MEMOIRE D'HABILITATION A DIRIGER DES RECHERCHES

Numéro d'ordre 51-2003

présenté par

M. PHAM Cong-Duc

pour l'obtention du diplôme

D'Habilitation à Diriger des Recherches

Simulations parallèles sur grappes de machines, Multicast fiable actif, Optimisations de systèmes de communication: Quelques contributions pour la résistance au facteur d'échelle.

Soutenue publiquement le 16 décembre 2003

Après avis de:

Ken	CHEN
Jean-Jacques	PANSIOT
Brigitte	PLATEAU

Devant le jury composé de:

Ken	**Chen**	**Rapporteur** ***Professeur à l'université Paris 13***
Serge	**Fdida**	**Examinateur** ***Professeur à l'université Paris 6***
Raymond	**Namyst**	**Examinateur** ***Professeur à l'université Bordeaux 1***
Jean-Jacques	**Pansiot**	**Rapporteur** ***Professeur à l'université de Strasbourg***
Brigitte	**Plateau**	**Rapporteur** ***Professeur à l'ENSIMAG***
Vincent	**Roca**	**Examinateur** ***Chargé de Recherche INRIA, projet Planète***
Behzad	**Shariat**	**Examinateur** ***Professeur à l'université Lyon 1***
Bernard	**Tourancheau**	**Examinateur** ***Directeur de recherche à SUN Labs Europe***

À mes parents.

Remerciements

Les travaux qui sont présentés dans ce document sont le fruit de travail (acharné!) de plusieurs étudiants, en thèse ou simplement en stage. Je tiens à remercier Patrick Geoffray et Roland Westrelin, étudiants en stage, puis en thèse avec B. Tourancheau lors de mon arrivée à Lyon. Beaucoup de mes travaux sur la simulation parallèle sur grappes de machines n'auraient pas été possibles sans leur aide. C. Albrecht, un étudiant venu d'Allemagne pour un court séjour doit également être mentionné pour son aide sur la simulation parallèle sur cluster, aide courte, mais efficace.

Moufida Maimour m'a contacté il y a plus de 3 ans (septembre 1999 pour un début de thèse en septembre 2000) pour commencer une thèse dans le domaine des réseaux. Je lui ai proposé de travailler sur le multicast et les réseaux actifs. Une grande partie des travaux présentés dans ce document viennent de cette collaboration. Je la remercie de la confiance qu'elle a pu mettre en moi, tout jeune chercheur fraîchement nommé. C'est une étudiante qui a toute mon estime, avec une autonomie remarquable et déjà une très bonne méthodologie de travail. Eric Lemoine a commencé sa thèse avec moi en septembre 2001 (mais a débuté son contrat CIFRE dès mai 2001) sur l'optimisation des sous-systèmes de communication en utilisant des cartes d'interface programmables. Il a développé une très grande expertise technique dans ce domaine particulièrement difficile. Il y a aussi dans cette aventure initiale d'autres étudiants, B. Gaidioz, J.P. Gelas, M. Herbert, qui, avec leur confiance et leur bonne humeur m'ont permis de toujours croire en la recherche même lors de moments difficiles liés à la constitution et à la restructuration de l'équipe.

Je remercie très chaleureusement Laurent Lefèvre, collègue nommé un an avant mon arrivée sur Lyon, pour son éternel sourire, sa bonne humeur inattaquable, son fort sens des responsabilités, sa complicité et son amitié au quotidien. Je remercie également Pascale Primet pour avoir pris les rennes de l'équipe et permettre à tous de pouvoir "chercher tranquillement". L'équipe RESO installée au LIP est maintenant une grosse équipe. L'ambiance chaleureuse et décontractée qui y règne est à mon avis unique et c'est grâce à l'apport de tous les autres membres de l'équipe.

Je n'oublie pas Bernard Tourancheau, celui qui m'a accueilli et aider à faire mes premières dents en tant que chercheur autonome. Je le remercie pour sa confiance en moi, pour son soutien et ses conseils avisés. Il m'a fait l'honneur d'accepter d'être présent dans ce jury. Je tiens ensuite à remercier les rapporteurs de ce document qui ont consacrés du temps à lire et à commenter ce travail: Ken Chen, Jean-Jacques Pansiot et Brigitte Plateau. Je remercie aussi les autres membres du jury: Serge Fdida, Vincent Roca et Behzad Shariat pour avoir acceptés d'examiner ce document et à me faire bénéficier de leur grande expérience.

En dernier lieu dans cette longue liste de remerciements, mais en première position dans mes pensées, je tiens à remercier d'une part mes parents pour m'avoir offert la possibilité de poursuivre des études supérieures jusqu'à l'obtention de la thèse. Cette HDR, qui n'a bien sûr pas la même valeur symbolique à leur yeux, est néanmoins le fruit de leur travail quotidien.

Table des matières

Chapitre 1

Introduction

1.1 PARCOURS

Ce document réalise une synthèse des activités d'enseignement et de recherche que j'ai menées après ma nomination en octobre 1998 à l'université Claude Bernard Lyon 1 (UCBL) sur un poste de Maître de Conférence. Ces travaux ont été réalisés au sein de la jeune équipe RESAM de l'université Lyon 1 puis dans l'équipe INRIA RESO/LIP (UMR CNRS-INRIA-ENS-UCBL) à l'Ecole Normale Supérieure.

Durant ma thèse, j'ai travaillé sur des problématiques de simulations parallèles dans le but d'accélérer la simulation et l'évaluation de modèles de réseaux de grande dimension. Les modèles considérés étaient des modèles de réseaux ATM avec des algorithmes de routage spécifiques. J'ai ensuite travaillé sur la simulation distribuée à grande échelle pendant mon année post-doctorale à l'UCLA (Université de Californie, Los Angeles, USA) et notamment sur l'architecture HLA (High Level Architecture) pour les simulations distribuées interactives. J'ai poursuivi dans ce thème de recherche à Lyon en m'orientant vers l'utilisation de plate-forme de calcul à base de grappes de machines (*clusters*) plutôt que sur des machines massivement parallèles comme c'était le cas lors de ma thèse. De nouvelles problématiques ont alors été étudiées pour accélérer au mieux les simulations sur ce type d'architecture.

Parallèlement à ces travaux, j'ai commencé à m'orienter aussi vers l'évaluation (grâce à l'expérience acquise en simulation) et la conception de protocoles de multicast fiable pour l'Internet et les grilles de calcul. J'étudie plus particulièrement des approches utilisant des éléments actifs/programmables dans le réseau. J'ai encadré une thèse sur le thème du multicast fiable actif qui a été soutenue fin novembre

2003. Depuis 2 ans, je dirige aussi des travaux sur l'optimisation des sous-systèmes de communication dans les machines serveurs (et en particulier l'interface entre le matériel et le système de communication), en collaboration avec SUN Labs Europe.

1.2 ACTIVITÉS D'ENSEIGNEMENT

Lors de mon arrivée à l'Université Claude Bernard Lyon 1, les filières réseaux existantes étaient le DESS IIR option Réseaux et le DESS CCI option Réseaux. La filière MIAG et Maîtrise d'informatique comportent également un module de réseau. C'est tout naturellement que j'ai effectué mes enseignements dans ces filières. Avec très peu d'expérience en enseignement effectué pendant ma thèse, les débuts ont été difficiles car il fallait créer les supports de cours, les sujets de TD et les fiches de TP. En effet, l'enseignement en réseau était encore récent et très peu de supports avaient été structurés auparavant, surtout pour les TDs et TPs.

En DESS IIR option Réseaux, j'ai monté et structuré 6 TPs réseaux en basant certaines d'entre eux sur d'anciennes fiches réalisées par L. Prylli, alors en thèse avec B. Tourancheau qui était à l'époque responsable du DESS IIR option Réseaux. J'ai aussi monté le cours Performance et Simulation dans cette formation car j'ai estimé que c'était un domaine important pour de futurs responsables des systèmes d'information. Avec l'aide importante de J. Bonneville, nous avons aussi structuré les TPs de réseaux grande distance X.25 avec des commutateurs X.25 OST PASS8.

En MIAG et en Maîtrise Informatique, j'ai monté le cours de Réseaux de Communication et les TDs associés. Les TDs ont démandés un investissement initial considérable puisqu'il fallait faire les sujets pour 20h de TD par étudiants. J'ai ensuite utilisé la partie Réseaux Grande Distance du cours Réseaux de Communication pour les DESS CCI (formation initiale et en apprentissage). J'ai renoncé en 2002 à enseigner en MIAG et en DESS CCI apprentissage à cause de mes charges dans d'autres formations.

J'ai ensuite commencé à enseigner le cours de Réseaux de Communication dans le Magistère d'Informatique et Modélisation, 2ème année (MIM2), localisé à l'ENS.

J. Bonneville m'a sollicité en 2002 pour enseigner en IUP réseaux, formation nouvellement crée au sein de l'UFR. J'ai accepté en montant un cours de Réseaux Haut-Débit et Qualité de Service. Pour 2003-2004, une autre personne s'est proposée, ce qui m'a permis de prendre d'autres formations.

En ce concerne les cours en DEA, j'interviens depuis l'année 2002-2003 dans le DEA DISIC à l'INSA, et depuis cette année dans le DEA DIF à l'ENS. Ces 2 formations sont co-habilitées par l'université Claude Bernard Lyon 1.

Depuis 1999, suite au départ de B. Tourancheau, j'ai été d'abord co-responsable (avec L. Lefèvre), puis responsable (depuis 2000) du DESS IIR option Réseaux. C'est une lourde responsabilité mais qui m'a permit d'établir des contacts avec beaucoup d'acteurs dans la région lyonnaise. C'est une expérience très enrichissante. Je suis toujours responsable de cette formation et j'essaie d'adapter chaque année l'organisation des interventions en fonction des orientations du marché. Par exemple,

les cours de sécurité ont pris plus d'ampleur ces 2 dernières années, des cours de Java et de programmation web ont été développés pour répondre à l'attente des entreprises, des cours de réseaux sans fils ont été introduits. Si l'on fait attention à préserver l'indépendance d'une formation de 3ème cycle par rapport aux lois très éphémères du marché de l'emploi, ce travail d'adaptation est absolument nécessaire sinon on prend le risque de former des étudiants qui ne pourront pas se positionner sur un marché en ce moment plus difficile que les autres années.

Actuellement, j'interviens donc en DESS IIR Réseaux, Maitrise Informatique, MIM2, DESS CCI, DEA DISIC et DEA DIF. J'ai eu la chance de concentrer ces enseignement dans le domaine des réseaux de communication qui est ma spécialité en recherche. Les enseignements réseaux, et surtout les TDs et TPs sont de plus en plus structurés. Plusieurs formations bénéficient de l'expérience accumulée, et l'arrivée d'Olivier Glück nouvellement nommé MCF UCBL en Septembre 2003 a contribuer à rendre les enseignements plus stables.

Avec le recul acquis grâce à ces années d'enseignement, je constate que je prends toujours plaisir à enseigner, et surtout à essayer de trouver la manière de présenter les connaissances sous une forme agréable et plus facilement assimilable par l'étudiant. Bien sûr, certaines notions sont complexes, mais néanmoins, avec un peu d'effort, il est possible de les enseigner mieux. Je prends conscience à présent que la recherche doit nous permettre, à nous enseignant-chercheur, de mieux faire passer le savoir (un savoir mis à jour grâce à nos recherches de pointes) vers les étudiants. Les cours en DEA sont importants puisque les auditeurs sont directement les étudiants potentiellement récupérable en thèse, mais la formation en premier et second cycle est très importante. Je n'ai pas fait d'enseignement en 1er cycle et je pense que c'est une bonne chose pour les nouveaux MCF nommés. En effet, je penses qu'enseigner en 1er cycle est plus difficile car la pédagogie prime sur le contenu avec de jeunes étudiants. Je penses maintenant être prêt pour le 1er cycle et je prendrai grand plaisir à le faire si l'occasion se présentait.

1.3 CONTEXTE DE RECHERCHE

J'ai assisté pendant ma thèse au formidable développement de l'Internet et surtout du World Wide Web[1] (figure 1.1). J'ai installé au début de ma thèse le premier serveur web du MASI (maintenant LIP6) et écrit les premières pages web du laboratoire. Le développement du web se poursuit, sans ralentir semble t-il, poussé par l'irrésistible montée en débit dans le cœur des réseaux d'opérateurs grâce aux techniques de multiplexage en longueur d'onde (DWDM) sur fibres optiques. De manière complémentaire, bien qu'à des ordres de grandeurs moindres, il y aussi depuis peu une augmentation des débits dans les réseaux locaux (Gigabit Ethernet et 10 Gigabit Ethernet) et dans les accès haut-débit chez les particuliers avec pour l'instant des

[1]Selon `http://www.zakon.org/robert/internet/timeline` le nombre de postes Internet passe durant la période sep.93-juil.97 de 2 millions à 26 millions. Le nombre de sites www passe de 204 à 1.203.096. Nous sommes actuellement à environ 40 millions de sites web pour plus de 162 millions de postes.

techniques comme ADSL mais d'autres solutions sont déjà à l'étude (VDSL, Ethernet First Mile (EFM), Passive Optical Network, Boucle Locale Radio...). Cette dernière évolution change complètement la manière dont on "surfe" à partir de chez soi et les particuliers vont être dans les années à venir la cible principale pour de nouvelles applications à grande échelle (visio-phone, VoD...).

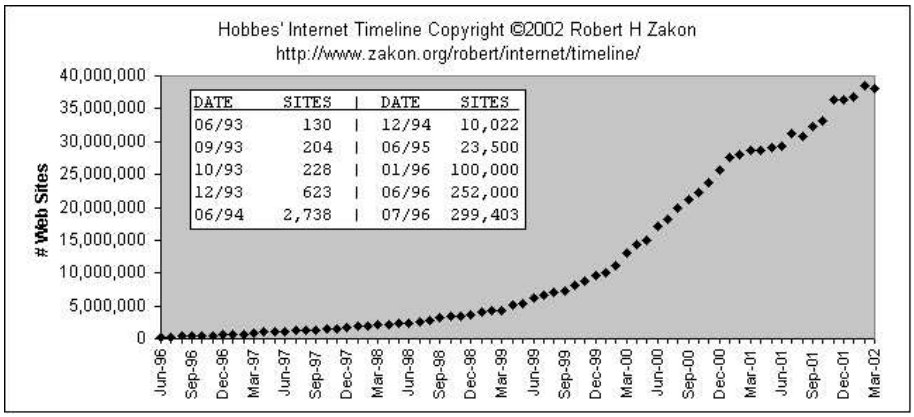

Figure 1.1 Evolution du nombre de site www.

L'Internet devient donc plus rapide généralement mais avec une très grande différence entre le cœur du réseau (2.5 Gbits/s à 40 Gbits/s) et les extrémités (de 36Kbits/s à quelques centaines de MBits/s pour les réseaux académiques). Cette différence est même supérieure à ce qui existait il y a quelques années avant le déploiement massif des techniques DWDM d'autant plus que certaines parties de l'Internet ont pu bénéficier de l'augmentation spectaculaire des débits dans les réseaux d'opérateurs sans pour cela voir leurs débits d'accès augmenter! Du fait de cette différence en débit qui existe, et existera toujours, les protocoles de l'Internet ont la rude tâche de devoir gérer une hétérogénéité en débit plus variée encore qu'auparavant, avec plus d'utilisateurs connectés! Par exemple, si on considère le protocole TCP, celui-ci devra pouvoir à la fois être efficace sur un lien au gigabits/s et sur un lien au kbits/s. De plus, il devra être équitable pour le plus grand nombre d'utilisateurs ayant des accès différents. Dans le cas des communications de groupes, le problème est encore plus ardu!

C'est dans ce contexte que les 3 thèmes de recherche que j'étudie (simulation parallèle, multicast fiable actif et optimisation de sous-systèmes de communication) ont pour point commun les mots-clés suivants: résistance au facteur d'échelle (*scalability*) et performance de bout-en-bout (c'est-à-dire d'application à application). Bien qu'assez différentes les unes des autres, les recherches effectuées dans ces thèmes m'aident à mieux comprendre et à mieux appréhender les réseaux haut-débit et les protocoles de communication.

1.3.1 Résistance au facteur d'échelle

La résistance au facteur d'échelle est une propriété indispensable de nos jours car les réseaux de communication sont de plus en plus grands et complexes. Cette préoccupation est au centre des recherches effectuées actuellement sur les réseaux IP (Internet) celles-ci concernent un grand nombre de domaines: routage, signalisation et gestion, QoS (voir le numéro spécial sur "Scalability in IP-oriented Networks", IEEE Communication Magazine, Vol. 41(6), June 2003).

Cette propriété est indispensable dans les protocoles de l'Internet, mais aussi dans les outils utilisés pour leur évaluation. En effet, il est tout aussi important de disposer d'outils d'évaluation permettant l'étude de systèmes complexes et de grande dimension pour pouvoir anticiper sur les performances des systèmes qui seront développés.

Ce souci de résistance au facteur d'échelle se trouve donc au centre de mes thèmes de recherche. Pour la simulation parallèle, je développe des techniques qui résistent au facteur d'échelle lorsque les modèles de simulation sont trop grands et comportent trop d'objets à simuler. Pour le multicast et l'optimisation des sous-systèmes de communication je propose des mécanismes qui permettent dans les réseaux réels de résister au facteur d'échelle lorsque le nombre de machines impliquées devient très grand.

Les travaux que j'effectue sur la simulation parallèle sont une suite de ce que j'ai fait dans ma thèse. Ce que je présente dans ce document concerne l'utilisation de grappes de machines comme plate-forme d'exécution pour les algorithmes de simulation parallèle. Les performances que l'on peut obtenir de l'exécution d'une simulation parallèle sur ce type d'architecture dépendent très fortement des latences des communications entre les processeurs et du nombre de processeurs. Il faut donc étudier très précisément la manière dont les messages sont envoyés et comment les charges sont réparties afin d'obtenir les latences les plus faibles, et donc des gains plus importants lorsque les modèles sont très grands.

Pour le multicast fiable et les sous-systèmes de communication, la résistance au facteur d'échelle doit être prise en compte dès la conception de la solution. Pour le multicast, cela concerne essentiellement les mécanismes de retransmission et de contrôle de la congestion car ceux-ci sont les plus pénalisants à grande échelle. Le problème est ardu car il y a de multiples causes complexes. Il y a d'une part les messages de contrôle envoyés par les récepteurs vers la source qui, à grande échelle, vont saturer celle-ci. Une conséquence directe est l'augmentation très forte des latences de recouvrements des erreurs, mais aussi des paquets orignaux. Il y a ensuite d'autre part les ralentissements dus à la présence dans le groupe d'éléments faibles (récepteurs sous des liens très congestionnés) qui limitent le débit possible. Des cas pathologiques sont malheureusement très fréquents où le débit de la source devient nul à cause d'un seul récepteur fautif. Pour les sous-systèmes de communication, cela concerne la robustesse du système et sa capacité à offrir un niveau de performance non dégradé.

La résistance au facteur d'échelle est un verrou important qui limite le déploiement

à grande échelle du multicast. Les solutions proposées devront pouvoir gérer la présence d'un grand nombre de récepteurs dans le groupe et fournir de faibles latences et des haut-débits! Il y a aussi la sécurité, mais ceci est un autre sujet. Pour la simulation parallèle, il est plus réaliste de dire que la résistance au facteur d'échelle est indispensable mais non suffisante, et ce qui limite son utilisation est plutôt la simplicité et la transparence pour l'utilisateur des mécanismes proposés et leur intégration dans des outils "clé-en-mains".

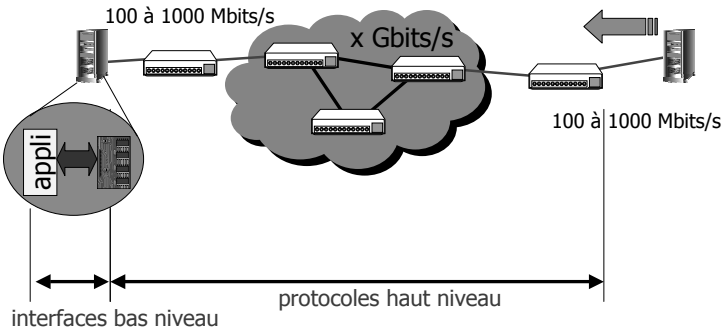

Figure 1.2 Performance de bout-en-bout.

1.3.2 Performances de bout-en-bout

Apporter des performances de bout-en-bout nécessite d'intervenir à 2 niveaux: un niveau très proche de la machine et du système d'exploitation, et un niveau plus élevé dans les couches de communication. Il est important de se rendre compte que l'optimisation d'une partie seule, sans toucher à l'autre, ne permet pas l'obtention de performance de bout-en-bout. La figure 1.2 illustre les 2 niveaux auxquels nous faisons référence.

Si nous prenons par exemple le débit comme critère de performance, ce débit peut être limité (inférieur à la capacité physique du lien) à l'intérieur du réseau de distribution (le nuage représentant l'Internet), auquel cas ce sont les protocoles de niveau réseau et transport qui doivent être optimisés (si le matériel n'est pas en cause). Si c'est au niveau de la machine d'extrémité, ce sont des mécanismes de plus bas-niveau, comme la gestion des tampons mémoire ou celle des interruptions, qui doivent être étudiés (à des débits élevés, les interruptions matérielles peuvent saturer le processeur par exemple) ou alors l'implémentation des protocoles. Nous voyons bien qu'intervenir à un seul endroit sur le chemin des données est inefficace: rien ne sert de délivrer rapidement des paquets dans le réseau d'une source vers une destination si la machine destinatrice présente un goulot d'étranglement, et inversement. C'est pourquoi mes recherches comportent les deux volets de la problématique.

ORGANISATION DU DOCUMENT

Ce document est composé de plusieurs chapitres, chacun d'eux expliquant rapidement la problématique scientifique du domaine de recherche avec ensuite une synthèse des travaux que j'ai effectués. Le document n'a pas été écrit dans le but de détailler techniquement les problémes étudiés, mais plutôt de présenter la problématique générale, les acquis actuels de la communauté de recherche, ma reflexion personnelle et les directions de recherche prises et celles qui me semblent les plus importantes dans un futur proche. Un certain nombre d'articles que j'ai publiés avec les étudiants en thèse sur ces thèmes ont été sélectionnés pour être inclus en annexe afin de présenter ce que le document ne fournit pas, c'est-à-dire les détails concernant les propositions réalisées. Le manuscrit est décomposé de la manière suivante:

- Chapitre 2: ce chapitre présente mes travaux sur la simulation parallèle sur des architectures de grappes de machines. Je présente en particulier les travaux d'optimisations pour réduire les latences des communication afin d'obtenir des gains sur ce type d'architecture.
- Chapitre 3: ce chapitre présente les travaux de thèse de Moufida Maimour sur les protocoles de diffusion fiable dans le contexte des réseaux actifs/programmables. Je présente mes motivations d'étudier ce problème, la méthodologie adoptée et les grandes lignes de nos propositions. Une petite partie sera consacrée à décrire comment nos propositions peuvent être bénéfiques pour les grilles de calculs.
- Chapitre 4: ce chapitre présente les travaux de thèse d'Eric Lemoine sur l'optimisation des sous-systèmes de communication. Cette thèse CIFRE s'effectue en collaboration avec SUN Labs Europe. Ce thème de recherche encore nouveau pour moi n'est pas encore entièrement finalisé, mais des résultats intéressants ont déjà été proposés.
- Chapitre 5: c'est la conclusion de ce document d'habilitation à diriger les recherches.

Chapitre 2

Simulations parallèles sur grappes de machines

2.1 INTRODUCTION

Les protocoles de l'Internet ne sont pas simples! Du moins, ils ne le sont plus depuis que l'Internet a pris une telle ampleur. La recherche dans le domaine des réseaux et des protocoles de communication nécessite d'évaluer, principalement de manière quantitative, les performances des systèmes en jeu afin d'en maîtriser et d'optimiser le fonctionnement. De même, toute proposition d'un nouveau protocole ou mécanisme doit être accompagnée d'une étude détaillée de son comportement, si possible à grande échelle. De manière non exhaustive, l'étude des protocoles de routage, des mécanismes de marquage, des communications point-à-multipoint et du contrôle de la congestion par exemple, nécessite souvent d'évaluer des systèmes de grande dimension. Il est alors nécessaire de disposer de nouveaux outils d'évaluation car la complexité des systèmes étudiés augmente ainsi que leur dimension, rendant des études de phénomènes transitoires, ou même stationnaires, délicates voire impossibles dans le contexte actuel.

Pour résoudre ces modèles nous avons le choix entre les méthodes analytiques et la simulation. La plupart des systèmes informatiques peuvent être considérés comme un ensemble de ressources et de clients concourant pour ces ressources. Des phénomènes d'attente inhérents à ces systèmes font que les méthodes de résolution analytiques utilisent naturellement le formalisme des files d'attente. Les réseaux de files d'attente donnent la possibilité de modéliser un système donné puis d'en

extraire des critères de performances. Ces méthodes sont les moins onéreuses puisque la résolution se fait analytiquement. En contre partie, le champ d'application de ces méthodes est relativement réduit car celles-ci imposent souvent des hypothèses de fonctionnement très restrictives qui nécessitent de fortes simplifications pouvant rendre l'étude non réaliste.

La simulation, contrairement aux méthodes analytiques, ne présente pas de limites théoriques quant au niveau de détail du modèle. Grâce à cela, elle est devenue un outil largement utilisé dans l'évaluation de performance des systèmes informatiques. Cependant cette fantastique souplesse se paye cher, et les principaux facteurs limitatifs sont le temps de simulation et la capacité mémoire. Par exemple, pour évaluer des algorithmes de routage, il est nécessaire de descendre jusqu'au niveau du paquet pour réellement obtenir des critères intéressants. Malheureusement, simuler à ce niveau implique l'exécution de millions d'événements, ce qui allonge dangereusement le temps de simulation, si celle-ci tient en mémoire!

Les méthodes traditionnelles de simulation séquentielle ne peuvent plus satisfaire à l'augmentation constante de la taille et de la complexité des modèles. Des techniques de simulation distribuée ont été proposées et j'ai moi-même travaillé sur ces aspects durant ma thèse. Ces techniques, encore méconnues du grand public et d'un usage délicat sont efficaces si on les applique de manière bien spécifique. J'ai ainsi montré pendant ma thèse qu'il était possible de réduire considérablement les temps de simulation et ainsi d'augmenter la taille des systèmes étudiés sur des modèles de réseaux ATM.

Cependant, un des verrous qui empêchent l'utilisation de telles techniques par les personnes dont l'évaluation de modèles de réseaux est la tâche quotidienne consiste en l'accès à des machines parallèles coûteuses et très sollicitées. La réalité que les chercheurs dans le domaine des réseaux vivent au quotidien est la suivante: ils utilisent les machines qui sont disponibles dans leur laboratoire (c'est très différent dans d'autres domaines de recherche comme la physique par exemple où l'utilisation de ressources distribuées est indispensable et chose courante). Inévitablement donc, les modèles étudiés sont plus petits, moins généraux, moins représentatifs de la réalité. Les travaux décrits dans ce chapitre sont motivés par ce constat et une des solutions que j'ai regardée est l'utilisation de petites grappes de machines connectées entre elles par un réseau d'interconnexion rapide. "Petit" signifiant 8 à 16 processeurs maximum. Nous verrons par la suite que de telles infrastructures sont beaucoup plus adaptées aux besoins des chercheurs dans le domaine des réseaux et des protocoles.

Organisation du chapitre

Le reste de ce chapitre est organisé comme suit: la section 2.2 rappelle rapidement les principes de base de la simulation et les techniques de simulation parallèle. Ils sont fournis dans ce document dans le seul but de clarifier la compréhension de la suite de mes travaux sur la simulation parallèle sur grappes de machines. La section 2.3 présente les spécificités des grappes de machines et la section 2.4 présente ensuite de manière synthétique les problématiques associées à la simulation parallèle sur

grappes de machines et les travaux que j'ai effectués. Une sélection des articles publiés pourra être trouvée à la fin du document en annexe.

2.2 RÉSUMÉ DES TECHNIQUES DE SIMULATION PARALLÈLE

2.2.1 Modèles

Simuler revient à imiter le comportement d'une personne, d'une machine, d'un phénomène etc., bref plus généralement d'un *système* physique pour l'étudier. Pour cela, il est nécessaire de faire des suppositions, plus ou moins précises, sur la façon dont un système donné fonctionne. L'ensemble de ces suppositions qui prennent le plus souvent la forme de relations mathématiques ou logiques constitue un *modèle*. Ce modèle servira ensuite à représenter le système, cela dans le but de mieux le comprendre et éventuellement d'étudier les impacts de telle ou telle modification.

Si les relations qui composent le modèle sont simples, alors il est possible d'utiliser un modèle mathématique qui fournira des résultats exacts sur le comportement du système—solution analytique. Par contre, si ces relations sont plus complexes, on atteint très vite les limites des modèles mathématiques, qui typiquement nécessitent très souvent des simplifications ou des hypothèses de fonctionnement très restrictives. Dans ce cas, la simulation est souvent le dernier recours pour l'étude du système. Cette dernière solution est de plus en plus appréciée et utilisée car il n'y a pas de contraintes théoriques quant à la complexité du système et du niveau de détail que l'on souhaite modéliser. En particulier, tandis que les méthodes mathématiques ne fournissent que des valeurs moyennes pour les critères de performance, la simulation permet en plus d'obtenir des estimations des distributions de ces derniers. De plus, celle-ci autorise l'étude du comportement d'un système non seulement à l'état stationnaire, mais aussi dans des états transitoires. Pour effectuer une simulation, on a recours à un ordinateur pour évaluer numériquement le modèle du système physique que l'on considère.

2.2.2 Simulation parallèle à événements discrets

Les modèles de réseaux de communication sont essentiellement des systèmes asynchrones car les paquets sont envoyés d'un élément à un autre sans aucune relation avec une horloge globale. Par conséquent, les techniques de simulation synchrone ou par incrément fixe ne pourront pas représenter fidèlement le comportement temporel du système. De plus, de telles approches sont difficilement réalisables dans un contexte distribué sans la connaissance précise de l'état global du système. Nous nous intéressons donc aux approches asynchrones et en particulier celles dites discrètes où les changements apportés au système se produisent à des points discrets dans le temps. Dans ce cas, le modèle va évoluer par bonds variables dans le temps selon la date d'occurence des événements.

Les simulations sont supposées être exécutées sur une architecture parallèle ou distribuée composée de n processeurs. La parallélisation consiste alors à diviser

le système global en un ensemble de sous-systèmes disjoints, chacun d'entre eux étant simulé par un processus logique (PL) s'exécutant sur un processeur physique $p_i, 1 \leq i \leq n$. Dans ce cas, les interactions qui peuvent exister dans le système réel sont représentées par des échanges de messages estampillés par une horloge locale du PL source. Les messages sont reçus par le PL destinataire et placés dans sa liste locale d'événements (*Future Event List*, FEV) en attendant d'être traduits en événements et exécutés. L'exécution d'un événement consiste en une éventuelle modification de l'état du PL et une éventuelle libération de nouveaux messages vers d'autres PLs. D'autres méthodes de parallélisation existent mais leur champ d'application est relativement limité [33]. Celle à laquelle nous nous intéressons offre de manière générale un niveau de parallélisme supérieur.

Cette distribution des horloges et des listes d'événements pose des problèmes de respect de la causalité. Inexistants dans le cas séquentiel, ces problèmes apparaissent maintenant car les différents PLs peuvent évoluer indépendamment et se trouver à des instants de simulation différents. En particulier, une *faute temporelle* se produit lorsqu'un message arrive à un PL avec une estampille inférieure à l'horloge locale du PL. Intuitivement ce cas représente une incohérence puisque le message appartient au passé. C'est pourquoi des mécanismes explicites de synchronisation sont nécessaires pour garantir l'intégrité de la simulation; ceux-ci peuvent être classés en deux catégories: conservateur et optimiste.

