Un Modèle de Transfert Haut Débit en Applications pour Grilles de Calcul

Rapport de Stage
Master 2 Recherche: "Systèmes et Logiciels"
Institute d'Informatique et Mathématiques Appliquées de Grenoble

Université Joseph Fourier, Saint Martin d'Hères, France

Carlos Jaime BARRIOS HERNÁNDEZ

Sous la direction de **Yves DENNEULIN**
Laboratoire Informatique et Distribution ID-IMAG

15 juin 2005
Montbônnot-Saint Martin, France

Résumé

Dans le cadre de la recherche pour le développement de systèmes de calcul intensif distribué à grand échelle, la modélisation de la performance permet non seulement la caractérisation d'un système déterminé, mais aussi une spécification pour la construction et l'utilisation des systèmes dérivés. Le transfert de données à haut débit est une caractéristique importante à étudier car elle permet ainsi de connaître les limitations et les possibilités réelles de la performance d'une grille de calcul ou de stockage. A partir de l'exploration théorique des différents modèles nous avons décidé d'analyser les grappes IDPOT et ICluster2 dans l'environemment de grid5000 avec le modèle LogP/LogGP, pour sa capacité à capturer des aspects qui permettent de décrire l'utilisation de réseaux pendant le transfert de données à haut débit. Nous allons présenter les différentes analyses de performance à partir des données trouvées dans la partie expérimentale ainsi que proposer un modèle qui décrit le comportement observé, en accord avec le modèle paramétrisé LogP utilisé pour faire l'évaluation du système et prendre les mesures.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Revisión Bibliographique

Dans le cadre de la recherche pour le développement de systèmes de calcul intensif distribué à grand échelle, la modélisation de la performance permet non seulement la caractérisation d'un système déterminé, mais aussi une spécification pour la construction et l'utilisation des systèmes dérivés. Le transfert de données à haut débit est une caractéristique importante à étudier car elle permet ainsi de connaître les limitations et les possibilités réelles de la performance d'une grille de calcul ou de stockage. Cette première partie du rapport de stage présente dans un contexte particulier la problématique et les différentes stratégies de modélisation et de mesure du transfert de données à haut débit en systèmes de calcul intensif distribués.

1.1 Introduction

Aujourd'hui, l'implémentation de systèmes de calcul intensif pour le traitement de problèmes à grand échelle, tels que les grappes ou les grilles[1], exigent une connaissance des caractéristiques et des limitations tant du matériel que du logiciel, ainsi que des conditions d'utilisation actuelles et de la disponibilité des ressources. Bien que l'utilisation des Grilles de Calcul ait été un défi technologique dédié principalement à des projets de recherche scientifique, l'évolution du calcul de haute performance a permis une transformation des organisations et des projets[9], non seulement dans le domaine éducatif ou scientifique, mais aussi dans l'environnement industriel ou commercial. La définition de Grilles de Calcul[5], suscite déjà, à elle seule, un ensemble de

[1]Clusters and Grids en anglais.

questions techniques et sociales fortement liées, telles que la fiabilité d'accès à la grille, la consistance et la sûreté de l'information et l'efficacité du système par rapport aux besoins des utilisateurs.

Dans ce contexte, l'analyse de la performance de systèmes de calcul intensif s'avère nécessaire afin d'identifier leurs limitations selon les besoins des utilisateurs. Cette analyse permet également de projeter les étapes à suivre pour le développement et l'exécution d'applications des grilles de calcul. Par ailleurs, il ne faut pas oublier le rôle fondamental qui joue la modélisation et l'évaluation de la performance dans toutes les phases du cycle de vie d'un système[18]. Le modèle offre une compréhension du comportement du système et permet de le confronter avec la réalité, de l'évaluer et de le caractériser en accord avec les paramètres de mesure ou classement définis globalement ou particulièrement. D'un autre point de vue, le modèle peut spécifier un système de haute performance pour sa construction, sa diffusion et sa scalabilité.

La modélisation du transfert à haut débit dans un environnement de grille de calcul permet de connaître les caractéristiques du logiciel pouvant être exécutées en termes de taille de données et du temps de communication qui donnera, finalement, une idée de l'efficacité. Il existe déjà quelques expériences documentées sur ce type d'approche et des modèles acceptés pour mesurer le transfert à haut débit. Dans ce rapport nous présenterons une expérience de modélisation et d'évaluation de transfert à haut débit sur un environnement de grille de calcul constitué par deux grappes : ID-Pot et I-cluster2, situées au laboratoire Informatique et Distribution[2] et à l'Institut de Recherche en Informatique et Automatique de la Region Rhône-Alpes[3] respectivement, à Montbonnot, France.

Nous préciserons d'abord le contexte de l'expérience, notamment en ce qui concerne les caractéristiques d'architecture des machines parallèles qui composent la grille et les caractéristiques du logiciel du système. Puis nous définirons la problématique traitée et les stratégies utilisées ainsi que les outils pour mesurer la performance, en accord avec les objectifs du projet. Par la suite nous présenterons quelques modèles pour mesurer le transfert à haut débit que l'on tente d'étudier dans pour cette expérience et enfin quelques conclusions à propos de cette première partie du document.

[2]ID-IMAG : http ://www-id.imag.fr

[3]INRIA Rhône Alpes : http ://www.inrialpes.fr

1.2 Grilles de Calcul

Une grille de Calcul[4] est une *infrastructure* de matérielle et logicielle qui fournit l'accès sûr, constant, compatible et peu coûteux aux possibilités informatiques de très haut niveau ou à haute performance[5]. Des années quatre vingt dix jusqu'à aujourd'hui la discussion autour des objectifs et des politiques d'utilisation des grilles de calcul ont été actives et bien connues des spécialistes, elles ne seront donc pas traitées dans ce document. En revanche, nous ferons une description de l'architecture requise par un système de grille de calcul et tenterons d'établir la différence entre un environnement conventionnel de calcul distribué et un environnement de type grille de calcul.

1.2.1 Architecture de Grille de Calcul

Un consensus se dégage sur le fait que l'infrastructure d'une grille de calcul doit être étudiée du point de vue de l'architecture du système et examinée à l'aide du logiciel nécessaire pour soutenir la grille. A partir de cette perspective, on peut identifier quatre types de systèmes que nécessitent les services et les composants d'une grille de calcul[5] :

- *Les interfaces* qui font le lien entre l'environnement de calcul massif et l'utilisateur,
- *es Grappes de Calcul*, qui permettent l'usage du parallélisme et le calcul entre des systèmes homogènes,
- *l'intranet*, qui permet une approche hétérogène et une distribution géographique,
- *l'internet* qui fournit les moyens pour la commande centralisée et la distribution de l'information.

Chacun de ces composants est un sujet particulier d'étude[5] et on pourrait en dire d'avantage. Cependant, dans le présent rapport nous nous limiterons à essayer d'identifier quelle est l'interaction des composants constituant les services de la grille. À partir de cela, nous pourrons identifier différents modèles d'adoption de grilles[9], à savoir :

- *Infra-Grille*[6] : Ce type de modèle permet de partager les ressources entre les différents départements d'un établissement. C'est un environnement Grille très intégré, généralement homogène et très demandé,

[4]Grid Computing en anglais.
[5]Non seulement dans cette discipline mais aussi dans d'autres domaines.
[6]Infra-Grid.

qui peut être composé de grappes de calcul[7].

- *Intra-Grille*[8] : Ce modèle est plus complexe, et peut être décrit comme un scénario pour l'intégration de ressources de plusieurs de divisions au sein d'une organisation. En fonction du type d'organisation, on peut l'appeler *Enterprise-Grille* ou *Campus-Grid*. Il est important de signaler qu'il existe également une intégration, mais pas forcement une homogénéité de composants. Cependant, cette intégration exige la définition de politiques d'implémentation et de commande.
- *Extra-Grille*[9] : conformément au niveau de complexité, dans ce modèle le scénario d'intégration de ressources a lieu entre l'organisation et une entité externe associée[10] Les conditions d'implémentation et d'utilisation sont définies par les politiques de services des organisations. Il important de dire aussi, que, de façon générale, l'Extra-Grille implique une distribution géographique des ressources et un *accès à distance* entre les organisations. Mais, ceci est à présent un sujet de discussion, car il peut exister une Intra-Grille avec des divisons physiquement distribuées. Dans ce travail nous définirons l'Extra-Grille comme étant un système géographiquement distribué dont le champ d'action est un réseau de type métropolitain.
- *Inter-Grille*[11] : Ce type de grille permet le partage des ressources de calcul et le stockage de données à travers le web public en collaboration avec d'autres organisations ou des particuliers. C'est un service à la demande d'une complexité croissante, ce n'est pas un système homogène et les politiques d'utilisation et de commande ne sont pas complètement définies, bien qu'existe un usage généralisé.

Comme le montre ce classement, lorsque le niveau de complexité augmente, l'infrastructure pour implémenter la grille de calcul est plus complexe. La recherche en ce domaine particulier a défini plusieurs degrés de complexité qui correspondent aux précédents niveaux d'intégration que l'on a déjà décrit dans le présent rapport. Ces catégories sont[9] :

1. Les Grilles pour l'optimisation d'infrastructures.
2. Les Grilles de Calcul avec le traitement virtuel.

[7]Plusieurs auteurs ont appelé ce type de grille des *grappes-grilles* ou *Clusters- Grids* en anglais.

[8]Intra-Grid ou Inner-Grid.

[9]Extra-Grid ou Outer-Grid.

[10]Ce modèle de collaboration est connu sous le nom de *Partner-Grid*.

[11]Inter-Grid.

3. Les Grilles de Données avec la virtualisation de données et le stockage.
4. Les Grilles de Services avec les services virtuels pour faciliter l'intégration.
5. Les applications virtuelles pour la composition de ressources à partir de quelques applications associées à travers d'interfaces de services.

Les catégories précédentes proposent des champs de recherche spécialisée en différents aspects[5], tels que la nature des applications, les modèles, les outils et les paradigmes de programmation, les architectures de systèmes, les méthodes et les algorithmes de résolution de problèmes, la sûreté et sécurité, la gestion de ressources, l'instrumentation, la modélisation et l'analyse de la performance, l'infrastructure et les protocoles de réseau et même des aspects plus sociaux et économiques de l'implémentation de la technologie de grilles de calcul, notamment le concept d'organisation virtuelle.

1.2.2 Calcul Distribué et Calcul par Grille

Les applications pour le calcul distribué comportent un certain nombre de processus coopératifs utilisant les ressources d'un ensemble de systèmes de calcul[17]. Du point de vue du calcul de haute performance, le calcul distribué est possible avec des environnements traditionnels, comme PVM, MPICH, ou avec des grilles de calcul. En effet, un environnement conventionnel distribué suppose un *ensemble*[12] *de noeuds* de calcul à partir desquels est constituée une machine virtuelle concurrente. Dans une grille de calcul, l'environnement est compris comme une *ensemble de ressources* virtuelle. L'accès à la totalité de ressources dans la grille de calcul par les utilisateurs se fait jusqu'à l'ensemble mais pas jusqu'aux noeuds individuels, comme c'est le cas dans des environnements distribués conventionnels. Une autre caractéristique importante est le fait que, dans les grilles de calcul, les ressources sont dynamiques et diverses contrairement aux ressources de calcul distribué. Une description plus détaillée peut être consultée en [17].

1.2.3 Le Contexte de Recherche : Grid 5000

Dans la recherche sur l'implémentation de systèmes de grille de calcul ou de stockage, l'étude du comportement de l'interaction entre noeuds

[12]pool en anglais.

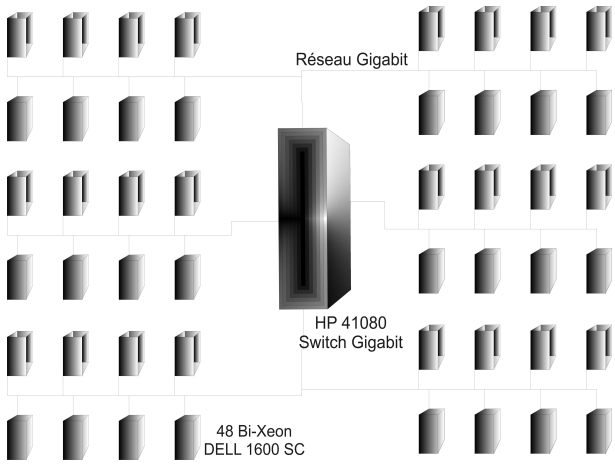


FIG. 1.1 – *La Topologie de la Grappe IDPOT à ID-IMAG.*

hétérogènes est très important, par exemple dans des projets comme GRID5000[13] en France. L'analyse, la modélisation et la mesure proposées dans ce travail ont lieu dans ce contexte. Nous allons maintenant décrire deux grappes que font une *extra-grille*, celle du laboratoire ID-IMAG et celle de l'INRIA, à Montbonnot.

La Grappe IDPOT

IDPOT est une grappe expérimentale au sein du laboratoire Informatique et Distribution. Depuis 2004, IDPOT travaille sous Debian, avec un noyau 2.4.26[13]. La grappe est constituée de 48 x Bi-Xeon 2.5 GHz + 1.5 Go de RAM ECC et 1 switch x 48 ports Gigabit.

La Grappe I-Cluster2

La Grape I-Cluster2 est la première grappe en France sur la base de processeurs Itanium-2[7]. I-Cluster2 est constituée par 104 noeuds reliés par un réseau Myrinet. Chaque noeud est un processeur dual Itanium-2 de soixante quatre (64) bits à neuf cents (900) mégahertz, trois (3) Gigaoctets de mémoire RAM et soixante douze Gigaoctets de disque dur local.

[13]Pour plus d'informations sur le projet GRID5000 se reporter à : http ://www.grid5000.org .

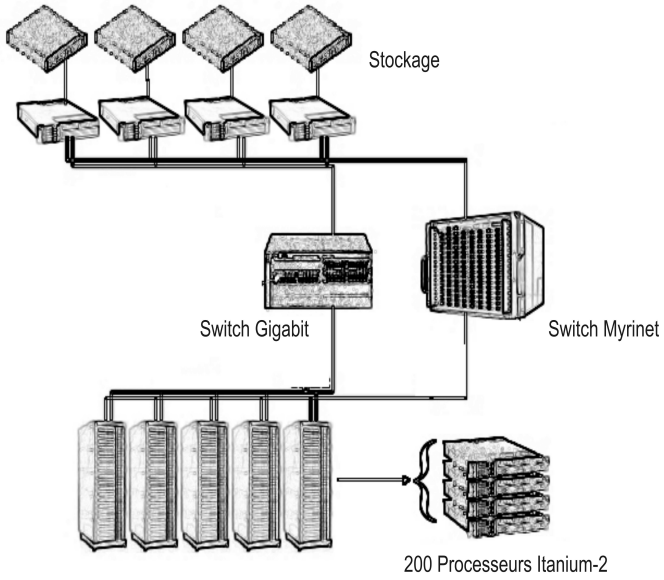


FIG. 1.2 – *La Topologie de la Grappe I-Cluster2 à INRIA Rhône Alpes.*

L'ensemble de la grappe fournit deux cent huit (208) processeurs, trois cent douze (312) Gigaoctets de mémoire et une capacité de disque de sept virgule cinq (7.5) Téraoctets. Cette configuration est complétée avec un serveur de douze (12) Gigaoctets de mémoire, un réseau d'administration, avec des *switches FastEthernet* liés à un *swtich* Gigabit de 24 (vingt quatre ports), et 4 (quatre) disques SCSI d ? ? ?un virgule un (1.1) Téraoctets chacun.

1.3 La Mesure de la Performance

La Mesure de la Performance d'un environnement grille est très liée aux techniques de mesure de la performance en réseau. Il est bien connu que le développement des outils pour mesurer la performance reposent sur des différents principes[19] et, comme il déjà été dit, la mesure de la performance permet non seulement de comprendre le comportement dynamique d'un système pour faire sa spécification mais aussi de déterminer des politiques d'utilisation, à partir de l'évaluation des données de cette mesure.

Les stratégies de mesure posent un problème particulier, car il n'existe

pas une terminologie *standard* ou des modèles de mesure *bien définis* qui permettent la réplication d'une expérience. Cependant, il existe des travaux sur cette problématique tel que ceux réalisés par [19] le groupe de travail sur les mesures du réseau ou NMWG [14] qui peuvent être consultées en [6], et qui sont utilisées dans le présent rapport pour définir les stratégies de mesure à réaliser.

Dans la figure 1.3 on peut regarder les propriétés intrinsèques ayant un rapport avec la performance et la fiabilité d'une entité de réseau, dans ce cas particulier, un système Grille, de forme hiérarchique. Les entités de réseau sont des noeuds et des chemins. Un noeud n'est pas forcement une entité physique, mais il peut être une rangée de dispositifs qui comprennent un switch, un système autonome ou un noeud virtuel. Ainsi, un chemin est une connexion unidirectionnelle entre deux noeuds[6]. La partie gauche de la figure 1.3, définit la structure topologique du système grille, tandis que l'autre partie définit le comportement des données dans le temps. C'est ce domaine spécifique qui est abordé dans ce projet.

1.3.1 Le Transfert de Données à Haut Débit

Dans une grille de calcul ou de stockage, le transfert de données est l'une des caractéristiques les plus importantes et également un défi d'étude. Pour garantir un transfert à haut débit et une disponibilité de communication durable en accord avec les besoins des utilisateurs il faut, comme il a déjà été dit, connaître la performance du système. On peut définir le transfert de données *haut débit*[15] comme une propriété intrinsèque d'une grille de calcul, dont la tâche est de transporter l'information entre les noeuds par des liens entre eux. Autrement dit, on traitera dans ce cas spécifique les relations entre les entités du réseau, voire, la relation entre les entités de la grille.

1.3.2 Les Mesures de la Transfert de Données à Haut Débit

Dans la figure 1.3 sont décrites les caractéristiques de réseau utilisées pour observer le comportement d'un système grille. On peut observer également

[14] Network Measurements Working Group.

[15] il importe de signaler le fait que les données sont toujours considérées comme étant à haut débit entre le contexte de calcul haute performance et le transfert.

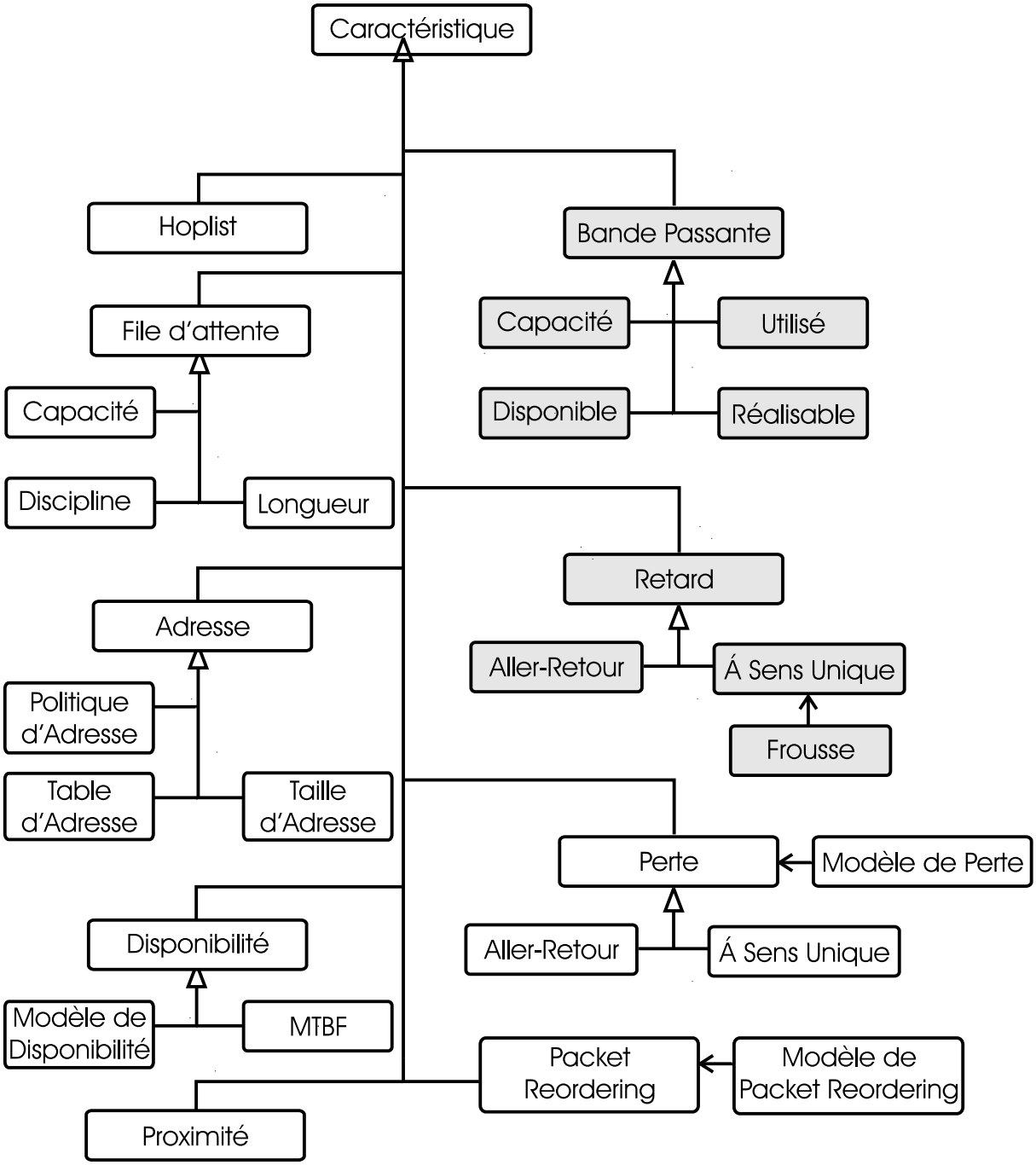


FIG. 1.3 – *Caractéristiques de Réseau qui peuvent être utilisées pour décrire des entités d'une Grille*

Niveau	Description	Exemple
1	Environemment Physique	code et bits dans le fill
2	Cape de lien avec "framing"	Ethernet ou SDN/Sonet
3	Cape de Réseau	IP
4	Cape de Transport	UDP, UDP+RTP, TCP ...

