N° d'ordre : 13 /2005 –M /IN

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE
« HOUARI BOUMEDIENE »
FACULTE DE GENIE ELECTRIQUE DEPARTEMENT INFORMATIQUE

MEMOIRE
Présenté pour l'obtention du diplôme de MAGISTER
EN : Informatique

Spécialité : Programmation et système

Par : Mr GUERROUMI MOHAMED

Sujet

Accessibilité aux données dans les réseaux mobiles ad hoc

Soutenu le 20/12/2005 devant le jury composé de :

M^me. M. IOUALLALEN BOUKALA	M.C	Présidente.
M^r. N. BADACHE	PR	Directeur de thèse.
M^elle. N. BENSAOU	M.C	Examinateur.
M^me. S.MOUSSAOUI		Invitée.

RESUME

Avec le développement de la technologie sans fil et l'utilisation croissante des réseaux mobiles ad hoc, beaucoup de problèmes et de difficultés viennent d'apparaître. Les réseaux mobiles ad hoc permettent aux utilisateurs d'interconnecter sans besoin d'une infrastructure préexistante. Ils sont bien plus complexes que les réseaux mobiles utilisant des stations fixes, parce que les unités mobiles peuvent agir comme des stations et des routeurs, et ils peuvent quitter le réseau à n'importe quel moment. Ceci cause la division fréquente et le changement imprévisible de la topologie du réseau et diminue l'accessibilité aux données.

Les réseaux mobiles ad hoc sont des réseaux fortement dynamiques et décentralisés comme les systèmes Peer to peer. Cette similitude peut conduire à exploiter le développement de ces systèmes pour construire un réseau mobile ad hoc à grande échelle. Avoir un réseau mobile ad hoc de taille importante nécessite d'améliorer le niveau d'accessibilité de données.

Dans ce travail nous proposons trois méthodes pour élever le niveau d'accessibilité, en utilisant la réplication de données. Dans ces trois méthodes, nous tiendrons compte de la fréquence d'accès des nœuds mobiles à chaque donnée. En ce qui est de l'évaluation des performances des méthodes proposées, nous présentons les résultats obtenus par la programmation de ces méthodes sur le simulateur Glomosim.

Sommaire

Introduction générale

Les technologies sans fil connaissent aujourd'hui un essor qui permet d'envisager de nouvelles perspectives dans le domaine des télécommunications. Les nouveaux moyens multimédias qui permettent de traiter l'information ont des caractéristiques particulières (une faible capacité de stockage, une source d'énergie autonome…) et accèdent au réseau à travers une interface de communication sans fil. Dans cet environnement mobile, aucune restriction n'est posée sur la localisation des usagers. Le nomadisme et le nouveau mode de communication utilisé, engendrent de nouvelles caractéristiques propres à l'environnement mobile : une fréquente déconnexion, un débit de communication et des ressources modestes, et des sources d'énergie limitées.

Les réseaux ad hoc sont des réseaux mobiles sans fil. Ils peuvent être définis comme une collection d'entités mobiles interconnectées par un lien sans fil formant un réseau temporaire sans l'aide de toute administration ou de tout support fixe. Un réseau ad hoc peut avoir une taille importante.

Dans les systèmes peer to peer (P2P) tous les nœuds possèdent des capacités et des responsabilités équivalentes. Ces nœuds peuvent coopérer et contribuer à fournir des services au réseau. Les réseaux Ad Hoc (MANET) et les systèmes P2P partagent certaines caractéristiques principales, auto organisation et décentralisation, et tous les deux ont besoin de résoudre le même problème fondamental de la connectivité. Ces similitudes peuvent aider à résoudre le problème de développement des applications ad hoc à grande échelle.

Dans un réseau ad hoc, les hôtes mobiles doivent former, de manière spontanée, une architecture générale qui peut être utilisée comme infrastructure du système. Les applications des réseaux ad hoc sont nombreuses. On trouve des applications dans le domaine militaire et d'autres applications tactiques comme les opérations de secours et les missions d'exploration.

L'utilisation de réseau mobile ad hoc croit d'une manière considérable. Elle est idéale dans les situations où l'installation d'une infrastructure n'est pas possible, parce que l'infrastructure est trop chère ou trop vulnérable. En raison du manque de support d'infrastructure dans les réseaux ad hoc, chaque nœud fonctionne comme un routeur et participe dans l'envoi de données pour d'autres nœuds. La plupart des recherches dans les réseaux mobiles ad hoc se concentrent sur le développement de protocoles de routage dynamiques pour trouver des itinéraires entre les nœuds. Bien que le routage soit une question importante dans les réseaux ad hoc, d'autres questions comme l'accès à l'information (données) est également très importante, puisque le but final de l'utilisation des réseaux ad hoc est que les nœuds mobiles puissent accéder à l'information et autant que possible à travailler comme sur un réseau statique avec tous les avantages de partage et de coopération.

Puisque les nœuds mobiles dans les réseaux ad hoc se déplacent librement, les déconnexions se produisent fréquemment, ceci cause la division fréquente du réseau. En cas de partition,

les nœuds mobiles d'une des partitions ne peuvent pas accéder à des données élémentaires tenues par les nœuds mobiles dans l'autre partition. L'enjeu aujourd'hui pour les réseaux ad hoc est de garder l'information disponible n'importe où et n'importe quand, pour n'importe quel usager et quelque soit l'emplacement de l'information, avec les meilleurs temps d'accès possibles. Vu les limitations des réseaux mobiles ad hoc, une question critique s'impose alors et nécessite une véritable réflexion. C'est la question de disponibilité et d'accessibilité des données. Une solution possible est la réplication de données. Cette technique consiste à créer des copies d'une même donnée sur plusieurs hôtes mobiles. L'objectif principal est d'améliorer les performances et surtout de permettre l'accès aux données.
Dans ce travail nous allons proposer une solution pour améliorer la disponibilité, en prenant en compte les limitations liées à l'environnement ad hoc.

Ce document est composé de cinq chapitres : le but du premier chapitre est de définir l'environnement mobile ad hoc et de mettre en évidence ses caractéristiques et ses domaines d'application. Dans ce chapitre nous allons parler du problème d'accessibilité de données qui s'impose quand le réseau mobile ad hoc est partitionné et nous allons souligner que la réplication est la meilleure solution à ce problème. Après la définition du système P2P et la description de ses principales applications, nous avons constaté que les deux environnements partagent certaines caractéristiques principales, et les réseaux ad hoc peuvent bénéficier du développement de ces systèmes.

Le deuxième chapitre est consacré à l'étude de la technique de la réplication de données dans l'environnement mobile et l'exploration des avantages et des inconvénients de cette technique. Après la description des stratégies de la réplication et la mise en évidence des techniques de chaque stratégie, nous allons étudier la réplication à travers quelques systèmes existants.

Le troisième chapitre est réservé à la présentation des protocoles principaux de la réplication de données dans les réseaux mobiles ad hoc. Nous allons décrire dans ce chapitre les principales caractéristiques, les fonctionnalités ainsi que les avantages et les inconvénients de chaque méthode. Les méthodes sont classées suivant le type de données manipulé (sans mises à jour, mises à jour périodiques et mises à jour apériodiques). Dans ce chapitre nous présenterons une étude comparative des méthodes étudiées.

Dans le quatrième chapitre, nous allons présenter la conception de notre solution. Cette dernière consiste à proposer trois méthodes de réplication visant à traiter le problème de disponibilité de données dans l'environnement mobile ad hoc. L'exécution de ces méthodes s'effectue sur deux parties : La première partie permet de créer les réplicas primaires de chaque nouvelle donnée créée et la deuxième partie consiste à gérer dynamiquement les

copies (réplicas) de données. Dans ces méthodes, chaque donnée est associée à deux types de fréquences d'accès, interne et externe.

Le dernier chapitre représentera l'évaluation de nos méthodes, en utilisant le simulateur glomosim. Ces méthodes seront comparées avec la méthode DCG-S1(Dynamic Connectivity based Grouping- Stability of radio links), parce que cette dernière donne des meilleures performances par rapport aux autres méthodes. Le résultat de nos expériences montre que nos méthodes améliorent l'accessibilité et diminuent le trafic.

Chapitre 1 : Réseaux Mobiles Ad Hoc

1.1 Introduction :

La technologie dans le domaine de la communication sans fil a vécu dans les dernières années une évolution exponentielle marquée non seulement par l'augmentation des dispositifs mobiles (handheld digital devices, personal digital assistants, or wearable computers) qui sont de plus en plus performants et leurs tailles sont de plus en plus petites, mais aussi par la diversité des applications dans ce domaine. Cette Evolution offre aux utilisateurs la possibilité d'utiliser des ordinateurs portables ou nomades. Ils ne sont plus forcés de travailler depuis leurs ordinateurs du bureau mais au contraire ils peuvent se déplacer librement avec leurs portables tout en accédant de manière transparente à leurs données et sans changement de profil ou d'environnement de travail.

Les réseaux mobiles ad hoc permettent aux utilisateurs de former d'une manière aléatoire et décentralisée un réseau de communication sans fil et sans infrastructures préexistantes. Leur but est de garder l'information accessible n'importe où et n'importe quand. Les réseaux mobiles ad hoc partagent certaines caractéristiques principales avec les systèmes peer to peer, comme l'auto organisation et la décentralisation. Ces systèmes trouvent leurs applications dans le domaine de communication Internet et le partage de fichiers.

1.2 L'environnement mobile

Un environnement mobile est un système composé de sites mobiles qui permet à ses utilisateurs d'accéder à l'information indépendamment de leurs positions géographiques. Les réseaux mobiles sans fil, peuvent être classés en deux catégories:
Les réseaux avec infrastructure qui utilisent généralement le modèle de la communication cellulaire utilisant les stations de base fixe, et les réseaux sans infrastructure ou les réseaux ad hoc. Plusieurs systèmes utilisent déjà le modèle cellulaire et ont connu une très forte extension ces dernières années (les réseaux GSM notamment) mais ont besoin d'une lourde infrastructure matérielle fixe.
Le modèle de réseau sans infrastructure préexistante n'utilise pas les stations fixes, tous les sites du réseau sont mobiles et communiquent d'une manière directe en utilisant leurs interfaces de communication sans fil. L'absence de l'infrastructure oblige les unités mobiles à se comporter comme des routeurs qui participent au cheminement de données aux autres unités.

1.3 Réseaux mobiles ad hoc

1.3.1 Définition

Un réseau mobile ad hoc est un système autonome formé dynamiquement par des nœuds mobiles qui sont reliés par l'intermédiaire des liens sans fil, sans utiliser une infrastructure préexistante ou une gestion centralisée [Bwe82], [Bro98](voir la figure1.1(A)). Les nœuds se déplacent d'une manière libre et aléatoire et s'organisent arbitrairement, ainsi, la topologie du réseau ad hoc peut changer rapidement et de manière imprévisible.

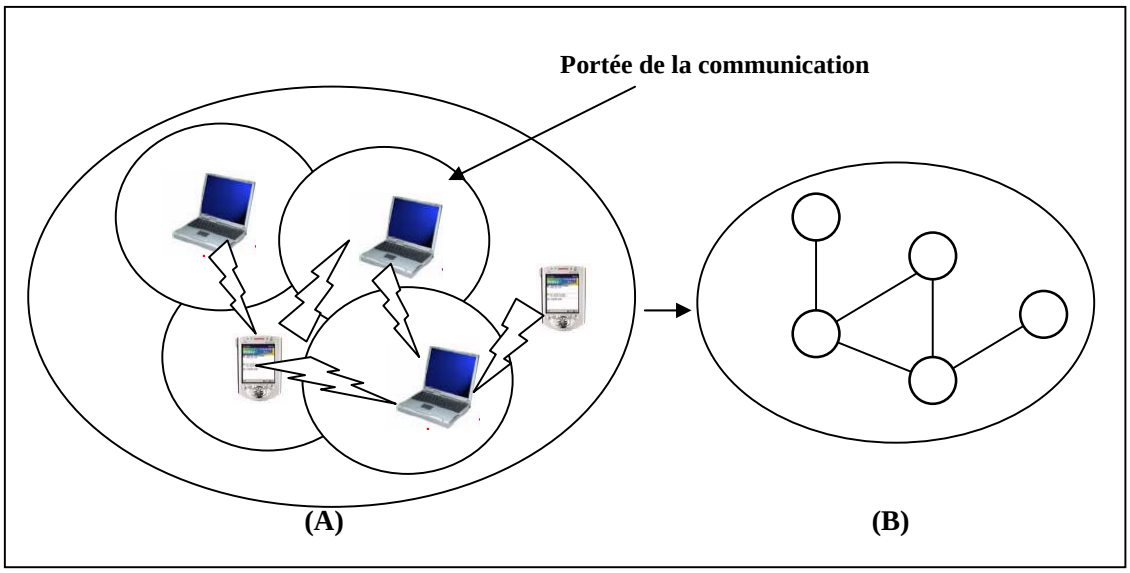


Figure 1.1 : Réseau mobile ad hoc

Un tel réseau appelé généralement MANET (Mobile Ad hoc NETwork), peut fonctionner dans un mode autonome ou peut être relié à l'Internet. En général, les itinéraires entre les nœuds dans un réseau ad-hoc peuvent inclure des sauts multiples, ce qui justifie que les réseaux ad hoc sont appelés les réseaux ad hoc sans fil de multi-sauts. Chaque nœud pourra communiquer directement avec n'importe quel autre nœud qui réside dans sa marge de transmission. Pour communiquer avec les nœuds qui résident au-delà de cet intervalle, un nœud mobile a besoin d'utiliser les nœuds intermédiaires.
Le graphe représenté par la figure 1.1(B) est une modélisation du réseau mobile ad hoc schématisé par la figure 1.1(A), où les cercles représentent l'ensemble des nœuds du réseau et les traits présentent la communication directe entre les nœuds.

1.3. 2 Les caractéristiques des réseaux ad hoc

Les réseaux mobiles ad hoc sont caractérisés par ce qui suit :

Un système autonome sans infrastructure préexistante: Un réseau ad hoc n'a besoin d'aucune infrastructure préexistante ou d'une gestion centralisé. Chaque nœud fonctionne en mode distribué de peer to peer, il agit comme un routeur indépendant et produit des données indépendantes. La gestion de réseau doit être distribuée à travers les différents nœuds, ce qui apporte une difficulté dans la détection et la gestion des défauts.

Routage de multi-sauts : Car tous les nœuds dans un réseau ad hoc jouent le rôle d'un routeur et aucun routeur donné par défaut. Les messages transmis par l'émetteur doivent été envoyés nœud par nœud afin d'arriver au destinataire.

Topologie dynamique : Les unités mobiles du réseau se déplacent d'une façon libre et arbitraire. Par conséquent la topologie du réseau peut changer à des instants imprévisibles, d'une manière rapide et aléatoire. Les liens de la topologie peuvent être unis ou bidirectionnels.

Bande passante limitée : Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un hôte soit modeste.

Variation des capacités de lien et de nœud : Chaque nœud peut être équipé d'une ou plusieurs interfaces par radio qui ont des capacités de transmission /réception variables et fonctionne à travers les différentes bandes de fréquence. Cette hétérogénéité dans les capacités de radio de nœud peut avoir comme conséquence des liens probablement asymétriques. En plus, chaque nœud mobile pourrait avoir une configuration logicielle et matérielle différente ayant pour résultat la variabilité dans des capacités de traitement. Concevoir des protocoles et des algorithmes de réseau pour ce réseau hétérogène peut être complexe, exigeant l'adaptation dynamique aux conditions changeantes.

Energie autonome : Les nœuds mobiles sont alimentés par des sources d'énergie autonomes comme les batteries. Une partie importante de cette énergie est consommée quand les unités transmettent ou reçoivent les messages à travers une connexion sans fil. Ce qui implique que les protocoles conçus pour les réseaux ad hoc doivent réduire au maximum le trafic, pour minimiser la consommation d'énergie.

Sécurité physique limitée : Les réseaux mobiles ad hoc sont plus touchés par le paramètre de sécurité par rapport aux réseaux filaires classiques. Cela se justifie par les contraintes et les limitations physiques qui font que le contrôle de données transférées doit être minimisé.

1.3.3 Les applications actuelles et futures des réseaux mobiles ad hoc

Les réseaux ad hoc sont utilisés dans toute application où le déploiement d'une infrastructure réseau filaire est trop contraignant, soit parce que difficile à mettre en place, soit parce que la durée d'installation des câbles prend beaucoup temps.
Les applications des réseaux ad hoc peuvent être classées par catégorie comme suit:

Applications militaires : les réseaux ad hoc sont en particulier convenus au scénario de champ de bataille où les soldats ont besoin d'un système de transmissions portatif, instantané et résilient opérant dans un environnement hostile.

Applications commerciales : actuellement un réseau ad hoc est utilisé pour les applications commerciales, notamment pour la création des réseaux personnels PANs (Personal Area Networks). Dans les réseaux PAN, les dispositifs mobiles, tels qu'un laptop, un mobilophone, et un assistant personnel digital (PDA) collaborent entre eux pour l'échange et la synchronisation de données. Les futures applications des réseaux ad hoc commerciaux incluront la gestion de réseau dans les bureaux provisoires et dans les conférences [Rar02]. Dans une certaine mesure, ces applications sont déjà possibles, quoique limitées à la transmission entre les nœuds voisins. Cette restriction surgit parce que des protocoles de routages pour ce type de réseaux n'ont pas été encore normalisés.

Applications de secours et de délivrance : Les réseaux ad hoc à l'avenir pourraient être déployés dans des situations de secours et de délivrance où l'infrastructure fixe a pu être détruite à cause d'un tremblement de terre ou à toute autre catastrophe.

Réseaux de capteur : Les réseaux ad hoc peuvent également relier des réseaux de capteurs. Une application typique de tels réseaux est pour la collection de conditions ambiantes, telles que des mesures de pollution. Les nœuds pour cette application devraient être petits, à prix réduit et de basse puissance.

1.4 Le problème d'accessibilité de données dans les réseaux mobiles ad hoc

Dans les réseaux ad hoc, chaque nœud mobile joue le rôle de routeur, et communique avec les autres. Même si la source et le nœud destinataire ne sont pas en contact direct, les paquets de données sont envoyés au nœud mobile de destination en utilisant les nœuds mobiles intermédiaires. La nature de déplacement aléatoire et imprévisible des nœuds mobiles, produit la division fréquente de réseau. La figure 1.2 présente un exemple de deux partitions. Dans cette figure, les nœuds de la partition P1 ne peuvent pas accéder aux données de la partition P2 et vice versa. Cette nature inhérente aux réseaux ad hoc fait que l'accessibilité de données soit toujours inférieure à celle fournie dans les réseaux fixes.

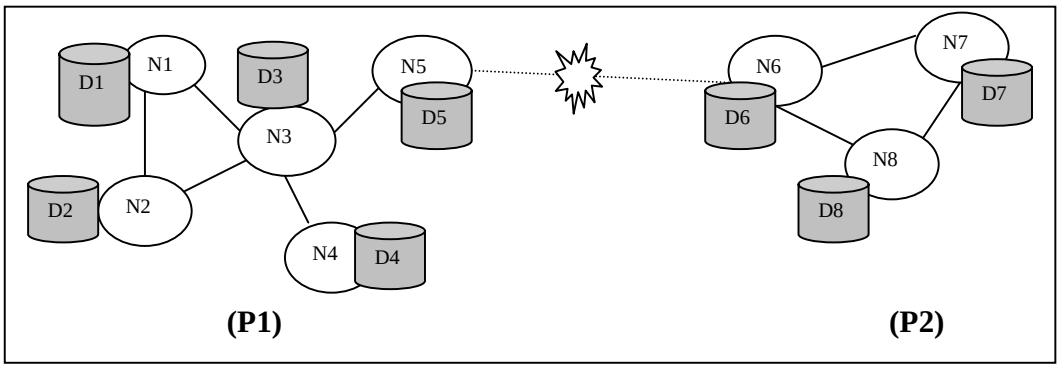


Figure 1.2 : Exemple de partitionnement.

Une solution possible, pour garder les données accessibles, est de créer pour chaque donnée, une copié identique de chaque donnée dans les nœuds de l'autre partition, avant que le partitionnement soit produit. Dans la figure 1.3, si les copies de données sont créées avant l'éloignement des deux partitions, chaque nœud mobile peut accéder aux données après la division de réseau.

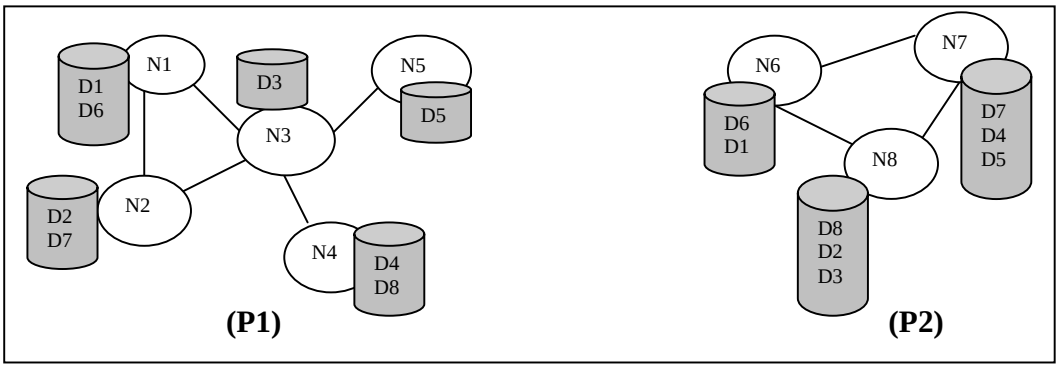


Figure 1.3 : Exemple de création des copies de données.

Cette technique est connue sous le nom de réplication de données. Elle est très efficace pour améliorer l'accessibilité de données. D'autre part, les nœuds mobiles ont généralement des ressources faibles, les nœuds mobiles ne peuvent pas tenir toutes les copies de données. La réplication de données dans les réseaux ad hoc doit prendre en considération la puissance d'énergie modeste et la bande passante limitée aussi bien que la mobilité de toutes unités du réseau.

1.5 Réseau mobile ad hoc et les systèmes peer to peer

Les Peer to peer (P2P) sont des systèmes ayant une organisation automatique, et de gestion décentralisée, dans lesquels les nœuds contribuent et coopèrent à fournir des services. Le réseau mobile ad hoc (MANET) est un système autonome de nœuds mobiles reliés par des liens sans fil, dont l'union forme un réseau de communication avec une arbitraire topologie. Un système P2P se compose par un ensemble de nœuds qui changent dynamiquement. Alors qu'un MANET consiste en des nœuds mobiles communiquant l'un avec l'autre en employant des liens sans fil de multi-sauts.

Les systèmes p2p peuvent être définis comme une architecture réseau, dans laquelle chaque ordinateur possède des capacités et des responsabilités équivalentes [Kor01], [Sch01]. Ces systèmes trouvent leurs applications dans plusieurs domaines, notamment dans le domaine de communication qui regroupe, la messagerie instantanée (AOL, ICQ, MSN Messenger, Yahoo Messenger), la téléphonie par Internet (Webphone, PhoneFree), ainsi que la vidéo conférence (CUseeME, ICUII) et le domaine de partage de fichier à travers plusieurs applications comme Napster, Gnutella, Kazaa, Freenet.

Les P2P et les réseaux ad hoc partagent certaines caractéristiques principales :

- Organisation automatique : La base des systèmes P2P et les réseaux ad hoc sans fil, est le concept de l'auto organisation [Ykx04]. Dans ces deux technologies, aucune entité centrale n'existe, pour contrôler et coordonner le réseau, et aucune pré-configuration n'est acceptée par ces deux environnements.

Le réseau est établi, dès que les participants décideront de créer un réseau, en établissant des connexions entre eux. En plus, le réseau modifié de manière permanente, parce que les nœuds changent le plus probablement leurs connexions entre eux.

- Topologie dynamique : Un nœud dans le P2P et le MANET peut joindre ou laisser le réseau à tout moment et la position d'un nœud dans un MANET change arbitrairement,

ce qui mène à l'instabilité des itinéraires pour tous les nœuds. Le résultat est le changement dynamique de la topologie.

1.6 Conclusion

Un réseau ad hoc est par conception très peu structuré. Il s'agit de nœuds, tous équivalents du point de vue du réseau, qui doivent se charger eux-mêmes d'établir toute l'infrastructure permettant les communications. Nous constatons que les réseaux mobiles ad hoc partagent des caractéristiques principales avec les systèmes P2P. Cette analogie peut aider à exploiter les idées P2P et le développement qu'a connu ces derniers, au profil des réseaux ad hoc.

Les nœuds dans un réseau mobile ad hoc se caractérisent par une mobilité aléatoire. Ce genre de déplacement peut partitionner le réseau en plusieurs parties. Un nœud mobile ne pourra jamais accéder aux données se trouvant aux autres partitions, ceci diminue le taux d'accessibilité de données. L'utilisation de la réplication peut rendre les données accessibles et réduire le temps de réponse. Cette technique doit aussi prêter attention aux ressources limitées caractérisant les nœuds mobiles. Dans le prochain chapitre nous allons étudier les principes de cette technique dans l'environnement mobile à travers les systèmes existants.

Chapitre 2 : La réplication dans l'environnement mobile

2.1 Introduction :

Le développement de la communication sans fil et l'informatique mobile permettent l'accès à l'information indépendamment de la localisation géographique, même en l'absence de l'infrastructure préexistante, cette nouvelle technologie connue sous le nom du réseau mobile ad hoc, ayant pour résultat un nouveau modèle de l'informatique répartie c'est l'informatique peer-to-peer (P2P). Le développement des applications ad hoc à grande échelle nécessite une forte disponibilité de données, le moyen le plus efficace pour garantir cette disponibilité est de produire plusieurs copies(réplication) des mêmes données sur plusieurs sites. Ceci garantit que la charge sur les nœuds et les liens du réseau restent gérables et que les utilisateurs peuvent obtenir l'information désirée dans un temps raisonnable. Cela pose néanmoins deux problèmes. D'abord, il faut savoir où et comment répliquer les données, et ensuite, comment s'assurer que les clients peuvent trouver ces données répliqué.

2.2 Définition de base:

2.2.1 Disponibilité :

La notion de disponibilité est définie dans [Phi94] comme une propriété d'une ressource ou d'un service qui permet de satisfaire les requêtes de l'utilisateur dans les délais impartis. La disponibilité englobe la notion de fiabilité qui représente seulement une assurance de réponse correcte sans contrainte de temps.

2.2.2 Réplication :

La réplication des composants devient un dispositif important exigé par beaucoup de systèmes informatiques pour plusieurs raisons. Elle consiste a créer une ou plusieurs copies de la même ressource sur différents serveurs dans une architecture distribuée. La ressource repliquée peut être: une donnée, un service, une tâche ou une configuration. Le premier but de la réplication est de fournir les systèmes disponibles et d'élever la tolérance aux pannes.

La réplication est également employée pour équilibrer la charge. Les clients peuvent alors émettre les requêtes à n'importe quelle copie dans le système, évitant la surcharge du système et améliorant le temps de réponse.

2.2.3 Objet : Toute information peut être partagée.

2.2.4 Replica(Reproduction):

Un Replica ou une reproduction est une copie d'un objet stockée dans un ou plusieurs emplacements (un emplacement peut être un serveur fixe ou mobile comme le cas d'un reseau adhoc), un emplacement peut stocker des reproductions d'objets multiples.

2.2.5 Cohérence (Consistency):

La cohérence est une propriété qui exprime la convergence entre le contenu des différentes copies d'un même objet, elle assure que deux reproductions ou plus ont la même valeur de données. La synchronisation de reproduction est employée pour établir l'uniformité.

2.3 Avantages et coût de la réplication :

Dans un système centralisé de base de données, tous les clients se relient au même serveur, tandis que dans un système de réplication de base de données, les clients peuvent choisir leur serveur parmi divers serveurs pour répondre à leurs requêtes. Ceci a comme avantages:

2.3.1 La disponibilité élevée

L'existence de plusieurs copies des mêmes données sur plusieurs sites permet d'améliorer la disponibilité et garantit la tolérance aux pannes. Par exemple, si une reproduction tombe en panne en raison d'une panne de logiciel ou de matériel, les reproductions restantes peuvent continuer à fonctionner. Or, dans un système centralisé, l'accés aux données devient complètement indisponible après une panne.

2.3.2 Une meilleure exécution

Le chargement de traitement transactionnel peut être distribué parmi toutes les reproductions (machines) dans le système. Ceci mène à deux améliorations:

2.3.3 Un plus grand débit

Les reproductions peuvent indépendamment être intérogées par les requêtes des utilisateurs (lecture ou mise à jour). En cas de mise à jour les modifications doivent être exécutées sur toutes les reproductions.

En raison du fait qu'une partie du chargement de transaction peut être manipulée d'une manière décentralisée, les reproductions peuvent traiter plus de transactions par unité de temps qu'un système centralisé où les demandes d'accès doivent toujours exécutées par un seul serveur central.

2.3.4 Temps de réponse plus courts

Le temps de réponse des demandes est réduit parce que ceux-ci peuvent être exécutés sur un replica local, ou un replica près du client.
Cependant, ces avantages peuvent être compromis par les coûts suivants:

2.3.5 Une communication plus élevée

Les replicas doivent communiquer pour s'assurer que des modifications sont appliquées à toutes les copies de données. Ceci augmente la charge, le traitement et la communication entre les utilisateurs. Le résultat peut etre la saturation du réseau, qui peut dégrader les performances de l'exécution globale.

