HAL
open science

Grid flow control : prototype of a grid gateway based on network processors

Sébastien Soudan, Pascale Primet

To cite this version:

Sébastien Soudan, Pascale Primet. Grid flow control : prototype of a grid gateway based on network processors. [Rapport de recherche] LIP RR-2006-18, Laboratoire de l'informatique du parallélisme. 2006, 2+11p. hal-02102285

HAL Id: hal-02102285

https://hal-lara.archives-ouvertes.fr/hal-02102285

Submitted on 17 Apr 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Laboratoire de l'Informatique du Parallélisme

École Normale Supérieure de Lyon
Unité Mixte de Recherche CNRS-INRIA-ENS LYON-UCBL n° 5668

Grid flow control: Prototype of a grid gateway based on Network Processors

Sebastien Soudan ,
Pascale Primet

May 2006

Research Report N° 2006-18

École Normale Supérieure de Lyon
46 Allée d'Italie, 69364 Lyon Cedex 07, France
Téléphone : +33(0)4.72.72.80.37
Télécopieur : +33(0)4.72.72.80.80
Adresse électronique : lip@ens-lyon.fr

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

INRIA

Grid flow control: Prototype of a grid gateway based on Network Processors

Sebastien Soudan , Pascale Primet

May 2006

Abstract

Due to network bandwidth growth, grid computing has been envisionned. As grids are aggregation of computer clusters based on high performance local networks and interconnected by very high speed core networks, access links to core networks remain bottleneck locations. In order to optimize grid computing overall performance, we propose to organize data transfers from one cluster to an other. This organization must allow ressource schedulers to schedule data transferts in addition of tasks. Based on a grid network resource reservation model this paper presents the design f a grid gateway, located at the LAN/WAN interface. To implement this approach, we have develop a prototype based on *Network Processors* Intel *IXP2400* which is able to control flows at 1 gigabits/s. We discuss the main design choices and experimental results

Keywords: Network Processor, Grid, IXP2400, gateway

Résumé

Les grilles de calcul hautes performances sont aujourd'hui consitutées de grappes d'ordinateurs interconnectées par des réseaux de cœur surdimensionnés. Cependant les liens d'accés dont les débits restent inférieurs aux débits disponibles au sein d'une grappe, constituent de sévères goulets d'étranglement. Nous proposons d'optimiser les performances globales des grilles par une organisation et un contrôle des transfert permettant à l'ordonnanceur de tâches de planifier les communications en même temps que les tâches. En ce basant sur ce nouveau modèle de réservation de ressource réseau dans la grille, ce rapport présente l'architecture d'équipements en charge de l'allocation et du contrôle de la ressource réseau à l'interface LAN/WAN. Pour implanter cette solution tout en conservant les débits d'accés (1 gigabits/s, voire 10 gigabits/s), nous avons développé un prototype basé sur la technologie des *Network Processors IXP2400* d'Intel. Nous discutons ici les principaux choix conceptuels et quelques résultats expérimentaux.

Mots-clés: Network Processor, Grille, IXP2400, passerelle

1 Introduction

La formidable croissance de la puissance des processeurs prédite par la loi de Moore a fortement influencé l'évolution du domaine du calcul au cours des 40 dernières années. Les avancées encore plus rapides des technologies de transmission optique sont à même de produire une nouvelle révolution non seulement dans le domaine des télécommunication mais aussi dans le domaine du calcul. Ces grilles peuvent répondre aux besoins de calcul croissants des grandes applications scientifiques (de biologie, d'astro-physique, de physique des hautes énergies, d'imagerie médicale) et industrielles (simulation numérique) [BFH03], [Ter05], [Dat05], [Glo05]. Pour créer cette agrégation complexe de ressources, il était naturel de se baser sur la technologie IP de l'Internet qui répond aux exigences d'interopérabilité, masque l'hétérogénéïté des équipements et des technologies et présente d'excellentes propriétés de robustesse et de résistance au facteur d'échelle. Les applications réparties sur une grille utilisent les protocoles `TCP-UDP` de l'Internet qui sont largement diffusés et disponibles sur les noeuds de calcul et permettent de communiquer sur les liens longue distance.

Cependant, on observe que dans le cadre des grilles aujourd'hui déployées, les capacités sans cesse croissantes et les services de plus en plus sophistiqués fournis par les infrastructures réseau ne sont pas accessibles aux applications. Le nuage réseau global présente un mode de partage non contrôlé globalement [FJ95] qui lui confère des performances, une sécurité et des fonctionnalités très variables et parfois incompatibles avec les algorithmes de calcul classiques. En conséquence, la performance d'exécution des applications, en grande partie déterminée par les mouvements de données, est souvent délicate à prédire et à obtenir en pratique. La possibilité de perte de messages en rafale, les délais de transmission importants et variables, le comportement très dynamique des liens entre processus distants remettent profondément en question les modèles et les techniques développées dans le contexte du calcul parallèle et du calcul distribué sur grappe de calcul. Il est pourtant fondamental de s'assurer que le potentiel considérable offert par l'agrégation de ressources ne soit pas inexploité voire gâché par l'inadéquation des modèles et des mécanismes de communication mis en oeuvre.

