UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

TECHNIQUE DISTRIBUÉE DE GESTION DE LA CHARGE SUR LE RÉSEAU ÉLECTRIQUE ET RING-TREE : UN NOUVEAU SYSTÈME DE COMMUNICATION P2P

Thèse de doctorat
Spécialité : génie électrique

Simon AYOUB

Jury : Philippe MABILLEAU (directeur de recherche)
Maxime Dubois (rapporteur)
François Bouffard (évaluateur)
Basile L. Agba (évaluateur)

Sherbrooke (Québec) Canada Mars 2013

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-96320-3

Our file Notre référence
ISBN: 978-0-494-96320-3

À mes parents, mes sœurs, mes frères et spécialement à ma famille

RÉSUMÉ

Le réseau de distribution et de transport de l'électricité se modernise dans plusieurs pays dont le Canada. La nouvelle génération de ce réseau que l'on appelle smart grid, permet entre autre l'automatisation de la production, de la distribution et de la gestion de la charge chez les clients. D'un autre côté, des appareils domestiques intelligents munis d'une interface de communication pour des applications du smart grid commencent à apparaître sur le marché. Ces appareils intelligents pourraient créer une communauté virtuelle pour optimiser leurs consommations d'une façon distribuée. La gestion distribuée de ces charges intelligentes nécessite la communication entre un grand nombre d'équipements électriques. Ceci représente un défi important à relever surtout si on ne veut pas augmenter le coût de l'infrastructure et de la maintenance.

Lors de cette thèse deux systèmes distincts ont été conçus : un système de communication peer-to-peer, appelé Ring-Tree, permettant la communication entre un nombre important de nœuds (jusqu'à de l'ordre de grandeur du million) tel que des appareils électriques communicants et une technique distribuée de gestion de la charge sur le réseau électrique.

Le système de communication Ring-Tree inclut une nouvelle topologie réseau qui n'a jamais été définie ou exploitée auparavant. Il inclut également des algorithmes pour la création, l'exploitation et la maintenance de ce réseau. Il est suffisamment simple pour être mis en œuvre sur des contrôleurs associés aux dispositifs tels que des chauffe-eaux, chauffage à accumulation, bornes de recharges électriques, etc. Il n'utilise pas un serveur centralisé (ou très peu, seulement lorsqu'un nœud veut rejoindre le réseau). Il offre une solution distribuée qui peut être mise en œuvre sans déploiement d'une infrastructure autre que les contrôleurs sur les dispositifs visés. Finalement, un temps de réponse de quelques secondes pour atteindre l'ensemble du réseau peut être obtenu, ce qui est suffisant pour les besoins des applications visées. Les protocoles de communication s'appuient sur un protocole de transport qui peut être un de ceux utilisés sur l'Internet comme TCP ou UDP.

Pour valider le fonctionnement de de la technique de contrôle distribuée et le système de communiction Ring-Tree, un simulateur a été développé; un modèle de chauffe-eau, comme exemple de charge, a été intégré au simulateur. La simulation d'une communauté de chauffe-eaux intelligents a montré que la technique de gestion de la charge combinée avec du stockage d'énergie sous forme thermique permet d'obtenir, sans affecter le confort de l'utilisateur, des profils de consommation variés dont un profil de consommation uniforme qui représente un facteur de charge de 100%.

Mots-clés : Algorithme Distribué, Demand Response, Gestion de la Charge Électrique, M2M (Machine-to-Machine), P2P (Peer-to-Peer), Réseau Électrique Intelligent, Ring-Tree, Smart Grid

REMERCIEMENTS

Je suis vivement reconnaissant à l'Université de Sherbrooke de m'avoir fourni un environnement de recherche exceptionnel.

Un grand MERCI à mon directeur de recherche, le professeur Philippe Mabilleau, pour son aide et son encadrement tout au long du projet, ainsi que pour tous les défis intéressants auxquels il m'a exposé et qui ont suscité ma curiosité scientifique.

Je remercie M. Alexandre Prieur et M. David Beauvais de CanmetÉNERGIE (Ressources Naturelles Canada) pour les discussions stimulantes que nous avons eus, ainsi que pour leur soutien.

Je remercie M. Alain Moreau du LTE (IREQ) pour ses précieux conseils et son appui, ainsi que pour les données réelles que LTE nous a fournies et que nous avons utilisées dans nos simulations.

Je remercie professeur Roland Malhame de l'École Polytechnique pour son aide durant mon examen général de doctorat et son support avec son modèle de chauffe-eau que nous avons utilisé et qui a facilité la mise en œuvre de notre algorithme de contrôle.

Je remercie tous les membres du jury : les professeurs Basile Agba, François Bouffard Maxime Dubois et Philippe Mabilleau pour leurs commentaires constructifs qui m'ont encouragé à développer davantage le simulateur et à générer d'autres résultats intéressants pendant le dernier mois avant la soutenance. Je remercie également le président du jury, le professeur Frédéric Mailhot, pour toutes les questions intéressantes qui m'ont été posées pendant la soutenance et pour les nouvelles idées soulevées qui pourraient mener à des développements futurs.

Je remercie ma conjointe, Vivian Issa, pour son support et ses idées inspirantes qui mènent à innover.

Finalement, je tiens à souligner l'importance de l'appui financier du Groupe de Recherche sur la Parole et l'Audio.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1 INTRODUCTION

Ce chapitre détaille la problématique visée par le projet de recherche en exposant l'évolution du réseau électrique traditionnel vers un réseau électrique intelligent le smart grid. Il présentera ensuite l'utilité d'une nouvelle recherche, les objectifs visés par cette thèse et les contributions scientifiques des travaux réalisés. Finalement, il présente le plan général de ce document.

1.1 Mise en contexte et problématique

« Le 14 août 2003, une panne d'électricité a frappé la partie centrale et nord-est des États-Unis ainsi qu'une partie de l'Ontario, au Canada. ... Les demandes d'électricité étaient donc élevées en raison des fortes charges de climatisation ... la seule façon possible de prévenir la panne aurait été de délester une charge d'au moins 1 500 à 2 500 MW autour de Cleveland et d'Akron...». C'est ce que nous pouvons lire dans le rapport provisoire : « Causes de la panne du 14 août 2003 dans le nord-est des États-Unis et au Canada », publié sur le site web de Ressources Naturelles Canada. Donc, une des causes de la panne de 14 août 2003 était la forte demande d'électricité et le délestage aurait pu prévenir l'étendue de la panne [1].

Le délestage consiste à couper l'approvisionnement d'un ou de plusieurs consommateurs pour rétablir l'équilibre entre la production et la consommation du réseau et réduire les risques de pannes entres autres. Outre la sécurisation du réseau, le délestage est utilisé pour réduire la pointe de demande en puissance. La pointe apparaît durant les périodes d'intense utilisation d'électricité. La réduction de la pointe permet à la compagnie d'électricité d'approvisionner plus de clients en utilisant la même installation de production.

En tout temps, la compagnie d'électricité doit assurer la demande en puissance. La puissance installée, qui est la puissance totale que peut fournir une installation (parc éolien, centrale hydroélectrique, ...), est bâtie pour satisfaire la puissance la plus élevée qui peut arriver sur le réseau. Mais, cette pointe de puissance se produit pendant peu de temps par année. On fait appel à des centrales qui ne servent que quelques heures par an. La Figure 1 [1] montre un réseau électrique qui est utilisé entre 95% et 100% de sa capacité totale pendant environ 300 heures par année et dont seulement 40% de celle-ci est utilisée d'une façon continue.

Le réseau électrique est donc sous-utilisé. Sur une base annuelle, il fonctionne à 65% de sa capacité totale (moyenne de l'ensemble des centrales hydroélectriques d'Hydro-Québec ou à 35% dans le cas de l'ensemble des parcs éoliens). Le facteur d'utilisation, qui est le rapport entre l'énergie électrique produite pendant un an et l'énergie qui aurait été produite si cette installation avait été exploitée pendant un an, en continu, à sa puissance maximale, est de 0,65.

$$FU = \frac{consommation\ annuelle\ totale\ (surface\ sous\ la\ courbe)}{consommation\ maximale\ possible\ (Puissance\ maximale \times 8760\ heures)} \approx 0{,}65$$

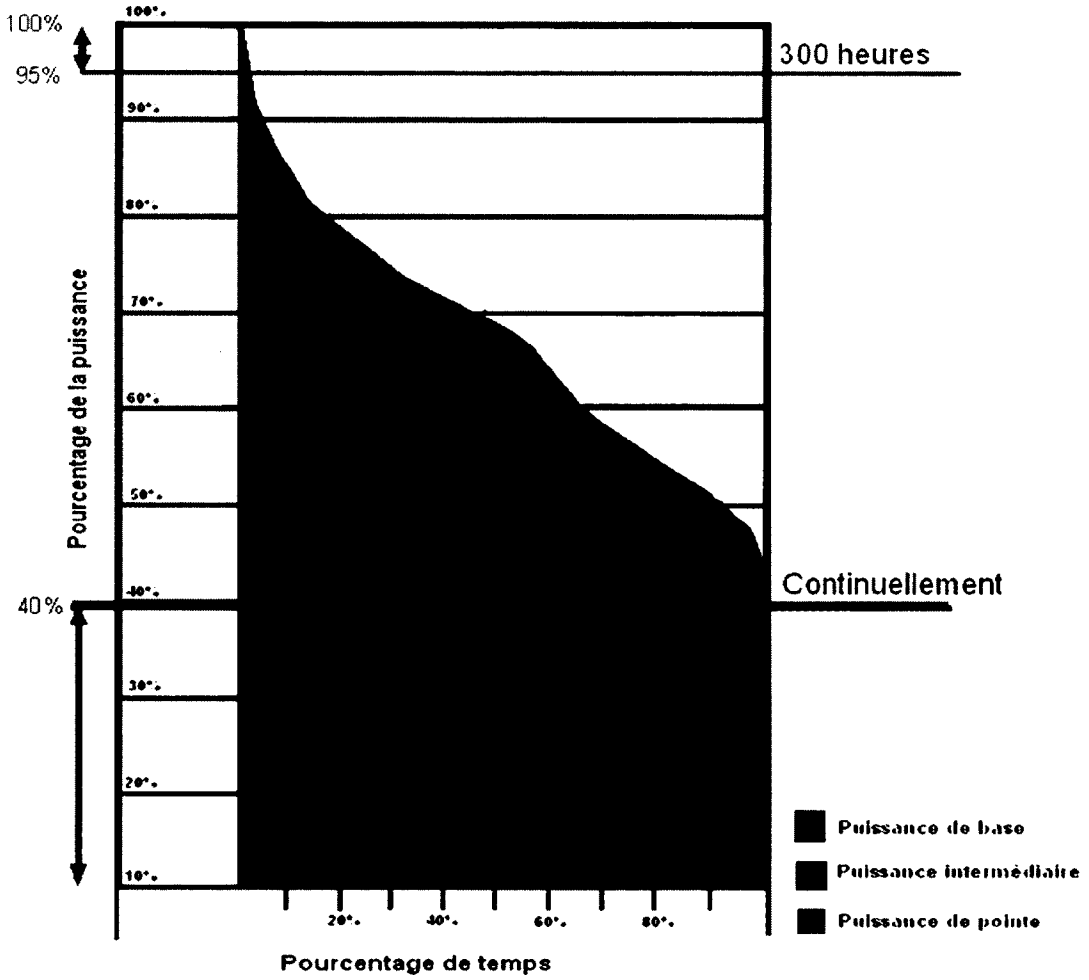


Figure 1: profil de la demande annuelle au Québec [1]

Aucune installation de production d'électricité ne peut fonctionner à plein régime tout le temps compte tenu des besoins de maintenance, des bris et de la disponibilité de la source d'énergie. Cependant, pour rentabiliser son investissement, la compagnie d'électricité cherche à avoir un

facteur d'utilisation aussi proche que possible de 1. Pour augmenter le facteur d'utilisation, une méthode possible est de réduire la pointe et niveler la courbe de consommation.

Pour mieux comprendre la situation, examinons le profil de demande quotidienne du réseau électrique (Figure 2) [1]. L'évolution de la demande au cours de la journée montre qu'il y a deux pointes : entre 11h00 et 12h00 ainsi qu'entre 17h00 et 18h00. Pendant ces heures, un nombre important d'appareils électriques fonctionnent en même temps.

Figure 2: profil de la consommation quotidienne au Québec [1]

Si, pendant les heures de pointes, on pouvait couper l'alimentation de certains appareils électriques (processus de délestage « load shedding ») et la rétablir pendant les périodes de faible demande (processus de déplacement de la pointe « load shifting »), on pourrait alors niveler le profil de demande. Des appareils électriques comme les chauffe-eaux et, dans une moindre mesure, les climatiseurs sont des candidats intéressants pour être gérés par un tel mécanisme. Car, dans le cas de ces appareils, l'arrêt de l'alimentation électrique de façon intermittente pour des durées limitées affecte peu le confort de l'utilisateur en raison des possibilités de stockage de l'énergie sous forme thermique qui leurs sont inhérentes. D'autres charges présentent également des caractéristiques de stockage de l'énergie, comme les

batteries des automobiles électriques ou hybrides, ce qui en font de bons candidats pour participer à un algorithme de gestion de la pointe.

1.1.1 Gestion de la charge « load management »

Du point de vue du distributeur, la gestion de la charge est l'ensemble des actions prises pour modifier la demande de l'électricité en fonction de contraintes financières et techniques reliées à la production et à la distribution de l'énergie.

La gestion de la charge est utilisée pour offrir différents services au réseau électrique comme la réduction de la pointe de consommation et l'équilibrage entre la production et la demande. Comme le réseau électrique doit être dimensionné pour supporter la pointe de la puissance, la réduction de la pointe permet de servir plus de clients en utilisant les mêmes installations. Ceci évite la construction des nouvelles centrales électriques pour répondre à la demande grandissante. Ceci permet également de réduire le coût de la production qui est plus élevé pendant les heures de pointes.

L'équilibre entre la production et la consommation doit être assuré en tout temps. Lorsqu'une panne d'électricité survient, l'opérateur du réseau peut délester « Load Shedding » certaines charges pour rétablir cet équilibre. Pour éviter de créer une autre pointe de consommation après la période de délestage, il faut répartir la consommation des charges pour niveler leur profil de consommation « Load Shifting ». Il est possible de gérer continuellement la consommation des charges pour créer des profils de consommation variable « Load Shaping ». Ceci peut être utilisé pour, par exemple, mieux intégrer les productions d'énergies fluctuantes.

Plusieurs programmes de « Demand Response » (DR) sont déjà implémentés et utilisés par les opérateurs du réseau [2]. Avec la DR, les charges doivent répondre à une demande de changement de comportements provenant de l'opérateur. Les programmes de DR peuvent agir selon des contraintes temporelles différentes : temps réel, court terme et moyen terme. La DR « temps réel » est, principalement, utilisée pour les cas d'urgences. La DR à court et moyen terme sont utilisés pour participer aux marchés de l'électricité et pour offrir des services auxiliaires « Ancillary services ».

Certaines compagnies d'électricité appliquent un tarif variable plus élevé pendant les heures de pointes. Ceci incite les clients à baisser leur consommation durant les heures de pointes et produit ainsi un effet d'autorégulation de l'amplitude de ces pointes. D'autres compagnies, imposent une limite de puissance maximale à consommer, elles facturent lourdement la puissance excédentaire utilisée et pénalisent ainsi ceux qui dépassent leur puissance souscrite. Des moyens dissuasifs moins contraignants peuvent également être utilisés comme, par exemple, une promotion via les médias de la réduction de la consommation électrique pendant certaines heures de la journée. Certaines compagnies ciblent des appareils spécifiques moins sensibles au délestage, comme les chauffe-eaux par exemple, et les contrôlent à distance. Les clients qui acceptent que leurs appareils soient contrôlés, bénéficient d'un rabais sur leurs factures d'électricité.

Le délestage est un moyen efficace pour réduire la pointe de consommation. Cependant, il est insuffisant de délester certains types de charges comme les chauffe-eaux, pendant seulement les heures de pointe. Car, après la période de délestage, les charges se trouvent affamées et commencent à consommer en même temps. Ceci crée une autre pointe de reprise de charge appelée PAYBACK.

Pour éviter la pointe de reprise de charge, les charges peuvent être divisées en plusieurs groupes. Chaque groupe est délesté d'une manière indépendante et en particulier peut l'être à une plage horaire spécifique. Avec ces techniques, il s'agit de rechercher la meilleure division des groupes i.e. nombre de groupes et nombre d'unités dans chaque groupe ainsi que le meilleur horaire i.e. heure d'interruption et durée d'interruption pour chaque groupe.

Le processus de déplacement de la pointe pourrait être appliqué sur différents types d'appareils électriques, surtout les appareils intelligents « smart appliances » qui sont capables de stocker l'énergie ou de différer leur fonctionnement; par exemple : le déplacement automatique du cycle de dégivrage d'un réfrigérateur, le déplacement de la consommation du lave-vaisselle ou de la sécheuse, ...

La gestion de la charge peut être étendue au delà de l'optimisation du facteur d'utilisation pour égaler la consommation à la production sur les réseaux qui intègrent des productions fluctuantes comme l'éolien et l'énergie solaire. En fait, le coût instantané de l'électricité varie selon la disponibilité de la production. Des charges capables d'accumuler de l'énergie peuvent

être utilisées pour stocker l'énergie quand elle est disponible (généralement à plus bas prix) et utiliser cette énergie pendant les heures de pointes (prix d'électricité plus élevé). Dans ce contexte, les charges modifient leur comportement en réponse à un signal traduisant la disponibilité de l'électricité souvent par le biais de son coût instantané.

Dans certains systèmes, les clients peuvent participer au marché de l'électricité via une agrégation de charge; plusieurs clients s'unissent pour former un groupe, ce groupe aura un pouvoir d'achat plus important et pourra offrir des engagements, en terme de pointe de consommation, au même titre que les grands clients industriels. Ces charges gèrent leur consommation afin de participer au marché et assurer le meilleur prix d'électricité.

Aujourd'hui, grâce aux technologies de l'information et de communication, la gestion de la charge peut être appliquée à chaque appareil individuellement et d'une façon personnalisée, une chose qui n'était pas possible avant. Il est devenu possible d'équiper des appareils électriques avec une interface de communication pour des applications de gestion de la charge plus sophistiquées.

Les techniques actuelles proposent une approche centralisée où une entité externe contrôle les équipements électriques qui se trouvent à l'intérieur d'un domicile. Ceci pose un problème de vie privée qui ne sont pas encore réglés dans le domaine de smart grid.

1.1.2 Réseau électrique intelligent « smart grid »

Actuellement, l'infrastructure du réseau électrique, dans plusieurs pays dont le Canada, se modernise. Cette modernisation doit permettre d'améliorer l'efficacité du réseau au niveau de la production, du transport, de la distribution ainsi que chez les consommateurs dans les maisons et bureaux. Elle doit permettre l'intégration au réseau électrique des énergies renouvelables (éolien, solaire, biomasse, ...). L'équilibrage des énergies renouvelables pourra être offert par une multitude de ressources distribuées comme la gestion de la demande [3]. La gestion de la demande est déjà utilisée pour offrir des services d'urgences, services économiques et services auxiliaires [2]. Cependant, la DR pour les services d'adaptation aux fluctuations de la production des ressources renouvelable est encore à ses débuts.

La nouvelle génération du réseau électrique doit aussi supporter les voitures électriques rechargeables. Pour amener le réseau électrique à cette modernisation, on fait appel aux technologies de l'information et de la télécommunication.

Les technologies de l'information et la télécommunication sont des domaines assez matures. Ils ont déjà fait leurs preuves d'efficacité et de fiabilité avec le guidage des satellites dans l'espace ou la mise en œuvre de l'intelligence des robots envoyés sur la planète Mars et qui effectuent des tâches de manière autonome. Ces technologies ont révolutionné beaucoup de domaines; leur application et intégration au domaine de la gestion a donné naissance à l'informatique de gestion, au domaine de la mécanique et de l'électronique a donné naissance à la Mécatronique, au réseau téléphonique a permis la création du service cellulaire ainsi que de la voix sur IP et de la vidéoconférence. Aujourd'hui, cette intégration et application au réseau électrique donne naissance au smart grid.

Le smart grid intègre au réseau électrique des composantes électroniques, logicielles et de communication. Ce n'est pas nécessairement une discipline distincte, mais plutôt un thème intégrateur (Figure 3).

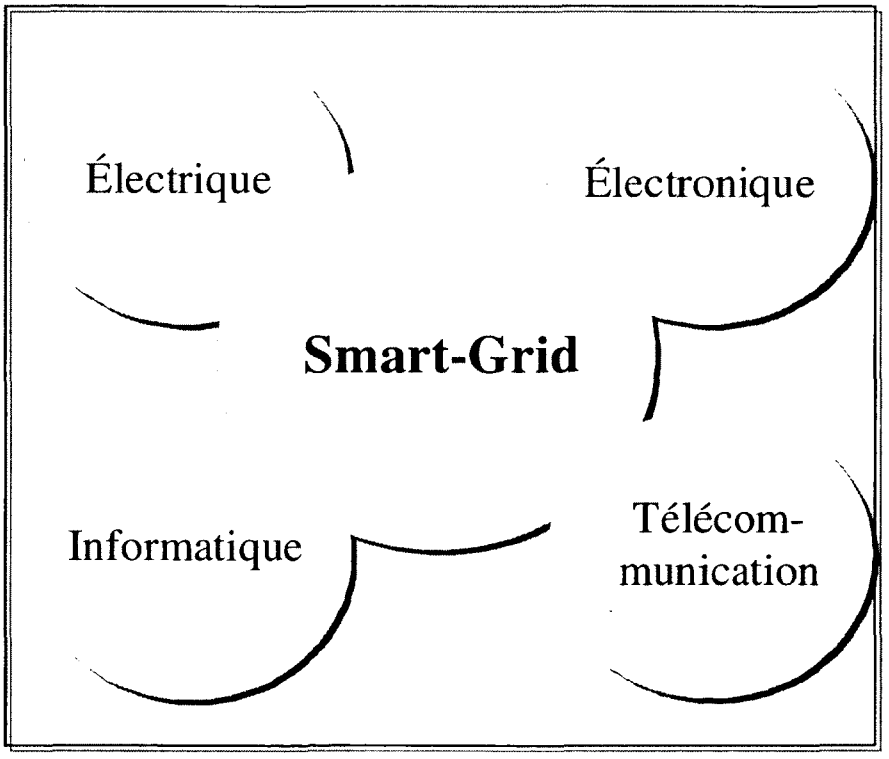


Figure 3: représentation du réseau électrique intelligent « smart grid »

La majorité des équipements utilisés aux différents niveaux du réseau électrique, seront informatisés et munis de capteurs et de moyens de communication bidirectionnelle. Les pannes, qui peuvent survenir aux différents niveaux du réseau, pourraient donc être détectées et gérées automatiquement.

Grâce aux compteurs intelligents, en voie d'installation dans plusieurs réseaux de distributions électriques, il sera possible de lire les compteurs à distance, d'appliquer des tarifs variables et de les diffuser chez le client [4]. Ce même compteur pourrait être utilisé pour des techniques de gestion de la charge; il pourrait recevoir du réseau les demandes de délestage et les relayer vers les appareils chez le client avec des technologies de réseautage sans fil comme ZigBee. Ce contrôle peut se faire par le biais d'une passerelle ou contrôleur résidentiel à l'intérieur de la maison ou le bureau comme le « Home Energy Management Gateway », « Home Area Network » ou autres. Les appareils peuvent être contrôlés selon des priorités préétablies.

C'est dans ce contexte qu'une nouvelle approche de gestion de la charge est présentée. Cette approche est complètement distribuée. Elle se base sur un principe de quota de consommation de puissance. L'algorithme distribué nécessite une communication directe entre les différentes charges. Un nouveau système de communication P2P appelé Ring-Tree est proposé pour permettre le partage d'informations entre un très grand nombre d'appareils. Le système est un réseau dédié « Overlay Network » qui utilise les services de transport de l'Internet ou tout autre infrastructure de communication déjà existante.

1.2 Utilité d'une nouvelle recherche

Le contrôle de la demande en fonction de la production est actuellement effectué de façon centralisée par l'intermédiaire de l'opérateur du réseau de distribution sous la forme de délestage généralement volontaire et supporté par un avantage tarifaire. Cette situation est en train d'évoluer considérablement pour différentes raisons :

- Utilisation de sources d'énergies fluctuantes comme l'éolien ou le solaire en complément des sources stables et contrôlables actuelles (nucléaire, thermique, hydro-électrique, ...)

- Introduction d'une tarification variable dans le temps pour inciter un ajustement décentralisé de la consommation
- Introduction des véhicules électriques rechargeables qui risquent de changer le profil de consommation typique et d'augmenter l'appel en puissance des clients

Actuellement c'est la consommation qui stimule la dynamique du réseau électrique, on évolue vers une situation où l'offre de production va prendre une part plus importante dans la conduite du réseau. Ceci nécessite donc une interaction plus importante avec les éléments consommateurs, surtout ceux qui ont la possibilité d'accumuler de l'énergie tel que : chauffe-eaux, certains chauffages, véhicules électriques, etc.

Les technologies de communication envisageables pour permettre l'interaction avec les dispositifs consommateurs sont soit dédiées et à déployer généralement via un investissement du distributeur d'électricité ou bien elles doivent s'appuyer sur l'existant, c'est-à-dire principalement Internet ou le réseau cellulaire.

Les technologies existantes sur Internet sont du type P2P, souvent propriétaires, et sont mal adaptées pour un temps de réponse de type temps-réel de quelque secondes à quelques minutes, avec un très grand nombre de nœuds (jusqu'à quelques millions). De plus elles sont mises en œuvre sur des ordinateurs personnels assez puissants sans limites de ressources.

1.3 Objectifs visés par cette recherche

Le premier objectif est de concevoir une technique distribuée de gestion de la consommation des charges électriquesqui sera adaptée aux besoins du distributeur, acceptable par le client et qui minimise, en termes de puissance, la reprise de charge sur le réseau.

Un autre objectif, est de concevoir un réseau de communication qui permet l'interconnexion via une infrastructure de communication existante telle que l'Internet, d'un grand nombre d'appareils pour les faire collaborer entre eux dans un but précis. Par exemple un ensemble de charges contrôlables tel que des unités de chauffage à accumulation ou des chauffe-eaux résidentiels qui collaborent entre eux pour éviter une pointe de consommation ou pour équilibrer une production d'énergie électrique fluctuante comme une ferme d'éoliennes.

1.4 Contributions scientifiques des travaux de recherche

Cette thèse présente une nouvelle approche distribuée de gestion de la charge. Des appareils électriques, tels que les chauffe-eaux d'une ville entière, partagent un quota de consommation de puissance et décident d'une façon autonome d'arrêter ou non leur consommation, sans l'intervention de la compagnie d'électricité. Pour prendre cette décision, ils forment, de façon automatique, une communauté et ils échangent de l'information en utilisant un réseau dédié « overlay network ».

Une telle structure de réseau dédié, avec un nombre de nœuds (appareils électriques) potentiels aussi élevé et qui puisse être mise en œuvre avec des ressources limitées au niveau de chaque nœud, n'existait pas. La thèse présente donc un nouveau réseau dédié P2P appelé Ring-Tree qui est en accord avec les besoins visés. Le système Ring-Tree définit une nouvelle topologie. Il est supporté par des algorithmes en permettant la création, l'exploitation et la maintenance.

La solution proposée (Figure 4), est intéressante car elle ne requiert pas une nouvelle infrastructure de communication. Des équipements électriques munis d'un système embarqué réseauté, communiquent via Internet ou via n'importe quel autre réseau de communication existant.

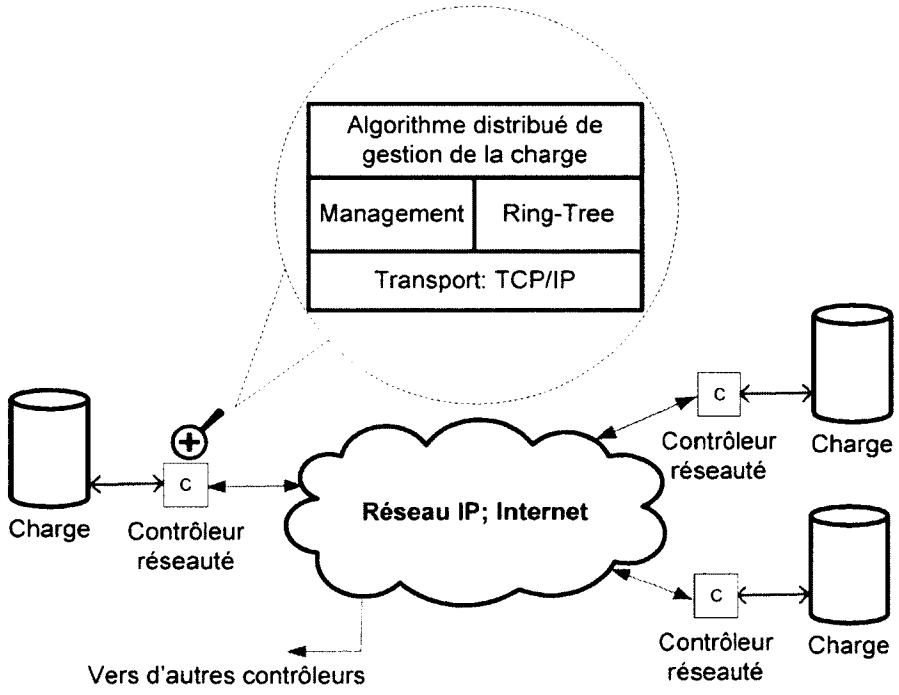


Figure 4: vue globale du système Ring-Tree et de l'algorithme de gestion de la charge

Une autre particularité intéressante est la gestion de la charge en continu pendant les 24 heures de la journée par opposition à un contrôle appliqué uniquement pendant les heures de pointes. Ceci permet un contrôle plus efficace de la reprise de charge. Il permettrait à un groupe de charges de participer à un marché de l'électricité où le prix varie continuellement selon la disponibilité de la production. Il permettrait également un équilibre en temps réel entre la consommation et la production même si celle-ci est très fluctuante.

1.5 Plan du document

Une technique de contrôle distribuée a été conçue et développée. Cette technique inclut un algorithme distribué qui s'exécute au niveau de la charge électrique et un quota de consommation variable dans le temps calculé selon le besoin du réseau électrique, la capacité des charges et le confort de l'usager. Pour valider le fonctionnement de la technique distribuée un simulateur, qui inclut un modèle mathématique de chauffe-eaux issu de la litérature, a été développé. L'algorithme distribué, la formulation mathématique des conditions nécessaires au cacul du quota, les détails du modèle de chauffe-eau ainsi que les résultats de simulation sont publiés dans un article présenté au CHAPITRE 2. La conception d'un système de communication P2P appelé Ring-Tree, supportant une communauté pouvant comporter des millions de membres, ainsi que des algorithmes de création, d'exploitation et de maintenance de l'architecture sont présentés au CHAPITRE 3. La mise en œuvre des solutions proposées dans cette thèse incluant le Ring-Tree et l'algorithme distribué basé sur une méthode d'histogramme est présentée au CHAPITRE 4. Le CHAPITRE 5 présente des suites possibles de ce projet ainsi que des propositions d'autres sujets de recherche en lien avec cette thèse et une conclusion.

CHAPITRE 2 Distributed Load Management Technique for a Community of Electric Loads

Avant-propos

Auteurs et affiliation :

S. Ayoub : candidat au doctorat, Université de Sherbrooke, Faculté de génie, département de génie électrique et de génie informatique.

P. Mabilleau : professeur titulaire, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

A. Prieur et D. Beauvais : chefs de projets de smart grid, CanmetÉNERGIE, Ressources Naturelles Canada, Varennes, Québec

A. Moreau : chercheur, Hydro-Québec, Institut de Recherche, LTE, Shawinigan, Québec

Date de soumission : 23 juin 2012

Revue : IEEE transactions on smart grid

Titre français : Algorithme Distribué de Gestion de la Charge d'une Communauté de Charge Électrique

Contribution au document :

Cet article présente la formulation mathématique, en utilisant la théorie des ensembles, de l'algorithme distribué de gestion de la consommation des charges électriques, basé sur un quota variable dans le temps, ainsi que le développement d'un simulateur de chauffe-eaux, comme exemple de charge, et les résultats de simulation du contrôle distribué appliqué à une communauté de chauffe-eaux pour changer son profil de consommation en puissance.

Contribution des auteurs :

L'auteur de la thèse, S. Ayoub, a conçu la technique de contrôle distribuée et a développé sa formulation mathématique (quota et algorithme local). Il a également réalisé les simulations menant aux résultats présentés en utilisant un simulateur qu'il a développé en Java. Le professeur P. Mabilleau a dirigé les travaux présentés dans l'article et a, conjointement avec l'auteur, élaboré les différents algorithmes. Les contributions de M. Prieur et de M. Beauvais ont permis de mieux expliquer, présenter

et situer la solution dans un contexte d'applications réelles et celle de M. Moreau a permis d'améliorer la simulation des chauffe-eaux et de valider les résultats du contrôle distribué.

Résumé français :

Pour gérer la consommation électrique des appareils communicants au sein d'une communauté où la décision de consommation est prise par chacun d'entre eux, une technique de gestion de la charge a été conçue et développée. cette technique permet à un groupe de charges électriques d'engendrer n'importe quel profil de consommation respectant certaines limites physiques des appareils. Ces appareils électriques gèrent leur consommation de façon à suivre un quota de consommation variable dans le temps respectant des conditions formulées. Les résultats de la simulation ont montré que l'algorithme distribué combiné avec du stockage de l'énergie permet à une communauté de chauffe-eaux de suivre un quota constant pour niveler leur profil typique de consommation, sans nuire à la disponibilité en eau chaude pour le client.

2.1 Introduction

The advanced smart grid is expected to take full advantage of the flexibility available from electric loads, generators and storage devices. In power systems, demand side participation, such as Demand Response (DR), distributed generation and on-site storage have many advantages such as mitigating price spikes and lines congestions [6]–[9]. Many large industrial and commercial customers are already engaged in DR wholesale programs in order to provide energy, reserve, capacity and regulation services [10]. By modifying the load profile, DR programs lead to significant reductions in the electricity price [11], [12].While the wholesale market is open to large consumers and generator of electricity in many regions, the participation of residential customers or smaller resources in the market is still at its inception.

Among the different DR strategies, the uses of price signals are being considered to induce a shift or a reduction in power consumption during high price of electricity [13]–[16]. To ensure greater participation of the residential sector, new energy management technologies and smart appliances are also being considered [17]. In [18] it is showed that the introduction into the market of smart appliances with embedded communication is possible when purchase incentives are in place. Furthermore, many studies [19]–[22] focused on integration of small resources in power system operations and markets. When the loads of several customers are aggregated, it is possible to flatten the overall load profile, obtain a higher load factor, and ultimately, lower per unit energy costs for all members of the group [23].

Under the current wholesale DR programs available in North America, the resource deployment is done using bulk or resource-specific load control and through customer self-dispatches using frequency sensitive relays. Several publications in that area are building on the centralized approach to propose optimal ways of performing the economic dispatch and the deployment of the resources [24]–[28]. In [29] a population of thermostatically controlled loads is made to regulate the output of a wind plant by varying the nominal thermostat temperature set points. In typical utility DR program targeting residential customers, the same approach, based on a centralized intelligence, is also being used. For example, a control signal is sent, forcing the cycling of a load, a new set point on a thermostat or the activation of a Electric Thermal Storage device [30]. Under such programs, the technology installed would

only support one way flow of information, a signal sent from the operator to the load and no feedback from individual resource.

In [31] user comfort is considered as one of the system constraint. Considering that DR operations in the residential sector is mostly based on a one way, point-to-multipoint communication, the setting up of a two-way communication channel between the operator and each individual load may be challenging. Also, should feedback information be available for each load, there may be concerns about privacy of the personal information shared with the control centre. To solve the technical challenge of having two-way flow of information and protecting data privacy, distributed algorithms and peer-to-peer communication are being proposed.

The peer-to-peer (P2P) concept [32] is well known and it is used for several information technology and communication applications: video and voice calls, file-sharing applications, and many others [33], [34]. The application of P2P and decentralized algorithms as the foundational information technology infrastructures of the future smart grid is discussed in [35]. Also, in [36] a distributed approach for energy storage was proposed.

This paper proposes a distributed load management approach, combined with energy storage capabilities, to manage the consumption of a community of electric loads. These loads exchange information using a communication network following a peer-to-peer approach. The goal pursued by the P2P community is to increase its load factor. A load levelling application is therefore used to flatten the usual peaks and valleys of a non-coordinated consumption. A fixed consumption quota is established a priori, chosen to ensure lowest peak demand while ensuring that users' comfort is not affected. Should variable quota (time dependent) being set, this would, to the extent of some physical limitations, create a desired consumption profile.

The next Section presents the design concept of this new approach. Section 2.3 presents the problem formulation and the distributed algorithm solution. As an example, the distributed load management technique is detailed using an electrical water heater load. The elemental electric water heater model used in the simulation is presented in Section 2.4. The simulation results and discussion, including the benefits of thermal energy storage, are presented in Section 2.5. The conclusion is presented in Section 2.6.

2.2 New Design Concept

The new design concept starts with building, in a distributed way, a community of devices such as smart appliances, energy management systems or any end-load of a specific region (feeder, substation, city, power area) (Figure 5). When the community is created, each device capable of computing and communicating would join the virtual community using any existent network infrastructure such as the Internet or a private or public IP based communication network. Within this community, the loads would share information on the state (i.e. temperature, battery level, power demand, etc.) of every member and those members would self-dispatch their consumption without direct load control from the power system operator.

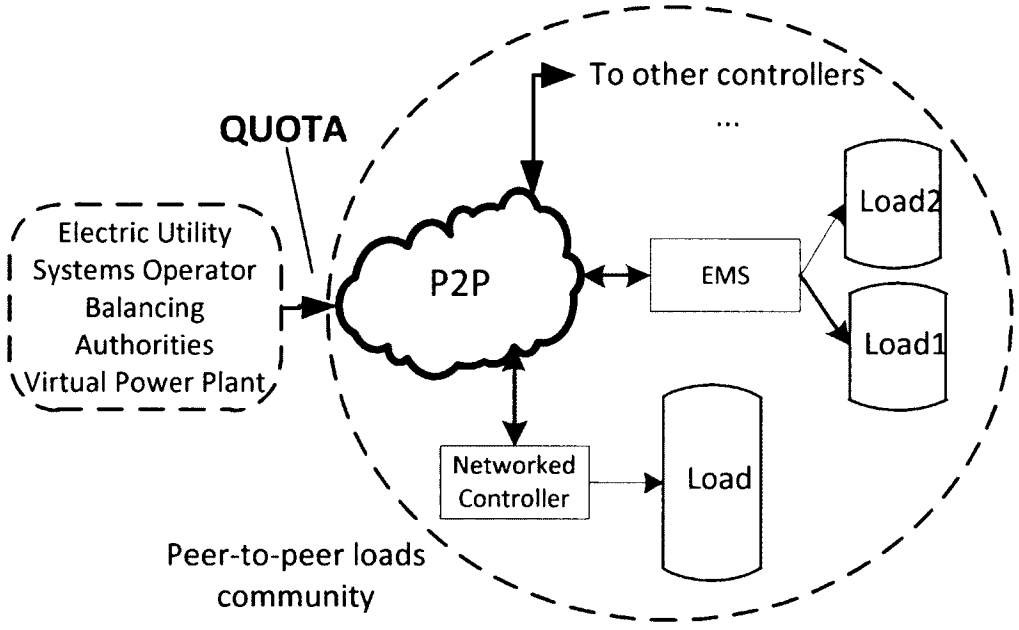


Figure 5: example of an overlay P2P network for smart grid applications

When compared to a centralized approach, the P2P has many advantages. Two-way flow of information is facilitated by the architecture of the IP-based network and the P2P communication scheme. It is robust and has high fault tolerance. There is no single point of failure to make sure that the whole network (control and communication) would not be compromised by single or multiple faults. Under such architecture, adding more nodes increases the load management potential and decreases the likelihood of failure. Loads join and leave the group dynamically. Privacy is part of the design since the personal information is not shared to a central entity.

To use of P2P communication for smart-grid applications, the overlay network must have, at least, the following characteristics:

- The resources required on each node (bandwidth, storage space, and computing power) must be limited in accordance with the limited resources available on each communicating device
- It must allow Multicast and Broadcast communication between a large number of nodes (up to millions)
- Smart-Grid applications, such as the distributed technique proposed in this paper, require exchanging small to medium size messages; the transmission Broadcast time between millions of nodes must be less than 1 minute

To flatten the overall load profile, a fixed consumption quota is applied. The appliances control their consumption in order to respect the quota by allowing the appliances, having the highest need, to consume first. Hence only a limited amount of power consumption is allowed during a period of time. With this new approach, the comfort of the customer is prioritized. The quota can be provided by a coordinator, an ISO, an electric utility or other authority, but the control decision is distributed and made by the appliances.

An algorithm has been developed so that the decision to turn the appliance on and off is taken by the individual load controller using the information exchanged with other loads about their electrical energy needs and the received quota proportional to the general load on the grid. In the specific case of water heaters, flattening the overall load profile reduces the peak demand because the highest demand of water heaters is coincident with the peak's grid. To store energy in thermal form, the thermostat set points are modified.

2.3 Problem Formulation and Distributed Algorithm

In this section the distributed technique is formulated. The objective function and constraints that help determine the quota's value are detailed.

2.3.1 Distributed Load Management Algorithm

A community is defined as a set N of nodes ξ and having a cardinality of $|N| = n(t)$ where t is the time; the size of the community $n(t)$ varies when a node joins or leaves the community.

In practice, a node ξ is a smart appliance, an energy management gateway, or any networked controller capable of computing, communicating, and controlling at least one load. However, in this paper a node controls only one load. Therefore, the terms node and load can be used interchangeably.

Each load has a state parameter $x(t)$ that represents its actual need for consumption. The state parameter, also called need parameter, is defined according to the type of the load; for example the state parameter of a water heater or an Electric Thermal Storage is the temperature, and the state parameter of a Plug in Electric Vehicle is the battery charge level. In these cases, a low value of the state parameter corresponds to highest consumption need. Hence, all the loads in the community can be ordered according to their state parameter.

A number of nodes $q(t)$ having the lowest state parameter values within the community N, constitutes a subset Q of the set N; the value of q varies with time according to the quota's value of the permitted power consumption.

The distributed algorithm in each node allows determining if a load belongs to the subset Q or not. A load belongs to the subset Q if its state parameter value is less than a threshold parameter value $x_{th}(t)$ and if the limited size $q(t)$ of Q is respected. A threshold parameter value $x_{th}(t)$ is defined as follows: if $x_i(t)$ is the state parameter of node ξ_i and $x_j(t)$ is the state parameter of node ξ_j then

$$\exists\, x_{th}(t): x_{th}(t) < x_i(t) \text{ and } x_{th}(t) > x_j(t)$$

$$\forall\, \xi_i \in Q^c \text{ and } \forall\, \xi_j \in Q \qquad (1)$$

where Q^c is the complement of Q in N.

The nodes in Q are the only nodes allowed to consume. By varying the value of $q(t)$ with time, the consumption power profile is reshaped according to $q(t)$. Knowing the quota's

value, the nodes exchange information about their state parameter in order to build the subset Q, and to determine the threshold parameter value $x_{th}(t)$.

Two types of quotas are defined: optional and mandatory. Loads respect the optional quota until their state parameter reaches x_{min}, then they start consume without respecting the quota as it is optional. The value of x_{min} is determined according to user's comfort. In mandatory quota, loads respect the quota without considering the value of x_{min}. The mandatory type of quota is used in order to reduce the load temporarily in response to an emergency grid condition initiated by a request from ISO or the electric utility, to avoid blackouts for example. In such situation the security of the grid is more important than user's comfort. To turn off all the appliances, a mandatory quota fixed to zero can be used.

The control is applied all the time and not only during peak periods. The time is divided into cycles. The duration of a cycle can be set, for example, to 1, 5 or 15 minutes. A new decision for consumption is made by every node at the end of each cycle. The length of a cycle can be variable in time.

Every cycle has the following actions:

1. Ordering, in ascending order, all the nodes of the set N according to the state parameter
2. Calculating the value of the power consumption quota according to load forecasting, storage capabilities, state of the loads, and generation availability with respect to the conditions formulated in sub section 2.3.3.
3. Quota's value reception by all the nodes
4. Subset Q construction by using the first q values in the ordered list
5. Determining the value of the threshold x_{th} which is higher than the highest value in the subset Q and lower than the lowest value in Q^c
6. Comparison by every node k of its state parameter x_k to the threshold x_{th} and to x_{min}
7. Decision for electric consumption is made according to x_{th}, x_{min}, and the type of quota
8. End of cycle

2.3.2 Objective

The objective is to change the behaviour of the appliances (nodes) of the community in a way that their power consumption profile meets the quotas' values at all time.

$$\sum_{k=1}^{q} p_k(t) = quota$$

And

$$\sum_{k=q+1}^{n} p_k(t) = 0 \quad (2)$$

where the loads are in ascending order according to the state parameter, $p_k(t)$ is the power consumption of load k at time t, $q = |Q|$, and $n = |N|$ hence,

$$\forall\, \xi_j \in Q, p_j(t) \neq 0 \text{ and } \forall\, \xi_i \in Q^c, p_i(t) = 0. \quad (3)$$

There is an obvious relation between the quota, which is the amount of power consumption permitted, and q(t) the cardinality of Q. If all the loads are identical i.e. have the same power rating r. The objective equation becomes:

$$\sum_{k=1}^{n} p_k(t) \leq r\, q(t). \quad (4)$$

Therefore, varying the quota's value reshapes the load profile of the community N by allowing the loads with highest need to consume first. Consequently, any power consumption profile can be created, if the variable quota respects the following conditions.

2.3.3 Constraints

If, at time t, the quota's value is higher than the power that can be consumed by all the loads, the subset Q will contain loads incapable of consuming power (i.e. $\xi_j \in Q$ and $p_j(t) = 0$) and the objective equation is not respected. To avoid this situation the quota's value must be determined according to condition 1:

$$\text{Condition 1: } q(t) < n(t) - s(t) \quad (5)$$

where s(t) is the cardinality of a set S of all the saturated loads within the community, saturated load is defined as a load having a state parameter value higher or equal to a

maximum value i.e. $x_k \geq x_{max}$ (Tableau I); x_{max} is determined according to the storage capacity of the load. Condition 1 can be presented in terms of power as follows:

$$quota < \sum_{i=1}^{n} r_i(t)\big(1 - s_i(t)\big) \tag{6}$$

$$s_i(t) = \begin{cases} 1 & if\ x(t) \geq x_{max} \\ 0 & elsewhere \end{cases} \tag{7}$$

where r_i is the power of load i.

Tableau I : load status definition

$X_k < x_{min}$	Unsatisfied load
$x_k \geq x_{max}$	Saturated load
$x_{min} \leq x_k < x_{max}$	Responsive load

Next, if the quota's value is lower than the power needed by the unsatisfied loads, some unsatisfied loads will overflow the subset Q (an unsatisfied load is defined as a load having a state parameter value less than a minimum value i.e. $x_k < x_{min}$ and x_{min} is determined according to the user's comfort level); if the quota is mandatory, the comfort level of the users having their loads situated outside Q will be affected; if the quota is optional these unsatisfied node will consume without respecting the quota (i.e. $\xi_i \in Q^c$ and $p_i(t) \neq 0$) and the objective equation will not be respected. To avoid this situation the value of an optional quota must be determined according to condition 2:

$$\text{Condition 2: } q(t) > u(t) \tag{8}$$

where u(t) is the cardinality of a set U of all the unsatisfied loads within the community. Condition 2 can be presented in terms of power as follows:

$$quota > \sum_{i=1}^{n} r_i(t)\, u_i(t) \tag{9}$$

$$u_i(t) = \begin{cases} 1 & if\ x(t) < x_{min} \\ 0 & elsewhere \end{cases} \tag{10}$$

where r_i is the power of load i.

Satisfaction of conditions 1 and 2 implies that:

$$U \cap Q^c = \emptyset \text{ and } S \cap Q = \emptyset. \tag{11}$$

Having more storage capabilities reduces the limitation of condition 1, and using a mandatory quota overrides the constraints of condition 2 but it may affect user's comfort.

When user comfort is prioritized and no energy is stored at initial time, the following fact can be expressed:

"the energy consumed by a load after control, over a period of time must be higher or equal to the minimum energy needed by the load, over the same period of time."

The minimum energy needed by a load is the smallest amount of energy required to keep user's comfort or proper functioning of the load. This is an important assumption because it sets a lower limit for the quota's value over a period of time. Hence, the following condition is formulated:

$$\text{Condition 3: } quota \geq \frac{\sum_{h=1}^{T}\sum_{k=1}^{n} p_k(h)}{T} \tag{12}$$

where $p_k(h)$ is the power consumption of load k during the period of time h, $h = 1..T$.

If the quota is constant over a period of time T, a flat demand profile is created when the quota is met at all time. In this case, the best possible theoretical result is achieved when an optional fixed quota is respected and equal to

$$quota = \frac{\sum_{h=1}^{T}\sum_{k=1}^{n} p_k(h)}{T}. \tag{13}$$

The importance of this value is that the energy needed over a period of time T doesn't change, the user's comfort is not affected, and the overall power profile is flattened over T. The value of the quota is equal to the average power consumption during the period of time T. In the following case study, a constant quota equal to the average power over T=24 hours is to be met by a community of loads.

2.4 Water Heater Case Study

The distributed algorithm described in Section III is applied on an electrical water heater load as an example. A fixed quota is used during 24 hours in order to flatten the load profile of a community of water heaters. The goal is to reach the best theoretical result as defined in Section III. The next sub section presents the electric water heater model used to simulate a community of water heaters, i.e. distributed loads. Sub section B presents the simulation of a community of uncontrolled water heaters. At last, sub section 2.4.3 presents the distributed technique applied on the water heaters.

2.4.1 Elemental Electric Water Heating Stochastic Model

The water temperature in a water heater is very stratified; the hot water temperature exiting from the top of the tank is very different from the average temperature. To simplify the simulation, a water heater model with average hot water temperature is used. The latter is evaluated by the differential equation (14) based on work in [27].

Hot water's temperature must be monitored continuously. For a given water tank size, solving the differential equation (14), provides the temperature x(t) at any moment. Equation (14) describes the discrete state dynamics in $^{\circ}\ C/min$, of the water heater [27].

$$\frac{dx(t)}{dt} = -a[x(t) - x_a(t)] - A(t)\kappa(t) + Rm(t)b(t) \qquad (14)$$

where x(t) is the average tank water temperature at time t, $x_a(t)$ is the ambient temperature, a is the thermal resistance of the tank walls, A(t) is the rate of energy extraction, in $^{\circ}\ C/min$, when water is in demand, κ (t) is the normalized (1-0) hot water demand jump process, R is the power rating of the heating element in $^{\circ}\ C/min$, m(t) is the thermostat binary state (1 for On, 0 for Off) and b(t) is the on-off control to change the behaviour of the water heater. The switching of κ (t) is characterized by the following transition probabilities [27]:

$$Pr[\kappa(t+h) = 1|\kappa(t) = 0] = \alpha_0 h \qquad (15)$$

$$Pr[\kappa(t+h) = 0|\kappa(t) = 1] = \alpha_1 h \qquad (16)$$

where α_0 and α_1 are positive and h is an infinitesimal time increment.

After solving (14), (17) is obtained:

$$x(t) = e^{-at}\left[\int_0^t e^{a\tau}(-A(\tau)\kappa(\tau) + Rm(\tau)b(\tau) + ax_a(\tau))d\tau + x_0\right] \quad (17)$$

Where x_0 is determined at t = 0. To simulate the dynamic water demand over 24 hours α_1's value is constant and α_0's value changes every 30 minutes. It is considered that α_0's variations follow the same curve as the daily hot water demand curve; when the hot water demand is high, the probability to extract hot water is also high, and when the hot water demand is low, the probability to extract hot water is also low. This technique considers that the probability for the consumer to start extracting hot water varies in time following the curve of the electricity demand and the consumer stops using water with a constant probability.

It is also considered that the curve of daily hot water demand is similar to the electric consumption curve. Hence, α_0 also follows the same curve as the electric consumption. This technique allows generating any consumption profile by varying properly α_0's values over 24 hours. Typical values of parameters α_0 and α_1 are given in [28].

2.4.2 Water Heater Simulation

Equation (17) was programmed for households of type A as defined by [27], where type A corresponds to families of 3 people with a 52 US gallon sized water tank and 4.5kW heating elements. The parameters of the water heater model associated to this type are [27]:

$A = 1.29\,^{\circ}\,C/min$, $a = 0.000156\text{min}^{-1}$, $R = 0.3279\,^{\circ}\,C/min$, $\alpha_0 = 0.012min^{-1}$, $\alpha_1 = 0.32min^{-1}$.

The parameter b(t) of the elemental water heater model permits to control the electrical consumption of each simulated water heater. The time increment h is set to 1 min. The 48 α_0's values are calculated according to the profile to be simulated. The given α_0's value used is an average.

All the following results are based on simulation of 10,000 water heaters and normalized average profiles are analyzed.