2.3.6 Une complexité plus élevée de système

Les replicas d'un même objet reçoivent d'une manière asynchrone les demandes de client qui modifient la base de données ou l'information partagée. Sûrement la synchronisation des différents replicas exige des algorithmes d'ordonnoncent complexe.

2.4 Stratégies de réplication :

Les stratégies de réplication qui ont été proposées dans la littérature peuvent être classifiées en deux grandes catégories: *pessimiste* et *optimiste.* Ces stratégies diffèrent dans les hypothèses adoptées et les techniques qu'elles emploient pour mettre à jour et maintenir la cohérence entre les replicas.

2.4.1 Réplication pessimiste :

Les réplications pessimistes donnent aux utilisateurs la possibilité d'avoir une seule copie fortement disponible et mise à jour pour chaque objet. Cette possibilité peut être réalisée de plusieurs manières, mais les concepts de base restes les mêmes. Elles interdisent tout accès à une reproduction qui est en mise à jours. Par conséquent elles garantissent une forte cohérence entre les replicas. Parmi les techniques utilisées pour ce type de réplication on a :

2.4.1.1 Copie primaire

Cette technique[Ald76], [Sto79] utilise une reproduction (replica) en tant que primaire, responsable de toute opération de mise à jour. Autrement dit, le nœud qui a la copie primaire gère toute requête de type « update » voir (figure 2.1). Les autres replicas agissent en tant que secondaires et reçoivent seulement les changements du primaire pour qu'ils soient à jour. Les opérations de lecture peuvent être effectuées à n'importe quel replica de l'objet logique.

Cette technique a été largement appliquée dans les systèmes commerciaux de base de données [Kro86], [Ora96], [Die94], [Ber97].

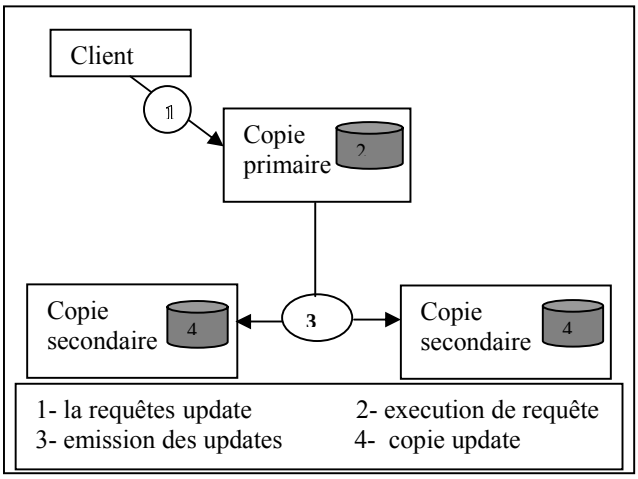


Figure 2.1: update message in primary copy

En cas de partition du réseau, seule la partition contenant la copie primaire peut y accéder, et les modifications sont envoyées aux sites des autres partitions lors du recouvrement de réseau. Il est possible de désigner une autre copie primaire parmi les copies secondaires qui existent si l'échec de la copie primaire est détecté. Sinon, le système doit toujours supposer qu'une partition s'est produite et aucune désignation ne peut être faite.

2.4.1.2 Notion de quorum :

Dans cette technique[Gif79] une opération de lecture ou d'écriture ne peut être effectuée sur un replica, que si elle est autorisée par un ensemble de nœuds qui ont le même replica. Cet ensemble de nœuds est appelé QUORUM. On distingue deux types de quorum, un quorum de lecture R et un quorum d'écriture W où les valeurs R et W doivent vérifier les conditions suivantes :

1- $R \cap W \neq \Phi$, cette condition assure la cohérence de données en cas de partitions de réseau. Elle signifie qu'un replica ne peut pas être lu dans une partition avec le quorum de lecture et modifié dans une autre partition avec un quorum d'écriture.

2- $W > V/2$ où v est le nombre total de nœuds ayant le même replica. Cette condition assure en cas de partition de réseau, qu'une opération d'écriture ne sera jamais produite dans deux partitions.

L'inconvénient de cette approche est que l'opération de lecture ou d'écriture est assez chère, car elle nécessite un avis de tous les membres de quorum.

2.4.1.3 Avantages et limitations :

Les techniques pessimistes de réplication s'adaptent bien aux environnements de gestion de réseau local statique. L'application de ces techniques dans un environnement distribuée à grande échelle pour garantir de bonnes performances et un niveau acceptable de disponibilité ne peut pas être assuré [Yma02], pour les raisons suivantes :

- Tandis que le nombre de replicas augmente, la disponibilité de la lecture s'améliore mais la disponibilité de modification diminuée. Donc il est difficile avec les mises à jour fréquentes et un nombre de sites plus élevé d'établir un système de réplication pessimiste[Yuv01], [Yuv02]. Ce qui permet de dire que les stratégies pessimistes s'adaptent mal aux réseaux à déconnexions fréquentes et aux réseaux à grande échelle. C'est pourquoi plusieurs réseaux statiques et mobiles ont opté pour des techniques optimistes, par exemple USENET [Spl98], DNS [All01], et systèmes de base de données mobiles Bayou [Pst97].
- Quelques activités humaines exigent le partage optimiste de données. Par exemple dans l'ingénierie coopérative ou le développement des logiciels, les personnes veulent travailler simultanés mais en séparation relative. Une technique pessimiste est mal adaptée. Il vaut mieux leur permettre de mettre à jour les données indépendamment et de réparer les conflits occasionnels après qu'ils soient produis.

2.4.2 Réplication optimiste :

La réplication optimiste permet aux utilisateurs d'accéder à n'importe quel replica, à n'importe quel moment. Autrement dit, la réplication optimiste ne limite pas la disponibilité. La demande d'utilisateur peut être servie localement et immédiatement, contrairement à la réplication pessimiste qui bloque les demandes de lecture s'il y a des modifications en cours d'exécution pour le même replica.

La réplication optimiste est utilisée dans plusieurs applications, y compris la gestion des données dans un environnement mobile, comme par exemple le système Bayou [Pst97], où les utilisateurs sont fréquemment déconnectés, ils peuvent avoir leurs propres copies privées et les modifier concurremment. Après que le travail soit effectué, une procédure de gestion de conflit est exécutée. Plusieurs techniques sont proposées pour concrétiser cette stratégie de réplication. On trouve :

2.4.2.1 Vecteurs De Version (Version Vectors) :

Pour chaque replica d'un objet, on associe un vecteur de N éléments où N est le nombre de nœuds dans lesquels une copie de cet objet est enregistrée. A chaque mise à jour un nouveau vecteur est crée en incrémentant l'entrée correspondant au nœud qui généré cette modification.

Supposant qu'on a deux vecteurs VVi et VVj correspondant respectivement aux nœuds i et j tel que le k^{eme} élément d'un vecteur indique le nombre de mises à jour effectués par le nœud k, les vecteurs sont échangés entre les nœuds et les conflits sont détectés comme suit :

(1) si *VVi = VVj* alors les replicas n'ont pas été modifiés.

(2) si VV*i* domine *VVj* alors l'objet a été modifié seulement au nœud identifie par i, et le nœud j doit substituer son replica et son vecteur (VVi domine VVj ssi VVi[k]>=VVj[k], et $1 \leq K \leq N$).

(3) autrement, il y a un conflit de mise à jour et l'incohérence ne peut pas être résolue.

La taille occupée par des vecteurs de version dépend linéairement du nombre de replicas créé dans le système. Ceci peut être considéré comme un obstacle significatif de scalabilite. En plus la limitation principale de cette approche est que l'ensemble des nœuds qui ont le même replica est considéré statique. Chaque nœud a une position fixe pré-assignée dans un vecteur de version. Ceci signifie que la création ou le retrait des nœuds qui tiennent le même replica est interdite par cette approche.

Une solution de ce problème est celle utilisée dans Bayou[Pst97]. Cette approche manipule la création et le retrait des nœuds comme des opérations de mises à jour.

2.5 Protocoles de réplication et quelques systèmes existants:

2.5.1 Coda :

Coda est un système de fichier distribué comme le système AFS [Msc86] (the Andrew File System). Il peut être considéré comme un systeme de replication basé sur une architecture client serveur dans un réseau fixe, où les fichiers sont dupliqués vers les clients qui les sauvgardent localement pour augmenter les performances en utilisant la stategie pesimiste de replication. en cas de panne de serveur ou de partition de réseau, la disponibilité nécessite une forte politique de cohérence pour assurer la similitude entre les différentes copies.

L'accès d'une machine mobile à l'information sur le système Coda est possible [Sat02]. Un client peut être dans l'un des trois états suivants : hoarding, emulation et reintegration (voir Fgure2.2).

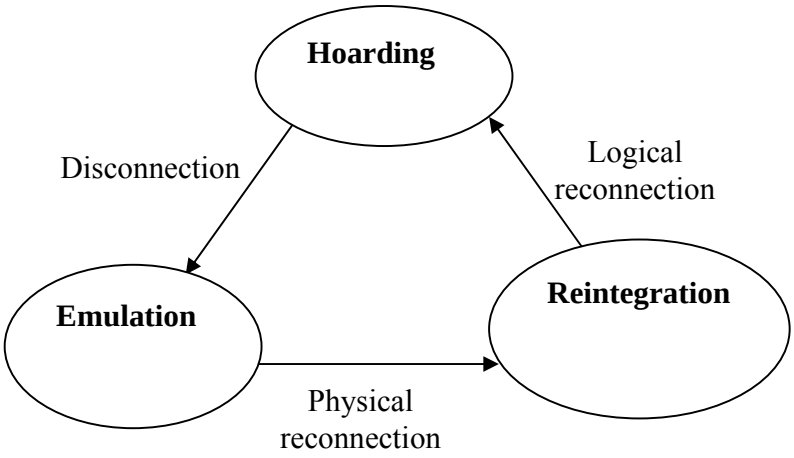


Figure 2.2 : Les états de client dans Coda

.

Dans l'état ***Hoarding***, le client est connecté à un serveur qui contient l'information. Tandis que dans cet état, le client peut envoyer ses demandes au serveur et exécuter son travail. Simultanément, il essayera également de maintenir son cache en dupliquant les données utiles (par exemple, fichiers, attributs de fichier, et répertoires). Dans ce cas le système Coda utilise la stratégie pessimiste de réplication (primary copy). En cas de déconnections le client passe à l'état d'***Emulation*** dans lequel toutes les demandes de fichier(lecture ou écriture) peuvent servir localement en utilisant la stratégie optimiste de réplication.

Quand la reconnexion se produit, le client est dans l'état de ***Reintegration*** dans lequel il transfert toutes les modifications effectuées en cours de l'état précédent au serveur et mis à jour son cache. Il est possible que pendant la réintégration la connexion avec le serveur soit détruite encore, remettant le client de nouveau dans l'état d'*Emulation*.

Evidemment, dans ce contexte on peut avoir différentes versions pour un même fichier, et la question qui doit être posée est comment cette incohérence peut être détectée et résolue. La solution adoptée par Coda est l'aproche basée sur le vecteur de version (version vectors), tel que chaque replica est assigné à un Coda Version Vector (CVV). Si un client modifie un fichier, le serveur disponible envoie ces modifications et le CVVc de ce client avec un message de mise à jour aux autres serveurs. Chaque serveur i caractérisé par son CVVi vérifie si CVVc >= CVVi. Si oui, la nouvelle mise à jour est appliquée au replica du serveur et son CVVi associé à ce replica est modifié. Si non, un conflit est détecté, ce fichier est marqué inopérable et son propriétaire est annoncé. Si le client est dans l'état émulation alors, ce procédé est retardé jusqu'à la phase de réintégration.

Enfin Coda peut être considéré comme une solution pertinente et fortement disponible pour les réseaux mobiles ou les systèmes faiblement reliés basés sur un serveur d'infrastructure. Néanmoins il n'est pas convenable pour un réseau ad hoc car la propagation des mises à jour est effectuée par un serveur d'infrastructure.

2.5.2 Roam

Roam est un système de réplication de fichier dans un environnement mobile [Rat98] et [Rat99]. Il est semblable aux systèmes Ficus [Hei97] et Rumor [Guy98]. Roam utilise la réplication optimiste signifiant que chaque client peut avoir un ou plusieurs replica dans sa mémoire locale et servir les demandes sans avoir besoin d'accéder au serveur.

Roam utilise la transmission « peer to peer » à travers le modèle de Ward(wide area replication domains) qui se compose d'un groupe de clients mobiles qui sont géographiquement près l'un de l'autre.

Tous les clients dans un groupe de Ward peuvent communiquer en utilisant la relation « peer to peer ». Dans telle situation la transmission peut être intermittente ou de mauvaise qualité(voir figure2.3). Un client peut potentiellement faire partie de plusieurs groupes où chaque groupe est représenté par un utilisateur maître qui relie son groupe avec les autres groupes. Le maître du groupe doit connaître tous les réplicas se trouvant dans son groupe. Les groupes et les maîtres des groupes sont créés, contrôlés, et détruits dynamiquement durant toute la vie du système.

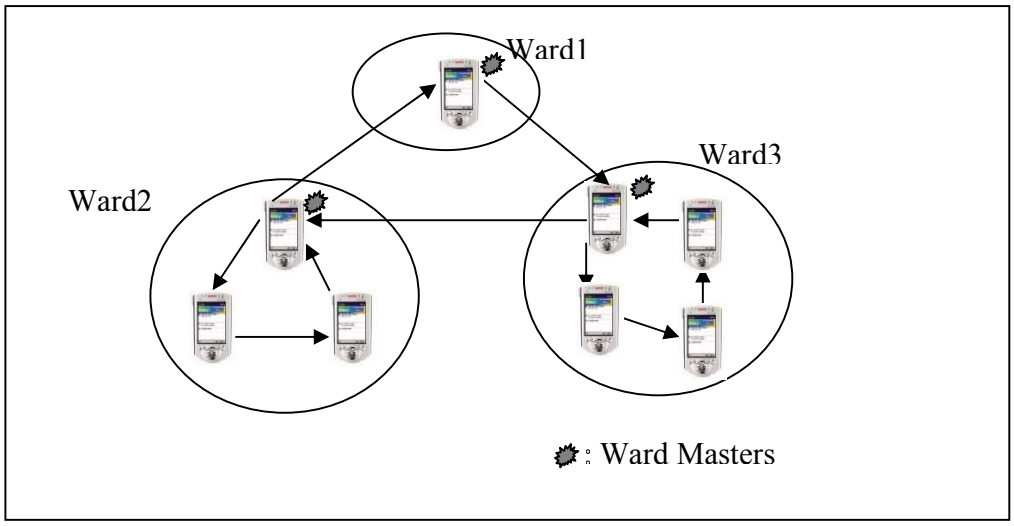


Figure2.3 : les modèles de Ward.

Pour contrôler la cohérence de données entre les clients d'un même groupe ou entre les maîtres de différents groupes, les clients communiquent dans une topologie de boucle.

Essentiellement, les clients communiquent seulement avec leurs voisins sur la boucle pour recevoir des mises à jour aux données partagées. Un replica cible tire toutes les mises à jour d'un replica source. La cible se renseigne sur toutes les mises à jour que la source avait faites ou reçues, mais la source n'apprend rien de la cible. Puisque les clients sont arrangés dans une boucle, et la transmission est à sens unique. Pour déterminer si un fichier est plus récent ou pas, pour faire les mises à jour appropriées, Roam utilise l'approche d*es vecteurs de version* (voir le paragraphe 2.4.2.1) pour chaque replica.
L'avantage principal de l'utilisation d'une topologie de boucle pour échanger des mises à jour est la limitation de nombre de messages envoyés entre les utilisateurs dans un groupe.

2.5.3 Bayou

Bayou [Ter95], [Pet97] est un système de réplication de base de données mobile. Il utilise la réplication optimisme sur un groupe de machines pour supporter des applications de collaboration et un modèle de transmission « peer to peer ». Contrairement au système Roam, le système Bayou n'impose aucune restriction sur la transmission entre les utilisateurs. N'importe quel utilisateur peut communiquer avec n'importe quel autre. En conséquence, la disponibilité des services peut être augmentée. Les serveurs propagent ensuite ces modifications entre-eux suivant un protocole de réconciliation incrémental (*anti-entropy protocol*) [Pet96].
Bayou est un système orienté applications. Il offre des mécanismes grâce auxquels l'application peut spécifier la résolution des conflits [Ter95],[Ter98] et paramétrer le protocole de réconciliation entre les serveurs [Edw97],[Pet97]. On distingue deux types de conflits :

(i) Les conflits *write/write* lorsque deux clients mettent à jour la même donnée de manière différente.

(ii) Les conflits *read/write* lorsqu'un client met à jour une donnée en se basant sur la lecture d'une donnée en train d'être mise à jour.

La détection de conflits s'effectue lors des opérations de mise à jour. Chaque opération de mise à jour est associée à une pré-condition définie par l'application (*dependency check*). Si celle-ci n'est pas satisfaite, il y a conflit et le serveur invoque alors une procédure de résolution de conflit (*merge procedure*) également définie par l'application.
Les auteurs donnent un exemple de système de réservation de chambre quand la réservation d'une chambre pour une heure est demandée. Le contrôle de dépendance (*dependency check*) s'assure que la salle est réellement libre pendant cette heure et alors la réservation est effectuée. Sinon la procédure de fusion(*merge procedure*) est exécutée

pour voir si la demande peut être satisfaite autrement. La procédure de fusion pourrait essayer de réserver une salle différente ou peut-être programmer la même salle à une heure différente.

2.5.4 Deno

Deno [Kel99] est un protocole de réplication pour les environnements mobiles et les réseaux qui ont une faible connectivité. Il combine une stratégie de mise à jour par vote (quorum voir le paragraphe 2.4.1.2) avec une stratégie de diffusion par Epidémie. L'avantage de la diffusion Epidémique est qu'elle ne pose aucune contrainte structurelle ou topologique. Un terminal mobile maintient ses voisins lors d'une diffusion et établit une communication par paire. Les avantages de la validation par vote sont :

(i) Le protocole est totalement décentralisé et aucune donnée n'est primaire : tous les terminaux jouent le même rôle,

(ii) La disponibilité est augmentée : il n'est pas besoin que tous les terminaux soient disponibles, un certain nombre de votes suffit,

(iii) La réconciliation n'est plus un problème : entre deux modifications de copies, il y a un vote pour choisir la copie cohérente.

Deux problèmes se posent pour effectuer un vote dans un environnement mobile : le maintien d'un groupe de votants et la non terminaison d'un vote. Chacun des participants doit maintenir l'ensemble du groupe des participants (ou son nombre) pour savoir qui a vote et si une décision est possible : le surcoût occasionné par la maintenance du groupe est d'autant plus important que la mobilité est forte. Les déconnexions peuvent également :

(i) Fausser un vote : Un vote peut échouer parce que certains terminaux ne peuvent voter positivement,

(ii) Etre bloquées : Un vote peut être dans l'attente de certaines voix.

Deno résout ces problèmes avec une technique de crédit ou porte-monnaie (*Currency Mechanisms*) [Kel00], [Cet00a]. Chaque votant possède un montant de monnaie par fichier. Ce montant reflète la capacité du terminal à maintenir celui-ci. Lorsqu'un vote est propagé, chaque votant peut décider de mettre un montant de monnaie dans le porte-monnaie. Lorsque le porte-monnaie atteint un certain montant prédéfini, le vote est entériné. Si le porte-monnaie ne peut plus être propagé, le vote échoue. Pour ne pas fausser le vote lors de déconnexions, des *proxies* sont déployés et votent de manière transparente selon des comportements définis par les terminaux portables [Cet00b]. Dans le protocole de base, la stratégie de cohérence assure une exécution faiblement cohérente. Une extension [Cet01] permet une cohérence forte en fournissant une exécution

globalement serialisable et un ordre unique de validation des transactions de mises à jour. Ces deux variantes autorisent les requêtes et les validations locales sur le terminal portable, sans aucun blocage.

2.5.5 AdhocFS :

AdhocFS [Bou02] est un système de distribution de fichier conçu pour un environnement Ad hoc. Il considère l'existence d'un serveur fixe appelé *Home server* et un ensemble d'outils mobiles sans fils qui sont fréquemment déconnectés du serveur nommé *Adhoc groups.* Cependant le but de AdhocFS est de garantir la disponibilité de données entre les utilisateurs mobiles en cas de l'absence de serveur. Les utilisateurs mobiles peuvent obtenir des replicas de différents données en connectant au serveur home. En cas de disconnexion les utilisateurs mobiles forment un groupe ad hoc, où chaque utilisateur mobile peut acceder a ses voisins par un seul saut, AdHocFS contrôle dynamiquement les changements de membres des groupes ad hoc.

Le groupe ad hoc partage quelques similitudes avec le modèle de Ward du système Roam. Cependant, il existe une différence principale, car dans le cas du Roam, les utilisateurs sont groupés sous une topologie de boucle ou la communication s'effectue dans un seul sens, alors que, AdHocFS exploite la connectivité élevée qui existe entre les membres du groupe ad hoc.

Le serveur home estampille les replicas d'un fichier. Chaque estampillage identifie la version actuelle de fichier enregistrée au serveur. Chaque fois qu'une mise à jour est appliquée au fichier, l'estampille est incrémentée. La valeur de replica copiée du serveur à un utilisateur mobile reflète, au début, une valeur stable qui sera modifiée pendant la période de déconnexion. Dans cette période AdHocFS utilise la stratégie optimiste de réplication qui permet à n'importe quel utilisateur de servir les demandes d'une manière indépendante. Dans de telles situations, n'importe quelle mise à jour est enregistrée avec des informations concernent le groupe ad-hoc actuel dans une structure de données nomme Log. Dans AdHocFS, l'estampillage et la structure Log constituent le CCL(Coherency Control List).

Quand un utilisateur mobile est reconnecté au serveur home, un protocole d'uniformité est exécuté entre eux, ce protocole compare les CCLs. Si le replica stable du serveur home domine le replica de l'utilisateur mobile, alors le serveur envoie les mises à jour manquantes à l'utilisateur mobile. Sinon, le serveur home reçoit les nouvelles mises à jour et les appliquer en conséquence à sa valeur et incrémente son estampille.

Trois inconvénients principaux peuvent être identifiés dans Ad-HocFS[Joa03]. Premièrement, seulement les serveurs home permettent d'accéder aux valeurs stables de

chaque fichier. Les utilisateurs mobiles tiennent seulement des valeurs expérimentales de repplica. Par conséquent, les utilisateurs mobiles en cas de déconnexion peuvent seulement accéder à la version expérimentale des fichiers. Deuxièmement les serveurs home agissent en tant que serveurs primaires. Ces caractéristiques peuvent perturber les mises à jour en cas de partition du réseau ou la panne des serveurs homes. Enfin, le traitement de l'incohérence exige que le contenu du CCLs de deux reprlicas soit examiné. Ce traitement peut prendre beaucoup de temps d'exécution et consomme la bande passante, ce qui agit négativement sur la qualité des services.

2.6 Conclusion

Dans ce chapitre, nous avons présenté les aventages et les inconviniants de la réplication de données. Nous avons vu que cette technique donne de méilleures performances. Elle permet d'augmenter la disponibilité des données et d'équilibrer la charge autour du serveur.

Les données répliquées peuvent être synchronisées en utilisant deux stratégies de réplications. La stratégie pessimiste qui n'autorise la modification que sur la copie primaire. De cette façon, elle augmente la cohérence entre les copies mais oblige les clients à transférer les modifications au serveur. Ceci peut inonder le serveur quand la taille du réseau et le nombre de modifications augmentent. La stratégie optimiste ne limite pas la diponibilité, elle autorise l'utilisateur à accéder à n'importe quelle copie et à n'importe quel moment.

Les solutions disponibles supposent l'existence d'un serveur fixe et conçoivent leurs propositions en se basant sur cette hypothèse. La réplication de données dans les réseaux mobiles ad hoc ne peut pas avoir cette hypothèse, parce que l'environnement ad hoc étend la notion de la mobilité à tous ses éléments. Ceci va compliquer l'utilisation de la réplication dans cette environnemt. Dans le prochain chapitre, nous allons donner une présentation synthétique des différentes solutions qui existent et qui résolvent le problème de disponibilité dans les réseaux mobiles ad hoc.

Chapitre3 : La disponibilité dans un réseau Ad hoc

3.1 Introduction

L'augmentation de dispositifs mobiles a donné l'occasion de développer de nouvelles applications que nous voulons faire fonctionées à grande echelle. En plus les services à accès distant et le partage des données constituent les principales applications dans un environnement ad hoc. Par exemple dans un champ de bataille, les soldats forment un réseau ad hoc. Ils devraient pouvoir recueillir la dernière information telle que les ordres et partager cette information entre eux.

Cependant, ces nouveaux services doivent faire face à plusieurs restrictions importantes dues aux caractéristiques liées à l'environnement ad hoc. Notamment La topologie dynamique, la limitation de la bande passante, la durée de vie limitée des batteries et les débranchements fréquents, dus aux mesures d'économie d'énergie, au partitionnement du réseau et aux changements de position [Cla03], ne sont pas des facteurs négligeables. Ils mettent des contraintes sur la disponibilité et l'intégrité du service offert.

Une solution possible pour ce problème est de répliquer les données sur plusieurs nœuds, en plus, dans tel environnement fortement dynamique, en termes de mobilité et de changements de position, un mécanisme de gestion de localisation est nécessaire afin d'assurer que les serveurs mobiles peuvent être localisés et atteints à tout moment [Imi92].

3.2 La réplication dans les réseaux mobiles ad hoc

En plus des avantages de réplication cités au préalable, la réplication dans les réseaux mobiles ad hoc vise à garantir les conditions suivantes:

1. Assurer la disponibilité en cas de panne totale de nœud, à cause de chute de capacité de la batterie ou d'une attaque de déni de service.
2. Garantir l'accessibilité même lorsque la copie initiale est inaccessible.
3. Améliorer la qualité du service (QOS) entre les clients et les serveurs.
4. Assurer l'actualisation et l'uniformité des différents replicas.

La réplication dans l'environnement ad hoc est employée pour améliorer la disponibilité du service qui est un ensemble de données tenues par un serveur mobile et pour augmenter le taux d'accessibilité de données.

3.2.1 Réplication de services :

Ce protocole vise à garantir la disponibilité des services sur un réseau ad hoc en cas de partitionnement, les auteurs dans [Wan02a], [Wan02b] et [Wan03] supposent que :

1- Les nœuds se déplacent selon des groupes corrélés, en utilisant un système GPS (global positioning System) pour savoir la vitesse de tous les nœuds mobiles. On peut déduire des informations sur le changement de la topologie de réseau et donc prévoir une future division de réseau. Par conséquent, les serveurs centralisés peuvent dupliquer des services à l'avance, pour garantir la disponibilité.
2- Chaque nœud est identifié par un seul identificateur. Il est capable de connaître sa position et de mesurer sa vitesse actuelle par l'intermédiaire de GPS.

Considérant un ou plusieurs services dans le réseau, un serveur est un nœud qui tient un ou plusieurs services, les autres nœuds sont des clients, Un nœud client devient un serveur quand il reçoit un réplica de service. Les clients ont besoin d'accéder d'une manière continue au moins à un des serveurs existants pour obtenir des données. Les clients envoient leur *identificateur, position* et *information* de vitesse quand ils envoient leurs demandes au serveur.

Pour garantir la disponibilité de service à tous les clients dans un réseau ad hoc partitionable, un serveur doit repliquer le service sur les nœuds qui vont partir avant qu'ils se séparent complètement. Ceci est illustré dans la **figure3.1**. Un serveur simple *S* sert deux groupes de mobilité (*Cj* et *C*k) qui se déplacent à différentes vitesses et directions, et *S* appartient au groupe *Cj*. Au début, les zones d'accessibilité des deux groupes superposent, ainsi le serveur *S* est accessible à tous les nœuds. Cependant, après que les groupes se séparent, les nœuds dans le groupe *Ck* ne peuvent pas accéder au service de S. Par conséquent, S doit dupliquer le service juste avant la séparation.
En effet, le serveur S doit *prévoir la division de réseau* et le temps convenable pour dupliquer son service. Par conséquent, *S* doit détecter les groupes de mobilité dans ses clients et son propre groupe pour savoir à quel client il devrait dupliquer le service.

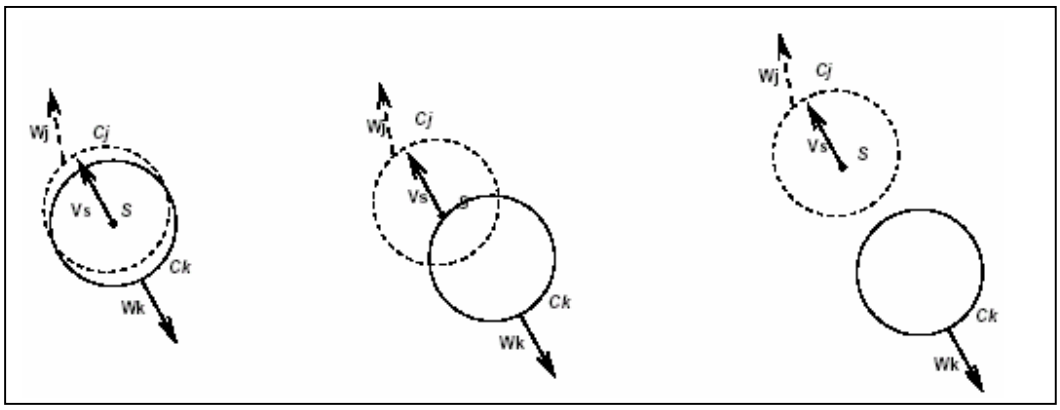


Figure 3.1 : la division des clients d'un serveur**.**

Pour effectuer cette tâche, les serveurs utilisent un algorithme SV (*sequential clustering)* [Wan02b]. A chaque intervalle du temps, chaque client exécute un algorithme DG (*distributed grouping algorithm)* [Wan03] qui permet pour chaque client de construire son groupe stable et de découvrir un ensemble de serveurs. Un client choisit le meilleur serveur parmi les serveurs existants, le meilleur serveur est celui dont la vitesse lui permettra de rester le plus de temps possible dans le groupe des clients.

