de 3).

- **Cluster**

  Un cluster est un sous-ensemble de nœuds qui partagent une propriété donnée et qui sont représentés par des cluster-heads (i.e. représentants). Dans le cas de la propriété de dominance, tous les autres nœuds du cluster sont adjacents au cluster-head.

- **Ensemble dominant connecté (Connected Dominating Set, CDS)**

  Un CDS d'un graphe est un ensemble de nœuds tel que tout nœud du réseau est voisin d'au moins un nœud du CDS, et tel que le CDS forme une structure connexe. Si V' est l'ensemble des nœuds du CDS, la définition plus formelle est :

  $\forall u \in V, \exists v \in V' / v \in N(u)$

  $\forall (u, v) \in V'^2, \exists c = \text{chemin}_{u \to v} / \forall w \in c, w \in V'$

3.7.2 Structures basées liens

Cette classe est basée sur les sous graphes du réseau : Contrairement à la première, cette classe se base sur la minimisation du nombre d'arêtes (liens) utilisées pour la communication. En minimisant le nombre des liaisons tout en gardant la connectivité du réseau, les diffusions des requêtes (pour les membres de la communauté) n'engendreront pas des problèmes de broadcast strom, et les congestions dues aux messages redondants diminueraient aussi. Les topologies les plus connues et se basant sur les liens du réseau sont :

- **Arbre couvrant (spanning tree, ST)**

  Soit G = (V, E) un graphe connecté. ST est un spanning tree de G lorsqu'il connecte tous les nœuds de G et il ne possède pas de cycle (i.e. qu'il respecte la définition d'arbre). ST = (V, E'). Notons n, le nombre de nœuds V, alors $|E'| = n - 1$. Si chaque lien porte un coût, il est possible de construire un **minimum spanning tree (MST),** minimisant le coût global des arêtes utilisées.

- **K-tree cores**

  Etant donné un arbre T, un K-tree core de T est définit comme un sous- arbre de T' avec exactement k feuilles minimisant la distance moyenne entre ce core et tous autres nœuds de l'arbre T. si k=2, alors le core T' est juste un chemin dans l'arbre.

- **(Degree-constrained spanning tree, DCST)**

  Un DCST d'un graphe G est un spanning tree de degré minimum.

- **K-arbre**

  Un k-arbre d'un graphe G est un spanning tree de G où le degré maximum des nœuds est limité à une certaine constante k.

- **Anneau (ring)**

  Un réseau est un anneau si chaque nœud a un degré exactement égal à deux. De plus il existe un seul cycle dans le réseau, Ce cycle relie chaque nœud à lui-même et contient tout les nœuds du graphe.

3.8 Conclusion

Après avoir présenté, dans ce chapitres, les réseaux ad hoc et les topologies virtuelles dans ces réseaux, nous tirons parti des avantages des deux grandes familles de topologies et proposons un nouvel algorithme de découverte de communauté se basant sur : l'ensemble indépendant maximal (MIS) qui représente l'ensemble de nœuds "privilégiés" pour assurer la tache de découverte des membres de communauté (backbone). Et l'arbre couvrant de degré limité (k-arbre) qui peut être utile pour organiser les nœuds du MIS reliés entre eux de façon à réduire les redondances ainsi diminuer les degrés de nœuds. Un k-arbre est construit et maintenu de façon distribuée. Lorsqu'une information à relayer est reçue par un nœud de l'MIS, il ne peut la relayer que via un lien faisant parti du k-arbre. Le nombre d'envois est optimisé et une information ne peut boucler. .

Chapitre 4

Algorithme distribué pour la découverte de communautés d'intérêts dans les réseaux Ad hoc

4.1 Introduction

Etant donné un réseau social distribué constitué d'unités ou hôtes mobiles (PDA, …) en interactions. Ces mobiles ont des centres d'intérêts (ou profils) différents et une capacité de mémoire (cache) limité. Les mobiles partageant le même centre d'intérêt forment un groupe social qu'on appelle une communauté d'intérêt. Ces communautés sont en générales de grandes tailles dont les membres sont référencés par les caches de ces mobiles. Ainsi, la question que l'on se pose est la suivante : Un mobile peut- il atteindre tous les membres de sa communauté via un parcours de cache(s) sachant que la taille de cette communauté est très supérieure à la taille de son cache ?

Donc, le but de notre travail porte sur la découverte de communautés d'intérêts dans un réseau social distribué (en particulier dans un réseau ad hoc) où chaque membre de ce réseau dispose d'un cache limité.

Pour ce faire, nous avons proposé un algorithme distribué basé sur les notions de nœuds indépendants et d'arbres couvrants. .

4.2 Modélisation et formulation du problème

On modélise le système par un graphe non orienté connexe G = (V, E) où V est l'ensemble des nœuds (i.e. les unités ou les hôtes mobiles) du réseau et E modélise l'ensemble des connexions qui existent entre ces nœuds. Une arête (u, v) ∈ E si et seulement si u et v peuvent mutuellement recevoir les transmissions de u et v. Ce qui implique que les liens entre les nœuds sont bidirectionnels. Dans ce cas, on dit que u et v sont voisins.

On définit la distance entre deux nœuds u et v dans le graphe G comme le nombre d'arêtes du plus cours chemin entre u et v.

L'ensemble des voisins directs d'un nœud u ∈ V est noté N(u) et celui à deux sauts est noté par $N^2(u)$.

$$N\,2(u) = \{v, dist(u,v) <= 2\}$$

Chaque nœud u du réseau à un identifiant unique id, un centre d'intérêt P appartenant à une classe de profil, et peut communiquer avec ses voisins à deux saut $N^2(u)$.

Nous supposons que chaque nœud a des connaissances sur ses voisins directs telles que leurs profils et leurs états locaux.

Définition (communauté d'intérêt) On définit une communauté d'intérêt **CI** d'un nœud u par l'ensemble des nœuds du graphe G ayant leur centre d'intérêt dans la même classe de profil que lui.

$$CI(u) = \{v \in G\,(V,E), \quad P\bullet(u) = P\bullet(v)\}$$

4.3 Proposition

L'idée principale de notre algorithme consiste à construire une topologie virtuelle permettant de résoudre le problème de découverte de communauté dans les réseaux mobiles. Les nœuds de cette topologie représentent le backbone de la communauté. Pour ce faire, nous avons proposé un algorithme distribué constitué de trois étapes principales (voir Fig.4.1) :

La première étape consiste à trouver un ensemble MIS (Maximal Independent Set) du graphe. Cette étape permet de retrouver l'ensemble maximal de nœuds qui couvrent tous les nœuds du graphe indépendamment de leurs profils (ou appartenances aux communautés).

La deuxième étape consiste à connecter uniquement les nœuds de l'ensemble MIS qui sont soit des nœuds de la même communauté, soit ils couvrent au moins un nœud de cette

communauté en créant des liens virtuels entre eux. On désigne par **graphe de couverture de la communauté CI_R,** la topologie virtuelle crée pour la Communauté d'Intérêt R. Ainsi, pour chaque communauté d'intérêt, on va construire son graphe de couverture.

La troisième étape de notre algorithme permet de résoudre le problème de cache limité des nœuds (degré des nœuds dépasse la taille du cache) en construisant pour chaque graphe de couverture de la communauté $\mathbf{CI_R}$ créé à l'étape précédente son k-arbre couvrant. Les nœuds de cet arbre couvrent des éléments de la communauté, et leurs degrés ne dépassent pas la taille limite des caches. On note cet arbre par **k-arbre couvrant de la communauté CI_R.**

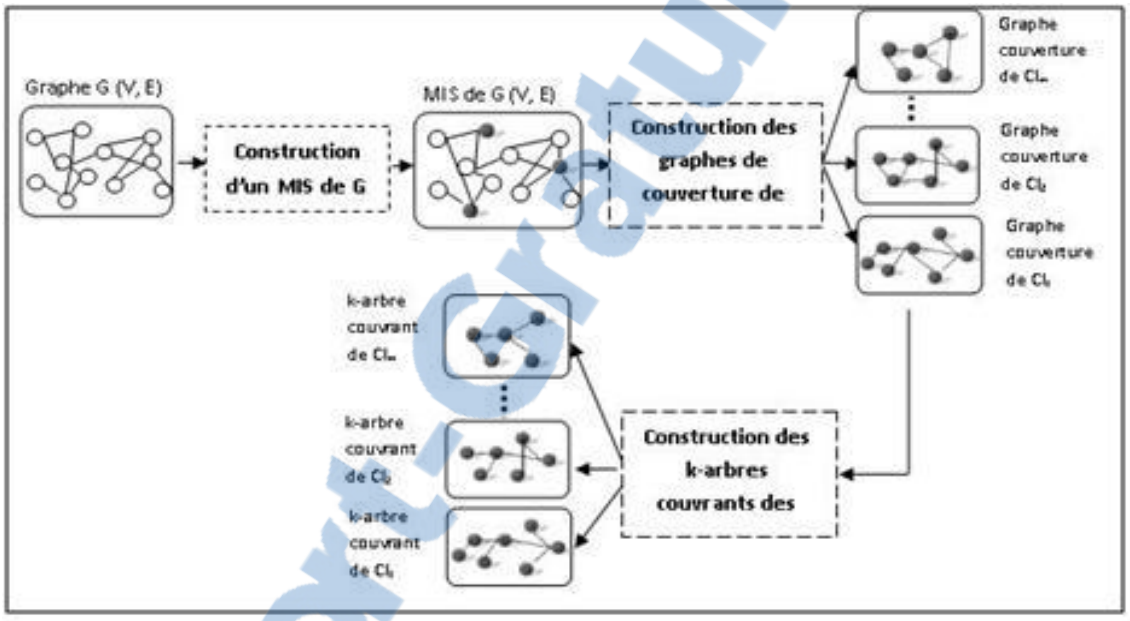

Fig.4.1. Schéma général du processus de découverte de communautés d'intérêts.

