Table des matières

Liste des figures

Remerciements

Je tiens à remercier en premier lieu ma directrice de mémoire M^me^ L. BOUALLOUCHE Maître de Conférences à l'université de Béjaïa, de m'avoir honoré avec la proposition de ce sujet. Je remercie également M^r^ D. AÏSSANI, Professeur à l'Université de Béjaïa, qui m'a co-encadré durant toute cette année, pour ses conseils et orientations. Je les remercie pour la confiance qu'ils m'ont accordée, pour leur disponibilité et tout le temps qu'ils m'ont consacré durant ce travail.

Je remercie tout particulièrement les membres de mon jury, qui ont accepté de juger ce travail. J'adresse mes très sincères remerciements à M^r^ B. MENDIL, Professeur à l'Université de Béjaïa, qui a bien voulu présider mon jury; à M^r^ A. BOUKERRAM, Maître de Conférences à l'Université de Sétif et à M^r^ M. BENMOHAMMED, Professeur à l'Université de Constantine, pour avoir accepté de juger mon travail. Je les remercie d'avance pour leurs critiques et suggestions.

Je tiens à exprimer toute ma reconnaissance à Ali LARBI pour avoir répondu à mon aide.

Je voudrais remercier toutes les personnes qui contribuent au bon fonctionnement de l'école doctorale.

Je ne pourrai clôturer ces remerciements sans me retourner vers les êtres qui me sont le plus chers, J'adresse mes remerciements à mes parents qui sans eux aucune réussite n'aurait été possible, et à toute ma famille.

Je voudrai remercier profondément mes amis Maroine BENARROUDJ, enseignant à l'université de M'Sila, et Adel CHEKAL, ingénieur en informatique, pour avoir toujours répondus présent pour mon aide, je leur suis infiniment reconnaissant.

Je voudrai remercier toutes les personnes qui ont contribuées, d'une manière ou d'une autre, au bon déroulement de ce travail.

Dédicaces

- *A mes chers parents qui m'ont apporté leur amour et leur soutient.*
- *A ma grand-mère.*
- *A mes frères et mes sœurs.*
- *A tous mes amis.*
- *A tous ceux qui sont chers.*

Mounir

Résumé

Le cache des nœuds mobiles est largement considéré comme une solution efficace pour améliorer les performances des systèmes. C'est le cas par exemple de la mise en cache coopérative, qui est fondée sur l'idée de partage des données en cache et la coordination entre plusieurs nœuds de réseau. Ces politiques coopératives peuvent être particulièrement efficaces pour accéder à l'information dans les réseaux mobiles Ad hoc. La plupart des politiques de remplacement de cache existantes effectuent le remplacement de manière indépendante, et prennent rarement en compte le partage et la coordination entre les caches de réseau pour le remplacement.

Notre travail consiste à proposer une politique coopérative de remplacement de cache dans les réseaux Ad hoc, basée essentiellement sur le principe de partage et de la coordination entre les nœuds du réseau afin d'améliorer les performances de ce dernier. Pour ce faire, nous avons mis en œuvre une politique qu'on a appelée *Mobile Zone Serveur (MZS)*, utilisant un paramètre de coopération *NBclients(i)*.

Les résultats de la simulation ont montré que MZS donne de meilleures performances en termes de taux de succès de requêtes et taux de succès en octets dans une zone en comparaison avec d'autres politiques de remplacement de cache.

Mots clés : Réseaux Ad hoc, Cache coopérative, Remplacement de cache, Simulation.

Abstract

The cache of mobile nodes is widely considered an effective solution to improve system performance. This is the case for example of cooperative caching, based on the idea of sharing cache data and coordination between multiple network nodes, these cooperatives policies can be particularly effective for accessing information in Ad hoc mobile networks. Existing cache strategies perform replacement independently, and rarely take into account the sharing and coordination between network caches for replacement.

Our job is to propose a cooperative caching policy in ad hoc networks, mainly based on the principle of sharing and coordination between the nodes of the network to improve the performance. To do this, we implemented a policy which has been called *Mobile Zone Server (MZS)*, using a parameter of cooperation *NBclients(i)*.

The simulation results showed that MZS gives better performance in terms of query success rate and bytes success rate compared with other cache replacement policies.

Keywords: Mobile Ad hoc networks, Cooperative caching, Cache replacement, Simulation.

Introduction générale

Les réseaux Ad hoc permettent aux utilisateurs de former spontanément un système de communication dynamique, ils sont utiles dans les situations où la connectivité réseau temporaire est nécessaire, et ils sont souvent utilisés dans les champs de bataille et en cas de catastrophe. Toutefois, les réseaux sans fil continuent à souffrir de la bande passante limitée, de la communication de faible qualité, des déconnexions fréquentes du réseau et des ressources locales limitées [73].

Plusieurs études sur les mécanismes d'utilisation des caches dans les réseaux Ad hoc ont montré que les mécanismes de mise en cache indépendantes ne peuvent pas améliorer efficacement les performances du système. C'est la raison pour laquelle les caches coopératifs, fondés sur l'idée de partage et de coordination des données entre caches, ont émergé comme une approche efficace pour surmonter la limitation de la mise en cache indépendante [24], [89].

Il y a plusieurs questions importantes relatives à la gestion du cache: remplacement, cohérence et pré-chargement. Dans notre travail, nous étudions le problème de remplacement de cache, qui a été largement considéré dans les réseaux câblés, mais il a reçu beaucoup moins d'attention dans les réseaux mobiles Ad hoc.

La plupart des stratégies existantes de remplacement de cache effectuent le remplacement de manière indépendante [60], [89]. Ces politiques de remplacement ne sont pas très efficaces dans la mise en cache des réseaux Ad hoc. A titre d'exemple, si on considère une région où les mobiles ont les caractères d'accès identiques ou similaires, et que chaque nœud exerce des remplacements de cache de façon autonome, il est très probable que les mêmes données seront stockées dans chaque nœud. Donc, il est inutile d'envoyer une demande aux voisins qui détiennent les mêmes éléments en cache. En conséquence, la stratégie de remplacement coopérative est plus efficace dans un tel état.

L'objectif de ce travail est de proposer une politique de remplacement coopérative de cache, basée sur le principe de partage de l'espace cache et la coordination entre les nœuds lors de remplacement, et qui servira à améliorer les performances globales des réseaux mobiles Ad hoc, tout en prenant en considération les caractéristiques de trafic Web, ainsi que les caractéristiques spécifiques des réseaux mobiles Ad hoc.

Organisation du mémoire

Ce mémoire est organisé en quatre chapitres comme suit :

Le premier chapitre présente une étude des réseaux mobiles Ad Hoc avec ses différentes caractéristiques, ainsi que les différentes normes des réseaux locaux sans fil, tel que le Bluetooth, la 802.11, HiperLAN 1 et 2. Parmi ces normes, nous avons prêté une attention particulière à la norme 802.11 établie par l'IEEE qui fonctionne en deux modes, le mode infrastructure (le PCF Point Coordination Function adapté aux réseaux mobiles cellulaires), et le mode appelé Ad hoc (le DCF Distributed Coordinated Function qui est plus adapté aux réseaux mobiles Ad Hoc). Ensuite, nous avons exposé les différentes approches de routage des réseaux mobiles Ad Hoc, ainsi que quelques protocoles de routage qui leurs sont dédiés.

Dans le deuxième chapitre, nous réalisons un état de l'art sur les caches. Nous donnons une classification des caches selon leurs localisations et leurs fonctions. Puis, nous détaillons le fonctionnement des caches en décrivant un certain nombre de concepts liés aux caches, ainsi que les métriques de performances des caches et l'apport d'utilisation de cache dans les réseaux mobiles Ad Hoc ; ensuite, nous abordons les caches coopératifs et leurs différents types. Enfin, nous exposons les différentes techniques de coopération dédiées aux réseaux mobiles Ad hoc.

Le troisième chapitre présente une classification des politiques de remplacement de caches qui existent dans la littérature, en donnant pour chaque classe ses avantages et ses inconvénients. Nous terminons par une proposition d'une politique de remplacement de cache, MZS ***Mobile Zone Server***, pour les réseaux mobiles Ad Hoc.

Dans le dernier chapitre, nous considérons quelques méthodes d'évaluation de performances, ainsi que quelques caractéristiques du trafic Web. Puis, nous décrivons le modèle de notre système et quelques caractéristiques du trafic. Nous terminons par la comparaison des résultats de simulation de notre politique de remplacement de cache MZS avec quatre autres politiques LFU, LRU, ZC et GC. Nous clôturons notre mémoire par une conclusion générale.

Chapitre 1

Les réseaux Ad hoc

I.1. Introduction

Les environnements mobiles offrent aujourd'hui une grande flexibilité d'emploi. En particulier, ils permettent la mise en réseau des sites dont le câblage serait trop onéreux à réaliser dans leur totalité, voire même impossible.

Le concept des réseaux mobiles Ad hoc essaie d'étendre les notions de la mobilité à toutes les composantes de l'environnement. Ici, contrairement aux réseaux basés sur la communication avec infrastructure (cellulaire), aucune administration centralisée n'est disponible, ce sont les hôtes mobiles eux-mêmes qui forment une infrastructure du réseau. Aucune supposition ou limitation n'est faite sur la taille du réseau Ad hoc, le réseau peut contenir des centaines ou des milliers d'unités mobiles.

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

Notre travail entre dans le cadre de l'étude du problème de cache dans les réseaux mobiles Ad hoc. Notre étude offre principalement une étude synthétique des travaux de recherche qui ont été fait, et qui se font à l'heure actuelle, dans le but d'améliorer les performances du réseau en proposant ou en améliorant une politique de remplacement de cache du réseau Ad hoc. Comme nous allons voir les avantages et les inconvénients de quelques politiques de remplacement déjà étudiées dans la littérature.

I.2. Définition

Les réseaux Ad hoc aux quels nous sommes intéressés sont ceux décrits et étudiés par le groupe de travail MANET (Mobil Ad hoc Network) de l 'IETF (Internet Engineering Tast Force). Une définition de ces réseaux est donnée formellement dans RFC 2501 [Naval Research Laboratory, January 1999] :

« *Un réseau Ad hoc comprend des plates-formes mobiles (par exemple, un routeur interconnectant différents hôtes et équipements sans fil) appelées nœuds qui sont libres de se déplacer sans contrainte. Un réseau Ad hoc est donc un système autonome de nœuds mobiles Ce système peut fonctionner d'une manière isolée ou s'interfacer à des réseaux fixes au travers des passerelles*».

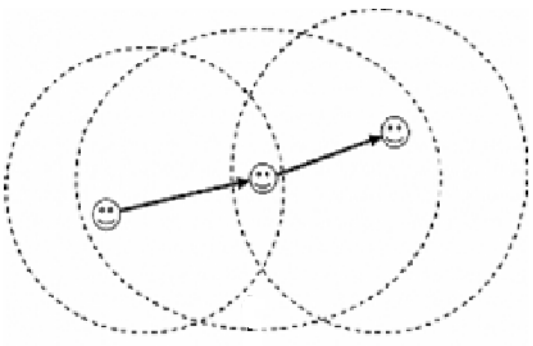


FIG I. 1. Le mode sans infrastructure (IBSS) (Independent Basic Service Set)

I.3. Les domaines des réseaux Ad hoc

Les réseaux Ad hoc sont des réseaux mobiles et sans fil capables de fonctionner sans infrastructure. Ils s'adaptent dynamiquement à leur environnement, et à leur topologie [70].

Cette idée de pouvoir se connecter en réseau n'importe où et n'importe comment intéresse particulièrement les militaires pour les opérations en milieu hostile ou pour les opérations de secours après une catastrophe naturelle. Mais les applications sont également nombreuses dans le monde civil, Par exemple :

I.3.1. Les services d'urgence

Opération de recherche et de secours des personnes, tremblement de terre, feux, dans le but de remplacer l'infrastructure filaire.

I.3.2. Le travail collaboratif et les communications dans des entreprises ou bâtiments

Dans le cadre d'une réunion ou d'une conférence par exemple.

I.3.3. Applications commerciales

Pour un paiement électronique distant (taxi) ou pour l'accès mobile à l'Internet, où service de guide en fonction de la position de l'utilisateur.

I.3.4. Réseaux de senseurs

Les capteurs, chargés de mesurer les propriétés physiques des environnements (comme la température, la pression...), sont dispersés (le plus souvent lâchés d'un avion ou d'un hélicoptère) par centaines, voire par milliers sur le site, effectuent leurs mesures et envoient les résultats à une station par l'intermédiaire d'un routage Ad hoc à travers le réseau.

I.3.5. Le cadre informatique

Dans le cadre de l'informatique, les réseaux Ad hoc peuvent servir à établir des liens entre ses différents composants. Dans ce cas, on parle non plus de LAN (Local Area Network) mais de PAN (Personnal Area Network)

D'une façon générale, les réseaux Ad hoc sont utilisés dans toute application où le déploiement d'une infrastructure réseau filaire est trop contraignant, soit parce qu'il est difficile à mettre en place, soit parceque la durée d'installation du réseau ne justifie pas de câblage à demeure.

I.4. Caractéristiques des réseaux Ad hoc

Les réseaux Ad hoc en tant que nouveau paradigme des réseaux sans fil présentent de nombreuses caractéristiques dont certaines leur sont bien spécifiques et les différencient des réseaux mobiles classiques [72]

I.4.1. Une 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.

I.4.2. Rapidité de déploiement

Les réseaux Ad hoc peuvent être facilement installés dans les endroits difficiles à câbler, ce qui élimine une bonne part du travail et du coût généralement liés à l'installation et réduit d'autant le temps nécessaire à la mise en route.

I.4.3. Une 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. En effet, seules les bandes des 2 et 5 GHz restent ouvertes pour les réseaux sans fil.

I.4.4. Des contraintes d'énergie

Les hôtes mobiles sont alimentés par des sources d'énergie autonomes donc restreintes, comme les batteries, par conséquence la durée de traitement est réduite. Donc le paramètre d'énergie doit être pris en considération dans tout contrôle fait par le système.

I.4.5. Une sécurité physique limitée

Les réseaux mobiles Ad hoc sont plus touchés par le paramètre de sécurité, que les réseaux filaires classiques. Cela se justifie entre autres par les vulnérabilités des liens radio aux attaques, ainsi que les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé.

I.4.6. L'absence d'infrastructure

Les réseaux Ad hoc se distinguent des autres réseaux mobiles par l'absence d'infrastructure préexistante et de tout genre d'administration centralisée. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau d'une manière continue.

I.4.7. Equivalence des nœuds du réseau

Dans un réseau classique, il existe une distinction nette entre les nœuds terminaux (stations, hôtes) qui supportent les applications et les nœuds internes (routeurs par exemple) du réseau, en charge de l'acheminement des données. Cette différence n'existe pas dans les réseaux Ad hoc car tous les nœuds peuvent être amenés à assurer des fonctions de routage.

I.4.8. Communication par lien radio

Les communications entre les nœuds se font par l'utilisation d'une interface radio. Il est alors important d'adopter un protocole d'accès au médium qui permet de bien distribuer les ressources radio et ceci en évitant le plus possible les collisions et en réduisant les interférences. Les technologies de communication sans fil sont alors indispensables à la mise en place d'un réseau Ad hoc.

I.5. Les normes des réseaux locaux sans fil

Plusieurs normes pour les réseaux locaux sans fil (WLANs, Wireless Local Area Networks) à portée limitée on été développées (visant des usages à l'échelle du bureau ou du bâtiment). Leur arrivée a soulevé un engouement nouveau pour les réseaux radio multi-sauts, qui étaient le domaine exclusif des militaires. Parmi ces normes, on peut citer: Bluetooth, HiperLAN (type 1 et type 2) et 802.11 avec ses différentes versions.

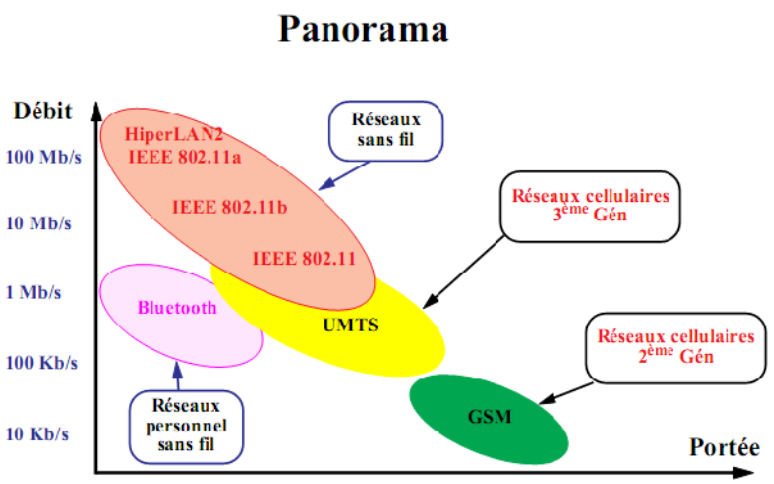


FIG I. 2 : Positionnement de différentes normes de réseaux locaux sans fil

I.5.1. Le Bluetooth

Bluetooth a plutôt pour objectif de faire disparaître les câbles entre les divers équipements numériques (périphériques d'ordinateur tels que clavier, imprimante, modem, ou encore appareil photo numérique, PDA, walkman, etc.). Les équipements Bluetooth ont donc des portées et des débits assez modestes, ainsi qu'une consommation électrique en rapport.

C'est le groupe de travail Bluetooth SIG (Bluetooth Special Interest Group) qui a élaboré les spécifications de Bluetooth, ils l'ont développé pour permettre la réalisation de réseaux personnels (PAN : Personnal Area Network). Ces réseaux doivent donc permettre des communications à courte portée, à des débits faibles ou moyens entre toute sorte d'équipements. Ils travaillent dans la bande ISM (Industrial, Scientific and Medical) des 2.4 GHz à des puissances leur permettant d'atteindre des portées allant du mètre à la centaine de mètres environ.

Les réseaux Bluetooth sont construits de manière centralisée. Un maître élu peut prendre en charge jusqu'à huit esclaves et forme ainsi un piconet. Dans un piconet, c'est le maître qui contrôle toutes les transmissions. Les esclaves ne peuvent émettre des paquets que s'ils y ont été invités par le maître. Ce dernier doit donc les interroger régulièrement pour savoir s'ils ont des données à envoyer (polling).

Plusieurs piconets peuvent être reliés afin de former une structure plus grande appelée scatternet. A cette fin, les mobiles peuvent quitter temporairement leur piconet pour aller s'attacher à un autre. La figure **FIG I.3** donne un exemple de scatternet. Un esclave alterne entre les piconets 1 et 2 afin d'en assurer la liaison et un mobile esclave du piconet 2 peut être aussi maître dans le piconet 3 (le processus de dé-association est réversible, il est possible de quitter temporairement un piconet puis de le rejoindre à nouveau et de revenir à la situation initiale)[26].

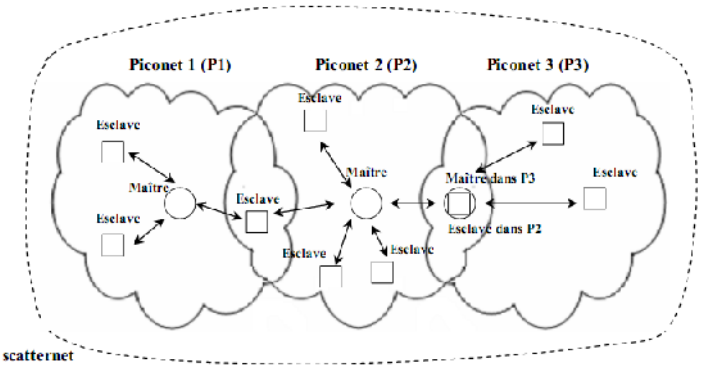


FIG I. 3 : Un scatternet bluetooth

I.5.2. L' HiperLAN 1 [12]

High Performance Local Area Network type 1 (HiperLAN 1) est un standard de l'European Technical Standard Institute (ETSI). Il décrit le fonctionnement d'équipements travaillant dans la bande des 5.15-5.30 GHz et permettant d'atteindre des débits de 23.5 Mbit/s sur une distance d'environ 50 mètres. L'architecture est totalement décentralisée. Il n'y a pas de notion de point d'accès mais les nœuds HiperLAN 1 peuvent cependant avoir des rôles de passerelles. Les caractéristiques les plus marquantes d'HiperLAN 1 sont :

Le mécanisme d'accès au médium évolué, qui gère les priorités. Il est possible d'obtenir des garanties de qualité de service, particulièrement utiles pour les flux multimédias.

La possibilité d'étendre le réseau au delà de la portée radio, par sauts successifs (des nœuds jouant le rôle d'intermédiaires entres ceux qui sont trop loin pour communiquer directement).

Les fonctionnalités d'HiperLAN 1 sont organisées comme présenté sur la figure **FIG I.4**. Nous pouvons remarquer que le mécanisme d'accès au médium est au cœur du système. C'est lui qui permet en particulier la gestion des priorités. Le fonctionnement de ce mécanisme est particulièrement intéressant puisqu'il est prévu pour fonctionner dans un contexte Ad hoc.

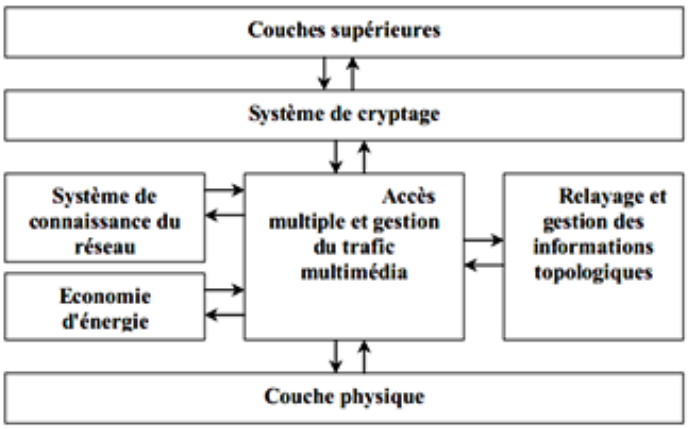


FIG I. 4 : L'organisation générale d'hiperLAN type1

I.5.3. L'HiperLAN 2 [13]

L'organisation générale d'HiperLAN type 2 est présentée sur la figure **FIG I.5.** On remarque en premier lieu que la norme prévoit la compatibilité avec diverses technologies (TCP/IP bien sûr, mais aussi ATM, UMTS et l'IEEE 1394 connue aussi sous le nom de Firewire). Contrairement au type 1, HiperLAN type 2 est basé sur une centralisation poussée. Les points d'accès sont d'ailleurs indifféremment appelés Access Points (AP) ou Central Controler (CC). Les points d'accès sont reliés entre eux par une infrastructure réseau filaire ou non-filaire, les mobiles s'attachent ensuite à ces points d'accès pour accéder aux ressources du réseau. HiperLAN 2 peut aussi fonctionner sans infrastructure fixe. Dans ce cas, un mobile est élu pour jouer le rôle de contrôleur central et les autres vont s'attacher à lui.

Bien entendu, dans ce type de configuration il n'est pas question de réseau Ad hoc au sens où le groupe MANET l'entend. Tout au plus, les mobiles pourront communiquer directement entre eux (un saut), ou alors par l'intermédiaire du contrôleur central (deux sauts). L'accès au médium reflète ces choix architecturaux, et c'est le contrôleur central qui décide de l'ordonnancement des communications dans la zone qu'il gère. Ces communications se font grâce à des trames de durée fixe (2 ms) qui sont elles-mêmes découpées en différents champs de longueur variable et qui portent soit les informations de contrôle du point d'accès, soit les données dans un sens ou dans l'autre. Il faut noter également que la couche physique d'HiperLAN type 2 est très semblable à la

couche physique de 802.11a. De part son architecture très ciblée, HiperLAN 2 est donc peu adapté aux réseaux Ad hoc.

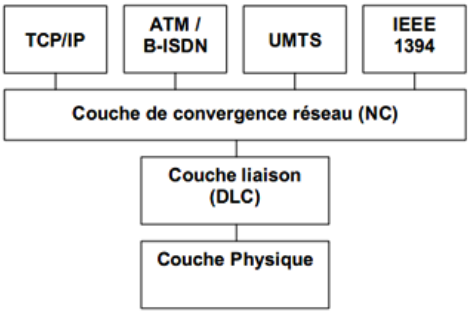


FIG I. 5. L'organisation générale d'HiperLAN type 2

I.5.4. La 802.11

802.11 est une norme établie par l'IEEE. Elle décrit les couches physiques et MAC d'interfaces réseau radio et infra-rouge. Les débits possibles varient entre 1 et 54 Mbit/s suivant les techniques et les éventuelles extensions de la norme employées. Les portées prévues varient entre quelques dizaines et quelques centaines de mètres en fonction de la vitesse choisie et de l'environnement. Cette norme cible deux contextes d'utilisation :

• Le mode "infrastructure" (l'utilisation privilégiée de 802.11), où des stations de base reliées entre elles par un réseau filaire assurent la couverture d'une certaine zone et prennent en charge les mobiles dans leur voisinage (**FIG I.6**).

• Le mode appelé Ad hoc, qui consiste en fait simplement à autoriser les communications entre deux mobiles à portée l'un de l'autre, sans intervention de stations ou d'autres mobiles extérieurs (**FIG I.6**).

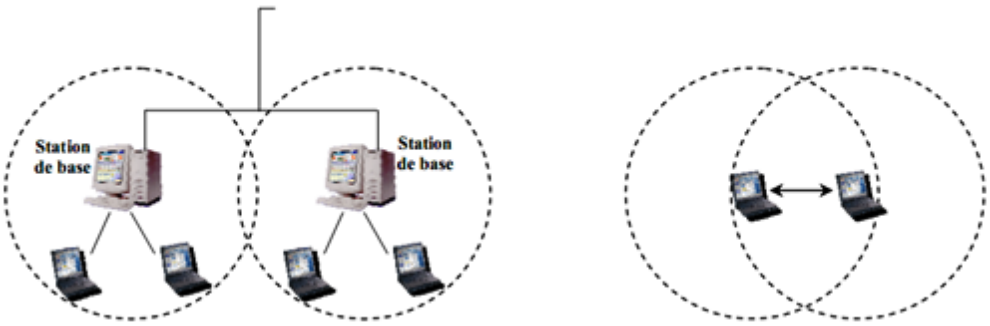


FIG I. 6 : Réseau 802.11 en mode infrastructure et Réseau 802.11 en mode Ad hoc

I.5.4.1. La couche physique

Initialement, le standard IEEE 802.11 permet l'utilisation de trois couches physiques différentes (FHSS, DSSS et IR), auxquelles 802.11a a ajouté OFDM :

FHSS (Frequency Hoping Spread Spectrum) [36]

La plupart des interférences nuisibles aux transmissions radio n'agissent en fait que sur des bandes de fréquence assez étroites. Si par malchance, de telles interférences ont lieu au moment où l'on transmet, alors notre signal sera fortement dégradé. Une technique pour protéger notre signal consiste à régulièrement changer de fréquence (**FIG I.7**). Bien sûr les paquets envoyés sur la bande perturbée seront affectés, mais ils ne représenteront plus qu'une minorité des transmissions et leur retransmission sera moins coûteuse. L'émetteur et le récepteur doivent connaître à l'avance le séquencement des sauts de fréquence, mais des informations portées par les paquets permettent à un mobile s'attachant au réseau de savoir à partir d'un paquet qu'il reçoit où en est le déroulement de la séquence.

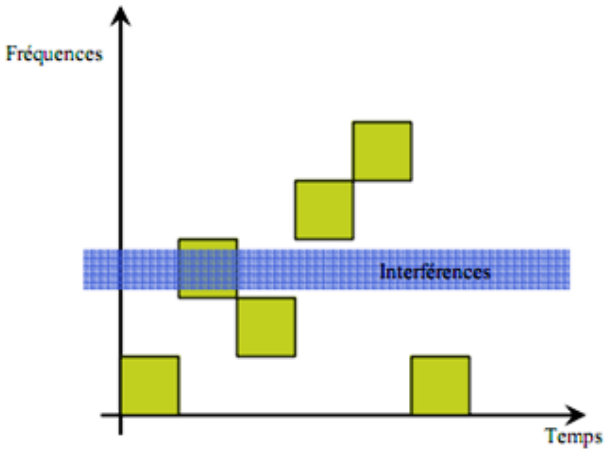


FIG I. 7 : Changer de fréquence régulièrement pour réduire l'impact de certaine interférence

DSSS (Direct Sequence Spread Spectrum) [36]

Toujours pour lutter contre les interférences importantes mais n'affectant que des plages de fréquences assez étroites, il existe la technique de l'étalement de spectre. Des manipulations sur le signal vont le faire occuper un spectre plus large. A la réception, une manipulation inverse est effectuée (figure 1.7). Cette technique est moins sensible aux interférences dues aux fréquences parasites à faible largeur spectrale.

IR : Infra Red

Deux technologies de réseaux infrarouges existent : les réseaux à ondes infrarouges directes et les réseaux à ondes infrarouges diffusées. La communication s'effectue en plaçant les entités communicantes sur une même ligne de vue et à une distance maximale de 2 m. Les échanges se font entre 1 et 4 Mb/s. Les réseaux à ondes infrarouges diffusés offrent toujours une bande passante de 4 Mb/s, mais une antenne à ondes infrarouges diffusés couvre une zone qui peut atteindre jusqu'à 100 m2.

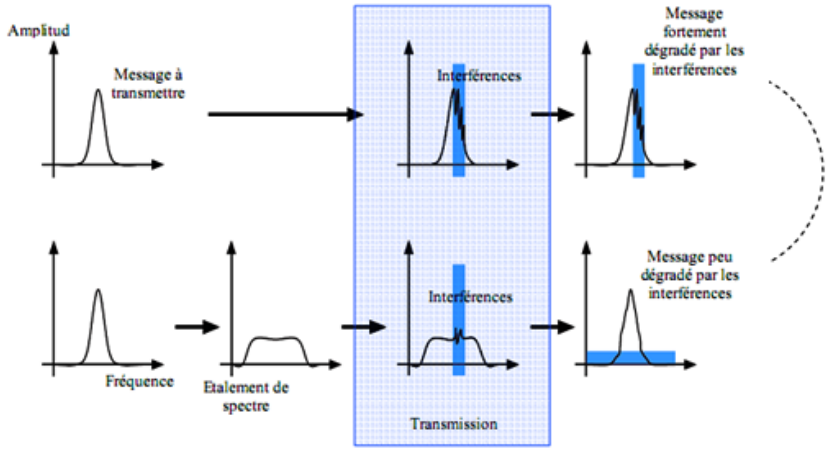


FIG I. 8 : Etaler le spectre pour réduire l'impact de certaines interférences

OFDM (Orthogonal Frequency Division Multiplexing) [36]

L'idée de base de l'OFDM réside dans le fait de répartir un canal binaire haut-débit en une multitude de canaux, lesquels étant modulés à bas-débits. Chacun de ces sous-canaux est modulé par une fréquence différente, l'espacement entre chaque fréquence restant constant. Ces fréquences constituent une base orthogonale et le spectre du signal OFDM présente une occupation optimale de la bande allouée.

On peut dire pour résumer rapidement cette méthode, qu'en présence de chemins multiples, à débit total équivalent, l'agrégation d'un certain nombre de canaux lents donne de meilleurs résultats qu'un seul canal très rapide.

I.5.4.2. La couche MAC

L'IEEE 802.11 définit deux modes de fonctionnement de la sous-couche MAC : le mode PCF (Point Coordination Function) qui utilise la station de base pour contrôler l'activité de la cellule et le mode DCF (Distributed Coordinated Function) qui n'utilise aucune entité de gestion centralisée pour communiquer. Dans les réseaux Ad hoc multi-sauts, il n'y a pas de stations de base fixes et c'est donc le mode DCF qui sera employé, le mode d'accès PCF n'étant pas applicable dans ces réseaux.

Description du mode DCF

Dans le monde filaire, lorsqu'un émetteur envoie un signal sur le câble, il peut y lire en même temps la valeur qui y est effectivement présente. Si jamais la valeur lue est différente de celle que l'émetteur écrit, c'est qu'un autre émetteur est actif au même moment et qu'il y a collision.

Cette écoute du signal sur le câble au moment de l'émission est à la base de la méthode d'accès CSMA/CD (Carrier Sense Multiple Access / Collision detection) bien connue d'Ethernet. CSMA/CD permet de détecter une collision et le cas échéant de retransmettre le paquet après un temps d'attente aléatoire. A la différence de la couche MAC de 802.11 qui utilise des acquittements pour détecter ces collisions et permettre la retransmission des paquets qui ont été perdus (en l'absence d'acquittement, l'émetteur sait qu'il doit retransmettre).

Avec Ethernet, l'idée est d'observer l'état du canal avant d'émettre. Si le canal est libre, alors nous pouvons envoyer notre trame (et si à ce moment-là nous détectons une collision, nous réémettons la trame un peu plus tard, après une attente de durée aléatoire).

Mais avec 802.11 si plusieurs mobiles étaient en attente d'émission, ils détecteraient tous le canal libre et émettraient au même moment. Il y aurait collision au récepteur et il faudrait attendre que le délai imparti pour le retour de l'acquittement soit écoulé pour s'en rendre compte, ceci pourrait être relativement long.