Ce protocole permet de garantir la disponibilité de service en cas de division de réseau. Cependant, il n'est pas destiné pour tous les réseaux ad hoc car il utilise la notion de groupe et le système GPS qui est très cher. Il nécessite un abonnement et les entités mobiles doivent équiper par un récepteur GPS. En plus la définition de réseau mobile ad hoc élimine l'utilisation des infrastructures préexistantes. Cependant l'utilisation du système GPS peut être considérée comme une infrastructure préexistante.

3.2.2 Réplication de données :

Les protocoles de réplication de données peuvent être divisés en deux classes : protocole de réplication de données sans mise à jour et protocole de réplication de données avec mise à jour qui peut être périodique ou apériodique.

3.2.2.1 Réplication de données sans mise à jour :

Dans [Har01], trois méthodes de réplication ont été proposées SAF, DAFN, DCG dans un modèle de système composé d'un ensemble M de nœuds tel que M= M1, M2, ……Mm

où m est le nombre total de nœuds et Mi ($1 \leq i \leq m$) l'identificateur de chaque nœud et d'un ensemble de données D = D1, D2….. Dn où n est le nombre total de ces données, Dj ($1 \leq j \leq n$) est l'identificateur de donnée. Chaque donnée est tenue comme une donnée originale par un nœud mobile particulier. Dans cet environnement l'auteur suppose que :

1. Chaque nœud mobile a l'espace mémoire de C données élémentaires pour créer les replicas.
2. Les données élémentaires ne sont pas mises à jour.
3. Toutes les données sont de même taille.
4. Les fréquences d'accès aux données élémentaires de chaque nœud mobile sont connues et ne doivent pas changer.

L'accès à une donnée élémentaire est réussi dans deux cas:
Soit le serveur mobile qui reçoit la demande tient l'original ou le replica de donnée élémentaire concernée, soit, il existe au moins un nœud mobile qui tient l'original ou le replica de cette donnée élémentaire demandée qui est relié par un seul saut ou par plusieurs sauts au serveur mobile. Autrement, la demande échouera.
Les replicas sont replacés (changés la place) dans une période spécifique (relocation period), la nouvelle place est déterminée à chaque période en se basant sur l'accès fréquent de chaque nœud mobile à chaque donnée élémentaire en prenant compte la topologie actuelle.

3.2.2.1.1 La méthode SAF (Static Access Frequency)

Dans cette méthode, chaque nœud mobile crée des replicas de C données élémentaires dans l'ordre décroissant des fréquences d'accès à chaque période de relocation, et Les nœuds mobiles n'ont pas besoin d'échanger l'information entre eux pour l'attribution de replica. Le réplica est crée quand l'accès à cette donnée est réussi ou le nœud a pu connecter à un autre nœud qui possède l'original ou le réplica de cette donnée.
Un exemple d'exécution de cette méthode est illustré dans la figure 3.2, tel que une donnée élémentaire Di est une donnée originale de nœud Mj si i=j sinon le Di est un replica. La fréquence d'accès de chaque nœud mobile à chaque donnée élémentaire est donnée dans le tableau1 de la figure3.2.

Data	Nœuds mobiles					
	M1	M2	M3	M4	M5	M6
D1	0.65	0.25	0.17	0.22	0.31	0.24
D2	0.44	0.62	0.41	0.40	0.42	0.46
D3	0.35	0.44	0.50	0.25	0.45	0.37
D4	0.31	0.15	0.10	0.60	0.09	0.10
D5	0.51	0.41	0.43	0.38	0.71	0.20
D6	0.08	0.07	0.05	0.15	0.20	0.62
D7	0.38	0.32	0.37	0.33	0.40	0.32
D8	0.22	0.33	0.21	0.23	0.24	0.17
D9	0.18	0.16	0.19	0.17	0.24	0.21
D10	0.09	0.08	0.06	0.11	0.12	0.09

Tableau 1 : Les fréquences d'accès.

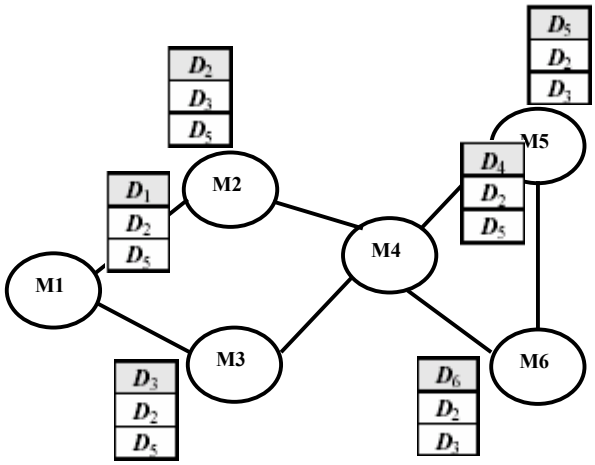


Figure 3.2 : exemple d'exécution de SAF.

Cette méthode permet de réduire le trafic. Cependant, elle diminue la disponibilité de données dans le cas où les nœuds mobiles ont les mêmes caractéristiques d'accès car la méthode est basée seulement sur la fréquence d'accès à des données élémentaires dans l'exemple précédant, les données élémentaires D 7, D 8, D 9, D10 ne sont pas assignées, en plus des replicas de même donnée élémentaire peuvent être assignées à plusieurs nœuds voisins par exemple D2 est assignée à M1, M3 et M4.

3.2.2.1.2 La méthode DAFN (Dynamic Access Frequency and Neighborhood)

Cette méthode tente de résoudre les problèmes de la méthode SAF, notamment le problème de produire plusieurs replicas de même donnée élémentaire. En premier lieu la méthode DAFN est exécutée comme la méthode SAF, ensuite s'il existe un replicas de la même donnée élémentaire entre deux voisins, le nœud qui a la fréquence d'accès inférieure à cette donnée élémentaire change le replica par un autre replica qui n'est pas assignée par exemple les replicas D2, D5 existent entre M1 et M2 (Figure3.2) sont remplacés respectivement par D7 dans M1 et par D8 dans M2 (Figure3.3). Cette méthode est exécutée en suivant les étapes suivantes :

- A chaque période de relocation, chaque nœud diffuse son identificateur et ses fréquences d'accès aux données élémentaires. Cette opération va permettre au nœud mobile de connaître tous les nœuds connectés.

- Chaque nœud mobile crée des réplicas localement, en basant sur la méthode SAF décrite précédemment.
- Après la création des réplicas, dans chaque ensemble des nœuds, le nœud qui a le numéro identificateur le plus bas commence à vérifier deux à deux la duplication des réplicas entre ses voisins. Si c'est le cas, le nœud qui a une fréquence d'accès inférieur, doit remplacer ce réplica par un autre. S'il existe parmi ces données dupliquées une qui est originale, alors le nœud qui a le réplica de cette donnée doit le changer. Le nouveau réplica est choisi parmi les réplicas possibles, où la fréquence d'accès à ce réplica est la plus haute.

La figure suivante montre un exemple d'exécution de cette méthode. Après l'allocation des réplicas, le nœud M1 commence à exécuter le processus d'élimination des redondances avec M2, où le réplica D2 est remplacé par D7 dans le nœud M1 et D5 par D8 dans le nœud M2. Cette opération doit se continuer, jusqu'à ce que tous les nœuds soient vérifiés deux à deux.

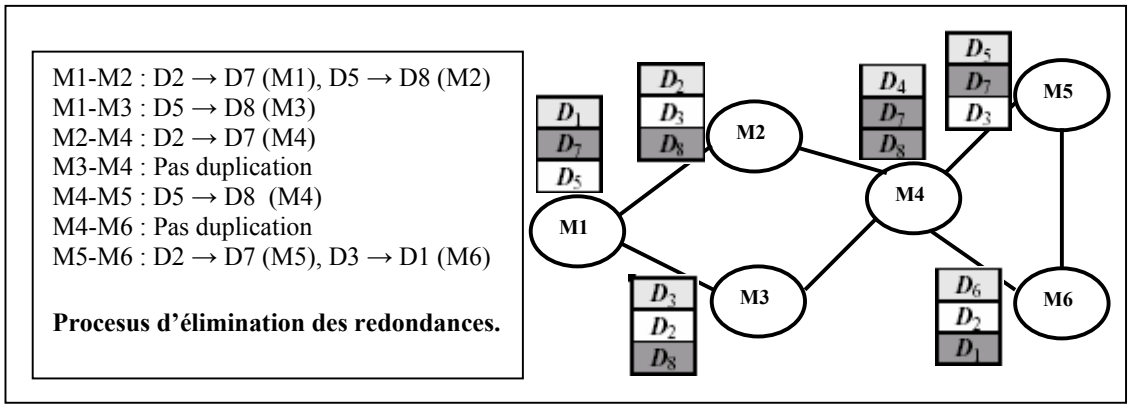


Figure 3.3: Exemple d'exécution de DAFN.

En conséquence, la méthode DAFN fournit une disponibilité de données plus au moins élevée que celle fournie par la méthode de SAF. Néanmoins, le trafic est plus élevé que celle fourni par la méthode de SAF, parce qu'à chaque période d'assignation, les nœuds échangent des informations entre eux. En plus, l'exemple d'exécution (figure3.3) montre que la méthode DAFN n'élimine pas complètement les replicas identiques entre les voisins par exemple D8 est assignée à M2, M3 et M4.

3.2.2.1.3 La méthode DCG (Dynamic Connectivity based Grouping) :

Au lieu de mettre en commun les replicas entre les nœuds voisins (la méthode DAFN), la méthode DCG partage les replicas entre les nœuds mobiles d'un large groupe voir

(Figure3.4). Un groupe est un graphe partiel dont les nœuds sont biconnectés, c.à.d reliés deux à deux. La méthode de DCG crée ces groupes en appliquant l'algorithme de *biconnected components* [Aho74]. Afin d'avoir un partage efficace, chaque groupe devrait être stable, c.à.d, le groupe n'est pas facilement cassé à cause de changement de la topologie du réseau. En supprimant chaque nœud arbitraire dans un groupe, par le regroupement deux à deux des nœuds mobiles, le groupe est considéré stable même si un nœud mobile disparaît du réseau ou un lien est débranché. L'exécution de cette méthode suit les étapes suivantes :

- A chaque période d'assignation, chaque nœud diffuse son identificateur et ses fréquences d'accès aux données élémentaires.
- Dans chaque ensemble des nœuds, le nœud mobile d'identificateur le plus bas exécute l'algorithme de *biconnected components* pour construire les groupes biconnected. Si un nœud représente un point d'articulation comme M4 dans la figure 3.4, dans ce cas, ce nœud doit appartenir au premier groupe trouvé, c-à-d G1.
- Dans chaque groupe, le nœud d'identificateur inférieur, calcule la fréquence d'accès du groupe à chaque donnée, en additionnant les fréquences d'accès des nœuds mobiles du même groupe à la même donnée élémentaire.
- Suivant l'ordre décroissant de fréquence d'accès du groupe, les réplicas sont assignés jusqu'à ce que l'espace mémoire dans le groupe devient plein, en commençant toujours par le nœud d'identificateur inférieur dans le groupe. La création de réplica prend en considération la valeur de fréquence d'accès des autres nœuds du même groupe à cette donnée, tel que un nœud crée un replica si sa fréquence d'accès à cette donnée est supérieure à toutes les fréquences d'accès des autres nœuds membres au même groupe.
- Après l'assignation de tous les réplicas, s'il existe des nœuds qui ont un espace libre dans le groupe, ces nœuds doivent occuper cet espace par des réplica qui ne sont pas assignés localement en choisissant celle qui a une fréquence d'accès supérieure.

La figure3.4, montre un exemple d'exécution de cette méthode, dans cet exemple les nœuds mobiles sont regroupés en deux groupes: G 1 = (M1, M2, M3, M4) et G2 = (M5, M6). Le tableau2 montre les fréquences d'accès de chaque groupe à chaque donnée élémentaire.

Data	Nœuds mobiles					
	M1	M2	M3	M4	M5	M6
D1	0.65	0.25	0.17	0.22	0.31	0.24
D2	0.44	0.62	0.41	0.40	0.42	0.46
D3	0.35	0.44	0.50	0.25	0.45	0.37
D4	0.31	0.15	0.10	0.60	0.09	0.10
D5	0.51	0.41	0.43	0.38	0.71	0.20
D6	0.08	0.07	0.05	0.15	0.20	0.62
D7	0.38	0.32	0.37	0.33	0.40	0.32
D8	0.22	0.33	0.21	0.23	0.24	0.17
D9	0.18	0.16	0.19	0.17	0.24	0.21
D10	0.09	0.08	0.06	0.11	0.12	0.09

Groupe	
G1	G2
1.29	0.55
1.87	0.88
1.54	0.82
1.16	0.19
1.73	0.91
0.35	0.82
1.40	0.72
0.99	0.41
0.70	0.45
0.34	0.21

Tableau 2 : Fréquences d'accès des groupes.

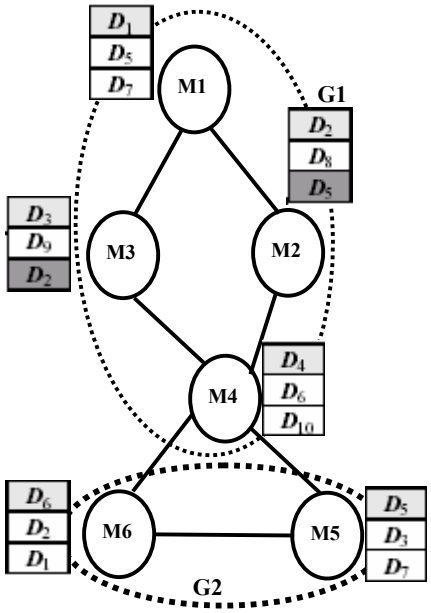


Figure 3.4 : Exemple d'exécution de DCG.

Cette méthode, par rapport à la méthode précédente partage les replicas entre un grand nombre de nœuds, ce qui donne la possibilité de créer une variété de réplicas et permet de dire que la disponibilité peut être augmentée. Cependant, le temps d'atribution des replicas et le trafic sont élevés. En plus il est beaucoup plus probable que le réseau change de topologie pendant l'exécution de cette méthode. Ce qui perturbe l'assignation des replicas.

3.2.2.1.4 Réplication de données basées sur la stabilité des liens

L'évaluation des trois méthodes proposées par T.hara dans[Har01], montre que la méthode DCG augmente le taux d'accessibilité de données, tandis que la méthode SAF diminue le taux de trafic. Dans [Hys03], T. hara et les autres essayaient d'augmenter la disponibilité des données, en prenant en compte la stabilité des liens reliant les nœuds mobiles. Plus les hypothèses utilisées dans [Har01], les auteurs dans ce cas supposent que le nœud mobile se déplace selon le modèle de déplacement aléatoire (Random waypoint Model). Dans ce modèle un nœud mobile choisit d'une manière aléatoire son point de destination et se déplace vers cette destination avec une vitesse quelconque. Chaque nœud mobile connaît sa destination, sa vitesse, son temps de repos et sa position géographique courante en utilisant le système GPS. Dans les réseaux ad hoc, deux nœuds mobiles peuvent être reliés directement entre eux par un lien sans fil.

Le déplacement des nœuds peut produire la déconnexion, quand la distance entre eux est plus longue que la gamme possible de communication. En comparant les mouvements de deux nœuds voisins, le temps de déconnexion peut être estimé et représenté par la variable tij, tels que i et j identifiés les deux nœuds voisins Mi et Mj. Selon l'état de Mi et Mj le calcule de tij peut avoir trois cas possibles :

1- Les nœuds Mi et Mj sont en repos, alors le tij est le minimum des temps de repos qui reste.
2- Les deux nœuds Mi et Mj se déplacent. Alors le temps tij est le temps, où la distance entre Mi et Mj devient plus grande que la gamme possible de communication. Tij doit être inférieur au temps d'arrivée de deux mobiles à leur destination.
3- L'un des deux nœuds se repose, tandis que l'autre se déplace. Le temps tij est toujours le moment où leur distance devient plus grande que la gamme de communication.

la stabilité de lien entre deux mobiles est exprimée par la relation Bij, en utilisant la période de relocation T

$$Bij = \begin{cases} 1 & \text{si } tij >= T \\ tij / T & \text{sinon} \end{cases}$$

Un lien qui a une valeur de Bij plus grande, est un lien plus stable par rapport à un lien qui a une valeur Bij plus petite. En utilisant cette définition de stabilité des liens, T.hara et les autres ont proposé les méthodes de réplication suivantes :

3.2.2.1.4.1 La méthode DAFN-S1 (DAFN - Stability of radio links: 1):

Le principe de cette méthode est le même que la méthode DAFN décrite précédemment, sauf que dans ce cas, la stabilité des liens entre les nœuds mobiles est prise en considération. L'exécution de cette méthode se réalise comme suit :

- À chaque période de relocalisation, chaque nœud mobile braodcasts son identificateur aux autres nœuds. À la fin de cette opération, chaque nœud devrait connaître tous les autres nœuds connectés et qu'il peut communiquer soit directement par un seul saut ou indirectement par plusieurs sauts.
- Chaque nœud détermine les données élémentaires à répliquer en se basant sur la méthode SAF, où les données sont répliquées dans l'ordre décroissant de fréquence d'accès. Puis chaque nœud mobile échange avec tous ses voisins, les informations

de ses données répliquées, la fréquence d'accès à ces données et son mouvement courant.

- Dans chaque ensemble de nœuds mobiles, le nœud qui a l'identificateur le plus bas commence à déterminer la nature du lien en calculant la valeur de Bij. Par exemple si i<j, le nœud mobile Mi calcule la valeur de Bij. Cette valeur doit être comparé à un seuil x pour déterminer la nature du lien reliant les nœuds Mi et Mj :
  1- Si Bij < X, le lien reliant Mi et Mj est un lien instable et doit être ignoré.
  2- Si Bij >= X, dans ce cas le lien reliant Mi et Mj est considéré stable et les étapes suivantes doivent être procédées : s'il existe une copie de même donnée dans les deux mobiles Mi et Mj, alors le replica doit être remplacé. Si les deux copies sont des réplicas, alors le nœud qui a une basse fréquence d'accès à cette donnée, doit la remplacer par une autre donnée.

Un réplica est remplacé par une autre copie de donnée qui a une fréquence d'accès plus élevée parmi les données qui ne sont pas assignées pendant la période de relocalisation.

3.2.2.1.4.2 La méthode DAFN-S2 (DAFN - Stability of radio links: 2) :

Cette méthode est exécutée comme la méthode précédente. La seule différence entre les deux méthodes, est que la méthode précédente calcule la valeur de Bij et utilise un seuil X pour déterminer la stabilité du lien. Alors que la méthode DAFN-S2 calcule la valeur de $(1\text{-}Bij)\times p$ pour chaque paire de nœuds mobiles $M\ i$ et $M\ j$, quand il y a une duplication d'une donnée élémentaire entre eux, où p est la fréquence d'accès de $M\ i$ à cette donnée dupliquée, tel que i<j. Cette valeur représente la probabilité que le nœud mobile accédera à cette donnée élémentaire après avoir été déconnecté de l'autre nœud mobile. Si $(1\text{-}Bij)\times p$ <= X, où X représente une valeur de seuil, le nœud concerné remplace cette donnée par une autre donnée de la même façon que DAFN-S1.

Data	Nœud mobile				
	M1	**M2**	**M3**	**M4**	**M5**
D1	0.65	0.25	0.17	0.22	0.31
D2	0.44	0.62	0.23	0.40	0.11
D3	0.20	0.47	0.50	0.18	0.23
D4	0.10	0.53	0.67	0.60	0.09
D5	0.51	0.28	0.72	0.58	0.71
D6	0.08	0.38	0.43	0.33	0.26
D7	0.17	0.32	0.11	0.49	0.62
D8	0.22	0.24	0.21	0.23	0.57

Tableau3 : Fréquence d'accès aux données.

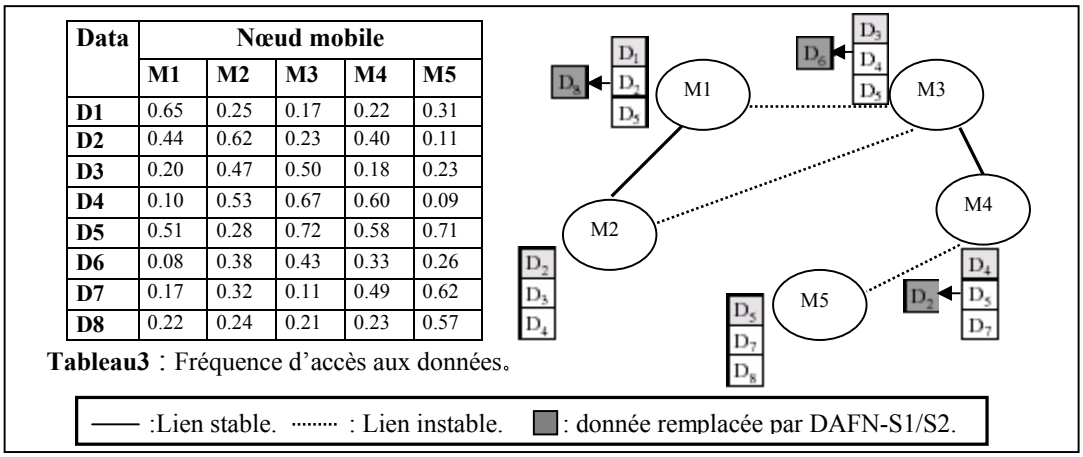


Figure 3.5 : Exemple d'exécution de DAFN-S1 et DAFN-S2.

La figure3.5 explique un exemple d'exécution de deux méthodes DAFN-S1 et DAFN-S2, où un ensemble de cinq nœuds mobiles constituent un réseau ad hoc. Chaque nœud a une donnée originale et possède une capacité mémoire lui permettant de tenir deux réplicas. Le tableau3 donne la fréquence d'accès à chaque donnée par n'importe quel nœud mobile. Les méthodes DAFN-S1 et DAFN-S2 permettent de distinguer les liens instables comme M1-M3, M2-M3 et M4-M5 qui seront ignorés et lien stable comme M1-M2 et M3-M4. Les données élémentaires entre les nœuds voisins reliés par des liens stables sont comparées, et les données identiques sont remplacées, comme D2 est remplacée par D8, D4 par D6 et D5 par D2.

3.2.2.1.4.3 La méthode DCG-S1 (DCG - Stability of radio links: 1):

Cette méthode est dérivée de la méthode DCG proposée dans [Har01]. La différence entre ces deux méthodes et que la méthode DCG-S1 prend en considération la stabilité des liens entre les nœuds mobiles après la construction des groupes biconnected. Le principe de cette méthode est le suivant :

- A chaque période de relocation, chaque nœud broadcasts son identificateur aux autres nœuds, ce qui lui permet de connaître tous les nœuds accessibles. Ensuite chaque nœud échange avec ses voisins les informations concernant son mouvement actuel.
- Construction des groupes biconnected dans chaque ensemble de nœuds, en commençant par le nœud qui a l'identificateur le plus bas. Deux nœuds mobiles Mi et Mj considérés connected si la valeur de Bij >= X, où *X* représente une valeur de seuil. Puis, chaque composant biconnected est mis à un groupe. Si un nœud mobile appartient à plus d'un composant biconnected, il doit appartenir seulement au premier groupe de composants biconnected trouvé. La figure3.6 présente trois cas différents de groupes biconnected.

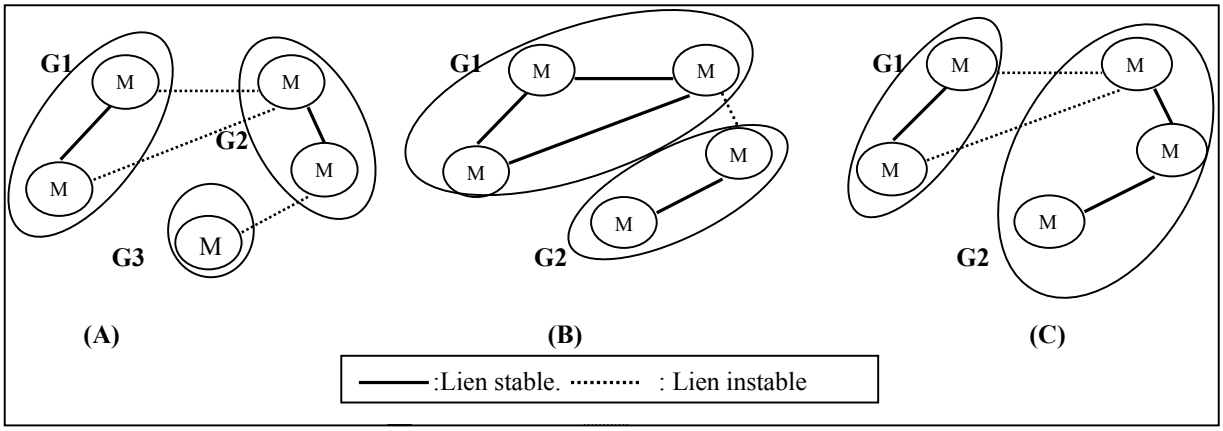


Figure 3.6 : exemple de construction des groupes biconnecteds relies par des liens stables.

- Dans chaque groupe, le nœud qui a l'identificateur le plus bas, calcule pour chaque donnée la somme des fréquences d'accès de tous les nœuds du même groupe, le résultat est une liste des données ordonnées dans l'ordre décroissant des fréquences d'accès du groupe (voir tableau4 de la figure3.7).
- Les replicas sont créés suivant l'ordre décroissant de fréquence d'accès du groupe jusqu'à ce que l'espace mémoire de tous les nœuds dans le groupe devient plein. Les données originales se trouvant dans le groupe ne doivent pas assigner. Chaque nœud doit vérifier avant de créer le réplica, si sa fréquence d'accès à ce réplica est la valeur supérieure parmi toutes les fréquences d'accès des autres nœuds du même groupe.
- Après avoir assigné toutes les données élémentaires, s'il reste un espace mémoire libre aux nœuds mobiles dans le groupe, alors les données accédées par les nœuds du groupe non répliquées, doivent être assignées jusqu'à ce que l'espace mémoire soit plein, en tenant compte de la fréquence d'accès du nœud aux données répliquées.

La figure3.7 présente un exemple d'exécution de cette méthode, le résultat de cet exemple est la création de trois groupes. Les réplicas sont assignés aux membres du groupe dans l'ordre décroissant de fréquence d'accès du groupe aux données répliquées en commençant toujours par le nœud qui a le numéro d'identificateur le plus bas. Par exemple dans le groupe G1 de la figure3.7, le nœud mobile M1 commence à créer les réplicas suivant l'ordre décroissant de fréquence d'accès du groupe G1. M1 ne doit pas créer un réplica dont l'original de ce réplica est tenu par un nœud membre (D2 dans ce cas).

Data	Nœud mobile					Groupe		
	M1	**M2**	**M3**	**M4**	**M5**	**G1**	**G2**	**G3**
D1	0.65	0.25	0.17	0.22	0.31	0.90	0.39	0.31
D2	0.44	0.62	0.23	0.40	0.11	1.06	0.63	0.11
D3	0.20	0.47	0.50	0.18	0.23	0.67	0.68	0.23
D4	0.10	0.53	0.67	0.60	0.09	0.63	1.27	0.09
D5	0.51	0.28	0.72	0.58	0.71	0.79	1.30	0.71
D6	0.08	0.38	0.43	0.33	0.26	0.46	0.76	0.26
D7	0.17	0.32	0.11	0.49	0.62	0.49	0.60	0.62
D8	0.22	0.24	0.21	0.23	0.57	0.46	0.44	0.57

Tableau4: Fréquence d'accès aux données.

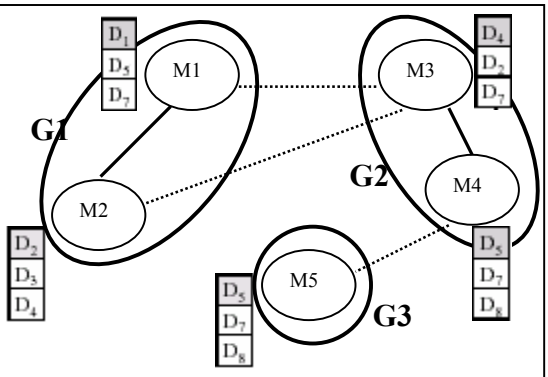


Figure3.7 : Exemple d'exécution de DCG-S1.

Avantages et inconvénients :

En premier lieu, les méthodes DAFN-S1, DAFN-S2 et DCG-S1 héritent tous les avantages et les inconvénients des méthodes DAFN, DCG décrites précédemment. En deuxième lieu, ces méthodes peuvaient améliorer la disponibilité de données car elles partagent les réplicas entre les nœuds qui reliés par des liens stables, notamment avec la méthode DCG-S1, où un plus grand nombre des nœuds peuvent construire un groupe stable et partagent entre eux une variété de données.