Le besoin de qualités et de différentiation de service est très important [FFR+04]. Dans une grille, il faut faire cohabiter des flux aux contraintes antagonistes tels que les flux de contrôle, les barrières de synchronisation et les transferts massifs de données. Le modèle de service unique *Best Effort* pour l'acheminement de paquets et l'omniprésent protocole de transfert TCP offrant une fiabilité et un ordre total, un partage équitable de la bande passante sans aucun contrôle des délais présente de sérieuses limitations ici. Le critère global d'équité (max-min fairness) classiquement recherché pour le partage de la bande passante n'est pas adapté à ces nouveaux usages et à des objectifs d'optimisation différents. Ici, la réservation de bande passante s'impose.

Ce rapport s'intéresse à ce problème particulier du contrôle des flux haut débit dans le cadre d'un modèle d'overlay de grille. La section suivante explicite notre modèle de contrôle des flux. La section 3 présente l'architecture d'un l'équipement baptisé *grid gateway* que nous proposons pour l'exécution de l'algorithme distribué d'allocation de bande passante et le contrôle en ligne des trafics et des flux traversant le lien d'accès. Nous présentons un prototype de cet équipement basé sur la technologie des processeurs réseau (network processor) que nous avons développé. Les résultats expérimentaux sont donnés en section 4. Enfin, la section 5 replace ces travaux dans l'état de l'art. La conclusion et les perspectives sont développées en section 6.

2 Réservation de bande passante et contrôle des flux de la grille

Dans une grille basée sur des réseaux TCP/IP, il est difficile de prévoir à l'avance la durée d'une tâche car on ne sait pas quand les données nécessaires seront disponibles sur les ressources de calcul. Dans un réseau filaire, pour qu'un transfert soit déterministe, il ne doit pas être perturbé par d'autres flux. C'est le principe de fonctionnement des réseaux locaux haute performance des grappes de calcul comme MYRINET où un circuit éphémère (wormhole) est alloué de bout en bout pour un transfert. Ainsi, les flux à faible latence des applications parallèles (MPI) utilisent abondamment ces garanties strictes que seule une forme plus ou moins virtuelle de réservation de ressource peut offrir.

Est-ce que cette approche peut s'étendre aux communications longues distances ? Dans [MPRZ05] nous avons proposé et étudié un nouveau modèle de réservation de bande passante adapté aux flux et aux réseaux très haut débit des grilles et avons proposé des algorithmes d'ordonnancement associés. Cette proposition se base sur un certain nombre d'observations: les grilles connectent un faible nombre de ressources interagissant (e.g. 10^3 et non 10^8 comme dans l'Internet), la capacité d'une seule source (c = 1Gbps) est comparable à la capacité des goulets d'étranglement (c = 1 ou 10Gbps), les délais entre sites sont importants, les transferts sont massifs, le nombre de flux actifs tend à être faible et les flux tendent à être très longs (heures, jours). Ceci élimine le problème de passage à l'échelle aussi bien fois en nombre de flux qu'en nombre d'états à mettre à jour qui freine le déploiement des schemas de réservation dans Internet (cf IntServ). Il a été observé que dans une grille, si un contrôle d'admission pro-actif n'est pas appliqué, les performances se détériorent rapidement, à cause du profil "en rafale" des demandes, même lorsque la charge est moyenne voire faible (moins de 50%). Nous ne détaillons pas ici le modèle ni les fonctions objectifs, ni les heuristiques proposés, mais en donnons les principes généraux.

L'idée sous jacente est de transposer l'approche adoptée dans les réseaux de grappes à des réseaux longue distance. Le but étant de faire en sorte que du point de vue du flux, celui-ci se trouve isolé dans un lien bien approvisionné. La fonction d'allocation prise en charge par les interface des réseaux rapides de grappes est ici déportée vers les passerelles d'accès, qui, dans notre modèle, constituent les seuls points d'engorgement. Nous considérons à la fois les liens d'entrées (ingress point) et les liens de sorties (egress point) du coeur de réseau longue distance. L'objectif est de garantir que la bande passante est suffisante et constante tout au long du chemin (et en particulier dans ces deux types de goulets d'étranglement) pour assurer le transport des différents flux sans perturbation. L'idée est d'imposer à chaque flux une limite de débit de telle sorte que la somme de ces débits soit inférieure aux bandes passantes des liens rencontrés. Les flux traversant plusieurs passerelles, les limites de débit doivent également être négociées avec les autres passerelles. Nous ajoutons un aspect temporel à l'approche de réservation de bande passante au niveau des passerelles. En effet, les volumes à transférer sont connus à l'avance. Les sources peuvent donc être autorisées à émettre pendant une fenêtre temporelle donnée comme le montre la figure 1. De ce fait chaque passerelle connaît par avance son "planning" d'utilisation et peut donc accepter (de son point de vue) un nouveau flux ou le rejeter. La mise en accord des passerelles d'accès pour planifier un flux de bout en bout se fait par l'échange de leurs ordonnancements respectifs et la négociation.