L'idée retenue pour 802.11 est donc, lorsque le canal devient libre, d'attendre une période de durée aléatoire supplémentaire appelée backoff avant d'émettre. Ce mécanisme s'applique lorsque le canal devient libre aussi bien après une de nos propres émissions qu'après toute autre émission. Ainsi, si plusieurs mobiles veulent émettre, il y a peu de chances pour qu'ils aient choisi la même durée. Celui qui a choisi le plus petit backoff va commencer à émettre, et les autres vont alors se rendre compte qu'il y a à nouveau de l'activité sur le canal et vont attendre.

Lorsque le canal devient libre, avant toute chose, il faut qu'il le reste pour une période DIFS (DCF Inter-Frame Space). Si le canal est resté libre durant toute cette période, alors les mobiles qui veulent émettre tirent un backoff aléatoire exprimé en un

nombre de time slots d'une durée fixe. Une fois ce tirage effectué, tant que le canal reste libre, les mobiles décrémentent leur backoff. Dès que l'un d'eux a terminé, il émet. Les autres mobiles, dès qu'ils détectent le regain d'activité sur le canal stoppent la décrémentation de leurs backoff et entrent en période de defering **FIG 1.9.**

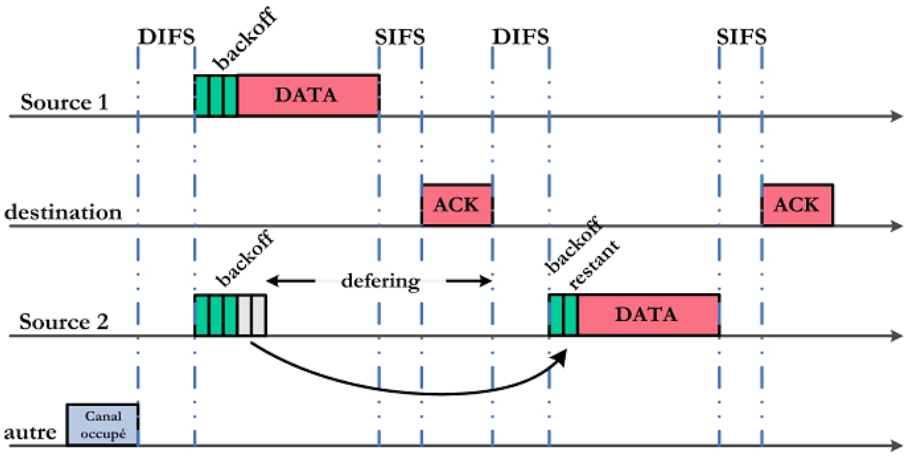


FIG I. 9 : Les mécanismes du backoff et du defering

Il faut noter que le temps de pause qui sépare un paquet de données de son acquittement est appelé SIFS (Short Inter-Frame Space) et qu'il est plus court que DIFS. Le mobile en période de defering ne pourra reprendre la décrémentation de son backoff que si le canal est à nouveau libre pendant DIFS. Le fait que SIFS soit plus court empêche que la décrémentation ne reprenne de manière inopportune entre les données et leur acquittement.

Lorsque les données du mobile "source1" ont été acquittées et que DIFS s'est écoulé sans activité sur le canal, "Source2" peut reprendre la décrémentation de son backoff (qui est déjà à 2 unités). Ici, aucun autre mobile ne vient l'empêcher de terminer et il peut donc finalement envoyer ses données.

Le mécanisme de backoff limite les risques de collision mais ne les supprime pas complètement. Aussi, si une collision se produit (détectée grâce à l'absence d'acquittement), un nouveau backoff va être tiré au hasard. Mais à chaque collision consécutive, la taille de la fenêtre va doubler afin de diminuer les chances que de telles collisions se répètent. La borne inférieure de la Contention Window est toujours zéro, et la borne supérieure va évoluer entre les valeurs aCWmin et aCWmax définies par la

norme. La borne supérieure de la fenêtre est ré-initialisée aCWmin sitôt qu'un paquet a été transmis correctement (ou lorsque les timers de ré-émission expirent) [IEE03]. Un exemple d'évolution de la fenêtre de contention est donné sur la **FIG I.10.**

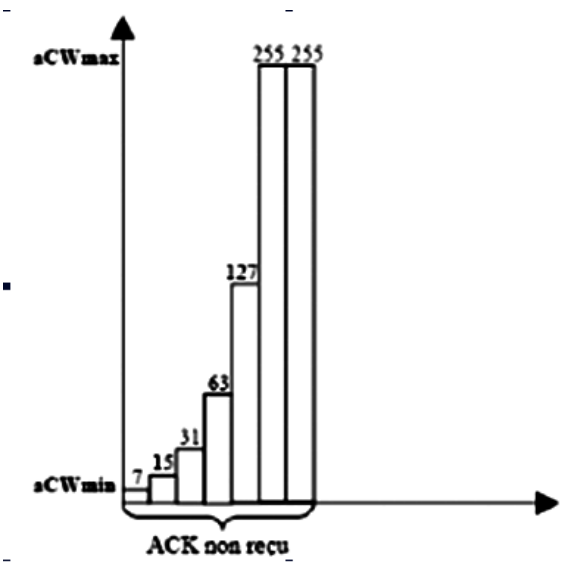


FIG I. 10 : Exemple de variation du backoff

I.5.4.3. RTS/CTS et le problème des nœuds cachés et des nœuds exposés

Le mécanisme CSMA/CA est insuffisant quand il s'agit du problème des nœuds cachés où du problème des nœuds exposés **FIG I.11**.

Problème des nœuds cachés

FIG I. 11 : Le problème du nœud caché

Problème des nœuds exposés

Considérons le cas présenté en **FIG I.12** où une station *B* initie une communication vers une station *A,* la station *C* écoute le canal radio, elle entend donc une communication en cours, car *C* est dans la zone de couverture de *B.* Dans ce cas, la station *C* déduit

qu'elle ne peut pas entamer une communication avec *D,* or si *C* transmettait, elle ne crée pas une collision dans les régions où *D* et *A* se situent.

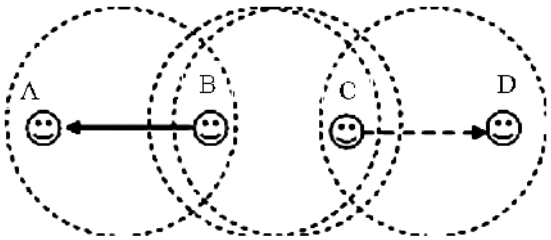


FIG I. 12 : Problème du nœud exposé.

Dans cette configuration, un émetteur détecte l'activité de l'autre, il croit que le canal est toujours occupé. Donc, il n'émettra pas ses données disponibles. Même s'il n'y aura pas de collisions au niveau d'un autre récepteur proche.

802.11 propose un mécanisme utilisant des paquets de contrôle appelés Request To Send (RTS) et Clear To Send (CTS). Un mobile qui veut émettre ne va plus directement envoyer son gros paquet de données, mais plutôt un petit paquet RTS pour lequel les chances de collision sont plus faibles. A ce paquet RTS, le destinataire va répondre par un petit paquet CTS qu'il diffuse à tout son voisinage comme illustré dans la **FIG I.13.**

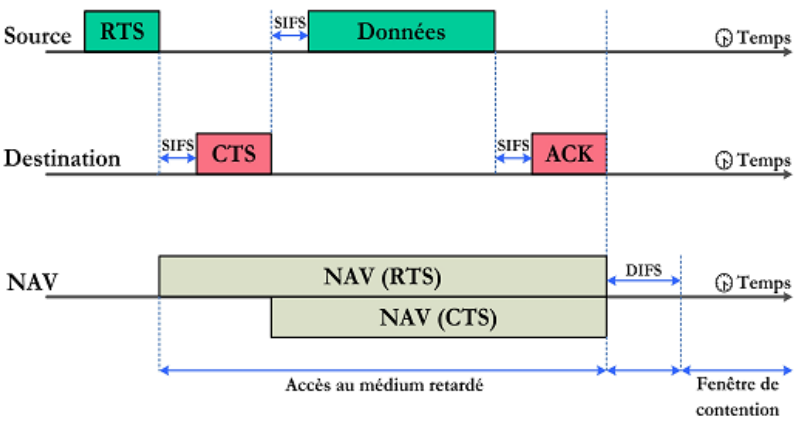


FIG I.13 : Le mécanisme RTS/CTS

Les paquets RTS et CTS contiennent des informations qui permettent de réserver le canal pour la durée de transmission des données qui vont suivre. Un mobile qui reçoit

un CTS alors qu'il n'a pas envoyé (ni même détecté de RTS) sait que quelqu'un d'autre va émettre et doit donc attendre. Le mobile qui a envoyé le RTS sait, quand il reçoit le CTS correspondant, que le canal a été réservé pour lui et qu'il peut émettre. Au niveau des mobiles, la réservation du canal est implémentée grâce au Network Allocation Vector (NAV). Dans chaque nœud, le NAV indique pour combien de temps le canal est utilisé par quelqu'un d'autre.

I.5.4.4. Différentes dérivées de la norme 802.11

Dans le but de palier certaines lacunes des réseaux sans fil et plus particulièrement du standard 802.11b, plusieurs nouvelles extensions ont été proposées :

- 802.11a : appelé également WiFi5, cette norme spécifie 8 canaux radio dans la bande de fréquence des 5 GHz. Elle permet d'obtenir un débit théorique de 54 Mbps (30 Mbps réels) ;
- 802.11b : est considérée comme la première norme sans fil exploitée par le grand public et les professionnels. Cette norme offre un débit théorique de 11 Mbps (6 Mbps réels) dans la bande des 2.4 GHz, avec une portée importante pouvant atteindre jusqu'à 300 mètres dans un environnement dégagé ;
- 802.11c : cette norme ne représente aucun intérêt pour le grand public, car elle représente uniquement une modification de la norme 802.11d afin de pouvoir établir un pont avec les trames 802.11 au niveau liaison de données ;
- 802.11d : qui représente un supplément à la norme 802.11 dont le but est de permettre aux différents points d'accès d'échanger des informations sur des plages de fréquences et des puissances selon les restrictions réglementaires autorisées dans différents pays ;
- 802.11e : qui vise à donner des possibilités en matière de qualité de service au niveau de la couche liaison de données pour les applications multimédia et temps réels.
- 802.11f : cette norme a été définie afin de permettre une meilleure utilisation d'infrastructures multi-vendeur. En proposant un protocole appelé IAPRP (pour Inter Access Point Roaming Protocol), 802.11f permet à un utilisateur itinérant de changer de point d'accès de façon transparente lors d'un déplacement dans l'infrastructure réseau.
- 802.11g : cette norme qui est plus performante (au moins en terme de débit et de sécurité) est directement compatible avec 802.11b et utilise une modulation OFDM

(pour Orthogonal Frequency Division Multiplexing). Elle offre un haut débit théorique de 54 Mbps (30 Mbps réels) sur la bande de fréquence des 2.4 GHz.

- 802.11h : le but de cette norme est de respecter l'utilisation des systèmes WLANs dans les pays européens dans la bande 5 GHz (HiperLAN 2 [ETSI TS-101 761-1. Technical Specification Broadband Radio Access Networks HIPERLAN Type 2. Avril 2000. Data Link Control Layer Part 1 : Basic DataTransport Functions.], d'où ce "h" de 802.11h) pour être en conformité avec la réglementation européenne en matière de fréquence et d'économie d'énergie.
- 802.11i : qui a été définie afin de remédier au problème de la sécurité des transmissions (gestion et distribution des clés, chiffrement et authentification). Cette norme s'appuie sur l'AES (Advanced Encryption Standard) et s'applique aux technologies 802.11a, 802.11b et 802.11g.
- 802.11j : dont la spécification a été proposée dans le but d'incorporer la réglementation japonaise, au même titre que 802.11h par rapport à la réglementation européenne.
- 802.11k : cette technologie qui utilise des signaux infrarouges, a pour caractéristique principale d'utiliser une onde lumineuse pour la transmission de données. En effet, le caractère non dissipatif des ondes lumineuses permet d'offrir un niveau de sécurité plus élevé.
- 802.11n : cette nouvelle norme a été ratifiée à l'unanimité au sein de l'IEEE (Institute of Electrical Electronic Engineers) au début 2006. Elle sera la première norme 802.11 à implémenter la technologie MIMO (Multiple Input Multiple Output). 802.11n devrait être à priori dix fois plus rapide que la norme 802.11g puisque les débits théoriques annoncés sont de plus de 500 Mbps, tout en restant compatible avec les normes 802.11b et 802.11g.

I.6. Le problème de routage dans les réseaux Ad hoc

I.6.1. Définition du routage

Le routage est une méthode d'acheminement des informations à la bonne destination à travers un réseau de connexion donné. Le problème de routage consiste à déterminer un acheminement optimal des paquets à travers le réseau au sens d'un certain critère de performance. Le problème consiste à trouver l'investissement de moindre coût

en capacités nominales et de réserves qui assure le routage du trafic nominal et garantit sa survivabilité en cas de n'importe quelle panne d'arc ou de nœud.

I.6.2. La difficulté du routage dans les réseaux Ad hoc

De fait qu'un réseau Ad hoc est un ensemble de nœuds mobiles qui sont dynamiquement et arbitrairement éparpillés d'une manière ou l'interconnexion entre les nœuds peut changer à tout moment, il se peut qu'un hôte destination soit hors de la portée de communication d'un hôte source, ce qui nécessite l'emploi d'un routage interne par les nœuds intermédiaires afin de faire acheminer les paquets de message à la bonne destination.

En effet, la topologie évoluant constamment en fonction des mouvements des mobiles, le problème qui se pose dans le contexte des réseaux Ad hoc est l'adaptation de la méthode d'acheminement utilisée avec le grand nombre d'unités existant dans un environnement caractérisé par de modestes capacités de calcul et de sauvegarde.

D'ailleurs, dans la pratique, il est impossible qu'un hôte puisse garder les informations de routage concernant tous les autres nœuds, dans le cas où le réseau serait volumineux.

I.6.3. Les contraintes de routage dans les réseaux Ad hoc

L'étude et la mise en œuvre d'algorithmes de routage pour assurer la connexion des réseaux Ad hoc au sens classique du terme (tout sommet peut atteindre tout autre), est un problème complexe. L'environnement est dynamique et évolue donc au cours du temps, la topologie du réseau peut changer fréquemment. Il semble donc important que toute conception de protocole de routage doive étudier les problèmes suivants :

1. La minimisation de la charge du réseau
2. Offrir un support pour pouvoir effectuer des communications multi-points fiables
3. Assurer un routage optimal
4. Offrir une bonne qualité concernant le temps de latence

I.7. Les protocoles de routage pour les réseaux Ad hoc

I.7.1. Modes de communication dans les réseaux Ad hoc

Avant de parler des protocoles de routage proprement dit, nous allons rappeler quels sont les principaux modes de communication dans les réseaux et particulièrement dans les réseaux Ad hoc :

- ✓ la communication point à point ou unicast, pour laquelle il y a une source et une seule destination ;
- ✓ la communication multipoint ou multicast, qui permet d'envoyer un message à plusieurs destinataires ;
- ✓ la diffusion ou broadcast, qui envoie un message à tous les nœuds du réseau. Ces trois modes de communication sont schématisés par la **FIG I.14**.

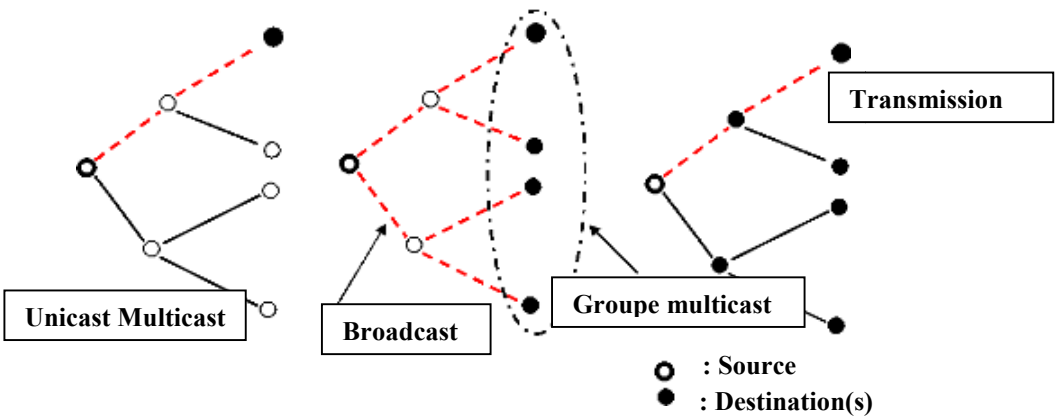


FIG I. 14 : Modes de communication dans les réseaux mobiles

I.8. Classification des protocoles de routage

Vue la difficulté de routage dans les réseaux Ad hoc, les stratégies existantes utilisent une variété de techniques afin de résoudre ce problème. Suivant ces techniques, plusieurs classifications sont apparues, parmi lesquelles (Voir dans la **FIG 1.15**).

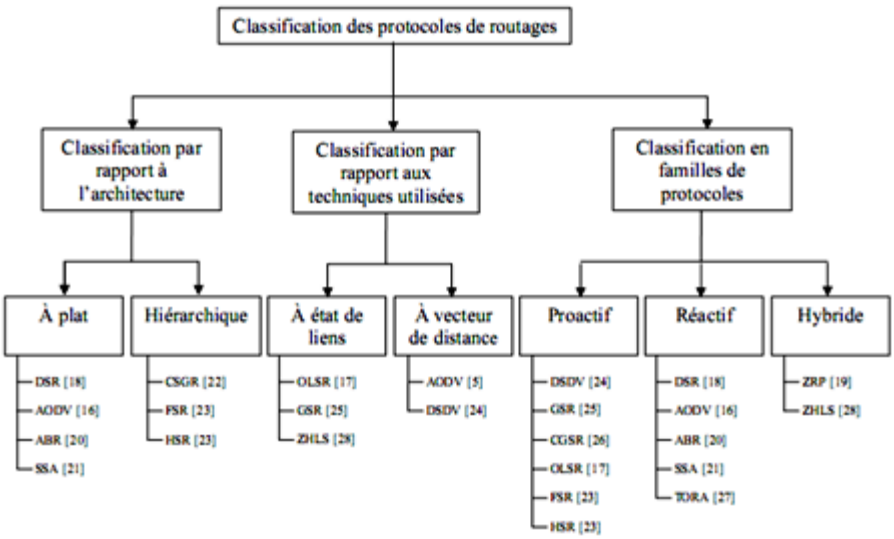


FIG I. 15 : Classification des différents protocoles de routages de réseaux Ad hoc

I.8.1. Par rapport à l'architecture

Le premier critère utilisé pour classifier les protocoles de routage dans les réseaux Ad hoc concerne le type de vision qu'ils ont du réseau et les rôles qu'ils accordent aux différents mobiles.

I.8.1.1. Les protocoles de routage à plat

Considèrent que tous les nœuds sont égaux (**FIG I.16**). La décision d'un nœud de router des paquets pour un autre dépendra de sa position. Parmis les protocoles utilisant cette technique, on cite l'AODV (Ad hoc On Demand Distance Vector).

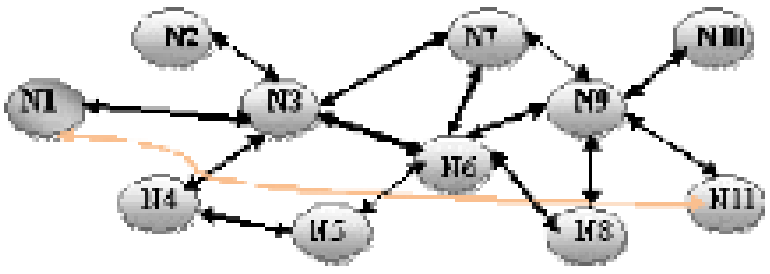


FIG I. 16 : Routage à plat

I.8.1.2. Les protocoles de routage hiérarchique

Fonctionnent en confiant aux mobiles des rôles qui varient de l'un à l'autre. Certains nœuds sont élus et assument des fonctions particulières qui conduisent à une

vision en plusieurs niveaux de la topologie du réseau. Par exemple, un mobile pourra servir de passerelle pour un certain nombre de nœuds qui se seront attachés à lui. Le routage en sera simplifié, puisqu'il se fera de passerelle à passerelle, jusqu'à celle directement attachée au destinataire. Un exemple donné sur la figure (**FIG I.17**), où le nœud N3 passe par les passerelles P1, P2 et P3 pour atteindre N7. Dans ce type de protocole, les passerelles supportent la partie majeure de la charge du routage (les mobiles qui s'y rattachent savent que si le destinataire n'est pas dans leur voisinage direct, il suffit d'envoyer à la passerelle qui se débrouillera) [26]. Un exemple de protocole utilisant cette stratégie est l'OLSR (Optimized Link State Routing)

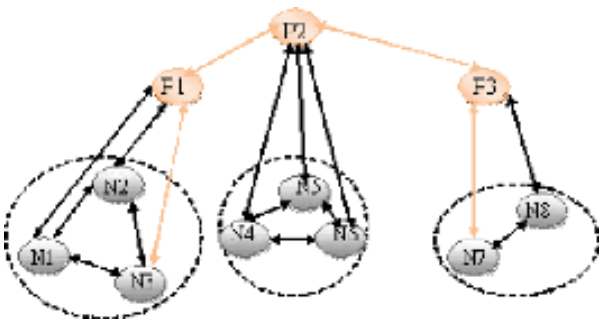


FIG I. 17 : Routage Hiérarchique

I.8.2. Par rapport à la technique utilisée

On distingue deux familles de protocole de routage hérités du monde filaire: les protocoles basés sur l'état des liens et ceux basés sur le vecteur de distance. Les deux méthodes exigent une mise à jour périodique des données de routage qui doivent être diffusées par les différents nœuds de routage du réseau. Les algorithmes de routage basés sur ces deux méthodes utilisent la même technique qui est la technique des plus courts chemins, et permettent à un hôte donné, de trouver le prochain hôte pour atteindre la destination en utilisant le trajet le plus court existant dans le réseau.

I.8.2.1. Les protocoles basés sur l'état de lien

La famille des protocoles à état de liens se base sur les informations rassemblées sur l'état des liens dans le réseau. Ces informations sont disséminées dans le réseau périodiquement ce qui permet ainsi aux nœuds de construire une carte complète du réseau. Un nœud qui reçoit les informations concernant l'état des liens, met à jour sa vision de la topologie du réseau et applique un algorithme de calcul des chemins

optimaux afin de choisir le nœud suivant pour une destination donnée. En général, ces algorithmes se basent sur le principe de l'algorithme de Djikstra [27] pour calculer les chemins les plus courts entre un nœud source et les autres nœuds du réseau. Les principaux protocoles de routage dans les réseaux Ad hoc qui appartiennent à cette classe sont les suivants : TORA, OLSR et TBRPF.

I.8.2.2. Les protocoles basés sur le vecteur de distance :

Les protocoles à vecteur de distance se basent sur un échange, entre voisins, des informations de distances des destinations connues. Chaque nœud envoie à ses voisins la liste des destinations qui lui sont accessibles et le coût correspondant. Le nœud récepteur met à jour sa liste locale des destinations avec les coûts minimums. Le processus de calcul se répète, s'il y a un changement de la distance minimale séparant deux nœuds, et cela jusqu'à ce que le réseau atteigne un état stable. Les calculs des routes se baser sur le principe de l'algorithme distribué de Bellman-Ford [9] (DBF). Les protocoles de routage basés sur le vecteur de distance les plus connus pour les réseaux Ad hoc sont : DSR, DSDV et AODV.

I.8.3. Classification par famille

Suivant la manière de création et de maintenance de routes lors de l'acheminement des données, les protocoles de routage peuvent être séparés en : Proactif, Réactif et Hybride.

I.8.3.1. Les protocoles de routage proactifs

Les protocoles de routage proactifs essaient de maintenir les meilleurs chemins existants vers toutes les destinations possibles (qui peuvent représenter l'ensemble de tous les nœuds du réseau) au niveau de chaque nœud du réseau. Les routes sont sauvegardées même si elles ne sont pas utilisées. La sauvegarde permanente des chemins de routage est assurée par un échange continu des messages de mise à jour des chemins, Le plus abouti de ces protocoles est OLSR.

I.8.3.2. Avantages et inconvénients des protocoles proactifs

Avec un protocole proactif, les routes sont disponibles immédiatement. Ainsi, l'avantage d'un tel protocole est le gain de temps lors d'une demande de route. Le problème est que les changement de routes peuvent être plus fréquents que la demande de

la route et le trafic induit par les messages de contrôle et de mise à jour des tables de routage peut être important et partiellement inutile, ce qui gaspille la capacité du réseau sans fil. De plus, la taille des tables de routage croit linéairement en fonction du nombre de nœud.

De ce fait, un nouvel type de protocole est apparu. Il s'agit des protocoles de routage réactifs.

I.8.4. Les protocoles de routage réactifs

Les protocoles de routage réactifs (dits aussi: protocoles de routage à la demande), représentent les protocoles les plus récents proposés dans le but d'assurer le service du routage dans les réseaux sans fils.

La majorité des solutions proposées pour résoudre le problème de routage dans les réseaux Ad hoc, et qui sont évaluées actuellement par le groupe de travail MANET (Mobile Ad hoc Networking working Groupe) de l'IETF (Internet Engineering Task Force), appartiennent à cette classe de protocoles de routage [54].

Les protocoles de routage appartenant à cette catégorie créent et maintiennent les routes selon les besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte globale de routes est lancée, et cela dans le but d'obtenir une information. Actuellement, le plus connu de ces protocoles est AODV.

I.8.4.1. Les avantages et les inconvénients des protocoles réactifs

A l'opposé des protocoles proactifs, dans le cas d'un protocole réactif, aucun message de contrôle ne charge le réseau pour des routes inutilisées, ce qui permet de ne pas gaspiller les ressources du réseau. Mais la mise en place d'une route par inondation peut être coûteuse et provoquer des délais importants avant l'ouverture de la route et les retards dépassent bien souvent les délais moyens admis par les logiciels, aboutissant à une impossibilité de se connecter alors que le destinataire est bien là.

De ce fait, un nouveau type de protocole est apparu. Il s'agit des protocoles de routage hybrides.

I.8.5. Les protocoles de routage hybrides [69]

Dans ce type de protocole, on peut garder la connaissance locale de la topologie jusqu'à un nombre prédéfini- a priori petit- de sauts par un échange périodique de trame de contrôle, autrement dit par une technique proactive. Les routes vers des nœuds plus lointains sont obtenues par schéma réactif, c'est-à-dire par l'utilisation de paquets de requête en diffusion. Un exemple de protocoles appartenant à cette famille est DSR (Dynamic Source Routing), qui est réactif à la base mais qui peut être optimisé s'il adopte un comportement proactif. Un autre exemple est le protocole ZRP (Zone Routinier Protocol).

I.9. Présentation de quelques protocoles de routage de réseaux Ad hoc

Dans ce qui suit, nous allons présenter quelques protocoles de routage des réseaux mobiles Ad hoc, à savoir AODV, OLSR, DSR et ZRP.

I.9.1. (AODV) Ad hoc On-demand Distance Vector [25]

Ad hoc On-Demand Distance-Vector Protocol (AODV) est un protocole réactif basé sur le principe des protocoles de routage à vecteur de distance. Il est pour l'essentiel une combinaison de DSDV et de DSR. Il emprunte, à DSR ses mécanismes de découverte et de maintenance des routes « Route Discovery » et « Route Maintenance » ; à DSDV, son routage « saut par saut », les numéros de séquences ainsi que la diffusion des mises à jour des tables de routage. Ce protocole fait l'objet d'un grand nombre de recherches dû à toutes ses caractéristiques; il est en effet l'un des principaux protocoles au sein du groupe de recherche MANET.

Comme le fait DSR, AODV utilise une requête de route dans le but de créer un chemin vers une certaine destination. Cependant, AODV maintient les chemins d'une façon distribuée en gardant une table de routage au niveau de chaque nœud de transit appartenant au chemin cherché. Une entrée de la table de routage contient essentiellement

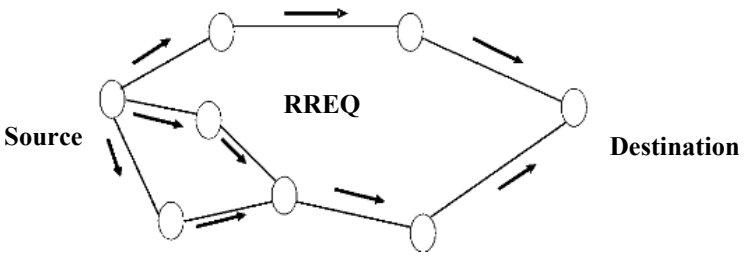
- ✓ L'adresse de la destination.
- ✓ Le nœud suivant.
- ✓ La distance en nombre de nœud (i.e. le nombre de nœud nécessaire pour atteindre la destination).
- ✓ Le numéro de séquence destination.

- ✓ Le temps d'expiration de l'entrée de la table.

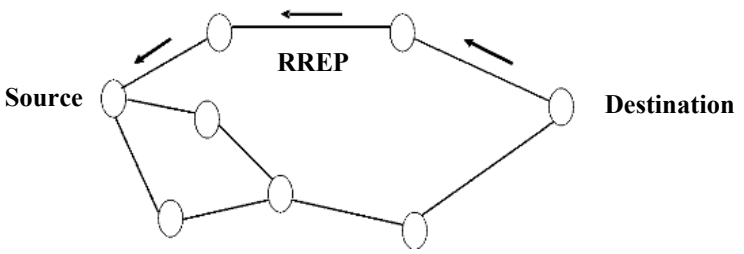
Quand un nœud de transit envoie le paquet de la requête à un voisin, il sauvegarde aussi l'identificateur du nœud à partir duquel la première copie de la requête est reçue. Cette information est utilisée pour construire le chemin inverse **(FIG I.18 (b))**. Qui sera traversé par le paquet réponse de route (cela induit que l'AODV ne supporte que les liens symétriques). Puisque le paquet réponse de route va être envoyé à la source, les nœuds appartenant au chemin de retour vont modifier leurs tables de routage suivant le chemin contenu dans le paquet de réponse.

Un nœud diffuse une requête de route (RREQ : *Route REQuest)* dans le cas où il aurait besoin de connaître une route vers une certaine destination et qu'une telle route ne serait pas disponible **(FIG I. 18 (a))**. Cela peut arriver si la destination n'est pas connue au préalable, ou si le chemin existant vers la destination est devenu défaillant (i.e. la métrique qui lui est associée est infinie). Le champ numéro de séquence destination du paquet RREQ contient la dernière valeur connue du numéro de séquence associé au nœud destination, cette valeur est recopiée de la table de routage.

Si le numéro de séquence n'est pas connu, la valeur nulle sera prise par défaut. Le numéro de séquence source du paquet RREQ contient la valeur du numéro de séquence du nœud source. Comme nous l'avons déjà dit, après la diffusion du RREQ, la source attend le paquet réponse de route (RREP : *Route REPly).* Si ce dernier n'est pas reçu pendant une certaine période (appelée *RREP_ WAIT_TIME),* la source peut rediffuser une nouvelle requête RREQ. A chaque nouvelle diffusion, le champ *Broadcast* ID du paquet RREQ est incrémenté. Si la requête RREQ est rediffusée un certain nombre de fois (*RREQ_RETRIES)* sans la réception de réponse, un message d'erreur est délivré.



(a) La propagation du paquet RREQ



(b) Le chemin pris par RREP

FIG I. 18 : Les deux requêtes RREQ et RREP utilisées dans le protocole AODV.

Afin de maintenir des routes consistantes, une transmission périodique du message "*HELLO"* est effectuée. Si trois messages "*HELLO"* ne sont pas reçus consécutivement à partir d'un nœud voisin, le lien en question est considéré défaillant. Les défaillances des liens sont, généralement, dues à la mobilité des nœuds dans le réseau. Les mouvements des nœuds qui ne participent pas dans le chemin actif n'affectent pas la consistance des données de routage. Quand un lien reliant un nœud *p* à un nœud qui le suit dans le chemin de routage devient défaillant, le nœud *p* diffuse un paquet *UNSOLICITED RREP,* avec une valeur de numéro de séquence égale à l'ancienne valeur du paquet *RREP* incrémentée de un, et une valeur infinie de la distance. Le paquet *UNSOLICITED RREP* est diffusé aux voisins actifs jusqu'à ce qu'il arrive à la source. Une fois le paquet est reçu, la source peut initier le processus de la découverte de routes.

L'AODV maintient les adresses des voisins à travers lesquels les paquets destinés à un certain nœud arrivent. Un voisin est considéré actif, pour une destination donnée, s'il délivre au moins un paquet de donnée sans dépasser une certaine période (*appelée active timeout period).* Une entrée de la table de routage est active si elle est utilisée par un voisin actif. Le chemin reliant la source à la destination en passant par les entrées actives des tables de routage est dit un chemin actif. Dans le cas de défaillances de liens, toutes les entrées des tables de routage participant au chemin actif et qui sont concernées par la défaillance sont supprimées. Cela est accompli par la diffusion d'un message d'erreur entre les nœuds actifs.