En plus la détermination de la stabilité des liens pour ces méthodes, est basée sur le système GPS qui est un système très coûteux et peut être considéré comme une infrastructure préexistante, alors que la conception des réseaux ad hoc néglige l'utilisation de toute infrastructure préexistante. En plus, cette stabilité est basée sur l'estimation de temps restant entre la déconnexion d'un lien et la vitesse de déplacement de nœud. Cependant, cette estimation n'est pas toujours précise, parce que dans un environnement réel, il y'a souvent quelques obstacles qui perturbent la communication par radio, comme les bâtiments par exemple.

3.2.2.2 Réplication de données avec mise à jour :

Dans [Har03], l'auteur a proposé un protocole dérivé de celui proposé dans [Har01]. Les données élémentaires sont mises à jour à des intervalles constants (mise à jour périodique) ou inconstants (mise à jour apériodique). Ceci est fait seulement par le nœud mobile qui tient la donnée originale. Après qu'une donnée élémentaire est mise à jour, ses replicas deviennent inaccessibles. Si les nœuds qui les tiennent ne sont pas reliés au nœud qui tient la copie originale, c'est parce que la mise à jour ne peut pas être propagée.

3.2.2.2.1 Réplication de données avec mise à jour périodique :

Dans [Har02] et [Har03], l'auteur a proposé un protocole dérivé de celui proposé dans [Har01]. Chaque donnée élémentaire est périodiquement mise à jour, L'auteur propose également trois méthodes d'attribution de replica (E-SAF, E-DAFN et E-DCG). L'allocation de replica est basée sur la valeur d'une variable PT qui représente la fréquence d'accès de chaque donnée et le temps restant jusqu'à ce que chaque donnée soit mise à jour, cette variable est définie par la relation suivante :

$$PT = pij \,.\, \tau j = pij \,.\, (Tj - tj).$$

Tel que :

Pij représente la probabilité qu'un nœud mobile *Mi* envoie une demande d'accès a une donnée élémentaire *Dj* à une unité de temps, c.-à-d., la fréquence d'accès.

τj dénote le temps restant jusqu'à ce que *Dj* soit mise à jour.

Tj dénote la période de mise à jour *de Dj (Ti est une valeur stable).*

tj dénote le temps qui est passé après la dernière mise à jour de *Dj*.

Plus les hypothèses adoptées dans [Har01], dans ces nouvelles méthodes, Hara suppose que chaque donnée élémentaire est périodiquement mise à jour. Cette mise à jour est effectuée seulement par le nœud qui a la copie originale de donnée. Le principe de ces trois méthodes est :

3.2.2.2.1.1 La méthode E-SAF (*Extended SAF*) :

Dans cette méthode à chaque période de relocation, chaque nœud crée suivant l'ordre décroissant de PT, un nombre C de réplicas.

La figure3.8 montre un exemple d'exécution de la méthode E-SAF, dans lequel les données élémentaires sont assignées à un nœud mobile suivant leur valeur décroissant de PT. La valeur PT est calculée en utilisant les données du tableau5

Data	Nœuds mobiles						
	M1	M2	M3	M4	M5	M6	τj
D1	0.65	0.25	0.17	0.22	0.31	0.24	50
D2	0.44	0.62	0.41	0.40	0.42	0.46	90
D3	0.35	0.44	0.50	0.25	0.45	0.37	120
D4	0.31	0.15	0.10	0.60	0.09	0.10	60
D5	0.51	0.41	0.43	0.38	0.71	0.20	100
D6	0.08	0.07	0.05	0.15	0.20	0.62	60
D7	0.38	0.32	0.37	0.33	0.40	0.32	40
D8	0.22	0.33	0.21	0.23	0.24	0.17	70
D9	0.18	0.16	0.19	0.17	0.24	0.21	10
D10	0.09	0.08	0.06	0.11	0.12	0.09	20

Tableau5

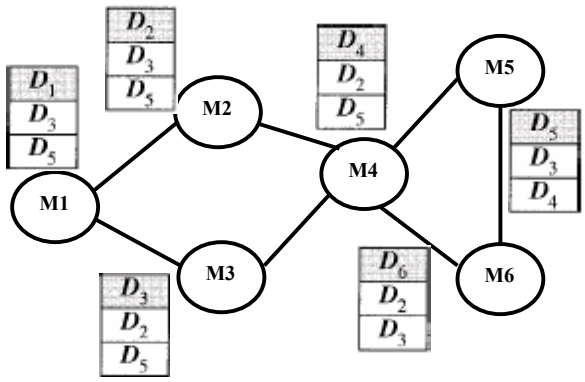


Figure 3.8 : Exemple d'exécution de E-SAF.

Dans cette méthode les nœuds mobiles n'ont pas besoin d'échanger les informations entre eux pour l'attribution du réplica. Ce qui permet de créer les réplicas en diminuant le trafic. D'autre part, dans la méthode E-SAF, chaque nœud crée les réplicas en se basant seulement sur la valeur de PT, ce qui conduit pour les nœuds qui ont les mêmes caractéristiques de créer les mêmes réplicas. Le résultat est la réduction de la disponibilité.

3.2.2.2.1.2 La méthode E-DAFN (*Extended DAFN*) :

Pour résoudre le problème de la méthode précédente, la méthode E-DAFN élimine la duplication de réplica entre les nœuds voisins. D'abord, cette méthode détermine l'attribution des réplicas comme la méthode E-SAF. Ensuite s'il existe une duplication d'un réplica entre deux nœuds voisins, le nœud qui a une valeur PT inférieure pour ce réplica, doit remplacer ce même réplica par un autre.

Dans cette méthode, à chaque période de relocation, les nœuds broadcast leur identificateur, puis chaque nœud créer localement ses réplicas, en se basant sur la méthode E-SAF. Ensuite dans chaque ensemble de nœuds, le nœud d'identificateur inférieur procède à l'élimination des réplicas dupliqués. S'il existe une duplication entre deux voisins, le réplica de valeur PT inférieure doit été remplacé par un autre réplica de valeur PT supérieure parmi les réplicas qui ne sont pas assignés.

Un exemple d'exécution de cette méthode est illustré par la figure3.9. Cet exemple présente le résultat obtenu par l'application de cette méthode sur l'exemple montré dans la figure3.8 précédente.

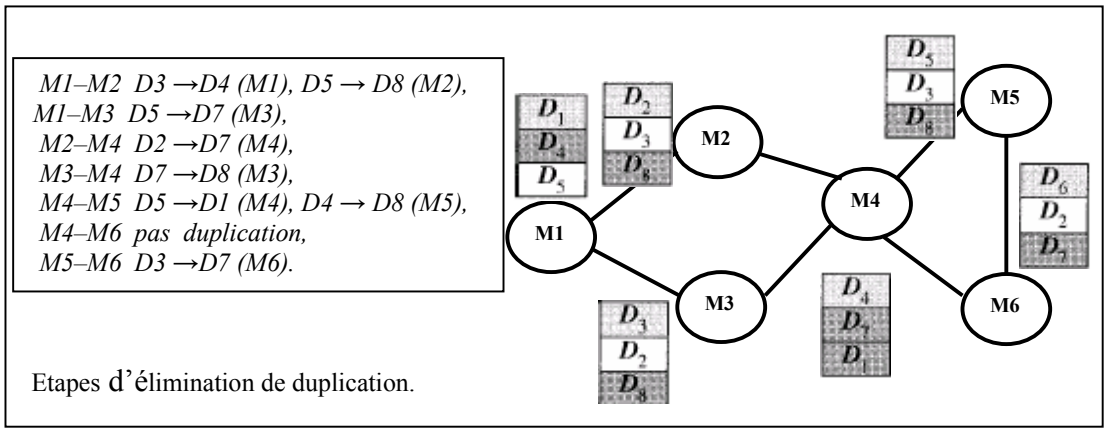


Figure 3.9 : Exemple d'exécution de la méthode E-DAFN.

Par rapport à la méthode E-SAF, la méthode E-DAFN permet d'assigner plusieurs réplicas en les partageant entre les nœuds voisins, la chose qui va augmenter la disponibilité.

Néanmoins, l'exécution de cette méthode produit un nombre important de messages, ce qui affecte les performances en ce qui concerne les points suivants : un temps d'exécution augmenté et un taux de trafic élevé.

3.2.2.2.1.3 La méthode E-DCG (*Extended DCG*) :

Cette méthode utilise la notion de *biconnected components* pour construire les groupes biconnected et partage les réplicas dans un groupe de nœuds, la chose que l'on n'a pas dans la méthode E-DAFN qui partage les réplicas entre les nœuds voisins seulement. Cette méthode est exécutée à chaque période de relocation. Les nœuds broadcast leur identificateur et leurs informations concernant la fréquence d'accès aux données, après cette diffusion, le nœud d'identificateur inférieur commence la construction des groupes biconnected dans chaque ensemble des nœuds, après la construction des groupes la méthode E-DCG suit les étapes suivantes :

- Pour chaque groupe le nœud d'identificateur inférieur calcule la fréquence d'accès de chaque groupe, et la valeur de PT pour chaque donnée élémentaire.
- Dans l'ordre décroissant de la valeur de PT de chaque groupe, les répliquas sont crées dans chaque groupe en commençant toujours par le nœud d'identificateur inférieur dans le groupe, les données originales se trouvant dans le groupe ne doivent pas assigner. Le réplica est crée dans le nœud qui a la valeur PT supérieure pour cette donnée, parmi tous les nœuds qui ont un espace libre dans le groupe.
- Après la création de tous les réplicas, les nœuds qui ont un espace mémoire libre vont créer dans l'ordre décroissant de la valeur de PT, les réplicas qui ne sont pas assignés.

Data	Nœuds mobiles						Groupe	
	M1	M2	M3	M4	M5	M6	G1	G2
D1	0.65	0.25	0.17	0.22	0.31	0.24	1.29	0.55
D2	0.44	0.62	0.41	0.40	0.42	0.46	1.87	0.88
D3	0.35	0.44	0.50	0.25	0.45	0.37	1.54	0.82
D4	0.31	0.15	0.10	0.60	0.09	0.10	1.16	0.19
D5	0.51	0.41	0.43	0.38	0.71	0.20	1.73	0.91
D6	0.08	0.07	0.05	0.15	0.20	0.62	0.35	0.82
D7	0.38	0.32	0.37	0.33	0.40	0.32	1.40	0.72
D8	0.22	0.33	0.21	0.23	0.24	0.17	0.99	0.41
D9	0.18	0.16	0.19	0.17	0.24	0.21	0.70	0.45
D10	0.09	0.08	0.06	0.11	0.12	0.09	0.34	0.21

τj	PT1	PT2
50	166,41	27,5
90	168,3	79,2
120	184,8	98,4
60	69,6	11,4
100	173	91
60	21	49,2
40	56	28.8
70	69,3	28.7
10	7	4.5
20	6.8	4.2

Tableau6 : les valeurs de PT par groupe.

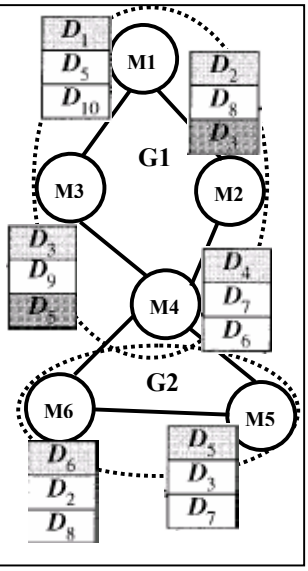


Figure 3.10 : Exemple d'exécution de la méthode E-DCG.

La figure3.10 présente un exemple d'exécution de cette méthode. Après la construction des groupes (G1 et G2), le nœud M1 dans G1 et le nœud M5 dans G2, commencent à exécuter le processus de création des réplicas qui sont crées, en respectant l'ordre décroissant de la valeur de PT de chaque groupe et en choisissant le réplica d'une PT supérieure parmi les nœuds du groupe. Les réplicas D7 et D5 sont créés au deuxième cycle de cette méthode, après avoir terminé l'assignation de tous les réplicas.
Cette méthode permet d'augmenter la disponibilité en partageant les réplicas dans un groupe de nœuds, ce qui met en disponibilité une variété de données. Cependant, elle augmente le trafic qui est du au nombre important de messages échangés. En plus il est trés probable que le réseau change la topologie pendant l'exécution de cette méthode, chose qui va perturber le processus de construction de groupe et l'assignation des réplicas.

3.2.2.2.1.4 Cache coopératif :

Dans[Lig04], les auteurs ont proposé deux méthodes de réplication, ils supposent un environnement ad hoc illustré par la figure suivante :

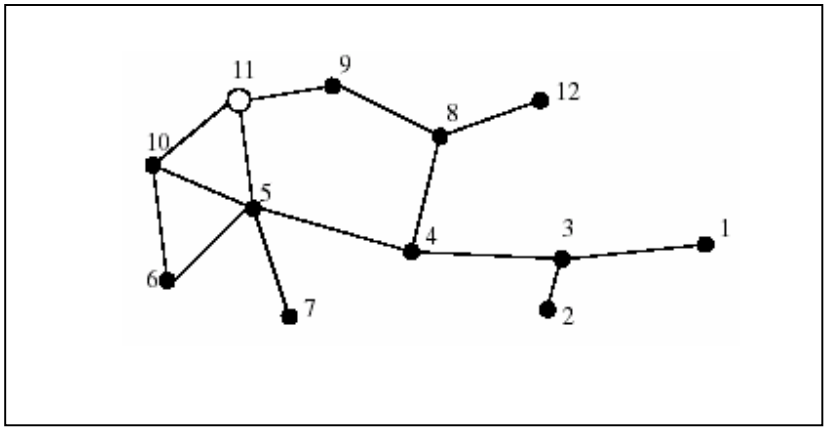


Figure 3.11 : un réseau mobile ad hoc.

Dans cet environnement, le nœud N11 est un serveur fixe qui contient une base de données de N items (d1, d2,……dn), les autres nœuds sont des nœuds mobiles qui ont une capacité de stockage limitée et qui se déplacent suivant le modèle rendom way point. Les données sont mises à jour seulement par le serveur qui sert les demandes suivant la politique FCFS (first-come-first-service). Quand le serveur envoie une donnée élémentaire à un nœud mobile, il envoie aussi sa durée de vie (période de mise à jour). Après l'expiration de la période de mise à jour, le nœud mobile doit obtenir la nouvelle version des données.

3.2.2.2.1.4.1 La méthode CacheData

En participant au routage de données aux autres nœuds, un nœud mobile peut cacher une donnée di quand il trouve que di est populaire, c.-à-d, il y avait beaucoup de demandes de di ou il a suffisamment d'espace libre. Par exemple, dans la figure3.11, les nœuds N6 et N7 envoient leur demande d'accès à di à travers le nœud N5, ce dernier trouve que cette donnée est populaire et crée une copie de cette donnée localement, les prochaines demandes peuvent être servies directement par le nœud N5. Si le nœud N1 envoie plusieurs demande d'accès à di, les nœuds N3, N4, N5 vont trouver que di est populaire et vont créer une copie de cette données localement. Pour réduire la consommation d'espace mémoire, un nœud mobile ne doit pas répliquer une donnée si les demandes d'accès à cette donnée viennent du même nœud.

Dans cette méthode, l'information demandée peut être répliquée prés de l'utilisateur, ceci diminue le temps de réponse et réduire le trafic. Néanmoins, le traitement exigé par cette méthode pour chaque demande d'accès peut prendre un temps d'exécution élevé notamment quand le nombre de nœuds et de demandes d'accès augmentent.

3.2.2.2.1.4.2 La méthode CachePath

Le concept de cette méthode peut être expliqué en utilisant la figure3.11. Supposant que le nœud mobile N1 demande la donnée di, cette demande doit être envoyée au serveur N11. Quand le nœud N1 reçoit cette donnée, le nœud N3 sait que N1 a une copie de di. Plus tard, si N 2 demande di, N 3 sait que le serveur N 11 est de trois sauts, alors que N1 se trouve d'un seul saut. Dans ce cas la demande de N2 est redirigée vers N1 en se basant sur les protocoles de routage qui fournissent le nombre de sauts entre la source et la destination. Pour réduire le trafic de réseau, un nœud cache un Path de donnée si la donnée est très près de lui.

L'utilisation de chemin d'accès permet l'accès rapide à l'information demandée. Néanmoins, les auteurs de cette méthode suppose l'existence de protocole de routage ce qui demande un espace mémoire supplémentaire et produit un trafic élevé.

3.2.2.2.2 Réplication de données avec mise à jour apériodique :

Dans ce cas les données peuvent être modifiées à n'importe quel moment par le nœud qui a la copie originale de donnée.

Dans [Hku04] les auteurs ont proposé une extension des méthodes proposées dans [Har01] et [Har03] pour adapter l'environnement où la mise à jour soit apériodique. L'idée de base était de définir une relation reliant la fréquence de lecture et la fréquence

d'écriture d'une donnée. Cette relation est définie par le taux de lecture / écriture comme suit :

RWR=Rij/Wj, où

Rij : dénote la probabilité qu'une demande d'accès à une donnée Dj soit envoyée par le nœud mobile Mi pendant une unité de temps.

Wj : dénote la probabilité qu'une mise à jour d'une donnée originale Dj soit effectuée par son propriétaire.

L'augmentation de la valeur de RWR signifie, que les évènements de lecture sont beaucoup plus produits que les évènements d'écriture. Donc il vaut mieux répliquer la donnée. Si la valeur de RWR est diminuée, ça implique que les évènements d'écriture sont beaucoup plus produits. Dans ce cas, nous ne pouvons pas répliquer ces données car, il y aura plus de chance que ces données soient modifiées. Dans cet environnement les auteurs supposent que :

- Chaque nœud mobile possède un espace mémoire limité et les réplicas sont crées à chaque période de relocation (*relocation period*).
- Chaque donnée originale est aléatoirement mise à jour par son propriétaire. Après qu'une donnée élémentaire soit mise à jour, les réplicas de cette donnée tenus par des nœuds qui ne peuvent pas accéder à l'originale, deviennent invalides.
- Les fréquences de lecture et d'écriture à chaque donnée pour chaque nœud sont connues et ne sont pas stables.
- Les données émergentes sont toujours répliquées en premier et sans condition. S'il n'y a pas d'espace, la donnée qui a un RWR le plus bas peut être supprimée. Au cas où, une donnée émergente est modifiée et le nœud mobile est déconnecté, le nœud peut décider de lire cette donnée ou attendre la connexion.
- Les données originales sont considérées comme des copies primaires et les réplicas comme des copies secondaires. Les mises à jour sont appliquées seulement sur la copie primaire puis les modifications sont propagées aux autres copies. Pour garder la cohérence entre les copies, les auteurs utilisent un des protocoles de cohérence comme le protocole LASY[Lls92].

En se basant sur ces conditions, les auteurs ont proposé les méthodes suivantes :

3.2.2.2.2.1 La méthode E-SAF+ :

Dans cette méthode, chaque nœud crée un nombre C de réplicas dans l'ordre décroissant de la valeur de RWR. Lors d'une création de réplica, une demande de réplication peut

être échouée. Dans ce cas, l'espace mémoire de ce réplica est remplacé temporairement par un des réplicas qui ont été assignées lors de la période de relocation précédente mais ne sont pas actuellement choisis. Ce réplica est choisi parmi les replicas possibles où sa valeur de RWR est la plus haute. S'il n'y a aucun replica possible à être assigné provisoirement, l'espace mémoire est maintenu librement. L'exécution de cette méthode suit les étapes suivantes :

- A chaque période de relocation, chaque nœud possède des données originales broadcast ses fréquences d'écritures pour ces données. Cette diffusion va permettre aux nœuds mobiles de calculer leur valeur de RWR pour chaque donnée.
- Se basant sur le profil de l'utilisateur, s'il existe des données émergentes Dj qui seront demandées par Mi avant le prochain période de relocation. Alors RWRij = MAXVALUE pour garantir la réplication d'une donnée émergente.
- Chaque nœud crée les réplicas suivant l'ordre décroissant de la valeur de RWR.

L'exécution de cette méthode ne produit pas beaucoup de messages, ce qui réduit le trafic. D'un autre coté la création du réplica est basée seulement sur la valeur de RWR, dans ce cas, les nœuds qui ont les mêmes caractéristiques vont créer les mêmes réplicas. Cependant, un nœud mobile peut accéder aux données tenues par les autres nœuds. Par conséquent cette méthode réduit le taux d'accessibilité quand plusieurs nœuds ont les mêmes caractéristiques d'accès.

3.2.2.2.2.2 La méthode E-DAFN+ :

Cette méthode est exécutée comme la méthode précédente. Sauf qu'elle partage les réplicas entre les nœuds voisins. A chaque période de relocation chaque nœud diffuse son identificateur et ses valeurs de RWR de chaque donnée. Donc chaque nœud peut savoir l'ensemble des nœuds auquel il est connecté. Dans chaque ensemble à partir de nœud d'identificateur inférieur le processus d'élimination des réplicas redondants entre les nœuds voisins est exécuté. En cas de redondance, le nœud qui a une valeur RWR inférieure pour ce réplica redondant remplace le réplica concerné par un autre réplica de RWR supérieure parmi les réplicas qui ne sont pas assignés.

Cette méthode hérite les avantages et les inconvénients de la méthode E-DAFN. Elle augmente la disponibilité par rapport à la méthode précédente. Mais, elle augmente aussi le taux de trafic.

3.2.2.2.2.3 La méthode E-DCG+ :

Cette méthode essaie d'augmenter la disponibilité en partageant les réplicas dans un groupe de nœud biconnected. Après la construction des groupes qui s'effectuée de la même façon que la méthode DCG à chaque période de relocation et l'exécution de première et deuxième étape de la méthode E-SAF+. La méthode E-DCG+ suit les étapes suivantes :

- Dans chaque groupe, une valeur de RWR du groupe à chaque donnée est calculée en additionnant le RWR des nœuds mobiles du groupe pour chaque donnée. Ces calculs sont effectués par le nœud mobile d'identificateur inférieur.
- Dans l'ordre décroissant de la valeur de RWR du groupe, les réplicas sont assignés, en prenant en considération aussi la valeur de RWR de chaque nœud pour chaque donnée.
- Après avoir créé tous les réplicas, s'il reste un espace mémoire libre aux nœuds mobiles dans le groupe, d'autres replicas sont assignées dans l'ordre décroissant de la valeur de RWR jusqu'à ce que l'espace mémoire soit plein. Chaque réplica est assignée à un nœud mobile dont la valeur de RWR à cette donnée est la plus haute parmi les nœuds qui ont l'espace mémoire libre.

L'avantage de cette méthode est le partage de données dans un groupe de nœuds, ce qui va augmenter la disponibilité par la présence d'une variété de données. Cependant, cette méthode prend un temps d'exécution plus élevé et augmente le trafic, en produisant un nombre élevé de messages.

3.4 Etude Comparative :

Dans ce paragraphe nous allons conclure et comparer les méthodes précédentes, en se basant surtout sur les paramètres et les métriques suivants :

3.4.1 Le taux d'accessibilité de donnée : l'accessibilité de donnée est le premier objectif de toutes les méthodes cherchant à augmenter la disponibilité de l'information dans l'environnement ad hoc. Une donnée est dite disponible si elle est atteignable, quand elle est demandée par n'importe quel utilisateur. Le taux d'accessibilité est exprimé par le rapport du nombre de demandes d'accès réussies, au nombre de toutes les demandes d'accès envoyées.

3.4.2 Le trafic : le trafic est un paramètre très important. Il doit être pris en considération par toutes les méthodes et tous les protocoles de réplication. Le trafic est exprimé par le nombre de messages produits durant l'exécution du système. Ce paramètre doit être minimisé autant que possible vu les limitations et les caractéristiques liées à l'environnement mobile ad hoc.

3.4.3 Le taux d'accessibilité invalide : ce paramètre est mesuré quand les mises à jour de données sont prises en considération par l'environnement. Il est exprimé par le rapport du nombre d'accès aux réplicas invalides, au nombre de toutes les demandes d'accès envoyées. Dans l'autre coté le taux d'exactitude de donnée peut être mesuré par le rapport du nombre d'accès aux réplicas valides, au nombre de toutes les demandes d'accès envoyées.

3.4.4 Comparaison des méthodes de réplication de données sans mise à jour :

Le **tableau3.1** présente une comparaison de six méthodes de réplication de données ou la mise à jour n'est pas prise en compte. Les méthodes sont exécutées périodiquement ou chaque nœud peut créer un nombre limité de réplicas. L'objectif de ces méthodes est de fournir un taux d'accessibilité élevé et de réduire le trafic.
D'après les résultats de simulation, la méthodes DCG donne un taux d'accessibilité élevé par rapport à la méthode DAFN, tandisque la méthode SAF donne un taux d'accessibilité bas. Cette différence est justifiée par le degré de redondance de réplica plus élevé dans la méthode SAF. L'évaluation de ces méthodes montre que l'accessibilité est améliorée quand la stabilité de lien est prise en compte, telle que la méthode DCG-S1 donne le niveau le plus élevé de disponibilité.

De l'autre coté, la méthode DCG et la méthode DCG-S1 donnent le trafic le plus élevé. DAFN et DAFN-S1 donnent un trafic moins élevé par rapport aux deux premières méthodes. Le trafic provoqué par ces méthodes est inversement proportionnel à la période de relocalisation, alors que l'accessibilité de données n'est pas sensible à cette période.

Méthodes	Principe de réplication	Taux de disponibilité	Trafic engendré	Temps d'exécution
SAF	Les réplicas sont crées périodiquement suivant l'ordre décroissant de fréquence d'accès de chaque nœud.	Disponibilité faible causée par les nœuds ayant les mêmes caractéristiques.	Un trafic réduit	Un temps d'exécution réduit.
DAFN	- Créer les réplicas de la même façon comme la méthode SAF. - Eliminer les redondances entre les nœuds voisins	Amélioration de taux de la disponibilité.	Un trafic plus au moins élevé.	Un temps d'exécution plus au moins élevé.
DCG	- Construire les groupes biconnected. - Calculer la fréquence d'accès de chaque groupe. - Créer les réplicas suivant l'ordre décroissant de fréquence d'accès du groupe et de chaque nœud à chaque donnée.	Un taux de disponibilité élevé.	Un trafic élevé	Un temps d'exécution élevé.
DAFN-S1	- Créer les réplicas de la même façon comme la méthode SAF. - Partager les réplicas entre les nœuds voisins reliant par des liens stables. - La stabilité de lien entre Mi et Mj est donnée par la relation Bij= tij/période relocalisation.	Disponibilité plus élevé par apport à la méthode DAFN	Trafic moins élevé par apport à la méthode DAFN	Un temps d'exécution moins élevé.
DAFN-S2	Même que DAFN-S1 sauf la stabilité est donnée dans ce cas par la relation $(1\text{-}Bij)\times p$	Même que la méthode DAFN-S1	moins élevé par apport à la méthodeDAFN	Un temps d'exécution moins élevé.
DCG-S1	Même que DCG plus la prise en compte des liens stable à l'intérieur du groupe.	plus élevé par apport à la méthode DCG	moins élevé par apport à la méthode DCG	Un temps d'exécution élevé.

Tableau 3.1

3.4.5 Comparaison des méthodes de réplication de données avec mise à jour périodique :

La mise à jour de données pose le problème de cohérence entre les réplicas du même objet. Dans ce cas, les méthodes de réplication de données visent à réduire au minimum le taux d'accessibilité invalide, augmenter la disponibilité et réduire le trafic. Le **tableau3.2** présente une comparaison des méthodes E-SAF, E-DAFN et E-DCG. On prend en considération les métriques décrites précédemment.

Méthodes	Principe de réplication	Taux de disponibilité	Trafic engendré	taux d'accessibilité invalide	Temps d'exécution
E-SAF	Périodiquement, chaque nœud crée les réplicas suivant la fréquence d'accès et le temps restant pour qu'une donnée soit mise à jour.	Disponibilité basse	Trafic bas	dépend de la période de mise à jour	Moins élevé.
E-DAFN	- Création des réplicas comme la méthode E-SAF. - Partage les réplicas entre les nœuds voisins.	Disponibilité élevé par rapport à la méthode E-SAF	Plus élevé par rapport à la méthode E-SAF	dépend de la période de mise à jour	Plus élevé par rapport à la méthode E-SAF
E-DCG	- Création des groupes biconnected. - Pour chaque groupe les réplicas sont crées suivant la fréquence d'accès du groupe et le temps restant pour que cette donnée soit mise à jour.	Disponibilité plus haute que celle de la méthode E-DAFN	Plus élevé par rapport à la méthode E-DAFN	dépend de la période de mise à jour	Plus élevé par rapport à la méthode E-DAFN
CacheData	Réplication de données populaires.	Disponibilité moyenne dépend de la stabilité du réseau adhoc	Élevé	Dépend de la connectivité du réseau	élevé
CachePath	Creation de chemins de données.	Disponibilité plus au moins un temps de réponse réduis	Plus élevé causé par le protocole d routage.	Dépend de la mobilité du réseau.	Plus élevé

Tableau 3.2.

L'étude d'évaluation des ces méthodes montre que la disponibilité augmente et le taux d'accessibilité invalide diminue suivant l'augmentation de la période de mise à jour. C'est parce que, quand la période de mise à jour est longue, les replicas créées sont valides pendant longtemps. Dans ces trois méthodes, la méthode E-DCG donne un temps d'accessibilité plus élevé par rapport aux deux autres méthodes. Cette étude montre aussi, que les performances de ces méthodes sont dégradées quand la période de mise à jour est plus courte que la période de relocalisation.