TAB. 1.1 – *Table des couches Réseau selon le Modèle OSI.*

que la partie droite de la figure représente le transfert d'information dans le système. Dans ce cas particulier on va analyser la bande passante et la latence, comme des caractéristiques pour modeler le transfert haut débit dans le contexte d'expérimentation décrit précédemment.

La Bande Passante

La Bande Passante est définie comme la quantité de données transférées par unité de temps. En d'autres termes, il s'agit de la quantité d'information ou des données pouvant circuler dans un médium physique de communication donné. Parmi les caractéristiques qui décrivent la bande passante on trouve :

- *La Capacité* : la quantité maximale de données par unité de temps qui peut être portée dans un lien ou dans un chemin .
- *L'Utilisation* : le trafic global qui existe actuellement sur le lien ou le chemin.
- *La Disponibilité ou la Bande Passante Disponible* : c'est la quantité maximale de données par unité de temps pouvant être fournis par un lien ou un chemin pendant l'utilisation courante.
- *La Possibilité ou la Bande Passante Réalisable* : comme son nom l'indique il s'agit de la quantité de données par unité de temps pouvant être fournies par un lien ou un chemin à une application, en tenant compte de l'utilisation courante, du protocole et du système opératif utilisé ainsi que de la capacité de la performance du destinataire.

Pour chaque caractéristique il est très important de spécifier le niveau, ou cap, considéré sur lequel est faite la mesure, en accord avec le model OSI défini en [8], comme le montre la table 1.1.

Traditionnellement, les processeurs sont interconnectés point à point, par le biais des liens de réseau[6], pour le type d'analyse qu'on va à réaliser, cette considération est insuffisante. Alors on va prendre en compte un modèle de

partage de bande passante[16] où le nombre de connexions est considéré. En [3] est proposé un modèle empirique intéressant pour le partage de bande passante :

$$B = b * (N + n) \tag{1.1}$$

Où B est le taux maximal de transfert de données qui peut être obtenu à partir du lien d'embouteillage dans un chemin ; b est le taux de transfert de données observé par connexion de façon expérimentale, n la quantité de connexions ouvertes ou utilisées pour l'application de la grille en usage pour la mesure, et N les autres connexions utilisées dans d'autres applications du même réseau.

Bien que de façon générale le taux maximal de transfert soit donné par le fabricant, l'intérêt de l'équation 1.1 est que l'on peut calculer une valeur empirique rapprochée de la valeur réelle. Ainsi, selon le modèle typique la bande passante partagée peut être : $\beta = \frac{B}{n}$, où n est le nombre de connexions qu'utilise le lien et B, le taux maximal de transfert de données pour chaque connexion[3].

Un autre aspect important est la Bande Passante Agrégée, qui est la quantité totale de données efficaces transférées pendant la durée d'une expérience[14], et que l'on peut exprimer comme :

$$BPA = \frac{n * b}{t} \tag{1.2}$$

dans l'équation 1.2 t est considéré comme le temps de l'expérience, n le nombre de clients ou connexions utilisés pour l'application du grille, et b la taille de données par client ou β par client. Cette mesure peut être exprimée en termes de bits/secondes ou en hertz.

La Latence

Dans le contexte de Grille de Calcul, la latence est considérée comme synonyme de retard[17] On va définir la latence comme le temps que prend un paquet de données pour aller d'un point indiqué à un autre. En plusieurs utilisations, la latence est calculée avec l'envoi d'un paquet de données qui retournera à l'expéditeur la mesure de temps d'aller-retour étant considérée

[16]Bandwidth Sharing en anglais.
[17]Delay en anglais.

comme la latence. Une modèle mathematique bien connu pour décrire le temps T requis pour envoyer x octets de données à un lien de réseau est l'équation suivante :

$$T = \alpha + \frac{x}{\beta} \tag{1.3}$$

Oú α est la latence et β la bande passante[3].

A partir des équations 1.1 et 1.3 en [15] est proposé un modèle decrit par l'expression suivante :

$$T = \alpha_{exp} + \frac{x}{\beta_{exp}} \tag{1.4}$$

Où α_{exp} est la latence expérimentale, qui est déterminée par une fonction linéaire de la latence physique pour la latence simulée, et β_{exp} qui est aussi une valeur expérimentale.

1.4 Les Modèles et Techniques de Mesure de Transfert à Haut Débit

Pour faire la mesure et l'analyse de la performance de transfert haut débit, il existe différents modèles et techniques proposés par plusieurs groupes de recherche. En fait, le transfert de données n'est pas seulement un problème des grilles de calcul et son étude en ce contexte est un héritage de la recherche en systèmes distribués, en réseaux de communication de données, comme Internet. Dans cette partie, nous présenterons seulement les modèles que nous avons considérés proches de la problèmatique choisie, car d'autres modèles existent plus anciens et plus variés qui sont aussi importants et peuvent être trouvés dans la bibliographie specialisée.

1.4.1 Le Modèle Traditionnel de SIMGRID

SIMGRID est un ensemble d'outils developpé pour la simulation des applications en environnements distribués et hétérogènes de calcul[2]. Aujourd'hui, est disponible la version 2-93[18]. Avant, nous avons présenté les mesures bien connues de latence et de bande passante. SIMGRID utilise le modèle

[18]SIMGRID 2-93 est disponible à http ://gcl.ucsd.edu/simgrid/dl/ .

présenté dans l'équation 1.3, comme un ensemble de liens de réseau en trois modes pour multiples transferts[3]. Mais, le modèle traditionnel de SIMGRID, bien que simple, a de sérieuses limitations en termes d'utilisation et de réalisme qui sont présentées en [3]. Comme ébauche de solution aux limitations est proposé le modèle expérimental qu'on a cité avant et représenté par l'équation 1.4.

1.4.2 LogP, LogP Paramétrisé et LogGP

Le modèle LogP[4] fournit un mécanisme de capture des aspects pertinents de passage de message en architectures à mémoire distribuée. Une méthode efficace d'implantation du modèle LogP est presentée en [10] pour faire les mesures des paramètres de LogP en messages de différentes tailles. Le modèle présente le nombre de processeurs comme P, la latence de réséau comme L, o est le temps qu'un processeur passe pour envoyer ou recevoir un message. Finalement, g est défini comme l'intervalle minimal de temps entre deux transmissions ou émissions consécutives de message pour un processeur, telle que la valeur réciproque réalisable de bande passante *end-to-end*. LogP est surtout pour messages courts, cas où o et g sont alors constants.

Une paramétrisation en LogP, définit cinq paramètres. Comme LogP, les paramètres P et L sont définis respectivement comme le nombre de processeurs et la latence, mais apparaissent aussi les paramètres $o_s(m)$ et $o_r(m)$ qui sont les temps CPU utilisés pour envoyer et recevoir un message de taille m. $g(m)$ est l'intervalle de temps minimal entre deux transmissions ou réceptions consécutives. Comme on peut l'observer, $g(m)$ implique tant $0_s(m)$ que $o_r(m)$, bien que finalement $o_s(m) \leq g(m) \geq o_r(m)$. On peut ainsi caractériser un réseau avec $N = (L, o_s, o_r, g, P)$.

Le modèle LogGP[1] est une incorporation du modèle LogP mais pour des messages longs. La définition de L, o, g et P est la même, avec en plus le paramètre G qui est l'espace par octet pour des messages longs. Il est aussi possible de représenter LogGP sous une forme parallèle [1], comme le montre la table 1.2.

Dans [21] se trouve un intéresant ensemble d'équations pour la comprehénsion de l'importance du paramètre G dans le modèle LogGP.

LogP/LogGP	LogP paramétrisé
L	$= L + g(1) - o_s(1) - o_r(1)$
o	$= (o_s(1) + o_r(1))/2$
g	$= g(1)$
G	$= g(m)/m$, pour un message m suffisamment long
P	$= P$

TAB. 1.2 – Le Modèle LogGP exprimé en termes de LogP parametrisé.

1.4.3 Le Modèle de Retard de Paquet : *Packet Tailgating*

[11] décrit un modèle déterministe de retard de paquets généré par une technique appelé *Packet Tailgating*, pour réaliser la mesure de bande passante par lien. Le modèle construit est un modèle multi-paquet qui unifie des modèles de un paquet et de paires paquets, qui sont bien décrits dans l'article.

Le modèle est représenté par l'équation :

$$t_l^k = t_0^k + \sum_{i=0}^{l-1} \left(\frac{s^k}{b_i} + \alpha_i + q_i^k \right) \tag{1.5}$$

Cette expression prévoit que le paquet k arrive au lien l en un temps de transmission t_0^k plus la somme de tous les latences α_i, les retards de transfert $\frac{s^k}{b_i}$ et les retards en ligne q_i^k des liens précédents. Les auteurs ont modélisé ce dernier élément avec l'équation :

$$q_l^k = max\left(0, t_{l+1}^{k-1} - d_l - t_l^k\right) \tag{1.6}$$

L'équation explique que le paquet k est aligné dans le routeur avant le lien l depuis le moment de son arrivée à ce *router* t_l^k jusqu'au début de la transmission. Aussi elle exprime quel est le temps quand le paquet précédent $k-1$ arrive au prochain routeur t_{l+1}^{k-1} moins la latence de ce lien d_l, avec la supposition que le premier paquet n'est pas déjà en ligne.

Avec la combinaison des équations 1.5 et 1.6, on obtient :

$$t_l^k = t_0^k + \sum_{i=0}^{l-1} \left(\frac{s^k}{b_i} + \alpha_i + max\left(0, t_{l+1}^{k-1} - d_l - t_l^k\right) \right) \tag{1.7}$$

Suite à ce modèle, des auteurs ont proposé une technique, appelé *Packet Tailgating* qui permet la mesure de caractéristiques en deux niveaux : au niveau de l'environemment complet du réseau et au niveau des liens. Pour ces mesures a été construit le prototype `nettimer`. La technique dérivée et son prototype du modèle proposé en [11] est robuste parce qu'elle permet la mesure de liens multicanaux, est considéré plus rapide et moins intrusive que d'autres techniques d'un paquet ou par paires de paquets. Mais il y a aussi d'importantes limitations comme, par exemple, ne pas pouvoir faire une mesure asynchrone ou qu'une faite en quelque lien du chemin peut perturber les mesures sur tous les liens. Les auteurs de [11] proposent différentes stratégies de solutions aux problèmes identifiés.

1.4.4 Autres Modèles et Techniques

D'autres modèles et techniques que nous allons présenter ont été développées pour faire des mesures spécifiques aux architectures distribuées et peuvent être utilisés comme stratégies d'analyse et de caractérisation de système de grille. En fait, pour le traitement de notre problème, tous les derniers modèles et techniques de mesure de la performance peuvent être utilisées.

Le modèle LoGPC

Ce modèle est proposé por Moritz et Frank dans [16] pour faire la mesure de la performance en environnements de passage de messages. Le modèle LoGPC est une extension des modèles LogP et LogGP, mais loGPC utilise un paramètre C qui représente le degré de contention[19]. Ce modèle est une approximation intéressante de la modélisation de la compétition pour des ressources entre des entités dans un système de grille.

Le modèle de Latence et RTT

Le RTT[20] est un aspect fondamental des modèles de transfert pour TCP. Intuitivement, le RTT peut être consideré comme deux fois la latence d'un lien dans une simple connexion ou deux fois la somme de toutes las latences

[19]Le degré de contention est le nombre de noeuds qui essayent de transmettre un message à travers le même lien ou chemin en même temps.

[20]Round Trip-Time en anglais : c'est le temps entre l'envoi d'un paquet depuis une source à une destination et la réception du paquet réponse pour la source depuis la destination.

des liens dans une réseau composé. Un modèle de RTT pour une route r est donné dans [3] :

$$RTT_r = \beta * 2 * \sum_{l \in r} Lat_l \tag{1.8}$$

Dans ce modèle β est un facteur multiplicateur, Lat_l est la latence pour chaque lien l.

Le modèle de Bande Passante Partagée

Dans le même document [3] est presentée la bande passante empirique telle que citer auparavant, comme un flux d'un lien simple limité pour la RTT. Les auteurs on proposé un modèle pour liens multiples avec plusieurs flux. Dans le cas de la bande passante partagée il est important dire que chaque flux est en compétition, il est donc évident qu'il existe à un moment un lien formant un goulet d'étranglement. Un lien est un tel goulet si la somme de toutes les bandes passantes localisés dans les routes des liens est égale au total de la bande passante du lien. Une representation plus formelle du goulet d'un lien l peut être :

$$\forall l \in r, if \sum_{l \in r} \lambda_r = C_l \tag{1.9}$$

Où $lambda_r$ est le transfert de données pour r, $linr$ est un ensemble de routes et C_l la capactié par lien avec toujours $C_l > 0$.

Le goulet d'étranglement équitable est un goulet l, pour lequel la bande passante est partagée de manière équitable :

$$\forall r \ni l, \lambda_r = \frac{1/RTT_r}{\sum_{r' \ni l} 1/RTT_{r'}} * C_l \tag{1.10}$$

Après ces définitions, dans [3] sont proposées cinq modèles différents de bande passante partagée :

– *Latence Inverse* : Pour chaque flux r_i, la bande passante partagée dans le lien en goulet est w_i, tel que :

$$w_i = \frac{1}{\sum_{l \in r_i} Lat_l} \tag{1.11}$$

– Latence Inverse Limitée : est similaire à la dernière expression avec la condition que la bande passante pour le flux r_i,BW, est limité par γ qui est la bande passante maximale.

$$BW_{maxr_i} = \frac{\gamma}{\sum_{l \in r_i} Lat_l} \tag{1.12}$$

– *RTT Inverse* : C'est similaire que la Latence Inverse mais la somme est remplacée par RTT_{r_i}

$$w_i = \frac{1}{RTT_{r_i}} = \frac{1}{\beta * 2 * \sum_{l \in r} Lat_l} \tag{1.13}$$

– *RTT Inverse Limité* : C'est pareil que la latence Inverse limitée mais la somme est remplacé par RTT_{r_i}.

$$BW_{maxr_i} = \frac{\gamma}{RTT_{r_i}} = \frac{\gamma}{\beta * 2 * \sum_{l \in r} Lat_l} \tag{1.14}$$

– *MaxMin Equitable* :C'est le traditionnel MaxMin équitable, où tous les flux sont égaux au goulet d'étranglement partagé, $w_i = 1$, alors :

$$\forall r \in R, \exists l \in r, \sum_{r' \ni l} \lambda_{r'} = C_l et \lambda_r = max\{\lambda_{r'}, r' \ni l\} \tag{1.15}$$

Où R est l'ensemble de routes et C_l la capacité par lien.

1.5 Conclusion

De cette première partie on peut conclure les aspects suivants :

– Plusieurs modèles et représentations de comportement pour faire la mesure et l'analyse de performance existent, mais chacun est un cas particulier d'analyses. Des modèles traditionnels plus généraux peuvent être utilisés comme point de départ pour ces cas particuliers.

– Le contexte de recherche est celui de la problématique du transfert à haut débit sur des systèmes de grilles de calcul et de stockage. Alors, le domaine plus spécifique d'analyses est la modélisation de la bande passante et de la latence.

– La représentation paramétrée permet la description des entités du réseau analysé dans le système, mais il faut aussi regarder les niveaux de mesure en accord avec le modèle d'architecture de grille de calcul en étude, pour connaître les limitations tant du système que des simulations obtenues pour le modèle et aussi de la confrontation avec les mesures réelles.

Chapitre 2

L'Expérimentation

Dans cette partie nous allons décrire les expériences réalisées pour le développement du projet, telle comme la mesure de *gap*, *overhead time sender*, *overhead time received* et la *latence*. Ces expériences ont pour but de mettre en évidence le comportement réel du système et présentent des premières analyses avant de construire les modèles de performance de l'infrastructure étudiée qui sont présentés dans la partie suivante.

2.1 Description des expériences

Dans le chapitre précédent, nous avons exposé les différents modèles mathématiques pour l'analyse de la performance en grappes et grilles de calcul et leur relation avec les techniques de mesure en réseau. Nous avons aussi détaillé l'intérêt principal de l'analyse et la connaissance du transfert de données à haut débit dans le cadre de recherche. A partir de l'exploration théorique des différents modèles nous avons décidé d'analyser les grappes avec le modèle *LogP*[4], pour sa capacité à capturer des aspects qui permettent de décrire l'utilisation de réseaux pendant le transfert de données à haut débit.

Principalement, l'expérience a consisté en l'exécution d'un code connu comme `logp_multitest` dans l'environnement d'étude. Le code, comme nous allons l'expliquer plus en détails ensuite, fait partie d'un outil plus complexe pour l'implémentation du modèle LogP par un système à passage de message, qui s'appelle *LogP MPI*.

Nous avons sélectionné deux grappes pour faire les mesures : IDPOT et ICluster2. Chacune d'elles ont des caractéristiques très particulières tant du

point de vue architecture que logiciel mais les deux font partie de l'ensemble de ressources du projet *Grid5000*, comme décrit dans la première partie. Sur chacune des grappes nous avons fait des expériences sur l'environnement OAR, pour l'utilisation de noeuds réservés. Nous avons fait la mesure depuis 2 jusqu'à a 32 sur deux types d'expériences : la première pour le transfert de donnés de taille fixe et la seconde pour le transfert de donnés de taille variable. Nous avons fait des expériences préliminaires avec des messages de taille très petite pour connaître le comportement général de l'outil et du système de communication, particulièrement sur la grappe IDPOT, mais nous ne donnons pas de résultats ici car notre principal intérêt est le transfert de messages de grand taille.

Les tailles de messages pour les expériences de taille fixe sont de1 Mega octets (1048574 octets), 10 Mo (14857400 bytes), 50 (52428700 bytes) et 100 Mo (104857400 bytes). Les tailles pour les expériences avec messages de taille variable vont de 1 Mo à 100 Mo, en intervalles donnés par l'outil.

Il faut préciser que les expériences ne sont pas nécessairement faites pendant un temps de non utilisation du système pour deux motifs : le premier est que dès qu'il y a des processus actifs dans l'environnement il y a toujours transfert d'information, le deuxième, est qu'il est important pour pour nous de connaître la performance réelle du système pendant son utilisation pour faire une comparaison entre le modèle théorique et les données expérimentales.

La figure 2.1, présente le modèle que nous avons utilisé et qui est bien décrit dans [4]. La latence L peut être observée comme le temps qui s'écoule entre l'envoi du premier bit d'un message de taille m depuis l'expéditeur P_0 jusqu'au récepteur P_1. $s(m)$ et $r(m)$ sont le temps d'envoi et de réception du message quand les deux commencent leurs opérations simultanément. $s(m) = g(m)$ est le temps auquel l'expéditeur est prêt à envoyer le prochain message. $r(m) = L + g(m)$ est le temps pendant lequel le message est reçu par le récepteur. Chaque fois que le réseau lui-même est le goulot d'étranglement de transmission, $O_s(m) < g(m)$ et l'expéditeur peut continuer de calculer après le temps $O_s(m)$. Rappelons que $O_s(m)$ et $O_r(m)$ sont l'overhead d'envoi et de réception, respectivement. En autres mots, $O_s(m)$ et $O_r(m)$ sont les temps que le processeur, des deux côtés, sont occupés à envoyer et recevoir un message de la taille m. L'espace[1] $g(m)$ est l'intervalle minimum de temps entre la transmission ou la réception de messages consécutifs. De ces dernières

[1]D'après l'anglais *gap*.

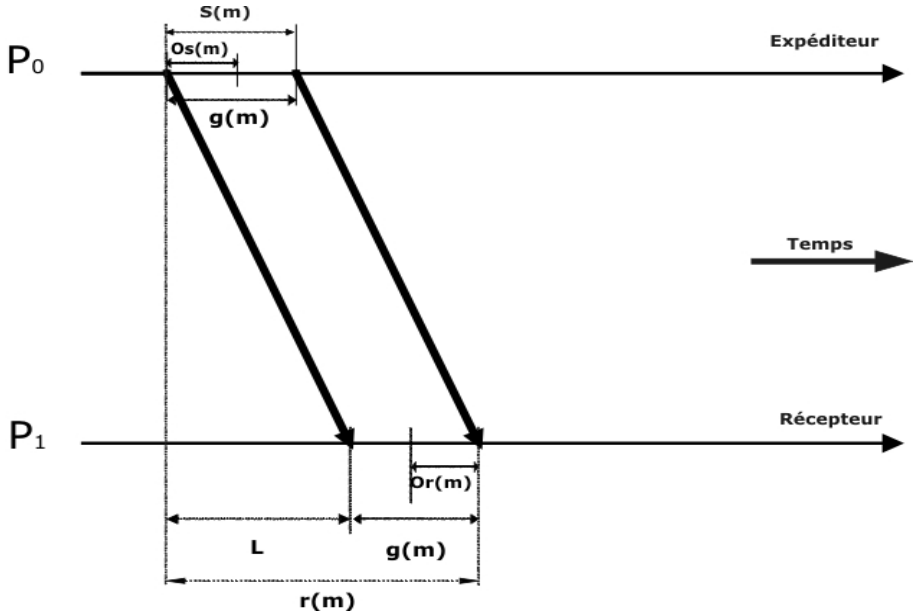


FIG. 2.1 – *Le Modèle LogP.*

descriptions on peut déduire que, pour des messages suffisamment grands, il est possible de commencer quand l'expéditeur est encore occupé, alors $O_s(m)$ et $O_r(m)$ peuvent se recouvrir.

