Notes de cours (ENS Lyon, M1)

Chapitre 3 : Réseaux sans fil

Table des matières

3 Réseaux sans fil

Les réseaux de téléphonie mobile que nous utilisons aujourd'hui souffrent souvent de lacunes :
- débit
- consommation des ressources
- flexibilité
- sécurité

Nouvelles architectures de réseaux sans fil.

Plan de la section :
- présentation des différentes générations de réseaux sans fil
- notion de routage dans les réseaux sans fil dynamiques, différents protocoles et algorithmes

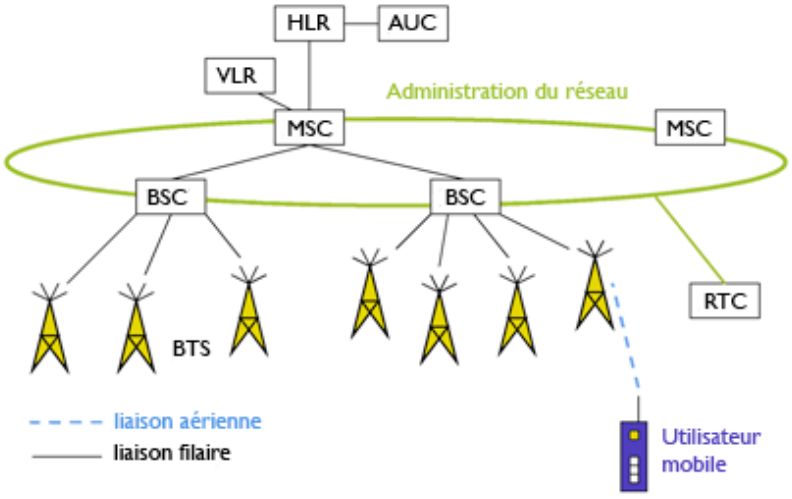
3.1 Différentes générations

3.1.1 Réseaux sans fil à stations de base

Les réseaux utilisés couramment de nos jours sont des réseaux sans fil à stations de base : GSM, GPRS, UMTS, Wi-Fi.

Caractéristiques :
- coût de déploiement important
- nécessité d'une infrastructure fixe, interconnectée de manière filaire.
- coût des licences, notamment pour l'UMTS

Téléphone sans fil : infrastructure hiérarchique



- Station mobile : MS, habituellement un téléphone mobile

- Partie radio du réseau : Stations de base, BTS (Base Tranceiver Station) : antennes chargées de communiquer avec les MS. La zone de communication couverte par une station de base est appelée cellule (*téléphone cellulaire*). Stations de contrôle BSC (Base Station Controller) : chargées de gérer un ensemble de stations de base.
- Partie routage : chargée d'acheminer la communication entre deux utilisateurs du réseau ou vers l'extérieur (comme pour appeler d'un portable vers un téléphone filaire fixe).
  - Centres de commutation de service mobile MSC (Mobile Switching Centers) reliés entre eux, effectuant le routage des communications
  - A chaque MSC est associé un enregistreur de localisation des visiteurs VLR (Visitor Location Register), qui gère les informations sur les abonnés se trouvant dans la zone gérée par le MSC (informations de localisation précises sur les utilisateurs mobiles).
  - Une base de données centrale (éventuellement dupliquée) HLR (Home Location Register) gère la liste des abonnés de l'opérateur du réseau (dont une information de localisation comme la zone de localisation).
  - Centre d'authentification AuC, base de données chargée d'assurer l'authentification des utilisateurs.
- Partie Opération et maintenance, chargée de mettre en place et de veiller au bon fonctionnement des différents éléments du réseau (des BSC à l'AuC).

Passerelles pour se connecter à d'autres réseaux extérieurs, notamment vers le téléphone fixe.

Coût de déploiement élevé, et nécessité de faire évoluer la configuration du réseau pour offrir la meilleure qualité de service possible.

- placement des stations de base
- taille des cellules
- fréquences attribuées à chaque cellule
- regroupement des stations de base affectées aux contrôleurs
- regroupement des contrôleurs affectés aux MSC
- ...

Paramètres qui influent grandement sur la qualité de la connexion. Une cellule trop grande risque d'être surchargée et de devoir refuser des connexions. Un changement de cellule par le téléphone mobile lors d'un déplacement est favorisé si les cellules appartiennent au même contrôleur...

Recherche de la configuration optimale : problème très complexe. Utilisation des fichiers de "log" des différents équipements du réseau, et application d'algorithmes génétiques.

En dehors de ces problèmes de coût, problèmes de sécurité qui obligent à mettre en place divers pare-feu, mécanismes d'authentification et autres, ce qui réduit encore les performances, déjà faibles en terme de débit.

On cherche à répondre aux quatre grands défis :

- haut débit
- faible consommation

– flexibilité, accessibilité, auto-configuration
– sécurité

Nouvelles architectures de réseaux proposées, que l'on détaille maintenant.

3.1.2 Réseaux radio maillés

But : supprimer les connexions filaires.

Dans ce type de réseaux, les stations de base utilisent également une liaison radio pour communiquer entre elles. Stations de base = relais radio pour les communications des téléphones mobiles. Mobile : rattaché à la station de base la plus proche, avec laquelle il communique exclusivement. Station de base : organise les communications "au mieux" pour éviter les pertes de paquets et pour optimiser l'utilisation de la bande passante.

Stations de base : suivant leur puissance et la distance entre elles, possibilité de communiquer → graphe des stations de base. Deux mobiles peuvent communiquer si les stations de base auxquelles ils sont rattachés peuvent communiquer : route dans le graphe. Idéalement, graphe connexe : il existe toujours une route reliant deux stations quelconques du réseau. Utilisation d'algorithmes de routage classiques dans les réseaux.

Il suffit que l'une des stations de base soit reliée à un réseau filaire pour permettre à l'ensemble des noeuds du réseau l'accès à ce réseau : communication avec un autre réseau, accès Internet...

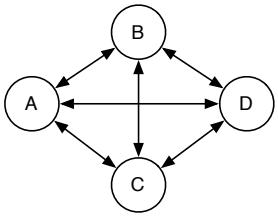
Variante des réseaux maillés : les réseaux satellitaires. Satellites = stations de base, se relayant entre elles les paquets des mobiles.

Une unique station terrestre suffit à assurer l'interconnexion entre les téléphones satellitaires et le réseau terrestre. Contrairement aux réseaux maillés classiques, ce n'est pas toujours le même satellite qui communique avec cette station.

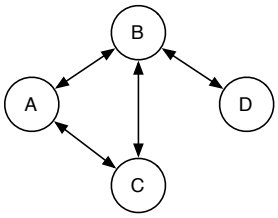
3.1.3 Réseaux P2P

Inspirés des technologies de P2P, on cherche ici à supprimer également les stations de base. Si plusieurs mobiles sont à portée de communication, ils peuvent échanger directement des informations sans l'aide d'une station de base.