La méthode E-DCG produit un trafic plus élevé par rapport aux autres méthodes, le trafic est inversement proportionnel à la période de mise à jour. Parce que chaque nœud mobile doit fréquemment régénérer les replicas qu'il possède après que les originaux étaient mis à jour.

3.4.6 Comparaison des méthodes de réplication de données avec mise à jour apériodique :

Dans ce cas les données sont mises à jour irrégulièrement. La mise à jour est effectuée sur la copie originale par le nœud qui la tient, les autres nœuds peuvent l'accéder seulement en lecture

Méthodes	Principe de réplication	Taux de disponibilité	Trafic engendré	taux d'accessibilité invalide	Temps d'exécution
E-SAF+	Périodiquement, chaque nœud crée les réplicas suivant la valeur de RWR.	Disponibilité basse	Trafic bas	Dépend de la fréquence d'écriture.	Moins élevé.
E-DAFN+	- Création des réplicas comme la méthode E-SAF+. - Partage les réplicas entre les nœuds voisins.	Disponibilité élevée par rapport à la méthode E-SAF+	Plus élevé par rapport à la méthode E-SAF+	Dépend de la fréquence d'écriture.	Plus élevé par rapport à la méthode E-SAF
E-DCG+	- Création des groupes biconnected. - Pour chaque groupe les réplicas sont crées suivant la valeur de RWR du groupe et la valeur de RWR de chaque nœud.	Disponibilité plus haute que celle de la méthode E-DAFN+	Plus élevé par rapport à la méthode E-DAFN+	Dépend de la fréquence d'écriture.	Plus élevé par rapport à la méthode E-DAFN

Tableau 3.3

Comme **le tableau 3.3** présente, la méthode E-DCG+ donne un niveau de disponibilité plus élevé par rapport aux deux autres méthodes. Le résultat de simulation de ces méthodes prouve que cette disponibilité diminue en parallèle avec l'augmentation de fréquence d'écriture. Parce que, quand la fréquence d'écriture est basse, les replicas créés sont valides pendant longtemps. Quand la fréquence d'écriture est haute, les replicas deviennent inaccessibles dans un temps très court, et ainsi, dans la plupart des

cas les demandes d'accès réussissent seulement quand les nœuds envoient la demande aux nœuds qui tiennent les originaux des données.
Le taux d'accessibilité invalide et le trafic dans ces méthodes sont proportionnels à la fréquence d'écriture. Quand la fréquence d'écriture est plus élevée, le trafic et le taux d'accessibilité invalide ont leur valeur maximale. Parmi ces méthodes, E-DCG+ donne le trafic le plus élevé et prend un temps d'exécution le plus long.

3.5 Conclusion :

La réplication des données est une technique très efficace pour élever la disponibilité, réduire le temps d'exécution et améliorer la qualité de service, néanmoins ses avantages ne peuvent se concrétiser réellement dans un environnement ad hoc qu'avec une méthode fiable et durable qui répond aux questions suivantes :
1. Quel est le nombre optimal de replicas pour chaque donnée ?
2. Quel est le nœud le plus approprié pour tenir le replica d'une donnée?
Cette méthode doit aussi améliorer les performances en tenant compte des caractéristiques du réseau ad hoc, garantir un taux d'accessibilité de données élevé, un trafic diminué et une gestion efficace de la mémoire de stockage.
Les solutions proposées par Hara ont des avantages et des inconvénients. La méthode DCG-S1 donne de meilleures performances par rapport aux autres méthodes de réplication sans mise à jour. Dans le prochain chapitre, nous allons essayer d'améliorer ces solutions en répondant aux questions précédentes.

Chapitre 4: Réplication de données pour améliorer la disponibilité dans les réseaux mobile ad hoc.

4.1 Introduction :

Les déconnexions fréquentes liées à l'environnement mobile ad hoc, les ressources limitées de ces outils mobiles et la nature de déplacement des usages font la complexité du réseau mobile ad hoc.
L'amélioration de la disponibilité de données et la minimisation de la consommation des ressources sont l'objectif de toute méthode visant à garder l'information accessible n'importe où et n'importe quand pour tous les usagers du réseau ad hoc.
L'utilisation de la réplication de données peut aider à atteindre ces objectifs. Elle permet de produire plusieurs copies d'une même source de données et de maintenir l'information accessible en cas de panne de serveur. Plusieurs stratégies de réplication ont été proposées pour les environnements fixes [Cal88], [Fuc94], [Har00], [Krb92], [Woj92] où les échecs ne se produisent pas fréquemment et où les ressources matérielles sont beaucoup plus importantes. Ces stratégies améliorent l'accessibilité et réduisent la charge autour du ou des serveurs. D'autres solution de réplication sont proposées pour l'environnement mobile [Bai94], [Hsw94], [Pib95], [Bou02]. Ces dernières supposent toujours l'existence d'un serveur fixe. Cette hypothèse est contraignante pour l'autonomie d'un réseau mobile Ad Hoc.
Dans ce chapitre nous allons essayer de proposer une solution pour améliorer la disponibilité en prenant compte des limitations liées à l'environnement ad hoc.

4.2 Description de l'environnement et hypothèses:

L'environnement considéré est un réseau mobile ad hoc, où chaque nœud mobile peut coopérer à la construction d'un cache commun, par le partage de son espace de stockage avec les autres nœuds. Un utilisateur mobile peut créer des replicas et les maintenir localement. Comme il peut générer de nouvelles données (données originales) et les partager avec les autres utilisateurs. En plus il peut mettre en cache des chemins ou des « Paths » qui permettent un accès plus rapide aux données distantes. Dans cet environnement un nœud mobile peut recevoir aussi des données à partir d'un réseau statique. La mise à jour des données n'est pas prise en charge par notre environnement, on peut trouver plusieurs applications où la mise à jour des données ne représente pas un

intérêt majeur. Par exemple les informations concernent une région, peuvent être délivrées sous forme d'une carte géographique indiquant les différents chemins et les établissements comme les hôpitaux, l'aéroport, les restaurants, les stations d'essence, les hôtels,…etc. ces informations sont délivrées par des stations fixes d'information. D'autres applications ne nécessitent pas la mise à jour comme les informations de la météo.

Dans cet environnement on suppose que :

- Chaque nœud mobile est désigné par un identificateur unique N_i tel que 1<=i=< NT, où NT est le nombre total des nœuds mobiles formants le réseau ad hoc.
- Chaque donnée créée par un nœud mobile est associée à un identificateur unique D_{ij} où i est l'identificateur du nœud qui crée cette donnée et j est le numéro séquentiel de création de cette donnée par cet utilisateur. Par exemple D_{11} et D_{12} sont respectivement la première et la deuxième donnée originale créée par le nœud N_1. Les données réceptionnées à partir d'un réseau statique gardent leur identificateur original et ne doivent pas être codifiées de cette façon.
- Un nœud mobile peut mettre en cache un « Path » pour accéder aux données qui ne sont pas localement disponibles. La structure de cette technique sera définie plus tard dans ce chapitre.
- Chaque nœud mobile possède un espace mémoire limité, pour maintenir localement les replicas, les données originales et les paths.
- Un nœud mobile envoie périodiquement ses informations concernant les données localement stockées aux nœuds voisins.

Pour chaque utilisateur, chaque donnée accédée est caractérisée par deux types de fréquence d'accès :

- Une fréquence d'accès externe qui représente le taux d'accès de l'utilisateur à une donnée externe c-à-d en dehors de son cache et,
- Une fréquence d'accès interne qui représente le taux d'accès des différents utilisateurs à une donnée interne c-à-d qui est locale au nœud.

4.3 Replication data methods for improving availability in mobile ad hoc network

En se basant sur l'environnement et les hypothèses décrites précédemment, nous proposons dans cette section trois méthodes de réplication de données pour améliorer la disponibilité de données.

4.3.1 Vue d'ensemble :

Il est clair que la disponibilité et l'accessibilité de données dans un réseau ad hoc est un problème difficile à résoudre à cause des caractéristiques liées à cet environnement, notamment en ce qui concerne la mobilité imprévisible des nœuds construisant le réseau, et leur capacité limitée de stockage et de traitement.

T.Hara[Har01] a proposé quelques méthodes pour assigner des replicas aux nœuds mobiles dans un réseau ad hoc afin d'élever le degré d'accessibilité aux données partagées. Les méthodes sont fondées sur l'hypothèse, que tous les nœuds du réseau connaissent les probabilités d'accès aux données et ces probabilités ne changent pas. En plus, les données ne sont pas mises à jour. Dans ces méthodes les auteurs essayent de repartir les replicas entre les nœuds. Dans la méthode SAF plusieurs copies de même données élémentaires peuvent se retrouver sur des nœuds voisins. Alors qu'il y a d'autres données qui ne sont pas assignées du tout faute d'espace. La deuxième méthode DAFN essaye de diminuer la redondance des copies d'une même donnée sur des nœuds voisins. Dans cette méthode les nœuds voisins échangent entre eux leurs fréquences d'accès aux données. L'algorithme de duplication considère alors cette vue d'ensemble des fréquences des voisins pour éviter les redondances. La troisième proposition DCG gère le réseau en groupes de nœuds ou "clusters". Elle duplique alors les données au sein d'un groupe en tenant compte des fréquences d'accès de tous les nœuds du groupe et limite la redondance à l'intérieur du groupe. Dans cette méthode chaque nœud doit diffuser ces caractéristiques d'accès sur le réseau. Malgré toutes ces améliorations le problème de replicas redondants entre les nœuds voisins reste toujours posé. Dans [Har03], T.Hara étend les trois méthodes proposées dans [Har01] en supposant que les données élémentaires sont périodiquement mises à jour. Ces méthodes répliquent les données élémentaires sur les nœuds mobiles en se basant sur les fréquences d'accès aux données, le temps restant jusqu'à ce que chaque donnée soit mise à jour, et la topologie actuelle de réseau.

Dans [Har04] T.Hara et S.Kumar ont proposé une extension de ces trois méthodes en supposant que les mises à jours s'effectuent de manière aléatoire. T.Hara et les autres ont proposé une nouvelle méthode (DCG-S1) [Tys03], dérivée de celle proposée dans [Har01]. Dans cette méthode, les auteurs utilisaient la notion de stabilité des liens, pour construire les groupes stables. Cette technique a permis d'élever l'accessibilité de données. Mais le trafic produit par cette méthode reste toujours élevé.

Yan et Coa [Lig04] ont proposé deux méthodes de réplication pour améliorer l'accès aux données. Les données sont maintenues par des serveurs fixes et les nœuds mobiles peuvent créer des réplicas à partir de ces serveurs. La première méthode CacheData permet aux nœuds intermédiaires(entre le demandeur d'accès et le serveur) de créer des

réplicas pour servir les futures demandes. La deuxième méthode CachePath permet aux nœuds mobiles de cacher un chemin vers des données et l'utilise pour réorienter les futures demandes.

Dans ce qui suit nous proposons une répartition des replicas entre les nœuds dans un réseau ad hoc en répondant aux questions suivantes :

1. Quel est le nombre optimal de replicas pour chaque données ?
2. Quel est le nœud le plus approprié pour tenir le replica d'une donnée?

Dans notre solution nous allons essayer de diminuer tant que possible le nombre du replica déployé et de garder le maximum possible les données prés de l'utilisateur. Nos méthodes s'exécutent en deux étapes :

La première étape est l'étape de génération de replicas d'un objet partagé après sa création (la création d'un objet consiste, pour un nœud à créer ou à recevoir à partir d'un réseau statique, une nouvelle donnée qui n'existe pas sur le réseau).

La deuxième étape est la gestion dynamique de ces replicas. Dans cette partie nous allons essayer de répliquer les données partagées d'une manière dynamique en prenant compte du changement de la topologie et de la fréquence d'accès à l'information.

Nous ne considérons, dans ces travaux que les données sans mise à jour c'est-à-dire accédées uniquement en lecture. Une deuxième proposition est envisageable pour les données mises à jour. La propagation des mises à jour et le maintien de la cohérence des copies font l'objet d'une autre question critique de recherche et d'autres travaux importants.

4.3.2 Hop-based replication data method :

Comme nous avons décrit précédemment cette méthode se base sur deux étapes, et le nombre de saut entre les réplicas est pris en considération. Dans ce qui suit nous allons décrire en détail le principe de ces deux étapes.

4.3.2.1 Génération de replicas primaires:

Cette étape est exécute dés qu'on a une création d'un nouvel objet original partagé ou après la réception d'une nouvelle donnée à partir d'un réseau statique. Elle consiste à générer les replicas primaires pour cette donnée en prenant en compte les nœuds existants et la topologie actuelle.

Le principe de cette étape consiste à générer les replicas en utilisant un compteur de saut (Hop counter) (voir figure1). Ce compteur est initialisé à zéro par le nœud qui crée l'objet original D_o et incrémenté de un à chaque saut, la réinitialisation à zéro de ce compteur se

fait après chaque création d'un replica. Le processus de création de ces replicas dits primaires s'exécute comme suit :

- Quand un nœuds crée un nouvel objet ou reçoit une nouvelle donnée à partir d'un réseau statique. Il doit envoyer un message de type création, contenant l'objet et le compteur de saut initialisé à zéro aux nœuds voisins.
- Chaque nœud qui reçoit ce type de message, incrémente le compteur à un, et crée localement un réplica de cet objet dans les cas suivants :
  - Un nœud est choisi, pour tenir un replica, si le compteur est égal à trois sauts et si ce replica n'existe ni dans le cache de ce nœud ni dans l'un de ses voisins, comme par exemple le nœud N9 de la figure 4.1. Après la création d'un replica, le nœud réinitialise le compteur à zéro et envoi le message de création aux nœuds voisins.
  - Si le compteur est égal à trois sauts, et en même temps il existe une copie de la même donnée dans l'un de ses voisins alors, dans ce cas le nœud récepteur initialise le compteur de saut à un et envoie le message de création aux autres nœuds.
  - Si le compteur est égal à deux sauts, et en même temps il n'existe pas un lien suivant (autre que celui par lequel le message est reçu) alors le nœud actuel doit être choisi, pour tenir le replica comme c'est le cas pour le nœud N8.
- Si un nœud reçoit une autre fois le même message pour le même objet, il doit l'ignorer.

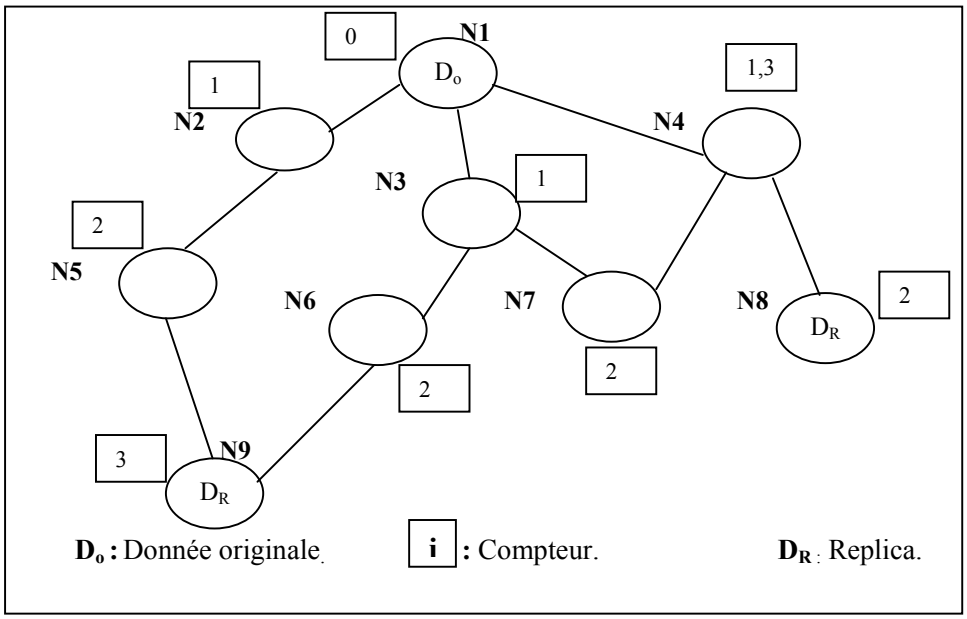


Figure 4.1 : Exemple de génération de replicas primaires.

Ces replicas doivent être stockés sur les nœuds qui sont choisis par notre algorithme. En cas de manque d'espace, les données qui ont une basse fréquence d'accès interne seront supprimées pour libérer de l'espace. Après la création de ces replicas primaires, ils sont considérés comme tous les autres replicas existants dans le cache et leur gestion dynamique est expliquée dans la deuxième partie de cette méthode.

Le but de cette répartition est d'offrir à chaque nœud une plus grande probabilité de présence d'une donnée accédée dans son cache ou dans son voisinage le plus proche. Si l'information recherchée n'existe pas localement, une probabilité plus forte de la trouver dans un nœud voisin est garantie. Ce nœud voisin peut servir les demandes au lieu d'un nœud éloigné. Ceci réduit le temps de réponse des requêtes d'accès, aussi bien que la bande passante et la consommation d'énergie, parce que les nœuds traitent un nombre réduit de requête. En plus, la charge de travail est répartie sur le réseau d'une manière uniforme, ce qui réduit aussi la surcharge au tour d'un nœud mobile. L'espace est utilisé de manière optimale en évitant la redondance non justifiée des replicas d'un même objet. La redondance est mesurée par le nombre de replica d'un même objet sur les nœuds voisins. Toutes fois, deux nœuds voisins peuvent cacher des replicas identiques dans le cas ou l'objet correspondant est très demandé par les deux nœuds (voir le paragraphe suivant).

4.3.2.2 Gestion dynamique de replicas :

Dans cette partie, nous voulons placer dynamiquement les replicas et les organiser dont le but de réduire au minimum le nombre de copies déployées, tout en maintenant la disponibilité. Les nœuds dans un réseau mobile ad hoc, peuvent se joindre ou quitter le réseau à n'importe quel moment. La nature de déplacement des nœuds cause le changement dynamique de la topologie, ce qui agit négativement sur la disponibilité d'un côté et exige d'un autre côté, une solution solide qui s'adapte à ces caractéristiques et garantit toujours la disponibilité.
Dans l'immédiat, on traite seulement le cas de nouveaux utilisateurs qui viennent de se joindre au réseau. Alors la question qui doit être posée est : comment peut-on repartir les replicas qui sont déjà assignés (par exemple D0 dans la figure4.1). La réponse est l'utilisation de la réplication à la demande tout en respectant tant que possible les conditions de saut utilisées dans la partie précédente. Supposant qu'on a deux utilisateurs

qui viennent de se relier au réseau. Ces deux utilisateurs sont représentés dans la figure 4.2 par les nœuds N10 et N11.

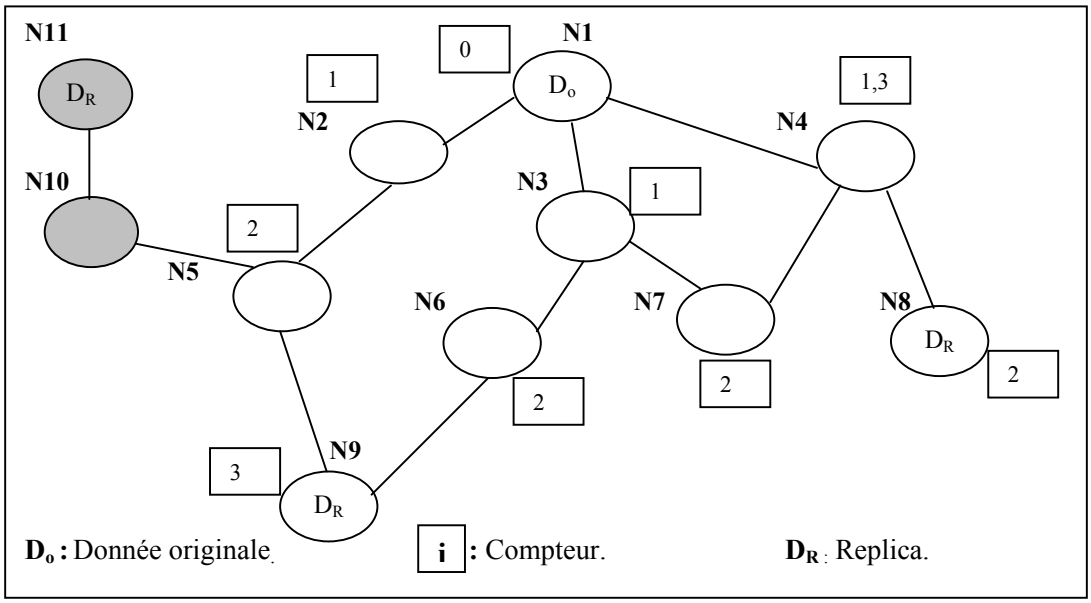


Figure 4.2 : Exemple de réplication à la demande.

Si le nœud N11 veut lire la donnée D0, alors il doit envoyer sa demande Hop par Hop vers N5 qui va l'envoyer à son tour vers N9. Cette demande est accompagnée d'un compteur de sauts. Lorsque N9 reçoit la demande, il trouve le compteur de sauts de cette demande égale à trois. Dans ce cas, au lieu d'envoyer la réponse, l'utilisateur N9 envoie une copie complète de la donnée à N11.

Cette même demande peut envoyer par N10 avant la demande de N11. Dans ce cas, N9 trouve le compteur de saut égal à deux, ce qui signifie que N10 est un nœud de frontière (comme le nœud N8). Alors N9 va envoyer une copie de la donnée vers N10. À la réception de cette copie, N10 se comporte comme l'indique la figure4.3:

- Si le nœud N10 n'a qu'un seul nœud voisin (N11) sans compter le nœud N5 (par le quel est reçu la copie), alors cette copie doit être envoyée à ce voisin (voir figure4.3.A). Car nous souhaitons autant que possible garder trois sauts entre deux copies identiques.
- Si le nœud N10 est un nœud frontière (voir figure 4.3.B) ou a plusieurs nœuds voisins (voir figure 4.3.C). Alors N10 cache ce replica localement. Le principe du compteur n'est pas respecté à ce niveau dans le but d'offrir le meilleur temps d'accès possible à ces voisins.

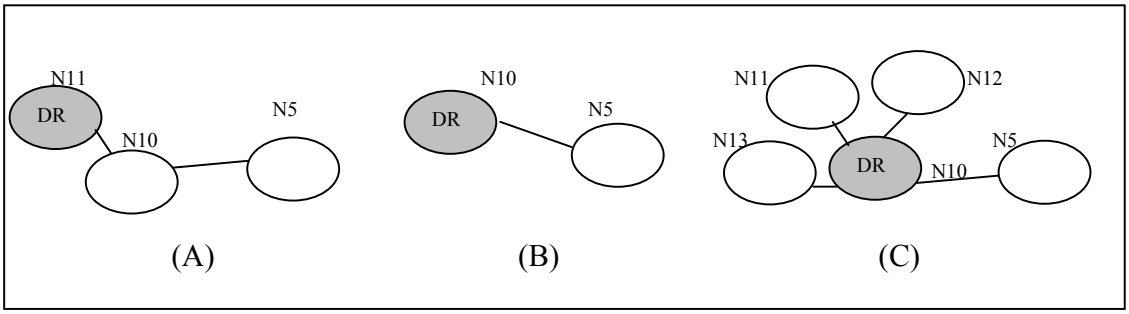


Figure 4.3 : Le comportement d'un nœud après la demande d'une donnée.

Avec cette répartition des replicas, on peut se trouver dans une situation où un utilisateur utilise une donnée d'une manière répétitive, alors que celle ci n'existe pas dans son cache. Pour réduire au maximum le nombre de transmission de messages et la consommation de la bande passante, on définit une variable **T** qui représente le taux d'accès à une donnée par un nœud telle que :

T= NB/U ou

NB : est le nombre de demandes d'accès à une donnée et,

U : est une unité du temps.

Pour une meilleure gestion du cache on distingue pour chaque utilisateur deux types de taux**,** un taux **TE** qui représente le taux d'accès d'un utilisateur à une donnée externe c-à-d en dehors de son cache et un taux **TI** qui représente le taux d'accès des différents utilisateurs à une donnée interne au nœud.

Ces variables sont définies pour chaque donnée élémentaire. Si un utilisateur détecte que sa variable **TE** liée à une donnée élémentaire dépasse un certain **seuil**(**TE >S**) c-à-d cette donnée est très demandée par cet utilisateur, il vaut mieux la répliquer dans son cache. Pour se faire on distingue les cas suivants :

- S'il y a suffisamment d'espace localement au nœud, la réplication est effectuée directement.
- S'il n'y a pas un suffisamment d'espace libre, alors les données qui ont le plus petit **TI** seront supprimées pour libérer l'espace. Ici la suppression ne perd pas l'objet, car il existe au moins une copie de cet objet dans le réseau c'est la copie originale.

Avec cette répartition, l'utilisateur a besoin de diffuser sa demande aux nœuds voisins, bien sûr si l'information recherchée n'existe pas dans son cache. Cette diffusion peut charger ou plus peut saturer le réseau et prendre un temps d'exécution plus élevé notamment dans un réseau ad hoc à grande échelle où le nombre de nœud est élevé. L'utilisation de la technique de *CachePath* proposée dans [Lig04], peut contribuer à réduire la consommation de l'énergie et à optimiser l'utilisation de la largeur de la bande passante à un certain degré. Cette technique consiste à cacher le path le plus proche pour atteindre l'objet(original ou replica), dans ce cas l'utilisateur peut satisfaire ses accès en un nombre réduit de sauts. Dans notre cas le *Path* est une relation triple qui relie l'identificateur de l'utilisateur, l'identificateur de l'objet et le nombre de saut qui sépare la donnée du nœud demandeur (*(Transmitter_ NodeAdd, Add_Data, Hop_num)*). Cette technique s'adapte à notre proposition de la façon suivante :

1- Lors de la réception du message de création des réplicas primaires, si un nœud trouve le compteur de saut égale un, il va comprendre que l'émetteur de ce message possède une copié (originale ou réplica) de ce nouvel objet. Alors il doit créer localement un path vers cet objet. Le path dans ce cas est de la forme *(Transmitter_ NodeAdd, Add_Data, 1)*, voir les nœuds N2, N3, N4 et N10 de la figure4.4 Les prochaines demandes d'accès à D0, qui seront réceptionnées par ces nœuds seront orientées, selon le Path qui convient le plus.

2- Un nœud lors de la réception du message de création de réplicas primaires. Il doit créer un path de la forme *(Target_NodeAdd, Add_Data, 1)*, s'il trouve le compteur de saut égale deux, comme les nœuds N5, N6et N7 de la figure4.4.

3- En contribuant au cheminement des réponses d'accès pour les autres nœuds, un nœud mobile peut créer des paths, suivant la valeur du compteur de saut. Comme l'exemple de nœud N11 de la figure 4.4, lors de l'envoi de la réponse au nœud N12. Il crée le path *(N12, D0, 1)* pour servir les futures demandes d'accès à cette donnée.

4- Un nœud ne doit pas cacher un Path vers un objet, s'il possède un replica de ce même objet.

5- Le Path est caché directement, s'il y a un espace suffisant dans le cache. Sinon les Paths moins référencés seront supprimés pour libérer l'espace. Un Path est toujours de taille très petite.

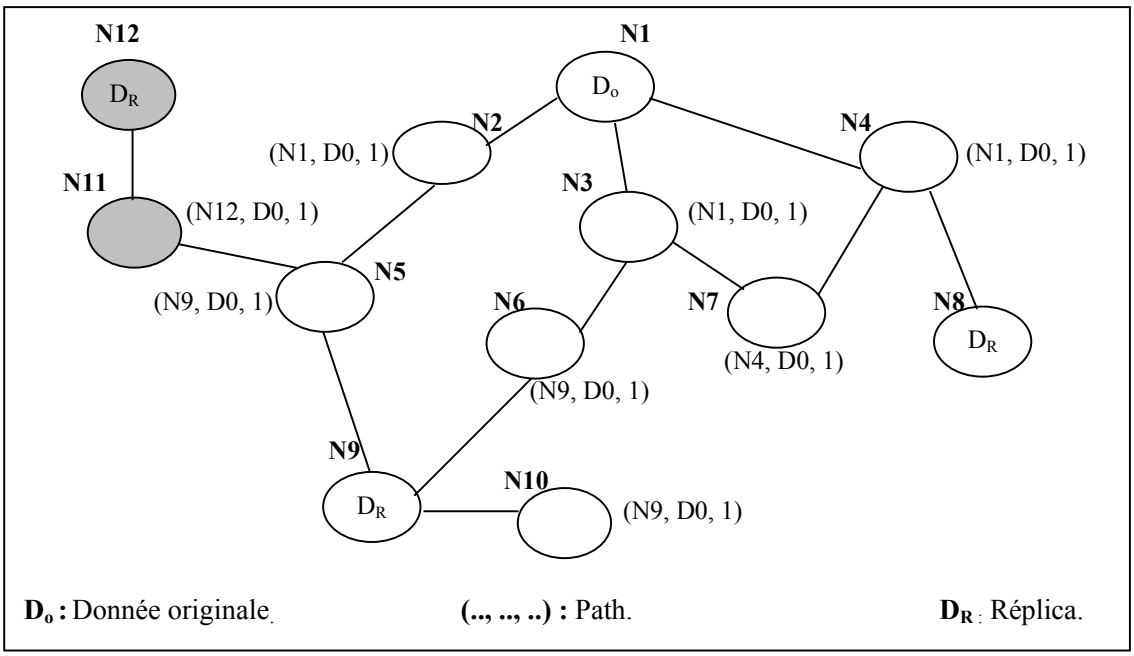


Figure 4.4 : Exemple de cache un path d'accès pour D0

A cause du déplacement aléatoire et instantané de l'utilisateur dans l'environnement ad hoc, les liens entre les utilisateurs peuvent être cassés. En plus le processus de gestion du cache peut supprimer les données, et le *Path* caché pour ces données devient alors invalide. Afin de garder seulement les Paths valides. On utilise la même technique que celle utilisée dans [Lig04]. Elle consiste à associer un Time-To-Live (TTL) à chaque Path. Le Path est considéré invalide si son TTL est expiré et dans ce cas le Path doit être supprimé. Cette durée de vie est mise à jour à chaque échange faisant référence à la donnée

4.3.2.3 Accès aux données :

Dans ce contexte, une demande de l'utilisateur pour accéder à une donnée se traite de la façon suivante :

1- Si le demandeur possède l'original ou un replica de la donnée en question. L'accès est effectué localement.