Ce modèle s'intéresse en particulier aux flux massifs et se base sur un modèle particulier

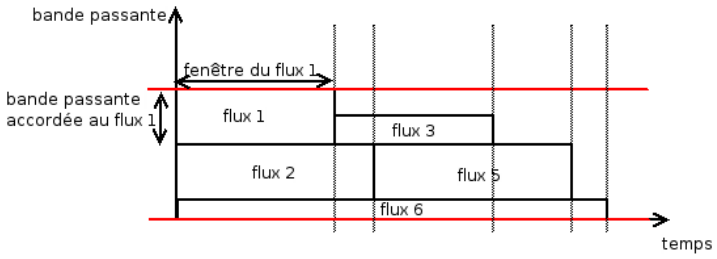

Figure 1: "Planning" de flux

de réseau avec un cœur virtuellement complètement maillé (type crossbar). Nous travaillons actuellement la généralisation de ce modèle théorique pour y intégrer les flux faibles latence et les autres flux *best effort* ainsi qu'un modèle de réseau hiérarchique plus complet. Ce modèle peut être mis en œuvre par un overlay de contrôle dont l'objectif est d'offrir un service d'ordonnancement, de gestion et de contrôle des flux de grille aux points d'engorgement potentiels. Ce service peut coopérer étroitement avec le système global de gestion des ressources ainsi qu'avec les processus d'applications.

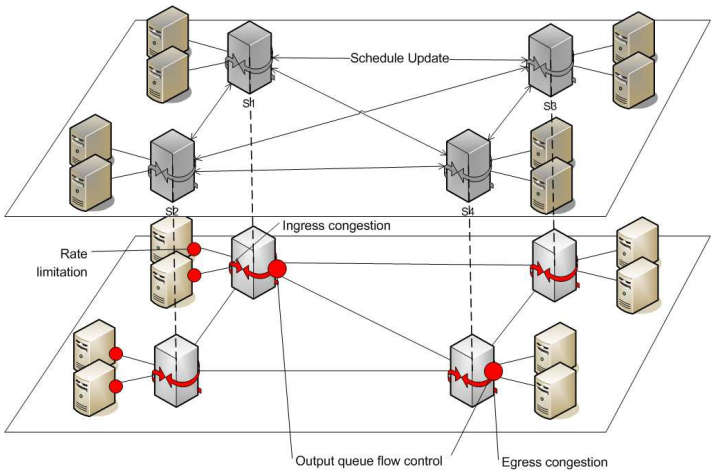

Figure 2: Modèle de réseau de grille proposé

Cet overlay s'appuie sur des mécanismes de contrôle activés à différents niveaux de la hiérarchie réseau. Il a été observé que les pertes de performance (latence et pertes de données) pour les connexions à très haut produit débit-délai sont dues à diverses formes de congestions et de dépassement de capacité et se produisent par rafale et au niveau des systèmes d'extrémité et des liens d'accès. Ainsi, même dans le cas de liens réservés, les mécanismes de contrôle de congestion à fenêtre de l'algorithme de TCP s'avèrent peu appropriés. Nous avons montré [VTK+06] qu'il est possible d'optimiser des connexions TCP sur des liens très longue distance à bande passante garantie en utilisant des techniques de limitation de débit et d'espacement de paquets à la source pour éviter les rafales comme illustré par les figures 3 et 4. Dans les deux figures, les deux connexions longue latence (100ms) se partagent un même lien gigabit via le protocole TCP-BIC (version amélioré de TCP-RENO pour les hauts produits débit-délai). Dans la courbe de gauche, elles se perturbent mutuellement et ne parviennent pas