2.2.3 Les protocoles de synchronisation

La première méthode proposée fut l'approche dite conservatrice [14]. Elle évite les fautes temporelles en autorisant un PL p à exécuter un événement que lorsqu'il peut être déterminé avec certitude que son exécution ne pourra pas produire de fautes temporelles dans le futur. De ce fait, chaque PL i communicant avec p se voit affecter une file d'événements et l'algorithme consiste à sélectionner la file possédant l'événement d'estampille minimale. Si cette file est vide, un blocage se produit. Pour éviter les situations d'inter-blocage, des *null-messages* n'ayant aucune signification du point de vue du modèle peuvent être utilisés pour faire avancer artificiellement le temps de simulation. La réception d'un *null-message* d'estampille T sur un lien informe le récepteur qu'aucun message d'estampille inférieure à T ne pourra être reçu sur ce lien. Une autre solution consiste à laisser la simulation se bloquer pour entreprendre alors une phase de déblocage [15, 16]. Dans les deux cas, une bonne capacité à "prédire" l'avenir (*lookahead*) est nécessaire pour propager rapidement les valeurs d'horloge et ainsi obtenir de bonnes performances.

D'un fonctionnement totalement différent, les approches optimistes, dont le plus connu des réprésentants est *Time Warp* [48], ne font aucune tentative pour éviter les fautes temporelles. Ils instaurent par contre un contrôle réactif qui consiste en un mécanisme de retour-arrière (*rollback*) leur permettant de corriger la trajectoire de la simulation lorsqu'une faute survient. Les inconvénients majeurs de ces approches optimistes sont d'une part, la nécessité de sauvegarder périodiquement l'état du PL pour pouvoir le restaurer et d'autre part, l'utilisation d'anti-messages pour corriger

la propagation des erreurs de trajectoire. De manière générale, les approches optimistes sont beaucoup plus complexes à mettre en œuvre que leurs homologues conservateurs, mais peuvent d'un autre côté exploiter de manière transparente le parallélisme présent dans le modèle.

2.2.4 Synthèse sur les différentes approches et le simulateur CSAM développé

Les modèles de réseaux de communication présentent un fort degré de synchronisation par envoi de messages. Combinés à une granularité la plus souvent faible (car simulation au niveau du paquet), ces modèles ont tendance à produire un grand nombre de fautes temporelles avec une approche optimiste. De plus, les optimisations existantes pour ces approches sont bien souvent inutiles et inefficaces avec ces modèles de réseaux (voir mon document de thèse [87]). Des précautions doivent également être prises pour gérer efficacement la sauvegarde des états dans les approches optimistes car la plupart des modèles réalistes de réseaux impliquent une taille d'état relativement grande. Maintenant, avec une approche conservatrice, la quantité de *lookahead* (capacité de prédire ce qui se passera) dans ces modèles de réseaux sera le plus souvent réduite aux délais des liens lorsque l'application induit un ordonnancement non FCFS (les algorithmes d'ordonnancement pour la QoS par exemple). Cependant cette quantité de *lookahead*, même réduite aux délais des liens, est bonne car ceux-ci sont nettement plus importants que les délais représentés par les temps de traversée dans les routeurs. Un des problèmes avec une méthode conservatrice est la présence possible d'un grand nombre de boucles dans les modèles. Des schémas de routage réalistes et un mode connecté peuvent introduire un problème de répartition des messages entre un chemin et son homologue dans l'autre direction. Ce phénomène provoque une augmentation du trafic de *null-message* qui pénalise le simulateur. Ce problème peut être amoindri dans une certaine mesure mais nécessite souvent de prendre en compte de manière détaillée les particularités de l'application et de les ajouter dans le code du simulateur.

Cependant mon sentiment général penche plus vers l'utilisation de méthodes conservatrices pour la simulation des modèles de réseaux de communication. La forte synchronisation au niveau message et les conséquences qui en découlent pour une approche optimiste m'incite à penser qu'il vaut mieux "prévenir que guérir".

CSAM est un outil que j'ai développé durant ma thèse pour simuler des modèles de réseaux de communication. Initialement conçu pour les réseaux ATM, il peut être utilisé pour simuler des modèles de réseaux de communication en général. Le noyau de simulation est écrit en C++ et un certain nombre d'objets de base (commutateurs, sources de trafic, algorithme de routage, contrôle d'admission...) ont été modélisés. La construction de nouveaux modèles peut se faire en créant des objets dérivés et en modifiant leur comportement. L'algorithme de synchronisation est uniquement conservateur et offre une transparence totale à l'utilisateur car le simulateur n'exploite que le *lookahead* des liens de communication (temps de propagation). La topologie et les paramètres associés aux objets du modèle (commutateurs, sources

de trafic etc.) sont ensuite décrits dans un fichier de configuration lu à l'initialisation de la simulation. Initialement développé pour être exécuté sur une machine CM-5 avec PVM (*Parallel Virtual Machine*, une librairie d'envoi de messages), CSAM a été ensuite porté sur CRAY T3E pour utiliser SHMEM et MPI (*Message Passing Protocol*, une autre librairie d'envoi de messages). Avec le support de MPI, CSAM peut être utilisé sur toute architecture supportant cette librairie de communication. Le support de MPI étant celui qui doit être préféré car il permet, par exemple, un portage facile vers les architectures à base de grappes de machines que nous allons aborder ci-après.

2.3 LES GRAPPES DE MACHINES: DÉFINITION ET ARCHITECTURE

On peut définir une grappe de machines comme étant un ensemble d'ordinateurs interconnectés par un réseau de communication. Ce qui différencie principalement une telle grappe d'une machine parallèle est le fait que chacun des éléments de la grappe est une entité pouvant être utilisée, si on le souhaite, de manière indépendante. Par exemple, les grappes à base de PCs (processeurs Intel par exemple) se différencient d'une machine parallèle multi-processeurs à base des mêmes processeurs par le fait que chaque PC a la possibilité d'être utilisé en dehors du contexte de la grappe, ce qui n'est pas le cas si l'on prend un processeur quelconque de la machine parallèle.

Dans la pratique, il est effectivement possible de construire des grappes à partir d'ordinateurs standards achetés dans le commerce de masse (des PCs en tour par exemple) et de les mettre sur des étagères. Il est aussi possible de gagner en place en prenant des machines en rack et de les monter dans une armoire: l'intégration est alors plus avancée.

Les avantages des grappes de machines sur les machines parallèles sont nombreux. Parmi ceux-ci, citons le coût moindre par processeur et la possibilité de mettre à jour sa grappe de manière incrémentale. La puissance en terme de capacité de calcul brute est généralement bonne, voire très bonne, ce qui justifie l'utilisation de telles architectures.

> Une liste appelée `Top500` recense les 500 plus puissantes machines du monde (voir `www.top500.org`). En janvier 2003, la machine la plus puissante était la machine *Earth Simulator* construite par NEC et installée au Japon. Cette machine affichait une performance mesurée de 35860 GFlops avec le test Linpack et comportait 5120 processeurs. La machine en seconde position était l'ASCI Q[1] avec 4096 processeurs (HP AlphaServer) pour 7.7 TFlops.
>
> Il y avait en janvier 2003 "93 clusters dans le Top500 dont 55 basés sur

[1] Les machines ASCI Q, White, Red, Blue peuvent être considérées comme étant des grappes suivant la définition donnée précédemment.

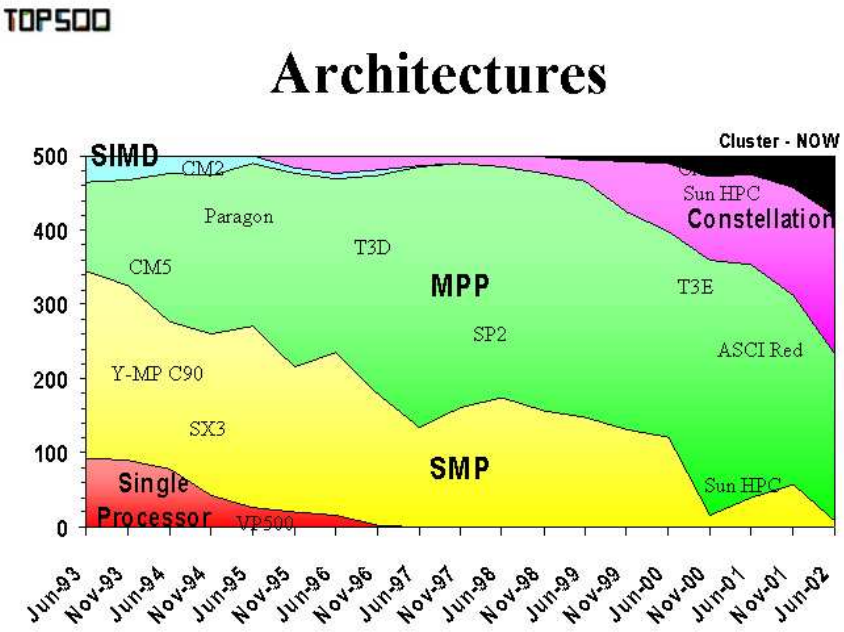

Figure 2.1 Répartition en nombre des machines selon leur type dans le Top500.

des processeurs Intel" et "Pour la première fois, deux clusters de PC sont dans le Top10; un cluster construit par Linux Networks et Quadrics pour le LLNL et un cluster construit par HPTi avec Myrinet pour le NNOA" (source BI-ORAP, numéro 34-Jan 2003. Linux Cluster Xeon 2.4 GHz, 2304 processeurs et 5.7 TFlops pour le premier; et Dual Xeon 2.2 GHz, 1536 processeurs et 3.3 TFlops pour le deuxième).

Pour voir l'évolution de la puissance, il faut se rappeler qu'en juin 2002 la première grappe de machines à base de PC dans la liste se trouvait à la 35ème position pour une performance de 0.825 TFlops avec 512 processeur AMD HELICS 1.4 GHz interconnectés avec un réseau Myrinet. La figure 2.1 montre la répartition en nombre des systèmes selon leur type dans la liste en juin 2002. Il y avait 80 grappes de machines dans la liste si l'on regroupe les grappes à base d'Intel, d'AMD, de SUN et d'Alpha (49 si on ne considère que les 2 premiers types de processeur qui sont vraiment des processeurs issus du marché de masse).

Les performances exprimées en GFlops dépendent bien sûr du programme test utilisé (LU, LINPACK...) et du profil des communications et des calculs dans ces tests. Pour les grappes, la performance du réseau d'interconnexion est un critère très important pour l'obtention de performance au niveau applicatif. Cela est bien sûr aussi le cas sur les machines parallèles, mais le problème est ici plus récurrent sur les grappes puisque le réseau d'interconnexion est lui aussi généralement issu du marché

de masse (bien que ce marché ne soit pas aussi large que celui des machines). Nous verrons rapidement dans le paragraphe 2.3.1 les différentes technologies disponibles.

Devant la flexibilité offerte par les grappes, il est tout à fait envisageable, au vu des prix, de construire de petites grappes dédiées par application type. En utilisant des nœuds de calcul bi-processeur, il est possible de construire une grappe de calcul affichant un rapport performance/prix imbattable! Nous verrons par la suite comment ces nœuds bi-processeurs peuvent être utilisés dans le cas de la simulation parallèle.

2.3.1 L'architecture matérielle

Les nœuds de calcul dans les grappes de machines sont généralement des PCs à base de processeurs Intel ou AMD avec aussi un marché pour des grappes à base de machines SUN ou Alpha. En ce qui concerne le réseau d'interconnexion, il est possible de constituer des grappes à partir de technologies réseaux très bon marché telles que FastEthernet (100MBits/s). Cependant, les faibles performances de ce réseau orientent plutôt de telles grappes vers une utilisation de type ferme de calcul avec des applications à très gros grain de calcul (rapport calcul/communication très grand comme par exemple l'application `seti@home`).

Pour construire de véritables grappes performantes, le réseau d'interconnexion doit être beaucoup plus rapide et offrir de faibles latences. Il existe plusieurs produits: Myrinet, SCI (Scalable Coherent Interface, GigaEthernet, FiberChannel, QsNet (Quarics), Atoll... Les lecteurs pourront se référer à la thèse de R. Westrelin [113] (sous la direction du Pr. B. Tourancheau) pour un état de l'art sur ces réseaux d'interconnexion.

Lorsque je suis arrivé sur Lyon en 1998 dans l'équipe du Pr. Tourancheau, les grappes de machines commençaient à être étudiées en France et une grande expérience sur les produits Myrinet [7] (tout nouveau produit de la société Myricom à cette date) avait été acquise par l'équipe, principalement dans le contexte de la thèse de L. Prylli [93]. Nos travaux liés aux grappes Myrinet ont été publiés à plusieurs reprises [38, 39][2].

Les produits Myrinet sont maintenant en large diffusion et présents dans un grand nombre de grappes de calcul très performantes: la première grappe de PC dans le Top500 par exemple (140 des 500 machines du Top500 utilisent Myrinet). L'architecture d'une grappe à base de produits Myrinet consiste en des machines avec une carte d'interface Myrinet reliées à un commutateur Myrinet. Le commutateur est de type crossbar avec une architecture très simple mais efficace qui supporte

[2] P. Geoffray, L. Lefèvre, C. Pham, L. Prylli, O. Reymann, B. Tourancheau, R. Westrelin, "High-Speed LANs: New Environments for Parallel and Distributed Applications", *Proceedings of EuroPar'99, LNCS 1685, August 31-September 3, 1999, Toulouse, France, pp633-642.*

P. Geoffray, C. Pham, L. Prylli, B. Tourancheau, R. Westrelin "Protocols and Software for Exploiting Myrinet Clusters", *Proceedings of the International Conference on Computational Science (ICCS 2001), LNCS 2073&2074, Technical Session on Global Computing - Internals and Usage (CCIU 2001), May 28-30th, 2001, San Francisco, CA, USA, pp233-242*

dans les dernières versions jusqu'à 128 ports à 2Gbits/s chacun[3]. En ce qui concerne la carte d'interface Myrinet, c'est une carte embarquant un processeur RISC, appellé LANai, de la mémoire et des composants DMA. La particularité de cette carte, et c'était précurseur à l'époque, était que le processeur pouvait être entièrement reprogrammé. C'est cette possibilité qui rend ces cartes très polyvalentes et performantes, et qui permet de nombreuses optimisations logicielles qui seront présentées dans le paragraphe suivant. La figure 2.2 représente une carte Myrinet avec le processeur LANai en zone centrale.

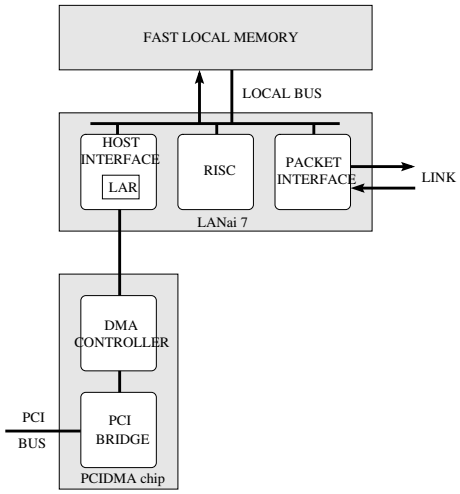

Figure 2.2 Architecture d'une carte Myrinet (source: R. Westrelin).

2.3.2 L'architecture logicielle

Il est courant de constater que les progrès réalisés dans le domaine du logiciel sont beaucoup moins rapides que ceux réalisés dans le matériel. Les raisons à cela sont surtout d'ordre conceptuels car il est extrêment difficile de concevoir des logiciels performants. Ces problèmes se retrouvent sur les grappes de machines pour les bibliothèques de communications entre les nœuds de la grappe sous les formes suivantes: faibles débits par rapport à la capacité de l'interface, trop grande latence, surcharge anormale du processeur hôte de la machine, perte de paquets...

Si l'on réutilise des bibliothèques existantes et conçues pour des réseaux IP (la suite TCP/IP par exemple), celles-ci ne pourront généralement pas fournir à l'utilisateur final toutes les performances disponibles (débit et latence dans la plupart des cas). Il y a plusieurs raisons à cela: (*i*) les protocoles utilisent des algorithmes qui ne sont pas adaptés aux haut-débit (fenêtre d'anticipation trop petite, numérotation limitée, contrôles d'erreur trop fréquents, routage coûteux,...), (*ii*) l'implémentation n'est pas performante sur le haut-débit (trop de recopies mémoires, trop de traversées de couches, trop d'appels systèmes, trop d'interruptions

[3]L'INRIA Rhône-Alpes a reçu en janvier 2003 un cluster de 208 processeurs Itanium-2 interconnectés en Myrinet avec un commutateur 128 ports.

matérielles qui sont mal gérées et qui surchargent le processeur hôte...) et, (*iii*) impossibilité de profiter des nouvelles fonctionnalités des cartes d'interfaces récentes. La figure 2.3 montre de manière imagée les résultats d'une mauvaise conception ou d'une mauvaise implémentation sur les performances finales.

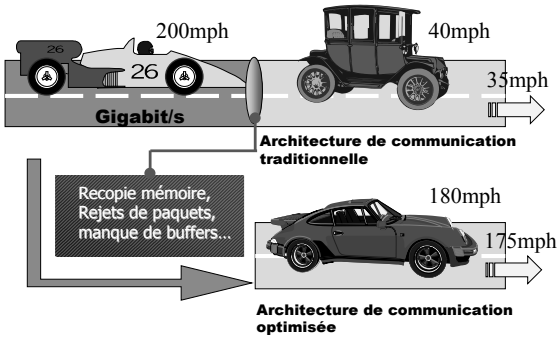

Figure 2.3 Coûts induits par les couches de communication.

Ce sont ces constats qui sont à l'origine du développement de la bibliothèque BIP (Basic Interface for Parallelism) [94] par L. Prylli sur les produits Myrinet. BIP s'est lui-même basé sur d'autres propositions antérieures telles que Active Messages [25], Fast Messages [82] et U-Net [26]. Tous ces systèmes ont en commun la volonté de raccourcir le chemin des données, en court-circuitant le système d'exploitation, et d'éviter le plus possible les recopies mémoires (voir figure 2.4).

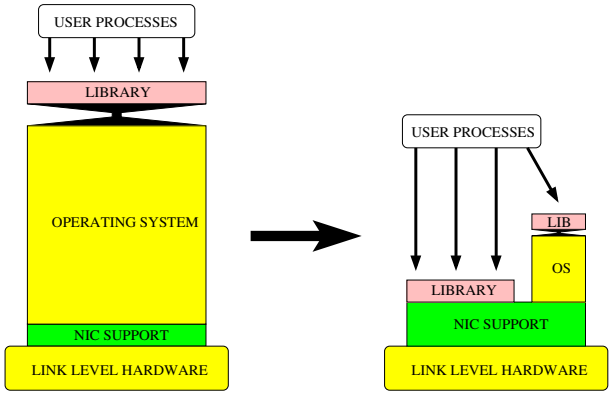

Figure 2.4 Nouvelles architectures pour les bibliothèques récentes.

BIP n'a pas été conçu pour l'utilisateur final. Pour celui-ci, une interface de programation standardisée comme MPI permet une plus grande portabilité. C'est dans ce but que MPI-BIP a été proposé. Grâce aux travaux de L. Prylli, R. Westrelin et P. Geoffray, BIP et MPI-BIP sont des implémentations performantes qui offrent de très bons débits et de très faibles latences comme illustré par la figure 2.5.

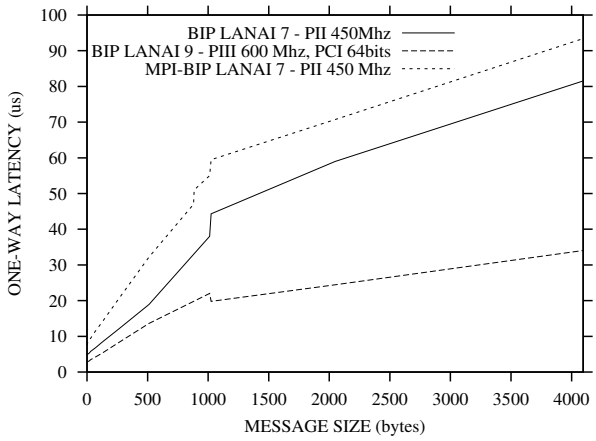

Figure 2.5 Latences (aller-retour divisé par 2) avec BIP et MPI-BIP.

2.4 SIMULATION PARALLÈLE ET GRAPPES DE MACHINES: PROBLÈMES ET SOLUTIONS

Après avoir travaillé sur des problématiques de synchronisation entre processus logiques [88, 89, 90] la problématique sur les grappes de machines est beaucoup plus dirigée vers l'obtention de la latence la plus faible dans les communication inter-processeur. De ce fait, mes recherches dans ce domaine ont plutôt porté sur l'utilisation optimale des couches de communication BIP et MPI-BIP décrites précédemment. CSAM a été testé avec succès sur des grappes de PCs interconnectés par du Myrinet et avec la librairie MPI-BIP [86][4]. Le noyau de simulation comporte des optimisations pour réduire le coût de gestion des messages externe et pour gérer des canaux de communication virtuels. Nous allons décrire dans la suite de cette section le cycle de vie des objets de simulation et les contraintes que cela pose pour le simulateur parallèle. Nous présenterons les verrous pour l'obtention de gains sur les grappes de PCs, en comparant les résultats avec ceux issus d'expérimentations précédemment effectuées sur des machines parallèles.

2.4.1 Le cycle de vie des objets de simulation

Le cycle de vie des objets de simulation dans le simulateur parallèle CSAM consiste essentiellement à récupérer des messages, à les traiter puis à produire des messages en sortie. Les messages qui arrivent et qui doivent être traités sont généralement mis à leur réception dans une liste d'événements et sont triés par ordre croissant des estampilles. Les messages qui sont produits et qui sont à destination d'un autre processeur sur une autre machine sont envoyés sur le réseau d'interconnexion, sinon ils sont directement insérés dans la liste des événements locaux.

Le destinataire d'un message est identifié par le couple (*node*, *ic*) où *ic* est le

[4]C. Pham, "High Performance Clusters: A Promising Environment for Parallel Discrete Event Simulation", *Proceedings of the PDPTA'99, June 28-July 1, 1999, Las Vegas, USA, Vol. III pp1299-1304.*

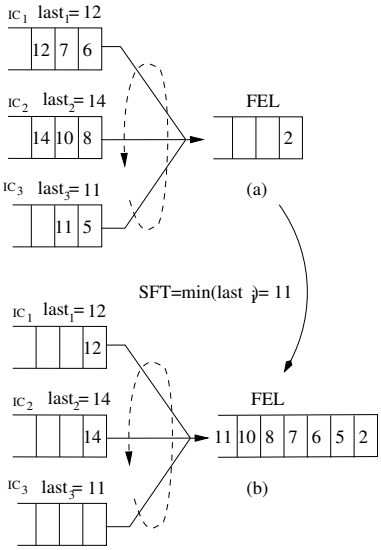

Figure 2.6 Fonctionnement des canaux virtuels et calcul de l'horizon d'exécution, (a) avant et (b) après récupération des messages externes.

numéro du canal de communication qui relie le destinataire à l'émetteur (rappel: l'approche conservatrice nécessite un canal de communication entre chaque entité). Les messages externes sont reçus par le système de communication (MPI, BIP...) et placés dans des buffers mémoire. Ils seront périodiquement récupérés pour être insérés dans la liste locale (FEL) et devenir des événements éligibles pour l'exécution. La valeur de *ic*, associée à l'estampille d'un message reçu, permet au simulateur de calculer à la volée le minimum des estampilles sur les derniers messages reçus et cela sur l'ensemble des canaux (ce qui revient à calculer le minimum des valeurs maximales par canal). Cette valeur représente l'horizon d'exécution (*Safe Future Timestamp*) à ne pas dépasser si l'on veut garantir une intégrité temporelle stricte (comme c'est le cas dans les approches conservatrices). La figure 2.6 illustre ce fonctionnement. Lorsqu'un processus logique n'a plus de messages à exécuter ou que son horloge logique a atteint cet horizon, il envoie un null-message et récupère tous les messages externes en attente dans les buffers de communication.

2.4.2 Contraintes du point de vue de la simulation parallèle

Du point de vue du simulateur parallèle, les modèles de réseaux de communication présentent une très petite granularité de calcul (temps d'exécution d'un événement très faible) et nécessitent l'envoi d'un très grand nombre de messages entre les processeurs (contrairement aux applications traditionnelles de calculs scientifiques où le grain est plus grand). Il est par exemple courant d'envoyer plusieurs millions de messages par minute. Ces messages sont soient des messages liés aux modèles (pour représenter les paquets, les cellules...), soient des messages liés au simulateur

(null-messages...). Par conséquent, l'impact de la latence des communications sur les performances est très important comme nous allons le voir par la suite.

2.4.3 Les principaux verrous pour obtenir des gains sur les grappes

Expérimentations sur Cray T3E

J'ai resumé dans la table 2.1 les résultats d'une série de simulations effectuées pendant ma thèse (jan 1997) sur une machine parallèle T3E. Cette machine était fournie par l'*Institut du Développement et des Ressources en Informatique Scientifique* (IDRIS) et comporte 256 processeurs (DEC Alpha EV5 300Mhz et 128 Mo de RAM) interconnectés en tore 3D par un réseau propriétaire. Les bibliothèques de communication disponibles sont SHMEM (SHared MEMory) et MPI qui affichent respectivement des latences de $7\mu s$ et $13\mu s$. J'ai simulé 0.31s de fonctionnement d'un modèle de réseau comportant 78 commutateurs ATM. Plus de 50 millions d'événements ont été simulés!

Tableau 2.1 Cray T3E avec SHMEM et MPI.

np.	*temps SHMEM (s)*	*gain SHMEM*
1	488	-
4	131	3.72
8	72	6.77
16	46	10.60
32	33	14.78
64	31	15.74
np.	*temps MPI (s)*	*gain MPI*
1	488	-
4	253	2.07
8	133	3.66
16	82	5.95
32	59	8.27
64	55	8.87

Sur le Cray T3E, la bibliothèque SHMEM montre de bons gains qui sont presque linéaires sur 4 et 8 processeurs. Les gains obtenus avec MPI sont beaucoup plus faibles à cause de la plus grande latence, principalement due à la traversée des couches protocolaires de communication. Le gain maximum est de 15.74 avec SHMEM sur 64 processeurs, mais peu d'améliorations peuvent être obtenues avec plus de 32 processeurs, principalement à cause des coûts de synchronisation. Le problème le plus difficile à résoudre avec autant de processeurs est celui de la répartition des charges. Si le gain ne diminue pas plus lorsque le nombre de processeurs augmente, c'est grâce à la faible latence sur le Cray. Sur des architectures moins per-

formantes, il serait plus raisonnable d'exécuter les simulations parallèles sur moins de processeurs et d'optimiser la répartition des calculs.

Expérimentations sur une grappe Myrinet

Je vais maintenant présenter les résultats d'une série de simulations effectuées sur une grappe expérimentale comportant 12 nœuds interconnectés par 3 commutateurs 8-port Myrinet. Chaque nœud est un Intel Pentium Pro 200 MHz avec 64Mo de RAM et un chipset 440FX. L'interface réseau est une carte Myrinet avec un LANai 4.1 et 256Ko de memory. La plate-forme de test a été fournie par le LHPC (Laboratoire pour les Hautes Performances en Calcul) qui est issu d'une coopération entre l'ENS-Lyon et Matra Système Information. Avec cette configuration en 1998, la latence pour envoyer un message de 70 octets (taille d'un message dans notre application) est d'environ $10\mu s$ avec BIP et $17\mu s$ avec MPI/BIP. La table 2.2 résume les gains obtenus avec MPI/BIP et montre le nombre de messages qui doivent être envoyés sur le réseau. Seulement 8 nœuds ont pu être utilisés sur les 12 disponibles au moment des expérimentations (décembre 98).

Tableau 2.2 Myrinet avec MPI/BIP.

#proc	*temps*	*gain*	*# de msg distant*
1	333	-	0
2	228	1.46	710,905
4	141	2.36	2,304,835
6	98	3.39	2,845,867
8	104	3.20	3,418,919

Si l'on compare les résultats obtenus sur la grappe Myrinet à ceux obtenus sur le Cray, nous pouvons voir d'une part que la version séquentielle va plus vite sur un Pentium Pro 200MHz que sur un DEC Alpha EV5 300MHz. Nous pouvons déjà voir l'avantage d'utiliser des plate-formes standards qui permettent d'avoir plus rapidement les processeurs les plus rapides. De nos jours les processeurs sont cadencés à plus de 2 GHz! Les performances affichées avec MPI/BIP sur Myrinet sont très encourageantes et le coût des envois de messages peut très certainement être amélioré. Une possibilité d'amélioration peut être d'utiliser BIP en natif plutôt que MPI/BIP (comme utiliser SHMEM en natif sur Cray). Cependant, le développement d'un support de BIP en natif aurait nécéssité beaucoup de temps (implémentation d'un contrôle de flux entre autre), j'ai préféré étudier les moyens d'améliorer les performances de MPI/BIP sur une architecture de grappes de PCs. De plus, l'interface MPI étant la plus portable, c'est la solution qui m'a semblée la plus pertinente sur le long terme. Je présente dans les 2 sous-sections qui suivent les travaux effectués pour optimiser l'envoi des messages et prendre en compte les architectures multi-processeurs, et plus spécialement les bi-processeurs.

2.4.4 Aggréger les messages pour réduire les coûts

Une des méthodes les plus couramment utilisées pour réduire les coûts d'envoi et de réception des messages consiste à aggréger à l'émission plusieurs petits messages en un nombre beaucoup plus réduit de long messages. La réception en est aussi optimisée car un seul paquet réseau est reçu sur la carte d'interface (donc moins d'interruptions matérielles par exemple). Cette méthode peut-être appliquée dans le cas du simulateur CSAM car celui-ci fonctionne en alternant période de traitement (traitement des événements locaux) et récupération des messages externes lorsque l'horizon d'exécution a été atteint (voir section 2.4.1). De cette manière, l'aggrégation des messages à l'émission, qui retarde l'envoi des messages, n'a que peu d'incidences néfastes puisque les messages, s'ils arrivaient plus tôt, devraient néanmoins attendre chez le récepteur la fin d'une période de traitement.

Avec un étudiant en master (étudiant Carsten Albrecht) de l'université de Lübeck (Allemage) en stage sous ma direction, nous avons modifié le noyau du simulateur CSAM pour supporter l'aggrégation des messages. Une stratégie simple, reposant sur une taille maximale des aggrégats, a été implémentée (des mécanismes supplémentaires permettent néanmoins d'éviter les blocages). Les premiers résultats n'ont pas été très concluants car les performances n'étaient pas bonnes. Nous avons cherché à améliorer ces résultats en décortiquant méthodiquement les performances des différentes composantes du système de communication et nous avons aussi mesuré de manière précise les coûts et les gains de l'aggrégation. Ces travaux ont été publiés dans l'article #1[5] qui peut être trouvé en annexe à la fin de ce document.

Performances des couches basses

La figure 2.7 représente la latence de MPI-BIP et de BIP à l'époque de nos expérimentations. Comme on peut le voir, cette latence n'est pas constante et présente même des sauts à plusieurs endroits.

Si l'on regarde la courbe de MPI-BIP, le premier saut correspond au passage par MPI d'une politique d'envoi asynchrone (taille < 156) à une politique d'envoi synchrone avec attente d'un accusé de réception de l'entête du message avant de transmettre les données (politique à rendez-vous). Le deuxième saut est dû à la couche BIP qui passe d'une stratégie de petit message (avec une copie) à une stratégie de long message (sans copie, mais avec un coût plus gran pour l'initialisation des moteurs DMA).

Ces premières mesures nous permettent de tracer une courbe de gain théorique si l'aggrégation est effectuée. La figure 2.8a montre les pourcentages théoriques des gains en fonction du nombre de messages (de 42 octets, taille d'un message dans le simulateur CSAM) aggrégés.