Mode ad-hoc simple saut : on ne communique que avec ceux directement à portée. Si le graphe de connexion est complet, tous peuvent communiquer avec tous. C'est par exemple le cas dans un espace confiné, comme une salle de réunion (Fig. (a)). Par contre, deux noeuds non reliés ne peuvent pas communiquer, comme A et D dans la Fig. (b).



(a) Graphe complet
Mode simple saut



(b) Graphe non complet

Un réseau mobile ad-hoc est constitué de stations munies d'une interface de communication radio. Un noeud peut communiquer directement avec ses voisins (les noeuds à portée de communication de sa propre interface), et il fait office de routeur pour les autres mobiles du réseau. Dans l'exemple de la Fig. (b), le noeud B relaye les messages du mobile D vers le mobile A et vice versa. Réseau **MANET** : Mobile Ad hoc NETwork. Très utile lorsqu'aucune connexion filaire est disponible, et pour le déploiement rapide d'un réseau. Les noeuds communiquent en acheminant le message par routage "multi-saut". On détaillera les algorithmes de routage par la suite.

Problème de consommation :
- plus de messages émis en mode ad hoc : transmission de mes propres messages + des messages routés : plus gourmand en énergie ?
- portées de communication largement réduites en ad hoc, les distances peuvent être divisées par 10 par rapport à un mode basé sur une infrastructure.
- consommation énergétique : varie au moins proportionnellement au carré de la distance de communication. Ainsi, transmettre 10 fois plus de message 10 fois moins loin permet de ne consommer qu'un dixième de l'énergie nécessaire au mode station de base.
- diminution des puissances d'émission : limite les risques de collisions entre les communications, et donc les retransmissions dues à la perte de paquets, ce qui permet de faire encore des économies d'énergie.

Configurer les portées de communication (la puissance d'émission) : problème important et complexe :
- portée suffisante pour que le réseau soit connexe
- portée pas trop grande pour limiter les consommations d'énergie et les risques de collision

Densité du réseau : nombre de mobiles par zone de communication. Dépend de la portée de communication. Pour avoir 90% de chances que le réseau soit connexe, il faut atteindre une densité de 15. Cela correspond à avoir un mobile tous les $22m^2$ pour une portée de communication de dix mètres.

3.1.4 Réseaux hybrides combinant ces différentes technologies

On peut imaginer une classe de réseaux hybrides mélangeant ces différents types de réseau : utilisation d'un mode ad hoc si un mobile n'est pas à portée de communication d'une borne, ... Arriver à trouver le juste mélange entre réseaux à stations de base centralisés, réseaux maillés, réseaux P2P.

3.2 Modélisation des réseaux ad hoc

3.2.1 Modélisation du réseau

Réseau radio : les connexions se font au gré de la transmission radio. Un signal est reçu s'il est K fois plus fort que le bruit ($K = 10$ typiquement).

Atténuation des ondes radio avec la distance : $1/r^\alpha$, où r est la distance et α dépend de l'environnement ($\alpha = 2$ dans l'air, mais peut augmenter avec les obstacles).

Ainsi, si P_e est la puissance d'émission du signal, et B le bruit ambiant constant, un émetteur radio sera reçu jusqu'à une distance

$$R = \left(\frac{P_e}{KB}\right)^{1/\alpha}$$

.

Espace homogène : α est constant, et l'ensemble des positions d'où l'on peut recevoir le signal constitue un disque centré sur l'émetteur. Cet ensemble est plus complexe lorsqu'il y a des obstacles (murs, bureaux, ...). Modèle classique = le plus simple = graphe de disques unitaires (*disk unit graph*) : deux noeuds d'un réseau ad-hoc peuvent communiquer s'ils sont à distance inférieure à R. Les distances sont normalisées pour que $R = 2$, un disque unitaire est centré sur chaque noeud, et deux noeuds peuvent communiquer si leurs disques s'intersectent.

Raffinement du modèle :

- Graphe de boules unitaires : noeuds dans l'espace plutôt que dans le plan.
- Puissances d'émission variables : disques ou boules de rayon variable.

Modèle valide dans un espace homogène si un seul noeud émet à la fois. Autre problème des réseaux ad-hoc : les différentes transmissions interfèrent les unes avec les autres, et le "bruit" provenant de plusieurs transmissions s'additionnent. Ainsi, un fort trafic le long d'une route peut briser des connexions qui seraient possibles autrement (en augmentant considérablement le bruit ambiant).

Liens unidirectionnels

Possibilité qu'un noeud soit entendu d'un autre mais pas l'inverse : lien unidirectionnel. Cela arrive notamment lorsque les puissances d'émission sont différentes suivant les émetteurs.

Liens à bannir : problème des acquittements que l'on ne peut pas envoyer (et l'acquittement est nécessaire pour s'assurer que le paquet est bien passé). D'autre part, s'il existe un lien de A vers B, pour pouvoir utiliser ce lien, A doit

savoir qu'il existe. Pour cela, B doit avoir dit d'une façon ou d'une autre à A qu'il le reçoit. Coût de la communication de B vers A : peut être important et prohibitif si l'on veut faire parvenir des acquittements.

On définit donc le **voisinage** d'un noeud comme l'ensemble des noeuds de sa zone de couverture (noeuds à qui il peut envoyer des messages) dont il peut recevoir les messages.

Deux modes d'émission dans un réseau ad-hoc :

- *broadcast* : envoi des paquets de contrôle à tous les voisins, pour une diffusion efficace de l'information.
- *unicast* : on spécifie le noeud suivant sur la route d'un paquet de données, et on demande un acquittement, pour éviter de perdre les paquets de données. (à l'inverse, la perte de paquets de contrôle ne compromet pas en général le fonctionnement du réseau).

Autres subtilités du modèle : lorsqu'un noeud possède plusieurs interfaces radio. Les protocoles deviennent tout de suite beaucoup plus complexes, car chaque interface possède sa propre zone de couverture et ses interfaces voisines.

Nécessité de savoir router dans ce modèle qui combine toutes les difficultés d'un réseau radio :

- La portée diminue avec le débit. Le relayage radio est donc indispensable pour éviter les réseaux filaires
- Dynamicité du réseau radio même si les émetteurs sont fixes : les connexions peuvent être perturbées par la pluie, le passage d'un véhicule, le déplacement des meubles...

3.2.2 Le problème du routage en ad-hoc

Graphe d'interconnexion : très grande dynamicité, et les mécanismes de routage classiques ne peuvent pas être utilisés.

Deux grandes familles de protocoles de routages pour les réseaux ad-hoc.

- Routage réactif, à la demande, ne génère les routes que lorsqu'elles sont demandées par les mobiles. Exemple : AODV (Ad-hoc On-demand Distance Vector routing) qui utilise un mécanisme de diffusion (broadcast) dans le réseau pour découvrir les routes valides.
- Routage proactif, par table de routage, qui maintient continuellement des tables de routage à jour. Exemple : OLSR (Optimized Link State Routing), qui utilise un mécanisme permettant de désigner un sous-ensemble de son voisinage responsable de la dissémination des informations de contrôle de topologie dans le réseau à moindre coût.