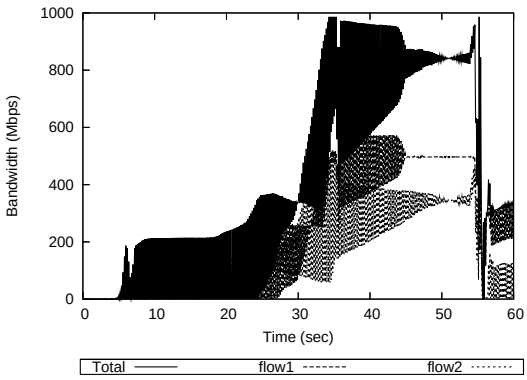

Figure 3: Flux BIC limités par token bucket

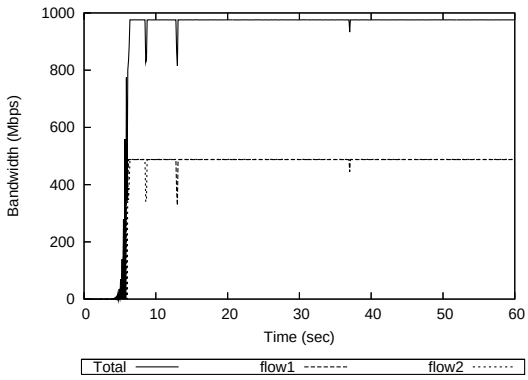

Figure 4: Flux BIC limités par PSPacer

à partager optimalement le lien à cause de problèmes de synchronisation de fenêtre et de phénomènes de rafale pendant la phase de slow start (pire cas). Dans la figure de droite, l'ajout d'une limitation de débit par espacement régulier des paquets via un mécanismes d'espacement de paquet, permet d'obtenir un débit stable et équitable pour chacune des connexions. Ce mécanisme permet de lisser les rafales dûes au mécanisme de contrôle de congestion par fenêtre de TCP et à la longue latence observées dans la figure de gauche. Ces expériences nous montrent que pour optimiser l'utilisation des liens et maximiser les débits des connexions, il est nécessaire de contrôler très finement les flux haut débit, tant à leur émission qu'au cours de leur traversée des liens partagés. Notre approche de réservation de ressources nécessite donc une bonne coopération de toutes les sources et le respect de l'ordonnancement.

Le contrôle des flux au point d'accès consiste à vérifier que chaque flux respecte le débit ainsi que de la fenêtre temporelle allouée. Dans la suite de ce rapport, nous nous interessons à la description et à l'implantation d'une architecture de passerelle de grille, élément actif de l'overlay de grille. Chaque passerelle de grille est localisée à l'interface entre le réseau local et le réseau longue distance. Elle est traversée par tous les flux sortants ou entrants dans le site (grappe) associé. Cette passerelle a en charge l'exécution de l'algorithme distribué d'ordonnancement de flux dans le plan contrôle ainsi que le contrôle de l'exécution de cet ordonnancement par chacun des flux traversants.

3 Conception d'une passerelle à base de *Network Processors*

L'architecture de la passerelle comporte deux plans fonctionnels distincts : le plan données et le plan contrôle. Le rôle du premier est de transmettre les paquets alors que le second contient l'"intelligence" de la passerelle. Intéressons nous d'abord au plan données. Les passerelles sont classiquement [KMC+00] vues comme une succession de blocs fonctionnels reliés par des files d'attentes. Ces blocs sont : *réception (Rx)*, *Routage + contrôle des flux* et *transmission (Tx)*. *Rx* est le bloc fonctionnel responsable de la réception des paquets, récupére les données arrivant sur les interfaces optiques et rassemble les paquets en mémoire. Les files d'attentes ne contiennent que des descripteurs de paquets. Ceux-ci permettent de trouver les données en mémoire et contiennent des informations sur les traitements réalisés et ceux à effectuer. *Tx* est le bloc fonctionnel chargé d'envoyer les paquets routés. Celui-ci reçoit des descripteurs de paquets provenant du bloc de routage par l'intermédiaire des files d'attente associées.

Lorsque Tx recoit un descripteur de paquet, le paquet correspondant est déjà prêt à être envoyé (toutes les modifications de son contenu ont été faites). La tâche de Tx est alors de découper le paquet et de les écrire sur la bonne interface. Le bloc fonctionnel de routage et de filtrage des flux représenté sur la Figure 5 réalise plusieurs fonctions. Il est en charge de router le paquet. Cette étape se décompose en plusieurs sous-étapes qui consistent à modifier les en-têtes en fonction de la route que doit suivre ce paquet. Cette étape, présente dans toutes les passerelles, est suivie d'une autre plus spécifique à notre passerelle : le contrôle de flux. Il s'agit d'identifier le flux auquel appartient le paquet, de vérifier que le paquet n'est pas en dehors du profil puis d'autoriser ou de rejeter le paquet. Cette étape utilise une table de flux autorisés maintenue par le plan contrôle.

