

Université de Nice - Sophia Antipolis
DEA Réseau et Systèmes distribués

Rapport de Stage

Transmission vidéo multipoint sur Internet

Présenté par
AL HAMRA Anwar

Sous la direction de
Thierry TURLETTI

Laboratoire d'accueil
Projet Planète, INRIA Sophia Antipolis

Soutenu le 2 juillet 2001

Remerciements

Je voudrais exprimer toute ma gratitude et tout mon respect à mon encadreur Monsieur Thierry Turletti, qui m'avait accueilli au sein du Projet Planète. Il m'a suivi de près et il a fait de son mieux pour que mon stage réussisse.

Je tiens à remercier le responsable du DEA, Monsieur Philippe Nain, et le responsable des stages, Monsieur Michel Riveill, pour leur sage direction et leurs aides inappréciables.

Je remercie aussi tous mes enseignants pour les informations qu'ils nous ont donné pendant cette année et pour leur générosité.

Merci à tous les membres de l'équipe pour la bonne ambiance qu'ils font régner.

Enfin, un merci du fond du cœur à tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

À mes parents...

À Wissam et Nizar...

TABLE DES MATIERES :

INTRODUCTION _____	1
--------------------	---

Chapitre 1 _____ TRANSMISSION MULTIPOINT VIDEO SUR INTERNET, ETAT DE L'ART

Introduction _____	3
1.1- Calcul de la bande passante allouée au flux _____	3
1.1.1- Calcul du taux de perte _____	4
1.1.2- Calcul du RTT _____	4
1.2- Scalabilité _____	5
1.3- Hétérogénéité des récepteurs _____	7
1.4- Conclusion _____	12

Chapitre 2 _____ UNE SOLUTION A LA TRANSMISSION MULTIPOINT

Introduction _____	13
2.1- Real Time Transport Protocol (RTP) _____	13
2.1.1- RTP Data Transfer Protocol _____	14
2.1.2- RTP Control Protocol (RTCP) _____	14
2.2- Organisation des agrégateurs dans le réseau _____	15
2.3- Classification des récepteurs _____	17
2.3.1- Préliminaires _____	17
2.3.2- Algorithme de classification des agrégateurs _____	18
2.4- Travail existant _____	19
2.5- Travail du stage _____	20
2.5.1- Calcul du RTT _____	20
2.5.2- Détection et résolution des collisions des SSRC et CNAME _____	23
2.5.3- Synchronisation des abonnements aux couches (join and leave) _____	24
2.6- Résultats attendus _____	25
2.7- Conclusion _____	26

Chapitre 3 _____ COMPARAISON DE L'APPROCHE [12, 13] AVEC LES APPROCHES EXISTANTES

Introduction _____	27
2.1- L'approche de [12, 13] versus l'approche de [1] _____	27
2.2- L'approche de [12, 13] versus les deux approches de [10] _____	28
2.3- L'approche de [12, 13] versus MLDA _____	29
2.4- Conclusion _____	37

Chapitre 4 _____ EXPERIMENTATIONS ET RESULTATS

Introduction _____	39
4.1- Expérimentations et résultats _____	39
4.2- Interprétation des résultats _____	43
4.3- Conclusion _____	44

_____ CONCLUSION

INTRODUCTION

La transmission vidéo même en point à point sur Internet n'assure aucune garantie au niveau de la perte, la bande passante et du délai représente un grand challenge, d'où la nécessité d'utiliser des protocoles de contrôle qui adaptent l'application à l'état du réseau. La principale difficulté pour adapter l'application aux conditions du réseau est due à l'hétérogénéité des récepteurs, donc la source, pour qu'elle puisse adapter son débit en fonction de l'état du réseau, a besoin des informations autour les états de réception des récepteurs dans la session.

Par suite, chaque récepteur doit envoyer un rapport de réception à la source dans lequel il indique sa bande passante disponible et son taux de perte.

Le récepteur peut calculer facilement son taux de perte, il suffit qu'il surveille la réception du flux de données sur une certaine période.

La majorité des trafics sur Internet sont des trafics TCP, par suite, le modèle utilisé par l'application pour calculer la bande passante disponible dans le réseau doit être équitable avec les autres trafics TCP sur Internet.

Les modèles TCP sont souvent utilisés, ils allouent au flux de données la même bande passante allouée avec le protocole TCP pour ce même flux de données dans les mêmes conditions du taux de perte et du délai.

Avec les modèles TCP, la bande passante disponible du récepteur est calculée en fonction de son taux de perte et le temps aller retour (RTT) entre lui et la source.

La solution la plus simple pour calculer le RTT entre les récepteurs et la source, est que chaque récepteur envoie un paquet de contrôle à la source, et le RTT sera calculé comme étant le temps écoulé entre le temps d'émission du paquet à la source par le récepteur, et la réception de l'acquittement correspondant de la source.

Cette solution est applicable pour les sessions avec un seul récepteur ou avec un nombre très petit de récepteurs, mais pour une session avec un grand nombre de récepteurs, cette solution présente un problème de scalabilité et gaspillage de la bande passante du réseau.

Par suite, les rapports de réception des récepteurs gaspillent de la bande passante dans le réseau et peuvent causer une congestion dans le réseau et un problème de scalabilité, surtout dans le cas des sessions avec un grand nombre de récepteurs.

Dans ce rapport, on adresse en détail tous les problèmes de la transmission vidéo multipoint sur Internet, et les solutions proposées, ainsi qu'on présente une solution qui assure une adaptation du débit de la source, et en même temps, il ne présente pas un problème de scalabilité des rapports de réception, ni un gaspillage de la bande passante dans le réseau.

Dans le chapitre suivant, on cite en détail ces problèmes et quelques solutions proposées. Dans le chapitre 2, on présente le protocole de transport RTP, et on détaille notre approche présentée dans [12, 13] qui présente une solution à la transmission multipoint. Notre approche propose la transmission des données en plusieurs couches et chaque récepteur évalue sa bande passante disponible et choisit les couches à joindre.

Ainsi, dans le chapitre 2 je présente le travail de mon stage qui est fait au sein du Projet PLANETE de l'INRIA Sophia Antipolis.

Dans mon stage j'ai étudié les problèmes de la transmission multipoint sur Internet, j'ai adressé le problème de synchronisation de l'abonnement des récepteurs aux différentes couches de données, ainsi que la détection et la correction des collisions entre les identificateurs alloués par le protocole RTP à chaque élément de la session (section 2.6).

Dans mon travail j'ai implémenté un algorithme de calcul de RTT entre les récepteurs et la source proposé dans notre approche.

Cet algorithme sert à calculer les bandes passantes TCP-Friendly des récepteurs [18].

les flux vidéos ont besoin d'un protocole de contrôle de congestion qui évite les variations brusques de débit, (c'est qui est pas le cas pour le TCP). En même temps, ce protocole de contrôle doit être équitable au niveau de la bande passante en comparaison avec les autres protocoles de contrôle.

Or, la majorité du trafic Internet est du trafic TCP, donc le protocole utilisé doit allouer au flux vidéo (sur un intervalle du temps, qui peut être long), la même bande passante que TCP alloue à ce même flux, sous les mêmes conditions du réseau, avec le même taux de perte et le même temps aller retour (RTT).

L'intérêt du calcul de la bande passante TCP-Friendly des récepteurs est qu'on peut adapter le débit de l'application aux capacités disponibles dans le réseau, en évitant les variations brusques de débit, ainsi qu'on assure un partage équitable de la bande passante du réseau entre les différentes applications, ce qui peut répondre aux besoins des applications vidéos sur Internet.

De plus, j'ai analysé les performances de cet algorithme et je l'ai comparé avec d'autres approches.

Dans le troisième chapitre, on présente une comparaison entre notre approche et quelques solutions proposées à la transmission multipoint.

Dans le quatrième chapitre, on présente les résultats trouvés suite à des expérimentations, et une conclusion est présentée dans le dernier chapitre.

CHAPITRE I

TRANSMISSION VIDEO MULTIPOINT SUR INTERNET ETAT DE L'ART

Introduction :

La transmission vidéo multipoint sur Internet vers des récepteurs hétérogènes est problématique. Elle nécessite des algorithmes adaptatifs pour d'une part, lutter contre la perte de paquets sur Internet, et d'autre part, adapter le débit de réception des flots vidéos aux caractéristiques de chaque récepteur.

Pour que la source puisse adapter le débit du flux de données aux caractéristiques des récepteurs, les récepteurs doivent envoyer à la source des rapports de réception dans lesquels ils indiquent leurs bandes passantes disponibles et leurs taux de perte moyens.

D'autre part, les flux vidéos ont besoin d'un protocole de contrôle de congestion qui évite les variations brusques de débit, (c'est qui est pas le cas pour le TCP). En même temps, ce protocole de contrôle doit être équitable au niveau de la bande passante en comparaison avec les autres protocoles de contrôle.

Or, la majorité du trafic Internet est du trafic TCP, donc le protocole utilisé doit allouer au flux vidéo (sur un intervalle du temps, qui peut être long), la même bande passante que TCP alloue à ce même flux, sous les mêmes conditions du réseau, avec le même taux de perte et le même temps aller retour (RTT).

L'intérêt de l'utilisation d'un protocole de contrôle qui calcule la bande passante disponible à un flux de données est majeur.

Un flux de données qui est transmis sans aucun contrôle de débit, peut affecter les flux de données transmis avec débits contrôlés. Dans le cas où une congestion apparaît dans le réseau, tous les flux contrôlés diminuent leurs débits, et les flux sans contrôle continuent à émettre avec leurs même débit, en ignorant l'état de congestion présenté, ce qui peut conduire à des congestions de large durée, ainsi qu'ils peuvent dominer à la bande passante disponible dans le réseau.

Pour récapituler, les problèmes principaux de la transmission vidéo multipoint sur Internet, sont les suivants :

- 1- Le calcul de la bande passante allouée au flux.
- 2- Le problème de scalabilité des rapports de réception des récepteurs.
- 3- L'hétérogénéité des récepteurs dû à leurs états de réception différents.

1.1- Le calcul de la bande passante allouée au flux:

Pour des raisons d'équité avec le trafic Internet, qui est en majorité, composé de flots TCP, l'algorithme utilisé peut calculer la bande passante allouée au flux vidéo, avec un modèle TCP [18], en fonction du taux de perte et du RTT.

1.1.1 Calcul du taux de perte :

Le calcul du taux de perte ne pose aucun problème, il suffit de surveiller la réception des paquets de données par le récepteur lui-même, et à l'aide du numéro de séquence de chaque paquet, le récepteur peut arriver facilement à calculer son taux de perte.

1.1.2 calcul du RTT :

La méthode la plus simple pour calculer le RTT, est que l'un des deux extrêmes (la source ou le récepteur), envoie un message à l'autre extrême. Le RTT est égal à deux fois la différence entre l'instant de réception de ce message et son instant d'émission. Mais cette méthode nécessite des horloges synchronisées entre les deux extrêmes. D'autre part, en réalité les réseaux sont souvent asymétriques, et dans ce cas, le temps d'aller n'est pas le même que le temps de retour, ce qui rend le calcul du RTT imprécis.

Une solution possible, est que l'un des deux extrêmes envoie un message en mémorisant son instant d'émission, et que l'autre répond avec un acquittement dans lequel est indiqué le T_{dlsr} , qui est la différence (en valeur absolue) entre l'instant de réception du message, et l'instant de génération de l'acquittement correspondant.

Le RTT est égal à:

$$RTT = T_{\text{réception de l'acquittement}} - T_{\text{émission du message}} - T_{dlsr} \quad \text{éq.(a)}$$

Avec $T_{dlsr} = | T_{\text{réception du message}} - T_{\text{émission de l'acquittement correspondant}} |$

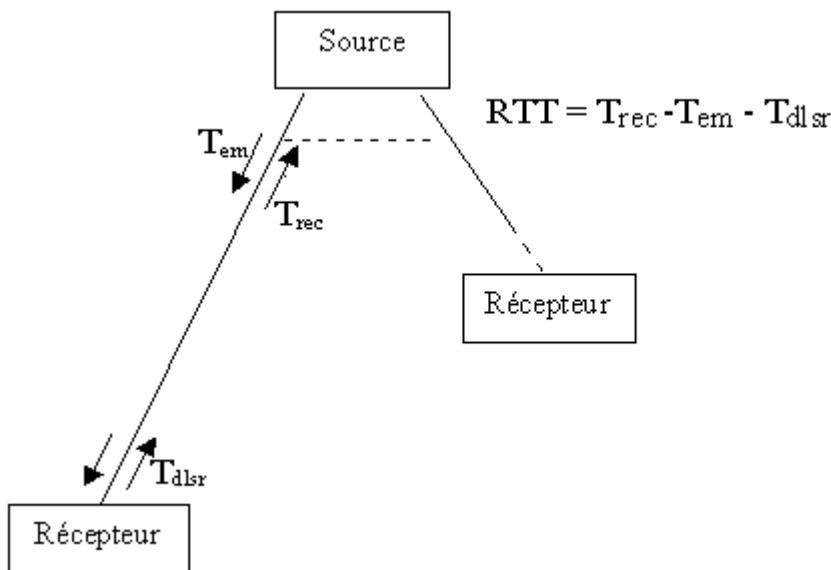


Fig. 1.1 Schéma représentant les différents paramètres dans le calcul de RTT.

Avec cette méthode, tous les récepteurs doivent échanger des messages de contrôle avec la source.

Ces messages envoyés par les récepteurs présentent un problème de scalabilité et un gaspillage de la bande passante du réseau et surtout dans le cas de sessions avec un grand nombre de récepteurs.

1.2 Scalabilité :

Au passage à l'échelle, la scalabilité est un paramètre très important qu'il faut prendre en compte lors de l'implémentation d'un algorithme ou d'une discipline.

Dans notre cas, le calcul du RTT présente un problème de scalabilité des rapports des récepteurs.

De nombreux algorithmes ont été proposés pour résoudre le problème de rapports de récepteurs [1,2].

[1] présente un algorithme proposé par Anindya Basu, S.Jamaloddin Golestani, mais avec une hypothèse forte, sur la distribution de récepteurs dans le réseau.

Comme le montre la figure 1.2, les récepteurs se distribuent dans le réseau sous la forme d'un arbre.

Avec cet algorithme, la source diffuse un message, et le RTT pour chaque récepteur est calculé comme étant le temps écoulé entre la diffusion de ce message, et l'instant de réception par la source de l'acquittement correspondant du récepteur.

L'algorithme propose une méthode qui permet à chaque récepteur de calculer facilement les RTT entre lui et chacun de ses fils (RTT différentiel).

Le récepteur, après la réception d'un acquittement de son fils, calcule le RTT différentiel de ce dernier.

Périodiquement, chaque récepteur envoie un acquittement agrégé au récepteur de niveau supérieur dont lequel il indique le RTT différentiel de chaque récepteur duquel il a reçu un acquittement.

La manière de distribution des récepteurs dans le réseau, présente en même temps le point fort et le point faible de cet algorithme.

D'une part, cette distribution assure la fusion des acquittements des récepteurs, ce qui évite le problème de scalabilité de ces acquittements.

De l'autre part, avec une autre distribution des récepteurs, par exemple des récepteurs distribués dans le réseau comme des feuilles, la fusion des acquittements des récepteurs devient inefficace, par suite, on aura un problème de scalabilité de ces acquittements.

Ainsi que chaque récepteur envoie son acquittement au récepteur de niveau supérieur (ou à la source s'il est en face d'elle), et ce dernier attend un certain temps avant l'émission de son acquittement agrégé au récepteur de niveau supérieur (ou à la source), et ainsi de suite jusqu'à l'arrivée à la source, donc, on a un retard ajouté au temps nécessaire pour que l'acquittement du récepteur arrive à la source, ce qui rend le calcul des RTT des récepteurs imprécis.

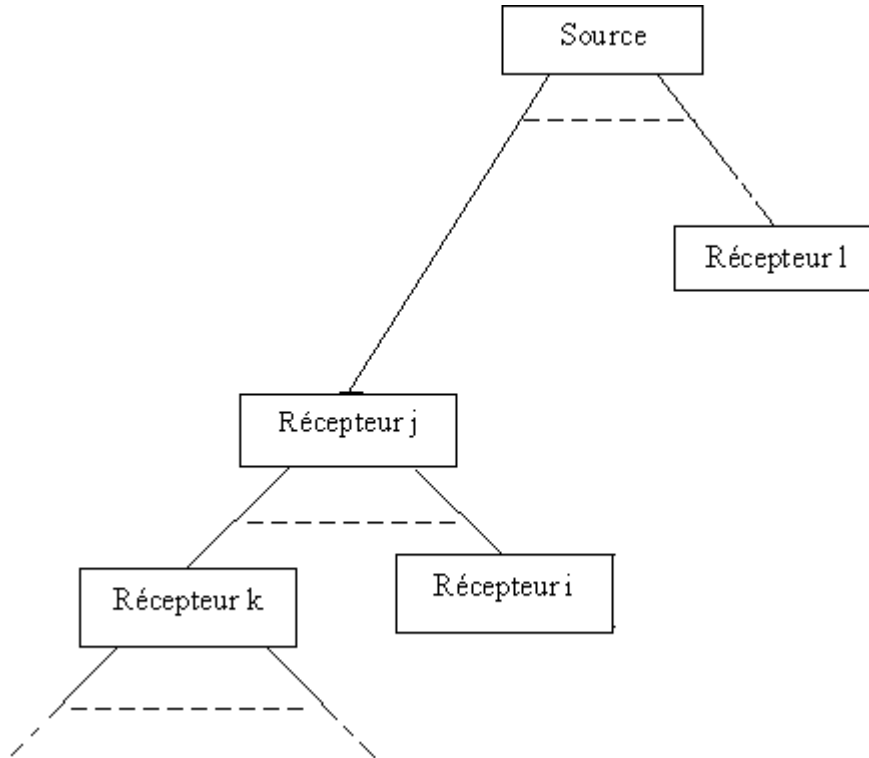


Fig. 1.2 La distribution des récepteurs dans le réseau

[2] présente une solution proposée par T.Turletti, J.Bolot et I.Wakeman, avec laquelle, chaque récepteur envoie son rapport avec une probabilité qui est fonction du nombre de récepteurs.