Avant de détailler les différentes étapes de notre algorithme, nous donnons l'ensemble de variables que possèdent les nœuds du réseau.

Chaque nœud u du réseau possède les variables suivantes :

- **ID(u) :** désigne identifiant de u, ID(u) ∈[1, n], avec n=|V|.
- **P(u) :** désigne le centre d'intérêt ou le profil du nœud u. P(u) ∈[1, m], avec m le nombre de classes de profils différentes du graphe G.
- **N(u) :** désigne l'ensemble des voisins direct du nœud u.
- **N^2(u) :** désigne l'ensemble des voisins à deux sauts du nœud u.
- **M(u) :** variable indiquant si le nœud u appartient à l'ensemble MIS(G) du graphe G ou non .

- **Deg(u) :** nombre de nœuds appartenant à l'ensemble MISA(u) (MIS Adjacent) qui sont voisins à u : $Deg(u) = |MISA(u)|$.
- **Cache(u) :** cache de taille K du nœud u. (ou taille m x k, où m : nombres de communautés). Ce cache permet de sauvegarder des informations pertinentes permettant de retrouver les membres de la communauté de u. Les informations qu'on trouve dans chaque ligne du cache sont :
  - **N-Ref :** le nœud référencé par u (N-Ref ∈ MIS(G)).
  - **N-Dest :** Le nœud qui mène vers N-Ref, avec N-Dest ∈ N(u), et le chemin entre les nœuds u et N-Ref doit être le plus court chemin entre eux en passant par N-Dest.
  - **Com :** variable booléenne indiquant si N-Ref fait parti de la communauté d'intérêt de u.
  - **Dist :** distance entre le nœud u et le nœuds référencé N-Ref.

  Ci dessous, un exemple d'une ligne du cache d'un nœud u :

Table 1. exemple d'une Ligne du cache d'un nœud u

u	N-Ref	N-Dest	Com	Dist
	3	2	1	3

Le nœud u (Table 1) référence le nœud 3 (N-Ref= 3) qui fait parti de la communauté recherchée (com=1), se trouvant à distance 3 (Dist= 3) de lui en passant par le nœud 2 (N-Dest=2).

4.3.1 Construction de l'ensemble MIS du graphe G

Plusieurs méthodes ont été proposées pour construire un ensemble MIS d'un graphe. Ces méthodes consistent à extraire un ensemble maximal de nœuds noté MIS tel qu'aucune paire de nœuds de l'ensemble MIS ne se trouvent adjacents dans le graphe, et aucun ensemble indépendant du graphe ne le contient (ensemble indépendant de cardinalité maximale).

Pour construire cet ensemble, on s'inspire de l'algorithme de Goddar et al. [29] qui est un algorithme auto stabilisant en un temps polynomial, ce qui est une bonne propriété dans le contexte des réseaux Ad hoc.

L'Auto-stabilisation est un paradigme pour les systèmes distribués qui permet au système d'atteindre, sans intervention extérieure, un état global correct à partir d'un état

quelconque. Ainsi, ces systèmes peuvent partir d'une configuration quelconque pouvant contenir des incohérences introduites par des fautes survenues dans le système et retrouver un comportement correct au bout d'un temps fini.

L'algorithme auto-stabilisant pour la construction de l'ensemble MIS du graphe G est illustrée ci-dessous. Chaque nœud u a une variable binaire unique M (u).

La règle 1 permet à un nœud de modifier sa valeur si tous les nœuds dans son voisinage fermé (ensemble de ces voisins y compris lui-même) ont la même valeur.

La règle 2 permet de modifier sa valeur s'il existe au moins un nœud de son voisinage qui a la même valeur que lui.

Algorithme 1 Algorithme de construction D'un MIS de G.

R_I.) **if** $M(u) = 0 \wedge (\forall v \in N(u), M(v) = 0)$ **then**

$M(u) = 1$

R_{II}.) **if** $M(u) = 1 \wedge (\exists v \in N(u), M(v) = 1)$ **then**

$M(u) = 0$

Apres stabilisation de l'algorithme :

L'ensemble des nœuds $\{u, M(u) = 1 \wedge (\forall v \in N(u), M(v) = 0)\}$ forme un MIS de G.

L'ensemble des nœuds $\{u, M(u) = 0 \wedge (\exists v \in N(u), M(v) = 1)\}$ forme l'ensemble des nœuds dominés (couverts).

L'exemple suivant illustre une exécution de l'algorithme1.

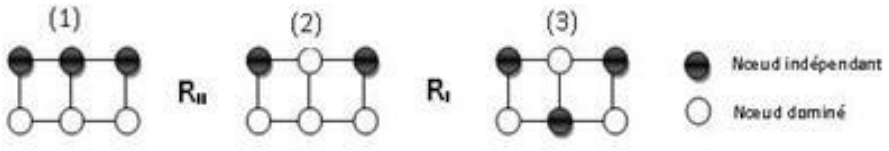

Fig.4.2. Graphes (1)-(3) représentent une exécution de l'algorithme 1

En exécutant R_I un nœud devient indépendant.

Par l'exécution de R_{II}, un nœud devient dominé.

4.3.2 Construction de la topologie virtuelle : graphe de couverture de la communauté CI_R

Une fois l'ensemble MIS du graphe G est construit, la deuxième étape de notre algorithme consiste à construire une topologie virtuelle représentée par un graphe G'(V', E')

où V' représente les nœuds de V appartenant à l'ensemble MIS de G et qui couvrent des nœuds de la communauté d'intérêt recherchée noté CI_R. Une arête existe entre deux nœuds s'ils sont MIS adjacents.

Définition (MIS adjacent) : on défini l'ensemble MIS adjacent **MISA(u)** d'un nœud u par l'ensemble des nœuds du MIS(G) qui lui sont à distance au plus 3.

$$MISA(u) = \{v \in MIS(G), Dist(u, v) <= 3\}.$$

A fin de créer cette topologie virtuelle qui couvre les éléments de la communauté ou le graphe de couverture de la communauté CI_R, nous avons proposé l'algorithme distribué dont le principe est le suivant (l'algorithme détaillé est illustré dans **Algorithme 2**, à la fin de ce chapitre):

Chaque nœud u de l'ensemble MIS envoie un message Req à ses voisins à distance 2 pour découvrir les nœuds de la communauté recherchée CI_R.

Chaque message Req transmis par un nœud contient son identifiant id, et la communauté recherché CI_R .

Nous supposons qu'un message envoyé par un nœud est reçu correctement en un temps fini par tous ses voisins à distance 2.

A la réception de ce message par un nœud $v \in N^2(u)$, ce nœud doit répondre en envoyant un message RRep à u contenant des informations selon les cas suivant :

- **Cas 1:** le nœud v appartient à l'ensemble MIS(G). Dans ce cas, trois configurations possibles se présentent :
  - Soit ce nœud appartient à la communauté recherchée (on note ce cas par **R2.1.a.** , voir Algorithme 2) , donc il répond en envoyant un message RRep contenant son id, une information indiquant qu'il appartient à la communauté C(v)=1, une information indiquant que c'est un nœud de l'MIS noté LM(v)= {(v, 1)} (LM(v) est une liste contenant des couples, la première valeur de chaque couple représente l'identifiant d'un nœud de l'MIS et la seconde indique l'appartenance de ce nœud à la communauté), et la liste de ses voisins directe qui font partie de la communauté LV(v)= $\{w \in N(v), P(w)= CI_R \}$. La figure suivante illustre ce cas :

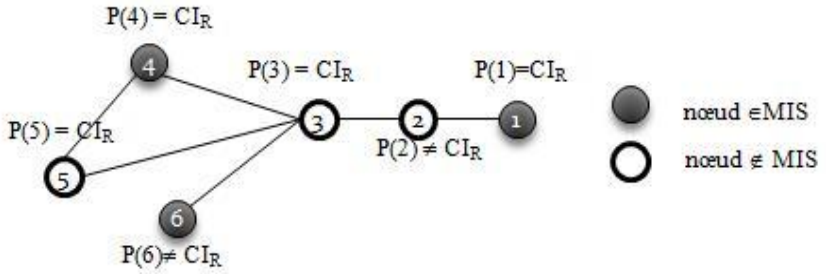

Fig. 4.3. Réponse d'un Nœud appartenant à MIS(G) et à la communauté.