Le protocole de routage AODV (tout comme le protocole DSR), n'assure pas l'utilisation du meilleur chemin existant entre la source et la destination. Cependant, des évaluations de performances récentes ont montré qu'il n'y a pas de grandes différences (en terme d'optimisation) entre les chemins utilisés par le protocole AODV et celles utilisées par les protocoles basés sur les algorithmes de recherche des plus courts

chemins. De plus, le protocole AODV ne présente pas de boucle de routage et évite le problème du comptage à l'infini "*counting to infinity"* de *Bellman-Ford,* ce qui offre une convergence rapide quand la topologie du réseau est dynamique.

I.9.2. (OLSR) Optimized Link State Routing [35]

OLSR est un protocole proactif et comme son nom l'indique, c'est un protocole à état de lien optimisé; il obtient aussi des routes de plus court chemin. Alors que dans un protocole à état de lien, chaque nœud déclare ses liens directs avec ses voisins à tout le réseau, dans le cas d'OLSR, les nœuds ne déclarent qu'une sous-partie de leur voisinage grâce à la technique des relais multipoints. En un nœud donné, cette technique consiste essentiellement à ignorer un ensemble de liens et de voisins directs, et de ne garder qu'un sous-ensemble des ces voisins considérés comme pertinent. Il est choisi de façon à pouvoir atteindre tout le voisinage à deux sauts (tous les voisins des voisins), cet ensemble est appelé l'ensemble des relais multipoints MPR (Multipoint Relay). Ces relais multipoints sont utilisés pour diminuer le trafic dû à la diffusion des messages de contrôle dans le réseau en appliquant la règle suivante: un nœud retransmet un message si et seulement si il ne l'avait pas déjà reçu, et il vient de le recevoir d'un nœud dont il est un relais multipoint. Pour maintenir à jour toutes les informations nécessaires au choix des relais multipoints et le calcul de la table de routage, les nœuds OLSR ont besoin de s'échanger périodiquement deux type, de message :

- ✓ Le message dit HELLO diffusé par chaque nœud, contenant la liste de ces voisins. Ces messages permettent à chacun de choisir son ensemble de relais multipoints.
- ✓ Le deuxième type est le message TC (Topology Control) diffusé périodiquement par chaque nœud et retransmet uniquement par les MPRs pour atteindre tout le réseau. Ce message contient la liste de ces voisins qui l'ont choisi comme un MPR.

I.9.3. (DSR) Dynamic Source Routing [40]

DSR est un protocole de routage réactif qui utilise une technique de routage par source dans laquelle les nœuds ne doivent pas nécessairement garder la trace de la route. Chaque paquet contient dans son en-tête la liste complète des adresses des nœuds à traverser vers la destination. Lors de la création des routes si un destinataire est dans le cache d'un nœud source, la route connue est utilisée. Sinon, une procédure de découverte de route est déclenchée par un message "Route Request". Le chemin vers la destination

est créé dans le paquet de recherche de route. Chaque nœud qui reçoit ce paquet ajoute à la route préexistante dans ce paquet sa propre adresse.

En cas de rupture d'un lien sur un trajet, le nœud situé en amont de cette rupture envoie à la source une indication concernant la rupture de lien. Les nœuds peuvent sur une base volontaire, garder dans leur cache les routes créées par les procédures de recherche de route. Ces routes permettent de répondre plus rapidement à d'autres requêtes de création de route. En cas de rupture de lien, il est possible à un nœud qui possède une route valide vers la destination de détourner par ce nouveau chemin, sauvant ainsi l'acheminement des paquets.

I.9.4. (ZRP) Zone Routing Protocol [33]

ZRP est un exemple de protocole de routage hybride (réactif/proactif). D'une part, il limite la portée de la procédure proactive seulement au voisinage local du nœud, et d'une autre part, la recherche dans tout le réseau, bien qu'elle soit globale, elle est exécutée de façon efficace en faisant intervenir une partie des nœuds de réseau, par opposition à interroger tout le réseau. ZRP divise le réseau en plusieurs zones de routage où une zone Z(k;n) pour un nœud n avec un rayon k, est définie comme l'ensemble des nœuds à une distance inférieure ou égale à k sauts de ce même nœud n. ZRP utilise deux protocoles totalement indépendants l'un à l'intérieure des zone, et l'autre entre ces zones. Le premier protocole est l'Intrazone Routing Protocol (IARP), qui opère à l'intérieur des zones. On peut avoir plusieurs protocoles proactifs, où les différentes zones peuvent fonctionner avec IARPs différent. Le deuxième protocole est l'Interzone Routing Protocol (IERP), qui est un protocole réactif utilisé pour le routage entre les zones. Il est utile uniquement quand le nœud destinataire ne se trouve pas dans la zone de routage de nœud source.

Ce découpage du réseau offre plusieurs avantages, notamment le passage à l'échelle. Par exemple, à la réception d'une requête réactive, le nœud est capable d'indiquer immédiatement si cette destination recherchée appartient ou non à sa zone, et pourrait ainsi l'aiguillées vers d'autres zones sans avoir à déranger ses voisins. Hélas, ce protocole présente l'inconvénient de l'absence de coordination entre les nœuds, il en résulte que les zones se chevauchent et un nœud peut être à la fois membre d'une zone et nœud frontière de plusieurs zones.

I.10. Conclusion

Dans cette partie, un état de l'art sur les réseaux Ad hoc avec ses différentes caractéristiques a été présenté, ainsi que les différentes normes des réseaux locaux sans fil, telle que le Bluetooth, la 802.11, HiperLAN 1 et 2. Parmi ces normes, nous avons prêté une attention particulière à la norme 802.11 établie par l'IEEE qui fonctionne en deux modes, le mode infrastructure le PCF (Point Coordination Function) adapté aux réseaux mobiles cellulaires, et le mode appelé Ad hoc, le DCF (Distributed Coordinated Function), qui est très adapté aux réseaux mobiles Ad hoc. Ensuite, nous avons exposé les différentes approches de routage des réseaux mobiles Ad hoc et quelques protocoles de routage qui leur sont dédiés. Le chapitre suivant présente un état de l'art sur les caches. Il commence par une classification des caches selon deux critères, leurs localisations et leurs fonctions, et se termine par une présentation du fonctionnement des caches, en décrivant un certain nombre de concepts liés aux caches, ainsi que les métriques de performance des caches et l'apport d'utilisation de cache dans les réseaux mobiles Ad hoc.

Chapitre 2

Les caches et les réseaux mobiles Ad hoc

II.1. Introduction

Le cache est une mémoire intermédiaire située entre le processeur et la mémoire centrale et dont le temps d'accès est inférieur à celui de la mémoire centrale. Un défaut de cache (un *miss* en anglais) peut se produire lorsqu'un document référencé n'est pas présent dans le cache, autrement c'est un succès (un *hit* en anglais).

Pour qu'un cache soit bénéfique, il faut que le coût de stockage d'une information soit plus petit que le coût de sa recherche depuis son serveur d'origine. Elle améliore notablement les temps d'exécution moyens des programmes. En mémorisant les informations les plus référencées par un programme, donc susceptibles d'être référencées à nouveau dans un futur proche.

Le mécanisme d'utilisation de cache a réussi à trouver sa place dans la plupart des systèmes et réseaux informatiques. Les processeurs d'ordinateur utilisent deux types de cache, les caches de données et les caches d'instructions. Les systèmes d'exploitation ont des caches (les buffers) pour les unités de disques. Les systèmes de fichiers distribués tels que le NFS et l'AFS se fondent fortement sur les caches pour de meilleures performances. Les routeurs cachent les routes récemment utilisées. Les serveurs du DNS (Domain Name System) cachent les correspondances hostname et adresse.

Nous nous intéressons dans ce chapitre à l'utilisation des caches dans les réseaux mobiles Ad hoc. Comment adapter efficacement ce mécanisme dans ce système réseau mobile pour

améliorer ses performances, en tenant compte des contraintes spécifiques comme la mobilité, l'énergie limitée et la capacité restreinte de stockage.

II.2. Types des caches Web

Il existe trois types de caches Web : les caches client, les caches serveur et les caches proxy. En général, les proxys peuvent être localisés prés du serveur (dans ce cas, ils agissent comme des serveurs miroirs), ou prés des clients dans le but de réduire le temps de latence [50].

II.2.1. Le cache client

Est disponible au niveau de la machine du client sur son browser Web.

Actuellement, les browsers Web offrent différentes options, comme définir la taille du cache de la mémoire vive et du disque où sont stockés les documents reçus. En utilisant ce cache, le client accède à une page sans faire de requête à un serveur si les fichiers correspondants sont stockés dans sa mémoire ou son disque.

II.2.2. Le cache serveur

Il offre les mêmes fonctionnalités au niveau du serveur : en disposant en mémoire vive les documents les plus fréquemment sollicités, il accélère l'accès à ses propres ressources.

II.2.3. Le cache proxy

Le cache proxy quant à lui est un dispositif commun à plusieurs clients permettant à ceux-ci d'optimiser leur accès au Web. La solution du proxy permet d'interposer une capacité de stockage entre le client et le serveur.

Le principe est simple, lorsqu'un client cherche à accéder aux ressources d'un serveur, il envoie une requête au proxy. Celui-ci cherche ce document dans son cache et le retransmet alors au client. Si le document demandé par le client n'existe pas dans le cache, le proxy envoie la requête au serveur lui-même et retransmet le document reçu au client en stockant au passage le document dans son cache s'il le juge intéressant.

II.3. La fonction des caches

Les caches peuvent être répartis en trois catégories de fonctionnement qui sont, les caches CPU (Central Processing Unite), les caches Web, et les caches mobiles.

II.3.1. Les caches CPU traditionnels

Comme la vitesse de la CPU est très grande, l'algorithme de remplacement de cache doit être très efficace. Dans ce type de fonctionnement, LRU (Least Recently Used) est très répandu, puisque son coût de traitement est très bas et de raisonnables performances sont obtenues [74], [49].

Les caractéristiques d'un cache CPU sont comme suit :

1. Toutes les données sont de taille égale.
2. Le cache est de très petite taille.
3. Les mises à jour des données sont effectuées dans le cache.
4. Les données sont souvent accédées séquentiellement.

II.3.2. Les caches Web

Contrairement à un cache CPU, la vitesse de communication est très lente pour les caches Web. Ainsi, pour ce dernier, nous pouvons utiliser des algorithmes plus compliqués, pour plus de performances [1], [49].

Les caractéristiques d'un cache Web sont comme suit :

1) Les tailles de données sont distribuées entre de très petites à de très grandes valeurs.

2) Nous pouvons utiliser des disques pour le cache. Ainsi, la taille de cache peut être assez grande.

3) La mise à jour est exécutée sur les pages Web, ainsi un certain contenu de cache peut facilement devenir incohérent.

4) Périodiquement, au cours d'une même journée, on peut avoir une grande variation de la charge d'utilisation. Par exemple, la charge d'utilisation à 9 heures du matin, peut

être très différente de celle de 5 heures de l'après midi, et même encore de celle de la nuit.

Le grand problème avec ce type de cache est les tailles des documents qui peuvent être très différentes [3], [65]. En outre, en raison des caractéristiques citées ci-dessus, nous devons ajuster les modèles de gestion de cache traditionnels. Par exemple, pour la politique LRU, en plus de la récence, nous devons considérer la taille des documents.

II.3.3. Les caches des systèmes mobiles

Les systèmes mobiles ont généralement des petits caches, et peu de données à traiter, à comparer avec les deux cas cités ci-dessus [3], [94]. Ces systèmes, à la différence des deux cas précédents, sont exposés à deux grands problèmes, qui sont : la mobilité, et l'énergie limitée des nœuds. En effet, les documents ayant été populaires dans une aire géographique donnée, et en raison de la mobilité, peuvent ne pas l'être dans une autre, ce qui crée la pollution de cache.

Le concepteur des algorithmes de remplacement de cache, pour les systèmes mobiles, doit prendre aussi en considération le facteur de consommation d'énergie, par exemple en faisant une estimation de coût en énergie nécessaire à la récupération d'un document à distance, si celui-ci doit être remplacé [70].

II.4. Fonctionnement de cache

Le cache est un espace mémoire et/ou disque sur une machine. Lorsque celle-ci reçoit une requête, le cache doit appliquer une stratégie pour savoir si le document doit être caché ou pas. En effet, il lui faut déterminer si le document peut servir à une future demande.

II.4.1. Support HTTP pour le cache

L'un des plus importants dispositifs de protocole HTTP utilisés dans le contexte des caches, est celui des requêtes conditionnelles, qui recherchent des documents, seulement si un certain nombre de conditions sont satisfaites. Les clients formulent les requêtes conditionnelles, en utilisant les métas informations du document en question. Les serveurs fournissent ces informations avec les documents dans *last-modified* et *ETag* de l'en-tête de leurs réponses. L'en-tête *Last-modified* contient la date et le temps de la dernière modification

du document. L'en-tête *ETag*, qui signifie *Entity* Tag, est un identifiant unique pour une instance particulière d'un document.

Un autre groupe d'entêtes est utilisé, pour indiquer quels documents peuvent être cachés, pour combien de temps ils peuvent être cachés, et s'ils peuvent être partagés entre différents clients, etc. En particulier, les en-têtes expires et âge qui sont utilisés par les caches pour expirer les documents cachés. L'en-tête complexe cache-control peut être utilisé que ce soit par les requêtes ou par les réponses pour contrôler le comportement du cache en différentes manières, peut-être la plus importante directive de Cache-control est No-cache. Elle permet à une requête d'interdire de donner une réponse à partir du cache, et elle permet à une réponse d'empêcher a n'importe quel cache de cacher la réponse.

II.4.2. Cachabilité des documents

L'objectif primaire d'un cache est de cacher certains documents, des réponses qu'il a reçues. Un document serait cachable, s'il peut être utilisé pour répondre à une future requête. Il différencie donc entre deux familles de documents, les documents cachables et les documents non cachables. Tous les documents statiques sont considérés comme cachables à l'inverse des documents dynamiques. Les contenus dynamiques sont des pages où les données fournies par le serveur changent dans le temps ou en fonction du client (pages personnalisées).

Par exemple, la réponse à un formulaire rempli par le client est personnalisée et unique, elle est non « cachable ». Un cache décide aussi si une réponse particulière est cachable en examinant différents composants de son en-tête. En particulier ce qui suit :

- Le code statut de la réponse
- La méthode de la requête
- Les directives de Cache-control
- Le validateur de la réponse
- Authentification de la requête

Ces différents facteurs agissent l'un sur l'autre d'une façon légèrement compliquée. Par exemple, quelques méthodes des requêtes sont non cachables à moins que ce soit permis par une directive de Cache-control. Quelques codes statut sont cachables par défaut, mais l'authentification et le Cache-control ont la priorité.

II.4.3. Défaut/succès de cache (cache hits/misses)

Quand un cache reçoit une requête pour un document, il vérifie si le document a été déjà caché. Si c'est non, on dit que la requête a provoqué un défaut de cache (cache miss). Un cache miss se produit pour les documents qui n'ont jamais été référencés précédemment, pour ceux qui ne sont pas cachables, ou pour ceux qui ont été supprimés pour faire de la place pour les nouveaux. Si le document est présent dans le cache, alors nous pourrions avoir un cache hit. Cependant, le cache doit d'abord décider si le document stocké est frais ou périmé. Un document caché est frais si son temps d'expiration n'a pas été écoulé ; autrement, il est périmé. Les documents frais sont donnés au client immédiatement, ils ne provoquent aucune latence et ne consomment aucune bande passante au serveur d'origine. Par contre, les documents périmés exigent la validation avec ce dernier.

II.5. Le problème de cohérence du cache

Les caches aident à réduire la latence et le nombre d'accès. Cependant, ils ont un effet secondaire indésirable. Ils fournissent parfois aux utilisateurs des documents qui ne sont pas à jour par rapport aux originaux sur des serveurs d'origine. Cela signifie que les utilisateurs sont fournis avec des pages périmées. Afin d'éviter ce problème, les caches doivent mettre à jour les documents qui y sont contenues. Le problème du maintien des documents en cache à jour avec les originaux est appelé le problème de cohérence de cache [29], [75].

II.6. Les mécanismes de cohérence de cache

Il existe deux types de techniques de cohérence du cache: la cohérence forte et la cohérence faible. La principale différence entre la cohérence forte et faible est que, dans la cohérence forte, pour assurer qu'il n'y a pas de perte de fraîcheur, à chaque fois le cache est obligé de garantir la fraîcheur du contenu servi par rapport au serveur d'origine. D'autre part, dans la cohérence de cache faible, le cache ne doit pas communiquer avec le serveur d'origine à chaque fois qu'il sert un document. Par conséquent, cache avec forte consistance est plus coûteux pour la mise en cache Web que la cohérence de cache faible [58].

Il y a au moins quatre techniques bien étudiées pour assurer la cohérence de cache. Ils comprennent client validation (client polling), les rappels invalidation, time-to-live (TTL) et If-Modified-Since [29].

II.6.1. Forte cohérence du cache

La forte cohérence du cache inclut la validation client et l'invalidation serveur.

II.6.1.1. Client polling (validation clients)

Dans Client polling, une vérification régulière de l'objet mis en cache avec ce lui de la version originale disponible sur le serveur d'origine. Cela permet de savoir si les objets mis en cache sont à jour avec l'original au niveau du serveur d'origine. S'ils ne le sont pas, ils sont mis à jour en allant chercher la dernière version du serveur d'origine.

Une méthode élégante pour la détection des fichiers ou des pages Web est de l'application des fonctions de hachage de type MD5 ou SHA-L. il est donc possible de détecter des changements même mineurs aux fichiers. En outre, ils peuvent être calculés très rapidement [58].

II.6.1.2. Invalidation rappels (invalidation serveur)

Il est possible de contourner l'exigence de vérification périodique d'inconsistance par les clients en confiant la tâche de détecter les objets périmés vers le serveur d'origine. Cette approche est employée par la technique de rappel invalidation pour assurer la consistance de cache.

Dans ce cas, le serveur doit tenir des registres de ces clients qui cachent ses objets. Il doit interagir avec ces procurations lorsque les objets sont modifiés. Le mécanisme de rappel invalidation permet la possibilité d'économiser la bande passante réseau, car les clients ne sont pas obligés d'interroger les serveurs de temps en temps. Toutefois, des questions telles que l'évolutivité, la sécurité et la confidentialité s'imposent en raison de l'exigence que les serveurs doivent suivre les caches pour chaque objet mis en cache. En plus de cela, les serveurs sont surchargés de travail supplémentaire.

Il existe d'autres variations de cette stratégie. On peut citer la stratégie de contrat (lease) et la stratégie d'abonnement [58].

II.6.1.3. La stratégie de contrat (Lease)

Variation de **(invalidation serveur),** Cette stratégie a été proposée dans [32], dans laquelle le serveur maintient, pour un document donné la liste des clients qui l'ont utilisés, et

auxquels, il fait la promesse de les notifier si ce document a été mis à jour pendant une période de temps donnée appelée lease (la durée de contrat). Par conséquent, un serveur ne supprime un client dans une liste que lorsque son contrat expire. De ce fait, une mise à jour peut être retardée par un client inaccessible pendant au plus la durée de son contrat. La figure **FIG II.1** illustre les interactions entre le cache client et le serveur d'origine avec la stratégie de contrat.

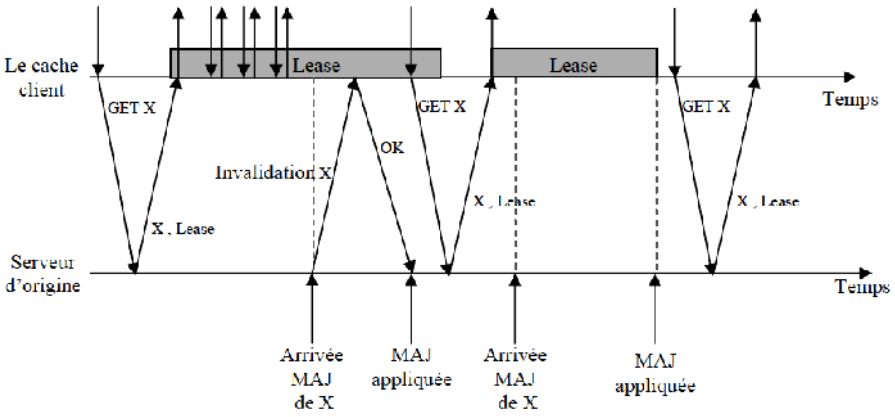


FIG II. 1 : L'invalidation par la stratégie de contrat (Lease)

II.6.1.4. La stratégie d'abonnement

Dans cette stratégie, les clients demandent explicitement au serveur de leurs accorder des abonnements à recevoir les invalidations [28]. Les études faites dans [29], ont montré que les documents accédés seulement une fois (one timers) constituent 60% de tous les documents de la trace étudiée. Ainsi, pour limiter l'invalidation uniquement aux documents qui bénéficieraient le plus, le serveur accorde les abonnements uniquement aux clients qui ont déjà une copie valide du document caché. En d'autres termes, le client peut demander un abonnement seulement avec en-tête if-modified-since, et le serveur le lui accorde si le document n'est pas modifié.

Ces abonnements peuvent être implémentés au niveau de la couche application ou la couche IP en utilisant le multicast. Au niveau de la couche application, le serveur d'origine maintient les listes des clients abonnés, et quand un document change, il invalide les clients de la liste correspondante en leur envoyant des messages d'invalidation unicast. Dans le cas de multicast, un groupe multicast est défini pour chaque document et le serveur envoie des messages d'invalidation aux adresses IP multicast correspondant aux documents changés. Les listes des clients sont stockées dans l'état d'adhésion multicast maintenu par les routeurs.

L'adresse multicast pour l'invalidation permet une diffusion efficace des mises à jour du serveur vers les caches clients.

II.6.2. Faible cohérence du cache

La technique faible cohérence du cache peut offrir une alternative à la technique forte cohérence du cache.

II.6.2.1. Time-To-Live (La durée de vie)

Le time-to-live (TTL) approche fixe une durée de vie des objets. Après la fin de la durée, les objets sont considérés comme périmés et les dernières versions des objets sont obtenues. La principale difficulté avec l'approche TTL est qu'il n'est pas facile de fixer la durée de vie des objets Web. En outre, les objets qui sont peu susceptibles de devenir périmés très rapidement ne doivent pas être mis à jour très souvent. En d'autres termes, leur durée de vie doit être maintenue élevée.

Adaptive server TTL : est une approche TTL. Dans cette technique, la durée de vie d'un objet est constituée juste au moment ou il y a un succès de cache.

Il existe plusieurs stratégies de validation adaptive TTL dont on peut citer :

- **Validation synchrone**

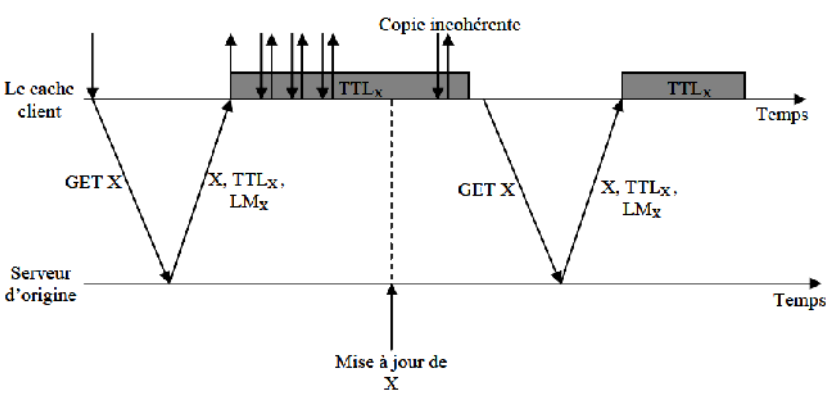


FIG II.2 : Validation de cache en utilisant les TTL

Quant un cache reçoit un document cachable envoyé par le serveur d'origine, ce dernier fournit avec ce document une valeur TTL (Time To Live) explicite, en utilisant les

champs d'en-têtes expire ou max-age, ainsi que le time stamp de la dernière mise à jour du document dans le champ d'en-tête last-modified.

Quand un document est caché, celui-ci peut être utilisé jusqu'à ce qu'il expire. Après cela, et au moment de référencement de ce même document, le cache client doit le valider avant de pouvoir l'utiliser encore. Cela se fait de manière synchrone puisque le client doit envoyer une requête conditionnelle (Par exemple une requête GET avec l'en-tête if modified since (IMS) contenant la date last-modified) comme illustré dans la figure **FIG II .2** au serveur d'origine et attendre sa réponse.

- **Validation asynchrone**

L'alternative à la validation synchrone est la validation asynchrone qui peut être implémentée de deux manières :

- Avec des threads séparés au niveau du client : ces threads vérifient périodiquement l'âge des documents dans le cache et valident ceux qui sont expirés.
- Avec des déclencheurs de temps (time triggers) : un trigger est armé lorsqu'un document est chargé dans le cache ou validé, et qui sera déclenché lorsque ce document expire. Quand un trigger est déclenché, il validera le document expiré.

La validation asynchrone élimine la latence. Cependant, elle maintient ou même augmente l'échange de message et la consommation de la bande passante, en raison des validations inutiles. Le coût en bande passante est particulièrement considérable quand une validation nécessite le chargement d'une nouvelle version d'un document inutilement.

II.6.2.2. Piggyback validation

L'utilisation de l'en-tête des requêtes réduit le coût en bande passante de la validation asynchrone mais pas le nombre de messages. Pour réduire le nombre de messages, on peut regrouper les requêtes de validation destinées à un même serveur dans un seul message. Cette approche est appelée Piggyback Cache Validation (PCV), la validation de cache par ferroutage et qui a été proposée par Krishnamurthy et Wills [46].

Dans PCV, quand un client doit envoyer une requête HTTP pour un serveur, il vérifie s'il possède d'autres documents de même serveur, et si un quelconque de ces documents a

expiré ou est sur le point de l'être, le client ajoute ses requêtes de validation sous forme d'en-têtes IMS avec la requête originale.

II.7. Le préchargement (Prefetching)

Il existe deux types de pré-chargement. Avec le pré-chargement informé ou hors ligne (off-line prefetching), l'utilisateur informe le système quant il va effectuer ses futurs accès. Avec le pré-chargement en ligne (on-line prefetching), le système se base sur les accès passes de l'utilisateur afin de prédire ses accès futurs

Le Prefetching représente un compromis entre la latence et la bande passante. Par prédiction, un cache sélectionne des documents au pré-chargement, en admettant qu'un client les demandera. Une prédiction correcte a comme conséquence une réduction de latence ; cependant, une prédiction incorrecte a comme conséquence une perte de bande passante [7] [38], [80].

II.8. Politiques de remplacement de caches

Comme la capacité mémoire des caches est fixe, ils ne peuvent contenir tous les documents qui leurs sont soumis. Alors, quand cet espace mémoire se remplis, le cache doit choisir un ensemble de documents à expulser pour faire de la place aux nouveaux.

Les politiques de remplacement de caches ont faits l'objet de recherches approfondies. Il en résulte donc de nombreux travaux. Plusieurs critères ont étés utilisés pour définir quand un objet doit être expulsé du cache. Ces critères sont:

- Le caractère récence de l'objet,
- La fréquence d'accès de l'objet, exprimée en fonction du nombre de requêtes à l'objet,
- La taille de l'objet,
- Le coût induit par le rapatriement de l'objet depuis le serveur d'origine,
- Le temps écoulé depuis la dernière modification de l'objet,
- La date d'expiration correspondant au temps à partir duquel l'objet est périmé.

Trois types de stratégies de remplacement, utilisant les critères énoncés ci-avant, ont été proposés :

(i) Les stratégies se basant sur un seule critère,

(ii) Les stratégies hybrides se basant sur plusieurs critères [43].

II.9. Les métriques des politiques de remplacement

L'une des questions à laquelle il faut répondre quand on fait une évaluation d'une politique de remplacement est : quelle sont les métriques ? Il existe de multiples métriques intéressantes, Certaines d'entres elles sont discutées ci-dessous :

II.9.1. Taux de succès (Hit rate)

Il représente le pourcentage des documents demandés qui sont servis à partir du cache par rapport au nombre total de documents demandés. Les petits documents sont généralement favorisés par les algorithmes de remplacement pour rester dans le cache afin d'augmenter le taux de succès.

II.9.2. Taux de succès en octets (Byte hit rate)

Il représente le pourcentage d'octets de données servies à partir du cache par rapport au nombre total d'octets servis. Cette métrique favorise pour le remplacement les petits documents tandis qu'un algorithme de remplacement basé sur le taux de succès favorise les grands.

II.9.3. Économie de bande passante

Cette métrique vise à quantifier le degré de diminution du nombre d'octets transférés à travers le réseau. C'est une métrique très importante pour beaucoup d'ISPs (Internet Service Provider) pour réduire leurs coûts, qui est directement liée à la bande passante.

II.9.4. Réduction de latence

La latence se réfère au délai de transmission d'une information d'un point à l'autre. Cette métrique mesure le degré de réduction de cette latence pour une politique de remplacement de cache donnée.

II.9.5. Performance de la CPU

Une politique de remplacement de cache excessivement compliquée pourrait faire de la CPU un goulot d'étranglement. Il est également important de noter que les politiques de

remplacement qui doivent examiner une grande quantité de données pourraient surcharger la mémoire et ainsi donc réduire sensiblement les performances de la CPU.

II.10. Les algorithmes de remplacement de caches

Après avoir décrit quelques métriques, on présente maintenant un survol sur les politiques de remplacement de cache et qui seront représentées plus en détail dans le chapitre suivant. Généralement, ces algorithmes peuvent être regroupés en trois catégories comme proposé par Aggarwal et al [3].

II.10.1. Les algorithmes traditionnels et leurs extensions directes

La première catégorie, inclut les algorithmes classiques de remplacement, dont on cite principalement LRU (Least Recently Used) et LFU (Least Frequently Used), comme leurs noms l'indiquent, LRU remplace le document le moins récemment utilisé, tandis que LFU remplace le moins fréquent, on a aussi l'algorithme FIFO (First In, First Out) qui est encore plus simple que LRU et LFU, les documents sont remplacés dans le même ordre dont ils ont été ajoutés au cache.

Parmi les extensions de LRU on a LRU-Threshold et LRU-Min proposées dans [1], EXP1 [64], HLRU [79], PSS (Pyramidal Selection Scheme) [3] et LRU-LSC [34]. LFU possède aussi plusieurs extensions dont on peut citer, LFU-Aging et LFU-DA proposées dans [4], α-Aging [93], etc.

II.10.2. Les algorithmes à base de clefs

Dans ce type d'algorithmes, les documents sont ordonnés à base d'une clef primaire, et une clef secondaire est utilisée pour ordonner les documents ayant une même clef primaire, et si ces mêmes documents ont la même clef secondaire, on utilise une clef tertiaire, et ainsi de suite. Par exemple, la politique de HYPER-G proposée dans [84] utilise la fréquence d'accès comme clef primaire, la récence comme clef secondaire et finalement la taille comme clef tertiaire. On a aussi plusieurs autres politiques dans cette catégorie comme : swLFU (Server-Weighted LFU) [42], SLRU (Segmented LRU) [5] et LRU-Hot [55], etc.

II.10.3. Les algorithmes à base de fonction

Ces politiques de remplacement ont eu recours à des fonctions dans l'intention d'inclure plusieurs facteurs tels que le temps depuis le dernier accès, le temps d'entrée dans le

cache, le coût de transfert, le temps d'expiration (TTL), la taille et ainsi de suite. Par exemple, l'algorithme HYBRID décrit dans [85] utilise une fonction exponentielle basée sur la fréquence d'accès, la taille, la latence au serveur et la bande passante au serveur. On peu citer encore, GD (Greedy Dual)-Size [16] suivie par GDSF [6] et GD* [37], TSP (Taylor Series Prediction) [87], LRV (Lowest Relative Value) [66], etc.

II.11. L'apport d'utilisation de cache dans les réseaux Ad hoc

L'introduction de caches permet de réduire la surcharge du réseau puisqu'ils diminuent le nombre de messages, De plus, l'introduction de cache atténue la charge des terminaux mettant à disposition des contenus puisqu'ils sont moins souvent sollicités. Elle diminue également le temps d'attentes de l'utilisateur puisque les contenus deviennent accessibles localement. Diminuant ainsi la consommation d'énergie et la consommation de la bande passante du réseau. Finalement, l'intégration de caches accroit la disponibilité des contenus puisque ces derniers restent accessibles même si leur fournisseur ne l'est plus [28].

II.11.1. Réduction de la latence

Comme illustré dans la Figure **FIG II.4** le gain en latence peut être considérable pour une requête servie depuis un nœud intermédiaire plutôt que depuis la source du document demandé.