Aussi, une autre solution efficace au problème de scalabilité, est l'utilisation des agrégateurs au sein du réseau[10,11].

Les agrégateurs se distribuent dans le réseau sous la forme d'un arbre, dont les feuilles s'appellent des *agrégateurs feuilles*, et la racine, *l'agrégateur final*.

Le rôle de ces agrégateurs, est d'agréger les rapports des récepteurs, et de les fusionner en un seul rapport. Plusieurs paramètres sont pris en compte dans cette opération de fusion.

Un agrégateur feuille est élu pour chaque groupe des récepteurs qui se trouvent dans une même région locale.

Chaque récepteur envoie son rapport à son agrégateur feuille, et ce dernier, chaque fois, il reçoit un rapport d'un récepteur, il fusionne avec les autres, et envoie un rapport final à l'agrégateur de niveau supérieur, et ainsi de suite, jusqu'à l'arrivée à *agrégateur final*, qui délivre à la source un rapport final agrégé.

Avec cette solution, tous les récepteurs peuvent envoyer leurs acquittements, et le nombre d'agrégateurs est fonction du nombre de récepteurs.

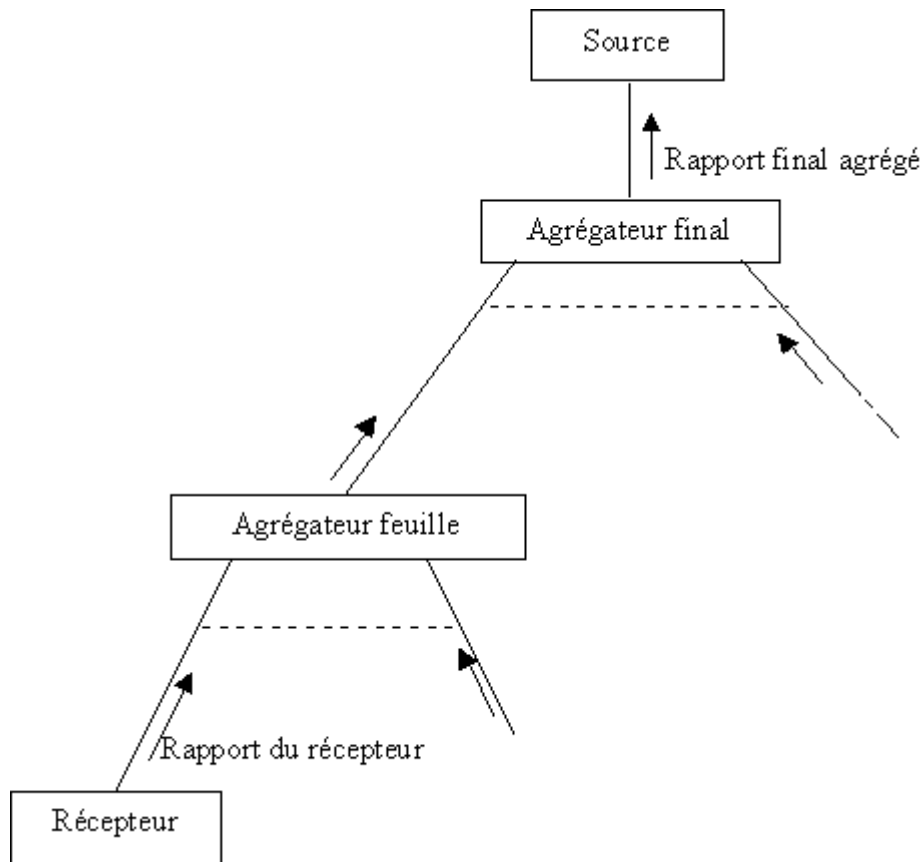


Fig. 1.3 La distribution des agrégateurs dans le réseau

1.3- L'Hétérogénéité des récepteurs:

Les solutions proposées pour le problème d'hétérogénéité des récepteurs sont nombreuses.

Les deux approches, MTCP[3] et RMTP[4] utilisent une fenêtre de congestion, comme TCP.

Avec ceux-ci, la fenêtre n'augmente que si la source reçoit les acquittements de tous les récepteurs.

En plus, ces deux approches, utilisent les mêmes protocoles que le TCP pour l'augmentation ou la diminution de la fenêtre.

Avec ces approches, la source adapte son débit selon l'état du récepteur le plus médiocre du groupe des récepteurs, et par suite, ne présentent pas une solution efficace au problème d'hétérogénéité. Aussi, ils utilisent une architecture complexe pour l'agrégation des acquittements.

Une autre solution est la transmission de données en multicouches[5,6,7,10,11].

Avec cette solution, la source émet les données en plusieurs couches, une couche de base, et des couches supplémentaires, et chaque récepteur, après l'évaluation de sa bande passante disponible, choisit les couches à joindre.

Dans la couche de base, la source émet les données nécessaires pour une réception avec une qualité minimale, et dans les couches supplémentaires, elle envoie le reste des données pour augmenter la qualité de réception.

Chaque récepteur s'abonne à la couche de base, et si sa bande passante le permet, il s'abonne aux couches supplémentaires, pour améliorer la qualité du flux qu'il reçoit.

Cette opération d'abonnement est détaillée dans le chapitre suivant.

La source affecte à chaque couche un niveau de protection, et la couche de base est la plus

importante. Cette protection dépend de l'algorithme utilisé, soit en affectant à chaque couche, un

niveau de priorité (et dans ce cas, le nombre maximal de couches générées doit être inférieur ou égal au nombre maximal de niveaux de priorités que le réseau puisse supporter), soit en envoyant les données avec redondance.

Ces couches peuvent être statiques, c.a.d. sont fixes en nombre et débit, ou bien dynamiques, qui sont variables, en fonction de l'état du réseau.

Dans le premier cas, la source fixe, au début de la transmission, le nombre de couches et le débit de chaque couche.

Dans le cas des couches dynamiques, chaque récepteur, envoie son rapport, en indiquant son taux de perte, et sa bande passante disponible, et selon ces rapports, la source adapte le nombre de couches et le débit de chacune.

Avec cette solution, même le débit de la couche de base peut être variable. Dans le cas où la bande passante la plus faible dans le groupe de récepteurs, est plus grande que le débit de cette couche de base, la source peut régler le débit de cette couche à la valeur de la bande passante minimale.

Dans le cas des couches dynamiques, et comme dans le calcul du RTT, les rapports des récepteurs présentent un problème de scalabilité.

Mais l'avantage de cette solution, est qu'elle reflète toujours l'état du réseau, ce qui n'est pas le cas avec les couches statiques.

La source, avec des messages de contrôle, transmet des informations concernant les couches, nombre, débit et adresse de chacune.

Les opérations "join and leave" qui permettent aux récepteurs de joindre ou quitter une telle couche, sont synchronisées avec des messages envoyés par la source (ces opérations sont détaillées dans le chapitre suivant).

Le routeur, parmi les couches qu'il reçoit, ne laisse passer que celles réclamées par les récepteurs ce qui présente une économie de la bande passante utilisée.

Vicisano L. Rizzo et J. Crowcroft [5], présentent une approche avec une transmission de données en un nombre de couches statiques, et chaque récepteur, après l'estimation de son taux de perte, décide de son abonnement aux couches.

Les inconvénients de ce mécanisme, est qu'il utilise des couches statiques, et que le calcul de la bande passante du récepteur se fait en fonction du taux de perte uniquement, sans prendre en compte son RTT entre lui et la source.

PLM(packet pair receiver-driven layered multicast)[6] est basé sur une transmission de données en multicouches, dont le nombre est fixe, ainsi que le débit de chacune.

Avec cette approche, la source émet deux paquets consécutifs, en indiquant dans leurs entêtes que ces sont des paquets de contrôle, et le récepteur, en fonction du temps écoulé entre la réception de ces deux paquets, détermine sa bande passante disponible et s'abonne aux couches adéquates.

Cette méthode permet d'éviter le problème des rapports des récepteurs, mais elle nécessite que les routeurs utilisent une discipline équitable pour servir les paquets dans les files d'attente, car sinon, le temps écoulé entre la réception de ces deux paquets, ne sera plus proportionnel à la bande passante disponible.

T. Jiang, E. Zegura and M. Ammar[7] présentent un mécanisme avec lequel, la source émet une couche de base dont le débit est fixe, et des couches supplémentaires avec des débits variables. Ces débits sont calculés en fonction des rapports de récepteurs, qui évaluent leurs bandes passantes disponibles en fonction du taux de perte et ne prennent pas en compte leurs RTT.

Aussi, cet approche, ne propose pas de solution pour éviter la saturation du réseau avec les rapports de récepteurs.

Cheung, Ammar et Li [8], présentent une solution, qui consiste à décomposer le groupe de récepteurs en sous-groupes. Vers chacun de ces sous-groupes, la source émet les données avec un débit qui est calculé en fonction des rapports des récepteurs du sous-groupe correspondant.

Une autre solution est présentée dans [9], avec laquelle la source émet une seule couche avec un débit maximal.

Chaque nœud intermédiaire, quand elle reçoit cette couche de données, calcule la bande passante disponible entre lui et les nœud prochain et/ou les récepteurs.

Puis, vers chacun des nœuds prochains et/ou les récepteurs, le nœud intermédiaire transfère la couche de données avec un débit adapté à la valeur correspondante de la bande passante disponible.

Notons que cette solution a l'inconvénient de demander une architecture complexe des routeurs.

[10] présente deux approches qui adressent le problème d'hétérogénéité des récepteurs, et qui permettent à la source d'adapter dynamiquement son débit en fonction de l'état du réseau. Ces deux approches sont cités ci-dessous :

a- Network-Based SAMM [10]: elle nécessite la contribution de nœuds intermédiaires pour calculer la bande passante disponible sans perte qui peut être allouée à ce flux.

Pour cela, il est nécessaire d'implémenter dans les routeurs un algorithme, qui, en surveillant les flux qui le traversent, calcule la bande passante totale allouée aux flux vidéo, ainsi que celle de chacun d'eux.

La source émet plusieurs couches de données, et pour chaque n paquets de données (n autour 32), elle émet un paquet de contrôle, dans lequel elle indiquant le nombre maximal de couches qu'elle peut générer.

Dans ce paquet de contrôle, la source fixe le champ R_E à la valeur du débit maximum voulu, qui est la somme des débits de toutes les couches.

Le routeur, après la réception de ce paquet, calcule la bande passante disponible sans perte de paquets qui le traversent, en fonction du taux de perte pendant l'intervalle du temps écoulé entre la réception de ce nouveau paquet de contrôle et le paquet précédent.

Après, il affecte à R_E le minimum entre la valeur de la bande passante disponible calculée, et la valeur courante de R_E .

Le récepteur, quand il reçoit ce paquet, répond avec un acquittement, en indiquant la valeur de R_E , qui représente sa bande passante disponible sans perte.

Cet approche, résout le problème de scalabilité des acquittements, en implémentant des agrégateurs dans le réseau.

La source, selon l'acquittement final agrégé, adapte le nombre de couches et le débit de chaque couche.

b- End-to-End SAMM [10]: très simple, il consiste à une évaluation des bandes passantes disponibles aux récepteurs, en surveillant la réception des paquets, ce qui évite de gaspiller de la bande passante dans le réseau avec les paquets de contrôle de la source.

Le récepteur, en évaluant le taux de perte des paquets de données, peut évaluer si le débit avec lequel il reçoit est inférieur ou supérieur que sa bande passante disponible.

La source adapte le nombre de couches et le débit de groupe, en fonction des rapports de récepteurs. Aussi, cette approche utilise des agrégateurs, pour fusionner les acquittements de récepteurs.

Avec ces deux approches, le récepteur calcule sa bande passante disponible sans perte, ce qui présente deux inconvénients :

Or la majorité du trafic Internet est un trafic TCP, par suite le calcul de la bande passante disponible sans perte du récepteur n'alloue pas à ce récepteur la même bande passante allouée avec un modèle TCP [18] dans les mêmes conditions du réseau, avec le même taux de perte et le même RTT, ce qui rend le partage de la bande disponible dans le réseau entre les différentes applications inéquitable.

Le deuxième inconvénient est que les applications vidéos ont besoin d'un protocole de contrôle qui évite les variations brusques de la bande passante disponible dans le réseau.

Le calcul de la bande passante disponible sans perte du récepteur se fait à la réception en comptant le nombre des paquets reçus pendant une période T .

Ainsi, à la présence d'une congestion dans le réseau, la bande passante disponible sans perte du récepteur se diminue rapidement, et dans le cas où la congestion disparut, cette bande passante augmente rapidement, par suite, on a une variation brusque dans la bande passante disponible de l'application dans le réseau.

Les modèle TCP sont utilisés pour éviter cette variation brusque du débit, ainsi que pour avoir un partage équitable de la bande passante disponible dans le réseau entre les différentes applications.

Notons que la première approche demande une architecture complexe des routeurs.

MLDA[11]: est une approche qui adresse le problème d'hétérogénéité de récepteurs, en proposant un algorithme pour calculer le RTT entre les récepteurs et la source.

La transmission des données se fait en un nombre dynamique des couches. A l'aide des rapports de réception des récepteurs, la source adapte son débit.

La source émet périodiquement des message de contrôle (SR), en indiquant le nombre de couches, le débit et l'adresse de groupe.

Chaque récepteur, après la réception du message SR, commence à surveiller la réception de paquets de données sur chaque couche sur une certaine période (T_0 pour la couche de base, T_1 pour la première couche supplémentaire, etc.).

Après le calcul de son RTT, le récepteur calcule sa bande passante disponible, attend un délai aléatoire, et diffuse son rapport (RR) à la source dans la couche la plus élevée à laquelle ce récepteur s'est abonné.

Les récepteurs qui se trouvent dans une même région locale reçoivent le message SR presque en même temps.

Pour éviter la congestion dans les files d'attente de la source avec les rapports de réception des récepteurs, qui peut conduire à une perte de rapports, chaque récepteur attend un délai aléatoire avant l'émission de son rapport. Pendant ce délai aléatoire, le récepteur écoute le port correspondant à sa couche la plus élevée, et s'il reçoit un RR d'un autre récepteur qui demande cette même couche, il détruit son rapport

cet algorithme s'appelle suppression partielle, et il présente une solution efficace pour résoudre le problème de scalabilité des rapports de récepteurs.

La source, avant d'émettre chaque SR, calcule le T_{dlr} de chaque rapport de récepteur reçu dans la période écoulée entre l'instant du dernier SR, et l'instant d'émission de ce SR.

Sous forme de réponse aux RR de récepteurs, la source introduit dans son message SR des nouveaux champs contenant le T_{dlr} d'un RR reçu, et l'identité du récepteur correspondant.

Pour les autres récepteurs qui ont détruit leurs rapports, la solution est la suivante :

Dans un réseau idéal (avec des horloges synchronisées et sans asymétrie), le RTT est égal à deux fois la différence (en valeur absolue), entre l'instant de réception du message de l'une des deux extrémités, et l'instant d'émission de ce même message de l'autre extrémité, donc:

$$RTT/2 = T_{rec} - T_{em} \quad (1)$$

Pour compenser la désynchronisation d'horloges, on ajoute le paramètre δ , et le paramètre σ pour compenser l'asymétrie du réseau, et l'équation (1) devient :

$$RTT/2 = T_{rec} - T_{em} + \sigma - \delta \quad (2).$$

Supposons $\theta = \sigma - \delta$,

$$\text{donc : } \theta = T_{rec} - T_{em} - RTT/2 \quad (3).$$

La source inclut dans chaque message SR l'instant d'émission de ce message.

Le récepteur, après la réception du message SR de la source, pour pouvoir calculer son RTT entre lui et la source, il a besoin de calculer la valeur de θ .

Chaque récepteurs qui obtient une réponse dans le message SR de la source, calcule son RTT selon l'équation (a). Ensuite, il calcule la valeur de θ avec l'équation (3) et annonce aux autres récepteur cette valeur. Pour cela, il diffuse un message, avec un ttl (temps de vie) faible, en indiquant la valeur de θ . Avec un faible ttl, ce message n'atteint que les récepteurs locaux. Appelons ce récepteur, R_s .

A présent, chaque récepteur qui a détruit son RR, échange des paquets de contrôle avec un récepteur R_s local pour calculer le RTT entre lui et la source.

Après la réception du message de R_s , dans laquelle le R_s indique sa valeur de θ , le récepteur envoie à son R_s un message (appelé demande de θ dont lequel il indique son identité, et il mémorise son instant d'émission.

Le R_s chaque fois il reçoit une demande de θ d'un récepteur, il répond avec un message (appelé réponse du R_s) dont lequel il indique l'identité du récepteur et le T_{dlsr} de sa demande de θ .

A la réception de la réponse du R_s , le récepteur calcule le RTT entre lui et son R_s , puis, il calcule son θ_{local} avec ce dernier, et par suite, la valeur de θ , sera la somme du θ_{local} et du θ du R_s .

L'intérêt de cette approche, est qu'elle adresse à savoir les problèmes d'hétérogénéité des récepteurs et de scalabilité des rapports de réception. Elle présente aussi un algorithme pour calculer les RTT des récepteurs.

1.4- Conclusion :

Pour récapituler les principaux problèmes de la transmission multipoint vidéo sur l'Internet, et des solutions proposées, il faut noter que l'hétérogénéité des récepteurs, et la scalabilité des rapports de réception représentent un grand challenge, d'où la nécessité d'un algorithme qui adresse ces deux problèmes, et ne gaspille pas de la bande passante dans le réseau avec les paquets de contrôle.