2- Si le demandeur n'a pas la donnée recherchée(originale/replica) et possède un path valide vers cette donnée, la demande doit être envoyée vers cette donnée en spécifiant le path.
3- Si le demandeur ne possède ni la donnée recherchée, ni un path valide qui mène vers cette donnée, la demande dans ce cas doit être envoyée aux nœuds voisins.

4- Le récepteur de la demande doit se comporter comme un demandeur lors de la réception d'une demande d'accès à une donnée.

5- Quand le récepteur reçoit une demande d'accès, dont la donnée demandée et le demandeur constituent un path valide au niveau du récepteur. Il va comprendre automatiquement l'invalidité de ce path et la demande doit être traitée comme le troisième cas.

4.2.2.4 Algorithme :

Pour générer les replicas de cette façon, on a besoin de deux procédures:

- Une procédure Création() qui est exécutée après chaque création d'un nouvel objet par le nœud correspond. Le résultat de cette procédure est l'envoi d'un message contenant le compteur de saut et l'objet à répliquer. Il permet aux nœuds qui vérifient les conditions de saut, de créer localement une copie de cet objet et d'informer les autres nœuds qu'il y a un nouvel objet partagé.
- La deuxième procédure Replica() est exécutée par tous les autres nœuds suite à la réception du message initial de réplication envoyé par le nœud qui crée l'objet.

Procédure Création()

```
{
  int CPT =0 ;
  Créer Objet ;

  /* message de création envoyé aux nœuds voisins */
  Envoyer_Creation (AddCreatNode,CPT, Objet) ;

}
```

Procédure Replica(AddCreatNode, CPT, Objet**)**

```
{   Si le message Envoyer_Creation (AddNode, CPT, Objet) déjà réceptionné
    Alors rien faire ;
    Sinon
      { CPT++ ;
        Si CPT=1 Alors {  Créer un Path de type (AddCreatNode, Objet, CPT);
                    Envoyer_Creation (AddCreatNode, CPT, Objet) aux nœuds voisins ;
                          }
       Si CPT=2 Alors  si le nœud est un nœud frontière Alors
                                {
                                   Créer un replica pour cette objet ;
                                }
                    sinon
                    {
                     Créer un Path de type (AddNextNode, Objet, CPT);
                     Envoyer_Creation (AddCreatNode, CPT, Objet) aux nœuds voisins ;
                    }
        Si CPT=3 Alors si voisinage(AddNeighborNode, Objet)=false ;
                          /*cette fonction vérifie l'existence de replica de l'objet
                          réceptionné aux nœuds voisins    */
                  Alors
                   {
                    Créer un replica pour cet objet ;
                    CPT=0 ;
                    Envoyer_Creation (AddCreatNode, CPT, Objet) aux nœuds voisins ;
                   }
                Sinon   {
                           Créer un Path (AddNeighborNode, Objet, 1) ;
                          CPT=1;
                    Envoyer_Creation (AddCreatNode, CPT, Objet) aux nœuds voisins ;
                     }
}
```

4.3.3 Importance-Based Replication Data Method :

Dans un réseau mobile ad hoc à grande échelle où le nombre des nœuds est plus ou moins élevé, chaque nœud peut générer plusieurs données originales. Alors, la génération des replicas primaires pour chaque objet en utilisant la méthode précédente, peut augmenter le trafic et surcharger les nœuds mobiles, notamment dans le cas où, plusieurs nœuds génèrent des objets originaux en même temps. En plus avec cette génération des replicas inconditionnée, on peut avoir répliqué des objets qui ne seront pas très utilisés par les autres nœuds. Par ce que, la génération des réplicas primaires pour une donnée qui ne

représente pas un intérêt aux autres nœuds, augmente le trafic et consomme l'espace mémoire des nœuds mobile et ne fournit aucun gain aux utilisateurs.
On propose alors une amélioration à notre méthode en rajoutant le traitement du cas particulier des données qui ne présentent pas nécessairement un intérêt pour les nœuds du réseau. Nous utilisons la règle suivante :

« *Un nœud dans un réseau mobile ad hoc lance la procédure de génération du replica primaires pour un nouvel objet qui vient de le créer, s'il connaît à l'avance que cet objet sera utilisé par les nœuds du réseau. Sinon, il envoi seulement un message de contrôle pour informer les autres nœuds de l'existence d'un nouvel objet partagé* ».

Pour motiver cette idée on donne l'exemple suivant : Dans un champ de bataille. Si un soldat détecte une réalité, alors il ne doit pas attendre qu'elle soit très demandée par les autres utilisateurs pour qu'il la réplique. Mais, puisqu'il connaît à l'avance que cette réalité qui vient de la récolter sera s'intéressée par plus d'un utilisateur, il exécute directement la procédure de création, L'information ou la donnée nouvelle est alors répliquées selon la méthode présentée dans les paragraphe précédents. Les messages de réplication servent à dupliquer et à informer de la présence de la donnée. En suite, n'importe quel soldat peut accéder à cette information et la retrouver rapidement.
Ce même scénario peut se retrouver dans un travail collectif, oũ un nombre important de chercheurs formants un réseau ad hoc, coopèrent à la réalisation d'un même projet. En faisant son travail, un chercheur peut aboutir à un résultat, dont les autres ont besoin pour continuer leurs travaux. Malgré que ce nouvel objet(résultat) ne soit pas encore demandé, mais il doit être répliqué de la façon précédente, puisqu'il sera l'objet d'intérêt pour les autres utilisateurs.
Il est évident que ces solutions générales au problème devront subir des adaptations spécifiques lorsqu'on les adoptera pour une application donnée. Comme, par exemple, le traitement du cas de réception d'une nouvelle donnée qui peut sensiblement différer d'une solution à une autre.

4.3.4 Size-Based Replication Data Method:

L'amélioration précédente peut prendre un temps d'exécution élevé, une consommation de bande passante et d'énergie importante, dans le cas où la taille de l'objet est très importante.

Pour remédier à ce problème, on rajoute ici le traitement de ce cas particulier. Dans la première partie, lors de la génération des replicas primaires, si la taille de l'objet est importante (dépasse un certain paramètre seuil), il vaut mieux d'envoyer un message de contrôle pour informer les autres nœuds de l'existence de cette nouvelle donnée. Ainsi, on limite la circulation de messages de taille importante sur le réseau. Une procédure spécifique est alors prévue pour ce cas.

En considérant aussi la notion de taille dans la gestion dynamique de replicas, on sait que deux messages de tailles sensiblement différentes n'ont pas le même temps de transmission, même si les deux messages ont la même source. Si deux données ont la même fréquence d'accès externe, on donne la priorité de réplication à la donnée qui a un temps d'accès élevé. Pour se faire, on définie la relation suivante :

R= ΣT_i (i=1….n)/FE telle que

Ti : est le temps de réponse et i est le nombre d'accès, n représente le nombre totale d'accès dans une unité du temps.

FE : est la fréquence d'accès externe.

Dans cette méthode, si un nœud mobile trouve que de deux données ou plus ayant la même fréquence d'accès externe dépassée un certain seul. La donnée qui sera répliquée en premier est celle qui a une valeur **R** supérieure. De cette façon on assure en cas de l'existence des données qui ont la même fréquence externe, que la priorité de réplication est réservée aux données éloignées ou qui ont une réponse de taille élevée.

4.4 Conclusion :

Dans ce chapitre, nous avons proposé trois méthodes de réplications, pour améliorer l'accessibilité et la disponibilité de données dans un réseau mobile ad hoc. En utilisant la technique du cache.

Ces méthodes s'exécutent en deux parties. La partie de génération de replicas primaires. Dans cette partie la première méthode génère les réplicas primaires de chaque nouvel objet. Or que la génération de ces réplicas dans la deuxième et la troisième méthode est basée respectivement sur l'importance et la taille de l'information.

Dans ces méthodes, nous avons pris en considération la priorité d'accès à chaque données pour chaque utilisateur, en utilisant la fréquence d'accès.

L'étape suivante sera réservé pour l'évaluation de ces méthodes, en utilisant un outil de simulation.

Chapitre 5 : Evaluation des performances

5.1 Introduction :

L'analyse analytique des protocoles des réseaux mobiles est rendue difficile par la complexité des protocoles d'accès aux canaux, l'influence de la mobilité non prédictive des nœuds et les caractéristiques des ondes radio et des communications sans fil. Pour se faire, on a souvent recours à des mesures réelles ou à la simulation.

L'objectif de simulation est de modéliser le monde réel et de mettre en évidence les caractéristiques et les interactions entre les activités du système modélisé. Elle permet aussi d'observer le comportement du système et de suivre son évolution dans le temps.

Dans ce chapitre nous allons évaluer nos méthodes en les comparant avec la méthode DCG-S1[Tys03] présentée précédemment, en utilisant le simulateur GloMoSim (*Global Mobile Information System Simulator*) [Bta00].

5.2 Simulateur et outils de développement

Différents logiciels de simulation de réseaux sont disponibles. Ils permettent de simuler toutes les couches réseau de manière déterministe(Un modèle de simulation est déterministe s'il ne contient aucune variable aléatoire). Il est possible de sauvegarder la totalité des événements qui se sont déroulés durant la simulation. Cette sauvegarde peut être utilisée pour détecter des défauts lors de la conception de nouveaux protocoles ou d'effectuer des statistiques sur les performances de ceux-ci.

Les deux simulateurs les plus populaires dans le domaine de recherche de gestion de réseau sans fil sont NS2 et GloMoSim. NS2 est un outil logiciel de simulation des réseaux informatiques. Il est devenu Aujourd'hui un standard de référence en ce domaine. C'est un logiciel libre disponible sur internet. Il supporte, depuis quelques années, la simulation de réseaux wireless.

GloMoSim est simulateur spécialisé aux environnements wireless. Il a été conçu afin de simuler les réseaux sans fil à grande échelle, grâce à l'approche d'agrégation de nœuds. Nous nous sommes concentrés sur le second simulateur vu la diversité des fonctionnalités qu'il offre pour la gestion de la mobilité.

5.2.1 GloMoSim

GloMoSim est un simulateur réseau modulaire pour la simulation et l'évaluation des performances des réseaux sans fil (locaux ou étendus) [Bzg98]. Il utilise la simulation parallèle à événements discrets fournie par le langage PARSEC (*Parallel Simulation Environment for Complex System*) développé par PCL (*Parallel Computing Laboratory*) de l'université UCLA (*University of California at Los Angles*). Parsec permet de bien séparer la description du modèle de la simulation de sa nature d'exécution (parallèle ou séquentielle). Pour réaliser une simulation, GloMoSim utilise une approche simple. En effet, si nous

supposons que chaque nœud dans la simulation est une entité a part, alors nous allons remarquer non seulement une dégradation mais aussi une limitation des performances de la simulation car l'initialisation d'un millier d'entités peut influer sur le temps d'exécution et sur l'utilisation de l'espace mémoire. Pour palier à ce problème, GloMoSim fait recours à une approche d'agrégation de nœuds où une seule entité peut simuler plusieurs nœuds du réseau dans le système [Bta00]. Chaque nœud a sa propre structure de données et son code qu'il doit s'exécuter sans interférence ni violation d'accès aux structures de données des autres nœuds. Chaque nœud à son propre état et l'ensemble des états complets de tous les nœuds est maintenu par l'entité. Cette dernière approche permet d'augmenter le nombre de nœud dans une simulation sans augmenter le nombre d'entité. Une entité n'est autre qu'une zone géographique où chaque nœud est déterminé par ses coordonnées dans cette zone.

5.2.1.1 Architecture

GloMosim a en plus, une structure de couches proche de celle du modèle OSI [Dar99]. En effet, chaque entité intègre les diverses couches réseaux qui peuvent être simulées sous forme de fonction. Au début de la simulation une fonction d'initialisation est appelée pour chaque couche de chaque nœuds. A la réception d'un message, une couche exécute la tache demandée. A la fin de la simulation, une nouvelle fonction est lancée pour mettre fin à la simulation et collecter les statistiques désirées. Pour faciliter la communication entre les diverses couches réseaux, un ensemble d'API (*Application Programming Interface*) est spécifié sous forme de messages échangés entre les couches.

Couche Application (CBR, HTTP, Telnet, FTP)
Couche Transport (TCP, UDP)
Couche Réseau (IP avec AODV, Flooding, Bellman-Ford, OSPF, DSR, WRP)
Couche Mac (CSMA, MACA, MACAW, FAMA, 802.11)
Couche Radio (Free Space, Rayleigh, Ricean, SIRCIM)
Couche Mobilité (Random drunken et Random waypoint)

Figure 5.1 : les couche de Glomosim.

5.2.1.2 Configuration de la Simulation

GloMosim prend en entrée un fichier décrivant la description de la simulation. Ce fichier (en format *.in*) contient un ensemble de variables de paramètres de la simulation avec les valeurs correspondantes. Parmi ces variables on trouve :

- ***Simulation-Time*** : temps de simulation en jour, heure, seconde, microseconde et nano-seconde. Si on ne précise pas l'unité, la seconde est utilisée par défaut.

- ***Terrain-Range-X*** : Largeur du terrain de la simulation en mètre ;
- ***Terrain-Range-Y*** : Longueur du terrain de la simulation en mètre ;
- ***Number-Of-Nodes*** : nombre des nœuds réseaux dans la simulation ;
- ***Node-Placement*** et ***Node-Placement-File*** : décrivent les emplacements des nœud dans le terrain : (aléatoire, uniforme, rangés, en groupe, etc.) ;
- ***Mobility*** : spécifie si les nœuds seront mobiles ou pas. Si une mobilité est activée, on spécifie alors la nature de mobilité (aléatoire ou pas), la liste suivante présente les différents types de mobilité :

```
- MOBILITY TRACE
- RANDOM-DRUNKEN
- MOBILITY-TRACE-FILE ./mobility.in
- MOBILITY PATHLOSS-MATRIX
- MOBILITY RANDOM-WAYPOINT
    MOBILITY-WP-PAUSE      30S
    MOBILITY-WP-MIN-SPEED  0
    MOBILITY-WP-MAX-SPEED  7
- MOBILITY  NONE
```

La mobilité None implique que la vitesse de déplacement des nœuds est égale à zéro. Dans ce type de mobilité les nœuds ne subissent aucun mouvement.
Pour le model RANDOM-DRUNKEN, si un nœud est actuellement à la position (x, y), il peut probablement se déplacer à (x-1, y), (x+1, y), (x, y-1), et (x, y+1) dans le terrain physique. Pour RANDOM WAYPOI NT, un nœud choisit aléatoirement une destination dans le terrain physique. Il se déplace vers cette destination avec une vitesse uniformément choisie entre MOBILITY-wp-min-speed et MOBILITY-wp-max-speed (meter/sec). Après qu'il atteigne sa destination, le nœud reste là pour une période de temps précisée par le paramètre MOBILITY-wp-pause.
- ***Power-Range :*** représente les limites de la portée de communication de chaque nœud.
- ***Bandwith* :** représente la bande passante utilisée par les nœuds pour transmettre des messages.
- ***Application*** : application à simuler (ftp, cbr, http ou telnet) ;
- ***Routing-Protocol*** : protocole de routage utilisé ;
- ***Transport-Protocol*** : protocole de transport utilisé (UDP ou TCP) ;
- ***MAC-Protocol*** : protocole de la couche Mac (802.11, CSMA, FAMA (*Floor Acquisition Multiple Acces*), MACA(*Multiple Access Collision Avoidance*);
- et les paramètres de statistiques et d'autres paramètres optionnels.

5.3 Paramètres de notre environnement de simulation :

Cette section présente la configuration que nous avons choisie pour notre environnement de simulation.

5.3.1 le choix de modèle de mobilité :

Parmi les models de mobilité offerts par le simulateur Glomosim et qui peuvent représenter la mobilité des nœuds mobiles dans un réseau ad hoc, on trouve le modèle *Random Waypoint* et *Random Druken.* Dans le premier modèle un nœud choisit aléatoirement une destination dans le terrain de la simulation et il se déplace vers cet endroit en utilisant une vitesse choisie arbitrairement entre une vitesse maximale et une vitesse minimale spécifiées auparavant par l'utilisateur. Lorsqu'il atteint sa destination, un nœud fait une pause pendant un temps aussi défini. Dans le cas de *Random Druken* un nœud se déplace périodiquement à une position choisie arbitrairement à partir de celles de ces voisins immédiats tout en suivant une vitesse donnée. En effet si un nœud se trouve à une position (x,y), il peut se déplacer aux positions suivantes (x-1, y), (x+1, y), (x, y-1) ou (x, y+1). Parmi ces deux modèles, nous avons choisi le modèle *Random Waypoint,* car il est plus proche du mouvement réel des nœuds par rapport aux autres modèles.

5.3.2 Couche application :

Dans cette couche chaque nœud peut générer des données originales, comme il peut générer des demandes d'accès aux autres données. La création des données et la génération des demandes d'accès suivent un processus de poisson avec une moyenne M qui représente l'intervalle du temps en seconde entre deux création ou deux demandes d'accès (M=60s).

La création des données originales est définie comme une application dans le fichier de configuration de Glomosim comme suit :

CREAT : <Source> <Taille de l'item> <Temps de début>.

Après la création d'une donnée le processus de création de réplica est exécuté. Les demandes d'accès sont aussi définies dans le fichier de configuration en utilisant la syntaxe suivante :

DEMAND : <Source> <Idf_Data> <NumSeq_Data> <Temps de début>.

5.3.3 Couche MAC :

Le protocole qu'on a utilisé dans cette couche, c'est le protocole IEEE 802.11 qui est très utilisé dans les réseaux sans fil. Dans ce protocole chaque paquet transmis doit suivi par un acquittement de réception, L'absence de cet acquittement après plusieurs transmissions indique la défaillance de lien entre le nœud émetteur et le nœud récepteur. Dans ce protocole une vérification du canal doit précéder son utilisation ce qui permet d'éviter les collusions.

5.3.4 Modèle de propagation :

Parmi les modèles de propagation qu'on a trouvé disponible au niveau de Glomosim, on a choisi le modèle Free space qui est un modèle simple et de propagation libre. Dans ce modèle le signal se propage de l'émetteur vers le récepteur en négligeant tous les obstacles entre eux, et le signal faiblit en fonction de la distance entre le nœud source et le nœud destination.

5.3.5 Surface du terrain :

On a choisi un terrain carré de 1000 m * 1000 m

5.3.6 Autres paramètres utilisées

Intervalle de génération de données: 60 seconde.
Intervalle de demande de données : 60 seconde.
Temps de simulation : 10 minutes.
Bande passante : 2 Mbits/s.
Pause time : 10 seconde.
Nombre de nœud : 50.
Nombre de donnée :50.

5.4 Les paramètres de comparaison :

5.4.1 La fréquence d'accès

La fréquence d'accès représente le nombre d'accès dans une unité du temps. Dans nos méthodes, un nœud mobile crée un réplica pour une donnée si sa fréquence d'accès déplace un certain seuil. Le calcule de fréquence de cette façon ne reflète pas le besoin de l'information dans le temps. Car on peut se retrouver dans le cas ou un nœuds demande une donnée d'une manière répétitive mais sa fréquence d'accès dans une unité du temps ne dépasse pas le seuil exigé, alors cette donnée ne sera jamais répliquée pendant qu'elle est demandée à chaque unité du temps. Pour cela nous avons opté pour une autre formule connue sous le nom de « Moving Averages » [Bur76], [Cro87a], [Cro87b], [Kum85], [Hun86], [Mon96] qui est une des formule d'analyse technique les plus connues. La moyenne mobile est un des indicateurs les plus anciens et les plus utilisés. Une moyenne mobile permet de calculer la valeur moyenne de la valeur sur une période donnée. Le terme de "mobile" indique qu'il s'agit d'un calcul glissant.

Il existe deux types de moyenne mobile, une moyenne mobile simple, qui est une méthode ad hoc permettant de donner une estimation des "alentours" d'une série, on suppose alors que la variable sera proche de sa moyenne récente. Une moyenne mobile simple, tout simplement est une moyenne sur une fenêtre glissante d'observations :

$$Y_t(m) = 1/m \sum_{I=1}^{m} y_{t+k-i},$$

Où k est librement fixé selon les besoins du modélisateur, pour une prévision, il est nécessaire que k <= 0. Par exemple, si les données sont de fréquence mensuelle, et qu'on souhaite prévoir Y_t, il est possible d'utiliser la fonction de moyenne mobile suivante :

$$f(t) = 1/12 \sum_{I=1}^{12} y_{t-i},$$

qui fournit la prévision : Y(t+1)= f (t) qui est la moyenne des 12 dernières observations.

Cette moyenne considère toutes les observations au même pied d'égalité. Il est très réaliste, que la prochaine valeur Y(t+1) puisse être proche d'une simple moyenne des dernières observations. Si on souhaite accorder plus de poids aux observations les plus récentes, on peut utiliser le modèle EWMA (Exponentially Weighted Moving Average) selon lequel :

$$Y(t+1) = \beta \sum_{k=0}^{\infty} (1 - \beta)^k y_{t-k}. \quad \text{tel que}$$

$\beta \in]0, 1[$ est le facteur de poids (smoothing constant). Pour ne pas être obligé de recalculer la prévision chaque fois qu'une nouvelle valeur de x est révélée, on utilise la formule de mise `a jour suivante : $Y(t+1) = \beta Yt + (1 - \beta)Y't$

Les moyennes mobiles peuvent être calculées sur différentes périodes, ce qui permet de dégager des tendances à court terme. La figure (Figuer5.2) suivante montre l'influence du facteur de poids (smoothing constant) sur l'utilisation de la moyenne. Dans cette figure la moyenne mobile est calculée sur une période d'une journée durant 20jours.

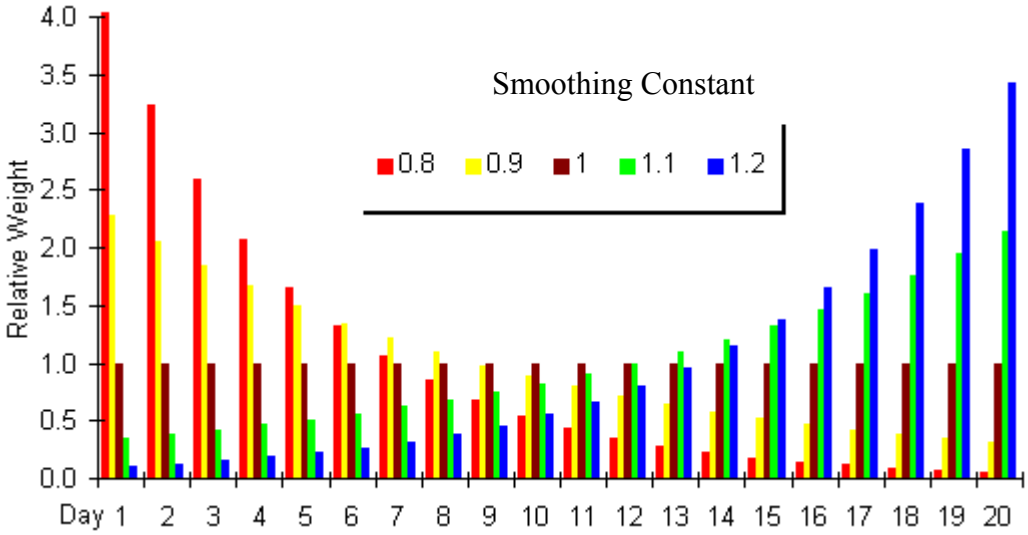


Figure 5.2 : smoothing constant et la moyenne mobile.

Si le smoothing constant est proche de 1, la prévision comporte une certaine rigidité et une grande dépendance au passé. Si au contraire le smoothing constant est petit, la prévision est plus souple et ne dépend que faiblement du passé.

Dans notre cas la moyenne mobile de la fréquence d'accès qu'elle soit soit interne ou externe est donnée par la formule suivante :

$$MAf_{ij} = \beta\, MAf_{ij*} + (1- \beta)\, f_{ij} \quad \text{tels que :}$$

MAf_{ij} : la moyenne mobile de la fréquence d'accès du nœud i à la donnée j pour la nouvelle période.

MAf_{ij*} : la moyenne mobile de la fréquence d'accès du nœud i à la donnée j pour l'ancienne période.

f_{ij} : la fréquence d'accès du nœud i à la donnée j pour la nouvelle période.

B : est le constant de poids, dans notre simulation $\beta = 0.5$. avec cette valeur on donne le poids à la nouvelle valeur de fréquence en même temps, on prend en considération l'historique d'accès de chaque nœud.

La MAf_{ij} est calculée à chaque unité du temps. Si la valeur de la MAf_{ij} externe dépasse un certain seuil K. Alors la donnée concernée doit être répliquée.

5.4.2 La vitesse de déplacement de nœud

Le modèle de mobilité que nous avons choisi pour notre configuration, est le modèle Random WayPoint qui est le modèle le plus proche au mouvement réel des nœuds mobiles. Dans ce modèle la vitesse de déplacement des nœuds est comprise entre une valeur minimale précisée par la variable MOBILITY-WP-MIN-SPEED et une valeur maximale donnée par la variable MOBILITY-WP-MAX-SPEED. Pour tester les métriques citées au paragraphe 5 nous avons choisi de modifier ce paramètre selon les intervalles suivantes :

- Vitesse très faible : dans ce cas la vitesse de déplacement des nœuds varie entre 0 et 1.
- Vitesse faible : la vitesse de déplacement varie entre 0 et 3.
- Vitesse moyenne : la vitesse de déplacement varie entre 0 et 5.
- Vitesse forte : la vitesse de déplacement varie entre 0 et 10.

5.4.3 La charge du réseau

Ce paramètre a une relation directe avec la consommation de la bande passante et l'augmentation de trafic. Une consommation importante de la bande passante peut dégrader les performances du réseau et augmente le risque de collusion, ce qui agit négativement sur la disponibilité de donnée. Donc il est très important de tester l'influence de ce paramètre sur le fonctionnement et le déroulement de ces quatre méthodes.

Dans notre cas ce paramètre est manipulé par la variation du nombre de demande d'accès envoyée et le nombre de création de donnée par seconde pour chaque nœud, ces processus suivent une loi de poisson avec une moyenne M. Dans ce paramètre nous allons aussi varier la taille de donnée entre 1 KO jusqu'à 4 KO.

5.4.5 Capacité de stockage

La taille de stockage des nœuds est un paramètre important pour augmenter la disponibilité et réduire le trafic. Nous allons varier ce paramètre (entre 1KO et 50KO) pour observer ses effets sur les métriques choisis ci-après. La taille du cache par défaut est 20KO.

5.4.6 La portée de communication

La portée de communication est délimitée par le rayon de puissance de communication des nœuds. Ce paramètre est accessible en utilisant la variable POWER-RANGE du fichier de configuration. La valeur par défaut de ce paramètre est 50M et l'intervalle de variation est [1m, 300m].

5.4.7 Scalability

Le but de ce paramètre consiste à suivre le comportement d'un protocole de réplication de données, quand le nombre de nœuds participant augmente. Le protocole de réplication doit toujours donner un taux d'accessibilité acceptable et réduire le nombre de message transmis. Dans ce paramètre nous allons varier le nombre de nœud entre 50 et 250 nœuds, dans un terrain de 5000 M * 5000 M.

5.5 Les métriques à mesurer

Basant sur les paramètres mentionnés précédemment. Nous allons mesuré les métriques essentiels pour évaluer les performances de nos méthodes proposées en les comparant avec la méthode DCG-S1[Tys03]. ces métriques sont :

5.5.1 Le taux de disponibilité

Cette métrique permet de connaître le taux d'accessibilité de données durant le temps de simulation. Formellement ce taux est défini par la formule suivante :

TD= NDR / TDE

Tels que NDR et TDE sont respectivement le nombre de demandes réussies et le total de demandes envoyées.

L'objectif de tous les protocoles de réplication est d'augmenter le TD le plus possible.

5.5.2 Le trafic

Le trafic est le nombre de message transmis par tous les nœuds pendant la durée de simulation, formellement le trafic est défini comme suit :

$T = \sum_{i=1}^{n} NmesTr_i$ tel que n et le nombre de nœud et $NmesTr_i$ est le nombre de message transmis par le nœud i durant la simulation.

5.6 Les résultats de simulation

Dans cette section nous allons simuler et évaluer les trois méthodes que nous avons proposé plus la méthode DCG-S1. Dans ces résultats, chaque point représente la moyenne de trois exécutions. Ces expériences sont déroulées durant dix minutes.