Approches hybrides que nous ne détaillerons pas ici.

Plusieurs types de routage sont à étudier dans les réseaux ad-hoc. On a vu qu'il y a deux types possibles d'émission des messages, en mode unicast ou broadcast. On retrouve les mêmes concepts dans les algorithmes de routage :

- Routage d'un noeud vers un autre, *unicast*, pour lequel il faut trouver une route jusqu'à un noeud précisément identifié. Il s'agit du cas le plus classique du routage, comme nous l'avons étudié pour les réseaux filaires et les réseaux P2P.

- Naturellement, dans les réseaux sans fil, la diffusion se fait à tous les voisins. On s'intéresse donc particulièrement dans ce type de réseaux aux algorithmes de routage d'un noeud vers un ensemble de noeud. Il s'agit du routage multipoint *multicast*.
- Cas particulier du routage multipoint : le routage d'un noeud vers tous les autres, appelé aussi diffusion ou *broadcast*. Les techniques utilisées sont généralement différentes des techniques de multicast.

Diffusion : brique de base des protocoles de routage, souvent nécessaire pour diffuser les informations nécessaires à la constitution des tables de routage.

Techniques de routage unicast et multicast : utilisation de tables de routage pour construire des arbres de diffusion restreints.

On va étudier principalement la diffusion unicast avec AODV (routage réactif). Pour le protocole de routage proactif OLSR, on s'intéressera tout d'abord à un algorithme de diffusion optimisé, puis on verra des algorithmes unicast et multicast.

3.3 Le routage réactif avec AODV

Les protocoles de routage réactif créent et maintiennent les routes selon les besoins. Lorsqu'une route est requise, lancement d'une procédure de découverte globale de routes, dans le but d'obtenir une information spécifiée et inconnue au préalable.

Aucun échange de paquets de contrôle pour construire des tables de routage, technique d'inondation et donc consommation d'une grande quantité de ressources pour découvrir une simple route entre 2 points du réseau.

Réseau dense : protocole très coûteux, mais plus avantageux dans les réseaux fluides pour lesquels l'échange d'information pour maintenir des tables de routage de faible taille à jour est coûteux.

Protocole AODV créé par les concepteurs de DSDV, un protocole proactif utilisant le routage par vecteur de distance. On s'intéresse ici principalement à un algorithme de routage unicast.

3.3.1 Découverte de routes avec AODV

L'algorithme AODV maintient une table sur chaque noeud, indexée par destination, donnant des informations sur cette destination et notamment le voisin auquel envoyer le paquet pour atteindre cette destination. Si la destination recherchée n'est pas dans la table, on doit donc découvrir une route jusqu'au destinataire. L'algorithme agit à la demande : recherche d'une route uniquement lorsque c'est nécessaire.

Paquet spécial de demande de route diffusé sur le réseau. Paquet qui contient
- l'adresse IP de la source et de la destination (indiquer qui recherche qui)
- un identifiant de requête : compteur local maintenu séparément par chaque noeud et incrémenté à chaque diffusion d'un paquet de demande de route.
- Compteur de séquence = horloge, incrémenté chaque fois qu'un paquet de demande de route ou un paquet de réponse est envoyé. Permet de différencier les nouvelles routes des anciennes. Le paquet de demande contient le numéro de séquence de la source, mais aussi la valeur la plus récente du numéro de séquence de la destination que le noeud source a vu (0 s'il ne l'a jamais vu).
- Enfin, le nombre de sauts : compteur dans le paquet qui sert à enregistrer le nombre de sauts que le paquet a effectué (valeur initiale à 0).

Algorithme de traitement d'un paquet de demande de route (par les voisins de l'émetteur source) :

1. Le couple (adresse source, id de requête) est examiné. A noter que ce couple identifie de façon unique le paquet. Table d'historique locale pour savoir si cette requête a déjà été vue et traitée. Si doublon : ignoré et arrêt du traitement. Sinon, couple inscrit dans la table d'historique pour rejeter de futurs doublons, et poursuite du traitement avec l'étape 2.
2. Recherche de la destination dans la table de routage locale. S'il possède une route récente vers la destination, envoi d'un paquet de réponse à la source en lui indiquant comment atteindre la destination : "passe par

moi !". Une route est "récente" si le numéro de séquence de destination stocké dans la table de routage est supérieur ou égal à celui du paquet de demande. Sinon, la route mémorisée est plus ancienne que la route précédente connue de la source, et on passe à l'étape 3.

3. Le récepteur du paquet ne connaît pas de route vers la destination : incrémente le nombre de sauts et rediffuse le paquet. Table locale des routes inverses : les données du paquet sont extraites pour créer une entrée dans cette table, qui sera utilisée pour construire la route inverse pour envoyer le paquet de réponse au noeud source. Démarrage d'un temporisateur pour la nouvelle entrée de route inverse. S'il expire, l'entrée est supprimée (beaucoup d'entrées qui ne seront pas utilisées sur les "mauvais" chemins employés lors de l'inondation).

Si la requête atteint le noeud destination, un paquet de réponse est construit avec les informations suivantes :

- Adresse source et adresse destination proviennent directement du paquet de demande.
- Numéro de séquence de destination : provient de son compteur en mémoire locale.
- Nombre de sauts : compteur initialisé à 0 qui va compter le nombre de sauts sur la route inverse.
- Durée de vie : champ pour contrôler la validité de la route, initialisé en copiant la valeur *Nombre de sauts* du paquet de demande.

Ce paquet est envoyé en mode unicast (mono-destinataire) au noeud voisin d'où provient le paquet de demande, puis il suit la route inverse notée dans les tables, et enfin atteint le noeud source. Le champ *Durée de vie* est décrémenté à chaque noeud et le paquet est détruit si la durée de vie atteint 0.

Lors de la réception d'un paquet de réponse (le long du parcours inverse vers la source), le paquet est examiné. Une entrée pour la route vers le noeud destination est inscrite dans la table de routage locale si au moins une de ces conditions est satisfaite :

- Aucune route vers la destination n'est connue
- Le numéro de séquence pour la destination dans le paquet de réponse est supérieur à la valeur présente dans la table de routage
- Les numéros de séquences sont égaux mais la nouvelle route est plus courte

D'où un "effet secondaire" de la requête : tous les noeuds sur le chemin inverse découvrent sans effort la route vers la destination. Les noeuds n'étant pas sur le chemin inverse suppriment l'entrée dans la table de routes inverses lorsque le temporisateur associé expire.

Grand réseau : gros volume de diffusions, même pour des destinations proches de l'émetteur.

Possibilité de modifier le processus de découverte pour limiter les inondations : diffusion d'un paquet de demande de route avec un champ *Durée de vie*, initialisé tout d'abord à 1, puis si aucune réponse n'arrive dans un délai raisonnable, on recommence avec un le champ à 2, puis 3,4,5... La recherche commence localement, puis elle est élargie progressivement.

3.3.2 Maintenance des routes

Propriété des réseaux ad-hoc : très grande dynamicité du réseau (variations rapides de topologie).