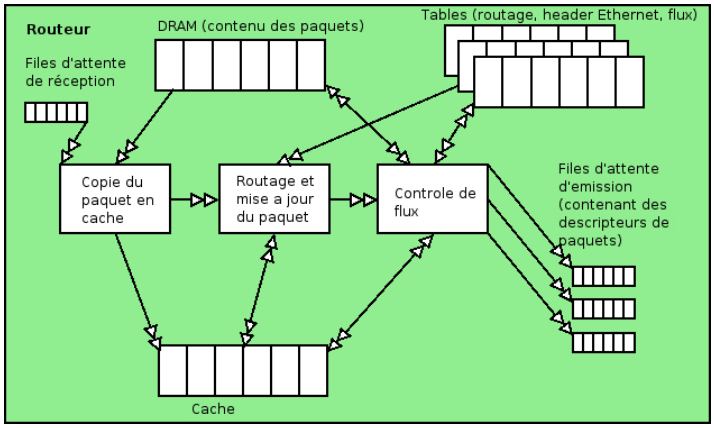

Figure 5: Bloc de routage et contrôle de flux

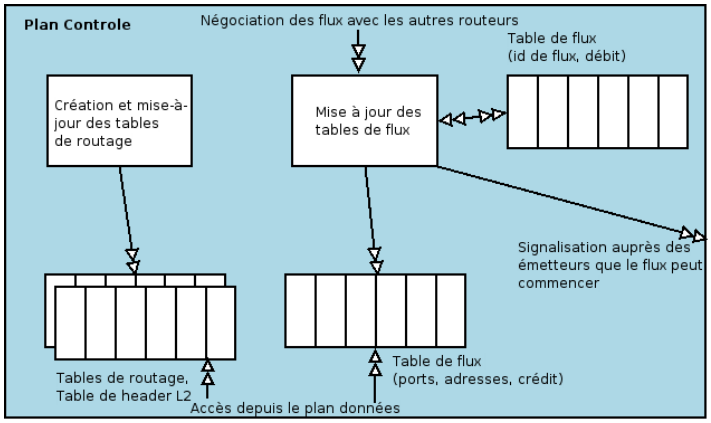

Figure 6: Plan Contrôle

Le bloc fonctionnel de routage et contrôle de flux est le seul qui nécessite des informations provenant du plan contrôle. Le plan contrôle, représenté sur la Figure 6, est responsable des choix des règles de routage et de contrôle de flux. Il a donc le rôle de maintenir à jour les tables de routage, d'en-tête de niveau 2 ainsi que celles de description des flux. Nous distinguons 2

parties dans le plan contrôle : une partie classique concernant la gestion des routes et en-tête de niveau 2, et une partie concernant la gestion des flux. Nous utilisons 2 tables construites par le plan contrôle. La première contient une description des flux (protocole, ports et adresses source et destination) ainsi qu'un compteur de crédit de bande passante (en paquets). Cette table permet le contrôle de flux comme nous l'avons défini plus tôt, c'est à dire la limitation du débit d'un flux dans le temps. La limitation de débit passe par la gestion du crédit de paquets ou de bits par seconde. Le compteur est décrémenté par le plan données lorsqu'un paquet correspond au flux décrit. Si le compteur atteint 0, les paquets sont rejetés. Les compteurs sont crédités par le plan contrôle plusieurs fois par seconde en fonction de la seconde table qui contient la description du profil de trafic accordé à un flux (débit en paquets par seconde, date de début, date de fin). Cette seconde table permet de réaliser simplement la limitation dans le temps. En effet, le compteur de crédit n'est incrémenté que dans la fenêtre temporelle du descripteur de flux et lorsque le flux n'existe pas encore dans la première table ou que son crédit est épuisé, le paquet est rejeté par le plan données.

Pour évaluer la faisabilité et les performances d'une telle architecture, nous avons implanté un prototype à l'aide de l'architecture de processeur réseau IXP2400. Nous détaillons ci-dessous les choix d'implantation des différents blocs. Les *IXP2xxx* sont des architectures de processeurs orientés réseau. Ces plate-formes disposent d'un nombre important (9) de processeurs, de plusieurs interfaces gigabits, ainsi que d'une quantité assez importante de mémoire. L'architecture *IXP* est développée par Intel. La carte que nous utilisons est l'*ENP-2611* de Radisys. Elle est equipée d'un *Xscale* à 600Mhz sur lequel tourne Linux ainsi que 8 MicroEngines[1] qui sont des processeurs dédiés au traitement de paquets pouvant gérer 8 threads. Les threads permettent de masquer les latences des mémoires. Les *ME* ont une mémoire locale de 640 mots de 32 bits. Cette mémoire est la plus rapide (temps d'accès de 3 cycles) mais aussi celle fournie en plus petite quantité. Elle sera donc utilisée dans notre passerelle comme cache par le bloc de routage. Les mémoires présentent dans l'IXP offrent des caractéristiques variées. Le Scratchpad (16ko), est accessible en 60 cycles et permet de créer des *rings* (buffers circulaires) dans lesquels on peut faire des opérations de type *put/get* atomiques. Cette mémoire est donc prédestinée aux files d'attentes. La SRAM fait 8Mo (temps d'accès de 90 cycles environ). Elle est adaptée au stockage des tables. Enfin, la DRAM qui fait 256Mo avec un temps d'accès de 120 cycles est mieux adaptée au stockage des paquets. Dernier point, la carte PCI que nous utilisons, l'*ENP-2611*, possède 3 interfaces optiques Ethernet gigabits/s et une interface 100Mbit/s cuivre pour le développement (boot de l'OS embarqué, accès au système de fichiers en particulier).