Cependant, ces seules données ne suffisent pas car il y a d'autres paramètres qui entrent en jeu. Après des tests supplémentaires, nous avons représenté dans

[5] C. Pham and C. Albrecht. "Tuning Message Aggregation on High Performance Clusters for Efficient Parallel Simulations". *Parallel Processing Letters, 9(4):521–532, 1999.*

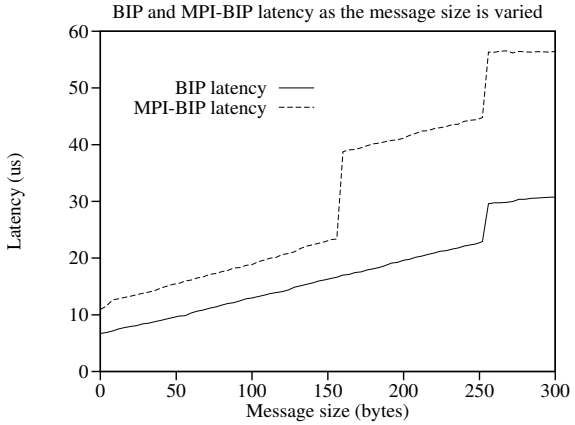

Figure 2.7 Latence de MPI-BIP et BIP en fonction de la taille des messages.

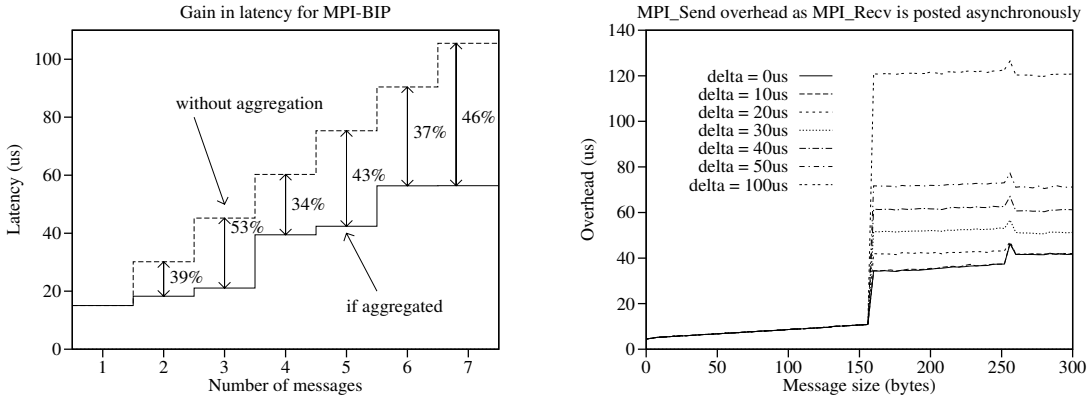

Figure 2.8 (a) Gain en latence, (b) coût d'un `MPI_Send`

la figure 2.8b le coût (en temps) d'un appel à la primitive `MPI_Send` lorsque du coté du récepteur l'appel au `MPI_Recv` correspondant est émis avec un retard δ. Ce comportement se retrouve dans le fonctionnement du simulateur CSAM à cause de l'alternance des phases de traitement et des phases de récupération de messages (là où les appels à `MPI_Recv` sont effectués). Un grand nombre d'applications parallèles et distribuées ont également ce comportement. Comme on peut le voir sur cette figure, pour des tailles de messages supérieures à 156 octets, plus les appels sont "désynchronisés", et plus l'appel à `MPI_Send` bloque longtemps (ce qui est préjudiciable à l'application).

Ces différents tests nous ont permis de mieux comprendre le comportement des couches basses de communication et d'optimiser la stratégie d'aggrégation. Dans ce cas, il est préférable de rester en dessous du seuil de 156 octets pour les aggrégats, ce qui permet de benéficier non seulement d'une latence plus faible, mais également d'un coût moindre dans les appels à `MPI_Send` en conservant une grande facilité de programmation. En effet, le coût du `MPI_Send` peut aussi être réduit en utilisant les versions asynchrones de l'appel (`isend`, `bsend`) mais cela complique de manière

significative le code à écrire. Après ces optimisations, les performances ont été nettement améliorées et les gains obtenus sont illustrés par la figure 2.9. Ensuite, suivant les modifications apportées à BIP ou à MPI-BIP, les paramètres d'aggrégation sont configurés en conséquence.

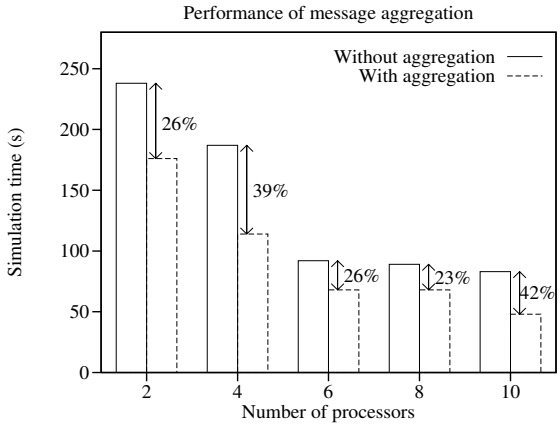

Figure 2.9 Performance du simulateur.

2.4.5 Prise en compte des machines multi-processeurs

Une des nouvelles directions de recherche était celle posée par les machines multi-processeurs, et plus particulièrement les bi-processeurs à l'époque de l'étude. En effet, ces architectures bénéficient d'un rapport performance/coût très intéressant, surtout grâce au fait que la carte d'interface, souvent coûteuse pour les réseaux spécialisés tels que Myrinet, est partagée.

La suite BIP s'est enrichie d'une composante SMP (noté BIP-SMP) grâce aux efforts de P. Geoffray, en thèse sous la direction de B. Tourancheau. L'architecture de la composante BIP-SMP est décrite dans la figure 2.10 où l'on peut voir qu'elle comporte 2 couches: une permettant un accès transparent à BIP ou à BIP-SMP, et s'insérant en dessous de l'API de BIP et une autre, au même niveau que BIP, qui exploite de manière spécifique la mémoire partagée. Plusieurs mécanismes très efficaces ont été implémentés dans BIP-SMP pour offrir des latences très faibles entre 2 processeurs de la même machine et des performances similaires à BIP entre 2 machines distantes. Ces travaux de P. Geoffray sont décrits dans [40] et les performances résumées dans la table 2.3.

Dans ce contexte où les performances sont asymmétriques, il est possible d'optimiser l'aggrégation des messages en prenant en compte la localité des messages. Des travaux effectués avec P. Geoffray m'ont permis de modifier le noyau de simulation en conséquence. Le tableau 2.4 résume les tests que j'ai effectués sur 4 machines mono-processeur et 2 machines bi-processeurs. `Aggr. x-y` indique une aggrégation entre 2 processeurs distants de x octets et une aggrégation interne de y octets. Lorsque l'on augmente le nombre de machines utilisées (et donc le nombre de processeurs), cette aggrégation asymmétrique n'apporte pas d'améliorations supplémentaires car

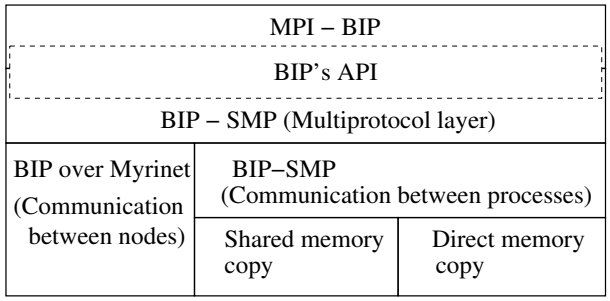

Figure 2.10 Architecture de BIP-SMP.

	BIP-SMP	MPI-BIP/BIP-SMP
Latence intra-nœud (Shared Memory)	1.8 μs	3.3 μs
Latence inter-nœud (Myrinet)	5.7 μs	7.6 μs
Débit intra-nœud (Shared Memory)	160 MBytes/s	150 MBytes/s
Débit inter-nœud (Myrinet)	126 MBytes/s	107 MBytes/s

Tableau 2.3 Performances des communications point-à-point.

dans ce cas le coût de la distribution et les fortes synchronisations implicites qui en découlent masquent les gains d'une aggrégation asymmétrique. Cependant, le gain absolu reste plus important; par exemple des gains supérieurs à 4.5 ont pu être obtenus sur une grappe de 8 nœuds. Ces travaux ont été publiés dans [37][6] puis une version étendue a été publiée dans l'article #2[7] qui peut être trouvé en annexe à la fin de ce document.

2.4.6 Déport de fonctionnalités sur la carte réseau

Les cartes Myrinet sont programmables et cette fonctionnalité a été largement employée pour la suite logicielle BIP. J'ai encadré de février 2001 à juillet 2001 Eric Lemoine en stage de DEA pour étudier la possibilité de déporter sur la carte des fonctions de haut niveau liées à la simulation parallèle (génération/suppression de null-messages, calcul des horizons d'exécution pour la synchronisation...). Un travail préliminaire a consisté à évaluer le gain que l'on pouvait obtenir en fonction de la quantité de traitement (en terme de temps CPU) à déporter sur la carte [57]. Il s'avère ensuite que pour réaliser la génération/suppression de null-messages ou le calcul des horizons d'exécution, il est nécessaire de partager des structures de données entre le processeur hôte et le LANai sur la carte de communication. Or, un tel partage de données est très coûteux et les gains sont alors négligeables

[6] P. Geoffray, C. Pham, B. Tourancheau, "Exploiting Clusters of Shared Memory Multiprocessors with BIP-SMP: the Parallel Simulation Application", *Proceedings of the Workshop on Cluster-based Computing, help in conjunction with the 1999 ACM International Conference on Supercomputing, June 20-25, 1999, Rhodes, Greece.*

[7] P. Geoffray, C. Pham, and B. Tourancheau. "A Software Suite for High-Performance Communications on Clusters of SMPs". *Journal of Cluster Computing, Vol 5(4), pp353-363, October 2002.*

	4 ext.		2x2 int.	
stratégie	temps	gain	temps	gain
No Aggr.	92	1.31	73	1.65
Aggr. 156-156	71	1.70	64	1.86
Aggr. 256-256	63	1.92	63	1.92
Aggr. 512-512	62	1.95	62	1.95
Aggr. 1024-1024	62	1.95	63	1.92
Aggr. 256-156	_	_	59	2.05
Aggr. 512-156	_	_	**56**	**2.16**
Aggr. 1024-156	_	_	59	2.05
Aggr. 512-256	_	_	61	1.98

Tableau 2.4 4 nœuds externes vs 2 x 2 internes.

pour une complexité de développement très grande. J'ai abandonné cette voie de recherche en ce qui concerne la simulation parallèle, mais l'utilisation de cartes réseau programmables dans le contexte des sous-systèmes de communication réseau sera traitée dans le chapitre 4.

2.5 CONCLUSIONS DU CHAPITRE ET PERSPECTIVES

La communauté de recherche en simulation parallèle a produit ces 20 dernières années des algorithmes et des mécanismes pour la synchronisation des processus, pour la sauvegarde des états dans les approches optimistes etc. Le défi en cette fin de siècle n'est plus le même, et les travaux sur la synchronisation sont marginaux. Je penses que la bonne évolution viendra des outils et des applications. On assiste de la part des équipes de recherche à plus de prototypes d'environnements intégrés, plus d'outils, plus de réflexions axées sur des applications particulières. L'équipe dans laquelle j'étais à UCLA s'oriente par exemple vers le développement d'environnements puissants comme le produit Qualnet[8]. J'utilise ces outils (PARSEC, Qualnet) pour évaluer les protocoles de multicast sur une grande échelle.

Lors de mon arrivée à Lyon en octobre 1998, l'utilisation des grappes de machines commençait tout juste à se développer. Les travaux de B. Tourancheau et de ses étudiants en thèse ont été précurseurs sur de nombreux points. J'ai pu profiter de cet environnement très motivant pour poursuivre mes travaux de recherche. L'efficacité des grappes est désormais démontrée dans de nombreux domaines scientifiques et la simulation parallèle est l'un de ces domaines. La méthodologie que nous avons utilisée dans le cadre de grappes Myrinet avec la suite logicielle BIP est facilement généralisable à d'autres architectures ou suites logicielles (MPI-MPC...). Le point commun que l'on retrouve habituellement est le fait que les performances des communications sont très souvent sensibles à la taille des messages et à l'asynchronisme.

[8] `http://www.scalable-networks.com/`

Les machines bi-processeurs sont les machines qui offrent le rapport performance/prix le plus élevé. Les machines quadri-processeurs et plus sont bien plus chères. Une grappe idéale pour des simulations parallèles est à mon avis une grappe de 4 à 8 machine bi-processeurs, pas plus. Un volet de mes recherches concernant les grilles de calcul, je ne penses pas que la simulation parallèle y trouve un jour sa place. C'est plutôt le domaine pour les simulations distribuées comme HLA où l'interactivité et la coopération sont plus importants que la performance pure.

Ces travaux ont été les premiers que j'ai réalisé en tant que chercheur autonome après mon séjour post-doctoral aux Etats-Unis. La présence d'une équipe compétente et disponible m'a été d'une aide considérable pour mener à bien ces travaux.

MES PUBLICATIONS LIEES A CE THEME

Livres, chapitre de livres

[1] L. Lefevre, C. Pham, C. Seitz, B. Tourancheau (eds). *First Myrinet users' group*, INRIA, Septembre 2000, ISBN.

[2] C. Pham. La simulation parallèle. A paraître dans Traité IC2 Hermès. Performance des réseaux de télécommunications: les méthodes.

Journaux

[1] P. Geoffray, C. Pham, and B. Tourancheau. A software suite for high-performance communications on clusters of smps. *Journal of Cluster Computing*, 5(4):353–363, 2002.

[2] C. Pham and C. Albrecht. Tuning message aggregation on high performance clusters for efficient parallel simulations. *Parallel Processing Letters*, 9(4):521–532, 1999.

Conférences

[1] P. Geoffray, C. Pham, L. Prylli, B. Tourancheau, and R. Westrelin. Protocols and software for exploiting myrinet clusters. In *Proceedings of the International Conference on Computational Science (ICCS 2001)*, number 2073&2074 in LNCS, pages 233–242, San Francisco, CA, USA, May 2001. Springer-Verlag.

[2] C. Pham. Comparison of message aggregation strategies for parallel simulations on a high performance cluster. In *IEEE MASCOTS 2000*, pages 358–365, San Francisco, CA, USA, August 2000.

[3] P. Geoffray, C. D. Pham, and B. Tourancheau. Exploiting clusters of shared memory multiprocessors with bip-smp: the parallel simulation application. In N. P. Carter and S. S. Lumetta, editors, *ACM International Conference on Supercomputing (ICS'99)*, Workshop on Cluster-based Computing, Rhodes, Greece, June 1999.

[4] P. Geoffray, L. Lefèvre, C. Pham, L. Prylli, O. Reymann, B. Tourancheau, and R. Westrelin. High-speed LANs: New environments for parallel and distributed applications. In *ACM/IFIP EuroPar'99*, number 1685 in LNCS, pages 633–642, Toulouse, France, August 1999. Springer-Verlag.

[5] C. Pham and C. Albrecht. Optimizing message aggregation for parallel simulation on high performance clusters. In *Proceedings of the IEEE MASCOT'99 Conference*, pages 76–83, College Park, MD, USA, October 1999.

[6] C. Pham. High performance clusters: A promising environment for parallel discrete event simulation. In *PDPTA'99*, volume III, pages 1299–1304, Las Vegas, NV, USA, June 1999. CSREA Press.

Chapitre 3

Le multicast fiable et les services actifs

3.1 INTRODUCTION

Un service de communication point à multipoint offre un moyen efficace de diffuser des unités de données à un groupe de récepteurs, en ce sens qu'une seule copie de chacune de ces unités est envoyée par la source (s'il n'y pas de pertes). Le multicast IP (RFC 1112) fournit au niveau réseau un support efficace pour la diffusion non fiabilisé des paquets pour un grand nombre d'applications: visio-conférence, ftp multidestinataire, mise à jour d'informations dupliquées réparties, applications coopératives et tableau blanc, simulations distribuées,... Certaines de ces applications prévoient la mise en rapport d'un nombre de participants de l'ordre de plusieurs milliers et peuvent aussi, en plus de l'efficacité du routage, nécessiter une grande fiabilité dans la délivrance des données. Les recherches que je mène avec Moufida Maimour se concentrent sur le multicast fiable pour la distribution de données, et plus particulièrement sur les mécanismes au niveau transport pour la résistance au facteur d'échelle. Je n'ai pas encore regardé les aspects liés au routage, même si ceux-ci possèdent indéniablement des impacts importants sur les performances au niveau transport.

Je précise néanmoins que dans le cas du multicast fiable que nous étudions la résistance au facteur d'échelle signifie pouvoir supporter dans une session multicast de plusieurs centaines à quelques milliers de récepteurs (ce qui est déjà un défi pour le multicast fiable). Les applications types sont par exemple la mise à jour de programmes, l'échanges de bases de données et de fichiers de grande taille, la soumission de travaux sur une grille de calcul,... Contrairement au multicast non

fiable plutôt utilisé pour la diffusion temps-réel de contenus multimédia, je n'envisage pas, et ne voit pas réellement l'intérêt, d'avoir des groupes de plusieurs millions de récepteurs pour l'envoi fiable de fichiers. Les mécanismes qui sont donc proposés dans ce chapitre ne sont donc pas fait pour ce scénario d'utilisation.

Le problème de la fiabilité en point à point est bien maîtrisé et des solutions satisfaisantes (du moins pour TCP) ont été déployées. Par contre l'assurance de la fiabilité dans le contexte du multicast est un problème plus ardu et les solutions sont moins évidentes, surtout sur des réseaux étendus et hétérogènes. La conception de protocoles de diffusion fiable efficaces est plus difficile et doit tenir compte des contraintes imposées par la résistance au facteur d'échelle. L'implosion et la surcharge de la source par les messages de contrôle (ACK/NACK) (voir figure 3.1), l'exposition des récepteurs à des transmissions dupliquées ou le contrôle de congestion en sont des exemples. Malgré plusieurs années d'efforts consacrées à la conception de protocoles de diffusion fiable au dessus d'IP multicast, il est toujours aussi difficile de fournir une solution de bout-en-bout capable de satisfaire à l'ensemble des contraintes liées au facteur d'échelle.

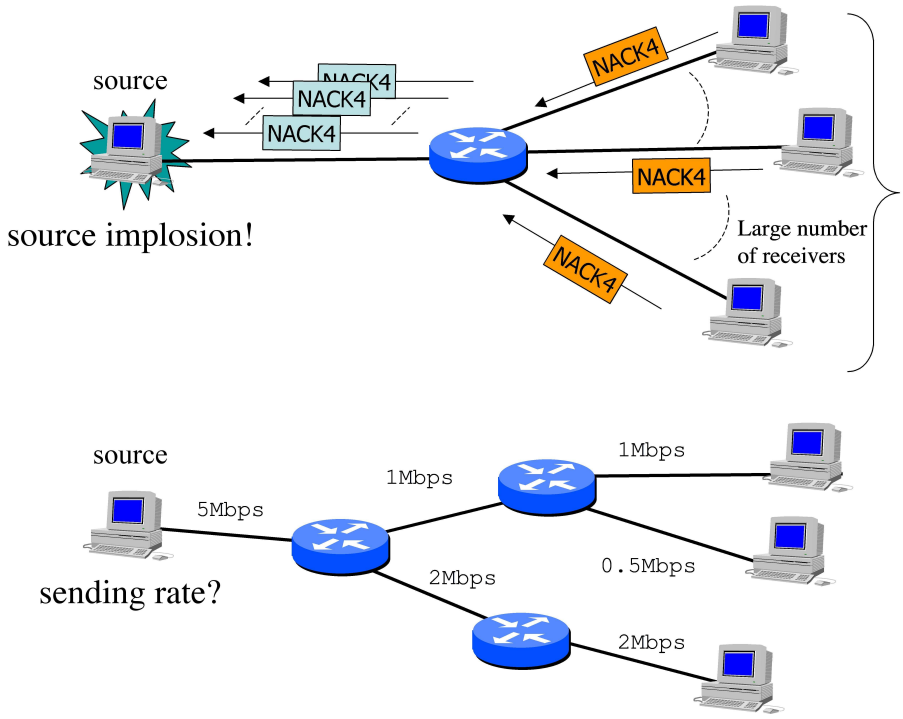

Figure 3.1 Implosion des NACKs et contrôle de congestion

La plupart des propositions récentes pour assurer la fiabilité utilisent du recouvrement local pour considérablement diminuer le temps de recouvrement [80]. Par exemple, un récepteur peut recevoir un paquet retransmis d'un autre récepteur dans le voisinage (SRM [29]), d'un récepteur désigné (RMTP [84], TMTP [117], LMS [83], PGM [103]) ou d'un serveur dédié (LBRM [45]). Parmi ceux-ci, PGM et LMS

utilisent les routeurs du réseau pour mettre en œuvre une structure hiérarchique. LMS exploite quelques informations sur la topologie disponible au niveau des routeurs. Avec des fonctions d'aiguillage spécifiques au niveau des routeurs, le protocole fait l'élection d'un récepteur comme *leader* pour chaque sous arbre pour satisfaire les requêtes de ses fils. PGM de son côté utilise une assistance de routeurs PGM pour choisir un récepteur (*Designed Local Retransmitter DLR*) qui doit cependant être sur le chemin vers la source. Des approches alternatives basées sur des codages FEC (Forward Error Correction) évitent, ou diminuent considérablement, les retransmissions à partir de la source ou des entités réparatrices (Asynchronous Layered Coding, RFC 3450-3453, en est un des représentants). Un des problèmes de ces approches est néanmoins la production des paquets FEC qui nécessitent souvent de garder (à un instant donné) en mémoire l'intégralité des données (tout le fichier de 10Go par exemple). Ces techniques FEC peuvent aussi être utilisées de manière conjointe à une approche conventionnelle (avec feedback). Dans ce cas, il est possible de voir le fichier comme plusieurs morceaux plus petits, et les ACK permettent de savoir si un morceau a été correctement reçu.

Une autre propriété souhaitable, et non des moindres, est la co-habitation des flux multicast avec les autres flux unicast gérés essentiellement par TCP. C'est ce que l'on nomme communément le contrôle de congestion. Dans les versions actuellement déployées de TCP par exemple ce contrôle est une combinaison de la phase de *slow-start* avec une phase de *congestion avoidance* [46]. Ce mécanisme permet aux entités sources des connexions TCP de partager relativement équitablement la bande passante disponible. Dans le cas du multicast, il est ardu de concevoir des mécanismes de contrôle de congestion qui offrent à la fois une bonne utilisation du réseau pour tous les récepteurs, et une compatibilité avec les flux TCP. Lorsque les données peuvent être décomposées en plusieurs couches (comme c'est assez facilement le cas pour des données vidéo), des approches utilisant des souscriptions à plusieurs groupes, de manière cumulative, sont possibles [75, 109, 11, 56].

De manière générale, la fiabilité et le contrôle de congestion en utilisant l'Internet tel qu'il est actuellement, c'est-à-dire avec un service IP non intelligent, est très difficile (manque d'information sur la topologie, sur le nombre de récepteurs, sur leur capacité...). Certaines approches qui ont été proposées (comme PGM [103] ou LMS [83] par exemple) ajoutent de nouvelles fonctionnalités dans les routeurs (et nécessitent donc des routeurs spécifiques). Cette démarche montre la nécessité de pouvoir adapter et ajouter dynamiquement des fonctionnalités de haut-niveau dans l'Internet pour offrir un support performant pour les communications de groupe.

Organisation du chapitre

Le reste de ce chapitre est organisé comme suit: la section 3.2 présente un état des verrous actuels qui limitent le déploiement et l'utilisation du multicast. La section suivante présente ensuite les travaux et les directions privilégiées par l'IETF et ses groupes de travail. Je présente ensuite dans la section 3.4 les concepts de la technologie des réseaux actifs/programmables et ses apports pour proposer des

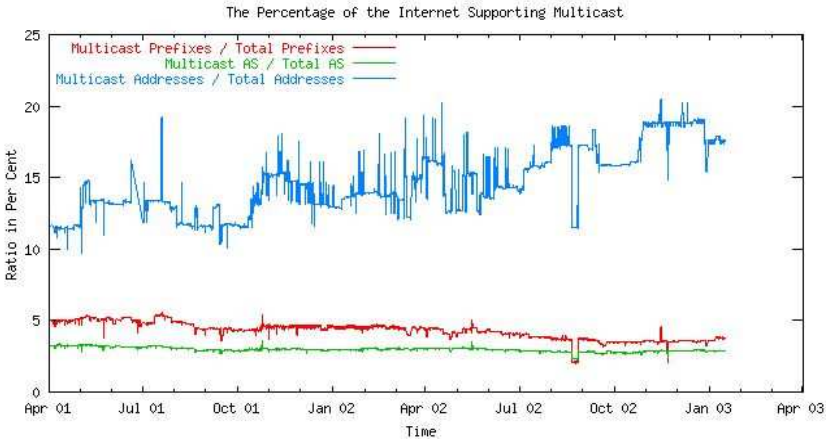

Figure 3.2 Chiffres représentatifs du déploiement du multicast.

solutions efficaces pour le multicast. Ensuite, je présente dans les sections 3.5, 3.6 et 3.7 les travaux que j'ai effectués avec Moufida Maimour sur la modélisation et l'analyse de protocoles de multicast fiable avec assistance des routeurs, la définition et l'analyse de nouveaux services, et enfin le protocole DyRAM que nous proposons. Nos travaux sur le support du multicast dans les grilles de calcul seront présentés dans la section 3.8. Les travaux concernant le protocole DyRAM, son implémentation et sa validation se font dans le cadre du projet RNRT VTHD++ et RNTL E-Toile. Une partie plus exploratoire concernant le multicast sur les grilles de calcul se fait dans le cadre d'une ACI GRID dont je suis le coordinateur.

3.2 LES VERROUS ASSOCIÉS AU MULTICAST

Bien qu'étant un sujet de recherche très fructueux depuis ces 15 dernières années, ayant donné lieu à un nombre impressionnant de contributions, force est de constater que le multicast reste très peu utilisé et est peu déployé. J'ai intégré ici quelques courbes issues du site `www.multicasttech.com/status/` qui donnent une idée du déploiement actuel du multicast dans l'Internet. La figure 3.2 montre la taille relative, en pourcentage, de l'Internet multicast par rapport à l'Internet global. Les critères utilisés sont le nombre de préfixes, de systèmes autonomes (AS) et d'adresses.

Le taux de pénétration du multicast dans l'Internet est quelque part entre la courbe des préfixes et la courbe des AS, soit moins de 5%, avec une certaine stabilité. On constate cependant qu'entre juillet 2002 et janvier 2003, l'Internet global augmente plus vite que l'Internet multicast (les nouveaux réseaux créés ne sont malheureusement pas multicast). Si l'on regarde au sein des AS multicast ceux qui possèdent des sources actives (ce qui donne une approximation de l'utilisation réelle du multicast), on voit qu'ils représentent environ un tiers du nombre des AS multi-

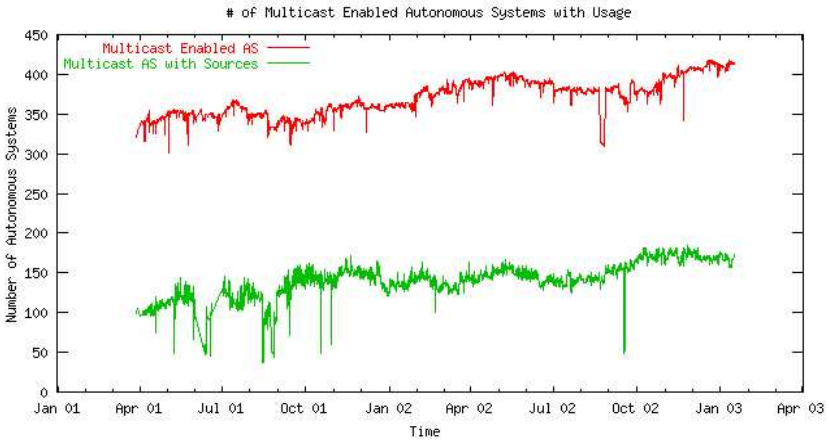

Figure 3.3 Systèmes Autonomes avec des sources actives.

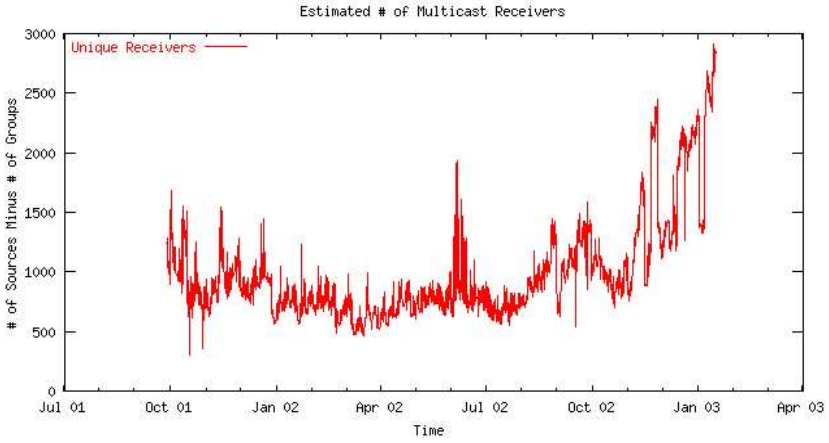

Figure 3.4 Approximation du nombre total de récepteurs (borne inférieure car les statistiques n'interceptent que les récepteurs utilisant RTCP).

cast (cf. figure 3.3). Cela se traduit en terme de nombre de récepteurs à quelques milliers de récepteurs (cf figure 3.4).

On le voit bien, l'Internet est bien loin d'adopter la technologie multicast, et les utilisateurs aussi, malgré un besoin important. Les raisons de cet "échec" sont nombreux et la difficulté consiste à résoudre un grand nombre de problèmes en même temps car ceux-ci sont très dépendant les uns des autres. Par exemple, même si le routage multicast à l'intérieur d'un domaine est relativement efficace (PIM-SM/DM, CBT...), le routage inter-domaine l'est moins, et cela pour des raisons de résistance au passage à l'échelle et de modèle économique.

Ce problème économique est le plus important chez les opérateurs. En effet, beaucoup d'opérateurs basent aujourd'hui leur modèle sur le débit d'accès du client (voire la quantité de traffic), donc ils pourraient ne pas être intéressés à priori par des services qui consisteraient à ne facturer qu'un paquet au lieu de plusieurs. Ceci est

une erreur car ce modèle n'est certainement pas le bon pour survivre dans le contexte actuel de concurrence sur les services. Les opérateurs ont intérêt à développer des services à valeur ajoutée: déployer le multicast amènerait à généraliser des usages haut débit et développer le concept des ASP (*Application Service Provider*) par exemple.

Dans le milieu industriel, un des verrous les plus forts concerne la faible sécurité du modèle d'IP multicast. En effet, dans des scénarios d'utilisation quotidienne du multicast par les entreprises pour la mise à jour de bases de données, de logiciels, etc. il est indispensable de pouvoir définir des règles pour contrôler soit les sources, soit les récepteurs. Cette sécurisation peut se faire à plusieurs niveaux: près du réseau, ou plus près de la couche transport voire applicative.