Fondamentalement, l'expérience a utilisé le code de *LogP/MPI* du modèle *LogP* pour envoyer un message d'une taille déterminée ou variable et observer les paramètres fournis pour regarder la performance du système.

2.1.1 LogP/MPI 1.3

L'outil LogP/MPI est un outil décrit dans l'article [10][2], pour faire des mesures de benchmark en environnements distribués.

LogP/MPI évalue l'exécution des messages envoyés et reçus pour des communications données de MPI. L'exécution est exprimée en terme de modèle paramétrisé de LogP, pour des messages de diverses tailles. L'exécution mesure entre une paire de processus en supposant que la communication dans les deux sens est symétrique. Les données sur l'exécution peuvent être sauvées,

[2]L'outil et sa documentation sont disponibles sur le site http ://www.cs.vu.nl/albatross/

classées et être rechargées pour une réutilisation. Les auteurs de LogP/MPI ont fournit une *API* pour rechercher des paramètres de LogP pour différentes tailles de message données pour faire des requêtes parallèles MPI adaptées aux coûts de communication.

LogpMultitest

`LogP_multitest` est un programme pour faire des mesures en parallèle proposé par Luiz-Angelo Estefanel du Laboratoire ID-IMAG[3]. Le programme permet de connaître le comportement des paires de processeurs dans l'environnement et d'envoyer et de recevoir des messages de taille fixe ou variable.

Les options utilisées sont les mêmes qu'on peut trouver dans le programme `logp_test` et qui sont bien détaillées dans le fichier README de l'outil, nous détaillerons les suivantes :

- Send : Indique qu'il utilisera un appel MPI_Send bloqueant.
- Recv : Indique l'utilisation similaire à MPI_Recv.
- -min-size : La plus petite taille de message à envoyer.
- -max-size : La plus grand taille de message à envoyer.
- -o : Indique le fichier de sortie. Chaque fichier de sortie décrit la mesure pour une paire de processus. Par exemple, un fichier s'appellera ficher.Id1.Id2.

Il est important de noter que la taille de la mesure est exponentielle, si bien que, par exemple, pour faire une mesure de 1 Mega Octets comme taille minimale, il faut spécifier une taille de `... -min-size 1048574*1048574...`.

2.1.2 Les Commande utilisées

Pour les expériences, la connexion à la grappe se fait par ssh, par exemple, pour la connexion avec IDPOT :

```
[barrios@wayama tmp]$ ssh cbarrios@frontal38
Password:
Last login: Fri Apr 29 10:00:37 2005 from nfs38
#  Bienvenue sur Frontal38 (IDPOT) (Grenoble - cluster bi-xeon)
###############################################################################
# Documentation : https://frontal38.imag.fr/
# Wiki de G5K   : https://helpdesk.grid5000.fr/
# Status de G5K : https://frontal38.imag.fr/cgi-bin/oargridmonika.cgi
# Accès G5K     : ssh oar.sites avec sites = (bordeaux|lyon|orsay|rennes
#                 |sophia|toulouse)
###############################################################################
```

[3]http ://www-id.imag.fr/Laboratoire/Membres/Estefanel_Luiz-Angelo

```
#
# ATTENTION : Des modifications auront lieu prochainement
#       - changement probable dans le nommage du cluster
#       - intégration du icluster2 dans Grid5000
#
#               Bon Taf !
#
cbarrios@nfs38:~$
```

Nous devons ensuite utiliser oar, le gestionnaire de travaux pour soumettre des travaux sur une grappe ou la grille.

La commande principale pour exécuter le code est :

```
mpirun -np XX logp_multitest -min-size=T*T -max.size=T*T -o XXpmultitest_1...
```

Les X indiquent la quantité de processeurs utilisés et T est la taille du message.

La sortie d'information est à travers d'un fichier de données en accord avec les paramètres de la commande, par exemple pour une mesure avec 2 processeurs, pour une taille fixe de 1 mega octet :

```
mpirun -np 2 logp_multitest -min-size 1048574*1048574 -max-size 1048574*1048574 -o 2plogp_multitest1Mb
```

La première partie de la commande est une instruction d'appel à *mpi*, pour l'exécution de code avec en paramètres le nombre de processeurs et le programme à exécuter. La deuxième partie limite la taille de message minimale et maximale, en ce cas, une taille fixe. Finalement la dernière partie est le nom de l'archive de sortie. La sortie pour cette commande sera dans le fichier `2plogp_multitest1Mb.0.1` qui contient :

```
# LogP network performance data: logp_test.Send.Recv.0.1
# Latency =      1.82
# time      bytes os        os_min    os_cnfint or        or_min    or_cnfint g
1116318384       0 0.0000004 0.0000000 0.0000002 0.0000003 0.0000000 0.0000002 0.0000004
1116318384 1048574 0.0013042 0.0011575 0.0008000 0.0037349 0.0036694 0.0003256 0.0033303
```

La première ligne est la description de la commande, la deuxième la latence mesurée au moment de faire l'essai. Les données sont organisées en colonnes : le temps (en microsecondes), la taille de message (en octets), l'overhead d'envoi, avec des mesures détaillées de Os minimum et maximum, ainsi que l'overhead de réponse avec des mesures minimale et maximale et finalement, le "gap" ou espace minimal de temps pour envoyer ou recevoir un message. Les données sont présentées en deux lignes, une avec une taille 0 et l'autre avec la taille demandée.

Pour faire les courbes, nous avons fait une réorganisation des données en tableaux résumés, où est utilisée seulement la deuxième ligne effective ainsi que la mesure de la latence et le nombre de processeurs utilisés. Un fragment d'un tableau résumé est :

```
#RESULTATS DE LOGPMULTITEST AVEC 51200 bytes, X Proccesseurs
#COMAND: ./mpirun -np xx logp_multitest size-min=51200*51200
# time      bytes os        os_min    os_cnfint or        or_min    or_cnfint g L np
#
#np = 2
# LogP network performance data: logp_test.Send.Recv.0.1
1114692549   51200 0.0001008 0.0000984 0.0000034 0.0001275 0.0001253 0.0000060 0.0001421 2.12 2
#
#np = 4
# LogP network performance data: logp_test.Send.Recv.0.1
1114692600   51200 0.0003168 0.0001282 0.0001947 0.0002692 0.0000893 0.0003812 0.0007016 1.93 4
# LogP network performance data: logp_test.Send.Recv.2.3
1114692600   51200 0.0000755 0.0000565 0.0000342 0.0001575 0.0001204 0.0000366 0.0001284 0.00 4
#
#np = 8
# LogP network performance data: logp_test.Send.Recv.0.1
1114692613   51200 0.0003092 0.0000674 0.0003868 0.0018233 0.0007843 0.0007926 0.0011752 12.75
8
# LogP network performance data: logp_test.Send.Recv.2.3
1114692613   51200 0.0000876 0.0000775 0.0000050 0.0001215 0.0001204 0.0000039 0.0001125 206.07
8
# LogP network performance data: logp_test.Send.Recv.4.5
1114692613   51200 0.0000505 0.0000484 0.0000030 0.0001011 0.0000875 0.0000271 0.0000828 0.00
8
# LogP network performance data: logp_test.Send.Recv.6.7
1114692613   51200 0.0001983 0.0000324 0.0002379 0.0013709 0.0001555 0.0008047 0.0006322 0.00
8
.            .     .
.            .     .
.            .     .
```

Ces tables fournissent la base de données principale pour les expériences et l'analyse des résultats. Chaque taille de message est identifiée avec une couleur différente.

Les graphiques présentées dans ce document sont construits avec l'outil *gnuplot*[4]. Avec ces données il est possible de construire des fonctions à partir de chacune des mesures sans traitement statistique. Cela montre le comportement de l'infrastructure, pendant l'expérimentation. Nous reviendrons sur cet aspect dans la suite.

[4]Pour plus information http ://www.gnuplot.info

2.2 Les Experiences sur IDPOT

Dans cette partie nous allons présenter les résultats des expériences obtenues sur la grappe IDPOT en deux parties : avec taille fixe et taille variable. Chaque expérience a été faite sur les grappes dans les conditions décrites auparavant.

2.2.1 Taille Fixe

Les expériences avec taille fixe sont présentées sur les figures : 2.2, 2.3, 2.4, 2.5 pour les mesures d'overhead d'envoi et de réception. Pour les mesures de gap ce sont les figures : 2.6 et 2.7.

Dans la figure 2.2 on peut regarder comm'exist de temps semblables entre pairs de processeurs jusqu'a ving deux (22) processeurs. À partir d'ici, la *distance* entre points agumente mais n'est pas un croissance en permanence, comme pour construir une fonction complêtemment linéale ou complêtemment exponentielle. La tendance est visible encore dans la graphique 2.3. Autre aspect interessant est que l'ordre de grandeur entre données de chaque figure, si bien il y a un croissance, est equivalent. Contraire dans les graphiques 2.4 et 2.5 où on peut observer un change d'ordre de grandeur des valeurs important.

En la figure 2.4 s'observe comm'il y a un tendance plus lineal et seulement au final exist d'espace considerable entre mesures de pairs de processeurs. On fait, est interessant regarder la variance de compartemment entre tailles differents de messages, parce que pour générale s'espére une croissance progresive mais on peut regarder que pour chaque taille il y a d'augmentation des valeurs de distribution des données differente.

En la graphique 2.5 on peut regarder comme les points preséntent une distribution plus inestable que dans les dernières figures. La première explication est qu'en pour le modèle LogP, avec messages de grand taille, peut être que le transfert commence quand l'expediteur soit occupé, comm'est expliqué dans [10] et [21][5].

Les graphiques 2.6 et 2.7 d'espace de temps entre deux transmissions consécutives des messages pour pair de processeurs montre comm'au augmenter la taille de message, le temps augmente de forme important. Évidentemment

[5]Encore, il faut de pris en compte l'utilisation de réséau pendan les epreuves, comm'on montre dans le graphique 2.8 des latences experimentées.