Nous n'avons fait que survoler les caractéristiques des IXP pour permettre la compréhension des choix d'implantation faits. Nous utilisons 7 *ME* répartis de la façon suivante : 1 pour *Rx*, 4 pour *Routage*, et 2 pour *Tx*. *Rx* est en assembleur et nous ne le détaillerons pas ici. Notons simplement que ce code place dans un *ring* (dans le Scratchpad) les descripteurs de paquets encodés sur 5 *mots* et ils contiennent les informations permettant de trouver le paquet dans la DRAM. Ces informations sont utilisées par le bloc suivant. Le code du bloc de routage fonctionne sur 4 *ME* (soit 32 *threads*). Un *thread* est responsable du traitement d'un paquet à la fois. Cela consiste à modifier les en-têtes (Ethernet et IP) en fonction de la table de routage, ainsi qu'à vérifier que le paquet est bien dans le profil du flux auquel il appartient, avant de transmettre ce paquet au bloc d'émission (ou de le rejeter). La fonction qui vérifie le profil d'un flux a besoin du quadruplet identifiant le flux auquel appartient un paquet à

[1] noté *ME* par la suite.

partir duquel elle peut rechercher l'entrée correspondante dans la table de flux et décider de transmettre ou de rejeter le paquet. Chaque entré dans la table fait 20 octets. Nous sommes dans un contexte de grille, nous supposons donc que le nombre de flux simultanés ne dépasse pas quelques milliers. La taille de la table n'est donc pas très importante. Notons que nous avons une classe de flux particulière utilisant la première entrée de la table qui est utilisée pour tous les paquets qui n'ont pas de flux décrits dans la table. Afin de conserver l'ordre des paquets, les *threads* des ME de routage sont ordonnés grâce à un signal propagé de thread en thread. En plus d'envoyer les paquets, le bloc *Tx* libére l'espace mémoire pris par le paquet en DRAM. Tout comme le bloc *Rx*, il est écrit en assembleur et provient du SDK Radisys, cependant la version utilisant 2 *ME* fonctionne et c'est donc celle-ci que nous utilisons. Le plan contrôle est situé sur le *Xscale* (donc sous Linux). Commen nous l'avons vu, il réalise deux fonctions qui sont la gestion des tables de flux et la gestion des tables de routage. La première est réalisée par un programme autonome `flow-sch` et la seconde est réalisée par le programme qui charge le code dans les ME. Le programme `flow-sch` correspond à la partie de droite de la Figure 6, il met à jour régulièrement la table de flux.

Le logiciel développé pour ce prototype de passerelle fait environ 20 000 lignes de code. Ce code est écrit dans plusieurs langages : C et assembleur ME, distribué entre plusieurs architectures cibles : *ME*, *Xscale* et entre plusieurs environnements *user land* et *kernel land*.

4 Expérimentations

L'*IXP* dispose de 3 interfaces optiques, nous avons donc utilisé 3 machines comme noeuds d'extrémités. La carte étant située sur une autre machine, il y a 4 machines dans la configuration. Les tests suivants ont été réalisés sur des machines bi-processeur P4 Xeon 2.8GHz avec des noyaux Linux 2.6.12.5.

L'application *StaticFwd* fait partie du SDK Radisys. Dans sa version originale, elle retransmet les paquets entrant par la première interface ressortent par la seconde, ceux entrant par la seconde ressortent par la troisième... et ce sans modification. Cette application utilise 3 *ME*, un pour *Rx*, 1 pour faire le "routage" et 1 pour *Tx* Nous avons modifié ce code pour qu'il n'utilise que les 2 premières interfaces et transmette les paquets entre celles-ci. Le but est d'évaluer les performances de l'*IXP2400* avec le minimum de traitements. Nous avons utilisé *Iperf* pour générer le trafic. Nous pouvons ainsi mesurer le débit en *full-duplex*, que nous pouvons espérer obtenir entre deux machines avec l'IXP au milieu. Du fait du routage statique, nous ne pouvons pas tester entre trois machines. Les débits *full-duplex* obtenus sont de 793+791 Mbits/s en TCP et 819+806 Mbits/s en UDP alors qu'en *back to back*, on obtient en TCP 890+903 Mbits/s et 860+821 Mbits/s en UDP et lorsque les machines sont reliées par l'intermédiaire du switch, on obtient en TCP 863+863 Mbits/s et 808+873 Mbits/s en UDP. Les performances sont donc un petit peu inférieures (10%) lorsque l'on passe par l'IXP avec l'application *StaticFwd*. Comparons maintenant ces résultats avec ceux obtenus sur la passerelle avec gestion des flux.

Pour évaluer la passerelle, nous nous plaçons dans le pire des cas du point de vue de la recherche de flux. Ce cas est celui où il n'y a pas d'entrée spécifique pour le flux considéré et donc celui-ci est comptabilisé avec la classe par défaut. Ce cas est le plus coûteux car il correspond au cas où la recherche du flux échoue. Afin que notre flux ne soit pas limité par le crédit de paquets par seconde, nous mettons le compteur à une très grande valeur. Nous voulons tester les capacités de la passerelle. La passerelle a trois interfaces, il y a une

machine de test à chaque extrémité. Nous génerons le trafic en anneau, chaque machine est configurée pour recevoir le trafic de la machine précédente et pour envoyer du trafic à la machine suivante. Cela dans le but de saturer les 3 liens. Les résultats en UDP que nous avons obtenu sont 793+807+876 Mbits/s. Il n'y a donc pas de perte de performance.