Pour le multicast fiable, il faut reconnaître que le coût de la fiabilité est très grand. Même si pour un groupe d'un millier de récepteurs le problème de l'implosion des ACK/NACK à la source n'est pas vraiment préoccupant (voir [62]) c'est le recouvrement des pertes qui peut être source de problèmes. A cela s'ajoute la complexité de la mise en oeuvre car IP multicast n'est pas activé partout (donc nécessité de mettre des tunnels). C'est une des raisons pour laquelle un certain nombre de travaux proposent des approches pour le multicast au niveau applicatif qui permettent de faire la gestion des groupes, les duplications de paquets..., sans avoir besoin d'un déploiement au niveau réseau. Ces approches sont désignées comme *application-level multicast* ou bien *end-system multicast* ou bien encore *host-based multicast* [105, 21, 85, 19, 20, 121, 101]. Certaines de ces approches peuvent se passer complètement de IP multicast, mais d'autres peuvent aussi profiter de nuages IP multicast lorsqu'ils existent [20, 121]. Il existe également des solutions basées sur des techniques peer-to-peer comme SCRIBE [13] qui permettent un service de distribution fiable de données extrêmement robuste au sein de populations très larges et fortement dynamiques. Notons aussi l'approche très intéressante proposée dans [62] qui ajoute des extensions à TCP pour pouvoir gérer les communications point à multi-points pour des groupes de taille moyenne (environ un millier).

3.3 L'IETF ET LES DIRECTIONS "STANDARDS"

Le multicast est un des thèmes importants de l'Internet et les groupes traitant du multicast dans l'IETF sont:

- idmr: Inter-Domain Multicast Routing
- msdp: Multicast Source Discovery Protocol
- pim: Protocol Independent Multicast
- ssm: Source-Specific Multicast
- msec: Multicast Security

- malloc: Multicast-Address Allocation
- rmt: Reliable Multicast Transport
- magma: Multicast & Anycast Group Membership
- bgmp: Border Gateway Multicast Protocol
- mboned: MBONE Deployment

Pour le multicast fiable, il y aussi un groupe de recherche "Reliable Multicast Research Group" de l'IRTF. Pour l'IETF, le groupe *rmt* est celui qui est le plus concerné. Les travaux qui y sont menés sont NORM avec des ACK/NACK, TRACK avec une structure en arbre et ALC avec des codages multi-couches. L'IETF a récemment publié un certain nombre de RFCs qui montrent, plus ou moins, les directions privilégiées. En particulier, les propositions basées sur la décomposition en couches (RFC 3450, 3451) ainsi que celles utilisant des codes FEC (RFC 3452, 3453) sont bien avancées. L'équipe INRIA/Planète a largement contribué à ces RFCs.

```
RFC 2887: The Reliable Multicast Design Space for Bulk Data Transfer
M. Handley, S. Floyd, B. Whetten, R. Kermode, L. Vicisano, M. Luby,
August 2000

RFC 3208: PGM Reliable Transport Protocol Specification
T. Speakman, J. Crowcroft, J. Gemmell, D. Farinacci, S. Lin,
D. Leshchiner, M. Luby, T. Montgomery, L. Rizzo, A. Tweedly, N. Bhaskar,
R. Edmonstone, R. Sumanasekera, L. Vicisano, December 2001

RFC 3269: Author Guidelines for Reliable Multicast Transport (RMT)
Building Blocks and Protocol Instantiation documents R. Kermode,
L. Vicisano, April 2002

RFC 3048: Reliable Multicast Transport Building Blocks for One-to-Many
Bulk-Data Transfer B. Whetten, L. Vicisano, R. Kermode, M. Handley,
S. Floyd, M. Luby, January 2001

RFC 3450: Asynchronous Layered Coding (ALC) Protocol Instantiation
M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, J. Crowcroft, December 2002

RFC 3451: Layered Coding Transport (LCT) Building Block M. Luby,
J. Gemmell, L. Vicisano, L. Rizzo, M. Handley, J. Crowcroft, December
2002

RFC 3452: Forward Error Correction Building Block M. Luby, L. Vicisano,
J. Gemmell, L. Rizzo, M. Handley, J. Crowcroft, December 2002

RFC 3453: The Use of Forward Error Correction in Reliable Multicast
M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, J. Crowcroft,
December 2002
```

Des nouveaux modèles sont également apparus, qui simplifient le modèle général de l'IP multicast qui est multi-source/multi-destination, et qui permettent soit de rajouter de la sécurité, soit de gagner en simplicité et donc en performance. En particulier, le modèle EXPRESS [44], EXPlicitly REquested Single-Source, (traité par le groupe SSM) se focalise sur le cas 1-vers-n.

3.4 RÉSEAUX ACTIFS/PROGRAMMABLE ET MULTICAST FIABLE

3.4.1 Les réseaux actifs

Le concept des réseaux actifs proposé dans [107, 108] consiste d'une part à donner aux paquets (rebaptisé *capsules* dans certaines approches à granularité fine) une certaine liberté d'action avec la possibilité d'inclure un peu de code et, d'autre part, à prôner l'ouverture des équipements avec des interfaces d'accès standardisées pour l'exécution de codes spécifiques (IEEE P1520 par exemple).

La réalisation pratique la plus courante consiste à donner la possibilité aux routeurs d'exécuter un traitement spécifique sur les paquets qui arrivent. La différence avec des approches utilisant le marquage, le plus souvent associé ensuite avec un ordonnancement spécifique (comme le fait DiffServ), est que le traitement spécifique à appliquer sur les paquets peut être fourni par une application de l'utilisateur final. Ce concept autorise le déploiement dynamique des protocoles et des services, permettant ainsi une grande souplesse aux applications qui adoptent cette philosophie. Cette souplesse rend les réseaux actifs intéressant et puissant, car elle donne une grande liberté à la conception et à la mise en place de divers mécanismes et algorithmes adaptés aux besoins précis d'une application pour un instant et, pourquoi pas aussi un lieu géographique donné.

Des environnements d'exécution, qui sont en concepts très proches des systèmes d'exploitation, sont nécessaires pour l'exécution des codes spécifiques. Cette ouverture des équipements n'est pas simple, et de nombreux problèmes restent difficiles à résoudre: sécurité des accès et fiabilité des codes à exécuter, coûts introduits par les environnements d'exécution et des traitements, isolation entre flux et facturation... On le voit, les problèmes sont complexes et nombreux. Un certain nombre de projets internationaux et nationaux ont été menés sur ce thème ces dernières années: CANES (US), ANAIS (FR), AMARRAGE (FR), AMARILLO (FR),...) et une action spécifique CNRS a également été lancé (2001-2002) pour étudier et promouvoir le concept des réseaux actifs. Des réalisations ont été faites: des environnements d'exécution actifs ont été écrits (ANTS [114], TAMANOIR [35], CLARA [81]...) et des équipements actifs ont aussi été proposés (Alcatel a réalisé des prototypes de routeurs actifs à partir d'une base conventionelle; la societé 6Wind commercialise de petits routeurs d'accès actifs).

Le concept des réseaux actifs est cependant à contre-courant de la philosophie aujourd'hui dominante, celle de l'Internet. Celle-ci suit une logique très simple : le réseau a pour but de véhiculer les informations et il faut qu'il se cantonne à ce rôle

de simple transporteur pour laisser aux *extrémités* les traitements plus sophistiqués. Dans ce cadre, les entités de transport de données (les paquets IP) sont "simplement" relayées par des routeurs jusqu'à leur destination finale. La philosophie Internet a ses atouts, en particulier sa simplicité de mise en oeuvre, qui a conduit au succès de l'Internet actuel. Elle n'est cependant pas la solution universelle. Car, il y a bien des cas dans lesquels un traitement intermédiaire ou le déploiement d'un nouveau protocole serait souhaitable, voir nécessaire. Les exemples ne manquent pas: supervision de réseaux [104], régulation de flux temps-réel [41], monitoring, simulation distribuée [120], et aussi multicast bien sûr que nous détaillerons plus tard.

Il faut signaler aussi que l'Internet a pour environnement une infrastructure principalement filaire, très vaste et presque sans contrôle, reliant le plus souvent des terminaux relativement puissants (des stations de travail, PC). Il existe d'autres systèmes, par exemple les environnements mobiles, pour lesquels la nécessité de changement de comportement, le manque de ressources sur les terminaux et le besoin de s'adapter à l'environnement exigent une certaine intelligence de la part du réseau: les réseaux actifs pourront y jouer un rôle important.

3.4.2 Multicast et réseaux actifs

Les principaux problèmes du multicast liés au facteur d'échelle sont dus à (*i*) un manque d'information (sur la topologie, sur les groupes, sur les pertes, sur les délais...), et (*ii*) un manque de distribution des traitements (traitement des ACK/NACK, des retransmissions). Le concept des réseaux actifs est particulièrement bénéfique dans le cas du multicast car on peut facilement voir qu'il est possible de déployer des services spécifiques, et relativement légers, pour pallier à ces manques.

Par exemple, pour résoudre le problème de l'implosion des messages de contrôle l'avantage d'une approche active est de permettre aux routeurs de pouvoir aggréger les messages. Ensuite, pour réduire les latences de recouvrement, il est possible de demander aux routeurs, s'ils possèdent du cache disponible, de retransmettre les paquets perdus. En ayant accès à des informations disponibles à même le réseau, il est ensuite possible d'optimiser les retransmissions aux seuls récepteurs concernés. Si cette possibilité n'était pas retenue (cache très coûteux), les routeurs actifs peuvent toujours aider à détecter les pertes et à localiser l'entité réparatrice la plus apte à réparer une perte donnée.

Peu après l'introduction du concept des réseaux actifs, la communauté scientifique, essentiellement nord-américaine au départ, a réalisé un certain nombre de contributions sur l'apport des réseaux actifs sur les protocoles de multicast fiables[1]: ARM [60], RMANP [12], APES [97], AER/NCA [51]. ARM et RMANP proposent tous deux des services de cache de paquets au niveau des routeurs et des services d'aggrégation des NACKs. La taille de cache nécessaire peut cependant être très

[1] Je n'ai énuméré ici que les premières propositions. Des propositions plus récentes ont été faites depuis, voir [102].

importante dans ces approches. APES propose l'utilisation conjointe de code FEC avec du recouvrement local afin de réduire la quantité de cache dans le réseau. AER/NCA est un protocole similaire a ARM, mais qui utilise de manière conjointe une suppression locale des NACK (basée sur des temporisateurs) avec des aggrégations au niveau de noeuds de retransmission (*repair server*, co-localisé avec des routeurs actifs). De plus, le recouvrement local est distribué sur un ensemble de *repair server* qui se déclarent auprès des récepteurs et de la source. Il n'est cependant pas clair dans cette approche comment les *repair server* décident de se déclarer, et à quel moment ils décident de se retirer.

3.4.3 Pourquoi rechercher dans cette direction?

Tous ces travaux sont très intéressants et montrent bien les bénéfices apportés par les réseaux actifs: traitement plus localisé et meilleur exploitation des informations de topologie ou parfois de sémantique. Cependant, il est assez facile de critiquer l'approche du multicast actif car le déploiement des réseaux actifs se heurte aussi à des problèmes de sécurité et de performance. Si l'on rajoute à cela le problème actuel du déploiement du multicast "simple", alors l'approche active semble un peu utopique aux premiers abords.

Si l'on y regarde d'un peu plus près, un certain nombre d'approches qui ont été proposées pour contourner le problème de la résistance au facteur d'échelle ou au manque de déploiement d'IP multicast introduisent eux-mêmes des fonctionnalités supplémentaires dans les routeurs de l'arbre de multicast (par le biais de protocoles supplémentaires) dont le déploiement à grande échelle est aussi aléatoire. C'est le cas par exemple de REUNITE [105], HBH [20], PGM [103, 36], LMS [83] ou de l'approche proposée dans [54].

Si les services actifs sont légers, la frontière est mince entre un protocole utilisant des services actifs, et un autre utilisant des services supplémentaires dans les routeurs, qu'il faudra aussi déployer à un moment donné. En utilisant le concept des réseaux actifs, la manière dont les nouveaux services seront déployés a le mérite d'être claire, même si la solution n'est pas encore finalisée: du code est chargé dans un environnement d'exécution similaire à une système d'exploitation. De plus, l'utilisation de routeurs actifs dans la phase de validation d'un protocole, n'implique pas automatiquement un déploiement opérationel sur des routeurs obligatoirement actifs et très ouverts (c'est à dire acceptant du code utilisateur). La validation sur routeurs actifs permet de tester en grandeur réelle le protocole pour passer ensuite, si les services sont légers et s'il y a consensus avec des constructeurs, à une approche plus traditionnelles. Comme la plupart des services actifs ne sont pas à déployer dans tous les routeurs, mais sur seulement une faible proportion d'entre eux, cela rend les deux catégories encore plus proches.

Les travaux sur le multicast actif précédemment cités représentent à mes yeux un premier essai de validité. Certains d'entre eux ne sont pas vraiment différents des protocoles non-actifs nécessitant des routeurs spécifiques, alors que d'autres proposent des services lourds comme le cache dans les routeurs. Cette lourdeur me

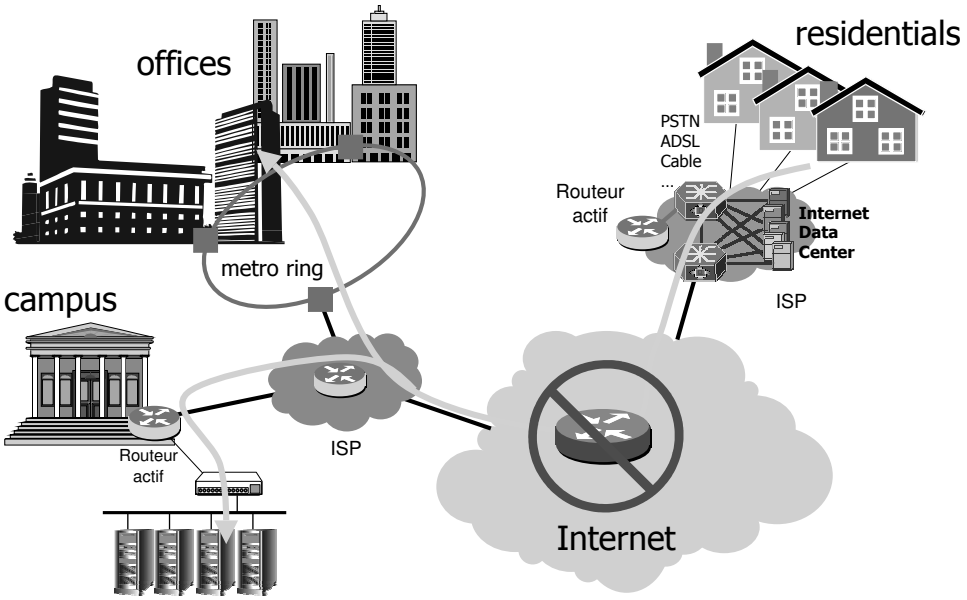

Figure 3.5 Déploiement de solutions actives.

semblaient trop pénélisante et j'avais la conviction qu'il était possible de proposer d'autres services légers (pour éviter par exemple le cache dans les routeur) dont le déploiement ne serait pas beaucoup plus difficile qu'une approche à base de routeurs spécifiques. **Nos recherches utilisent donc plus le modèle de routeur programmable que celui de routeur purement actif. Le terme actif sera employé pour indiquer que des environnements d'exécution sont utilisés pour le prototypage et le déploiement**.

Un scénario de déploiement que je pense possible est celui représenté par la figure 3.5 où les services supplémentaires sont introduits à la périphérie des réseaux d'opérateur, là où l'intelligence apporte la meilleure plus-value. Ces routeurs "augmentés" (ceux basés sur des environnements actifs en constituent un sous-ensemble) peuvent être déployés au sein d'un fournisseur d'accès, qui souhaite par exemple mettre en place des services de haut-niveau, ou bien également à l'initiative d'un domaine privé tels que les campus, les laboratoires, les centres de calcul ou les grandes entreprises. Avec l'apparition de machines très puissantes (et souvent très abordables) pouvant servir de routeurs actifs, de manière complémentaire aux routeurs traditionnels, ce scénario est tout à fait envisageable à moyen terme.

La manière dont je souhaitais aborder ce nouveau thème après mes recherches sur la simulation était donc d'apporter, d'une part, des contributions sur le dimensionnement de ces systèmes (avec des méthodes analytiques simples) et, d'autre part, sur la définition de nouveaux services légers qui peuvent grandement améliorer la

résistance au facteur d'échelle et les performances de bout-en-bout pour les applications. Nous verrons plus tard, dans la section 3.8 qu'il est aussi envisageable de proposer des services à déploiement spécifique en prenant en compte les besoins des applications d'une grille de calcul par exemple.

3.5 MODÉLISATION ET ANALYSE DU MULTICAST FIABLE ET DES SERVICES ACTIFS

La principale caractéristique d'un multicast fiable utilisant des services actifs est que les routeurs incorporent des traitements supplémentaires au traitement de routage habituel, augmentant ainsi le temps de séjour d'un paquet actif dans le routeur. La démarche générale consiste donc à trouver, en fonction du taux de perte, le nombre de paquets de données et de contrôle que chaque élément dans le réseau (routeur, source et récepteur) est susceptible de traiter. De précédentes études, dans un contexte non actif, ont déjà utilisé cette méthodologie avec succès. Par exemple, les auteurs dans [49] ont montré que les protocoles utilisant un recouvrement local à partir de serveurs co-localisé affichaient de meilleures performances que les approches de bout-en-bout et ceux effectuant du recouvrement à partir d'autres récepteurs. La méthode d'analyse est basée sur celle proposée dans [91] et réutilisé ensuite dans [61, 50] pour étudier d'autres protocoles de multicast fiable.

Lorsque nous avons commencé cette étude, les 3 services actifs dans les routeurs les plus courants sont (*i*) le cache des paquets pour le recouvrement local, (*ii*) l'aggrégation des messages de contrôle et, (*iii*) la retransmission partielle pour éviter le problème de l'exposition (subcast). Une approche comme ARM utilise les 3 services, alors que AER, par exemple, n'utilise que le premier avec une politique locale de suppression des messages de contrôle. Aucun travail n'avait étudié la combinaison des 3 services, ni étudié les contraintes en terme de puissance de calcul des routeurs actifs pour l'obtention de meilleures performances. C'est ce qui nous a semblé être un manque pour bien comprendre les approches à assitance de routeurs.

Nous avons donc souhaité pouvoir évaluer les performances de ces différentes stratégies. Pour cela Moufida Maimour a dérivé des protocoles génériques qui bénéficient du cache des paquets de données au niveau des routeurs actifs. Ces protocoles sont notés S_1, S_2, S_2^s, S_3 et S_3^s. Cependant, chacun d'eux adopte une stratégie différente pour la suppression des NAKs qui peut être combinée ou non avec le subcast des paquets retransmis. Ce subcast, s'il est effectué au niveau de la source sera noté par l'exposant ^{s}. S_2 et S_2^s utilisent une suppression locale, alors que S_1, S_3 et S_3^s utilisent une suppression globale dans les routeurs. Précisons que dans le cas de S_2^s, le subcast n'est possible que pour les récepteurs non liés à un routeur actif. En effet, il n'est pas facile d'offrir un service de subcast avec une stratégie de suppression locale car les routeurs ne connaissent pas l'identité des récepteurs ayant subi des pertes. Pour S_3, le multicast partiel est intégré dans les routeurs actifs. Le tableau 3.1 résume les différentes propriétés des protocoles étudiés.

Une évaluation analytique de ces protocoles qui utilise le temps de traitement

Protocole	locale	globale	subcast routeur	subcast source
S_1		oui		
S_2	oui			
S_2^s	oui			oui
S_3		oui	oui	oui
S_3^s		oui	oui	oui

Tableau 3.1 Résumé des propriétés des protocoles étudiés.

dans les différents composants pour en tirer les débits maximum est ensuite effectuée. Cette modélisation analytique reste simple mais donne des "bornes" facilement exploitables.

3.5.1 Modèle d'analyse

Le modèle de réseau et de pertes est inspiré de celui utilisé dans [50]. Nous considérons le cas d'une seule source S qui diffuse à R récepteurs (Fig.3.6). Le nombre des récepteurs est supposé multiple de B pour qu'on puisse les distribuer sur $N = R/B$ groupes locaux dans le cas où tous les routeurs sont actifs. Chaque groupe local dans ce cas est constitué de B récepteurs. Nous adoptons un modèle où les routeurs actifs sont placés à la périphérie du réseau de cœur. Cela pour deux raisons essentielles:

- Le réseau de cœur est fiable. En effet, il a été montré dans [115] que les liens qui subissent le plus de pertes sont à la périphérie. Ces liens sont de deux types: le lien "source" qui relie la source au réseau et l'ensemble des liens "terminaux" qui relient les récepteurs au réseau.
- Le réseau de cœur est un réseau à haut débit. Si on y place des routeurs actifs, il sera ralenti par les traitements effectués par ces derniers.

Le modèle ne suppose pas que tous les routeurs à la périphérie soient actifs. En effet, on considère que F routeurs ($0 \leq F \leq N$) parmi N sont actifs (Fig.3.6). Chaque routeur actif A_i ($i = 1 \,..\, F$) est responsable de B récepteurs $R_{i1}, \cdots, R_{iB}$, qui forment son groupe local. Un récepteur relié dans l'arbre du multicast à un routeur actif est dit **lié**. Les autres récepteurs non liés sont dits **libres** (Fig.3.6).

Pour la modélisation des pertes, p_l est la probabilité qu'une perte se produise au niveau du lien source ou d'un lien terminal. La probabilité, vue par un récepteur, qu'une perte de bout en bout aura lieu est alors $p = 1 - (1 - p_l)^2$. Dans cette première analyse, la corrélation temporelle n'est pas prise en compte; elle le sera dans les simulations que je présenterai plus loin dans la section 3.7. Les pertes au niveau des liens terminaux sont supposées mutuellement indépendantes.

Cette configuration de base peut être améliorée en liant la source à un routeur actif que nous désignerons par A_S (Fig.3.7). Cela permet de déterminer les meilleures configurations. Les cas particuliers où $F = 0$ et $F = N$ sont également examinés.

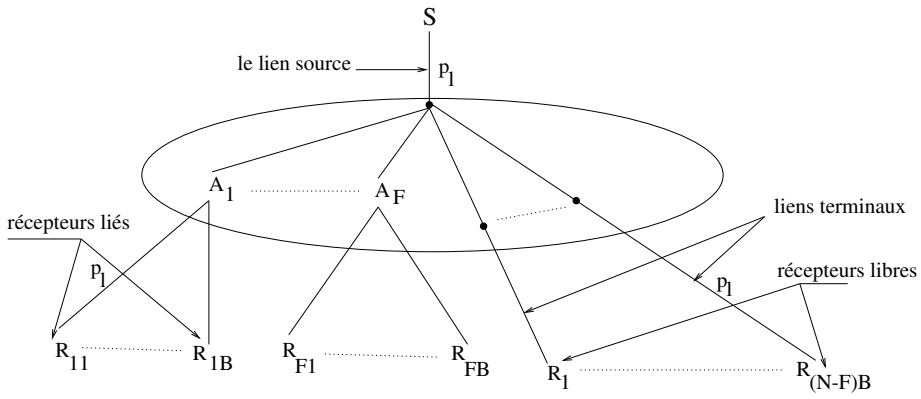

Figure 3.6 Modèle des réseaux et des pertes: configuration de base

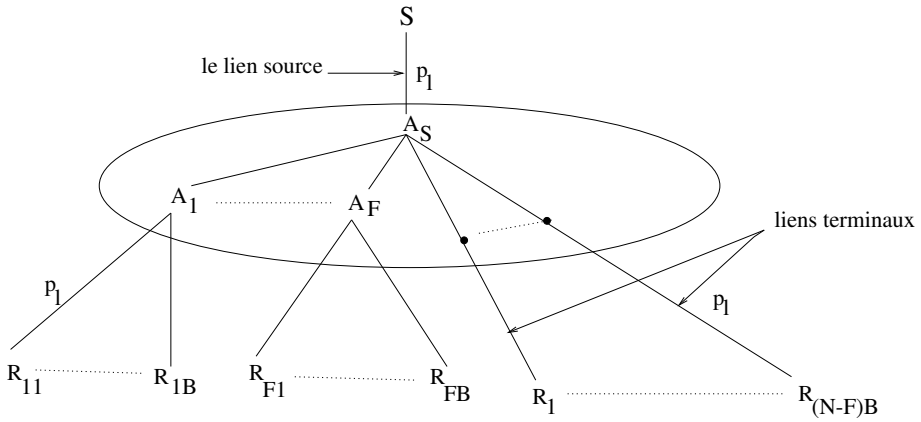

Figure 3.7 Modéle des réseaux et des pertes: configuration améliorée

3.5.2 Principales hypothèses de travail

Nous supposons qu'un NACK envoyé par unicast suit le chemin inverse du multicast. Cette supposition est primordiale pour pouvoir bénéficier des services actifs pour le multicast fiable. Cela peut être réalisé en implémentant un service de routage spécifique aux NACKs. Une autre approche consiste à permettre à un récepteur de connaître l'identité du routeur actif auquel il est relié (comme dans [51]). Nous supposons donc que l'une ou l'autre des approches est utilisée.

Nous supposons également dans cette première analyse que les NACKs ne sont jamais perdus. Cette hypothèse ne pénalisera aucun des protocoles analysés. Par contre elle permet la simplification des expressions analytiques. Cette hypothèse sera levée dans les simulations que nous présenterons plus loin dans la section 3.7. Nous supposerons également que l'on dispose de suffisamment de moyens de stockage de telle sorte que lorsqu'un NACK arrive à un routeur actif, ce dernier aura le paquet demandé dans son cache.

3.5.3 Principaux résultats de cette première analyse

Un des premiers résultats est d'avoir développer un modèle que nous pourrons réutiliser lorsque nous proposerons et validerons de nouveaux services. L'analyse et l'ensemble des résultats (avec les courbes) peuvent être trouvés dans l'article [74][2]. Je ne résumerai ici que les principaux résultats obtenus qui ont guidés nos reflexions futures.

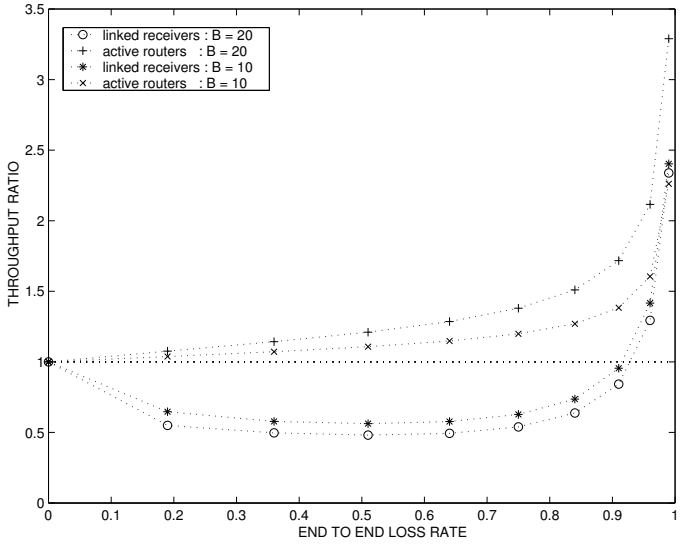

Figure 3.8 Suppression locale vs suppression globale S_2/S_3.

Suppression locale vs suppression globale

Afin de comparer précisément les deux stratégies de suppression des NACK, la figure 3.8 montre pour les récepteurs liés et les routeurs actifs le gain en débit en fonction de la probabilité de pertes. Plusieurs tailles de groupes locaux sont utilisées.

La figure 3.8 montre le gain entre S_2 et S_3. Pour des taux de pertes pas trop grands, S_3 est meilleure que S_2 pour les récepteurs liés car ceux-ci peuvent bénéficier du subcast. Dans S_2, un récepteur lié peut continuer à recevoir des NACKs en provenance de son routeur actif à chaque fois qu'un récepteur voisin (dans le même sous-groupe) subit des pertes.

Bénéfice du subcast des paquets retransmis

Le subcast a l'avantage de décharger les récepteurs et/ou les routeurs actifs selon si le subcast à partir de la source est disponible ou non. Pour comprendre le bénéfice du subcast à partir des routeurs actifs associés aux récepteurs liés, la figure 3.9 montre le gain pour un tel récepteur dans le cas de S_3 et S_2. Nous pouvons voir

[2]M. Maimour, C. Pham, "A Throughput Analysis of Reliable Multicast Protocols in an Active Networking Environment", *Proceedings of the Sixth IEEE Symposium on Computers and Communications (ISCC 2001), 3-5 July, 2001, Hammamet, Tunisia.*

que le subcast permet un plus grand débit dans S_3. Ce gain dépend de la taille du groupe local et du taux de pertes. Il est plus bénéfique de faire du subcast lorsque le groupe local est grand. Pour de très faibles et très forts taux de pertes, le subcast n'offre pas de gains très remarquables. Le choix de faire du subcast ou non peut donc se faire en fonction des pertes observées.

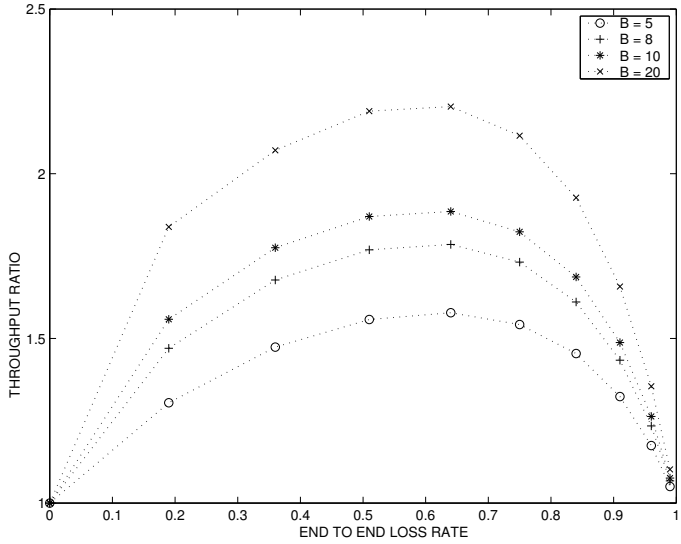

Figure 3.9 Bénéfice du subcast pour les récepteurs liés (rapport S_3/S_2).

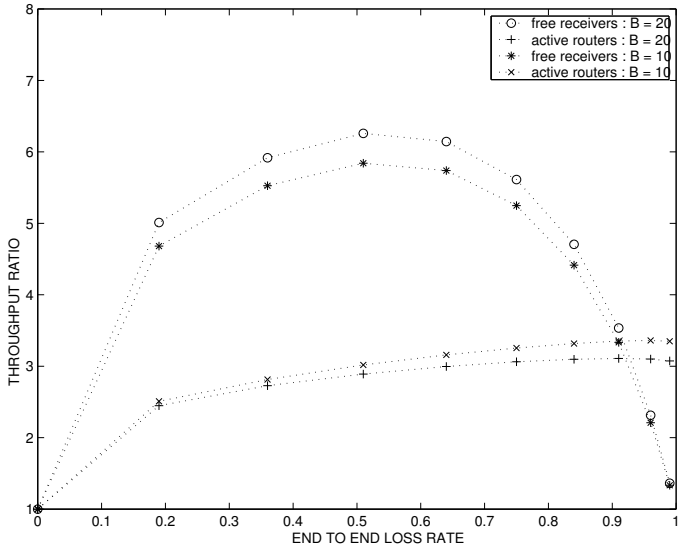

Figure 3.10 Bénéfice du subcast à partir de la source, rapport S_2^s/S_2

Pour montrer le bénéfice du subcast à partir de la source, la figure 3.10 représente le gain dans les récepteurs libres (non liés) entre S_2^s and S_2. Deux tailles différentes

pour le groupe local sont utilisées. Dans le modèle, le nombre de récepteurs libres ($(N-F)B$) est proportionnel à la taille du groupe local (B). Au niveau des récepteurs libres, il est possible d'avoir des gains de 5 et presque 6 pour des taux de pertes de 20% and 50% respectivement. Pour les routeurs actifs, contrairement au cas S_3^s et S_3, S_2^s est meilleure que S_2 même pour des taux de pertes élévés. En effet dans S_3^s le routeur actif reçoit tous les NACKs générés dans le groupe local, dont le nombre augmente considérablement avec p.