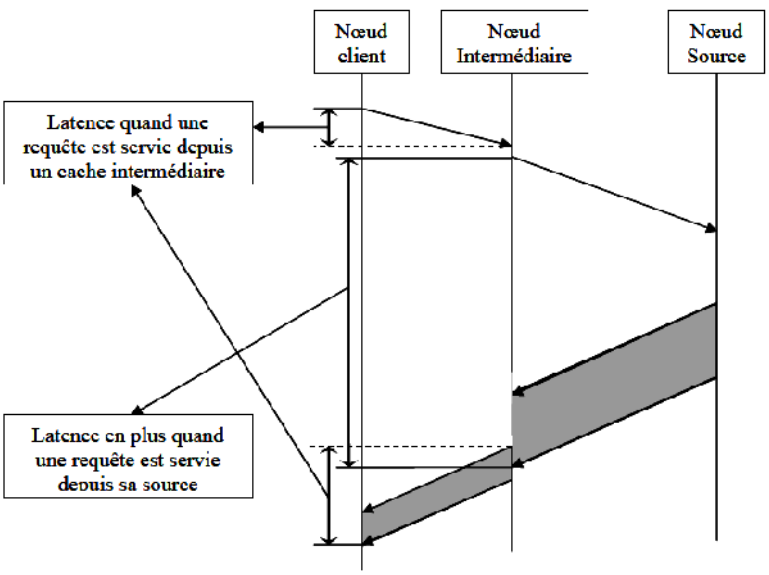


FIG II.4 : Réduction de la latence par la technique du cache

II.11.2. Sauvegarde de la bande passante

Un autre avantage pour l'utilisation des caches dans les réseaux mobiles Ad hoc et dans les réseaux en général, est la réduction de la consommation de la bande passante. Avec cette réduction, l'avantage évident est non seulement la réduction des coûts globaux des transferts à travers le réseau, mais elle réduit également la charge de lien vers la source, et donc elle diminue indirectement la latence.

II.11.3. Augmentation de l'accessibilité des données

Vu le caractère de topologie dynamique des réseaux mobiles Ad hoc qui entraine de fréquentes déconnexions et d'éventuelles partitionnements de réseaux, une source de données peut devenir inaccessible, mais les requêtes qui lui sont destinées peuvent être satisfaites par les caches des autres nœuds accessibles.

II.12. Cache coopérative

La mise en cache coopérative a été largement étudiée pour économiser la bande passante et de réduire la latence et de congestion dans l'Internet [6]. Dans un environnement de mise en cache coopérative, un certain nombre de serveurs de cache travaille ensemble au service d'un ensemble de clients pour s'aider à caché plus d'objets collectivement sans surcharger un seul serveur cache.

Fondamentalement, l'architecture de mise en cache Web coopératif peuvent être divisée en deux catégories, hiérarchique et distribuée [82].

II.12.1. Caches hiérarchiques [67]

Dans les schémas de mise en cache hiérarchique, les serveurs de cache sont organisés comme une hiérarchie arborescente et les caches sont placés à plusieurs niveaux du réseau. La mise en cache hiérarchique a été proposée en le projet Harvest [19] et est largement utilisée dans les serveurs proxy d'aujourd'hui, c'est à dire Squid [76].

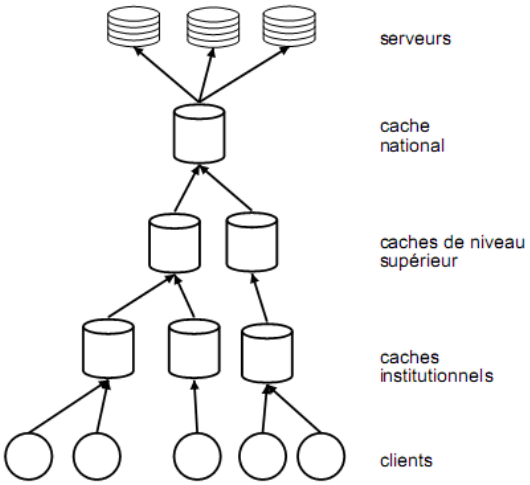


FIG II. 6 : Structure hiérarchique des caches proxys coopératives.

II.12.2. Caches distribuées [67]

Dans la mise en cache distribuée, les serveurs cache tendent de maintenir des relations peer-to-peer, et les caches sont placés au niveau bas du réseau [44].

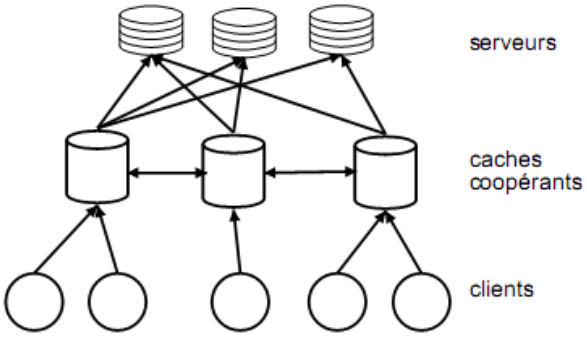


FIG II. 5 : Schéma des caches proxys coopératifs

II.13. Cache coopérative (IMANET)

Yin et al. Ont proposé trois schémas de coopération mise en cache pour les réseaux Ad hoc: CachePath, CacheData et HybridCache [90].

En CacheData, cache intermédiaire pour servir les requêtes a' venir au lieu de récupérer les données du serveur.

Dans CachePath, les nœuds cachent les chemins des données, pour rediriger les demandes futures des nœuds voisins vers elles.

En HybridCache, les avantages de CachePath et CacheData sont combinés pour améliorer les performances. Shen et al. Ont présenté un schéma de mise en cache d'adaptation de coopération pour les réseaux mobiles Ad hoc [70]. Dans leur schéma, ils définirent une zone du terminal mobile comme un groupe de terminaux mobiles voisins d'un nombre de sauts. Si une demande est produite, elle est d'abord diffusée aux terminaux mobiles dans la zone pour récupérer les données demandées. Si les données ne sont pas en cache dans la zone, une communication peer-to-peer est utilisée pour transmettre la demande à la station de base, c'est-à-dire que la demande est transmise vers le nœud qui se trouve au long du chemin de serveur, si la requête est satisfaite par le cache de ce nœud ou le cache d'un de ses voisins de la même zone, la donnée sera transféré au demandeur, sinon le nœud enchainera cette demande au nœud suivant jusqu'à l'arrivé de la demande au serveur d'origine.

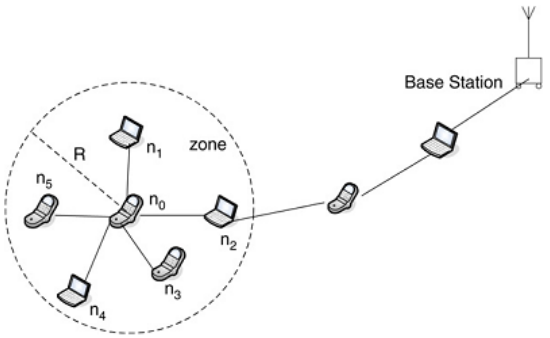


FIG II. 7 : Zones de réseau Ad hoc

II.13.1. La zone Coopérative [92]

Chaque nœud dispose d'un cache pour stocker les données fréquemment utilisées. Les données dans le cache ne satisfassent pas seulement les propres demandes locales de nœud, mais aussi les demandes des autres nœuds de réseau qui le traverse. En cas de *miss* cache local, le nœud recherche d'abord les données dans sa zone avant de transmettre la demande au nœud suivant qui se trouve sur un chemin vers le serveur. Toutefois, la latence peut être plus longue si les voisins intermédiaires de nœud n'ont pas une copie de l'objet demandé.

II.13.1.1. La zone coopérative d'un seul saut

Une recherche antérieure [70] a présenté un projet de zone de coopération, où une zone de coopération contient une série de terminaux mobiles dans le secteur de K sauts (appelé régime K-Zone).

Une zone coopérative d'un seul saut est une zone qui ne contienne que l'ensemble des terminaux voisins situés dans une distance d'un seul saut. Le choix d'une zone d'un seul saut pour le reste de notre travail est basé sur les raisons suivantes:

(1) Évitez les inondations: Dans le système K-Zone, une demande devrait être diffusée à tous les terminaux mobiles au sein de K sauts, ce qui conduit à des coûts de communication importants. Dans notre système, une demande a seulement besoin d'être diffusée une fois à ses voisins pour récupérer les données souhaitées.

(2) Réduire le temps de la latence d'accès: Dans le schéma K-Zone, un nœud doit attendre les réponses de tous les nœuds à l'intérieur de K sauts avant de transmettre une requête à la station de base. Notre système n'a besoin que de communiquer avec le voisinage d'un nœud, ce qui est plus simple et plus rapide.

(3) Economie d'énergie: Dans le système K-Zone, si K est grand, on risque d'inonder le système par les messages. Ainsi, le système va consommer beaucoup plus d'énergie que dans le régime d'un seul saut. D'ailleurs, dans un MANET, les terminaux mobiles se déplacent librement, donc la topologie du réseau change fréquemment, entraînant des coûts supplémentaires pour obtenir les informations sur la topologie K-zones.

(4) Intuitivement, si une donnée ne peut être trouvée dans les terminaux voisins, il y a seulement une petite chance qu'elle se trouve dans un nœud lointain.

Une simulation a montré que la probabilité de cache hit en dehors de la zone d'un seul saut ne dépasse pas 0,03%, ce qui est négligeable. Donc, la limitation de la zone à un saut est raisonnable [17].

II.14. Conclusion

L'utilisation des caches dans les réseaux mobiles Ad hoc s'est avérée être une technique très importante pour améliorer d'une manière significative les performances de ces réseaux. Néanmoins, cette solution pose un certain nombre de problèmes liés à la gestion des caches, comme la consistance des données cachées et les techniques de remplacement de caches et les techniques de coopération.

Nous allons présenter dans le chapitre suivant une classification des politiques de remplacement de caches, qui existent dans la littérature, en donnant pour chaque classe ses avantages et ses inconvénients. Nous terminons, par une proposition d'une politique coopérative de remplacement de cache, MZS (Mobile Zone Serveur) pour les réseaux mobiles Ad hoc, basée sur une fonction. Cette fonction calcule une valeur pour chaque document en prenant en considération sa fréquence, sa récence et sa taille, ainsi qu'un paramètre de coopération.

Chapitre 3

Les politiques de remplacement de caches

III.1. Introduction

Les caches représentent des mémoires ayant des capacités de stockage finies. Par conséquent, ils ne peuvent contenir qu'un nombre fini de documents. Lorsque l'espace libre devient insuffisant pour contenir un nouveau document, il est nécessaire de remplacer les documents jugés moins utiles, par les nouveaux documents, et cela, suivant une politique de remplacement de cache.

Plusieurs politiques de remplacement de cache ont été proposées, dans le but d'améliorer les performances des caches. Dans ce chapitre, nous allons présenter une synthèse des politiques qui existent dans la littérature, en les classifiant en cinq classes, et en donnant pour chaque classe les avantages et les inconvénients. Nous terminons par une proposition d'une politique coopérative de remplacement de cache pour les réseaux mobiles Ad hoc.

III.2. Classification des politiques de remplacement de cache

Pour présenter les différentes propositions de politiques de remplacement de cache qui existent, d'une manière structurée, nous procédons d'abord à leurs classifications, selon les paramètres (caractéristiques) des documents, qui peuvent influencer le processus de remplacement. La plupart de ces paramètres sont décrits dans Krishnamurthy et Rexford [45]

On donne ci après un récapitulatif des paramètres importants :

1) récence : période depuis la dernière référence de document ;
2) fréquence : nombre d'accès à un document ;

3) taille : taille de document ;

4) coût de chargement: coût de chargement d'un document depuis son serveur d'origine;

5) temps de modification : période depuis la dernière modification ;

6) temps d'expiration (TTL) : est la période de temps pendant laquelle le document restera valide (cohérent). Quand cette période expire le document devient périmé et peut être remplacé immédiatement.

Ces facteurs peuvent être incorporés à la décision de remplacement.

Dans la littérature, la plupart des propositions utilisent les quatre premiers facteurs.

La récence et la fréquence sont des facteurs importants mais très différents. Par conséquent, les stratégies qui utilisent ces facteurs indépendamment, doivent être mises dans des classes séparées. De plus, certaines stratégies combinent la récence et la fréquence, donc elles doivent être traitées dans leur propre classe.

Certaines stratégies calculent une fonction spécifique incorporant différente facteurs. Dans la plupart du temps des poids sont utilisés et associés à chaque facteur, et donc une classe séparée est nécessaire pour ce genre de stratégies. Et enfin, une dernière classe pour les stratégies dédiées aux environnements mobiles. Ces stratégies sont adaptées aux spécificités et caractéristiques de ces environnements. Donc cinq classes de stratégies de remplacement de caches sont mises en évidence, à savoir :

- Les stratégies de remplacement de cache basées sur la récence
- Les stratégies de remplacement de cache basées sur la fréquence
- Les stratégies de remplacement de cache basées sur la taille.
- Les stratégies de remplacement de cache basées sur les fonctions
- Les stratégies de remplacement de cache dédiées aux environnements mobiles

Kin-Yeung Wong [43] a classé la majorité des politiques existantes dans la littérature en 2006 comme montre le tableau ci-dessous

Catégorie	Conception logique	Bonne pour	
Basé sur la récence	Un objet récemment référencée sera référencé de nouveau en un avenir proche.	Lorsque de nombreux utilisateurs sont intéressés par les mêmes objets Web au même moment.	
Basé sur la fréquence	Seul un petit ensemble d'objets est populaire, et ces objets devraient être mis en cache.	Lorsque les utilisateurs ont tendance à accéder à des sites Web dont les objets avec popularités plutôt stable.	
Basé sur la taille	Suppression d'un objet plus grand peut faire de la place pour de multiples petits.	Lorsque les utilisateurs ont tendance à accéder à l'information des sites Web.	
Basés sur des fonctions	Contenir plus de paramètre pourrait parvenir à un supérieur taux de réussite	Lorsque le système a suffisamment de ressources mémoire et de capacités de calcul.	
Logique de conception et l'environnement préférable			
catégorie	**Les politiques de remplacement disponibles**	**Représentant de la politique**	
Basé sur la récence	LRU, LRU-threshold, LRU*, LRU-hot, LRU-LSC, SB-LRU,SLRU, HLRU, Pitkow/Recker, EXP1, value-aging, generational replacement	LRU	
Basé sur la fréquence	LFU, LFU-Aging, LFU-DA, Window-LFU, swLFU, Aged-swLFU, α-Aging, HYPER-G	LFU-DA	
Basé sur la taille	SIZE, LRU min, partitioned caching, PSS, CSS, LRU-SP	La taille	
Basés sur des fonctions	GD-Size, GDSF, GD*, PGDS, server-assisted cache replace-ment, TSP, Bolot/Hoschka, MIX, M-Metric, HYBRID, LNC-R-W3, LRV, LUV, LR, N-gram	GD-Size	
Quelques politiques disponibles			
catégorie	**Taux de réussite**	**Taux de réussite en octets**	**Complexité**
Basé sur la récence	Constant	Constant	Constant
Basé sur la fréquence	Constant	Constant	Constant
Basé sur la taille	Constant	Constant	Constant
Basés sur des fonctions	Le Meilleur	Le Meilleur	Le Plus haut
Performances globales			

Table III.1 : Catégories des politiques de remplacement du cache [43].

III.2.1. Les stratégies de remplacement de cache basées sur la récence

III.2.1.1. Least Recently Used (LRU)

LRU est basé sur l'hypothèse qu'un document qui a récemment été demandée sera demandé à nouveau. Il ne supprime pas un document qui a récemment été consulté. De cette façon, un document populaire qui est consulté plusieurs fois en un court intervalle de temps sera conservé dans le cache.

III.2.1.2. LRU-Threshold [1]

Un objet ne sera pas caché si sa taille dépasse un seuil donnée, sinon elle procède exactement comme LRU.

III.2.1.3. LRU* [22]

Dans cette stratégie tous les documents demandés sont stockés dans une liste LRU. Chaque document a un compteur de nombre de fois qu'il a été référencé. Quand un document caché est référencé alors il est déplacé au début de la liste LRU et son compteur est incrémenté de un. À chaque fois qu'il y a un remplacement, le compteur de documents le moins récemment utilisé est vérifié. S'il est à zéro, le document est supprimé du cache. Autrement, le compteur est décrémenté de un, et le document est déplacé en tête de la liste.

III.2.1.4. LRU-Hot [55]

LRU-Hot contrôle deux listes LRU : une pour les documents chauds (populaires) et une pour les documents froids (pas aussi populaires). Un document est chaud, si sa fréquence d'accès au serveur original est au-dessus d'un certain seuil. Cette information est envoyée avec le document au client. Selon cette information, le document est inséré dans la liste correspondante. Ces listes sont traitées différemment. LRU-Hot utilise deux compteurs de référence : un compteur de base et un compteur pour les documents chauds. Les deux compteurs sont initialisés à zéro. Chaque requête augmente le compteur de base de un. Le compteur chaud est incrémenté de un après chaque α requêtes (α > 1). Quand un document est reçu, il est stocké au début de sa liste correspondante, et une valeur d'accès lui est affectée et est égale à la valeur actuelle du compteur de base. Sur un remplacement, le cache recalcule les valeurs des deux documents se trouvant dans les queues des deux listes.

$$hot\_value =$$

$$tail_{hot} - hot\,reference\,counter$$

$$cold\_value =$$

$$tail\_cold - cold\,reference\,counter$$

Le document avec la plus petite valeur est enlevé du cache. L'algorithme est réitéré jusqu'à ce qu'il y ait assez d'espace pour le nouveau document entrant.

III.2.1.5. LRU-LSC [34]

Cette stratégie utilise une liste LRU pour déterminer « l'activité » de différents objets en cache. Au moment du remplacement des objets de moindre activité (à partir de la fin de la liste LRU) ils seront placés dans une seconde liste. La nouvelle liste est triée selon une certaine valeur $spc_i = Ci/Si * t_i$, les objets sont retirés de cette nouvelle liste jusqu'à ce que leur taille cumulée soustraite de la taille de tous les objets mis en cache soit inférieure à une taille donnée.

C_i= Coût pour aller chercher l'objet *i* à partir de son serveur d'origine; C_i a un sens plus général tel que le nombre de saut et la latence ;

S_i=Taille de l'objet *i* ;

t_i= Temps la dernière demande de l'objet *i* ;

III.2.1.6. SLRU (Segmented LRU) [5]

La stratégie SLRU divise le cache en deux segments : un segment non protégé et un segment protégé (réservé pour les documents populaires). Sur la première requête pour un document, celui-ci est inséré dans le segment non protégé. Si le document est référencé à nouveau (un cache hit), alors le document est déplacé au segment protégé. Cette politique était destinée initialement aux caches disques [41]. Elle a été utilisée pour le Web [5] parce qu'elle prend en considération la fréquence et la récence d'accès dans les décisions de remplacement. En effet, les deux segments sont contrôlés avec la stratégie LRU, mais uniquement les documents du segment non protégé qui sont remplacés. Les documents du segment protégé sont déplacés vers le segment non protégé suivant la stratégie LRU. SLRU exige l'utilisation d'un paramètre qui détermine quel pourcentage de l'espace de cache est alloué au segment protégé.

III.2.1.7. HLRU [79]

HLRU introduit un schéma, pour supporter l'historique d'un certain nombre de références à un document spécifique. Soient r1,r2,…..,rn des requêtes à un document caché à des temps t1,t2,…,tn. La fonction de l'historique est définie comme suit :

$$\text{Hist}(x,h) = \begin{cases} t_i & \text{si il y a exactement h-1 références} \\ & \\ \text{Entre ti et tn,} & \end{cases}$$

Hist(x,h) définie le temps des h dernières références d'un document caché x. HLRU remplace le document ayant la valeur maximale de hist. S'il y a beaucoup de documents avec un hist=0, alors LRU est utilisée pour le remplacement.

III.2.1.8. PitKow/Recker [63]

Elle permet de supprimer les documents qui sont LRU, sauf si tous les documents sont accédés le jour même, dans ce cas supprimer le plus grand.

III.2.1.9. EXP1 [64]

C'est une extension de LRU qui utilise la période de temps entre la date courante et la date de dernier accès pour peser l'importance d'un document. EXP1 choisit un document *i* si

$\frac{1}{\mu_i} = \min\{\frac{1}{\mu_i}\}$ avec *i*=1...N (ensemble de N documents). μ_i est calculé à base des périodes successives entre les différents accès au document *i*.

III.2.1.10. Generational Replacement [60]

Tous les objets sont stockés dans n (n>1) listes. Chaque liste i<n contient des objets qui ont été demandés i fois. La liste n contient tous les objets qui sont demandes n fois où plus, Une demande d'un objet entraîne sa suppression dans sa liste actuelle et de son insertion dans la liste suivante (au début). Les objets en liste n sont insérés au début de liste n. Le remplacement aura lieu à la fin de la liste 1.

III.2.1.11. Value-Aging [93]

La politique « valeur de vieillissement » utilise la formule suivante pour le remplacement : $V_{new}(i)$ est mis à jour à chaque demande de l'objet (i):

$$V_{new}(i) = V_{old}(i) + C_t * \sqrt{\frac{C_t - t_i}{2}} \quad \text{où}$$

C_t = L'heure actuel.

t_i = Temps de la dernière demande de l'objet (i).

III.2.1.12. Avantages

- Elles considèrent la localité temporelle, comme facteur principal, et les traces empiriques des requêtes du Web présentent ce comportement de localité temporelle, ce qui est donc avantageux.
- Elles sont simples à mettre en œuvre. La plupart de ces stratégies utilisent une liste LRU. Les nouveaux documents demandés sont insérés à la tête de la liste. Quand il y a un hit le document est enlevé de sa position actuelle et inséré à la tête et le remplacement se fait à la fin de la liste.

III.2.1.13. Inconvénients

- Les variantes LRU simples ne combinent pas la récence et la taille. Et comme les documents Web ont des tailles très différentes, la taille devrait être considérée à chaque remplacement.
- Elles ne considèrent pas la fréquence. Ceci a pu être un indicateur très important dans plusieurs environnements.

III.2.2. Les stratégies de remplacement de cache Basées sur la fréquence

III.2.2.1. LFU (least frequently used) [62]

LFU est basée sur la pensée que les documents demandés avec une probabilité fixe, Les documents demandés avec la plus grande probabilité devraient donc être conservés dans le cache. Quand il est nécessaire, un document avec les moins fréquences d'accès est choisi pour être retirer.

III.2.2.2. LFU-Aging [4]

Avec LFU, les documents qui étaient très populaires, pendant une période de temps, peuvent demeurer dans la cache, même lorsqu'ils ne sont pas demandés pendant une longue autre période. Ceci est dû à la grande valeur de leur compteur de fréquence, c'est ce qu'on appelle la pollution de cache. Pour éviter cette pollution, une technique de « Aging » peut être introduite. Le LFU-Aging utilise donc un seuil, si la valeur moyenne de tous les compteurs de fréquence excède ce seuil, tous les compteurs de fréquence sont divisés par 2.

III.2.2.3. LFU-DA [4]

Avec LFU-DA, quand un document *i* est référencé, alors sa valeur *Ki* est (re)calculée comme suit :

$$Ki = Fi + L,$$

où *Fi* et *L* sont respectivement, la fréquence de document *i* et le facteur Aging qui est initialisé à zéro. LFU-DA choisit le document avec la plus petite valeur de *Ki*. Cette valeur Ki est affectée à *L* du nouveau document entré en cache.

III.2.2.4. SWLFU (Server-Weighted LFU) [42]

Cette stratégie utilise pour chaque document *i* un poids de fréquence Wi, ce poids indique l'appréciation du serveur de ce document *i* pour qu'il soit caché. Par conséquent, le serveur peut influencer la cachabilité d'un document. La stratégie LRU est appliquée aux documents en second cas, la où on a la même valeur Wi.

III.2.2.5. A-swLFU (Aged-swLFU)

Cette stratégie expulse l'objet *LRU*, à chaque k remplacements.

k = 0 correspond à la *swLFU* stratégie originale.

k = 1 correspond à *LRU*.

k > 1 donne un mélange de récence et fréquence.

III.2.2.6. α-âge [93]

Il s'agit d'une explicite méthode de vieillissement avec une fonction périodique de vieillissement $f(v) = \alpha * fi \quad 0 \leq \alpha \leq 1$.

Chaque impulsion d'horloge virtuelle (par exemple, 1 heure) la valeur de chaque objet est diminuée à α fois sa valeur originale. *f (v)* augmenté de un après chaque hit.

Avec la variation de α de 0 à 1, on peut obtenir un spectre d'algorithmes allant de la *LRU (α = 0)* à *LFU (α = 1)*.

fi= nombre de demandes de l'objet i.

III.2.2.7. HYPER-G [84]

Cette stratégie combine *LRU* et *LFU*, et elle utilise aussi la taille des documents. Au début, le document le moins fréquemment utilisé est choisi. S'il y a plus d'un document qui répond à ce critère, HYPER-G choisit le moins récemment utilisé. Si ceci ne donne toujours pas un document unique à remplacer, le plus grand document est choisi.

III.2.2.8. Fréquence relative de Benhamida [10]

Cette stratégie utilise une fréquence relative F_i^r au lieu du nombre d'accès pour chaque Document *i*. Elle est calculée comme suit :

$$F_i^r = F_i/(now - D_i)$$

F_i : est la fréquence du document *i* ;

D_i : est la date d'entrée du document *i* dans le cache ;

now : est la date courante.

III.2.2.9. Avantages

Elles considèrent la fréquence d'accès, qui est un facteur très important dans les environnements statiques où la popularité des documents ne change pas beaucoup sur une période de temps spécifique.

Ces stratégies présentent comme avantage le fait de combiner la récence et la fréquence, et par conséquent, de même pour leurs deux avantages. En effet ces stratégies peuvent aussi nous éviter les problèmes rencontrés dans l'utilisation uniquement de l'une de ces deux approches à part.

III.2.2.10. Inconvénients

- **Complexité :** Les stratégies basées sur LFU exigent une gestion plus complexe de cache. LFU peut être mise en œuvre, par exemple avec une file d'attente prioritaire.
- Les procédures spéciales utilisées dans les mécanismes de remplacement de cache et qui combine la fréquence et la récence. En effet, la plupart d'entre elles introduisent une complexité additionnelle, cependant, elle ne prend pas en considération la taille des documents.

- **Pollution de cache :** Les documents qui étaient très populaires pendant une période de temps peuvent demeurer dans le cache même lorsqu'ils ne sont pas demandés pendant une longue autre période. C'est pour cela qu'on fait appelle à la technique de Aging qui n'est rien d'autre qu'une technique basée sur la récence.
- **Valeurs similaires :** Beaucoup de documents peuvent avoir la même fréquence. Dans ce cas-ci, un autre facteur de comparaison est nécessaire.

III.2.3. Les stratégies de remplacement de cache basées sur la taille

III.2.3.1. SIZE [84]

Cette stratégie remplace le plus grand document. La stratégie *LRU* est appliquée aux documents de même taille.

III.2.3.2. LRU-Min [1]

C'est une variante de la stratégie *LRU* qui essaye de minimiser le nombre de documents à remplacer. Soient *Lo* et *T* qui dénote, respectivement, une liste et un seuil.

(1) T reçoit S, où S est la taille du document demandé.

(2) Lo reçoit tous les documents qui ont une taille supérieure ou égale à T (Lo peut être vide).

(3) Appliquer *LRU* pour *Lo* jusqu'à ce que la liste soit vide ou l'espace libre de cache soit au moins T.

(4) si l'espace libre du cache n'est pas au moins *S*, alors *T* reçoit *T/2* et aller à l'étape (2).

III.2.3.3. Partitioned Caching [57]

Cette stratégie classe tous les objets Web selon leur taille en trois groupes (petit, moyen, grand). Le seuil pour ce classement est dérivé antérieurement à partir de traces Web. Chaque classe a son espace cache propre et dirigé de manière indépendante des deux autres classes avec la stratégie *LRU*. Soit *Sc* la taille du cache et *Sc1, Sc2, Sc3* avec ($Sc1 + Sc2 + Sc3 = Sc$) la taille des trois partitions (petite = 1, moyenne = 2, grande = 3). Murta et al [1998] ont montré expérimentalement que cette stratégie de remplacement de Cache Web doive détenir : *Sc1 <Sc2 < Sc3*.

III.2.3.4. PSS (Pyramidal Selection Scheme) [3]

Cette stratégie fait une classification pyramidale des documents en fonction de leur taille. Tous les documents de la classe i ont une taille comprise entre 2^{i-1} et 2^{i}-1. Chaque classe a une liste *LRU* séparée. À chaque fois qu'il y a un remplacement, les valeurs *LRU* des documents de chaque classe sont comparées. *PSS* choisit le document *i* si sa valeur $Si \times \Delta Ti$ est la plus grande parmi toutes les valeurs de ces documents. *Si* et ΔTi sont respectivement, la taille du document i et le nombre d'accès au cache depuis le dernier accès au document i.

III.2.3.5. CSS (Cubic Selection Scheme) [77]

CSS est une extension de la stratégie *PSS*. *CSS* est basé sur un cube comme structure de données. Le cube *CSS* est formé par une matrice dont les éléments sont des listes d'objets qui correspondent à une seule classe. Les objets dans une même classe ont une taille [*log Si*] et une fréquence de demande [*log fi*] soit *MaxF* la fréquence maximale possible, la hauteur de la matrice est donnée par [*log MaxF*]+1 et la largeur par [*log M*]+1. M est la taille maximale d'un objet.

III.2.3.6. LRU-SP [68]

LRU-SP est une autre extension de *PSS*, Tout comme *PSS,* elle utilise des différentes classes. Une classes d'objet *i* est déterminée par [*log(Si/fi)*], chaque classe maintien une liste *LRU* séparé. Une demande de caché un objet *i* peut causer son réarrangement dans une autre liste. En cas de remplacement, les valeurs d'objet le moins récemment utilisé dans chaque classe sont comparées. *LRU-SP* choisi un objet *i* si leur valeurs ($\Delta TiSi)/fi$ est la plus grande parmi toutes les valeurs d'objets.

ΔTi est le nombre d'accès depuis le dernier accès a l'objet *i* est demandé (comme dans *PSS*).

III.2.3.7. Avantages

Les stratégies basées sur la taille combinent la récence et la fréquence, s'ils sont bien définis, chaque stratégie peut éviter les inconvénients des deux stratégies.

III.2.3.8. Inconvénients

La prise en compte de la taille et la fréquence nécessite une bonne gestion du cache. Une modification de la valeur de la fréquence peut entraîner une réorganisation des listes, parce que l'objet correspondant doit être inséré dans une nouvelle liste.

III.2.4. Les stratégies de remplacement de cache basées sur des fonctions

III.2.4.1. GD (Greedy Dual)-Size [16]

La GD-Size maintient pour chaque document une valeur caractéristique *Hi*. Une requête pour un document *i* exige de recalculer *Hi* comme suit :

$$H_i = \frac{C_i}{S_i} + L,$$

où *Ci* est le coût de chargement du document *i* depuis son serveur ;

Si est la taille de document *i* ;

L est le facteur courant de vieillissement qui est initialisé à zéro. La GD-Size choisit le document avec la plus petite valeur de *Hi*. La valeur de ce document est affectée à *L*.

III.2.4.2. GDSF [4]

GDSF calcule *Hi* comme suit

$$H_i = f_i \frac{C_i}{S_i} + L,$$

où *fi* est la fréquence du document *i*.

Une forme généralisée de *GDSF* a été décrite dans [93]. Elle calcule *Hi* comme suit

$$H_i = \frac{f_i^a}{S_i^\beta} + L$$

Ci = 1 ;

α et *β* sont des poids paramétrables qui indiquent l'importance des facteurs utilisés.

III.2.4.3. GD* [37]

*GD** calcule *Hi* comme suit:

$$H_i = \left(\frac{(f_i * C_i)}{S_i}\right)^{\frac{1}{\beta}} + L$$

β est un facteur de pondération, qui est estimé dans un mode en ligne. β caractérise la corrélation de référence. La corrélation de référence est mesurée par la distribution de référence de l'inter arrivé pour les objets populaires.

III.2.4.4. Server-assisted cache replacement [23]

Cette stratégie renforce les politiques de remplacement en fournissant les proxys avec la fonction de répartition du temps de la prochaine demande pour chaque objet. La distribution est estimée par la collecte de statistiques par objet sur le temps inter requête. Le serveur génère quotidiennement un histogramme d'inter requête en observant ces demandes. Ces statistiques sont intégrées dans le calcul du bénéfice d'un objet.

III.2.4.5. TSP (Taylor Series Prediction) [87]

TSP calcule Hi comme suit

$$H_i = \frac{f_i \times C_i}{S_i \times T_t}$$

et

$$T_t = t_p - t_c$$

où Ci est le coût de chargement du document i depuis son serveur ;

S_i est la taille de document i ;

f_i est la fréquence de document i ;

T_t décrit "l'accroissement" temporelle de la requête vers le document i ;

t_p est une prédiction de temps prévu de la prochaine requête pour ce document i ;

t_c est le temps courant. Le t_p est déterminé avec la série de Taylor de second degré.

III.2.4.6. Stratégie de Bolot et Hoschka [11]

Cette stratégie utilise la fonction suivante pour calculer la valeur de document. :

$$f(i) = W1 + W2si + \frac{W3+W4\,Si}{Ti},$$

où W1, W2, W3 et W4 sont des paramètres de réglage. Les valeurs des ces paramètres dépendent de la métrique de performance qui devrait être optimisée.

III.2.4.7. MIX [59]

La stratégie MIX utilise la fonction suivante pour calculer la valeur de l'objet i:

$$f(i) = \frac{l_i^{r1} f_i^{r2}}{T_i^{r3} S_i^{r4}}$$

r_i (i = 1, ..., 4) sont les paramètres de réglage, qui devraient être déterminés par des essais expérimentaux. Il n'y a pas d'intervalles définis pour ces paramètres de réglage. Les auteurs de l'algorithme utilisaient $r_i = 1$ pour i = 2, 3, 4 et $r_1 = 0{,}1$.