Dans le chapitre suivant, on présente le protocole de transport RTP, le travail de mon stage, ainsi qu'un algorithme proposé dans [12, 13] qui présente une solution à tous les problèmes cités ci-dessus.

CHAPITRE II

UNE SOLUTION A LA TRANSMISSION MULTIPOINT

Introduction :

L'étude du chapitre précédent nous amène à conclure de la nécessité d'un algorithme qui adresse les principaux problèmes de la transmission multipoint, à savoir, l'hétérogénéité des récepteurs ainsi que la scalabilité d'émission des rapports de ces derniers, et en même temps, d'éviter de gaspiller de la bande passante dans le réseau avec les paquets de contrôle.

Dans ce chapitre on présente une solution de contrôle de transmission multipoint proposée dans [12] et [13], ainsi que le travail de mon stage.

Notre approche consiste à utiliser une transmission des données avec un nombre de couches dynamiques pour résoudre le problème d'hétérogénéité des récepteurs. Cette approche utilise des agrégateurs pour résoudre le problème de scalabilité dû à l'émission des rapports des récepteurs, ainsi qu'une méthode qui permet de calculer le RTT entre les récepteurs et la source.

Cette approche utilise le protocole RTP pour transférer les paquets de données et de contrôle (RTCP).

Dans la section 2.1 on décrit le protocole RTP. La section 2.2 adresse plusieurs méthodes pour l'organisation des agrégateurs dans le réseau. Dans la section 2.3, on présente l'algorithme utilisé pour la classification des récepteurs.

La section 2.4 présente le travail déjà fait au niveau de l'implémentation de cette approche, et le travail de mon stage est présenté dans la section 2.5.

Dans la section 2.6, on présente les résultats attendus après la fin de l'implémentation du programme, et une conclusion est présentée dans la section 2.7.

2.1- Real Time Transport Protocol (RTP):

RTP est un protocole de transport en temps réel qui est conçu pour satisfaire les besoins des applications multimédia sur internet.

RTP est utilisé juste au dessus du protocole **UDP**, et on distingue, le protocole de transfert des données **RTP** du protocole de contrôle associé **RTCP**.

2.1.1- RTP Data Transfer Protocol :

Un paquet de donnée est décomposé d'une entête et d'un champ de donnée. L'entête elle même est décomposée en deux partie, une partie fixe (entête fixe) de douze octets, qui est commune entre toutes les applications qui utilisent RTP, et une entête supplémentaire qui est fonction de l'application considérée.

Dans l'entête du paquet de donnée, la source transmet toutes les informations nécessaires pour le traitement de ce paquet, à savoir, le type, la taille des données, le numéro de séquence de ce paquet, l'identité de la source (SSRC) ou la liste des sources (CSRC liste) dans le cas où plusieurs sources ont contribué à ce paquet de donnée, etc.

2.1.2- RTP Control Protocol (RTCP):

RTCP est basé sur la transmission périodique des paquets de contrôle à tous les participants à la session, et sa fonction principale est d'obtenir des rapports de réception des flux vidéos. Il y a cinq types de messages RTCP :

1- SR (Sender Report):

C'est le rapport (ou le message de contrôle) de la source qui contient des informations concernant les données envoyées par cette source, et des statistiques sur la réception des flux envoyés par les autres sources dans le cas d'une session à plusieurs sources.

2- RR (Receiver Report):

C'est le rapport de réception des récepteurs, dans lequel le récepteur indique son état de réception (sa bande passante disponible et son taux de perte). Dans le cas d'une session à plusieurs sources, ce rapport peut contenir des statistiques sur la réception d'au plus 31 sources.

3- SDES (Source description RTCP packet):

Le protocole **RTP** identifie chaque élément de la session indépendamment des couches inférieures, pour cela, il affecte à ceux-ci plusieurs identificateurs qui sont le *SSRC*, *CNAME*, *NAME*, *EMAIL*, *PHONE*, *LOC*, *TOOL*, *NOTE* et *PRIV*.

Ces identificateurs sont choisis aléatoirement, et les deux premiers (*SSRC* et *CNAME*) doivent être unique sur toute la session, d'où la nécessité d'utiliser des algorithmes pour la détection et la correction des collisions.

Chaque élément de la session (source ou récepteur) transmet ces identificateurs dans un paquet *SDES*, chaque fois qu'il transmet un rapport, donc chaque *SR* ou *RR* est suivi d'un paquet *SDES*.

Notons que ces identificateurs sont affectés chaque fois que l'on lance l'application.

Dans le cas où la source émet plusieurs types de données, par exemple, un flux audio et un flux vidéo, un même paquet ne peut pas transporter ces différents types. Dans ce cas, RTP affecte à la session des identificateurs SSRC pour chaque application, à condition que le SSRC de chaque élément (source ou récepteur) soit nouveau pour chaque application.

Afin d'un seul CNAME est affecté à chaque élément de la session pour toutes les applications, permettre aux différents éléments de synchroniser les flux reçus. D'où la raison de l'utilisation de plusieurs identificateurs et non pas un seul.

4- BYE:

Chaque élément de la session doit envoyer ce paquet pour indiquer la fin de sa contribution à la session.

5- APP:

dans le cas des applications spécifiques, les données peuvent être transmises dans des paquets spécifiques de type APP. Ce paquet consiste en une entête de 12 octets et un champ de données qui est fonction de l'application considérée.

Le format de ce paquet APP est représenté dans le schéma ci-dessous :

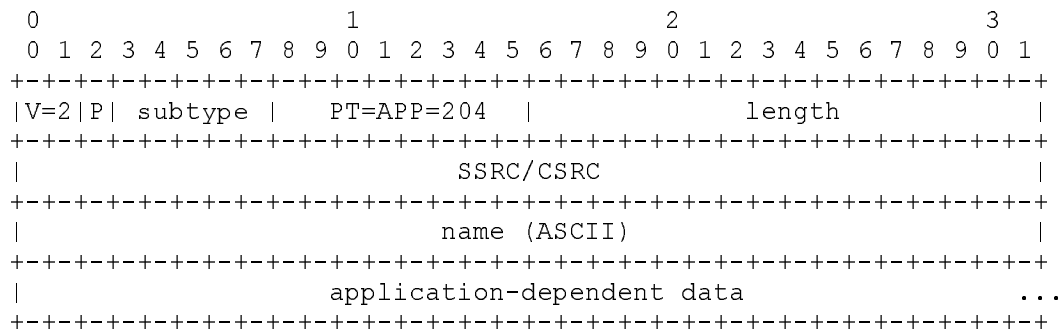


Fig.2.1 Format du APP

Plus de détails concernant le protocole RTP et le rôle de chaque champ de ces paquets sont présentés dans [14]

2.2- Organisation des agrégateurs dans le réseau :

La figure ci-dessous (Fig. 2.2) représente un exemple d'implémentation hiérarchique multi-niveau des agrégateurs dans le réseau. Les agrégateurs sont distribués sous la forme d'un arbre dont la racine est le AA0 (*Agrégateur final*), et AAi représente un agrégateur de niveau i. Les agrégateurs de niveau le plus bas s'appellent agrégateurs feuilles.

Chaque région contient un agrégateur feuille qui fusionne les rapports des récepteurs qui y appartient, et l'envoi à l'agrégateur de niveau supérieur.

Le rapport final agrégé contient le nombre des récepteurs dans chaque classe.

Les agrégateurs peuvent être placés manuellement, ou automatiquement en utilisant des fonctions qui s'occupent de lancer des agrégateurs dans le réseau.

[15] présente une méthode de placement automatique des agrégateurs dans le réseau. Cette méthode consiste à implémenter une fonction en utilisant le *custom concast*. Le *custom concast* est une application qui permet de déterminer une adresse destination pour un groupe de sources.

Chaque groupe de récepteurs applique cette fonction ou cette application pour choisir un agrégateur feuille, et chaque groupe des agrégateurs feuilles applique cette fonction pour choisir un agrégateur de niveau supérieur, et ainsi de suite jusqu'au le choix de l'agrégateur final.

Le datagramme du *custom concast* contient un ensemble des groupes de sources et une destination unique pour chacun de ces groupes.

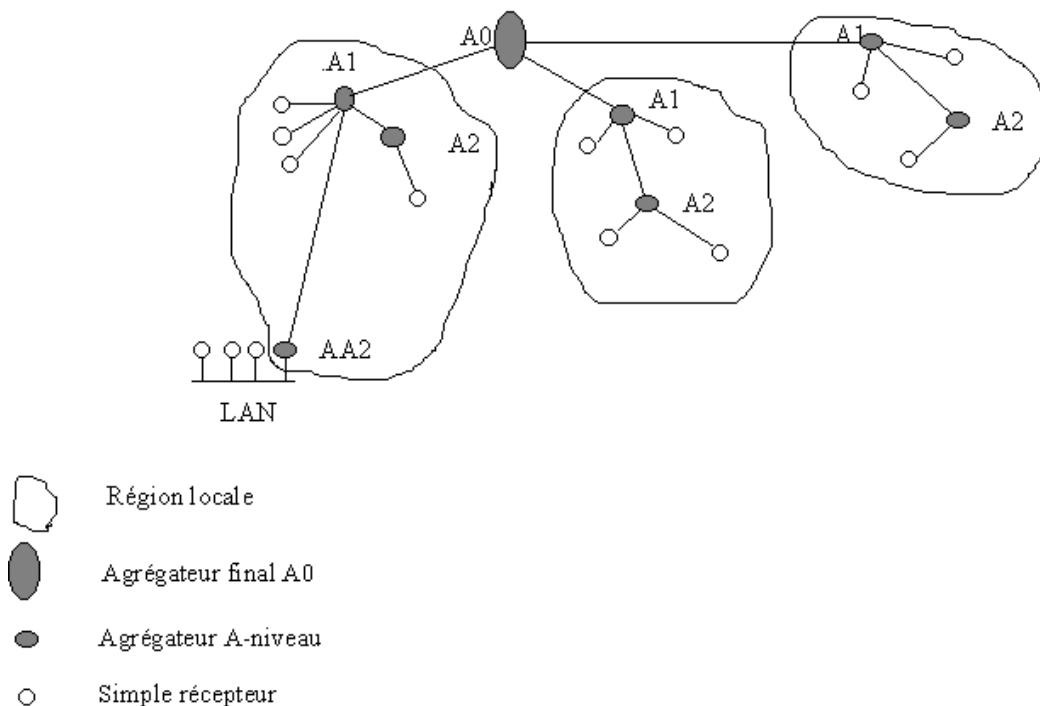


Fig. 2.2 Utilisation hiérarchique multi-niveau des agrégateurs

Chaque récepteur envoie son rapport à l'adresse destination du groupe auquel il appartient, et la source reçoit un seul rapport final agrégé.

La signalisation dans cet algorithme permet à l'application d'avoir des informations concernant la configuration du réseau, pour pouvoir décomposer le groupe des récepteurs en sous-groupes, et déterminer les adresses des destinations correspondantes.

Une autre méthode est présentée dans [16], qui consiste à utiliser dans le réseau plusieurs *nœuds actifs* appelé *cluster* (noté AS1).

Dans cet algorithme, les récepteurs réagissent comme des clients qui lancent des Agents (agrégateurs) dans des positions stratégiques dans le réseau.

Avant que le client (récepteur) lance un Agent dans le réseau, il écoute le réseau pour recevoir des informations qui lui permettent d'effectuer un rendez-vous avec un *nœud actif*. Ces informations peuvent être obtenues en écoutant une adresse multipoint connue, ou en utilisant le DHCP décrit dans [17].

Une fois le client a affecté un rendez-vous avec un AS1, il lance son Agent.

AS1 implémente un protocole de contrôle (ASCP) qui utilise un HM (Home Manager). Ce dernier reçoit les demandes des agents des récepteurs, et choisit un Agent principal et des agents secondaires pour remplacer l'agent principal dans le cas où ce dernier tombe en panne.

Dans le cas où un récepteur tombe en panne, le récepteur est annulé de la liste des récepteurs, et la même chose pour son Agent.

Notons que dans les deux cas, les agrégateurs participent seulement aux flux RTCP et non pas aux flux des données ou de contrôle.

2.3- Classification des récepteurs :

2.3.1- Préliminaires :

La classification des récepteurs sert à regrouper les récepteurs homogènes dans une même classe, pour laquelle la source prend les mêmes décisions d'adaptation. Cette classification ne doit pas se faire seulement en fonction des rapports des récepteurs actuels, mais elle doit prendre en compte les rapports récents du passé. Dans ce cas, une procédure doit être ajoutée pour oublier les rapports très anciens.

Avant de décrire les méthodes utilisées pour classier les récepteurs, il faut citer les paramètres qui sont pris souvent en compte lors de la classification.

Parmi ces paramètres, le taux de perte et la bande passante disponible des récepteurs.

Le taux de perte permet à la source de déterminer le niveau de protection nécessaire pour chaque couche, et il peut être calculé en surveillant la réception du flux de donnée sur un intervalle de temps.

La bande passante disponible est utilisée par la source pour adapter les débits des différentes couches. Dans le cas où l'on calcule les bandes passantes disponibles des récepteurs avec une équation TCP, qui utilise un modèle de TCP [18], un algorithme scalable doit être utilisé pour calculer les RTT entre les récepteurs et la source.

l'équation TCP utilisée dans notre approche est la suivante :

$$BW = S/[R*(2*p/3)^2 + t_{RTO}*(3*(3P/8)^2)*P*(1 + 32P^2)] \quad (1)$$

Où S est la taille moyenne des paquets de données en bytes estimée par le récepteur, R est la valeur du RTT en secondes entre le récepteur et la source calculée en secondes.

p est le taux de perte moyen du récepteur compris entre 0 et 1.

t_{RTO} (en secondes) = $4*R$.

Dans une session avec une seule source et un seul récepteur, le récepteur envoie un acquittement pour chaque paquet de données ou pour un groupe de paquets selon l'algorithme utilisé. Si l'acquittement correspondant au paquet (ou au groupe de paquets)

de données n'arrive pas à la source avant t_{RTO} à partir de l'instant de son émission, il est considéré perdu.

Plusieurs approches sont proposées pour modéliser le processus de perte: Le modèle de Gilbert est un modèle Markovien simple à deux états, un état de perte représenté par B (Bad), et un état dans lequel le paquet arrive à la destination, et est représenté par G (Good). p désigne la probabilité de passer de l'état G à l'état B, et q la probabilité de passer de l'état B à l'état G. Les temps moyens de séjour dans l'état G et dans l'état B sont respectivement $1/p$ et $1/q$. La moyenne de perte dans ce modèle est : $p_l = p/(p + q)$.

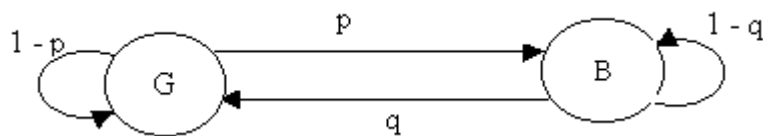


Fig. 3.3 Le modèle de Gilbert

Dans le cas où la probabilité de perte dans l'état B n'est pas 1, et la probabilité de réception du paquet n'est pas 0, la moyenne de perte devient : $p_l = p \cdot p_B / (p + q) + q \cdot p_G / (p + q)$.

La classification des récepteurs peut se faire en fonction d'un seul paramètre ou un groupe de paramètres dépendants ou indépendants. Elle se fait en regroupant les rapports similaires dans une même classe; Pour cela, il faut une méthode pour mesurer la distance entre les différents rapports et choisir la classe la plus proche.

Cette distance peut être mesurée avec une équation euclidienne L^p donnée par la formule suivante : $d(x, y) = (\sum(x_i - y_i)^p)^{1/p}$

où x_i et y_i sont les différents paramètres des rapport x et y respectivement.

2.3.2- Algorithme de classification des agrégateurs:

[12] présente un algorithme simple pour la classification des récepteurs. Chaque classe est représentée par un point représentatif et un poids, et chaque fois qu'un nouveau point rejoint cette classe, il change la position du point représentatif ainsi que son poids. L'agrégateur reçoit les rapports des récepteurs, et quand un nouveau rapport arrive, il joint la classe la plus proche, qui correspond à la distance euclidienne minimale, et dans le cas où cette distance est plus grande qu'un certain seuil, une nouvelle classe est créée, à condition que le nombre total des classes soit inférieur à 5, qui est le nombre maximal des couches.

Notons que chaque classe correspond à une couche.

Chaque agrégateur envoie régulièrement son rapport agrégé vers l'agrégateur de niveau supérieur. L'agrégateur feuille considère le point représentatif obtenu avec la période passée comme un point représentatif pour la nouvelle période, et les agrégateurs de niveaux supérieurs réinitialisent leurs points représentatifs après l'émission de leurs rapports.

Avec un facteur γ le poids de chaque classe de l'agrégateur feuille est diminué chaque fois que ce dernier envoie son rapport, et quand le poids descend sous un certain seuil, la classe est éliminée.

cette méthode prend en compte les rapports des récepteurs récents du passé, et élimine les rapports très anciens.

Pseudo-Code de classification des récepteurs:

d_{th} = seuil défini pour créer des nouvelles classes (si la distance entre un rapport de réception reçu et toutes les classes existantes est supérieur à ce seuil, et si le nombre des classes est inférieur à 5, une nouvelle classe est créée, autrement, le rapport joint la classe la plus proche).