Densité des routeurs actifs

La figure 3.11 montre l'impact sur les performances de S_3 de la densité de routeurs actifs. La figure représente le gain en fonction de cette densité par rapport à un cas où il n'y a pas du tout de routeurs actifs. Nous avons également étudié plusieurs scénario en fonction de la puissance de traitement des routeurs actifs. Avec une puissance de traitement identique aux récepteurs, le gain est d'un ordre de grandeur plus grand si tous les récepteurs sont liés. Cas plus intéressant, si la puissance de traitement du routeur actif est divisée par 10 il est encore possible de doubler le débit si 55% des routeurs sont actifs.

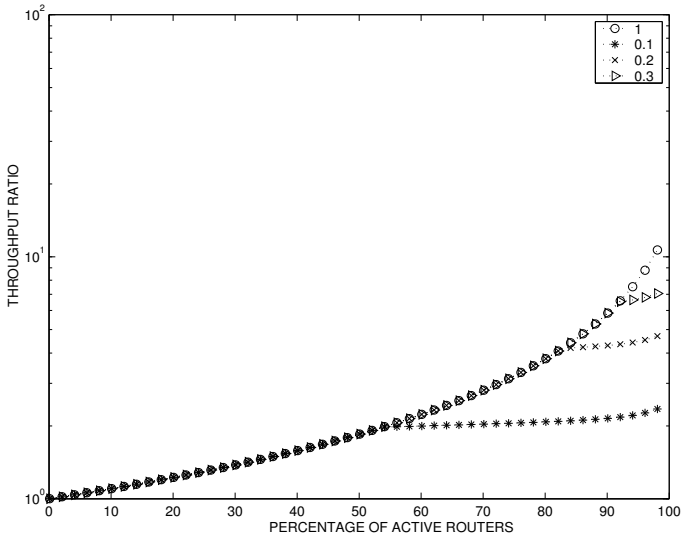

Figure 3.11 Gain en débit pour S_3 en fonction de la densité de routeurs actifs. $p = 0.05$, $B = 10$ et $N = 1000$.

3.6 PROPOSITION DE NOUVEAUX SERVICES ACTIFS

Les services que nous avons évalués jusqu'à présent sont le cache des paquets, l'aggrégation des messages de contrôle et le multicast partiel des paquets retransmis. Notre position vis-à-vis du cache de paquets qui est un service lourd dans les

routeurs nous a conduit à chercher d'autres services légers pour améliorer les performances du recouvrement local. C'est dans cette démarche, qui me semble plus facilement déployable, que nous avons par exemple proposé et évalué 4 nouveaux services qui sont:

- l'élection dynamique d'un retransmetteur: un routeur actif d'assistance peut élire un récepteur pour retransmettre un paquet perdu par un de ces voisins.

- la détection rapide des pertes dans les routeurs: un routeur actif d'assistance peut générer un NACK vers la source s'il détecte une perte de séquence dans le flux des paquets.

- l'aggrégation des RTTs: les routeurs actifs d'assistance participent à l'aggrégation des RTTs par segment afin de générer une valeur de RTT plus précise permettant une régulation du débit plus fluide par la source.

- le partionnement des récepteurs en sous-groupes pour améliorer la gestion des groupes hétérogènes.

Pour les 2 premiers services, nous avons, à l'aide d'une démarche similaire à l'évaluation analytique précédente, modélisé et étudié leurs apports et leurs performances. Ces résultats ont été publiés dans l'article #3[3] et #4[4] qui peuvent être trouvés en annexe à la fin du document. Pour les 2 derniers services consistant en l'aggrégation des RTTs et au partionnement des récepteurs, des résultats ont été présentés dans #5[5] qui peut être trouvé en annexe à la fin du document, et [68][6] et [66][7]

Ces 4 services sont des services légers qui ne consomment que très peu de ressources dans les routeurs, mais qui, combinés avec un recouvrement local, permettent de réduire les latences de bout-en-bout et augmenter la satisfaction des récepteurs.

[3]M. Maimour, C. Pham, "An Analysis of a Router-based Loss Detection Service for Active Reliable Multicast Protocols", *Proceedings of the 11th IEEE International Conference on Networks (ICON 2002), August 27-30, 2002, Singapore.*

[4]M. Maimour, C. Pham, "Dynamic Replier Active Reliable Multicast (DyRAM)", *Proceedings of 7th IEEE Symposium on Computers and Communications (ISCC 2002), July 1-4 2002, Taormina, Italy, pp275-282.*

[5]M. Maimour, C. Pham, "AMCA: an Active-based Multicast Congestion Avoidance Algorithm", *Proceedings of 8th IEEE Symposium on Computers and Communications (ISCC 2003), June 30-July 3 2003, Kemer, Antalya, Turkey.* Best paper award.

[6]M. Maimour, C. Pham, "A RTT-based Partitioning Algorithm for a Multi-rate Reliable Multicast Protocol", *Proceedings of IEEE High Speed Network and Multimedia Communications (HSNMC 2003), July 23-25 2003, Estoril, Portugal.*

[7]M. Maimour, C. Pham, "Dealing with heterogeneity in a fully reliable multicast protocol", *Proceedings of IEEE International Conference On Networks (ICON 2003), Sydney, Autralia, September 2003.*

3.7 LE PROTOCOLE DYRAM

Nous avons intégré dans un protocole que nous avons appelé DyRAM, pour *Dynamic Replier Active reliable Multicast*, les nouveaux services que nous avons proposé, ainsi que les services d'aggrégation globale et de subcast. L'un des objectifs majeurs de DyRAM est de se passer du cache dans les routeurs et d'obtenir une latence de recouvrement faible. En ce sens, il est bien différent de ARM [60], AER [51] ou MAF [102]. DyRAM est présenté en détail dans l'article #4 qui peut être trouvé en annexe à la fin du document.

3.7.1 Modélisation et simulation

Pour évaluer DyRAM, nous avons choisi d'utiliser la simulation, et dans un premier temps, le langage PARSEC, que j'avais utilisé en simulation parallèle durant mon séjour post-doctoral, puis le simulateur *ns* pour pouvoir étudier l'interaction avec des flux TCP et proposer des mécanismes de contrôle de congestion. La simulation nous a permis de modéliser plus finement l'interaction entre les différents services actifs et d'ajouter des modèles de pertes plus réalistes en prenant en compte la corrélation temporelle.

3.7.2 Principaux résultats de cette deuxième analyse

Recouvrement local et élection

Les figures 3.12a et 3.12b montrent pour DyRAM le nombre de retransmissions (M_S) à partir de la source et le temps moyen de transfert. Ces résultats ont été obtenus avec des simulations de sessions multicast comportant 48 récepteurs distribués en 12 groupes locaux. Les courbes montrent que le recouvrement local permet de diminuer la charge de la source et le temps de recouvrement lorsque le nombre de récepteurs augmente, et plus particulièrement lorsque le taux de perte est grand.

Détection rapide des pertes

Le service léger de détection rapide des pertes a pour but de réduire les latences de recouvrement. Nous avons effectué des simulations qui montrent que ce service est particulièrement bénéfique à DyRAM. Dans les figures 3.13a et 3.13b, les performances des protocoles DyRAM- et DyRAM+ sont présentés. DyRAM- n'a pas de service de détection rapide, alors que DyRAM+ en bénéficie. Les valeurs du délai de recouvrement sont normalisées par rapport au RTT et sont exprimées en fonction du nombre de récepteurs. En général, le service de détection rapide des pertes permet à DyRAM+ de terminer le transfert plus rapidement: avec $p = 0.25$ et 96 récepteurs par exemple, DyRAM+ peut être 4 fois plus rapide.

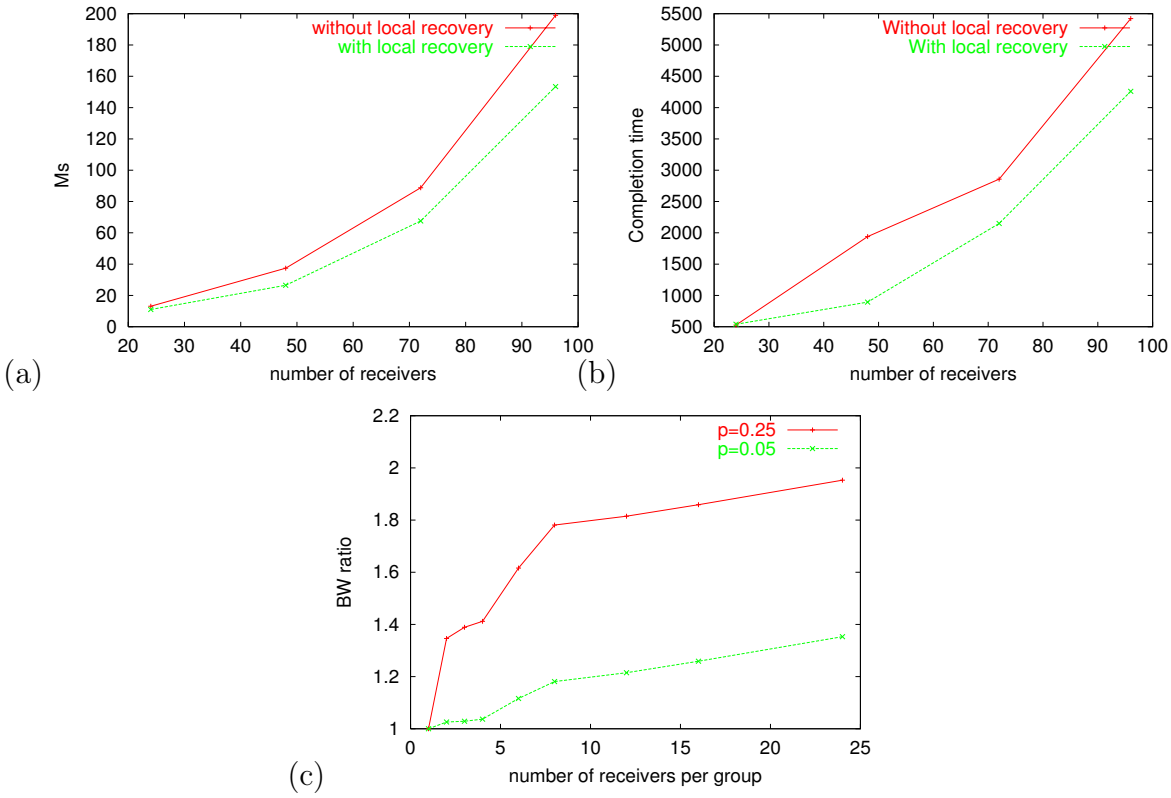

Figure 3.12 (a) Charge à la source, $p = 0.25$. (b) Temps de transfert, $p = 0.25$. (c) Bande passante consommée (rapport).

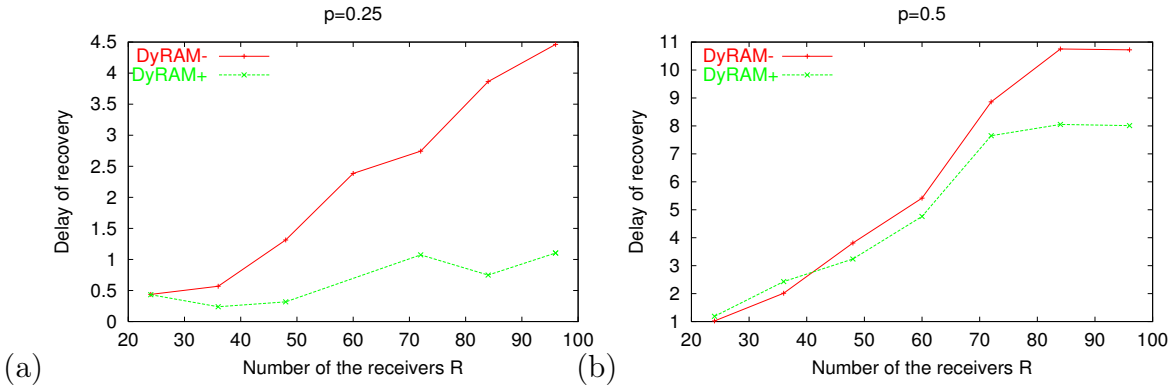

Figure 3.13 Délai de recouvrement normalisé avec (a) $p = 0.25$ et (b) $p = 0.5$

Contrôle de la congestion

Le contrôle de congestion est un domaine ardu en multicast car il est difficile d'une part de récupérer et de prendre en compte l'état global de l'ensemble des récepteurs, et aussi d'autre part de satisfaire l'ensemble des récepteurs lorsqu'il y a hétérogénéité (comme c'est très souvent le cas).

Dans une démarche similaire à celle que nous avons adoptée pour proposer les services actifs précédents, nous avons proposé et évalué un service actif permettant d'améliorer l'estimation des RTTs des récepteurs vers la source. Les routeurs actifs dans l'arbre de multicast sont impliqués dans cette tâche, et vont d'une part estimer les RTTs vers leurs récepteurs fils et le RTT vers leur routeur actif père, ou la source s'ils n'ont pas de père; et d'autre part, effectuer l'aggrégation des informations de RTTs afin de ne propager qu'une valeur pertinente vers la source. Comme dans RMANP [3] ou NCA [51], nous bénéficions de l'arbre physique de multicast pour effectuer l'aggrégation des RTTs, au contraire de TRAM [18] ou MTCP [96] qui utilisent, et doivent donc maintenir, un arbre logique.

L'algorithme AMCA [69] (Active-based Multicast Congestion Avoidance Algorithm) que nous proposons permet avec ce service léger de prédire de manière très juste la congestion (par observation de la variation des RTTs) et par conséquent d'éviter les pertes de paquets. L'approche est compatible avec TCP comme le montre les simulations représentées dans la figure 3.14.

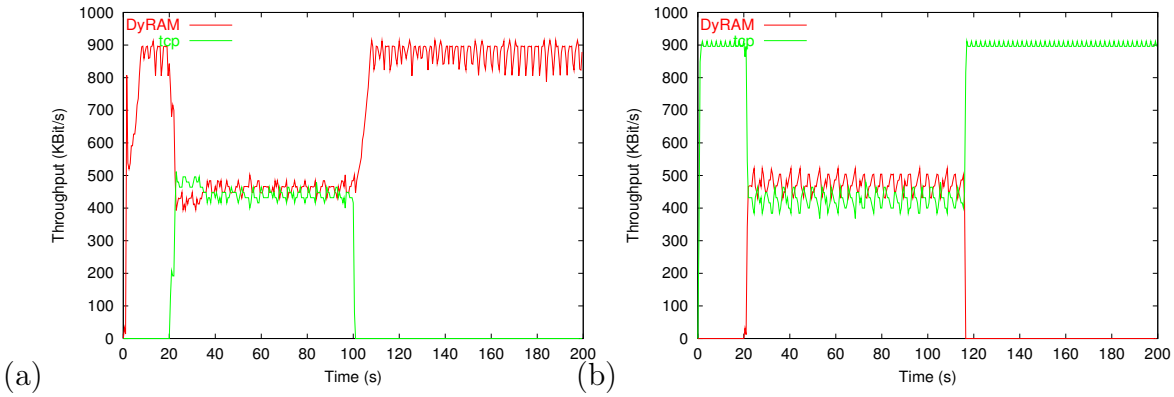

Figure 3.14 Contrôle de congestion AMCA. (a) TCP en tâche de fond, (b) DyRAM en tâche de fond.

L'approche proposée est similaire à celle adoptée par TCP Vegas [9] mais souffre moins du problème du re-routage qui peut faire croire à une congestion à cause d'une augmentation du RTT. En effet, l'approche utilisée consiste à mesurer la variation de RTT section par section (une section est un ensemble de liens conduisant/provenant vers/d'un routeur actif). Les routeurs actifs aggrègent et propagent les variations mesurées section par section.

3.7.3 Implémentation

Un prototype de DyRAM a été développé pour tout d'abord quantifier le coût des services actifs proposés, et surtout le coût de traversé d'un routeur actif. Les routeurs actifs sont des PCs (PC Pentium II 400 MHz, 512 Ko cache, 128Mo RAM, Linux 2.4, Java 1.3.1) avec l'environnement TAMANOIR [35]. Cet environnement orienté vers la haute-performance est développé dans notre équipe par L. Lefèvre et J. P. Gelas. Les résultats qui ont été publiés dans [72][8] montrent que le temps de traversé d'un paquet de données dans un routeur actif est de l'ordre de $20\mu s$, celui de traitement des NACK et des paquets de retransmission est compris entre $120\mu s$ et $135\mu s$. Ces temps sont améliorables avec des processeurs plus rapides et sont donc très encourageants. L'élection dynamique d'un retransmetteur prend un temps variable qui dépend d'une part du nombre de récepteurs sous le routeur actif, et d'autre part du nombre d'itérations nécessaires pour trouver le bon retransmetteur.

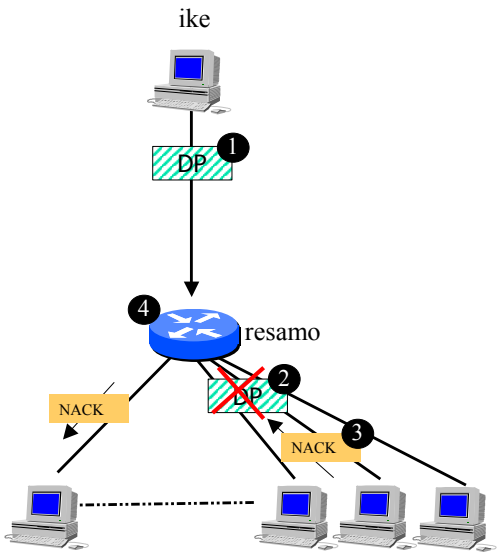

Figure 3.15 Topologie pour la mesure du coût d'élection.

La figure 3.15 montre la topologie de test et les coûts de l'élection sont illustrées dans la figure 3.16. Dans cette dernière figure, 5 courbes correspondants à un nombre croissant de récepteurs fils pour un noeud actif (de 5 à 25) sont représentées. Chaque courbe montre le temps passé pour l'élection lorsque le récepteur élu est en position $i \in [0, max]$.

[8] M. Maimour, J. Mazuy, C. Pham, "The Cost of Active Services in Active Reliable Multicast", *Proceedings of IEEE 4th Annual International Workshop on Active Middleware Services (AMS 2002), July 24-26, 2002, Edinburg, UK, pp67-72.*

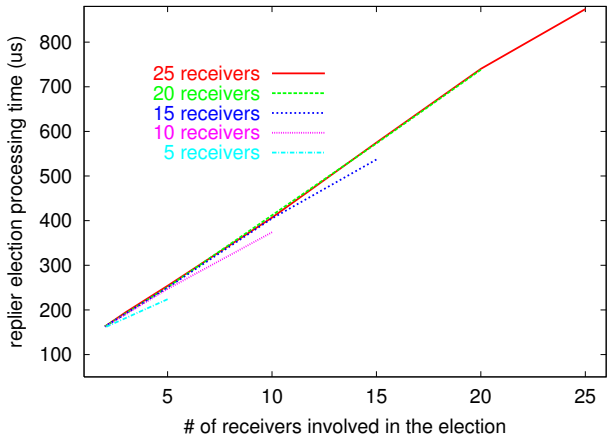

Figure 3.16 Coût d'une élection.

Dans le cadre du projet RNTL E-Toile sur la grille de calcul, nous implémentons un support similaire à `ftp` pour utiliser le protocole de multicast DyRAM à des fins de distribution de code et de données pour le calcul scientifique sur une grille de calcul. Ce point sera discuté plus en détail dans la section 3.8.

3.8 LE MULTICAST SUR LES GRILLES DE CALCUL

3.8.1 La grille de calcul

L'idée maîtresse d'une grille de calcul [30] est d'aggréger et d'utiliser un super calculateur virtuel constitué grâce à l'interconnexion performante des ressources de calcul de sites distribués géographiquement, et disposant de machines parallèles ou de grappes de machines. La globalisation des ressources informatique répond à la fois à un souci économique où l'on cherche à partager des équipements de calcul coûteux mais aussi à des soucis correspondants au passage au facteur d'échelle pour l'étude de problèmes de plus en plus grand.

De telle infrastructure de grille sont de véritables défis mais peuvent aussi être la réponse au besoin de puissance de calcul, sans cesse croissant, d'applications scientifiques dans des domaines très variés: physique des haute-énergies, prédiction météorologique, problèmes mathématiques et combinatoires, exploration du génôme,... Ces dernières années ont vu émerger un certain nombre de logiciels "middleware" et d'environnements permettant l'accès à ces ressources distribuées: Condor [63], Globus [31], Legion [42] pour n'en citer que quelques uns. Des projets nationaux et internationaux ont été initiés sur toute la planète pour étudier le potentiel du calcul de type grille (*grid computing*): DataGrid (www.eu-datagrid.org), EuroGrid (www.eurogrid.org), GriPhyn (www.gryphyn.org), PPDG (www.ppdg.net), E-Toile (http://www.urec.cnrs.fr/etoile/e-toile.htm)...

L'infrastructure réseau privilégiée sur laquelle repose actuellement les grilles est une interconnexion de réseaux IP avec des liens à très haut-débit (2.5 à 10 Gbps).

Cependant, dans certain cas cette interconnexion peut être simplement l'Internet mondial. Cependant, ce type d'infrastructure souffre de grandes faiblesses, en particulier, au niveau de la fourniture de garantie de QoS et du support de communication de groupe. De nouveaux protocoles adaptés aux besoins spécifiques des applications doivent être mis en oeuvre. La très grande diversité des flux en terme de taille de message, de communications (point à point ou multicast), type de message (contrôle ou données), requière une intelligence supplémentaire dans le réseau (avec du monitoring par exemple [92]) pour mieux supporter les besoins de la grille.

Dans le cadre de l'implication de notre équipe dans les problèmatiques GRID, je suis coordinateur d'une ACI GRID sur les services réseaux pour la grille de calcul. Notre équipe propose dans cette action de développer une architecture active dédiée à la grille et d'étudier le problème du transport de données sur la grille. Nous regardons plus précisément les problèmes liés à l'optimisation des protocoles de transport (TCP, travaux de Pascale Primet) et ceux des communications multipoints (multicast). La page web du projet est `http://www.ens-lyon.fr/~cpham/PROJETS/ACIgrid.htm`.

3.8.2 Le support du multicast fiable

Il est assez facile de définir 3 catégories d'applications que l'on trouve de manière récurrente sur les grilles de calcul et pouvant utiliser un support multicast:

1. Applications interactives
   - simulations distribuées : de type DIS, HLA ou bien couplage de code
   - visualisation distante, généralement en temps réel
2. Gestion de base de données
   - réplication de bases de données/fichiers/objets: ex: Datagrid où chaque fichier fait environ 1 à 2 Go, au total quelques Po.
   - maintien de la cohérence, synchronisation, gestion des caches...
3. Distribution de codes et de données
   - applications du type ferme de calcul, massivement parallèle (seti@home...)

Du point de vu académique, toutes ces applications sont intéressantes. Du point de vue pratique, la visualisation à distance n'est pas encore vraiment une demande forte (surtout de la part des industriels qui sont déjà réticents à déporter du calcul). Pour les bases de données, une distribution à l'échelle mondiale n'est pas

quelque chose d'indispensable pour les utilisateurs. Les applications les plus dominantes actuellement sont la diffusion de programmes et de données pour le calcul scientifique. Ces applications aujourd'hui impliquent un nombre restreint de participants, et il n'est pas clair qu'il existe un vrai besoin pour de très grands groupes. Cependant, même avec une dizaine de sites, les quantités de données échangées sont si grandes que le temps de transfert peut rapidement devenir problématique. De plus, il existe un besoin croissant de pouvoir exécuter des applications distribuées plus interactives ou réalisant plus d'opérations collectives plus complexes. C'est dans ce contexte qu'un support de diffusion performant est nécessaire.

3.8.3 Vers le concept d'une grille de calcul active

Avec une vue logique plus proche d'un système distribué que d'une pure infrastructure de communication, il est assez tentant d'étendre pour la grille les fonctionnalités assez basiques que l'on trouve actuellement dans l'Internet. Cela afin d'offrir les services à valeur ajoutée par rapport aux services de routage de base qui n'ont guère évolués depuis ces 10 dernières années. Cette initiative est un peu similaire à celles de la communauté pair-à-pair (P2P) ou à celles des *overlays* et des *web-services*.

Il est possible d'aller un peu plus loin et d'intégrer plus profondément des services supplémentaires dans l'infrastructure réseau et voir le réseau d'interconnexion de la grille comme une ressource intelligente et programmable supplémentaire, au même titre que les ressources de calcul des sites de la grille, capable de réaliser un certain nombre d'opérations pour l'application. La figure 3.17 illustre par exemple comment le réseau peut contribuer à réaliser une opération de calcul du maximum.

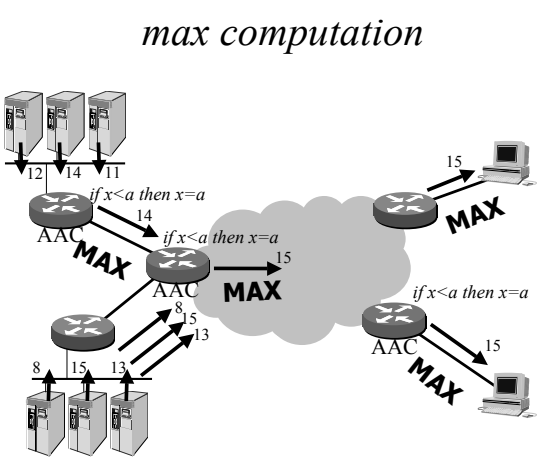

Figure 3.17 Exemple d'une opération *max* avec support du réseau.

Comme il est difficile de prévoir ce que sera une application typique de la grille, s'il y en aura une, une telle approche peut permettre une meilleure et rapide adéquation

entre les besoins de l'application et les ressources proposées. La réalisation pratique d'une telle architecture peut reposer sur des routeurs actifs placés à la périphérie du réseau d'opérateur (voir figure 3.18). Dans ce cas, nous introduisons l'acronyme AAC qui signifie *Application-Aware Component* et qui sont des éléments basés sur des routeurs actifs. Les principes généraux d'une telle architectures ont été décrits dans [58][9] et [8][10]. Plus d'information sur le concept P2P peuvent être trouvées sur `http://www.peer-to-peerwg.org` et `http://openp2p.com`.

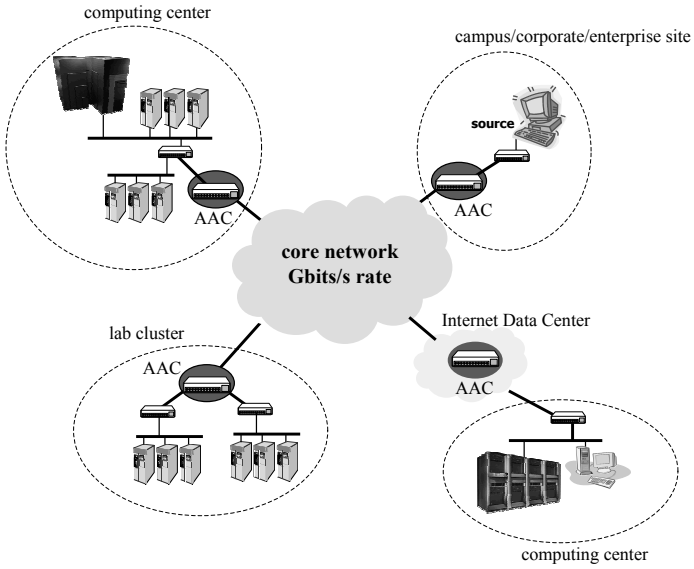

Figure 3.18 Une architecture de grille active.

3.8.4 DyRAM sur la grille de calcul

Il peut y avoir une grande variété d'infrastructures de grilles mais dans la plupart des cas elles utilisent des architectures réseaux assez similaires: des ressources locales de calcul sont connectées ensembles (Fast/Giga Ethernet, Fiber Channel, SCI, Myrinet...) et accèdent au reste du monde (l'Internet, les autres sites) au travers d'un ou de plusieurs routeurs. Nous pouvons donc supposer que les ressources de calcul sont distribuées au travers d'un réseau de type Internet avec un réseau très haut-débit dans le cœur (typiquement des réseaux déployés par les opérateurs) et plusieurs réseaux d'accès à plus faible débit (jusqu'à 1Gbits/s) à la périphérie. La figure 3.18 représente une telle infrastructure. Pour des raisons de simplicité, j'ai

[9]L. Lefèvre, C. Pham, P. Primet, B. Tourancheau, B. Gaidioz, J. P. Gelas, M. Maimour, "Active Networking Support for The Grid", *Proceedings of the third International Working Conference on Active Networks (IWAN'01), September 30 and October 1-2 2001, Philadelphia, USA, pp16-33.*

[10]F. Bouhafs, B. Gaidioz, J.P. Gelas, L. Lefèvre, M. Maimour, C. Pham, P. Primet, B. Tourancheau, "Designing and experimenting an active grid architecture", Future Generation Computer System, version étendue de [58]. A paraître 2004.

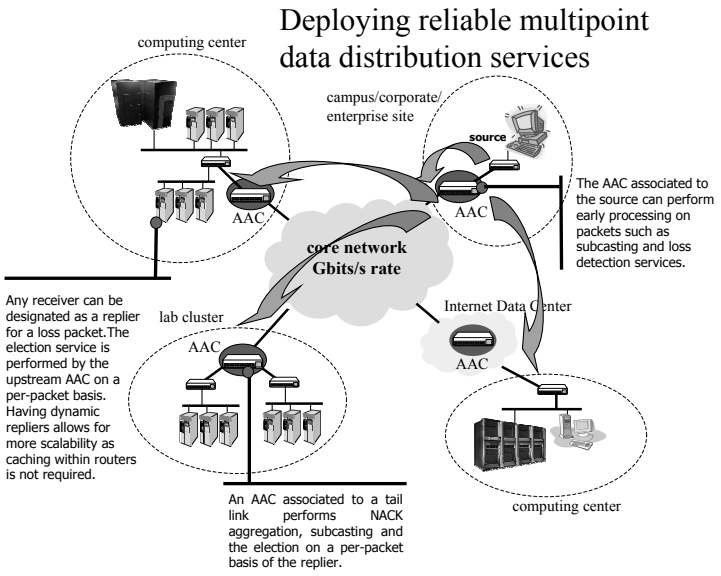

Figure 3.19 Déploiement de DyRAM sur une grille de calcul.

représenté chaque réseau d'accès par un routeur mais dans le cas réel, un tel réseau contiendrai plusieurs routeurs.

Les AACs pourront se situer dans les *Internet Data Center* (et donc déployés par un opérateur, mais cela est assez aléatoire pour l'instant) ou dans un domaine privé, à l'initiative d'un centre de calcul ou d'un campus par exemple. Dans le cas d'un déploiement de grilles de calcul, c'est le scénario le plus adapté. Les services actifs liés à DyRAM pourront soient se trouver chargés au préalable dans les AACs (par l'administrateur du site local), ou bien téléchargés à partir de la source ou d'un *repository* (comme c'est déjà possible avec TAMANOIR), voir figure 3.19. En principe, le routeur le plus proche du réseau de coeur est un bon candidat pour être AAC. De plus, pour bénéficier d'une faible latence de recouvrement, il est souhaitable d'avoir des AACs proches des ressources de calcul. Dans une telle architecture, les AACs qui se trouvent donc à la périphérie font tous l'aggrégation des NACKs et le subcast. Du coté de la source, des travaux préliminaires détaillés dans [71] ont montré que la détection précoce des pertes n'est bénéfique que si elle est effectuée assez proche de la source. Par conséquent, un AAC proche de la source est le meilleur candidat pour installer les services de détection précoce des pertes. Les autres AACs réalisant uniquement l'élection pour le recouvrement local. Nous avons étudié et détaillé ces aspects dans le papier #5[11] qui peut être trouvé en annexe à la fin de ce document.