III.2.4.8. M-Metric [81]

La valeur d'un objet i est donnée par:

$$f(i) = f_i^f T_i^r S_i^s$$

f, s et r sont des paramètres de pondération. f doit avoir une valeur positive pour donner le poids au plus demandé. r devrait être négatif pour donner plus de poids aux dernières demandes. s peut être positif ou négatif. Une valeur positive donne plus de poids aux grands objets, une valeur négative donne plus de poids aux petits objets.

III.2.4.9. HYBRID [85]

La fonction d'un objet i de serveur S dépend des paramètres suivants:

C_s (le temps de contact de serveur), b_s (bande passante disponible pour le serveur s). La fonction est définie par

$$f(i) = \frac{(C_s + \frac{W_b}{b_s}) f_i^{W_n}}{S_i}$$

où W_b et W_n sont des paramètres de pondération. Les estimations pour C_s et bs sont basées sur le temps de recherche des documents à partir du serveur s dans un passé récent.

III.2.4.10. LNC-R-W3 [72]

Le profit $p(i)$ d'un objet i est calculé comme

$$p(i) = \frac{f_i l_i}{s_i},$$

f_i est calculé en tenant compte de la dernière demande K (K>1) (fenêtre coulissante de demandes K):

$$f_i = \frac{K}{(t-t_k)s_i^b}.$$

où t correspond au temps réel et t_k est le temps de la plus ancienne demande dans la fenêtre coulissante. b est utilisée pour pondérer l'importance de la taille des différents objets.

III.2.4.11. LRV [66]

Cette stratégie choisit l'objet avec la plus faible valeur relative, la fonction est la probabilité que le document est consulté à nouveau. Cette probabilité P_r est calculée comme suit:

$$P_r(f_i, T_i, s_i) =$$

$$\begin{cases} P(1, s_i)\left(1 - \check{D}(T_i)\right), si\ f_i = 1 \\ P(f_i)\left(1 - \check{D}(T_i)\right), \ sinon \end{cases}$$

$P(f_i)$ est la probabilité que l'objet i soit demandé $f_i + 1$ fois étant donné qu'il est demandé f_i fois. Pour les objets avec une seule demande, cette probabilité dépend de la taille de l'objet et est donnée par $P(1, s_i)$. $\check{D}(Ti)$ indique la répartition du temps Inter Access. Rizzo et Vicisano [2000] on montré des moyens efficaces pour calculer de manière adaptative $P(fi)$, $P(1, s_i)$ et les paramètres qui influent sur $\check{D}(Ti)$ (basé sur l'historique de l'accès précédent).

III.2.4.12. LUV [20]

Cette stratégie calcule pour chaque objet i de valeur V (i):

$$V(i) = W(i)p(i).$$

$W(i)$ est le coût relatif pour aller chercher l'objet à partir de son serveur d'origine et est défini par $W(i) = Ci / Si$.

$p(i)$ est la "probabilité" que l'objet *i* est référencé dans l'avenir. $p(i)$ est calculée par

$$p(i) = \sum_{k=1}^{f_i} F(t_c - t_k),$$

où t_{ck} est le temps réel et t_k est le temps de la plus ancienne demande dans une fenêtre coulissante après k demandes. $F(x)$ doit être baissée, pour donner plus de poids à des récentes références. Une possibilité est la suivante :

$$F(x) = \left(\frac{1}{2}\right)^{\lambda x} \quad (0 \leq \lambda \leq 1) .$$

III.2.4.13. LR(Logistic Regression)-Model [31]

L'objectif de ce modèle est d'exprimer les résultats d'une variable dépendante Y, en termes de prédicteurs, (1, X_1, ..., X_k) et de leurs coefficients respectifs (β_0, β_1, ..., β_k). La probabilité est donnée par LR

$$P_{LR} = P(Y = 1 \; ou \; 0 \mid 1, X_1, \dots, X_k) = \frac{1}{1-e^{-z}} \quad ,$$

où

$$z = \sum_{j=0}^{k} \beta_j X_j \quad -\infty < z < +\infty .$$

Étant donné un ensemble de données observées, les coefficients sont obtenus en utilisant la méthode d'estimation par maximum de vraisemblance (phase d'apprentissage). Une fois que ces coefficients sont connus, la probabilité P_{LR} peut être calculée pour les autres objets (phase de prévision).

Dans le contexte de la mise en cache, la manifestation d'intérêt est de savoir si un document est ré-accédé au moins une fois dans les prochains accès de N. Les prédicteurs sont des facteurs comme la récence, la fréquence.

III.2.4.14. The N-gram based replacement policy [88]

Tout comme la politique LRV, la politique N-gram prévoit également la probabilité qu'un document sera évalué à l'avenir. Pour évaluer la future fréquence d'accès $W(P)$ du document, qui est ensuite ajouté à la formule de la politique GDSF

$$K(P) = L + W(P) + F(P)\frac{C(P)}{S(P)}.$$

Lorsque l'espace de cache est insuffisant, le N-gram remplace d'abord le document avec la plus basse valeur $K(P)$.

III.2.4.15. Avantages

Cette classe inclue la plupart des stratégies de remplacement de cache proposées dans la littérature. Parmi les avantages de ces stratégies on a :

- Ces stratégies n'exigent pas une combinaison fixe des facteurs utilisés, à travers un choix approprié des paramètres, on peut optimiser n'importe quelle métrique de performance.
- Elles prennent en considération un certain nombre de facteurs qui permettent de manipuler différentes situations de charge de travail.

III.2.4.16. Inconvénients

Certaines propositions, associent des poids aux paramètres qu'elles utilisent pour le remplacement, mais le choix des poids appropriés pour un critère de performance est une tâche difficile.

III.2.5. Les stratégies de remplacement de cache dédiées aux environnements mobiles

III.2.5.1. La politique TDS (Time and Distance Sensitive) [52]

Une politique de remplacement de cache, Appelée TDS dédiée aux réseaux Ad hoc, a été proposée dans [52]. Elle prend en considération les facteurs de la distance et un facteur temporel pour le remplacement.

TDS se base sur deux facteurs pour choisir une victime à remplacer. Le premier est la distance δ, mesurée par le nombre de sauts par rapport aux points d'accès ou terminaux mobiles, qui

ont la donnée demandée. Puisque δ est étroitement lié à la latence, si la donnée élémentaire avec un δ plus élevé est choisie en tant que victime, alors la latence d'accès serait haute. Par conséquent, la donnée avec la moindre valeur δ est choisie en tant que victime. En raison de la mobilité des terminaux, la topologie de réseau peut changer fréquemment, et les valeurs δ peuvent devenir obsolètes. Par conséquent un deuxième paramètre τ est utilisé, qui capture le temps écoulé depuis la dernière mise à jour de δ. La valeur de τ est obtenue par

$\frac{1}{t_{cur}} - t_{update}$, où t_{cur} et t_{update} sont respectivement le temps courant et le temps de la dernière mise à jour de δ pour un document dans le cache. Trois variantes de TDS sont présentées, TDS_D, TDS_N et TDS_T qui choisissent comme victime le document ayant respectivement le minimum de $(\delta + \tau)$, $(\delta \times \tau)$ et τ.

L'inconvénient de cette politique de remplacement est qu'elle ne prend pas en considération la récence et la fréquence des accès.

III.2.5.2. La politique ZC (Zone coopérative)[18]

Dans [18], un schéma de gestion de cache appelé ZC (Zone coopérative) a été présenté, et dans le quel une politique de remplacement de cache basée sur le calcul d'une valeur a été proposée, où le document ayant la valeur la plus basse, est celui qui est enlevé du cache. Pour chaque document dans le cache, quatre facteurs sont considérés pour le calcul de cette valeur :

La popularité A_i: Elle est la probabilité d'accès d'un document se trouvant dans le cache. Un document avec une petite probabilité d'accès devrait être choisi pour le remplacement, donc le serveur enregistre pour chaque document sa probabilité d'accès A_i initialisée à zéro. Pour chaque accès à ce document, cette probabilité est mise à jour suivant cette formule :

$$A_i^{new} = \alpha \frac{1}{t_c - t_l} + (1 - \alpha) A_i^{old},$$

où t_c est le temps courant ;

t_l es le temps du dernier accès ;

α est un facteur constant pour peser l'importance de la récence d'accès.

La distance δ : Elle est mesurée en nombre de sauts entre le nœud demandeur du document et le nœud répondant (la source ou à partir d'un cache). Les documents avec une petite

distance sont favorisés pour le remplacement. Ce paramètre est particulièrement important pour la conservation de la bande passante et la réduction de la latence.

La cohérence : Un document d_i est valide pour une durée de vie limitée, qui est connue à partir du champ TTL_i. Les documents avec des petits TTL sont favorisés pour le remplacement.

La taille S : Les documents avec une plus grande taille sont choisis pour le remplacement. Dans ce cas le cache peut contenir plus de documents et ainsi peut satisfaire plus de demandes d'accès. Basée sur les facteurs cités ci-dessus, la fonction qui calcule la valeur associée à chaque document est donnée par la formule suivante :

$$W_1 A_i + W_2 \delta_i + W_3 TTL_i + W_4 IS_i ,$$

où w_1, w_2, w_3 et w_4 sont des poids associés à chaque facteur tels que :

$$\sum_{j=1}^{4} W_j = 1 .$$

Le document avec la plus petite valeur de cette fonction est choisi comme victime pour le remplacement.

Parmi les inconvénients de ZC est qu'elle nécessite un « Cache Admission Control » pour réduire le nombre de fichiers dupliqués dans une zone et la difficulté de fixer les poids W associés à chaque facteur. Aussi elle donne plus d'importance aux fichiers cachés dans toute la zone qu'en local.

III.2.5.3. Politique de remplacement de cache orientée énergie [51]

Une autre politique de remplacement de cache orientée énergie est proposée dans [51] ; Cette dernière étant dédiée à un schéma de caches coopératifs. Cette politique tente à minimiser la consommation d'énergie des accès de tous les nœuds mobiles à un document donné. Pour cela, cette politique calcule pour chaque document d_j son bénéfice Comme suit :

$$benefit(dj) = \sum_{i=1}^{q} P_{ij} E_{ij} ,$$

où

q est le nombre de nœuds mobiles ;

P_{ij} est la probabilité qu'un nœud i accède au document d_j ;

E_{ij} est le coût en énergie pour qu'un nœud i accède au document d_j.

Quant un document d_x doit être caché et que l'espace dans le cache est insuffisant cette politique sélectionne un ensemble V de documents à remplacer dans le cache telle que :

$$\sum_{j \in V} taille(d_j) \geq taille(d_x) \text{ , et que}$$

$$\sum_{j \in V} benefit(d_j) \text{ soit le minimum possible.}$$

Le gain en énergie après le remplacement sera:

$$gain(d_x) = benefit(d_x) - \sum_{j \in V} benefit(d_j).$$

III.2.5.4. La politique de remplacement de cache PIX [2]

Acharya a proposé une politique de remplacement de cache appelée ***pix***, elle est basée sur une fonction de coût (a cost-based caching algorithm) dans laquelle le document avec le plus petit coût est choisi comme victime pour le remplacement.

Soit d_i un document se trouvant dans le cache, son coût est calculé selon la formule suivante :

$$pix(d_i) = p_i / x_i \text{ ,}$$

où

- p_i est la probabilité d'accès ;
- x_i est la fréquence de diffusion du document d_i.

La motivation pour le remplacement basé sur ***pix*** est que la victime devrait être le document qui a la plus basse utilité local (c.à.d. la plus basse probabilité d'accès) et le plus bas coût de ré-acquisition (c.à.d. la fréquence de diffusion la plus élevée).

III.2.5.5. La politique Min-SAUD [86]

Une autre politique de remplacement de cache appelée Min-SAUD (Minimum Stretch integrated with Access rates, Update frequencies, and cache validation Delay) a été proposée dans [86], pour les environnements mobiles sans fil où la consistance doit être vérifiée avant l'utilisation du document caché. Elle est basée sur une fonction de gain qui prend en

considération plusieurs paramètres qui affecte le remplacement d'un document dans le cache, à savoir, *la probabilité d'accès, la fréquence de mise à jour, la taille du document, le temps de recherche, et le coût de validation de cache*. Cette fonction de gain est calculée pour chaque document d_i selon la formule suivante :

$$gain(i) = \frac{pi}{si}\left(\frac{bi}{1+xi} - v\right),$$

où

P_i est la probabilité d'accès au document i ;

S_i est la taille du document i ;

X_i est le rapport de taux de mise à jour sur le taux d'accès au document *i* ;

bi est le temps de récupération depuis le serveur (en cas de cache *miss*).

V est le délai de validation du cache.

L'idée de cette politique est de maximiser le gain total pour les documents maintenus dans le cache. Ainsi, pour trouver l'espace pour un document d_j au $K^{ième}$ accès, la politique Min-SAUD identifie comme victime un ensemble optimal de document V^*_k tel que :

$$V_k^* = min_{V_k \subseteq (cache)} \sum_{t \in V_k} gain(d_i) \text{ et}$$

$$taille(cache) \geq S_j + \sum_{d_i \in cache} S_i - \sum_{d_i \in V_k} S_i .$$

III.2.5.6. FSDV Frequency, Size and Distance based Value [48]

Dans ce qui suit, nous allons proposer une politique de remplacement de cache basée sur le calcul d'une valeur pour chaque document présent dans le cache qu'on appellera FSDV (Frequency, Size and Distance based Value). La fonction qui calcule cette valeur inclue quatre paramètres, la fréquence d'accès, la distance en nombre de sauts séparant le nœud qui répond à une demande d'un document de celui qui le demande, le temps écoulé depuis la dernière mise à jour de cette même distance et la taille de document. Donc pour chaque document d'identifiant id on calcule la fonction f(id) puis on choisit pour le remplacement le document ayant la plus petite valeur.

$$f(id) = \frac{frequence(id) \times distance(id)}{taille(id) \times age\_distance(id)} .$$

Mais l'utilisation de la fréquence d'accès pose le problème de la pollution de cache. Ce problème est posé lorsqu'un document était très populaire dans une période de temps, alors il peut demeurer toujours dans le cache même s'il reste non référencé pour une longue autre période de temps.

Pour remédier à ce problème, on utilise un autre paramètre, Age_Max qui représente le seuil maximum pour la moyenne de l'âge des documents présents dans le cache. Quand la moyenne dépasse ce seuil, toutes les fréquences des documents seront divisées par deux. L'inconvénient de cette politique est quelle fait abstraction sur la récence.

III.2.5.7. UMC (universal mobile caching) [71]

La politique Universal Mobile UMC propose un algorithme de remplacement d'auto-optimisation basée sur un ensemble de simples critères de base et utilisable pour la mise en cache en général. En ajustant automatiquement le poids de différents critères, la politique UMC peut s'adapter à divers environnements structurellement et l'évolution des charges de travail.

Cette formule peut être exprimée comme suit:

$$H(p) = W_L.\log(L(p)) + W_C.\log(C(p)) - W_B.\log(B(p)),$$

L(p): Probabilité d'accès à p ;

C(p): Coût de la récupération de p ;

B(P): Bénéfice d'expulser p.

L'Universal Mobile cache algorithme s'appuie sur ce petit jeu des propriétés de l'objet (la probabilité L(p), le coût C(p), bénéfice B(p)), et se combine directement à fournir une auto-optimisation de la mise en cache globale.

L'estimation de la probabilité de l'accès futur a été faite en utilisant deux estimateurs différents: la fréquence d'accès, et la récence du dernier accès. Sur la base de la performance des critères composant, la distribution des crédits pour chacun des critères est de la manière suivante :

$$Credit_c = (Per\ f_0 - Per\ f_i).Credit_p,$$

où

$Credit_c$: le crédit en cours pour les critères ;

$Credit_p$: le précédent crédit pour les critères ;

Per f_0: la combinaison générale du système (Taux de succès, Taux de succès par octets) ;

Per f_i: la combinaison (Taux de succès, Taux de succès par octets).

Les crédits sont répartis entre les critères après chaque évaluation de leur performance relative. Quand un critère dégrade la performance du cache, les crédits seront redistribués de sorte que cela ne dégrade pas les performances globales.

L'inconvénient de cette politique est quelle nécessite des calculs périodiques de la performance globale, ce qui impose un coût supplémentaire en terme d'énergie et de la bande passante.

III.2.5.8. SXO (Size*Order) [90]

Cette politique se concentre sur deux paramètres qui sont plus faciles à obtenir. Le premier paramètre est la taille des données. Si le second paramètre est ordre(i), l'ordre de (i) selon les intérêts d'accès. Soit *N=Ordre(i),* avec *i* est la $N^{ième}$ données fréquemment consultées. Intuitivement, les données qui sont moins susceptibles d'être accessibles doivent être remplacées en premier. La politique de remplacement (Taille * Ordre) (SXO), combine ces deux paramètres dans la fonction de valeur suivantes:

$$Value\ i = Si \square Ordre\ i\ ,$$

L'élément de données avec la plus grande *value i* est remplacé en premier, Bien que *Order i* n'est pas disponible, il peut être dérivé à partir du taux d'accès à *i*, comme suit:

$$Ordre\ i = a_i = \frac{K}{T - T_{a_i}(K)}\ .$$

où *T* est le temps actuel et $T_{ai}(K)$ est le moment du $K^{ième}$ plus récents accès.

L'inconvénient majeur de cette politique est quelle néglige l'impact de la distance des nœuds.

III.2.5.9. OBS (On-Bound Selection) [20]

Elle n'est pas limité à l'accès aux données sans fil, elle est également applicable aux réseaux câblés tels que les applications client-serveur Internet et la mise en cache Web.

$$OBSF_{ij}(t) = \frac{(f_{ij}^{a}(t))^2}{f_{ij}^{a}(t)+f_{j}^{u}(t)},$$

$$OBSF_{ij} = \lim_{t\to\infty} OBSF_{ij}(t) = \frac{(\mu_{ij})^2}{\mu_{ij}+\lambda_j}.$$

où

$f_{ij}^{a}(t)$: La fréquence d'accès à l'objet Oj par le nœud *i* au moment *t*

$f_{j}^{u}(t)$: La fréquence de modifications de l'objet Oj par le serveur au moment *t*

Dans la politique de remplacement OBS, l'algorithme maintient l'historique d'accès au cache et la mise à jour des données, qui peut être utilisé pour calculer le facteur OBS.

La politique de remplacement calcule les facteurs OBS de l'ensemble des données du cache et les données nouvellement accessibles, puis cherche l'élément avec le plus petit facteur d'OBS. Si les données avec le plus petit facteur OBS ne sont pas les données nouvellement accessibles, l'élément correspondant est remplacé par les données nouvellement accessibles, dans le cas contraire, les données consultées ne seront pas mises en cache.

III.2.5.10. General_Opt [91]

Les notations suivantes sont utilisées :

- n : le nombre d'éléments de données dans la base de données.
- fi: le coût de l'extraction de donnée *i* du cache.
- c: le coût moyen de valider la cohérence des données d'articles dans le cache.
- vi: le coût de l'obtention de la mise à jour de la donnée *i* à partir du serveur.
- ai: le taux d'accès moyen à la donnée *i*.
- ui: le taux de mise à jour de donnée *i*.
- Si: la taille de la donnée *i*.
- Pai : la probabilité de référencier la donnée *i*.
- Pui : La probabilité d'invalider la donnée *i* du cache.

- V: l'ensemble de tous les éléments de données mises en cache.

Sur la base des notations ci-dessus, le remplacement General_Opt du cache devrait optimiser l'expression suivante:

$$max \sum_{i \in V} value(i) ,$$

où

$$value(i) = P_{ai}(f_i - c - P_{ui} \cdot v_i).$$

Cette fonction peut s'expliquer par un modèle de coût d'accès au cache. L'algorithme General_Opt remplace l'ensemble des articles qui minimisent le coût total d'accès.

III.2.5.11. GroupCaching [92]

Elle considère les caches des nœuds voisins (d'une zone d'un seul saut) comme un seul cache, et les nœuds appartenant à cette zone forment un groupe. Quand un nœud reçoit un objet de données de la destination, chaque nœud sait au préalable le reste de l'espace du cache disponible dans chaque mobile de son groupe et l'*id* et la *date d'entrée* de leurs objets mises en cache. Tout d'abord, quand un nœud reçoit un objet, il le met en cache si est suffisant. Sinon, le récepteur vérifie les espaces des caches disponibles de ses membres du groupe.

Si l'espace du cache disponible de quelques membres du groupe est suffisant pour stocker l'objet, le récepteur choisit aléatoirement un nœud pour mettre l'objet dans son cache.

Deuxièmement, si l'espace disponible du cache de tous les membres du groupe n'est pas suffisant pour mettre en cache l'objet reçu, le récepteur recherche dans le groupe pour voir s'il existe un membre qui a déjà cette donnée en cache. Si c'est le cas, l'objet n'est pas mis en cache. Sinon, le nœud récepteur sélectionne le membre qui a les plus anciennes données mises en cache dans le groupe et lui envoie l'objet pour faire le placement.

Quand un membre de groupe reçoit un objet du récepteur, il effectue à plusieurs reprises les opérations de remplacement LRU jusqu'à l'augmentation de l'espace du cache disponible qui permet de mettre en cache l'objet.

L'inconvénient de cette politique est quelle néglige l'importance des fichiers populaires.

III.3. Proposition d'une politique de remplacement de cache pour les réseaux Ad hoc

La conception des politiques de remplacement de caches destinées aux réseaux mobiles Ad hoc doit prendre en considération, non seulement les caractéristiques de ces réseaux, mais aussi celles de leur trafic (comme la fréquence, la récence des documents et leurs tailles) pour augmenter les performances des caches en local, ainsi que l'intégration d'un schéma de coopération afin d'améliorer les performances globales du système, que ce soit en économie de l'énergie ou en réduction de la latence globale.

Dans tous les travaux qu'on a consulté, La plupart des politiques proposées dans le cadre des réseaux mobiles Ad hoc, est basée sur des schémas de coopération, afin d'améliorer les performances globales de ces réseaux.

On prend comme exemple les politiques suivantes :

- ✓ La politique proposée dans le schéma de coopération de cache appelé ZC (Zone Coopérative) [18] a essayé d'inclure tous les paramètres par sommation et en associant pour chacun un poids. Cependant, dans cette politique, la difficulté réside dans la détermination optimale des poids pour le remplacement. De même, la coopération se limite au processus de la recherche de document et le système de contrôle du cache.
- ✓ La politique TDS [52] utilise la distance en nombre de sauts et un paramètre temporel, mais elle ne prend pas en considération la fréquence et la taille des documents qui sont des paramètres très importants et influent considérablement sur les performances des caches. Et la coopération se limite au processus de recherche d'un document et l'agrégation des données par le système de contrôle du cache.
- ✓ La politique de remplacement de cache (SXO) [90], combine ces deux paramètres (Taille, Ordre) et le schéma de coopération *cache path et cache data* pour garantir l'agrégation et le maintien du plus court chemin, Néanmoins, elle ne garantie pas la répartition des données dans le réseau.
- ✓ La politique de remplacement de cache GroupCaching [92] est purement coopérative, Ce qui assure la répartition des données sur le réseau, l'inconvénient est qu'elle néglige l'impact de la mobilité et la satisfaction des caches en local.

La table ci dessous résume le genre de coopérations adoptées pour chaque politique et ses inconvénients.

La politique de remplacement	Type de coopération	Inconvénient
ZC	Coopération lors de la recherche d'un document, et la mise en cache est contrôlée par le système de contrôle du cache.	Pas de coopération entre les caches de la même zone lors de remplacement.
TDS	Coopération lors de la recherche d'un document et l'agrégation des données par le système de contrôle du cache.	Elle ne prend pas en considération la fréquence et la taille des documents.
SXO	Elle intègre un schéma de coopération *cache path et cache data* pour garantir l'agrégation des données.	Elle ne garantie pas la répartition des données dans le réseau.
GroupCaching	purement coopérative et elle assure la répartition des données.	Elle néglige l'impact de la mobilité et la satisfaction des caches en local.

Table III.2 : Le genre de coopérations adoptées dans quelques politiques et ses inconvénients

Dans ce qui suit, nous allons proposer une politique de remplacement de cache basée sur le principe de répartition des données dans le réseau, tout en gardant le maximum de satisfactions des caches en local, qu'on appellera **Mobile Zone Serveur (MZS).** Cette politique permettra d'appliquer une stratégie basée sur l'importance d'un objet en local et dans sa zone coopérative (d'un seul saut) définie par :

$$H_i = \frac{f_i}{S_i} + NBclients(i)\ ,\ \text{où}$$

f_i : la popularité de fichier *i* calculé comme suit :

$f_i = \frac{K}{T-Ti}$, où *T* est le temps actuel et T_i est le moment du $K^{ème}$ plus récent accès.

T-Ti représente l'âge du fichier ;

K représente la fréquence du fichier ;

S_i : la taille du fichier.

$NBclients(i)$: le nombre de voisins d'un seul saut qui ne possèdent pas le fichier *i* dans leurs caches.

Pour faire un compromis entre l'importance d'un fichier dans sa zone et son importance en local, l'intégration d'une valeur *NBclients(i)* directement dans la politique de remplacement peut résoudre ce problème.

La valeur *NBclients(i)* diminue les duplications automatiquement dans une zone puisque un fichier dupliqué aura une petite valeur de *NBclients(i)*, augmentant ses chances d'être retiré du cache.

Cacher un fichier même s'il est dupliqué dans sa zone permet de garder ses chances d'être ré-accédé une autre fois au futur proche en local.

La mobilité pose un grand problème dans les politiques de remplacements du cache adoptées pour les réseaux Ad hoc, Cependant, dans notre cas, le nœud qui change de zone garde ses fichiers les plus récents et les plus populaires en cache (maximum de satisfaction locale) qui peuvent être importants dans la nouvelle zone franchie.

Notre politique garantit que chaque nœud mobile aura le maximum de clients pour ses fichiers locaux y compris lui-même, donc il se comporte comme un serveur de fichiers mobile dans sa zone ; c'est la raison pour laquelle nous avons choisi le nom de cette politique (**Mobile Zone Serveur**).

III.4. Conclusion

La problématique principale des politiques de remplacement du cache concerne le choix des documents qui doivent être remplacés quand l'espace libre dans le cache devient insuffisant. Plusieurs politiques ont été proposées dont l'objectif est de bien choisir les documents inutiles et il s'est avéré que les politiques qui utilisent un schéma coopératif et combinent deux ou plusieurs paramètres donnent de meilleures performances.

Dans ce chapitre, nous avons présenté la majorité des politiques de remplacement. Ensuite, nous les avons classées selon les paramètres utilisés. Nous avons également proposé une politique de remplacement de cache pour les réseaux mobiles Ad hoc dans le but d'améliorer les performances de ce dernier.

Dans le chapitre suivant, nous allons présenter quelques méthodes d'évaluation de performances, ainsi que quelques caractéristiques du trafic Web. Ensuite, nous passons à la description du modèle de notre système et de quelques caractéristiques du trafic. Nous terminons par la présentation des résultats de simulation de notre politique de remplacement de cache **Mobile Zone Serveur (MZS)** en comparaison avec quatre autres politiques **LFU**, **LRU**, **ZC** et **GroupCaching**.

Chapitre 4

Caractérisation du trafic et évaluation de performances

IV.1. Introduction

Les performances de n'importe quelle politique de remplacement de cache dépendent du trafic Web ou plus précisément des caractéristiques de ce trafic, de la capacité du cache et de la politique de remplacement elle-même. C'est pour cela que toute étude des techniques et mécanismes utilisés dans le cadre des caches Web doit commencer par la compréhension des propriétés et caractéristiques des documents Web à cacher.

Les critères d'évaluation de performances des politiques de remplacement sont liés aux objectifs de l'utilisation du cache dans le Web, tels que la réduction du trafic dans les réseaux, la minimisation de la charge des serveurs et la réduction des latences des clients, etc. Tous ces objectifs nous aident à mettre en évidence les paramètres de mesure de performances des politiques de remplacement.

Dans ce chapitre, nous allons présenter les différentes méthodes d'évaluation de performances existantes, ainsi que les principales caractéristiques du trafic. Nous prêtons une particulière attention aux aspects pertinents liés aux caches. Ensuite, nous allons présenter le modèle général du système à étudier, l'environnement de simulation ainsi que quelques résultats d'études comparatives de notre politique de remplacement de cache avec d'autres politiques existants.

IV.2. Les méthodes d'évaluation de performances

Avant d'entamer la caractérisation du trafic, il sera utile de décrire les principales méthodes d'évaluation de performances qui sont utilisées pour évaluer les nouvelles

technologies. Ces méthodes d'évaluation de performances sont réparties en quatre grandes catégories : mesures directes, analyse mathématique, simulation basée sur les traces, simulation basée sur le Benchmarking.

IV.2.1. Mesures directes

Les mesures directes concernent les méthodes d'observation et d'auscultation du comportement de Web en temps réel. Elles sont particulièrement utiles pour évaluer l'exécution réelle du Web. Mais il est souvent très difficile de mettre en application ces méthodes. Par exemple, observer l'exécution de bout en bout comme vue par de vrais utilisateurs exigerait d'équiper leurs navigateurs. Les mesures directes sont généralement non répétables, puisqu'il est impossible de reproduire entièrement les conditions dans lesquelles les mesures initiales ont été prises.

IV.2.2. Analyse mathématique

Cette technique d'évaluation de performance consiste à réduire le système étudié en un modèle mathématique et de faire par la suite de l'analyse numérique. Autrement dit, elle permet de représenter le système en utilisant des formalismes mathématiques. Ces formalismes mathématiques modélisent la dynamique du système, en utilisant un ensemble de variables liées par des équations d'évolution. Cette technique est un moyen rapide et efficace.

Il existe plusieurs formalismes et méthodes souvent utilisés pour évaluer les performances de réseau, on peut citer, entre autres : les méthodes de graphes, les réseaux de Petri et les files d'attente.

Toutefois, la modélisation analytique implique un niveau d'abstraction considérable, on ne peut pas utiliser ces méthodes sans faire beaucoup de suppositions simplificatrices qui peuvent compromettre la qualité des résultats.

IV.2.3. La simulation

La simulation s'impose comme la technique la plus utilisée dans le domaine d'évaluation de performance des systèmes informatiques, à cause essentiellement du fait que les modèles analytiques ont des limitations dans la gamme de comportements qu'elles peuvent modéliser.

Simuler un système consiste à expérimenter sur un modèle en exécutant plusieurs trajectoires possibles.

IV.2.3.1. La simulation basée sur les traces

Les traces de Web sont des enregistrements d'informations telles que le temps d'arrivée des requêtes, les URLs demandés, la taille des réponses, et les informations d'entêtes. Les fichiers logs des accès collectés par les serveurs Web représentent un exemple de ces traces. Ces fichiers logs peuvent être aussi collectés auprès des navigateurs ou proxys.

Les différentes propriétés du trafic, telles que la distribution des tailles de documents, ou les intervalles de temps des inter-arrivés, peuvent être inférés depuis les traces collectées de différentes manières, c'est ce qu'on appelle le processus d'analyse de traces. D'autres expériences utilisent ces traces d'accès pour simuler le comportement de diverses techniques et mécanismes de Web. Ces expériences s'appellent les trace-driven simulations.

Généralement n'importe quelle trace manifeste des propriétés particulières qui lui sont propres [34]. Une trace des accès de cache d'un proxy d'un laboratoire ou d'une université de recherche peut présenter des propriétés tout à fait différentes d'une trace semblable des utilisateurs résidentiels ; les traces obtenues pendant les jeux olympiques peuvent être différentes des traces des turbulences de marché boursier. Donc la question qui se pose est comment essayer de reproduire une trace pour un comportement plus général.

IV.2.3.2. Simulation basée sur le benchmarking

Benchmarking est l'utilisation d'un trafic synthétique, artificiellement généré pour tester les aspects spécifiques des équipements de Web ainsi que leurs comportements.

L'ultime but de la synthétisation de trafic Web est d'imiter le comportement réel du Web aussi étroitement que possible. Ce trafic est généré en analysant de vraies traces, puis en extrayant les caractéristiques du trafic, et en reproduisant ces caractéristiques dans un trafic synthétique.

Cette reproduction de trafic synthétique peut exiger de faire des suppositions au sujet du comportement du Web, par exemple en plus de la distribution de popularité, on peut également avoir besoin de la corrélation des références, c'est pour cela qu'il est difficile de décider quand cesser de considérer de plus en plus des aspects du comportement de Web.

Parfois, il est souvent nécessaire d'évaluer les systèmes complets dans des conditions de surcharge. Par exemple, pour découvrir comment un server Web se comporte quand il reçoit plus de requêtes que ce qui est fournie par des traces disponibles. Dans ces situations, l'utilisation du trafic synthétique devient inévitable.