2.4.3 Distributed Algorithm Applied on Water Heaters

Depending on load forecasting, potential of energy storage, state of the loads, the availability of generation, and/or market price, a power consumption quota is determined. A time dependent quota that respects the conditions discussed earlier allows creating any load profile. In this paper, the objective is to flatten the overall load profile of all the loads. Hence, a constant quota is used over 24 hours. To study the quota's value effect on the results, several simulations were run using different quota's value. The results are showed in Section 2.5

Every t minutes the hot water temperature (state parameter) x(t) is measured and a decision is made. Only water heaters having the lowest temperature consume electricity. The total number of water heaters operating at the same time is equal to the quota's value. Other water heaters will turn off. If a water heater's temperature drops to the minimum temperature comfort level x_{min}, the water heater starts consumption independently of an optional quota. This case happens when the quota's value doesn't respect condition 2, the optional quota is then not respected and a peak reappears, as shown in Section 2.5

One way to implement the algorithm is to have each node (i.e. water heater) to receive, via the data network, from other nodes of the community a list of controller identification numbers (ID) ordered by temperature. Each node receives information about a limited number q(t) of water heaters having the lowest temperature. A given water heater decides to consume only if there is a need for consumption (i.e. its temperature is under x_{max}) and it has a lower hot water temperature than at least N-q other water heaters' temperature. Water heaters having the lowest temperature always have the priority to consume.

In a set N of n nodes participating in load management program, there is a subset Q of q loads having the lowest state parameter value where $N = Q \cup Q^c$. Each element in the set N is composed by an ordered pair (x(t), ID) where ID is a unique node identification number. For this case study, n(t) and q(t) are time invariant.

$$N = \{(x(t), ID)\ of\ all\ loads\}. \tag{18}$$

Elements in Q have higher priority to use the energy because their temperature is lower than those of Q^c. In this case a water heater ξ_i starts consumption only if the following condition is satisfied.

$$\left[(x_i(t) < x_{max}) \wedge \left((x_i(t), ID) \in Q\right)\right] \vee [x_i(t) < x_{min}]. \quad (19)$$

In mandatory quota the water heater must respect the quota independently of hot water temperature. In this case the quota is always respected and ξ_i starts consumption only if the following condition is satisfied.

$$\left[(x_i(t) < x_{max}) \wedge \left((x_i(t), ID) \in Q\right)\right]. \quad (20)$$

According to the distributed algorithm based on the threshold value described in Section II, the power consumption of the water heater is driven by its parameters as follows:

$$p_i(t) > 0 \; if \; [b_i(t) = 1 \; and \; m_i(t) = 1]. \quad (21)$$

The value of $m_i(t)$ varies according to the following:

$$m_i(t+1) = \begin{cases} 0 & if \; x_i(t) > x_{max} \\ 1 & elsewhere \end{cases}. \quad (22)$$

When the quota is mandatory, the value of $b_i(t)$ varies according to the following:

$$b_i(t+1) = \begin{cases} 1 & if \; x_i(t) < x_{th}(t) \\ 0 & elsewhere \end{cases}. \quad (23)$$

When the quota is optional, the value of b(t) varies according to the following:

$$b_i(t+1) = \begin{cases} 1 & if \; x_i(t) < x_{th}(t) \; or \; x_i(t) < x_{min} \\ 0 & elsewhere \end{cases}. \quad (24)$$

In order to determine $x_{th}(t)$ a histogram of temperature is constructed by all the nodes communicating via the P2P network. The values under $x_{min}(t)$ are the unsatisfied nodes, and the values over $x_{max}(t)$ are the saturated nodes. The threshold value $x_{th}(t)$ is determined using the information in the histogram and the quota's value. The use of a histogram minimizes the size of the messages exchanged between the nodes because there is no need to exchange a list of pairs (x(t), ID) but only a table of temperature distribution.

2.5 Simulation results

Experimental data [38], collected during eight months from 51 dwellings, were used to create a realistic profile for average daily hot water consumption. This data was used to determine the values of α_0 as discussed earlier and the values of κ (t) in turn. To generate a typical

power consumption profile similar to the one presented in [39], 10,000 water heaters were simulated over 24 hours (Figure 6 uncontrolled profile). The normalized energy consumption is around 14 kWh per day.

2.5.1 Results of Load Management with Distributed Algorithm

The objective here is to have an unvarying power distribution of the domestic hot water load over 24 hours. In order to achieve this, a fixed quota that respects conditions 1, 2 and 3 should be used. When the distributed algorithm is applied to the realistic profile (Figure 6), using a normalized fixed quota=0.72kW, the normalized peak demand 1.2 kW is reduced by 40%. Quota's value doesn't respect condition 2. The optional quota is not respected between 1AM and 7AM because the saturated loads weren't able to use all the energy allowed by the quota.

Condition 3 stated that constant power consumption quota must be higher or equal than the energy needed by a community over a period of time divided by this period of time. This value corresponds to the mean value over 24 hours of the realistic profile and it is equal to 0.6kW $\left(\frac{14}{24} \approx 0.6\right)$. Any constant quota having values lower than 0.6kW will be exceeded. Using an optional quota, it is not possible to force the appliances to consume less than the minimum energy needed during 24 hours. Therefore, the value of the quota must be equal or higher than the forecasted mean value of a load profile.

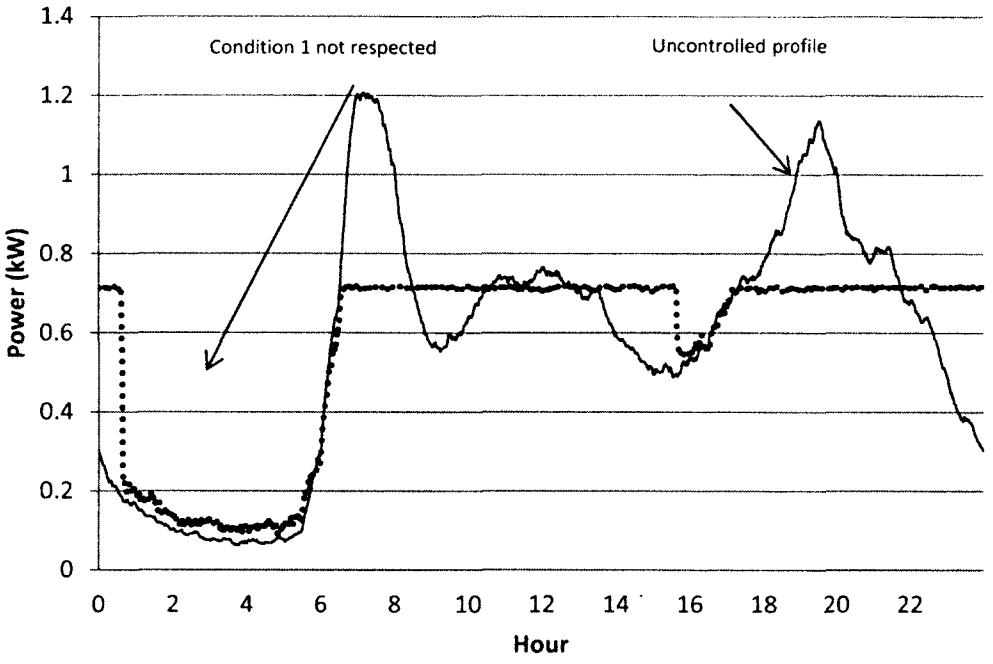


Figure 6 : average power profile, normalized quota=0.72kW, $x_{min} = 60°$ C, $x_{max} = 65°$ C

When an optional quota of exactly 0.6kW was used, conditions 1 and 2 weren't satisfied ().

The simulation confirmed that, as stated in condition 3, all quotas lower than 0.6kW were exceeded. When an optional quota higher or equal to 0.72 was applied (Figure 8), the peak reduction decreased when increasing the quota's value and condition 1 wasn't respected at all time.

Several simulations using different values of quotas were processed (Figure 9). The target quotas were reached only when the quota passes a certain value (0.7kW).

Condition 3 is not yet verified. By using energy storage it is possible to use a quota equal to the average power over 24 hours (0.6kW) as shown in the next sub section.

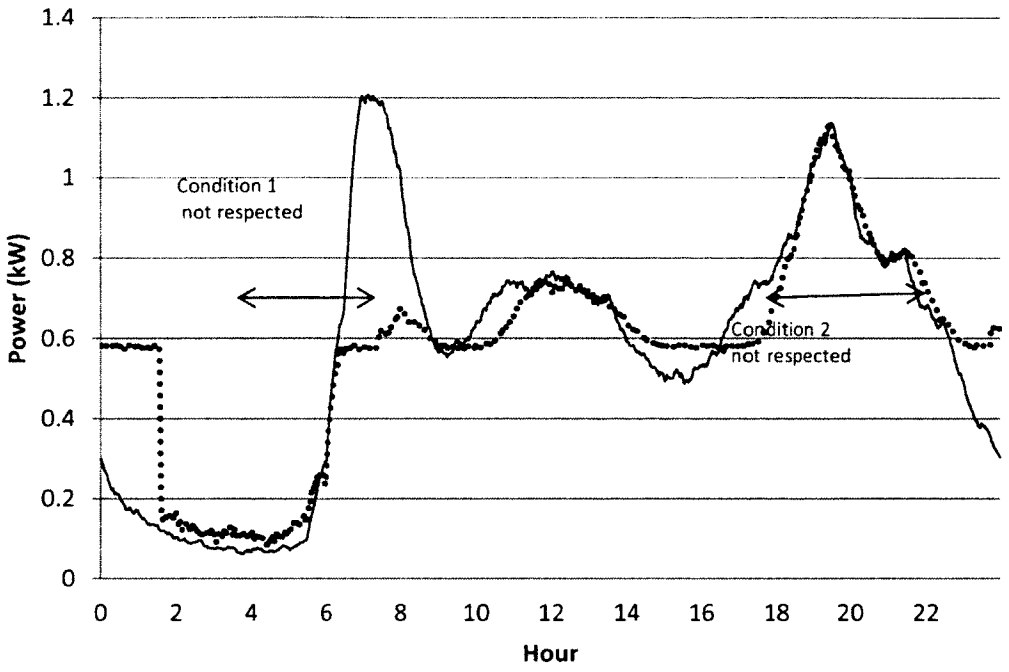


Figure 7: average power profile, normalized quota=0.6kW, $x_{min} = 60°$ C, $x_{max} = 65°$ C

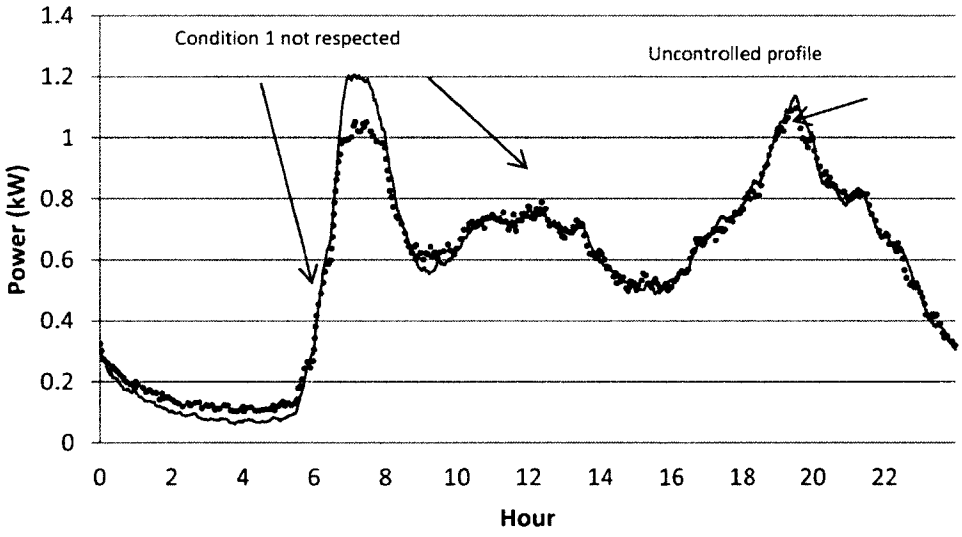


Figure 8: average power profile, normalized quota=1.26kW, $x_{min} = 60°$ C, $x_{max} = 65°$ C

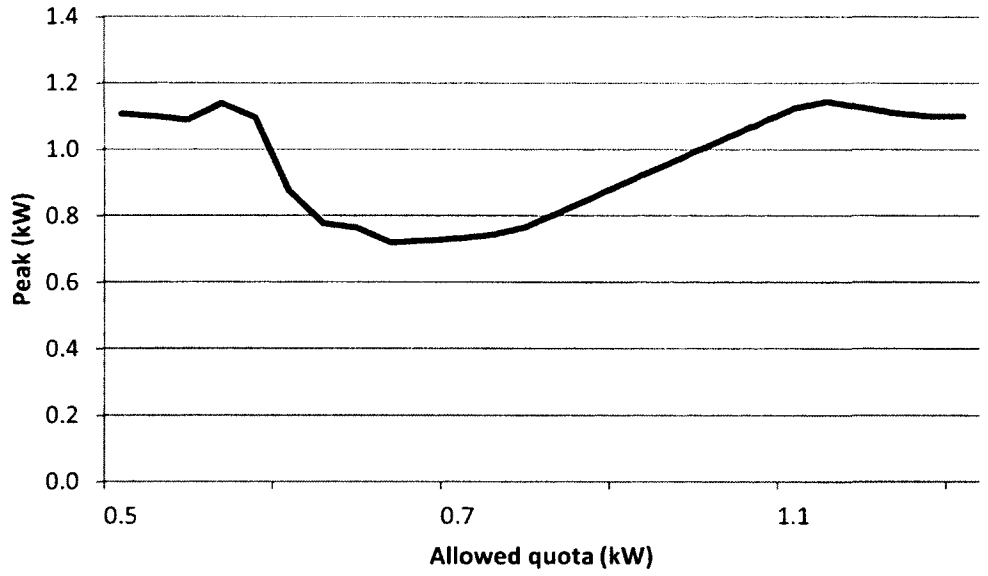


Figure 9: effects of the quota's value on the peak's value, x_{min} = 60° C, x_{max} = 65° C

2.5.2 Thermal Energy Storage

To store the electric energy in thermal form, x_{min} is set to 55°C and x_{max} to 70°C. The algorithm, allowed the lower temperature to reach 55°C instead of 60°C, and the higher temperature to rise to 70°C instead of 65°C.

The best possible theoretical result for this profile is reached (Figure 10). This corresponds to a peak reduction of 50% and a load factor of 0.99. The normalized energy consumed after control is around 14 kWh per day.

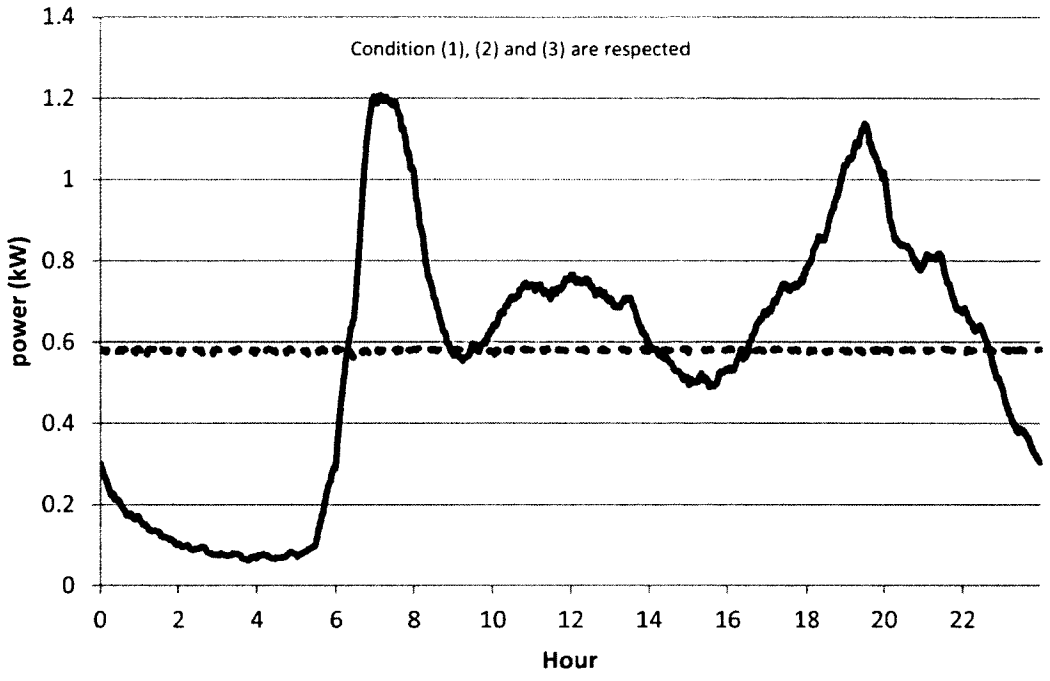


Figure 10: average power profile, normalized quota=0.6kW, $x_{min} = 55°$ C, $x_{max} = 70°$ C

Several simulations using different values of quotas were processed (Figure 11). The target optional quotas were respected only when their values were greater than the mean value (0.6kW) as stated in condition 3.

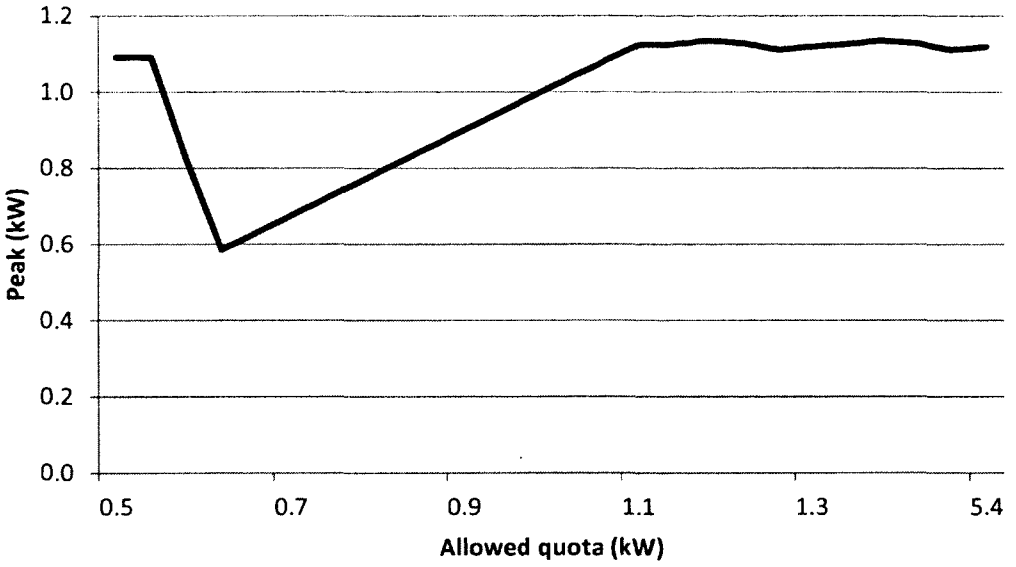


Figure 11: effects of the quota's value on the peak's value, $x_{min} = 55°$ C, $x_{max} = 70°$ C

2.5.3 Result analysis

The results showed that a constant quota is effective in flattening the load profile and in reducing the peak when its value is properly set. When the quota's value doesn't respect the stated conditions there was less benefits. If the optional quota value is too low, in order to respect user's comfort the algorithm ignores the quota allowing any water heater to start consumption as soon as its temperature drops to the minimum level (x_{min}). If the quota's value is too high, the peak is nearly the same and therefore there is no benefit.

The results obtained are specific to the example power profile. The peak reduction depends on the shape of the load profile, for example if the load profile is uniformly distributed over 24 hours, there is no need for any control.

In practice, maintaining hot water at higher temperature may require the use of a thermostatic mixing valve to ensure constant and safe outlet temperatures. Also, the durability of the tank depends on the water temperature and the impact of the water heater tank design should be considered.

When water temperature is higher, hot water consumption is reduced because the consumer will use a lesser quantity of hot water mixed with a greater quantity of cold water. A constant

amount of hot water extraction were used regardless the temperature of the water. Using a more realistic water heater simulations would improve the results. A better result means achieving the perfectly uniform load using less energy storage.

2.6 Conclusions

This paper presented a distributed load management technique for appliances that possess embedded communication interface for smart grid applications. An overlay P2P network, similar to those used in other information technology applications, is proposed for the communication between these appliances. With this technique, there is no centralized intelligence to control the community's energy consumption. The loads self control their consumption. When the load management is applied at the appliance level, it is possible to create, within physical limitations, any load consumption profile. This technique uses the flexibility inherent to energy storage to change the power profile of a group of devices.

A simulation of smart water heaters, using a fixed quota during 24 hours, showed that this distributed load management technique, combined with energy storage, allowed the appliances to create a steady load profile without affecting user comfort. This corresponds to the best possible theoretical result stated by condition 3, $\text{quota} = \frac{\sum_{h=1}^{T}\sum_{k=1}^{n} P_k(h)}{T}$.

Load levelling is one of the many applications that could be used to reduce the stress on power system. For example the same technique may be used to balance the high penetration of variable renewable energy sources. By using variable quota, regulation services could be offered by a community of load to balance the fluctuations of variable renewable energy. Varying the quota within reasonable range and the pre-determined conditions would reshape the load consumption profile to match the variability profile of renewable generators. Load forecast and load states (temperature, power demand) would be used to accurately estimate the up and down regulation potential and the right time-dependent quota to balance the fluctuation of the renewable energy.

CHAPITRE 3 ARBRE-ANNEAU « RING-TREE »

Le CHAPITRE 2 a montré comment une communauté de nœuds permet à des appareils électriques communicants de gérer leur consommation électrique d'une façon distribuée. Pour créer cette communauté, il a été proposé d'utiliser un réseau dédié « Overlay Network » de type « peer-to-peer » (P2P) capable de supporter un grand nombre de nœuds. Ce chapitre présente un nouveau réseau dédié appelé arbre-anneau « Ring-Tree » qui permet la création, l'exploitation et la maintenance d'un tel réseau. La motivation pour développer ce réseau dédié est liée aux limitations rencontrées lorsque l'on veut partager des données au sein d'une communauté avec un nombre important de nœuds, c'est-à-dire de quelques milliers à quelques millions. Des fonctionnalités comme la diffusion à tous les nœuds du réseau nécessitent la connaissance d'un moyen pour les rejoindre, comme par exemple la liste de leurs adresses, ce qui peut représenter un volume d'informations considérable si la taille de la communauté est importante. La diffusion de message d'un centre de contrôle est déjà utilisée par des programmes de gestion de la charge pour contrôler des équipements chez les clients d'une façon centralisée. Avec le P2P il serait possible à n'importe quel nœud de diffuser des messages à tous les autres nœuds du réseau, un service qui n'est pas disponible présentement dans le domaine de smart grid.

Un réseau dédié « overlay network » est un réseau logique bâti par-dessus une infrastructure de communication existante telle que l'Internet ou tout autre réseau IP. Les nœuds de ce réseau utilisent les services de la couche transport d'Internet donc principalement via les protocoles TCP ou UDP. Un réseau dédié P2P est composé et géré par les nœuds qui exploitent le réseau. Chaque nœud est à la fois source, destination et routeur.

Le réseau P2P permet aux nœuds qui le composent, appelés aussi « peers », d'échanger des données. Ces échanges peuvent se servir d'un serveur central ou ils peuvent être complètement décentralisés. Initialement, ces réseaux servaient pour des applications de partage des fichiers entre les nœuds [40]-[42]. Aujourd'hui, leur utilisation s'applique à plusieurs types d'applications : jeux vidéo virtuels « virtual reality » [43], communication multicast, diffusion audio et vidéo [44][45], QoS (Quality of Service), et bien d'autres applications et fonctionnalités.

L'utilisation des réseaux dédiés P2P présente plusieurs avantages pour les applications de smart grid. Il réduit les ressources nécessaires à la communication telles que l'infrastructure

physique. Il réduit également les coûts de maintenance du réseau et ceux d'opération des programmes de gestion de la charge, entre autres. Car la maintenance du réseau informatique est assurée par les nœuds eux-mêmes. Une fois que le réseau P2P est bâti par les nœuds, des solutions distribuées peuvent y être déployées. Ces solutions requièrent peu ou pas de gestion centralisée.

Actuellement, les solutions centralisées présentent un problème de vie privée bien connu dans le domaine du smart grid; une entité externe contrôle ou permet de contrôler les équipements à l'intérieur du domicile; des informations sur l'utilisation de l'appareil sont partagées avec un centre de contrôle. Une solution distribuée serait plus acceptée par les consommateurs, car avec le contrôle distribué c'est l'équipement lui-même qui gère sa consommation et non pas un centre de contrôle. Les informations sur l'utilisation de l'équipement ne seront pas partagées avec une entité centrale comme c'est le cas présentement. L'existence d'un centre de contrôle n'est pas nécessaire.

Les solutions centralisées sont limitées par rapport au nombre de nœuds qu'ils peuvent gérer, car l'ajout de nœuds augmente la charge au centre de contrôle ce qui augmente davantage le coût des investissements. Ceci limite l'extensibilité des solutions centralisées. Tandis que, l'ajout des nœuds dans un système P2P distribué augmente le gain potentiel pour l'application et réduit la probabilité de partitionnement du réseau.

Alors, le réseau P2P présente plusieurs avantages pour le smart grid. Mais, le choix d'un réseau P2P et de son architecture dépend grandement de l'application et de ses caractéristiques. Les caractéristiques des applications smart grid sont très différentes des autres applications utilisant le P2P. Par exemple, contrairement aux applications de partage de fichiers, les applications de smart grid n'échangent pas de fichiers de grande taille mais des messages de petites et moyennes tailles. Comme toute application P2P, les nœuds peuvent joindre et quitter le réseau dynamiquement mais les équipements électriques sont peu dynamiques, une fois que l'équipement est connecté, il ne quitte le réseau qu'en cas de bris ou de défectuosité. Les applications smat grid utilisant le P2P peuvent être caractérisées par les suivants:

- Participation d'un grand nombre d'équipements (de l'ordre de grandeur de 10^6)
- Interaction temps réel entre les équipements (de l'ordre de la minute pour l'ensemble du réseau

- Diffusion de message en continu à tous les équipements
- Messages échangés de petite et moyenne taille (de l'ordre de grandeur de 10^3 octets)
- Temps de diffusion de messages limité (<1 minute)
- Les nœuds rejoignent et quittent le réseau dynamiquement
- Une fois connectés les nœuds quittent rarement le réseau
- Les noeuds sont des systèmes embarqués et disposent de ressources limitées (communication, stockage et puissance de calcul)

Il n'existe pas une architecture P2P générique qui fonctionne pour toutes les applications. Les besoins spécifiques des applications conditionnent les services que le réseau P2P doit offrir et ces services affectent à leur tour la conception de ces réseaux. Les réseaux dédiés P2P peuvent être caractérisés par les éléments suivants [46]:

- Choix de la structure du réseau logique (topologie)
- Choix des identificateurs des nœuds
- Gestion des identificateurs par les nœuds
- Stratégie de routage
- Stratégie de maintenance

La conception du Ring-Tree tient compte des caractéristiques et des besoins des applications du smart grid. Le système Ring-Tree définit une nouvelle topologie. L'identificateur de chaque nœud permet la localisation de celui-ci dans la topologie. Outre le routage et la maintenance, l'identificateur sert aux différents types de communication Unicast, Multicast et Broadcast. Une table de voisinage de taille limitée a été conçue pour servir au routage et à la maintenance.

La section 3.1 présente une description générale du Ring-Tree. Ensuite, sont présentés la topologie (section 3.2), l'identificateur des nœuds (section 3.3), la table de voisinage (section 3.4), la table des IDs (section 3.5), le routage (section 3.6), l'algorithme de recouvrement pour la maintenance du réseau (section 3.7) et la méthode de la formation de la communauté (section 3.9). Enfin, la section 3.10 présente le rôle d'un serveur de support. Une analyse qui aide à la sélection des valeurs des paramètres n et p qui caractérisent le Ring-Tree est montrée à la section 3.11. La section 3.12 conclue le CHAPITRE 3.

3.1 Description générale

Les réseaux P2P diffèrent selon les services qu'ils offrent aux applications. La majorité des réseaux P2P offre le service de partage des fichiers. D'autres réseaux sont optimisés pour la communication locale entre les nœuds à proximité, une fonctionnalité nécessaire pour les jeux de réalité virtuelle. Le Ring-Tree offre principalement la communication « Broadcast » entre tous les nœuds ainsi que la mise à jour et le partage d'une variable distribuée (histogramme) dans tous les nœuds.

Comme tout réseau dédié P2P, le Ring-Tree permet l'ajout dynamique des nœuds dans le réseau. Grâce à sa structure hiérarchique qui se compare à un arbre, le temps de propagation de l'information entre un nombre important de nœuds, est limité. Il est principalement proportionnel au nombre de niveaux dans cet arbre. Grâce à l'ajout des liens pour former les anneaux, la topologie a une meilleure tolérance à la disparition non planifiée des nœuds. Un algorithme de recouvrement permet l'autoréparation « self healing » de la structure. Un identificateur de nœuds (ID) comparable à une adresse IP sert au routage au niveau de la couche Ring-Tree. Cet identificateur permet le repérage de chaque nœud dans le réseau. Une table de voisinage de taille limitée sert au routage au niveau Ring-Tree et à la maintenance, d'une façon distribuée, de la structure. Le système est adapté pour des équipements possédant des capacités restreintes. Les ressources requises par chaque nœud sont limitées, en particulier le nombre de connexions maintenues avec les nœuds voisins dans la structure est limitée. La structure du « Ring-Tree » répond donc aux spécifications suivantes :

- Communication entre un nombre important de nœuds (ordre de grandeur du million)
- Ordonnancement intrinsèque des nœuds (ID) au niveau de la structure logique, permettant le repérage de chaque nœud dans le réseau
- Ressources requises (taille de la table de voisinage, ...) pour chaque nœud limitées
- Temps de propagation de l'information (Broadcast, ...) limité
- Ajout dynamique des nœuds dans le réseau
- Autoréparation de la structure en cas de disparition d'un ou de plusieurs nœuds

3.1.1 Présentation en couches

Par rapport au modèle standardisé OSI (Open System Interconnection), le Ring-Tree se situe entre la couche Application et la couche Transport (Figure 12). Il offre à la couche Application

des services de communication au niveau du réseau dédié tels que : Broadcast, Multicast, Unicast et mise à jour et diffusion d'un histogramme. Il s'occupe également de la connexion des nœuds et de la maintenance du réseau sans l'intervention de la couche Application.

Le système exploitant le Ring-Tree, par exemple l'algorithme de gestion de la charge décrit au CHAPITRE 2, réside dans la couche Application. L'application utilise des fonctionnalités du Ring-Tree tel que Join() pour rejoindre une communauté de charge et Broadcast() pour diffuser un message à tous les membres d'une communauté. Une même application peut créer plusieurs nœuds dans une même communauté ou dans des communautés différentes. Il sera donc possible d'échanger des messages entre ces nœuds à partir d'une même application.

Les fonctionnalités internes du Ring-Tree telles que les protocoles de communication, le routage, la connexion et la maintenance sont invisibles pour la couche application. Certaines fonctionnalités servent, uniquement, l'application et d'autres servent la communauté Ring-Tree tel que le routage et la maintenance du réseau. Le Tableau II présente une liste de fonctions primitives disponibles pour l'application.

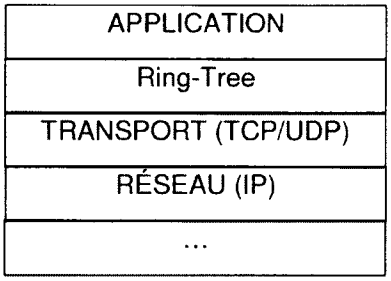


Figure 12: diagramme de l'architecture relativement au modèle OSI

Tableau II: fonctions primitives du Ring-Tree

Nom de la fonction primitive	Description
Join(CID)	Rejoindre une communauté qui a l'identificateur CID
Leave(CID)	Se déconnecter de la communauté CID
Unicast(CID, ID, MSG)	Envoyer un message MSG au nœud qui a l'identificateur ID représentant sa position dans la communauté CID
Multicast(CID, ID, MSG)	Envoyer un message MSG au nœud qui a l'identificateur ID ainsi qu'à tous ses descendants dans la communauté CID
Broadcast(CID, MSG)	Envoyer un message MSG à tous les nœuds de la

	communauté CID
Receive(CID)	Recevoir les messages destinés à l'application et provenant de la communauté CID
NodesStat(CID)	Obtenir de l'information, déterminée selon le niveau de sécurité et de confidentialité requis, sur le réseau
Update (CID, HIST, VAL)	Mettre à jour l'histogramme partagé en y insérant la valeur VAL

3.1.2 Ring-Tree Layer et Management

Le Ring-Tree est composé de deux parties principales : Ring-Tree Layer (RTL) et Management (Figure 13). La partie Management offre à l'application, les services de connexion et de déconnexion à l'aide des primitives Join() et Leave(). Elle s'occupe de la gestion, la mise à jour et la diffusion de la table de voisinage qui change à chaque ajout ou disparition d'un nœud voisin. Management gère la taille dynamique de la communauté via le traitement des requêtes d'ajout de nouveaux nœuds et la détection et le remplacement des nœuds voisins disparus. À l'aide de l'algorithme de recouvrement, Management maintient le nœud et ses descendants attachés au réseau. À des intervalles réguliers, des messages keepalive peuvent être envoyés aux nœuds voisins pour s'assurer que la communication est toujours en bon état. Sinon une connexion TCP peut être utilisée. Après la détection d'un bris de communication, Management exécute l'algorithme de recouvrement, met à jour la table de voisinage et diffuse cette table aux nœuds concernés. Pendant la mise à jour de la table, Management met un fanion d'état à 0 pour empêcher l'utilisation de la table par d'autres parties du Ring-Tree. Le fanion d'état indique si le nœud est connecté correctement et si la table est à jour et prête à être utilisé. RTL, ne peut utiliser la table pour, par exemple, le routage, l'envoi unicast, Multicast ou Broadcast, etc. tant que le fanion n'est pas remis à 1.

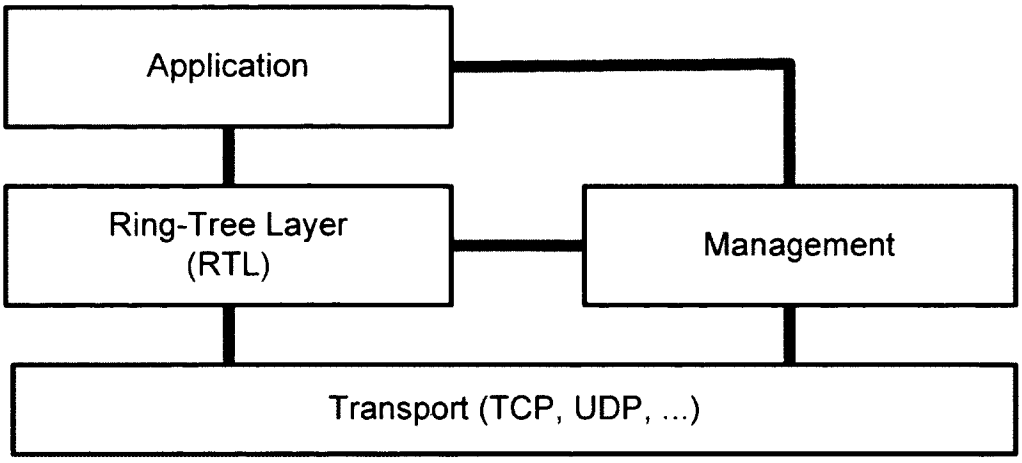


Figure 13: diagramme en couches réseau de l'architecture

La partie RTL interagit avec la couche application plus souvent que Management. Elle achemine les messages Unicast, Multicast et Broadcast ainsi que le service de mise à jour et de lecture de l'histogramme partagé pour l'application. Elle exécute, au besoin, le protocole interne ARM (Address Resolution Management) qui sert à trouver l'adresse IP d'un nœud dont on connait l'ID.

RTL s'occupe de la réception des messages provenant de la couche Transport. Certains messages sont livrés à l'application, d'autres sont traités par RTL ou Management selon le type de message reçu. Les messages de type multicast et Broadcast sont relayés vers les nœuds concernés de la table de voisinage. Si un message keepalive est reçu, la RTL répond au voisin pour lui confirmer que le nœud est toujours actif. Les autres messages de gestion tels que la mise à jour de la table de voisinage provenant d'un voisin, la demande d'insertion provenant d'un nouveau nœud, la demande de réinsertion d'un nœud déconnecté, etc. sont envoyés à la partie Management. La Tableau III présente une liste de fonctions internes à la couche Ring-Tree.

Tableau III: fonctions interne à la couche Ring-Tree

Nom de la fonction	Description
Recovery()	implémente l'algorithme de recouvrement pour gérer la disparition des nœuds voisins
Insert()	insère un nouveau nœud au réseau. Réinsère un sous réseau après la disparition d'un nœud parent

Keepalive()	surveille le lien logique avec les nœuds voisins en envoyant périodiquement des messages keepalive de petite taille (<100 octets), (service optionnel)
TableUpdate()	permet de mettre à jour la table de routage
ARM(ID)	retourne l'adresse IP d'un nœud possédant l'identificateur ID (service optionnel)

3.2 Topologie

La topologie du réseau dédié est choisie selon les protocoles de communication et les applications. Certaines applications visent des topologies qui possèdent plus de redondance en tenant compte du réseau physique [47]. D'autres applications telles que le partage de fichiers et la diffusion de contenu multimédia visent des topologies qui tiennent compte de la capacité de la source.

La topologie du réseau dédié a un impact important sur la performance du routage [48]. Pour avoir un réseau extensible et mieux gérable, il faut utiliser une topologie hiérarchique structurée [49]. Plusieurs études ont été effectuées sur les réseaux hiérarchiques [50]-[53]. Les topologies hiérarchiques peuvent l'être à multi-niveaux comme la topologie tore [54] ou à deux niveaux (Figure 14) [55].

Dans la catégorie des architectures structurées on retrouve, entre autres, des topologies en arbre [56][57] et en anneau [58]. Avec les topologies structurées les algorithmes de routage sont simples. Certains auteurs proposent des topologies non structurées (maille) [43], des topologies hybrides (maille et arbre) [59] ou des systèmes qui permettent de créer des topologies différentes, à partir de zéro, à l'aide de protocoles configurables [54].

Le Ring-Tree se situe dans la catégorie des réseaux structurés hiérarchiques multi-niveaux. Des concepts existants dans d'autres réseaux sont exploités dans le Ring-Tree mais sa topologie est complètement nouvelle. Les algorithmes de création et de maintenance et les protocoles de communication définis dans le réseau Ring-Tree exploitent les avantages de cette nouvelle topologie.

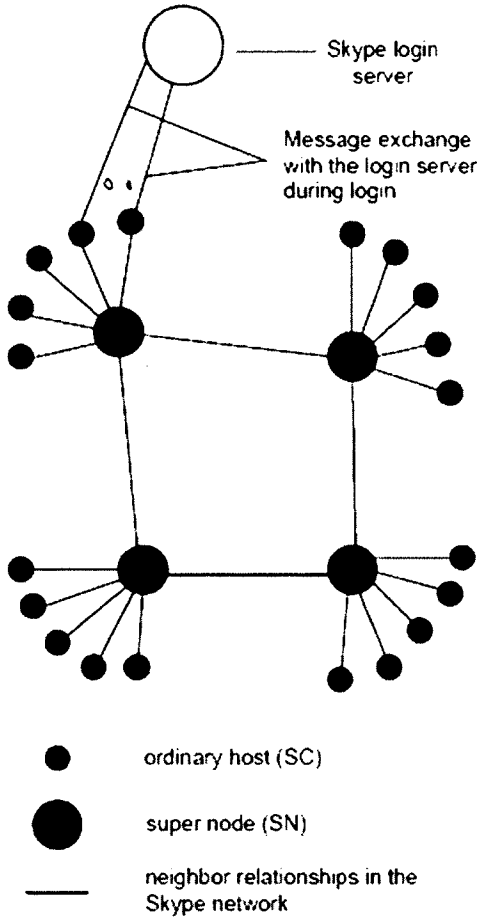


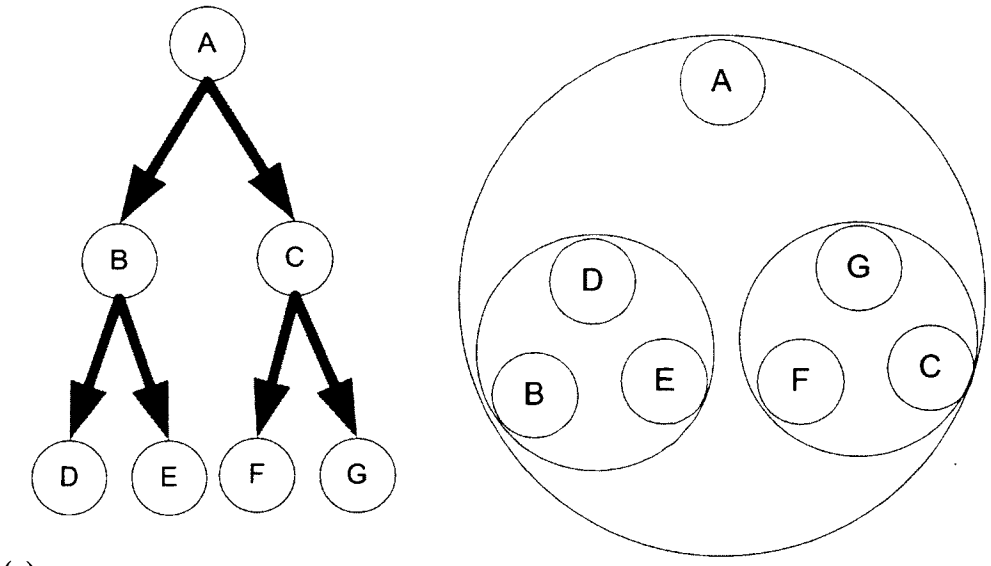
Figure 14: représentation d'un exemple d'une topologie à deux niveaux (Skype)

3.2.1 Topologie Ring-Tree

La topologie du Ring-Tree peut être vue comme une topologie en arbre, mais les nœuds de chaque niveau forment un anneau entre eux et avec leur parent. Un nœud peut appartenir à plusieurs anneaux. Pour chaque anneau, il établit deux liens : un lien avec le voisin situé à gauche et un autre avec celui situé à droite. La Figure 15 montre un réseau de sept nœuds organisés selon deux topologies : arbre et arbre-anneau.

Si le nœud A quitte le réseau involontairement, la topologie en arbre se partitionne tandis que celle en arbre-anneau, grâce au lien entre les nœuds B et C, ne se partitionne pas et il est possible de recréer un autre lien virtuel pour refermer l'anneau. De plus, si, dans la topologie en arbre, le nœud B quitte le réseau involontairement, les nœuds D et E se retrouvent séparés l'un de l'autre et séparés du réseau. Les deux nœuds D et E doivent se reconnecter au réseau. Dans la topologie arbre-anneau seulement un des deux nœuds soit D ou E doit se reconnecter

au réseau. La topologie en arbre-anneau présente une meilleure tolérance à la disparition involontaire des nœuds par rapport à la topologie en arbre.



(a)

(b)

Figure 15: (a) topologie en arbre (b) topologie en arbre-anneau

La topologie en arbre est bien adaptée aux communications de type « one-to-many » où un nœud source diffuse des informations à plusieurs nœuds. Elle est extensible car elle supporte un grand nombre de nœuds qui peuvent se joindre au réseau. Elle réduit le temps de propagation des messages grâce au parallélisme dans la communication. Mais elle est sensible à la disparition du nœud central (« centre point failure »). Dans les applications qui requièrent une communication de type many-to-many, comme l'application visée par le présent projet, une topologie en arbre, sensible à la disparition involontaire d'un nœud, nuit grandement à la performance des applications. Elle n'est pas optimisée pour les applications de type « many-to-many » ou pour la mise à jour distribuée d'un histogramme.

Une topologie en anneau augmente la robustesse de la structure face à une disparition involontaire d'un nœud. Mais, le temps de propagation de l'information augmente avec l'augmentation du nombre de nœuds dans l'anneau. Elle est donc moins extensible qu'une topologie en arbre.

La topologie en arbre et en anneau existent déjà. Mais une topologie qui combine les deux serait une innovation. Cette combinaison permet au Ring-Tree d'avoir les avantages de la topologie en anneau qui augmente la robustesse de la structure, réduit le problème de « centre

point failure » et et également ceux de la topologie en arbre qui accroît l'extensibilité du réseau et réduit le temps de propagation.

3.2.2 Nœud enfant et parent

Il existe deux types de nœuds : nœud enfant et nœud parent appelé aussi grappin. Ce concept est similaire à celui que l'on retrouve dans Skype ou Kazaa [42], où il y a des nœuds ordinaires et des super nœuds. Un nœud grappin appartient à plusieurs anneaux. Par exemple dans la Figure 15 les nœuds A, B et C sont des nœuds grappins et les autres sont des nœuds enfants. Le nœud grappin est responsable de faire le lien entre les nœuds de l'anneau incluant leurs descendants et le reste du réseau. Il doit effectuer la fusion (agrégation) des informations provenant des autres nœuds et acheminer le trafic entre les anneaux où il est relié. N'importe quel nœud peut être un nœud Grappin dépendamment de sa position dans le réseau.

Un parent est responsable de l'insertion et de la réinsertion des nœuds. Après une disparition d'un enfant, avant d'accepter une demande d'insertion, un parent attend un temps prédéfini pour donner plus de chance aux descendants du nœud disparu de se réinsérer dans cet anneau. Une demande de réinsertion concerne un ensemble de nœuds tandis qu'une demande d'insertion concerne un seul nouveau nœud. Un parent priorise donc une demande de réinsertion sur les demandes d'insertion.

Chaque nœud est un parent sauf les nœuds de niveau feuille. Un nœud peut être parent dans plusieurs anneaux. Mais, il ne peut être enfant que dans un seul anneau. Chaque nœud est un enfant sauf le nœud parent du niveau tronc qui est le seul, dans tous le réseau, qui n'est pas un enfant.

Le niveau de l'anneau où le nœud est situé comme enfant définit le niveau d'un nœud sauf le nœud parent du niveau tronc qui lui a le niveau 0.

3.2.3 Paramètres n et p

La topologie Ring-Tree nommée T(n, p) est caractérisée par deux paramètres, n et p. Le paramètre n est le nombre de nœuds maximal que chaque anneau doit supporter. Le paramètre p est le nombre de niveaux maximal que le réseau peut avoir. Le réseau peut donc contenir un maximum de $N = n^p$ nœuds.

Le Ring-Tree a donc une topologie hiérarchique à p niveaux. Le niveau le plus bas (niveau du tronc) a le numéro « 0 », la numérotation des autres niveaux augmente en ordre croissant jusqu'au niveau le plus haut (niveau feuille) qui a le numéro « p-1 ».

La topologie Ring-Tree est structurée en anneaux. Chaque anneau peut avoir un maximum de n nœuds. Les nœuds sont numérotés en ordre croissant dans le sens horaire (choix arbitraire) de « 0 » à « n-1 ». Le nœud parent a donc le numéro « 0 ».

Théoriquement n et p peuvent avoir n'importe quelle valeur. Une étude sur les valeurs optimales possibles pour n et p est présentée à la section 3.11.

3.2.4 Exemple de topologie

La topologie Ring-Tree T(3, 3) présentée à la Figure 16 contient 9 anneaux de niveau 2 qui est le niveau feuille, 3 anneaux de niveau un et un anneau de niveau zéro qui est le niveau tronc. Ce réseau peut contenir un maximum de 27 nœuds (n=3, p=3 et $N=n^p=27$ nœuds). Les nœuds sont numérotés de 0 à 2 à l'intérieur de chaque anneau.

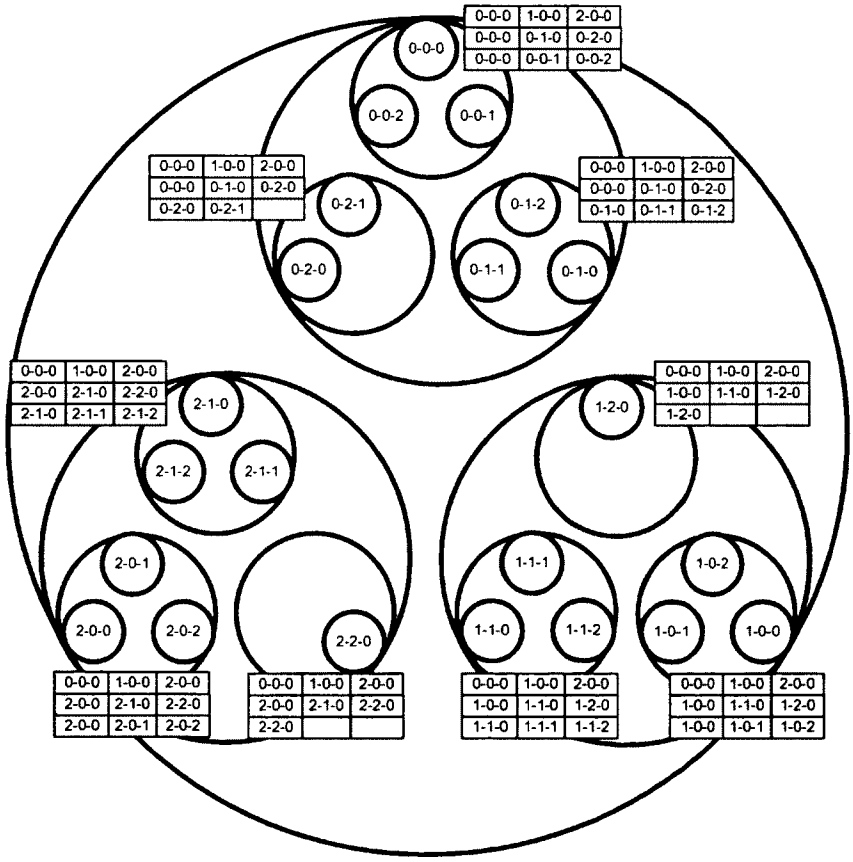


Figure 16: exemple de topologie (les tables des nœuds sont décrites à la section 3.4)

Un nœud enfant hérite une partie du numéro de son parent. Par exemple le nœud 1-1-2 hérite la partie 1-1 de son parent 1-1-0 et son parent hérite le premier 1 de son parent 1-0-0. Ce numéro s'appelle identificateur ID.

3.3 Identificateur

Chaque nœud est identifié par un numéro d'identification unique nommé ID. Cet ID correspond à sa position logique dans le réseau. Cet ID est important car, selon sa position, un nœud peut avoir un rôle différent dans la communauté. Par exemple, un nœud placé à la position 0 dans son anneau est un nœud parent. Un nœud parent a l'autorité d'insérer et de réinsérer des nœuds dans l'anneau. Un nœud surveille toujours son suivant, si un bris de communication avec le suivant est détecté et si le suivant est un parent, le nœud devient orphelin responsable de trouver une autre position permettant de réinsérer les nœuds de son anneau et ses descendants dans le réseau. À partir de son ID, un nœud peut connaître son rôle dans la communauté.

Par convention, un ID est composé de p champs séparés par des tirets « - ». La position de chaque champ correspond à un niveau du réseau. La position de chaque champ représente, de gauche à droite, les niveaux de 0 à p-1. La valeur de chaque champ varie entre 0 et n-1 avec n le nombre maximal de nœuds dans un anneau. Cette valeur représente une position dans l'anneau. La valeur zéro représente un nœud parent et les autres nombres augmentent dans le sens horaire (choix arbitraire) représentants des nœuds enfants. Chaque nœud possède donc un numéro de position dans son anneau et son ID représente sa position dans le réseau.

3.3.1 Construction de l'ID

Lorsqu'un nœud ξ rejoint le réseau, son parent immédiat lui attribue un ID partiel contenant de l'information sur le niveau de l'anneau où il est placé comme enfant, la position exacte dans cet anneau ainsi que la position de ses parents. Cet ID contient i+1 champs. La position du dernier champ (position i) représente le niveau du nœud. La valeur de ce champ représente la position où ce nœud est situé dans l'anneau. La position d'avant dernier champ (i-1) représente le niveau de son parent immédiat et sa valeur représente la position du parent dans son anneau. Les positions des champs suivants (i-2, i-3, ...) représentent les niveaux des autres

parents de niveaux inférieurs et leurs valeurs représentent les positions des parents dans leurs anneaux respectifs et ainsi de suite jusqu'au parent du niveau tronc représenté par le premier champ de l'ID (position 0). À l'aide de ces informations, le nouveau nœud sait exactement où il est situé dans l'ensemble du réseau.

La taille de l'ID i+1 est inférieure ou égale à p-1, p étant le nombre maximal de niveaux. Comme la taille de l'ID doit être composée de p champs, le nœud ajoute zéros pour les p-i champs manquants. Les zéros indiquent que le nœud ξ est maintenant parent pour tous les anneaux de niveaux supérieurs à son niveau i et auxquels il est relié.

Par exemple, si dans une topologie $T(n = 3, p = 5)$ un nouveau nœud reçoit l'ID [0-2-1]. Comme la taille de l'ID reçu est plus petite que p, le nœud peut déduire qu'il n'est pas au niveau feuille. Dans ce cas le nœud est de niveau 2 (5-3=2). Il complète son ID en ajoutant deux zéros car la taille p=5. Cet ID devient [0-2-1-0-0]. Le nœud déduit qu'il est parent (sans enfants pour l'instant) aux niveaux 3 et 4. Il occupe la position 1 dans son anneau de niveau 2 car le champ à droite contient la valeur 1. Son parent de niveau 1 occupe la position 2 dans son anneau et son parent de niveau 0 occupe la position 0 dans son anneau.