[11] M. Maimour, C. Pham, "Experimenting Active Reliable Multicast on Application-Aware Grids", *Journal of Grid Computing, Version étendue de [73]. A paraitre 2004.*

3.8.5 Valorisation et projets

Ces travaux ont fait l'objet de plusieurs communications dans des conférences internationales, mais également de séminaires chez SUN Labs et de démonstrations à IPDPS 2003 (stand ACI GRID). Des démonstrations dans le cadre du projet RNTL E-Toile et RNRT VTHD++ ont également été effectuées et ont prouvées la faisabilité du concept. Le prototype de DyRAM précédemment développé par un étudiant de maîtrise (J. Mazuy) a été revu en profondeur pour fournir à la fois une API et une librairie pour l'utilisation de DyRAM par des applications. Un programme similaire à `ftp` est également développé pour une utilisation dans le projet E-Toile par les partenaires industriels. Ce développement est effectué par F. Bouhafs, ingénieur expert INRIA, recruté dans le cadre du projet E-Toile, et travaillant sous ma direction.

3.9 CONCLUSIONS DU CHAPITRE ET PERSPECTIVES

Le domaine de recherche associé au multicast est très vaste. J'ai orienté mes recherches ces 3 dernières années vers la fiabilité et les solutions actives. Ces approches avaient déjà été étudiées depuis environ 2 ans aux US, mais j'ai senti que des contributions étaient possibles en proposant des services légers novateurs permettant plus de performance et une plus grande résistance au facteur d'échelle. Ces travaux se sont ensuite bien intégrés dans le contexte actuellement très porteur des grilles de calcul.

Ce travail a trouvé le support de 3 projets: une ACI GRID à caractère exploratoire, un RNRT VTHD++ et un RNTL E-Toile aux caractères plus appliqués. Cela a donné une dimension différente au travail de recherche, très intéressante car mélangeant la recherche académique avec de la recherche très appliquée. L'enrichissement personnel est très grand, j'ai pu à la fois encadrer une étudiante en thèse et un ingénieur expert (Faycal Bouhafs, INRIA, projet E-toile) sur ces problématiques. Des implémentations réelles existent, qui ont fait ressortir un grand nombre de problèmes pratiques souvent non traités dans des études uniquement théoriques. J'ai par exemple beaucoup regardé les aspect liés à la configuration du routage multicast (PIM-SM, MBGP, DVMRP) pour la construction de démonstrations à grande échelle. Cela m'a notamment permis de mieux connaître les règles d'ingénierie du multicast IP. Pouvoir mener un projet de la phase de réflexion et de conception sur le papier à une implémentation réelle, utilisée par des partenaires industriels est un aspect que je privilégie dans mes orientations de recherche.

Cette recherche a donné les résultats que j'escomptais: elle a montré la faisabilité de l'approche service actif léger pour l'assurance de la fiabilité dans les communications multicast. Des publications, ainsi que des séminaires et démonstrations nous ont permis de diffuser ces résultats sur la scène internationale et aussi au niveau national grâce aux projets industriels. Sans aller jusqu'à dire que nous avons fait le tour du problème, je pense que l'aspect novateur dans le futur sera moindre sur

ce thème. Il reste un énorme travail d'ingénierie afin de tester et comparer les différentes approches de multicast fiable sur une grande échelle. J'espère pouvoir y contribuer avec l'aide d'équipes comme INRIA PLANETE, INRIA ARES, le LSIIT, Renater et d'autres industriels.

Pour en savoir plus...

Des études recensant les problèmes/solutions liés au déploiement du multicast ont été publiées, j'invite les lecteurs à consulter [22, 1, 20] pour de plus amples discussions. En ce qui concerne l'intégration du multicast dans les technologies actuelles de l'Internet (routage inter-domaine, QoS, DiffServ), les lecteurs pourront consulter [106, 98]. Les lecteurs intéressés par le routage multicast peuvent consulter l'habilitation d'Eric Fleury [28] qui pointe sur des travaux de référence. En ce qui concerne le contrôle de congestion, [59] formalise de manière originale ce problème en utilisant le paradigme de Fair Queueing (et donc utilise une approche de bout-en-bout conforme à la philosophie d'Internet) et propose un mécanisme intéressant appelé PLM (Packet-pair Layered Multicast). D'autres papiers sur le contrôle de congestion en multicast peuvent être trouvés sur `http://www-scf.usc.edu/~junsoole/research/multicast_congestion/`.

MES PUBLICATIONS LIEES AU THEME

Journaux

[1] F. Bouhafs, B. Gaidioz, J.P. Gelas, L. Lefèvre, M. Maimour, C. Pham, P. Primet, B. Tourancheau. Designing and Experimenting an Active Grid Architecture. *Future Generation Computer System*, 2004. A paraître.

[2] M. Maimour, C. Pham. Dynamic Replier Active Reliable Multicast (DyRAM). *Journal of Cluster Computing*, 2004. A paraître.

[3] M. Maimour, C. Pham. Experimenting Active Reliable Multicast on Application-Aware Grids. *Journal of Grid Computing*, 2004. A paraître.

[4] M. Maimour, C. Pham. Modélisation et évaluation des protocoles de multicast fiable dans le contexte des réseaux actifs. *Réseaux et Systèmes Répartis-Calculateurs Parallèles (RSR-CP), Numéro spécial sur la Performance et Réseaux et des Systèmes*, 13(6), 2001.

Conférences

[1] M. Maimour, C. Pham. Dealing with heterogeneity in a fully reliable multicast protocol. In *Proceedings of IEEE International Conference On Networks (ICON 2003)*, Sydney, Autralia, September 2003.

[2] M. Maimour, C. Pham. A RTT-based Partitioning Algorithm for a Multirate Reliable Multicast Protocol. Proceedings of the IEEE High Speed Network and Multimedia Communications (HSNMC 2003), Estoril, Portugal, July 2003.

[3] M. Maimour and C. Pham. AMCA: an Active-based Multicast Congestion Avoidance Algorithm. In *Proceedings of the 8th IEEE Symposium on Computers and Communications (ISCC 2003)*, Antalya, Turkey, July 2003.

[4] M. Maimour and C. Pham. A New Active-based Multicast Congestion Avoidance Algorithm. Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE'03), May 4-7, 2003, Montreal, Canada.

[5] M. Maimour and C. Pham. Towards an application-aware communication framework for computational grids. In *Proceedings of the Asian Computing Science Conference (ASIAN 2002)*, Hanoi, Vietnam, December 2002.

[6] M. Maimour, J. Mazuy, and C. Pham. The cost of active services in active reliable multicast. In *Proceedings of the 4th IEEE Annual International Workshop on Active Middleware Services (AMS 2002)*, pages 67–72, Edinburg, UK, July 2002.

[7] M. Maimour and C. Pham. An analysis of a router-based loss detection service for active reliable multicast protocols. In *Proceedings of the 11th IEEE International Conference on Networks (ICON 2002)*, Singapour, August 2002.

[8] M. Maimour and C. Pham. Dynamic replier active reliable multicast (dyram). In *Proceedings of the 7th IEEE Symposium on Computers and Communications (ISCC 2002)*, pages 275–282, Taormina, Sicily, July 2002.

[9] M. Maimour and C. Pham. A loss detection service for active reliable multicast protocols. In *Proceedings of the International Network Conference (INC'2002)*, Plymouth, UK, July 2002.

[10] M. Maimour and C. Pham. An active reliable multicast framework for the grids. In *Proceedings of the International Conference on Computational Science (ICCS 2002)*, volume 2330 of *Lecture Notes in Computer Science*, pages 588–597, April 2002.

[11] L. Lefèvre, C. Pham, P. Primet, B. Tourancheau, B. Gaidioz, J.P. Gelas, and M. Maimour. Active networking support for the grid. In Noaki Wakamiya Ian W. Marshall, Scott Nettles, editor, *IFIP-TC6 Third International Working Conference on Active Networks (IWAN 2001)*, LNCS 2207, pp16–33,October 2001. ISBN: 3-540-42678-7.

[12] M. Maimour and C. Pham. A throughput analysis of reliable multicast protocols in an active networking environment. In *Proceedings of the Sixth IEEE Symposium on Computers and Communications (ISCC 2001)*, pages 151–158, Hammamet, Tunisia, July 2001.

[13] M. Maimour and C. Pham. Active reliable multicast for efficient data distribution on an internet-based grid computing infrastructure. In *Proceedings of the International Conference on Internet Computing (IC'2001)*, pages 437–443, Monte Carlo Hotel, Las Vegas, USA, June 2001.

Rapports de recherche

[1] M. Maimour and C. Pham Dealing with Heterogeneity in an Active-based Multicast Congestion Avoidance Protocol. INRIA research report RR-4796, March 2003. Also available as a LIP/ENS research report under 2003-20.

[2] M. Maimour and C. Pham. A RTT-based Partitioning Algorithm for a Multi-rate Reliable Multicast Protocol. INRIA research report RR-4779, March 2003. Also available as a LIP/ENS research report under 2003-16.

[3] M. Maimour and C. Pham. AMCA : an Active-based Multicast Congestion Avoidance Algorithm. INRIA research report RR-4689, January 2003. Also available as a LIP/ENS research report under 2003-07.

[4] M. Maimour and C. Pham. An Analysis of a Router-based Loss Detection Service for Active Reliable Multicast protocols. INRIA research report RR-4636, November 2002. Also available as a LIP/ENS research report under 2003-06.

[5] M. Maimour and C. Pham. Dynamic Replier Active Reliable Multicast (DyRAM). INRIA research report RR-4635, November 2002 and a LIP/ENS research report under 2003-05.

[6] M. Maimour and C. Pham. A Throughput Analysis of Reliable Multicast Protocols in an Active Networking Environment. TR01-2001. January 2001.

Chapitre 4

Optimisation des sous-systèmes de communication

4.1 INTRODUCTION

Les réseaux de communication sont constitués de liens, de routeurs et des machines terminales. Ces machines peuvent être des stations de travail ou bien des serveurs. Ces serveurs pouvant être soit des serveurs de fichiers (tyoe NFS) dans le cas d'un réseau local, soit des serveurs de contenu dans le cas de l'Internet et du web par exemple. Lorsque l'on parle de performance sur un réseau, on pense généralement d'abord, et dans cet ordre: (*i*) liens physiques, (*ii*) protocoles de transport et, (*iii*) vitesse de la machine terminale. Dans ce dernier cas, rares sont ceux qui voit immédiatement une différence entre vitesse du processeur et performances réelles pour les applications, qui sont généralement bien moins bonnes que si l'on ne considère que la vitesse du processeur. Les raisons de ces différences sont nombreuses: côut des couches protocolaires, mauvaise conception et implémentation des protocoles, coût des communications avec le système d'exploitation et efficacité du sous-système de communication (SSC).

Dans une architecture de machine, c'est le SSC qui a la charge de gérer les entrées/sorties de paquets. Le SSC est normalement une partie générique du système d'exploitation pour la gestion des interruptions, des buffers mémoire, du cache, etc, et n'est donc pas composé du driver de la carte d'interface. Pour nos travaux, nous modifions le SSC et le driver, comme illustrée par la figure 4.1. Les protocoles de haut-niveau tels que IP ou TCP ne sont pas inclus dans le SSC. La carte

d'interface n'est également pas inclue dans le SSC, mais sur les nouvelles générations de cartes programmables, une partie du SSC peut y être déportée et exécutée. L'architecture d'un SSC a un impact important sur les performances réseaux finales d'une machine. Les influences sont parfois subtiles et les interactions souvent très complexes avec les composants matériels tels que le processeur, la mémoire (principale et celle du cache), les différents controleurs (d'interruptions par exemple), les cartes d'interfaces, l'architecture du bus, etc. Ces divers surcoûts d'entrée/sortie dans le SSC peuvent être classés en deux catégories: ceux induits par octet, sollicitant donc le système mémoire, et ceux induits par paquet, sollicitant plus le(s) processeur(s).

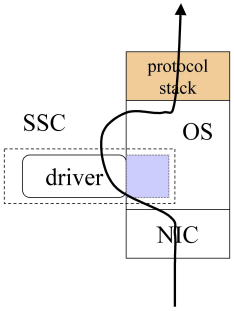

Figure 4.1 Le sous-système de communication, le driver et leur place dans une architecture de machine.

Un des grands problèmes actuels est que les performances des processeurs et des architectures mémoire, bien qu'en constante évolution, augmentent bien moins vite que la performance des réseaux. Par exemple, il semble peu probable que les serveurs de l'Internet, qui, aujourd'hui, utilisent des processeurs de l'ordre du GHz et sont connectés à des réseaux locaux Gigabit Ethernet (1Gb/s), soient équipés de processeurs à 10GHz au moment de l'arrivée sur le marché des réseaux 10 Gigabit Ethernet. La figure 4.2 reprise d'un tutorial du Pr. Nick McKeown illustre bien ce problème.

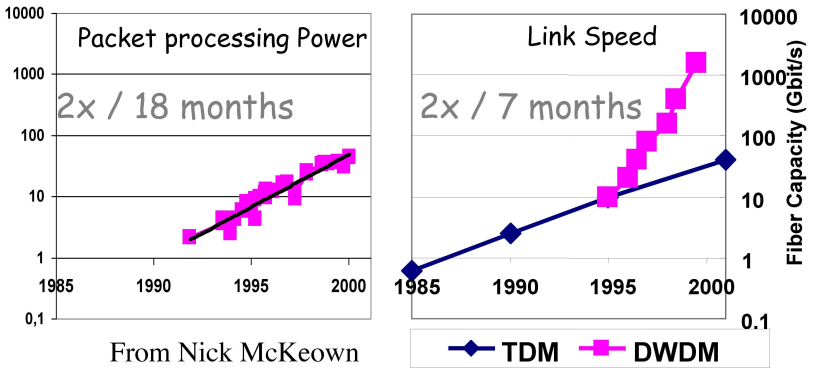

Figure 4.2 Evolution de la vitesse de processeur vs. évolution de la vitesse des liens.

Comme le débit disponible dans les réseaux de coeur des opérateurs (et donc sur certains liens de l'Internet) a très largement augmenté ces dernières années avec le déploiement massif de technique de multiplexage en longueur d'onde (DWDM) sur fibres optiques, le goulot d'étranglement va donc très certainement se déplacer vers les machines d'extrémités auparavant "protégées" par un réseau local de faible capacité. Ces machines, lorsqu'elles sont des machines serveurs sont alors exposées au problème d'échelle qui réduit fortement leur capacité de traitement (baisse du débit sortant, augmentation du temps de réponse) et leur robustesse. Pour illustrer l'extrême dynamicité à laquelle doit faire face un serveur web par exemple la figure 4.3[1] représente le nombre de requête sur le moteur de recherche `Google` du mot-clé `cnn` lors de l'attaque du 11 septembre 2001 sur le WTC aux USA. On peut constater plus de 6000 requêtes par minute pendant une longue période de temps. La figure 4.4 montre les effets dramatiques bien connus de la charge sur un système peu robuste. Dans ce cas, peu importe d'avoir des liens très rapides ou des protocoles de transport très performants, la performance de bout-en-bout n'est pas là!

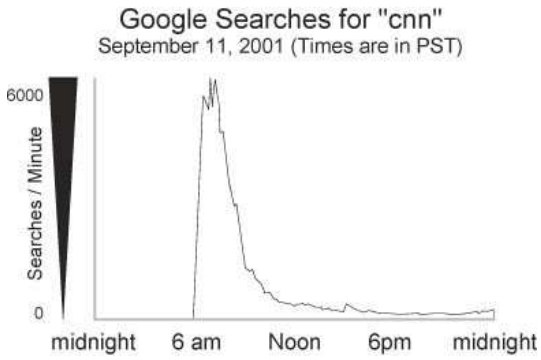

Figure 4.3 Requêtes/minute sur Google pendant l'attaque du 11/9/01.

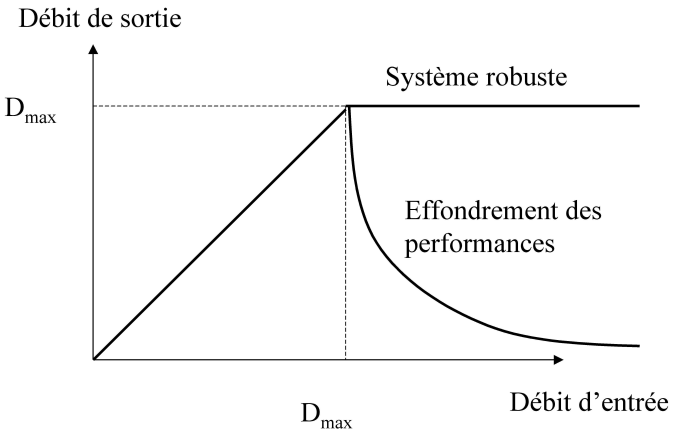

Figure 4.4 Effondrement des performances d'un système.

[1]Que l'on peut trouver sur `http://www.google.com/press/zeitgeist/9-11-search.html`

Dans le cas d'un serveur web, ces problèmes sont très importants. En effet, ces serveurs sont très sollicités et il n'est pas rare d'avoir des milliers de connexions concurrentes (surtout si le site est très populaire!). La robustesse est dans ce cas un critère important, notamment pour la résistance aux attaques visant à rendre indisponibles certains serveurs!

J'ai commencé à m'intéresser à ce thème lorsque j'ai travaillé sur la problématique des clusters et les moyens disponibles pour améliorer les performances des simulations parallèles tournant sur ces architectures. Sensibilisé aux problèmes de la performance et des coûts liés aux SSCs avec la suite logicielle BIP [94] proposée par Loic Prylli (voir chapitre 1), j'ai cherché à optimiser l'envoi et la réception des messages dans les communications entre les machines d'un cluster. Les cartes d'interface réseaux programmables telles que les cartes Myrinet, dont BIP tire profit, m'ont semblé être une solution possible pour accélérer les simulations parallèles. En effet, j'ai pensé qu'il était possible de déplacer un certain nombre de focntionnalités dans la carte d'interface comme l'aggrégation des messages, le calcul de l'horloge minimale, ou bien encore l'envoi des messages de contrôle (null-messages). Eric Lemoine a effectué un stage de DEA sur ce sujet sous ma direction. Les résultats ont cependant montré que la capacité de traitement des cartes d'interface par rapport à la vitesse des processeurs ne favorisait pas les gains en performance. Par contre, le résultat du stage a aussi montré qu'effectuer un traitement léger au plus près du fil permettait de gagner en robustesse pour certaines applications. Eric Lemoine a continué en thèse sous ma direction (et co-encadré par L. Lefèvre à 50%) sur ces perspectives avec un financement CIFRE en collaboration avec SUN Labs Europe à Grenoble. Par la même occasion, une coopération forte s'est faite avec cette entreprise. La thèse d'Eric se focalise sur l'optimisation du sous-système de communication dans les machines serveurs. Il a pour l'instant plus particulièrement regardé le cas des serveurs web sur des architectures de machines multi-processeurs.

Organisation du chapitre

Ce chapitre présente les problèmes auxquels nous nous attaquons et résume les recherches effectuées dans la cadre de la thèse d'Eric Lemoine et les résultats qu'il a obtenus. Ce chapitre est organisé de la manière suivante: la section 4.2 liste les différents coûts qui sont induits par les systèmes de communication et les principaux travaux qui ont été faits pour optimiser de tels systèmes. La section 4.3 présente les optimisations sur les machines serveur multi-processeurs. La section 4.4 présente l'approche KNET proposée par Eric Lemoine avec des fonctions de classification sur la carte d'interface réseau. Les premiers résultats sont présentés dans la section 4.5.

4.2 LES COÛTS D'ENTRÉE/SORTIE DANS LES SOUS-SYSTÈMES DE COMMUNICATIONS

Comme précédemment mentionné, les coûts peuvent être classés en deux catégories: ceux induits par octet, sollicitant donc le système mémoire, et ceux induits

par paquet, sollicitant plus le(s) processeur(s). C'est une raison par exemple pour laquelle on cherche à augmenter la taille des paquets pour réduire les coûts par paquets (avec les Jumbo frames dans le cas de GigaEthernet par exemple). Les opérations qui induisent un coût par octet sont par exemple (source [24]):

- le transfert du device vers la mémoire (DMA, copies),
- les API, qui peuvent plus ou moins engrendrer des copies intermédiaires de données lors du passage d'un domaine à un autre (*kernel* vers *user* par exemple),
- les manipulations sur la partie donnée des paquets (checksum par exemple).

Ceux qui induisent un coût par paquet sont (source [52]):

- la manipulation des structures de données telles que celles associées aux sockets de communication,
- la gestion des buffers mémoire (création, destruction, concatenation par exemple)
- les opérations du système d'exploitation telles que le scheduling, la synchronisation, les interruptions logicielles,
- les opérations spécifiques aux protocoles telles que la manipulation de l'entête, l'encapsulation, etc.

Il existe des solutions pour réduire ces coûts. Par exemple, pour réduire le coût par paquet lié aux interruptions matérielles, il est possible de faire ce qu'on appelle de l'*interrupt batching* ou *interrupt coalescing* qui consiste à utiliser une interruption pour récupérer plusieurs paquets provenant de la carte d'interface. Cette solution ajoute néanmoins de la latence dans la récupération des messages puisque ceux-ci doivent attendre entre 2 interruptions. Une variante consiste à déclencher un mode de scrutation (*polling*) sur réception d'une interruption jusqu'à ce qu'il n'y ait plus de paquets en attente. Cette solution proposée dans [23, 77] (la solution de [77] a ensuite été implémentée dans Linux [100]) permet pendant de forte charge à ne pas utiliser d'interruptions matérielles, évitant ainsi un effondrement des performance.

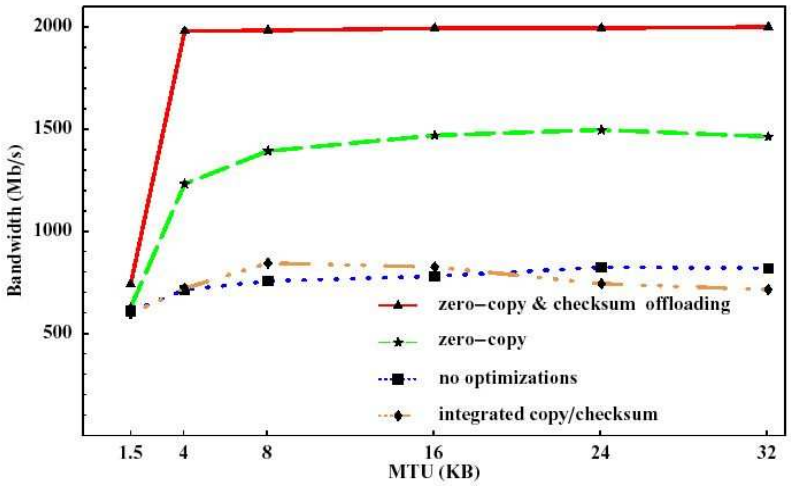

Figure 4.5 Performance de TCP avec différentes optimisations, figure reprise de [17].

Une autre solution (complémentaire) consiste aussi en l'utilisation de paquets plus grands. Cependant cette solution n'est pas extensible indéfiniment. Pour réduire les coûts associés aux mouvements de données, des techniques appelées *zero-copy* permettent d'éviter les copies inutiles lors du passage du paquet dans le SSC et dans les piles protocolaires. Ces techniques ont été appliquées avec succès pour BIP et aussi dans [53, 17] entre autres. Wang et Liu [110] ont écrit un rapport récapitulant les principaux problèmes posés par les SSC dans un environnement haut-débit et les solutions qui peuvent être appliquées. J'invite le lecteur à consulter ce rapport pour obtenir plus de précisions, notamment sur les différentes méthodes pour réduire les coût du checksum que je n'ai pas détaillées dans ce document.

Pour illustrer l'impact de ces coûts, la figure 4.5 issue de [17] montre le débit de TCP au dessus d'un réseau Myrinet avec différentes optimisations. On peut voir par exemple que les gains peuvent dépasser 100% avec des techniques de zero-copy et de déport du checksum.

4.3 AMÉLIORER L'UTILISATION DES MACHINES SERVEURS MULTI-PROCESSEUR

4.3.1 Interruptions matérielles et threads

Dans toutes les architectures de machines, une arrivée de paquet génère une interruption matérielle (Receive INTerrupt, RINT) qui est prioritaire. Généralement, la carte d'interface effectue un transfert DMA pour mettre le paquet dans l'anneau de réception du driver. Le traitement a effectuer ensuite est réalisé dans ce que l'on appelle la routine RISR (Receive Interrupt Service Routine). Selon la conception du SSC, cette routine peut soit copier le paquet vers des zones mémoires de l'OS (et donc la copie se fait dans le contexte du RINT), voir figure 4.6a; soit uniquement

mettre à jour une file de descripteurs, voir 4.6b. Dans les 2 cas, le traitement du paquet pour remonter jusqu'à l'application (ou du moins jusqu'à la socket buffer) est du ressort d'une interruption logicielle programmée plus tard.

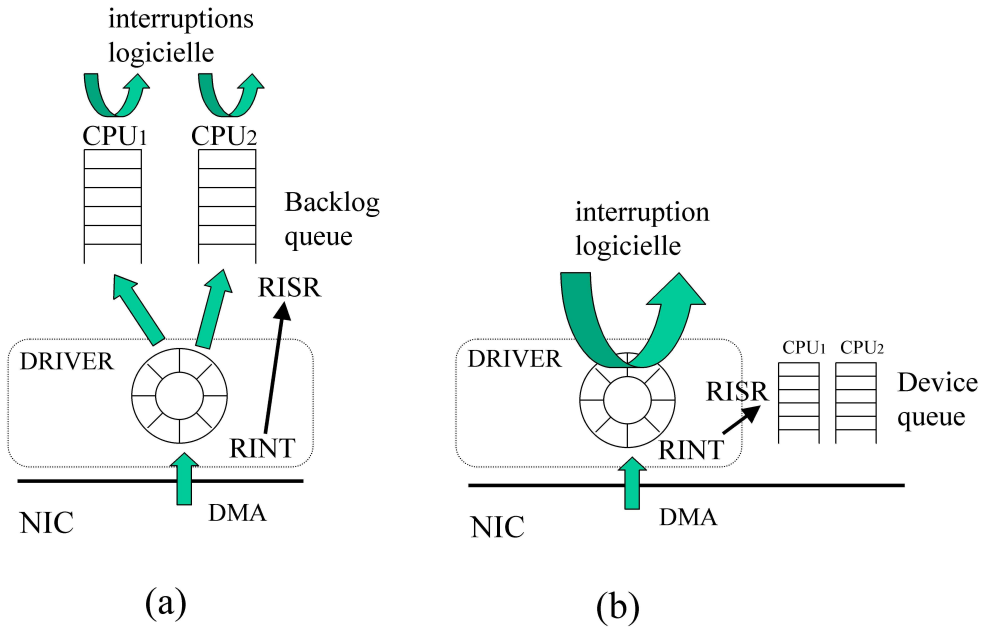

Figure 4.6 (a) copie du paquet dans le contexte du RINT, (b) copie du paquet dans le contexte d'une interruption logicielle.

Le mode de fonctionnement illustré par la figure 4.6a a été longtemps utilisé dans la plupart des architectures de communication. Il pose cependant un problème fort, celui du Receive Livelock qui, à forte charge, fait écrouler les performances du système (voir figure 4.4). Dans la figure 4.6b, le traitement effectué pendant le RINT est minimal et le plus gros du traitement est effectué dans une interruption logicielle par le biais d'un thread dédié. Dans [100], les auteurs implémentent une telle méthode dans Linux et le traitement effectué pendant le RINT consiste uniquement à mettre dans une file (associée à un CPU i) l'identificateur du device effectuant la requête, à désactiver les RINT et à programmer une interruption logicielle (voir figure 4.6b). C'est cette dernière qui récupèrera les paquets à partir de l'anneau de réception en faisant de la scrutation puis à rétablir les RINT après avoir récupérer et traiter tous les paquets en attente (en éliminant ainsi aussi le besoin de faire de l'*interrupt batching* et ses problèmes de latence).

Cette méthode implémentée dans Linux, que les auteurs appellent NAPI, est efficace sur des systèmes mono-processeur, mais présente des inconvénients sur les systèmes multi-processeurs. En effet, la scrutation permet à un processeur de bien récupérer et traiter tous les paquets en attente jusqu'à épuisement de l'anneau de réception, mais empêche aussi tout autre processeur présent dans la machine de récupérer des paquets. Dans un système multi-processeur, NAPI sérialise les opéra-

tions d'un processeur à un autre. La figure 4.7 illustre ce phénomène pour une machine à 4 processeurs. De plus, rien n'empêche les paquets d'une même connexion (TCP par exemple) d'être traités par des processeurs différents, entraînant ainsi des coûts supplémentaires dus à la perte de localité des données dans les caches.

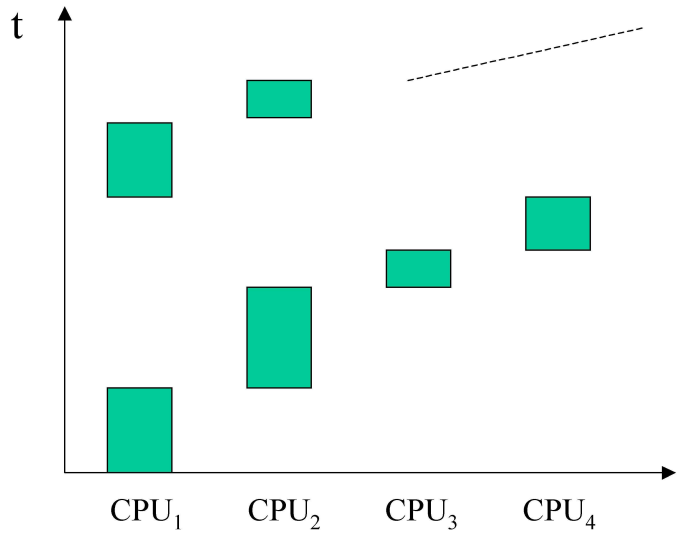

Figure 4.7 Sérialisation des accès à l'anneau de réception par les processeurs.

4.3.2 Augmenter les performances des protocoles de plus haut-niveau sur ces architectures

Sur les machines multi-processeurs, il existe de multiples moyens d'optimiser le traitement des protocoles de plus haut niveau: meilleure implémentation, prédiction des en-têtes, utilisation de pipelines, parallélisation des traitements. Nous nous sommes plus particulièrement intéressés sur le dernier point même si cette idée n'est pas récente puisqu'elle a été expérimentée dès les années 90 [47]. Depuis lors, un grand nombre de travaux ont porté sur ce sujet (des synthèses pouvant être trouvées dans [43, 27, 79, 118, 119]). Il est possible de paralléliser les traitements à plusieurs niveaux. La figure 4.8 issue de [118] montre les différentes alternatives: au niveau des (*i*) fonctionnalités, (*ii*) couches, (*iii*) paquets et, (*iv*) connexions.

Dans les architectures récentes de machines, qui possèdent quand même une capacité de traitement très supérieure à ce que l'on trouvait il y a 5 ans, le parallèlisme au niveau du paquet ou de la connexion est le plus indiqué. En effet, paralléliser au niveau de la fonction, ou des couches, lorsque les processeurs sont rapides et que les fonctions/couches sont en nombre limités ne conduit pas à des performances optimales, sans compter les difficultés à réaliser ce type de parallélisation sur une architecture de communication en couches. La parallélisation au niveau du paquet offre de bonnes performances, mais celles-ci peuvent être limitées par d'une part, les synchronisations et les locks sur la mémoire (liés à l'implémentation), et d'autre part par la necessité de réordonner les paquets traités par différents processeurs

(liés au protocole, TCP par exemple). Pour ces raisons, Eric Lemoine a surtout regardé le parallélisme sur les connexions qui exploite naturellement l'indépendance (dans la plupart des cas) des connexions entre elles. En choisissant un parallélisme sur les connexions, les choix d'architectures sont également multiples. La figure 4.9 également issue de [118] montre les différentes alternatives possibles.