Lorsque le nœud 3 reçoit le message Req (1, CI_R) du nœud 1, il doit lui répondre en envoyant le message RRep (3, C=1, LM= {(3,1)}, LV= {4, 5}).

- Soit ce nœud n'appartient pas à la communauté recherché mais il couvre des nœuds de cette communauté (**R2.1.b.**), alors il sert de relais et doit répondre (Fig. 4.4) en envoyant un message RRep contenant son id, une information indiquant que c'est un nœud de l'MIS, une information indiquant qu'il n'appartient pas à la communauté, et la liste de ses voisins directe qui font partie de la communauté.

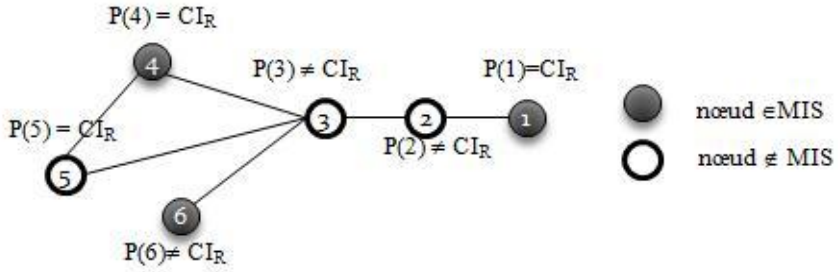

Fig.4.4. Réponse d'un Nœud appartenant à MIS(G), n'appartient pas à la communauté mais couvre ses éléments.

Lorsque le nœud 3 reçoit le message Req (1, CI_R) du nœud 1, il doit lui répondre en envoyant le message RRep (3, C=0, LM= {(3,1)}, LV= {4,5}).

- Soit ce nœud n'appartient pas à la communauté recherché et il ne couvre aucun nœud de cette communauté (**R2.1.c.**), alors il répond (Fig. 4.5) en envoyant un message RRep vide.

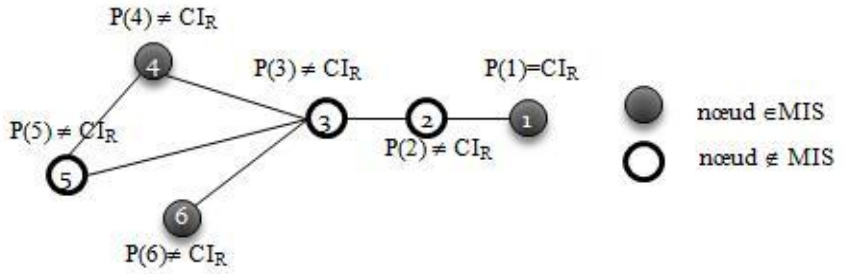

Fig. 4.5. Réponse d'un Nœud appartenant à MIS(G), n'appartient pas à la communauté et ne couvre aucun de ses éléments.

Lorsque le nœud 3 reçoit le message Req (1, CI_R) du nœud 1, il doit lui répondre en envoyant le message RRep() vide.

- **Cas 2 :** Le nœud n'appartient pas à l'ensemble MIS(G), Trois autres configurations se présentent aussi :

  - Soit ce nœud appartient à la communauté recherchée (**R2.2.a.**), or, il est clair que ce nœuds est couvert par au moins un nœud de l'ensemble MIS, ainsi, il répond en envoyant un message RRep contenant la liste de ses voisins directes qui font partie de l'ensemble MIS noté LM.

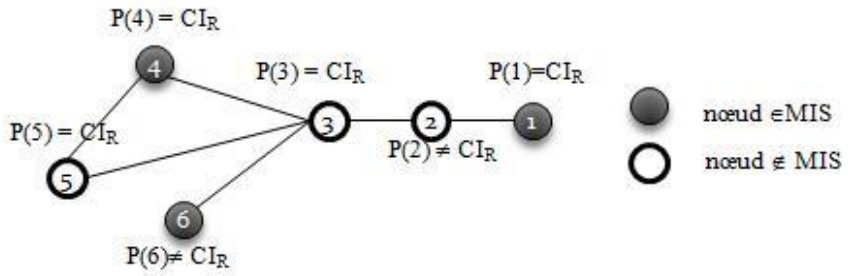

Fig. 4.17. Réponse d'un Nœud n'appartenant pas à MIS(G), mais appartient à la communauté.

Lorsque le nœud 3 reçoit le message Req (1, CI_R) du nœud 1, il doit lui répondre en envoyant le message RRep (3, C=1, LM= {(4,1), (6,0)}, LV= {4,5}).

  - Soit ce nœud n'appartient pas à la communauté, mais il existe des nœuds de l'ensemble MIS qui le couvrent et qui appartiennent à la communauté (**R2.2.b.**), ainsi il doit répondre en envoyant un message RRep contenant la liste de ces nœuds.

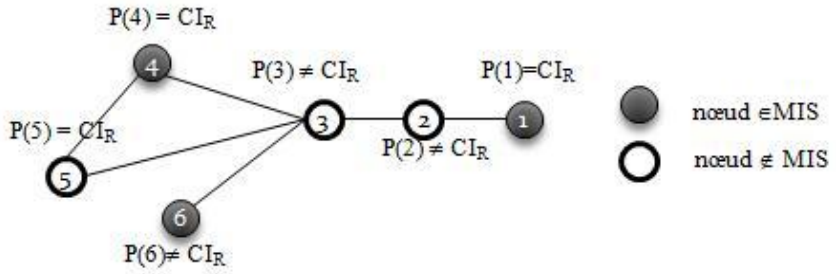

Fig. 4.7. Réponse d'un nœud n'appartenant ni à MIS(G), ni à la communauté mais couvert par des nœuds du MIS(G) appartenant à la communauté.

Lorsque le nœud 3 reçoit le message Req (1, CI_R) du nœud 1, il doit lui répondre en envoyant le message RRep (3, C=0, LM= {(4,1)}, LV= {4,5}).

- Soit ce nœud n'appartient pas à la communauté, et il n'est couvert par aucun nœud de l'ensemble MIS qui fait partie de la communauté (**R2.1.c.**), alors, il répond en envoyant un message RRep vide.

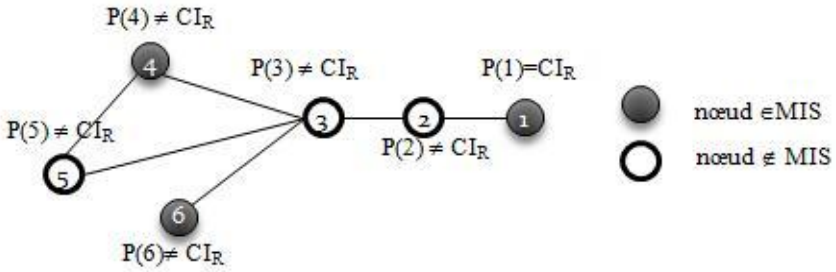

Fig. 4.8. Réponse d'un nœud n'appartenant ni à MIS(G), ni à la communauté non couvert par des nœuds du MIS(G) appartenant à la communauté.

Lorsque le nœud 3 (Fig.4.8) reçoit le message Req (1, CIR) du nœud 1, il doit lui répondre en envoyant le message RRep() vide.

Maintenant, à la réception du message RRep, le nœud doit faire des mises à jour locales. Il doit exécuter une procédure d'enregistrement des informations collectées dans son cache et mettre à jours son degré. Donc, le nœud vérifie si les informations contenues dans le message n'existent pas déjà dans son cache, alors il met à jour ce dernier ainsi que son degré sinon il ignore ce message.

Par exemple à la réception du message RRep (3, C=1, LM= {(3,1)}, LV= {4, 5}) par le nœud 1, il va enregistrer ses informations dans son cache :

Cache(1).N-Ref = 3, Cache(1).N-Dest = 2, cache(1).c = 1 .

A la fin du processus, chaque nœud de l'ensemble MIS aura construit une topologie virtuelle locale représentant l'ensemble des MIS adjacents qui couvrent les membres de la communauté recherché CI_R. Autrement dit, chaque nœud de l'ensemble MIS(G) découvre tout les nœuds de son ensemble MIS adjacent qui sont soit eux même des membres de la communauté recherchée, soit ils couvrent des membres de cette communauté.