À partir d'un ID, il est possible de déduire le niveau du nœud. Il est également possible de déduire sa position dans le réseau. Si on examine de droite à gauche les valeurs d'un ID d'un nœud quelconque, tant qu'on trouve des zéros, cela veut dire que ce nœud est un parent à tous ces niveaux de p-1 jusqu'à ce qu'on rencontre une valeur différente de zéro. La position de cette valeur différente de zéro représente le niveau du nœud et sa valeur représente la position du nœud dans son anneau. Les valeurs suivantes à gauche représentent les positions de son parent, du parent de son parent, etc. jusqu'au tronc.

Par exemple, dans une topologie $T(n = 3, p = 3)$ l'ID [2-1-1] veut dire que le nœud [2-1-1] est un nœud feuille. Il est placé à la deuxième position dans son anneau de niveau 2, son parent est placé à la deuxième position dans son anneau de niveau 1 et le parent de son parent est placé à la troisième position dans son anneau qui est de niveau 0.

3.4 Table de voisinage

Une quantité d'information limitée est conservée par chaque nœud dans une structure de donnée appelée table de voisinage ou table de connaissance. Chaque case de la table contient de l'information (adresse IP, ...) sur un seul nœud. La taille de cette table est $n \times p$, avec n le

nombre maximal possible de nœuds dans un anneau et p le nombre maximal possible de niveaux dans le réseau. Cette table est utilisée principalement pour le routage et par l'algorithme de recouvrement.

Chaque ligne de la table correspond à un niveau hiérarchique du réseau. La ligne supérieure correspond au niveau « 0 » qui est le niveau du « tronc », les lignes suivantes correspondent aux autres niveaux en ordre croissant jusqu'à la dernière ligne « p-1 » le niveau « feuille ».

Les cases d'une même ligne contiennent de l'information sur les nœuds d'un même anneau. Chaque colonne correspond à la position du nœud dans l'anneau. La première colonne (position « 0 ») contient de l'information sur le nœud parent. Les colonnes suivantes contiennent de l'information sur les nœuds descendants numérotés en ordre croissant. Le nœud situé à la première case est le nœud parent de tous les autres nœuds situés dans les autres case de la même ligne.

3.4.1 Construction de la table de voisinage

Lorsqu'un nœud se joint au réseau, son parent lui attribue un ID. À partir de cet ID, le nœud peut déterminer où se placer dans la table. Il y a une correspondance entre l'ID du nœud et sa position dans la table. La position du dernier champ de l'ID (avant l'ajout des zéros à la fin) et la valeur de ce champ correspondent respectivement aux numéros de ligne et de colonne de la case correspondant à sa position dans la table (Figure 17). La valeur de ce champ n'est jamais zéro. Ceci indique qu'un nœud se joint toujours au réseau comme étant enfant.

Son parent immédiat lui envoie la partie supérieure de sa table de voisinage. Il lui envoie donc le contenu de toutes les lignes de 0 à i, i étant le niveau du nœud. Les nœuds d'un même anneau de niveau i ont en commun les i+1 premières lignes du tableau.

La partie inférieure du tableau doit contenir tous les descendants directs du nœud ξij. Ces descendants doivent se placer dans les colonnes autres que celle à la position 0 car cette position correspond au parent qui est dans ce cas le nœud ξij.

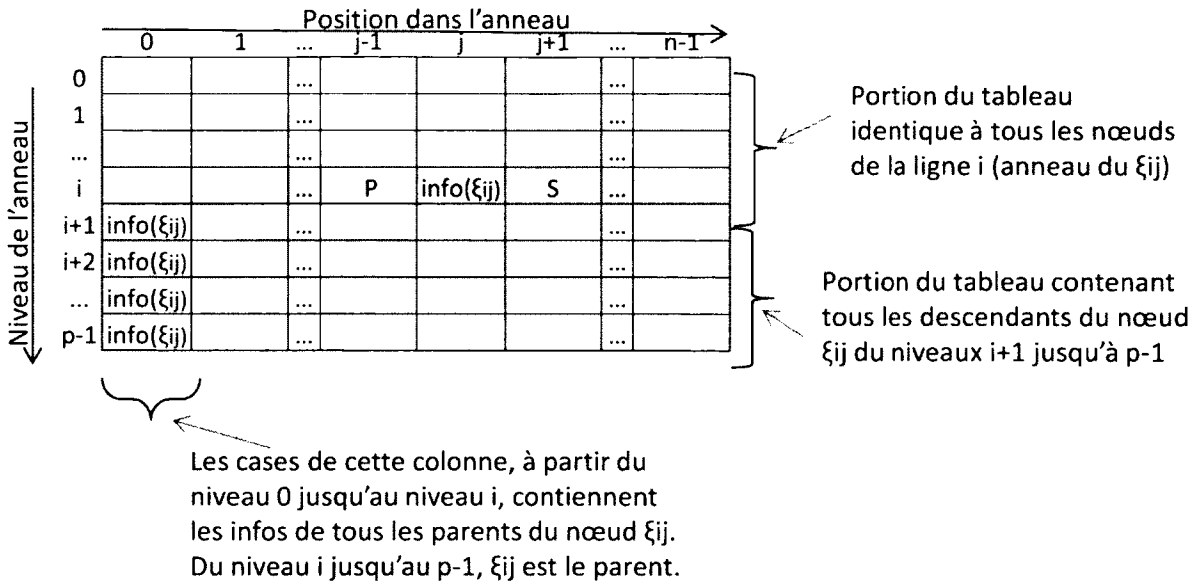


Figure 17: représentation de la construction de la table de voisinage

3.4.2 Niveau d'un nœud

Le niveau d'un nœud est le niveau de l'anneau à lequel le nœud appartient comme étant enfant. Chaque nœud est un enfant dans exactement un seul anneau à l'exception du premier nœud du réseau qui n'est pas un enfant. Le premier nœud du réseau est le nœud parent du niveau tronc, il a donc le niveau zéro.

Un nœud peut appartenir à plusieurs anneaux de niveaux différents dans le réseau. Mais, le nœud ne peut être enfant que dans un seul de ces anneaux. Le niveau de ce seul anneau à lequel le nœud est placé comme étant enfant définit donc le niveau du nœud.

Le niveau du nœud peut être déterminé à partir de la taille de l'ID reçu au moment de la connexion au réseau. La taille de cet ID est plus petite ou égale à p. Si la taille de l'ID reçu est x, le niveau du nœud est x-1. Le premier nœud du réseau est une exception. Il a le niveau 0.

À l'exception du nœud ξ_{00} qui est le premier nœud du réseau, la ligne i de la table de voisinage ou de la table des IDs est la seule ligne où un nœud ξij apparaît dans une colonne autre que la première colonne quelque soit ξij. Le nœud ξij est un enfant dans cet anneau. La ligne i détermine donc le niveau du nœud ξij. Ce niveau est identique à tous les autres nœuds enfants situés à la même ligne dans les colonnes supérieurs à 0.

3.4.3 Connaissance mutuelle et non mutuelle

On dit qu'un nœud A « connaît » un nœud B si A détient de l'information sur B. Cette information inclut principalement une façon permettant de le rejoindre ou de lui envoyer un message par exemple son adresse IP. Il existe deux types de connaissance : connaissance mutuelle et connaissance non mutuelle. Une « connaissance mutuelle » veut dire que les deux nœuds se connaissent i.e. tous les deux détiennent de l'information l'un sur l'autre. Une « connaissance non mutuelle » veut dire que le nœud A connaît B mais B ne connaît pas A.

Chaque nœud a une connaissance mutuelle avec chaque nœud appartenant au même anneau que lui ainsi qu'avec chacun de ses descendants directs (Figure 18). Chaque nœud a une connaissance non mutuelle de tous les nœuds de l'anneau de son parent, de tous les nœuds de l'anneau de parent de son parent et ainsi de suite jusqu'à l'anneau du tronc. La Figure 19 illustre cette chaîne de connaissance non mutuelle, représenté par la courbe foncée, pour le nœud qui a l'ID 211.

La chaîne de connaissance non mutuelle est utilisée principalement par l'algorithme de réinsertion. Un nœud exécute cet algorithme lorsqu'il surveille son parent et son parent disparaît (section 3.7).

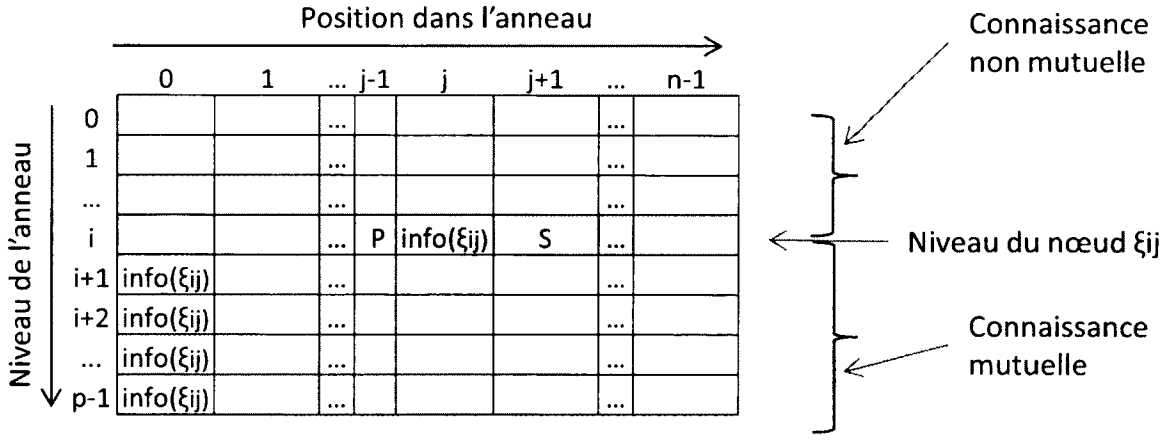


Figure 18: connaissance mutuelle et non mutuelle dans la table de voisinage

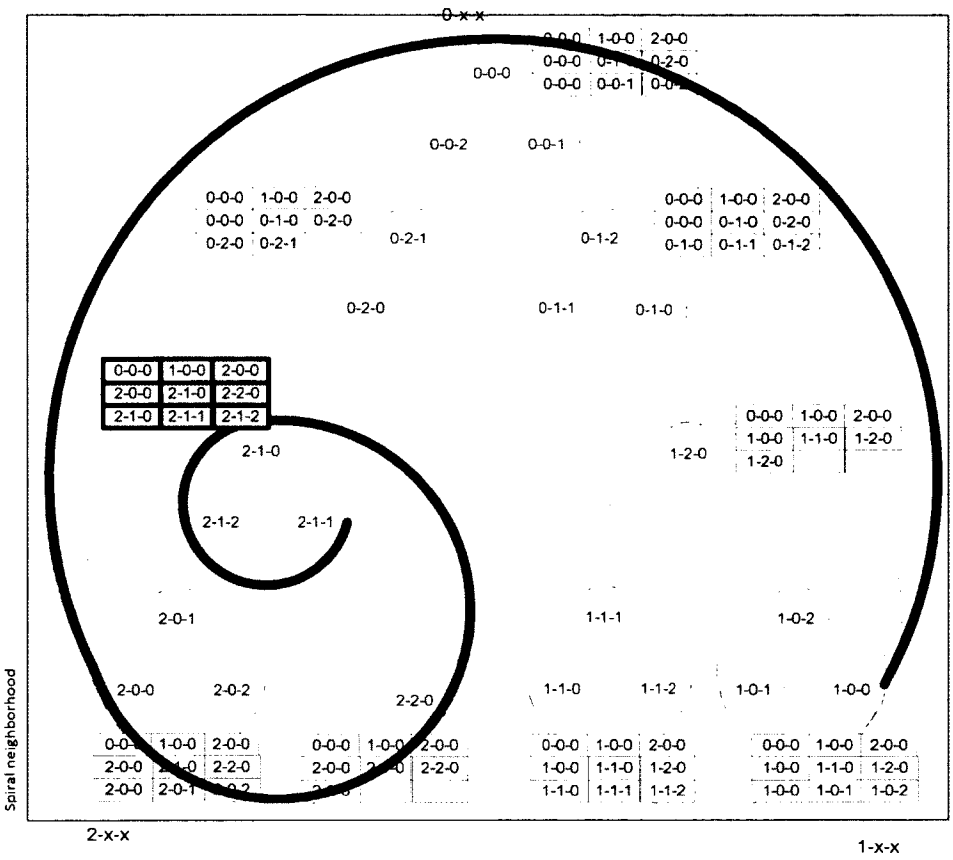


Figure 19: schéma de la spirale de la table de voisinage

3.4.4 Nœud précédent et nœud suivant

Un nœud utilise sa table de voisinage pour localiser ses précédents et ses suivants sur les différents anneaux (Figure 20). La localisation des nœuds suivants et précédents dans la table de voisinage est nécessaire pour l'algorithme de surveillance. Elle permet au nœud ξ d'identifier les nœuds qu'il doit surveiller (les nœuds suivants). Elle permet également au nœud ξ d'identifier les nœuds qui le surveillent (les nœuds précédents).

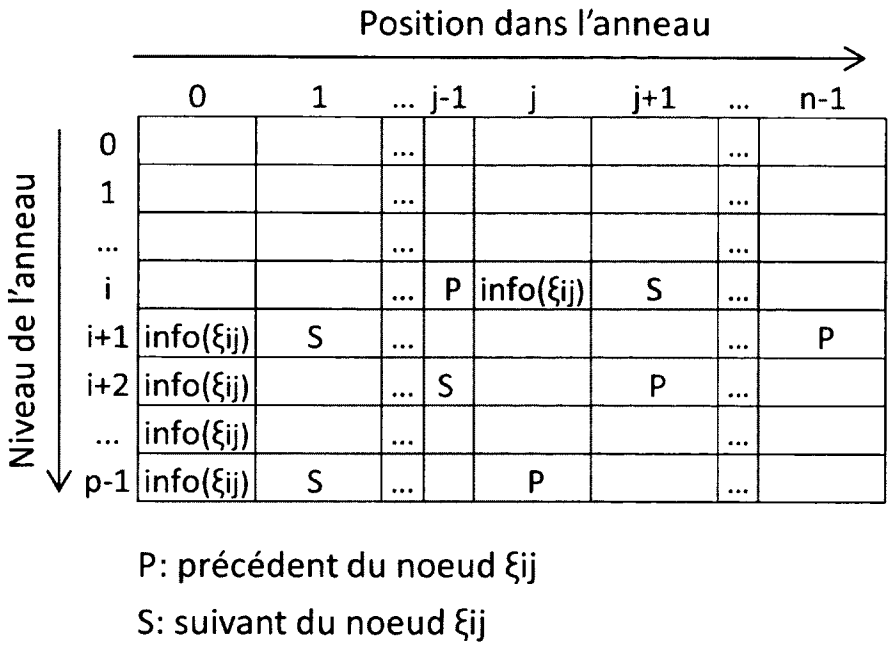


Figure 20: représentation du nœud précédent et du nœud suivant dans la table de voisinage

Le précédent d'un nœud ξ situé à la position j dans un anneau (ligne) de niveau i est le premier nœud qui se trouve à gauche du nœud ξ i.e. à la position k la plus proche avec $k < j$. Si $j = 0$, le précédent serait le dernier nœud qui existe dans l'anneau i.e. le nœud situé à la fin de la ligne i. Chaque ligne est vue comme un cercle car elle correspond à un anneau fermé. Le suivant du nœud ξ est le premier nœud qui se trouve à sa droite i.e. à la position u la plus proche avec $u > j$. Si ξ est le dernier nœud de l'anneau i.e. situé à la fin de la ligne i, le suivant serait le premier nœud i.e. le nœud situé à la position 0.

Si le nœud ξ est seul dans un anneau, il n'aurait pas ni précédent ni suivant dans cet anneau. S'il existe un autre nœud que ξ dans un anneau, ce nœud serait à la fois suivant et précédent. S'il existe plusieurs nœuds dans un anneau, le nœud suivant du nœud ξ serait le premier nœud existant et situé à la position supérieure à celle de ξ et le nœud précédent serait celui situé à la position inférieure, sauf si le nœud ξ est situé à la position 0, dans ce cas, le nœud précédent serait le nœud situé à la position la plus élevée et si le nœud ξ est situé à la dernière position (position la plus élevée) le nœud suivant serait celui situé à la première position (position 0).

Soit j la position du nœud ξ et soit k la position du nœud précédent, k appartient à l'ensemble des ID des nœuds existant dans l'anneau, k est alors déterminé comme suit:

Si j>0 min(j-k) avec k<j

Si j= 0 max(k), avec k<n

Soit j la position du nœud ξ et u la position du nœud suivant, u appartient à l'ensemble des ID des nœuds existant dans l'anneau, u est donc déterminé comme suit:

Si j<max(u) min(u-j) avec u>j

Si j=max(u) u=0

Dans chaque anneau un nœud peut avoir un maximum d'un précédent et d'un suivant. Comme le nombre total d'anneaux auxquels un nœud peut appartenir est de p-i anneaux (i étant le niveau du nœud), chaque nœud peut avoir un total de p-i suivants et p-i précédents.

3.5 Table des IDs

Pour attribuer une position à un nouveau nœud, le nœud parent responsable de l'insertion du nouveau nœud utilise une table des IDs appelé aussi table de conversion. Cette table contient les IDs de tous les nœuds qui se trouvent ou qui peuvent se trouver dans sa table de voisinage. La taille de cette table est donc identique à celle de la table de voisinage $n \times p$. Connaissant les paramètres n et p, un nœud crée la table des IDs, à partir de son propre ID. La création de cette table se fait localement par n'importe quel nœud, il n'est donc pas nécessaire d'échanger cette table.

Supposons qu'un nœud ξij possède l'ID : $ID_0ID_1 \ldots ID_i$, avec i le niveau du nœud, ID_k champ de l'ID et $k = 0..p-1$. Seulement la partie gauche de chaque ID est représentée. Comme chaque ID est de taille p, il faut ajouter des zéros pour le compléter.

La première ligne de la table des IDs est toujours la même. Elle contient les IDs des nœuds de niveau tronc numéroté de 0 à n-1. Le premier champ ID_0 de l'ID peut avoir n'importe quelle valeur provenant de cette première ligne.

Dans les cases de la deuxième ligne, le premier champ de l'ID ne change pas (toujours ID_0), le deuxième champ augmente en ordre croissant en commençant par zéro à partir de la première case de cette ligne. La première case contient donc ID_00, la deuxième case contient ID_01 et ainsi de suite jusqu'à la dernière case qui contient $ID_0(n-1)$.

Dans les cases de la troisième ligne, les deux premiers champs de l'ID ne changent pas et le troisième champ augmente en ordre croissant en commençant par zéro à partir de la première case de cette ligne. Dans les cases de la quatrième ligne, les trois premiers champs de l'ID ne

changent pas et le quatrième champ augmente en ordre croissant de la même façon que les autres lignes avant et ainsi de suite jusqu'au remplissage de la table (Figure 21).

Comme i est le niveau du nœud ξij, les lignes à partir de i+1 jusqu'à p-1 contient les enfants de ce nœud. Toutes les cases à partir de la position $(i+1),1$ jusqu'à $(p-1),(n-1)$ correspondent aux IDs des enfants du nœud ξij.

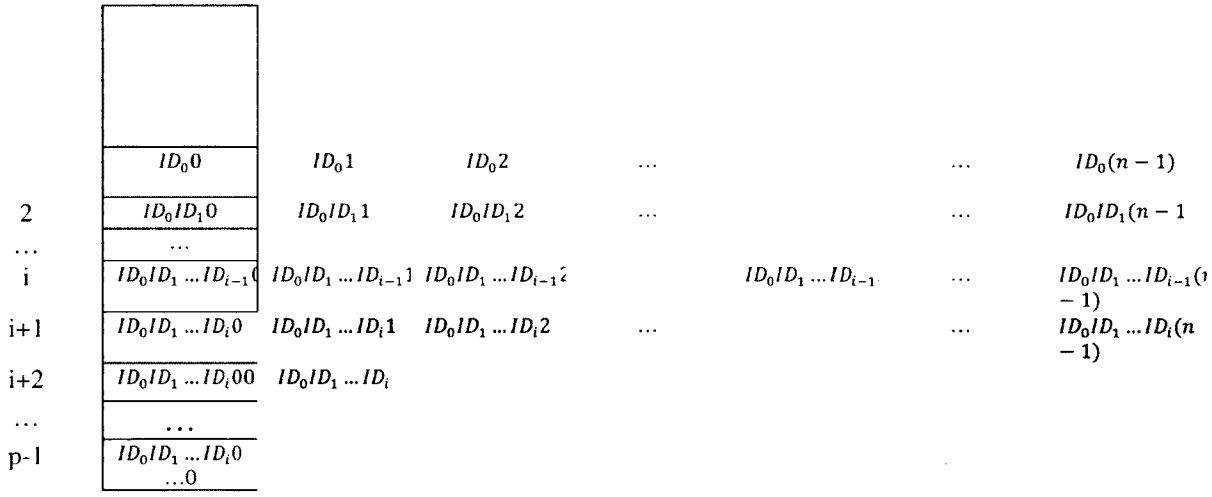


Figure 21: table des IDs

La partie inférieure de la table (partie foncée) contient les IDs de tous les enfants du nœud ξij. Les IDs non attribués de cette partie peuvent être alloués par le nœud ξij aux nouveaux nœuds qui demande l'insertion ou la réinsertion.

La partie supérieure de la table peut être utilisée par certains algorithmes tels que les algorithmes de routage « ID unicast » et Multicast présenté dans la 3.6.

3.6 Routage et échange des messages

La table de voisinage sert au routage pour acheminer les messages selon les communications Unicast, Multicast et Broadcast. La partie inférieure de la table (i.e. du niveau i jusqu'au niveau p-1) sert principalement au routage. La partie supérieure de la table (i.e. du niveau i-1 jusqu'au 0) sert principalement à l'algorithme de réinsertion et à certains types de routages.

3.6.1 Types de messages

Les messages sont divisés en deux catégories : charge utile et management. Les messages de la catégorie charge utile contiennent de l'information destinée à l'application. Les messages de la catégorie management contiennent de l'information concernant la gestion du réseau.

Un message de la catégorie charge utile peut être du type : Unicast, Multicast ou Broadcast. La mise à jour d'une variable partagée sur le réseau, comme l'histogramme, est un exemple de message de ce type. Un message de la catégorie management peut être de type : mise à jour de la table de voisinage (MAJ), requête d'insertion, surveillance du voisin (Keepalive), etc. Pour réduire le trafic relié à la MAJ, seulement la partie de la table qui est affectée par la mise à jour peut être envoyée et non pas toute la table.

3.6.2 Identificateur de message (optionnel)

Pour améliorer la fiabilité de la communication, chaque message peut avoir un identificateur. Cet identificateur sert principalement aux nœuds qui se réinsèrent dans le réseau après la disparition de leur nœud parent pour vérifier qu'ils ont reçu tous les messages multicast et Broadcast qui leurs sont destinés.

Cet identificateur est composé de l'ID et de l'IP du nœud source combiné à un numéro de séquence. Ce numéro de séquence est géré par le nœud source. Pour éviter que le numéro de séquence redémarre à zéro pendant le processus de disparition-détection-réinsertion, sa taille doit être d'un nombre de bits suffisants pour distinguer tous les messages présents dans le réseau pendant l'absence d'un noeud. Pour réduire la taille de ce numéro de séquence, il est possible d'associer un numéro de séquence différent pour chaque type de message.

3.6.3 Unicast

La communication unicast Ring-Tree est basée sur la position du nœud destinataire. Elle permet à un nœud source (S) d'envoyer un message à un nœud destinataire (D) occupant une position donné dans le réseau. Elle est appelé également ID unicast, car la position du nœud est représentée par son ID.

Il existe deux façons d'envoyer d'ID unicast : communication directe et communication via Ring-Tree. Avec la communication directe, S possède l'adresse IP du nœud D, il communique

directement avec lui. Si S ne possède pas l'adresse IP du nœud D, connaissant l'ID du D, il utilise le protocole ARM, décrit à la section 3.6.4, pour obtenir son IP.

Avec la communication unicast via Ring-Tree les messages provenant du nœud source S sont routés jusqu'à destination par les autres nœuds du réseau. Les nœuds Ring-Tree agissent donc comme routeurs. Pour choisir un routeur, S compare l'ID du D avec les IDs des nœuds de sa table de voisinage (table des IDs). Le nœud qui est le plus proche au nœud destinataire est choisi comme routeur. Si ce routeur n'a pas l'IP du nœud destinataire D, il effectue le même algorithme que le nœud source pour trouver un autre routeur plus proche du nœud destinataire et ainsi de suite.

Pour choisir le routeur le plus proche, un nœud compare l'ID du nœud destinataire avec les IDs de tous les nœuds représentés dans sa table de voisinage. Le routeur le plus proche est le nœud qui a la partie commune avec l'ID du nœud destinataire cadrées à gauche de son ID la plus large. L'algorithme de cette fonction est présenté en ANNEXE C.

Il est toujours possible d'envoyer un message à un nœud dont on connaît son adresse IP, quelque soit sa position dans le réseau. Mais, la communication unicast via Ring-Tree permet l'envoie des messages à un nœud qui occupe une position donnée quelque soit la valeur de son adresse IP. Dans ce dernier cas, si le nœud est remplacé par un autre, le nouveau nœud reçoit les messages et non pas le nœud qui a quitté sa position.

3.6.4 Protocole ARM (« Address Resolution Management »)

Le protocole ARM est nécessaire dans le cas où un nœud quelconque A désir obtenir l'adresse IP d'un autre nœud quelconque B dont il connaît son ID. Si le nœud B ne fait pas partie de la table de voisinage de A, le nœud A envoie un message ARM au nœud le plus proche de B présent dans sa table de voisinage. Pour choisir le nœud le plus proche à B, le nœud A compare l'ID du B avec les IDs des nœuds de sa table de voisinage (table des IDs). Le nœud le plus proche est determiné en utilisant la méthode décrite à la sous-section 3.6.3 (algorithme en ANNEXE C). Si ce nœud n'a pas un accès direct à l'information du nœud destinataire, il effectue le même algorithme que le nœud source A pour trouver un autre nœud plus proche à B et ainsi de suite. Le nœud possédant de l'information sur le nœud destinataire, envoie l'IP du nœud destinataire au nœud source. Le message ARM contient l'adresse IP du nœud A pour que le premier nœud qui connait l'adresse IP de B, envoie cette adresse au nœud A.

3.6.5 Multicast

Lorsqu'un message est diffusé à un groupe de nœuds, la communication s'appelle Multicast. Le service multicast le plus connu est Mbone[60]. Il en existe aussi des commerciaux tel que Akamai[61] et iBeam[62]. Le Multicast peut fonctionner au niveau IP ou au niveau du réseau « overlay ». Le multicast IP est peu déployé sur Internet, car il requiert des modifications aux routeurs d'où la motivation d'acheminer les messages au niveau du réseau « overlay ».

Le Ring-Tree permet la communication multicast au niveau « overlay ». Ce type de communication permet de rejoindre une partie du réseau définit par un nœud parent et tous ses descendants (Figure 22).

Ce type de communication est utilisé lorsqu'un nœud doit diffuser une information qui concerne ses descendants seulement. Par exemple une mise à jour de la table de routage. Tous les descendants ne peuvent avoir de l'information que par leurs grappins. Ce type de communication s'appelle également Broadcast vers le haut i.e. vers les nœuds au niveau p-1 qui est le niveau feuille.

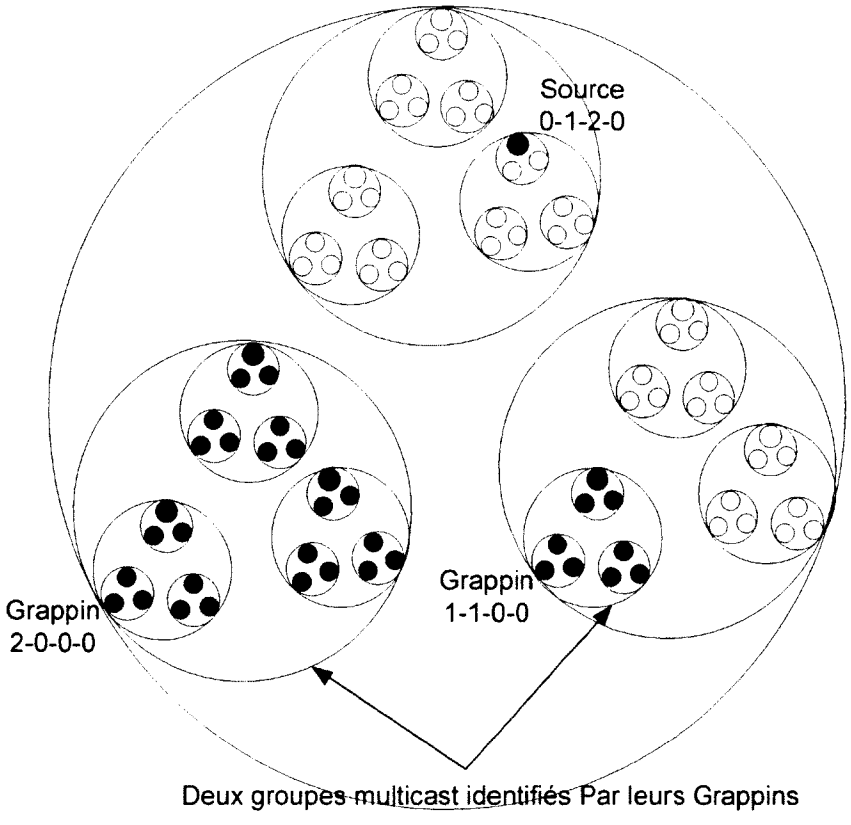


Figure 22: représentation des exemples de groupe de multicast

Lorsqu'un nœud reçoit un message de type multicast et lorsqu'il est le destinataire, il l'achemine à tous ses enfants. Ceci revient à acheminer le message à l'application et à retransmettre le message à tous les nœuds qui se trouvent dans la partie inférieure de sa table de voisinage (Figure 23). Chaque enfant à son tour retransmet le message à tous ses enfants i.e. tous les nœuds qui se trouvent dans la partie inférieure de sa table de voisinage et ainsi de suite jusqu'au niveau feuille.

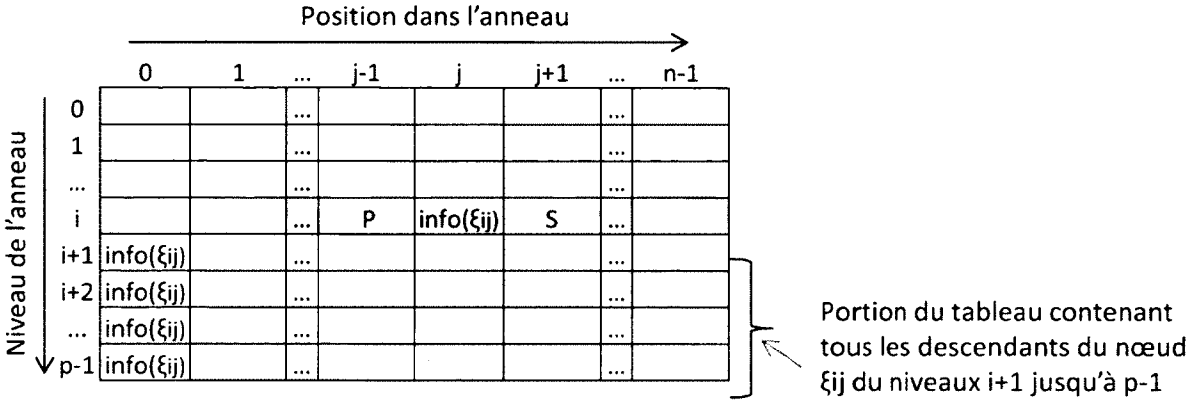


Figure 23: représentation du groupe multicast dans la table de voisinage

Pour acheminer efficacement le message à tous ses descendants, le nœud commence à envoyer le message aux nœuds de la première ligne ensuite aux nœuds de la deuxième ligne jusqu'au niveau feuille. De cette façon l'envoie multicast profite du parallélisme offert par la topologie. Pendant que le nœud envoie le message aux nœuds de niveaux supérieurs les nœuds des niveaux inférieurs envoient, en même temps, le message à leurs niveaux supérieurs.

Un nœud quelconque S peut envoyer des messages multicast directement au nœud grappin du groupe Multicast. Pour cela il faut avoir son adresse IP. S'il ne possède pas cette adresse, le nœud source peut l'obtenir via le protocole ARM décrit à la section 3.6.4. Le nœud S peut également s'en servir de la topologie Ring-Tree pour acheminer les messages ID multicast. Dans ce cas, même si le nœud grappin disparait le nœud qui le remplace continue à recevoir les messages et les acheminer au groupe multicast. Il existe donc deux façons pour envoyer un message ID multicast : communication directe et communication via Ring-Tree.

Avec la communication directe, S obtient l'adresse IP du nœud D et il communique directement avec lui. Avec la communication multicast via Ring-Tree les messages provenant du nœud source S sont routés jusqu'à destination par d'autres nœuds du réseau agissant

comme routeurs. Pour choisir un routeur, S compare l'ID du D avec les IDs des nœuds de sa table de voisinage (table des IDs). Le nœud qui est le plus proche au nœud destinataire est choisi comme routeur. Si ce routeur n'a pas l'IP du nœud destinataire D, il effectue le même algorithme que le nœud source pour trouver un autre routeur plus proche du nœud destinataire et ainsi de suite.

Il est toujours possible d'envoyer un message à un nœud dont on connaît son adresse IP, quelque soit sa position dans le réseau. Mais, la communication ID multicast permet l'envoie des messages à un nœud qui occupe une position donnée quelque soit la valeur de son adresse IP. Dans ce cas, si le nœud est remplacé par un autre, le nœud remplaçant reçoit les messages et non pas le nœud qui a quitté sa position.

3.6.6 Broadcast

La communication Broadcast permet à un nœud source S quelconque d'envoyer un message à tous les nœuds du réseau. Chaque nœud utilise sa table de voisinage pour l'envoie et le routage des messages. Il existe deux types d'envoi Broadcast : Broadcast Direct et Broadcast Circulaire.

Avec le Broadcast Direct un nœud source S envoie le message à tous ses descendants (similaire à multicast définit à la section 3.6.5) ainsi qu'à tous les nœuds du même niveau que lui. Lorsqu'un nœud reçoit un message de type Broadcast, il l'achemine à l'application et il le transmet à tous les nœuds de tous les anneaux à lequel il appartient sauf l'anneau de lequel il l'a reçu. Pour accélérer la transmission du message, un nœud commence toujours par envoyer le message à son parent ensuite à ses frères (i.e. les nœuds du même niveau) ensuite à ses descendants. Il commence donc toujours à envoyer le message au nœud qui existe dans la première case de la ligne au niveau i (même niveau que lui) ensuite à tous les nœuds de cette même ligne. Ensuite à tous ses descendants i.e. de la ligne i+1 jusqu'à la ligne p-1.

En résumé, le nœud source S transmet le message à tous les nœuds à partir de son niveau i jusqu'au niveau p-1. Chaque nœud qui reçoit le message, le transmet à tous les nœuds sur tous les anneaux à partir de son niveau jusqu'au niveau p-1 sauf ceux de l'anneau de lequel il l'a reçu le message. L'ordre des nœuds à lesquels le message est envoyé est choisi d'une façon à profiter du parallélisme de la topologie et par conséquent réduire le temps total de Broadcast.

Avec le Broadcast direct, un nœud donné doit acheminer un même message à plusieurs nœuds l'un à la suite de l'autre. Ceci revient à envoyer, à partir d'un nœud donné, un même message plusieurs fois, ceci multiplie le temps de transmission par le nombre de nœuds à lesquels il faut envoyer le message. Le temps maximal pour un message Broadcast direct, est estimé à $(np - 1)$ sauts. Un saut correspond au passage d'un message d'un nœud à un autre.

Une autre méthode possible pour envoyer un message Broadcast c'est de faire circuler le message sur l'anneau jusqu'au nœud parent. Le nœud source n'envoie pas le message à son parent directement sauf s'il est le précédent du parent. Chaque nœud envoie le message à ses suivants seulement. Il envoie à tous ses suivants de tous les anneaux à lesquels il est relié. Lorsque le nœud parent reçoit le message, il l'envoie à son tour à tous ses suivants de tous les anneaux à lesquels il est relié incluant l'anneau où se trouve le nœud duquel il l'a reçu le message. Lorsque le message revient au nœud qui a débuté l'envoie dans l'anneau, la transmission arrête. Ceci permet au nœud qui a débuté le Broadcast dans cet anneau de s'assurer que le message est reçu par tous les nœuds de cet anneau.

Il est possible d'inverser le sens d'envoi des messages i.e. un nœud envoie le message à ses précédents au lieu de l'envoyer à ses suivants. Comme le précédent surveille son suivant, la réception de ces messages de son suivant lui permet de vérifier qu'il est toujours présent. Ceci réduit le nombre de message « keepalive » nécessaire à la surveillance des nœuds.

Voici un exemple de routage de Broadcast circulaire : le nœud 211, de la Figure 24, envoie un message de type Broadcast à son suivant le nœud 212. Le nœud 212 achemine le message à son suivant 210. Le nœud 210 l'achemine au nœud 220 et au nœud 211. À la réception du message, le nœud 211, assuré que le message a parcouru l'anneau, arrête la transmission, car c'est lui qui l'a débuté dans cet anneau. Le nœud 220 l'achemine au nœud 200 qui est le seul nœud suivant. Le nœud 200 achemine le message à tous les nœuds suivants soit les nœuds 000, 210 et 201. Le nœud 210, assuré que le message a parcouru l'anneau, arrête la transmission. Le nœud 201 achemine le message à 202. Le nœud 000, à son tour, envoie le message aux nœuds 100, 010 et 001 et ainsi de suite. Chaque nœud qui a débuté le message dans son anneau s'assure que le nœud a parcouru tous les nœuds de l'anneau. Pour cela, un champ est ajouté dans le message Broadcast pour indiquer le nœud qui a débuté le message dans chaque anneau.

Il est possible de montrer qu'avec ce type de routage, un message Broadcast arrive à tous les nœuds en un maximum de $(n-1)(2p-1)$ sauts. Le temps de mise à jour moyen, si on néglige le temps d'exécution, est de $(n-1)(2p-1)T_{tr}$ avec T_{tr} le temps de transmission moyen pour qu'un message traverse un saut.

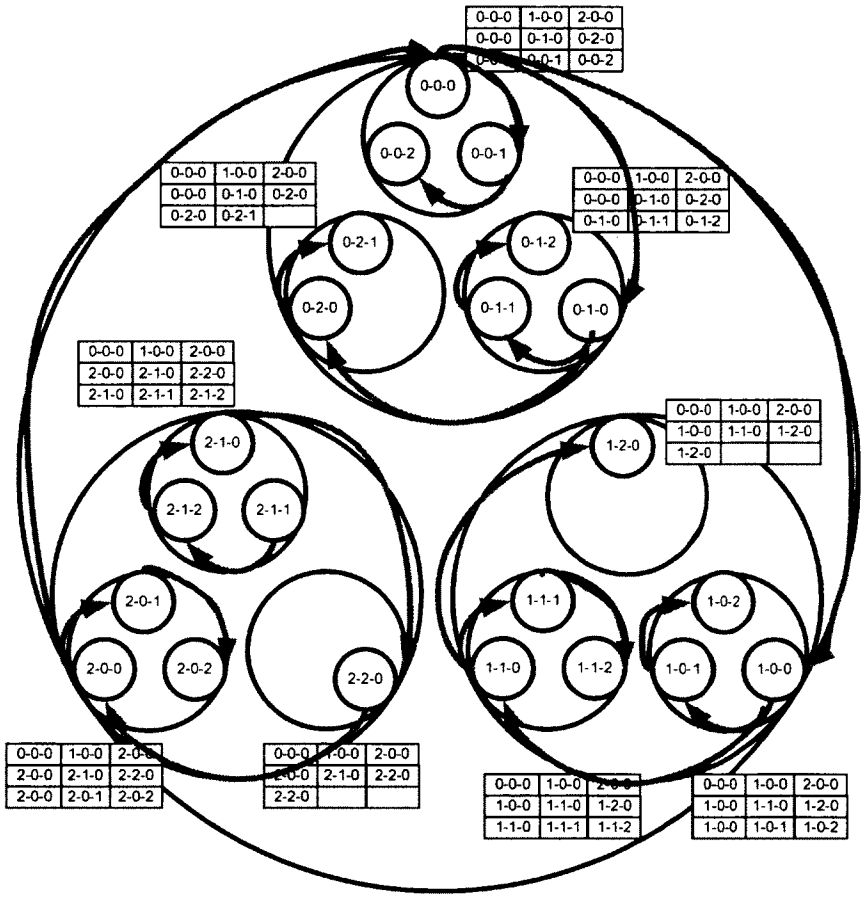


Figure 24: exemple de Broadcast circulaire envoyé par le nœud 2-1-1

3.6.7 Autres types de routage et agrégation de données

Grâce à la structure Ring-Tree et à la table de voisinage, différents protocoles de communication peuvent être définis. Voici un exemple de protocole de communication qui permet à un nœud d'acheminer un message vers le nœud grappin du niveau tronc (i.e. parent du niveau 0) sans communication direct. Ce protocole consiste à envoyer le message d'un nœud de niveau i à son parent immédiat. Le parent immédiat appartenant au niveau j ($j < i$)

achemine le message à son parent de niveau k (k < j < i) et ainsi de suite jusqu'au parent de niveau 0.

Il est possible de faire une agrégation de données au niveau du nœud parent. Un parent attend de recevoir les données de tous ses descendants avant de faire une agrégation ou fusion de données ou tout autre traitement de données. Une fois l'agrégation ou le traitement de données est terminé, il achemine le résultat à son parent; le parent fait de même et ainsi de suite jusqu'au tronc. Le dernier parent i.e. le parent de niveau 0, reçoit le résultat compilé de tous les nœuds du réseau.

Ce nœud est le premier nœud du réseau. Il possède l'ID 0. Tous les nœuds du réseau connaissent le nœud 0 car son IP se trouve à la case (0,0) de toutes les tables de voisinage. Un nœud peut donc envoyer un message au nœud 0 directement. Mais, si un grand nombre de nœuds lui envoie des messages d'une façon directe et en même temps, le nœud 0 peut être surchargé. Le protocole déjà décrit permet d'éviter cette situation car les messages arrivent au nœud 0 via ses enfants seulement. Pour éviter des situations similaires, il est possible d'instaurer une règle générale. Cette règle se base sur le concept de connaissance mutuelle définit à la section 3.4.3. Cette règle stipule qu'un nœud n'envoie de messages à un autre nœud que s'il y a une connaissance mutuelle entre les deux nœuds et un nœud n'accepte pas de message d'un nœud sauf les messages de management tel que l'insertion, la réinsertion et ARM, que s'il y a une connaissance mutuelle avec ce nœud. Si une telle règle est activée, l'unicast via Ring-Tree tel que définit à la section 3.6.3 et le Multicast via Ring-Tree (section 3.6.5) ne fonctionne plus à moins qu'on les traite comme étant des exceptions à cette règle.

Un autre protocole de communication de type Multicast peut être défini pour permettre de diffuser un message à tous les nœuds à partir d'un niveau quelconque j jusqu'au niveau k inclusivement, avec j < k. Ce type de Multicast basé sur le niveau est appelé multicast de niveau ou Multicast_K, k est le niveau à lequel la diffusion de message doit s'arrêter. Ce type de communication peut être utile pour des mécanismes de réinsertion d'un nœud orphelin de niveau quelconque k dans un anneau de niveau strictement inférieur à k (section 3.7). Un nœud orphelin de niveau k est un nœud qui a détecté la disparition de son parent immédiat de l'anneau k. Ce nœud devient le nouveau parent de cet anneau mais séparé du réseau i.e. orphelin. S'il possède au moins un enfant, il doit s'insérer dans un anneau de niveau

strictement inférieur à k pour qu'il puisse garder ses descendants. Ce nœud orphelin peut alors envoyer un message Multicast_K pour demander sa réinsertion dans le réseau.

3.7 Algorithme de recouvrement

Il ne suffit pas de créer une topologie, il faut aussi la maintenir sans affecter grandement la performance. Le maintient de la structure est nécessaire dans le cas de bris de communication avec un nœud. Comme chaque nœud overlay est un routeur, si la communication est brisée, il ne serait plus possible d'acheminer des messages via ce nœud ni de rejoindre ses descendants. Le bris de communication est souvent causé par une disparition involontaire d'un nœud. L'effet de cette disparition sur la structure Ring-Tree dépend de la position du nœud dans la topologie et du nombre de ses descendants. L'impact au réseau est plus important si le niveau du nœud disparu est plus bas i.e. plus proche du tronc. Plus un nœud est proche du tronc, plus le nombre de ses descendants potentiels est grand.

Pour maintenir la structure, trois étapes sont nécessaires : la surveillance des nœuds, la détection des bris de communication et le recouvrement. La surveillance des nœuds et la détection des bris de communication sont présentées à la sous section suivante. Deux approches de recouvrement sont présentées : une approche centralisée (section 3.10) et une approche distribuée (sous section 3.7.2).

3.7.1 Surveillance des nœuds et détection des bris de communication

Un nœud quelconque ξ est toujours surveillé par son précédent. Le précédent est le nœud qui existe dans la table de voisinage dans le même anneau (même ligne) que ξ et le premier nœud situé à sa gauche. Lorsque le nœud ξ est situé dans la première colonne, son précédent est le dernier nœud à droite existant sur l'anneau. Le nœud ξ surveille toujours son suivant. Le suivant est le nœud qui existe dans la table de voisinage au même anneau (même ligne) que ξ et le premier nœud situé à sa droite. S'il n'y a aucun nœud à sa droite, ξ surveille le nœud parent situé dans la case 0. Rappelons que chaque ligne correspond à un anneau fermé.

Chaque nœud, sur chaque anneau à lequel il appartient et quelque soit sa position sur cet anneau, surveille son suivant s'il existe et il est surveillé par son précédent s'il existe. Comme un nœud de niveau i appartient à tous les anneaux du niveau i jusqu'à p-1 (total de p-i

anneaux), il surveille donc un maximum de p-i suivants et il est surveillé par un maximum de p-i nœuds précédents.

Pour surveiller son suivant, un nœud lui envoie un message « keepalive ». À chaque fois qu'un nœud reçoit de son précédent un message keepalive, il doit lui répondre par un message de confirmation « ACK ». À défaut de le recevoir et après un certain temps d'attente prédéfini, le précédent considère le nœud comme étant déconnecté.

Les messages « keepalive » peuvent être encapsulés dans les autres messages échangés habituellement entre ces nœuds (mécanisme de piggyback). Un minuteur est remis à zéro à chaque réception du message du nœud suivant. L'envoie des messages « keepalive » seuls n'est donc nécessaire que s'il n'y a pas eu une communication entre les deux nœuds depuis un temps prédéfini.

Lorsqu'un nœud Ring-Tree établit des connexions TCP avec les autres nœuds, un bris de connexion peut être rapporté à la couche Ring-Tree s'il y a lieu. Lorsqu'un bris de connexion est rapporté, seulement un nœud précédent a l'autorité de considérer son suivant comme étant déconnecté. Lorsqu'un précédent considère son suivant comme étant déconnecté, le précédent exécute l'algorithme de recouvrement.

Lorsqu'un nœud disparaît, un bris de communication est détecté par ses p-i nœuds précédents. Mais, les p-i précédents ne détectent pas nécessairement le bris de communication en même temps. Il est possible que des bris de communications soient détectés par seulement quelques précédents. Ceci arrive lorsque le nœud n'est pas disparu mais les connexions entre seulement quelques nœuds sont brisées temporairement. La conception d'un algorithme de recouvrement, doit tenir compte de ces situations.

3.7.2 Algorithme de recouvrement distribué

Cas de bris de communication simple

Lorsqu'un nœud précédent détecte un bris de communication avec un frère (i.e. un nœud du même anneau qui n'est parent) dans un anneau de niveau quelconque k, il établit une connexion avec le nœud suivant s'il existe (Figure 25); il met à jour sa table de voisinage; il envoie un message pour une mise à jour partielle de la table de voisinage. Il envoie ce message sous forme multicast à tous les nœuds de l'anneau k ainsi qu'à tous ses descendants.

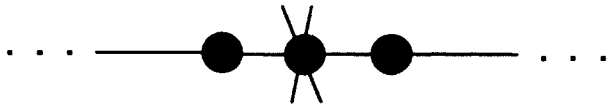


Figure 25: bris de communication simple

Cas de bris de communication multiple

Si le nœud suivant ne répond pas, le précédent continue à contacter les nœuds subséquents dans l'anneau jusqu'à ce qu'un nœud répond (Figure 26); il établit un lien avec ce dernier; il met à jour sa table de voisinage; il envoie un message pour une mise à jour partielle de cette table. Il envoie ce message sous forme multicast à tous les nœuds de l'anneau k ainsi qu'à tous ses descendants.

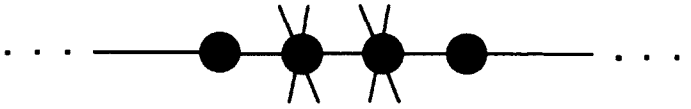


Figure 26: bris de communication multiple

Si aucun subséquent ne répond, le nœud est donc devenu seul dans cet anneau; il met à jour sa table de voisinage; il envoie un message, sous forme multicast, à tous ses descendants pour une mise à jour partielle de la table de voisinage.

Cas de bris de communication avec un nœud parent

Si un nœud précédent $\xi\mu$ détecte un bris de communication avec son parent, il se met en état appelé « orphelin actif » (section 3.8). Dans cet état, l'orphelin actif $\xi\mu$ est responsable de réinsérer, dans le réseau, tous les nœuds de son anneau incluant lui-même ainsi que tous ses descendants. Tous les nœuds de son anneau (i.e. ses frères) sont donc considérés comme étant ses descendants de niveau μ. L'orphelin actif $\xi\mu$ envoie un message à ses descendants de niveau μ et à tous les autres descendants pour leur demander de se mettre en état appelé « orphelin passif » ou « attente ». Ensuite, il envoi des requêtes de réinsertion aux nœuds de bas niveaux (i.e. strictement inférieur à μ) pour demander une réinsertion.

Une demande de réinsertion sert à reconnecter un nœud ainsi que tous ses descendants au réseau. Pour garder tous ses descendants, un orphelin actif de niveau μ doit chercher un anneau de niveau strictement inférieur à μ, sinon il sera obligé de se libérer des nœuds des hauts niveaux. Une demande de réinsertion inclut donc toujours le niveau maximal à lequel

l'insertion est demandée. Après insertion, le niveau du nœud $\xi\mu$ est déterminé selon le niveau de l'anneau à lequel il réussi à s'insérer.

Les nœuds qui doivent potentiellement être contactés par $\xi\mu$ pour une demande d'insertion sont les nœuds de bas niveaux. Leurs adresses existent dans la table de voisinage du nœud $\xi\mu$, aux lignes inférieures à μ (i.e. la partie supérieure de sa table de voisinage). Le nœud $\xi\mu$ demandant une insertion au niveau strictement inférieur à μ, envoie sa demande au nœud précédent du parent au niveau μ-1. Si ce dernier n'accepte pas la demande de réinsertion, le nœud $\xi\mu$ contacte les nœuds, à partir du niveau μ-2, l'un après l'autre en suivant l'ordre spirale jusqu'à ce qu'un nœud accepte sa réinsertion. N'importe quel nœud de niveau inférieur à μ-1 (i.e. de niveau μ-2 jusqu'à 0) peut, s'il a au moins une place, accepter une demande de réinsertion de niveau μ-1. Mais, seulement le nœud parent de l'anneau μ-1 peut accepter une demande de réinsertion de niveau μ-1 car les autres nœuds de cet anneau sont des descendants de ce même niveau. Seulement le parent d'un anneau quelconque a le droit d'insérer ou de réinsérer des nœuds à l'anneau.

Lorsque la demande de réinsertion est acceptée par un nouveau parent, le nouveau parent assigne à $\xi\mu$ un nouveau ID représentant le nouveau niveau de l'anneau à lequel le nœud est accepté ainsi que sa position dans cet anneau (section 3.3.1); il met à jour sa table de voisinage; il envoie un message multicast de mise à jour à tous ses descendants de niveaux supérieurs ou égal au niveau du nœud ajouté pour les informer du nouveau ajout; il envoie le nouveau ID ainsi que la table de voisinage partielle (section 3.4.1) à $\xi\mu$. À la réception de ces informations, le nœud $\xi\mu$ passe de l'état orphelin actif à l'état connecté; il assigne des nouveaux IDs à ses descendants à partir de son ID et de leurs positions dans les différents anneaux; il envoie un message multicast à tous ses descendants pour mettre à jour la table de voisinage et pour les informer de leurs nouveaux IDs; par ce fait même, les nœuds descendants passent de l'état orphelin passif à l'état connecté.

Si un orphelin actif ne réussit pas à s'insérer dans aucun des niveaux inférieurs à son niveau initial, après un nombre d'essai prédéterminé, il se considère séparé du réseau; il envoie un message multicast à tous ses descendants pour défaire le sous réseau. À la réception de ce message, chaque nœud passe de l'état orphelin passif à l'état non connecté; il essaye de se joindre de nouveau au réseau à partir d'une liste de nœuds actifs. La Figure 27 montre l'organigramme de l'algorithme proposé.