IV.3. Caractérisation du trafic Web

Dans la littérature, plusieurs travaux ont été consacrés à la caractérisation du trafic Web [80], [79], [9], [55] ainsi qu'à l'étude des performances des caches [10], [1]. Parmi les caractéristiques communes du trafic Web identifiées on a :

- La distribution de la popularité des documents Web suit une loi Zipf-like ;
- La distribution de la taille des documents suit une loi heavy-tailed (Pareto) ;
- La localité de références manifestées par les flux de requêtes.
- Le pourcentage élevé des documents référencés une seule fois (les one-timers).
- Le taux de modification des documents

IV.3.1. La popularité des documents

Le comportement des politiques de remplacement est très influencé par la distribution de la popularité des documents, puisque ceux qui sont populaires tendent à rester dans les caches. Plusieurs chercheurs ont observé que la fréquence relative avec laquelle les pages Web sont demandées suit la loi de Zipf [64]. La popularité indique qu'il existe un ensemble de documents plus demandés alors que le reste des documents est moins demandé. Ce type d'accès existe dans plusieurs systèmes tels que : le modèle d'accès mémoire au programmes machine [61], [14], la popularité des mots utilisés dans la langue anglaise[2], et la popularité des livres empruntés au niveau d'une bibliothèque publique [95],

La loi Zipf est une loi statistique très célèbre qui est observée dans le comportement de plusieurs systèmes complexes de nature différente. C'est une description du rapport entre la fréquence de l'occurrence d'un événement et son rang.

Surnommé par le professeur George Kingsley Zipf (1902-1950) de Harvard, cette loi a été à l'origine appliquée au rapport entre les mots dans un texte et leur fréquence d'utilisation.

Elle affirme que la probabilité relative d'une requête pour le i^{eme} plus populaire document est proportionnelle à 1 /i, alors cette relation est exprimée de la façon suivante :

$$p_i = k\, i^{-1}$$

où *K* est une constante de proportionnalité.

Glassman [51] était peut-être le premier à utiliser la loi Zipf pour modéliser la popularité des pages Web, puis, plusieurs d'autres auteurs ont également appliqué cette loi [9], [79], [22], [55]. Cependant, plusieurs études sur l'applicabilité de la loi Zipf aux documents Web, ont conclu que la distribution de la popularité des documents Web suit généralement une distribution Zipf-like [5], [41] dont la probabilité relative d'une requête pour le i^{eme} plus populaire document est proportionnelle à $1/i^{\beta}$ avec β compris entre 0 et 1. Donc la distribution de la popularité des documents Web ne suit pas la loi Zipf strict (pour la quel β=1), mais plutôt une loi plus général Zipf-like, avec le paramètre β qui varie d'une trace à l'autre mais qui est proche de 1. Selon cette loi, la popularité p_i d'un document de rang *i* est donnée par la formule suivante :

$$p_i = k\, i^{-\beta} \text{ avec } 0< \beta <1.$$

Où k est une constante de proportionnalité.

IV.3.2. La taille des documents

Dans l'évaluation de performance des caches Web, et contrairement au cache mémoire (où les documents sont de même taille), la distribution des tailles des documents Web joue un rôle important dans la mesure où elle influe directement sur les performances des politiques de remplacement de cache. En effet, un cache d'une taille donnée aura la capacité de contenir plus de documents de petite taille, ce qui implique un taux de succès élevé [22]. Cependant il ne peut contenir que moins de documents de grande taille, ce qui implique un petit taux de succès mais éventuellement un grand taux de succès en octets.

Plusieurs études des caractéristiques du trafic ont montré que cette distribution est de type heavy-tailed [37], [43], [42]. Une distribution est dite heavy-tailed si :

$$p[X > x] \approx x^{-\square} \qquad x \to \infty, 0 < \alpha < 2.$$

Si la forme asymptotique d'une distribution est hyperbolique alors elle est heavy-tailed, indépendamment de son comportement pour les petites valeurs de la variable aléatoire.

Une forme simple de la distribution heavy-tailed est la distribution de Pareto. Sa fonction de densité de probabilité est donnée par :

$$p(x) = \alpha k^{\square} x^{-x-1} \qquad \alpha, k > 0 , x \geq k.$$

Et sa fonction de distribution cumulative est donnée par :

$$F(x) = P[X \leq x] = 1 - (k/x)^{\alpha} .$$

Le paramètre *k* représente la plus petite valeur possible de la variable aléatoire *x*, donc *k* est la plus petite taille que peut prendre un document ; aussi si *k* augmente les petites valeurs de taille de documents sont ignorées.

IV.3.3. La localité des références

La localité des références caractérise la capacité de prévoir de futurs accès aux documents à partir des accès passés, elle est exploitée pour développer les politiques et les mécanismes liés aux caches, pour décider quels sont les documents à garder ou enlever du cache, quels sont les documents à pré-charger, c'est-à-dire, les charger dans le cache avant qu'il soient accédés.

Il y a deux types de localité : temporelle et spatiale.

IV.3.3.1. La localité temporelle

La localité temporelle se réfère à la tendance qu'un même document soit re-référencé fréquemment dans un petit intervalle de temps, autrement dit, un document qui vient d'être référencé est plus probable qu'il le soit encore dans le futur proche [22].

La localité temporelle est modélisée à l'aide d'une pile LRU appelé LRU stack model ou LRUSM [55], cette pile est utilisée pour stocker les documents les plus récemment accédés, et chaque position dans la pile est associée à une probabilité de re-référencement qui ne dépend que de cette position et non du document qui l'occupe [15].

Il existe deux modèles différents de la localité temporelle, le modèle statique et le modèle dynamique, dans le cas statique les probabilités sont associées préalablement et d'une façon permanente aux différents positions dans la pile. Dans le modèle dynamique, ces probabilités sont calculées dynamiquement en fonction des fichiers occupant actuellement chaque position dans la pile [15].

IV.3.3.2. La localité spatiale

La localité spatiale exprime le fait que certaine accès à quelques documents nécessitent fréquemment des accès à d'autres documents [79]. Par exemple, un accès à une page HTML peut nécessiter des accès aux documents encastrés ou hyperlinkés dans cette page. La localité spatiale implique que le référencement d'un document peut être un facteur prédictif de future référence.

IV.3.4. Les documents référencés une seule fois (one-timers)

Ce sont les documents qui ne sont accédés qu'une seule fois, dans la littérature ils sont communément appelés les one-timers. Des études ont montré que ces documents constituent un grand pourcentage du trafic Web (de 50 % jusqu'à 70 %) [55], [41], [22], [37]. Il est évident que ces documents ne doivent pas être cachés car en réalité ils ne seront jamais accédés dans le futur, et ils ne font que réduire les performances des caches. Par conséquent, les politiques de remplacement de cache doivent se doter de mécanisme de contrôle d'admission au cache et avoir la capacité de distinguer ces documents parmi d'autres.

Du point de vue positif, les one-timers peuvent être considérés comme une conséquence logique et inévitable de l'efficacité des mécanismes et techniques utilisés dans le cadre des caches Web comme les politiques de remplacement, le perfetching, la réplication, les proxys, etc. L'un des objectifs de ces techniques est de réduire la charge des proxys et des serveurs originaux, donc les one-timers s'expliqué par le fait que ces derniers ne sont pas atteint par les requêtes répétées pour le même document parce qu'elles étaient servies au niveau des caches.

IV.3.5. Le taux de modification des documents

Une autre caractéristique des documents Web dont dépend l'efficacité du cache est le taux de mise à jour des documents Web. Le taux de mise à jour est important parce qu'il affecte les mécanismes qui assurent qu'un cache ne sert pas un contenu obsolète aux clients.

Pour caractériser l'effet des changements de documents sur les caches, Douglis et al dans [63] ont étudié le taux de modification, la fraction d'accès ayant retourné des documents qui ont été modifiés depuis leurs derniers accès. Parmi toutes les réponses sur les quelles il était possible de déterminer si les documents ont été modifiés, le taux de modification était autour de 15 %. En d'autres termes, 15 pour cent de requêtes ne pourraient pas être satisfaites depuis le cache par les vieilles copies de documents car ces derniers ont été changés.

IV.3.6. Le type des documents

Actuellement la majorité écrasante de tous les téléchargements, en termes de type et de nombre de documents accédés, implique des pages HTML et des images. Ensemble, elles comptent autour de 90 % de tous les accès au Web, et plus de 70 % de volume téléchargé, [63], [84], [4], [37]. Les documents multimédias forment très peu d'accès mais un pourcentage en rapport considérable des tailles téléchargés : 14 % observé dans [84], 24 % dans [37], et jusqu'à 6 % dans [63].

IV.4. Modélisation et génération de trafic

Parmi les étapes de toute étude de performance, la modélisation constitue l'étape la plus importante et la plus cruciale. En effet, elle permet de représenter les différents aspects à étudier en faisant abstraction du système réel. Dans notre cas, il s'agit de modéliser une politique de remplacement du cache dans le cadre des réseaux mobiles Ad hoc. Ensuite d'utiliser la simulation pour prouver l'exactitude et l'efficacité de notre proposition en la comparant aux politiques de remplacement de cache traditionnelles et à d'autres politiques qui ont été proposées dans le cadre des environnements mobiles, et ce, en terme de taux de succès (hit rate) des accès aux caches, de taux de succès en octets (byte hit rate) et de taux de succès en zone (zone hit rate).

IV.4.1. Modèle de simulation

Notre modèle de simulation fonctionne comme suit. À l'arrivée d'une requête pour un document au niveau d'un nœud, ce dernier consulte son cache, si le document existe alors pour assurer la consistance, il vérifie la fraicheur de ce document grâce au TTL qui lui est associé. Si le document est frais (c'est un hit), le document est retourné directement en réponse au demandeur. Si le document n'existe pas dans le cache ou bien il n'est pas frais alors la requête est acheminée vers la source de ce document. Quand un nœud reçoit une

réponse pour un document, si l'espace libre dans le cache est suffisant pour recevoir ce document, alors il est inséré directement dans le cache, sinon la politique de remplacement de cache est invoquée pour libérer de l'espace pour ce nouveau document. La figure ci-dessous montre le comportement du cache recevant une requête pour un document.

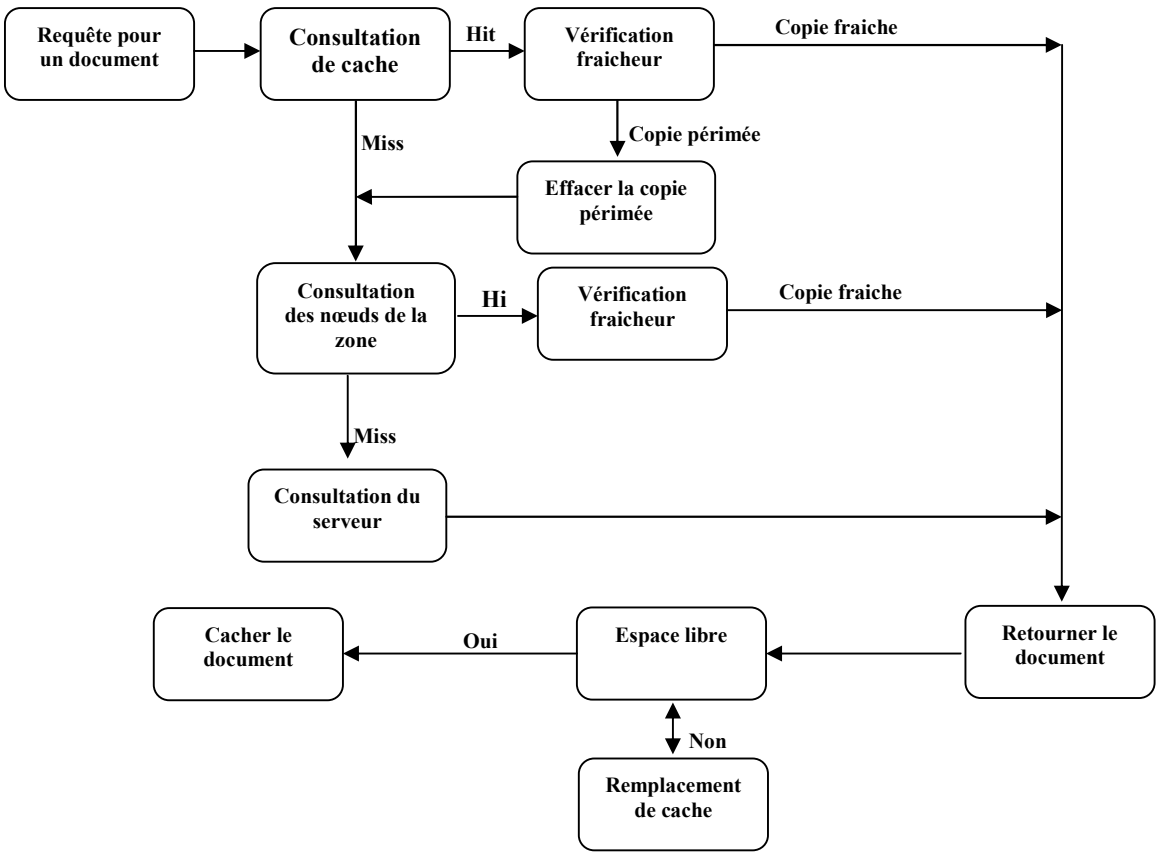


FIG IV. 1 : Comportement du cache recevant une requête pour un document.

IV.4.2. Modèles de Mobilité [30]

Il existe plusieurs modèles de mobilité utilisé dans des simulateurs des réseaux mobile Ad hoc. Tous ces modèles de mobilité expriment le mouvement et les arrêts imprévisibles des nœuds.

IV.4.2.1. Mobilité Déterministe

Ces conventions étant établies, l'algorithme de la mobilité déterministe est simple. A chaque quantum de temps on avance d'une distance v (la vitesse).

- ✓ On construit l'instance :

  Position Courante = position Initiale fournie par l'utilisateur;
  positionVisée = positionVisée fournie par l'utilisateur;
  v = vitesse fourni par l'utilisateur;
- ✓ exécution de l'algorithme :

  // On projette sur l'axe x grâce au cosinus de l'angle alpha et on l'ajoute a l'ancienne position

  x = x + v* cos(alpha);

  // On projette sur l'axe y grâce au sinus de l'angle alpha

  y = y + v * sin(alpha);

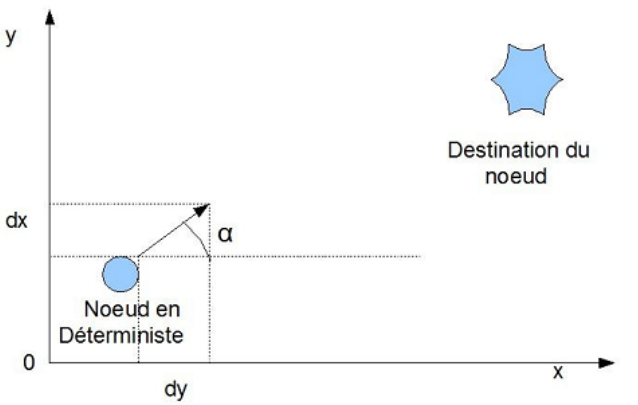


FIG IV. 2 : La mobilité déterministe

IV.4.2.2. Mobilité RandomWalk

Dans RandomWalk l'utilisateur est amené à fixer les bornes de chaque paramètre fixé au hasard. RandomWalk génère selon ces paramètres une position vers laquelle il se dirigera. Arrivé à cette position il en générera une suivante ainsi de suite jusqu'à la destruction du nœud ou la modification de sa mobilité.

IV.4.2.3. Mobilité RandomWayPoint

Réagit comme RandomWalk à l'exception qu'il tire un temps d'attente au hasard entre minAttente et maxAttente à chaque fois qu'il atteint un point visé.

IV.4.2.4. Mobilité Poursuite

Mobilité poursuite est composée d'une poignée Mobilité Déterminisme et d'une poignée Mobilité RandomWalk.

IV.4.3. La cohérence de cache

En raison d'économie de l'énergie et de la bande passante dans les réseaux sans fil, la plus part des politiques de remplacement de cache des réseaux ad-hoc, préfèrent utiliser la faible cohérence, en attribuant à chaque document caché un quantum de temps (TTL), les objets en cache sont supprimées si le temps de la durée de vie (TTL) expire [39].

IV.4.4. La génération du trafic

L'utilisation de la simulation pour l'évaluation de performances des politiques de remplacement de cache nécessite la génération d'un trafic représentatif. Dans la littérature, il existe plusieurs outils dont le but est d'étudier les caractéristiques de trafic Web et de reproduire ces caractéristiques sur un trafic représentatif. Dans notre étude, nous avons généré un trafic synthétique similaire à l'outil réalisé par N. Markatchev et C. Williamson en 2002, le WebTraff et plus précisément le package ProWGen (Proxy Workload Generator) qui est un outil basé sur une approche analytique pour la génération de trafic d'un proxy Web. Et on a implémenté un système de détection de voisinage d'un seul saut (HELLO Message), ainsi qu'un générateur des TTLs (Time To Live) associés aux documents. Ces TTLs sont utilisés par les politiques de remplacement de cache pour assurer la consistance des documents.

IV.4.5. Environnement de simulation

On a développé un simulateur orienté processus ou chaque nœud est représenté par un thread, chaque thread a une liberté de mobilité dans une interface déterminée, et qui lance périodiquement des requêtes à des fichiers selon une loi zipf, la génération de notre trafic zipfien est effectuée à l'aide d'une pile de probabilité. L'implémentation de notre simulateur a été faite en Java (Eclipse), sous le système d'exploitation Windows. Nous avons comparé notre politique, en premier, avec la même politique mais sans le paramètre ***NB_clients*****,** pour montrer l' apport de ce en terme de *hit_zone,* et nous l'avons comparée aussi avec quatre autres politiques de remplacement de cache que nous avons implémentées sur le même simulateur, deux politiques classiques LRU (Least Recently Used) et LFU (Least Frequently Used), et deux politiques proposées dans le cadre des réseaux mobiles Ad hoc GroupCaching

[42] et ZC (Zone coopérative) [84]. La comparaison a été faite en termes de deux paramètres, le taux de succès, le taux de succès en octets dans une zone.

IV.4.6. Validation du simulateur

Pour valider les simulateurs des politiques de remplacement de cache cités ci dessus nous avons utilisé des fichiers de courtes traces sur lesquels il était possible de suivre l'exécution des politiques de remplacement. Ensuite, pour chaque politique, on lance son exécution tout en affichant pour chaque remplacement effectué l'état du cache avant et après le remplacement, puis on vérifie si le document remplacé était vraiment celui qui faudrait être choisi par la politique de remplacement simulée. Cette procédure a été répétée plusieurs fois pour chacune des politiques de remplacement de cache.

IV.4.7. Les caractéristiques statistiques du trafic généré

Dans ce qui suit, nous allons présenter quelques caractéristiques statistiques du trafic généré. Pour cela, nous avons pris un échantillon de 1000 documents pour lesquels nous avons mis en évidence le changement de leurs popularités en fonction de leurs rangs, la distribution de leurs tailles, et enfin la distribution des TTLs qui leurs sont associés.

IV.4.7.1. La popularité des documents

La figure **FIG IV.3** montre le changement des popularités des documents en fonction de leurs rangs, ce qui confirme bien que la popularité suit une loi zipf. Nous avons choisi de présenter ce graphe avec l'échelle logarithmique pour mieux visualiser les résultats.

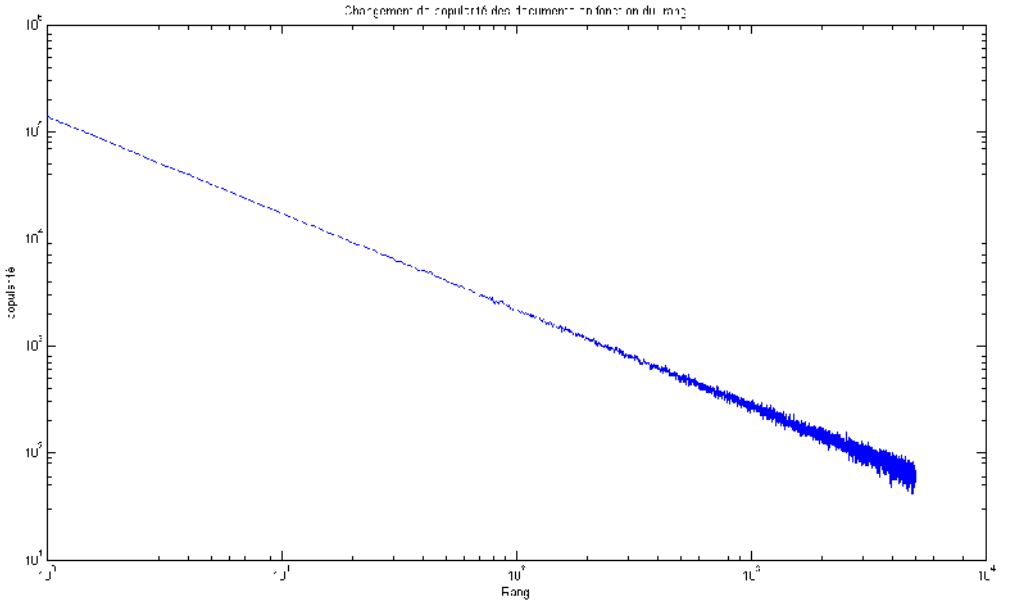


FIG IV.3 : La popularité des documents

IV.4.7.2. La distribution des tailles de documents

Comme nous l'avons mentionné précédemment, la distribution des tailles des documents de trafic suit une loi heavy-tailed avec α=0.7 et la taille minimale=1000. La figure **FIG IV.4** montre la variation des tailles de documents de l'échantillon choisi.

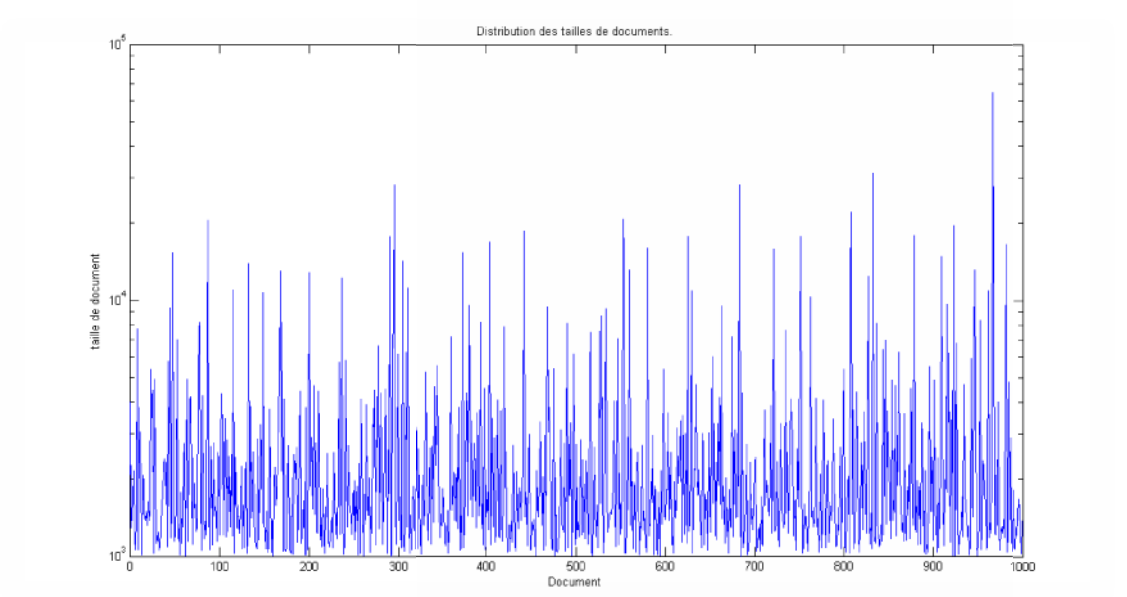


FIG IV. 4 : La distribution des tailles de documents

IV.4.7.3. La distribution des TTL de documents

La figure **FIG IV.5** montre la distribution des TTL de 1000 documents qui ont été générés selon une loi exponentielle de paramètre λ=1000 ms.

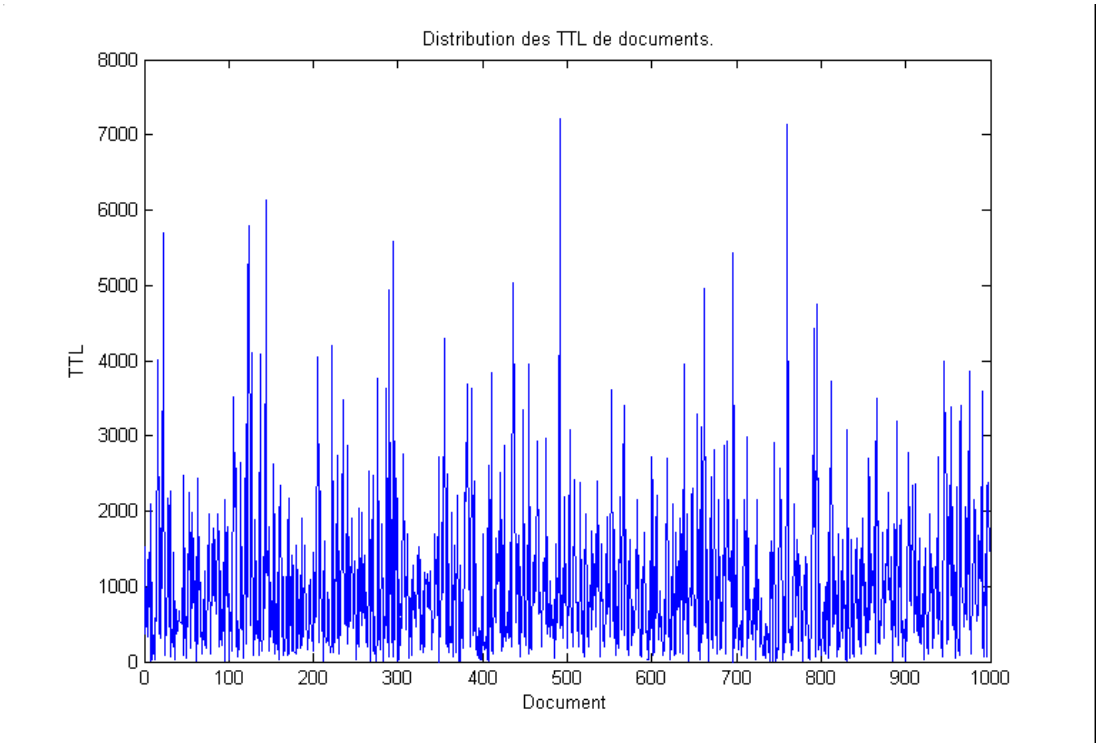


FIG IV. 5 : La distribution des TTL de documents

IV.5. Résultats de la simulation

Nous allons présenter les résultats de simulation de notre politique de remplacement ***Mobile Zone Serveur*** en comparaison avec les résultats obtenus des quatre politiques citées précédemment (LFU, LRU, ZC et GroupCaching). On commence par présenter l'impact de paramètre ***NBclients*** sur notre politique $\boldsymbol{H_i = \frac{f_i}{S_i} + NBclients(i)}$, en la comparant avec la même politique, mais sans le paramètre ***NBclients,*** $\boldsymbol{H_i = \frac{f_i}{S_i}}$ (on l'appellera la politique standard). Ensuite nous effectuerons une comparaison des performances des politiques en faisant varier le paramètre de loi de distribution zipf, puis en faisant varier la taille du cache. Et enfin, nous terminons par la présentation des résultats de la simulation en faisant varier le nombre de nœud dans notre région de simulation.

La table ci dessous montre les paramètres utilisés dans la simulation de toutes les politiques pour exprimer le comportement similaire à la réalité des réseaux Ad hoc, ces

paramètres sont choisis soigneusement, et déjà expérimentés dans plusieurs travaux dans le domaine[92], [18].

Simulateur	Orienté objet
Temps de simulation	6000 secondes
Taille de réseaux	1500×500 m
Nombre de mobiles	50~200 nœuds
Rayon de transmission	100 m
Modèle de mobilité	RandomWayPoint
La vitesse des nœuds	1~10 m/s aléatoirement
Nombre de fichiers	1000 fichiers
Le temps moyen de lancement d'une requête	0.2 seconds
Probabilité de requête aux fichiers de nœud (la localité temporelle)	50 %
Taille minimale des fichiers	1 KO
Taille du cache	200 KO ~1000 KO
TTL	λ=5000 ms
Le pourcentage des fichiers validés	15%

Table VI.2 : Paramètres de simulation

IV.5.1. Métriques d'évaluation des performances

Les métriques d'évaluations des performances dépendent des objectifs de l'utilisation de cache, qui sont la réduction de la charge du serveur, la réduction du trafic dans le réseau et la réduction de la latence moyenne des clients.

Donc, les critères qui peuvent mesurer les performances sont le taux de succès des requêtes et le taux de succès en octets. Pour évalué les performances de notre politique coopérative, il est évident de calculer ces métriques par rapport à toute la zone de chaque nœud et non pas par rapport à chaque nœud isolé.

IV.6. Impact du paramètre de la loi de distribution Zipf

La fréquence d'accès aux documents a un effet sur toutes les métriques de performances du cache. Plus les fréquences d'accès sont décentrées, plus les requêtes sont concentrées sur quelques documents, et par conséquent les taux de succès augmentent.

Pour les petites valeurs de paramètre de la loi de distribution Zipf, les fréquences sont moins décentrées c.à.d. qu'elles appartiennent à des plages de valeurs restreintes.

IV.6.1. Impact du paramètre *NBclients* sur notre politique

Le paramètre ***NBclients*** est utilisé pour réduire le nombre de fichiers dupliqués dans une zone. Pour montrer son apport à notre politique, on a comparé les performances de notre politique avec la même politique, mais sans le paramètre *NBclients,* et qu'on a appelé la politique *Standard,*

IV.6.1.1. Taux de succès dans une zone de MZS en comparaison avec la politique Standard

La figure ci-dessous montre la variation des taux de succès dans une zone en fonction de paramètre de la distribution zipf des deux politiques.

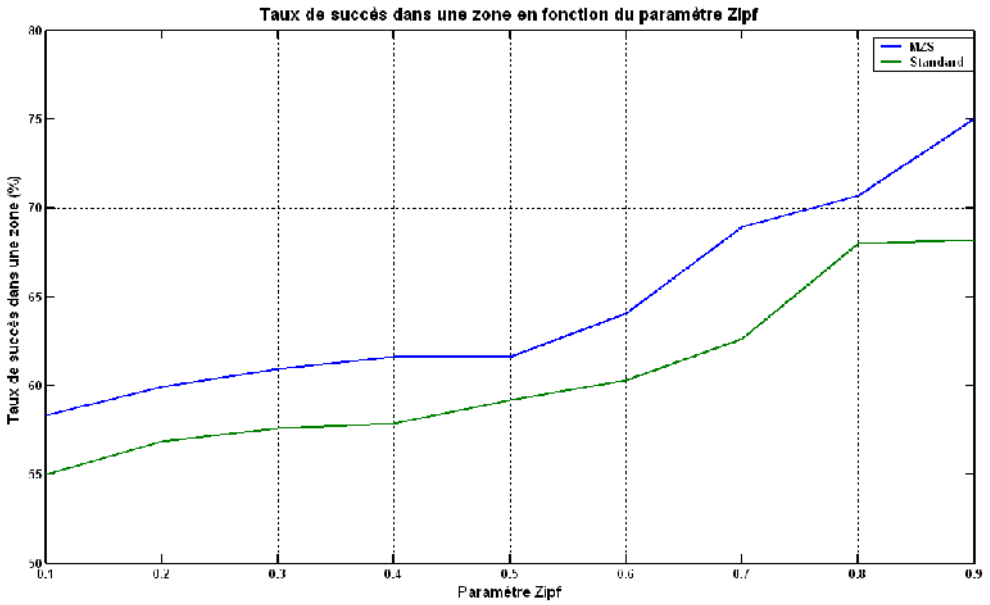


FIG IV. 6 : Taux de succès dans une zone de MZS et Standard

On remarque que le paramètre *NBclients* a amélioré le taux de réussite dans une zone avec toutes les variations du paramètre zipf.

On remarque aussi que les courbes sont ascendantes. Ceci peut être expliqué par la sensibilité des deux politiques à la popularité centrée, ce qui augmente le taux de succès dans le cache avec les grandes valeurs du paramètre zipf, aussi parce que le paramètre de popularité est utilisé dans les deux politiques.

IV.6.1.2. Taux de succès en octets dans une zone de MZS en comparaison avec la politique Standard

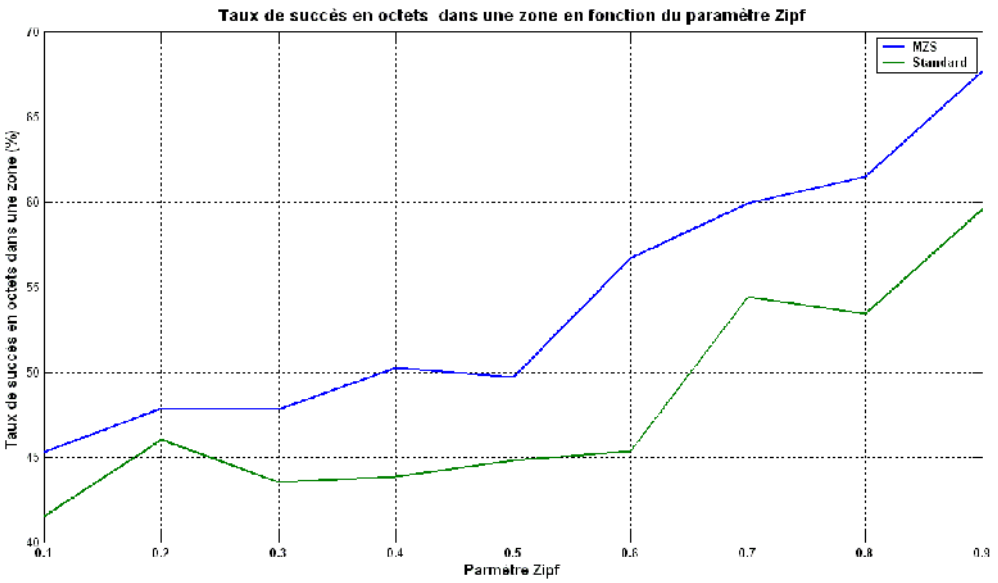


FIG IV. 7 : Taux de succès en octets dans une zone de MZS et Standard

En plus de l'amélioration du taux de réussite dans une zone, le taux de réussite en octets dans une zone est amélioré aussi avec toutes les variations du paramètre zipf.