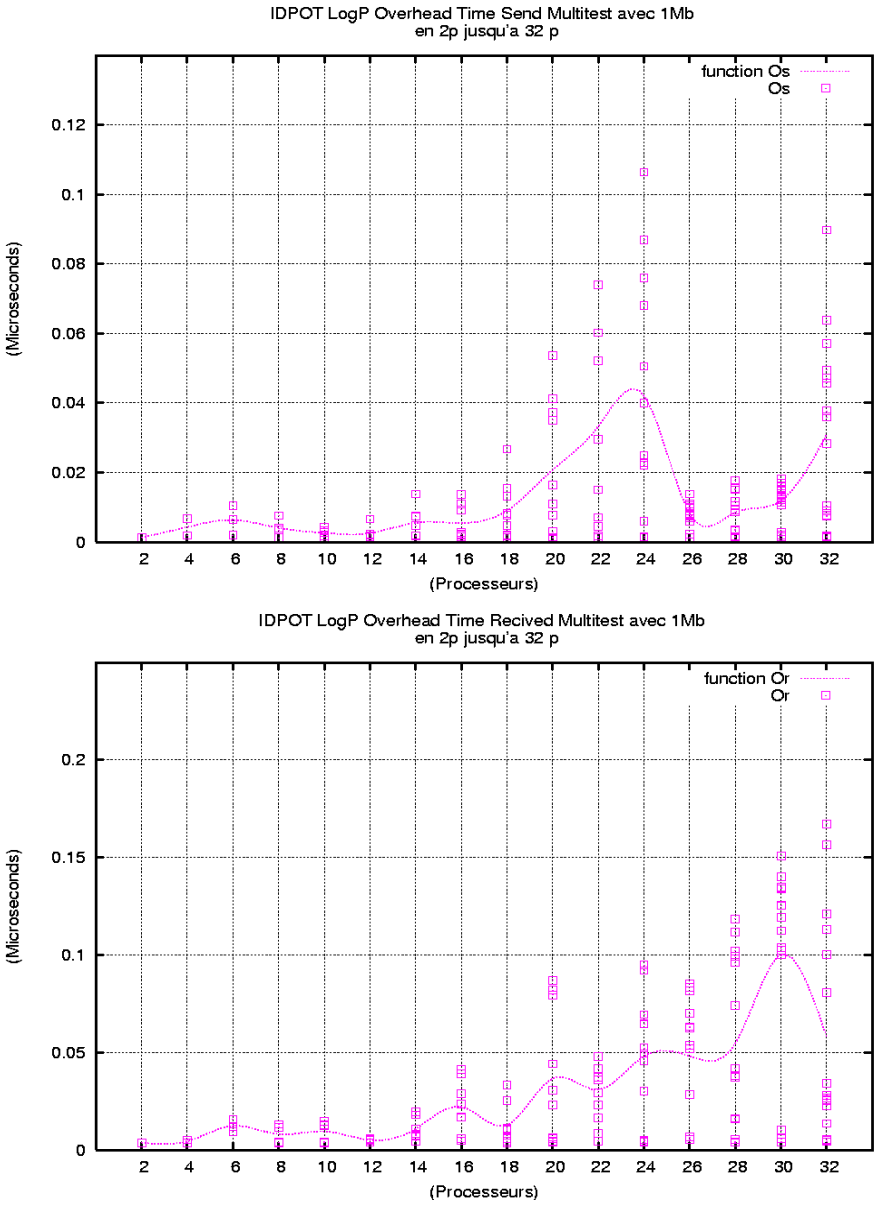


FIG. 2.2 – *Experience de mesure de Os et Or pour un message de taille 1 Mo.*

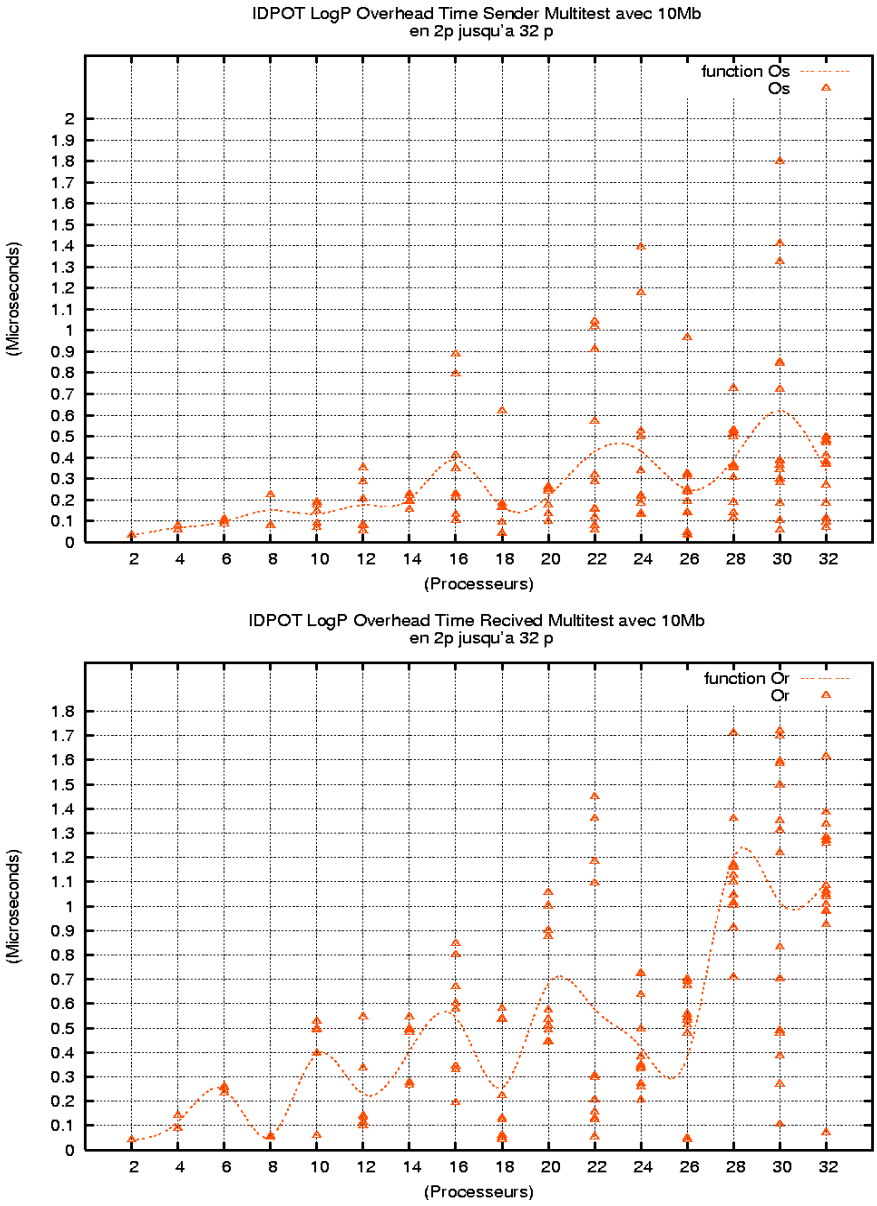


FIG. 2.3 – *Experience de mesure de Os et Or pour un message de taille 10 Mo.*

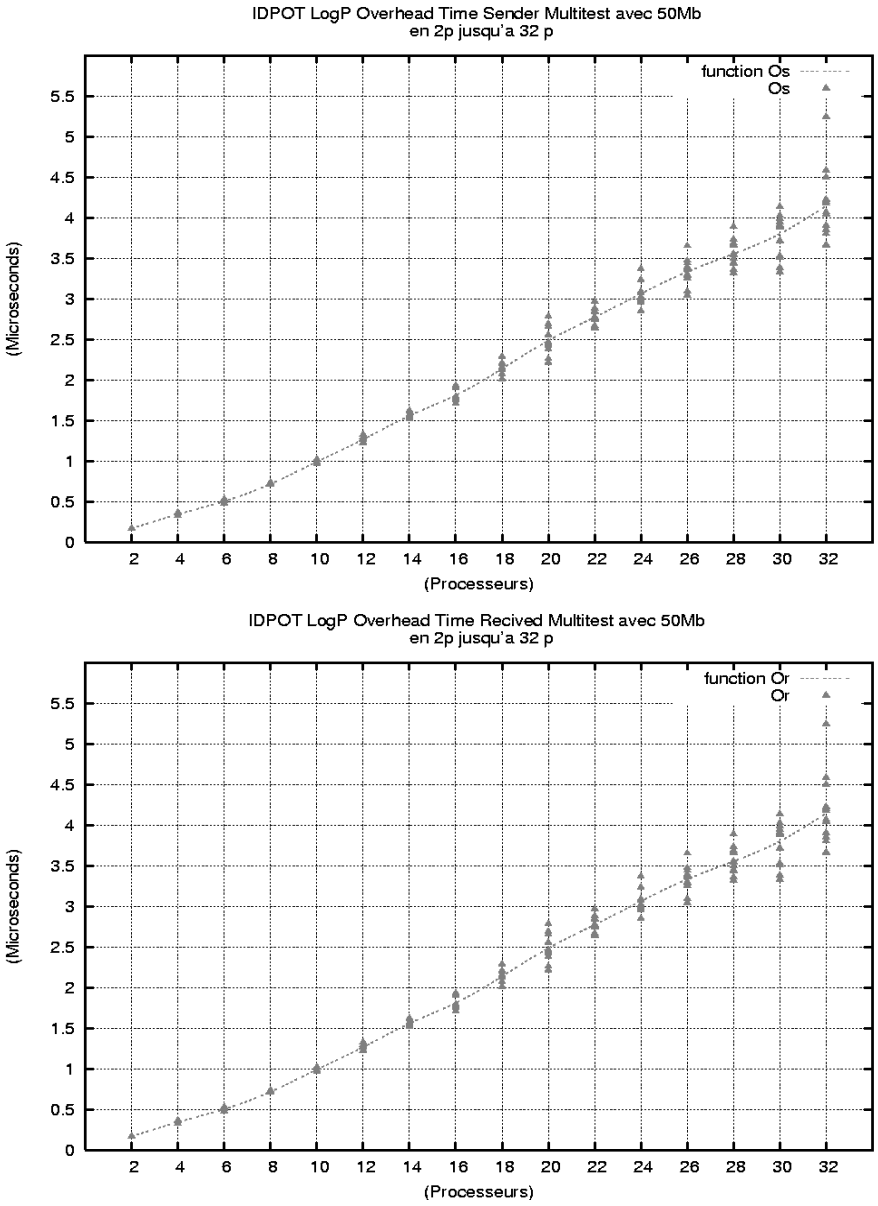


FIG. 2.4 – *Experience de mesure de Os et Or pour un message de taille 50 Mo*

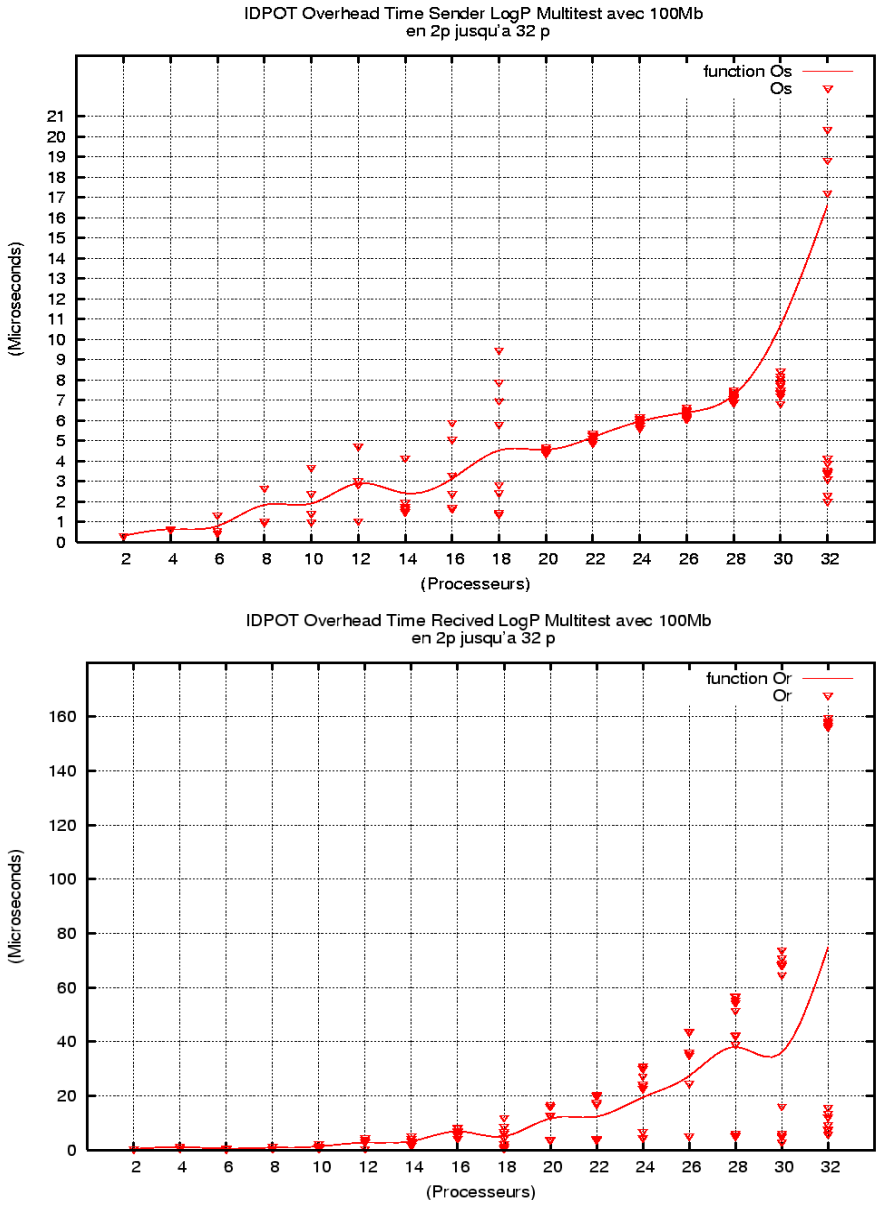


FIG. 2.5 – *Experience de mesure de Os er Or pour un message de taille 100 Mo*

toutes les mesures sont rélationées et des resultats des cettes deux graphiques sont équivalents aux mesures de *Os* et *Or* des respectives tailles qu'on a montré avant. Mais est important regarder et identifier la distribution des données pour chaque variable du modèle LogP, au realiséer l'analyse des données qu'on va à expliquer dans la dernière partie de rapport.

La latence pour chacune des tailles est représentée sur la figure 2.8. En cette graphique comparative est visible comm'au augmenter la taille de message la latence mesurée augmente. Aussi, on peut regarder que pour les pairs de processeurs que sont bas, le compartemment present une similarité qualitative, mais il faut d'observer les rangs des valeurs. Les pics que se presentent et les valeurs *zero* en la graphique sont par la proprieté qu'on à expliqué avant sur la transfert affectée pour la taille de message dans le modèle LogP et l'utilisation des réséau. Cettes valeurs ne sont pas néccesairemment zero, sinon des valeurs que sont très petites.

2.2.2 Taille variable

Les expériences pour des tailles variables sont réalisées avec différents nombres de processeurs, de 2 à 16, pour regarder le comportement sur les messages. Par exemple la figure 2.9 montre le gap pour 2 et 16 processeurs. On peut regarder dans le première partie de la graphique, comm'est la similarité des valeurs pour chque une des characteristiques. Aussi, est important observer comme les mesures augmentent dans le temps au augmenter la taille de messages[6].

La deuxième partie de las graphique, permettre depuis cett'experience regarder le compartemment pour pairs de processeurs. En cette graphique, par exemple, on a éxecuté le code sur seize (16) processeurs et par un message depuis 1Mo jusqu'ua 100Mo. Chaque pair est identifié et presente un compartemment particulier. Par exemple, on peut regarder la importance difference de temps entre les noeuds 10 et 11 sur les autres. On fait, est par l'utilisation du système pendant les essais et aussi, par les proprietés de communication de réséau qu'ont fair presentées avant et aussi, pour l'état de chaqu'un des processeurs et des noeuds d'IDPOT. Par autres essais de la même experience on va a trouver different compartemment pour noeud, mais la mesure général en des essais de taille fixe est equivalent, comm'on montre

[6]Les mesures sont en bytes pour les tailles de messages et en microseconds pour les temps de mesure.

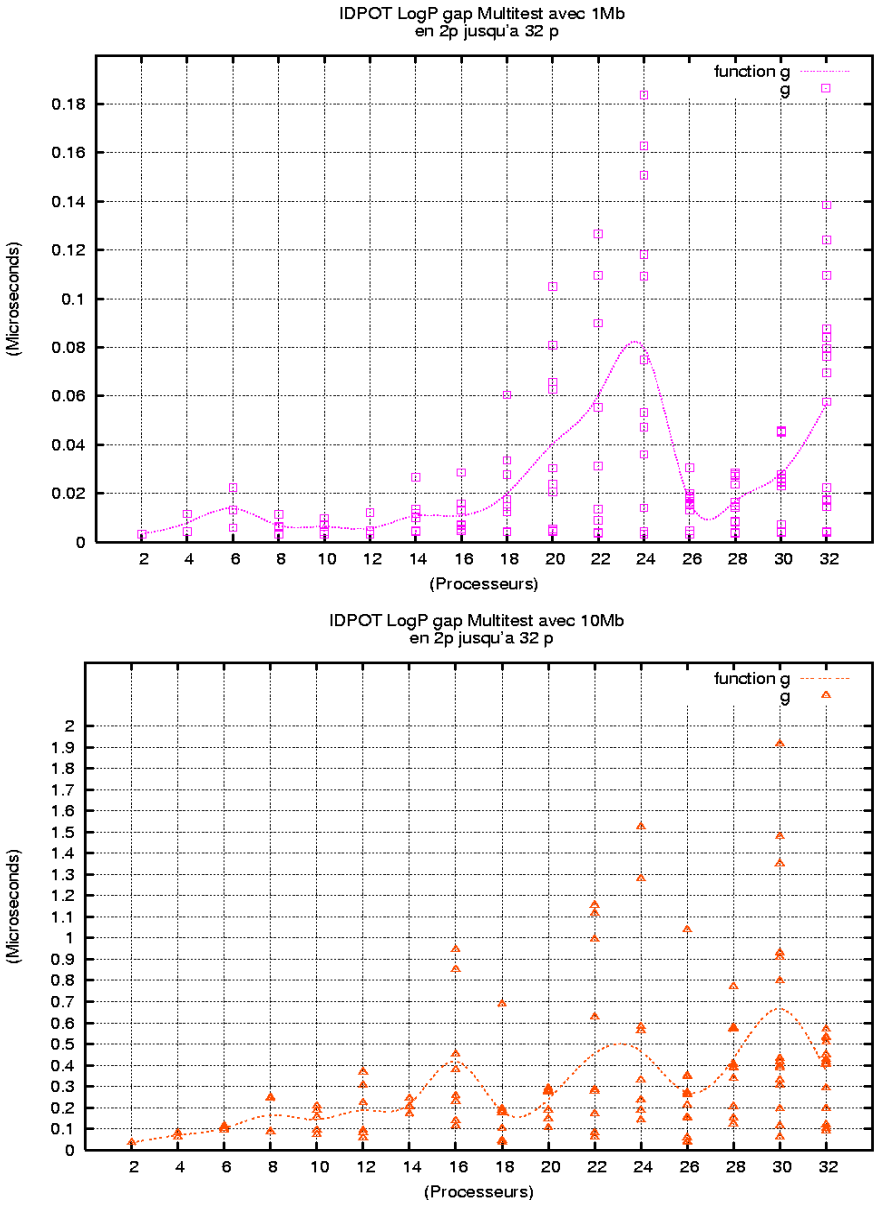


FIG. 2.6 – *Experience de mesure de gap pour messages de 1Mo et 10Mo en IDPOT.*

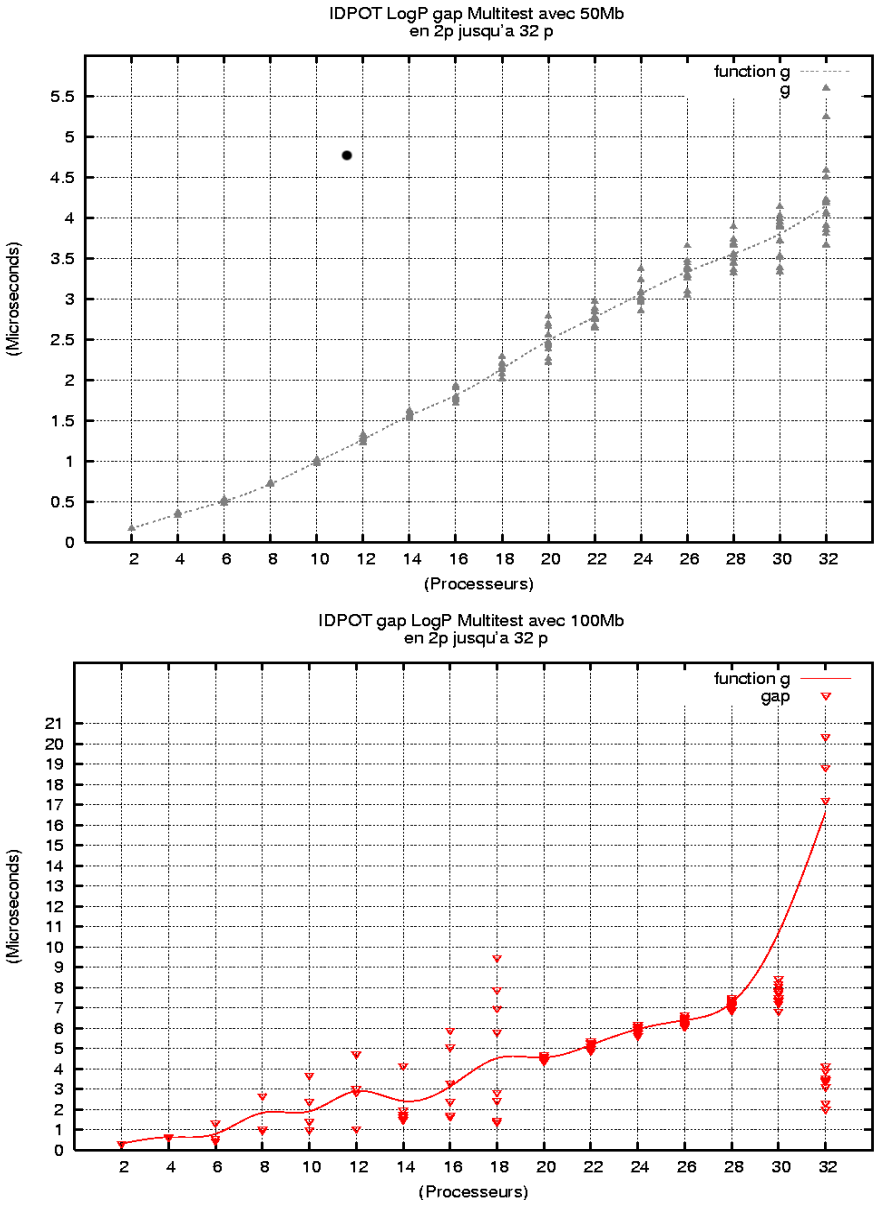


FIG. 2.7 – *Experience de mesure de gap pour messages de 50Mo et 100Mo en IDPOT.*

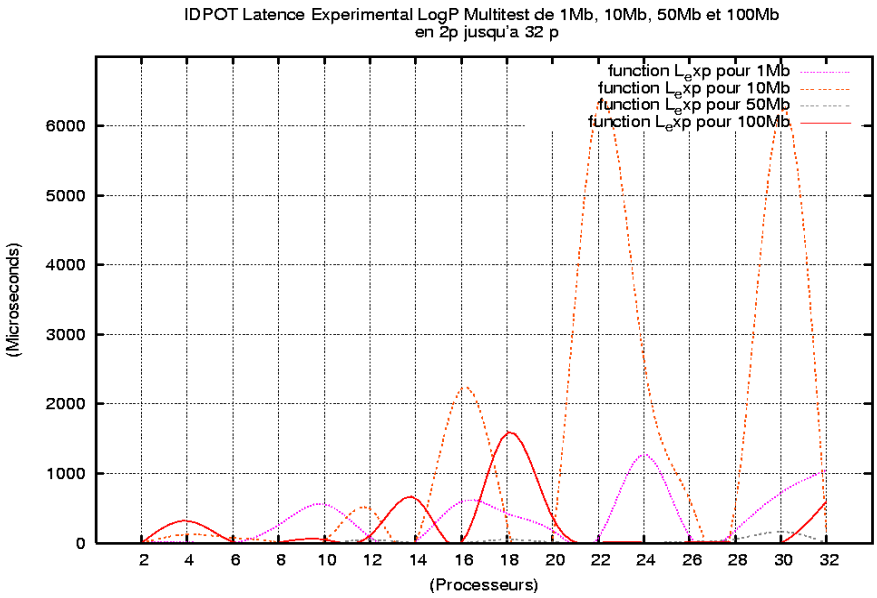


FIG. 2.8 – *Comparaison de mesures de latence expérimentée depuis 1 Mo jusqu'a 100 Mo en IDPOT.*

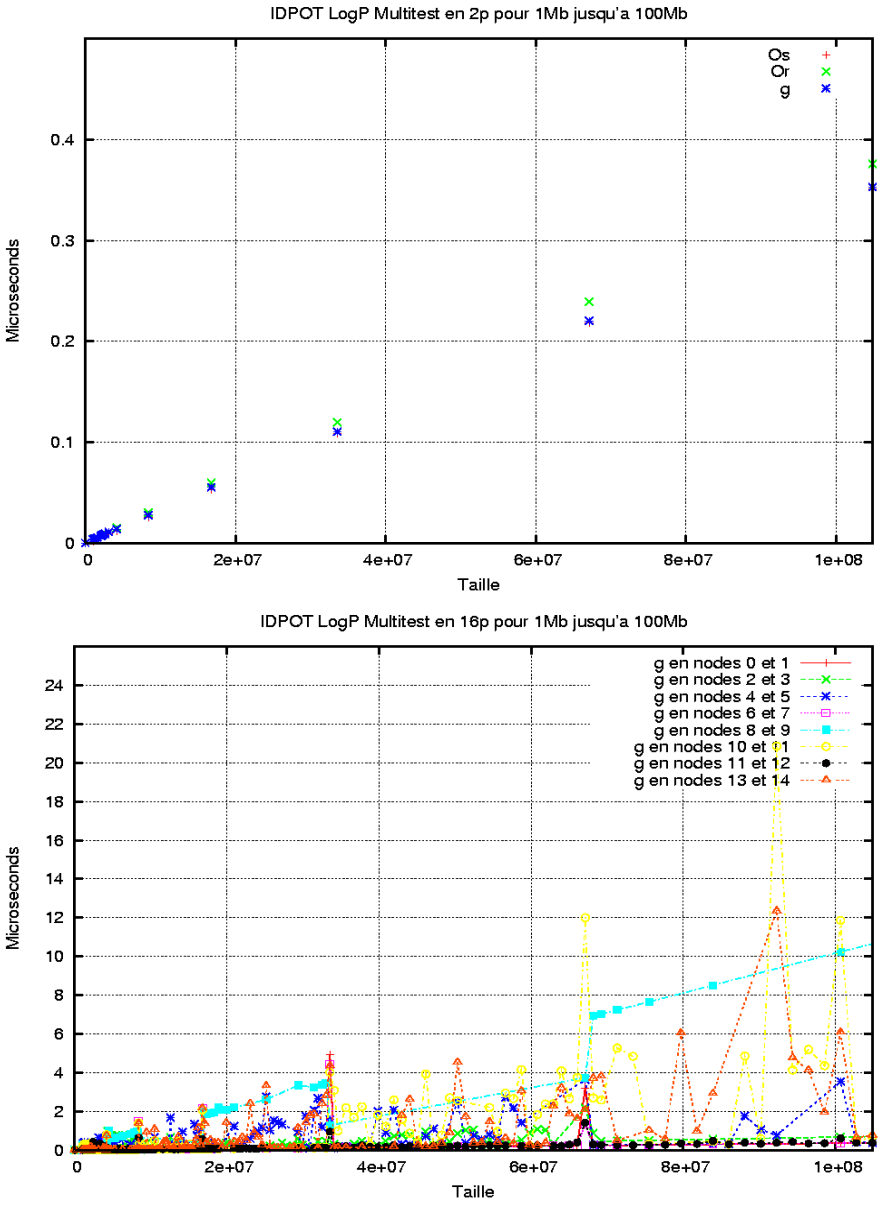


FIG. 2.9 – *Experience de mesure en IDPOT pour messages depuis 1Mo à 100Mo en 2 et 16p.*

en toutes les graphiques.

Infortunémment, les essais ne sont pas complets pour les problèmes en IDPOT en la date de la recherche, mais c'est une anlyses que peut donner information important sur l'état réel d'environemment et qu'on proppose pour la continuation de la recherche en ce domain sur cette infrastructure.

2.3 Les Experiences sur le ICluster2

Les mêmes expériences ont été menées sur le ICluster2 et les données sont présentées de la même manière.

2.3.1 Taille Fixe

Pour ICluster2 les mesures de taille fixe sont présentées sur les figures : 2.10, 2.11, 2.12, 2.13 pour les mesures d'overhead en envoi et réception. Pour les mesures de gap ce sont les figures : 2.14 et 2.15.

En la figure 2.10 on peut regarder les mesures de Os et Or pour une taille de 1Mo. Comm'est claire, il y a des valeurs plus bas qu'en IDPOT pour les proprietés de la réseau. Mais on peut identifier une difference important dans le cas de la mesure de Or depuis de 24 processeurs. La tendance se mantennent pour messages de 10Mo sur 22 processeurs comm'est visible dans la figure 2.11 et en 16 processeurs pour 50Mo et 100Mo selon les données des figures 2.12 et 2.13 respectivemment.

Dans les mesures de *gap* que sont presentées, es interessant regarder comme le rangs de valeurs sont chaque fois plus grand au augmenter la taille, d'une forme plus regulier qu'en IDPOT. En la figure 2.14 où on peut regarder les données par pairs pour messages de 1Mo et 10Mo respectivemment, et comme le rang des valeurs n'est pas consideré *grand* en relation avec la même experience realisée en IDPOT. On fait, est identifié un nombre important où se present une augmentation, comm'est le cas de 16 processeurs pour 1Mo et 18 pour 10Mo de taille de message. Autre point important est la relative haut messure initial de *gap* pour 2 processeurs en relation à l'utilisation des suivants cantités jusqu'a 12. Une première explication peut être que les conditions d'utilisation du système en général et la complêxité d'environemment de communicatino, en ce cas, pour mesure d'un message que n'est pas grand sur deux processeurs affectent le transfert du message. Mais on va à analyser plus detaillémment cette mesure.

Par des tailles de 50Mo et 100Mo presentés dans la figure 2.15 on peut dire qu'est interessant regarder comme la distribution des données est plus grand pour les messages de 100Mo. Aussi, n'est pas visible en ICluster2 une condition de linéalité qu'on peut regarder dans les experiences pour 50Mo en IDPOT, mais il faut de pris en compte que le rang des valeurs trouvées est equivalent dans les deux grappes, malgré la dispersion des données qu'est visible en ICluster2.