L'union de toutes les topologies locales forme la topologie virtuelle globale représentant le graphe de couverture de la communauté CI_R (backbone de la communauté CI_R).

Les degrés des nœuds de ce graphe peuvent être supérieur à la taille limite (k) du cache (c.-à-d. pour un nœud u, le nombre de nœuds de MISA(u) qui couvrent les éléments de sa communauté dépasse la taille de son cache), ainsi, La troisième étape de notre algorithme consiste à résoudre le problème du cache limité en construisant un k-arbre couvrant de ce graphe.

4.3.3 Construction du K-arbre couvrant du graphe de couverture de communauté CI_R

Pour construire le k-arbre couvrant du graphe de couverture de la communauté CI_R, (avec k-arbre : est un arbre de degré maximum au plus k), nous nous somme inspiré de l'algorithme auto stabilisant proposé par Blin et Al.[30] permettant de construire un arbre couvrant de degré minimum (MDST). Cet algorithme permet de construire un arbre $T = (V_T, E_T)$ couvrant un graphe G(V, E), tel que le degré maximum des nœuds dans T soit le minimum possible, c'est-à-dire $deg(T) = \min\{deg(T') : T' \text{ arbre couvrant de } G\}$, avec $deg(T) = \max\{deg(v) : v \in V_T\}$ représente le degré maximum d'un arbre T.

Nous allons présenter dans cette section le principe du fonctionnement de l'algorithme de construction du k-arbre couvrant.

Soit G = (V, E) un graphe non-orienté connexe (graphe de couverture de la communauté CI_R) et T un arbre couvrant de G.

Soit e = {u, v} une arête ne faisant pas partie de l'arbre T.

L'ajout de e à T crée un cycle dit *élémentaire* associé à e, notée C_e, et on échangeant une arête de C_e dans l'arbre et l'arête e donne un nouvel arbre couvrant de G. La Fig.4.9. (a) donne un arbre couvrant de G (arêtes pleines) avec un unique cycle élémentaire obtenu par l'ajout de e = {u, v} (arête en pointillée) à l'arbre.

Soit w un nœud dans le cycle élémentaire C_e, différent de u et v, de degré maximum ($deg_T(w) = deg(T)$).

Si l'échange entre e et une arête de C_e adjacente a w réduit de un le degré de w, alors l'arête e est dite ***arête améliorante***. Les deux nœuds constituants cette arête, doivent avoir leurs degrés inferieurs d'au moins deux par rapport au degré maximum de l'arbre.

De façon formelle (voir Fig.4.9), soit e = {u, v} une arête n'appartenant pas à l'arbre couvrant T. Soit w un nœud du graphe, tel que $w \in Ce$ et $degT(w) = deg(T)$. L'arête e est améliorante [30] si et seulement si

$$\mathrm{degT}\,(w) \geq \max(\mathrm{degT}\,(u), \mathrm{degT}\,(v)) + 2$$

Si u et/ou v ne vérifient pas les conditions de l'équation précédente, ils sont dits *bloquants*.

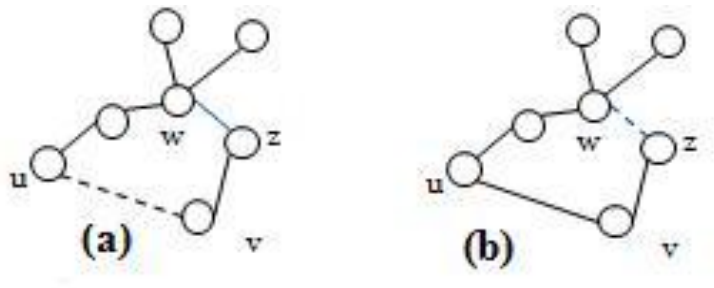

Fig. 4.9. Echanges d'arêtes dans un cycle élémentaire pour réduire le degré maximum d'un arbre couvrant.

On reprend l'algorithme proposé dans [30] et nous l'adaptons à notre cas pour résoudre le problème du k-arbre couvrant. Cet algorithme est composé de plusieurs modules auto-stabilisants listés ci-dessous :

1. Construction et maintient d'un arbre couvrant le graphe de couverture de communauté CI_R.
2. Calcul du degré maximum de l'arbre courant.
3. Si le degré maximum de l'arbre courant > k (taille limite des caches) alors,
   - Calcul du cycle élémentaire des arêtes ne faisant pas partie de l'arbre mais qui appartiennent au graphe G.
4. Réduction du degré maximum de l'arbre couvrant courant.

Le fonctionnement général de l'algorithme auto-stabilisant est le suivant :

On considère que chaque nœud possède un identifiant unique. A tout moment, un arbre couvrant est maintenu. Si cet arbre couvrant n'existe pas, il est construit en utilisant les identifiants des nœuds pour sélectionner un nœud racine et la distance de chaque nœud à la racine pour éliminer les cycles .

Le degré maximum de l'arbre est calculé par récupération de l'information des feuilles vers la racine, puis par propagation du résultat dans l'arbre.

Chaque nœud a des connaissances sur le statut de ses arêtes sortantes, à savoir, si elles appartiennent à l'arbre couvrant courant ou non. Chaque arête n'appartenant pas l'arbre va initier un processus de détection de son cycle élémentaire. Cette détection se fait grâce à un parcours en profondeur de l'arbre. De plus, ce processus détecte le degré maximum du cycle élémentaire.

Soit e une arête ne faisant pas partie de l'arbre couvrant et soit C_e son cycle élémentaire. Si le degré maximum dans C_e est égal au degré maximum de l'arbre et si l'arête vérifie les conditions pour être améliorante, alors l'arête améliorante e est échangée avec une arête adjacente à un nœud de degré maximum dans C_e.

- **Construction et maintient d'un arbre couvrant**

Nous avons adapté l'algorithme auto-stabilisant construisant un arbre en largeur (BFS tree : Breadth First Search tree) proposé par Afek, Kutten et Yung dans [31]. Ce module utilise les variables suivantes lorsqu'il est exécuté sur un nœud v : l'identifiant de la racine de l'arbre ($root_v$), un pointeur vers le parent de v dans l'arbre ($parent_v$) et la distance dans l'arbre entre v et la racine ($distance_v$). La racine de l'arbre retourné par ce module est le nœud ayant le plus petit identifiant dans le réseau.

Deux règles sont utilisées dans ce module, la première règle permet de changer le parent d'un nœud et la deuxième permet de réinitialiser l'état d'un nœud. Plus précisément, lorsqu'un nœud v possède un voisin u avec un identifiant de nœud racine plus petit que celui de v, alors v change l'identifiant de sa racine pour celle de u. Ensuite, v choisit u comme son nouveau parent dans l'arbre (si plusieurs nœuds voisins vérifient la condition énoncée alors v prend comme parent le nœud d'identifiant le plus petit). En revanche, si un nœud v détecte une incohérence dans son état par rapport à l'état de ses voisins, alors v utilise la deuxième règle pour se déclarer comme nouvelle racine. De plus, v se choisit comme racine et parent, et met sa distance à 0. L'état d'un nœud v est cohérent par rapport à l'état de son voisinage si v a comme parent un nœud dans son voisinage ayant la même racine et a une distance égale à celle de son parent plus un (un nœud racine a une distance à 0).

Nous allons détailler, dans cette partie, le processus de réduction du degré d'un nœud ayant un degré maximum ou bloquant dans l'arbre. Pour cela, nous donnons le principe de fonctionnement des différentes procédures permettant de le faire :