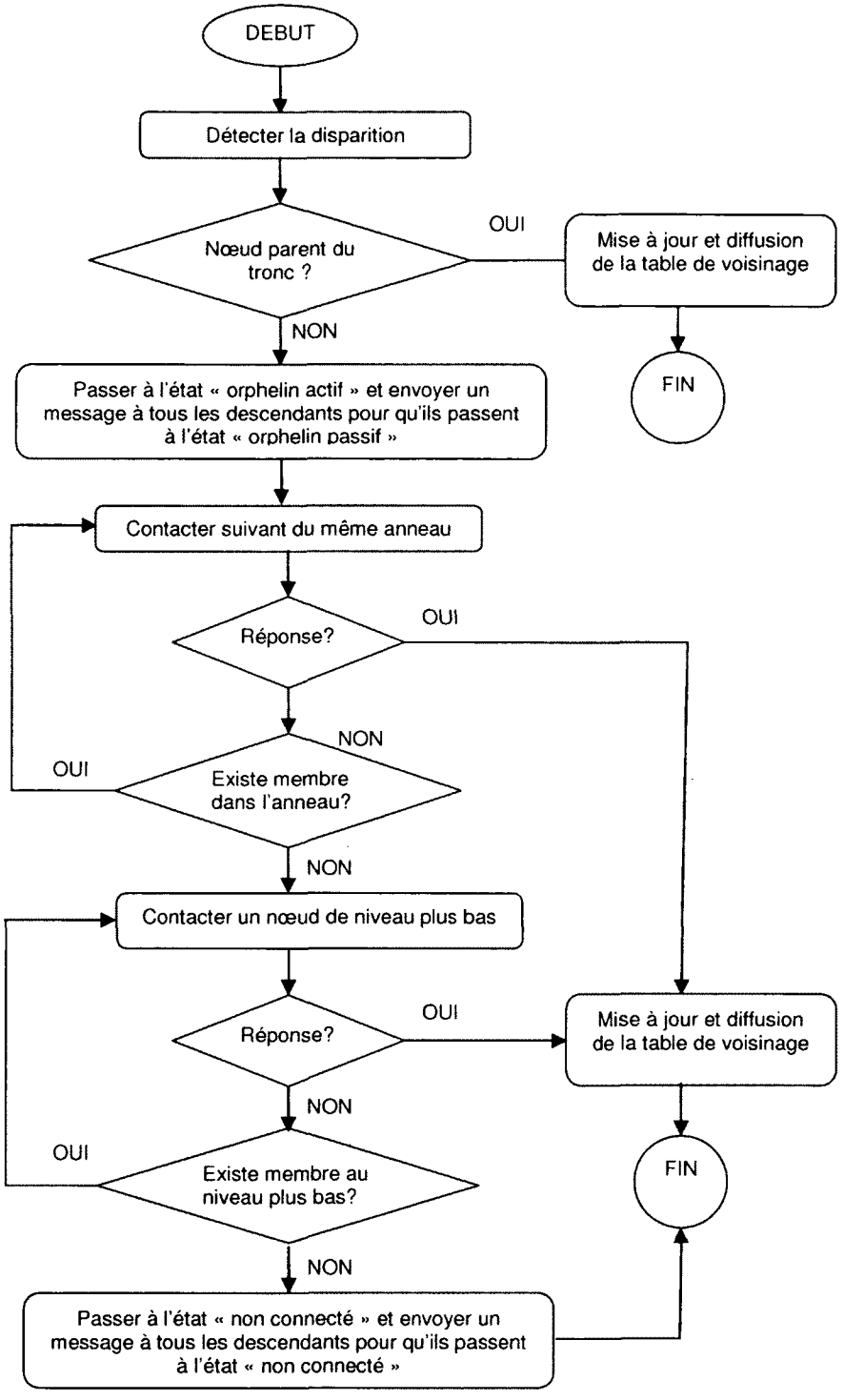


Figure 27: organigramme de l'algorithme de recouvrement

Disparition d'un nœud parent

Un nœud de niveau i est un parent pour p-i anneaux différents de niveau i+1 jusqu'à p-1. Lorsque ce nœud disparaît, ceci est détecté par les précédents de ces p-i anneaux (Figure 28).

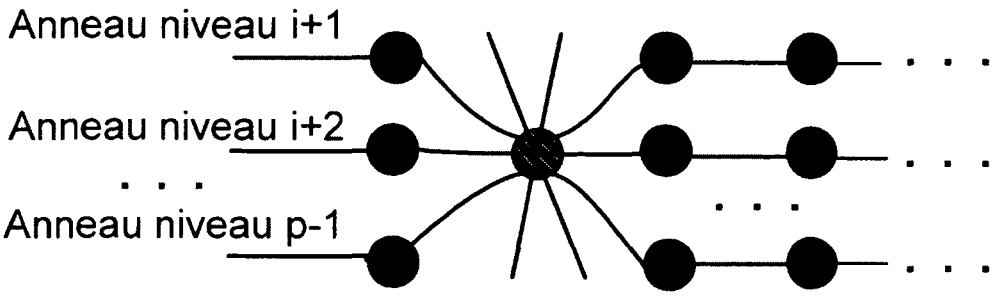


Figure 28: bris de communication avec un parent

Sur chacun de ces anneaux, le précédent qui a détecté la disparition de son parent devient un parent de cet anneau mais orphelin actif. Un maximum de p-i-1 précédents deviennent donc des orphelins actifs. Chaque orphelin actif essaye de se réinsérer, comme étant enfant, dans un niveau plus bas (Figure 29). Le même algorithme est effectué par tous les orphelins actifs.

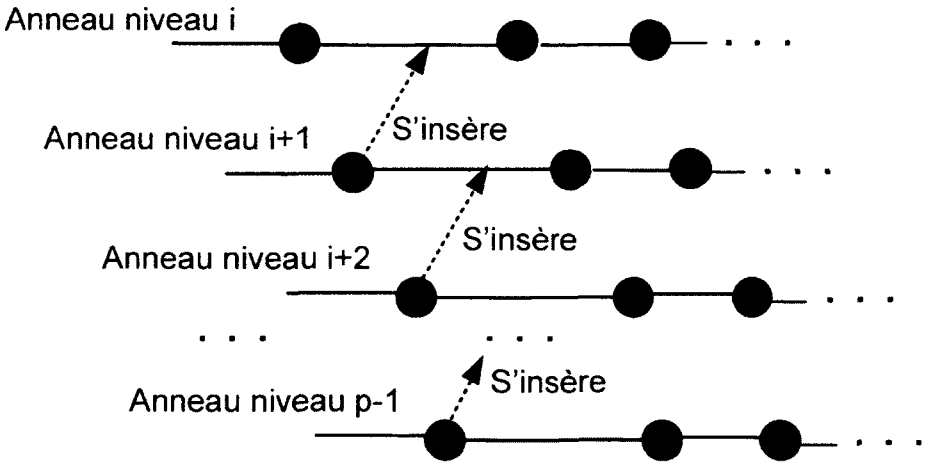


Figure 29: rétablissement après une disparition d'un nœud parent

La disparition d'un nœud quelconque ξij est donc détecté par tous ses précédent situés aux différents niveaux de niveau i jusqu'à p-1 (Figure 30). Après la détection de la disparition du nœud ξij, chaque précédent de niveau quelconque k appelé P_k devient le parent de l'anneau k, avec k = i..p-1. Le précédent P_k essaie de s'insérer dans un anneau de niveau inférieur à k.

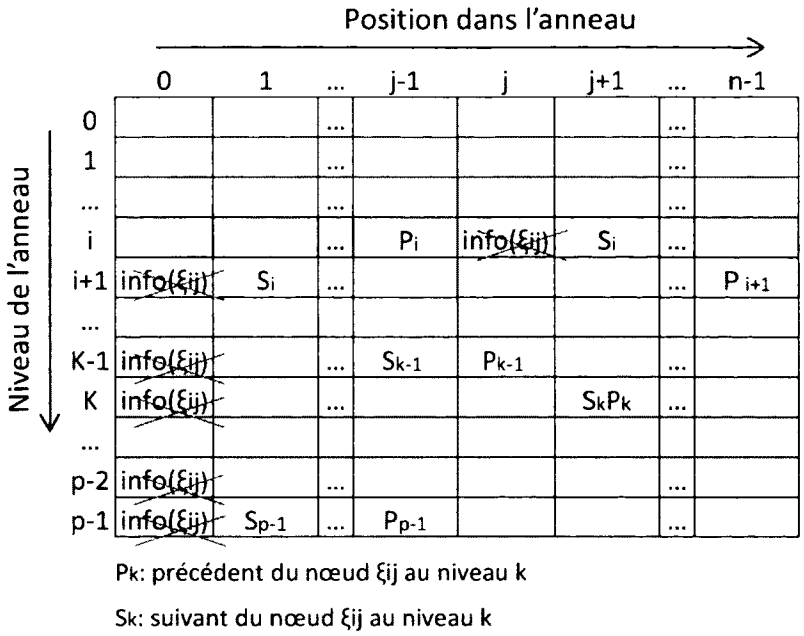


Figure 30: représentation de la disparition du nœud ξij dans la table de voisinage

Comme déjà expliqué, lorsque le précédent P_k détecte un bris de communication avec son parent ξ_{ij} au niveau de l'anneau k, il change son état à « orphelin actif », il envoie un multicast à tous les nœuds de l'anneau k ainsi qu'à tous ses descendants pour leur demander de se mettre en état « orphelin passif ». Il envoie une demande de réinsertion de niveau k-1 au nœud P_{k-1}. Si P_{k-1} a également détecté un bris de communication avec le nœud ξ_{ij}, il serait donc en état orphelin actif (parent de l'anneau) il peut accepter la demande de réinsertion. S'il n'a pas encore détecté un bris de communication avec ξ_{ij} il peut soit refuser la demande de réinsertion du nœud P_k soit envoyer un message keepalive urgent à son parent pour s'assurer qu'il est là. Si le nœud parent ne répond pas, P_{k-1} passe à l'état orphelin actif; accepte la demande de réinsertion de P_k; lance l'algorithme de recouvrement. Une schématisation de l'algorithme de recouvrement distribué est présentée en ANNEXE B.

3.7.3 Gestion des différentes situations

Situation 1 : disparition du nœud 0 du niveau tronc

La détection du nœud parent du niveau tronc doit être géré différemment. Car, ce nœud n'a aucun niveau inférieur. Lorsque le précédent du nœud parent de niveau tronc détecte sa disparition, il n'essaye pas de se réinsérer dans un niveau inférieur mais tout simplement il le

remplace (modifie son ID), ensuite il envoie un multicast à tous les nœuds du réseau pour mettre à jour leur table de voisinage.

Situation 2 : décision collective lorsqu'un nœud disparaît

Au lieu qu'un seul nœud précédent décide que son suivant frère est disparu, il est possible de coordonner la décision avec le parent de l'anneau. Dans ce cas, un nœud de niveau quelconque k n'est considéré disparu que si le précédent et le parent de l'anneau ensemble le décident. La disparition d'un nœud est détectée par le précédent ou par le parent. Mais, le parent seul autorise la décision de déconnexion du nœud ainsi que la mise à jour de la table de voisinage indiquant cette disparition.

Lorsque le parent est disparu, au lieu qu'un seul nœud précédent décide pour chaque anneau séparément que le parent est disparu, grâce à la structure et à la table de voisinage, il est possible de créer un réseau de nœuds précédents. Les nœuds de ce réseau peuvent collaborer pour décider collectivement via un vote ou autre mécanisme si un nœud est considéré déconnecté. Comme pour un nœud donné il existe un maximum de p-i anneaux à lesquels il est relié, la taille de ce réseau est p-i nœuds.

Situation 3 : nœud considéré déconnecté et qui essaie de contacter un nœud du réseau

Si, pour une raison quelconque, un nœud ξ reçoit un message d'un nœud considéré disparu, ce dernier ne doit pas apparaitre dans la table de voisinage de ξ. Le nœud ξ refuse donc le message et informe le nœud qu'il ne fait pas partie de sa table de voisinage. Le nœud considéré disparu doit se reconnecter au réseau.

Situation 4 : accommodation possible d'une demande de réinsertion

Lorsqu'une demande de réinsertion de niveau quelconque r est reçue par un nœud ξh de niveau h, avec h inférieur à r, si ξh a une place au niveau inférieur à r dans sa propre table de voisinage il accepte la demande de réinsertion. Si ξh n'a aucune place au niveau inférieur à r dans sa propre table de voisinage, il est possible qu'un ou plusieurs de ses descendants de niveau inférieur à r aient une place. Une façon possible d'accommoder la demande de réinsertion serait d'acheminer la demande vers ses descendants. Le nœud qui demande la réinsertion recevrait possiblement plusieurs réponses favorables. Il pourrait donc choisir une position convenable à partir de la liste de réponses reçue.

Situation 5 : demande de réinsertion non relié à un bris de communication

Il est toujours possible à un nœud de changer sa position dans le réseau. Pour cela, il peut envoyer une demande de réinsertion de type Multicast_K via un message incluant son adresse IP et le niveau d'insertion recherché. Si le nœud est de niveau k, le niveau d'insertion recherché serait strictement inférieur à k.

Situation 6 : orphelin qui reçoit une demande de réinsertion

Un nœud en mode orphelin (actif ou passif) doit refuser les demandes de connexions provenant des nouveaux nœuds. Cependant, il peut être programmé pour refuser ou accepter les demandes d'insertions provenant d'un nœud orphelin actif de niveau supérieur. Si un nœud orphelin reçoit une demande de réinsertion de niveau quelconque k et s'il a au moins une place disponible dans un niveau k ou inférieur, il place le nouveau nœud dans sa table de voisinage et lui envoie une demande qu'il se mette en état orphelin passif.

Situation 7 : orphelin passif qui surveille ses suivants

Un nœud en état orphelin passif peut continuer à surveiller ses suivants ou il peut, après un temps prédéfini, se mettre en état non connecté. S'il se met en état non connecté il envoie un message de déconnexion à tous ses descendants pour qu'ils se placent en état non connecté également. Si le nœud orphelin passif est programmé pour surveiller ses suivants pendant qu'il est en état orphelin passif, s'il perd un suivant il exécute l'algorithme de recouvrement tout en restant en état orphelin passif sauf s'il perd son parent où alors il se met en état orphelin actif avant d'exécuter l'algorithme de recouvrement.

Situation 8 : disparition de tous les nœuds du niveau tronc en même temps

Si les nœuds du niveau tronc disparaissent tous en même temps, le réseau devient séparé en un maximum de n sous-réseau. Il ne sera pas possible aux descendants d'un nœud du niveau tronc de rejoindre le reste de la communauté. La probabilité que cela arrive dépend de la probabilité qu'un nœud disparaisse, de la valeur du paramètre n et du nombre des nœuds existant au niveau tronc. Si α est la probabilité qu'un nœud quelconque disparaît involontairement et si tous les nœuds ont la même probabilité de disparaître, alors la probabilité que tous les nœuds d'un anneau donné disparaissent en même temps est α^n. Plus le réseau est dense (i.e. nombre de place utilisée sur le nombre de place disponible), plus la probabilité qu'un nœud se sépare est faible.

Priorité d'une demande de réinsertion sur une demande d'insertion

Un parent est toujours responsable de l'insertion et de la réinsertion des nœuds. Après une disparition d'un enfant, avant d'accepter une demande d'insertion, un parent attend un temps prédéfini pour donner plus de chance aux descendants du nœud disparu de se réinsérer dans cet anneau. Une demande de réinsertion concerne un ensemble de nœuds tandis qu'une demande d'insertion concerne un seul nouveau nœud. Un parent priorise donc une demande de réinsertion sur les demandes d'insertion.

3.8 Diagramme d'états

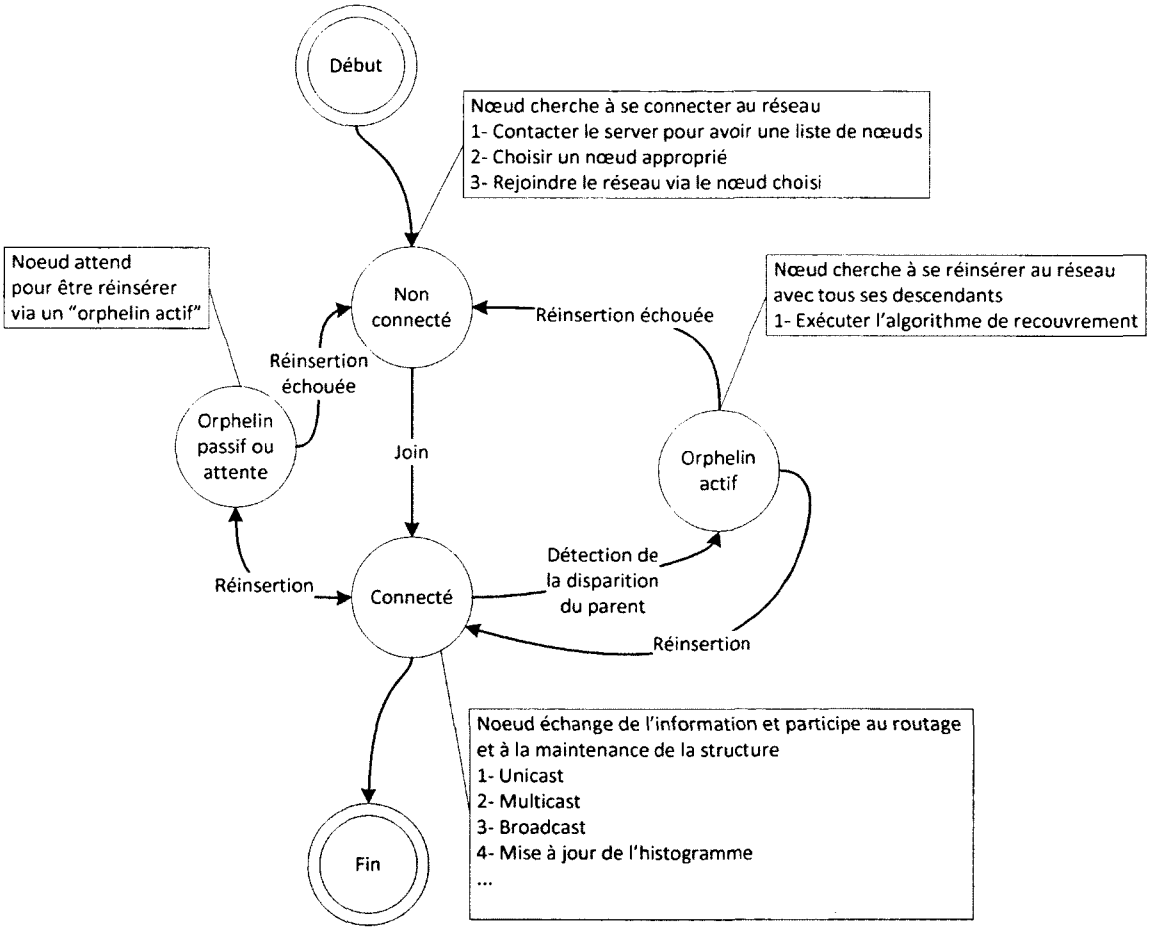


Figure 31: diagramme d'états de connexions du nœud

Chaque nœud démarre dans un état « non connecté » (Figure 31). Le but du nœud dans cet état est de se joindre à la communauté spécifié par l'application. Une fois le nœud est inséré dans

la communauté, il passe à l'état « connecté ». Dans cet état, son but principal est d'échanger des messages, de participer au routage et à la maintenance de la structure. S'il détecte un bris de communication avec son parent, il passe à l'état « orphelin actif ». Son but devient alors de se réinsérer dans la communauté. Ses frères et ses descendants, informés par le bris de communication, passent à l'état « orphelin passif » appelé aussi « attente ». Après un certain temps d'attente prédéfini (Timeout) un orphelin passif envoie un message à ses frères et ses descendants pour passer à l'état « non connecté » avant de passer, lui-même, à ce même état.
Si le nœud Ring-Tree reçoit une demande de déconnexion provenant de l'application, quelque soit son état, il passe à l'état « fin ».
Un « orphelin actif » est normalement un précédent qui détecte un bris de communication avec son parent. Un « orphelin passif » est un nœud descendant d'un « orphelin actif ».

3.9 Formation de la communauté

Pour rejoindre une communauté quelconque T(n,p), un nœud peut se servir d'un serveur de support joignable via une adresse préprogrammée résolue par DNS (Domain Name System) comme ring-tree.com afin d'obtenir une liste de nœuds actifs du réseau. Un nœud actif est un nœud qui a eu un contact récent avec le serveur. Le serveur maintient une liste de nœuds et veille, via un minuteur, qu'ils sont toujours présents. Les valeurs des paramètres n et p peuvent être fournit en même temps que l'envoie de la liste ou ils peuvent être préprogrammées.
Le premier nœud qui contacte le serveur pour avoir une liste des nœuds actifs, reçoit une liste vide et devient automatiquement le premier nœud actif du réseau. Il construit sa table de voisinage de taille $n \times p$ (Figure 32). C'est un nœud de niveau 0, malgré qu'il ne soit pas enfant dans ce niveau. Il est le parent de tous les anneaux à lesquels il appartient.

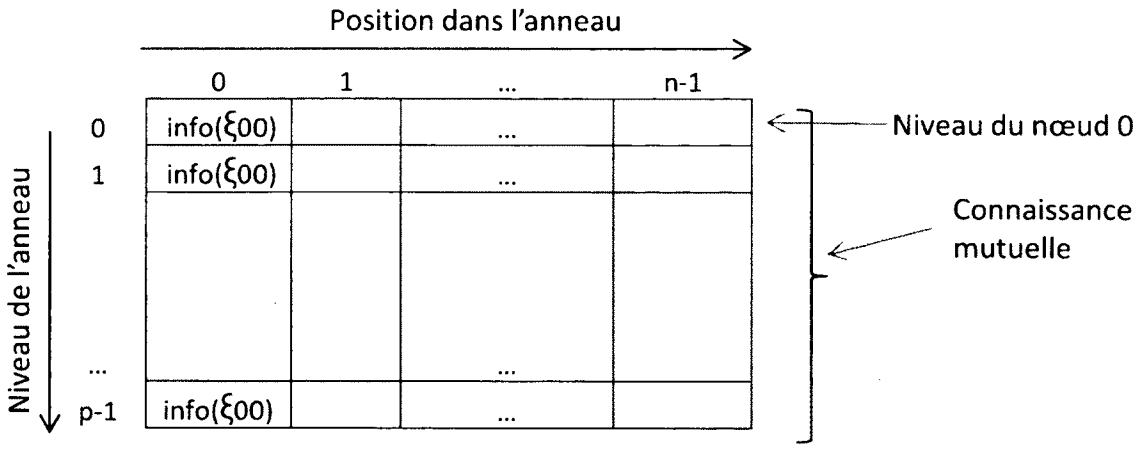


Figure 32: table de voisinage du nœud 0

Le deuxième nœud reçoit une liste qui contient l'adresse du premier nœud et il devient le deuxième nœud actif. Il effectue une demande d'insertion auprès du premier nœud qui est le parent du niveau tronc. Le parent accepte le nouveau nœud, lui fournit un ID partiel et une table de voisinage partielle tel que présenté aux sections 3.3.1et 3.4.1.

Chaque nouveau nœud demande l'insertion à un parent de la liste des nœuds actifs. À partir de cette liste, le nœud choisit celui auprès duquel il va faire sa demande d'insertion d'une façon aléatoire ou selon un critère permettant d'optimiser la structure construite. Parmi les critères possibles on peut identifier: la proximité du tronc afin de créer une structure qui réduit la probabilité de partitionnement ou bien la distance "réseau" minimale afin de créer une structure où les temps de communication seront les plus courts possibles. La distance réseau peut être obtenue via le protocole Ping et être basée sur le temps d'aller et de retour (RTT) ou le nombre de sauts. Le parent, s'il a une place disponible, dans n'importe quel anneau dont il est parent, accepte le nouveau nœud. Il lui fournit un ID partiel correspondant à sa position et une table de voisinage partiel.

Amélioration possible

Avec le routage au niveau IP, un message traverse un routeur une seule fois tandis qu'avec un routage au niveau « overlay » le même message peut traverser un routeur plusieurs fois [44]. Le routage au niveau overlay est moins performant que celui au niveau IP et pour améliorer sa performance il faut tenir compte des couches inférieures comme la couche IP [63].

Pour tenir compte des couches inférieures et améliorer la performance de la communication, les positions des nœuds Ring-Tree dans le réseau peuvent être choisies selon des techniques existantes. Par exemple, Kazaa place les nœuds voisins selon leurs préfix IP ou selon le temps minimum qui les sépare (Round Trip Time). Il est aussi possible de placer les nœuds selon le nombre de sauts minimum qui les sépare. De cette façon un message traverse un plus petit nombre de routeurs afin de se rendre à sa destination. Le message se rend plus rapidement à destination et utilise moins de ressources.

3.10 Serveur de support

Un serveur peut être utilisé pour : l'authentification des nouveaux nœuds, l'approvisionnement d'une liste de nœuds actifs, l'ajout des nœuds et le maintient de la structure. Il ne participe pas à la communication ni au routage des messages.

Différentes applications telles que la localisation de ressources, la gestion de la charge sur le réseau électrique ou n'importe quel autre application désirant créer une communauté Ring-Tree, peut utiliser un tel serveur. Un serveur peut servir plusieurs communautés.

Lorsqu'un nœud rejoint le serveur, ce dernier lui fournit les valeurs des paramètres n et p spécifique à la communauté à laquelle il demande de se connecter. Ces paramètres peuvent être différents d'une communauté à une autre.

Si l'on utilise une approche centralisée pour l'ajout de nœuds et la maintenance du réseau, le serveur a une carte de position de tous les nœuds dans le réseau Ring-Tree. Il assigne une place disponible à chaque nœud désirant s'insérer au réseau et il met à jour la carte du réseau. Tout bris de communication détecté par un précédent, selon la méthode présentée à la section 3.7.1, est rapporté au serveur pour qu'il choisisse une autre place convenable à l'ensemble de nœuds déconnectés. Dans ce cas, l'algorithme distribué de recouvrement décrit à la section 3.7.2 n'est plus nécessaire. Seuls la surveillance et la détection des nœuds disparus sont effectuées d'une façon distribuée mais c'est le serveur qui s'occupe de la réinsertion des nœuds déconnectés. Pour cela, il doit tenir compte du niveau de nœud déconnecté. Un nœud déconnecté de niveau i ayant au moins un descendant, doit être placé dans un anneau de niveau inférieur ou égal à i. Le nœud concerné et ses descendants restent connectés entre eux et c'est l'ensemble de ces nœuds qui est considéré par la réinsertion. Le serveur possède la

carte de position de tous les nœuds de réseau, il peut facilement effectuer l'ajout des nœuds déconnectés.

Même dans le cas d'une approche distribuée de l'algorithme de réinsertion, le serveur peut prendre part à l'ajout des nœuds dans le réseau. Un nœud désirant s'insérer dans le réseau contacte le serveur pour avoir une liste de nœuds actifs. Chaque nœud parent ayant au moins une place disponible dans sa table de voisinage demande au serveur de s'ajouter sur la liste des nœuds actifs. Si le serveur a une liste de taille limitée, et si la liste est pleine (le nombre de nœuds actifs dans le réseau peut être très important), il demande au nœud de le recontacter plus tard. Le nœud attend un temps prédéfini avant de redemander de s'ajouter à la liste. Si le nœud n'a aucune place disponible, il envoie un message au serveur pour retirer son adresse de la liste.

3.11 Valeurs optimales des paramètres n et p

La topologie Ring-Tree est caractérisée par les paramètres n et p. Les valeurs de ces paramètres affectent la taille maximale possible de la communauté $N = n^p$. La valeur N est déterminée selon la prévision à long terme sur l'étendue de la communauté. Elle peut être différente d'une communauté à une autre.

Pour une même valeur N, plusieurs valeurs de n et p sont possibles. Théoriquement ces paramètres peuvent avoir n'importe quelles valeurs. Mais, ces dernières affectent plusieurs éléments de performance tel que :

- Nombre maximal des sauts d'un message Broadcast circulaire : $(n-1)(2p-1)$
- Nombre maximal de transmission d'un message Broadcast direct : $np-1$
- Taille de la table de voisinage : $n \times p$
- Taille de l'ID : p
- Nombre des nœuds affectés par une disparition d'un parent de niveau i : $p-i-1$
- Probabilité de disparition de tous les nœuds d'un même anneau : α^n
- Trafic des données au niveau de chaque nœud

Pour un N donné, si la valeur de p est grand (i.e. n petit) la topologie ressemble plus à un arbre qu'un anneau et si la valeur de p est petit (i.e. n est grand) le comportement du réseau s'approche d'un anneau. Une topologie qui ressemble à un arbre réduit le temps de communication Broadcast mais le trafic de données au niveau des nœuds est différent d'un

niveau à un autre. Les nœuds situés au niveau tronc de la topologie (i.e. les nœuds de niveau 0) seront plus chargés que tous les autres nœuds dans un réseau plein (i.e. lorsque toutes les positions de la topologie sont occupées). Si les nœuds sont situés en anneau, la valeur de n est plus grande par rapport à p, le trafic va être plus distribué (dans le cas extrême où tous les nœuds sont placés en un seul anneau les nœuds vont avoir la même charge de trafic) mais le temps de communication sera plus long. Le CHAPITRE 4 montre un exemple numérique via une application de Smart-Grid.

La taille de la table de voisinage a un effet sur la mémoire requise au niveau d'un nœud Ring-Tree ainsi que sur la taille des messages échangés lors de la connexion, la réinsertion et la mise à jour. Le choix des valeurs n et p affecte la taille de la table de voisinage. Par exemple, pour une communauté de $N = 10^{10}$ nœuds il est possible de choisir $n = 10^5$ et $p = 2$ ou bien $n = 10$ et $p = 10$, le premier choix nécessite une table de voisinage de taille $n \times p = 2 \times 10^5$ tandis que le deuxième choix nécessite une taille de seulement 100 cases.

Pour trouver les valeurs optimales du couple (n, p) minimisant la taille de la table de voisinage, il est possible d'utiliser la Tableau IV. La valeur optimale du couple (n, p), pour une communauté de taille approximative $N = 10^{10}$, est (3, 21). Ceci donne une valeur de N légèrement supérieure à 10^{10} et une taille de la table de voisinage de 63 cases.

La Tableau IV présente également les valeurs optimales des couples (n, p) minimisant le temps de Broadcast direct. Selon cette table, pour une communauté de 1024 nœuds, il faut choisir (n=4, p=5) ou bien (n=2, p=10) et pour une communauté de 1.5 millions il faut utiliser (n=3, p=13).

Une table identique à celle obtenu pour le temps de Broadcast direct et la taille de la table de voisinage peut être calculée pour chacun des éléments de performance listés ci-haut. Une table calculée pour le temps de Broadcast circulaire, montre que la valeur optimale du paramètre n est presque tout le temps égale à 2 quelque soit N, il faut augmenter p pour avoir le nombre de N nécessaire.

Pour avoir une communauté proche d'un million, il est possible de choisir (n=2, p=20). Avec ces valeurs, le temps de Broadcast direct est de 39, une valeur identique à celle obtenue avec (n=3, p=13) permettant de créer une communauté de taille supérieure (1.5 millions) pour le même temps de Broadcast.

Tableau IV: valeurs de (n, p) minimisants le temps de diffusion d'un Broadcast direct

N	couples (n,p) possibles	Nombre de transmissions	Taille de la table de voisinage
4	(2, 2)	3	4
8	(2, 3)	5	6
9	(3, 2)	5	6
16	(4, 2); (2 ,4)	7	8
27	(3, 3)	8	9
32	(2, 5)	9	10
36	(6, 2)	11	12
64	(4, 3); (2, 6)	11	12
81	(3, 4)	11	12
128	(2, 7)	13	14
243	(3, 5)	14	15
256	(4, 4); (2, 8)	15	16
512	(2, 9)	17	18
729	(3, 6)	17	18
1024	(4, 5); (2, 10)	19	20
2187	(3, 7)	20	21
4096	(4, 6); (2, 12)	23	24
6561	(3, 8)	23	24
8192	(2, 13)	25	26
19683	(3, 9)	26	27
32768	(2, 15)	29	30
59049	(3, 10)	29	30
65536	(4, 8); (2, 16)	31	32
177147	(3, 11)	32	33
262144	(4, 9); (2, 18)	35	36
531441	(3, 12)	35	36
1594323	(3, 13)	38	39
2097152	(2, 21)	41	42
4782969	(3, 14)	41	42
14348907	(3, 15)	44	45
16777216	(4, 12); (2, 24)	47	48
43046721	(3, 16)	47	48
129140163	(3, 17)	50	51
134217728	(2, 27)	53	54
387420489	(3, 18)	53	54
1162261467	(3, 19)	56	57
3486784401	(3, 20)	59	60
10^{10}	(3, 21)	62	63

Le temps de Broadcast n'est pas le seul critère de sélection des valeurs n et p. Une petite valeur de n (une grande valeur de p, pour un N donné) réduit le temps de diffusion d'un message Broadcast mais réduit également la robustesse de la structure.

Pourquoi il faut augmenter la valeur de n par rapport à p pour un N donné?

Chaque nœud du niveau tronc est responsable d'un maximum de $^{N}/_{n}$ nœuds. Si un nœud du niveau tronc disparait, $^{1}/_{n}$ du réseau sera affectée. Plus la valeur de n est grande, plus le nombre de nœuds affectés par une disparition d'un nœud du niveau tronc est petit.

La valeur du paramètre n affecte la probabilité de disparition de tous les nœuds en même temps. Si α est la probabilité qu'un nœud quelconque disparaît involontairement et si tous les nœuds ont la même probabilité de disparaître, alors la probabilité que tous les nœuds d'un anneau donné disparaissent en même temps est α^n. Plus la valeur de n est grande, plus la probabilité que tous les nœuds d'un anneau donné disparaissent en même temps est petite.

Lorsqu'un nœud de niveau i disparait, un nombre de p-i-1 précédents deviennent des orphelins actifs. Plus la valeur de n est grande par rapport à p, plus le nombre d'orphelins actifs est petit.

Utilisation d'un n variable

Si tous les nœuds du niveau tronc disparaissent en même temps le Ring-Tree se partitionne. Il est possible d'utiliser une valeur de n variable dépendamment du niveau du nœud. Par exemple un n plus grand peut être choisi pour les nœuds du niveau tronc et un n plus petit pour les autres niveaux. Ceci augmente la robustesse de la structure face aux disparitions des nœuds du niveau tronc.

La structure Ring-tree permet d'utiliser différentes valeurs de n pour les différents niveaux de la topologie (par exemple des valeurs de n plus grandes pour les niveaux plus bas) sans affecter grandement les algorithmes et les protocoles du réseau.

Les valeurs des paramètres n et p ne sont pas nécessairement fixes tout au long de la vie de la communauté. Il est possible de concevoir des mécanismes d'expansion et de réduction du réseau. Mais, la réduction de la communauté peut amener à la rejection de quelques nœuds du réseau. Pour étendre une communauté, il suffit d'augmenter, dynamiquement, la valeur de n ou de p. À la réception des nouvelles valeurs, chaque nœud augmente la taille de sa table de voisinage. Il sera donc en mesure de recevoir plus de nœuds descendants. Si la valeur de n ou de p est diminuée durant la vie de la communauté, chaque nœud peut envoyer aux nœuds concernés un message pour qu'ils se mettent en état non connecté. Si la valeur de n est diminuée, les nœuds des colonnes à droite (n-1, n-2, ...) seront rejetés. Si la taille de p est

diminuée, les nœuds des lignes en bas (p-1, p-2, ...) seront rejetés. Ces nœuds rejetés doivent donc essayer de se reconnecter à la communauté mais une place disponible ne leur est pas garantie.

3.12 Conclusion

Ce chapitre a présenté les détails du système de communication Ring-Tree. Ce système de communication est conçu pour fonctionner d'une façon distribuée P2P au niveau « overlay ». Il vise principalement des applications de smart grid où des millions d'appareils électriques ayant des ressources limitées, échangent des messages de petite à moyenne taille via des communications « Broadcast ». Cependant, l'utilisation du Ring-Tree peut être envisagée pour d'autres types d'applications telles que la localisation des ressources au sein d'un groupe d'utilisateurs, système de stockage virtuel, service de diffusion Multicast, etc.

Les éléments clés du Ring-Tree sont la topologie, l'ID de nœuds et la table de voisinage. Ces éléments permettent différentes utilisations de la structure. Ils permettent la conception de multiples protocoles de communication pour rejoindre une partie ou tous les nœuds du réseau. Ils permettent également, la conception de différents algorithmes de maintenance de la structure.

La topologie structurée hiérarchique du Ring-Tree T(n, p) est caractérisée par deux paramètres n et p, n étant le nombre maximal de nœuds qu'un anneau quelconque peut supporter et p est le nombre de niveau maximal du réseau. La taille maximale du réseau est de n^p nœuds.

Le choix optimales des valeurs n et p dépend de l'application utilisant le Ring-Tree. Pour les applications où les nœuds sont très dynamique, il est mieux d'augmenter n et de réduire p. Car une plus grande valeur de n augmente la robustesse mais augmente le temps de diffusion des messages. Pour les applications où les nœuds ne sont pas très dynamique et la probabilité de disparition des nœuds est basse, il est mieux de réduire n et augmenter p. Pour augmenter la robustesse de la structure, il est possible de choisir une valeur de n fixe plus grande pour le niveau du tronc et une ou des valeurs plus petites pour les autres niveaux plus hauts. Une étude sur les valeurs optimales de n et p a été présentée à la section 3.11.

L'identificateur unique ID permet au nœud de connaitre sa position dans la topologie. Cet ID sert à déterminer le niveau du nœud dans la topologie. Il sert également à construire la table de

voisinage et celle des IDs. La table des IDs sert principalement à l'attribution d'un ID à un nouveau nœud et aux algorithmes de routages basés sur l'ID.

La table de voisinage, de taille limitée n × p, permet de garder de l'information sur une partie du réseau. Il permet à un nœud donné de rejoindre les autres nœuds de la communauté via des messages unicast, multicast et Broadcast et de participer au routage des messages. Cette table joue un rôle important à la maintenance de la structure (surveillance des nœuds suivants, détection des bris de communication et réparation).

Finalement, le Tableau V résume les éléments de performance principaux du Ring-Tree.

Tableau V: éléments de performances du réseau Ring-Tree

Mémoire requise	$n \times p$ pour la table de voisinage p pour chaque ID
Nombre d'envoie maximal pour la diffusion d'un message à tous les nœuds du réseau	$np - 1$ pour le Broadcast circulaire $(n-1)(2p-1)$ pour le Broadcast direct
Nombre des nœuds impliqués dans le cas de disparition d'un nœud	$p - i - 1$, avec i le niveau du nœud disparu
Probabilité de partitionnement du réseau	α^n, avec α la probabilité de disparition d'un nœud quelconque

CHAPITRE 4 MISE EN ŒUVRE DU SYSTÈME

Le CHAPITRE 2 a proposé une approche décentralisée pour le contrôle de la consommation des charges électriques. Le contrôle s'effectue à l'aide d'une consigne appelée quota. Le quota est une valeur de puissance qui guide la consommation d'une communauté des charges électriques. Lorsque la valeur de quota varie dans le temps, le profil de consommation électrique de la communauté suit cette variation. Pour suivre les valeurs de quota, les charges construisent un histogramme de leur paramètre d'état. Ce paramètre, appelé aussi paramètre de besoin, représente la « température » de l'eau chaude lorsque la charge est de type chauffe-eau. À l'aide de l'histogramme et la valeur du quota, chaque charge exécute un algorithme de contrôle local. L'algorithme de contrôle calcule une valeur seuil qui permet à la charge de décider de consommer en tenant compte la valeur et le type du quota ainsi que le confort de l'usager.

Pour construire l'histogramme, les charges s'appuient sur un réseau de communication qui leur permet d'échanger des données. Le réseau de communication leur permet, entre autres, de recevoir la valeur et le type du quota. Le réseau Ring-Tree, présenté au CHAPITRE 3, peut être utilisé pour ce type d'application de contrôle distribué. Il peut fonctionner avec des équipements ayant des ressources limitées comme des microcontrôleurs. Car, il requiert peu de mémoire et de capacité de calcul. C'est un réseau dédié « overlay », qui fonctionne sur une infrastructure de communication IP (Internet Protocol). Il permet à un nombre important de nœuds de construire l'histogramme en quelques secondes lorsque, par exemple, l'infrastructure du réseau IP utilisée est de type Internet haute vitesse.

Ce chapitre documente la mise en œuvre de l'ensemble des systèmes : le contrôle de charges électriques distribué présenté au CHAPITRE 2 et le réseau de communication Ring-Tree présenté au CHAPITRE 3. La technique de contrôle distribué peut fonctionner avec d'autres réseaux de communication que le Ring-Tree et ce dernier peut être utilisé par tout autre application qui requiert la collaboration entre un grand nombre de nœuds. Cependant, ce chapitre présente l'implémentation du Ring-Tree et son utilisation avec la technique de contrôle proposée.

La technique de contrôle est assez générique. Elle peut s'appliquer sur différents types de charges : chauffe-eau, véhicule électrique, chauffage, etc. La présente étude se limite à un seul type de charge qui est le « chauffe-eau ».

La validation de la mise en œuvre de l'ensemble des systèmes est réalisée par des simulations. Le simulateur, programmé en Java, inclut des chauffe-eaux virtuels ainsi que l'algorithme de contrôle distribué et le réseau de communication Ring-Tree. Les nœuds Ring-Tree programmés sont statiques. La gestion des ajouts et des retraits dynamiques des nœuds est détaillée au CHAPITRE 3.

La simulation effectuée sur une seule machine est limitée à quelques milliers de nœuds. Une évaluation de la performance des systèmes est étalée pour couvrir le cas où un nombre important de nœuds (millions) participerait à la communauté.

Ce chapitre utilise les termes charge, contrôleur et nœud d'une façon interchangeable. Un nœud est une entité qui a la capacité de communication et de calcul. Un contrôleur est une entité capable d'acquérir des données sur l'état de la charge (ex : température du chauffe-eau, niveau de charge d'une batterie, etc.) et de contrôler la consommation électrique de la charge, via des relais par exemple. Dépendamment de l'implémentation physique, les capacités du nœud et celles du contrôleur peuvent être intégrées à la charge (charge intelligente, « smart appliance »). C'est pourquoi le reste du chapitre ne fait pas la distinction entre ces termes. Les charges considérées ici sont intelligentes.

La section 4.1 explique comment le réseau Ring-Tree est utilisé pour l'algorithme de contrôle distribué. Le fonctionnement général des systèmes est expliqué à l'aide d'un diagramme des cas d'utilisations et des exemples des types d'applications du domaine du réseau électrique sont présentés. La section 4.2 présente les résultats de simulation ainsi que la performance des systèmes. L'évaluation de la performance est étalée pour couvrir le cas où un nombre important (millions) de nœuds participerait à la communauté. La section 4.3 étudie les limites physiques du schème de commande et présente des équations qui permettent d'automatiser le calcul du quota. Elle présente également les bornes d'erreurs associées au calcul des histogrammes. Finalement, une évaluation quantifiée du degré de complexité de l'implémentation est exposée à la section 4.4.

4.1 Fonctionnement de l'algorithme de contrôle avec le Ring-Tree

Les différents nœuds (chauffe-eaux) utilisent le réseau de communications Ring-Tree pour construire un histogramme de leurs températures. Après la construction de l'histogramme, le nœud racine qui est le parent du niveau du tronc (nœud 0) dans un réseau Ring-Tree, le diffuse à tous les autres nœuds. Cet histogramme est, ensuite, utilisé localement par l'algorithme de contrôle distribué. Chaque nœud a, donc, deux fonctionnalités principales : maintenir à jour un histogramme de températures et exécuter l'algorithme de contrôle local.

Pour maintenir à jour un histogramme, les nœuds du niveau feuille construisent des histogrammes partiels et les envoient vers le tronc. Ces histogrammes sont fusionnés au niveau des nœuds parents de la topologie Ring-Tree. Cette action s'appelle phase 1; c'est la phase de la construction de l'histogramme. Ensuite, l'histogramme construit est diffusé par le nœud parent du niveau tronc vers tous les autres nœuds. Cette deuxième action s'appelle phase 2, c'est la phase de la diffusion de l'histogramme. Lorsque les deux phases sont complétées, les nœuds se mettent en attente. Les trois étapes construction (phase 1), diffusion (phase 2) et attente définissent un cycle. La durée d'un cycle dépend, entre autres, du type de la charge. Dans la simulation, la durée d'un cycle a été fixée à 1 minute i.e. à chaque minute l'histogramme est mis à jour et il est disponible localement dans chaque nœud pour être utilisé par l'algorithme de contrôle. L'algorithme de contrôle est exécuté à la fin de chaque cycle.

L'algorithme peut, également, être exécuté selon le besoin de l'opérateur (Figure 33), à la réception d'une nouvelle valeur de quota par exemple. Dans ce cas, la décision de la charge, au pire cas, sera basée sur des informations qui sont vieilles d'une minute.

L'opérateur peut demander une copie de l'histogramme. Grâce à l'histogramme, l'opérateur peut simuler l'exécution de l'algorithme de contrôle pour prévoir le comportement de la communauté. En tout moment, l'opérateur peut envoyer une nouvelle valeur du quota ou modifier son type (obligatoire, optionnel) selon les besoins du réseau. Dans certains cas, l'utilisation de l'histogramme n'est pas nécessaire pour la décision locale. Par exemple, lorsqu'un quota obligatoire de valeur zéro est envoyé aux nœuds pour délester toutes les charges.

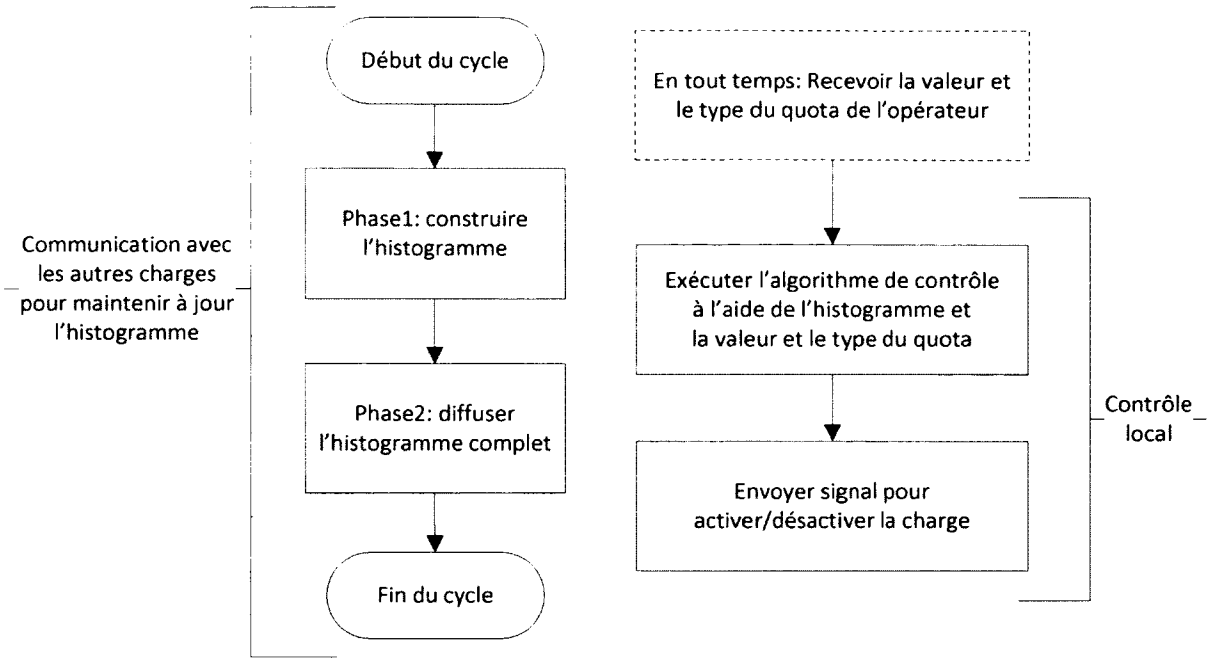


Figure 33: représentation du fonctionnement global de l'ensemble des systèmes

4.1.1 Construction et diffusion de l'histogramme

Les nœuds utilisent le réseau de communication Ring-Tree pour créer et partager une structure des données contenant un histogramme de température. Cet histogramme associe à chaque valeur de température le nombre de chauffe-eaux ayant cette valeur. En pratique, on effectue une quantification qui regroupe plusieurs valeurs de températures en une seule classe appelée aussi plage des valeurs. Pour construire l'histogramme, la structure des données circule entre tous les nœuds de la communauté et chaque nœud incrémente la valeur de la classe qui correspond à sa température. La représentation de l'histogramme et des températures de références qui définissent les plages des valeurs est schématisée à la Figure 34. En simulation, chaque plage des valeurs couvre un demi-degré Celsius $\Delta T_k = 0{,}5°C\ \forall k$.

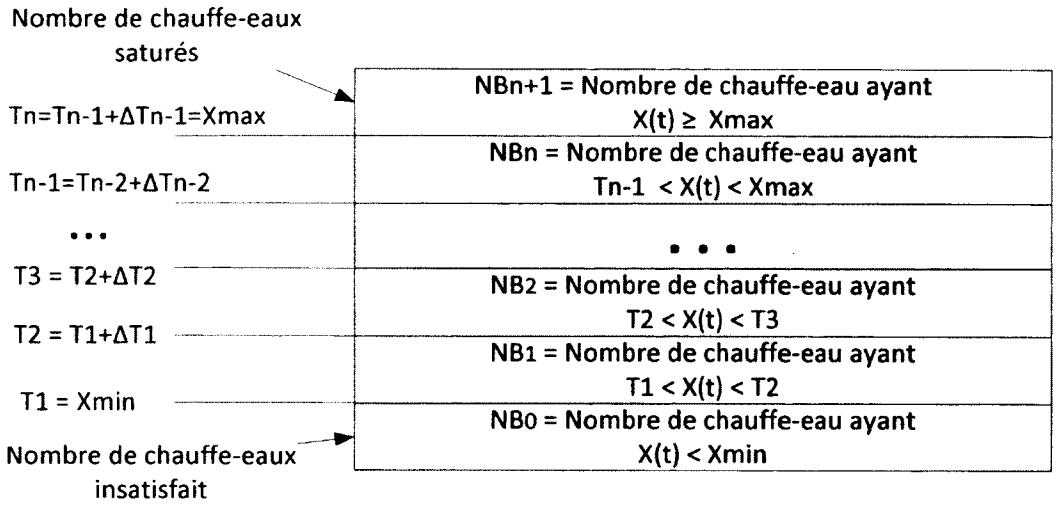


Figure 34: représentation de l'histogramme et les températures de références

La construction et la diffusion de l'histogramme se font sur deux phases : phase 1 (construction) et phase 2 (diffusion). Pendant la phase 1, les nœuds envoient des histogrammes partiels partants des nœuds du niveau feuille de la topologie vers les nœuds du niveau tronc. Ces histogrammes sont fusionnés au niveau des nœuds parents pour former des histogrammes partiels plus complets avant d'être transmis vers le tronc du réseau. Les histogrammes partiels arrivent au parent du niveau 0 (niveau du tronc de la topologie Ring-Tree) appelé nœud 0 ou nœud racine. Le nœud racine fusionne les histogrammes et ajoute sa propre information pour compléter l'histogramme. À la fin de la phase 1, le nœud racine détient un histogramme complet. La phase 2 correspond, simplement, à la diffusion de l'histogramme à l'ensemble des charges. Le nœud racine envoie un message de type Broadcast qui contient l'histogramme complet.

Deux protocoles de communications peuvent être utilisés : communication directe et communication circulaire i.e. suivant l'anneau. Les nœuds peuvent transmettre les données selon la communication directe ou circulaire dans une des deux phases ou dans les deux phases. Mais, dans ce qui suit, le même type de communication (directe ou circulaire) est utilisé dans les deux phases.

L'algorithme de la communication directe est présenté à la Figure 35 et un diagramme qui schématise les étapes des phases 1 et 2 pour une topologie simple $T(n = 3, p = 3)$, est

présenté à la Figure 36. Avec la communication directe, chaque nœud de niveau feuille (i.e. niveau $p-1$) envoie, à son parent immédiat, un histogramme partiel. Lorsqu'un parent reçoit tous les histogrammes partiels de ses enfants; il les fusionne (une simple addition); il incrémente la case qui correspond à sa température; et il transfère l'histogramme résultant à son parent immédiat et ainsi de suite jusqu'à ce que l'histogramme atteint le parent du niveau du tronc (niveau 0).