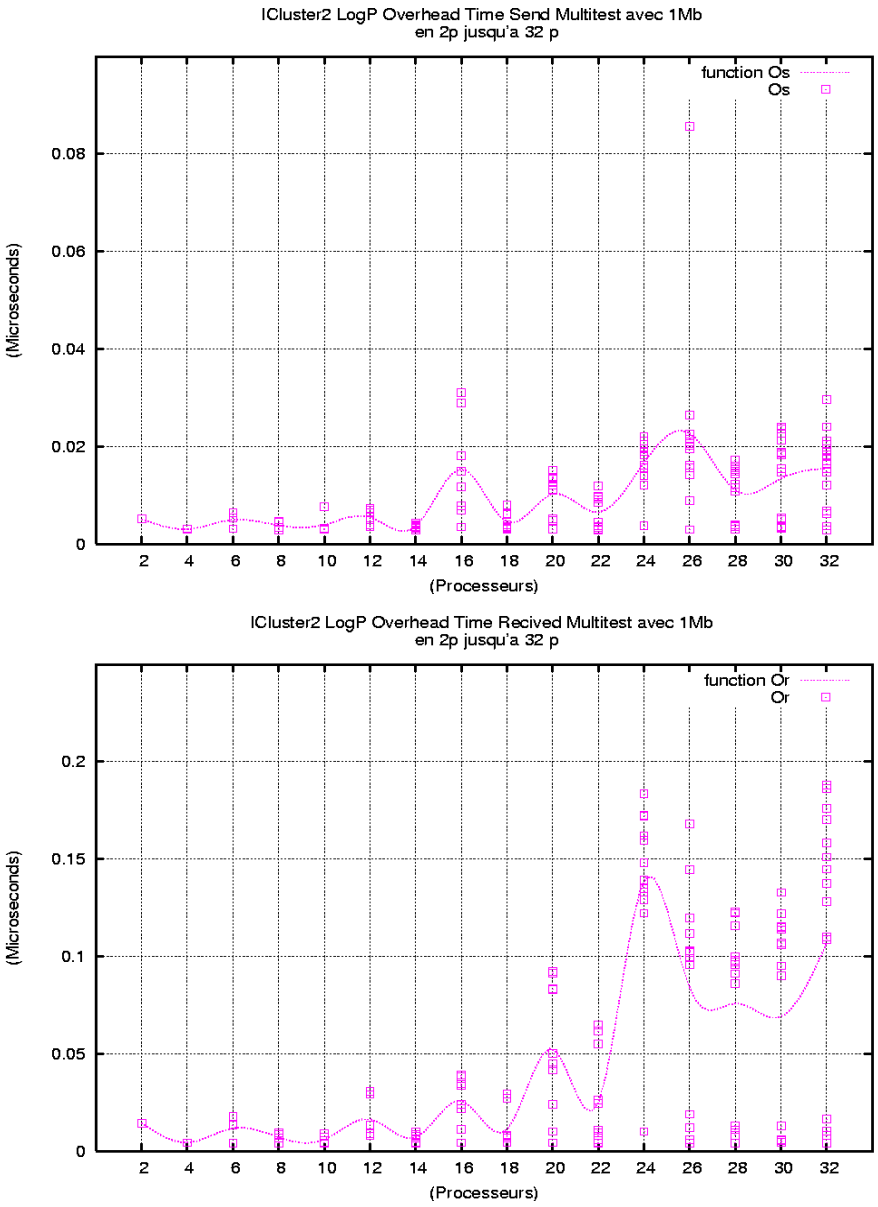


FIG. 2.10 – *Experience de mesure de Os et Or pour 1Mo en ICluster2*

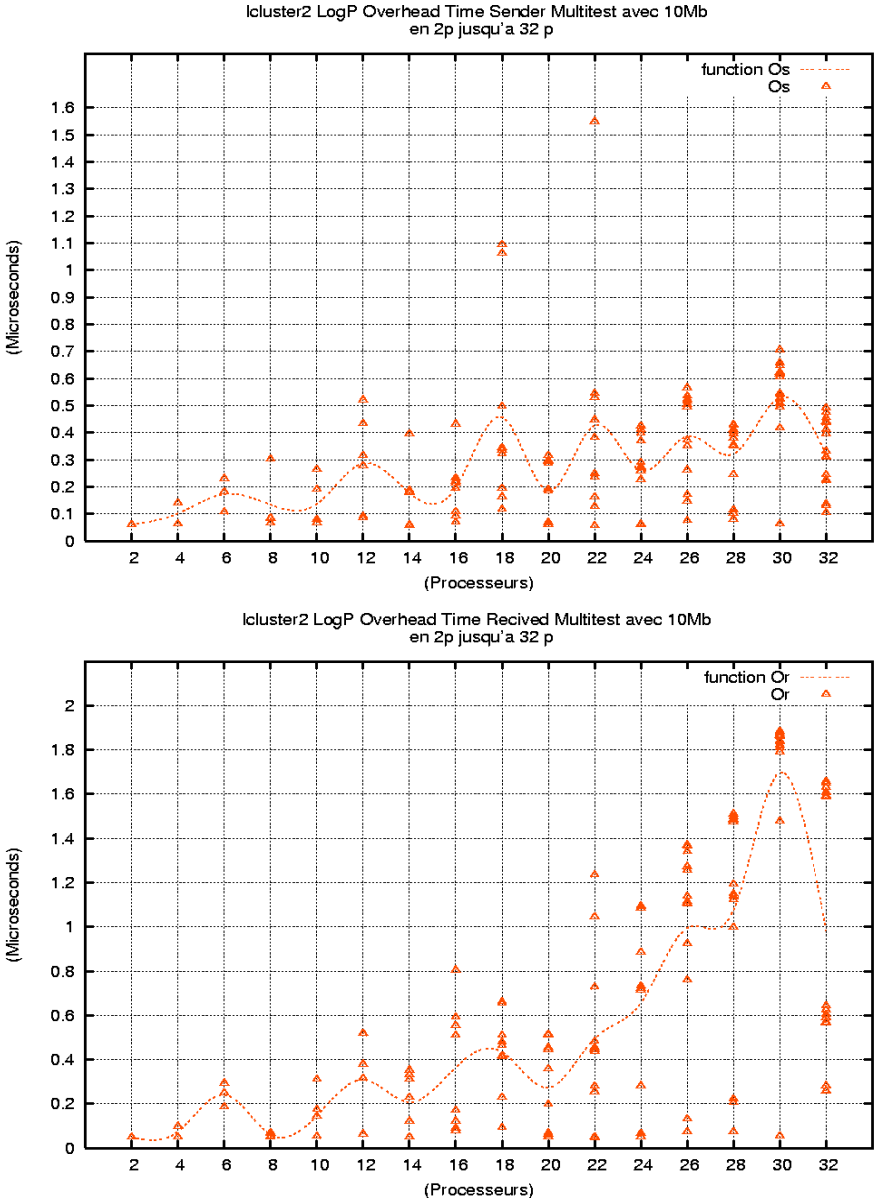


FIG. 2.11 – *Experience de mesure de Os et Or pour 10 Mo en ICluster2*

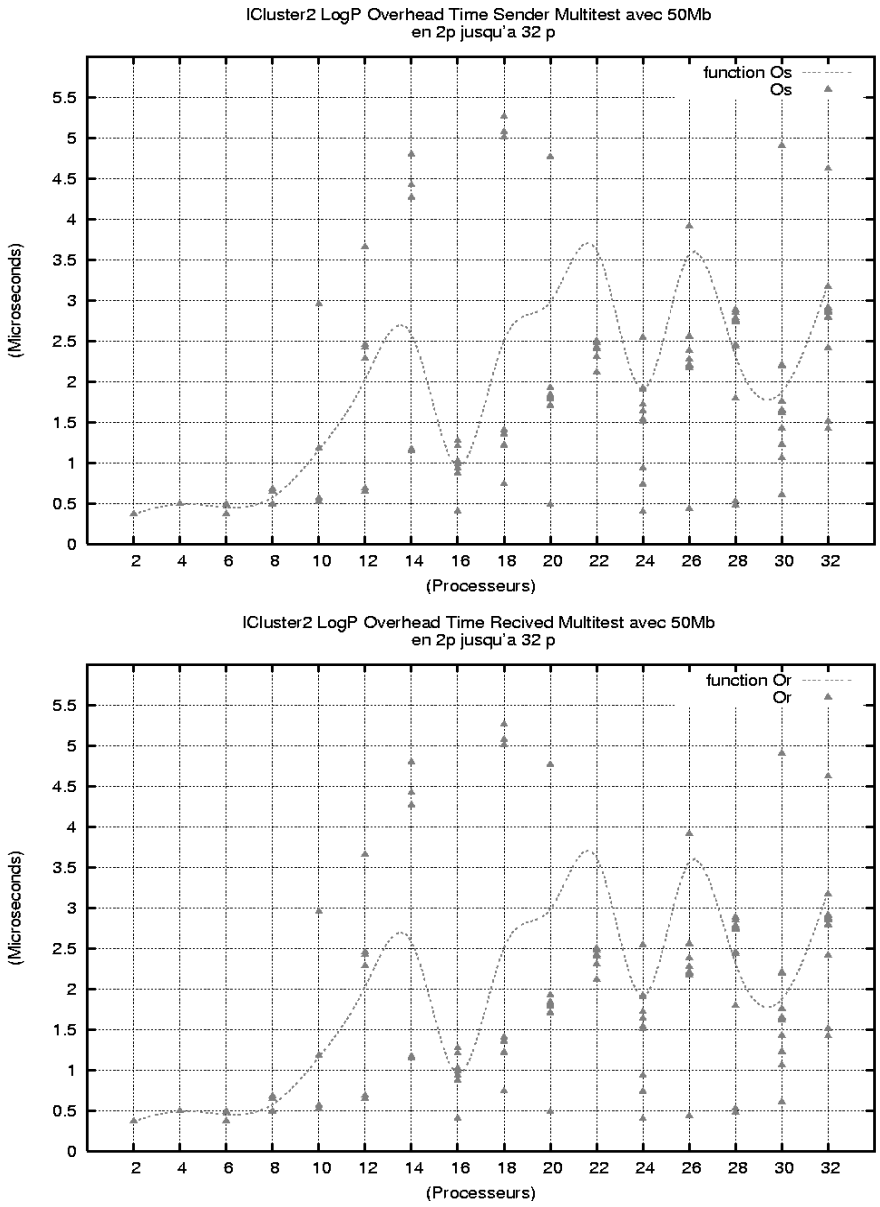


FIG. 2.12 – *Experience de mesure de Os et Or pour 50 Mo en Icluster2*

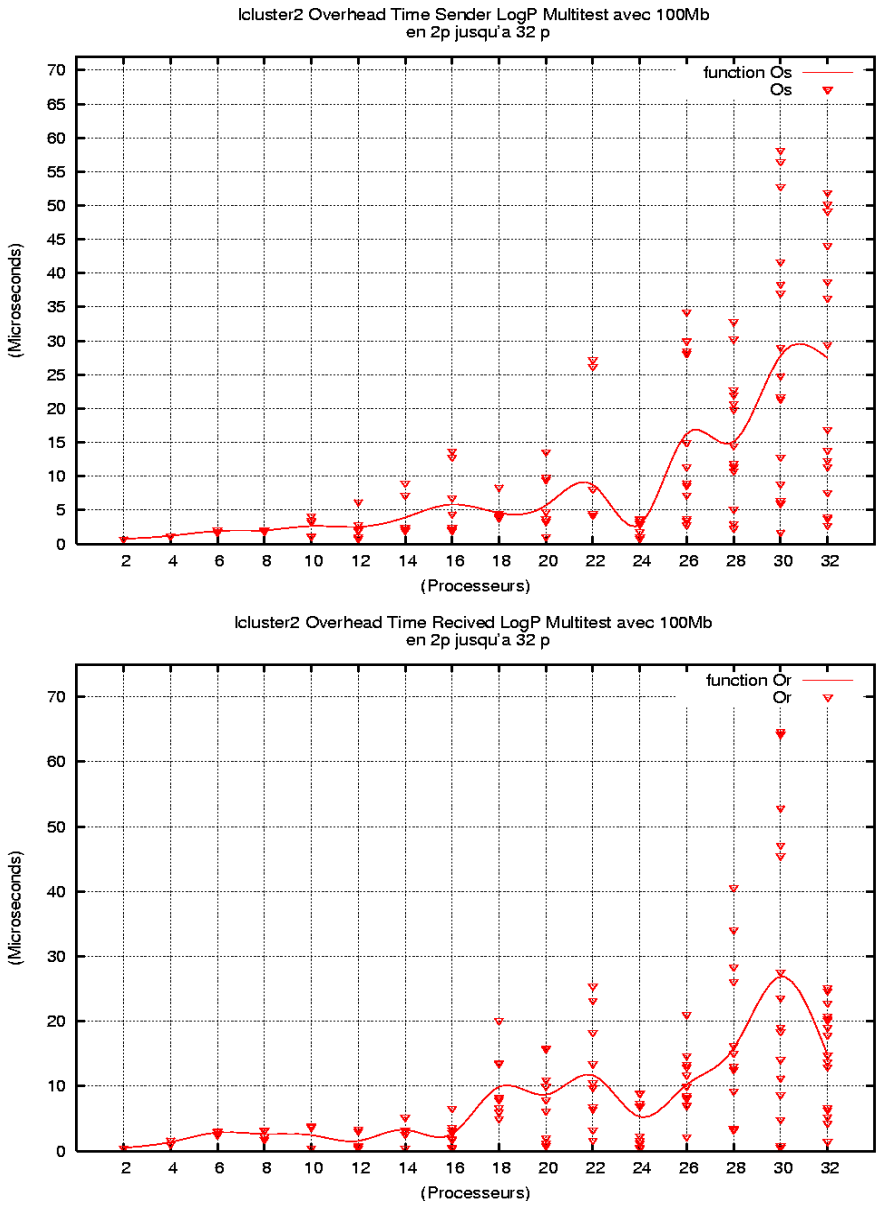


FIG. 2.13 – *Experience de mesure de Os et Or pour 100 Mo en Icluster2*

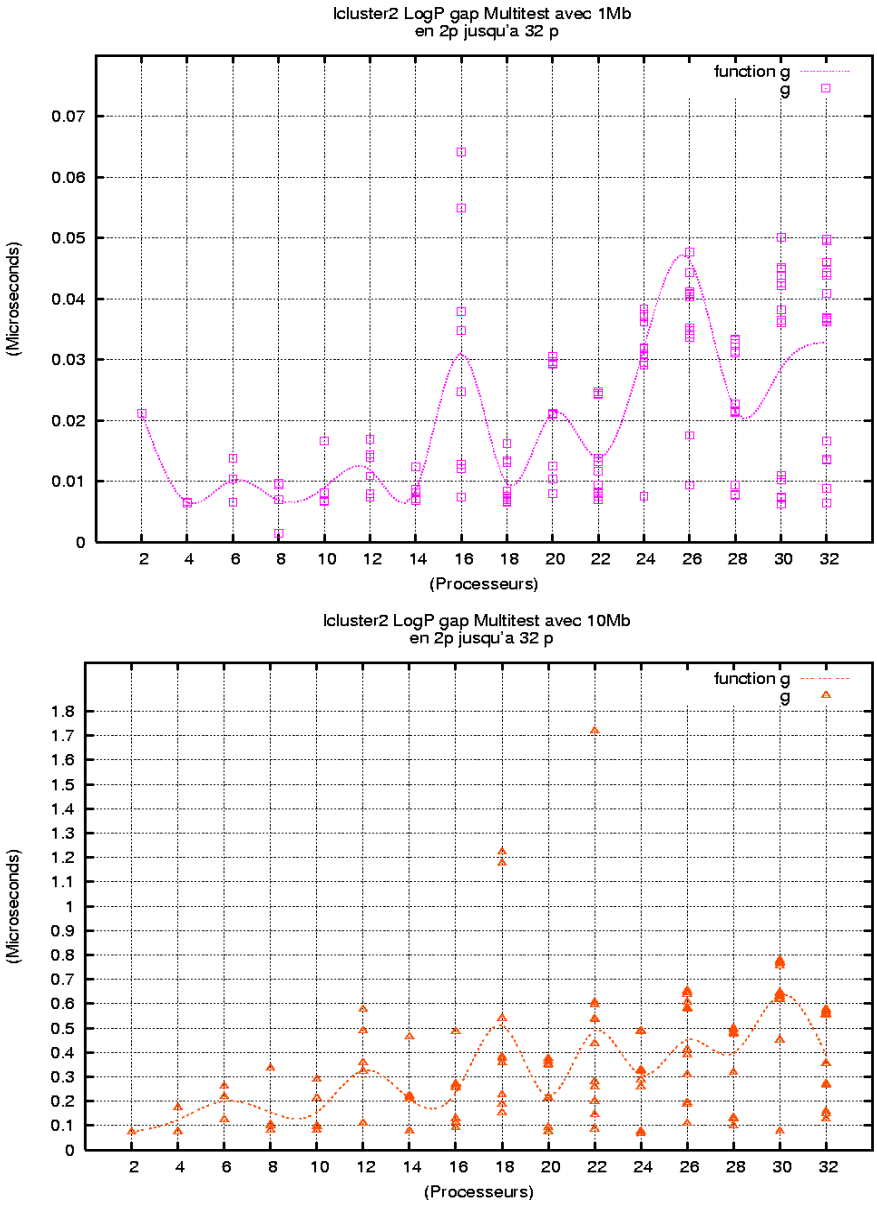


FIG. 2.14 – *Experience de mesure de gap en Icluster2 pour messages de taille 1Mo et 50Mo.*

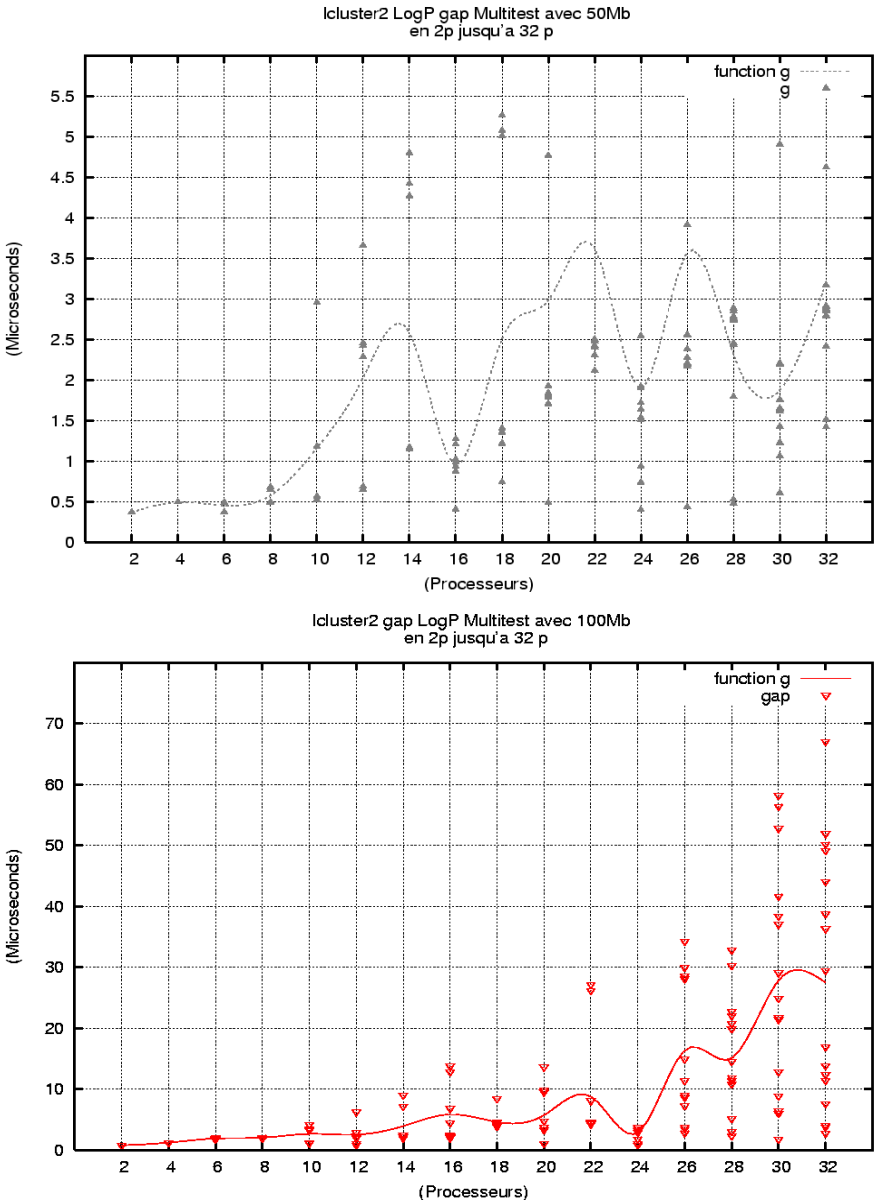


FIG. 2.15 – *Experience de mesure de gap en Icluster2 pour messages de taille 50Mo et 100Mo.*

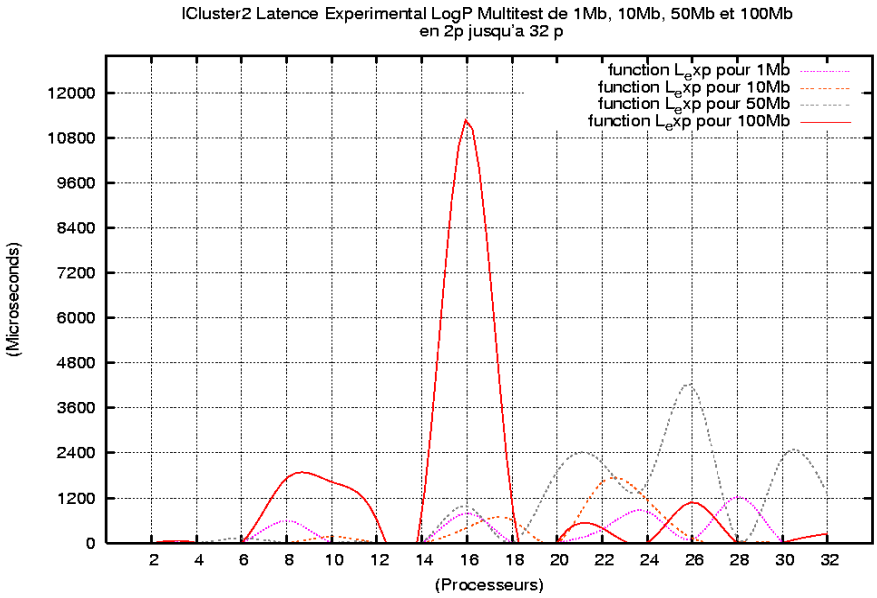


FIG. 2.16 – *Expérience de mesure de Latence sur ICluster2*

La *latence* pour chacune des tailles est présentée sur la figure 2.16. On peut regarder comme la latence est relativemment haut respect la latence trouvée en IDPOT. Mais aussi, il y a un compartemment plus stable. En attendant qu'en IDPOT on trouvé plus des pics en ICluster2 non, garantisée pour le type de système de communication qu'on a referencié plusieurs fois. Mais il faut de regarder speciellement l'haut valeur presenté au utiliser 16 processeurs en 100Mo.