Il nous faut maintenant tester les fonctionnalités de contrôle des fenêtres temporelles. Pour effectuer ce test nous utilisons un trafic *half-duplex* entre 2 machines et lui imposons différentes limites de débits sur différents intervals temporels. Nous remarquons sur le graphique donné à la Figure 7 que le débit n'est pas très lisse. Cela est dû au fait que nous mettons à jour le crédit de paquets par secondes tous les 10^{e}de seconde. Ce paramètre peut être ajusté. Le temps caractéristique de TCP est de l'ordre du RTT (ici 0.2/0.3 ms) donc le débit du flux TCP a le temps de croître du fait de son mécanisme de contrôle de congestion jusqu'à ce que le crédit soit épuisé. Cependant le rôle de notre passerelle est de limiter le débit, pas de le lisser même si cela pourrait être une fonctionnalité intéressante à ajouter. L'émetteur est censé se conformer à l'accord négocié pour son flux, s'il ne le fait pas, cela n'est pas à son avantage puisque certains de ses paquets seront perdus ce qui l'obligera à les renvoyer. Il faut noter que dans le cas de cette expérience, les flux ne sont pas lissés par la source et les RTT sont très faibles (pire cas dans un contexte grille).

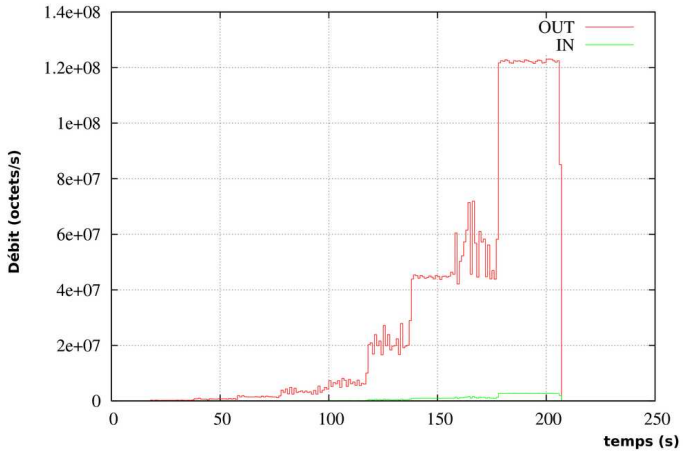

Figure 7: Débit limité par le passerelle

Dernière évaluation, nous effectuons les mêmes tests que pour *StaticFwd* et qu'en *back to back*. C'est à dire que nous utilisons 2 machines reliées par notre passerelle. Nous obtenons en TCP 849+833 Mbits/s et 840+857 Mbits/s en UDP. Ces résultats sont meilleurs que ceux obtenus avec *StaticFwd* alors que le code est plus complexe mais notre code utilise plus de ME. Les résultats que nous obtenons sur l'IXP sont bons, notre passerelle est capable de gérer les 3 interfaces à la fois avec des débits *full-duplex* proches des débits maximaux. Par ailleurs, l'ensemble de ces contrôles se fait sur la carte et ne charge pas le processeur central. L'implantation actuelle de la passerelle n'est pas couplée avec l'algorithme d'ordonnancement de flux. Il faut donc prévoir que celui-ci prendra des ressources supplémentaires or dans l'architecture proposée, sa place est sur l'hôte, il utilisera donc des ressources qui ne sont pas actuellement utilisées et ne perturbera pas le contrôle des flux actifs.

5 Travaux relatifs

Tandis que le partage des ressources de calcul et de stockage ont été largement étudiées [CFK99], l'idée d'intégrer la gestion de la ressource réseau dans les environnements de grille gagne de l'attention. Par exemple, la réservation de ressource réseau a été investiguée dans le contexte des grilles par [FFR+04]. L'architecture de réservation et d'allocation de ressource de Globus (GARA) proposée, introduit l'idée des réservations à l'avance et de la gestion bout en bout de la QoS pour différents types de ressources (bande passante, stockage, calcul). Les algorithmes et les méthodes proposées sont très classiques et se basent sur les principes Intserv et DiffServ . Dans les faits cette proposition n'est pas encore déployée de manière très intensive dans les grilles car les réseaux longues distances n'offrent pas les services requis de bout en bout. Notre approche est beaucoup plus souple dans la mesure où elle se base sur une topologie réseau particulière quoique réaliste sans goulet d'étranglement dans le coeur et s'intéresse plus particulièrement au domaine privé qui présente les liens les plus congestionnés. Le groupe Grid High-Performance Networking (GHPN) du Global Grid Forum (GGF) spécifie actuellement le concept et les interfaces de service réseau de grille (grid network service). Les travaux présentés ici s'intègrent parfaitement dans ce contexte. Le problème du partage optimal de bande passante a aussi été étudié dans les grilles par [BHD03]. Le modèle de réseau et le modèle de réservation adoptés sont différents de ceux proposés dans ce rapport. Par ailleurs, dans le cadre de projet ambitieux d'utilisation flexible des réseaux optiques dans les grilles [Opt05], la problématique de réservation de bande passante dans les mondes optiques et Ethernet est aussi étudiée. Dans le prolongement des études développées dans ce rapport, nous étudions comment adapter notre modèle à ce contexte d'interconnexion avec les réseaux optiques contrôlables par l'utilisateur.