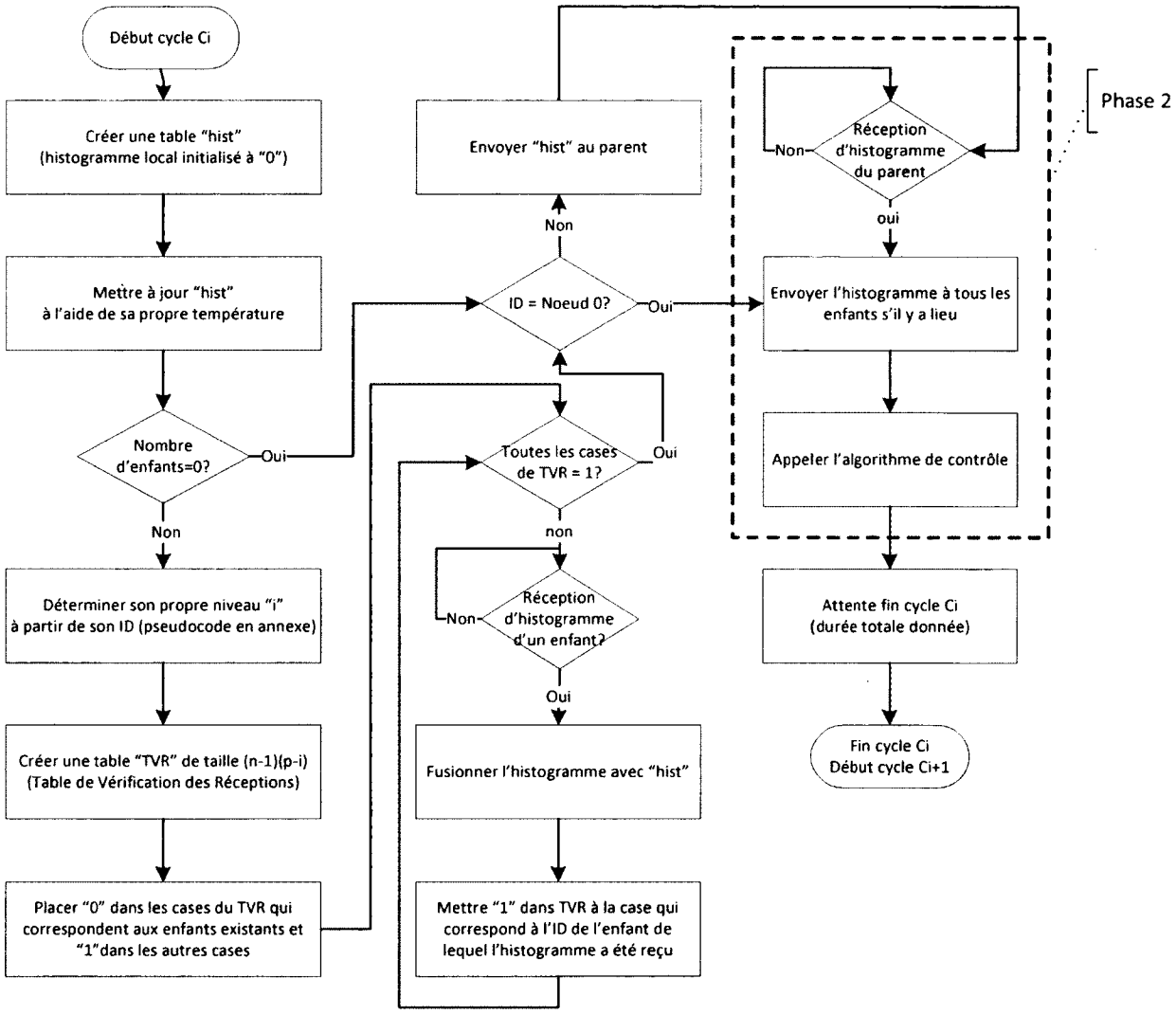


Figure 35: organigramme de l'algorithme de la communication directe

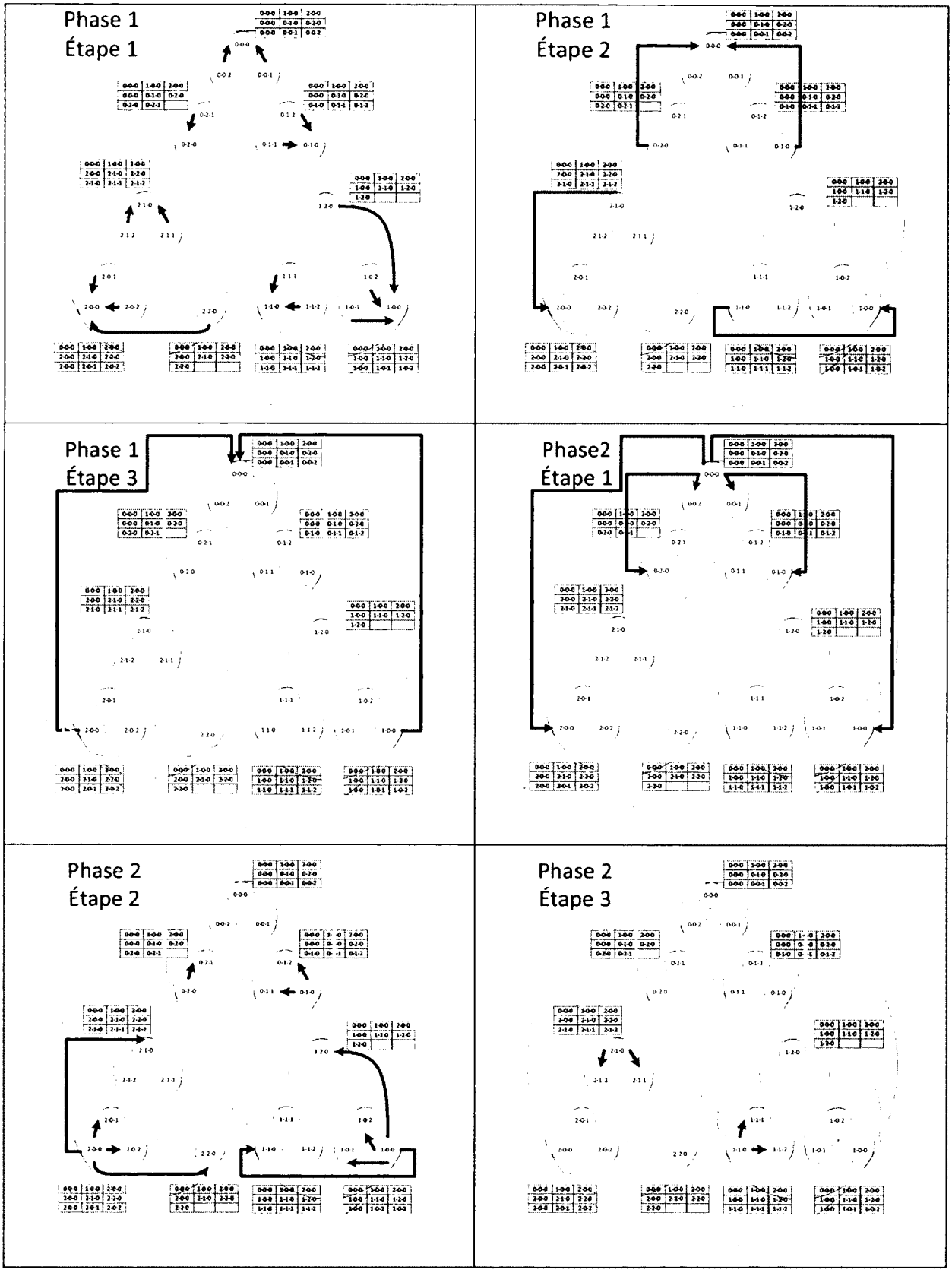


Figure 36: diagramme des étapes des deux phases 1 et 2 de la communication directe

Pour vérifier qu'il a reçu tous les histogrammes de ses enfants, un parent utilise une table de vérification des réceptions appelé TVR. La taille de cette table est égale au nombre des enfants du nœud i.e. $(n-1)(p-i)$ avec i le niveau du nœud. Chaque case de la TVR correspond à un enfant du nœud. Au début de chaque cycle, les valeurs de cette table sont initialisées à zéro. À chaque fois que le parent reçoit un histogramme d'un enfant, il met la valeur 1 dans la TVR à la case qui correspond à cet enfant. Lorsque toutes les cases sont mises à 1, le parent sait qu'il a reçu tous les histogrammes partiels de ses enfants.

Avec la communication circulaire, le nœud situé à la position « suivant » du parent débute la circulation de l'histogramme sur l'anneau. À la réception de l'histogramme, les autres nœuds de l'anneau, chacun à son tour dans un sens horaire, fusionne l'histogramme reçu de son précédent avec les histogrammes reçus de ses enfants; ajoute sa propre information à l'histogramme; transfère l'histogramme résultant à son suivant. L'histogramme atteint le parent de l'anneau. La communication circulaire de la phase 1 et celle de la phase 2, pour un exemple de topologie simple $T(n = 3,\ p = 3)$, sont schématisées aux Figure 37 et Figure 38 respectivement.

Avec la communication circulaire, la table des vérifications des réceptions (TVR) indique la réception de l'histogramme de chaque anneau. La taille de la table TVR, avec la communication circulaire, est égale à $p - i$. Rappelons que chaque nœud de niveau i est un parent de $p - i$ anneaux qui sont situés à partir du niveau $i + 1$ jusqu'au $p - 1$. La taille de la TVR de la communication circulaire est plus petite que celle de la communication directe qui indiquait la réception de l'histogramme de chaque enfant.

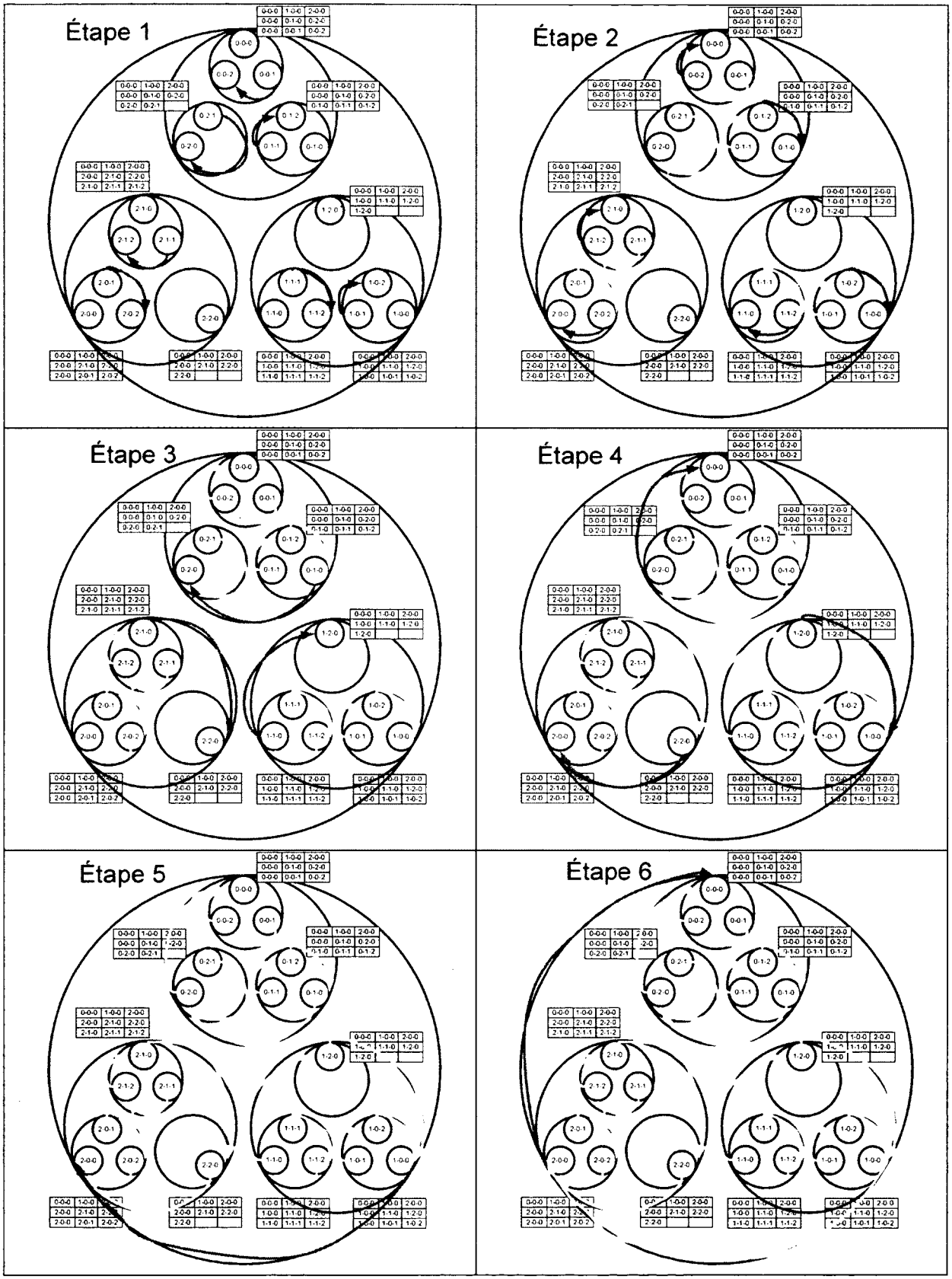


Figure 37: diagramme de la phase 1 avec la communication circulaire

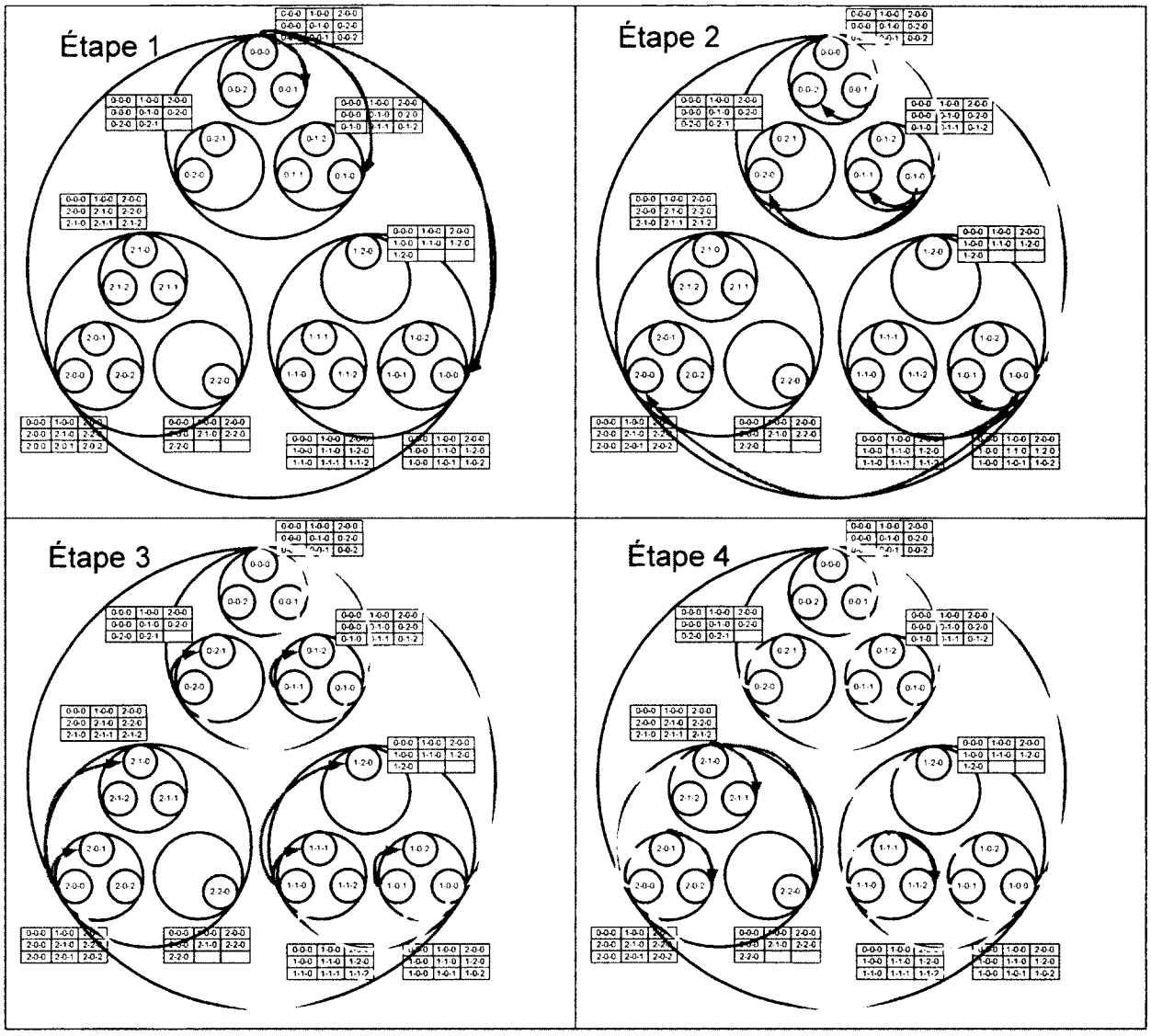


Figure 38: diagramme de la phase 2 avec la communication circulaire

Quelque soit l'approche utilisée, pour accélérer la communication, lorsqu'un nœud doit envoyer un message à d'autres nœuds qui sont situés aux niveaux différents dans la topologie, il commence toujours à l'envoyer aux nœuds situés aux niveaux les plus bas. Les nœuds de niveau plus bas ont potentiellement plus d'enfants. Ils sont alors priorisés afin de leur permettre de rejoindre leurs enfants plus rapidement et d'utiliser plus efficacement le parallélisme de la topologie. Par exemple, à l'étape 1 de la phase 2 de la communication circulaire (Figure 38), le nœud 0-0-0 envoie le message au nœud 1-0-0 avant de l'envoyer au nœud 0-1-0 et ensuite au nœud 0-1-1. Pendant que le nœud 1-0-0 envoie le message à son

enfant de niveau 1, après l'avoir reçu du nœud 0-0-0, ce dernier envoie le message à son enfant de niveau 1 également.

4.1.2 Algorithme de contrôle

L'algorithme de contrôle (Figure 39) permet à chaque nœud de prendre une décision d'activer la consommation électrique de la charge tout en respectant la valeur et le type du quota ainsi que le confort de l'usager. À l'aide de l'histogramme, l'algorithme calcule une valeur de température seuil. Si la valeur de la température seuil est plus élevée que celle de la température moyenne de l'eau dans le réservoir du chauffe-eau, l'algorithme décide d'activer la consommation.

Chaque nœud peut obtenir la valeur de la température moyenne de l'eau dans le réservoir du chauffe-eau via un ou plusieurs capteurs physiques attachés à la charge. En simulation, la température moyenne x(t) est calculée à partir d'un modèle stochastique issu de la littérature et des profils de consommation d'eau réels fournis par le LTE (IREQ).

L'histogramme contient les nombres des nœuds saturés $x(t) > x_{max}$, insatisfaits $x(t) < x_{min}$ et responsives $x_{min} < x(t) \leq x_{max}$. Les charges saturées doivent, obligatoirement, arrêter leurs consommations quelque soient la valeur et le type du quota. Les charges insatisfaites doivent consommer sauf si le quota est de type obligatoire. Les charges responsives décident de consommer tout dépendamment de la valeur de la température seuil x_{th}.

En réalité, la température seuil est une plage des valeurs limitée par deux températures de référence $\text{Top} = x_{th} - \varepsilon$ et $\text{Bottom} = x_{th} + \varepsilon$. En utilisant la valeur de quota et l'histogramme, chaque nœud exécute l'algorithme de la fonction qui calcule la valeur seuil (ANNEXE D) pour trouver cette plage des valeurs et déterminer les deux températures : Top et Bottom. Tous les nœuds qui ont une température supérieure à la température Top n'ont pas le droit de consommer, tous ceux qui ont une température plus basse que la température Bottom ont le droit de consommer et les autres nœuds décident d'une façon aléatoire s'ils ont le droit de consommer.

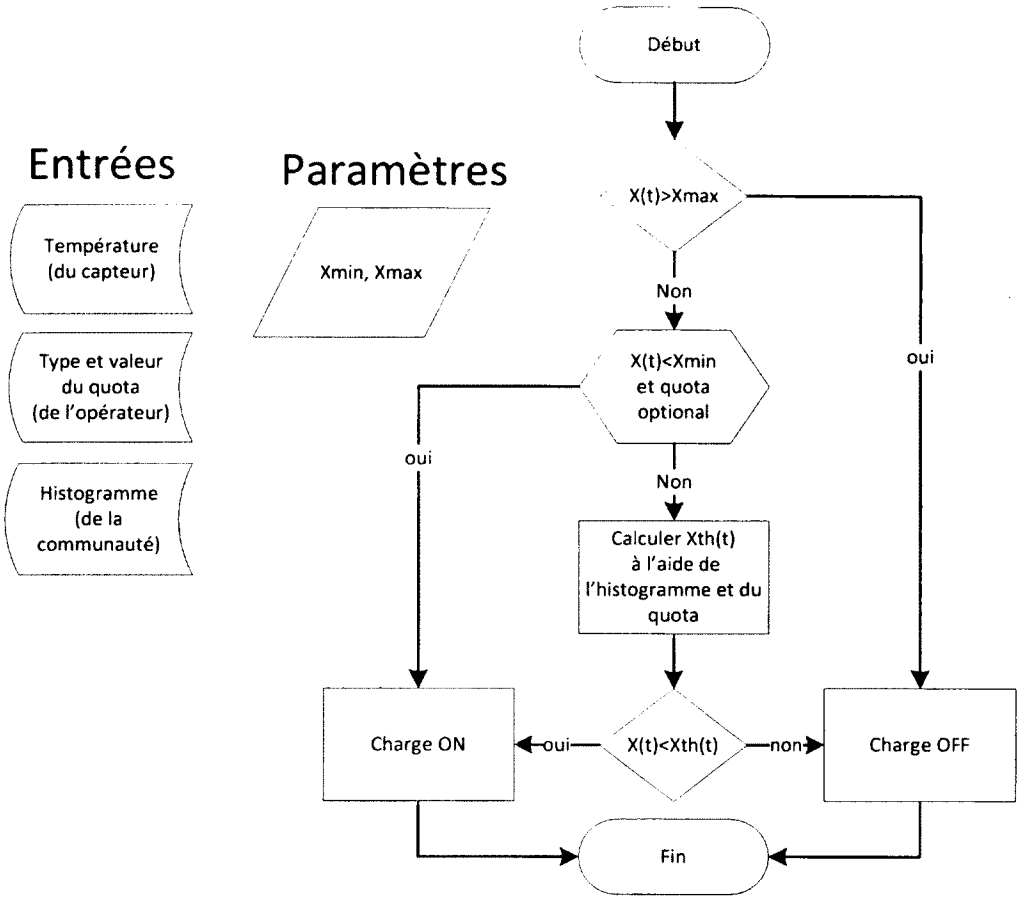


Figure 39: organigramme de l'algorithme de contrôle

4.1.3 Cas des utilisations des systèmes de contrôle

La technique de contrôle est constituée de deux systèmes : système de la communauté des charges et système de l'opérateur. La communauté des charges représente l'ensemble des charges. C'est un système distribué complètement autonome. La taille de la communauté varie mais son comportement reste le même. La communauté maintient à jour un histogramme et elle exécute l'algorithme de contrôle sans l'intervention de l'opérateur. L'opérateur est une entité qui fait le lien entre les services offert par la communauté des charges et les demandes provenantes du réseau électrique. Pour répondre aux besoins du réseau électrique, l'opérateur utilise les services offerts par la communauté via des mécanismes de contrôle. L'opérateur peut être, par exemple, le centre de conduite du réseau ou un « agrégateur » qui gère des programmes de gestion des charges.

Le lien entre la communauté et l'opérateur se fait à l'aide d'une consigne (type et valeur de quota) qui est envoyée à la communauté via un réseau de communication. L'utilisation du

réseau de communication « overlay » Ring-Tree est proposée. Ce réseau peut s'appuyer sur Internet comme infrastructure physique.

Le diagramme des cas d'utilisations, présenté à la Figure 40, montre une vue globale des systèmes. Les cas d'utilisations du système de l'opérateur et de celui de la communauté sont décrits dans le Tableau VI et le Tableau VII respectivement.

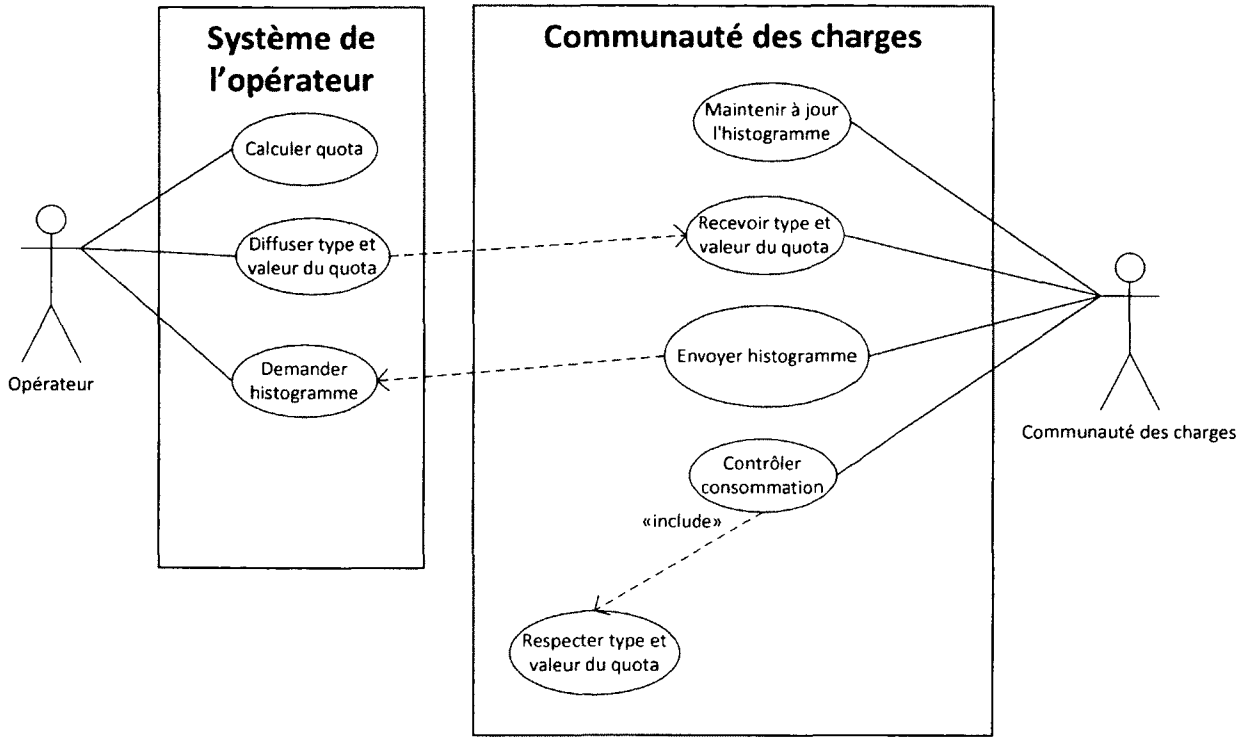


Figure 40: diagramme des cas d'utilisations

Tableau VI: système de l'opérateur

Nom du cas d'utilisation	Description
Diffuser type et valeur du quota	En tout moment, l'opérateur peut diffuser la valeur et le type du quota à l'ensemble de la communauté via le nœud racine ou tout autre nœud.
Demander histogramme	En tout moment, l'opérateur peut demander une copie de l'histogramme et des températures de références. L'histogramme montre le stockage d'énergie au sein de la

	communauté. Le stockage d'énergie peut être calculé relativement à x_{min}.
Calculer quota	Le calcul de la valeur du quota et la détermination de son type (obligatoire ou optionnel) dépend du service visé. La valeur du quota est calculée en tenant compte des prévisions sur la consommation et de la taille de la communauté. La taille de la communauté, potentiellement variable, peut être déduite de l'histogramme. Le calcul du quota doit tenir compte des contraintes formulées au CHAPITRE 2.

Tableau VII: système de la communauté des charges

Cas d'utilisations	**Description**
Maintenir à jour l'histogramme	Continuellement, des échanges des données s'effectuent selon des protocoles des communications définis qui permettent de maintenir à jour l'histogramme.
Recevoir le type et la valeur du quota	Tout nœud peut être programmé pour diffuser la valeur et le type du quota reçus de l'opérateur. Il est possible de désigner un seul nœud qui occupe une position donnée dans la topologie de traiter la demande.
Envoyer l'histogramme	Tout nœud peut être programmé pour envoyer une copie de l'histogramme suite à une demande provenant de l'opérateur. Selon le concept P2P ou M2M (machine to machine) chaque nœud est un serveur et client à la fois. Il est, également, possible de désigner un seul nœud qui occupe une position donnée dans la topologie pour traiter la demande de l'opérateur.
Contrôler la consommation	La décision concernant le contrôle de la consommation de la charge est prise par l'algorithme de contrôle local.
Respecter le type et la valeur du quota	Dépendamment du type du quota, le respect de sa valeur est assuré par l'algorithme de contrôle local.

4.1.4 Exemples d'applications

Plusieurs types d'applications peuvent utiliser cette solution. La solution permet de gérer la consommation des charges pour réduire la pointe « load shedding », pour répartir la consommation « load shifting », et pour créer des profils de consommation variés « load shaping ». Elle peut être utilisée pour des applications différentes comme les suivantes :

- Réduction de la pointe de consommation (plus que 60% de réduction de la pointe matinale du profil simulé a été atteinte sans affecter le confort de l'usager)
- Augmentation du facteur de charge (un facteur proche de 100% a été atteint avec l'utilisation d'un quota optionnel de valeur constante)
- L'équilibrage de la production des énergies fluctuantes par la création des profils variés des charges (différents profils qui respectent le quota ont être générés)
- Réduire le courant sur une ligne de transmission pour éviter une surcharge

Les charges peuvent être vues comme une réserve de puissance (négawatts) qui peut être déployée en partie ou en totalité en peu de temps. Dans certains cas d'urgences qui peuvent survenir au réseau électrique, le système de délestage doit réagir en quelques secondes (< 30 secondes). Ce type d'application peut être supporté sans coût supplémentaire.

4.2 Résultats

Le modèle de chauffe-eau programmé inclut un paramètre qui représente l'état de la consommation électrique de la charge (éteinte ou allumée). À chaque pas de simulation (1 minute), la valeur de ce paramètre est multipliée par la puissance nominale de la charge pour obtenir sa consommation en kW.

La simulation du système au complet incluant le Ring-Tree, les protocoles de communication pour maintenir à jour l'histogramme et l'algorithme de contrôle distribué donne des résultats conformes à la théorie. Plusieurs résultats ont été générés. Tous les résultats ont montré que lorsque les valeurs de quota sont choisies d'une façon à respecter les conditions formulées dans le CHAPITRE 2, la communauté des charges suit exactement cette variation. Les protocoles de communication permettent aux charges de construire et de diffuser l'histogramme en quelques sauts. Un saut correspond au passage d'un message d'un nœud à un autre. Le temps associé à chaque saut dépend de la bande passante des deux nœuds.

Le résultat de simulation d'un réseau Ring-Tree $T(n = 4, p = 5)$ pour une communauté de $N = 1024$ nœuds contrôlés à l'aide d'un quota qui varie avec le temps est présenté à la Figure 41. Ce résultat montre le profil typique de la consommation électrique d'une communauté de chauffe-eaux. Elle montre comment ce profil a été modifié selon des valeurs de quota variable (présentées en pointillés).

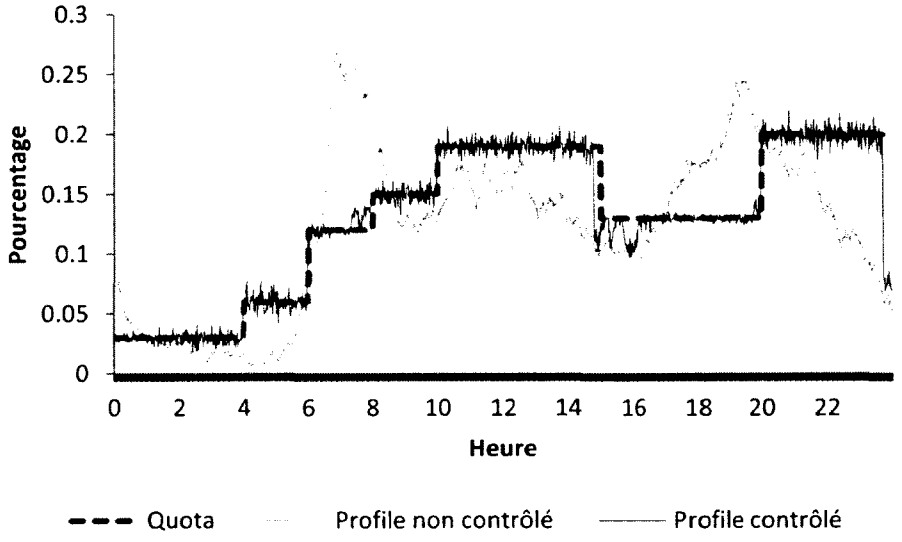


Figure 41: profile de consommation des charges contrôlées avec quota variable

Le nombre de sauts nécessaire pour compléter le cycle de construction et de diffusion de l'histogramme selon la communication circulaire en considérant le chemin le plus long est de $2(n-1)p$. Selon la théorie, le nombre des sauts nécessaires pour compléter la phase 1 est de $(n-1)p$ et celui de la phase 2 est de $(n-1)p$. La simulation de la topologie de 1024 nœuds $T(n = 4, p = 5)$ a montré que le cycle de communication se termine en 30 sauts. Cette valeur correspond bien à la valeur théorique. Pour l'exemple de la topologie présenté auparavant à la Figure 38 pour une topologie $T(n = 3, p = 3)$, le nombre des sauts de la phase 2 égal à 4 et non pas à 6 car la topologie de cet exemple ne contient pas tous les nœuds, la phase 2 s'est terminée plus tôt.

Ce protocole de communication permet, donc, à un million et demi de nœuds, placé dans une topologie $T(n = 3, p = 13)$ par exemple, de terminer la construction et la diffusion de l'histogramme en 52 sauts. Si le temps moyen nécessaire pour effectuer un saut est égal à 50

msec avec un paquet de taille de 1500 octets, la construction et la diffusion de l'histogramme peut se terminer en 2,6 secondes.

Les résultats montre également que le quota appliqué a permis de réduire la pointe de 7h00 du profile typique des chauffe-eaux de plus de 60%. Au Québec, la majorité des chauffe-eaux (2.400.000 chauffe-eaux) sont chauffés à l'électricité. Leurs contributions à la pointe de puissance (37 262MW prévus pour 2012-2013) est d'environ 5%. Si un million et demi de chauffe-eaux des clients résidentiels participent à un programme de gestion des charges, une réduction de 60% de leur pointe correspond à peu près à 700MW (ou 700 négawatts).

Selon un communiqué d'Hydro-Québec diffusé en date de 24 octobre 2008[1], des projets d'achat de 2000MW d'énergie éolienne ont été approuvés par la régie de l'énergie. Ces projets représentent des investissements de l'ordre de 5,5 milliards de dollars, dont 1,1 milliard de dollars en transport. Alors, une réduction de pointe de 700MW ne sera pas négligeable.

4.2.1 Performance du Ring-Tree

L'étude suivante permet d'expliquer les performances attendues du système. Deux éléments de performance seront examinés : le trafic des données, au niveau du réseau et au niveau du nœud le plus chargé, exprimé en nombre de paquets ainsi que la rapidité de la construction et de la diffusion de l'histogramme exprimé en nombre de sauts. Deux topologies seront utilisées comme exemples, $T(n = 3, p = 13)$ qui permet de contenir plus qu'un million et demi de nœuds et une topologie $T(n = 10, p = 6)$ qui peut contenir un million de nœuds.

D'une façon générale, un nœud Ring-Tree a un nombre limité de voisins. Le degré d'un nœud est de $(n - 1)(p - i)$ avec i le niveau du nœud. Le degré (ou valence) d'un nœud, selon la théorie des graphes, est le nombre de liens reliant ce nœud. Les nœuds les plus chargés sont les nœuds de niveau 0 ($i = 0$). Quelque soit le protocole de communication conçu pour le Ring-Tree, un nœud ne peut pas échanger des données simultanément avec plus que $(n - 1)p$ voisins. Avec les protocoles des communications basées sur la communication circulaire, le nombre des voisins avec lesquels un nœud de niveau 0 échange des données ne dépasse pas $2 \times p$.

[1] http://www.hydroquebec.com/4d_includes/la_une/PcFR2008-131.htm

Pour une topologie $T(n = 3, p = 13)$, les nœuds du niveau 0 sont limités à 26 voisins. Dans le cas de communication circulaire, un nœud ne communique simultanément qu'avec p voisins. Pour $T(n = 3, p = 13)$, ce chiffre se limite à 13 voisins pour le nœud 0. Le trafic au niveau du nœud 0 est donc limité à 13 histogrammes partiels. Chaque histogramme peut être envoyé dans un paquet IP de taille maximale à peu près égale à 1500 octets. Le volume de pointe est égal à la taille du message multipliée par la valeur du paramètre p. Cela représente un volume de pointe de données en transit égale à 19.5 ko (kilo octets) au niveau des nœuds les plus chargés. Pour un million de nœuds placés dans une topologie $T(n = 10, p = 6)$, le trafic de pointe serait de 9 ko. Car, le nœud le plus chargé ne communique simultanément qu'avec 6 voisins.

Le trafic de pointe de la topologie $T(n = 10, p = 6)$ est très réduit par rapport à celui de $T(n = 3, p = 13)$ (9 ko vs 19.5 ko). Mais, le temps de construction et de diffusion de l'histogramme est plus grand (108 sauts vs 52 sauts). L'avantage principale de la topologie $T(n = 3, p = 13)$ par rapport à $T(n = 10, p = 6)$ est le temps nécessaire pour la construction et la diffusion de l'histogramme. Avec la topologie $T(n = 3, p = 13)$ le trafic de pointe est presque doublé mais le temps de communication est à peu près divisé par deux.

Tel que mentionné au CHAPITRE 3, le choix des valeurs de n et p est important. Il est clair, selon le tableau de performance (Tableau VIII), que ces valeurs affectent le temps de construction et de diffusion de l'histogramme ainsi que le trafic de pointe au niveau des nœuds du tronc. Il faut donc choisir ces valeurs dépendamment des capacités des nœuds et selon les critères nécessaires à l'application. Si les nœuds possèdent plus de bande passante (Internet haute vitesse par exemple), il est possible de choisir une petite valeur de n pour améliorer le temps de construction et de diffusion de l'histogramme. Mais, le volume de pointe va augmenter. Car, avec une valeur de n plus petite, le trafic au niveau du tronc va être partagé avec un plus petit nombre de nœuds. Pour choisir les valeurs de n et de p, il faut faire un compromis entre la rapidité de la construction et diffusion de l'histogramme et le trafic de pointe au niveau des nœuds les plus chargés (niveau du tronc). Dans une situation réelle, les usagers peuvent avoir, selon leur abbonnement auprès de leurs fournisseurs d'accès, des capacités des bandes passantes et des limites d'échange de données différentes. Dans ce cas, il est possible de placer les nœuds ayant une plus grande capacité au niveau du tronc.

Tableau VIII: tableau de performance du Ring-Tree

Description	Formule générale	Exemple 1	Exemple 2
Topologie	$T(n,p)$	$T(3,13)$	$T(10,6)$
Nombre maximal des nœuds dans un anneau	n	3 nœuds	10 nœuds
Nombre maximal des niveaux dans la topologie	p	13 niveaux	6 niveaux
Nombre maximal des nœuds dans le réseau	$N = n^p$	1.594.323 nœuds	1.000.000 nœuds
Nombre maximal des nœuds au niveau i	$\begin{cases}(n-1)n^i & \text{si } i = 1..n-1 \\ n & \text{si } i = 0\end{cases}$	1.062.882 nœuds au niveau p-1 (niveau feuille)	900.000 nœuds au niveau p-1 (niveau feuille)
Nombre maximal des sauts nécessaire pour construire et diffuser l'histogramme selon la communication circulaire	$2(n-1)p$	52 sauts	108 sauts
Volume maximal des données échangées simultanément avec un nœud de niveau i	$2(p-i)$	26 paquets (niveau 0)	12 paquets (niveau 0)
Volume moyenne de pointe de données	$\frac{\sum_{i=1}^{i=p-1} 2(p-i)(n-1)n^i + 2pn}{N}$	$\approx$ 4 paquets	$\approx$ 3.2 paquets
Nombre maximal des nœuds impliqués dans le cas de disparition d'un parent au niveau i	$p-i-1$	12 nœuds (parent niveau 0)	5 nœuds (parent niveau 0)

Pour augmenter la rapidité de la communication sur un anneau, il est possible d'envoyer un message dans les deux sens de l'anneau (horaire et antihoraire). Pour éviter qu'un nœud de l'anneau reçoive le même message deux fois (une fois du nœud précédent et une fois du nœud suivant), il est possible d'ajouter un nombre maximal des sauts à effectuer dans chaque sens. Chaque nœud qui reçoit le message décrémente la valeur du nombre maximal des sauts avant de le transférer à son voisin (précédent ou suivant dépendamment du sens de l'envoi). Lorsque le nombre maximal des sauts atteint zéro le transfert s'arrête. Ce mécanisme

ressemble au TTL (Time To Live) déjà utilisé dans les réseaux informatiques. Le nombre maximal des sauts peut être déterminé selon le temps nécessaire pour que le message atteigne le nœud milieu par rapport au nœud source, d'une façon à ce que le message prend le même temps dans chaque sens. Ce temps dépend de la bande passante des nœuds de l'anneau. Pour s'assurer que le message peut atteindre tous les nœuds de l'anneau, la somme de nombre maximal des sauts dans le sens horaire avec celui du sens antihoraire doit être égale au nombre des sauts total dans l'anneau. Le nombre maximal des sauts dépend du nombre des nœuds qui existent dans l'anneau.

Il faut tenir compte du trafic des données nécessaires à la gestion du réseau et au maintien de la structure. Ce trafic est, principalement, généré dans les cas des disparitions ou des ajouts des nœuds. La dynamique des nœuds dans la communauté est très basse pour les types d'applications visés. Les bris de communication sont détectés automatiquement et un algorithme de recouvrement s'exécute pour réparer la structure. Le nombre des nœuds impliqués dans le cas de disparition d'un parent est de $p - i - 1$, avec i le niveau du nœud disparu. Les nœuds du niveau du tronc représentent le pire cas. Dans ce cas, $p - 1$ nœuds seront impliqués. Si un nœud de niveau 0 disparaît, pour une communauté $T(n = 3, p = 13)$, seulement 12 nœuds, au maximum, seront impliqués dans la réparation de la structure et avec $T(n = 10, p = 6)$, 5 nœuds seront impliqués. Généralement, le taux de disparition accidentelle des nœuds est très bas. Le trafic généré, pour une large communauté, reste raisonnable.

Pour réduire le trafic de gestion du réseau, des techniques existantes peuvent être utilisées comme le « piggy-back » qui permet d'ajouter les messages de gestion aux segments de données envoyés par l'application.

4.3 Limites physiques du schème de commande

Le choix des valeurs du quota doit tenir compte des limitations physiques des charges. Car il n'est pas possible de forcer une batterie, par exemple, de charger au-delà de sa capacité de stockage x_{max}, ni de forcer un chauffe-eau à augmenter sa température à une valeur non sécuritaire. Les valeurs du quota dépendent également du besoin de cosommation de l'usager i.e. de l'énergie nécessaire à son confort x_{min} ainsi que de l'énergie stockée dans la charge

par rapport à x_{min}. Les variations maximales du quota $\max\left|\frac{dq}{dt}\right|$ dépendent également du type du quota.

Si le quota est de type obligatoire, la variation maximale du quota égale à la puissance associée à toutes les charges par unité de temps $\max\left|\frac{dq}{dt}\right| = \frac{\sum_{k=1}^{n} p_k}{dt}$ avec p_k la puissance nominale de la charge k. Dès qu'une charge reçoive un quota de type obligatoire fixé à zéro, comme dans le cas de délestage, elle arrête sa consommation. Pour une longue durée de délestage, toutes les charges deviennent affamées. Dans ce cas, si un quota de valeur maximale est envoyé, toutes les charges vont consommer en même temps. Il n'y a, donc, pas de limite sur la variation des valeurs de ce type du quota. Le temps de réponse de la communauté d'un tel cas est très rapide. Car, il dépend seulement de la bande passante du réseau informatique pour recevoir la consigne et de la vitesse de commutation au niveau du matériel contrôlé.

Si le quota est de type optionnel, les limitations de ses valeurs sont représentées par deux équations. L'équation (25) représente la valeur minimale du quota et l'équation (26) représente la valeur maximale. Toute valeur de quota inférieure à l'équation (25) ne sera pas respectée, car l'algorithme de contrôle va prioriser le confort x_{min}. Toute valeur supérieure à l'équation (26) n'est pas possible d'être atteint à cause de la capacité de stockage maximale permise x_{max}.

L'équation (25) est la suivante :

$$\boldsymbol{quota} = \frac{E_{demandée} + E_{perte} - E_{stockée}}{dt} \qquad (25)$$

avec dt la durée du cycle de contrôle (1 minute), $E_{demandée}$ l'énergie requise par la charge durant dt due à l'utilisation de l'usager, E_{perte} l'énergie associée à la perte et $E_{stockée}$ l'énergie stockée dans la charge au début du cycle.

La valeur du quota est constante durant dt. L'énergie requise et celle des pertes peuvent être estimées à partir des prévisions à court terme. L'énergie stockée peut être calculée à partir de l'histogramme par rappot à x_{min}. Son équation dépend du type de la charge. Pour un chauffe-eau, $E_{stockée} = mc(x(t) - x_{min})$ avec m la masse de l'eau dans le réservoir, c la capacité thermique massique de l'eau liquide, x(t) la température moyenne de l'eau chaude dans le réservoir.

Si la demande et la perte sont inférieures à l'énergie stockée, l'équation (25) montre qu'il est possible de tirer de l'énergie de la charge, car la valeur du quota peut être négative. Mais, ceci n'est possible que si la charge est équipée par une telle capacité (par exemple V2G vehicle-to-grid).

L'équation (26) est la suivante :

$$quota = \frac{E_{demandée} + E_{perte} - E_{stockée} + E_{\grave{A}Stocker}}{dt} \quad (26)$$

avec $E_{\grave{A}Stocker}$ l'énergie qui doit être stockée au niveau des charges durant dt. Cette énergie doit respecter la capacité maximale de stockage x_{max}. Sa valeur maximale pour un chauffe-eau égale $E_{\grave{A}Stocker} \leq E_{max} = mc(x_{max} - x_{min})$.

Le temps de réponse de la communauté à une variation du quota est limité par le temps de la construction et de la diffusion de l'histogramme (phase 1 et phase 2). Pour un nombre important de nœuds les deux phases se terminent en quelques secondes. Cela respecte la majorité des applications du réseau électrique.

Quant aux bornes d'erreurs associées au calcul des histogrammes, s'il n'y a pas des disparitions accidentelles des nœuds, ces erreurs sont presque nulles. Il y a une légère approximation de calcul, qui n'affecte pas la performance, due au tirage au hasard parmi les nœuds qui sont situés à la même plage des températures. Dans ce cas, un nœud peut être placé au quota à la place d'un autre. Ces deux nœuds ont, à peu près, la même température (dans la simulation $\max(|x_1 - x_2|) = 0{,}5°C$ avec x_1 et x_2 les températures du premier et deuxième chauffe-eau respectivement). L'impact au confort de l'utilisateur est très minime. Si la consommation de l'eau chaude des deux charges est statistiquement identique, la température de celle qui a été placée à l'extérieur du quota va baisser plus rapidement. Au prochain cycle, cette charge va avoir plus de priorité à consommer grâce à l'algorithme de contrôle. Ce qui assure l'équité dans l'accès aux ressources entre les charges.

Si, pendant le processus, les nœuds disparaissent accidentellement, il y aura un effet possible sur l'exactitude de la commande. Mais, comme le temps du cycle de construction et de diffusion de l'histogramme est de quelques secondes (inférieur à 30 secondes même avec des délais importants), il faut que le nombre des nœuds qui disparaissent et apparaissent soit important pour avoir un effet perceptible. Lorsque les nœuds sont dynamiques, les bornes d'erreurs associées au calcul local dépendent du nombre et des positions des nœuds qui disparaissent accidentellement.

4.4 Évaluation quantifiée du degré de complexité de l'implémentation

L'algorithme de contrôle distribué et le système de communication Ring-Tree sont des solutions informatiques qui peuvent être implémentés sur des machines à ressources limitées. Ces machines doivent fonctionner en continu pendant des nombreuses années. Ces caractéristiques sont communes à la majorité des systèmes embarqués déjà déployés pour différentes types d'applications.

Les possibilités d'implémentations matérielles sont très vastes. Une implémentation possible contiendrait les éléments suivants:

- Capteurs pour mesurer le paramètre de besoin (ex : capteurs des températures)
- Relais pour la commutation de la liaison électrique de l'équipement contrôlé
- Microcontrôleur ayant les capacités suivantes :
    - Mémoire de masse pour loger l'algorithme de contrôle distribué et la couche des protocoles du Ring-Tree
    - Interface de communication de type Ethernet ou Wi-Fi pour permettre la communication, via le réseau informatique, avec les autres nœuds
    - Interface d'entrée pour permettre la lecture des données qui représente le paramètre de besoin
    - Interface de sortie pour envoyer la commande de contrôle au relais

Sur le marché, il existe des mémoires de masse de type flash assez puissantes pour loger l'algorithme de contrôle et la couche des protocoles du Ring-Tree ainsi qu'un système d'exploitation temps réel incluant la pile de protocole TCP/IP. Il est possible d'utiliser un ordinateur à carte unique (SBC « Single-Board Computer »), existant au marché, ayant des interfaces d'entrées pour lire les valeurs du paramètre de besoin (température), et de sorties pour contrôler un relais connecté à l'alimentation de la charge ainsi qu'une interface de communication (Ethernet ou Wi-Fi) pour la communication via le réseau. La Figure 42 montre les composantes logicielles et matérielles requises d'une telle implémentation.

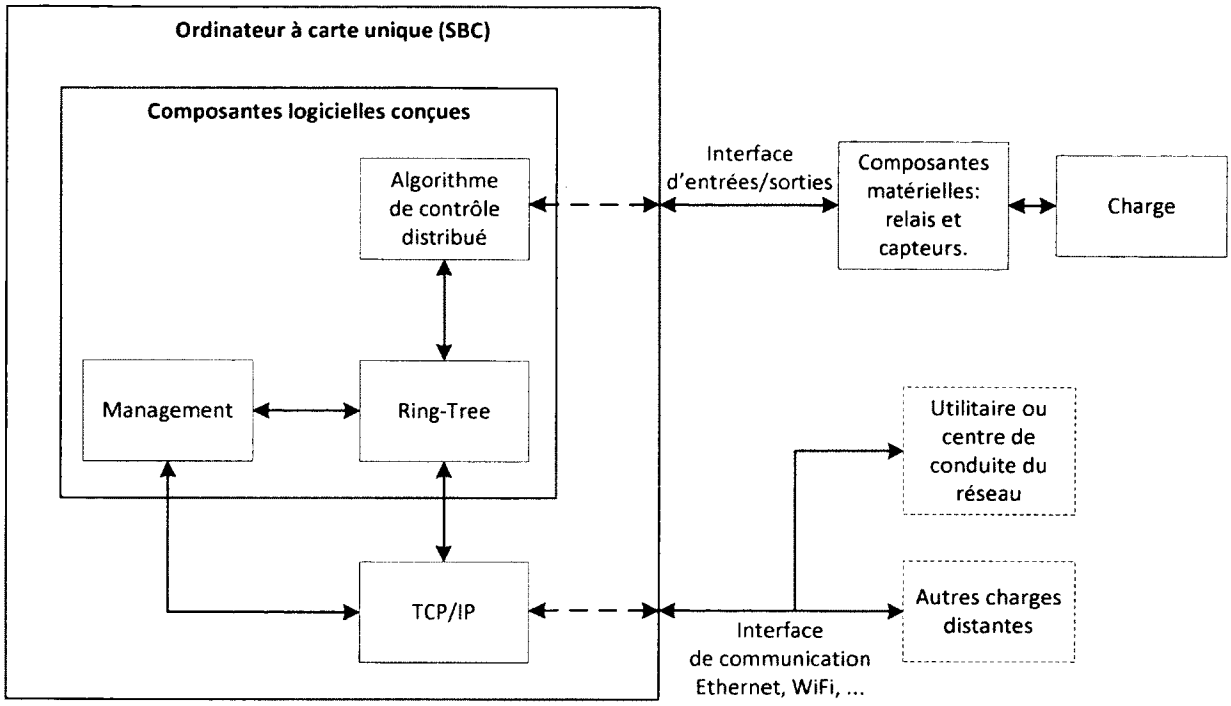


Figure 42: diagramme d'une implémentation possible de la solution à l'aide d'un SBC

Le coût du microcontrôleur, qui possède les capacités requises incluant la capacité de communication réseau (Ethernet ou Wi-Fi), est inférieur à 30$. Les prix des capteurs sont abordables. Mais, le relais (approuvé CSA), le boîtier et l'intégration de ces éléments à la charge peuvent coûter plus chère. Dans le cas où ces éléments sont intégrés à la fabrication de l'équipement (équipement intelligent « smart appliance »), le coût peut devenir presque nul. Les machines électroniques sont de moins en moins coûteuses et de plus en plus puissantes.

Un accès à un réseau IP, comme l'Internet, est requise pour le fonctionnement du système. En 2010, 8 ménages canadiens sur 10 (73 % au Québec) avaient accès à Internet dont 96 % de type haute vitesse[2]. La bande passante offerte par ce type de connexion est amplement suffisante pour l'implémentation de la solution.