N_{max} = nombre maximal des classes (5)

r = rapport du récepteur reçu

Chercher la classe la plus proche $d(r, \hat{C}) = \min_C d(r, C)$

Si ($d(r, \hat{C}) \geq d_{th}$)

 Si (nombre des classes $< N_{max}$)

 Ajouter une nouvelle classe C_{NEW}

 &

 Ajuster le poids (le nombre des récepteurs) de la classe $\hat{C} = C_{NEW}$

Calculer le point représentatif de la classe \hat{C}

$x_{\hat{C}} = (\text{poids}(\hat{C}) * x_{\hat{C}} + r) / (\text{poids}(\hat{C}) + 1)$

Augmenter le poids de la classe \hat{C} de 1

Mécanisme d'agrégation des rapports des récepteurs:

W_{min} = seuil de suppression de la classe

γ = poids de la mémoire

Au début de la période

 pour toutes les classes

 diminution du poids de la classe par le facteur

$\text{poids}(C) = \text{poids}(C) * \gamma$

 Si ($\text{poids}(C) < W_{min}$)

 enlever la classe

Recevoir un nouveau rapport

Envoyer le rapport agrégé vers l'agrégateur de niveau supérieur

2.4- Travail existant :

Deux équipes de L'INRIA (le projet PLANETE à Sophia Antipolis et le projet TEMICS à Rennes) ont commencé l'implémentation de la solution présentée dans [12, 13]. Dans cette section, on récapitule le travail effectué par ces deux équipes, et dans la section suivante, le travail de mon stage.

L'approche implémentée est utilisée pour une transmission des données en plusieurs couches, et en fonction de l'état de réception, la source adapte les caractéristiques des couches. Cette approche propose l'utilisation d'agrégateurs pour résoudre le problème de scalabilité d'émission des rapports des récepteurs.

L'équipe de Rennes s'occupe de l'adaptation des caractéristiques des couches dans le cas d'une source ou plus, et l'équipe à Sophia Antipolis s'occupe du problème de scalabilité d'émission des rapports des récepteurs et de contrôle de congestion.

Dans la version du code utilisé, la transmission des données se fait en une seule couche, et chaque T_{control} (appelée période de contrôle) la source diffuse un paquet de contrôle RTCP, composé d'un option SR, suivi d'une option SDES et d'une ou plusieurs options APP.

Dans ces options APP, les informations concernant les caractéristiques des couches sont transmises.

Chaque récepteur envoie à son agrégateur feuille son rapport de réception en indiquant le taux de perte moyen des flux des données à la réception et sa bande passante disponible sans perte.

Ces rapports de réception sont envoyés une fois dans chaque période T_{control} .

Les fonctions correspondantes à la fusion des rapports des récepteurs sont déjà implémentées, et par suite, le problème de scalabilité des rapports des récepteurs est résolu.

2.5- Travail du stage :

L'approche de [12, 13] propose une solution pour mesurer les RTT entre les récepteurs et la source dans laquelle seuls les agrégateurs feuilles échangent des messages avec la source, et chaque récepteur calcule son RTT avec la source comme étant la somme de son RTT avec son agrégateur feuille et le RTT de son agrégateur feuille avec la source.

Dans mon stage, j'ai implémenté l'algorithme de calcul des RTT entre les récepteurs et la source en se basant sur l'idée citée ci-dessus.

Après l'intégration du code correspondant au calcul du RTT entre les récepteurs et la source, chaque récepteur calcule le RTT entre lui et la source et transmet son rapport de réception à son agrégateur feuille en indiquant sa bande passante disponible calculée avec un modèle de TCP avec l'équation (1) au lieu de sa bande passante disponible sans perte, ainsi que le taux de perte moyen des flux des données à la réception.

J'ai aussi adressé le problème de la détection et la correction des collisions entre les identificateurs (SSRC et CNAME) des différents éléments de la session ainsi que les opérations d'abonnement des récepteurs aux couches supplémentaires, et j'ai présenté des solutions qui peuvent être implémentées et intégrées facilement avec les codes présents.

2.5.1- Calcul du RTT :

La méthode suivie consiste à échanger des options APP entre la source et l'agrégateur feuille d'une part, et le récepteur et son agrégateur feuille d'autre part. Les options APP échangées pendant chaque période de contrôle T_{control} pour le calcul des RTT entre les récepteurs et la source sont décrites ci-dessous :

Chaque agrégateur feuille envoie périodiquement à chaque T_{control} (avec un délai aléatoire) une option APP (appelée demande du RTT de l'agrégateur feuille) à la source en indiquant dans le champ de donnée le numéro de séquence de ce paquet, et il mémorise son instant d'émission.

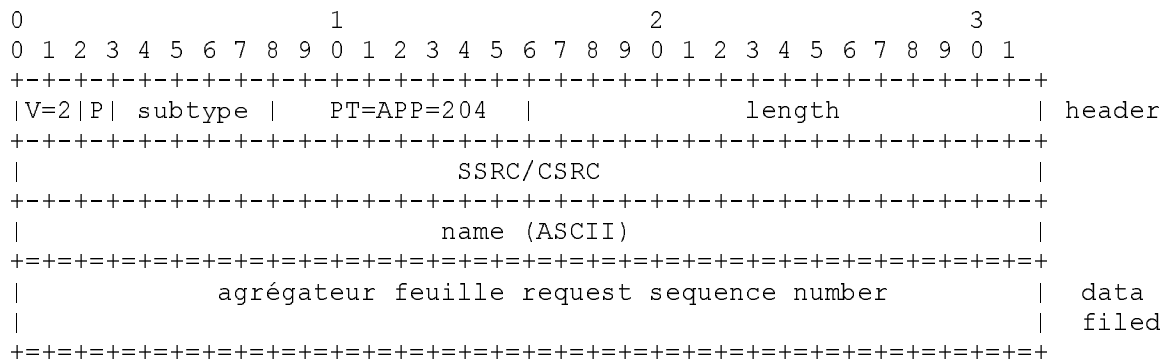


Fig.2.4 Format de la demande du RTT des agrégateurs feuilles

Chaque fois que la source reçoit une demande du RTT d'un agrégateur feuille, elle la mémorise avec son instant de réception.

Avec le message périodique diffusé par la source à chaque T_{control} , une option APP est ajoutée (appelée réponse de la source), dont le champ de donnée est composé de plusieurs sous-réponses aux demandes de RTT reçues.

Comme montre la figure 2.5 ci-dessous, chaque sous-réponse de ce champ de données est composée de plusieurs champs correspondants au T_{dlsr} d'une demande reçue (qui est le temps écoulé entre la réception de la demande et l'émission de la réponse correspondante), le numéro de séquence de cette demande, ainsi que le SSRC de l'agrégateur feuille correspondant.

Après la réception de la réponse de la source, chaque agrégateur feuille calcule le RTT entre lui et la source avec l'équation suivante:

$$RTT_{\text{Agrégateur}} = T_{\text{réception de la réponse de la source}} - T_{\text{émission de la demande de l'agrégateur}} - T_{\text{dlsr}} \quad (1)$$

A la réception de la réponse de la source, il cherche sa réponse dans le champ de donnée du paquet reçu, en comparant son SSRC au SSRC indiqué dans chaque sous-réponse. Tout de suite, l'agrégateur feuille calcule le RTT entre lui et la source selon l'équation (1), puis il répond les récepteurs à leurs demandes en diffusant un paquet RTCP (option APP) vers ses récepteurs (appelé réponse de l'agrégateur feuille). Le champ de donnée de cette option APP est composé en plusieurs sous-réponse chacun correspondant à une demande d'un récepteur reçue, et dans laquelle est indiqué le T_{dlsr} d'une demande d'un récepteur reçue (qui est le temps écoulé entre la réception de la demande et l'émission de la réponse correspondante), le numéro de séquence de cette demande, et le SSRC du récepteur correspondant. L'agrégateur feuille indique aussi dans ce champ de donnée la valeur de RTT entre lui et la source.

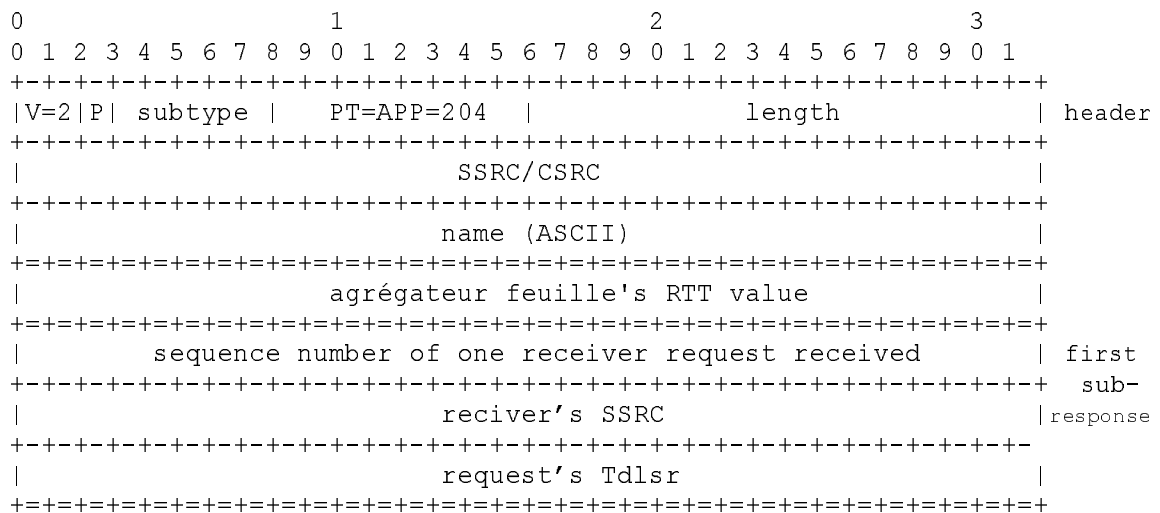


Fig. 2.7 Format de la réponse de l'agrégateur feuille aux demandes du RTT des récepteurs

Lorsque le récepteur reçoit la réponse de son agrégateur feuille, il va chercher sa réponse dans le champ de donnée, en comparant son SSRC au SSRC indiqué dans chaque sous-réponse, puis il calcule son RTT avec la source selon l'équation suivante:

$$RTT_{Récepteur} = T_{réception\ de\ la\ réponse\ de\ l'agrégateur} - T_{émission\ de\ la\ demande\ du\ récepteur} - T_{dlsr} - RTT_{Agrégateur}$$

Avec cette méthode, tous les récepteurs peuvent calculer avec précision leurs RTT avec la source.

Grâce au fait que seuls les agrégateurs finaux participent au calcul du RTT, cet algorithme ne présente pas un problème de scalabilité ni un gaspillage de la bande passante dans le réseau.

2.5.2- Détection et résolution des collisions des SSRC et CNAME :

Comme c'est indiqué dans la section 2.1, RTP affecte à chaque élément de la session (source ou récepteur) parmi eux les deux identificateurs (SSRC et CNAME). Ces

deux identifiants doivent être uniques dans la session, d'où la nécessité d'un algorithme qui détecte la collision correspondant au cas où deux éléments différents de la session génèrent un même SSRC ou un même CNAME.

Chaque option SR ou RR est suivie d'une option SDES qui indique les deux identifiants SSRC et CNAME. Dans la version du code utilisée, les récepteurs envoient des rapports de réception à leurs agrégateurs feuilles, et ces derniers renvoient un rapport agrégé aux agrégateurs de niveau supérieur et ainsi de suite.

Chaque rapport de réception consiste en un paquet RTCP formé de trois options, une option SR, suivie d'une option SDES et une option APP. Le rapport de la source consiste aussi en un paquet RTCP formé de trois options, une option SR, suivie d'une option SDES et une option APP.

Contrairement à la spécification RTCP, ces messages de contrôle périodiques ne sont pas échangés entre les différents éléments de la session, et par suite une détection de collision entre les identifiants n'est pas possible.

Ci-dessous, on présente une solution simple à la détection et correction des collisions. Cette solution peut être intégrée facilement avec les algorithmes déjà implémentés.

La source mémorise pour chaque élément de la session (y compris la source elle-même) son adresse IP avec ses deux identifiants (SSRC et CNAME), et chaque fois qu'elle reçoit un paquet SDES d'un élément, elle compare si un même identifiant est utilisé par un autre élément avec une adresse IP différente.

Chaque récepteur et chaque agrégateur envoie un paquet RTCP à la source, formé d'une option SDES.

Pour éviter la congestion du réseau avec ces paquets dans le cas où plusieurs récepteurs s'abonnent ensemble à la session, chaque récepteur attend un délai aléatoire avant l'émission de ses identifiants à la source.

Ce délai aléatoire est choisi en fonction du nombre des récepteurs dans la session.

Dans le cas d'une collision, la source inclut dans son rapport une liste des éléments qui ont une collision, et chaque récepteur ou agrégateur concerné, chaque fois qu'il reçoit le rapport de la source, régénère son identifiant (ou les deux dans le cas où il y a une collision avec les deux identifiants choisis).

2.5.3- Synchronisation des abonnements aux couches (join and leave):

Le récepteur, pour calculer sa bande passante disponible entre lui et la source, doit calculer le taux de perte moyen sur toutes les couches.

Par expérimentations, ils ont trouvé [11] que si deux récepteurs essaient de rejoindre en même temps deux couches différentes, le récepteur qui essaie de rejoindre la couche la plus basse, a de grandes chances de constater des pertes de paquets.

Ces pertes ne sont pas dues à l'augmentation du débit qu'il reçoit, mais apparaissent à la suite de l'autre récepteur qui essaie de rejoindre la couche la plus élevée.

Ainsi, des algorithmes pour synchroniser l'abonnement aux différentes couches sont nécessaires. L'approche présentée dans [11] adresse ce problème de synchronisation.

Dans cette approche, la source transmet les données en plusieurs couches, et elle diffuse dans la couche de base des messages de contrôle périodiques autour des caractéristiques des différentes couches. Chaque récepteur, après la réception du message de contrôle de

la source, commence à surveiller la réception des paquets sur la couche de base sur une certaine période T_0 .

Puis, le récepteur essaye de joindre la première couche supplémentaire (s'il n'était pas déjà abonné), et il surveille de nouveau la réception des paquets sur cette couche supplémentaire sur une certaine période T_1 , et ainsi de suite jusqu'à la surveillance des toutes les couches.

Après la fin des périodes de surveillance, le récepteur calcule sa bande passante disponible entre lui et la source, puis il quitte les couches qui ne lui conviennent pas.

Tous les récepteurs s'abonnent à la couche de base, et si la bande passante disponible du récepteur le permet, ce dernier s'abonne à des couches supplémentaires comme suit :

Soit C_0 le débit de la couche de base, C_i celui de la couche supplémentaire d'ordre i , et soit R la bande passante disponible du récepteur : le récepteur choisit K couches supplémentaires à joindre à condition que $C_0 + \dots + C_k < R < C_0 + \dots + C_k + C_{k+1}$.

Pour réduire l'effet de désynchronisation dans la réception des messages de contrôle de la source par les récepteurs, une période de synchronisation T_{sync} est ajoutée à la période T_0 .

$$T_{\text{sync}} = (RTT_{\text{max}} - RTT_i) / 2$$

où RTT_{max} est le RTT maximal sur tous les récepteurs dans la session, et RTT_i est le RTT du récepteur i .

Pour cela, les récepteurs incluent leurs RTT dans leurs rapports vers la source, et cette dernière inclut dans son message de contrôle la valeur du RTT_{max} .

Cet algorithme est simple et facile à intégrer avec les codes déjà implémentés.

2.6- Résultats attendus :

Dans cette section, on présente les résultats attendus du programme après la fin de l'implémentation.

La source diffuse périodiquement des paquets RTP des données, et périodiquement elle envoie des messages de contrôle SR, pour décrire la transmission des données.

Chaque récepteur surveille la réception des paquets de données, calcule son taux de perte, ainsi que son RTT avec la source selon la méthode décrite dans la section précédente, et il envoie périodiquement un rapport RR à son agrégateur feuille.

Le rapport RR contient le taux de perte estimé par le récepteur, ainsi que la bande passante disponible entre lui et la source.

Chaque fois que l'agrégateur reçoit un RR, il le fusionne avec les autres RR reçus selon l'algorithme décrit dans la section 2.3 et il envoie un rapport agrégé à l'agrégateur de niveau supérieur, et ainsi de suite jusqu'à l'arrivée à l'agrégateur final.

Après la réception du RR final agrégé de l'agrégateur final, la source adapte les caractéristiques des couches comme c'est indiqué dans la section 2.4.

Pour éviter la congestion dans les routeurs et les files d'attente avec les messages de contrôle périodiques (les rapports de réception et les demandes du RTT des récepteurs et des agrégateurs), un délai aléatoire est ajouté à ces messages périodiques

RTP spécifie une allocation 5% de la bande passante totale de la session aux paquets RTCP. Ainsi, les périodes des messages cités ci-dessus sont fonction du nombre des participants à la session.

Le choix du nombre des couches des données est très important, et il dépend de plusieurs paramètres. D'une part, un grand nombre des couches est nécessaire pour satisfaire les besoins des récepteurs hétérogènes. D'autre part, une petite période de contrôle (par suite un petit nombre de couches) permet à la source de répondre rapidement aux fluctuations du réseau à l'aide des rapports de réceptions des récepteurs.

Pendant chaque période de contrôle, chaque récepteur surveille toutes les couches pour calculer son taux de perte (T_0 pour la couche de base, T_1 pour la première couche supplémentaire, etc.), et par suite la période totale de surveillance est proportionnelle au nombre de ces couches.

La période de contrôle doit être suffisante pour que les récepteurs puissent surveiller toutes les couches, ainsi un grand nombre des couches demande des récepteurs une large période de surveillance et par suite une large période de contrôle.

Ainsi, le choix du nombre maximal des couches dépend de l'application considérée. Avec notre approche, le nombre maximal des couches est cinq.

2.7- Conclusion :

Dans ce chapitre on a présenté le protocole RTP, plusieurs algorithmes pour organiser des agrégateurs dans le réseau, ainsi que notre approche présentée dans [12, 13].