- **Cycle Search :** périodiquement chaque arête ne faisant pas partie de l'arbre courant recherche son cycle élémentaire. Lorsqu'une arête e = {u, v}, ne faisant pas partie de l'arbre a découvert son cycle élémentaire, celle-ci peut déterminer si elle est améliorante ou non à l'aide de la procédure Action on Cycle.
- **Action on Cycle :** si un cycle élémentaire ne contient pas de nœud de degré maximum, alors aucune amélioration n'est possible dans ce cycle élémentaire. Sinon si le cycle élémentaire ne contient aucun nœud bloquant, une amélioration est possible. Dans le cas contraire où il y aurait un nœud bloquant, la procédure Deblock est appelée pour réduire le degré du nœud bloquant, soit on appelant la procédure Improve soit en rappelant récursivement la procédure Deblock pour un nouveau nœud bloquant rencontré.
- **Improve :** la procédure Improve envoie un message de type Remove relayé le long du cycle élémentaire. Lorsque ce message atteint sa destination, c'est-à-dire l'arête e′ = {w, z}, le degré des nœuds extrémités de e′ et son statut (arête de l'arbre ou non) sont vérifiés. Si l'arête e′ n'est plus dans l'arbre ou n'est plus adjacente à un sommet de degré maximum, alors ce message est ignoré et supprimé du réseau. Sinon, l'arête e′ est supprimée afin de réduire de un le degré d'au moins un nœud de degré maximum, ainsi, le statut de l'arête e′ est modifié (elle ne fait plus partie de l'arbre). Ensuite, la procédure Reverse Orientation est appelée afin de modifier les liens de parenté dans une partie du cycle élémentaire. Un message de type Remove ou Back est envoyé dans une partie du cycle élémentaire afin d'inverser l'orientation des pointeurs parent dans une partie du cycle élémentaire. Lorsque le message de type Remove ou Back a atteint l'arête e initiatrice de l'amélioration dans le cycle élémentaire, alors e est ajoutée à l'arbre couvrant.
- **Reverse Orientation :** après la suppression de l'arête e′, l'orientation d'une partie du cycle élémentaire doit être modifiée sinon après l'ajout de l'arête e (initiatrice de l'amélioration) on n'obtiendrait plus un arbre couvrant correctement orienté. La réorientation est réalisée grâce la circulation d'un message de type Remove ou Back. La procédure Reverse Orientation supprime l'arête e′ et vérifie suivant l'orientation de e′ si un message de type Remove ou Back doit être envoyé pour corriger l'orientation dans le cycle élémentaire. Si e′ a une orientation opposée à la direction suivie par le message Remove, alors un message de type Remove est utilisé, sinon c'est un message de type Back qui est utilisé.

  Deblock : pour réaliser l'échange d'une arête ayant comme extrémité un nœud bloquant w ∈ {u, v} (avec e = {u, v} l'arête initiatrice d'un processus d'amélioration), l'algorithme

commence par essayer de réduire de un le degré du nœud w. Pour réaliser cette tache, w diffuse dans tout son sous-arbre un message de type Deblock contenant son identifiant. Lorsqu'un descendant w′ de w, adjacent à une arête f ne faisant pas partie de l'arbre, reçoit un message de type Deblock celui-ci démarre la détection du cycle élémentaire C_f associé à f comme décrit ci-dessus pour une arête améliorante. Si le cycle élémentaire C_f permet de réduire le degré du nœud bloquant w, alors un échange d'arêtes est réalisé dans C_f pour rendre w non bloquant (comme si w était un nœud de degré maximum). Ainsi, cela permet de réaliser un échange d'arêtes dans le cycle C_e associé à l'arête e qui ne fait pas partie de l'arbre. Cependant, si le cycle élémentaire C_f ne permet pas de réduire le degré du nœud bloquant w, alors la procédure continue récursivement en diffusant un message de type Deblock avec l'identifiant du nœud w′.

La figure suivante illustre le cycle des appels à ces procédures durant le processus de réduction du degré maximum d'un arbre couvrant.

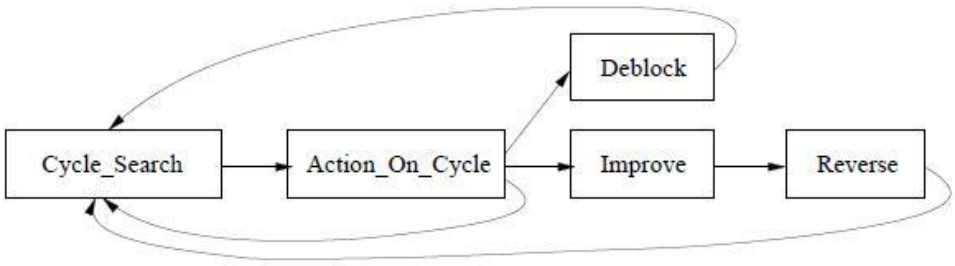

Fig. 4.10. Processus de réduction du degré maximum d'un arbre couvrant.

A la fin du processus de construction de l'arbre couvrant, on aura construit un arbre couvrant de degré soit égal à k, (problème résolu) soit le degré de l'arbre est toujours supérieur à k, ce qui veut dire qu'il n'ya aucune arête du graphe G qui est améliorantes, ainsi, on va s'appuyer sur des arêtes virtuelles pour la réduction des degrés. L'ajout de ces arêtes est fait de proche en proche.

Maintenant, pour chaque nœud désirant atteindre les membres de sa communauté, ce nœud est couvert par au moins un des nœuds de l'ensemble MIS(G), qui est soit de sa communauté soit son cache correspondant à cette communauté n'est pas vide (chaque nœud a autant de cache limité que de communautés), ainsi, il suffit de rajouter le nœud trouvé à son cache.

Le nœud trouvé appartient au graphe de couverture de la communauté CI_R, plus précisément, au k-arbre couvrant du graphe de couverture de la communauté, ainsi, on est sur d'atteindre tous les membres de la communauté en parcourant cet arbre, puisque les nœuds de ce dernier sont, soit des membres, soit couvrent des membres de la communauté.

Nous donnons dans ce qui suit, l'algorithme distribué détaillé qu'on a proposé dans la deuxième étape du processus de découverte de communauté. Cet algorithme permet de construire le graphe de couverture de communauté CI_R :

Algorithme 2 Algorithme de découverte de communauté sur un nœud u

ID: identifiant de u.
P: profil du nœud u.
N : voisins direct du nœud u.
N^2 : voisins à deux sauts du nœud u.
M : variable indiquant si le nœud u appartient à l'ensemble MIS(G) ou non.
Deg: degré du nœud u = |MISA(u)|.
Cache: cache du nœud u.

R1) **if** (M = 1) **then**
envoie Req(id, CI_R) à N^2(u)
R2) À la Réception d'un Req(id_v, CI_R)
R2.1) **if** (M=1) **then**
R2.1.a) **if** (P= CI_R) **then**
C=1
LM= {(u, 1)}
LV= {w ∈ N(u), P(w)= CI_R }
envoie RRep (id, C, LM, LV) à id_v
end if
R2.1.b) **if** (P≠ CI_R) ∧ (∃ w ∈ N(u), p(w)= CI_R) **then**
C=0
LM= {(u, 1)}
LV= {w∈ N(u), P(w)= CI_R }
envoie RRep (id, C, LM, LV) à id_v
end if
R2.1.c) **if** (P≠ CI_R) ∧ (∀w ∈ N(u), p(w) ≠ CI_R) **then**
C= 0
LM= {}
LV= {}
envoie RRep (id, C, LM, LV) à id_v

end if
end if
R2.2) If (M≠1) **then**
R2.2.a) If (P= CI_R) **then**
C=1;
LM= {(w, C(w)), w∈N(u) ∧ M(w)=1} ;
LV= {w∈N(u), P(w)=CIR}.
envoie RRep (id, C, LM, LV) à id_v ;
end if
R2.2.b) if (P≠ CI_R) ∧(∃ w ∈N(u), M(w)=1∧ P(w)= CI_R) **then**
C=0;
LM= {(w, 1), w ∈N(u), M(w)=1∧ P(w)= CI_R } ;
LV= {w∈N(u), P(w)= CI_R }.
envoie RRep (id, C, LM, LV) à id_v ;
end if
R2.2.c) if (P≠ CI_R) ∧(∀w ∈N(u), M(w)=1∧ p(w)≠ CI_R) **then**
C= 0;
LM= {};
LV= {} ;
envoie RRep (id, C, LM, LV) à id_v ;
end if
end if
R3) À la reception d'un RRep (id, C, LM, LV)
R3.1) if (LM={}) ∨ (∀ (v, C (v) ∈ LM, v∈ cache) **then**
ignorer RRep();
end if
R3.2) if (∃ (v, C (v)) ∈ LM, v∉ cache **then**
cache.N-Ref= v;
chache.N-Dest= w ∈ N(u), Dist(w, v) est minimale;
cache.Com=C(v) ;
Deg= Deg+1 ;
end if ;
R4) if M≠1 then
If (∃ v ∈ N(u), M(v)=1) ∧ ((p(v)= CI_R) ∨ (cache(v) ≠ {}) **then**
cache.N-Ref= v;
cache. N-dest=v ;
cache.Com=C(v) ;
Deg=Deg+1;
end if
end if

4.4 Exemple de déroulement

Pour décrire notre algorithme, nous donnerons dans cette partie un exemple illustrant le processus de découverte de communauté d'intérêt que nous avons proposé. Pour cela, nous définissons un graphe G(V, E) modélisant un réseau mobile (Fig.4.11), où chaque nœud est doté d'un cache limité à k(dans notre exemple, k=2).

4.4.1 Génération du graphe physique

Les nœuds du graphe représentent les hôtes mobiles du réseau: 25 nœuds.
Chaque nœud a un identifiant (représenté par un nombre à l'intérieur du cercle) et un centre d'intérêt appartenant à une classe de profil (représenté par un nombre à l'extérieur du cercle). Dans cet exemple, on décrit deux classes de profils. Chaque classe identifie une communauté d'intérêt.