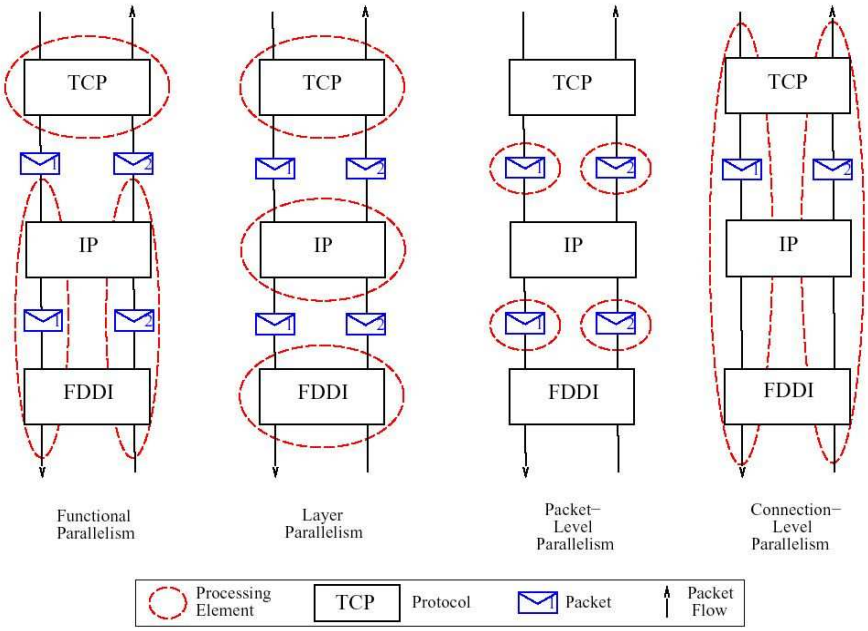

Figure 4.8 Différentes alternatives pour paralléliser les traitements, figure reprise de [118].

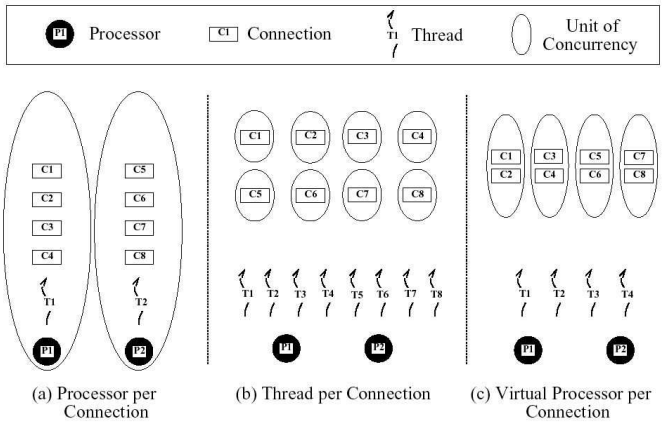

Figure 4.9 Différentes alternatives pour paralléliser au niveau des connexions, figure reprise de [118].

En ce qui concerne le choix entre processeur par connexion (avec multiplexage des connexions bien sûr) et thread par connexion, [118] montre que le premier choix affiche plus d'équité, surtout lorsque le nombre de connexions est grand. Bien que ce

résultat soit très dépendant de l'implantation, il est effectivement plus ardu d'obtenir de l'équité avec une approche thread par connexion que processeur par (groupe de) connexion(s).

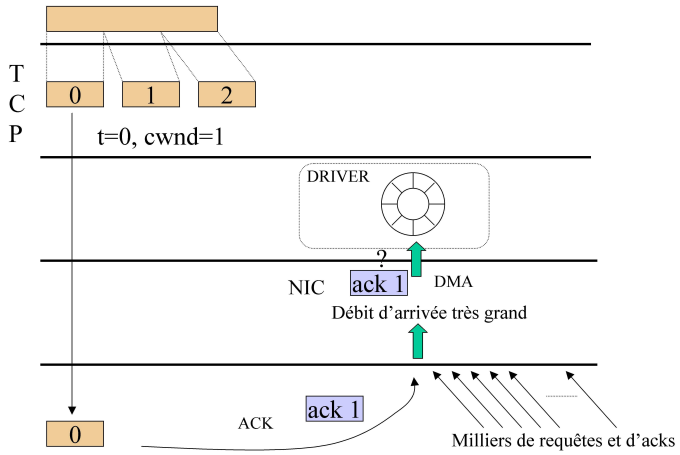

Figure 4.10 Illustration du problème de remontés des ACKs.

4.4 LA PROPOSITION KNET

Le système KNET proposé cherche à optimiser l'utilisation des machines multi-processeurs en augmentant l'efficacité et la robustesse. La robustesse est augmentée en évitant le receive-livelock par une méthode similaire à celle proposé dans [77] et implémentée dans NAPI [100]. Cependant, au contraire de NAPI où la robustesse se fait au détriment de l'efficacité, KNET cherche également à augmenter celle-ci en autorisant plusieurs processeurs à récupérer et à traiter les paquets entrant. C'est un point très important pour les serveurs (web ou fichiers) car un protocole de transport comme TCP est bien connu pour avoir un comportement dit *ACK-clocked*, c'est-à-dire que les segments de données ne sont envoyés du serveur que si les accusés de réception arrivent bien. Un retard trop important d'un ACK[2] peut faire chuter le débit en émission à cause du contrôle de congestion qui sera instauré (le retard ayant été interprété comme une perte, dans le meilleur des cas on divise la fenêtre de congestion par deux, dans le pire des cas, celle-ci sera réinitialisée à 1 pour ensuite faire un *slow-start*). La figure 4.10 illustre ce problème en montrant les interactions avec le sous-système de communication. Dans cet exemple, le serveur a un fichier à envoyer qui est découpé en 3 paquets. Comme c'est une nouvelle connexion, TCP envoie le premier paquet, et met en attente les 2 autres (paquet 1 et 2). Le paquet arrive au client, qui renvoie un ACK(0)[3]. Si l'on utilise un système comme NAPI, ce

[2]Eric Lemoine a vérifié sur des expérimentations que ce phénomène était très visible, et que le serveur pouvait passer un temps assez long en attente d'un ACK qui avait été soit perdu soit retardé, surtout lorsque le RTT est grand. La perte d'un ACK étant provoquée par une saturation des anneaux de réception à cause d'une mauvaise gestion des remontées de paquets.

[3]En réalité, TCP raisonne en octet et non en paquet.

n'est pas plusieurs processeurs qui récupèrent et traitent les paquets entrant mais un seul processeur i à un instant donné, comme illustrée précédemment par la figure 4.7. Si le débit d'entrée est très grand car il y a des milliers de connexions simultanées, alors l'ACK(0) peut très bien être rejeté et perdu par faute de place dans l'anneau de réception. Le serveur n'entre pas dans un état d'effondrement car le processeur fait des traitements utiles, mais le débit des connexions sortantes est très affecté. En permettant la remontée et le traitement en parallèle des paquets sur les processeurs, les anneaux se vident plus vite.

KNET utilise un parallélisme au niveau de la connexion afin de limiter les problèmes de contentions et de locks sur les structures mémoire des protocoles (blocs de contrôle de TCP entre autres). Un des points clés de KNET est d'utiliser une classification des paquets au niveau de la carte d'interface. Cette classification permet d'aiguiller très tôt un paquet vers une file d'attente associé à un processeur donnée. De cette manière, le parallélisme se fait bien au niveau de la connexion et permet de préserver la localité des données.

4.4.1 Classification sur la carte d'interface

La classification est un point important dans la proposition KNET, non pas par son importance théorique, mais par les effets qu'elle produit. La classification en général est également un domaine où de nombreuses recherches ont été effectuées, notamment pour les routeurs de l'Internet et le problème de consultation de la table de routage (*lookup*) et pour la Qualité de Service. Des propositions très similaires à nos travaux ont aussi été faites [34, 78, 99].

Dans KNET, la classification est effectuée par la carte d'interface et nous utilisons une carte programmable (Myrinet) pour le faire. Le fait que la carte soit programmable ne doit pas être vu comme une contrainte ou une nécessité. Il faut plutôt voir ces recherches du point de vue suivant: **"pour améliorer l'efficacité et la robustesse, que peut-on ajouter comme fonctionnalités aux cartes d'interfaces actuelles?"**. En effet, beaucoup de cartes d'interfaces actuelles réalisent déjà de nombreuses fonctions auparavant effectuées par le processeur hôte: calcul du checksum, cryptage, matching de tag pour le cluster computing, etc. Il existe de plus de nombreuses propositions et produits pour complètement coder sur la carte des protocoles de haut-niveau comme TCP (une recherche sur `www.google.com` avec "TCP Offload Engine (TOE)" comme mot-clés donne plusieurs produits commerciaux!).

Effectuer de la classification très tôt sur la carte d'interface permet plus de robustesse que de l'effectuer dans le noyau (dans le driver par exemple). Dans [34, 78, 99], la classification est effectuée dans le noyau par le RISR. Le principal inconvénient de ce type d'approche est d'ouvrir une brèche pour les *receive-livelock* car il est très dangereux d'autoriser les interruptions pendant le traitement des paquets lorsque la charge est forte.

4.4.2 Fonctionnement de KNET

Le fonctionnement général de KNET est représenté dans la figure 4.11. Le driver possède 1 anneau de réception par processeur et un paquet entrant est mis dans un des anneaux grâce à la classification effectuée par la carte. Comme précédemment indiqué, les paquets d'une même connexion sont mis dans le même anneau. Le RINT génèré par la réception d'un paquet (si cette interruption n'est pas masquée) déclenche l'exécution du RISR qui va mettre le device dans la file des devices associée à un processeur donné, tout comme dans [77, 100]. Le RISR a également la charge de réveiller le thread réseau associé à un processeur donné pour récupérer les paquets de l'anneau correspondant (contrairement à NAPI qui programme une interruption logicielle).

La classification dans la carte d'interface a été implémentée en utilisant des cartes programmes Myrinet, ce qui permet de tester sur des liens à 2Gbits/s. La classification est pour l'instant très simple car elle utilise exclusivement l'adresse IP de la source (donc du client dans le cas d'un serveur web) en effectuant l'opération simple suivante:

```
ip_src & (nb_proc-1)
```

Dans le cas qui nous intéresse, c'est-à-dire un serveur web chargé, la répartition sur les processeurs est relativement uniforme. Il n'est d'ailleurs pas du tout certain qu'une classification plus complexe soit meilleure.

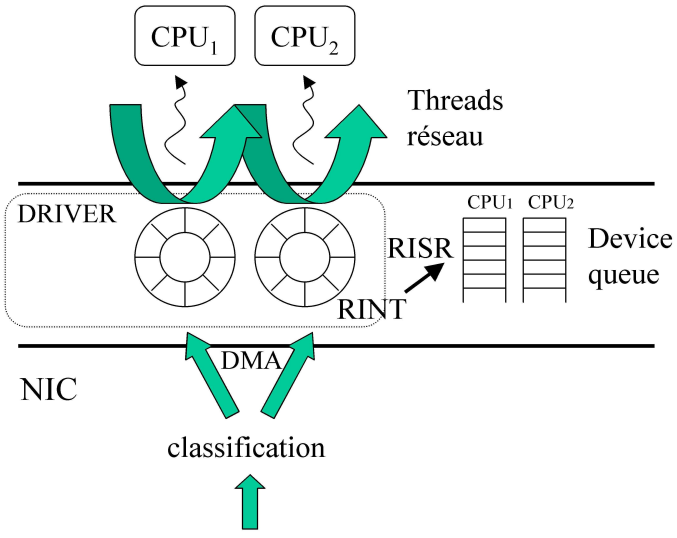

Figure 4.11 Fonctionnement général de KNET.

Le driver de la carte a été modifié pour supporter plusieurs anneaux de réception et descripteurs. Un module Linux permet au noyau de créer les threads réseau qui seront associés à chaque processeur. L'investissement d'Eric a été très grand, pour

d'autre part acquérir les compétences systèmes nécessaires et, d'autre part pour ensuite implémenter les propositions sur des systèmes réels.

4.5 LES PREMIERS RÉSULTATS

Les premiers résultats que je vais résumer ici ont été publiés dans #7[4] qui peut être trouvé en annexe à la fin de ce document, et dans un autre article soumis à USENIX.

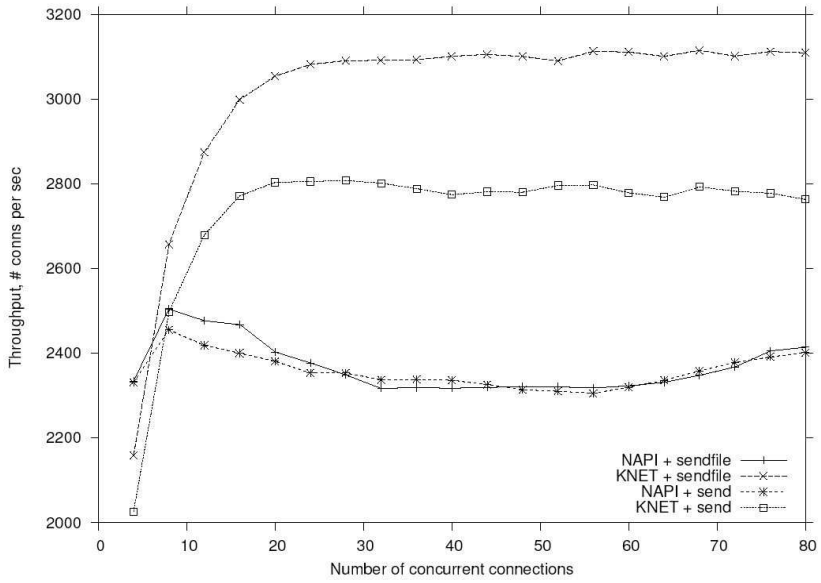

Figure 4.12 MTU 1500 octets, fichier de 20K.

Les travaux d'Eric se concentre sur les serveurs web. Les tests réalisés utilisent donc une machine multi-processeur qui sera le serveur web, et plusieurs autres machines qui agiront comme des clients. Les machines sont connectées par un réseau Myrinet qui offre un débit de 2Gbits/s full-duplex. Ces premières expérimentations ont pour but de "stresser" le système de communication du serveur pour vérifier les bonnes propriétés de KNET. Le système d'exploitation utilisé est Linux, version 2.4.20. Les serveurs web utilisés sont `Webfs` [55] et `Apache2` [2]. Les générateurs de trafic web sont `Sclient` [4] et `WebStone2.5` [76]. En fait c'est une version modifiée par Eric de `Sclient` qui est utilisée: une connexion se déroule de la manière suivante, une requête `http` est émise vers le serveur, celui-ci la traite et renvoie les données demandées (fichiers de 20Ko et 5Ko). Lorsque le client récupère le fichier, il effectue une nouvelle requête.

La figure 4.12 montre le débit en émission exprimé en nombre de connexions satisfaites par seconde (du coté serveur donc) entre NAPI et KNET pour des MTU

[4]E. Lemoine, C. Pham, L. Lefèvre, "Packet classification in the NIC for improved SMP-based Internet servers", A paraître dans IEEE ICN'04.

de 1500 octets et des fichiers web de 20K sur une machine serveur quadri-processor (PIII 550MHz, 512MB RAM, ServerWorks CNB20HE). Le débit est en fonction du nombre de connexions simultanées par client, sachant qu'il y a 4 machines bi-processeurs clientes sollicitant le serveur. 2 appels systèmes sont également comparés: `sendmsg()` et `sendfile()` qui utilise un mécanisme de zero-copy. Dans ce cas, les problèmes dus au ralentissement des ACKs est encore plus exacerbé. On constate dans la figure 4.12 que KNET offre plus de débit que NAPI: les processeurs sont mieux utilisés (la machine est occupée à 100%). Pour NAPI, comme un seul processeur est actif à la fois pour le traitement des paquets réseau, il n'y a pas de différences entre `sendmsg()` et `sendfile()`. Pour KNET, ce n'est pas la même chose: l'utilisation de `sendfile()` permet au serveur de traiter plus de demandes grâce à l'efficacité de KNET. Lorsque le MTU est mis à 500 octets pour augmenter le nombres de paquets, les performances de KNET sont encore meilleures comme illustrée par la figure 4.13. On voit que NAPI est très vite saturé. Avec des MTU de 9000 octets, KNET et NAPI se valent, voir figure 4.14. Dans cette expérience, KNET-RR est une version de KNET sans classification n'optimisant donc pas l'utilisation du cache (les paquets d'une même connexion pouvant être traités par des processeurs différents). On peut constater l'impact important des défauts de cache dans les performances.

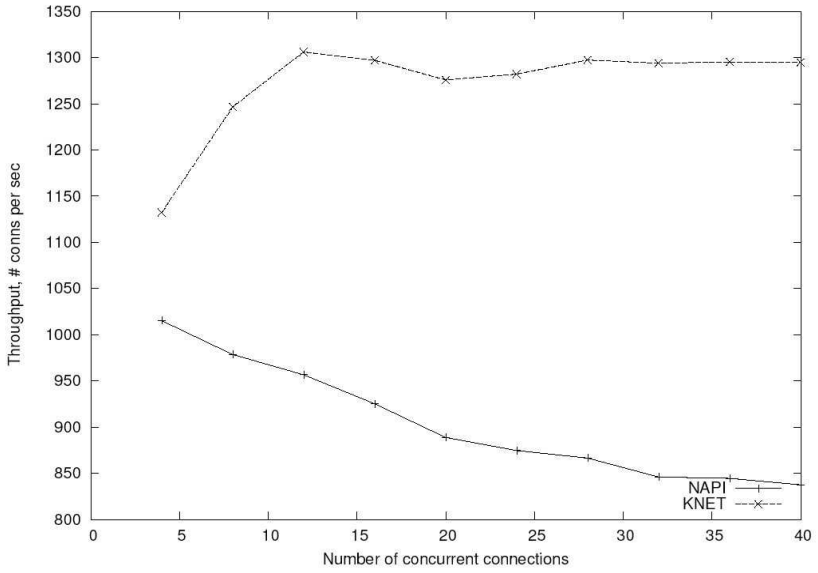

Figure 4.13 MTU 500 octets, fichier de 20K, avec sendfile().

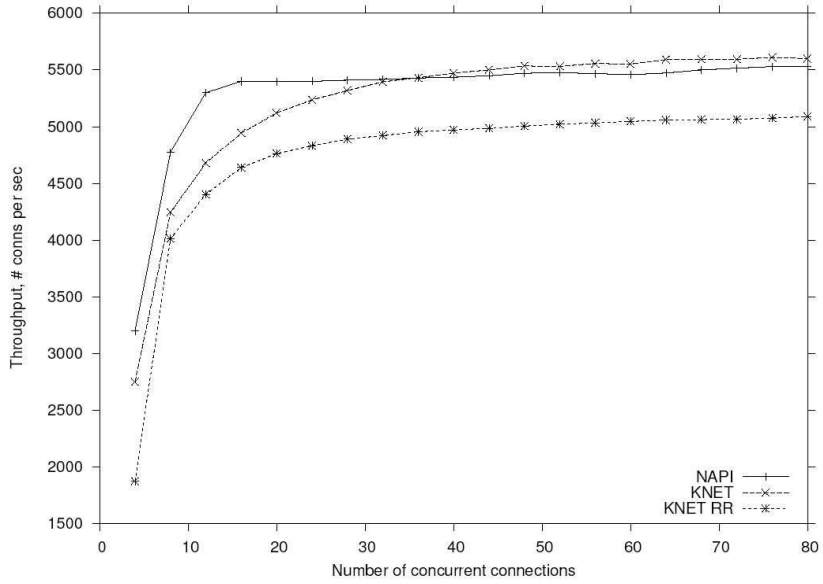

Figure 4.14 MTU 9000 octets, fichier de 20K, avec sendfile().

La figure 4.15 montre les résultats pour une taille de fichier supplémentaire, 5K, pour comparer avec les tests précédents sur des fichiers de 20K.

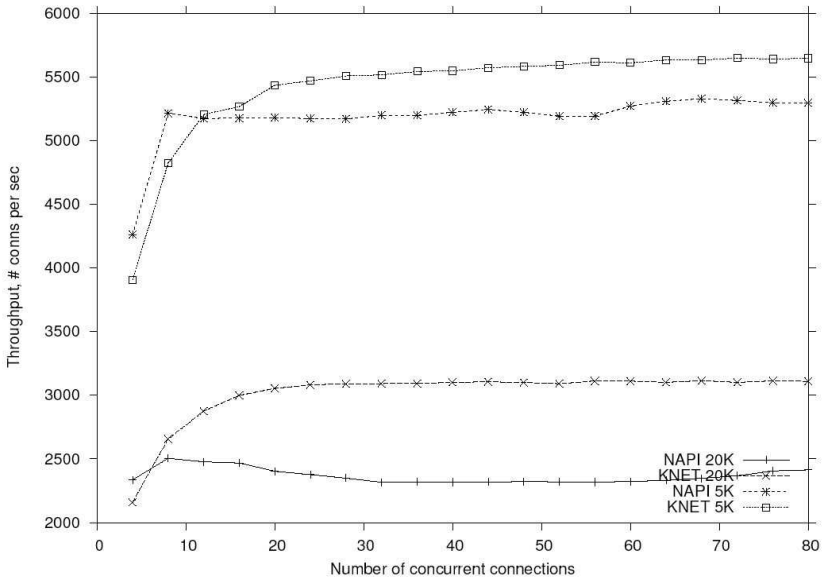

Figure 4.15 1500 octets, fichiers de 20K et 5K, avec sendfile().

4.6 CONCLUSION DU CHAPITRE ET PERSPECTIVES

Les performances des technologies réseaux ont connu un bond assez phénoménale ces dernières années, touchant à la fois le coeur du réseau avec des techniques

DWDM sur fibres optiques que les accès particuliers avec des techniques comme ADSL ou prochainement VDSL. On constate donc une augmentation globale des performances réseaux perçues par l'utilisateur final et l'apparition de nouvelles applications (rendues maintenant possibles) comme le streaming vidéo ou les grilles de calcul. De ce fait, le trafic sur certains équipements serveurs d'extrémité peut augmenter de manière considérable. Dans ce contexte très haut-débit, des équipes de recherche ont trouvé des solutions pour améliorer les performances des protocoles de transport tels que TCP pour exploiter au mieux le débit disponible (High Speed TCP, Scalable TCP, etc). La conséquence est maintenant que les performances sont souvent limitées par la capacité du sous-système de communication à générer, recevoir, transmettre et traiter les données à la vitesse du réseau.

L'optimisation des sous-systèmes de communication est donc à mon avis un volet très important dans la quête pour l'obtention des performances de bout-en-bout. Nos recherches doivent nous permettre de mieux comprendre les facteurs de bas niveau qui entrent en jeu dans l'obtention des performances. Notre direction de recherche va dans la proposition de mécanismes à mettre en oeuvre dans les cartes d'interface. A terme, il est envisageable que de telles propositions soient effectivement implantées dans des produits grand public.

C'est un sujet difficile dans lequel Eric Lemoine a acquis une expérience précieuse. J'ai également pu bénéficier de cette expérience qui me permet d'avoir une vue plus large que si mes recherches ne concernaient que les couches protocolaires de plus niveau comme TCP. Ses travaux sont bien intégrés dans les activités scientifiques de SUN Labs et de SUN Microsystems. Des coopérations ont été développées avec des équipes aux Etats-Unis dans lesquelles Eric a pu apporter son expérience et ses résultats, notamment en ce qui concerne les comparaisons entre KNET et les solutions déployées dans Solaris^{TM}.

Eric est en fin de thèse, il continue les tests sur des machines de plus en plus puissantes (SUN v60x xeon 2.8Ghz par exemple) pour vérifier que KNET est toujours performant. En effet, lorsqu'un processeur est suffisant pour traiter toutes les connexions, NAPI se comporte bien. Eric cherche à montrer que les performances de KNET dans ce cas sont au moins égales à ceux de NAPI. L'utilisation de trafic web plus réaliste comme ceux fournis par `SpecWeb99`, en combinaison avec `Sclient` est aussi prévue. Ceci afin de voir le comportement de KNET sur du trafic réel avec des pics de demandes. D'autres fonctionnalités à mettre sur la carte sont aussi à l'étude par Eric comme celle permettant l'allocation/désallocation des buffers réseau afin de garantir qu'un buffer alloué par un processeur i soit désalloué par le même processeur.

Pour en savoir plus

Depuis déjà quelques années, il existe des architectures matérielles pour supporter un grand nombre de threads. Les lecteurs peuvent consulter la page web du Pr Levy (`http://www.cs.washington.edu/homes/levy/`, projet Simultaneous Multithreading) pour trouver des papiers sur les architectures matérielles hyperthreadées.

Ce type d'architecture a été récemment mis en oeuvre sur des processeurs Intel (que Planète Saturn commence à commercialiser pour les particuliers :-)).

Des produits TCP Offload Engine de chez Alacritech^{TM}, Intel^{TM}, Adaptec^{TM}ou Lucent^{TM}sont disponibles sur le marché pour notamment les produits Gigabit Ethernet. Ces cartes se concentrent généralement sur le zero-copy et le déport de checksum. Cela montre d'une certaine manière qu'il existe un problème de performance qui ne concernent pas que les applications académiques. De plus en plus d'entreprises utilisent des serveurs (base de données, fichiers, web, intranet, etc.) ou des NAS (Network Attached Storage) dont la robustesse doit être garantie. Un article introductif peut être trouvé sur `http://www.10gea.org/SP0502IntroToTOE_F.pdf`.

MES PUBLICATIONS LIEES AU THEME

Conférences

[1] E. Lemoine, C. Pham, L. Lefèvre. Packet Classification in the NIC for Improved SMP-based Internet Servers. In *IEEE ICN'04*, 2004. A paraître.

[2] L. Lefèvre, E. Lemoine, C. Pham, B. Tourancheau. Fast Forwarding with Network Processors. In *PFHSN'02*, Poster Paper, 2002.

Chapitre 5

Conclusions et Perspectives

Nommé Maître de Conférence en octobre 1998 à l'université de Lyon dans l'équipe de Bernard Tourancheau, j'avais derrière moi 3 ans de thèse et une année de séjour post-doctoral, soit 4 ans d'expérience dans la recherche (presque rien donc) et quelques vacations en ce qui concerne l'enseignement. La première année fut très difficile, avec les charges de cours et surtout les préparations des supports, il est tentant de laisser pour un temps la recherche. L'équipe dans laquelle je suis tombé était une petite équipe de recherche, avec une moyenne d'âge de l'ordre de 25 ans! Il est courant de dire par exemple que dans une petite entreprise on a certes moins d'avantages, mais le travail est plus intéressant car plus polyvalent. Pour reprendre cette comparaison, j'ai été très vite plongé dans les cours magistraux à donner, les étudiants à encadrer (certains sont plus agés que moi!), dans la recherche et surtout dans la recherche (en vrac) de financement, de stagiaires, de thésards, de contrats, de meubles, de reconnaissance, de temps, ... Au coté de Bernard et de Laurent j'ai beaucoup appris et nous étions très motivés!

L'enseignement me procure toujours du plaisir et le contact avec les étudiants est important. Il faut faire attention à ne pas se déconnecter de la réalité et des enjeux socio-économiques. Je l'ai déjà évoqué dans l'introduction, le reste de la conclusion sera focalisé sur l'aspect recherche qui a été plus développé dans ce mémoire.

Je n'ai sans doute pas encore le recul nécessaire pour dire si les directions de recherches que j'ai choisies étaient bonnes ou non. Je me suis rendu compte que c'était un exercice très difficile. Ecrire un article de recherche n'est pas difficile en soit, le sujet est très clair et bien focalisé. Ecrire une page sur ses perspectives de recherche à moyen et long terme est nettement plus difficile. Cependant, ces

recherches ont été très enrichissantes pour moi et c'était ce que j'avais envie d'étudier. En ce qui concerne les 3 thèmes présentés dans ce document, le premier est dans la continuité des travaux de thèse, le deuxième sur le multicast est motivé par l'aspect résistance au facteur d'échelle, et le troisième par des retombées du premier et les aspects de performance de bout-en-bout. Dans les 3 thèmes, l'idée maîtresse est de distribuer l'intelligence afin de gagner en efficacité et en performance.

5.1 CONCLUSIONS SUR LES THÈMES DE RECHERCHES

Simulation parallèle

En ce qui concerne la simulation parallèle sur grappes de machines, ces recherches ne seront pas poursuivies. Notre équipe était pionnière sur les grappes il y a quelques années suite aux travaux de L. Prylli et B. Tourancheau. De nombreuses équipes possèdent maintenant cette compétence. En ce qui concerne les techniques de simulation, j'utiliserai celles développées pour évaluer les systèmes de grande taille (modèle de multicast en l'occurence) mais il n'y aura pas de contributions dans le domaine des techniques de simulation eux-mêmes. Si l'on regarde les contributions de la communauté "simulation parallèle" ces dernières années, l'essentielle consiste en des optimisations[1] des 2 approches de base, i.e. conservateur ou optimiste, et en des études de cas sur telle ou telle application (TCP, programmes parallèles,...). Etant membre du comité de programme de PADS, la principale conférence sur ce sujet, en 2001 et 2002, j'ai pu assister à la "crise" et aux discussions relatives à la baisse observée des soumissions. Il y aussi le constat que la communauté de recherche dans ce domaine ne se renouvellait pas: 80% des auteurs se retrouvent d'année en année. La décision qui a été prise a été d'élargir PADS à la problématique simulation en général, et non plus seulement aux simulations parallèles et distribuées. Ce que je veux dire c'est qu'il est possible d'avoir des performances sur un environnement maîtrisé comme les grappes de machines, sans trop de difficultés, et cela grâce aux progrès réalisés dans les architectures matérielles, les librairies de communication et les optimisations connues en simulation parallèle. Il est donc grand temps de passer en phase de production!

Il existe bien sûr de nouvelles directions de recherches très exitantes dans le domaine de la simulation, mais si l'on y regarde de plus près, elles sont très liées au concepts actuels des grilles de calcul. Ces problématiques sont essentiellement du domaine de la simulation distribuée et interactive: des équipes travaillent sur l'utilisation du web comme support et infrastructure pour les simulations distribuées, d'autres regardent l'aspect ubiquité de l'Internet pour proposer des simulateurs prenant en compte un grand nombre de plate-formes: portable, PDA/Palm, etc. Dans ce cas, c'est l'hétérogénéité des machines qu'il faut prendre en compte. Nos travaux sur les grilles de calcul et sur le multicast par exemple sont très similaires

[1] Ces optimisations concernent par exemple la gestion de sauvegardes d'états, l'utilisation de la mémoire, l'utilisation d'estimations locales et de recalcul,...

et l'aspect réseau est omni-présent: comment gérer l'hétérogénéité des récepteurs, comment utiliser le web pour proposer des services supplémentaires (web-services), comment utiliser des overlays... Comme je souhaite me focaliser sur le domaine des réseaux et systèmes, je regarderai ces aspects, mais à partir de la problématique réseau.

Multicast

Comme je l'ai enoncé dans le chapitre 3 sur le multicast, les problèmes fondamentaux du multicast sont bien cernés et les mécanismes pour assurer la fiabilité sont bien connus (ACK/NACK avec aggrégation, FEC). Il reste un grand travail de déployement et de validation des mécanismes pour s'assurer que les solutions proposées soient bien résistantes au facteur d'échelle, surtout dans un contexte très haut-débit. C'est d'ailleurs sans doute dans ce domaine que de nouvelles études peuvent être réalisées. En effet l'arrivée des réseaux très-haut débit a montré les faiblesses et l'inadéquation de protocoles comme TCP pour exploiter toute la bande passante disponible dans le réseau. En laissant de coté les limitations dues à l'implémentation, c'est par exemple le contrôle de congestion et les mécanismes de réaction aux pertes ou erreurs qui sont en cause. Ces mêmes problèmes sont aussi présents pour le multicast. A ma connaissance il n'y a pas encore eu d'études sur l'adéquation d'un contrôle de congestion multicast dans un contexte très haut-débit. Un des domaines dans lequel haut-débit et architecture controlée peuvent être plus facilement trouvés est sans doute la grille de calcul. Ces grilles informatiques sont généralement basées sur une interconnexion très rapide: VTHD à 2.5Gbits/s pour e-Toile, TeraGrid à 40Gbits/s en sont des exemples. L'infrastructure est généralement controlée dans le sens où le groupe d'utilisateurs est fermé et les solutions techniques déployées peuvent être originales (c'est-à-dire sans la contrainte de compatibilité forte avec l'Internet et les protocoles associés). Dans ce contexte, il est possible de proposer des solutions similaires à nos propositions sur DyRAM avec des services évolués hébergés par les routeurs du centre de calcul. Une des solutions d'évaluation de nos propositions dans ce contexte pourra être l'utilisation d'un émulateur réseau très haut-débit (dans le cadre du projet Grid5000 par exemple) pour effectuer des tests à grande échelle.