L'intérêt de cette notre approche est qu'elle adresse les principaux problèmes de la transmission multipoint (la scalabilité des rapports de réception et l'hétérogénéité des récepteurs), et présente un algorithme pour calculer avec précision les RTT entre les récepteurs et la source sans avoir un problème de scalabilité.

Dans le chapitre suivant, par comparaison de notre approche avec les approches présentées dans [1, 10, 11], on démontre que notre approche adresse tous les problèmes de la transmission multipoint et présente une économie de la bande passante du réseau.

CHAPITRE III

COMPARAISON DE L'APPROCHE [12, 13] AVEC LES APPROCHES EXISTANTES

Introduction :

Notre approche présentée dans [12, 13] présente beaucoup d'intérêts, il adresse tous les principaux problèmes de la transmission multipoint, il utilise des agrégateurs pour résoudre le problème de scalabilité d'émission des rapports de réception.

Notre approche propose la transmission des données en plusieurs couches dynamiques pour résoudre le problème d'hétérogénéité des récepteurs du à leurs différents états de réception, il présente une solution scalable pour calculer le RTT entre les récepteurs et la source, ainsi qu'il présente une économie de la bande passante du réseau utilisée.

Dans ce chapitre on va comparer notre approche aux approches présentées dans [1], [10] et [11].

Approche [1] présente une solution pour le calcul de RTT entre la source et les récepteurs, par comparaison avec notre approche, on démontre que notre approche présente une solution plus efficace et plus précise pour le calcul de RTT entre la source et les récepteurs.

[2] présente deux approches qui adressent le problème d'hétérogénéité des récepteurs. Dans la comparaison de ces deux approches avec notre approche, on démontre que notre approche présente l'avantage qu'elle utilise un modèle TCP pour calculer les bandes passantes disponibles des récepteurs (voir section 2.3.1).

MLDA adresse tous les principaux problèmes de la transmission multipoint, elle propose des solutions pour résoudre le problème de l'hétérogénéité des récepteurs, le problème de scalabilité d'émission des rapports de réception, ainsi qu'elle présente une solution pour le calcul de RTT entre les récepteurs et la source. Par comparaison de MLDA avec notre approche, on démontre que notre approche présente une économie de la bande passante du réseau utilisée.

3.1-Notre Approche versus Approche [1] :

[1] présente une solution pour le calcul du RTT entre les récepteurs et la source. Dans cette approche, et comme c'est décrit dans le premier chapitre, les récepteurs sont répartis dans le réseau selon une distribution hiérarchique multi-niveau (voir figure 1.2). La source diffuse périodiquement des messages de contrôle, et le RTT de chaque récepteur est calculé comme étant le temps écoulé entre l'instant d'émission du message de contrôle de la source, et l'instant de réception de l'acquittement correspondant du récepteur par la source.

Avec cet algorithme, chaque récepteur fusionne les rapports de réception reçus des autres récepteurs, et envoie périodiquement un rapport agrégé au récepteur de niveau supérieur. A la réception du message de contrôle de la source, le récepteur attend un certain délai avant l'émission de son rapport agrégé. Par suite, un délai est ajouté au RTT entre ce récepteur et la source, qui correspond au temps écoulé entre la réception du message de la source par ce récepteur, et la génération de l'acquittement correspondant. Ainsi, lorsqu'un récepteur reçoit un rapport d'un récepteur de niveau inférieur, il le fusionne avec les autres, et attend l'instant d'émission de son rapport agrégé, ce qui ajoute un délai au temps de propagation de l'acquittement du récepteur vers la source. La méthode utilisée ne prend pas en compte ces délais ajoutés, ce qui rend le calcul du RTT entre les récepteurs et la source imprécis.

Dans notre approche, chaque agrégateur feuille calcule son RTT avec la source, en envoyant une option APP représentant sa demande de calcul de RTT, et il mémorise l'instant d'émission de cette demande. La source répond avec une option APP dans laquelle elle indique le Tdlsr de chaque demande d'un agrégateur feuille reçue (qui est le temps écoulé entre la réception par la source de la demande de calcul du RTT de l'agrégateur feuille et l'instant de génération de la réponse de la source correspondante). A la réception de la réponse de la source, l'agrégateur feuille calcule le RTT entre lui et la source comme étant la différence entre l'instant d'émission de cette demande et l'instant de réception de la réponse correspondante de la source, et en prenant en compte le Tdlsr de cette demande.

Avec la même méthode, le récepteur calcule son RTT avec son agrégateur feuille. L'agrégateur feuille inclut dans sa réponse la valeur du RTT entre lui et la source. Le RTT du récepteur sera égal à la somme du RTT de ce récepteur avec son agrégateur feuille et le RTT entre l'agrégateur feuille correspondant et la source. Avec cet algorithme, le délai écoulé entre l'émission de chaque demande du RTT et la réception de la réponse correspondante est pris en compte, ce qui permet de calculer les RTT entre les récepteurs et la source avec précision.

3.2- Notre Approche versus Approches [10] :

Dans [10], deux approches sont proposées : Network-Based SAMM Algorithm et End-to-End SAMM Algorithm. La première approche nécessite la contribution de nœuds intermédiaires pour calculer les bandes passantes disponibles des récepteurs en fonction du taux de perte du flux de données.

La deuxième approche ne demande pas la contribution des nœuds intermédiaires, mais aussi elle calcule les bandes passantes disponibles des récepteurs à la réception par les récepteurs en fonction du taux de perte du flux de données.

Ces deux approches utilisent des agrégateurs pour résoudre le problème de scalabilité des rapports de réception.

L'avantage majeur de notre approche par rapport aux approches [10] est que notre approche calcule la bande passante disponible du récepteur avec une équation TCP (voir section 2.3.1).

3.3- Notre Approche versus MLDA :

MLDA présente beaucoup d'intérêts : Elle adresse le problème d'hétérogénéité des récepteurs, elle propose un algorithme scalable pour calculer les RTT entre les récepteurs et la source, aussi, elle propose l'algorithme *suppression partielle* pour résoudre le problème de scalabilité des rapports de ces derniers.

Les deux approches calculent les bandes passantes disponibles des récepteurs avec un modèle de TCP et sans la contribution de nœuds intermédiaires.

Aussi, ces deux approches proposent une transmission des données en plusieurs couches pour résoudre le problème d'hétérogénéité des récepteurs.

Afin de rendre la comparaison entre notre approche et MLDA plus compréhensible, on va étudier une session avec une seule source.

MLDA propose l'algorithme de suppression partielle des rapports de réception des récepteurs pour résoudre le problème de la scalabilité des rapports des récepteurs.

Notons que cet algorithme ne demande pas l'implémentation des agrégateurs dans le réseau, ce qui peut être considéré comme un avantage de MLDA par rapport à notre approche.

Avec cet algorithme de suppression partielle des rapports de réception des récepteurs, chaque récepteur surveille la réception des paquets sur chaque couche pendant une certaine période (T_0 pour la couche de base, T_1 pour la première couche supplémentaire, etc.).

A la fin des périodes de surveillance des couches, le récepteur calcule sa bande passante disponible, et choisit les couches de données à joindre.

Ensuite, le récepteur écoute sur une période T_{wait} le port de la couche la plus élevée à laquelle il s'est abonné, et s'il reçoit un rapport de réception d'un autre récepteur, il détruit son rapport.

Avec MLDA, le récepteur envoie son rapport sur sa couche la plus élevée, pour cela chaque récepteur écoute le port de sa couche la plus élevée.

Avec cet algorithme (*suppression partielle*), la suppression des rapports de réception se fait en fonction des bandes passantes disponibles indiquées dans ces rapports de réception. Cependant, la source ne peut pas évaluer le nombre des récepteurs qui sont abonnés à chaque couche, car les rapports de réception détruits ne sont pas comptés.

Dans l'option APP, la source inclut des informations concernant les couches (nombre, débits et adresse de groupe), la valeur de la période de contrôle $T_{\text{contrôle}}$, la période de surveillance de chaque couche (T_0, T_1 , etc.), ainsi que le RTT_{max} sur tous les récepteurs. Cette valeur est utilisée par les récepteurs pour synchroniser l'abonnement des récepteurs aux différentes couches.

Chaque récepteur, quand il reçoit le rapport de la source, commence à surveiller la réception des paquets sur la couche de base sur une certaine période T_0 .

Puis, le récepteur essaye de joindre la première couche supplémentaire (s'il n'était pas déjà abonné), et il surveille de nouveau la réception des paquets sur cette couche supplémentaire sur une certaine période T_1 , et ainsi de suite jusqu'à la surveillance des toutes les couches.

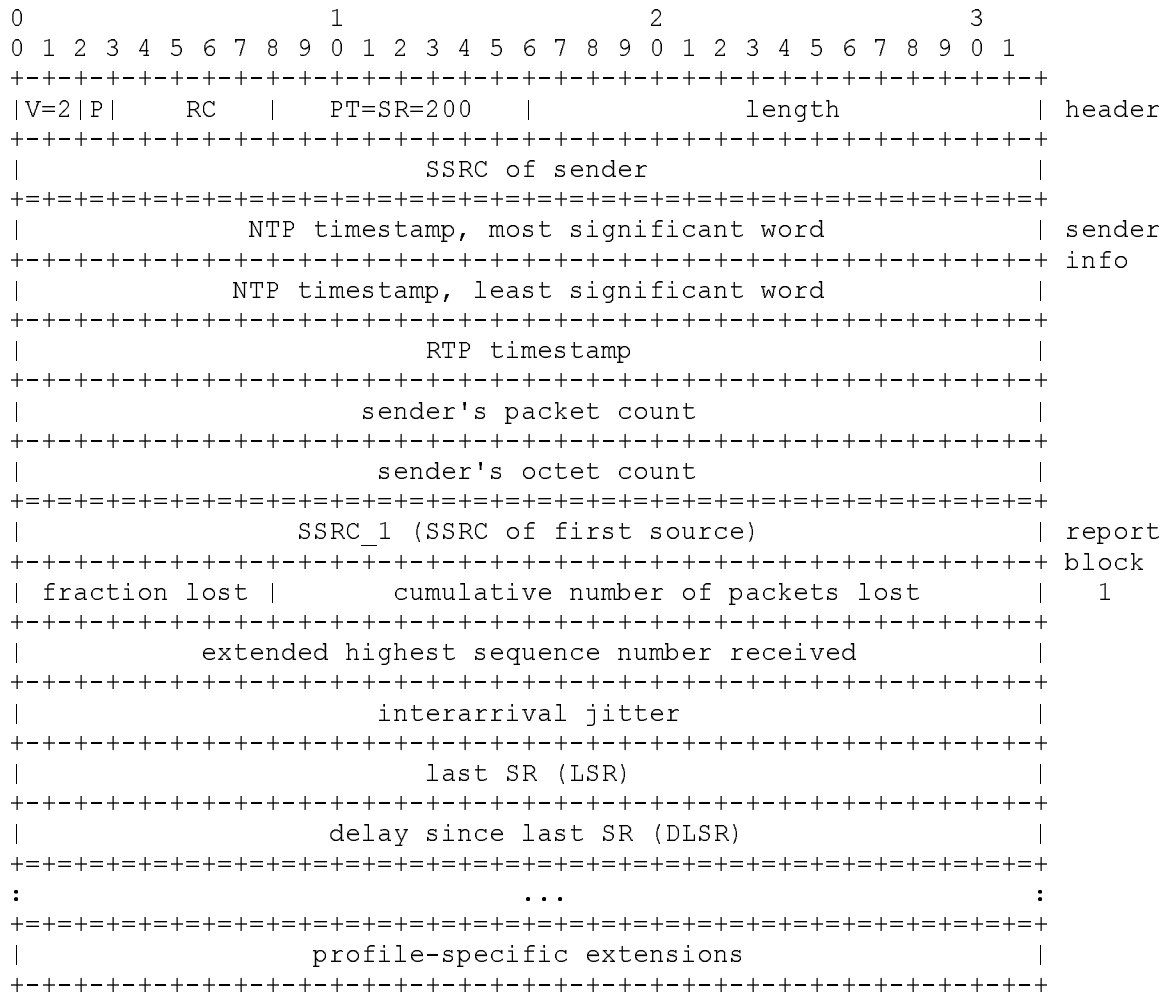


Fig.3.2 Format du SR

Après la fin des périodes de surveillance, le récepteur calcule sa bande passante disponible entre lui et la source, puis il quitte les couches qui ne lui conviennent pas.

Pour réduire l'effet de désynchronisation dans la réception des messages de contrôle de la source par les récepteurs, une période de synchronisation T_{sync} est ajouté à la période T_0 .

$$T_{\text{sync}} = (\text{RTT}_{\text{max}} - \text{RTT}_i)/2$$

Ainsi, le rapport de la source est représenté sur $(288 + 224 + 672 + 96*n* T_{\text{contrôle}} / T) = (1568 + 96*n* T_{\text{contrôle}} / T)$ bits.

Le récepteur à son tour, envoie son rapport à la source, ce rapport consiste aussi en trois options, une option *RR* suivie d'une option *SDES* et d'une option *APP*.

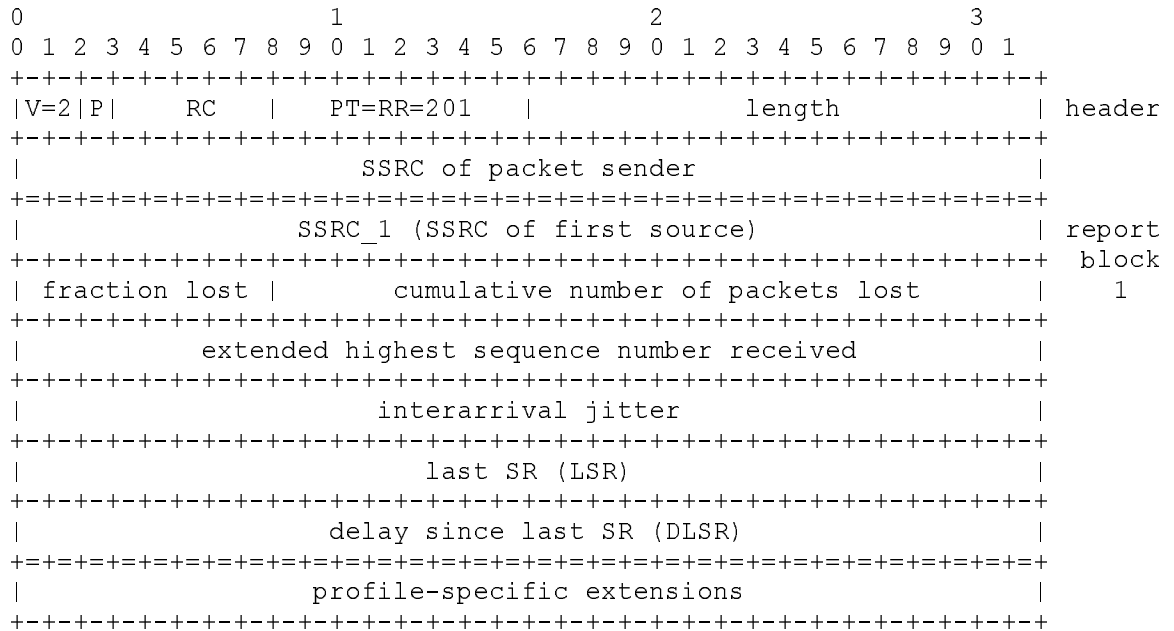


Fig.3.4 Format du RR

Comme pour le rapport de la source, les données sont transmises dans l'option APP, par suite le dernier champ de l'option RR (profile-specific extensions) est vide. Comme montre la figure 3.4, cette option RR est représentée sur 256 bits.

L'option SDES envoyée est la même qu'avec le rapport de la source, et elle est représentée sur 288 bits (voir figure 3.2).

Les données transmises dans le champ de données de l'option APP correspondent au numéro de séquence du rapport de réception envoyé, la bande passante disponible du récepteur, son taux de perte moyen, ainsi que la valeur de son RTT.

Cette option APP est représentée sur 224 bits comme montre la figure 3.5.

Ainsi, le rapport de réception des récepteurs est représenté sur $(256 + 288 + 224) = 768$ bits. Par suite, avec MLDA, la bande passante requise par les paquets de contrôle à chaque période de contrôle $T_{\text{contrôle}}$ est : $1568 + 96*n* T_{\text{contrôle}} / T + 768 = (2336 + 96*n* T_{\text{contrôle}} / T)$ bits.

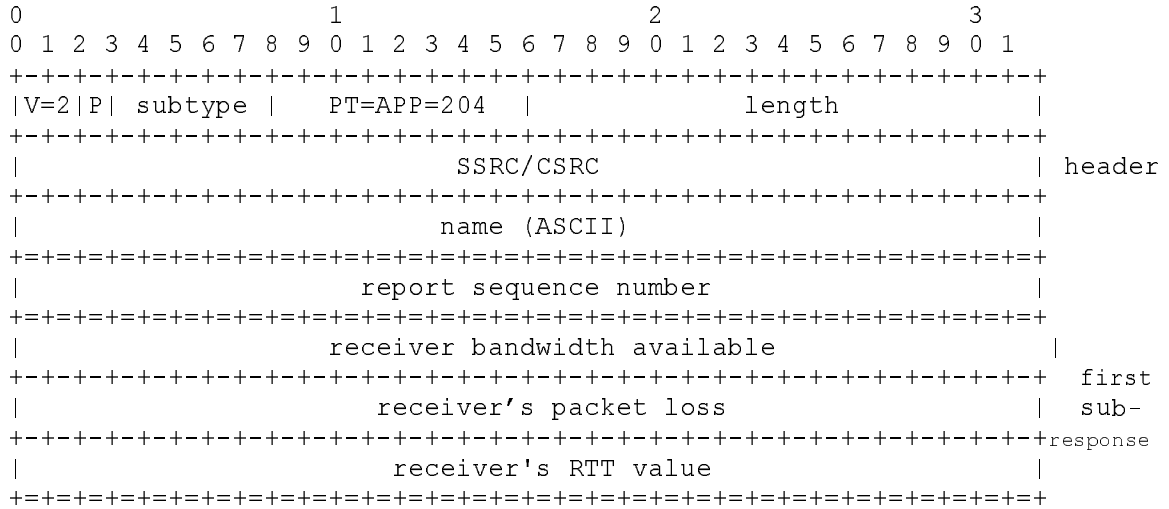


Fig. 3.5 Format de l'option APP du rapport du récepteur

Passons maintenant à notre approche. La source diffuse périodiquement à chaque $T_{\text{contrôle}}$, un rapport formé de trois options, une option *SR*, suivie d'une option *SDES* et d'une option *APP*.