[2] http://www.statcan.gc.ca/tables-tableaux/sum-som/l02/cst01/comm30b-fra.htm

4.5 Conclusion

Ce chapitre a montré l'intégration de l'algorithme de contrôle distribué au réseau de communication Ring-Tree. Il a expliqué comment les différentes charges (chauffe-eaux) utilisent le réseau Ring-Tree pour construire et diffuser leur histogramme des paramètres de besoin (températures). Les phases 1 et 2 permettent aux charges de maintenir à jour leur histogramme. Le chapitre a expliqué, à l'aide des différents diagrammes, l'ordre d'envoi des messages ainsi que le moment d'exécution de l'algorithme de contrôle. L'algorithme de contrôle peut être exécuté à la fin de cycle de construction et de diffusion de l'histogramme (à chaque 1 minute) ou selon le besoin de l'opérateur. L'algorithme de contrôle a été présenté à l'aide d'un organigramme. La fonction qui calcule la valeur seuil à partir de la valeur du quota et de l'histogramme a été expliquée.

Le fonctionnement de l'algorithme distribué avec le Ring-Tree a été présenté à l'aide d'un exemple simple d'un réseau $T(n = 3, p = 3)$ de taille limitée à 27 nœuds. Les résultats ont été générés à partir des simulations d'une communauté $T(n = 4, p = 5)$ de taille 1024 nœuds. Ces résultats ont confirmé qu'il est possible de créer des profils de consommation différents si on change la valeur du quota dans le temps.

Le réseau Ring-Tree présente beaucoup d'avantages pour les applications de « Smart Grid ». Il permet de respecter les contraintes temporelles requises pour la majorité des applications du réseau électrique. Les possibilités d'utilisation du Ring-Tree sont vastes. Chaque utilisation peut générer un trafic différent et nécessiter une capacité de bande passante différente. Le choix des paramètres n et p peut affecter la performance. D'une façon générale, pour une communauté de taille constante, le choix d'une valeur plus grande de n permet de mieux répartir le volume du trafic aux différents nœuds. Mais, ce choix augmente le temps de communication. Si les nœuds possèdent une plus grande bande passante, il est possible d'augmenter p pour réduire le temps de communication mais le trafic va être moins réparti entre les nœuds. Les nœuds de bas niveaux vont être plus chargés. Pour les applications visées, le trafic reste raisonnable même avec un n=3 et un p très grand. Une évaluation de la performance peut être effectuée pour étudier l'impact potentiel sur l'exactitude de la commande lorsqu'un grand nombre de nœuds disparait en même temps. Cette situation est peu probable pour ce type d'applications.

Le chapitre a, également, détaillé l'approche de contrôle décentralisée proposée dans cette thèse. Cette approche est composée de deux éléments principaux : opérateur et communauté des charges. L'opérateur fait le lien entre le besoin du réseau électrique et la communauté des charges à l'aide de la consigne quota. Les équations nécessaires au calcul du quota ont été présentées. Ces équations permettent de calculer les variations maximales du quota. Elles tiennent compte de la capacité maximale de stockage x_{max}, du limite de confort de l'utilisateur x_{min}, du besoin d'énergie de l'usager ainsi que de l'énergie stockée dans la charge par rapport à x_{min}. Le stockage d'énergie dans les charges ainsi que la prévision de la consommation d'électricité peuvent être estimés à l'aide de l'histogramme. Le stockage d'énergie peut être calculé par rapport à la température de référence x_{min}. L'automatisation du calcul du quota peut être programmée selon les services à offrir au réseau électrique i.e. le profil de consommation électrique visé.

Finalement, une quantification des spécifications matérielles nécessaire pour chacun des contrôleurs a été effectuée. L'intégration de la solution aux équipements déjà déployés chez les clients peut être coûteuse. Si le contrôleur est intégré à la fabrication de l'équipement « Smart Appliance » le coût matériel de la solution devient négligeable.

Les solutions présentées peuvent être utilisées pour offrir différents services au réseau électrique comme les services d'urgences, les services économiques, les services auxiliaires et les services d'adaptation aux fluctuations de la production des énergies renouvelables (éolien, solaire, ...). L'utilisation des techniques de gestion des charges pour équilibrer les fluctuations de la production des énergies renouvelables est encore à ses débuts.

L'approche décentralisée et les solutions distribuées conçues et développées dans cette thèse ouvrent la voie au déploiement des solutions sophistiquées de gestion des charges. Un projet basé sur les solutions présentées dans cette thèse est, présentement, en cours de développement chez CanmetÉNERGIE (Ressources Naturelles Canada) à Varennes (ANNEXE E). L'auteur a aidé l'équipe à intégrer le réseau Ring-Tree ainsi que l'algorithme de contrôle distribué à leur plateforme de développements. Plusieurs études reliées au projet se développent également.

CHAPITRE 5 CONCLUSION

Lors de cette thèse, deux systèmes distincts ont été conçu : un système distribué de gestion de l'appel de puissance des clients sur le réseau électrique (présenté au CHAPITRE 2) et un système de communication P2P appelé Ring-Tree permettant de supporter un grand nombre de nœuds (présenté au CHAPITRE 3). Le CHAPITRE 4 a présenté la mise en euvre de l'ensemble des systèmes. Les sections 5.1 et 5.2 rappelle les concepts les plus importants à retenir pour ces deux systèmes. La section 5.3 énonce les contributions scientifiques à la recherche. La section 5.4 présente les perspectives de recherches offertes par les contributions.

5.1 Gestion de la charge distribuée

La gestion de la charge distribuée, présentée au CHAPITRE 2, se base sur le concept d'un quota de consommation en puissance combiné avec du stockage d'énergie. Des équipements électriques chez les clients, comme des chauffe-eaux intelligents i.e. capable de calculer et de communiquer, collaborent entre eux pour respecter un quota de consommation déterminé selon l'état de charge sur le réseau électrique. L'utilisation d'un quota optionnel permet la réduction de la pointe de consommation tout en priorisant le confort de l'utilisateur. L'utilisation d'un quota obligatoire, priorise la sécurité du réseau sur le confort du client. Le quota obligatoire est utilisé seulement dans des cas d'urgence, par exemple, pour éviter une interruption d'électricité sur le réseau. L'utilisation d'un quota variable dans le temps permet de modifier le profil de consommation typique d'une communauté de charges. Lorsque la valeur de quota optionnelle varie dans le temps, en respectant les 3 conditions énoncées au CHAPITRE 2, n'importe quel profil de consommation peut être crée.

Pour mieux contrôler la reprise de charge sur le réseau électrique, la gestion distribuée est appliquée en continue et non pas seulement pendant les heures de pointe. Il est donc possible d'ajuster le profil de consommation, en utilisant un quota variable, de façon à équilibrer une production d'énergie fluctuante ou pour participer à une activité de type DR.

Pour assurer une meilleure protection de la vie privée des utilisateurs, la gestion de la charge proposée est distribuée i.e. la décision de contrôle est prise par l'équipement lui-même et non pas par la compagnie d'électricité ou une entité externe au domicile. Les informations sur la

consommation ne sont pas partagées avec un centre de contrôle. Une approche distribuée serait mieux acceptée par les consommateurs qu'une approche centralisée où un centre de contrôle qui gère les équipements chez les clients.

Le système de gestion de la charge nécessite la communication entre un grand nombre d'équipements électriques. Un défi important à relever sans augmenter le coût de l'infrastructure et de la maintenance.

5.2 Réseau de communication Ring-Tree

Une structure de réseau P2P a été conçue, opérant au dessus de la couche transport d'un réseau interconnecté conventionnel (TCP/IP et Internet). La particularité de cette architecture est de pouvoir supporter un nombre important de nœuds, jusqu'à plusieurs millions, en utilisant des ressources limitées sur chaque nœud. L'architecture est supportée par des algorithmes en permettant la création, l'exploitation et la maintenance. Parmi les algorithmes proposés on retrouve :

- L'ajout d'un nouveau nœud au réseau
- L'envoi d'un message vers un nœud spécifique dans le réseau (unicast)
- La diffusion d'un message à tous les nœuds du réseau (broadcast)
- La diffusion d'un message à un groupe de nœuds du réseau (multicast)
- Le parcours du réseau par un message contenant une liste ordonnée mise à jour sur l'ensemble des nœuds du réseau (Histogramme des valeurs d'une variable locale à chaque nœud mais dont la valeur est partagée)
- La reconfiguration du réseau suite au retrait non planifié d'un nœud (incluant l'agrégation d'ilots créés suite à ce retrait) pour en maintenir l'intégrité
- La création initiale du réseau

Le système Ring-Tree est suffisamment simple pour être mis en œuvre sur des puces (contrôleurs) associées aux dispositifs tel que des chauffe-eaux, chauffage à accumulation, bornes de recharges électriques, etc. Il n'utilise pas un serveur centralisé (ou très peu, seulement pour rejoindre le réseau). Il offre une solution réellement distribuée qui peut être mise en œuvre sans déploiement d'une infrastructure autre que les contrôleurs sur les

dispositifs visés. Finalement, le temps de réponse, estimé, de quelques secondes pour l'ensemble du réseau est en accord avec les besoins des applications visées.

Le système Ring-Tree se distingue par le choix de sa topologie qui n'a jamais été exploitée ou définie auparavant. Les topologies en anneau et en arbre se retrouvent dans des structures P2P existantes mais aucune structure combinant ces deux topologies ne se retrouve dans la littérature. Cette combinaison permet de tirer profit des avantages des deux topologies : la hiérarchisation multi-niveaux possible avec les arbres et la consolidation possible de l'information circulant sur un anneau.

Le Ring-Tree se distingue aussi par le choix de l'identificateur qui correspond à la position logique des nœuds dans le Ring-Tree. La topologie Ring-Tree permet de définir une méthode d'adressage logique associant un identifiant unique pour chaque nœud lié à sa position dans la structure. Cet identificateur est utilisé dans les différents algorithmes tels que le routage des messages et la création et maintenance du réseau.

La topologie Ring-Tree permet la mise en place d'algorithmes pour l'opération et la maintenance du réseau P2P où le nombre de voisins d'un nœud donné (le nombre de nœuds qui doivent être connus par ce nœud pour mettre en œuvre ces algorithmes) est limité à une table de taille nxp pour un réseau avec n^p nœuds au total. Cette caractéristique rend la topologie Ring-Tree très pertinente pour des réseaux avec un grand nombre de nœuds.

Le système est robuste vis-à-vis de la disparition non planifiée de nœuds. Un mécanisme de surveillance des voisins permet de détecter, efficacement, les bris de communication. Lorsqu'un bris de communication est détecté, un algorithme de recouvrement permettant la réparation de la structure ainsi que la réinsertion des nœuds orphelins est exécuté. Cet algorithme peut être centralisé ou distribué. Le recouvrement distribué est effectué complètement par les nœuds de la communauté. Le recouvrement centralisé utilise un serveur de support. Le serveur de support ne participe pas à la communication; il s'occupe d'associer une nouvelle position au nœud déconnecté lorsqu'un bris de communication est rapporté.

La topologie Ring-Tree est flexible et configurable facilement à l'aide des paramètres n et p. Le paramètre n définit le nombre de nœuds maximal possible dans un anneau et le paramètre p définit le nombre de niveaux maximal possible dans le réseau. Il est possible, pour une

communauté donné, d'utiliser des valeurs de n variables selon le niveau dans la topologie. Une valeur de n plus élevé pour l'anneau du tronc augmente la robustesse de la structure. Les valeurs des paramètres n et p peuvent être modifiées durant la vie de la communauté.
Des protocoles de communication et de routage unicast, multicast, multicast de niveau, Broadcast direct et Broadcast circulaire ont été présentés. Grâce à la structure Ring-Tree et à la table de voisinage, d'autres algorithmes de communication peuvent être conçu facilement.

5.3 Contributions

La propriété intellectuelle de la technologie Ring-Tree et de celle du système de gestion de la charge sont protégées par une demande de brevet déposée le 8 juin 2012 par SOCPRA[3] au nom de Simon Ayoub et de professeur Philippe Mabilleau. Le document de la demande de brevet est présenté en ANNEXE A.

La description de l'algorithme de gestion de la charge ainsi que la formulation mathématique d'une valeur seuil de décision et de celle du quota de consommation (variable dans le temps, optionnel, obligatoire) ainsi que la description de l'algorithme distribué ont été soumis pour être publier dans le journal « IEEE Transactions on Smart Grid » présenté au CHAPITRE 2.

Pour valider le fonctionnement de l'ensemble des systèmes (technique degestion de la charge et système de communication Ring-Tree), l'auteur a développé un simulateur écrit en Java.

L'auteur a également assisté l'équipe de CanmetÉNERGIE à Varennes à développer un simulateur, écrit en C++, qui inclut le système de communication Ring-Tree, des protocoles de communications pour la construction et la diffusion de l'histogramme, l'algorithme de contrôle et le modèle de chauffe-eau comme exemple de charge entre autres.

5.4 Développements futurs

La micro-production d'électricité va permettre aux clients de satisfaire à une partie ou à la totalité de leur besoin en énergie. Leur profil de consommation électrique, perçu du réseau, sera donc différent, surtout lorsque leur production d'électricité est basé sur l'énergie

[3] http://www.socpra.com

éolienne ou solaire qui sont des sources fluctuantes. De plus, l'arrivée des voitures électriques rechargeables risque de modifier davantage le profil typique de consommation, normalement bien connu pour un réseau électrique donné.

Des mécanismes de gestion de la charge plus évolués souvent combinés avec du stockage d'énergie vont donc devenir nécessaires pour améliorer l'équilibrage entre la production et la consommation. Par exemple, s'il n'y a pas assez de production, des mécanismes automatiques de réduction de la consommation chez les clients sont employés selon des priorités et des préférences préétablies; s'il y a trop de production, des solutions de stockage sous forme thermique, électriques ou autres peuvent être utilisés. L'utilisation de l'algorithme de gestion de la charge avec un quota variable, combiné avec du stockage d'énergie, permet de créer n'importe quel profil de consommation, dans certaines limitations physiques des équipements, sans affecter le confort de l'utilisateur.

L'introduction sur le marché des équipements électriques intelligents munis d'un microcontrôleur et d'une interface de communication permet plusieurs solutions de gestion de l'énergie chez les clients. Mais, la communication entre des millions d'équipements intelligents reste un défi technique. Le réseau Ring-Tree, s'appuyant sur l'Internet existant et ne nécessite donc pas d'infrastructure supplémentaire de communication, relève ce défi et ouvre la voie aux solutions distribuées. La participation des appareils domestiques (chauffe-eaux, unités de chauffage, etc.) au smart grid pour équilibrer la demande et la production sans contrôle centralisé devient alors possible.

ANNEXE A – DEMANDE DE BREVET

A DISTRIBUTED LOAD MANAGEMENT METHOD FOR CONTROLLING LOADS IN A COMMUNITY AND AN OVERLAY NETWORK TOPOLOGY SYSTEM FOR NETWORK NODE DEVICE COMMUNICATION

Technical Field

The present relates networked devices and communication between networked devices. The present also relates to a distributed load management method for controlling loads distributed in a community. More particularly the present relates to a distributed load management method for controlling the loads in order to reduce demand peaks in electricity by spreading out power consumption over time. The present further relates to an overlay network topology system and to a network node device for communicating with other node devices of a same community within the overlay network topology system.

Background

Demand for energy is growing continuously and the actual trend is to shift some energy consumption to electricity. Within the next few years, plug-in electric vehicles will be on the market in growing numbers and without load management, the only alternative for meeting the growing demand is to build more electric power plants and reinforce the power-grid. Renewable and decentralized generations will add more challenges for load management.

Load management aims to balance the supply of electricity on the grid with the electrical load. This can be achieved by real time direct control of some appliances by the utility or by using variable tariffs to influence consumer behaviour. There are many techniques for load management, some of them acting on water heaters. Most of them use direct control where user appliances are controlled by the electric utility and control is applied to groups of users on a given schedule. So only a limited number of water heaters consume electricity at the same time during the peak period. In such techniques and without an appropriate controller, user comfort can be affected if a water heater's temperature drops below the comfort level. There is no feedback from the water heater to the electric utility in order to control each water heater separately and the water heater has no control to do it on its own.

Another known concept is the Smart Grid which is a form of network that delivers electricity from suppliers to consumers using two-way digital communications to control appliances at

consumers' homes for saving energy, reduce costs and increase reliability and transparency if the risks inherent in executing massive information technology projects are avoided. The Smart Grid is envisioned to overlay the ordinary electrical grid with an information and net metering system that includes smart meters.

The idea of communications from suppliers to consumers to control appliances is not new, and systems have been implemented using analog technology for many years. The growth of an extensive digital communication network for the internet has made it practical to consider a more sophisticated type of electric Grid. The increased data transmission capacity has made it conceptually possible to apply sensing, measurement and control devices with two-way communications to electricity production, transmission, distribution and consumption parts of the power grid at a more granular level than previously. These devices could communicate information about grid condition to system users, operators and automated devices, making it possible for the average consumer to dynamically respond to changes in grid condition, instead of only utilities and very large customers.

Like existing utility grids, a Smart Grid includes an intelligent monitoring system that keeps track of all electricity flowing in the system, but in more detail. Like the existing grid, it also has the capability of integrating renewable electricity such as solar and wind, but has the potential to do so more effectively. When power is least expensive the user can allow the Smart Grid to turn on selected home appliances such as washing machines or factory processes that can run at arbitrary hours. At peak times it could turn off selected appliances to reduce demand.

Smart Grid deployment is under way in many countries. This transformation enables new capabilities that involve an important change regarding load management. With the Smart Grid, electric utilities are deploying advanced meters and devices with microprocessors and two-way communication that allow it to better control the grid, to interact with the customers and to collect real time information about their electricity usage, etc.

Microsoft presents their vision of Smart Grid, referring to it as "smart energy ecosystem", by proposing a "Smart Energy Reference Architecture" (SERA). Web services are part of their vision. Whirlpool proposed architecture called Whirlpool Smart Device Network (WSDN) that includes the Internet as part of the communication infrastructure. Cisco delivers an IP-

based communications infrastructure for the Smart Grid. Lately, they announced a partnership with Itron who will embed Cisco IP technology within its OpenWay® meters. However, the transformation to the grid presents a large challenge for the implementation of load management measures, not to mention the challenges added by plug-in electric vehicles and decentralized electric generations.

An overlay network is a virtual network that is adapted to run on top of a physical network. It is used for a wide number of applications such as media streaming, peer-to-peer file sharing, multicast communication, virtual reality, quality of service (QoS), video and voice calls (VoIP), etc. Participant nodes are adapted to create and build the virtual network. Moreover, the participant nodes are adapted to organize themselves in a topology that allows them to exchange information.

The topology is chosen according to the communication protocols and applications that use the overlay network. For instance, resilient overlay networks preferably require topologies that have a higher level of redundancy with respect to the physical network. Video streaming and file sharing applications preferably require topologies that respect the capabilities of the source.

A network topology can be unstructured or structured. In unstructured systems there are redundant connections that improve the quality of service but increase the overhead necessary to maintain a loop free structure. In structured systems it is easier to have a loop free structure and a simple routing algorithm.

Depending on the applications and protocols some topologies may be better adapted than others; a tree topology decreases the communication delay but presents a center point failure and is not optimized for many-to-many communication. A ring topology increases the number of links but the communication delay increases with the number of nodes in the ring.

An overlay network topology can be hierarchical, having ordinary nodes on one level and super nodes on another level. It can be hierarchical and structured such as a cubic or a tore network topology. However, none of these topologies is well adapted for applications where millions of nodes need to send continuously flowing messages to the entire network. This functionality is required for some applications where a large number of intelligent communicating devices such as intelligent sensors or controllers exchange information continuously in order to achieve a common goal.

Summary

It has been discovered that it is possible to balance power network load drawn by a community of loads that are connected to a power grid while still maintaining an acceptable level of comfort by individually controlling the load according to various power grid conditions. Each load can be given semi-autonomous or fully autonomous control over the choice to draw power or to postpone drawing power as a function of the needs of other loads and/or the power grid's ability or capacity to supply power.

In the following, it shall be understood that the power consuming load or a distributed community load can be any type of load that is adapted to maintain an acceptable temperature over a period of time even after having been disconnected from the power grid. In some embodiments, the load is a water heater. In other embodiments, the load is a heating system or an air conditioning system. In yet another embodiment, the load is an electric thermal storage system. In a further embodiment, the load is a combination of types of loads that are electrically connected to a power grid.

According to one aspect, there is a plurality of loads distributed in a community that are each electrically connected to a power grid. Each of the plurality of distributed community loads is controllable by a corresponding controller. The controller can be any type of electronic device that is connectable to a network such as an Internet network or any other type of network.

According to one aspect, the controller is an electronic circuit, such as an electronic circuit board having at least one network port for two-way communication, one input port for measuring a local need parameter (i.e. temperature, battery charge, etc.) and one output port for controlling the device operations. In addition, the controller can have a control signal port for setting a mode of operation, when more than one mode of operation is required. The electronic circuit can have a processor for processing information data of the input ports and determining therefrom a power consumption instruction for the corresponding load according to the mode of operation, if applicable.

According to another aspect, there is a load management method for controlling a plurality of distributed community loads with a plurality of controllers that are connectable to a network. The plurality of distributed community loads are each connected electrically to a power grid and can be controlled by a corresponding one of the plurality of controllers.

Each of the plurality of controllers is adapted to collect a local data from the corresponding load. The local data is indicative of at least a local need parameter.

In the following, it shall be understood that the local need parameter can be any type of parameter or combination of parameters that represent an actual need of the load's subject such as a water temperature in a water heater, a room temperature in the case where the load is a heating system or an air conditioning system, a storage temperature in the case of an electric thermal storage system, a battery charge level in the case of a rechargeable battery such as those used in automotive vehicles, a quantity of coolant material that is in a particular phase in the case of a heat exchanger using a bi-phase coolant etc.

The local need parameter can also represent a predetermined acceptable threshold or a combination of predetermined acceptable thresholds such as a minimum or maximum temperature, battery charge level or quantity of coolant material in a particular phase. In the case of a water heater, the local need parameter can represent a minimum water temperature. In the case of a heating system, the local need parameter can represent a minimum room temperature. In the case of an air conditioning system the local need parameter can represent a maximum room temperature. In the case of an electric thermal storage system, the local need parameter can represent a minimum thermal storage temperature. In the case of a rechargeable battery, the local need parameter can represent a minimum battery charge. Also, in the case of a rechargeable battery for a vehicle, the local need parameter can represent a combination of a minimum battery charge, a time of use and a minimum distance of travel. In the case of a heat exchanger, the local need parameter can represent a minimum quantity of coolant material in a particular phase.

In one embodiment, the local need parameter is a temperature of a hot water heater that is measured by a sensor at a single position within the water heater tank. Using this measured temperature, the controller decides when to turn on or off the electric consumption of the water heater.

The sensor could be positioned on an exterior wall of the tank. The sensor could be positioned where the water temperature is representative of the temperature of water exiting from the tank, such as near a top position of the tank.

In an alternate embodiment, the local need parameter is a temperature of a hot water heater that is measured at multiple positions within the water heater tank. There are several

temperature sensors located at various positions in the water heater tank to measure the temperature at those positions. A single temperature is then derived from the multiple measurements and the controller is then adapted to use that single temperature to turn on or off the electric consumption of the water heater.

Each of the plurality of controllers is adapted to receive a community data from at least one other node of the network. The community data is indicative of at least one of a power grid supply parameter or a community need parameter of the plurality of distributed community loads.

In the following, it shall be understood that the power grid supply parameter can be any type of parameter that represents an actual state of the power grid such as an available power on the grid or a predetermined power consumption value such as a power consumption quota or a power consumption price for a predetermined period of time, etc.

In the following, it shall also be understood that the community need parameter can be any type of parameter that represents an actual state of the other loads of the plurality of distributed community loads such as a water temperature in a water heater, a room temperature in the case where the load is a heating system or an air conditioning system, a storage temperature in the case of an electric thermal storage system, etc. The community need parameter can further represent a power consumption need of each of the plurality of distributed community loads or a ranking of an actual state of the other loads or of all the distributed community loads.

Based on the collected local data and the received community data, each of the plurality of controllers is adapted to set a power consumption of its corresponding load for reducing the power consumption of the load while maintaining an acceptable comfort level.

According to one embodiment, each of the plurality of controllers is adapted to set the power consumption of its load to an active status or to allow load power consumption when a value of the local need parameter is beyond an acceptable local threshold and a value of the power grid supply parameter is within an acceptable supply range.

There are various ways of applying this method to various types of loads. In one case where the loads are water heaters, the controller is adapted to turn on its water heater when the water temperature reaches a value that is below a predetermined threshold only if the power

available in the grid is within an acceptable range and/or only if the price of power is within an acceptable range at that period of time.

According to another embodiment, each of the plurality of controllers is adapted to set the power consumption of a corresponding load to an active status when a value of the local need parameter is beyond an acceptable local threshold and a value of the community need parameter is within an acceptable community threshold.

Again, there are various ways of applying this method to various types of loads. In one case where the loads are water heaters, the controller is adapted to turn on its water heater when the water temperature reaches a value that is below a predetermined threshold only if the water temperature is among the lowest in comparison with the water temperature of the other loads connected to the grid.

According to another embodiment, each of the plurality of controllers is adapted to set the power consumption of a corresponding load to an active status when a value of the local need parameter is beyond an acceptable local threshold, a value of the community need parameter is within an acceptable community threshold and the power grid supply parameter is within an acceptable supply range.

Another object of the present is to provide a distributed communication and management system that permits reducing message transmission delay while assuring reliability and scalability. Another object of the present is to provide a distributed communication and management system that simplifies and lowers the cost of managing such system.

According to one aspect the present relates to a network node device for communicating with other node devices of a same community within an overlay network topology system. The topology system has a plurality of node devices organized in a number of p hierarchically linked ring network levels, where p ranges from 0 to p-1. The plurality of node devices of a same ring is interconnected in groups of n node devices, where n ranges from 0 to n-1. One of the n node devices is a parent node for linking the n node devices of a first ring to node devices of a second ring at another hierarchy by being a parent node to at least the second ring.

The node device has a network connection interface to communicate over an interconnected network of node devices. The interface is related to a network address.

The node device further has a registry of connectable node devices adapted to store node device network address and identifiers each representing a node device of the community. The identifiers are indicative of the ring network level and ring network position of the corresponding node device according to the overlay network topology.

The node device further has a registry update manager adapted to modify the registry for handling a change in the community due to drops, adds and moves of node devices that affect the registry

The node device further has a forwarding message manager adapted to receive an incoming message via the network connection interface and to handle the incoming message in accordance with a content of the incoming message and the registry. When the network node device is a parent node the forwarding message manager is adapted to send an outgoing message according to the incoming message to a plurality of node devices of the registry, each having a node position on a different one of the rings that are linked by the network node device. When the network node device is not a parent node the forwarding message manager is adapted to send the outgoing message according to the incoming message to another node device of the same ring network level according to the registry.

According to another aspect the present relates to an overlay network topology system that has a node device as defined above.

Brief Description of the Drawings

The invention will be better understood by way of the following detailed description of embodiments of the invention with reference to the appended drawings, in which:

Figure 1A is a block diagram of an Internet peer-to-peer water heater network for load management, according to one embodiment;

Figure 1B is a block diagram of an Internet peer-to-peer load network for load management, according to one embodiment;

Figure 2 is a graph representing a sample profile of 10,000 water heaters where a hot water consumption peak is noticeable at 7 am and at 7 pm, according to one embodiment;

Figure 3 is a graph representing a sample profile of 1,000 water heaters where a hot water consumption peak is noticeable at 7 am and at 7 pm, according to one embodiment;

Figure 4 is a graph representing a sample profile of 100 water heaters where a hot water consumption peak cannot be distinguished, according to one embodiment;

Figure 5A is a flow chart of a load management method for controlling distributed community loads, according to one embodiment;

Figure 5B is a diagram representing all the loads in a community, according to one embodiment;

Figure 5C is a diagram representing a set of saturated loads and a set of unsatisfied loads of a community, according to one embodiment;

Figure 6 is a graph representing a normalized average daily power consumption profile, according to one embodiment;

Figure 7 is a graph representing a worst case power consumption profile, according to one embodiment;

Figure 8 is a graph representing a best possible result for the typical consumption profile, according to one embodiment;

Figure 9 is a graph representing a power consumption profile where a load management method was applied with a normalized quota of 0.72kW, according to one embodiment;

Figure 10 is a graph representing a power consumption profile where a load management method was applied with a normalized quota of 0.6kW, according to one embodiment;

Figure 11 is a graph representing a power consumption profile where a load management method was applied with a normalized quota of 1.26kW, according to one embodiment;

Figure 12 is a graph representing the reached consumption peak in function of allowed quota, according to one embodiment;

Figure 13 is a graph presenting the effects of various quota values on various peak consumption values, according to one embodiment;

Figure 14 is a graph presenting a 50% consumption peak reduction, according to one embodiment;

Figure 15A is a prior art representation of a tree topology;

Figure 15B is a prior art representation of a linked tree topology;

Figure 16 is a representation of a two level ring-tree topology, according to one embodiment;

Figure 17 is a prior art representation of a four level tree topology having 15 nodes;

Figure 18 is a representation of a three level ring-tree topology adapted to have up to 27 nodes, according to one embodiment;

Figure 19 is a diagram representing a subset of layers from the Open System Interconnection (OSI) model, according to one embodiment;

Figure 20 is a representation of a three level ring-tree topology in which there is illustrated a sending of multicast message, according to one embodiment;

Figure 21 is a representation of a three level ring-tree topology in which there is illustrated a sending of broadcast message, according to one embodiment;

Figure 22A is a representation of a three level ring-tree topology in which there is illustrated a neighbouring table corresponding to each of the nodes;

Figure 22B is illustrated a neighbouring table corresponding to a node;

Figure 23A is a prior art graphical representation of a single node failure in a ring topology;

Figure 23B is a prior art graphical representation of a multi-node failure inside a ring;

Figure 24 is a graphical representation of a parent node failure in a ring-tree topology, according to one embodiment;

Figure 25 is a graphical representation of a network insertion sequence, according to one embodiment;

Figure 26 is a flowchart presenting the steps to resolve node failures, according to one embodiment;

Figure 27 is a representation of a three level ring-tree topology in which there is illustrated a neighbouring table corresponding to each of the nodes and the spiral communication path according to the neighbouring table of one node, according to one embodiment;

Figure 28A is a state diagram of a node representing a node join to the network, according to one embodiment;

Figure 28B is a state diagram of a node representing a node join to the network, according to one embodiment;

Figure 29 is a block diagram representing the modules of a node, according to one embodiment; and

Figure 30 is a block diagram representing the modules of a node, according to one embodiment.

Detailed Description

The present relates to a distributed method for the load management of water heaters. This method assumes that water heaters of a city or a province, belonging to the same electric utility, are equipped by a communications-enabled controller. The controller allows them to communicate with each other via any network such as the Internet, in a peer to peer way, a Smart Grid's two way communication system or any other communication infrastructure. When water heaters are able to communicate with each other, they can assure a distributed load management without the intervention of the electric utility or any direct control. The decision to turn on and off is taken by the water heater itself using information received from other water heaters about their electrical energy needs and from the electric utility about the general load on the grid.

Figure 1A shows an illustration of an Internet peer-to-peer water heater network 102 for load management. This network 102, not addressed here, must support communication between thousands and up to millions of water heaters of the same electric utility. A networked controller 104 can be embedded in the appliance or it can be part of a home gateway and it may use the residential Internet connection. A skilled person will understand that it is possible for the method to take advantage of the principle of storing energy in thermal form. This principle is used in some other energy industries (e.g., space heating and cooling). Energy is stored during off-peak times in thermal energy storage for later use during times of peak demand. In such techniques, consumption of electricity will occur during off-peak times where the need for electricity is lower and in some cases the price of the electricity is cheaper. For example, electrical energy can be stored in Electric Thermal Storage to be used later for space heating during the peak-demand period; there is no consumption during peak-demand. Another example; ice can be produced during the night (an off-peak time) and used to cool environments during the day. These techniques don't necessarily use direct control.

According to an alternate embodiment, there is presented in Figure 1B a block diagram of a peer-to-peer network 102 for load management. In this network 102 an energy management system EMS 103 is adapted to control a plurality of loads 106 and a network controller 104 is adapted to control a single load 106. A coordinator 108 is also connected to the network 102, the coordinator 108 may be any one of an electric utility, a systems operator, a balancing authority, a virtual power plant, etc.

Note that the peer-to-peer network 102 may use any type of communication network such as an internet network, a cellular network, etc.

Electric Water Heater Model

The water temperature in the water heater 106 is very stratified with height; so that the hot-water temperature exiting from the top of the tank is very different from the average temperature. In the present, the average water temperature of the water heater is being considered. The average water temperature is evaluated by the differential equation (1) developed and based on works known in the art. This water heater model is known in the art, however the approach of the present is different, as will be shown in the following section.

Elemental electric water heating stochastic model

For each water heater, the water temperature must be monitored at all times. Solving the differential equation (1) provides the temperature x(t) at any moment.

$$\frac{dx(t)}{dt} = -a[x(t) - x_a(t)] - A\kappa(t) + Rm(t)b(t) \quad (1)$$

Where x(t) is the average tank water temperature at time t, xa(t) is the ambient temperature, a is the thermal resistance of the tank walls, A(t) is the rate of energy extraction when water is in demand, $\kappa(t)$ is the normalized (1-0) hot water demand jump process, R is the power rating of the heating element, m(t) is the thermostat binary state (1 for On, 0 for Off), b(t) is the on-off control used by the control-algorithm. The switching of $\kappa(t)$ is characterized by the following transition probabilities:

$$Pr[\kappa(t+h) = 1 | \kappa(t) = 0] = \alpha_0 h$$

$$Pr[\kappa(t+h) = 0 | \kappa(t) = 1] = \alpha_1 h$$

Where α_0 and α_1 are positive and h is an infinitesimal time increment.

From equation (1) the following equation (2) is obtained:

$$x(t) = e^{-at}C + e^{-at}\int_0^t e^{a\tau}(-A(\tau)\kappa(\tau) + Rm(\tau)b(\tau) + ax_a)d\tau \quad (2)$$

Where C is obtained considering that at the initial time, x(t) is equal to the ambient temperature.

In a simulation of the dynamic water demand over 24 hours the α_1 was considered to be constant and the α_0 value was varied every 30 minutes. Therefore, in 24 hours there were 48 different values of α_0. The α_0's variation were set according to the same curve as a daily hot water demand curve: when the hot water demand is high, the probability to extract hot water

is also high, and when the hot water demand is low, the probability to extract hot water is also low. The α_0's variation where also set according to the electric consumption curve, as the curve of daily hot water demand is similar thereto. So, α_0 also follows the same curve pace of the electric consumption.

This method considers that the probability for the consumer to start extracting water varies in time following the curve of the electricity demand and the consumer stops using water with a constant probability. To simulate different profiles it was only required to change the 48 values of α_0 in a way that it follows the pace of the needed profile. It was therefore possible to generate any consumption profile by varying the values of α_0 over 24 hours.

Water heater simulation

For simulation purposes, equation (2) was programmed in Java programming language, according to households with a family of 3 people having a 52 gallon sized water tank and 4.5kW heating elements. In such families, it was determined that the water consumption is about 266 liters per day with an average of 17 water events per day. The thermostat set point is 60°C with a 5°C deadband. The parameters of the water heater model associated to this type of household are:

R=0.3279 °C/min

a=0.000156 min-1

A=1.29 °C/min

α_0=0.012 min-1

α_1=0.32 min-1

In the simulation the time increment h is set to 1 min and the value of α_1 doesn't change with time, as mentioned earlier. The 48 α_0's values were calculated according to the profile to be simulated and the α_0's value is considered to be an average. To simulate a sample profile where there is a peak at 7 am and another one at 7 pm, and α_0's value was chosen in order to follow the pace of a typical profile, and a simulation of 10,000 water heaters was run as shown in Figure 2.

All simulated water heaters start with initial temperature value of 20°C. The simulation was run for 48 hours in order to reach a steady state and the data of the last 24 hours were analysed.

Figures 3 and 4 show simulations for the sample profile using 1,000 and 100 simulated water heaters. It is noticed that with a small number of simulated water heaters the shape of the sample profile gets lost. This is caused by the statistical aspect of the elemental water heater model. Consequently, the following results were simulated with 10,000 water heaters.

Water Heater distributed Method

There is presented in Figure 5A, a distributed algorithm 400 for load management. Shown in Figure 1A, each water heater 106 is equipped with a networked controller 104 that is adapted to perform the distributed algorithm 400. The controller 104 allows the water heaters to communicate between each other; it can measure the hot water's temperature, and it controls the electrical consumption of the water heater.

Depending on load forecasting, potential of energy storage, state of the loads, the availability of generation, and/or market price, a power consumption quota 410 is determined. A time dependent quota that respects the conditions discussed further allows creating any load profile. The quota can be calculated inversely proportional to a variable tariff; high prices mean low quotas and low prices mean high quotas. The quota can be represented by power units. For example one unit is equal to 1kW. In this case, if a water heater operates at 4.5kW it needs 4.5 units to operate at full power. In the simulations, there is considered that one unit is equal to 4.5kW; all water heaters are identical and have a single heating element of 4.5kW. Therefore, the number of water heaters can express the quota. This simplifies the presentation. For example, if there are 10,000 water heaters with single heating element of 4.5kW participating in the load management program, and the quota is 18MW then only q=4000 water heaters are allowed to consume at the same time.

There are two types of quotas: optional and mandatory. In optional quota water heaters respect the quota until their hot water temperature reaches a minimum level Min. Then they will consume without respecting the quota as it is optional. In mandatory quota, water heaters respect the quota without considering the hot water temperature. This type of quota can be used when the security of the grid is more important than user's comfort. It can be used to avoid blackouts for example. To turn off all the water heaters, the electric utility can send them a mandatory quota fixed to zero q=0.

Once the quota is calculated, its value q is sent 412 via the network 102 to all the controllers 104. If the controllers are connected to a Smart Grid, this information can be available to the controllers via a Smart Meter.

Knowing the quota's value q, as each controller monitors 414 the water temperature of its associated water tank, the controller has real time information about the water temperature x(t). It receives 416, via the network 102, from the community of participating water heaters 106, a list of controller identification numbers (ID) ordered by temperature 426. Each controller receives 416 the temperature of only a set of q water heaters having the lowest temperature. If its own ID is present 418 in the ordered list and its hot water temperature is lower than the maximum value (Max) 420 then it starts heating 422. In other words, a given water heater decides to consume only what it needs to consume (its temperature is under Max) and it has a lower hot water temperature than at least (N-q) water heaters. If not, it will update 424 the information in the list and pass it on. Water heaters having the lowest temperature always have the priority to consume.

In a set $X_N(t)$ of N water heaters 106 participating in the load management program, there is a subset $X_q(t)$ of q water heaters 106 with the lowest temperature where $X_N(t) = X_q(t) \cup \overline{X}_q(t)$. Each element in the set is a couple $(x(t), ID)$ of temperature $x(t)$ and a unique controller identification number ID.

$$X_N(t) = \{(x(t),\ ID)\ \ of \quad all \quad water \quad heaters \quad \} \tag{3}$$

The size N of $X_N(t)$ may vary with time t when water heaters 106 leave and join the load management program. The size q of $X_q(t)$ varies with time depending on q's value. It is considered that N and q are time invariant. Elements in $X_q(t)$ have higher priority to use the energy because their temperature is lower than those of $\overline{X}_q(t)$.

In one embodiment, each controller 104 monitors the water temperature 414 by calculating or measuring its own temperature every t minutes and communicates it to other water heaters 104 in the community. A decision is made every t minutes and only water heaters 106 with the lowest temperature will consume electricity. The total number of water heaters operating at the same time is based on the general power consumption quota. Waters heaters 106 that

do not have a temperature within the range of temperature considered as being the lowest are turned off.

In another embodiment, when in optional quota setting 430, if a water heater's temperature reaches the minimum temperature comfort level (Min) 428, the water heater starts consumption independently of the quota. In this case, the quota may not be respected and a peak demand reappears, as we will see in the results section. A given controller i turns on the water heater only if the condition (4) is satisfied.

$$\left[\left(x_i(t) < Max\right) \wedge \left((x_i(t), ID_i) \in X_q(t)\right)\right] \vee \left[X_i(t) < Min\right] \quad (4)$$

In yet another embodiment, when in mandatory quota setting 430, the water heater must respect the quota independently of hot water temperature. In this case the quota is always respected. When the condition in (5) is respected, the controller i turns on the water heater, if not it turns it off.

$$\left[\left(x_i(t) < Max\right) \wedge \left((x_i(t), ID_i) \in X_q(t)\right)\right] \quad (5)$$

According to example of the present invention, each load (e.g. water heater) has a state parameter x(t) that represents its actual need for consumption. The state parameter, also called need parameter, is defined according to the type of the load; for example the state parameter of a water heater or an Electric Thermal Storage is the temperature, and the state parameter of a Plug in Electric Vehicle is the battery charge level. In these cases, low value of the state parameter corresponds to a higher consumption need. Hence, all the loads in the community can be ordered according to their state parameter.

A number of nodes q (i.e. loads) having the lowest state parameter values within the community N, constitutes a subset Q of the set N; the cardinality of Q is q(t) and its value varies with time according to the quota's value of the permitted power consumption.

The distributed algorithm in each node allows determining if a load belongs to the subset Q or not, as shown in Figure 5B. A load belongs to the subset Q if its state parameter value is lesser than a threshold parameter value $x_{th}(t)$ and if the limited size q(t) of Q is respected. A threshold parameter value $x_{th}(t)$ is defined as follows: if $x_i(t)$ is the state parameter of node ξ_i and $x_j(t)$ is the state parameter of node ξ_j then

$$\exists\, x_{th}(t): x_{th}(t) < x_i(t) \text{ and } x_{th}(t) > x_j(t)$$
$$\forall\, \xi_i \in Q^c \text{ and } \forall\, \xi_j \in Q \quad (6)$$

where Q^c is the complement of Q in N.

The nodes in Q are the only nodes allowed to consume. By varying the value of q(t) with time, the consumption power profile is reshaped according to q(t). Knowing the quota's value, the nodes exchange information about their state parameter in order to build the subset Q, and to determine the threshold parameter value $x_{th}(t)$.

In one example, the control is regularly applied and not only during peak periods. The time is divided into cycles. The duration of a cycle can be set for example to 1, 5, 15 minutes or any other suitable period of time. A new decision for consumption is made by every node at the end of each cycle. The length of a cycle can be variable in time and in a cycle the following actions are performed: ordering, in ascending order, all the nodes of the set N according to the state parameter; calculating the value of the power consumption quota q according to load forecasting, storage capabilities, state of the loads, and generation availability with respect to the formulated conditions; receiving a quota value by all the nodes; constructing subset Q by using the first q values in the ordered list; determining the value of the threshold x_{th} which is higher of the highest value in the subset Q and lower than the lowest value in Q^c; comparing by every node k, its state parameter x_k to the threshold x_{th} and to x_{min}; and deciding if electric consumption is required according to x_{th}, x_{min}, and the type of quota.

According to one embodiment an objective is to change the behaviour of the nodes of the community in a way that their power consumption profile meets the quota's values at all time. In one example:

$$\sum_{k=1}^{q} p_k(t) = \text{quota}$$

and

$$\sum_{k=q+1}^{n} p_k(t) = 0 \quad (7)$$

where the loads are in ascending order according to the state parameter, $p_k(t)$ is the power consumption of load k at time t, $q = |Q|$, and $n = |N|$ hence,

$\forall\, \xi_j \in Q, p_j(t) \neq 0$ and $\forall\, \xi_i \in Q^c, p_i(t) = 0$. (8)

There is a relation between the quota, which is the amount of power consumption permitted, and q the cardinality of Q. If all the loads are identical (i.e. they have the same power

rating r), the quota's values becomes proportional to $q(t)$, the time dependent cardinality of Q. Consequently, the objective equation is:

$$\sum_{k=1}^{n} p_k(t) \leq r\, q(t). \quad (9)$$

Therefore, varying the quota's value reshapes the load profile of the community N by allowing the loads with a higher need to consume first. Consequently, any power consumption profile can be created, if the variable quota respects certain conditions.

Conditions to be respected

If, at time t, the quota's value $q(t)$ is higher than the power that can be consumed by all the loads, the subset Q will have loads incapable of consuming power (i.e. $\xi_j \in Q$ and $p_j(t) = 0$) and the objective equation is not respected. To avoid this situation the quota's value must be determined according to a first condition (i.e. condition 1):

$$q(t) < n(t) - s(t) \quad (10)$$

where $s(t)$ is the cardinality of a set S of all the saturated loads within the community.

Note a saturated load is defined as a load having a state parameter value that is higher or equal to a maximum value (i.e. $x_k \geq x_{max}$). The maximum value (i.e. x_{max}) is determined according to the storage capacity of the load.

However, if the quota's value is lower than the power needed by the unsatisfied loads, some unsatisfied loads will overflow the subset Q; if the quota is mandatory, the comfort level of the users having their loads situated outside Q will be affected; if the quota is optional these unsatisfied node will consume without respecting the quota (i.e. $\xi_i \in Q^c$ and $p_i(t) \neq 0$) and the objective equation will not be respected. To avoid this situation with an optional quota the quota's value must be determined according to a second condition:

$$q(t) > u(t) \quad (11)$$

where $u(t)$ is the cardinality of a set U of all the unsatisfied loads within the community.

Note, an unsatisfied load is defined as a load having a state parameter value less than a minimum value i.e. $x_k < x_{min}$. The minimum value (i.e. x_{min}) is determined according to the user's comfort level.

As presented in Figure 5C, the satisfaction of the first and second conditions implies that:

$$U \cap Q^c = \emptyset \text{ and } S \cap Q = \emptyset. \quad (12)$$

Having more storage capabilities reduces the limitation of the first condition, and using a mandatory quota overrides the second condition but it may affect user's comfort.

If the quota is constant over a period of time, a flat profile can be created when the quota is met at all time. When user's comfort is prioritized, the following assumption can be made: "the minimum energy needed by a load over a period of time after control must remain the same as before control".

This assumption sets a lower limit for a constant quota's value used to flatten a profile over a period of time T = 24. Hence, a third condition is be formulated:

$$q(t) \geq \frac{\sum_{h=1}^{T}\sum_{k=1}^{n} p_k(h)}{T} \qquad (13)$$

where $p_k(h)$ is the power consumption of load k during the period of time h, h = 1..T.

Hence, the best possible theoretical result is achieved when an optional fixed quota is respected and equal to:

$$q(t) = \frac{\sum_{h=1}^{T}\sum_{k=1}^{n} p_k(h)}{T}. \qquad (14)$$

Consequently, the energy needed over a period of time T does not change, the user's comfort is not affected, and the overall load is flattened. The value of the quota is equal to the average power consumption during the period of time T. In the following case study, a constant quota equal to the average power over 24 hours is to be met by a community of loads.

According to the distributed algorithm based on the threshold value described above, the power consumption of the water heater is driven by its parameters as follow:

$$p_k(t) > 0 \; if \; [b_k(t) = 1 \text{ and } m_k(t) = 1]. \qquad (15)$$

The value of m(t) varies according to the following:

$$m_k(t+1) = \begin{cases} 0 & \text{if } x_k(t) > x_{max} \\ 1 & \text{elsewhere} \end{cases}. \qquad (16)$$

When the quota is mandatory, the value of b(t) varies according to the following:

$$b_k(t+1) = \begin{cases} 1 & \text{if } x_k(t) < x_{th}(t) \\ 0 & \text{elsewhere} \end{cases}. \qquad (17)$$

When the quota is optional, the value of b(t) varies according to the following:

$$b_k(t+1) = \begin{cases} 1 & \text{if } x_k(t) < x_{th}(t) \text{ or } x_k(t) < x_{min} \\ 0 & \text{elsewhere} \end{cases}. \qquad (18)$$

In one embodiment, in order to determine $x_{th}(t)$, a histogram of temperature is constructed by all the nodes. The values under $x_{min}(t)$ are the unsatisfied nodes, and the values over $x_{max}(t)$ represents the saturated nodes. The threshold value $x_{th}(t)$ is determined using the information in the histogram and the quota's value. The use of a histogram minimizes the size of the messages exchanged between the nodes because there is no need to exchange a list of pair $(x(t), ID)$ but only a table of temperature distribution.

Results without any control

Presented in Figure 6 is a graph showing the average power needed by a water heater over 24 hours (based on a simulation of 10,000 water heaters). This profile is comparable to the typical power consumption profile. During the night between 1 AM and 6 AM the energy consumption dropped below 0.5 kW. During the day and evening consumption varied between 0.5 kW and 1.2 kW. There were two higher peaks at 7 AM and 7 PM. Around noon and mid-evening the level of consumption was lower than the peaks but it lasted for a longer period. The normalized energy consumption during 24 hours is around 14 kWh per day.

Results with load management distributed algorithm

One objective is to have a relatively more uniform power distribution of the Domestic Hot Water load profile. The peak reduction depends on the shape of the curve. In one power consumption profile, a power consumption is uniformly distributed over 24 hours. In such a profile, there is no peak to reduce and there is no need for control. In an alternate power consumption profile, a power consumption is concentrated in only one unit of time, as shown in Figure 7. Although non realistic, the reduction percentage of this peak is relatively greater and it is harder to control for changing the profile to a uniform shape.

To a more realistic profile obtained from a simulation based on typical hot water usage in a number of dwellings, there is applied a probabilistic mathematical model as described in the previous section. The objective is to transform this profile to a uniform profile as shown in the graph of Figure 8.

According to one embodiment, when there was applied the distributed algorithm with a normalized quota q=0.7kW to the realistic profile, it was possible to reduce the peak by 40% from 1.2kW to 0.7kW, as shown in Figure 9.

In a perfectly uniform daily profile the value of the power is constant during 24 hours. Considering that this value corresponds to the mean value of the realistic profile presented

earlier equal to 0.6kW, if there are applied constant quotas with values lower than 0.6kW those will not be respected. Since, with an optional quota, it is not possible to force the users to consume less than the energy needed during 24 hours, the value of the quota must be higher than the expected mean value of a random profile.

When the aim was to reduce the highest peak of the morning by 0.6kW, the optional quota wasn't respected and the reduction wasn't fully reached. In Figure 10, there can be seen a reduction of the peak of the morning by 0.55 kW but the peak of the evening is the same as the peak of the original profile. The quota was respected until 7 o'clock where another peak, lower than the original, reappeared and after that the shape of the controlled profile followed the original shape. The value of this quota is too low. It doesn't permits to reduce all the peaks while respecting the comfort of the user.

When there was applied the distributed algorithm with an optional quota of 0.72 and higher, the quota was always respected. But the benefit decreased with increasing the value of the quota. In Figure 11 for example, the quota is equal to 1.26kW.

This quota is respected during the 24 hours. The peak is 1.11, but there can be seen that the shape of the curve is almost the same as the original one. Therefore it is noticed that there is no benefit for high quotas.

Several simulations were done with different values of quota. In Figure 12 there is shown the reached peak in function of allowed quota which is the aimed peak. The aimed quotas are reached only when the quota passes a certain value 0.7kW in this case.

This confirms that the quota is only effective in reducing the peak when its value is within a certain range. If the quota is too low, it doesn't reduce the peak. This is because the algorithm, in order to respect user comfort, ignores the optional quota allowing any water heater to start consumption as soon as its temperature drops to the minimum level (Min=60°C). At the higher quota levels, the peak is nearly the same and therefore there is no benefit.

Electric energy stored in a thermal form in water Heater

To store the electric energy in thermal energy form the temperature was allowed to drop to a minimum value of 55°C instead of 60°C, and the temperature was allowed to increase to a maximum value of 70°C instead of 65°C.

Several simulations were done with different values of quota, as shown in Figure 13. Again, the quota was only effective in reducing the peak when the quota's value was within a certain range. But, it was possible to reduce the peak from 1.2kW to 0.6kW. This is the best possible result that can be achieved for this profile, as shown in Figure 14. This corresponds to a peak reduction of 50% and a load factor of 100%.

This result is specific for this profile. Considering the worst case profile described earlier, the best possible result may not be achieved unless the difference between Max and Min (Max-Min), becomes unrealistically high.

When the value of the quota is too low or too high, there is no benefit. When the optional quota is too low compared to the consumption need, the controllers don't respect it because it gives priority to the user comfort. When the quota is too high it means that the reduction aimed is low therefore the benefit is low.

Discussion

In practice, maintaining hot water at a higher temperature may require the use of a Thermostatic Mixing Valve (TMV) to ensure constant and safe outlet temperatures. Different material for the water heater tank must be considered as well because the durability of the tank depends on the water temperature among other factors.

When hot water is maintained at a higher temperature, hot water consumption is reduced because the consumer will use a lesser quantity of hot water mixed with a greater quantity of cold water. The water heater model used in our simulations considers a constant amount of hot water extraction. We believe that using a variable amount of hot water extraction depending on the high temperature will improve our results on reducing the peak demand.

Another factor that may improve our results is when the quota is variable with time. The quota must indicate the load on the grid. When the load is higher the quota is lower and when the load is low the quota must be higher. Our results were simulated with a constant quota; the quota is fixed during the simulation of 48 hours. When the quota is higher during off-peak time more energy will be stored in hot water because a higher number of water heaters will be allowed to raise their temperature to the Max (70°c). There is believed that a variable quota improves the results.

Despite all these simplifications, a reduction of 50% is a very encouraging result. Considering that 2,400,000 electric water heaters are responsible of 5% of a grid peak of

36,000MW and a community of 1,600,000 water heaters participates in a load management program and 50% of their peak is reduced, then a reduction of 600MW can be reached.