IV.6.2. Taux de succès dans une zone de MZS en comparaison avec LRU et LFU

La figure **FIG IV.8** montre le Taux de succès de requêtes dans une zone obtenu avec les politiques de remplacement LRU, LFU et MZS.

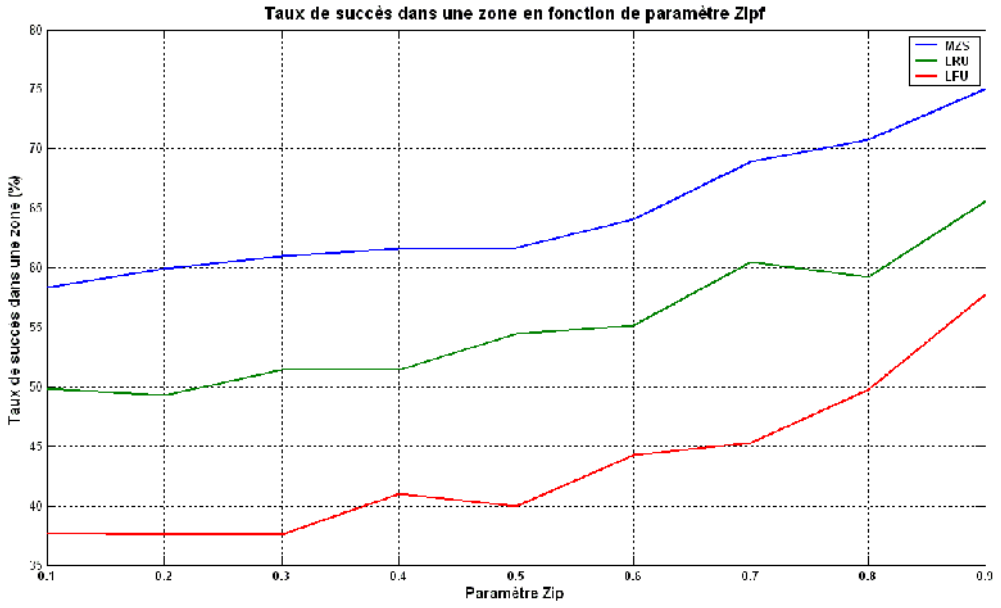


FIG IV. 8 : Taux de succès dans une zone de MZS, LRU et LFU

On peut constater que MZS donne de meilleures performances pour toutes les variations de paramètre zipf.

Ceci peut être expliqué par la maximisation du nombre de fichiers dans une zone, et la favorisation des fichiers de grande taille pour le remplacement dans la politique MZS.

On aperçoit que LFU donne les plus mauvais résultats, par ce que les nœuds ont les mêmes intérêts pour les fichiers populaires, ce qui induit des duplications des fichiers dans une même zone, et par conséquence la détérioration de taux de succès dans une zone.

IV.6.3. Taux de succès en octets dans une zone de MZS en comparaison avec LRU et LFU

Le taux de réussite en octets de MZS est plus élevé que celui de LRU et LFU, malgré que MZS favorise les fichiers de grande taille pour le remplacement, contrairement aux politiques LRU et LFU qui ne font pas de distinction lors du remplacement.

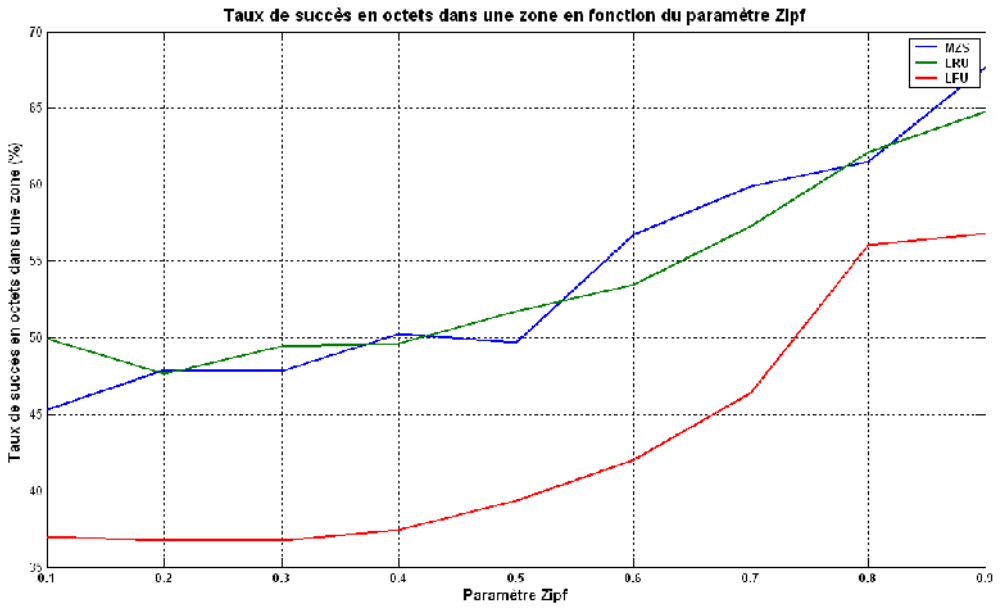


FIG IV. 9 : Taux de succès en octets dans une zone de MZS, LRU et LFU

IV.6.4. Taux de succès dans une zone de MZS en comparaison avec GC et ZC

Sur le graphe ci -dessous, On remarque que MZS donne un meilleur taux de succès en zone par rapport à ZC et GC avec toutes les variations du paramètre zipf, et cela peut être expliqué comme suit :

MZC maximise le nombre de fichiers dans une zone tout en gardant le maximum de satisfaction locale.

GC minimise le nombre de remplacements dans une zone, elle exploite l'espace libre de chaque nœud dans une zone pour placer les fichiers des autres nœuds voisins, une stratégie qui détériore en conséquence la satisfaction locale, donc le nombre de succès dans une zone.

ZC fournit les plus faibles résultats, puisque elle ne donne aucun intérêt à la satisfaction locale.

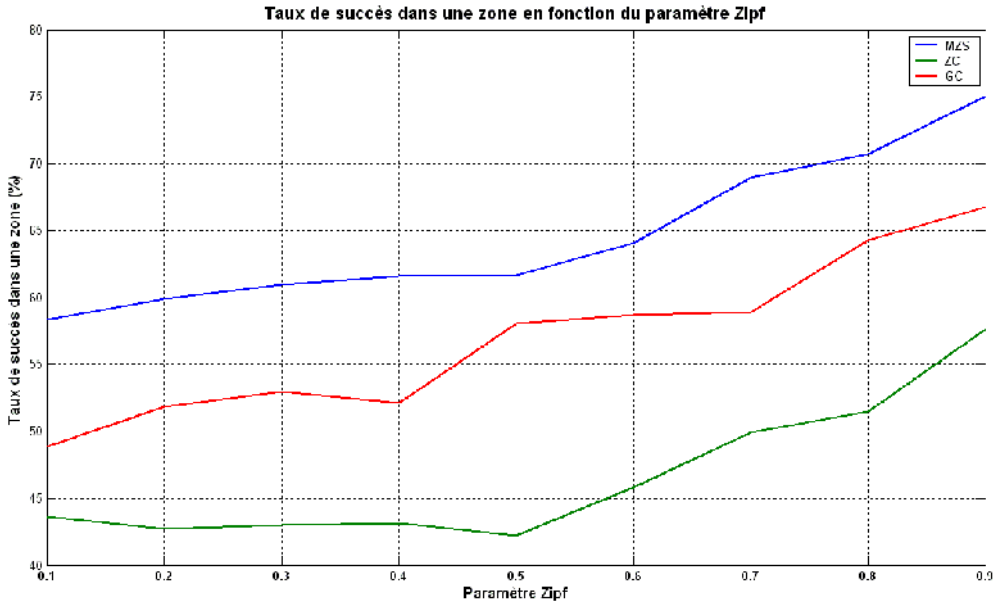


FIG IV. 10 : Taux de succès dans une zone de MZS, GC et ZC

IV.6.5. Taux de succès en octets dans une zone de MZS en comparaison avec GC et ZC

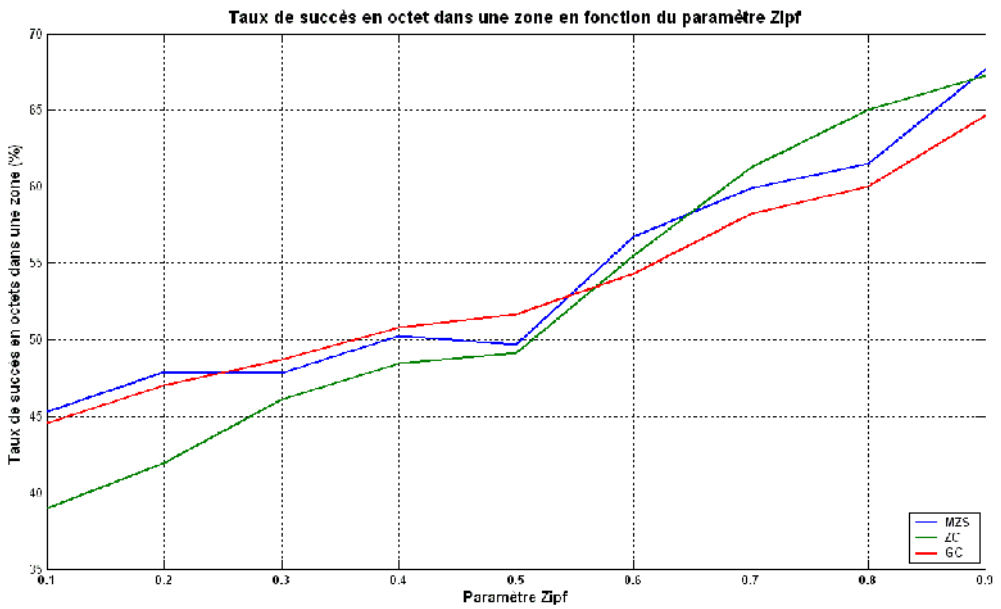


FIG IV. 11 : Taux de succès en octets dans une zone de MZS, GC et ZC

Le taux de succès en octets dans une zone de la politique MZS est meilleur que celui de GC et ZC, malgré que GC ne favorisent pas les fichiers de grande taille pour le remplacement.

IV.7. Influence de la taille de cache

L'augmentation de la taille des caches permet de cacher plus de documents et par conséquent, elle permet d'avoir de meilleures performances. Nous allons faire varier la taille du cache de 200 Ko à 1000 Ko et voir les performances qui en résultent.

IV.7.1. Impact du paramètre *NBclients* sur notre politique

Nous allons comparer les performances de MZS et Standard pour des tailles de cache différentes, afin de montrer l'importance du paramètre NBclient et son apport à notre politique.

IV.7.1.1. Taux de succès dans une zone de MZS en comparaison avec la politique Standard

La figure **FIG IV.12** montre le taux de succès de requêtes obtenu avec les politiques de remplacement MZS et Standard. On peut constater que MZS donne des meilleures performances pour toutes les tailles de cache, en particulier pour les petites. Mais on remarque que les taux de succès tendent vers l'égalité à chaque fois qu'on augmente la taille du cache. Ceci peut être expliqué par l'augmentation du nombre de fichier dans une zone.

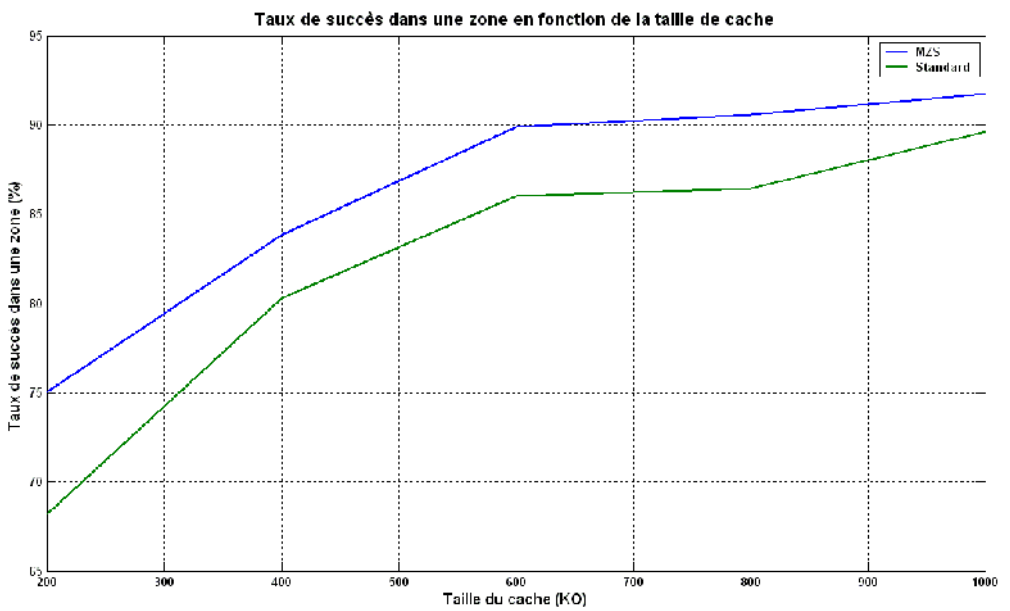


FIG IV. 12 : Taux de succès dans une zone de MZS et Standard

IV.7.1.2. Taux de succès en octets dans une zone de MZS en comparaison avec la politique Standard

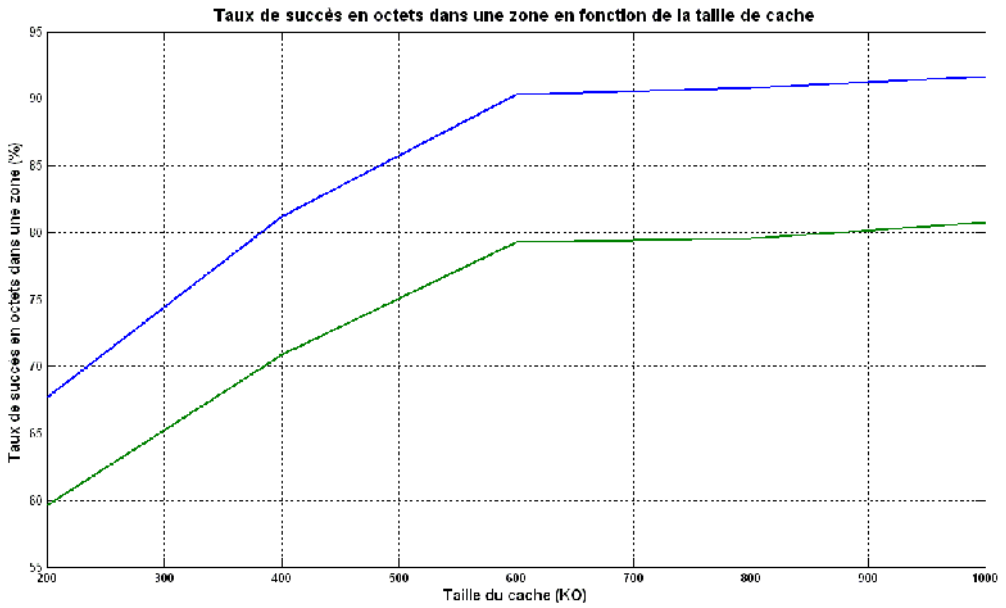


FIG IV. 13 : Taux de succès en octets dans une zone de MZS et Standard

On remarque bien que le taux de succès en octets est plus grand avec MZS pour toutes les variations de la taille.

IV.7.2. Taux de succès dans une zone de MZS en comparaison avec les politiques LRU et LFU

La figure **FIG IV.14** montre que la politique de remplacement MZS fournit un meilleur taux de succès dans une zone par rapport aux deux autres politiques LRU et LFU en particulier pour les petites tailles de cache

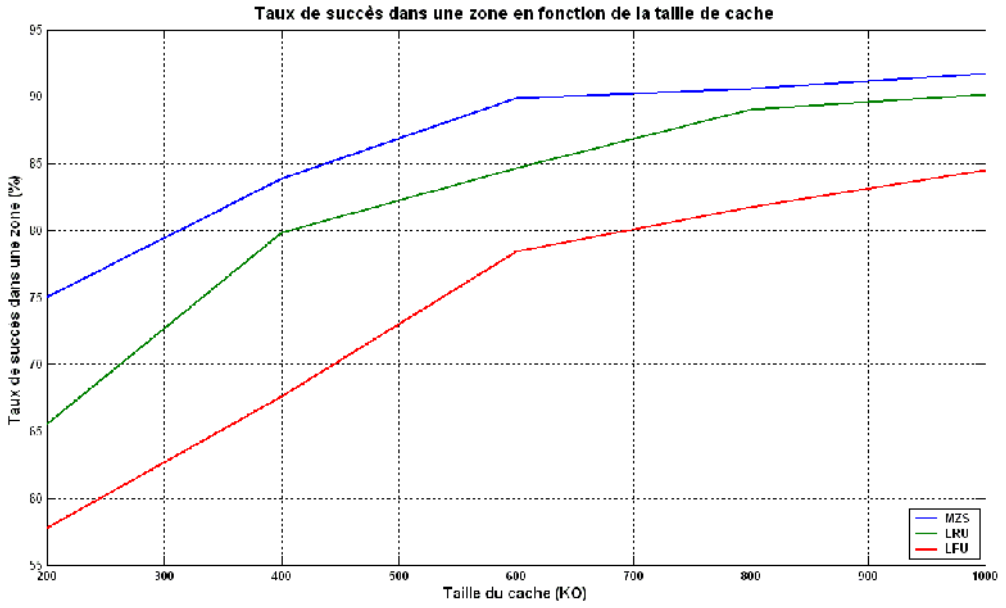


FIG IV. 14 : Taux de succès dans une zone de MZS, LRU et LFU

IV.7.3. Taux de succès en octets dans une zone de MZS en comparaison avec les politiques LRU et LFU

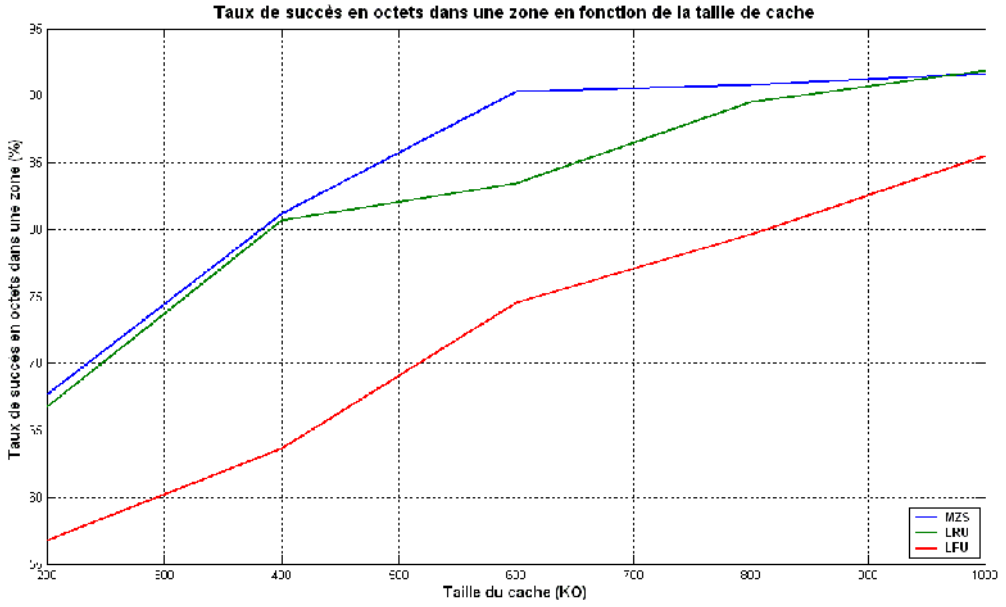


FIG IV. 15 : Taux de succès en octets dans une zone de MZS, LRU et LFU

MZS est toujours meilleure en taux de succès en octets dans une zone que les deux autres politiques LRU et LFU.

IV.7.4. Taux de succès dans une zone de MZS en comparaison avec les politiques ZC et GC

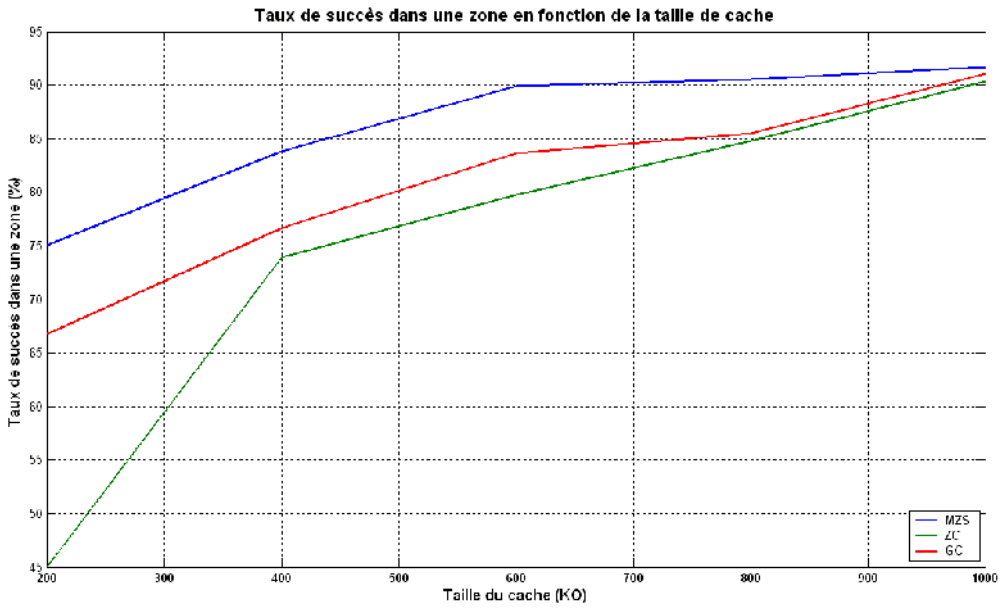


FIG IV. 16 : Taux de succès dans une zone de MZS, ZC et GC

On remarque que MZS est toujours meilleure en taux de succès dans une zone que les deux autres politiques ZC et GC, particulièrement au niveau des petites tailles du cache. Mais on remarque que les taux de succès tendent vers l'égalité à chaque fois qu'on augmente la taille du cache.

IV.7.5. Taux de succès en octets dans une zone de MZS en comparaison avec les politiques ZC et GC

La figure **FIG IV.17** montre que la politique de remplacement de cache MZS donne un meilleur taux de succès en octets par rapport aux deux politiques de remplacement de cache GC et ZC et ce pour presque toutes les variations de la taille du cache.

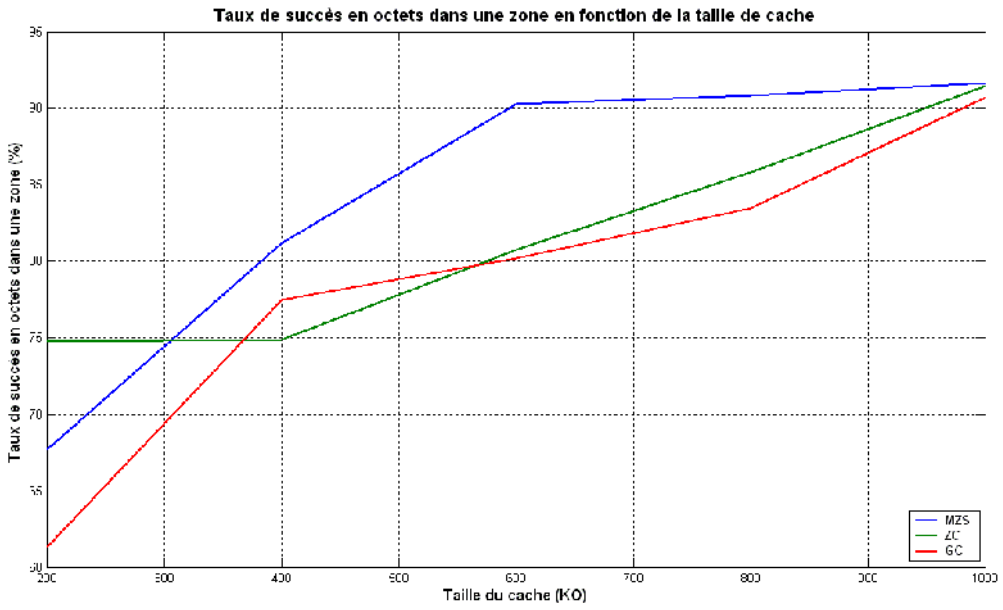


FIG IV. 17 : Taux de succès en octets dans une zone de MZS, ZC et GC

IV.8. Influence du nombre de nœuds

L'augmentation du nombre de nœuds dans la région de simulation permet de construire des zones plus grandes en nombre de nœud, et par conséquent, elle permet d'avoir plus de coopération entre cache, donc de meilleures performances. Nous allons faire varier le nombre de nœuds dans notre région de simulation de 50 à 200 et voir les performances qui en résultent.

IV.8.1. Impact du paramètre *NBclients* sur notre politique

IV.8.1.1. Taux de succès dans une zone de MZS en comparaison avec la politique Standard

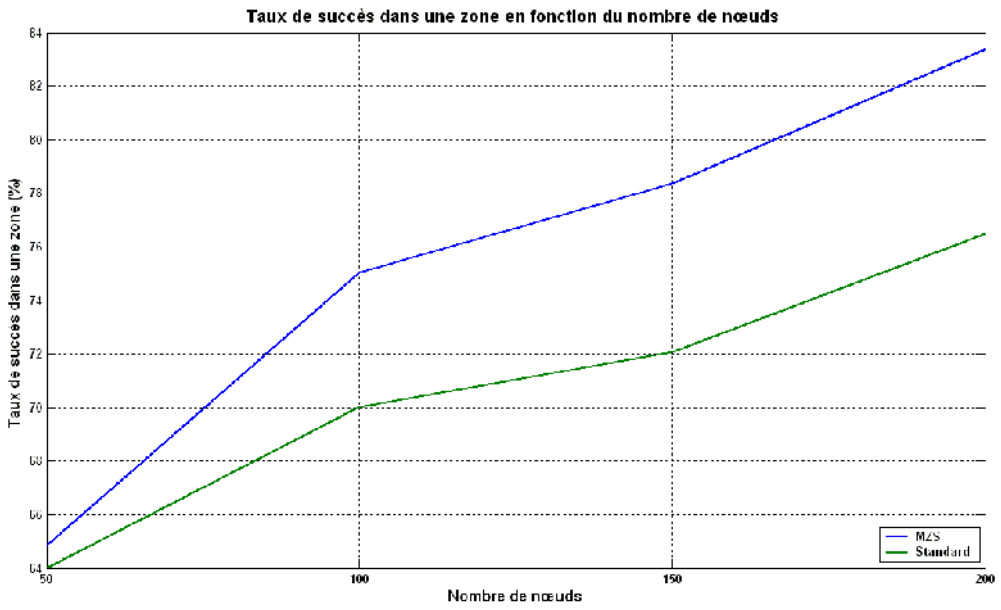


FIG IV. 18 : Taux de succès dans une zone de MZS et Standard

On remarque dans ce graphe que le taux de succès dans une zone est toujours meilleur avec notre politique MZS, par rapport à la politique standard, particulièrement quant on augmente le nombre de nœuds dans la région de simulation,

On peut expliquer ces résultats par l'augmentation du nombre de nœuds dans une zone qui augmente aussi le niveau de coopération, par conséquent le taux de succès dans une zone.

Dans le cas d'un nombre restreins de nœuds, et à la limite, MZS se comporte comme la politique standard.

IV.8.1.2. Taux de succès en octets dans une zone de MZS en comparaison avec la politique Standard

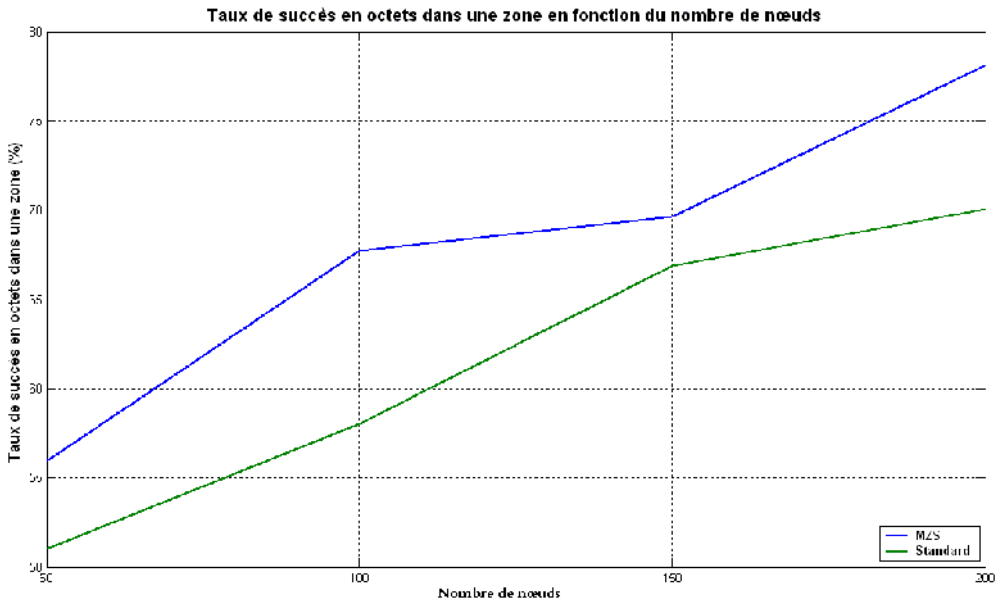


FIG IV. 19 : Taux de succès en octets dans une zone de MZS et Standard

Le nombre de succès en octets dans une zone de MZS est amélioré en augmentant le nombre de nœuds, du faite de l'augmentation du nombre de succès dans une zone.

IV.8.2. Taux de succès dans une zone de MZS en comparaison avec les politiques LRU et LFU

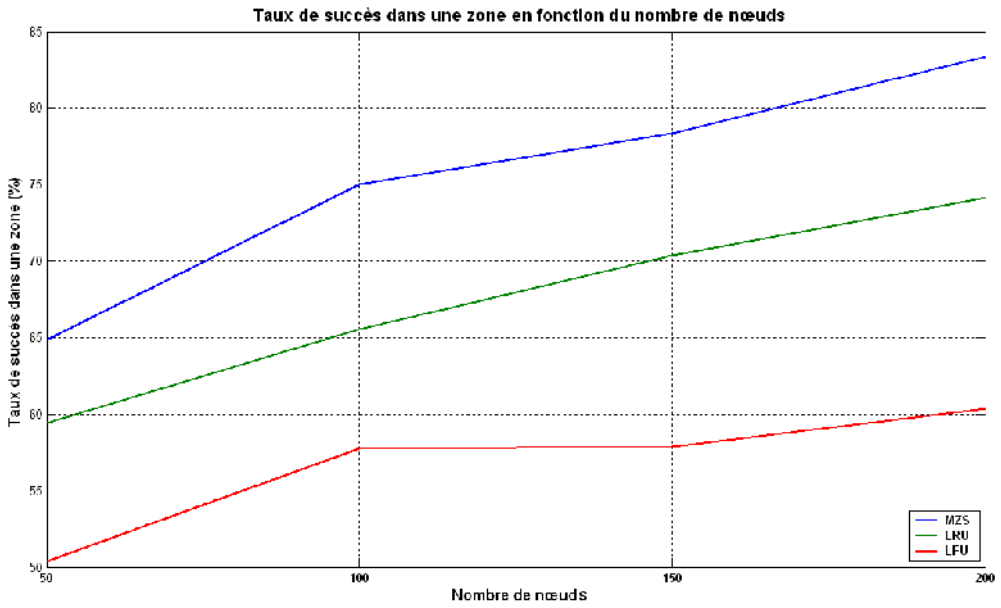


FIG IV. 20 : Taux de succès dans une zone de MZS, LRU et LFU

Ce graphe nous montre que MZS est toujours meilleure que LRU et LFU en taux de succès dans une zone, surtout en augmentant le nombre de nœuds dans la région de simulation.