Les données sont transmises dans l'option APP dans le champ de données, par suite le dernier champ de cette option (profile-specific extensions) est vide. Cette option SR est représentée sur 224 bits (voir figure 3.1).

L'option SDES envoyée est la même qu'avec le rapport de la source dans MLDA, et elle est représentée sur 288 bits (voir figure 3.2).

La source inclut dans les données transmises dans l'option APP, des informations concernant les couches (nombre, débits et adresse de groupe), la valeur de la période de contrôle, la période de surveillance de chaque couche (T_0, T_1 , etc.), ainsi que le RTT_{max} sur tous les récepteurs.

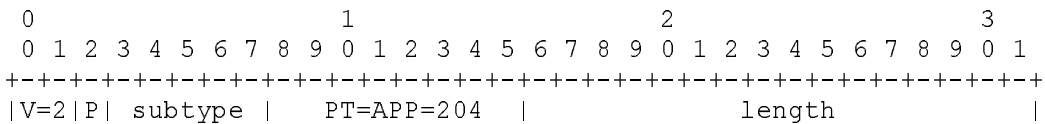
De même, la valeur RTT_{max} est utilisée par les récepteurs pour synchroniser leurs abonnements aux différentes couches.

Comme montre la figure 3.6, cette option est représentée sur 672 bits. Ainsi, le rapport de la source est représenté sur $(224 + 288 + 672) = 1184$ bits.

Notre approche utilise des agrégateurs pour résoudre le problème d'hétérogénéité. Avec cette approche, des paquets de contrôle sont échangés entre la source et les agrégateurs feuilles d'une part, et entre les agrégateurs feuilles et les récepteurs d'autre part.

Chaque groupe de récepteurs locaux se connecte à un même agrégateur feuille. par suite, les paquets de contrôle échangés entre ce groupe local et l'agrégateur feuille correspondant sont locaux.

Ainsi, la bande passante utilisée par les paquets de contrôle dans le réseau est supposée entre les agrégateurs feuilles et la source.



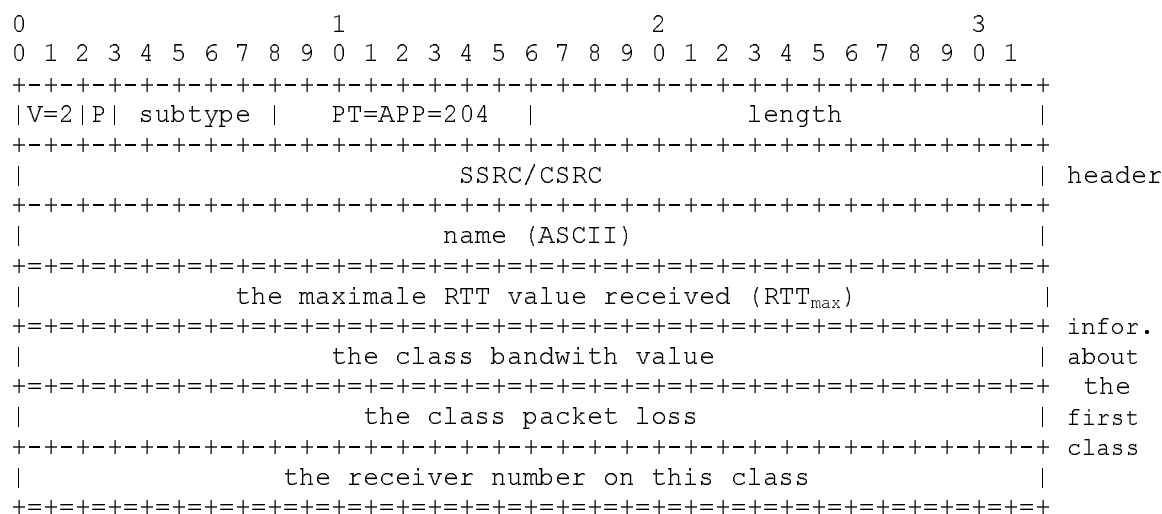


Fig. 3.7 Format de l'option APP du rapport du récepteur

Pour calculer le RTT entre lui et la source, chaque agrégateur feuille envoie à la source une demande du RTT.

Cette demande consiste en un seul paquet RTCP (option APP), dont le champ de données contient le numéro de séquence de cette demande (voir figure 2.4). Cette option est représentée sur 128 bits.

La source à son tour, répond aux demandes des agrégateur feuilles en diffusant un paquet RTCP formée d'une option APP (voir figure 2.5).

Dans cette option APP, la source inclut des sous-réponses, chacune consiste en trois champs, le numéro de séquence d'une demande reçue, le SSRC de l'agrégateur feuille correspondant, ainsi que le Tdlsr de cette demande, qui est la différence en valeur absolue entre l'instant de réception de cette demande et l'instant de génération de la réponse correspondante par la source.

Cette option consiste en une entête de 96 bits et un champ de données de $N_{\text{agr-feuille}}$ sous-réponse chacune représentée sur 96 bits, par suite $96 \cdot (1 + N_{\text{agr-feuille}})$ bits en total.

Ainsi, la bande passante utilisée par les paquets de contrôle pendant chaque période de contrôle est : 1184 bits pour le rapport de la source + 1152 bits pour le rapport de réception agrégé + 128 bits pour la demande du RTT de l'agrégateur feuille + $96 \cdot (1 + N_{\text{agr-feuille}})$ bits pour la réponse de la source donc $(2560 + 96 \cdot N_{\text{agr-feuille}})$ bits en total.

Pour fixer les idées, prenons un exemples d'une session avec une seule source, 300 récepteurs ($n = 300$) et 15 agrégateurs finaux, avec une période T égale à $10 \cdot T_{\text{contrôle}}$ ($T = 10 \cdot T_{\text{contrôle}}$).

Avec MLDA, la bande passante demandée pour cette session dans chaque période de contrôle est : $1568 + 96 \cdot n \cdot T_{\text{contrôle}} / T = 1568 + 96 \cdot 300 \cdot T_{\text{contrôle}} / 10 T_{\text{contrôle}} = 4448$ bits/ $T_{\text{contrôle}}$.

Avec l'approche de [12, 13], la bande passante demandée pour cette session est :

$$2560 + 96 \cdot N_{\text{agr-feuille}} = 2560 + 96 \cdot 15 = 4000 \text{ bits}/T_{\text{contrôle}}$$

Par suite le gain dans la bande passante du réseau est $((4448 - 4000) / 4448) \cdot 100 \approx 10\%$.

Ce gain dans la bande passante du réseau présenté par notre approche par rapport à MLDA permet à notre approche d'avoir une valeur de la période de contrôle plus petite qu'avec MLDA.

En effet, RTP préconise une allocation de 5% de la bande passante de la session pour les paquets de contrôle.

La source après la réception des rapports de réception des récepteurs, ajuste les caractéristiques des couches, et calcule la bande passante totale disponible dans cette session.

Ensuite, la source calcule la bande passante disponible pour les paquets de contrôle, qui correspond à 5% de la bande passante totale disponible dans la session, et la valeur de la période de contrôle correspondante.

Avec le même exemple pris en haut, on va calculer la valeur de la période de contrôle pour notre approche et pour MLDA avec une bande passante totale disponible dans la session de 100 Kbits/sec.

Le 5% de la bande passante totale disponible alloué aux paquets de contrôle est : $5 * 25 * 1024 / 100 = 1280$ bits/sec.

Les paquets de contrôle nécessite une bande passante de 4448 bits/ $T_{\text{contrôle}}$ avec MLDA et 4000 bits/ $T_{\text{contrôle}}$ avec notre approche.

La période de contrôle est calculée comme suit :

pour MLDA : $T_{\text{contrôle}} = 4448 / 1280 = 3.475$ secondes.

Pour notre approche : $T_{\text{contrôle}} = 4000 / 1280 = 3.125$ secondes.

Comme montre le calcul ci-dessus, notre approche permet d'avoir un gain dans la bande passante du réseau qu'avec MLDA (de $(4448 - 4000) * 100 / 4448 \approx 10\%$ avec l'exemple pris), ce qui lui permet d'avoir des périodes de contrôle plus courtes.

Dans cette section, la comparaison entre notre approche et MLDA est faite dans le cas d'une session avec une seule source, mais la même étude peut être généralisée pour les sessions avec plusieurs sources.

3.4- Conclusion :

Dans ce chapitre on a comparé notre approche aux approches présentées dans [1], [10] et [11].

Dans la comparaison avec l'approche [1], on a montré que notre approche présente une méthode plus efficace et plus précise pour calculer les RTT entre les récepteurs et la source.

Dans la comparaison avec les approches [10], on a étudié l'importance de l'utilisation d'un modèle TCP [18] pour le calcul des bandes passantes disponibles des récepteurs.

Avec MLDA, on a montré l'économie dans la bande passante dans le réseau présentée par notre approche, ce qui lui permet d'avoir des périodes de contrôle plus courte, par suite, de répondre rapidement aux fluctuations de la bande passante dans le réseau.

Chapitre 3. Comparaison de l'approche [12, 13] avec les approches existantes

Dans le chapitre suivant, on présente les valeurs de RTT entre les récepteurs et la source trouvés trouvées suite à des expérimentations.

CHAPITRE 4

EXPERIMENTATIONS ET RESULTATS

Introduction :

Le calcul du RTT entre les récepteurs et la source nécessite l'échange de paquets de contrôle ce qui peut présenter un problème de scalabilité et un gaspillage de la bande passante dans le réseau, surtout dans les sessions qui ont un grand nombre des récepteurs.

D'autre part, un algorithme qui permet d'ajuster la période de contrôle de la transmission des données dans la session ainsi que la fréquence de calcul du RTT entre les récepteur et la source en fonction de l'état du réseau présente beaucoup d'intérêt.

Pour résoudre ces problèmes de scalabilité et de gaspillage de la bande passante dans le réseau dus au calcul des RTT entre les récepteurs et la source, l'approche que nous avons suivie, présentée dans [12, 13] fait en sorte que seuls les agrégateurs feuilles échangent des paquets de contrôle avec la source.

Dans cette approche, la source calcule la bande passante totale disponible pour la session, ainsi que celle des paquets de contrôle, et ajuste la valeur de la période de contrôle.

dans ce chapitre on va présenter les résultats des expérimentations que nous avons effectuées dans le réseau local de l'INRIA–Sophia Antipolis et dans le Mbone entre les sites de Sophia et de l'IRISA Rennes.

4.1- Expérimentations et résultats :

La figure 4.1 représente l'architecture de la session simulée. Cette architecture consiste en une seule source, un agrégateur final (AG_{Final}), deux agrégateurs feuilles (AG_{F1} et AG_{F2}), et trois récepteurs (R_1 , R_2 et R_3).

Les deux récepteurs (R_1 , R_2) sont connectés à l'agrégateur feuille (AG_{F1}), et le troisième récepteur (R_3) est connecté au deuxième agrégateur feuille (AG_{F2}).

Les deux agrégateurs feuilles sont connectés à l'agrégateur final (AG_{Final}), qui à son tour est connecté à la source.

L'agrégateur final participe seulement aux rapports de réception envoyés par les deux agrégateurs feuilles, il fusionne ces rapports et envoie un rapport final agrégé à la source.

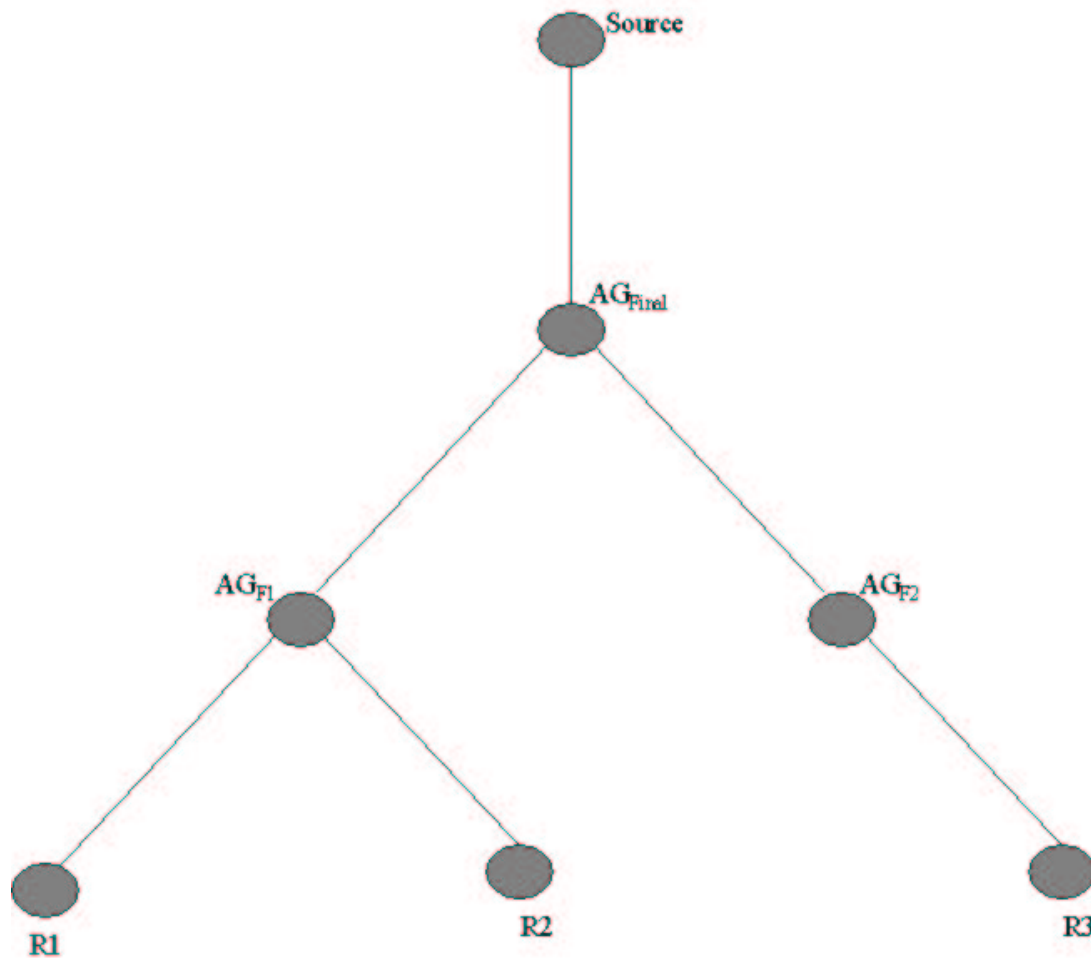


Fig. 4.1 Architecture de la session simulée

Les deux agrégateurs feuilles participent aux rapports de réception envoyés par les récepteurs et au calcul de RTT entre les récepteurs et la source.

Ces agrégateurs feuilles envoient leurs rapports de réception agrégés à l'agrégateur final, et à la source il envoient des demandes de RTT (voir ch.2, section 2.5).

Les récepteurs envoient leurs demandes de RTT et leurs rapports de réception à leurs agrégateurs feuilles.

Notre approche utilise un modèle TCP [18] pour calculer les bandes passantes disponibles (BW) des récepteurs, et l'équation utilisée est la suivante :

$$BW = S / [R * (2 * p / 3)^2 + t_{RTO} * (3 * (3P/8)^2) * P * (1 + 32P^2)] \quad (1)$$

Où S est la taille moyenne des paquets de données en bytes estimée par le récepteur, R est la valeur du RTT en secondes entre le récepteur et la source calculée en secondes.

p est le taux de perte moyen du récepteur compris entre 0 et 1.

t_{RTO} (en secondes) = $4 * R$.

Dans une session avec une seule source et un seul récepteur, le récepteur envoie un acquittement pour chaque paquet de données ou pour un groupe de paquets selon l'algorithme utilisé. Si l'acquittement correspondant au paquet (ou au groupe des paquets)

de données n'arrive pas à la source avant t_{RTO} à partir de l'instant de son émission, il est considéré perdu.

Avec cette équation (1), quand le taux de perte est égale à zéro, la bande passante disponible du récepteur est infinie. Dans la pratique, cela signifie que le récepteur pourra s'abonner à toutes les couches de données.

N'étant pas encore disponible, les expérimentations ont été faites avec une transmission de données en une seule couche correspondante à la couche de base.

Les trois expérimentations ont été faites avec la même architecture de la session ou la même distribution des éléments dans le réseau (figure 4.1).

a- La première expérimentation consiste en un test local avec des machines locales à l'INRIA-Sophia Antipolis. Toutes les machines utilisées sont des machines Linux.

Pour vérifier les valeurs de RTT trouvées avec notre algorithme, on a lancé en même temps que le test, une application ping entre la source et une autre machine Linux de l'INRIA-Sophia Antipolis.

L'application ping permet de calculer le RTT entre deux machines en donnant le nom de ces deux machines.