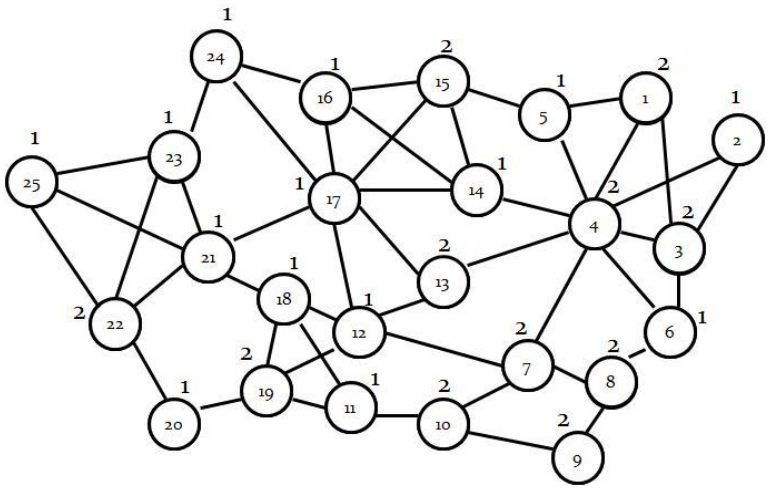

Fig.4.11. Exemple d'un réseau mobile représenté sous forme d'un graphe.

Le centre d'intérêt (profil) de chaque nœud {2, 5, 6, 11, 12, 14, 16, 17, 18, 20, 21, 23, 24, 25} est égal à 1. Ainsi, ces nœuds appartiennent à la communauté d'intérêt 1.

Le profil des nœuds {1, 3, 4, 7, 8, 9, 10, 13, 15, 19, 22} est égal à 2. Par conséquent, ces nœuds appartiennent à la communauté d'intérêt 2.

4.4.2 Construction de l'ensemble MIS du graphe G

La première étape de notre algorithme consiste à trouver l'ensemble MIS du graphe G(V,E). Après l'exécution est stabilisation de l'algorithme auto stabilisant décrit dans la

section 4.3.1 (**Algorithme 1:** Algorithme de construction D'un MIS de G) nous obtenons l'ensemble MIS de G représenté dans la Fig.4.12.

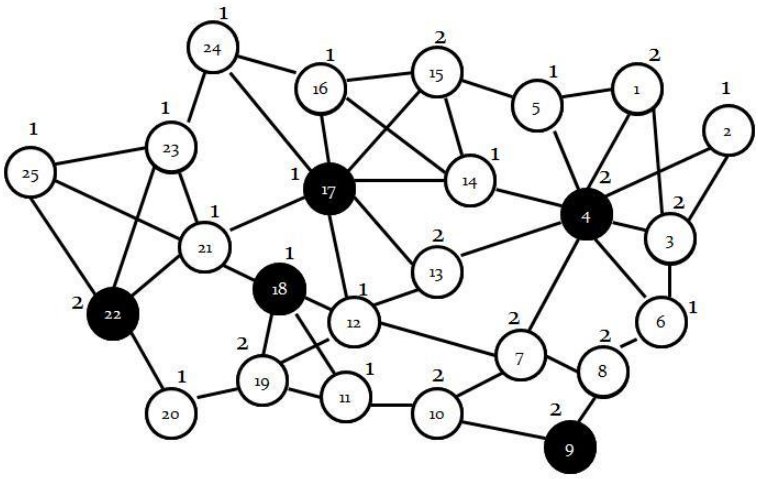

Fig.4.12. Résultat du calcul de l'ensemble MIS de G.

L'ensemble des nœuds {4, 9, 17, 18, 22} colorés en noir représente les nœuds indépendants de G (l'ensemble MIS), tandis que l'ensemble de nœuds {1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 18, 19, 20, 21, 23, 24, 25} colorés en blanc représente les nœuds dominés de G.

4.4.3 Construction de la topologie virtuelle : graphe de couverture de la communauté 1

La deuxième étape de notre algorithme consiste à créer le graphe de couverture de la communauté CI_R. Dans cet exemple, nous allons nous limiter à créer le graphe de couverture de la communauté d'intérêt1(CI_1). (Les mêmes étapes seront appliquées pour la communauté d'intérêt 2).

Pour ce faire, nous allons appliquer l'algorithme distribué décrit dans la section 4.3.2. Après exécution de l'Algorithme 2 par chaque nœud MIS du graphe G, nous obtenons la topologie virtuelle représentant le graphe de la communauté 1 sauvegardée dans les caches des nœuds du MIS.

Nous allons montrer pour chaque nœud de l'ensemble MIS = {4, 9, 17, 18, 22} le contenu de son cache :

- **Cache du nœud 4 :** Après avoir envoyer le message Req(4, 1) pour chercher les membres de la communauté 1 à N^2 (4)={8, 12, 15, 16, 17} (R1. De l'algorithme 2) et recevoir les repenses RRep de ces derniers, le résultat de son cache est représenté par la table 2.

Table 2. Cache du nœud 4.

4	N-Ref	N-Dest	Com	Dist
	17	13	1	2
	18	14	1	3

Le nœud 4 référence (établit des liens virtuels) les nœuds :

- Nœud 17 : résultat de l'exécution de la règle R2.1.a (ce nœud appartient à l'ensemble MIS et à la communauté 1).
- Nœud 18 : résultat de l'exécution de R2.2.a) par le nœud 12(Ce nœud appartient à la communauté 1, n'appartient pas à MIS de G mais couvert par le nœud 18 du MIS).

On représente le cache par la topologie virtuelle locale illustrée dans la Fig. 4.13. (1):

- **Cache du nœud 9 :** le cache du nœud 9 est représenté par la table 3.

Table 3. Cache du nœud 9.

9	N-Ref	N-Dest	Com	Dist
	4	8	0	3
	18	10	1	3

Ce nœud référence les nœuds :

- Nœud 4 : Résultat de l'exécution de la règle R2.2.a) par le nœud 6.
- Nœud 18 : Résultat de l'exécution de la règle R2.2.a) par le nœud 11.

Sa topologie virtuelle locale est illustrée dans la Fig.4.13. (2)

- **Cache du nœud 17:** le cache du nœud 17 est représenté par la table 4

Table 4. Cache du nœud 17.

17	N-Ref	N-Dest	Com	Dist
	4	14	0	2
	18	12	1	2
	22	21	0	2

Ce nœud référence les nœuds :

- Nœuds 4 et 22 : Résultat de l'exécution de la règle R2.2.b) (nœuds appartiennent à l'ensemble MIS de G, n'appartiennent pas à la communauté1 mais couvre des éléments de cette communauté) .

– Nœud 18 : Résultat de l'exécution de la règle R2.1.a).

Sa topologie virtuelle locale est représentée par la Fig. 4.13. (3)

- **Cache du nœud 18:** le cache du nœud 18 est représenté par la table 5

Table 5. Cache du nœud 18.

18	N-Ref	N-Dest	Com	Dist
	17	12	1	2
	22	21	0	2

Ce nœud référence les nœuds 17 et 22: Résultat de l'exécution de la règle R2.1.a) et R2.2.b) (resp.). Sa topologie virtuelle locale est représentée par la Fig. 4.13. (4).

- **Cache du nœud 22:** le cache du nœud 22 est représenté par la table 6

Table 6. Cache du nœud 22.

22	N-Ref	N-Dest	Com	Dist
	17	12	1	2
	18	21	1	2

Ce nœud référence les nœuds 17 et 22 : Résultat de l'exécution de la règle R2.1.a) et R2.2.b) resp. Sa topologie virtuelle locale est représentée par la Fig.4.13. (5)

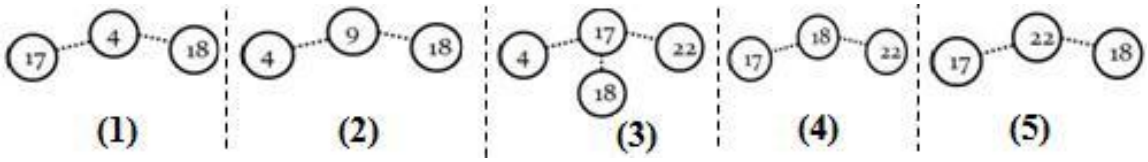

Fig.4.13. Topologies locales des nœuds de l'ensemble MIS.

L'union de toutes ces topologies virtuelles locales représente la topologie virtuelle globale du graphe qu'on a nommé graphe de couverture de la communauté CI_1 (ou backbone de CI_1). Cette topologie est illustrée dans la Fig.4.14, graphe(a). Le graphe (b) représente une vision globale de cette topologie dans G. Les arêtes discontinues représentent des liens virtuels.