Quelques projets d'équipements d'extrémité ou de bordure ont été réalisés à base de *Network Processors*, on peut noter [NK05]. Les architectures IXP ont été utilisées avec succès pour réaliser des fonctions d'émulation de délai, d'espacement de paquets, d'encryptage à très haut débit (gigabit). Les *network processors* traitant des flux gigabit commence à apparaitre. En ce qui concerne la limitation de débit du coté des hôtes, qui est une des piéces de l'architecture proposée, [TKK+05] propose un mécanisme de limitation et de lissage par flux.

6 Conclusion

Nous avons présenté un modèle d'overlay de contrôle des flux de grille permettant d'accroitre le déterminisme des performances des communications dans une grille en introduisant des mécanismes de réservation de bande passante. Pour implanter cette solution, nous avons décrit une architecture d'équipement de type passerelle de grille qui doit supporter des traitements à très haut débit. Nous avons développé les choix d'architecture et d'implantation de cette passerelle à base de processeurs réseau d'INTEL, IXP2400.

Les *IXP* offrent de nombreuses solutions quant à la disposition des blocs fonctionnels et offre un très haut niveau de performances, cependant le développement dans ce type d'environnement s'avère particulièrement ardu.

Notre passerelle supporte le gigabits/s mais elle ne dispose pas de *slow path* qui permettrait de gérer les paquets qui nécessitent plus de traitement.

Nos perspectives sont l'implantation de l'algorithme distribué d'ordonnancement des flux, le déploiement de cette solution dans le contexte de la plate-forme Grid5000, l'extension du

modèle au cas où le réseau de cœur n'est pas considéré comme sur-dimensionné et au cas où les congestions dans le réseau local sont prises en compte ainsi que le cas où le réseau dispose d'un plan contrôle unifié (GMPLS par exemple) de bout en bout.

References

[BFH03] Fran Berman, Geoffrey Fox, and Anthony J.G. Hey. *Grid Computing: Making The Global Infrastructure a Reality*. 2003. ISBN: 0-470-85319-0.

[BHD03] L. Burchard, H-U. Heiss, and C. A. F. De Rose. Performance issues of bandwidth reservations for grid computing. In *Proc. IEEE the 15th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD'03)*, pages 82–90, November 2003.

[CFK99] K. Czajowski, I. Foster, and C. Kesselman. Resource co-allocation in computational grids. In *Proc. IEEE the eighth International Symposium on High Performance Distributed Computing*, pages 219–228, August 1999.

[Dat05] Datatag. Page web, 2005. http://www.datatag.org.

[FFR+04] I. T. Foster, M. Fidler, A. Roy, V. Sander, and L. Winkler. End-to-end quality of service for high-end applications. *Computer Communications*, 27(14):1375–1388, 2004.

[FJ95] S. Floyd and V. Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM Transaction on Networking*, 3:365–386, August 1995.

[Glo05] Global Grid Forum. Page web, 2005. http://www.ggf.org.

[KMC+00] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.

[MPRZ05] Loris Marchal, Pascale Primet, Yves Robert, and Jingdi Zeng. Optimizing network resource sharing in grids. In *Proceedings of the Intl. Conference IEEE GLOBECOM'05, USA*, November 2005.

[NK05] Kiyohide Nakauchi and Katsushi Kobayashi. Studying congestion control with explicit router feedback using hardware-based network emulator. In *In Proceedings of the International workshop on Protocols for Long Distance Networks (PFLDNET 2005)*, Lyon, France, 2005.

[Opt05] OptIPuter. Page web, 2005. http://www.optiputer.net/.

[Ter05] Teragrid. Page web, 2005. http://www.teragrid.org.

[TKK+05] R. Takano, T. Kudoh, Y. Kodama, M. Matsuda, H. Tezuka, and Y.Ishikawa. Design and evaluation of precise software pacing mechanisms for fast long-distance networks. In *In Proceedings of the International workshop on Protocols for Long Distance Networks (PFLDNET 2005)*, Lyon, France, 2005.

[VTK+06] P. Vicat-Blanc Primet, T. Takano, Y. Kodama, T. Kudoh, O. Gluck, and C. Otal. Large scale gigabit emulated testbed for grid transport evaluation. In *In Proceedings of the International workshop on Protocols for Long Distance Networks (PFLDNET 2006)*, NARA, Japan, February 2006.