Dans les figures représentant les valeurs de RTT trouvées suites aux expérimentations, le RTT-ping représente les valeurs de RTT trouvées avec l'application ping.

Notons que l'application ping mesure le RTT en point à point, par suite les chemins suivis par les paquets de l'application ping et les paquets des applications multipoint sont différents, mais ça donne un ordre d'idée.

La liaison entre Sophia Antipolis et Rennes n'était pas congestionnée pendant les expérimentations, nous aurons du simuler des pertes de paquets aux récepteurs.

Ainsi, le taux de perte de R1 a été forcé à 2%, celui de R2 à 5% et celui de R3 n'a pas été modifié (il est resté nul), ce qui correspond à une bande passante infinie pour R3.

Les valeurs des RTT entre les récepteurs et la source trouvées avec notre algorithme sont représentées dans la figure 4.2 (voir page 45), ainsi que les valeurs de RTT trouvées avec l'application ping.

Comme le montre la figure 4.2, il y a une différence entre les valeurs de RTT trouvées avec notre algorithme et celles trouvées avec l'application ping.

Cette différence est due principalement à la dérivation entre les horloges des différentes machines. Cette dérivation est plus faible entre deux machines Linux ou Solarise qu'entre une machine Linux et une machine Solarise.

Pour montrer cet effet de dérivation entre les horloges des différentes machines sur les résultats trouvés, on a lancé un deuxième test local avec des machines toutes Linux.

b- La deuxième expérimentation consiste aussi en un test local avec la même session de la figure 4.1, mais cette fois ci, la source est lancée sur une machine Solarise, les agrégateurs (feuilles et final) et les récepteurs sont lancés sur des machines Linux.

En même temps que le test, on a lancé l'application ping entre la source et une autre machine Linux de l'INRIA-Sophia Antipolis.

La figure 4.4 (voir page 47) représente Les valeurs des RTT entre les récepteurs et la source trouvées avec notre algorithme, ainsi que celle trouvées avec l'application ping. Comme le montre les deux figures 4.2 et 4.4, les valeurs de RTT trouvées sont inférieures dans le deuxième test que dans le premier, ce qui montre l'effet de dérivation des horloges des machines.

N'ayant pas de machines dédiées à ces expérimentations, nous avons fait les tests avec des machines qui effectuaient d'autres tâches en parallèle. Ceci explique les valeurs pics du RTT des récepteurs obtenues dans les deux expérimentations locales.

Les bandes passantes des récepteurs trouvées avec ces deux expérimentations locales sont représentées dans les deux figures 4.3 et 4.5 (voir page 46 et 48).

Dans les courbes représentant les bandes passantes disponibles des récepteurs (4.3, 4.5 et 4.7), seules les bandes passantes disponibles des récepteurs R1 et R2 sont représentées, car R3 a un taux de perte nul, par suite une bande passante disponible infinie calculée avec l'équation 1.

D'après l'équation (1), la bande passante disponible du récepteur est inversement proportionnelle au RTT et au carré du taux de perte du récepteur.

R1 et R2 sont connectés à un même agrégateur feuille, par suite, ils ont des valeurs proches de RTT, ainsi que R1 a un taux de perte plus petit que celui de R2.

Par suite, R1 a des valeurs de RTT proches que ceux de R2 et un taux de perte plus petit que celui de R2, donc théoriquement, la bande passante de R1 doit être plus grande que celle de R2.

Les deux figures 4.3 et 4.5, montre en chaque point que la bande passante disponible du récepteur R1 est plus grande que celle du récepteur R2.

Comme le montre ces deux figures, l'utilisation d'un modèle TCP [18] pour calculer les bandes passantes disponibles des récepteurs permet d'adapter le débit de l'application à l'état du réseau, ainsi qu'il évite les variations brusques dans la bande passante allouée à cette application.

c- La troisième expérimentation est faite avec une source à l'INRIA-Rennes, et des récepteurs et agrégateurs (feuille et final) à l'INRIA-Sophia Antipolis.

Dans ce test, tous les machines utilisées sont des machines Linux sauf la source, qui est lancée sur une machine Solarise.

En même temps que le test, on a lancé l'application ping entre la source et une autre machine de l'INRIA-Sophia Antipolis.

Les valeurs de RTT entre les récepteurs et la source trouvées avec notre algorithme sont représentées dans la figure 4.6 (voir page 49), ainsi que celles trouvées avec l'application ping.

Dans cette expérimentation, la différence entre les valeurs de RTT trouvées avec notre algorithme et celles trouvées avec l'application ping est due principalement à La

dérivation entre les horloges des machines utilisées, ainsi que les paquets multicast et les paquets de l'application ping suivent des chemins différents.

Les valeurs pics du RTT obtenues représentent des moments de congestion dans le Mbone entre les sites de Sophia et l'IRISA Rennes.

les valeurs des bandes passantes disponibles des récepteurs sont représentées dans la figure 4.7 (voir page 50).

Les applications vidéo ont besoin d'un protocole de contrôle de congestion qui évite les variations brusques de débit.

Notre approche présente l'intérêt qu'elle calcule la bande passante TCP-Friendly [18] des récepteurs ce qui assure une variation lente de la bande passante allouée à l'application (comme le montre la figure 4.7), ainsi qu'elle assure un partage équitable de la bande passante du réseau entre les applications vidéos et le trafic Internet.

4.2- Interprétation des résultats :

Dans la section précédente, on a présenté les expérimentations que nous avons effectuées dans le réseau local de l'INRIA-Sophia Antipolis et dans le Mbone entre les sites de Sophia et de l'IRISA Rennes.

En même temps que ces expérimentations, on a lancé l'application ping qui calcule le RTT en point à point entre deux machines.

La différence entre les valeurs de RTT trouvées avec notre algorithme et celle trouvée avec l'application ping est due principalement à la dérivations entre les horloges des machines utilisées, ainsi que les paquets multicast et les paquets de l'application ping suivent des chemins différents.

Une deuxième raison qui peut expliquer cette différence entre les valeurs de RTT trouvées est que quand on lance l'application ping entre deux machines A et B, la valeur du RTT entre ces deux machines est mesurée comme étant la différence en valeur absolue entre l'instant d'émission du paquet de données par la machine A et la réception par cette même machine A de l'acquittement correspondante de l'autre machine B.

Avec l'application ping, le paquet de données est traité dans la couche réseau. Tandis que avec l'algorithme de notre approche, les paquets reçus et émis sont traités dans la couche de transport, ce qui ajoute un délai aux valeurs calculées de RTT entre les récepteurs et la source.

Les courbes représentant les bandes passantes disponibles des récepteurs trouvées suite à ces expérimentations montre l'intérêt de notre approche qui calcule la bande passante TCP-Friendly des récepteurs [18].

Cette approche adapte le débit de l'application aux capacités disponibles dans le réseau en assurant une variation lente du débit de l'application et un partage équitable de la bande passante du réseau entre les applications vidéos et le trafics Internet.

4.3- Conclusion :

Dans ce chapitre, on a présenté les expérimentations que nous avons effectuées et analysé les résultats obtenus.

A une erreur près due principalement à la dérivation entre les horloges des machines utilisées, notre algorithme permet le calcul des RTT entre les récepteurs et la source sans présenter un problème de scalabilité dû aux paquets de contrôle échangés, et en même temps il présente une économie de la bande passante dans le réseau.

A ce stade nous arrivons au bout de notre stage. Nous allons dans le chapitre suivant conclure, en rappelant les problèmes de la transmission vidéo multipoint sur Internet, quelques solutions proposées, ainsi que le travail de notre stage.

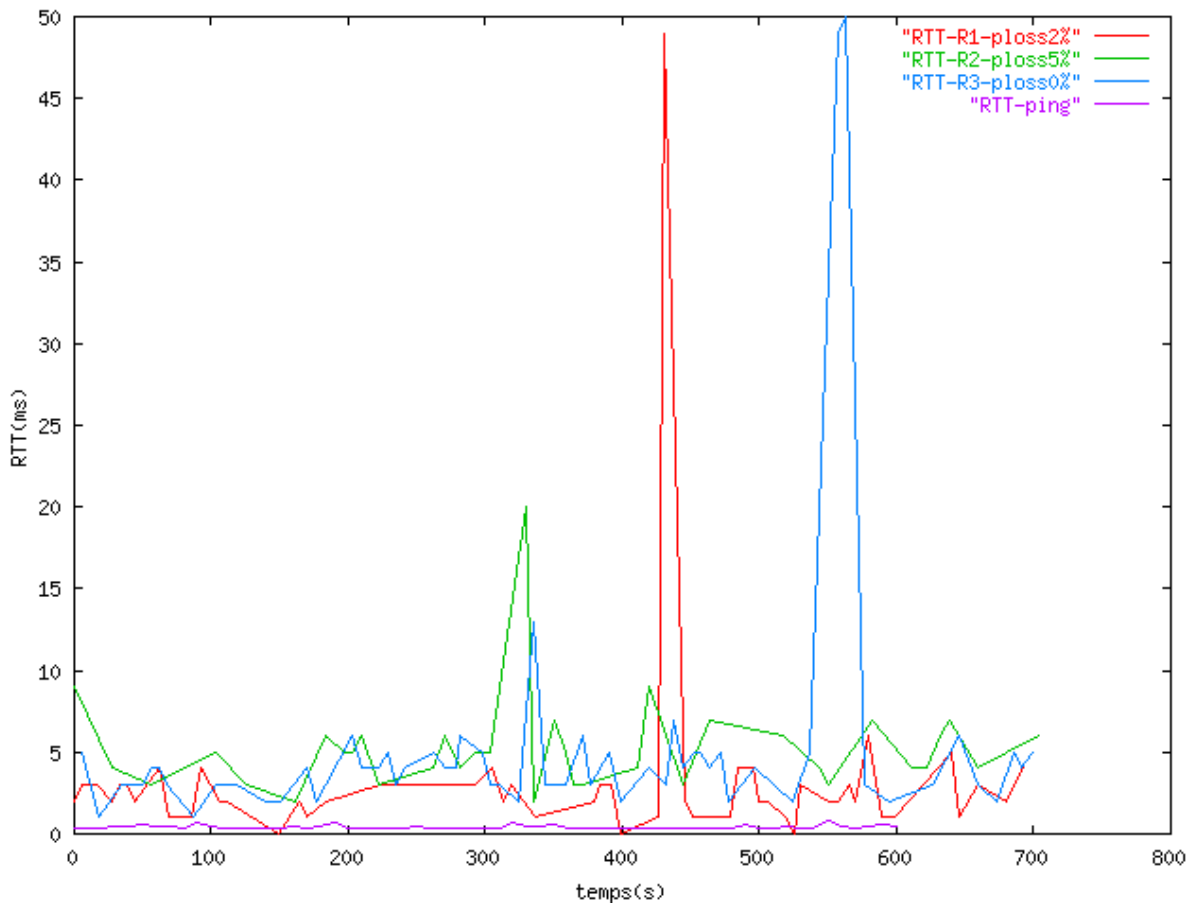


Fig. 4.2 Valeurs de RTT des récepteurs
trouvées suite à la première expérimentation
effectuée dans le réseau local de l'INRIA avec des machines Linux.

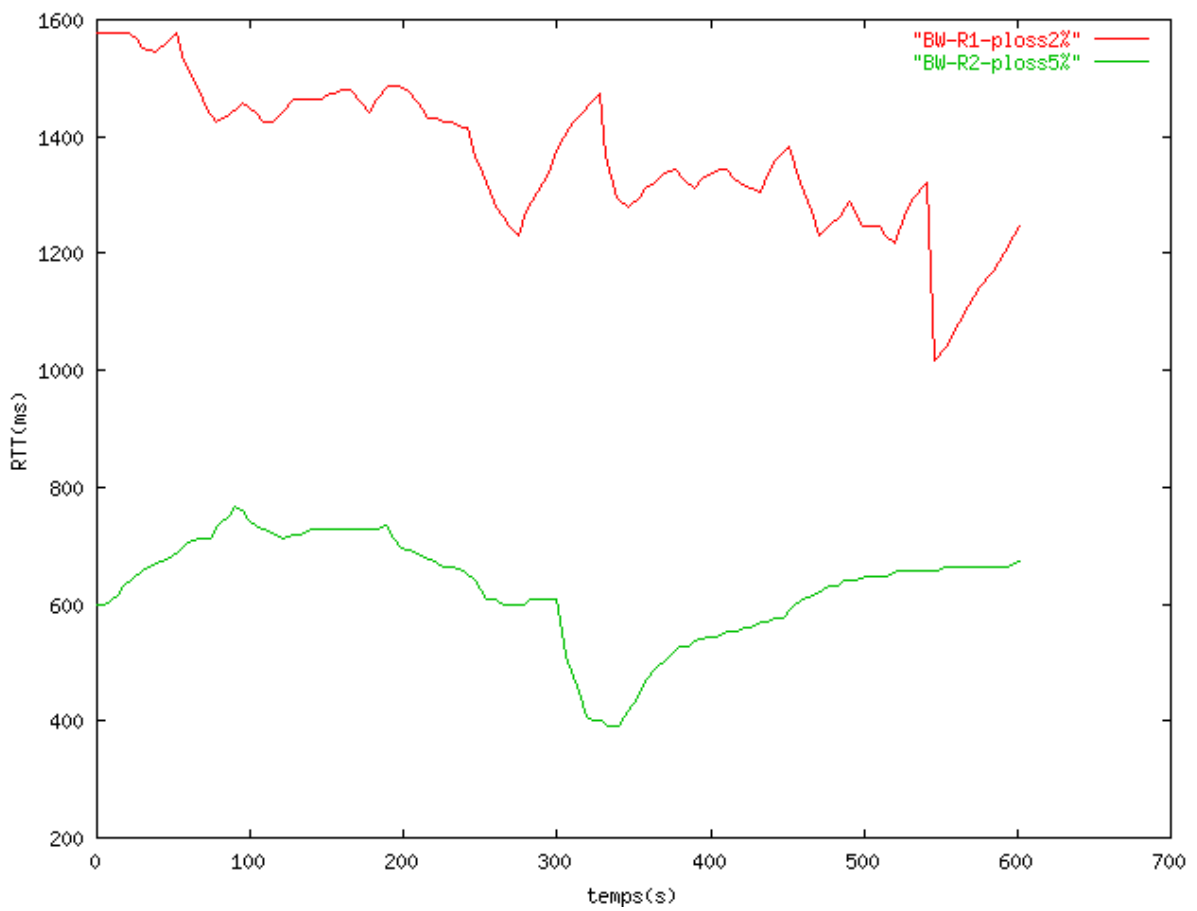


Fig. 4.3 Valeurs des bandes passantes disponibles des récepteurs trouvées suite à la première expérimentation effectuée dans le réseau local de l'INRIA avec des machines Linux.

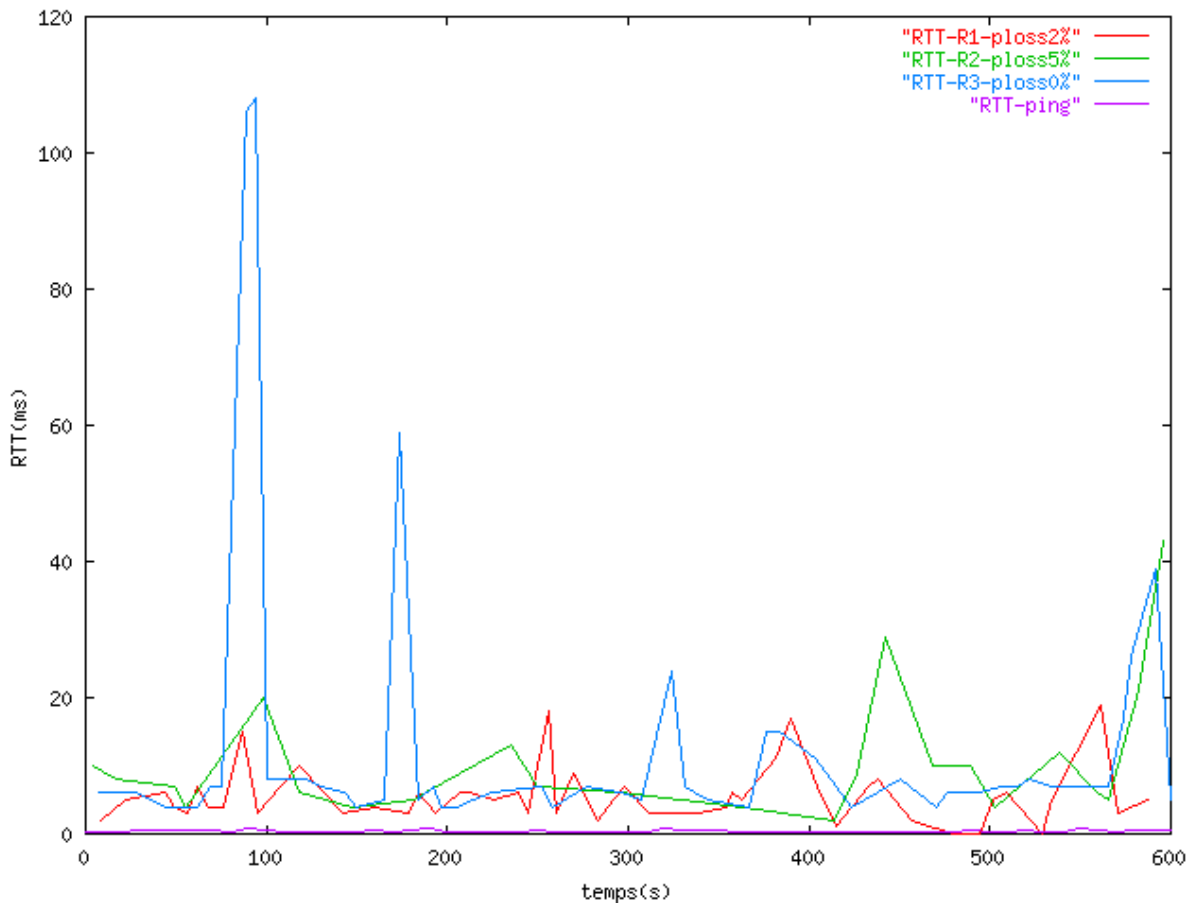


Fig. 4.4 Valeurs de RTT des récepteurs trouvées suite à la deuxième expérimentation effectuée dans le réseau local de l'INRIA avec des machines toutes Linux sauf la source qui est lancée sur une machine Solarise.