Diffusion périodique d'un paquet spécial HELLO, et chaque voisin doit répondre. Si un voisin qui était auparavant actif ne répond pas, on sait qu'il ne se trouve plus dans notre portée radio (ou qu'il est déconnecté).

Information utilisée pour purger les routes qui ne marchent plus. Pour chaque destination possible, le noeud N garde en mémoire l'id des noeuds qui lui ont communiqué un paquet à router vers cette destination récemment. Ce sont les *voisins actifs* de N pour cette destination.

N maintient une table de routage indexée par destination, contenant
- le noeud sortant à utiliser pour atteindre la destination,
- le nombre de sauts jusqu'à la destination,
- le numéro de séquence de la destination le plus récent,
- la liste des voisins actifs pour la destination.

Lorsqu'un voisin devient inaccessible en N :
- inspection dans la table de routage des destinations dont les routes passent par le voisin disparu, pour purger les lignes correspondantes de la table de routage.
- récupération des voisins actifs pour ces destinations, pour indiquer à ces voisins actifs que leur route via N est maintenant invalide et doit être purgée de leur table.
- les voisins actifs vont aviser leurs voisins actifs respectifs, et ainsi de suite de proche en proche, et ainsi toutes les routes qui dépendaient du noeud disparu sont purgées des tables de routage.

En fait, lors de la rupture d'un lien d'une route active, AODV tente de réparer la connectivité localement en diffusant une requête de recherche de route dans le voisinage. Si cette tentative échoue, alors la route est purgée comme expliqué précédemment, et une nouvelle recherche de route est lancée par la source.

Différence fondamentale entre AODV et Bellman-Ford classique dans les réseaux filaires : dans AODV les noeuds ne diffusent pas périodiquement la totalité de leur table de routage. Ceci permet d'économiser à la fois la bande passante et les batteries.

3.4 Le routage proactif avec OLSR

Routage proactif : même philosophie que les protocoles de routage utilisés dans les réseaux filaires conventionnels : méthode par états de liens (*link state routing*) et par vecteur de distance (*distance vector routing*). Mise à jour périodique des données de routage, diffusée par les différents noeuds de routage du réseau.

On détaille OLSR, routage ad hoc par états de liens, utilisant un mécanisme de diffusion optimisé en utilisant des multipoints relais. Stabilité de l'algorithme classique à états de liens. Petit rappel sur l'algorithme à états de liens : un noeud découvre ses voisins, et informe tout le réseau de son voisinage par diffusion. OLSR : optimise cet algorithme pour les réseaux mobiles ad-hoc, en réduisant la taille des paquets de contrôle et en minimisant l'inondation du trafic des paquets de contrôle.

Algorithme totalement décentralisé, et s'adapte aux changements de topologie du réseau (grande dynamicité des réseaux ad-hoc).

3.4.1 Inondation optimisée par multipoints relais

Inondation : technique la plus rudimentaire de diffusion, mais brique de base dans de nombreux algorithmes de routage, comme nous l'avons déjà vu. Elle consiste à répéter un message dans tout le réseau : chaque noeud qui reçoit le message pour la première fois répète le message, et le message inonde ainsi le réseau de proche en proche.

Particularité des réseaux radio : bande passante limitée, mais capacité d'atteindre plusieurs noeuds par une seule transmission, ce qui permet d'optimiser le processus de diffusion. Réseau dense (chaque noeud peut atteindre beaucoup de voisins) : possibilité de diffuser un message à n noeuds en moins de n émissions.

OLSR : technique utilisant une connaissance "locale" de la topologie. Un noeud découvre ses voisins directs et ses voisins à deux sauts. Découverte basée sur les paquets "hello".

Messages "hello"

Chaque noeud émet régulièrement un message "hello" contenant la liste des noeuds qu'il entend, message avec une durée de vie $TTL = 1$.

Un noeud possède ainsi la liste des noeuds qu'il entend (les voisins à un saut), et aussi la liste des noeuds entendus par ceux-ci. Cela permet de repérer les liens unidirectionnels, mais surtout d'avoir une vision à deux sauts de la topologie du réseau.

Au noeud u, les voisins de u sont les noeuds entendus par u et dont leur message "hello" contient u (ce qui représente donc un lien symétrique). L'ensemble des voisins de u est noté $N(u)$. Un message "hello" spécifie l'état des noeuds entendus, s'il s'agit d'un lien symétrique ou juste entendu (et cet état est mis à jour).

Les voisins à deux sauts de u sont les noeuds listés comme voisins (lien symétrique) dans les messages "hello" des voisins de u, mais différents de u et de ses voisins. Ces voisins à deux sauts sont dans l'ensemble $N^2(u)$.

Alors, l'idée des multipoints relais consiste à utiliser cette connaissance à deux sauts pour optimiser localement la diffusion. Ainsi, chaque noeud u désigne un ensemble de multipoints relais parmi ses voisins, qui lui permettent d'atteindre tous les noeuds à deux sauts de lui.

Calcul des multipoints relais (MPR)

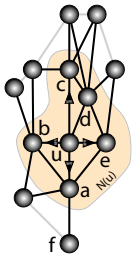
Chaque noeud u élit un ensemble $M(u) \subset N(u)$ de multipoints relais, de sorte que $N^2(u) \subset N(M)$ (avec $N(M) = \cup_{m \in M} N(m)$).

Un noeud est multipoint relais d'un autre, donc c'est une relation binaire.

Idée pour la diffusion : seuls les multipoints relais de u retransmettent un message de diffusion en provenance de u. Dans sa table locale, v garde comme information ses voisins et aussi les voisins pour lesquels il est un multipoint relais.

Ainsi, on atteint le voisinage à deux sauts en $|M(u)| + 1$ émissions. Il faut donc un nombre minimal de multipoints relais pour obtenir un gain maximal.

Sur l'exemple, $\{a, b, c, e\}$ est un ensemble de MPR acceptable pour u (qui n'a pas connaissance des arcs grisés). Cependant, la solution $\{a, b, d\}$ est meilleure. On remarque que a est forcément MPR de u car il est le seul lui permettant d'atteindre f.



Cependant, le problème du choix des MPR est en fait un problème difficile, NP-complet, car cela revient à trouver un *ensemble dominant* dans un graphe (ensemble de noeuds tel que tout sommet du graphe est soit dans l'ensemble, soit voisin d'un noeud de l'ensemble). Or, étant donné le contexte dynamique, on est amené à recalculer cet ensemble assez souvent. Il nous faut donc un algorithme rapide et efficace : trouver une solution facile à implémenter qui donne un résultat proche de l'optimal dans la majorité des cas. Résultat d'inapproximabilité de ce problème : il n'est pas possible d'obtenir une heuristique avec un facteur d'approximation meilleur que $O(\log n)$ dans le cas général.

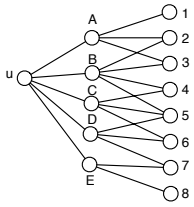
Heuristique pour l'élection des MPR, qui donne une solution optimale à $\log(n)$ près, où n est le nombre de voisins du noeud calculant son ensemble de MPR. L'idée consiste à sélectionner de manière gloutonne les MPR en préférant

ceux qui atteignent un maximum de voisins à 2 sauts non encore atteints par les MPR déjà choisis.