A skilled reader will understand that if load management at the appliance level were applied, it would be possible to achieve a better theoretical profile, for the case of water heaters, using thermal energy storage and quota of consumption power.

According to one embodiment, there is a distributed load management method applied to water heaters. Water heaters communicate together and with the electric utility to have the information necessary for the distributed algorithm. This distributed algorithm reduces the peak by about 40%. By increasing the hot water's temperature to 70°C during off-peak time and allowing it to drop to 55°C during peak-time, results are improved by additional 10%.

A skilled reader will understand that efficiency of a collaborative method is increased when the right quota is chosen. When the quota is too low another peak reappears.

Overlay Network Topology

In the present there is described an overlay network topology system, architecture and protocols. The system is a communication network that allows distributed devices, herein referred as nodes, to build and manage a community of devices in order to exchange messages or information of small and medium size. Further described in the present is how to construct and manage a community of nodes within the system. The primitives or commands necessary in order to use the system are also described.

A community of nodes could be any group of nodes adapted to communicate with each other. The group of nodes share a common need or interest.

According to one embodiment, each node of the group is a device such as a sensor or a controller for a heating unit, a hot-water heater, an electrical car, an electrical entrance of a building or any other type of unit or combination of units that require electrical energy to operate. In these examples, each device has a common need for electricity; however it is possible in other examples that the common need be different. For instance, in another example, each device has a common need for water.

According to one embodiment, the group of nodes share a common need and have a common objective. For instance, when the common need is electricity, the common objective could be to reduce the global electricity consumption of all participating devices. When the common

need is water, the common objective could be to reduce the global water consumption of all participating devices.

According to one embodiment, the common objective is achieved by distributing the commonly needed resource according to each device requirements. As the devices are adapted to communicate with each other, the requirements of each is known and the resource can be effectively distributed.

In yet another embodiment, each node of the group is a device for distributing information data such as a computer terminal or a surveillance system. Each device shares a common interest such as sharing or exchanging multimedia content, participating in an online computer game, detection and identification of moving objects, etc.

According to one embodiment, the number of nodes in a community is variable and the system is adapted to dynamically handle system size variations while assuring an acceptable level of fault tolerance.

Prior Art Tree Topology

There is presented in Figures 15A and 15B prior art network topologies such as a tree topology 1500 in Figure 15A and a linked node tree topology in Figure 15B. In a tree topology 1500 there is one root node 1502 and n direct children 1504 and each of the n direct children has n children 1506, etc. In a linked node tree topology 1510 there is a root node 1512 and n direct children 1514 that are sequentially linked, each of the n direct children has n children 1516 that are also sequentially linked, etc.

Ring-Tree Topology

In a ring-tree topology, there are n root nodes forming a root ring. Each root node is adapted to have n direct children forming a ring with this root node and each direct child is adapted to have n children forming another ring, etc.

Presented in Figure 16, according to one embodiment, is a ring-tree topology 1600 of two levels, a first level 1602 referred here as L0 and a second level 1604 referred here as L1. Each ring has a Super node or herein forth referred to as a parent node 1606. The ring-tree topology 1600 can be seen as a tree topology where nodes of the same parent form a ring between each other and with their parent (i.e. the parent node links the ring with the rest of the network or community). Each parent in the tree topology has n nodes situated on one level. Associated to each parent node of the Ring-Tree topology is n-1 children nodes 1608

situated on each level until L-1 levels. In other words, in a tree topology a parent situated on level 1 can have n children situated on level 2; in a Ring-Tree topology, a parent node of level 1 can have n-1 children of level 2 and another n-1 children of level 3, etc. until the level p-1.

According to one embodiment, the value of n may differ according to the p level in the ring-tree topology. For example the value of n may be greater for a lower p level to increase the robustness of the topology.

According to one embodiment, there is associated to each node (1606 and 1608) a neighbouring table that has a list of node identifiers representing each neighbouring node.

According to one embodiment, a parent node 1606 of level L1 1604 holds information about all nodes of the same ring and all its descendants only. It doesn't hold information about the descendants of its neighbour. Its neighbourhood table is limited to a size $p \times n$. The total number of nodes in the community is $N = n^p$ nodes, $M = (n-1) \cdot n^{(p-1)}$ of them are leaf nodes and $G = n^{(p-1)}$ are Parent nodes. The proportion of Parent nodes is 1/n. Recall that n is the number of participants in a ring and p is the number of level in the community.

Each ring, via its parent node, can be seen as a node by other parent nodes on the same level. If for example a parent node needs to send a message to its neighbour it can specify if the message must reach only the neighbour or the neighbour and its entire descendant (Multicast Message). So a Parent node can be addressed as a single node or as a group of nodes composed by the Parent node and its entire descendants. Descendants of the neighbour are not visible to a given node (i.e. nodes do not hold information about descendants of the neighbor). So a given node deals with a limited number of participants but it can still reach the whole community.

According to one embodiment, a parent node has more roles or responsibilities than has a child node. As a parent node has a greater number of node links than has a child node, the parent is responsible for routing information to a greater number of nodes. Moreover, the parent node is responsible for assuring a link between the children nodes and the nodes of a lower level.

When a node exists on level L where L < (p-1), this node is a Parent node; because all nodes are Parent nodes except the child nodes (i.e. nodes of level p-1).

Presented in Figure 17 is a prior art four level tree topology having 15 nodes. In a tree topology the number of links between the nodes is limited to a single link. When the single link disappears, the corresponding node gets disconnection from the network. It is known that the tree topology presents a relatively high vulnerability to node disconnections.

Presented in Figure 18 is a three level ring-tree topology adapted to have up to 27 nodes, according to one embodiment. With such topology a node can have a maximum of n*p direct bidirectional links with its neighbours.

Returning to Figure 16, according to one embodiment, one of the nodes of the topology 1600 is an assignment server 1610. The assignment server 1610 is adapted to assign a position to a new comer node as further described below.

According to an alternate embodiment, the assignment server 1610 is a device that is not part of the topology 1600. The assignment server 1610 is adapted to communicate with at least some of the nodes of the topology 1600 and assign a position to a new comer node as further described below.

OSI Model

In Figure 19 there is a diagram presenting a subset of layers from the Open System Interconnection (OSI) model. There is an OSI model Transport Layer 1910 and an OSI model application layer 1912. Between those two layers (1910 and 1912), there is a Network Topology Layer 1914 and a Management Layer 1916. According to one embodiment, the Network Topology Layer is a Ring Tree Topology Layer (RTTL) 1914 which will further be described in the following.

Transport Layer

The transport layer 1910 of the OSI model can offer any transport protocol such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). Primitives of the RTTL 1914 use services provided by the Transport Layer 1910.

Application Layer

Each client application of the application layer 1912 uses services offered by the RTTL 1914 and Management Layer 1916. The primitives or commands necessary to use these services are detailed in the Ring-Tree Topology Layer section. One same client application can create different nodes in different communities or in a same community. So via the application layer 1912, it is possible to exchange information between different communities.

Management Layer

The management layer 1916 is responsible of maintaining the community in a transparent way to the application layer 1912. To manage the dynamic size of the community, each node has a role in maintaining the network.

According to one embodiment, each node is adapted to supervise the nodes listed in its neighbouring table in case one of neighbouring nodes disappears or disconnects from the network. When one of the neighbouring nodes disappears from the network, a node replacement rule or algorithm is applied and one of the nodes that supervised the disappeared neighbouring node is adapted to replace that node.

Depending on the replacement rule applied, according to one embodiment, it is the supervising node positioned at the highest ring level number that would replace a neighbouring node that disappears.

If a Parent leaves a ring, it will be replaced by another node of the same Ring.

Ring-Tree Topology Layer

The Ring-Tree Topology Layer 1514 (RTTL) can serve different types of client applications. This layer uses Transport Layer 1510 of the reference model OSI and offers services to the client application.

According to one embodiment, the RTTL allows a node to join a specific community that is identified by a community identifier. The node may use a command such as: "Join (CommunityID, ServerAddress)", where the parameter "CommunityID" is the community identifier and the parameter "ServerAddress" is the address of the server that manages the community.

According to another embodiment, the RTTL allows a node to get the IP address of another node for which the position ID in the community is known. The Address Resolution Message (ARM) messages are transparent to the Client Application layer 1512. So users of Client Application 1512 don't need to implement ARM protocol. The node may use a command such as: "ARM (CommunityID, ID)", where the parameter "CommunityID" is the community identifier and the parameter "ID" is the position of the node within the community. A node can use such a command to send a message to a particular destination node when the IP address of the destination node is unknown to the node. A node can also

use such a command to update its list of neighbouring nodes for which the node is supposed to store the IP address.

According to another embodiment, the RTTL allows a node to send a unicast message to a node. The node may use a command such as: "Send_ID_Unicast (CommunityID, ID, message)", where the parameter "CommunityID" is the community identifier, the parameter "ID" is the position of the node within the community and the parameter "message" is the message that is to be sent to the node.

According to another embodiment, the RTTL allows a node to send a multicast message to a group of nodes. The nodes may use a command such as: "SendMulticast (CommunityID, destination, message)", where the parameter "CommunityID" is the community identifier, the parameter "destination" is the position of the parent node of which the group of nodes are the descendants and the parameter "message" is the message that is to be sent to the group of nodes.

According to another embodiment, the RTTL allows a node to send a broadcast to all the nodes of a community. The node may use a command such as: "SendBroadcast (CommunityID, message)", where the parameter "CommunityID" is the community identifier and the parameter "message" is the message that is to be sent to all the nodes of the community.

According to another embodiment, the RTTL allows a node to leave a community. The node may use a command such as: "Leave (CommunityID)", where the parameter "CommunityID" is the community identifier. It should be understood that a node that is part of more than one community could leave one community and still remain part of the other communities.

According to another embodiment, the RTTL allows a managing server or node to receive a message. The server or node may use a command such as "Receive (CommunityID)", where the parameter "CommunityID" is the community identifier.

According to another embodiment, the RTTL allows a managing server or a node to check on the total number of nodes in a community. The managing server or node may use a command such as: "NodesStat (CommunityID)", where the parameter "CommunityID" is the community identifier. This command returns the total number of participating nodes in a specific community. A skilled person will understand that it is also possible for this

command to return other kinds of related information such as the maximum number of p levels and the maximum number of nodes n per level in the network, or a table of nodes for each of the p levels of the network for indicating the number of nodes that are present at each p level.

According to yet another embodiment, the managing server is adapted to authenticate new nodes, to provide a list of active nodes, to add new nodes or to reinsert an orphan node.

A skilled person will understand that a single managing server may be adapted to manage a plurality of communities or applications.

Functional Overview

Transparently to Client Application, messages can be transmitted from one node to another. There are two types of messages: management messages and data messages. Management messages are messages that are required for building and maintaining the network of nodes or community. Data messages are messages that allow communicating information between nodes. Depending of their use, both management messages and data messages can be sent as Unicast, Multicast or Broadcast.

According to one embodiment, a message is sent from one node to a neighbouring node by following a ring path in any direction (i.e. clockwise direction or counter clockwise direction). In the case of a Unicast message, the message is sent from one neighbouring node to another until the message reaches the destination node. In the case of a Multicast message, the message is sent from one neighbouring node to another until the message has reached all destination nodes. In the case of a Broadcast message, the message is sent from one neighbouring node to another until the message has reached all nodes of the community.

According to an alternate embodiment, a message is sent directly from a first node to a second node. The first node can only send a message directly to the second node if that second node is a neighbouring node and is identified in the neighbouring table.

According to one alternative, the system can use the transport layer 1910 for sending the message. By using the IP address corresponding to the second node as stored in the neighbouring table, the first node can directly send the message to the second node. In one example, a Unicast communication uses a Ring Tree Internet Protocol (RTIP) for sending a message based on an IP address of the destination node.

According to another alternative, the system can use the RTTL 1914 for sending the message. By using the ID corresponding to the second node as stored in the neighbouring table, the first node can directly send the message to the second node. In one example, a Unicast communication uses a Ring Tree Identifier (RTID) for sending a message based on an identifier of the destination node.

Unicast Data Transfer

According to one embodiment, a node is adapted to send a message to another node using an RTID communication. The message is sent to a specific node position in the community irrespective of the IP address of the node occupying the position.

According to an alternate embodiment, a node is adapted to send a message to another node using an RTIP communication. The message is sent using an IP address of the destination node, irrespective of the node position or ID in the community. If a destination node changes its position in the community, RTIP messages will still reach the node whatever its new position contrary to RTID messages.

When it is required to send a message to a node irrespective of the node position in the community, the RTIP communication can be used. When the message is specific to a node position in the community, the RTID communication can be used.

According to one embodiment, to send an RTIP message, the Client Application uses primitive: Send_IP_Unicast (CommunityID, IP, message) . To send an RTID message, the Client Application uses primitive: Send_ID_Unicast(CommunityID, ID, message). It is possible that in one Client Application, the creation of nodes and communication between different nodes are done across various communities, therefore in this embodiment the CommunityID is specified in the Unicast transfer.

A skilled person will understand that in a Client Application having only one community, the CommunityID parameter in the Unicast transfer can be omitted.

Multicast Data Transfer

As presented in Figure 20, it is possible to send a message to a group of nodes by selecting the Parent of the group as a message destination. In this case, a source node addresses a message to a Parent and specifies that the message is a multicast message, the Parent node then diffuses (Broadcast Descendant) the message as a multicast to its children on all levels but not their neighbours of the same level. Children who are also Parents of other levels

diffuse the message as a multicast to their neighbours and to all their children and so on. This way, the multicast message can reach the Parent destination and all its descendants until leaf level.

In Figure 20, there is illustrated a sending of multicast message according to one embodiment. Node 0-1-2-0 sends a multicast message to Parent 2-0-0-0. The Parent 2-0-0-0 receives the multicast message and in response re-transmits the multicast message to his children of all three levels. Node 0-1-2-0 also sends a multicast message to Parent 1-1-0-0. The Parent 1-1-0-0 receives the multicast message and in response re-transmits the multicast message to his children of all two levels.

According to one embodiment, the Client Application uses primitive: SendMulticast (CommunityID, destination, message) to send a multicast message. In this case, the field destination represents the Parent RTID of the Multicast Group.

Broadcasting

It is also possible for a node to send a message to all nodes of the community by sending a broadcast message. As presented in Figure 21, when a message is to be broadcasted, a node sends the message to its direct neighbours of all levels, in a predetermined direction (i.e. clockwise or counter clockwise). When a node that is a Parent receives the message, it retransmits the message to its direct neighbour of the same level and also to its direct neighbours of other levels (i.e. children nodes). Every node that receives the message, in turn retransmits the message to its direct neighbour or neighbours always in the same predetermined direction until all nodes in the community have received the message.

According to one embodiment, in each ring, the broadcasted message does come back to the node that initiated the messaging in the ring. The node that initiated a message in a ring adds its ID, so nodes know when the transfer has to stop. This is a way, for the node that initiated the sending of the message in a ring, to verify that the message reached all the nodes of the ring. So it will stop the transfer preventing the message to run forever.

According to one embodiment, if in a ring the node that initiated the messaging disappears before receiving the message, its precedent node will stop the transfer. The precedent node supervises a node presence and if the node disappears, the precedent node stops the message from being transferred infinitely in the ring.

It shall be understood that a parent node is a precedent node to all of its children nodes and that a child node is a precedent node to another child node of the same ring level, according to the message transfer direction. According to one embodiment, when messages are transferred within a ring in a clockwise direction, a first child node that is a precedent node to a second child node is located, on the same ring, immediately next to second node in an anti-clockwise direction. In this case, the first node considers itself as a precedent node if, in its neighbourhood table, there is a node at the right of its own ID. If this node is situated at the end of a line in the neighbourhood table this node is the precedent node of the parent node of this ring.

A skilled person will understand that there are other known ways of preventing a broadcast message to run forever in the network. One well known mechanism is the use of TTL (Time To Live). For the Ring-Tree network, the TTL is defined in number of hops according to the following equation: $(n-1)(2p-1)$. Actually, to reach all nodes, a circular broadcast message takes a maximum average time of $(n-1)(2p-1)t$ with t is the average transmission time between two nodes. To send a message between the two distant nodes in the community when the network is full of nodes i.e. worst case for the broadcast transmission time, it takes a maximum of $(n-1)(2p-1)$ hops. This defines the TTL value.

According to yet another embodiment, a broadcast message is to send as a multicast message to the Root Parent. The message will reach all nodes of the community because the Root Parent will send the message as a multicast to all its children including Root nodes. The children will send the message as a multicast to their children and so on until reaching leaf nodes (see Multicast Data Transfer section).

According to yet another embodiment, a broadcast message is directly sent to a plurality of nodes. A sending node sends a message directly to all the nodes of the same ring and also to all the descendant nodes. If the sending node is situated as a child in ring level i, the sending node sends the direct broadcast message to all the nodes in ring level i and also to all the nodes in ring level i+1, and further on until ring level p-1. When a receiving node receives a direct broadcast message, the receiving node sends the message to all the nodes in all the rings to which it belongs except to the nodes of the ring from which it received the message. With this direct broadcast, a message may reach all the nodes in n*p-1 transmission time.

To send a broadcast message, Client Application can use primitive: SendBroadcast(CommunityID, message). Broadcast message can be done by any node even if the source doesn't belong to the community. This option can be disabled for security reason if necessary.

Aggregation of Messages

According to one embodiment, in order to unicast, multicast or broadcast a plurality of messages at once and each originating from different nodes, a parent node is adapted to aggregate into a single message the plurality of messages coming from its descendants and then send out the single message.

Community Management

According to one embodiment, a client application may use the command Join (CommunityID, ServerAddress) in order to join a network. The CommunityID must be recognized by the server. So, prior to using this primitive, the CommunityID must be defined on the server.

A Ring-Tree server administrator may add new communities on the server. According to one embodiment, the Join function contacts a server that returns a list of active nodes of the community. In a transparent way to the Client application, the Join function contacts one or many active nodes in order to join the community. Once the node is attached to the network, the network management will assure that the node will stay connected to the network despite the dynamic size change due to joining and leaving of nodes in the community.

According to one embodiment, each participating node can be a source or a destination and all nodes are routers. The maintenance of the system is assured by all participating nodes. Depending on the location of a node (i.e. parent or child), the role of the node may vary. For instance a parent node may be responsible of routing a greater number of messages than would a child node. Despite that some nodes can be charged in communication or in maintaining the network more than others, any node can assume any role in the community

Node Identification

Each node has a unique ID that represents its position in the whole topology. According to one embodiment, the length of the ID is p. So the ID contains p fields separated by dashes "-", p is the maximum number of levels of the network. These fields, from left to right, represent level 0 to p-1 respectively. Value of each field vary from 0 to n-1, where n is the

maximum number of nodes in a ring. Number "0" represents a parent node, other numbers increments in clockwise direction, represent child nodes. So each node has a position number in its ring and its ID represents its position in the network.

As each node is a child in only one level, this level defines the level of the node. For example: in a topology $T(n=5, p=10)$ a node of level 4 means that this node is a child on level 4 and it is a parent on all levels higher than 4. For levels under 4, it inherits all the position of its parent.

Examine, from right to left, the values of an ID of a given node: while there is zeros it means that the node is parent of those levels. The first value different from zero determine the level of the node and it actually means that the node is a child in this level and the value specifies its position in its ring. Values that come after determine the positions of its parent, its grandparent, its great grand parent, etc until the Root parent.

When a newcomer node joins the network, its parent gives it an ID indicating the ring level where it is placed as a child, the position in the ring and the parent's position so the newcomer will know where it is situated in the whole network. The newcomer completes its ID by zeros specifying that it's a parent for higher levels. For example: in a topology $T(n=3, p=5)$ when a newcomer receives the ID $[0-2-1]$ the node will complete this ID by zeros, the ID becomes: $[0-2-1-0-0]$. By examining this ID we conclude that this node belong to a community with p=5 levels (because there is 5 fields in the ID) numbered from 0 to 4. It also means that the node is a parent in the 4th and the 3rd levels because the fields in position p-1 and p-2 have the values of "0". The node level is 2, because it is a child in a ring of level 2 (level number p-3), it is placed on position 1 because the field on position 2 has the value of "1", its parent is placed in 3rd position in level 1 (level number p-4) and its Grandparent is the Root Parent (it has number "0" of root level).

Neighborhood Table

Each node retains limited information about the network. So a given node maintains a table containing information about $n \times p$ members. Those members are related to the node and the table contains IP addresses and other information about them. Actually, a given node have information about all its children (i.e. on all levels), parent, uncles, grand parent, grand uncles, great grandparent and great grand uncles until the nodes at the root level. They don't

hold information about the children of their: uncles, grand parent, grand uncles, great grand parent, great grand uncles, and so on until the root nodes. So the root nodes themselves are known but not necessarily all their children.

All nodes hold information about the root nodes but root nodes don't hold information about all the other nodes. A root node can be seen as a famous person known by everyone but this famous person don't necessarily know everyone. If the position of the nodes is closer to the root, the node is more important and it will be known by more nodes.

When a newcomer node joins the network, its parent gives him an ID indicating its position in the network. It also gives him part of its neighbourhood table. Actually, the newcomer share the lower part of the table with its parent i.e. it shares the same information from level number 0 (root) to level number (i-1) where i is the node's level. The rest of the table will be filled by information about future children of the newcomer. But at this point, this part of the table is left empty because the node doesn't have children yet. At leaf level i.e. level number $p-1$, nodes of the same ring have similar tables because a leaf node cannot have children so it will inherit the whole table from its parent.

Presented in Figure 22A, is a network $T(n=3, p=3)$, according to one embodiment. This network can have a maximum of $N = n^p = 3^3 = 27$ nodes, but it only has 22 nodes, there are still some available places, this is why some tables are not fully filled. In this network, there is one ring on level number 0, three rings on level number 1 and nine leaf rings i.e. on the leaf level number 2. At leaf level, nodes of one ring share the same table, that's why the table is represented only once for each leaf ring. So we have a total of nine different tables in this example. Each field in the table can have other information than the IP address, such as an identifier of the node, a ring level indicator or a ring position indicator. Here, only the IP addresses are stored in the table, the ring level indicator and position indicator are determined by the IP addresses' position in the table.

According to one embodiment, there is a local node identifier table corresponding to the neighboring table. The identifier table is created locally at each node according to the corresponding neighboring table and holds corresponding node identifiers such as IP addresses.

Because, root nodes are great grandparents and great grand uncles to all leaf nodes of the network, all nine tables share the same first line which is contain information about those root nodes. Consider node number 0-1-0. This node shares the same first two lines with node number 0-0-0 and only the 3rd line is different, because, the 3rd line contains information about the children of node number 0-1-0 where node number 0-0-0 doesn't know. Actually, a given children knows its uncles, grand uncles, grandparents, all great grand uncles and all great grandparents, but they doesn't know this children. This means, in the example below, node 0-1-1 knows or hold information about its uncle 0-2-0 but 0-2-0 doesn't know 0-1-1. Node 0-0-0 knows all its direct children 0-1-0, 0-2-0, 0-0-1 and 0-0-2, but it doesn't know its grandson 0-1-1, 0-1-2 and 0-2-1.

The neighbourhood table is used for Unicast, Multicast and Broadcast transfer. It is also used to maintain the network. The table must be updated each time a node join the network or disappear from it.

According to one embodiment the neighbouring table size properties (i.e. value of n and p) are set according to a destined application of the ring tree. In applications that are mostly used for broadcasting messages a neighbouring table size allowing to perform robust broadcast may be desirable. In applications that require a shorter broadcasting time a neighbouring table size allowing to perform timely broadcasting may be desirable. In applications that have nodes with a limited amount of memory, a reduced neighbouring table size may be desirable.

Mutual and Non-Mutual Knowledge

According to one embodiment as presented in Figure 22B, a neighboring table 2200 has a node information at position (i, j) presented as "info(ξij)". All node information at positions indicated by section 2202 is shared by all the nodes of ring level i. Node information at positions indicated by section 2204 is node information of the descendants of node ξij which are positioned at higher ring levels (i.e. ring levels i+1 to p-1). Node information in column 2206 contain node information of the parent nodes of ξij at ring levels 0 to i. The column 2206 also contains node information of ξij at ring levels i+1 to p-1, since ξij is a parent node for those levels.

In the neighboring table 2200 of Figure 22B, there is shown that there is mutual knowledge between node ξij and the nodes that appear in all ring levels from i to p-1. This means that

both the node ξij and a node of the table at ring level i to p-1 keep information about each other. However, there is non-mutual knowledge between ξij and the nodes that appears in all levels from 0 to i-1. Since node ξij keeps information about those nodes but those nodes do not keep any information about ξij.

Precedent and Next Node

Further presented in Figure 22B, there a precedent node Pk and a next node Sk in which k varies between 0 and p-1. The precedent of node ξij is located at position j-1 of ring level i, that is the first node located on her left in the neighboring table and indicated as Pi. If ξij was the first node of the line i, the precedent would be the node situated at the end of the same ring since every line of the neighboring table corresponds to a ring of the network. The next node of node ξij is the first node located on her right side in the table and indicated as Si. If ξij was the last node of the line i, the next node would be the first node in the same ring i.e. the node situated at position j=0 of the same line i (i.e. parent node).

Note that if the node ξ is alone in a ring, it would not have a precedent nor next node in the same ring. However, if there is one other node than ξ in a ring, that node would be both precedent and next of node ξ.

Let j be the position of node ξ and k the position of precedent node, k belongs to the set of node IDs existing in the same ring as ξ, k is determined as follow:

If $j>0$ $\min(j-k)$ where $k<j$

If $j= 0$ $\max(k)$, where $k<n$

Let j be the position of node ξ and u the position of next node, u belongs to the set of node IDs existing in the same ring as ξ, u is determined as follow:

If $j<\max(u)$ $\min(u-j)$ where $u>j$

If $j=\max(u)$ $u=0$

Therefore, in each ring a node can have a maximum of p-i precedent and p-i next nodes.

Node Level

A node ξ is a child in only one level. This level defines the level of the node. In this case level i defines the level of node ξij.

Protocol Data Unit (PDU)

There are two types of PDU: Management PDU and Data Transfer PDU. The Management PDU type is used to exchange all necessary data for the distributed management system. The

data transfer PDU type is used to send all necessary data for the application using the system. According to one embodiment, here is a list of data transfer PDU messages: unicast, multicast, broadcast and ARM. According to another embodiment, here is a list of management PDU messages: Status (to ensure that a neighbour is still connected. It is necessary to detect unexpected failure), Join (to add a node to the community), Leave (to quit a node from the community), Link (to insert a node in a ring) and Multicast.

Unicast Data Transfer Specification

Prior of sending a message via RTIP Unicast, similarly to ARP message, a node must send an address resolution message (ARM) to get the IP of the node that occupy a geometrical position in the network. This means that the source must at least knows the ID of the destination and send an ARM message. Once source node has destination's IP node, it can exchange messages directly using traditional IP protocol without passing by the Ring-Tree network for routing. Source node may know the IP of destination without the need of sending an ARM message. For more details see ARM protocol specification section.

To send RTID Unicast message, the source node may have information about the destination node in its neighbourhood table. If not, the source compares the destination's ID with information that hold in its neighbourhood table and then it will choose the nearest node to the destination to act as a router. Messages then will transit via this node. If the selected node doesn't have the IP of the destination, it will choose the nearest node to the destination to act as a second router and so on until reaching the destination. Contrary to ARM message, IP of the destination node is not returned to the source. Communication between source and destination must pass by the chosen routers. If not an IP Unicast can be used. With this kind of communication if the destination disappears or leave the network, its substitute will receive the messages.

To choose the nearest node to the destination, a source compare the destination's ID with the IDs present in its neighbourhood table. Comparing from left to right, the node that have the more values in common with the destination's ID will be the nearest node to the destination i.e. it will be the selected router.

ARM Protocol Specification

ARM messages are necessary in case we need to know the IP of a given node when the ID is known. Contrary to ARP, ARM message is not a broadcast message. The source node knows

the position of the desired node, it may have its IP directly from its neighbourhood table. If not, it can send an ARM message to the nearest node that knows the destination node's IP. But before that, it adds its IP to the ARM message so when a node that retain the desired IP receive the request, it will return the desired IP directly to the source.

The same algorithm for ID Unicast can be used to find the nearest node but the difference here is that the nearest node will simply return the IP if it has it or re-route the ARM message to the nearest node to destination that exist in its neighbourhood table and son on until reaching a node that knows the destination's IP. Once a node has the destination's IP it will send it directly to the node that initiated the ARM message.

Multicast Data Transfer Specification

There is two type of Multicast in Ring-Tree Network: Ring-Tree IP (RTIP) Multicast and Ring-Tree ID (RTID) Multicast. With RTIP, the source sends an ARM message to get the IP of the destination Parent. And then, it will send multicast messages directly to the destination. In the message it is specified that the message is a Multicast. Once the Parent receives the message it will diffuse it as a multicast message to all its children on all levels. Children that are Parents will diffuse it to all their children and so on until reaching leaf nodes. With RTID, the message may be routed via many nodes before arriving to destination as in the case of ID Unicast described in the Unicast data transfer specification section. In RTID, the multicast Group will continue receiving multicast messages even if the Parent disappears or leaves the network.

Broadcasting Data Transfer Specification

To send a broadcast message a node sends a message to its neighbour and all its children. When the message arrives to its parent, it sends it to its neighbour and all its children. So the message will reach the Grandparent of the source as well as all its children. When the Grandparent receives the broadcast, it will send it to its neighbour and all its children and so on until reaching all nodes of the network. In each ring, the node that initiated the message adds its ID so others nodes know when to stop resending it. To know the IP of neighbour and all children, a node uses its neighbourhood table.

Management Protocol Specification

Overview

According to one embodiment, the management system is distributed i.e. each node is responsible to manage the system. Status update messages are sent by all nodes at regular intervals, this allows other nodes to know that their neighbours still alive.

To reduce the cost in terms of bandwidth on each node, it is necessary to reduce the size of management messages and the overhead.

According to another embodiment, each parent node sends a status update message to all of its children at regular intervals. This allows the children nodes to know that their parent is still alive.

Network Formation

Topology Bootstrap

When a newcomer wishes to join a community, it contacts a server or a Ring-Tree Server (RT-Server) in order to get a list of active node members. According to one embodiment, the newcomer is adapted to send a join request to any member of the list. However, as each Ring can only have a maximum of n members, the Ring maybe full and the newcomer must try to join another member of the list.

According to another embodiment, the list of active node members includes only active node members of non-full Rings. The list can be updated on the RT-Server by parents that still have available places in their rings.

The RT-Server may have additional responsibilities. According to one embodiment, the RT-Server can validate if a node is allowed to join a community before supplying the list of active nodes. Other security functions can be added at the server level.

According to yet another embodiment, the RT-Server can also supply statistical information about the nodes such as the total number of registered nodes, the number of nodes online for each community, etc.

The newcomer is adapted to be inserted as a member in a free place of the selected non-full Ring. There is then assigned to the newcomer a position that serves as an ID number in the Ring. It will inherit part of the ID of its parent and the rest of the ID fields will be filled with zeros. If the newcomer occupies a position other than a leaf node (i.e. it can have children), the newcomer is then adapted to send its IP to the server to be part of the Non Full Active Member List for in turn receiving newcomers.

According to yet another embodiment, when for a given parent, rings of all levels are full, the parent informs the server that it is full so its ID is retreated from the Non Full Active Member List.

It is understood that all members don't need to know about the newcomer. The newcomer will be known by only the members of its own ring and all children of its neighbour. So in the best case scenario only 1 member will be updated. But if some parent's neighbourhood table are modified, they need to update their children's neighbourhood tables.

It is further understood that the newcomer is adapted to share the same lower part of the neighbourhood table with nodes of the same Ring where inserted. Besides serving for routing, this table serves the newcomer, in case of failure, to contact one of the parents node to stay connected to the network.

Topology Recovery

For reducing center point failure problems such as commonly found in tree based architecture, according to one embodiment, the Ring-Tree network is partitioned if all root nodes disappear at the same time. The probability for that to happen is α^n, with α the probability for a node to disappear and n is the number of nodes on root level.

A skilled person will understand that in a Ring-Tree network, the more there are root level nodes, the more the network is reliable.

Node Failure Recovery

Presented in Figure 23A is a graphical representation of a single node failure in a ring topology, according to the prior art. If a non-Parent node disappears , its precedent neighbour stops receiving "status update" message. Then its precedent node will simply contact the next node to reform the ring. It will update the neighbourhood table and send it by multicast to all affected nodes. No additional overhead is required and the ring is then ready to accept a new member replacing the disappeared node. If there is no other node in the ring, the node which left will be the only node in the ring.

Presented in Figure 23B is a graphical representation of a multi-node failure inside a ring, according to the prior art. If a multiple non-Parent node disappears, a precedent neighbour will stop receiving "status update" message. Then it will contact the next node until reaching a node that will reply and reform the ring. It will update the neighbourhood table and send it by multicast to all affected nodes. No additional overhead is required and the ring is then

ready to accept a new member replacing the disappeared nodes. If there is no other node in the ring, the node which left will be the only node in the ring.

Presented in Figure 24, there is a graphical representation of a parent node failure, according to one embodiment. When a parent node disappears, this will be detected by precedent nodes on all ring levels related to the parent node. The precedent becomes an active orphan (i.e. a node that has lost its parent); it will send a multicast message to all the affected nodes asking them to change their state to passive orphan (i.e. a node that has lost its parent or a parent at a lower level).

In Figure 24, the precedent of the disappeared parent node is an active orphan node and the children of the disappeared parent node are passive orphan nodes that are in a wait state.

Presented in Figure 25, there is a graphical representation of a network insertion sequence, according to one embodiment. If a parent node of ring level 0 disappears, a child node on ring level 1 (referred herein after as the chosen precedent node) is inserted in the ring level 0. The chosen precedent node has in its neighbourhood table the IP address of its uncles (i.e. nodes on level 0), and is adapted to contact the uncles in order to request an insertion at level 0.

According to one embodiment, the lower level ring (ex.: ring level 0) that lost a node (i.e. disappeared Parent node to ring level 1) will not accept newcomer after a timeout is passed in order to give priority to the chosen precedent node to be inserted in the ring. If the chosen precedent node doesn't succeed to join the lower level, it will try with another lower level until succeeding to join.

According to one embodiment, if after trying to join all lower levels including root level, it will consider itself partitioned and inform all its descendants of the partitioned status. When in the partitioned status, the descendants are adapted to re-contact the server to re-join individually.

If the Parent node is a Parent for many levels then all the rings will run the same algorithm to join a lower level. When a parent to an X number of levels disappears, the X rings will be disconnected from the network. In order to rejoin the network, a chosen precedent node of each disconnected ring will contact at least one lower level in order to be reconnected to the network separately.

In Figure 26 there is a flowchart presenting the steps to resolve node failures according to one embodiment. At step 2602, there is discovered that a node is not answering and that the node disappeared. At step 2604, the next node according to the neighbouring table is contacted (i.e. following the spiral representation of Figure 27). At step 2606, there is verified if the contacted node is present in the network and if that contacted node is not present, there is verified if there are other nodes in the same ring. If there are other nodes in the same ring, a next node of the ring is contacted according to the neighbouring table as in step 2604. If there are no other nodes in the same ring, at step 2608, there is verified if the contacted node is present. If the contacted node is present, at step 2610, the ring is reformed.
If the contacted node is not present, at step 2612, a node of a ring at a lower level is contacted according to the neighbouring table (i.e. following the spiral order as presented in Figure 27). At step 2614, there is verified if the contacted node is present and if that contacted node is not present, there is verified if there are nodes in a ring at a lower level. If there are nodes in the ring of lower level, a next node of that ring is contacted according to the neighbouring table as in step 2612, and so on until reaching the lowest level (i.e. root ring).
According to yet another embodiment, there is a precedent node that detects that a next node is disconnected. The precedent node informs the central managing server that the next node is disconnected. The central managing server manages (by updating the neighborhood table, by managing reinsertion, etc.) the precedent node according to its position in the network topology and attempt to manage its recovery if required.

Node Join

Presented in Figure 28A, is a state diagram of a node representing a node join to the network, according to one embodiment. When in the not connected state, the node contacts the server to get a list of active nodes or the addresses of a set of appropriate active nodes. Based on a predetermined selection criterion such as a ping test, the server chooses an active node or a set of appropriate active nodes for allowing the node to join the network via the active node or via one node from the set. Once the node is connected to the network, the node is able to communicate with other nodes of the network through a unicast, multicast or a broadcast of information.
Once the node joins the network it receives, from its parent, an ID representing its position in the network at level i (where i is between 0 and p-1); it also receives part of its parent

neighborhood table (i.e. lower part of the parent table from level 0 to level i, i is the level of the node).

When the node drops out of the network, the node may attempt to re-join the network. According to one embodiment, the node is adapted to contact a selected node of lower level according to the neighbouring table (i.e. following the spiral order as presented in Figure 27) for joining the network via the selected node.

According to yet another embodiment, as presented in state diagram of Figure 28B, when an attempt to insert the active orphan (i.e. the precedent of the disappeared parent node) fails, the active orphan sends a message to all the passive orphan (i.e. the children and children of higher levels of the disappeared parent node) to inform them that they are disconnected from the network. The passive orphan may then try to connect to the network on their own or pass to a non-connected state.

Topology Expansion

According to one embodiment, the RT topology T is characterized by its n and p parameters $T_{n,p}$. It is possible to modify these parameters dynamically to pass from topology $T_{n,p}$ to $T_{n+i,p+j}$.

In order to expand the size of the network all nodes must be informed by a trusted and authorized source such as the RT server. The RT server already has this information and it must be informed by any modification of the n and p values. Recall that the RT server informs each newcomer node by the value of these parameters as those values can be different for different communities.

Once a node is informed by the new values of n and p parameters it is adapted to expand the size of the neighbourhood table in order to allow more nodes to enter the network.

The topology expansion is easier than the topology reduction. Because when the size of n or p is decreased it may cause some nodes to leave the network and try to relocate later.

Node Modules

Presented in Figure 29 is a block diagram representing modules that may be found in a node or network node device of the system 2900 described above. The network node device has a network connection interface 2902 to communicate with other nodes of the system 2900. The interface 2902 is related to a network address such as an IP address.

The network node device further has a registry 2904 of connectable node devices. The registry 2904 is adapted to store node device network address and identifiers each representing a node device of the system or community 2900. The identifier is indicative of the ring network level and ring network position of the corresponding node device according to the overlay network topology system 2900.

The network node device further has a registry update manager 2906 adapted to modify the registry 2904 when a change in the community occurs due to drops, adds and moves of node devices that affect the registry 2904.

The network node device further has a forwarding message manager 2908. The forwarding message manager 2908 is adapted to receive an incoming message via the network connection interface 2902 and to handle the incoming message in accordance with a content of the incoming message and the registry 2904.

When the network node device is a parent node the forwarding message manager 2908 is adapted to send an outgoing message according to the incoming message to a plurality of node devices of the registry 2904, each having a node position on a different one of the rings that are linked by the network node device.

When the network node device is not a parent node (i.e. child node only) the forwarding message manager 2908 is adapted to send the outgoing message according to the incoming message to another node device of the same ring network level according to the registry 2904.

According to one embodiment, the network node device further has a device application 2910 adapted to add packet data to the incoming message or retrieve packet data from the incoming message. The packet data could be indicative of a variety of information such as an electrical consumption quota, a temperature table of the other nodes, a battery charge level, an electrical production level of a supply or even a multimedia file to be shared.

According to yet another embodiment, the network node device has a message sending manager 3002, as presented in Figure 30. The message sending manager 3002 is adapted to send a message by unicast, multicast or broadcast. When unicasting, the message is sent to another node device of the community according to the registry. When multicasting, the message is sent to a group of node devices, each of the group of node devices having a same parent node (or same parent node at lower levels) at a determined ring network level. In one

example, when multicasting, the message is sent to a group of node devices, each of the group of node devices having a same parent node (or same parent node at lower levels) and being at a ring level that is lower than a predetermined ring level that is lower than p-1. In another example, when multicasting the message is sent successively from a descendent node to a parent node up to a level 0 parent. When broadcasting, the message is sent to all the node devices of the community.

Note that for the above, the n and p values may be determined according to an acceptable maximum number of hops required for a circular broadcast or a direct broadcast. The n and p values may also be determined according to an acceptable neighboring table size, the number of acceptable nodes affected from a disconnected parent node or an acceptable probability of losing all the nodes of a same ring.

What is claimed is:

CLAIMS

1. A load management method for controlling a plurality of distributed community loads with a plurality of controllers that are connectable to a network, each of the plurality of distributed community loads being electrically connectable to a power grid and being controllable by a corresponding one of the plurality of controllers, the method comprising: each of the plurality of controllers collecting a local data indicative of at least a local need parameter;each of the plurality of controllers receiving a community data indicative of at least one of a power grid supply parameter or a community need parameter of the plurality of distributed community loads; andeach of the plurality of controllers determining and setting a power consumption of a corresponding load according to the local data and the community data.
2. The method of claim 1, wherein each of the plurality of controllers setting the power consumption of a corresponding load to an active status when a value of the local need parameter is beyond an acceptable local threshold and a value of the power grid supply parameter is within an acceptable supply range.
3. The method of any one of claims 1 or 2, wherein each of the plurality of controllers setting the power consumption of a corresponding load to an active status when a

value of the local need parameter is beyond an acceptable local threshold and a value of the community need parameter is within an acceptable community threshold.

4. The method of any one of claims 1 to 3, wherein the acceptable community threshold is determined according to the power grid supply parameter.
5. The method of any one of claims 1 to 4, wherein the community need parameter is indicative of at least a power consumption need of each of the plurality of distributed community loads.
6. The method of any one of claims 1 to 5, wherein the power supply parameter is indicative of an available power within the power grid.
7. The method of any one of claims 1 to 6 wherein the power supply parameter is indicative of a power supply price based on a predetermined time period.
8. The method of any one of claims 1 to 7, wherein the local need parameter is indicative of a required power consumption of the corresponding one of the plurality of distributed community loads.
9. The method of any one of claims 1 to 8, wherein the plurality of distributed community loads are water heaters and the local data is indicative of at least a temperature.
10. The method of any one of claims 9, wherein the local data is indicative of a water column temperature profile.
11. A network node device for communicating with other node devices of a same community within an overlay network topology system having a plurality of node devices organized in a number of p hierarchically linked ring network levels, where p ranges from 0 to p-1, and the plurality of node devices of a same ring being interconnected in groups of n node devices, where n ranges from 0 to n-1, one of the n node devices being a parent node for linking the n node devices of a first ring to node devices of a second ring at another hierarchy by being a parent node to at least the second ring, the network node device comprising:

    a network connection interface to communicate over an interconnected network of node devices, the interface being related to a network address;

    a registry of connectable node devices adapted to store node device network address and identifiers each representing a node device of the community and

each being indicative of the ring network level and ring network position of the corresponding node device according to the overlay network topology;

a registry update manager adapted to modify the registry for handling a change in the community due to drops, adds and moves of node devices that affect the registry; and

a forwarding message manager adapted to receive an incoming message via the network connection interface and to handle the incoming message in accordance with a content of the incoming message and the registry, wherein when the network node device is a parent node the forwarding message manager is adapted to send an outgoing message according to the incoming message to a plurality of node devices of the registry, each having a node position on a different one of the rings that are linked by the network node device, when the network node device is not a parent node the forwarding message manager is adapted to send the outgoing message according to the incoming message to another node device of the same ring network level according to the registry.

12. The network node device of claim 11, further comprising a device application adapted to add data to the incoming message or retrieve data from the incoming message.
13. The network node device of anyone of claims 11 or 12, wherein the registry of connectable node devices is a table having n columns and p rows.
14. The network node device of anyone of claims 11 to 13, wherein the registry of connectable node devices has n x p positions each being adapted to store the node device network address and the identifier.
15. The network node device of claim 14, wherein the identifier is a position of the n x p positions and each of the n x p positions is indicative of a ring level and node position within the community.
16. The network node device of anyone of claims 11 to 15, wherein the registry of connectable node devices comprises a node device of the community that is a parent node to the network node device and a node device of the community that is a next neighbour.

17. The network node device of claim 6, wherein the next neighbour is a node device of a same ring as the network node device.
18. The network node device of anyone of claims 11 or 12, wherein the registry of connectable node devices comprises all node devices of a same ring as the network node device and a parent node to the network node device.
19. The network node device of anyone of claims 11 or 12, wherein the registry of connectable node devices comprises parent nodes of all ring network levels hierarchically corresponding to the network node device.
20. The network node device of anyone of claims 11 to 19, being adapted to communicate with a server, the server being adapted to determine a replacement node device when there is detected a node device drop in the community.
21. The network node device of anyone of claims 11 to 20, wherein the replacement node device is determined according to a ping test result.
22. The network node device of anyone of claims 11 to 21, wherein the server is a node device of the community.
23. The network node device of anyone of claims 12 to 19, wherein the device application is adapted to determine a replacement node device according to the registry when there is a detected a drop of a node device in the community.
24. The network node device of anyone of claims 12 to 19, wherein the device application is adapted to determine a new parent node device according to the registry when there is a detected a drop of the parent node device in the community.
25. The network node device of anyone of claims 12 to 19, wherein the device application is adapted to determine a replacement node device according to a second registry indicative of the registry of the dropped node device.
26. The network node device of anyone of claims 11 to 25, wherein the nodes of a same ring are determined according to a ping test.
27. The network node device of anyone of claims 11 to 26, wherein the plurality of node devices are organized in a number of p hierarchically linked ring network levels according to ping test results.

28. The network node device of anyone of claims 11 to 27, further comprising a message sending manager that is adapted to send a message by unicasting the message to another node device of the community according to the registry.
29. The network node device of anyone of claims 11 to 28, further comprising a message sending manager that is adapted to send a message by multicasting the message to a group of node devices, each of the group of node devices having a same parent node at a determined ring network level.
30. The network node device of anyone of claims 11 to 29, further comprising a message sending manager that is adapted to send a message by broadcasting the message to all the node devices of the community.
31. The network node device of anyone of claims 11 to 30, wherein the message is send to all the node devices of the community according to a directional ring path that is either counter clockwise or clockwise as defined by the registry of each node device by which the message passes.
32. An overlay network topology system comprising: A node device as defined in anyone of claims 11 to 19 and 23 to 31.
33. The overlay network topology system of claim 32, further comprising a server adapted to determine a replacement node device when there is detected a node device drop in the community.
34. The overlay network topology system of claim 33, wherein the replacement node device is determined according to a ping test result.
35. The overlay network topology system of anyone of claims 33 or 34, wherein the server is a node device of the community.

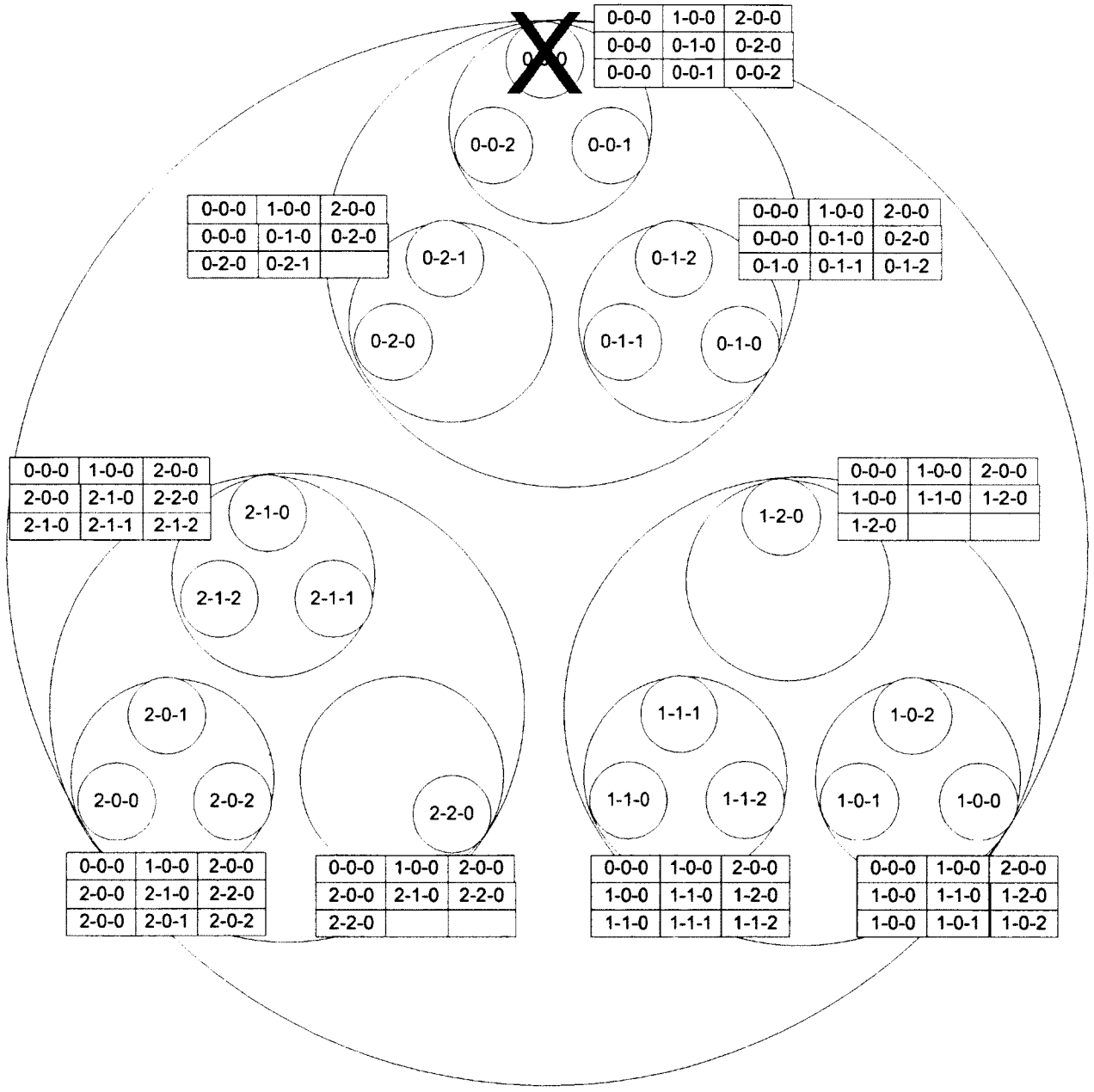
Abstract

A load management method for controlling a plurality of distributed community loads with a plurality of controllers. The controllers are connectable to a network and the distributed community loads are connectable to a power grid. The loads are controllable by a corresponding one of the controllers according to a local need and a power grid supply or a community need of the loads. A network node device adapted to communicate with other node devices of a same community within an overlay network topology system. The system has a plurality of node devices organized in a number of hierarchically linked rings. The plurality of node devices of a same ring is interconnected in groups of node devices. One of the node devices is a parent node for linking the node devices of a first ring to node devices of a second ring at another hierarchy by being a parent node to both first and second rings.