2.3.2 Taille Variante

Les expériences sur le Icluster2 avec taille variable sont présentées sur le graphique 2.17. En contradiction avec la performance regardée dans IDPOT les valeurs sont plus grand en ICluster2, pour les mêmes conditions qu'on a expliquée déjà avant. Aussi, on peut regarder un compartemment simileir au IDPOT dans la mesure pour pairs de noeuds, en ce cas, on peut trouver pour 8 processeurs quelque stabilité, avec variations importants pour les processeurs 6 et 7. Les autres mesures ne sont pas presentés pour raisons d'espace, mais pourra être interessant faire un analyse plus detaillé de ce aspect en ICluster2.

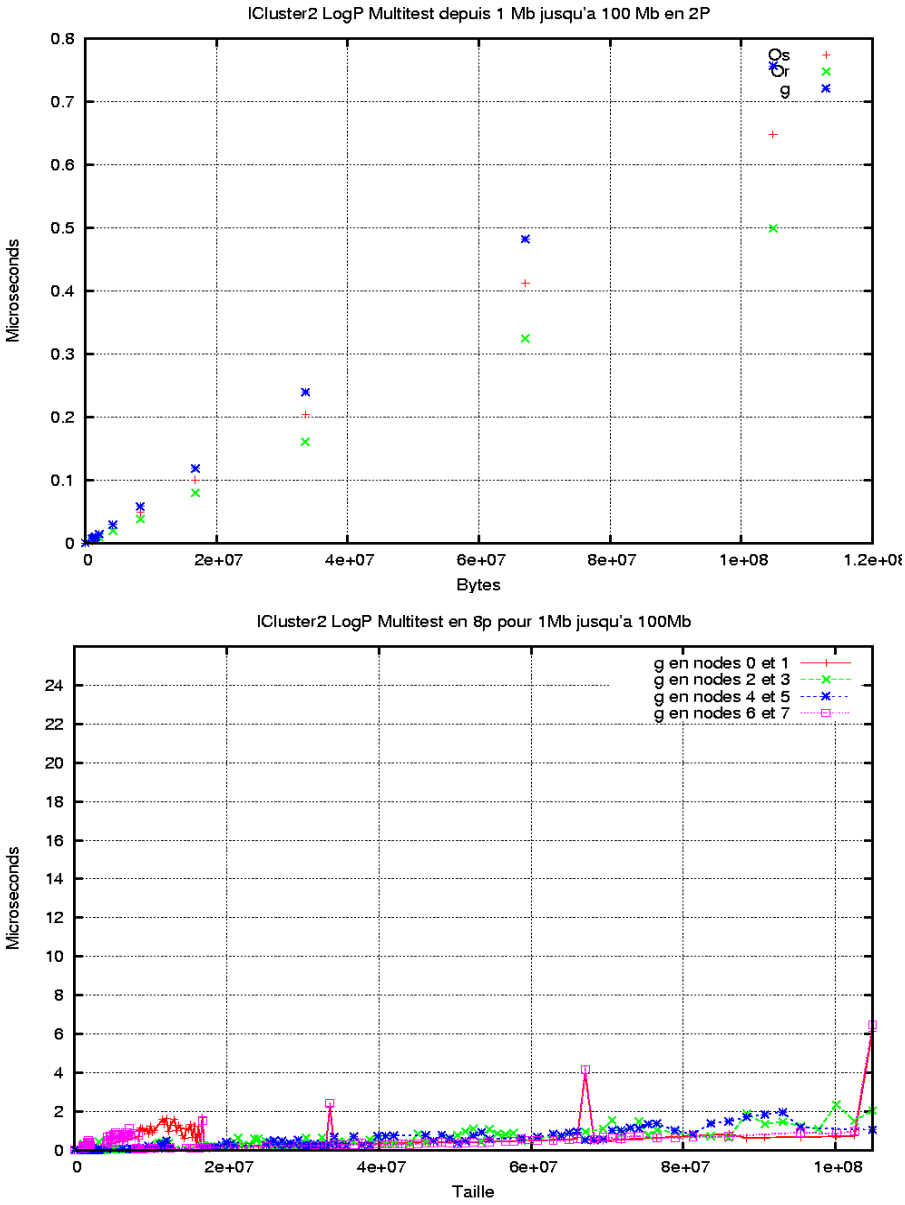


FIG. 2.17 – *Experience de mesure avec messages de taille variante en 2p et 8p de ICluster2*

2.4 Autres Expériences Non présentées

D'autres expériences ont été réalisées mais ne sont pas présentées dans cette partie du rapport utilisées pour l'analyse de résultats et la construction de modèle. Par exemple des expériences d'inondation, *flooding*, pour de petites tailles pour regarder le temps de communication normal de données, ainsi que quelques essais sur la grappe *gdx* d'Orsay qui ne sont pas présentés à cause de l'indisponibilité des ressources au moment de faire le rapport et de l'impossibilité de refaire les expériences.

Une autre expérience faite mais non présentée concerne la mesure entre différentes grappes, à cause de l'instabilité de grid5000 en ce moment.

2.5 Conclusion sur la partie expérimentation

Depuis realisés les experiences, on peut conclure les suivants aspects :

- Bien que l'infrastructure en étude, en ce cas les grappes IDPOT et ICluster2, sont completemment differents en architecture de materiel et de logiciel, le compartemment pendant les epreuves a été interesant pour regarder deux environemments specifiques, par exemple, dans le cas de IDPOT, on peut regarder un réseaux plus hétérogenée qu'ICluster2 mais moins complèxe. Les deux grappes, comme partie de Grid5000 peuvent fournir information important pour faire des modèles de compartemment de grid5000 en général.
- A partir des premières graphiques avec les données trouves pendant chaqu'une des experiences qu'on a realisé, sont identifiées de première vu, les points d'inestabilité, selon les tailles de messages et le nombre de processeurs utilisés. Ces points peuvent être identifies comme points de *bottleneck*, comm'est claire.
- Malgré l'experience de mesure en toute l'environemment n'a pas fait possible de realiser pour les problèmes avec plusieurs des noeuds que sont partie de Grid5000, avec les données qu'on a trouvé, est possible faire une première approach au étude de la performance de transfert haut débit en les grappes de Grid5000, depuis des resultats et des données que sont joins ici.
- Les essais avec taille variante ont fournissent des données pour IDPOT et ICluster2, que permettent identifier l'echantillon de taille de données où se present l'inestabilité, pour paires de processeurs. Aussi, comm'on

peut regarder dans les graphiques pour deux processeurs, la performance initial es prochain entre les deux infrastructures en étude, alors, comm'on va a montrer dans la partie suivant, est possible identifier paramètres de comparaison importants.

– Le code *LogP_multitest* utilisé pour les essais permettre la conaissance de la latence pendant le moment de l'epreuve, a partir du modèle LogP. Aussi, fournit des autres données de mesure que peuvent être utilisés pour la construction de modèles plus specifiqués d'après les données experimentelles.Ce aspect est interesant parce que il'est possible de faire une simulation de la performance avec les données et les resultats du modèle specifique, car on peut valider le modèle de mesure utilisé.

Chapitre 3

Analyses des Données et Modélisation

Dans cette dernière partie nous allons présenter les différentes analyses de performance à partir des données trouvées dans la partie expérimentale ainsi que proposer un modèle qui décrit le comportement observé, en accord avec le modèle paramétrisé LogP utilisé pour faire l'évaluation du système et prendre les mesures. Finalement, nous présenterons les conclusions du travail ainsi que les perspectives de développement.

3.1 Description des Analyses

Il est acquis que l'analyse de la performance d'environnements distribués implique l'observation du comportement statistique des données[22], dans notre cas, des temps de transfert de messages d'une taille déterminée selon le nombre de processeurs.

En général, les données acquises sont utilisées pour la construction de fonctions. Dans la partie précédente, nous avons regardé les fonctions expérimentales pour chacune des mesures en accord avec une fonction cubique pour faire une courbe expérimentale qui peut montrer le comportement approximatif d'environnement pendant l'expérience. Maintenant nous allons étudier les données statistiquement ainsi que de nouveaux résultats de calcul issus d'une modélisation.

3.2 Analyses de Transfert Haut Débit sur IDPOT

Le graphique 3.1 montre les résultats de calcul de la *moyenne* et de la *médiane* pour l'overhead d'envoi. Le comportement est qualitativement prédit par plusieurs modèles décrits auparavant, en augmentant la taille de message et le nombre de processeurs, Os augmente. On peut observer le même comportement pour Or et le gap g qui sont présentés dans les graphiques 3.2 et 3.3. Il est à noter la différence entre les valeurs de temps pour chaque mesure et l'instabilité des valeurs quand la taille de message augmente. Nous avons mentionné que peut être pour les grandes tailles de messages, le transfert peut commencer quand l'expéditeur est encore occupé, cet aspect explique en partie la non linéarité des donnés ainsi que la différence de temps, car en général $Os < Or$. En augmentant la taille du message, la différence de temps augmente aussi.

Un deuxième aspect concerne les *pics* constatés sur tous les graphiques de mesures. Le comportement d'une grappe n'est pas linéaire, il est important de regarder les caractéristiques de réseaux et l'infrastructure de communication d'IDPOT qui sont detaillées dans la première partie de ce rapport[1]. Une première approche suggère, par exemple, l'utilisation de l'infrastructure par une autre application pendant les essais. Bien que pour cette partie nous n'ayons pas présenté les graphiques de la latence au moment de l'experimentation, l'utilisation des noeuds pour d'autres travaux affectent les mesures. Autre point important, la caractéristique du matériel de communication par noeud et dans la grappe en général. IDPOT est constitué principalement pour un réseau GigaEthernet, bien que l'utilisation des noeuds en particulier pour des messages de grand taille, fasse varier de manière importante la performance observée entre les expériences à des moments différents.

Un autre point intéressant est la différence de valeurs observée entre la *moyenne* et la *médiane*. La moyenne est calculée en additionnant les valeurs de toutes les observations puis en divisant cette somme par le nombre d'observations qui font partie de l'ensemble. Ce calcul permet d'obtenir la valeur moyenne de toutes les données. Dans notre cas, nous avons additionné les différentes mesures capturées dans les observations pour chaque paire de processeurs et chaque taille de message. La valeur médiane correspond à l'observation qui se trouve au centre d'une liste ordonnée de données. Elle cor-

[1] Au moment des essais, entre le 01/03/2005 et le 01/06/2005.

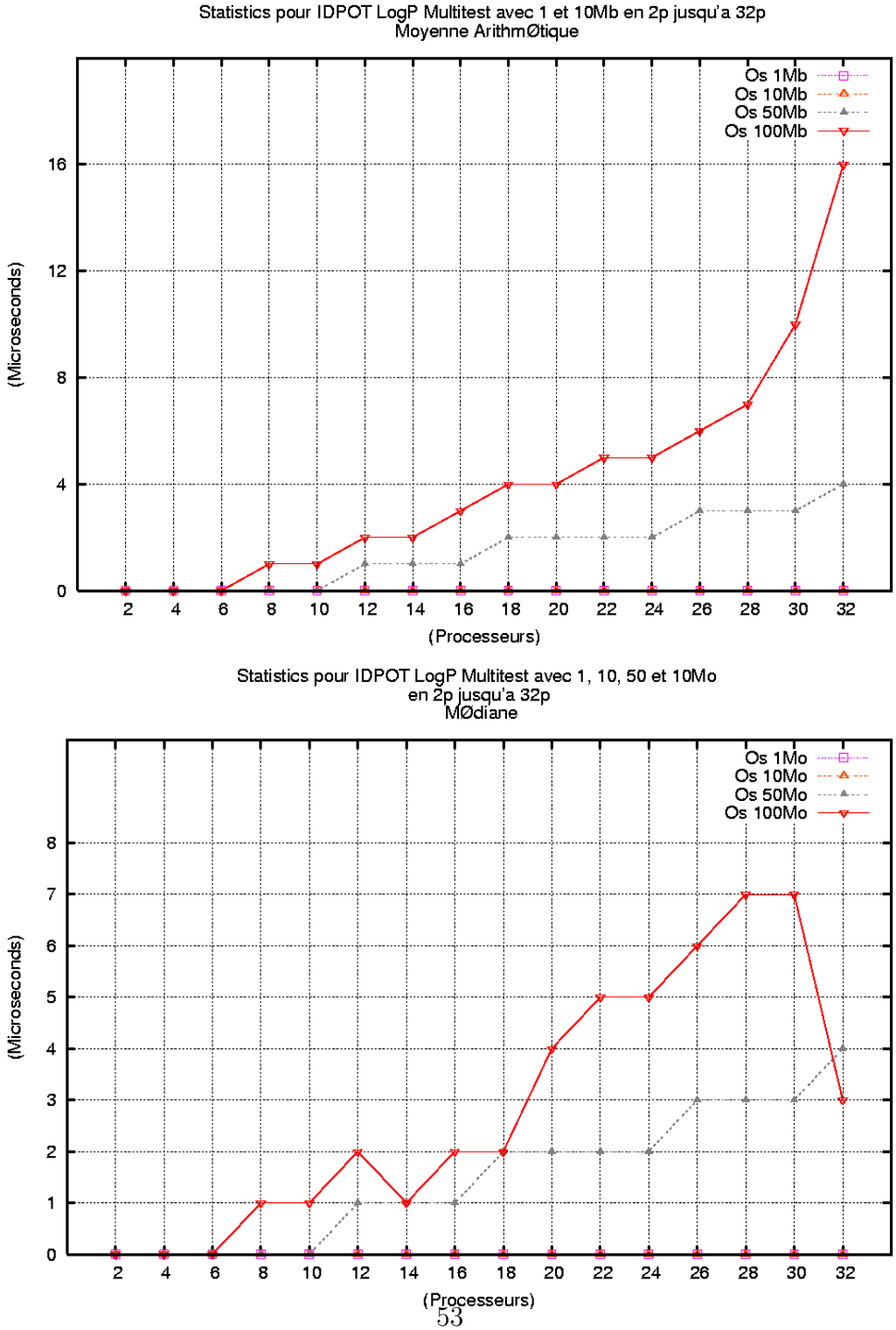
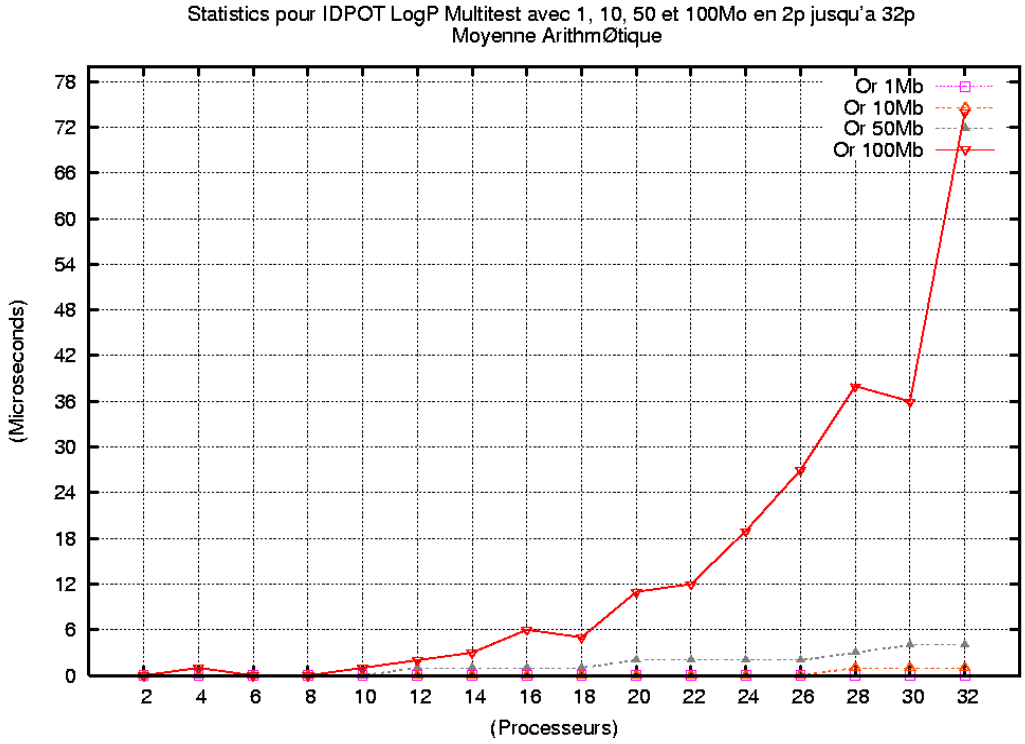


FIG. 3.1 – *Statistiques de l'expérience de mesure de Os sur IDPOT*



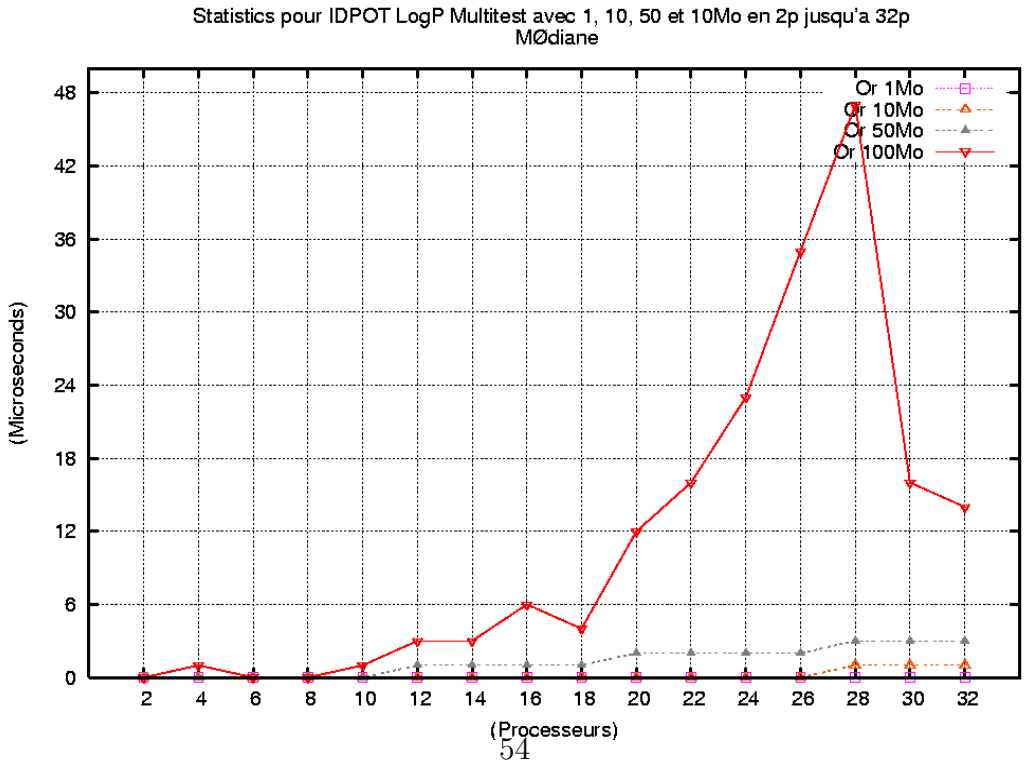


FIG. 3.2 – *Statistiques de l'expérience de mesure de Or en IDPOT*

P	1Mo	10Mo	50Mo	100Mo
2	1.82	1.90	0.25	1.63
4	13.68	122.80	0.26	318.49
6	1.30	1.70	0.13	15.71
8	259.31	7.12	1.55	6.62
10	560.44	0.85	0.23	48.59
12	83.01	107.01	45.41	101.96
14	25.57	19.04	0.83	642.61
16	598.43	350.83	0.95	6.83
18	425.37	0.12	45.96	1582.44
20	179.54	21.03	10.95	350.98
22	69.87	38.29	1.07	9.53
24	1279.5	0.33	2.35	12.47
26	92.24	0.49	16.33	17.1
28	179.99	57.67	53.28	37.95
30	723.46	73.20	159.69	32.25
32	1050.93	1.03	3.81	526.7

TAB. 3.1 – *Table des Latences Pour IDPOT.*

respond plus précisément à un pourcentage cumulé de 50%, c'est-à-dire que 50% des valeurs sont supérieures à la médiane et 50% lui sont inférieures[20].