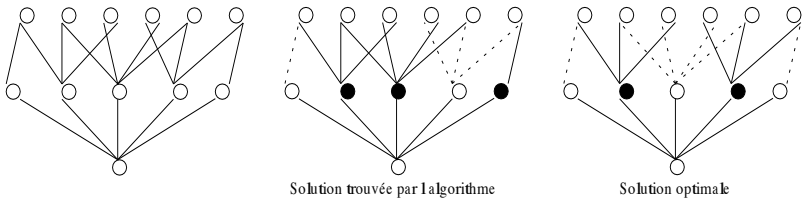
On note par $MPR(u)$ l'ensemble des MPR de u. Alors, l'algorithme de sélection des MPR est le suivant.

1. Initialiser $MPR(u) = \{\}$.
2. Identifier l'ensemble des noeuds de $N(u)$ qui sont les seuls ayant un lien avec un des voisins du second niveau (comme a dans notre exemple, seul lien avec le voisin de second niveau f). Ajouter ces noeuds à $MPR(u)$, et éliminer tous les noeuds de second niveau couverts par ces derniers de $N^2(u)$.
3. Tant que $N^2(u) \neq \{\}$ faire
   (a) Calculer le degré de chaque noeud v dans $N(u)$ (degré = nombre de voisins de $N^2(u)$ couverts par v).
   (b) Choisir un des noeuds de degré maximal et l'ajouter à l'ensemble des relais multipoint $MPR(u)$, et éliminer tous les noeuds de second niveau couverts par celui-ci de $N^2(u)$.

Exemple où l'algo est optimal : on choisit d'abord A et E comme MPR vu qu'ils sont les seuls à atteindre 1 et 8. Cela élimine de $N^2(u)$ les noeuds $1, 2, 3, 7, 8$. Le degré de B est alors 2, celui de C 3 et celui de D 2. On élit donc C, qui couvre les 3 noeuds de $N^2(u)$ qu'il reste.



Contre-exemple où l'optimal est en 2 MPR alors que l'algo en trouve 3.



3.4.2 Algorithme d'inondation (broadcast)

Une fois les ensembles de MPR calculés (et recalculés périodiquement lors de l'échange des messages "hello"), il est très facile de définir un algorithme de routage par inondation pour faire du broadcast (diffusion à tous les noeuds du réseau).

Si v reçoit un message de diffusion par u, et si u est voisin de v (lien qui n'est pas unidirectionnel entre u et v), alors v marque le message comme reçu. Si c'est la première fois que v reçoit ce message, et si $v \in MPR(u)$, et si le TTL du message reste strictement positif après décrémentation, alors v retransmet le message.

Petite subtilité : si v reçoit le message pour la première fois d'un noeud u_1 dont il n'est pas MPR, il ne retransmettra jamais le message, même s'il le reçoit ensuite d'un noeud u_2 dont il est MPR. L'idée est que tout voisin w de v sera quand même atteint, car w est un voisin à deux sauts de u_1 donc u_1 a dû élire un MPR lui permettant d'atteindre w.

On peut alors prouver que tout noeud du réseau sera atteint par une inondation par multipoints relais si l'on suppose que

- tous les voisins d'un émetteur reçoivent systématiquement le message émis,
- les ensembles de MPR sont valides (tout voisin à 2 sauts d'un noeud est couvert par au moins l'un de ses MPR),
- et le graphe défini par les relations de voisinage est connexe.

L'efficacité de cet algorithme a été vérifiée dans plusieurs simulations. Analyse simple de la complexité : si un noeud a d voisins dont m l'ont choisi comme MPR, on peut estimer qu'il émettra le message avec une probabilité de l'ordre de m/d. On s'attend à observer en moyenne nm/d émissions. Si les noeuds sont distribués uniformément, m/d restent proches du nombre moyen de MPR par noeuds/degré moyen. Analyse empirique en accord avec les résultats des simulations pour le modèle du graphe de disques unitaires.

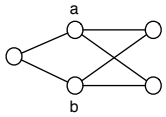
Analyse qui laisse entrevoir un facteur d'optimisation global proportionnel au facteur d'optimisation local, ce qui n'était pas évident à priori.

Fiabilité des MPR

Pour conclure sur la diffusion par MPR, remarquons que les transmissions radios restent peu fiables, notamment en mode broadcast. Il peut donc être intéressant de s'assurer que chaque noeud reçoit plusieurs fois un même message de diffusion pour limiter les problèmes de perte de message.

Les MPR permettent d'intégrer facilement cette variante, en indiquant la redondance k voulue dans la diffusion (*MPR coverage* dans OLSR). On adapte alors l'heuristique de calcul des MPR pour que tout voisin à deux sauts soit atteint par au moins k MPR si cela est possible.

Exemple où $\{a, b\}$ offre un MPR coverage de 2, au lieu de 1 pour $\{a\}$.



Autre possibilité dans OLSR : prendre en compte la propension des noeuds à vouloir relayer (*MPR willingness*). Les noeuds indiquent dans le message "hello" s'ils sont prêts à relayer. Sélection préférentielle de ces noeuds dans le choix des MPR. Dilemme classique : il faut alors inciter les noeuds à coopérer et à bien vouloir servir de relais...

3.4.3 Algorithme de routage unicast

Je rappelle tout d'abord le routage par états de liens classique, utilisé lorsque l'on connaît tout le graphe de connexion.

OSPF - Open Shortest Path First

Chaque noeud découvre ses voisins par l'échange régulier de messages "hello", et diffuse régulièrement dans le réseau la liste de ses voisins. Tous les noeuds peuvent alors reconstruire le graphe complet, et chaque noeud peut calculer un arbre de plus court chemin vers toutes les destinations connues pour construire sa table de routage.

Ce sont les listes de voisins qui sont appelées "états de liens", et lors d'une panne, le noeud qui détecte la panne diffuse immédiatement sa nouvelle liste de voisins qui invalide la précédente.

Problème de cohérence entre les différentes visions du réseau en chaque noeud, lors de la mise à jour. A noter également que seuls les liens symétriques sont utilisés et diffusés.

OLSR : principe et performance

OLSR est également basé sur les états de liens, mais on ne calcule plus des plus courts chemins sur le graphe de connexion entier, mais uniquement à partir d'un sous graphe. Cette technique de sous graphe est particulièrement utile pour les réseaux ad-hoc vu que le graphe de connexion peut être très dense.

Ainsi, un réseau ad-hoc peut posséder $O(n^2)$ liens. Si chaque noeud diffuse régulièrement la liste de ses voisins comme dans OSPF, on obtient un trafic en $O(n^3)$.

Utilisation de multipoints relais : cela multiplie le coût de la diffusion par m/d (m nbre de MR, et d : degré), facteur qui peut atteindre $\log n/n$ dans le cas d'un réseau très dense. De plus, OLSR ne diffuse pas la liste de tous les voisins, mais uniquement la liste de ses MPR sélecteurs, ce qui permet de gagner encore un facteur $\log n/n$.