5.6.1 Fréquence d'accès et l'accessibilité

Dans les méthodes que nous avons proposé. Un nœud mobile doit répliquer une donnée si sa fréquence d'accès à cette dernière dépasse un certain seuil, la fréquence d'accès est calculée en utilisant la fonction de la moyenne mobile décrite précédemment. Le but de cette expérience est de voir l'influence du changement de seuil sur le taux d'accessibilité aux données pour les trois méthodes proposées. La figure 5.3 montre le résultat de l'expérience.

Dans cette figure, l'axe horizontal représente les valeurs de fréquence d'accès choisies comme seuil et l'axe vertical représente le taux d'accessibilité de données.

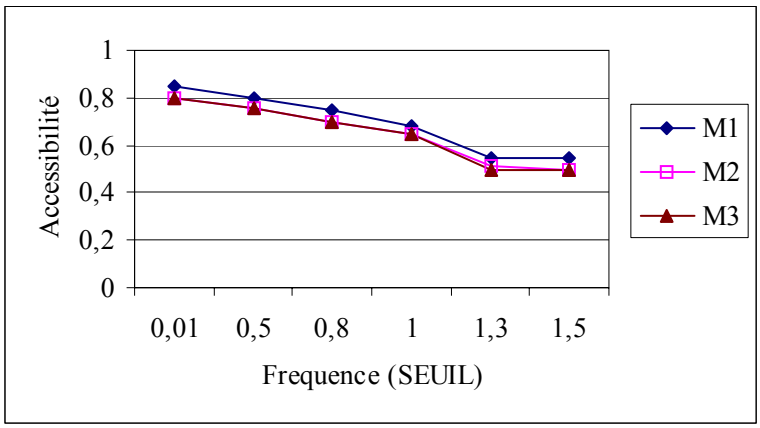


Figure5.3 : Fréquence d'accès et l'accessibilité.

Ce résultat montre que quand le seuil augmente, le taux d'accessibilité de données diminue puis il reste stable après une certaine valeur de seuil. Quand le seuil est très petit, toutes les fréquences d'accès vont dépasser ce seuil et les données concernées sont répliquées localement, ce qui explique le niveau élevé d'accessibilité. Quand le seuil augmente le niveau d'accessibilité diminue, par ce que le seuil choisi reste toujours supérieur à la plupart des fréquences d'accès des données, ces dernières ne peuvent pas être répliquées localement, ce qui oblige l'utilisateur à transmettre ses demandes à l'extérieur. Ces demandes peuvent ne pas être satisfaites.

La première méthode donne un taux d'accessibilité plus ou moins élevé par rapport aux deux autres méthodes, ceci est dû grâce à la procédure de création des réplicas primaires qui permet d'approcher toutes données créées aux utilisateurs du réseau. Cette accessibilité sera diminuée quand la charge augmente ou la capacité de stockage diminue, ce qu'on va voir après.

5.6.2 Fréquence d'accès et le trafic

La figure 5.4 montre l'influence du changement de seuil de fréquence d'accès sur le trafic. Dans cette figure nous remarquons que le trafic augmente d'une manière non considérable avec l'accroissement du seuil.

Quand la valeur du seuil est petite, presque toutes les données demandées sont répliquées ce qui réduit le nombre de message transmis et réduit le trafic. Quand le seuil augmente la plus part des données ne sont pas répliquées et presque toutes les demandes d'accès sont transmises aux autres nœuds, et donc, le résultat est un trafic plus important.

Mais la différence n'est pas importante car le trafic de réplication, produit dans le premier cas, est compensé par le trafic d'accès dans le deuxième cas. Le trafic d'accès dépend de

l'importance du nombre des accès distants aux données. Le trafic induit par la réplication dépend de la valeur du seuil fixé.

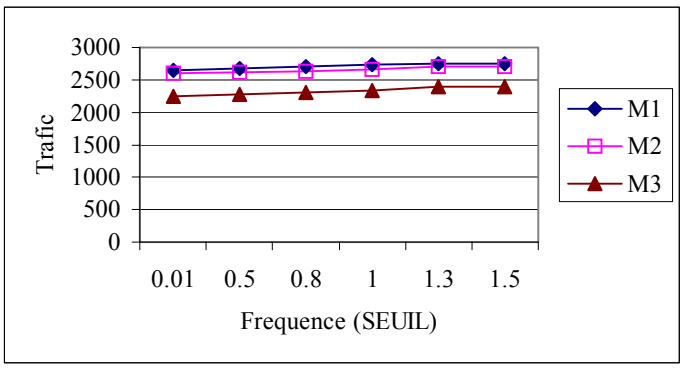


Figure 5.4 : Fréquence d'accès et le trafic.

La troisième méthode donne le meilleur trafic. Cette différence est due à la procédure de création des réplicas primaires. Cette procédure n'est exécutée dans cette méthode que pour une nouvelle donnée créée qui sera très demandée et lorsque sa taille ne dépasse pas un certain seuil. En plus dans cette méthode la priorité de réplication est donnée aux données éloignées. Toutes ces conditions induisent l'écart du trafic engendré par l'application des différentes solutions proposées.

5.6.3 Capacité de stockage et l'accessibilité

Dans la figure 5.5, nous remarquons que le taux d'accessibilité pour toutes les méthodes augmente avec l'augmentation de capacité de mémoire de stockage, ceci est du à la possibilité de répliquer localement les données demandées.
Pour la méthode DCG-S1, les réplicas sont alloués périodiquement, après la construction des groupes de nœuds, suivant l'ordre décroissant des fréquences d'accès du groupe. Si la capacité de stockage est large, le nœud mobile peut créer plusieurs réplicas de différentes données ce qui conduit à élever le taux d'accessibilité.
Dans le cas de nos méthodes, nous observons que si la mémoire de stockage est large la première méthode M1 donne le meilleur taux d'accessibilité. Ceci est dû à la procédure de création de réplicas primaires qui permet à chaque nouvelle donnée de se répliquer de sorte qu'elle soit proche de l'utilisateur. Lorsque la capacité de mémoire est petite, la méthode M3 donne un niveau d'accessibilité plus ou moins élevé par rapport aux autres méthodes. Ceci est dû à la procédure de création de réplicas primaires qui s'exécute, dans ce cas seulement, pour les données très demandées et des tailles qui ne dépassent pas un certain seuil, par ce que la création des réplicas primaires pour les données qui ne sont pas très demandées ou qui ont une taille importante nécessite une capacité de stockage plus grande et nécessite la suppression des données localement stockées qui peuvent être des données accédées.

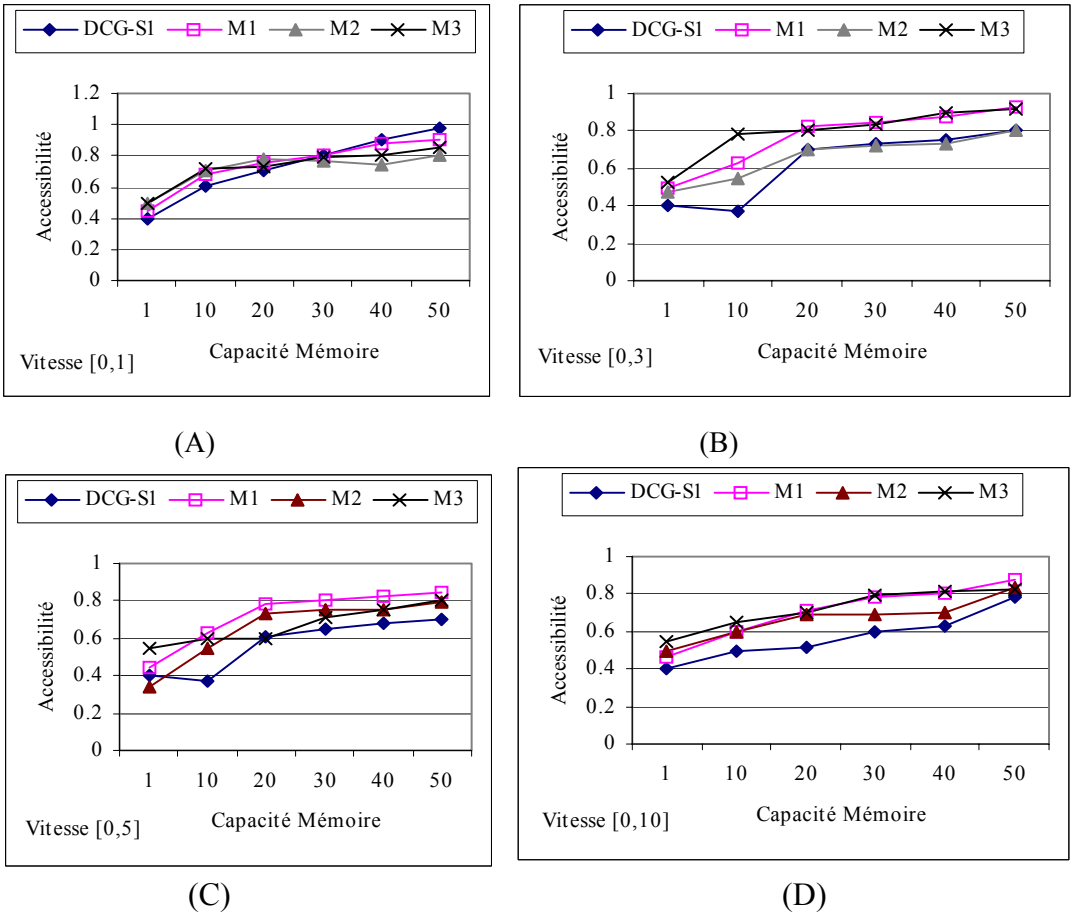


Figure 5.5 : Capacité de stockage et l'accessibilité.

Pour voir clairement l'influence de la variation de la vitesse de déplacement des nœuds sur l'accessibilité de données, nous avons regroupé les quatre états de la figure5.5, dans un seul état présenté par la figure5.6, où chaque point dans cette figure est la moyenne de tous les points représentés par chaque état de la figure5.5. par exemple pour calculer l'accessibilité quand la mobilité est égale à un pour la méthode M1, il suffit de calculer la moyenne de tous les points représentés par l'état de la figure5.5(A) pour cette méthode.

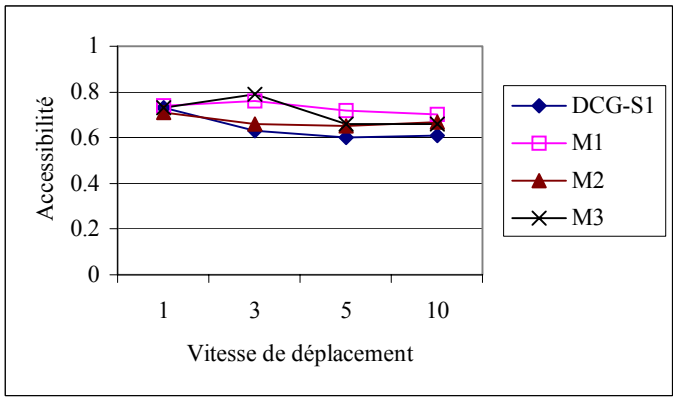


Figure5.6 : Vitesse de déplacement et l'accessibilité.

Dans cette figure nous observons que la méthode DCG-S1 est plus ou moins influencée par l'augmentation de la vitesse de déplacement des nœuds par rapport à nos méthodes. L'augmentation de la vitesse peut augmenter le changement de la topologie du réseau, ceci diminue la possibilité de trouver des liens stables entre les nœuds, ce qui rend la construction des groupes pour la méthode DCG-S1 presque une mission impossible. Dans ce cas chaque nœud construit un groupe et réplique les données suivant ses fréquences d'accès à ces données. Les nœuds mobiles de mêmes caractéristiques vont répliquer les mêmes données, ce qui va réduire le taux d'accessibilité notamment dans le cas de mémoire de stockage très petite. Pour nos méthodes nous remarquons qu'elles ne sont pas très influencées par l'augmentation de vitesse de déplacement des nœuds et la méthode M1 donne toujours le taux d'accessibilité le plus élevé en cas de capacité mémoire élevée même quand la vitesse de déplacement augmente par ce que M1 ne limite pas la création des réplicas primaires pour les données partagées par leurs importance ou leur taille.

5.6.4 Capacité de stockage et le trafic

La figure 5.7 montre que la méthode DCG-S1 donne le trafic le plus élevé. Presque la totalité de ce trafic est généré par l'exécution de la procédure de construction de groupe. Cette procédure produit un nombre très élevé de message et prend beaucoup temps d'exécution.

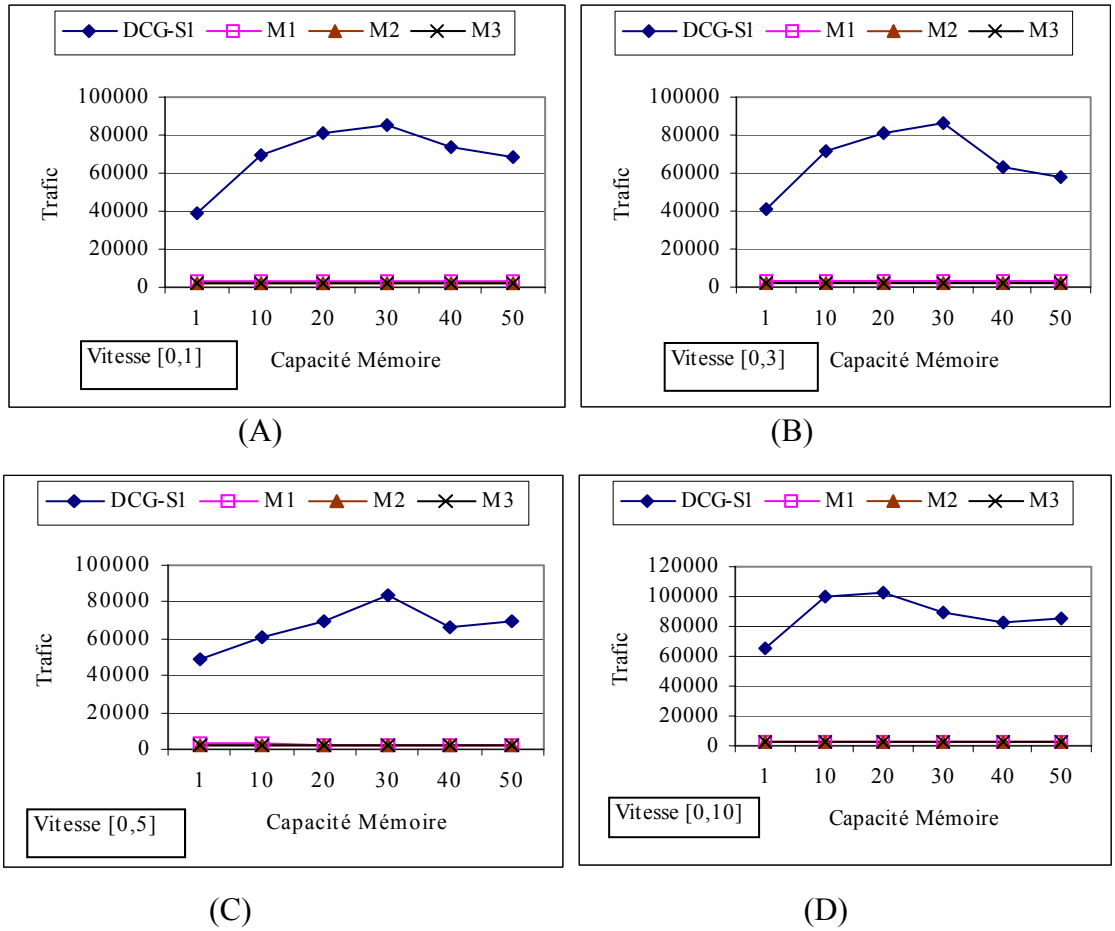
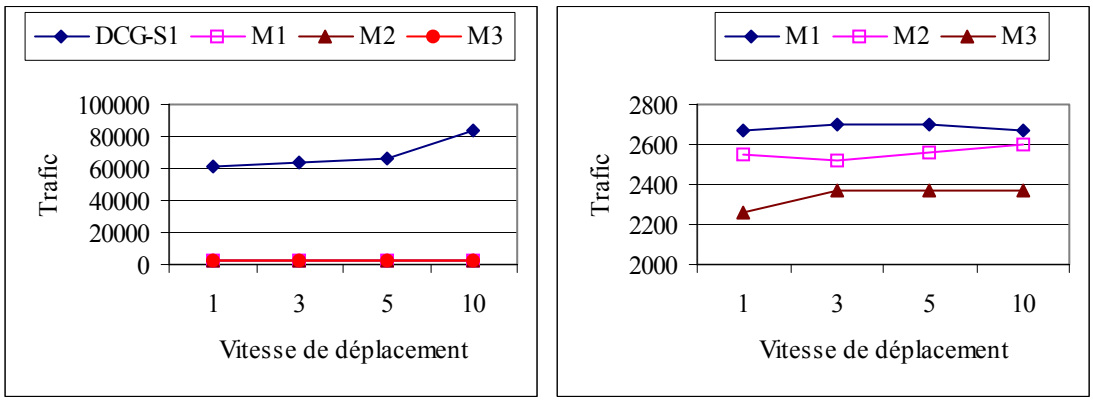


Figure 5.7 : Capacité de stockage et trafic.

D'autre part, le trafic augmente quand la capacité de la mémoire devient plus grande, puis il diminue d'un certain point. Quand la capacité de la mémoire est petite, le trafic est petit parce que le nombre de réplicas relocalisés est petit. Quand la capacité de la mémoire est grande, le trafic est également petit parce que tous les nœuds mobiles tiennent les réplicas de la plupart des données(plus d'accès locaux) et la relocalisation de réplicas se produit ainsi rarement.
Le trafic pour nos méthodes reste relativement petit par rapport à la méthode DCG-S1. Le trafic généré dans nos méthodes est produis par la procédure de génération de réplicas primaires et par les demandes d'accès si la donnée concernée n'est pas disponible localement.
Pour observer lisiblement l'influence de changement de vitesse sur le trafic généré par les quatre méthodes, nous avons regroupé les quatre états présentés de la figure 5.7, dans la figure 5.8.
Nous observons dans la figure 5.8 (A) que le trafic produit par la méthode DCG-S1 augmente quand la vitesse de déplacement des nœuds augmente. L'augmentation de la vitesse de déplacement peut favoriser l'instabilité de réseau et empêcher la construction des groupes ce qui réduit le taux d'accessibilité et augmente le trafic. Le trafic généré par les méthodes M1, M2 et M3 reste toujours relativement très petit.
Pour observer l'évolution du trafic dans ces méthodes, nous avons changé l'échelle et édité l'état présenté par la figure 5.8 (B). Nous remarquons que les trois méthodes ne sont pas influencées par l'augmentation de la vitesse et la méthode M3 donne toujours le trafic le plus petit. La méthode M1 exécute la procédure de création des réplicas primaires pour toute nouvelle donnée créée ce qui élève le nombre de messages produits.



(A) (B)

Figure 5.8 : Vitesse de déplacement et le trafic.

La méthode M2 crée les réplicas primaires sauf pour les nouvelles données qui seront très demandées par les autres nœuds ceci diminue le nombre de messages produits.
La méthode M3 crée les réplicas primaires de la même façon que la méthode M2, mais pour les données qui ne dépassent pas une certaine taille (2 ko dans notre simulation). La méthode

M3 donne aussi la priorité de réplication aux données ayant un temps de réponse plus élevé ce qui explique un trafic plus petit pour cette méthode.

5.6.5 La charge et l'accessibilité

Cette expérience vise à observer l'influence de l'augmentation du nombre de requêtes d'accès sur le taux de disponibilité de données. Dans la figure 5.9, les axes horizontaux représentent le nombre de requêtes transmises par seconde. En général nous remarquons que les méthodes ne sont pas très influencées par la charge.

Dans cette figure nous observons que la méthode DCG-S1 donne un niveau élevé d'accessibilité. Quand la charge est faible et la vitesse de déplacement est faible, cette disponibilité diminue avec l'augmentation de la charge et la vitesse de déplacement, mais elle est plus influencée par la variation de la vitesse de déplacement que par l'augmentation de la charge, parce que quand le nœud mobile se déplace rapidement, ses liens avec ses voisins deviennent plus instables. Ce qui va engendrait des singletons et empêcher la formation de groupes et par la suite le partage de données avec les autres nœuds. D'où, une baisse de la disponibilité.

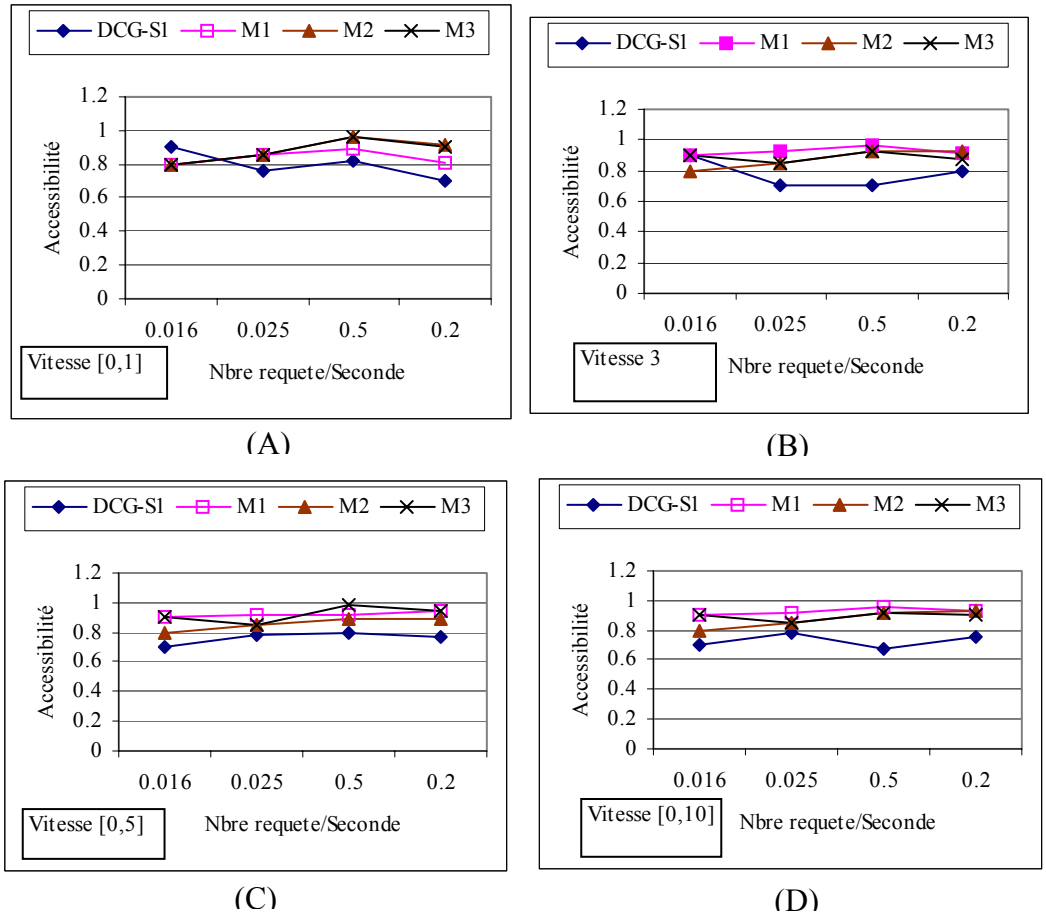


Figure 5.9 : La charge et l'accessibilité.

Pour nos méthodes nous remarquons qu'elles sont peu influencées par la charge et la variation de la vitesse. D'ailleurs nous observons que l'accessibilité augmente quand la charge

augmente, parce que dans nos méthodes les données sont répliquées localement si ses fréquences d'accès dépassent le seuil fixé. Les demandes d'accès sont d'abord envoyées à distance si la donnée concernée n'est pas disponible localement. Ces requêtes ne peuvent pas toutes être satisfaites. Mais après la création des réplicas pour ces données, puisque leurs fréquences d'accès dépassent le seuil, tous les accès suivants seront satisfaits plus rapidement.

5.6.6 La charge et le trafic

Dans la figure 5.10, nous remarquons que le trafic augmente avec l'augmentation de la charge. Ceci est dû à l'augmentation du nombre de message transmis par les nœuds du réseau. Quand le nombre de demande augmente, le trafic augmente, notamment dans le cas où les données demandées ne se trouvent pas localement et il n'existe pas un chemin qui mène vers ces données.

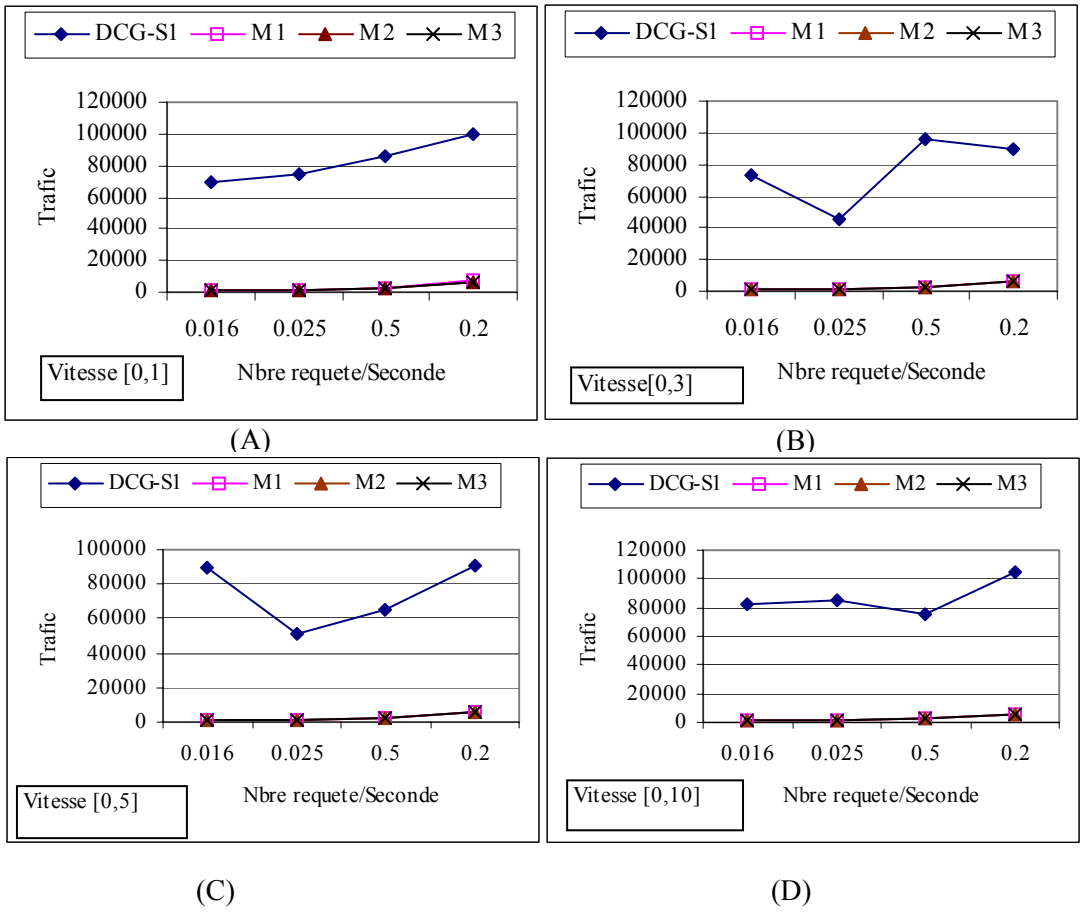


Figure 5.10 : La charge et le trafic.

Pour la méthode DCG-S1, nous observons dans la figure 5.10(A), que quand la charge augmente le trafic augmente. Dans cette méthode, si l'information recherchée n'est pas disponible chez le demandeur, la demande d'accès doit être diffusée dans le réseau, ce qui va augmenter le nombre de messages produits. Nous remarquons aussi pour cette méthode, que

le trafic est influencé par la variation de la vitesse de déplacement des nœuds. Le déplacement rapide des nœuds mobiles peut créer des liens instables entre les nœuds voisins et les groupes ne peuvent pas se former et les possibilités de partager des données sont alors limitées.
Dans le cas de nos méthodes, nous observons que le trafic n'est pas influencé par le changement de vitesse, mais par l'augmentation de la charge. Le trafic dans ces méthodes reste toujours très petit par rapport à DCG-S1. Car le trafic est principalement induit par les accès et les fréquences de ces accès.

5.6.7 La porté de communication et l'accessibilité

Nous observons dans la figure 5.11, que l'accessibilité augmente avec l'augmentation de la portée de communication. Quand la portée de communication est petite, chaque méthode donne presque la même accessibilité de données. C'est parce que le nombre de nœuds mobiles communicant entre eux est petit.
Pour la méthode DCG-S1, quand le rayon est très petit, les groupes de nœuds à liens stables ne peuvent pas se construire et la relocalisation de réplicas se produit ainsi rarement. Ceci diminue la disponibilité. Quand la portée de communication augmente l'accessibilité augmente, parce que la plupart des nœuds mobiles sont reliés entre eux et les réplicas peuvent être relocalisées.