Le degré du graphe de couverture de la communauté 1est égal à 4 et est supérieur à la taille limite du cache des mobiles k=2. Ainsi La troisième étape de notre algorithme consiste à

résoudre ce problème en construisant un K-arbre couvrant de degré 2 du graphe de couverture de cette communauté.

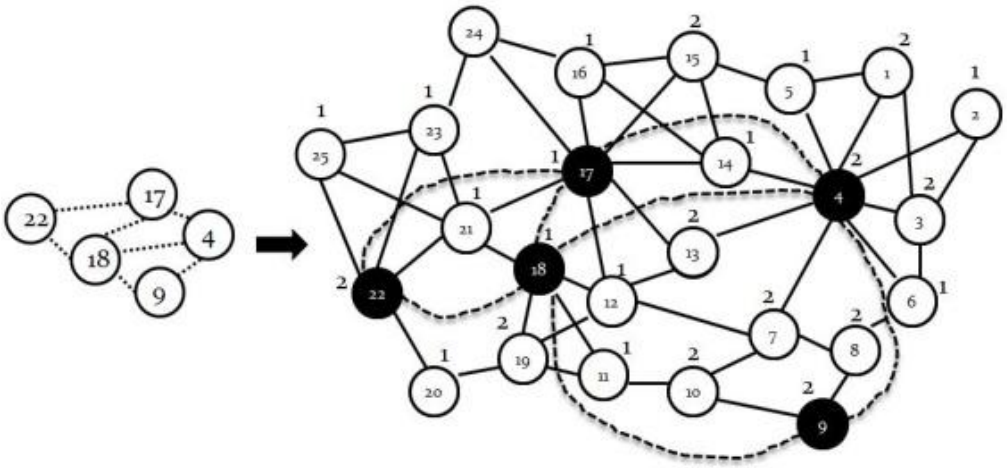

Fig. 4.14. Graphe de couverture de la communauté d'intérêt 1.

4.4.4 Construction du K-arbre couvrant de la communauté d'intérêt 1

Le k- arbre couvrant de la communauté 1(avec k=2) est construit en appliquant les différents modules auto stabilisants de l'algorithme décrit dans la section 4.3.3.

Après exécution de l'algorithme de construction d'un arbre couvrant sur le graphe de couverture de la communauté 1 on obtient l'arbre couvrant illustré dans la Fig.4.15.

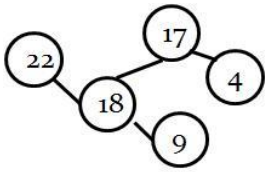

Fig. 4.15. Arbre couvrant de la communauté d'intérêt 1.

Le degré maximum de l'arbre courant est égal à 3 et est supérieur à k. (le degré du nœud 18 est de 3). Ce degré va être réduit comme suit :

L'arête (22,17) (retourné par cycle_ search) est une arête non améliorante (vérifiée par Action_On_Cycle), car le cycle créé en ajoutant cette arête à l'arbre contient un nœud bloquant (le nœud 17, voir graphe (1) de Fig. 4.16), ainsi, il faut débloquer en premier, ce nœud (Deblock). L'arête (4, 9) est une arête améliorante (graphe (2) Fig.4.16), échanger cette arête avec une arête adjacente au nœud 17 (l'arête (4, 17), graphe (3) Fig.4.16) puis échanger l'arête (22, 17) avec une arête adjacente au nœud 18 (soit l'arête(18,22), graphe (4), Fig.4.16).

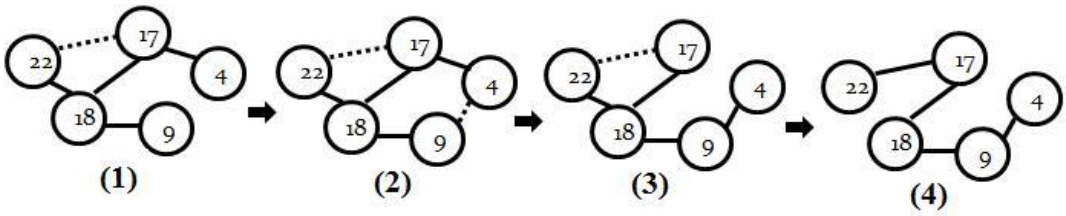

Fig. 4.16. Processus de création du k-arbre couvrant de CI_1

Le graphe (4) de la Fig.4.16 représente le k-arbre couvrant de la communauté d'intérêt1.

Pour chaque nœud désirant retrouver les membres de la communauté 1, par exemple, pour leurs envoyer une information quelconque, il lui suffira de transmettre cette information sur ce k-arbre.

4.5 Complexité

Nous allons évaluer la complexité totale de notre approche de découverte de communautés d'intérêts dans les réseaux ad hoc, en donnant les complexités des différents algorithmes intervenant dans cette approche, à savoir, l'algorithme de construction de l'ensemble MIS du graphe G(V, E) modélisant le réseau , puis l'algorithme de construction du graphe de couverture de la communauté d'intérêt, et en fin l'algorithme de construction du K-arbre couvrant de la communauté.

L'algorithme qu'on a utilisé pour la construction de l'ensemble MIS du graphe G est un algorithme auto stabilisant inspiré de celui proposé dans [29], les auteurs ont prouvé que cet algorithme retourne un ensemble MIS du graphe au bout d'un temps fini. La complexité en nombre de messages de cet algorithme est **O(n)**, avec n=|V|, le nombre des nœuds du graphe G.

Pour la construction du graphe de couverture de communauté, on a proposé un algorithme distribué dont la complexité en nombre de message est $O(n.\Delta^2)$, avec Δ, représente le degré maximum dans le graphe.

En effet, dans le pire cas, un nœud cherchant les membres de sa communauté situés à deux sauts envoie Δ^2 messages. Ce nœud a Δ voisins et chacun de ces derniers possède Δ autres voisins. Tout les nœuds cherchent les membres de la communauté, ainsi la complexité de l'algorithme est **$O(n. \Delta^2)$.**

L’algorithme de construction du K-arbre couvrant de la communauté est une adaptation de l’algorithme auto stabilisant proposé dans [30]. Les auteurs ont montré que l’algorithme retourne un arbre couvrant de degré minimum au bout d’un temps fini. La complexité, en nombre de message, de cet algorithme est **O(m.n.log(n))**. Avec m=|E|, le nombre d’arête du graphe. Cette complexité reste la même pour la construction de notre k-arbre couvrant.

Conclusion générale & Perspectives

Dans ce travail, nous avons proposé un nouvel algorithme distribué pour la découverte de communautés d'intérêt dans les systèmes sociaux distribués, en particuliers, dans les réseaux mobiles ad hoc où les entités constituant ces réseaux sont dotées de caches limités.

Notre principale contribution consiste en la création d'une topologie virtuelle dynamique orientée découverte de communautés (backbone de communauté) et respectant la contrainte du cache limité.

Cette topologie est basée, en premier, sur la réduction de l'ensemble de nœuds qui assurent la tache de découverte et référencement des membres de communautés. Cet ensemble est représenté par l'ensemble indépendant maximal (MIS) du réseau. Puis, la réduction des degrés des nœuds du MIS, à fin de satisfaire la contrainte de caches limités qui est garantie par la construction d'un k- arbre couvrant l'ensemble MIS.

Lorsqu'une information à relayer pour les membres d'une communauté est reçue par un nœud de l'ensemble MIS, il ne peut la relayer que via un lien faisant parti du k-arbre. Le nombre d'envois est optimisé et une information ne peut boucler.

Tous les calculs sont réalisés de façon complètement distribuée et, par conséquent, l'algorithme pourrait être adapté dans des systèmes à grande échelle.

Les réseaux ad hoc doivent souvent faire face aux changements dynamiques de topologie. Par conséquent, la maintenance de la topologie virtuelle est une question importante. Généralement, il y a deux types de mécanismes pour manipuler le changement de topologie : reconstruction périodique (proactif) et la reconstruction à la demande (réactif). Dans le premier cas, la période passée avant la reconstruction de la topologie a un impacte sur performances du système. Si elle est trop courte, les couts de communications seront élevés, si elle est trop longue, peut-être les vieux MIS ne peuvent pas garantir la propriété de couverture dans la nouvelle topologie. La construction réactive est efficace dans le cas ou la topologie change rarement et perdra son efficacité lorsque des changements fréquents sont produits. Nous envisageons alors de tester notre algorithme sur un réseau dans les deux cas de découverte réactive et proactive des communautés.

Dans la réalité, les mobiles sont dotés de caches limités mais de taille différente ainsi, nous estimons de prendre en compte des différentes tailles des caches des nœuds couplé avec leur consommation d'énergie dans la découverte et référencement des membres de communautés. Construire une topologie qui se base sur d'autres concepts de graphe telle que les CDS peut être envisagée.