MPR sélecteurs d'un noeud : liste de ses voisins qui l'ont choisi comme MPR. Les routes sont alors calculées sur le graphe orienté constitué des liens MPR vers MPR sélecteur.

Topologie du réseau

Chaque noeud maintient une base d'informations sur la topologie du réseau. Ces informations sont collectées en analysant les messages de contrôle qui circulent :

- les paquets "hello" contenant la liste des voisins et des MPR : messages émis par un noeud à ses voisins (qui servent aussi à calculer les MPR)
- les paquets "tc" (topology control) qui contiennent, pour un noeud MPR donné, la liste de ses voisins qui l'ont choisi comme MPR : ces messages sont régulièrement diffusés par les noeuds MPR dans tout le réseau, en utilisant l'inondation par multipoints relais décrite précédemment.

Chaque noeud maintient, pour chaque destination dans le réseau, des tuples (*dest*, *last*, *seq*, *time*). L'adresse de la destination est *dest* ; *last* est un relais multipoint de *dest*, *seq* est un numéro de séquence et *time* le temps au bout duquel ce tuple expire et doit être détruit. Ces entrées dans la table de topologie sont créées à partir des informations contenues dans les paquets "tc" : *dest* est un des MPR sélecteurs dans un paquet "tc", *last* est le noeud qui est à l'origine du paquet "tc", et le numéro de séquence *seq* est le numéro associé à l'ensemble des MPR sélecteurs dans le paquet "tc".

Le calcul de la table de routage est alors basé sur les informations présentes dans la base d'informations de topologie, et aussi sur les informations de voisinage local.

Tables de routage

Les tables de routage contiennent les informations (*dest*, *next*, *dist*), indiquant pour chaque destination *dest* le prochain saut à effectuer *next* et la distance estimée *dist*, en nombre de sauts, séparant *dest* du noeud local.

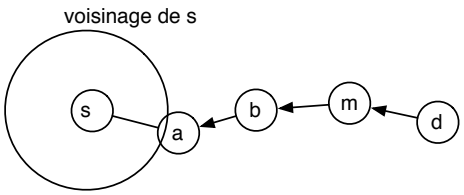
Mise à jour de la table de routage localement à chaque modification des informations de voisinage ou de topologie. Aucune génération de messages dans le réseau : il s'agit d'un simple calcul local.

Algorithme de calcul de la table de routage :

1. Détruire toutes les entrées ultérieures de la table
2. Insertion dans la table de tous les voisins directs (voisins avec un lien symétrique, à distance 1 saut).
3. Boucle pour h initialisé à 1 et incrémenté de 1 à chaque étape, jusqu'à ce qu'il n'y ait plus de nouvelles entrées dans la table de routage. A chaque étape, on insère les destinations à $(h+1)$ sauts.
   (a) Pour chaque tuple (Td,Tl,Ts,Tt) dans la base des informations de topologie, si Td n'est pas une entrée de la table de routage, et si Tl correspond à une entrée de la table de routage (Tl, *next*, h), alors on insère une nouvelle entrée dans la table de routage (Td, *next*, $h+1$).

Cet algorithme correspond à une construction de l'arbre des plus courts chemins dans le sous-graphe décrivant la topologie du réseau.

Exemple sur la figure suivante :



Le noeud source s cherche à atteindre le noeud destination d. d a choisi des MPR, dont m, donc m diffuse sur le réseau le fait que d est un de ses MPR sélecteurs, ce qui correspond à un arc $m \rightarrow d$ dans le sous-graphe de topologie, connu dans les informations de topologie, notamment au noeud source s. Noter que l'on utilise donc les arcs dans le sens inverse en comparaison du sens utilisé lors de la diffusion par MPR.

Preuve de plus court chemin

Les routes sont calculées sur un sous-graphe. Il reste à prouver que ce sont des routes de plus court chemin pour le graphe de connexions complet.

Montrons par récurrence sur d que tout noeud à distance d est atteint par une route optimale avec la méthode de construction des tables de routage expliquée précédemment.

Pour $d = 1$, les routes sont optimales car les liens avec les voisins sont connus (étape 2 de l'algorithme de calcul de la table de routage).

Si la propriété est vraie pour $d \geq 1$, considérons un noeud u à distance $d+1 \geq 2$. u possède un voisin v à distance d, et v un voisin w à distance $d-1$. u doit donc posséder un MPR m qui couvre w. Or m est à distance au plus d, donc par hypothèse de récurrence, il existe une route optimale de longueur d jusqu'à m, et l'entrée (m, $next$, d) dans la table de routage. Dans les informations de topologie, le noeud source sait que u est un MPR sélecteur de m, d'où un tuple (u, m, seq, $time$). A l'étape 3(a) de l'algorithme, on insérera donc l'entrée (u, $next$, $d+1$) dans la table de routage, qui correspond à la route optimale de longueur $d+1$ qui permet d'aller à u en passant par m.

Vers une sous-topologie optimale

Ainsi, non seulement les MPR optimisent la diffusion, mais également ils définissent une sous-topologie du réseau suffisante pour la construction de routes optimales.

Défaut par rapport à OSPF : un noeud ne peut pas calculer la table de routage d'un autre noeud dont il ne connaît pas le voisinage. De tels calculs sont utilisés par exemple dans le protocole de routage multicast basé sur OSPF.

Autre question : est ce qu'on pourrait trouver une meilleure topologie que celle des MPR ? Déjà, l'inclusion des liens avec les voisins est nécessaire dans la sous-topologie. Ensuite, si l'on considère un noeud u à distance 2, il est nécessaire qu'un lien avec un des voisins soit connu pour trouver une route optimale. Une hypothèse raisonnable consiste à dire que tous les noeuds connaissent les mêmes liens pour les noeuds à plus de deux sauts (comme cette information doit être diffusée, l'hypothèse semble raisonnable). L'ensemble des liens qui connectent à u et qui sont diffusés dans le réseau doit ainsi forcément définir un ensemble de MPR pour u, car tout noeud à distance 2 de u doit pouvoir atteindre u par un de ces liens.

La notion de MPR apparaît donc de façon intrinsèque dans le problème de diffusion d'une sous-topologie la plus petite possible, mais suffisante pour construire des routes optimales.

Pannes et utilisation des MPR sélecteurs

Pourquoi diffuse-t-on la liste des MPR sélecteurs et non pas simplement les MPR ? La raison est de pouvoir réagir de façon efficace en cas de panne, car la cassure d'un lien doit être répercutée sur la vision générale de la topologie assez rapidement s'il s'agit d'un lien entre un noeud et un MPR. Dans ce cas, le MPR peut diffuser rapidement une nouvelle liste dans laquelle le lien cassé n'apparaît plus.

Par exemple si le lien $b \leftarrow a$ est cassé, a est MPR et b est un de ses MPR sélecteurs, donc a va diffuser immédiatement une nouvelle liste sans b, et la source s de trafic vers d est assurée de recevoir cette nouvelle liste et peut mettre à jour sa table de routage.