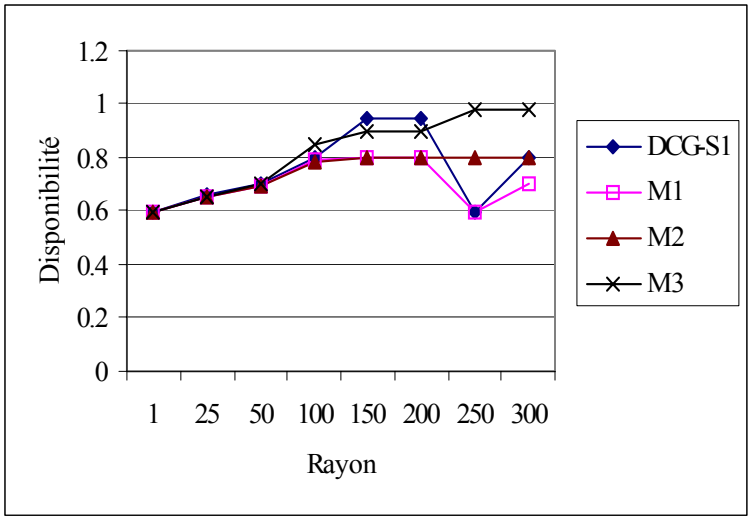


Figure 5.11 : La portée de communication et l'accessibilité.

Pour nos méthodes, nous remarquons que l'accessibilité augmente en parallèle avec l'augmentation de la portée de communication. Dans cette figure, nous observons que la méthode M3 donne le meilleur taux d'accessibilité de données, parce que cette méthode utilise une meilleure gestion de cache. Dans cette méthode les replicas primaires ne se produisent que pour les nouvelles données qui seront très demandées et leur taille ne dépassant pas un certain seuil. Pour la méthode M1, l'accessibilité diminue un peu quand la

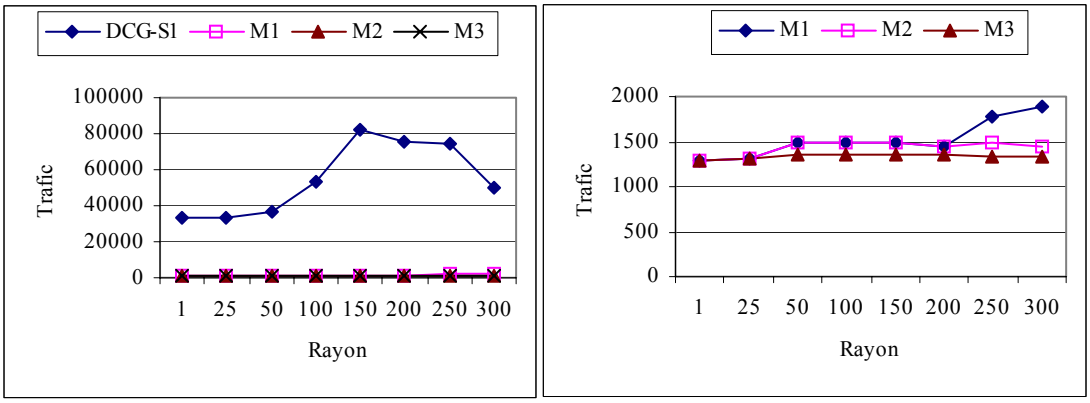
portée de communication devient plus grande, parce que dans ce cas, le nombre de nœuds qui peuvent être choisis par la procédure de création des réplicas primaires diminue(il y a plus de nœuds en trois sauts). Ceci peut diminuer l'accessibilité et augmente le temps de réponse.

5.6.8 La porté de communication et le trafic

Dans cette expérience, présentée par la figure5.12, nous remarquons que le trafic produit par les quatre méthodes est influencé par la variation de la portée de communication et la méthode DCG-S1 donne toujours le trafic le plus élevé.

Dans la figure 5.12(A), nous observons que lorsque la portée de communication est petite, le trafic provoqué par la méthode DCG-S1 est petit. C'est parce que le nombre de nœuds mobiles reliés entre eux est petit, la relocalisation des nœuds et la duplication ne causent pas ainsi un grand trafic. Quand la portée de communication est très grande, le trafic produit par la méthode DCG-S1 diminue parce que le nombre de nœuds mobiles dans un groupe est très grand et la relocalisation ou le déplacement de copies se produit plus rarement.

Les méthodes que nous avons proposées, ne sont pas très influencées par la variation de la portée de communication. Pour voir clairement l'effet du changement de rayon sur ces méthodes, nous avons changé l'échelle et édité le résultat présenté par la figure 5.12(B).



(A) (B)

Figure5.12 : La portée de communication et le trafic.

La méthode M3 donne toujours le plus petit trafic. Dans cette dernière le trafic reste inchangé quand le rayon augmente. La méthode M2 donne un trafic plus ou moins élevé que M1, parce que dans cette méthode, les réplicas primaires ne sont pas conditionnés par la taille, c à d la procédure de création de ces réplicas est exécutée pour toute nouvelle donnée qui sera demandée. Nous remarquons que le trafic dans cette méthode reste ainsi inchangé quand le rayon est élevé. Pour la méthode M1, le trafic est toujours élevé par rapport aux méthodes (M2, M3), ceci est du à l'exécution inconditionnée de la procédure de création des réplicas primaires. Mais dans tout les cas nos méthodes donnent un trafic très petit relativement à la méthode DCG-S1.

5.6.9 Scalability et l'accessibilité

Dans l'ensemble nous remarquons que l'accessibilité dans la figure 5.13, augmente avec l 'augmentation du nombre de nœud. Parce que la connectivité augmente. Nous observons pour la méthode DCG-S1, que la disponibilité diminue puis augmente cette instabilité est toujours liée à l'instabilité des liens reliant les nœuds du réseau, parce que la base de cette méthode est la construction des groupes stables. Ce qu'on a remarqué durant la simulation que parfois ces groupes ne peuvent pas se former.

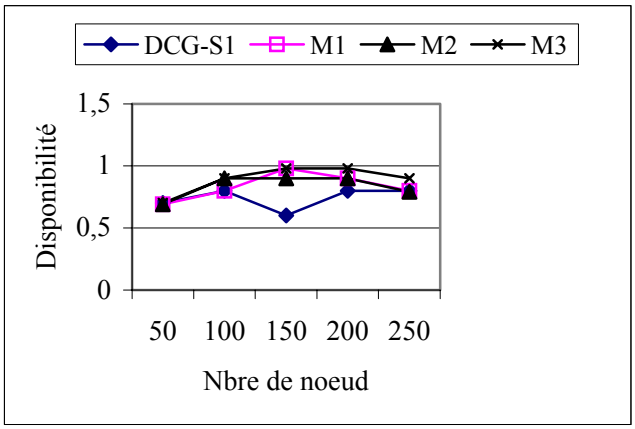


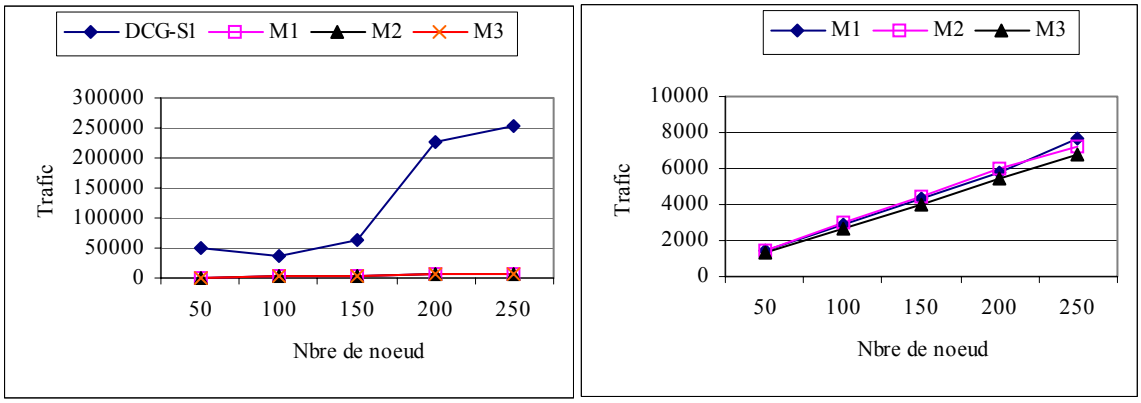
Figure 5.13 : Scalability et l'accessibilité.

La méthode M1 donne un niveau d'accessibilité élevé puis diminue un peu quand le nombre de nœuds devient plus grand, parce que dans cette méthode, tout nœud qui crée une nouvelle donnée, doit exécuter la procédure de création des réplicas primaires, ce qui permet de rapprocher les données aux utilisateurs. Mais quand le nombre de nœud augmente le nombre de données nouvelles augmente aussi. Ceci va consommer un espace mémoire de quelques nœuds (les nœuds choisis par la procédure de création des réplicas primaires). C'est pour cette raison que l'accessibilité diminue un peu quand le nombre de nœud augmente. Les méthodes M2 et M3 donnent le même taux d'accessibilité au départ mais quand le nombre de nœud augmente, l'accessibilité donnée par la méthode M3 dépasse celle fournie par la méthode M2. C'est parce que la méthode M3 n'exécute pas la procédure de création des réplicas primaires pour chaque nouvelle donnée créée. En plus M3 préfère répliquer les données qui ont un temps de réponse plus élevé que celles ayant un temps de réponse plus petit. D'où une meilleure gestion de charge par rapport aux autres méthodes.

5.6.10 Scalability et le trafic

Nous remarquons dans cette expérience, que le trafic augmente avec l'augmentation du nombre de nœud. Dans la figure 5.14(A), la méthode DCG-S1 donne le trafic le plus élevé, dans cette derrière l'exécution périodique de la procédure de construction des groupes produit

un nombre très élevé de messages, ce nombre de messages croît avec l'augmentation du nombre de nœuds car chaque nœud produit périodiquement un nombre élevé de message.



(A) (B)

Figure 5.14 : Scalability et accessibilité.

Dans le cas de nos méthodes, nous observons dans la figure 16(B), que le trafic augmente quand le nombre de nœuds augmente, mais cette augmentation reste très petite par rapport à la méthode DCG-S1. Dans ces méthodes, la méthode M3 donne le plus petit trafic.

5.7 Conclusion

Après les résultats de simulation que nous avons obtenu. Nous pouvons conclure que les méthodes que nous avons proposées donnent de bonnes performances. Le taux d'accessibilité est toujours élevé et le trafic provoqué par ces méthodes reste satisfaisant.

La méthode M1 donne un niveau d'accessibilité élevé, mais quand le nombre de création de données nouvelles augmente, le taux d'accessibilité diminue un peu et le trafic augmente, parce que cette méthode exécute la procédure de création des réplicas primaires pour chaque nouvelle donnée créée. Cette technique permet de rapprocher toute information aux utilisateurs. Mais elle consomme l'espace mémoire, donc cette méthode est très efficace quand les outils mobiles ont un espace mémoire plus ou moins élevé ou le nombre de création de données nouvelles est plus ou moins petit.

La méthode M2 donne un niveau d'accessibilité élevé presque comme la méthode M1. Le trafic produit par cette méthode est inférieur à celui provoqué par M1. La procédure de création de réplicas primaires dans cette méthode, ne s'exécute que pour les données très demandées. Ce qui fait que l'espace mémoire consommé par cette méthode est inférieur à celui consommé par la méthode M1. Donc la méthode M2 est très efficace quand l'espace mémoire des outils mobiles n'est pas important.

La méthode M3 donne un taux d'accessibilité élevé et fournit un trafic inférieur par rapport à M1 et à M2. Dans cette méthode la procédure de création des réplicas primaires, ne s'exécute que pour les nouvelles données qui sont très demandées et de taille limitée par un certain seuil. Ceci fait que la consommation de l'espace mémoire soit inférieure à celle consommé par M1 et M2. Pour M3, les données accédées qui ont un temps de réponse supérieur avec une fréquence d'accès dépassant le seuil demandé, sont prioritaires à la réplication par rapport à celles qui ont un temps de réponse inférieur et dont la fréquence d'accès dépasse le seuil demandé. Cette spécificité permet de réduire le trafic et donne une meilleure gestion du cache. La méthode DCG-S1 donne un taux d'accessibilité élevé mais, la procédure de construction des groupes stables produit un trafic très élève et prend un temps d'exécution très long, ceci va augmenter la consommation de l'énergie et dégrade les performances du réseau. Les résultats de la simulation nous montrent que le trafic et le temps d'exécution dans cette méthode croît considérablement quand la taille du réseau augmente. Vu les résultats obtenus pour les méthodes proposées, nous pensons que ses dernières passent mieux à l'échelle.

Conclusion générale

Les réseaux sans fil ad hoc ou les MANET (Mobile Ad hoc NETwork), sont des réseaux ne disposant d'aucune infrastructure préexistante et formés de nœuds mobiles interconnectés par des liaisons sans fils. Leurs architectures changent au gré de l'apparition et du mouvement des nœuds. L'absence d'infrastructure se traduit par la nécessité de mettre en place des solutions adaptées reposant sur la participation de l'ensemble des nœuds formant le réseau ad hoc.

Le changement fréquent de la topologie, le partitionnement du réseau, la bande passante limitée et les ressources d'énergie et de stockages faibles font la complexité de l'environnement ad hoc. Dans tel environnement limité par ces contraintes, le problème de disponibilité de données reste toujours compliqué. La réplication de données qui permet de créer un nombre de copie de même source de donnée en plusieurs nœuds, représente la solution idéale pour augmenter l'accessibilité de données et garder disponible de l'information même en cas de partitionnement du réseau. Cette solution doit tenir compte de ces contraintes et ne doit pas générer un trafic élevé.

T.hara, dans sa solution représenté par la méthode DCG-S1, utilise dans sa conception la notion de construction des groupes stables et partage les réplicas entre les nœuds du même groupe. Cette méthode a permis d'augmenter le taux d'accessibilité. Mais la procédure de construction des groupes produisait un grand nombre de messages, ce qui augmente la charge du réseau et consomme beaucoup de l'énergie et de la bande passante. Le nombre de message généré augmente rapidement, quand la taille du réseau augmente, cette spécification fait que la méthode DCG-S1 ne sera pas vérifier la scalability du réseau.

Notre solution présentée par les trois méthodes proposées, a permis d'augmenter le taux d'accessibilité et de diminuer le traffic. Dans cette solution nous avons essayé de garder les données prés de l'utilisateur et d'approcher autant que possible l'information au demandeur. Les trois méthodes s'exécutent en deux parties. La première partie, permet de créer les réplicas primaires de chaque nouvelle donnée. Dans cette partie un nœud peut cacher un Path vers une donnée qui ne se trouve pas localement. La deuxième partie consiste de la gestion dynamique du réplicas. Dans ces méthodes, chaque donnée est associée à une fréquence d'accès interne et une fréquence d'accès externe, un nœud demande la réplication d'une donnée si sa fréquence d'accès externe dépasse un certain seuil. La première méthode exécute la procédure de création du réplicas primaires pour chaque nouvelle donnée, alors que la deuxième méthode l'exécute seulement pour les données qui seront demandée, la troisième méthode exécute cette procédure pour les données demandées et qui leur taille ne dépasse pas une centaine limite. Le trafic généré par ces méthodes reste toujours bas quand la taille du réseau augmente. Ceci permet de dire que ces méthodes peuvent vérifier la scalability du réseau.

Dans les travaux futures, nous allons en premier temps étendre ces méthodes pour prendre le cas de mise à jour de données. Nous voulons ainsi profiter de la similitude du système P2P et du réseau mobile ad hoc et d'appliquer la technique de la table de hachage distribuée (DHT *Distributed Hash Tables*) qui permet une bonne recherche de données en minimisant le nombre de nœuds contactés. Bien sure cette technique ne sera pas directement appliquée vu la différence des ressources entre les deux environnements.

BIBLIOGRAPHIES

[Aho74] : A.V. Aho, J.E. Hopcroft, and J.D. Ullman, “The Design and Analysis of Computer Algorithms”, Addison-Wesley, 1974.

[Ald76]: P. A. Alsberg and J. D. Day, “A principle for resilient sharing of distributed resources”, In Proceedings of the International Conference on Software Engineering, 1976.

[All01]: ALBITZ, P. AND LIU, “DNS and BIND (4th edition)”, O’Reilly & Associates. ISBN 0-596-00158-4. 2001.

[Bai94]: D. Barbara and T. Imielinski, “Sleepers and workholics: Caching strategies in mobile environments”, Proc. ACM SIGMOD’94, pp.1-12, 1994.

[Ber97] : P. BERNSTEIN, E. NEWCOMER, “Principles of Transaction Processing”. Morgan Kaufmann. 1997.

[Bou02]: Malika Boulkenafed, Valerie Issarny, “AdHocFS: Sharing Files in WLANs”, INRIA-Rocquencourt Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay C´edex, France 2002

[Bro98] : J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, and J. Jetcheva, “A performance comparison of multi-Hop wireless ad hoc network routing protocols,” Proc. Mobicom’98, pp.159-164, 1998.

[Bta00] : L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, M. Gerla, “GloMoSim : A Scalable Network Simulation Environment”, computer Science Department, University of California at Los Angles, 2000.

[Bur76]: I. W. Burr, “Statistical Quality Control Methods”, Volume 16, New York: Marcel Dekker, Inc (1976).

[Bwe82] : D.J. Baker, J. Wieselthier, and A. Ephremides, “A distributed algorithm for scheduling the activation of links in a self-organizing, mobile, radio network”, Proc. IEEE ICC’82, pp.2F6.1-2F6.5, 1982.

[Bzg98] : R. Bagrodia, X. Zeng, and M. Gerla, “GloMoSim - A Library for Parallel Simulation of Large-scale Wireless Networks”, computer Science Departement, University of California at Los Angles, 1999.

[Cal88] : M.J. Carey and M. Livny, “Distributed concurrency control performance: Astudy of algorithm, distribution, and replication”, Proc.14th VLDB Conference, pp.13-5,1988.

[Cet00a] : U. CETINTEMEL et P. J. KELEHER, “Light-Weight Currency Management Mechanisms in Deno”, Proceedings of the 10th International Workshop on Research Issues on Data Engineering : Middleware for Mobile Business Applications and E-Commerce (RIDE'2000), p. 17ñ24, San Diego, California, USA, février 2000.

[Cet00b] : U. CETINTEMEL et P. J. KELEHER, " Performance of Mobile, Single-Object, Replication Protocols" , Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems (SRDS'2000), p. 218-227, Germany, October 2000.

[Cet01] : U. CETINTEMEL, P. J. KELEHER et M. FRANKLIN, "Support for Speculative Update Propagation and Mobility in Deno" , Proceedings of the 22nd International Conference on Distributed Computing Systems *(ICDCS'2001)*, Mesa, Arizona, USA, avril 2001.

[Cla03]: Claudio Basile, Marc-Olivier Killijian, and David Powell, "A survey of dependability issues in mobile wireless networks". Technical report, LAAS CNRS Toulouse, France, February 21 2003.

[Cro87a]: S. V. Crowder, "A Simple Method for Studying Run-length Distributions of Exponentially Weighted Moving Average Charts", 1987, *Technometrics*, 29, 401-408.

[Cro87b]: S. V.Crowder, "Average Run Lengths of Exponentially Weighted Moving Average Charts", 1987b, Journal of Quality Technology, 19, 161 -164.

[Dar 99] : DARPA ATO Sponsored Research, "Design of Mobile Adaptive Networks Using Simulation And Agent Technology", UCLA, 1999.

[Die94]: D. J. DIETTERICH, "DEC data distributor: for data replication and data warehousing", In Int. Conf. on Management of Data (Minneapolis, MN, USA, May 1994), pp. 468. ACM. 1994.

[Edw97] : W. K. EDWARDS, E. D. MYNATT, K. PETERSEN, M. J. SPREITZER, D.B.TERRY et M. M. THEIMER, "Designing and Implementing Asynchronous Collaborative Applications with Bayou", Proceedings of the 10th ACM Symposium on User Interface Software and Technology (UIST'97), p. 119ñ128, Banff, Alberta, Canada, octobre1997.

[Fuc94] : A.W. Fu and D.W.L Cheung, "A transaction replication scheme for a replicated database with node autonomy," Proc. 20th VLDB Conference, pp.214-225, 1994.

[Gif79] : D. K. Gifiord, "Weighted voting for replicated data", In Proceedings of the Seventh Symposium on Operating System Principles SOSP 7, pages 150–162, Asilomar Conference Grounds, Pacific Grove CA, ACM, New York. 1979.

[Guy98] : R. Guy, P. Reicher, D. Ratner, M. Gunter, W. Ma, and G. Popek, "Rumor: Mobile Data Access Through Optimistic Peer-to-peer Replication". In Proceedings: ER'98 Workshop on Mobile Data Access, 1998.

[Har00] : T.Hara, K.Harumoto, M.Tsukamoto, S.Nishio, "Dynamic replica allocation using database migration in broadband networks",Proc. IEEE ICDCS 2000, pp.376-384,2000.

[Har01]: T. Hara. "Effective replica allocation in ad hoc networks for improving data Accessibility". In Proc. of IEEE Infocom 2001, Anchorage, Alaska, U.S.A., volume 3, pages 1568-1576, April 2001.

[Har02]: T. Hara "Replica Allocation in Ad Hoc Networks with Periodic Data Update". Proceedings of the Third International Conference on Mobile Data Management (MDM.02) 0-7695-1500-2/02 2002 IEEE .

[Har03]: Takahiro Hara. “Replica allocation methods in ad hoc networks with data update”, ACM-Kluwer Journal on Mobile Networks and Applications, vol 8(4), pp 343-354, 2003.

[Har04]: Takahiro Hara and Sanjay Kumar Madria. “Dynamic Data Replication Using Aperiodic Updates in Mobile Adhoc Networks”, 9th international conference, DASFAA 2004

[Hei97] : T. W. Page, Jr., R. G. Guy, J. S. Heidemann, D. H. Ratner, P. L. Reiher, A. Goel, G. H. Kuenning, and G. Popek.“Perspectives on Optimistically Replicated, Peer-to-Peer Filing”. Software Practice and Experience, December 1997.

[Hku04] : Takahiro Hara and Sanjay Kumar Madria. “Dynamic Data Replication Using Aperiodic Updates in Mobile Adhoc Networks”, 9th international conference. (Eds.): DASFAA 2004, LNCS 2973, pp. 869–881, 2004. © Springer-Verlag Berlin Heidelberg 2004.

[Hsw94]: Y. Huang, P. Sistla, and O. Wolfson. “Data replication for mobile computer”, Proc. ACM SIGMOD’94, pp.13-24, 1994.

[Hun86]: Hunter, J. S. “The Exponentially Weighted Moving Average”, 1986, *Journal of Quality Technology*, 18, 203 -210.

[Hys03] : Takahiro Hara, Yin-Huei Loh, Shojiro Nishio. “ Data Replication Methods Based on the Stability of Radio Links in Ad Hoc Networks”**,** Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA’03) 1529-4188/03 2003 IEEE.

[Imi92]: T. Imielinski and B.R. Badrinath. “Querying in highly mobile distributed environments”, Proceedings of the 18th VLBD Conference, 1992.

[Joa03]: Joao Pedro Barreto. “Information sharing in mobile networks: a survey on replication strategies”, Instituto Superior T´ecnico/Distributed Systems Group, Inesc-ID Lisboa, September 2003

[Kel00]: P. J. KELEHER et U.ETINTEMEL. “Consistency management in Deno”, Mobile Networks and Applications (MONET), vol. 5, 2000.

[Kel99] : P. J. KELEHER, “Decentalized Replicated-Object Protocols”, Proceedings of the 18th Annual ACM Symposium on Principles of Distributed Computing (PODC'99), p.43-151, Atlanta, Georgia, USA, mai 1999.

[Kor01]: G. Kortuem. “When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad Hoc Networks”, in First IEEE International Conference on Peer-to-Peer Computing, 2001, pp. 75-91.

[Krb92] : N. Krishnakumar and A.J. Bernstein. “High performance escrow algorithms for replicated databases”, Proc. 18th VLDB Conference, pp.175-186, 1992.

[Kro86] : N. P.KRONENBERG, H. M.LEVY, AND W. D.STRECKER. “A closely-coupled distributed system”. ACM Trans. on Comp. Sys. (TOCS) 4, 4, 130–146, 1986.

[Kum85]: H. Kume, “Statistical Methods for Quality Improvement”, Tokyo: AOTS Chosakai, Ltd. 1985.

[Lig04]: Liangzhong Yin and Guohong Cao : “Supporting Cooperative Caching in Ad Hoc Networks”, IEEE INFOCOM, March 2004.

[Lls92]: R. Ladin, B. Liskov, L. Shrira, S. Ghemawat, Providing High Availability Using Lazy Replication. TOCS 10(4): 360-391 (1992).

[Mon96]: D. C. Montgomery, “Introduction to Statistical Quality Control”, Third Edition, New York: John Wiley & Sons, Inc. 1996.

[Msc86] : H. Morris James, H. Conner Michael, H. Howard John, S. Rosenthal David, and F. Donelson Smith. “Andrew: a distributed personal computing environment”. Communications of the ACM, 29(3):184–201, 1986.

[Ora96] : ORACLE. Oracle7 Server Distributed Systems Manual, Vol. 2. 1996

[Pet96]: K. Petersen, M. J. Spreitzer, D. B. Terry, and M. M. Theimer. Bayou: Replicated Database Services for World-wide Applications. In *Proceedings of the Seventh ACM SIGOPS European Workshop*, September 1996.

[Pet97]:]: K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, and A. J. Demers. “Flexible Update Propagation for Weakly Consistent Replication”. In Proceedings of the 16th ACM Symposium on Operating Systems Principles, October 1997.

[Phi94] : Philippe Quéinnec : Techniques de réplication de données pour les systèmes répartis à grande échelle, 1 avril 1994

[Pib95]: E. Pitoura and B. Bhargava, “Maintaining consistency of data in mobile distributed environments”, Proc. IEEE ICDCS’95, pp.404-413, 1995.

[Pst97]: PETERSEN, K., SPREITZER, M. J., TERRY, D. B., THEIMER, M. M., AND A. J. DEMERS, “Flexible. Update Propagation forWeakly Consistent Replication”. In 16th Symp. on Op. Sys. Principles (SOSP) (St. Malo, France, October 1997), pp. 288–301. 1997.

[Rar02] : R. Ramanathan and J. Redi, “A Brief Overview of Ad Hoc Networks: Challenges and Directions”, IEEE Communications, no. 50th Anniversary Commemorative Issue, pp. 20-22, May 2002.

[Rat98] : D. Ratner. “Roam: A Scalable Replication System for Mobile and Distributed Computing”. PhD thesis, UCLA, January 1998.

[Rat99] : D. Ratner, P.Reiher, G.J.Popek and R.Guy. “Peer Replication with Selective Control”. In *MDA '99: First International Conference on Mobile Data Access*, December 1999.

[Sat02] : M. Satyanarayanan, ”The evolution of coda”. ACM Transactions on Computer Systems (TOCS), 20(2):85–124, 2002.

[Sch01]: R. Schollmeier, “A Definition of *Peer-to-Peer* Networking for the Classification of *Peer-to-Peer* Architectures and Applications”, in First IEEE International Conference on Peer-to-Peer Computing, 2001, pp. 101-102.

[Spl98]: H.SPENCER, AND D.LAWRENCE, “Managing Usenet”. O’Reilly & Associates. ISBN 1-56592-198-4. 1998.

[Sto79] : M.Stonebraker. “Concurrency control and consistency of multiple copies of data in distributed ingress”.IEEE Transactions on Software Engineering, SE-5:188–194,1979.

[Ter95]: D. B. Terry, M. M. Theimer, K. Petersen, A. J. Demers, M. J. Spreitzer, and C. H. Hauser.“Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System”. In Proceedings of the 15th Symposium on Operating Systems Principles, December 1995.

[Ter98]: D. B. TERRY, K. PETERSEN, M. J. SPREITZER et M. M. THEIMER, ´ The Case for Non-transparent Replication : Examples from Bayou, *IEEE Data Engineering Bulletin*, vol. 21, n_4, December 1998.

[Wan02a]: K. Wang and B. Li. “Efficient and guaranteed service coverage in partitionable mobile ad-hoc networks”. In Proceedings of IEEE INFOCOM 2002.

[Wan02b] : Karen Wang and Baochun Li. “Group mobility and partition prediction in wireless ad-hoc networks”. In Proceedings of IEEE International Conference on communications (ICC 2002), volume2. New York City, New York, April 28-May 2 2002.

[Wan03]: Baochun Li and Karen Wang. “Nonstop: Continuous multimedia streaming in wireless ad hoc networks with node mobility”. IEEE Journal on Selected Areas in Communications, 21(10), December 2003.

[Woj92] : O.Wolfson and S. Jajodia, “Distributed algorithms for dynamic replication of data”, Proc. ACM PODS’92, pp.149-163, 1992.

[Ykx04]: Lu Yan, Kaisa Sere, Xinrong Zhou, Jun Pang , “Towards an Integrated Architecture for Peer-to-Peer and Ad Hoc Overlay Network Applications”, Turku Centre for Computer Science (TUCS), Abo Akademi University and Centrum voor Wiskunde en Informatica (CWI) P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands 2004

[Yma02]: Yasushi Saito and Marc Shapiro. “Optimistic replication”, Hewlett-Packard Laboratories, Palo Alto, CA (USA) Microsoft Research Ltd., Cambridge (UK) 2002.

[Yuv01] : H. YU, A.VAHDAT,”The Costs and Limits of Availability for Replicated Services. In 18th Symp. on Op. Sys. Principles *(SOSP)* (Lake Louise, AB, Canada, October 2001), pp. 29–42.

[Yuv02] : H.YU, A.VAHDAT, “Minimal replication cost for availability”. In 21st Symp. on Princ. of Distr. Comp. (PODC) (Monterey, CA, USA, July 2002).