Références Bibliographiques

1. Wasserman S., Faust K. Social network analysis. Cambridge University Press, Cambridge 1994.
2. Mauricio G., Resende C. Detecting dense sub graphs in massive graphs. In 17th International Symposium on Mathematical Programming 2000.
3. Guillaume JL., Stevens LB., Latapy M. Statistical analysis of a p2p query graph based on degrees and their time-evolution. In Lecture Notes in Computer Sciences (LNCS), proceedings of the 6-th International Workshop on Distributed Computing (IWDC), 2004.
4. Holger E., Lutz-Ingo M., Bornholdt S. Scale-free topology of e-mail networks. Physical Review E, 66, 2002.
5. Jeong H., Tombor B., Albert R., Oltvai Z., Barabasi A. The large-scale organization of metabolic networks. Nature, 407, 651, 2000.
6. Faloutsos M., Faloutsos P., Faloutsos C. On power-law relationships of the internet topology. In Proceedings og the internatonal ACM conference SIGCOMM, pages 251_262, 1999.
7. Kleinberg J. M., Kumar R., Raghavan P., Rajagopalan S. The Web as a graph : Measurements, models, and methods. In T. Asano, H. Imai, D. T. Lee, S. Nakano, and T. Tokuyama, editors, Proc. 5th Annual Int. Conf. Computing and Combinatorics, COCOON, number 1627. Springer-Verlag, 1999.
8. Ferrer R., Cancho i., Ricard V.S. The small-world of human language. Technical report, Santa Fe Working paper 01-03-016, 2001.
9. Albert R., Jeong H., and Barabási A.-L. Error and attack tolerance in complex networks. Nature, 406 :378_382, 2000.
10. Travers J., Milgram S. An experimental study of the small world problem. Sociometry, 32(4) : 425_443, 1969.
11. [WS98] Watts, D. J., Strogatz, S. H., Collective dynamics of small-world networks, Nature 393, 440-442 ,1998.
12. Ravasz E., Somera A. L., Mongru D. A., Oltvai Z. N., Barabasi A.-L.. Hierarchical Organization of Modularity in Metabolic Networks. Science, 297(5586) :1551{1555, 2002.
13. Pimm SL. The structure of food webs. Theoretical Population Biolology,Vol 16, pp 144–158, 1979

14. Flake GW, Lawrence S, Lee Giles C, Coetzee FM, Self-Organization and Identification of Web Communities. IEEE Computer, Vol 35, No 3, pp 66–71, 2002
15. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. PNAS, 99(12) :7821-7826, 2002.
16. Auber D, Chiricota Y., Jourdan F., Melançon G. Multiscale visualization of small world networks. In Proc of the 9th IEEE Symposium on Information Visualization (InfoVis 2003), page 10, Seattle, USA, 2003.
17. Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D, Defining and identifying communities in networks. Proceedings of the National Academy of Science of the USA, Vol 101, No 9, pp 2658–2663,2004.
18. Newman MEJ. Modularity and community structure in networks. Proceedings of the National Academy of Science of the USA, Vol 103, No. 23, pp 8577–8582, 2006
19. Pothen A., Simon H.D., and Liou K.P. Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl., 11(3) :430452, 1990.
20. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. PNAS, 99(12) :7821–7826
21. M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical Review E (Statistical, Nonlinear, and Soft Matter Physics). , 2002.
22. M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, Physical Review E, 70, 066111, 2004.
23. Arenas A., Fernandez A., Fortunato S., Gomez S., Motif-based communities in complex networks, Journal of Physics A: Mathematical and Theoretical, 41, 224001, 2008.
24. Duch J. Arenas A. Community detection in complex networks using extremal optimization. Physical Review E (Statistical, Nonlinear, and Soft Matter Physics), 72(2) :027104, 2005.
25. Palla G, Derényi I, Farkas I, Vicsek T, Uncovering the overlapping community structure of complex networks in nature and society. Nature, Vol 435, pp 814-818, 2005.
26. Milo R., Shen-Orr S., Itzkovitz S., Kashtan N., Chklovskii D., Alon U., Network motifs: Simple building blocks of complex networks, Science, 298, pp. 824–827, 2002.
27. Chan S., Hui P., Xu K. Community Detection of Time-Varying Mobile Social Networks. In Proceeding of The First International Conference on Complex Sciences: Theory and Applications (Complex 2009), Shanghai China, 2009.

28. Haddad M., Kheddouci H. A Virtual dynamic topology for service discovery in mobile ad hoc networks. IEEE Wireless Communications and Networking Conference 2007, Hong Kong, Chine, pp. 6, 2007.
29. Goddard W., Hedetniemi S.T., Jacobs D.P., Srimani K. Self-Stabilizing Protocols for Maximal Matching and Maximal Independent Sets for Ad Hoc Networks. In Proceedings of IPDPS. pp.162-162, 2003.
30. Blin L., Gradinariu M. Butucaru P., Rovedakis S. Self-stabilizing minimum-degree spanning tree within one from the optimal degree. In Proceedings of the 23rd IEEE International Conference on Parallel and Distributed Processing Systems (IPDPS), pages 1-11, Rome, Italy, 2009.
31. Afek Y., Kutten S., Yung M. Memory-efficient self stabilizing protocols for general networks. In Proceedings of the 4th international workshop on Distributed algorithms (WDAG), volume LNCS 486, pages 15–28, New York, NY USA, Springer-Verlag New York, Inc, 1991.
32. Haddad M., Kheddouci, H. A survey on graph based service discovery approaches for ad hoc networks, International Transactions on Systems Science and Applications (ITSSA), Volume 4, Issue 2, pp. 145-152, 2008.
33. Martinez, N. D., Artifacts or attributes? Effects of resolution on the Little Rock Lake food web, Ecological Monographs 61, 367-392 (1991).
34. S.N. Dorogovtsev and J.F.F. Mendes. Evolution of Networks: From Biological Nets to the Internet and WWW. Oxford University Press, Oxford, 2003.
35. M. E. J. Newman, The structure and function of complex networks. SIAM Review 45, 167–256, 2003.
36. P. Erdös and A. Rényi. On random graphs. Publicationes Mathematicas 6, 290(7), 1959.
37. M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. Computer Communications Rev., 29 :251–262, 1999.
38. Maslov S., Sneppen K., and Zaliznyak A., Pattern detection in complex networks: Correlation profile of the Internet, Preprint cond-mat/0205379 ,2002.
39. Maslov S. and Sneppen K., Specificity and stability in topology of protein networks, Science 296, 910-913,2002.
40. J. Kleinberg. The Small-World Phenomenon: An Algorithmic Perspective. In Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC), pages 163–170, 2000.

41. R. Albert, H. Jeong, A.-L. Barabási, Diameter of the world-wide web, Nature (London) 401 (1999) 130–131.

42. L. C. Freeman, A set of mesures of centrality based on betweeness ,Sociometry 40, 35,1977.

43. A. Mahdian, H. Khalili, E. Nourbakhsh, and M. Ghodsi. Web graph compression by edge elimination. In Proceedings of the Data Compression Conference (DCC'06), page 459. IEEE Computer Society, 2006.

44. S. Fortunato, M. Barthélemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. USA 104 (2007) 36-46.

45. B. W. Kernighan and S. Lin. An e_cient heuristic procedure for partitioning graphs. Bell System Technical Journal, 49(2) :291_308, 1970.

46. Santo Fortunato, Vito Latora, and Massimo Marchiori. Method to _nd community structures based on information centrality. Physical Review E, 70(5) :056104, 2004

47. Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. Physical Review E, 70(6) :066111, 2004.

48. Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks. preprint arXiv :cs/0702048, 2007.

49. L. Donetti and M. A. Muñoz. Detecting network communities: a new systematic and efficient algorithm. Journal of Statistical Mechanics, 2004(10):10012, 2004.

50. L. Donetti and M. A. Muñoz. Improved spectral algorithm for the detection of network communities. In Modeling cooperative behavior in the social sciences, volume 779, pages 104_107, 2005.

51. Roger Guimera and Luis A. Nunes Amaral. Functional cartography of complex metabolic networks. Nature, 433 :895_900, 2005.

52. Arenas A., Fernandez A., Fortunato S., Gomez S., Motif-based communities in complex networks, Journal of Physics A: Mathematical and Theoretical, 41, 224001, 2008.

53. Milo R., Shen-Orr S., Itzkovitz S., Kashtan N., Chklovskii D., Alon U., Network motifs: Simple building blocks of complex networks, Science, 298, pp. 824–827, 2002.

54. Duch J. Arenas A. Community detection in complex networks using extremal optimization. Physical Review E (Statistical, Nonlinear, and Soft Matter Physics), 72(2) :027104, 2005.

55. Stijn van Dongen. Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht, May 2000.