Si c'est le MPR sélecteur qui diffusait la liste de ses MPR, en cas de rupture du lien ça serait à b de diffuser la nouvelle, et il est moins évident que la nouvelle arrive à s rapidement (et, plus généralement, aux sources de trafic utilisant ce lien). Il faudrait pour cela que b commence par recalculer ses MPR, puis informe ses nouveaux MPR par des messages "hello", avant de pouvoir effectuer une diffusion qui atteigne les sources de trafic concernées. Et en cas de déconnexion du réseau par la cassure du lien, le noeud n'a aucune chance de pouvoir les contacter.

3.5 Le routage multicast

La plupart des applications utilisent du routage unicast, mais le multicast est en train d'évoluer régulièrement, son utilité ne cesse de se justifier, par exemple dans des applications de téléconférence ou de jeux en réseaux.

Difficulté dans la conception d'un protocole de routage multicast : choisir la structure à mettre en place.

- arbre de diffusion, qui peut être centralisé et partagé ou bien avoir un arbre par source,
- utiliser une grille reliant tous les membres d'un groupe,
- inondation en essayant d'optimiser,
- ...

Les choix de structures sont différents des choix effectués dans le cadre des réseaux filaires. On étudiera ici les difficultés propres aux réseaux sans fil.

Comme pour le routage unicast, il existe des protocoles réactifs ou proactifs, et on s'intéresse ici rapidement au routage multicast dans AODV, puis dans OLSR.

3.5.1 MAODV : Multicast AODV

MAODV est basé sur AODV, et fonctionne de manière réactive également, à la demande. Il utilise le même principe que AODV, et les mêmes formats de requêtes de recherche de routes.

Le protocole MAODV construit un arbre partagé centré sur un noyau (le leader du groupe) pour chaque groupe multicast du réseau. Le noeud leader est le premier participant au groupe, il est chargé de le gérer et de le maintenir en place. L'arbre peut contenir des noeuds qui ne font pas partie du groupe. Un numéro de séquence est associé à chaque groupe.

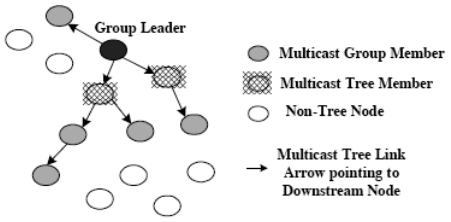


Table de routage

Un noeud MAODV maintient une table de routage comme dans AODV pour le routage unicast, mais aussi une table de routage pour la structure en arbre du groupe. Cette table contient l'adresse du groupe multicast (identifiant du groupe), le numéro de séquence, l'adresse du leader du groupe, le nombre de sauts jusqu'au leader, et les informations sur le saut suivant, qui correspondent aux voisins qui sont dans l'arbre. Les voisins peuvent être fils ou père, chaque noeud possède au plus un père, et le leader ne peut posséder des fils que s'il y a d'autres membres dans le groupe.

Message de requête

Les messages de demande de routes et de réponse sont ceux de AODV, adaptés pour le multicast. Une requête est envoyée lorsqu'un noeud "source" désire envoyer un message multicast. Le même type de message de requête REQ est utilisé lorsqu'un noeud désire rejoindre un groupe multicast. Les deux types de requêtes sont différenciées par un flag dans le message. Dans les deux cas, il s'agit donc de trouver une route vers l'arbre multicast.

Si le noeud source connaît le leader du groupe, il peut envoyer son message directement au leader en mode unicast. Sinon, le message REQ est relayé saut par saut comme dans AODV jusqu'à trouver un noeud qui appartient déjà à l'arbre multicast, et ayant un numéro de séquence supérieur ou égal à celui contenu dans le message de la requête. Le noeud récepteur de la requête répond alors en mode unicast au noeud demandeur, par un message de réponse REP comme les messages AODV.

Les requêtes sont envoyées avec des TTLs de plus en plus grands, comme pour les requêtes AODV unicast de recherche de route, et si le noeud n'obtient pas de réponse après plusieurs tentatives de connexion à un arbre multicast, il se considère comme étant le premier membre de ce groupe, et il initialise le numéro de séquence du groupe.

Réponse et activation de branches

Dans AODV, le message de réponse à une demande de route active directement la route, alors que dans MAODV, la réponse indique seulement une route possible jusqu'à l'arbre, sans l'activer. En effet, une seule route peut être rattachée à l'arbre, ou une branche rajoutée à l'arbre si le noeud rejoint le groupe, afin d'éviter les boucles.

Le noeud qui a émis la requête attend une certaine période, puis choisit la meilleure route à utiliser pour atteindre l'arbre parmi les réponses qu'il a reçues. Il envoie alors un message MACT pour activer la route jusqu'à l'arbre, ou activer une nouvelle branche de l'arbre multicast. La meilleure route est la plus récente (en regardant le numéro de séquence), et la plus courte parmi ces dernières (en terme de nombre de sauts jusqu'à l'arbre).

Opérations de maintenance

Les opérations de maintenance lors de départ de noeuds sont plus complexes en multicast que pour le protocole AODV unicast.

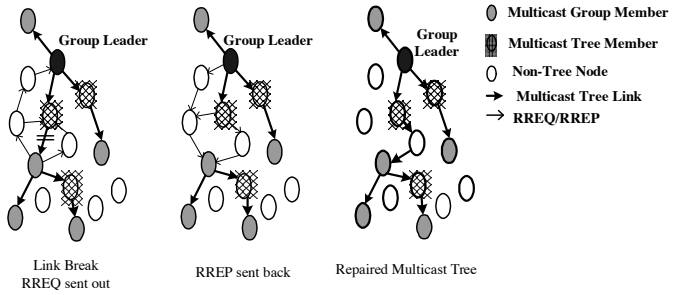
Le leader du groupe envoie périodiquement un message Hello de groupe. Ce message est broadcasté dans tout le réseau, pour indiquer l'adresse du leader du groupe et le numéro de séquence courant du groupe. Le numéro de séquence de groupe est incrémenté avant chaque envoi de ce message Hello.

Pour un départ d'un membre du groupe, si le noeud n'est pas une feuille de l'arbre, il doit rester dans l'arbre et faire office de routeur pour le groupe multicast. Sinon, il faut enlever la branche de l'arbre désormais inutilisée, cela se fait par l'envoi d'un message MACT avec un flag spécial. Les noeuds servant

de routeur pour le noeud qui quitte le groupe sont également supprimés de l'arbre, et ce jusqu'à ce qu'un membre du groupe soit rencontré.