Avec ces aspects, il est possible d'identifier quelques caractéristiques que nous allons décrire tant pour IDPOT que pour le ICluster2. L'analyse que nous proposons implique des observations conjointes des deux grappes et une comparaison.

A partir de ces données nous allons calculer une nouvelle valeur de *Latence* depuis le modèle LogP paramétrisé présenté dans la table 2 de la première partie. Nous allons présenter les données dans cette section dans la table 3.1, le modèle utilisé sera détaillé dans la section de modélisation.

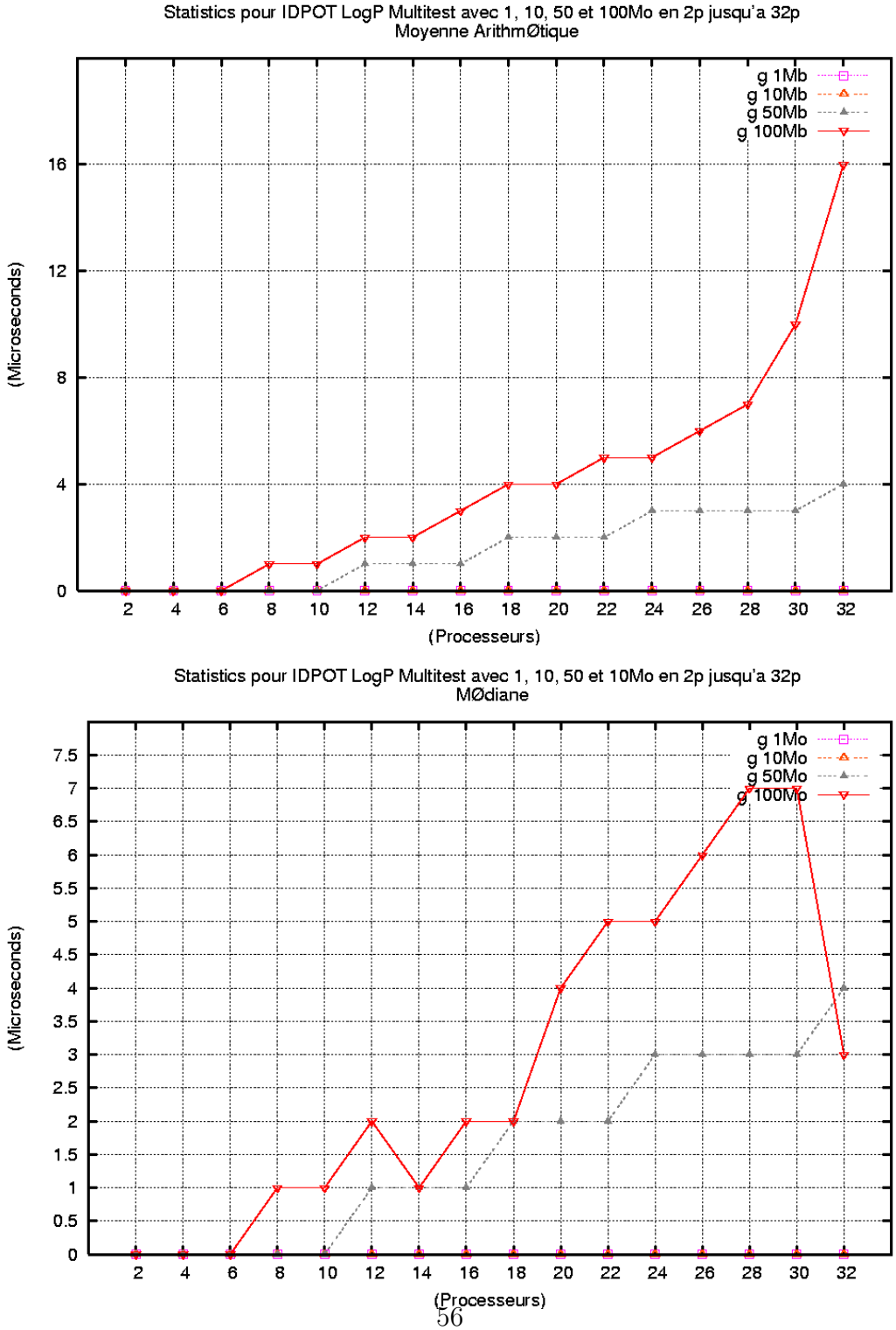


FIG. 3.3 – *Statistiques de l'expérience de mesure de g sur IDPOT*

3.3 Analyses de Transfert Haut Débit en ICluster2

Les mêmes considérations que pour IDPOT sont prises en compte dans l'expérimentation sur le ICluster2. Dans les graphiques 3.4 et 3.5 le comportement est visible pour différents tailles de messages de la grappe. La performance observée est semblable au comportement sur IDPOT. Dans les deux cas, tant le temps *Os* que *Or* augmente selon la taille du message ainsi qu'avec le nombre de processeurs.

Il est intéressant de remarquer que, comme pour les expériences sur IDPOT, la croissance du temps se présente sous une forme exponentielle à première vue. Si nous considérons les derniers aspects, ce phénomène peut être expliqué mais il faut aussi prendre en compte d'autres propriétés de transfert de données en environnements distribués, qui ont été décrit dans la premiére partie.

Sur le graphique 3.6 nous montrons les valeurs de g trouvées dans les différentes expériences sur le ICluster2. On peut constater une certaine stabilité dans les valeurs jusqu'à 20 processeurs et après les valeurs augmentent dramatiquement pour une taille de message de 100 Mo. Le comportement s'observe sur IDPOT aussi, mais la différence est moins accentuée.

En faisant une comparaison entre les résultats des deux grappes, a première vue, on va atteindre des valeurs plus élevées sur IDPOT que sur le ICluster2. Dans les premières valeurs pour des tailles de messages de 1Mo, 10Mo et 50Mo le comportement est identique et les intervalles des valeurs sont bas. Mais pour des messages de taille 100Mo la performance est très variable, on peut identifier des points de *bottleneck*. Dans le cas du Icluster2, l'instabilité commence à partir du transfert sur 22 processeurs. Pour IDPOT, bien que l'instabilité ne soit pas dramatique comme sur le ICluster2, pour la même taille de 100 Mo, on peut identifier un point important sur 12 processeurs. Cette constatation peut être étendue aux messages de 50Mo sur les deux grappes, mais l'espace des valeurs sur IDPOT est plus important que sur le Icluster2.

En observant les données et les courbes issues de l'expérience, sans traitement particulier, il est possible de constater le même comportement et d'identifier, a priori, l'échantillon où se présente l'instabilité, pour chacune des mesures. Ce travail est intéressant pour regarder la performance de transfert pour des tailles de messages importantes, mais regarder les résultats trouvées

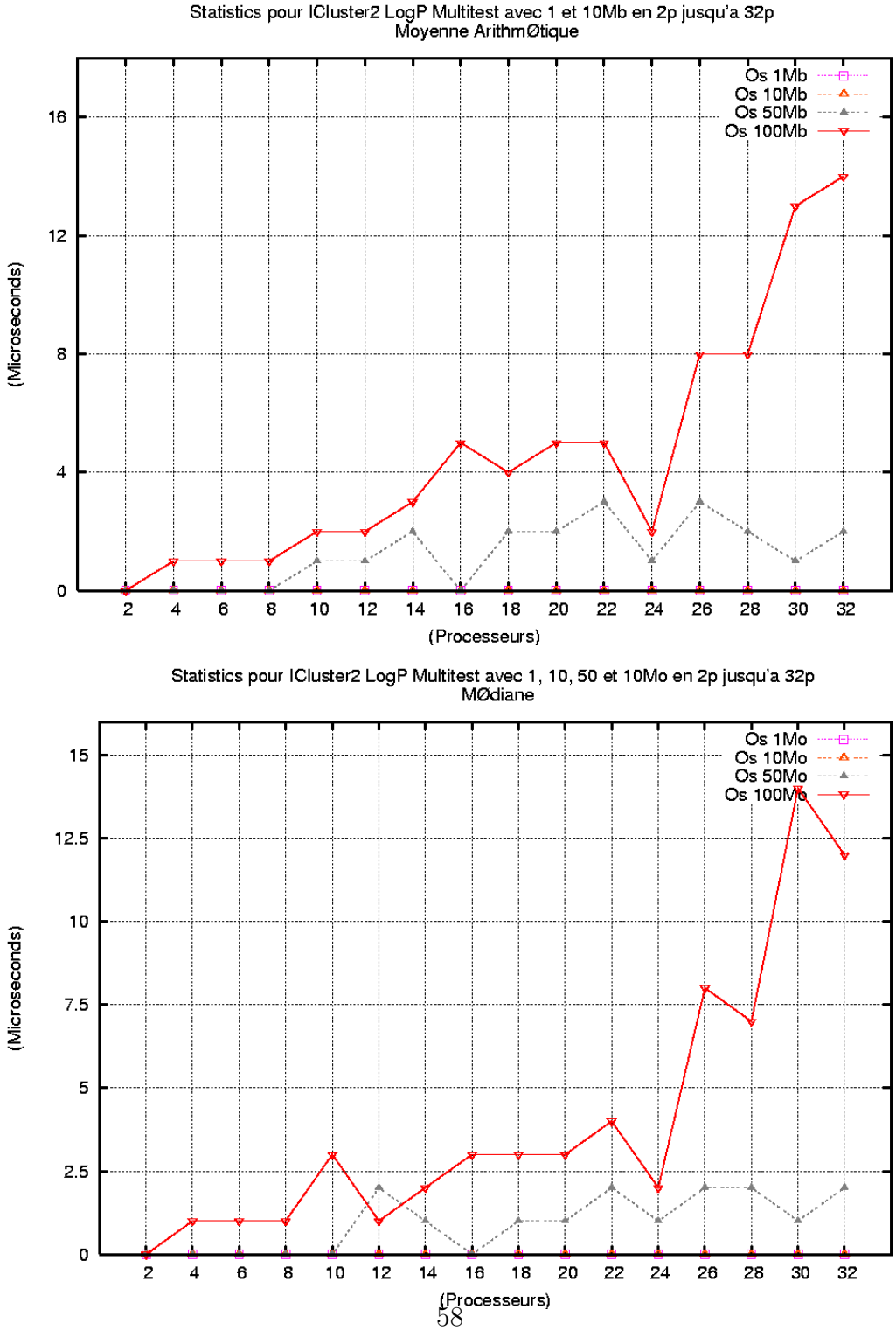


FIG. 3.4 – *Statistiques de l'Experience de mesure de Os en ICluster2*

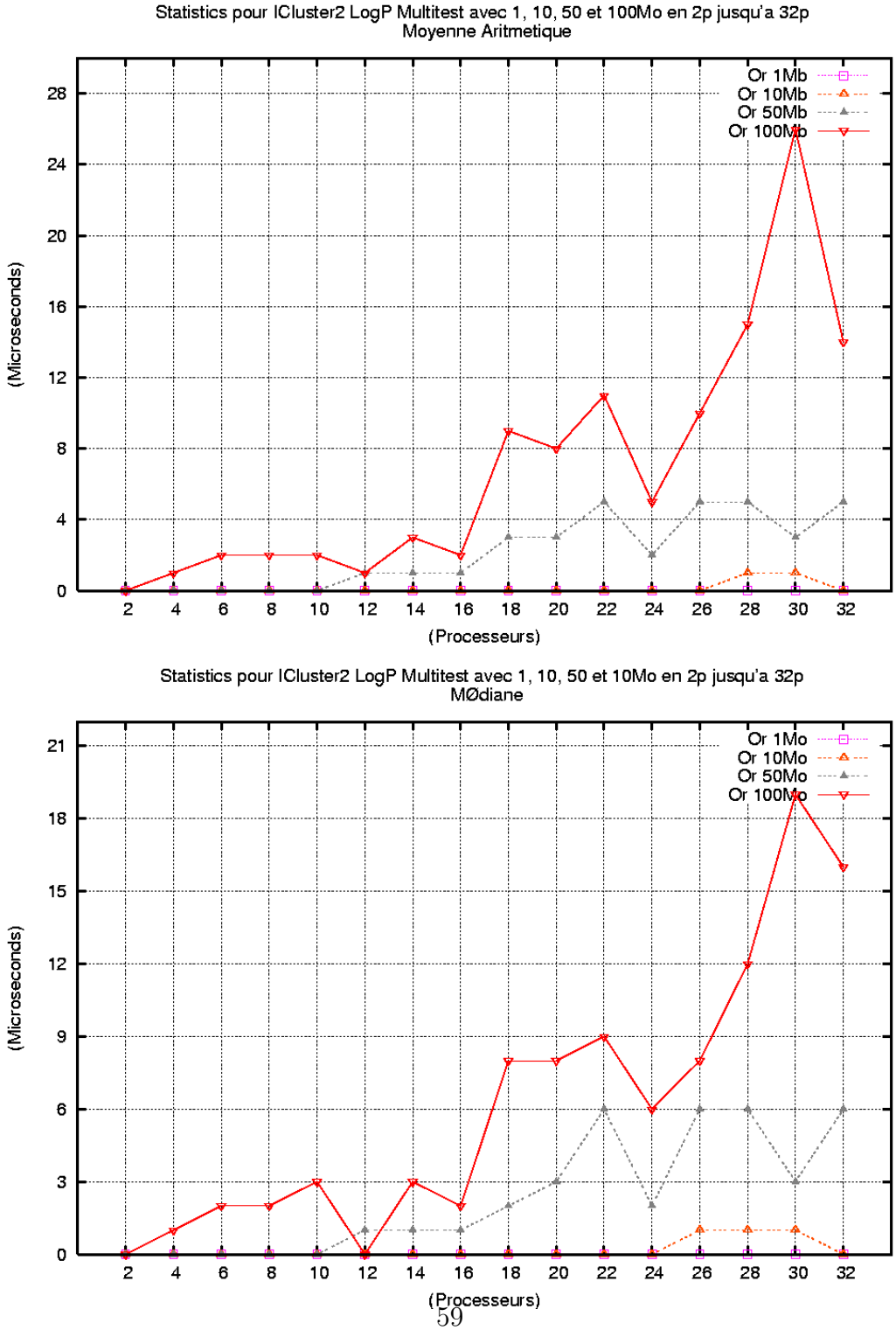


FIG. 3.5 – *Statistiques de l'expérience de mesure de Or sur le ICluster2*

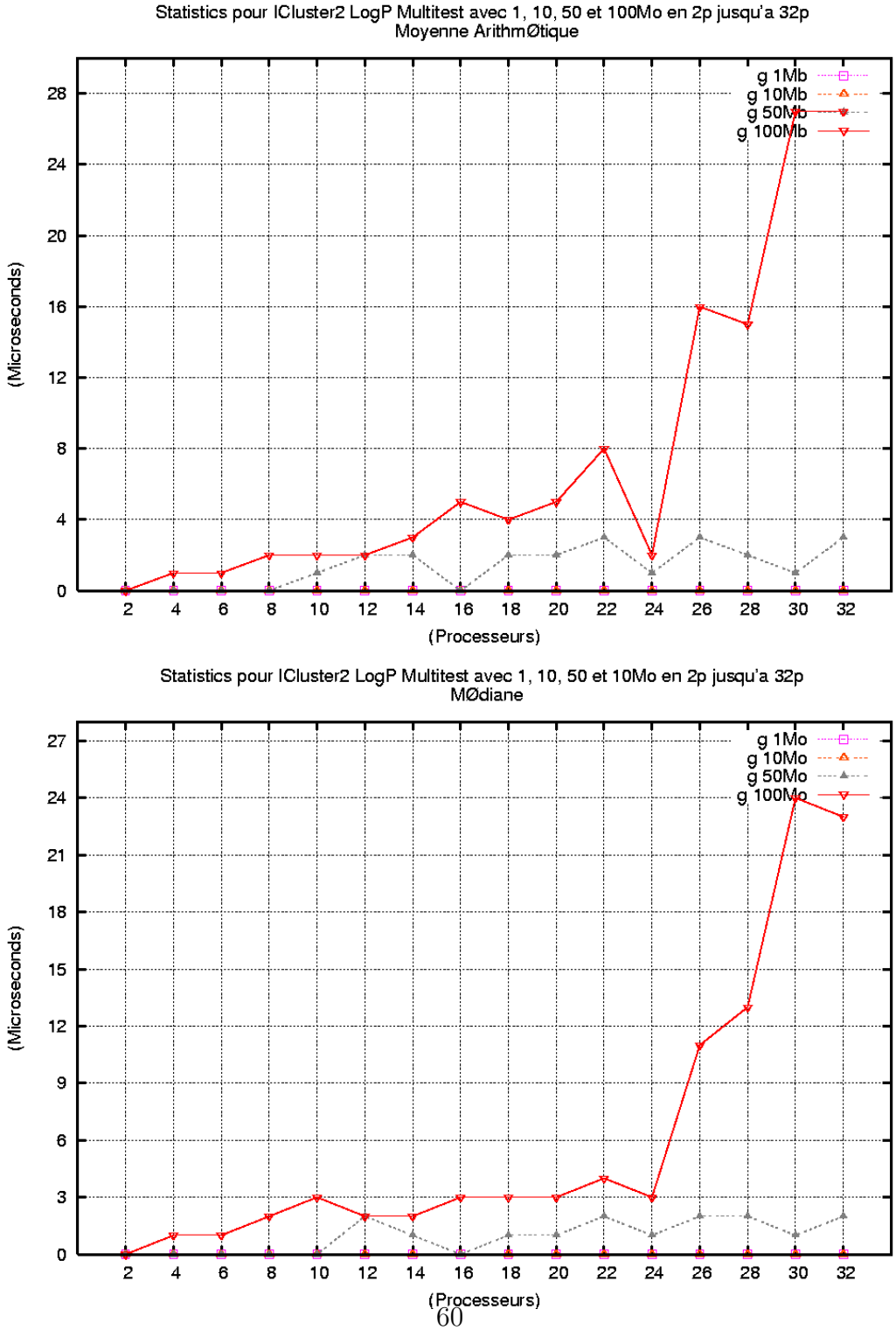


FIG. 3.6 – *Statistiques de l'expérience de mesure de g sur le ICluster2*

P	1Mo	10Mo	50Mo	100Mo
2	0.00	0.04	0.2	0.4
4	0.72	0.05	0.23	0.58
6	0.01	0.21	128.75	2.74
8	597.20	0.04	0.57	1274.40
10	0.00	166.90	0.16	1624.06
12	0.88	0.27	1.46	617.73
14	0.00	0.17	1.45	784.07
16	793.93	407.81	982.55	11307.16
18	0.01	615.26	3.47	1046.16
20	0.81	0.23	1904.37	2.14
22	362.38	1622.94	2138.81	391.28
24	842.16	1151.45	1589.17	4.13
26	94.74	126.26	4201.38	1083.35
28	1129.57	20.25	97.45	9.57
30	27.26	1.98	2266.08	10.4
32	4.80	4.23	1312.65	244.16

TAB. 3.2 – *Table des Latences Pour ICluster2.*

pour les autres tailles amène à identifier des coïncidences sur le nombre de processeurs où se présente des variations importantes. Un étude plus détaillée sur ce point serait intéressante mais n'est pas faite dans ce travail.

Comme pour IDPOT, nous présentons une table 3.2 des latences calculées avec le modèle mathématique basé en *LogP* et les données trouvées lors de l'expérience.

3.4 Modélisation

A partir du modèle *LogP* présenté auparavant et l'équation (4) présentée dans la première partie nous proposons le calcul de latence suivant :

$$\bar{Lat}_p(m) =\mid L_{exp} + \bar{g}(m) - \bar{O}_s(m) - \bar{O}_r(m) \mid \tag{3.1}$$

Où $\bar{Lat}_p$ est la latence calculée pour chaque paire de processeurs pour le transfert d'un message de taille m, depuis les moyennes de $g(m)$, $O_s(m)$ et $O_r(m)$ respectivement. Les résultats sont présentées dans les tables 3.1 et 3.2. Il est important de considérer la latence expérimentale nommée L_exp.

A partir des résultats on peut construire le graphique 3.7.