IV.8.3. Taux de succès en octets dans une zone de MZS en comparaison avec les politiques LRU et LFU

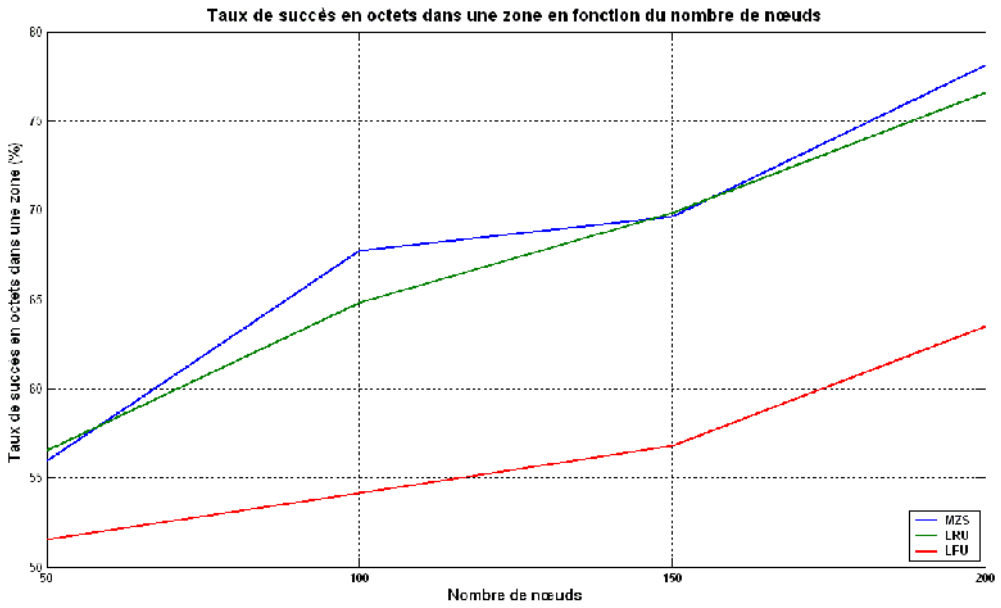


FIG IV. 21 : Taux de succès en octets dans une zone de MZS, LRU et LFU

Le taux de succès en octets dans une zone augmente à chaque fois que le nombre de nœud augmente. Ceci peut être expliqué par l'augmentation du nombre de nœuds dans une zone.

MZS paraitre la plus performante même en comparaison avec des politiques qui ne favorise pas les grands fichiers pour le remplacement comme LRU et LFU.

IV.8.4. Taux de succès dans une zone de MZS en comparaison avec les politiques ZC et GC

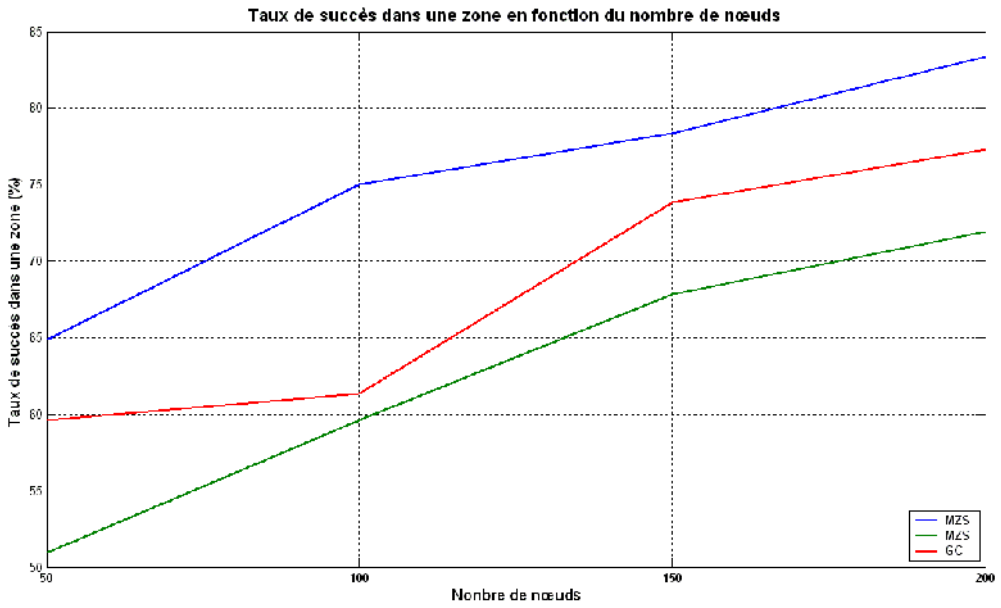


FIG IV. 22 : Taux de succès dans une zone de MZS, ZC et GC

La politique MZS est plus performante que les deux autres politiques de remplacements. Ceci peut être interprété comme suit :

GC est la politique la plus proche en performances de MZS, la différence est que MZS ne détériore pas la satisfaction des nœuds en local contrairement à GC et ZC, ces deux dernies maximisent aussi le nombre de fichiers dans une zone, mais négligent l'impact de la satisfaction locale.

ZC est la moins performante puisque il n'y a pas de coopération entre les nœuds de la même zone dans cette politique lors du remplacement de cache.

IV.8.5. Taux de succès en octets dans une zone de MZS en comparaison avec les politiques ZC et GC

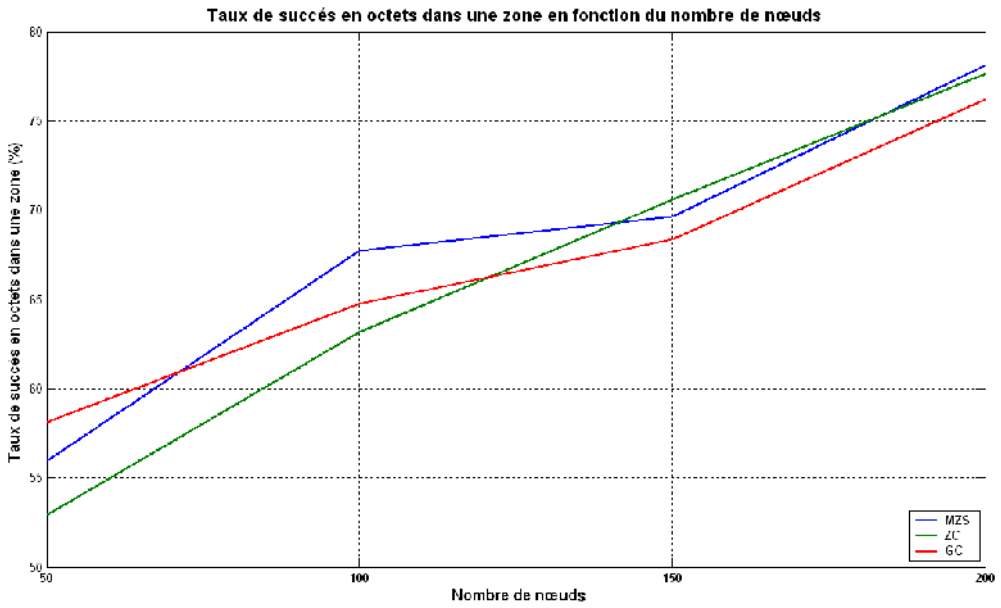


FIG IV. 23 : Taux de succès en octets dans une zone de MZS, ZC et GC

MZS est meilleure que les deux politiques pour la plupart des variations de nombre de nœud. Ceci peut être expliqué comme suit :

L'augmentation du nombre de nœuds provoque une coopération intense entre les nœuds de la même zone dans les politiques MZS et GC, augmentant ainsi le taux de succès en octets dans une zone, Mais, dans la politique GC, le taux de réussite dans le cache reste le facteur nuisible.

La performance en taux de succès en octets dans une zone de ZC augmente à chaque fois qu'on augmente le nombre de nœuds. Parce que les nœuds ne cachent pas un document qui se trouve déjà dans la zone, donc la ZC est très influencé par cette augmentation du nombre de nœuds.

On peut expliquer le faible taux de succès en octets de MZS parfois par rapport à GC et ZC comme suit :

Premièrement, Notre test est effectué avec une petite taille du cache, pour bien montrer l'influence du nombre de nœud.

Deuxièmement, MZS favorise les grands fichiers pour le remplacement, contrairement à GC, et même dans la politique ZC, ou l'impact de la taille n'est aussi influent que celui dans MZS.

IV.9. Conclusion

Après l'étude de performance des Mécanismes de remplacement de caches des réseaux mobiles Ad hoc, on peut conclure que les politiques de remplacement de caches classiques fournissent de moins bonnes performances et elles sont inadaptées pour ce type de réseaux, et ce, par le fait qu'elles se basent généralement sur un seul facteur dans leur décision de remplacement, comme le cas de LFU qui se base uniquement sur la fréquence.

Par conséquent, pour plus de performances du cache indépendamment des autres nœuds du réseau, la conception des politiques de remplacement de caches destinées aux réseaux mobiles Ad hoc ou aux environnements mobiles en général doit prendre en considération les facteurs qui sont liés aux caractéristiques du trafic comme la fréquence des documents et leurs tailles, et aussi la récence qui caractérise le comportement indépendant des nœuds dans le réseau.

Et même l'association de ces facteurs s'avère insuffisante pour être adoptée dans les réseaux Ad hoc, à cause des caractéristiques spécifiques de ces réseaux mobiles. C'est pourquoi toutes les politiques récentes dédiées aux réseaux Ad hoc utilisent un schéma de coopération entre les nœuds du réseau afin d'augmenter significativement les performances globales de tout le système.

Pour conclure, une politique de remplacement de cache dédiée aux réseaux Ad hoc, doit prendre en considération l'équilibre entre la satisfaction locale et la satisfaction des autres nœuds du réseau. C'est le cas de notre politique MZS, qui s'est avérée bien équilibrée, et qui fournit les meilleurs taux de réussites.

Conclusion générale

Les réseaux mobiles Ad hoc sont formés avec des nœuds mobiles, tels que les ordinateurs portables, PDA, ou téléphone cellulaire. Ces appareils mobiles peuvent créer un réseau sans fil de façon dynamique sans l'aide de toute infrastructure de réseau. Tout nœud peut se déplacer de façon arbitraire et communiquer avec les autres en utilisant des liaisons sans fil multi sauts. Chaque nœud agit comme un routeur et transmet les données à d'autres voisins dans sa zone de couverture de transmission. Les réseaux Ad hoc sont très utiles dans certains environnements, tels que les champs de batailles et les secours en cas de catastrophe.

Ces dernières années, des études de recherche dans les réseaux Ad hoc on été centrées principalement sur la conception des protocoles de routage multi-sauts pour la transmission de données entre nœuds. Toutefois, le but ultime d'un protocole de routage est de créer un chemin de routage pour la communication des données. Par conséquent, la façon d'améliorer l'accessibilité et la disponibilité des données, en utilisant les techniques de mise en cache, a émergé comme un important domaine de recherche.

Les techniques du cache sont une solution efficace pour accroître la performance de communication. Le cache a été largement utilisé dans différents domaines tels que la conception du processeur, multiprocesseur, l'architecture de mémoire et la conception de routeur. Par ailleurs, Internet utilise les caches Web pour réduire la charge sur les serveurs proxy [73] et l'architecture coopérative de cache [83] pour réduire les requêtes du trafic réseau et la latence moyenne des données de manière significative.

Dans un réseau Ad hoc, si le cache dispose de données fréquemment demandées, ces données cachées peuvent être servies pour d'autres plus tard. Le demandeur n'a pas besoin de récupérer les données de serveur distant (source de données). Par conséquent, l'accessibilité des données sera améliorée.

En outre, les techniques coopératives de mise en cache permettent la mise en cache coordonnée entre plusieurs nœuds, comme la réorientation des demandes de données, le placement en cache, ou le remplacement du cache.

La mobilité est l'un des grands problèmes rencontrés dans les réseaux Ad hoc. À titre d'exemple, le réseau peut être partitionné en de nombreux réseaux indépendants. Par

conséquent, le demandeur ne peut pas récupérer les données souhaitées sur le serveur distant ou dans un autre réseau.

Dans ce travail, nous avons proposé une politique coopérative de remplacement de cache pour les réseaux mobiles Ad hoc, que nous avons appelée MZS (Mobile Zone Serveur), qui se base sur quatre paramètres, la fréquence d'accès, le temps écoulé depuis le dernier référencement du document, la taille du document, et un paramètre de coopération que nous avons appelé *NBclient*. Les résultats de simulation ont montré que la politique MZS donne de meilleures performances pour la plupart des métriques de performances, en particulier pour le taux de succès des requêtes dans une zone, donc la plus efficace en cas de partitionnement de réseau. Ce qui est très fréquent dans les réseaux Ad hoc. De même, nos résultats montrent que la meilleure politique pour les réseaux mobiles Ad hoc est celle qui prend en considération les caractéristiques spécifiques de ces réseaux, ainsi que celles du trafic Web.

Perspectives

Vu le caractère dynamique de la topologie des réseaux mobiles Ad hoc, qui entraine de fréquentes déconnexions, d'éventuelles partitionnements de réseaux, et une répartition non équilibrée des nœuds dans un réseau, il serait intéressant, comme perspective, d'étendre notre proposition en intégrant un schéma de coopération (cache data, cache path, cache hybride) dans notre politique pour renforcer et élargir la coopération dans tout le réseau. En effet, celle-ci est réduite dans notre politique à des nœuds voisins d'un seul saut, et ce, afin d'obtenir une meilleure performance globale du système.

Une autre perspective serait d'envisager une coopération entre zones pour récupérer les données qui ne se trouvent pas dans une zone depuis la zone la plus proche, c'est-à-dire de créer un schéma de coopération hiérarchique intra zone et extra zone.

De même, il serait intéressant de partitionner les données en deux priorités, de telle manière que la première contienne les données uniques dans une zone d'un seul saut, et la deuxième contienne les données qui se trouvent uniques dans les zones voisines.

Cependant, avec l'intégration d'une de ces techniques envisagées, plusieurs questions seront posées, comme la localisation des documents cachés et les remplacements coordonnés.

Bibliographie

[1] Abrams .M, Standridge .C .R, Abdulla .G, Wiliams .S, and Fox .E .A, ”Caching Proxies: Limitations and Potentials”, Proceedings Of the 4th International WWW Conference, Boston, December 1995.

[2] Acharya .S, “Broadcast Disks: Dissemination-Based Data Management for Asymmetric Communication Environments”. PhD dissertation, Brown Univ, USA, Mai 1998.

[3] Aggarwal .C, Wolf .J .L, and Yu .P .S, “Caching on the World Wide Web”, IEEE Transactions on knowledge and data engineering, 11(1), 1999.

[4] Arlitt .M .F, Cherkasova .L, Dilley .J, Friedrich .R .J and Jin .T .Y, ”Evaluating content management techniques for Web proxy caches”. ACM SIGMETRICS Performance Evaluation Review, Vol. 27 N°. 4, March 2000, pp. 3–11.

[5] Arlitt .M, Friedrich .R, and Jin .T .Y, ”Performance evaluation of Web proxy cache replacement policies”, Computer performance evaluation: modeling techniques and tools: International conference N°10, Palma de Mallorca, Vol. 1469, 1998, pp. 193-206.

[6] Arlitt .M, Friedrich .R, and Jin .T, ”Workload characterization of a Web proxy in a cable modem environment”. ACM SIGMETRICS Performance Evaluation Review, New York, NY, USA, Vol. 27, N°. 2, 1999.

[7] Aumann .Y, et al,“Predicting event sequences: Data mining for prefetching web-pages”, In Proc. of ACM KDD, 1998.

[8] Bahn .H, Koh .k, Min .S .L, and Noh .S .H, ”Efficient replacement of non uniform objects in web caches”. IEEE comput, Vol. 35, N°. 6, June 2002, pp. 65–73.

[9] Bellman .R, “Dynamic Programming”, Princeton University Press, Princeton, New Jersey, 1957.

[10] Benhamida .N, « les politiques de gestion du cache d’un serveur web ». Mémoire de Magistère en Informatique de l’Université A/Mira, Bejaia. 2007.

[11] Bolot .J, and Hoschka .P, “Performance engineering of the World Wide Web: Application to dimensioning and cache design”. In Proceedings of the 5th International World Wide Web Conference. Elsevier, Amsterdam, Pays-bas, 1996.

[12] Broadband Radio Access Networks (BRAN). High Performance Radio Local Area Network (HIPERLAN) type 1. Functional specification. Technical Report EN 300 652 ref. REN/BRAN-10-01, ETSI, 1998.

[13] Broadband Radio Access Networks (BRAN). High Performance Radio Local Area Network (HIPERLAN) type 2. Requirements and architectures for wireless broad band access. Technical Report TR 101 031 V2.2.1 (1999-01) ref. RTR/BRAN- 0022001, ETSI, 1999.

[14] Buntand .R and Murphy .J,”The measurement, of locality and the behavior of programs”. The Computer Journal, Vol. 27, N°. 3, 1984, pp. 238-245.

[15] Busari .M and Williamson .C, “A synthetic workload generation tool for simulation evaluation of Web proxy caches”, Computer Networks Vol. 38, N° 6, 2002, pp. 779-794.

[16] Cao .P, and Irani .S, ”Cost-aware WWW proxy caching algorithms”. In Proceedings of the USENIX Symposium on Internet Technologies and Systems, 1997, pp. 193–206.

[17] Chan .E, Li .W, Chenb .D, ”Energy saving strategies for cooperative cache replacement in mobile Ad hoc networks”, published in: journal Pervasive and Mobile Computing, Vol. 5, Issue 1, February, 2009.

[18] Chand .N, Joshi .R .C and Misra .M, “Cooperative cache management in mobile Ad hoc netwoks”, Mobile Technology, Applications and Systems, 2005 2nd International Conference on, Guangzhou, India, 15-17 Nov. 2005, pp. 1 – 7.

[19] Chankhunthod .A, Danzig .P .B, Neerdaels .C, Schwartz .M .F, Worrell .K. J, “A hierarchical internet object cache”, in: USENIX Annual Technical Conference,1996, pp. 153-164.

[20] Chen .H, Xiao .Y, ”On-bound selection cache replacement policy for wireless data access”, IEEE Transactions on Computers, Vol. 56, N°. 12, 2007, pp. 1597 – 1611.

[21] Cheng .K, Kambayashi .Y, ” LRU-sp: a size-adjusted and popularity-aware LRU replacement algorithm for web caching”. Published in: · proceeding comp sac 24th international computer software and applications conference IEEE computer society Washington, DC, USA, 2000.

[22] Chuang .J, and Sirbu .M, ”Adding quality of service to network storage”. In Proceedings of the Workshop on Internet Service Quality Economics, 1999.

[23] Cohen .E, krishnamurthy .B, Krishnamurthy .E, rexford .J, “evaluating server-assisted cache replacement in the web “,in proceedings of the 6th european symposium on algorithms,1998.

[24] Danzig .P .B, Hall .R .S, Schwartz .M .F, ”A case for caching file objects inside internetworks”, in: SIGCOMM'93: Conference Proceedings on Communications Architectures, Protocols and Applications, ACM Press, New York, NY, USA, 1993, pp. 239-248.

[25] Das .S, Perkins .C, Belding-Royer .E, RFC 3561, Ad hoc On-demand Distance Vector (AODV) Routing, Memo. The Internet Society. 2003. Disponible sur : http://www.ietf.org/rfc/rfc3561.txt.

[26] Dhoutaut .D, « Etude du standard IEEE 802.11 dans le cadre des réseaux Ad hoc : de la simulation à l’expérimentation ». Thèse Doctorat, Institut National des Sciences Appliquées, Lyon, 2003.

[27] Dijkstrat .E .W, A note no two problems in connection with graphs. Numerische Mathematik, 1959, pp. 269-271.

[28] Dilley .J, Arlitt .M, Perret .S, and Jin .T, ”The Distributed Object Consistency Protocol”. Technical Report HPL-1999-109, Hewlett-Packard Labs, 1999.

[29] Dingle .A, Partl .T, ”Web Cache Coherence”, Charles University, Prague, Fifth International World Wide Web Conference, Paris, France, May 6-10, 1996, pp. 907-920.

[30] Eschard .L, Gapihan .N, Manteau .J, Rover .A, « Projet UML/Java, Émulation du protocole DSR sur réseau mobile ad’ hoc », Rapport Final, 2010.

[31] Foong .A .P, Hu .Y, and Heisey .D .M,”Essence of an effective web caching algorithm”. In proceedings of the international conference on internet computing, 2002, pp. 69-276.

[32] Gray .C .G, and Cheriton .D .R, ”Leases: An efficient fault-tolerant mechanism for distributed file cache consistency”, In Proceedings of the 12th ACM Symposium on Operating System Principles, 1989, pp. 202–210.

[33] Haas .Z, and Pearlman .M, The Zone Routing Protocole (ZRP) for Ad hoc Net- works.IETF MANET Draft, June 1999.

[34] Hosseini-Khayat .S,"Investigation of generalized caching", Ph.D. dissertation. Washington University, St. Louis, MO. 1997.

[35] Jacquet .P, and Clausen .T, RFC 3626, Optimized Link State Routing Protocol (OLSR), Memo. The Internet Society. 2003. Disponible sur: http://www.ietf.org/rfc/rfc3626.txt.

[36] Jangeun .J, Peddabachagari .P, and Sichitiu .M, Theoretical maximum throughput of IEEE 802.11 and its applications. Network Computing and Applications, Second IEEE International Symposium on. North Carolina State Univ, April 2003, pp. 249- 256.

[37] Jin .S, and Bestavros .A, "GreedyDual*: Web caching algorithms exploiting the two sources of temporal locality in Web request streams". In Proceedings of the 5th International Web Caching and Content Delivery Workshop. Vol.24, 2000, pp.174-183.

[38] Jinag .Z, and Kleinrock .L,"Web prefetching in a mobile environment". IEEE Personal Communications, Vol. 5, N°. 5, Oct 1998, pp 25 - 34.

[39] Joa-Ng .M, and Lu .I, "A Peer-to-Peer Zone-Based Two-Level Link State Routing for Mobile Ad hoc Networks". IEEE Journal on Selected Areas in Communications, Vol. 17, N°. 8, 1999, pp. 1415-1425.

[40] Johnson .D .B, Maltz .D .A, and Hu .Y, "The Dynamic Source Routing Protocol for Mobile Ad hoc Net- works (DSR)". Draft IETF, juillet 2004. Disponible sur : http://tools.ietf.org/id/draft-ietf-manet-dsr-10.txt.

[41] Karedla .R, Love .J, and Wherry .B, "Caching strategies to improve disk system performance". IEEE Computer, Vol. 27, N°. 3, Mars 1994, pp. 38-46.

[42] Kelly .T, Jamin .S, and Mackie-Mason .J .K, "Variable QoS from shared Web caches: User centered design and value-sensitive replacement", In Proceedings of the MIT Workshop on Internet Service Quality Economics, 1999.

[43] Kin-Yeung .W,"Web Cache Replacement Policies: A Pragmatic Approach", Macao Polytechnic Institute, IEEE Network,Vol. 20, N°. 1, January/February 2006, pp. 28-34.

[44] Korupolu .M .R, Dahlin .M, "Coordinated placement and replacement for large-scale distributed caches", IEEE Transactions on Knowledge and Data Engineering, Vol. 14, N°. 6, 2002, pp. 1317-1329.

[45] Krishnamurthy .B, and Rexford .J, "Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching, and Traffic Measurement". Addison-Wesley, Reading, MA. 2001.

[46] Krishnamurthy .B, and Wills .C .E, "Study of piggyback cache validation for proxy caches in the World Wide Web". In Proceedings of the USENIX Symposium on Internet Technologies and Systems, 1997, pp. 1–12.

[47] Lai .K .Y, Tari .Z, Bertok .P, "Location-aware cache replacement for mobile environnements", in: Global Telecommunications conference, GLOBECOM '04. IEEE, Vol. 6, 2004, pp. 3441–3447.

[48] Larbi .A, « Mécanisme de remplacement de cache dans les réseaux Ad hoc », mémoire de Magistère en Informatique, Université A/Mira, Bejaia, 2008.

[49] Leey .R, Goshimay .K, Kambayashiy .Y, and Takakuraz .H, "Caching Scheme for Mobile Web Information Retrieval", Proceedings of the 2nd International Workshop on Web Dynamics, Part of WWW, 2002.

[50] Legedza .U, Guttag .J, “Using Network-level Support to Improve Cache Routing”, Appears in the Proceedings of the 3rd International WWW Caching Workshop, Manchester, England, June 1998.

[51] Li .W, Chan .E and Chen .D, “Energy-efficient Cache Replacement Policies for Cooperative Caching in Mobile Ad hoc Network”. IEEE Wireless Communications and Networking Conference, March 2007, pp. 3347-3352.

[52] Lim .S, Lee .W .C, Cao .G, and Das .C .R, “A novel caching scheme for improving internet-based mobile Ad hoc networks performance”, In Journal Ad Hoc Networks, Vol. 4, N°. 2, March 2006, pp. 225-239.

[53] Mahanti .A and Williamson .C, “Web proxy workload characterization”. Technical report, Department of computer Science, university of Saskatchewan, 1999.

[54] Marti .S, Iuli .T .J .G, Lai .K, and Baker .M, “Mitigating Routing Misbehavior in Mobile Ad hoc Networks”. ACM MOBICOM ,Boston MA ,USA, 2000, pp. 255-265.

[55] Menaud .J .M, Issarny .V, and Banatre .M, ”Improving effectiveness of Web caching. In Recent Advances in Distributed Systems”. Lecture Notes in Computer Science, Vol. 1752. Springer-Verlag, Berlin, Allemagne, 2000, pp. 375–401.

[56] Meskauskas .P, “Mobile Ad hoc Networking”, Seminar on Telecommunications Technology, Helsinki, October 12, 1998.

[57] Murta .C .D, Almeida .V .A .F, and Meira .W, “Analyzing performance of partitioned caches for the www”. In proceedings of the 3rd international www caching workshop, 1998.

[58] Nagaraj .S .V,”web caching and its applications”, ISBN 1-4020-8049-2, Copyright 2004 by Kluwer Academic Publishers.

[59] Niclausse .N, Liu .Z, and Nain .P, “A new efficient caching policy for the world wide web”. In proceedings of the workshop on internet server performance”, 1998, pp. 119–128.

[60] Osawa .N, Yuba .T, and Hakozaki .K, “Generational replacement schemes for a www proxy server”. In High-Performance Computing and Networking (HPCN’97). Lecture notes in computer science, Vol. 1225. SpringerLink, Berlin, Germany, 1997, pp. 940–949.

[61] Peachey .J, “The bradford-zipf distribution and program behavior”. Master’s thesis, university of Saskatchewan. Department of computational science, 1982.

[62] Phillips .G and Breslau .L and Fan .L and Cue .p and Shenker .S,”web caching and zipf-like distributions: evidence and implications”. INFOCOM '99. 18th annual joint conference of the IEEE computer and communications societies, Proc. IEEE, New York, Vol. 1, 21-25 mars 1999, pp. 126-134.

[63] Pitkow .J, Recker .M, “A Simply Yet Robust Caching Algorithm Based on Dynamic Access Patterns”, Technical report VU-GIT-94-39, GVU Technical Report, 1994. With existing cache management techniques. In Proceedings of the 3rd International Web Caching Workshop. Manchester, Angleterre. Juin 1998.

[64] Reddy .M, and Fletcher .G .P, «Intelligent Web caching using document life histories: A comparison with existing cache management techniques”. In Proceedings of the 3rd International Web Caching Workshop. Manchester, Angleterre. Juin 1998.

[65] Ren Q, and Dunham .M .H,"Using Semantic Caching to Manage Location Dependent Data in Mobile Computing", The 6th Annual International Conference on Mobile Computing and Networking, New York, NY, USA, august 2000.

[66] Rizzo .L, and Vicisano .L, "Replacement policies for a proxy cache". IEEE/ACM Trans. Vol. 8, N°. 2, Apr 2000, pp. 158–170.

[67] Rodriguez .P, Spanner .C, Biersack .E .W,"Analysis of web caching architectures: Hierarchical and distributed caching", IEEE/ACM Transactions on Networking, Vol. 9, N°. 4, 2001, pp. 404-418.

[68] Romano .S and Elaarag .H, "A quantitative study of recency and frequency based web cache replacement strategies". CNS '08 Proceedings of the 11th communications and networking simulation symposium ACM New York, 2008.

[69] Sailhan .F, « localisation de ressources dans les réseaux Ad hoc », Thèse doctorale université Paris VI, 2005.

[70] Sailhan .F, Issarny .V, "Cooperative caching in Ad hoc networks", in: MDM'03: Proceedings of the Fourth International Conference on Mobile Data Management, Springer-Verlag, London, UK, 2003, pp. 13-28.

[71] Santhana krishnan .G, Amer .A, Chrysanthis .P .K, "Ageneralized target-driven cache replacement policy for mobile Environments towards universal mobile caching", in: Proc. Fourth ACM International Workshop on Data Engineering for Wireless and Mobile Access, 2005, pp. 73-80.

[72] Scheuermann .P, Shim .J, and Vingralek .R, "A case for delay-conscious caching of web-documents", in proceedings of the 6th international www conference, Vol. 29, N°. 8-13, 1997, pp. 997-1005.

[73] Shim .J, Scheuermann .P, and Vingralek .R, "Proxy Cache Algorithms: Design, Implementation, and Performance," IEEE Trans. Knowledge and Data Eng, Vol. 11, N°. 4, July/Aug. 1999.

[74] Silberschatz .A, and. Galvin .P .B, "Operating Systems Concepts". Addison Wesley, Reading, MA, fourth edition, 1994.

[75] Sosa .V .J, Navarro .L,"Influence of the Document Validation/Replication Methods on Cooperative Web Proxy Caching Architectures", Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS'02), Western Multiconference (WMC'02), San Antonio, Texas, Jan 2002, pp. 238-245.

[76] Squid Web Proxy Cache, disponible en ligne sur : http://www.squid-cache.org/

[77] Tatarinov .I, "An efficient LFU-like policy for web caches". Tech. Rep. Ndsu-csortr-98-01, computer science department, North Dakota state university, Wahpeton, 1998.

[78] Toh .C .K, "Associativity-Based Routing for Ad hoc Mobile Networks". Wireless Personal Communications Journal, Vol. 4, no. 2, Mars 1997, pp. 1-36.

[79] Vakali .A,"LRU-based algorithms for Web cache replacement". In International Conference on Electronic Commerce and Web Technologies. Lecture Notes in Computer Science, Vol. 1875. Springer-Verlag, Berlin, Allemagne, 2000, pp. 409–418.

[80] Vanderwiel .S and Lilja .D .J,"A Survey of Data Prefetching Techniques", Technical Report, University of Minnesota, 1996.

[81] Wessels .D, "Intelligent caching for world-wide-web objects". M.s. Thesis, university of Colorado at boulder, 1995.

[82] Wessels .D, “Web Caching”, O’Reilly, Sebastopol, CA. 2001.

[83] Wessels .D, Claffy .K, ”ICP and the Squid Web cache”, IEEE Journal on Selected Areas in Communication 16 (3), 1998, pp. 345-357.

[84] Williams .S, Abrams .M, Standridg .C .R, Abdulla .G, and Fox .E .A,”Removal policies in network caches for World Wide Web documents”. In Proceedings of ACM SIGCOMM. ACM Press, New York, USA, 1996, pp. 293–305.

[85] Wooster .R .P, and Abrams .M, «Proxy caching that estimates page load delays”. In Proceedings of the 6th International World Wide Web Conference. Santa Clara, USA, Vol. 29, N°. 8-13, 1997, pp. 977-986.

[86] Xu .J, Hu .Q .L, Lee .D .L, and. Lee .W .C, (Minimum Stretch integrated with Access rates, Update frequencies, and cache validation Delay), “Performance Evaluation of an Optimal Cache Replacement Policy for Wireless Data Dissemination“, IEEE Transactions On Knowledge and Data Engineering, Vol. 16, N°. 1, 2004.

[87] Yang .Q, Zhang .H .H, and Zhang .H,” Taylor series prediction: A cache replacement policy based on second-order trend analysis”. In Proceedings of the 34th Hawaii International Conference on Systems Sciences. IEEE Computer Society, Piscataway, New Jersey, USA, 2001.

[88] Yang .Q, Zhang .H .H, Li .T,”Mining web logs for prediction models in WWW caching and prefetching”. In: Proceeding of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 2001.

[89] Yeung .K .H, Wong .C .C, and Wong .K .Y, “A cache replacement policy for transcoding proxy servers”, In Proceedings of World Multi conference on Systems, Cybernetics and Informatics. Vol. 12, 2001, pp. 234–237.

[90] Yin .L, Cao .G, ”Supporting cooperative caching in Ad hoc networks”, IEEE Transactions on Mobile Computing , Vol. 5, N°. 1, 2006, pp. 77-89.

[91] Yin .L, Cao .G, Cai .Y, “A generalized target-driven cache replacement policy for mobile Environments”. Department of Computer Science & Engineering, The Pennsylvania State University, Vol. 65, N°. 5, 2005, pp. 583-594.

[92] Yi-Wei .T and Yeim-Kuan .C,”A Novel Cooperative Caching Scheme for Wireless Ad hoc Networks: GroupCaching”, International Conference on Networking, Architecture, and Storage, July 2007, pp. 62-68.

[93] Zhang .J, Izmailov .R, Reininger .D, and Ott .M, ”Web caching framework: Analytical models and beyond”. In Proceedings of the IEEE Workshop on Internet Applications. IEEE Computer Society, Piscataway, New Jersey, USA, Aug 1999, pp. 132 - 141. 41

[94] Zheng .B, and Lee .D .L, ”Semantic Caching in Location-Dependent Query Processing”, The 7th International Symposium on Spatial and Temporal Databases, Springer-Verlag, LNCS 2121, 2001, pp. 97-113.

[95] Zipf .G .K,” Human Behaviour and the Principle of Least Effort”, 1949: reprinted: An Introduction to Human Ecology (Hafner, New York), 1st ed, 1972, pp. 19-55.