Un autre point qui m'interpelle concerne la notion de *fairness* dans le multicast qui est encore largement basée sur celle définie par TCP. Il existe des alternatives possibles mais la principale difficulté lorsque l'on cherche a choisir une autre notion de fairness est de la faire accepter! Cependant, cette notion de *fairness* à-la-TCP n'est fondamentalement pas "fair" dans le cas du multicast où la source émet à plusieurs récepteurs en même temps. Le cas *TCP-friendly* est à mon avis un extrême; celui où la source récupère $n_{recv} * D_{tcp}$ est l'autre extrême. Entre les deux, il doit y avoir un compromis intéressant à la fois pour les utilisateurs et pour les opérateurs.

En parlant d'opérateurs ou d'ISP, cela m'amène à parler du modèle économique du multicast. Ce problème que j'ai rapidement évoqué dans le chapitre 3 est un problème très difficile à résoudre car ce n'est pas un problème technique! Les opéra-

teurs ou ISP sont réticents à permettre le multicast sur leurs routeurs car c'est une consommation de ressources importante lorsqu'il faut gérer le routage et dupliquer les paquets sur les interfaces de sortie, surtout lorsque les destinataires sont chez d'autres opérateurs ou ISPs! Tout le monde, au presque, s'accorde pour dire que si les applications sont présentes, le multicast sera là. Depuis des années, les particuliers étaient la cible privilégiée pour des applications multimédia telles que le streaming vidéo. Malgré l'arrivée des techniques DSL permettant un débit confortable chez le particulier, je ne vois toujours pas si ce type d'application sera réellement moteur. Par contre, on peut constater une montée fulgurante des techniques pair-à-pair (P2P) avec des environnements comme `Napster`, `Gnutella`, `iMesh`, etc. Dans ce contexte, il peut être intéressant d'étudier le multicast comme une solution parmi d'autres, qui sera utilisé en des points précis du réseau, pour distribuer l'information. Dans ce cas, il faut réfléchir à la manière dont le multicast doit coopérer avec les autres solutions et s'il n'y a pas des mécanismes à proposer pour rendre cet ensemble de technologies transparent à l'utilisateur.

Y a t-il un avenir pour le multicast et IP multicast? Je pense que oui. Peut-être pas pour le particulier et pour les applications que l'on cherche à lui faire accepter (sans succès car le besoin n'est pas forcément là, du moins à moyen terme!), mais sans doute pour les grands industriels, les grande entreprises et bien sûr les académiques (physiciens, biologistes, astronomes, etc.). Encore une fois, les grilles de calcul seront peut-être le domaine dans lequel le multicast déployé sur une échelle plus réduite que l'Internet pourra montrer sa réelle efficacité pour distribuer les données entre les utilisateurs de la grille car le besoin est cette fois bien réel.

Optimisation de sous-systèmes de communication

Ce domaine de recherche est le plus récent pour moi, mon expérience est toute jeune. Cependant, lorsque l'on regarde l'évolution de l'architecture des ordinateurs et celle d'un réseau de communication, on peut constater quelques similitudes: le processeur central, tout comme les débits dans un reseau de communication, a une capacité de traitement qui augmente sans arrêt alors que les autres composants restent conceptuellement les mêmes voire conservent les mêmes performances. Dans les 2 domaines, l'architecture conceptuelle (dans une vision macroscopique bien sûr) n'a pas beaucoup évoluée depuis plus de 20 ans: un réseau de communication est toujours constitué de routeurs et de liens; dans un ordinateur, c'est principalement un processeur avec des bus et de la mémoire. On peut donc constater que dans les deux domaines les protocoles/mécanismes ont une durée de vie assez longue (TCP d'un coté, NAPI de l'autre coté par exemple). Si l'on reste sur une architecture inchangée, la marge de manœuvre n'est pas grande: on déplace des buffers d'un espace à un autre, on diminue les interruptions, on limite les copies mémoire, mais pas beaucoup plus. Sur un ordinateur, avec le déport de fonctions (carte programmble, Network Processor), c'est architecture qui change, donc la marge de manoeuvre s'agrandie: déport de checksum, classification, TOE, etc. Dans le domaine des reseaux de communication, cette voie serait représentée par les reseaux actifs qui permettent de

déporter des fonctions avancées dans le réseau. Cette voie est plein de perspectives, les premiers résultats de la thèse d'Eric Lemoine sont très encourageants.

Les machines serveurs et plus particulièrement les serveurs web se doivent d'être robustes. C'est sans doute une constatation triviale mais pratiquement ce n'est presque jamais le cas. La robustesse plutôt que les performances pures est une voie de recherche intéressante. Si l'on observe les récentes propositions de la communauté système, on est frappé de voir que beaucoup de propositions sont similaires, par le concept, à certains mécanismes que l'on trouve aussi dans les réseaux de communication: études actuelles sur le contrôle des ressources [112, 111, 6, 5] qui proposent de faire de la classification et du traitements différenciés, du contrôle d'admission et de l'isolation entre les flux, etc. Dans un certain sens, la communauté système rejoint la communauté réseau sur certains aspects liés au contrôle de ressources car celles d'un serveur doivent être partagées par un nombre de plus en plus grand de clients (clients web bien souvent) en maintenant une certaine qualité de service, avec une dégration limitée et controlée des performances.

5.2 PERSPECTIVES

Ma vision que je vais présenter ici est une vision à plus long terme. Cette vision est bien sûr basée sur ma compréhension actuelle des problèmes scientifiques/économique et du contexte international. Elle n'est pas directement reliée aux thèmes de recherche que j'ai présenté dans ce document, elle reflète simplement les tendances que je perçoies.

Le domaine des réseaux de communication évolue à une vitesse phénoménale. Après l'introduction des techniques de multiplexage en longueur d'onde qui ont permis d'avoir des capacités énormes sur les fibres optiques, et qui ont initié un grand nombre de recherches sur les protocoles adaptés au très haut-débit et sur les grilles de calcul, je penses que l'un des domaines qui verra un essor fantastique est celui des réseaux sans-fils. On peut le constater tous les jours, l'accès sans-fils est une technologie indispensable à la continuité de la connectivité sur l'Internet et par là même maintenir la "connectivité" à la société ("vivre ensemble, séparément"). En France et dans les autres pays occidentaux, les systèmes de communication sans-fils de 3ème génération sont en cours d'étude et de déploiement. Dans certains pays d'Asie (comme le Japon), c'est déjà une réalité depuis quelque temps. Certains opérateurs pensent que l'on risque de passer directement de la 2.5G (GPRS) à la 4G (tout IP).

Je ne suis pas un spécialiste dans ce domaine, cependant je vois des interactions fortes qui peuvent se créer entre les réseaux filaires et ceux sans-fils de plus en plus répandus. Ce n'est certe pas une nouveauté (les interactions entre réseaux satellitaires et l'infrastructure filaire ont été étudiées depuis longtemps) mais l'ampleur du phénomène introduit de plus en plus de mobilité. Avec des débits de plus en plus importants dans le sans-fils, les utilisateurs vont sans doute souhaiter avoir exactement les mêmes applications que s'ils étaient dans un environnement filaire.

Cela ouvre des horizons de recherche très larges: sécurité, mobilité, performances des protocoles de transport, routage, interactions aux interfaces, etc. Je resterai à l'écoute de ces opportunités à chercher dans de nouvelles directions.

5.3 CONCLUSION DU DOCUMENT

Cette section est la conclusion de ce document d'HDR. L'écriture de ce document a été très bénéfique et m'a permis de prendre le temps de synthétiser les contributions effectuées et de réfléchir aux perspectives à moyen terme. Ce document a été écrit dans un style didactique, sans jamais trop entrer dans les détails techniques. Comme je l'ai indiqué dans l'introduction, une sélection des articles publiés peut être trouvée en annexe, qui permettra aux lecteurs d'avoir accès à ces détails. Je remercie donc ceux qui ont pris le temps de lire ce manuscrit jusqu'au bout, et j'espère que cette lecture aura été intéressante.

Bibliographie

[1] K. Almeroth, "The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment", *IEEE Network, January /February 2000.*

[2] Apache Software Foundation. Apache2. `http://httpd.apache.org/docs-2.0/`.

[3] A. Azcorra, M. Calderón, M. Sedano, and J. I. Moreno. Multicast congestion control for active network services. *European Transactions in Telecommunications*, 10(3), Mau/June 1999.

[4] G. Banga, P. Druschel. Measuring the Capacity of a Web Server In *USENIX Symposium on Internet Technologies and Systems, 1997.*

[5] Thiemo Voigt, Renu Tewari, Douglas Freimuth and Ashish Mehra. Kernel Mechanisms for Service Differentiation in Overloaded Web Servers. In *Usenix Annual Technical Conference, Boston, MA, USA, June 2001.*

[6] A. Bavier, T. Voigt, M. Wawrzoniak, L. Peterson. SILK: Scout Paths in the Linux Kernel. *Technical Report 2002-009, Dept. of Information Technology, Uppsala University, Feb. 2002.*

[7] Nanette J. Boden, Danny Cohen, Robert E. Felderman, Alan E. Kulawik, charles L. Seitz, Jakov N. Seizovic, and Wen-King Su. Myrinet - a gigabit-per-second local-area network. In *IEEE Micro*, volume 15, Arcadia, CA, February 1995. Myricom. http://www.myri.com.

[8] F. Bouhafs, B. Gaidioz, J.P. Gelas, L. Lefèvre, M. Maimour, C. Pham, P. Primet, B. Tourancheau. Designing and experimenting an active grid architecture. *Future Generation Computer System*, 2004. A paraître.

[9] L. Brakmo and L. Peterson. End to end congestion avoidance on a global internet. *IEEE Journal on Selected Areas in Communications*, 13(8), October 1995.

[10] R. E. Bryant. Simulation of packet communications architecture computer systems. *Massachussets Institute of Technology, MIT-LTC-TR-188, 1977.*

[11] John Byers et al. FLID-DL: congestion control for layered multicast. In *IEEE J-SAC, Special Issue on Network Support for Multicast Communication, 20(8), pp. 1558 - 1570, October 2002.*

[12] M. Calderón, M. Sedano, A. Azcorra, C. Alonso. Active Networks Support for Multicast Applications. *IEEE Networks, May/June 1998.*

[13] M. Castro, P. Druschel and A.M. Kermarrec and A. Rowstron. SCRIBE: a large-scale and decentralised application-level multicast infrastructure. In *IEEE Journal on Selected Areas in Communication (JSAC), 2002.*

[14] K. M. Chandy and J. Misra. Distributed Simulation: A Case Study in Design and Verification of Distributed Programs In *IEEE Transactions on Software Engineering, 1979, pp440-452.*

[15] K. M. Chandy and J. Misra. Asynchronous Distributed Simulation via a Sequence of Parallel Computations. In *Communication of the ACM, Vol. 24(11), April 1981, pp198-206.*

[16] K. M. Chandy, J. Misra and L. M. Haas. Distributed Deadlock Detection. In *ACM Transactions on Computer System, May 1983, pp144-156.*

[17] Jeffrey S. Chase, Andrew J. Gallatin and Kenneth G. Yocum. End System Optimisations for High-Speed TCP. In *IEEE Communications Magazine, April 2001.*

[18] D. M. Chiu, M. Kadansky, and J. Provino. A congestion control algorithm for tree-based reliable multicast protocols. In *IEEE INFOCOM'02.*

[19] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. A Case for End-System Multicast. In *ACM SIGMETRICS 2000.*

[20] Luís Henrique M. K. Costa, Serge Fdida. Enabling the Progressive Multicast Service Deployment. In *Proc ISCC'01, Hammanet, Tunisia.*

[21] J. Crowcroft and K. Carlberg MAPEX: Application Level Multicast Architectural Requirements for APEX *Internet draft.*

[22] C. Diot, B. Levine, J. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. In *IEEE Network, 2000.*

[23] P. Druschel, G. Banga. Lazy Receiver Processing (LRP): A Network Subsystem Architecture for Server Systems. *USENIX OSDI, 1996.*

[24] P. Druschel. Operating System Support for High-Speed Networking. *PhD dissertation, Univ. Arizona, 1994.*

[25] T. von Eicken. *Active Messages: an Efficient Communication Architecture for Multiprocessors.* PhD thesis, University of California at Berkeley, November 1993.

[26] T. von Eicken, A. Basu, M. Welsh. Incorporating memory management into user-level network interfaces In *TR CS Dept., Cornell University, 1997.*

[27] D. Feldmeier. A survey of High-Performance Protocol Implementation Techniques In *HighPerformance Networks, Ed. A. N. Tantawi, pp. 29–50, Kluwer Academic Publishers, Boston, 1994.*

[28] E. Fleury. Communication de groupe - du parallèlisme au ad-hoc. Habilitation à Diriger les Recherches, INSA, UCB Lyon. Décembre 2002.

[29] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ ACM Transactions on Networking*, 5(6), 1997.

[30] Foster, I., Kesselman, C. and Tuecke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications, 15 (3). 200-222. 2001.*

[31] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *Intl J. Supercomputing Applications, 11(2):115-128, 1997*

[32] R. M. Fujimoto. Lookahead in Parallel Discrete Event Simulation. In *Proceedings of the 1988 International Conference on Parallel Processing, pp34-41.*

[33] R. M. Fujimoto. Parallel Discrete Event Simulation. In *Communication of the ACM, Vol. 33(10), October 1990, pp31-53.*

[34] A. Garg. Parallel STREAMS: A Multi-Processor Implementation. In *USENIX, 1990.*

[35] Jean-Patrick Gelas and Laurent Lefèvre. Tamanoir: A high performance active network framework. In C. S. Raghavendra S. Hariri, C. A. Lee, editor, *Active Middleware Services, Ninth IEEE International Symposium on High Performance Distributed Computing*, pages 105–114, Pittsburgh, Pennsylvania, USA.

[36] Jim Gemmell and al. The PGM Reliable Multicast Protocol In *IEEE Networks*, 2003. Multicasting: An Enabling Technology.

[37] P. Geoffray, C. Pham, B. Tourancheau. Exploiting Clusters of Shared Memory Multiprocessors with BIP-SMP: the Parallel Simulation Application. In *Proceedings of the Workshop on Cluster-based Computing, help in conjunction with the 1999 ACM International Conference on Supercomputing, June 20-25, 1999, Rhodes, Greece.*

[38] P. Geoffray, L. Lefèvre, C. Pham, L. Prylli, O. Reymann, B. Tourancheau, and R. Westrelin. High-speed LANs: New environments for parallel and distributed applications. In *ACM/IFIP EuroPar'99*, number 1685 in LNCS, pages 633–642, Toulouse, France, August 1999. Springer-Verlag.

[39] P. Geoffray, C. Pham, L. Prylli, B. Tourancheau, and R. Westrelin. Protocols and software for exploiting myrinet clusters. In *Proceedings of the International Conference on Computational Science (ICCS 2001)*, number 2073&2074 in LNCS, pages 233–242, San Francisco, CA, USA, May 2001. Springer-Verlag.

[40] P. Geoffray, L. Prylli and B. Tourancheau. BIP-SMP: High Performance message passing over a cluster of commodity SMPs. In *Supercomputing'99 (SC99)*.

[41] Rim Hammi, Ken Chen, Jean-Pierre Blin & Frédéric Loras. A Framework for Responsive Control of Video Communication Using Active Networks. *XVIIIth World Telecommunications Congress (WTC'2002), Marne-la-Vallée, France, 22-27 Septembre 2002.*

[42] A. Grimshaw, A. Ferrari, F. Knabe and M. Humphrey. Legion: An Operating System for Wide-area computing. *IEEE Computer, 32(5):29-37, May 1999*

[43] M. Heddes and E. Rutsche. A survey of parallelism in communication subsystems. In *Research Report RZ 2570*, IBM Zurich Research Laboratory, 1994.

[44] Hugh W. Holbrook and David R. Cheriton. IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications. In *SIGCOMM*, 1999.

[45] Hugh W. Holbrook, Sandeep K. Singhal, and David R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. In *SIGCOMM*, Oct. 1995.

[46] V. Jacobson. Congestion avoidance and control. In *ACM SIGCOMM '88*, 1988.

[47] Jain, N., Schwartz, M., Bashkow, T. R. Transport Protocol Processing at GBPS Rates In *Proceedings of the SIGCOMM ?90 Symposium, Sept. 1990.*

[48] D. R. Jefferson. Virtual Time. *ACM Transaction on Programming Languages and Systems, Vol. 7(3), July 1985, pp405-425.*

[49] S. K. Kasera, J. Kurose and D. Towsley. A Comparaison of Server-Based and Receiver-Based Local Recovery Approaches for Scalable Reliable Multicast. In *Proceedings of IEEE INFOCOM, San Francisco, CA, USA, March 1998.*

[50] S.K. Kasera, G. Hjalmtysson, D. Towsley and J. Kurose. Scalable Reliable Multicast Using Multiple Multicast Channels. *IEEE/ACM ToN, Jun 2000.*

[51] S. Kasera and S. Bhattacharya. Scalabe fair reliable multicast using active services. *IEEE Network Magazine's Special Issue on Multicast*, 2000.

[52] J. Kay, J. Pasquale. The Importance of Non-Data Touching processing overheads in TCP/IP. In *ACM SIGCOMM 1993.*

[53] H. Keng, J. Chu. Zero-Copy TCP in Solaris. In *USENIX, 1996.*

[54] E. Kim et al. A Simple Router Assist Mechanism For Tree-Based Reliable Multicast Protocols. In *Proc. ICACT 2002.*

[55] G. Knorr. Webfs. `http://bytesex.org/webfs.html`.

[56] B. Li and J. Liu Multirate Video Multicast over the Internet: an overview. In *IEEE Network magazine, Special issue on "Multicasting: An Enabling Technology", January/February 2003.*

[57] L. Lefèvre, E. Lemoine, C. Pham, B. Tourancheau. Fast Forwarding with Network Processors. *Poster and short paper at the 7th IEEE/IFIP International Workshop on Protocols for High-Speed Networks (PfHSN), pages 42-48, Berlin, Germany, April 2002.*

[58] L. Lefèvre, C. Pham, P. Primet, B. Tourancheau, B. Gaidioz, J. P. Gelas, M. Maimour. Active Networking Support for The Grid. *Proceedings of the third International Working Conference on Active Networks (IWAN'01)*, 2001.

[59] Arnaud Legout and Ernst W. Biersack. Revisiting the Fair Queueing Paradigm for End-to-End Congestion Control. In *IEEE Network Magazine, 16(5):38–46, September 2002.*

[60] L. Lehman, S. Garland, and D. Tennehouse. Active reliable multicast. In *Proc. of the IEEE INFOCOM, San Francisco, CA*, March 1998.

[61] B.N. Levine and J.J. Garcia-Luna-Acceves. Comparaison of Known classes of Reliable Multicast Protocols. In *Proceedings of the International Conference and Network Protocols (ICNP)'96, Oct 1996.*

[62] S. Liang and D. Cheriton. TCP-SMO: Extending TCP to Support Medium-Scale Multicast Applications. In *IEEE INFOCOM'02.*

[63] M. Litzkow and M. Livny. Experience With The Condor Distributed Batch System. In *IEEE Workshop on Experimental Distributed Systems, October 1990.*

[64] M. Maimour, C. Pham. Dynamic replier active reliable multicast (dyram). *Journal of Cluster Computing*, 2004. A paraître.

[65] M. Maimour, C. Pham. Active reliable multicast on application-aware grids. *Journal of Grid Computing*, 2004. A paraître.

[66] M. Maimour, C. Pham. Dealing with heterogeneity in a fully reliable multicast protocol. In *Proceedings of IEEE International Conference On Networks (ICON 2003)*, Sydney, Autralia, September 2003.

[67] M. Maimour and C. Pham. AMCA: an Active-based Multicast Congestion Avoidance Algorithm. *Rapport de Recherche LIP 2003-07.*

[68] M. Maimour, C. Pham. A RTT-based Partitioning Algorithm for a Multirate Reliable Multicast Protocol. *Proceedings of the IEEE High Speed Network and Multimedia Communications (HSNMC 2003), Esteril, Portugal, July 2003.*

[69] M. Maimour and C. Pham. AMCA: an Active-based Multicast Congestion Avoidance Algorithm. In *Proceedings of the 8th IEEE Symposium on Computers and Communications (ISCC 2003)*, Antalya, Turkey, July 2003.

[70] M. Maimour and C. Pham. Towards an application-aware communication framework for computational grids. In *Proceedings of the Asian Computing Science Conference (ASIAN 2002)*, pages 140–152, Hanoi, Vietnam, December 2002.

[71] M. Maimour and C. Pham. An analysis of a router-based loss detection service for active reliable multicast protocols. In *Proceedings of the 11th IEEE International Conference on Networks (ICON 2002)*, Singapour, August 2002.

[72] M. Maimour, J. Mazuy, and C. Pham. The cost of active services in active reliable multicast. In *Proceedings of the 4th IEEE Annual International Workshop on Active Middleware Services (AMS 2002)*, pages 67–72, Edinburg, UK, July 2002.

[73] M. Maimour and C. Pham. An active reliable multicast framework for the grids. In *Proceedings of the International Conference on Computational Science (ICCS 2002)*, volume 2330 of *Lecture Notes in Computer Science*, pages 588–597, April 2002.

[74] M. Maimour and C. Pham. A throughput analysis of reliable multicast protocols in an active networking environment. In *Proceedings of the Sixth IEEE Symposium on Computers and Communications (ISCC 2001)*, pages 151–158, Hammamet, Tunisia, July 2001.

[75] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven Layered Multicast. In *SIGCOMM 1996.*

[76] Mindcraft. WebStone 2.5. `http://www.mindcraft.com/webstone/`.

[77] J. Mogul, K. Ramakrishnan. Eliminating Receive-Livelock in an Interrupt-Driven Kernel. In *USENIX, 1996.*

[78] Erich M. Nahum. Performance issues in parallelized network protocols. In *USENIX OSDI, 1994.*

[79] Erich M. Nahum. Networking Support For High-Performance Servers. In *PhD dissertation*, University of Massachusetts Amherst, 1997.

[80] J. Nonnenmacher et al. How Bad is Reliable Multicast Without Local Recovery? In *Proc. of the IEEE INFOCOM*, March 1998.

[81] M. Ott, G. Welling and S. Mathur. CLARA : A Cluster based Active Router Architecture. *In Proceedings of the Hot Interconnects VIII*, August 2000.

[82] Pakin, S., Karamcheti, V., Chien, A,: Fast messages (FM): Efficient, portable communication for workstation clusters and massively-parallel processors. *IEEE Concurrency*, 1997.

[83] Christos Papadopoulos, Guru M. Parulkar, and George Varghese. An error control scheme for large-scale multicast applications. In *Proc. of the IEEE INFOCOM*, March 1998.

[84] S. Paul and K. Sabnani. Reliable multicast transport protocol (RMTP). *IEEE JSAC, Spec. Issue on Network Support for Multipoint Communications*, 15(3), April 1997.

[85] D. Pendarakis, S. Shi, D. Verma, M. Waldvogel. ALMI: An Application Level Multicast Infrastructure *3rd USENIX Symposium on the Internet*

[86] C. Pham. High performance clusters: A promising environment for parallel discrete event simulation. In *PDPTA'99*, volume III, pages 1299–1304, Las Vegas, NV, USA, June 1999. CSREA Press.

[87] C. Pham. Simulation distribuée de réseaux ATM: étude et perspectives. Thèse de 3ème cycle, Université Paris 6, sous la direction du Prof. S. Fdida. Soutenue le 4 juillet 1997. http://www.ens-lyon.fr/~cpham/Paper/these2e.ps.gz

[88] C. Pham and S. Fdida. Relaxation of Synchronization Constraints in Parallel Simulation of ATM Networks. In *Proceedings of the Anglo-French Workshop on Formal Methods, Modelling and Simulation for System Engineering, Dassault Electronique, St-Quentin en Yvelines, France, February 13-14 1995.*

[89] C. Pham and S. Fdida. The Statistically Correct Approach to Distributed Simulation of ATM Network. In *Proceedings of the 6th IFIP Conference on Performance of Computer Networks, Istanbul, Turkey, October 23-26, 1995.*

[90] C. Pham and S. Fdida. Length-Based Blocking and Local Estimation in Distributed Simulation: A Case Study. In *Proceeding of the 29th Annual Simulation Symposium Louisiana, New Orleans, April 8-11 1996.*

[91] S. Pingali, D. Towsley, and J.F. Kurose. A Comparaison of Sender-initiated and Receiver-initiated Reliable Multicast protocols. In *Proc. of ACM SIGMETRICS'94, Vol.14, pp221-230, 1994.*

[92] P. Primet Habilitation à Diriger les Recherches. Juillet 2002.

[93] L. Prylli. Réseaux et Systèmes de Communication Locaux: Etude des Couches Logicielles de Base. Thèse de 3ème cycle sous la direction du Pr. B. Tourancheau. ENS-Lyon, Numéro d'ordre 85, 22 janvier 1998.

[94] Loïc Prylli and Bernard Tourancheau. BIP: a new protocol designed for high performance networking on myrinet. In *Workshop PC-NOW, IPPS/SPDP98*, Orlando, USA, 1998.

[95] L. Prylli, B. Tourancheau, and R. Westrelin. The design for a high performance mpi implementation on the myrinet network. In *EuroPVM/MPI'99*, 1999.

[96] I. Rhee, N. Ballaguru, and G. N. Rouskas. Mtcp : Scalable tcp-like congestion control for reliable multicast. In *Proceedings of IEEE INFOCOM'99.*

[97] Dan Rubenstien, Sneha Kumar Kasera, Don Towsley and Jim Kurose. Improving Reliable Multicast Using Active Parity Encoding Services. In *Proceedings of IEEE Infocom 1999.*

[98] L. Sahasrabuddhe and B. Mukherjee. Multicast Routing Algorithms and Protocols: A Tutorial. In *IEEE Network, Jan./Feb. 2000.*

[99] J. Salehi, J. Kurose, D. Towsley. The Effectiveness of Affinity-based Scheduling in Multi-processor Network Protocol Processing. In *IEEE/ACM Transactions on Networking, Vol. 4(4), 1996.*

[100] J. H. Salim, R. Olsson, A. Kuznetsov. Beyond Softnet. In *USENIX, 2001.*

[101] A. El-Sayed, V. Roca and L. Mathy. A survey of Proposals for an Alternative Group Communication Service. In *IEEE Network magazine, Special issue on "Multicasting: An Enabling Technology", January/February 2003.*

[102] P. Spathis, K. L. Thai. MAF: un protocole de multicast fiable In *CFIP 2002.* Montréal, January 2002.

[103] T. Speakman et al. Pgm reliable transport protocol specification. internet draft, 1998.

[104] R. State, O. Festor, E. Nataf. A Programmable Network Based Approach for Managing Dynamic Virtual Private Networks In *Proceedings of PDPTA 2000*, Las Vegas, June 26-29.

[105] Ion Stoica, T. S. Eugene Ng and Hui Zhang. REUNITE: A Recursive Unicast Approach to Multicast. In *IEEE INFOCOM 2000*.

[106] A. Striegel, G. Manimaran, "A Scalable Approach for DiffServ Multicasting", *Proc. of IEEE ICC'2001*.

[107] D. L. Tennehouse et Wetherall. Towards an Active Network Architecture. *Proceedings MMCN'96*.

[108] D. L. Tennehouse et al. A survey of active network research. *IEEE Comm. Mag.*, pp80–86, January 1997.

[109] Lorenzo Vicisano, Luigi Rizzo, and Jon Crowcroft. TCP-like Congestion Control for Layered Multicast Data Transfer. In *INFOCOM 1998*.

[110] P. Wang, Z. Liu Operating System Support for High-Performance Networking, A Survey. www.cs.iupui.edu/~zliu/doc/os_survey.pdf

[111] M. Welsh and D. Culler. Adaptive Overload Control for Busy Internet Servers. In *USENIX 2003*.

[112] M. Welsh, D. Culler and E. Brewer. SEDA: An Architecture for Well-Conditioned, Scalable Internet Services. In *SOSP 2001*.

[113] R. Westrelin. Propositions autour de l'architecture logicielle des interfaces réseau programmables. Thèse de 3ème cycle sous la direction du Pr. B. Tourancheau. Univ. Lyon 1, Numéro d'ordre 178-2001, 28 septembre 2001.

[114] D. Wetherall. Active network vision and reality: lessons from a capsule-based system. *Operating Systems Review*, 34(5):64–79, December 1999.

[115] M. Yajnik, J. Kurose, and D. Towsley. Packet Loss Correlation in the MBone Multicast Network. In *Proceedings of Global Internet Conference, November 1996*.

[116] L. Yamamoto and G. Leduc. An active layered multicast adaptation protocol. In *IWAN*, pages 179–194, 2000.

[117] Rajendra Yavatkar, James Griffoen, and Madhu Sudan. A reliale dissemination protocol for interactive collaborative applications. In *ACM Multimedia*, 1995.

[118] David J. Yates. Connection-Level Parallelism For Network Protocols On Shared-Memory Multiprocessor Servers. In *PhD dissertation*, University of Massachusetts Amherst, 1997.

[119] D. J. Yates, E. M. Nahum, J. F. Kurose, and D. Towsley. Networking support for large scale multiprocessor servers. In *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems, 1996.*

[120] S. Zabele et al. Improving Distributed Simulation Performance Using Active Networks. In *World Multi Conference, 2000.*

[121] B. Zhang, S. Jamin and L. Zhang Host Multicast: A Framework for Delivering Multicast To End Users. In *IEEE INFOCOM'02.*

NOTES

NOTES

ANNEXES

Une sélection des articles publiés

1. C. Pham and C. Albrecht. "Tuning Message Aggregation on High Performance Clusters for Efficient Parallel Simulations". *Parallel Processing Letters, 9(4):521–532, 1999.*

2. P. Geoffray, C. Pham, and B. Tourancheau. "A Software Suite for High-Performance Communications on Clusters of SMPs". *Journal of Cluster Computing, Vol 5(4), pp353-363, October 2002.*

3. M. Maimour, C. Pham, "An Analysis of a Router-based Loss Detection Service for Active Reliable Multicast Protocols", *Proceedings of the 11th IEEE International Conference on Networks (ICON 2002), August 27-30, 2002, Singapore, pp49-56.*

4. M. Maimour, C. Pham, "Dynamic Replier Active Reliable Multicast (DyRAM)", *Proceedings of 7th IEEE Symposium on Computers and Communications (ISCC 2002), July 1-4 2002, Taormina, Italy, pp275-282.*

5. M. Maimour, C. Pham, "AMCA: an Active-based Multicast Congestion Avoidance Algorithm", *Proceedings of the 8th IEEE Symposium on Computers and Communications (ISCC 2003), Antalya, Turquie.*

6. M. Maimour, C. Pham, "Experimenting Active Reliable Multicast on Application-Aware Grids", A paraître dans *Journal of Grid Computing*, 2004.

7. E. Lemoine, C. Pham, L. Lefèvre, "Packet Classification in the NIC for Improved SMP-based Internet Servers", A paraître dans *Proceedings of the IEEE ICN 2004.*