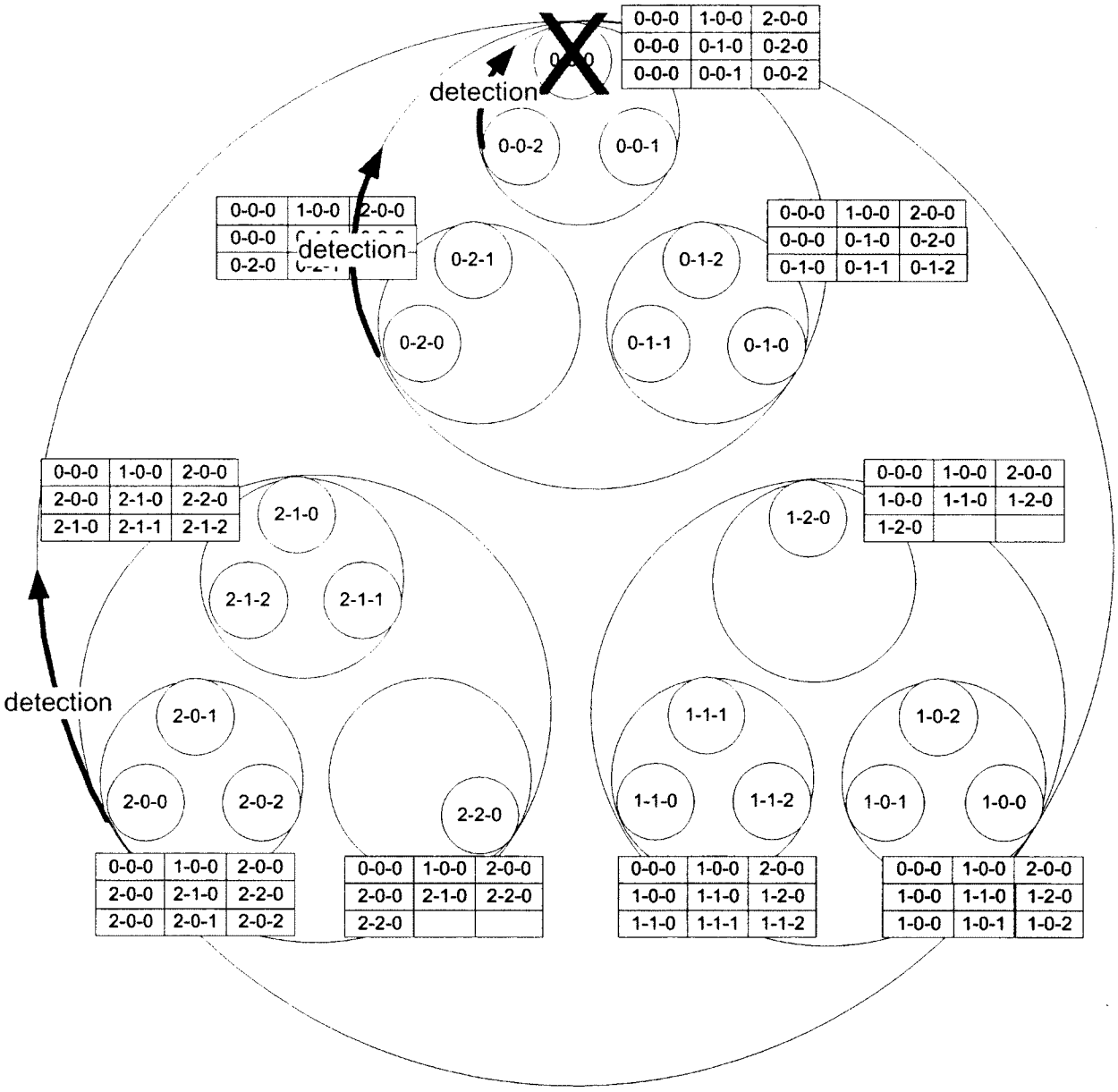
ANNEXE B – DIAGRAMMES DE L'ALGORITHME DE RECOUVREMENT

Node disappearance diagrams (example)

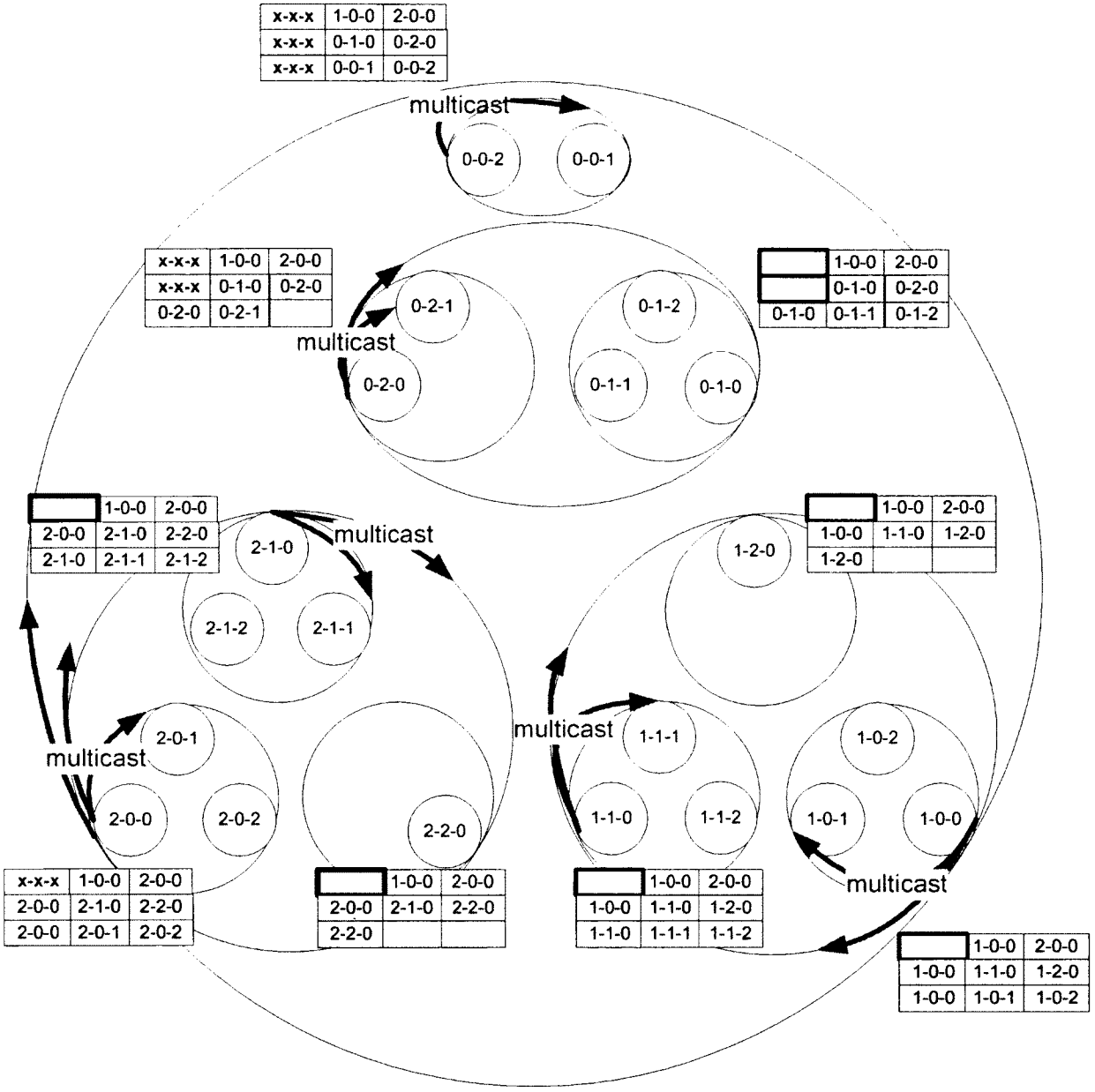
1- Disappearance: node 0-0-0 disappears



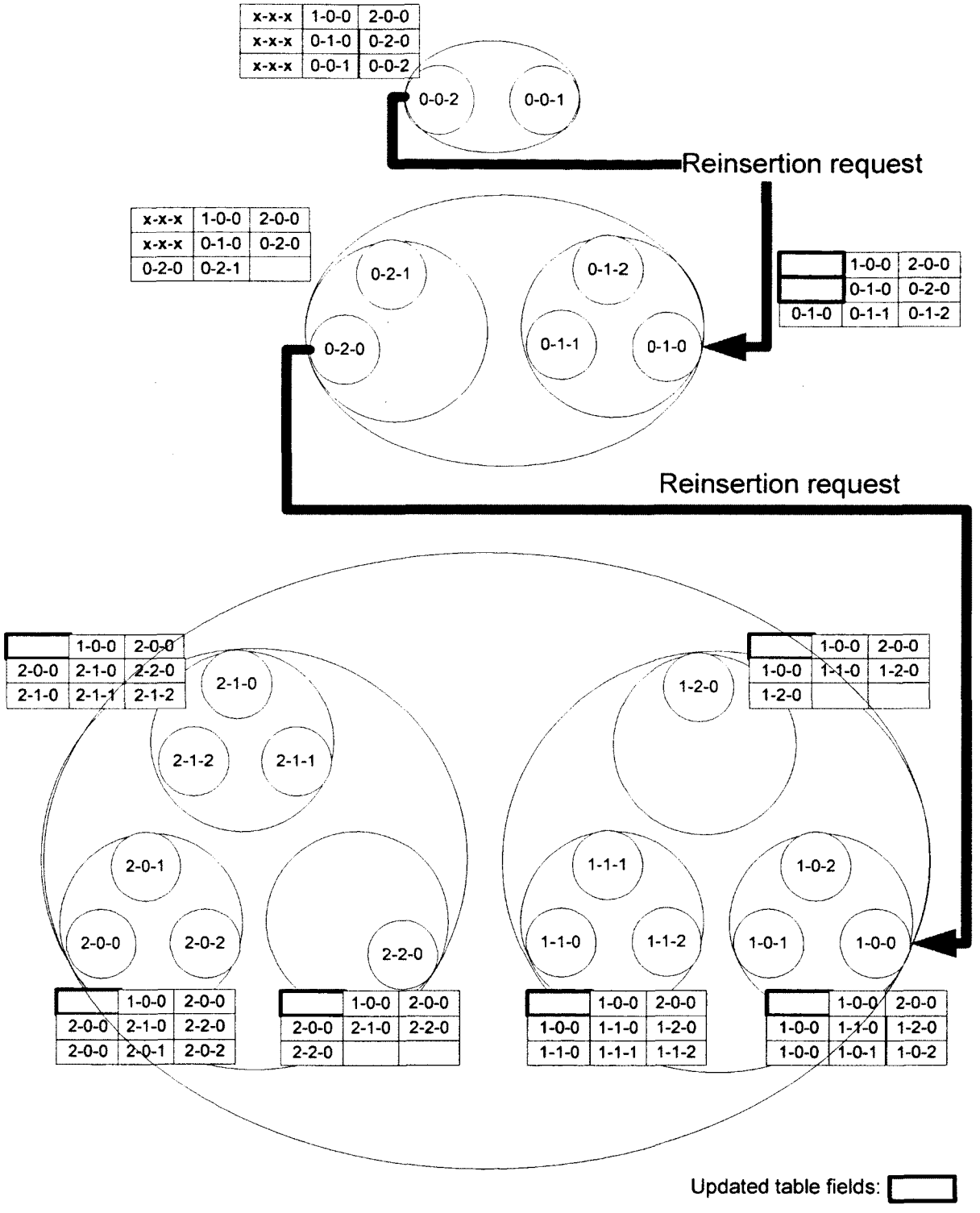
2- Detection: nodes 0-0-2, 0-2-0 and 2-0-0 detect that their next neighbor disappear. This detection is assured by "keepalive " messages sent continuously by every node to its next neighbor.



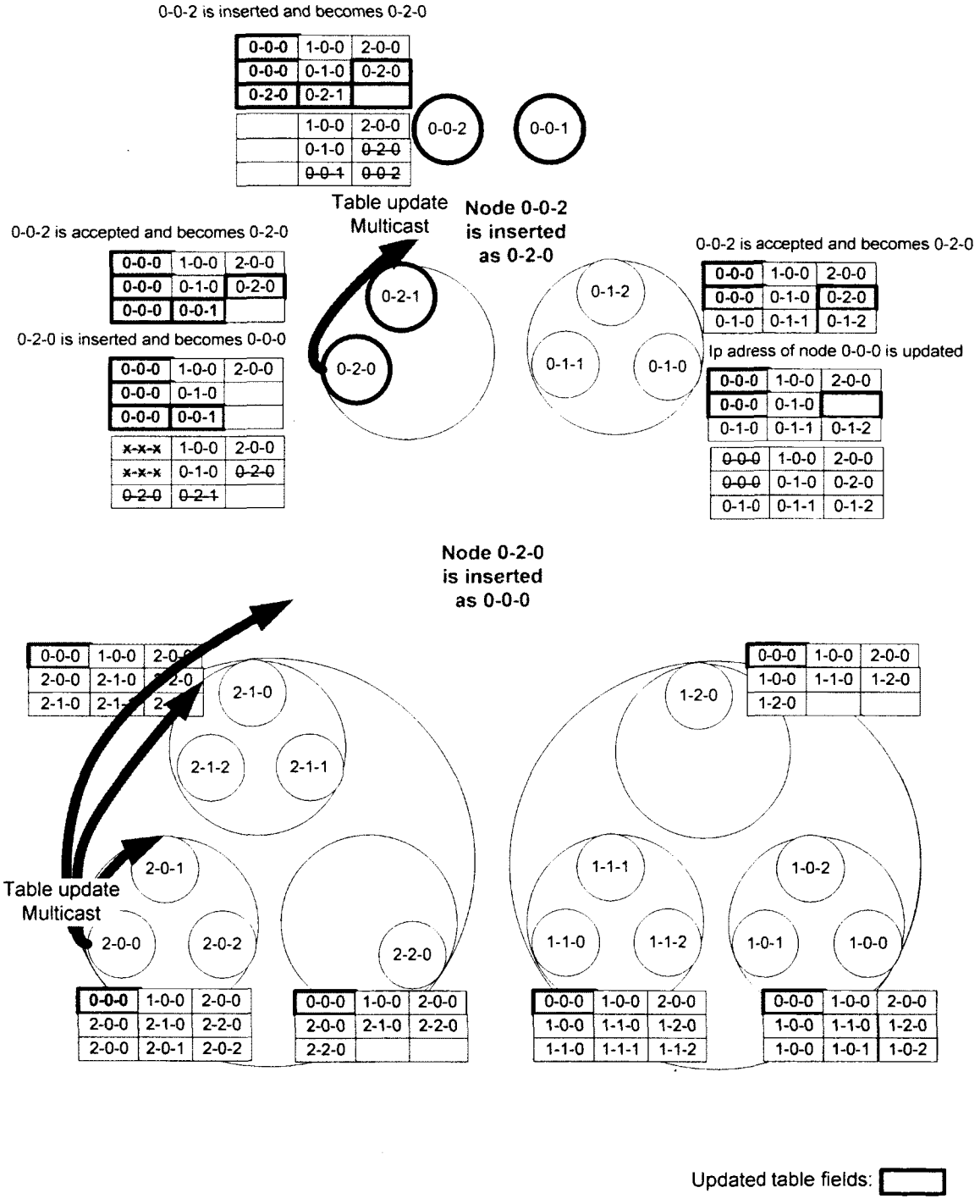
3- Nodes 0-0-2, 0-2-0 and 2-0-0 change their status from "connected" to "active orphan" and send a multicast message to set their descendants in « passive orphan » state.



4- Insertion: every orphan node sends a reinsertion request to join one lower level except the orphan node of the lowest level. If the node of the next lower level didn't satisfy the reinsertion request, the orphan node send an Insertion request to the following node according to the spiral order; for example: the orphan node 0-0-2 will contact nodes 0-1-0, 0-2-0, 1-0-0, 2-0-0 and so on until it gets a satisfied answer to the reinsertion request. After several failed attempts, the node will change its status to "Not Connected" and it will broadcast a "Disconnection" message to all its descendants. Upon receiving the "Disconnection" message, all the descendants change their status to "Not Connected". The disconnected nodes join the network separately, using an arbitrary delay in order to avoid over charging the server with "Get List of Active Nodes" requests.

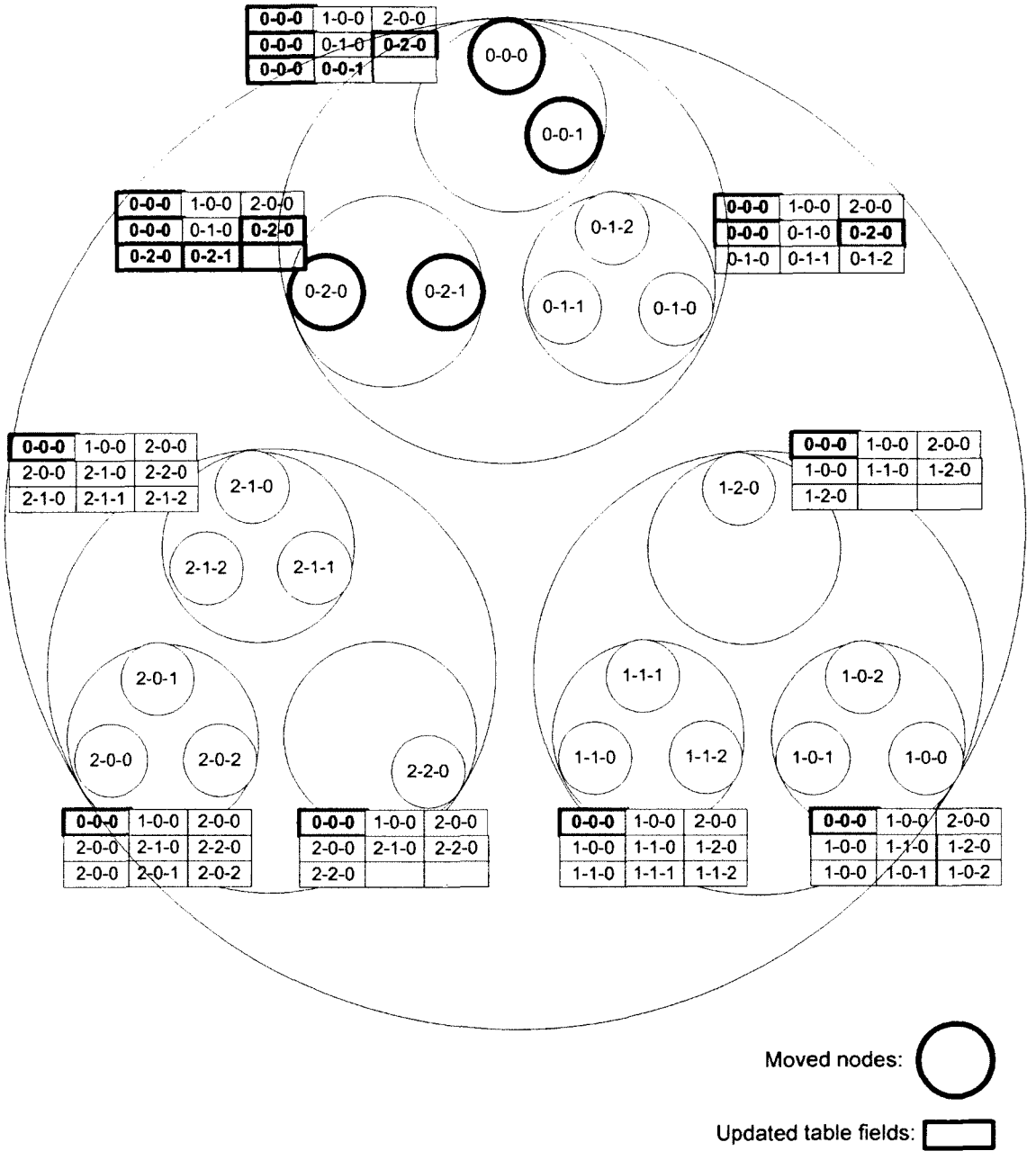

x-x-x 1-0-0 2-0-0
x-x-x 0-1-0 0-2-0
x-x-x 0-0-1 0-0-2
0-0-2
0-0-1
Reinsertion request
x-x-x 1-0-0 2-0-0
x-x-x 0-1-0 0-2-0
0-2-0 0-2-1
0-2-1
0-2-0
0-1-2
0-1-1
0-1-0
1-0-0 2-0-0
0-1-0 0-2-0
0-1-0 0-1-1 0-1-2
Reinsertion request
1-0-0 2-0-0
2-0-0 2-1-0 2-2-0
2-1-0 2-1-1 2-1-2
2-1-0
2-1-2
2-1-1
2-0-1
2-0-0
2-0-2
2-2-0
1-0-0 2-0-0
2-0-0 2-1-0 2-2-0
2-0-0 2-0-1 2-0-2
1-0-0 2-0-0
2-0-0 2-1-0 2-2-0
2-2-0
1-0-0 2-0-0
1-0-0 1-1-0 1-2-0
1-2-0
1-2-0
1-1-1
1-1-0
1-1-2
1-0-2
1-0-1
1-0-0
1-0-0 2-0-0
1-0-0 1-1-0 1-2-0
1-1-0 1-1-1 1-1-2
1-0-0 2-0-0
1-0-0 1-1-0 1-2-0
1-0-0 1-0-1 1-0-2
Updated table fields:

5- Insertion completion: Similarly to the Join request, Insertion request must be negotiated between all nodes of the insertion level. The node in Orphan state accepts only reinsertion requests and refuses Join Requests. Node 0-2-0 is accepted in level 0 and becomes node 0-0-0 (available place). It inherits part of the table of node 1-0-0 (1st line). Node 0-0-2 is accepted in level 1 and becomes 0-2-0 (available place). It inherits part of the table of node 0-1-0 (1st and 2nd lines). The new inserted nodes (0-0-0 and 0-2-0) multicast an update table messages to their descendants.



Updated table fields:

6- Procedure completed



ANNEXE C – ALGORITHMES

Localisation d'un nœud dans la table de voisinage à partir de son ID

//Node position discovery from ID

//Knowing its ID a node call this function to find its position in the whole community.
//once the position is known, the node can situate itself in the neighborhood table. This information can be used by the recovery distributed algorithm.

Find_Node_level(ID);

ID: | | | | | |

ID_0 ID_1 ID_2 ... ID_{p-1}

```
Node_level=p-1
While ID[Node_level]==0
        Node_level--;
Return Node_level;
```

//then the own_node_id can be located in the neighborhood table of size [p][n] as follows:
table[Node_level][ID[Node_level]]

Recherche de la position du nœud IDd ou de celle du nœud le plus proche si IDd n'existe pas dans la table des IDs appelé IDT

/**

/* Description : Cette fonction compare l'ID du nœud destinataire IDd avec les IDs de tous les

/* nœuds représentés dans la table des ID (appelé IDT). Le nœud le plus proche c'est le nœud

/* qui a la partie gauche de son ID la plus commune avec l'ID du nœud destinataire.

/* Cette fonction retourne la position (X, Y), dans la table des ID (IDT), du nœud destinataire

/* IDd. Si le nœud IDd n'est pas dans la table IDT, la fonction retourne la position du nœud le

/* plus proche à IDd.

/* Entrées : IDd, IDT, n, p

/* Sorties : position (X, Y) du nœud IDd ou du nœud le plus proche à IDd dans la table IDT

**/

Début

x=0, y=0, Rmax=0, i=0, j=0

```
tant que i<p-1
faire
        tant que j<n-1
        faire
                k=0, R=0
                tant que k<p et IDT(i,j)[k] == IDd[k]
                faire
                        R++
                fin faire
                si k==p
                        retourner (i, j, IDd) //on retourne la position du nœud IDd
                sinon
                        si R>Rmax
                                Rmax=R
                                x=i, y=j;
                        fin si
                fin si
                j++
        fin faire
        i++
fin faire
retourner (X, Y, proche) //on retourne la position du nœud le plus proche à IDd
```

ANNEXE D – HISTOGRAMME

En communiquant entre eux, les nœuds créent une table ou une structure de données contenant un histogramme qui représente la répartition de leurs températures. Si on normalise l'histogramme, en divisant les valeurs de chaque classe par le nombre total des nœuds, il peut donc être interpréter comme la densité de probabilité de la variable aléatoire température. C'est une variable aléatoire non stationnaire. Le schéma suivant montre la forme générale de cette table de répartition des températures ainsi que les plages de températures.

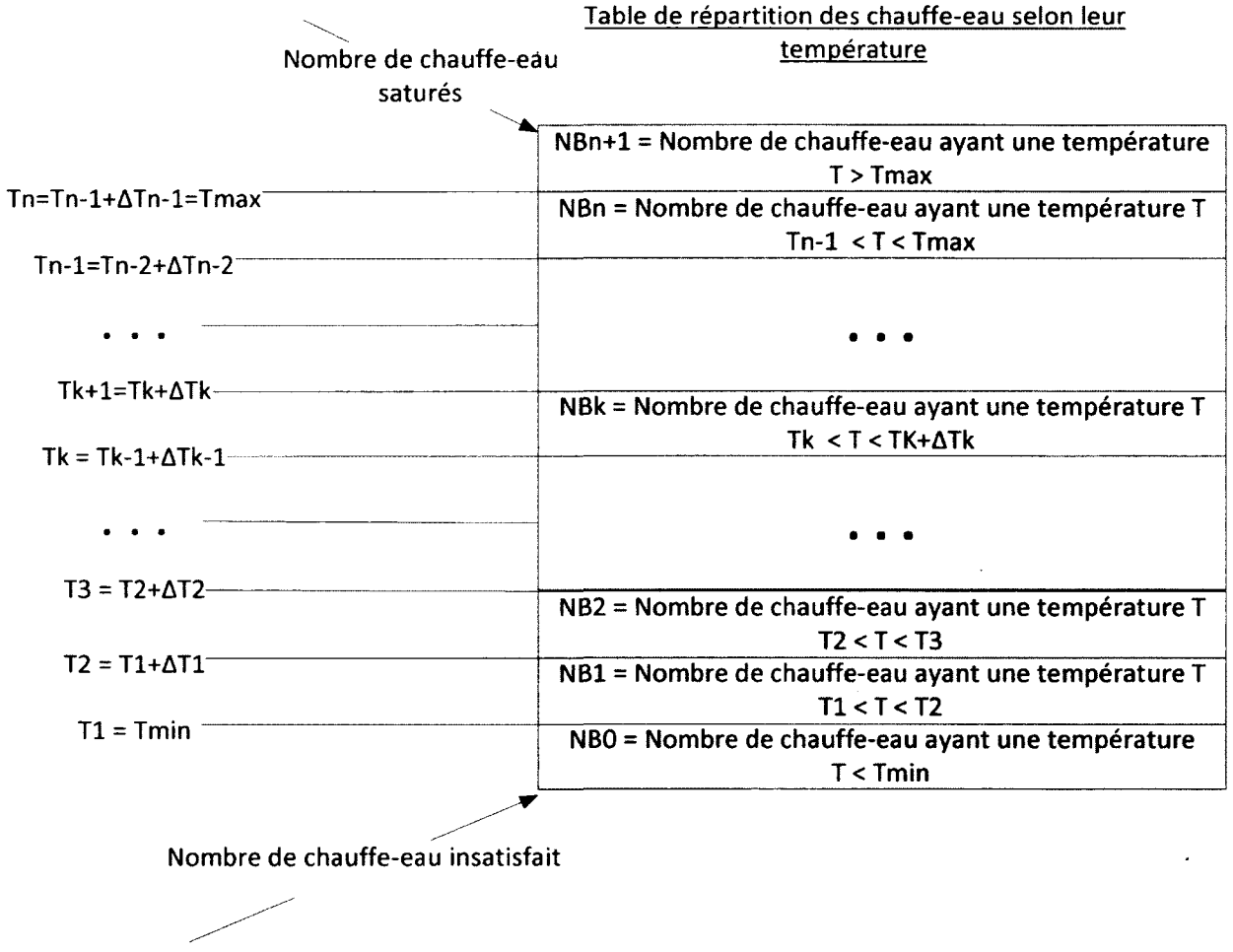


Figure 43: histogramme des températures

Cette table contient le nombre des nœuds saturés T>Tmax et ceux insatisfait T<Tmin. Elle peut donc servir pour déterminer la valeur d'un quota variable respectant les conditions un et deux tel que définit au CHAPITRE 2.

Pour déterminer si un chauffe-eau peut consommer ou pas, une valeur de température seuil est déterminée. Tous les nœuds, qui ont une température supérieure à cette valeur seuil arrêtent de consommer et tous ceux qui ont une température inférieure commencent à consommer.

En réalité cette température seuil est une plage de valeurs limitée par deux températures de référence : TOP et BOTTOM. L'algorithme suivant is_in_quota() sert à trouver la plage de valeur seuil et à déterminer les deux températures de références : TOP et BOTTOM. Tous les nœuds qui ont une température supérieure à la température de référence TOP n'ont pas le droit de consommer, tous ceux qui ont une température plus basse que la température BOTTOM ont le droit de consommer et le reste des nœuds décident d'une façon aléatoire s'ils ont le droit de consommer ou pas.

Algorithme is_in_quota()

Prérequis

La table de répartition des températures de tous les chauffe-eaux selon des plages de valeurs ainsi qu'une table de référence de température sont déjà créées.

Les deux conditions suivantes doivent être traitées avant ou au dans la fonction is_in_quota()

```
si (t > T_MAX)                  //température trop élevée
                retourner 0   //chauffe-eau arrête de consommer
fin si
si (t < T_MIN et type_quota est optionnel)  //température en dessous de celle du confort
                retourner 1   //chauffe-eau commence à consommer
fin si
```

Pseudo-code

```
/*****************************************************************************/
/*      Auteur: Simon Ayoub ing. jr
/*      Description: Algorithme de la fonction " is_in_quota(int t)", retourne 1 si la
température t est dans le quota, sinon elle retourne 0
/*      t: température du chauffe-eau en question
/*      hist[NB_T]: Histogramme de températures, chaque entrée contient le nombre de
chauffe-eaux appartenant à une plage de température
/*      NB_T: Nombre d'intervalles de température
/*      quota: nombre de chauffe-eaux (unité de puissance) maximal qui peut être utilisé en
même temps; il peut être variable avec le temps
/*      type_quota: optionnel ou obligatoire
/*      T_MIN: en dessous de cette température le confort du client peut être affecté
```

```
/*      T_MAX: en dessus de cette température, la température est trop élevée
/*      tempRef(i): Table de températures de référence représentant chaque plage de valeurs
/*************************************************************************
**/

Début
        somme = hist(0)        //la case (0) contient le nombre de chauffe-eaux qui ont T<Tmin
        i = 0
        tant que somme < quota
                i = i + 1
                somme = somme + hist(i)
        fin tant que

//si la température est en dessous de celle du confort (le quota est optionnel), on tire au hasard
        si (i = 0)
                si T<Tmin
                        pourcent = (somme - quota)/hist(i)
                        tirer au hasard un nombre "rand" entre 0 et 1
                        si rand < pourcent
                                retourner 0     //chauffe-eau OFF
                        sinon
                                retourner 1     //chauffe-eau ON
                        fin si
                sinon
                        retourner 0          //chauffe-eau OFF
                fin si
        fin si

        si t<tempRef(i-1) //t appartient à la plage hist(i - 1)
                retourner 1     //chauffe-eau ON
        sinon   si t >tempRef(i) //t appartient à la plage hist(i + 1)
                        retourner 0     //chauffe-eau OFF
                fin si
        sinon
//t appartient à la plage hist(i) il faut tirer un nombre au hasard pour décider si on chauffe ou
pas
                pourcent = (somme - quota)/hist(i)
                tirer au hasard un nombre "rand" entre 0 et 1
                si rand < pourcent
                        retourner 0     //chauffe-eau OFF
                sinon
                        retourner 1     //chauffe-eau ON
                fin si
        fin si
fin
```

ANNEXE E – SUITE DU DÉVELOPPEMENT

Présentement, le développement des concepts proposés lors de cette thèse se poursuit en collaboration avec le centre de recherche CanmetÉNERGIE situé à Varennes, Québec. L'objectif du projet est de gérer la consommation d'un million de chauffe-eaux simulés et d'un ou plusieurs nœuds réels incluant un chauffe-eau expérimental (Figure 44), à l'aide de l'algorithme distribué de gestion de la charge présenté au CHAPITRE 2 et la technologie Ring-Tree présenté au CHAPITRE 3.

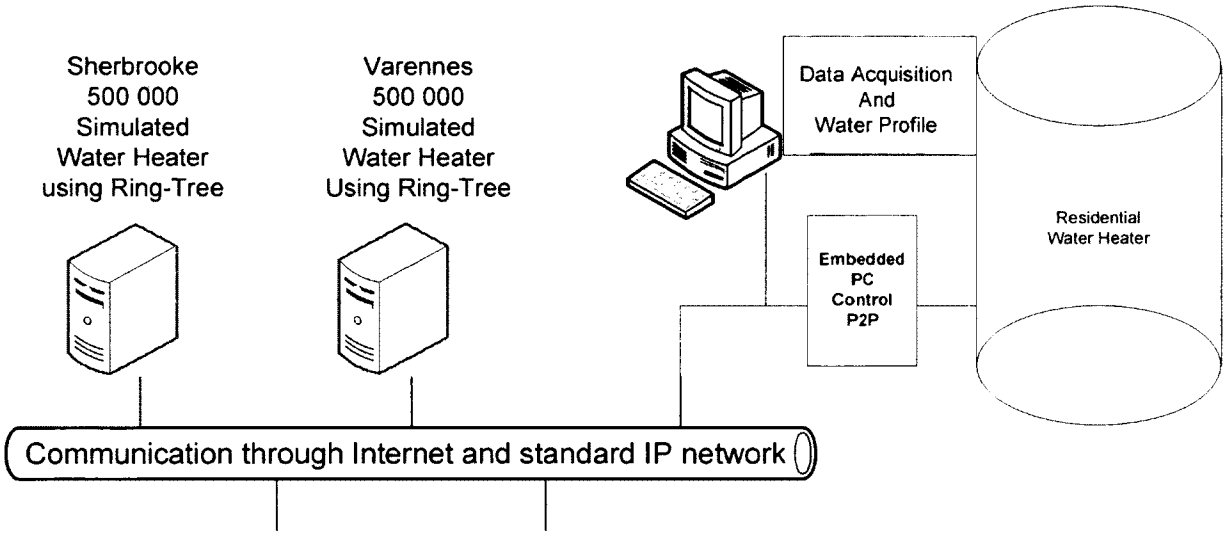


Figure 44: vue globale de la démonstration de simulation de million de nœuds

Un simulateur de chauffe-eaux permettant de simuler un grand nombre de nœuds programmé en langage C++ a été crée. L'algorithme de gestion de la charge, utilisant différents types de quota (obligatoire, optionnel, constant et variable), a été intégré au simulateur. Une communication selon le réseau Ring-Tree permettant la création d'un histogramme de température a été réalisée. Le mécanisme basé sur la valeur seuil de température permettant aux chauffe-eaux de décider, individuellement, s'ils appartiennent au quota a été intégré. Un chauffe-eau expérimental (Figure 45) incluant un système de tirage d'eau chaude, un système de refroidissement et un système d'acquisition de données et de contrôle (SCADA, TS-7500) ont été montés. Un outil de visualisation (LabView) pour aider à la validation et à l'étude de performance des différentes parties du système a été développé.

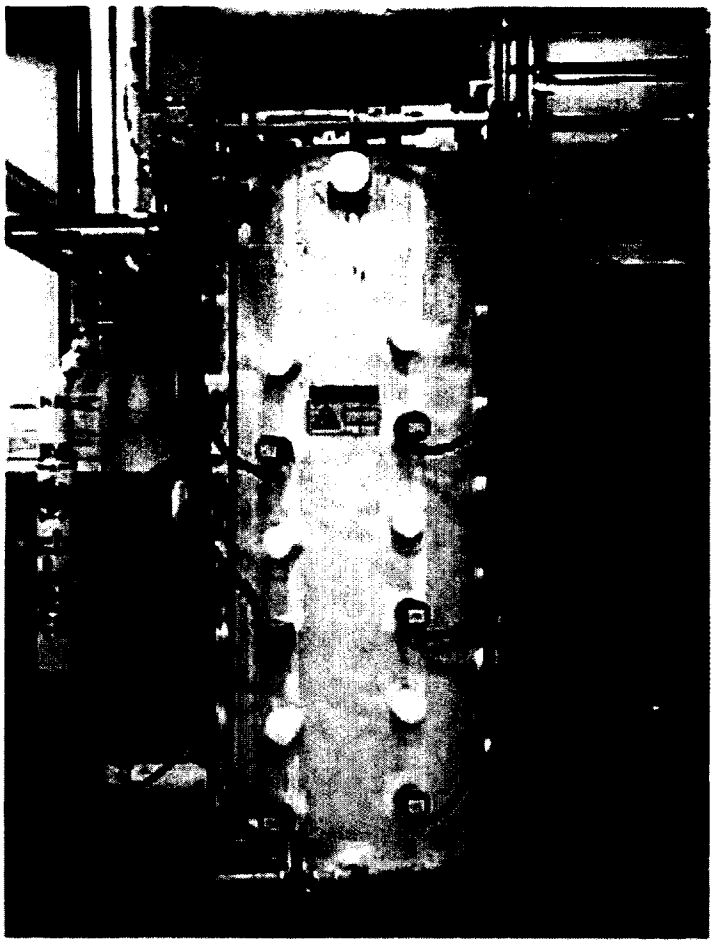


Figure 45: chauffe-eau expérimental[4]

Deux types de nœuds ont été conçus : un pour contrôler un chauffe-eau simulé et un pour contrôler un chauffe-eau réel. La Figure 46 présente une vue globale des différentes composantes logicielles de ces deux types de nœuds.

- « Water-Heater Model » représente un chauffe-eau simulé
- « Load Management Algorithm » représente l'algorithme de gestion de la charge
- « Ring-Tree Layer » représente la couche de communication Ring-Tree

Plusieurs objectifs et projets à venir : l'utilisation d'un quota variable pour équilibrer la consommation et la production d'énergies fluctuantes ou pour participer à une activité de type Demand-Response (DR); l'utilisation de l'algorithme distribué de gestion de la consommation pour gérer d'autres types de charges comme les bornes de recharges des

[4] Source: CanmetÉNERGIE

voitures électriques; l'étude des différentes possibilités de contrôle de charge avec une communauté P2P (Figure 47).

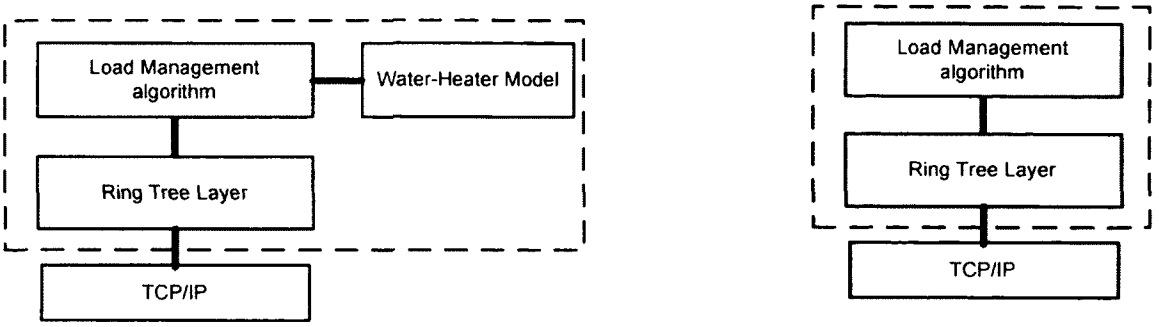


Figure 46: composantes logicielles a) chauffe-eau simulé b) chauffe-eau expérimental

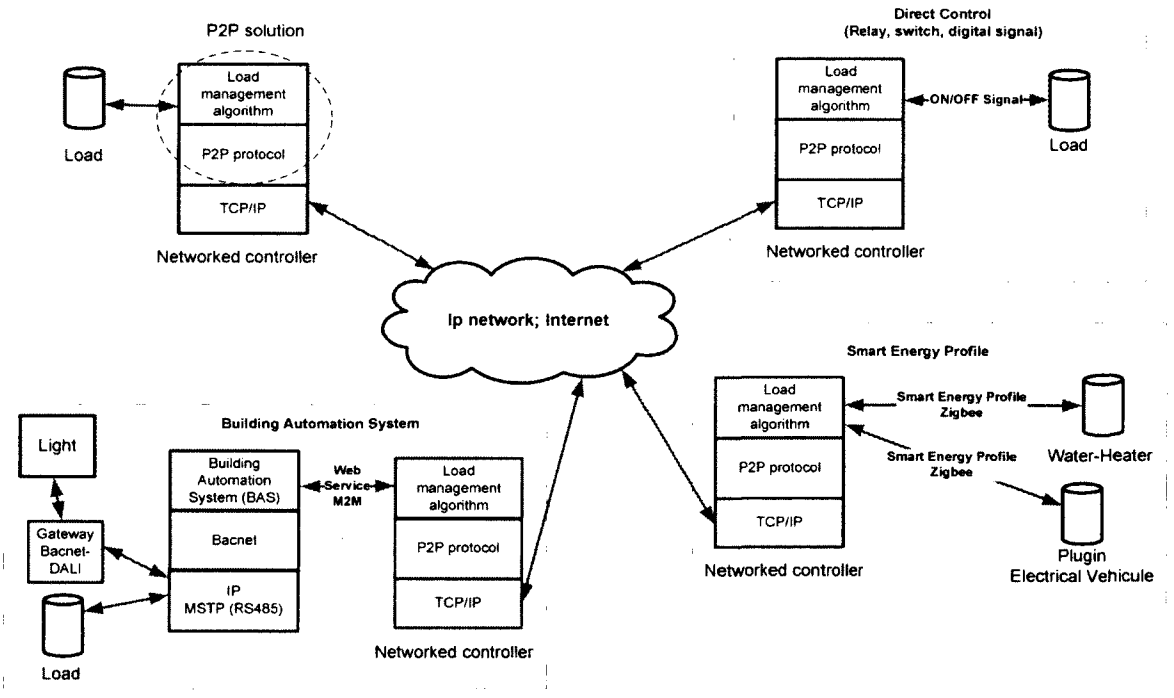


Figure 47: différentes possibilités de contrôle de charge avec une communauté P2P5

[5] Source: CanmetÉNERGIE

LISTE DES RÉFÉRENCES

[1] Ressources naturelles Canada, U.S. Department of Energy, « Rapport final sur la mise en œuvre des recommandations du Groupe de travail », Septembre 2006, [En ligne] : http://www.rncan.gc.ca/sites/www.nrcan.gc.ca.energy/files/pdf/eneene/pdf/outpan-fra.pdf

[2] Rahul Walawalkar, Stephen Fernands, Netra Thakur and Konda Reddy Chevva. Evolution and current status of demand response (DR) in electricity markets: insights from PJM and NYISO. Energy, pages 1553 - 1560, 2010.

[3] David Beauvais, Alexandre Prieur et François Bouffard "Smart Grid : la contribution des ressources énergétiques distribuées pour équilibrer les énergies renouvelables" 2012-11-19, [en ligne] http://canmetenergie.rncan.gc.ca/energies-renouvelables/smart-grid/publications/3170.

[4] Federal Energy Regulatory Commission, « Assessment of Demand Response & Advanced Metering», staff report, Dec. 2012. [En ligne] : http://www.ferc.gov/legal/staff-reports/12-20-12-demand-response.pdf

[5] Michel Bernier, "Design et efficacité énergétique en mécanique du bâtiment", *École Polytechnique de Montréal*, Janvier 2001.

[6] R. L. Earle, "Demand elasticity in the California power exchange day ahead market," *Elect. J.*, vol. 13, no. 8, pp. 59–65, Oct. 2000.

[7] R. H. Patrick and F. A. Wolak, "Real-time pricing and demand side participation in restructured electricity markets," in *Proc. Conf. Retail Participation in Competitive Power Markets*, Stanford Univ., 2001.

[8] A. C. Tellidou and A. G. Bakirtzis, "Demand response in electricity markets," in *Proc. 15th Int. Conf. Intell. Syst. Appl. Power Syst.*, 2009, pp. 1–6.

[9] S. Rassenti, V. Smith, and B. Wilson, "Controlling market power and price spikes in electricity networks: Demand-side bidding," *Proc. Natl. Acad. Sci.*, vol. 100, no. 5, pp. 2998–3003, May 2003.

[10] ISO/RTO Council, "North American Wholesale Electricity Demand Response Program Comparison", 2011 Edition, [En ligne] : http://www.isorto.org/atf/cf/%7B5B4E85C6-7EAC-40A0-8DC3-

003829518EBD%7D/IRC%20DR%20M&V%20Standards%20Implementation%20Comparison%20(2012-01-20).xls

[11] Khodaei, A.; Shahidehpour, M.; Bahramirad, S.; , "SCUC With Hourly Demand Response Considering Intertemporal Load Characteristics," *IEEE Transactions on Smart Grid*, vol.2, no.3, pp.564-571, Sept. 2011

[12] Kladnik, B., Artac, G. and Gubina, A. (2012), "An assessment of the effects of demand response in electricity markets." *Euro. Trans. Electr. Power.*, 2012

[13] A. K. David and Y. C. Lee, "Dynamic tariffs: Theory of utility-consumer interaction," *IEEE Trans. Power Syst.*, vol. 4, no. 3, pp. 904–911, Aug. 1989

[14] L. Goel, Q.Wu, and P.Wang, "Reliability enhancement and nodal price volatility reduction of restructured power systems with stochastic demand side load shift," in *Proc. IEEE Power Eng. Soc. Gen. Meet.*, 2007, pp. 1–8.

[15] R. Rajaraman, J. V. Sarlashkar, and F. L. Alvarado, "The effect of demand elasticity on security prices for the poolco and multi-lateral contract models," *IEEE Trans. Power Syst.*, vol. 12, no. 3, pp. 1177–1184, Aug. 1997.

[16] L. Goel, Q. Wu, and P. Wang, "Nodal price volatility reduction and reliability enhancement of restructured power systems considering demand-price elasticity," *Elect. Power. Syst. Res.*, vol. 78, pp. 1655–1663, 2008.

[17] T. Joseph Lui, Warwick Stirling,and Henry O. Marcy, "Get Smart", *IEEE power & energy magazine* 1540-7977, May-June 2010.

[18] Bucher, M.; Koch, S.; Andersson, G.; , "A dynamic household appliance stock model for load management introduction strategies," *Energy Market (EEM), 2011 8th International Conference on the European*, vol., no., pp.717-722, 25-27 May 2011

[19] S. Koch, M. Zima, and G. Andersson, "Potentials and applications of coordinated groups of thermal household appliances for power system control purposes," presented at the *IEEE-PES/IAS Conference on Sustainable Alternative Energy*, 2009.

[20] D. Infield, J. Short, C. Home, and L. Freris, "Potential for domestic dynamic Demand-Side Management in the UK," presented at the *IEEE Power Engineering Society General Meeting*, 2007.

[21] E. Peeters, D. Six, M. Hommelberg, R. Belhomme, and F. Bouffard, "The ADDRESS project: An architecture and markets to enable active demand," presented at the *6th International Conference on the European Energy Market 2009*, May 2009.

[22] Huang KY, Huang YC. "Integrating direct load control with interruptible load management to provide instantaneous reserves for ancillary services." *IEEE Trans Power Syst* 2004;19:1626–34.

[23] Paper, P. G.-W. (n.d.). PMA ONLINE MAGAZINE. [En ligne] : http://www.retailenergy.com/articles/loadagg.htm

[24] S. H. Lee and C. L. Wilkins, "A Practical Approach to Appliance Load Control Analysis: A Water Heater Study," *IEEE Trans. Power Apparatus and Systems*, vol. 102, No.4, pp. 1007-1013, 1983.

[25] R. F. Bischke, "Design and Controlled Use of Water Load Management," *IEEE Trans. Power Apparatus and Systems*, vol. 104, No.6, pp. 1290-1293, 1985.

[26] M. H. Nehrir and B. J. LaMeres, "A multiple-block fuzzy logic-based electric water heater demand-side management strategy for leveling distribution feeder demand profile," *Electric Power Systems Research* 56, pp. 225-230, 2000.

[27] J.-C. Laurent, G. Desaulniers, R. P. Malhame and F. Soumis, "A column generation method for optimal load management via control of electric water heaters," *IEEE Trans. Power Systems*, vol. 10, No.3, pp. 1389-1400, August 1995.

[28] J.-C. Laurent, and R. P. Malhame, "A Physically-Based Computer Model Aggregate Electric Water Heating Loads," *IEEE Summer Power Meeting, Vancouver B.C.*, paper 93 SM496-0 PWRS, 1993.

[29] D.S. Callaway, "Tapping the energy storage potential in electric loads to deliver load following and regulation, with application to wind energy," *Energy Conversion and Management*, vol. 50, no. 5, pp. 1389 – 1400, 200

[30] Larry Hughes, "Meeting residential space heating demand with wind-generated electricity", *Renewable Energy* 35, 1765–1772, 2010.

[31] Pengwei Du; Ning Lu; , "Appliance Commitment for Household Load Scheduling," *IEEE Transactions on Smart Grid*, vol.2, no.2, pp.411-419, June 2011

[32] Karl Aberer, Luc Onana Alima, Ali Ghodsi, Sarunas Girdzijauskas, Seif Haridi, Manfred Hauswirth, "The essence of P2P: A reference architecture for overlay

networks", *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, 2005

[33] Joaquin KELLER, Gwendal SIMON, "SOLIPSIS : A Massively Multi-Participant Virtual World", Proc. *Int. Conf. Parallel Distrib. Proces. Tech. Applic.*, pp. 262-268, 2003

[34] Birke, R.; Leonardi, E.; Mellia, M.; Bakay, A.; Szemethy, T.; Kiraly, C.; Cigno, R.L.; Mathieu, F.; Muscariello, L.; Niccolini, S.; Seedorf, J.; Tropea, G.; , "Architecture of a network-aware P2P-TV application: the NAPA-WINE approach," *IEEE Communications Magazine*, vol.49, no.6, pp.154-163, June 2011

[35] Andres Carvallo, John Cooper, "The Advanced Smart Grid – Edge power Driving Sustainability", *Artech House*, p.8, 2011.

[36] Nourai, A.; Sastry, R.; Walker, T.; , "A vision & strategy for deployment of energy storage in electric utilities," *IEEE Power and Energy Society General Meeting, 2010* , vol., no., pp.1-4, 25-29 July 2010

[37] R. P. Malhame and C.Y. Chong, "Stochastic Hybrid-State Systems for Electric Load Modeling," in Proc. Of the 22nd *IEEE Conference on Decision and Control*, p 1143-9 vol.3, 1983.

[38] Alain Poulin, André Laperrière, Dominic Lallier-Daniels, "Residential Hot Water Consumption Analysis", *Hydro-Québec Laboratoire des technologies de l'énergie (LTE)*, 2011

[39] Marcel Lacroix, Nicolas Galanis, Alain Moreau, "Electric Water Heater Design For Load Shifting And Customer Retention", *Hydro-Quebec (LTÉE)*, December 1996

[40] Xiang, H. W. Jing, H. Yahong. "ABC:A Cluster-based Protocol for Resource Location in Peer-to-Peer Systems", *18th International Parallel and Distributed Processing Symposium*, 2004. Issue, 26-30 April2004 Page(s): 84a-

[41] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for Internet applications". In *Proceedings of SIGCOMM 2001*, August 2001.

[42] J Liang, R Kumar, K Ross, "The KaZaA Overlay: A Measurement Study", *Proceedings of the 19th IEEE Annual Computer Communications Workshop* (2004)

[43] J. Keller, G. Simon, "SOLIPSIS: A Massively Multi-Participant Virtual World", *Proc. Int. Conf. Parallel & Distributed Techniques & Applications (PDPTA 2003), CSREA Press, 2003*, vol. 1, pp. 262-268

[44] ChaeY. Lee, Ho Dong Kim, "Reliable overlay multicast trees for private Internet broadcasting with multiple sessions", *Computers & Operations Research 34 (2007)* 2849 – 2864

[45] Kamal Jain, Laszlo Lovasz, Philip A. Chou. "Building scalable and robust peer-to-peer overlay networks for broadcasting using network coding", *Distributed Computing March 2007*, volume 19, Number 4, pp. 301-311

[46] Karl Aberer, Luc Onana Alima, Ali Ghodsi, Sarunas Girdzijauskas, Seif Haridi, Manfred Hauswirth, "The essence of P2P: A reference architecture for overlay networks", *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing (2005)*

[47] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. « Resilient Overlay Networks. » *SOSP* '01

[48] Z. Li, P. Mohapatra, "On investigating overlay service topologies", *Computer Networks 51 (2007)* 54-68

[49] K.C. Almeroth, "The evolution of multicast: from the MBone to interdomain multicastto Internet2 deployment", *IEEE Network*, Volume 14, Issue 1, Jan/Feb 2000 Page(s):10 – 20

[50] L. Garcès-Erice, E.W. Biersack, P.A. Felber, K.W. Ross, and G. Urvoy-Keller, "Hierarchical Peer-to-peer Systems", *Institut EURECOM, 06904 Sophia Antipolis, France*

[51] T. Wolf, S. Y. Choi, "Aggregated Hierarchical Multicast – A Many-to-Many Communication Paradigm Using Programmable Networks", *IEEE Transactions on Systems, Man, And Cybernetics-part C: Applications and reviews*, Vol. 33 no.3, August 2003

[52] C. Shields, J.J. Garcia-Luna-Aceves, "HIP: a protocol for hierarchical multicast routing", *Elsevier science,* 2000

[53] Wongyong Yoon, Dongman Lee, Hee Yong Youn, Seungik Lee, Seok Joo Koh, "A combined group/tree approach for scalable many-to-many reliable multicast", *INFOCOM 2002*

[54] Márk Jelasity, Alberto Montresor, Ozalp Babaoglu. « T-MAN: Gossip-based fast overlay topology construction », *Computer Networks 53 (2009)* 2321–2339

[55] Baset, S. A., Schulzrinne, H. G., « An Analysis of the Skype Peer-to-Peer internet Telephony Protocol », *Proceedings of the 25th IEEE International Conference on Computer Communications, INFOCOM (2006)*

[56] T. Ballardie, P. Francis, J. Crowcroft, "Core Based Trees (CBT)", *ACM SIGCOMM Computer Communication Review*, Volume 23, Issue 4 (Octobre 1993) Pages: 85-95

[57] C. Shields, J.J. Garcia-Luna-Aceves, "The Ordered Core Based Tree Protocol", *INFOCOM, Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, Volume 2, Issue , 7-12 Apr 1997 Page(s):884 - 891 vol.2

[58] Ayman Shaker and Douglas S. Reeves, « Self-stabilizing Structured Ring Topology P2P Systems », *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing (P2P'05)*, 2005

[59] Yang-hua Chu; Rao, S.G.; Seshan, S.; Hui Zhang, "A case for end system multicast", *IEEE Journal on Selected Areas in Communications*, Volume 20, Issue 8, Oct 2002 Page(s): 1456 – 1471

[60] Hans Eriksson, "Mbone: The Multicast Backbone", *Communications of the ACM*, Vol. 37, No.8, August 1994

[61] Akamai Technologies, Inc. [En ligne] : http://www.akamai.com

[62] iBeam Broadcasting Corp. [En ligne] : http://www.ibeam.com

[63] Dennis M. Moen, "Overview of Overlay Multicast Protocols", *George Mason University*,[En ligne] : http://netlab.gmu.edu/XOM/pdfs/Multicast%20Overview.pdf

[64] Rusitschka, S.; Gerdes, C.; Eger, K.; , "A low-cost alternative to smart metering infrastructure based on peer-to-peer technologies," Energy Market, 2009. EEM 2009. 6th International Conference on the European , vol., no., pp.1-6, 27-29 May 2009

[65] Galli, S.; Scaglione, A.; Zhifang Wang; , "For the Grid and Through the Grid: The Role of Power Line Communications in the Smart Grid," Proceedings of the IEEE , vol.99, no.6, pp.998-1027, June 2011