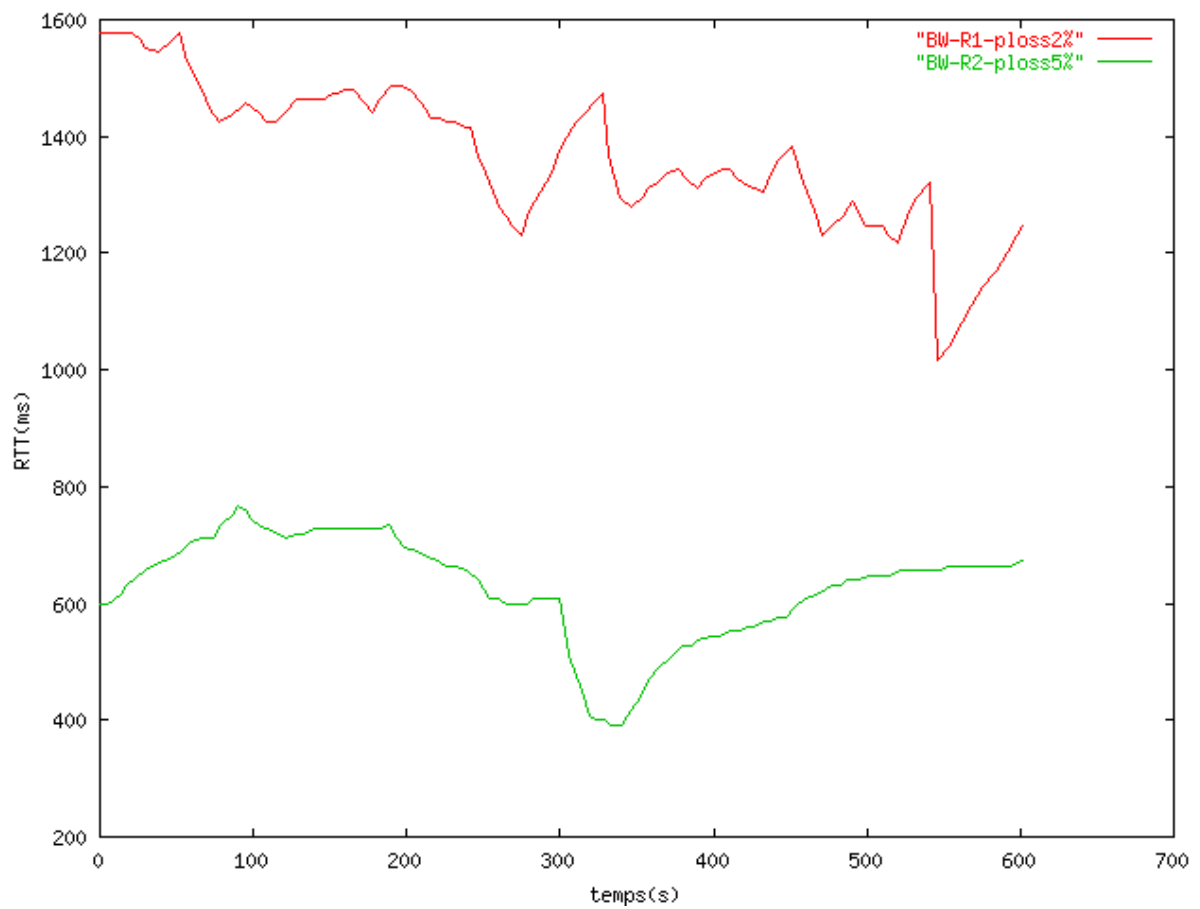


Fig. 4.5 Valeurs des bandes passantes disponibles des récepteurs trouvées suite à la deuxième expérimentation effectuée dans le réseau local de l'INRIA avec des machines toutes Linux sauf la source qui est lancée sur une machine Solarise.

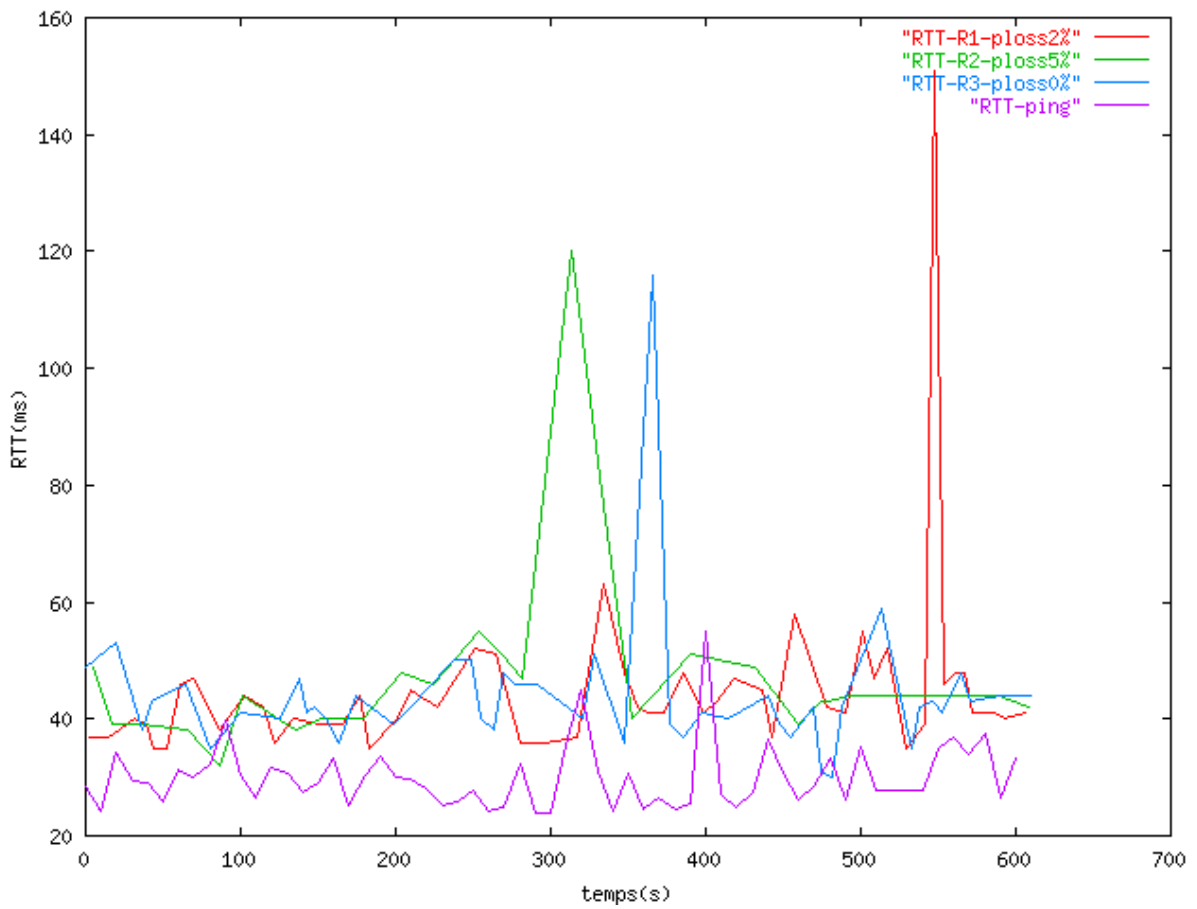


Fig. 4.6 Valeurs de RTT des récepteurs trouvées suite à la troisième expérimentation effectuée dans le Mbone entre les sites de Sophia et de l'IRISA Rennes.

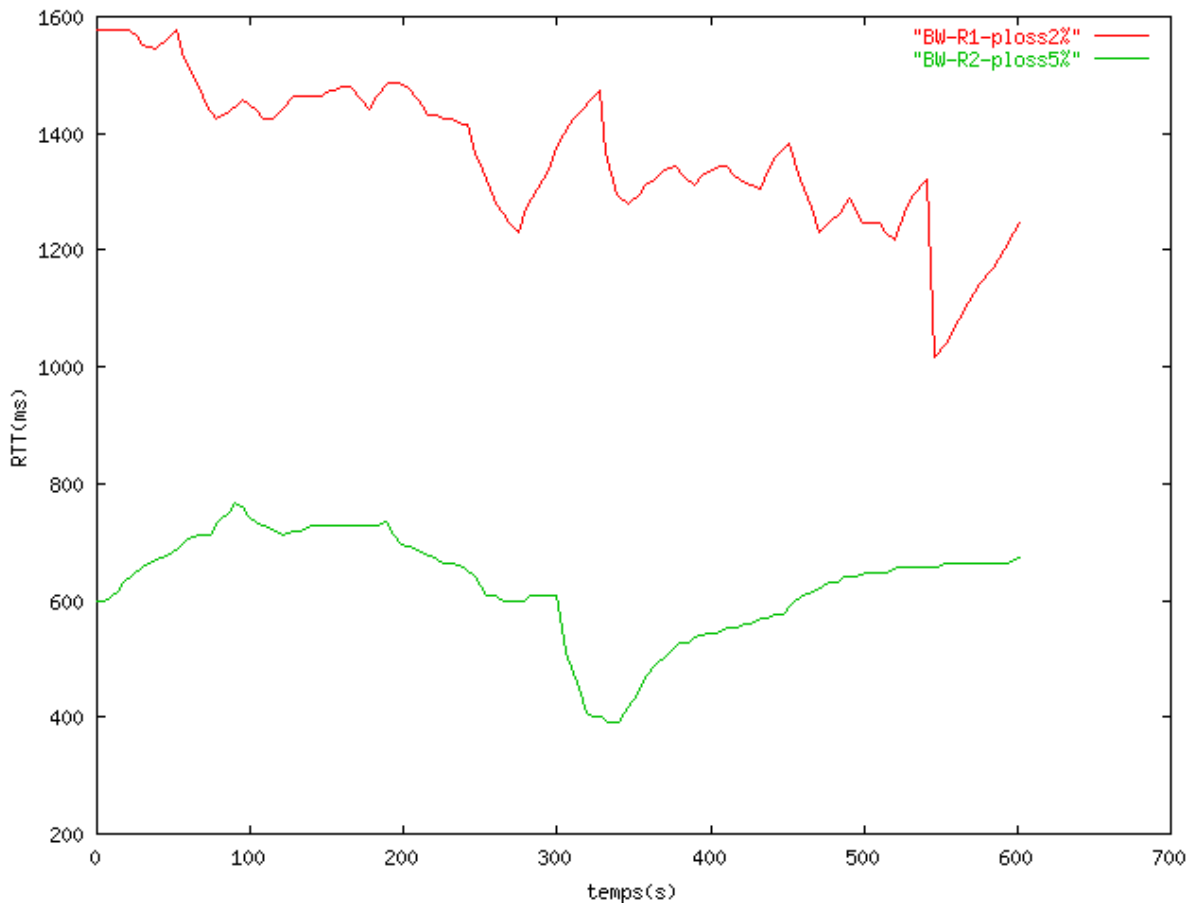


Fig. 4.7 Valeurs des bandes passantes disponibles des récepteurs trouvées suite à la troisième expérimentation effectuée dans le Mbone entre les sites de Sophia et de l'IRISA Rennes.

CONCLUSION

La transmission vidéo multipoint sur Internet reste un challenge, étant donné que le réseau Internet ne présente aucune garantie au niveau de la bande passante disponible et délai (latence, gigue).

L'hétérogénéité des récepteurs reste le problème le plus difficile à résoudre. À cause de cette hétérogénéité, l'état de réception des données n'est pas le même pour tous les récepteurs. Pour cette raison, les récepteurs doivent envoyer périodiquement à la source des rapports de réception concernant leurs états de réception, selon lesquels la source adapte son débit.

Dans son rapport de réception, le récepteur inclut sa bande passante disponible ainsi que son taux de perte.

Pour avoir un partage équitable de la bande passante du réseau entre les applications vidéos et le trafic Internet, qui est en majorité un trafic TCP, chaque récepteur peut calculer sa bande disponible avec un modèle TCP.

Les modèles TCP allouent au récepteur la même bande passante allouée avec le TCP sous les mêmes conditions du réseau, avec le même taux de perte et le même temps aller retour (RTT) entre ce récepteur et la source.

Le calcul du RTT entre les récepteurs et la source, présente un problème de scalabilité et gaspillage de la bande passante du réseau, car il demande l'échange des paquets de contrôle entre les récepteurs et la source.

Par suite, ces rapports de réception présentent un problème de scalabilité et un gaspillage de la bande passante du réseau.

Dans ce document, on a présenté quelques solutions proposées pour le calcul des RTT entre les récepteurs et la source, par suite les bandes passantes disponibles des récepteurs. On a étudié aussi quelques solutions qui adressent le problème d'hétérogénéité des récepteurs, ainsi que notre approche [12, 13] qui adresse tous les problèmes.

Notre approche propose une transmission des données en multicouches pour résoudre le problème de l'hétérogénéité des récepteurs.

Cette approche utilise des agrégateurs pour résoudre le problème de scalabilité des rapports de réception des récepteurs, et propose un algorithme scalable pour le calcul des RTT entre les récepteurs et la source, ainsi qu'elle présente une économie de la bande passante utilisée dans le réseau.

Dans ce document on a présenté aussi le travail effectué dans le cadre d'un stage de fin de DEA au sein de projet PLANETE de l'INRIA Sophia Antipolis.

Dans ce stage on a étudié les problèmes de la transmission multipoint sur Internet, on a adressé le problème de synchronisation de l'abonnement des récepteurs aux différentes couches de données, ainsi que la détection et la correction des collisions entre les identificateurs alloués par le protocole RTP à chaque élément de la session (section 2.6).

Dans ce travail on a implementé un algorithme de calcul de RTT entre les récepteurs et la source proposé dans notre approche qui sert à calculer les bandes passantes TCP-Freindly des récepteurs [18].

De plus, on a analysé les performances de cet algorithme et on l'a comparé avec d'autres approches.

BIBLIOGRAPHIE

- [1] A. Basu and S. J. Golestani, "Estimation of Receiver Round Trip Times in Multicast Communications," 1999, http://www-scf.usc.edu/~junsoole/research/tcp_friendly/.
- [2] J.C. Bolot, T. Turletti and I. Wakeman, "Scalable Feed-back Control for Multicast Video Distribution in the Internet," in *proc. of SIGCOMM*, August 1994, pp.58-67.
- [3] I. Rhee, N. Balaguru and G.N. Rouskas "MTCP » :Scalable TCP-like congestion control for reliable multicast," in *Proceeding of the conference on computer communications (IEEE Infocom)*, New York, USA, Mars 1999.
- [4] S. Paul, K. Sabnani, J.C.Lin and S.Bhattacharyya, "Reliable Multicast Transport Control (RMTP)," *IEEE Journal on selected Areas in Communications*, April 1997.
- [5] L. Vicisano, L. Rizzo and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," in *Proceeding of the conference on computer communications (IEEE Infocom)*, San Francisco, USA, Mar. 1998.
- [6] A. Legout and E.W. Biersack, "Fast convergence for cumulative layered multicast transmission scheme," Tech. Rep., Eurecom, Sophia-Antipolis, France, Oct. 1999, under submission.
- [7] T. Jiang, E. Zegura and M. Ammar, "Inter-receiver fair multicast communication over the internet," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Basking Ridge, New Jersey, June 1999.
- [8] S.Y. Cheung, M.H. Ammar and X. Li, "On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution," in *proc. of IEEE Infocom*, 1996.
- [9] E. Amir, S. McCanne and H. Zhang, "An Application-level Video Gateway," in *Proc. of ACM Multimedia*, November 1995.
- [10] B. J. Vikers, C. Albuquerque and T. Suda, "Source Adaptive Multi-layered Multicast Algorithm for Real-Time Video Distribution," Technical Report ICS-TR 99-45, University of California, Irvine, June 1999.
- [11] D. Sisalem and A. Wolisz, "MLDA : A TCP-Friendly Congestion Control Frame for Heterogeneous Multicast Environments," in *Eighth International Workshop on Quality of Service (IWQoS 2000)*, Pittsburgh, PA, June 2000.
- [12] K. Salamatian and T. Turletti, "Classification of receivers in large multicast groups using distributed clustering," in *Proceedings of Packet Video*, May 2001.

- [13] X. Hénocq, C. Guillemot, K. Salamatian and T. Turetletti, "Joint source and channel rate control for multilayered multicast video transmission based on a distributed clustering algorithm".
- [14] "RTP : A Transport Protocol for Real-Time Applications," Internet Engineering Task Force, INTERNET-DRAFT, draft-ietf-avt-rtp-new-06, January 14, 2000.
- [15] K.L. Calvert, J. Griffioen, A. Sehgal and S. Wen, " Concast : Design and implementation of a new network service," in proceeding of International Conference on Network Protocols, Toronto, Ontario, 1999.
- [16] E. Amir, S. McCanne and H. Zang, "An application level gateway," *in proc. Of ACM Multimedia*, San Francisco, California, November 1995.
- [17] R. Droms, "Dynamique host configuration protocol," *RFC-1541*, Octobre 1993.
- [18] "TCP-Friendly Rate Control (TFRC) : Protocol Specification," Internet Engineering Task Force, INTERNET-DRAFT, draft-ietf-tsvwg-tfrc-00, 17 November 2001.