Les problèmes propres au ad-hoc sont bien sur les ruptures de lien dues à la mobilité, et cela peut entraîner des envois à une partie seulement du groupe multicast. Les ruptures de lien peuvent être détectées de la même façon que dans AODV, mais AODV se contente d'effacer la route des tables de routage et de rechercher une nouvelle route, alors que dans MAODV la procédure a lieu localement, seul le noeud fils du lien cassé peut initier la procédure de réparation. Le noeud fils envoie en mode broadcast un message de requête REQ en demandant à rejoindre l'arbre, indiquant également son nombre de sauts jusqu'au leader (pour éviter qu'un de ses descendants renvoie une réponse). Lorsqu'une réponse est obtenue, le noeud envoie un MACT pour raccorder sa branche à l'arbre, et il envoie également un MACT avec un flag de mise à jour à ses descendants pour qu'ils puissent changer leur compteur de sauts jusqu'au leader du groupe.

Exemple :



Si aucune réponse n'est reçu après une tentative de re-connexion de la branche (après plusieurs essais), la source suppose alors que l'arbre est partitionné, et il faut sélectionner un nouveau leader pour l'arbre partitionné. Si le noeud source est un membre du groupe, il devient le leader. Sinon, il oblige un de ses descendants à devenir le leader.

Il arrive aussi que deux arbres d'un même groupe multicast retrouvent une connexion. Dans ce cas, il faut fusionner les deux arbres et décider du nouveau leader (un des deux leaders).

3.5.2 MOLSR : Multicast OLSR

MOLSR est l'extension multicast du protocole de routage OLSR. Il utilise les différentes tables (voisinage, topologie, routage) pour minimiser le trafic de contrôle et économiser la bande passante. Possibilité d'utiliser MOLSR dans un environnement hétérogène où certains noeuds ne supportent que OLSR (trafic multicast relayé par les noeuds OLSR).

On ne dispose pas comme pour AODV d'un arbre multicast du groupe, avec un leader de groupe, mais on construit et maintient un arbre multicast pour chaque couple (source, groupe). Ceci est fait de façon totalement distribuée

(pas d'entité centrale, de leader), et permet de définir les plus courts chemins de la source vers les membres du groupe. On parle d'*arbres à base de source*.

OLSR réagit aux modifications de topologie en recalculant les MPR et la table de routage. MOLSR exploite ces mises à jour pour optimiser les arbres multicast et détruire les branches obsolètes.

Informations locales

- Un noeud conserve localement l'identifiant des groupes multicast dont il fait parti.
- La table des arbres multicast maintient en chaque noeud les informations sur les différents groupes multicast auxquels ce noeud participe. Chaque entrée de la table contient des renseignements sur l'arbre défini par (source, groupe) : l'adresse de la source, l'adresse du groupe multicast, l'adresse du père du noeud dans l'arbre, et la liste des fils de ce noeud. A chaque noeud dans la table, on associe également une durée de vie. Comme pour AODV, un noeud peut ne pas être membre d'un groupe, mais faire parti de l'arbre pour router les messages vers les abonnés.
- La table de routage multicast indique les plus court chemins vers tous les noeuds multicast (identique à la table de routage unicast, mais n'utilise que les noeuds gérant le multicast, ces noeuds étant connus sur le réseau car l'information est diffusée). Cette table est recalculée à chaque fois que la table de routage OLSR est mise à jour, ou bien lorsqu'un nouveau noeud multicast est découvert.

Construction de l'arbre multicast

Les routeurs multicast diffusent dans le réseau le message MC-Claim (en utilisant la diffusion optimisée d'OLSR par l'intermédiaire des MPR). Chaque noeud effectuant des opérations multicast connaît ainsi les autres membres du réseau multicast.

Lorsqu'une source désire émettre des données multicast vers un groupe spécifique G, elle diffuse un message Source-Claim pour que les membres du groupe détectent sa présence et se rattachent à l'arbre associé à cette source. L'invitation de la source est diffusée par MPR.

Les branches de l'arbre sont construites en partant des feuilles. Lorsqu'un membre du groupe reçoit le message Source-Claim, et s'il ne fait pas partie de l'arbre définit par (source, G), alors il se rattache automatiquement à l'arbre :

- Il regarde dans sa table de routage multicast pour choisir le prochain saut qui lui permet d'atteindre la source en un nombre minimal de sauts. Ce noeud devient son parent dans cet arbre multicast.
- Il envoie un message pour confirmer à son parent qu'il devient son fils.
- Le noeud parent qui reçoit un tel message se rattache à son tour à l'arbre (source, G) s'il n'appartient pas à ce dernier, en suivant la même procédure. De plus, il actualise sa liste de fils.

La construction et l'activation de la branche se fait donc de façon récursive jusqu'à ce que la source ou un participant de l'arbre soit atteint.

Maintenance des arbres

Les arbres sont rafraîchis périodiquement par l'intermédiaire des messages de confirmation de parents et par les Source-Claim. Lorsqu'un noeud reçoit un Source-Claim et qu'il appartient déjà à l'arbre multicast, il se contente de mettre à jour les informations de durée de vie.

Les modifications de topologie sont détectées par l'échange de messages de contrôle d'OLSR. Chaque noeud répercute alors ces modifications également sur les structures multicast qu'il gère. Lors d'un changement dans le voisinage ou dans la topologie du réseau, la table de routage multicast doit être recalculée. Ces changements dans la table de routage peut amener à devoir changer un parent dans l'arbre (route vers la source). Pour cela, les tables sont mises à jour et éventuellement il notifie l'ancien parent qu'il quitte cette branche.

Les relations de parenté sont également mises à jour régulièrement grâce aux messages que chaque participant à l'arbre envoie régulièrement à son père.

Destruction des arbres

Un noeud peut quitter un groupe multicast, ou bien seulement une branche d'un arbre. Dans tous les cas, il est amené à détruire différentes branches de l'arbre et des liaisons entre les noeuds appartenant à l'arbre.

Un noeud quitte une branche spécifique s'il découvre une meilleure route vers la source, comme indiqué dans la maintenance de tables.

Lorsqu'un noeud quitte un groupe, il doit quitter tous les arbres multicast concernant ce groupe. Pour cela, il doit détruire toutes les branches de ces arbres. Un noeud qui est une feuille peut quitter l'arbre multicast par un simple message à son père (message Leave). Si le père devient une feuille et s'il n'est pas abonné au groupe, il effectue la même opération de destruction de branche. Sinon, s'il est sur la route pour atteindre d'autres membres du groupe pour certaines sources, il doit rester dans l'arbre pour effectuer le routage.

Lorsqu'il quitte un groupe, un noeud peut envoyer un message Leave à tous ses parents pour tous les arbres auxquels il appartient, en une seule diffusion.

Fiabilité des arbres multicast

Pour rendre plus fiable OLSR, on a vu comment on introduit de la redondance et de la diffusion multiple. De la même façon, on peut rendre les chemins multicast plus robustes et plus fiables en dédoublant les branches de l'arbre. On possède ainsi un chemin de secours en cas d'erreur sur un lien.

Ainsi, en plus de choisir son père, un noeud choisit également un *oncle* dans l'arbre, voisin de son père si possible. L'oncle est également à portée de communication du grand-père (le noeud connaît son voisinage à deux sauts), et ainsi le grand-père peut transmettre les données au noeud en passant soit par le père soit par l'oncle.