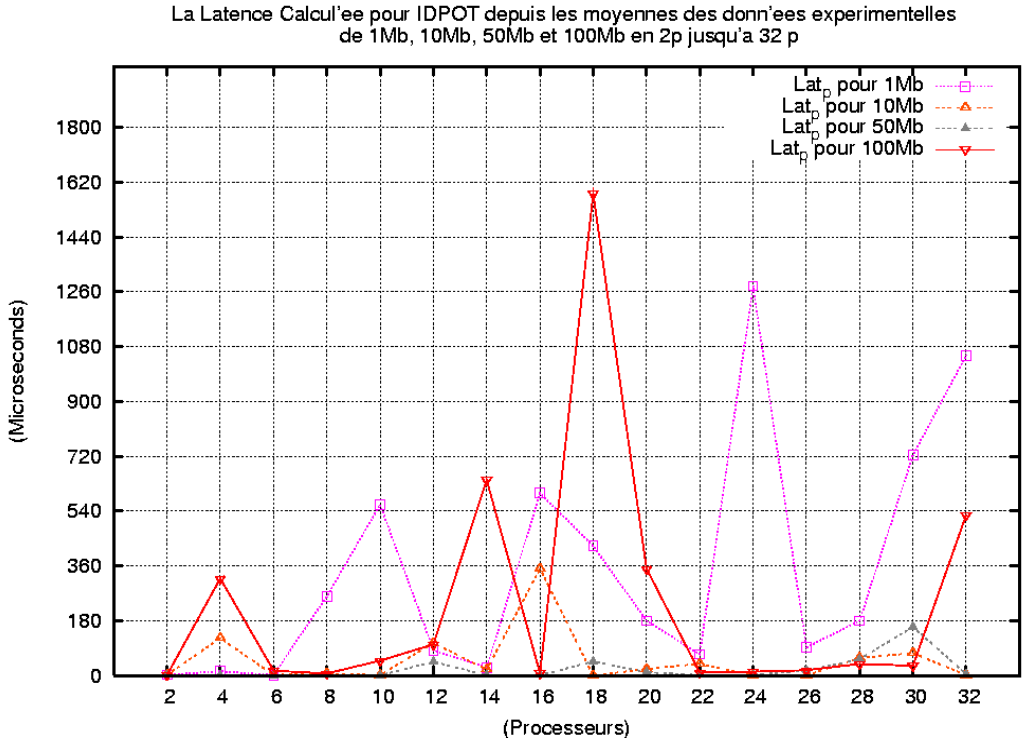
Nous proposons une mesure de $RTT(m)$ avec l'expression suivante :

$$\bar{RTT}_p(m) =\mid 2(Lat_p(m) + \bar{g}(m)) \mid \tag{3.2}$$

En accord avec la théorie détaillée dans la première partie, la mesure de RTT est importante parce qu'elle permet de connaître la valeur réelle de $r(m)$, qui est fondamentale pour le transfert TCP. Dans ce cas particulier nous allons calculer le valeur de $r(m)$ pour connaître le temps d'envoi du message de taille m depuis une source vers la destination et la réponse finale pour la destination, comme expliqué dans la première partie. Nous avons utilisé $\bar{RTT}_p(m)$ car la mesure est entre pairs, il est donc clair que nous avons calculé la moyenne pour chaque nombre de processeurs et pour chaque taille.

A parti des tables 3.3 et 3.4 il est possible de construire des graphiques pour analyser le compartemment du $r(m)$ calculé selon la taille de m sur la figure 3.8. Sur celui-ci, on peut voir que le temps r est équivalent sur les deux grappes, sauf pour 100Mo où se présente un pic avec l'utilisation de 18 et 16 processeurs respectivement. Le calcul pour des messages de grand taille est affecté par les caractéristiques déjà expliquées. On peut également identifier une performance plus irrégulière sur IDPOT que sur le ICluster2 et aussi d'importantes variations dans les autres tailles sur une des grappes, plus que sur l'autre. La question intéressante qu'il faudrait de regarder plus en détails dans le travail en cours, est la différence de comportement pour des tailles identiques. L'influence de certains aspects sur les résultats est claire, comme les architectures de grappe, l'état de la grappe[2] mais aussi, peut être

[2]Pendant l'experimentation, IDPOT a présenté d'importantes défaillances, mais les essais sont reproduits pour capturer les données et vérifier les valeurs.



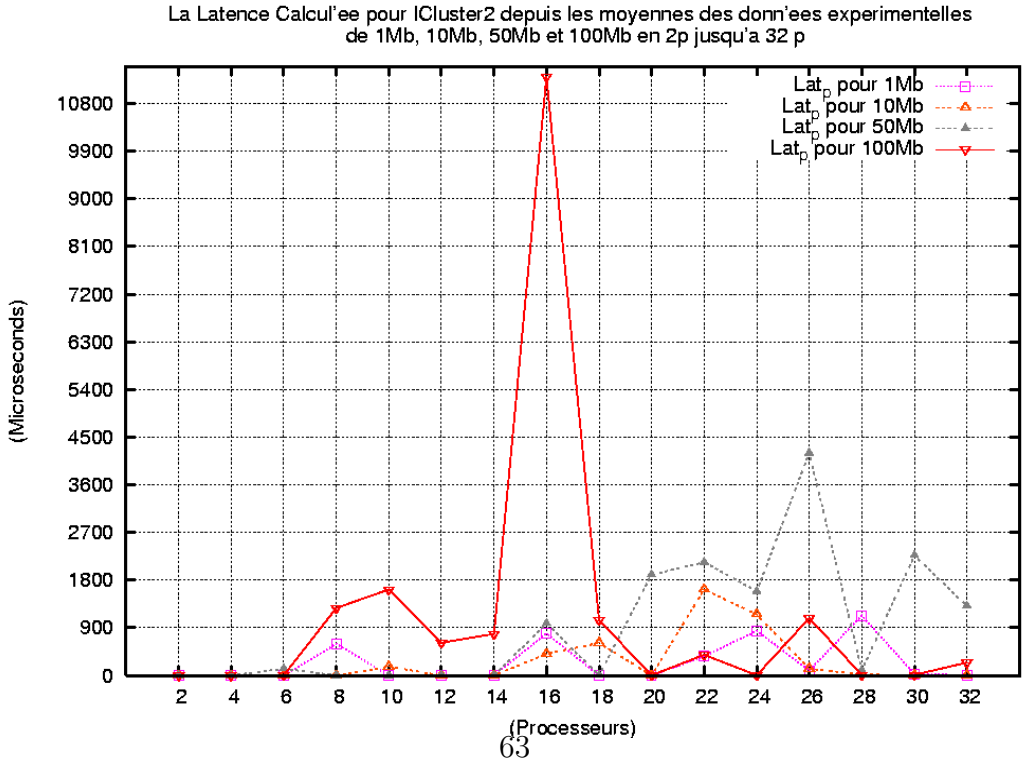


FIG. 3.7 – *Le Calcul de la Latence pour IDPOT et ICluster2 depuis les moyennes des données trouvées.*

P	1Mo	10Mo	50Mo	100Mo
2	3.64	3.87	0.84	3.88
4	27.37	245.75	0.17	638.25
6	2.62	150.39	1.26	33.01
8	518.64	14.56	4.53	16.92
10	1120.89	44.63	1.53	100.98
12	166.02	987.38	93.35	209.72
14	51.15	46.09	1.45	1290.03
16	1196.89	4463.61	1.73	7.41
18	850.78	569.38	96.19	3173.92
20	359.17	67.49	26.87	711.06
22	139.17	12674.66	7.68	8.72
24	2559.17	5213.42	1.43	13.04
26	184.50	1245.43	39.32	21.44
28	360.01	1016.68	113.66	61.42
30	1446.97	12515.98	326.98	43.12
32	2101.97	185.45	0.66	1086.65

TAB. 3.3 – *Table de $r(m)$ Pour IDPOT.*

P	1Mo	10Mo	50Mo	100Mo
2	0.05	0.07	0.34	0.70
4	1.45	0.14	1.45	1.26
6	0.01	0.03	258.40	1.70
8	1194.42	0.23	0.01	3452.81
10	0.02	334.11	1.99	3253.81
12	1.79	0.11	1.13	1240.49
14	0.01	0.07	2.27	1575.98
16	1587.93	816.08	1967.00	22626.01
18	0.01	1231.55	11.97	2101.5
20	1.65	0.03	3814.69	7.13
22	724.79	3246.86	4284.87	800.07
24	1684.38	2303.51	3182.12	13.97
26	189.58	253.43	8409.89	2199.37
28	2459.18	41.29	199.52	10.97
30	54.57	5.24	4535.91	34.88
32	9.67	9.24	2631.7	453.28

TAB. 3.4 – *Table de $r(m)$ Pour ICluster2.*

une faiblesse du modèle. C'est un point intéressant d'étude pour des travaux futurs.

Finalement nous proposons les calculs de o et de G afin de décrire complètement les caractéristiques des réseaux, avec une extension de Logp à LogGP qui est plus adapté aux messages de grande taille.

Le calcul de o est fait par l'équation :

$$\bar{O}_p = \frac{\bar{O}s(m) + \bar{O}r(m)}{2} \tag{3.3}$$

Et la valeur de G est calculée par l'expression :

$$\bar{G} = \frac{\bar{g}(m)}{m} \tag{3.4}$$

On peut alors, pour chacune des grappes évaluées, construire une table de ses caractéristiques respectives, selon la taille du message. Ainsi, chaque performance peut être résumée pour $N = (L, o, g, G, P)$ comme expliqué en [10]. Les caractéristiques pour IDPOT sont présentées dans les tables : 3.5, 3.6, 3.7 et 3.8 pour 1Mo, 10Mo, 50Mo et 100Mo respectivement.

Pour le ICluster2, on peut montrer les caractéristiques avec l'utilisation du modèle LogGP dans les tables :3.9, 3.10, 3.11 et 3.12.

Les données des tables pour IDPOT et ICluster2 sont reliées aux figures utilisées pendant tout le rapport.

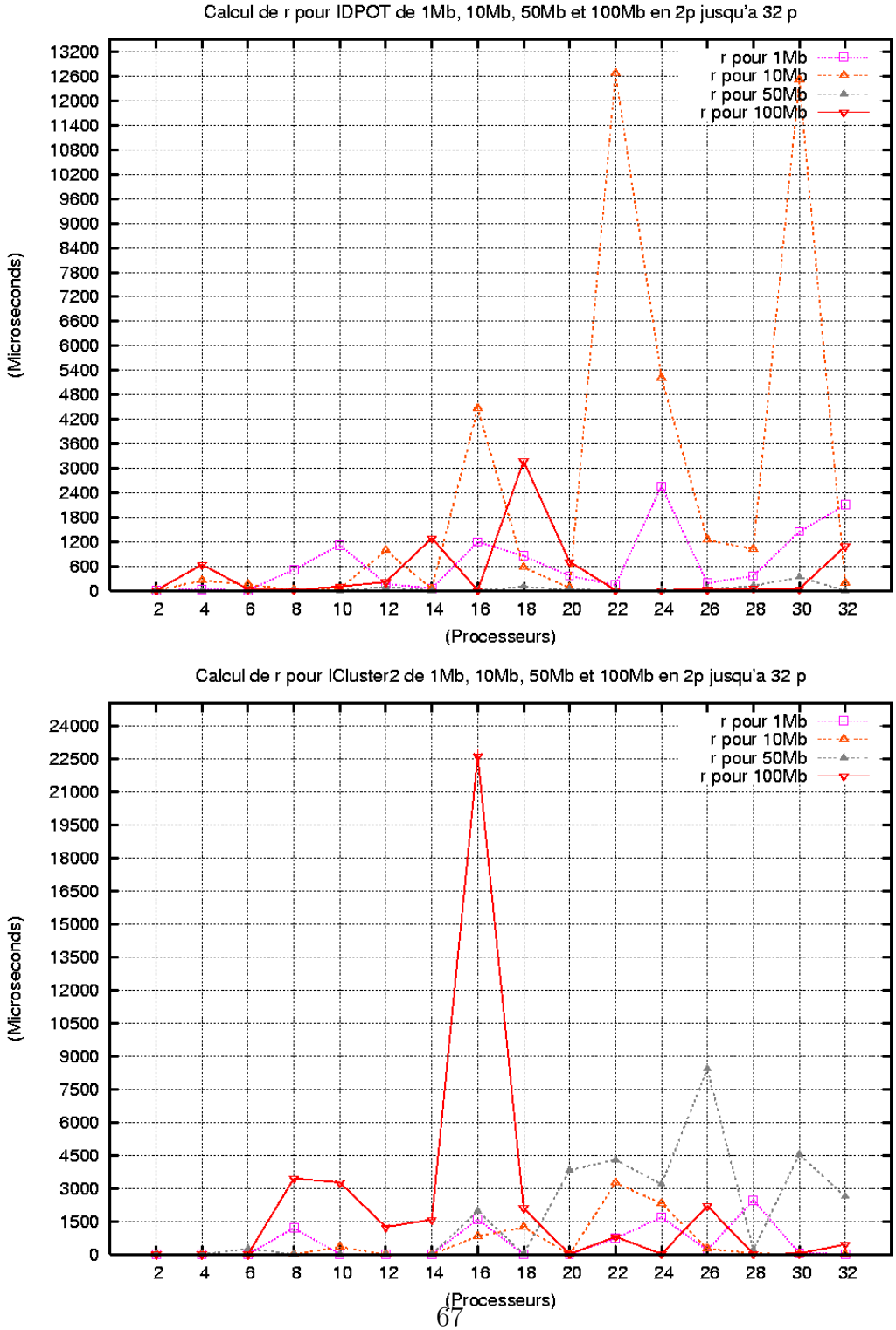


FIG. 3.8 – *Le Calcul de RTT pour IDPOT et ICluster2 depuis les données expérimentales.*

P	L	o	g	G
2	1,82	0,00	0,00	3,18E-09
4	13,68	0,00	0,00	7,58E-09
6	1,30	0,01	0,01	1,32E-08
8	259,31	0,01	0,01	6,50E-09
10	560,44	0,01	0,00	6,06E-09
12	83,01	0,00	0,00	5,38E-09
14	25,57	0,01	0,01	1,02E-08
16	598,43	0,01	0,01	1,03E-08
18	425,37	0,01	0,01	1,90E-08
20	179,54	0,03	0,02	3,85E-08
22	69,87	0,03	0,04	5,74E-08
24	1279,50	0,05	0,05	7,62E-08
26	92,24	0,03	0,01	1,59E-08
28	179,99	0,03	0,00	1,60E-08
30	723,46	0,06	0,01	2,69E-08
32	1050,93	0,04	0,03	5,43E-08

TAB. 3.5 – *Table de LogGP pour IDPOT avec une taille de message de 1Mo*

P	L	o	g	G
2	1,9	0,04	0,03	3,23E-09
4	122,8	0,09	0,07	6,68E-09
6	75,09	0,17	0,10	9,75E-09
8	7,12	0,10	0,16	1,57E-08
10	22,17	0,26	0,14	1,37E-08
12	493,5	0,20	0,19	1,79E-08
14	22,83	0,30	0,22	2,06E-08
16	2231,38	0,47	0,42	4,01E-08
18	284,51	0,21	0,18	1,74E-08
20	33,51	0,45	0,24	2,27E-08
22	6336,87	0,5	0,46	4,35E-08
24	2606,24	0,43	0,47	4,44E-08
26	622,44	0,31	0,27	2,58E-08
28	507,91	0,80	0,43	4,11E-08
30	6257,32	0,82	0,67	6,37E-08
32	92,35	0,72	0,37	3,56E-08

TAB. 3.6 – *Table de LogGP pour IDPOT avec une taille de message de 10Mo*

P	L	o	g	G
2	0,25	0,18	0,17	3,19E-08
4	0,26	0,36	0,34	6,52E-08
6	0,13	0,49	0,50	9,60E-08
8	1,55	0,69	0,72	1,37E-07
10	0,23	0,93	0,99	1,89E-07
12	45,41	1,19	1,26	2,41E-07
14	0,83	1,44	1,56	2,97E-07
16	0,95	1,67	1,81	3,45E-07
18	45,96	1,98	2,14	4,08E-07
20	10,95	2,34	2,48	4,74E-07
22	1,07	2,63	2,77	5,29E-07
24	2,35	2,88	3,06	5,84E-07
26	16,33	3,11	3,33	6,36E-07
28	53,28	3,49	3,55	6,78E-07
30	159,69	3,90	3,80	7,25E-07
32	3,81	4,53	4,15	7,91E-07

TAB. 3.7 – *Table de LogGP pour IDPOT avec une taille de message de 50Mo*

P	L	o	g	G
2	1,63	0,34	0,31	2,96E-09
4	318,49	0,84	0,63	6,04E-08
6	15,71	0,65	0,79	7,55E-08
8	6,62	1,36	1,84	1,75E-07
10	48,59	1,59	1,90	1,81E-07
12	101,96	2,80	2,90	2,77E-07
14	642,61	2,79	2,41	2,30E-07
16	6,83	4,97	3,12	2,98E-07
18	1582,44	4,75	4,52	4,31E-07
20	350,98	8,07	4,55	4,34E-07
22	9,53	8,73	5,17	4,93E-07
24	12,47	12,68	5,95	5,67E-07
26	17,10	16,85	6,39	6,09E-07
28	37,95	22,61	7,24	6,91E-07
30	32,25	23,33	10,69	1,02E-06
32	526,70	45,66	16,62	1,59E-06

TAB. 3.8 – *Table de LogGP pour IDPOT avec une taille de message de 100Mo*

P	L	o	g	G
2	0,00	0,01	0,02	2,02E-08
4	0,72	0,00	0,01	6,21E-09
6	0,01	0,01	0,01	9,75E-09
8	597,20	0,01	0,01	6,55E-09
10	0,00	0,00	0,01	8,53E-09
12	0,88	0,01	0,01	1,13E-08
14	0,00	0,01	0,01	8,01E-09
16	793,936	0,02	0,03	2,97E-08
18	0,01	0,01	0,01	9,13E-09
20	0,81	0,03	0,02	2,03E-08
22	362,38	0,02	0,01	1,32E-08
24	842,16	0,08	0,03	3,05E-08
26	94,74	0,05	0,05	4,43E-08
28	1229,57	0,04	0,02	2,06E-08
30	27,26	0,04	0,03	2,73E-08
32	4,80	0,06	0,03	3,13E-08

TAB. 3.9 – *Table de LogGP pour ICluster2 avec une taille de message de 1Mo*

P	L	o	g	G
2	00,04	0,05	0,07	6,91E-09
4	00,05	0,09	0,12	1,18E-08
6	00,21	0,21	0,20	1,91E-08
8	00,04	0,10	0,15	1,47E-08
10	166,90	0,14	0,15	1,48E-08
12	00,27	0,30	0,33	3,12E-08
14	00,17	0,19	0,21	2,01E-08
16	407,81	0,28	0,23	2,23E-08
18	615,26	0,45	0,51	4,89E-08
20	00,23	0,23	0,22	2,10E-08
22	1622,94	0,46	0,49	4,67E-08
24	1151,45	0,46	0,31	2,93E-08
26	126,26	0,69	0,45	4,32E-08
28	20,25	0,70	0,40	3,77E-08
30	01,98	1,12	0,64	6,07E-08
32	04,23	0,65	0,39	3,72E-08

TAB. 3.10 – *Table de LogGP pour ICluster2 avec une taille de message de 10Mo*

P	L	o	g	G
2	0,20	0,28	0,37	6,98E-08
4	0,23	0,41	0,49	9,39E-08
6	128,75	0,33	0,45	8,55E-08
8	0,57	0,57	0,57	1,10E-07
10	0,16	0,66	1,16	2,20E-07
12	1,46	1,74	2,02	3,86E-07
14	1,45	2,02	2,59	4,93E-07
16	982,55	1,10	0,95	1,81E-07
18	3,47	2,78	2,51	4,79E-07
20	1904,37	3,19	2,98	5,68E-07
22	2138,81	4,70	3,63	6,92E-07
24	1589,17	2,10	1,89	3,61E-07
26	4201,38	4,65	3,57	6,80E-07
28	97,45	3,73	2,31	4,41E-07
30	2266,08	2,72	1,87	3,57E-07
32	1312,65	4,33	3,20	6,10E-07

TAB. 3.11 – *Table de LogGP pour ICluster2 avec une taille de message de 50Mo*

P	L	o	g	G
2	0,40	0,57	0,74	7,09E-09
4	0,58	1,25	0,10	1,15E-08
6	2,74	2,31	0,61	1,80E-08
8	1724,40	2,24	0,50	1,92E-08
10	1624,06	2,49	2,47	2,52E-08
12	617,73	1,96	1,70	2,40E-08
14	784,07	3,49	3,82	3,74E-08
16	11307,16	4,16	3,95	5,57E-08
18	1046,16	7,07	1,83	4,38E-08
20	2,14	6,90	4,23	5,44E-08
22	391,28	8,58	5,76	8,35E-08
24	4,13	3,91	1,24	2,72E-08
26	1083,35	9,34	7,10	1,56E-07
28	9,57	12,31	16,93	1,44E-07
30	10,40	20,19	28,90	2,65E-07
32	244,16	14,46	21,70	2,62E-07

TAB. 3.12 – *Table de LogGP pour ICluster2 avec une taille de message de 100Mo*

Chapitre 4

Conclusion finale

D'après d'analyses des données et la modélisation on peut suggérer les conclusions suivantes :

- Pour le transfert haut débit dans les deux grappes il existe une importante variation et instabilité pour le transfert de messages de taille 100Mo. On peut également constater quelques comportements irréguliers pour des essais de plus petite taille avec l'utilisation de plusieurs processeurs. Les travaux existants consultés expliquent ce phénomène par des points de *bottleneck*, on peut donc affirmer que ces points que nous avons appellé comme points instables sont dues à des goulets. Il est intéressant de regarder, par exemple, que, pour des valeurs relativement faibles, ces points apparaissent. Toutefois, la théorie explique ce comportement par l'état d'occupation des noeuds ou du réseau ou pour d'autres choses comme la diffusion des données.
- Autre conclusion importante, à partir de ce travail on peut construire un modèle qui décrit les caractéristiques de chaque grappe étudiée. Il est donc possible, à partir de cette première étude, de proposer un modèle général pour la performance de *grid5000* et d'identifier des aspects importants comme la taille limitée des messages qui peuvent être transférés effectivement par l'environnement et faire une description détaillée de chaque noeud élément de *grid5000*.
- Bien que le modèle LogGP et les programmes développés pour son implémentation soient bien acceptés pour la réalisation de ce type de mesures de performance, et dans ce travail a montré être un outil efficace, il est possible de proposer une analyse du modèle mathématique et des autres aspects qui ne sont pas suffisamment pris en compte mais

affectent les données de mesure particulièrement pour des systèmes hétérogènes comme *grid5000*[1] qui peuvent générer un nouveau modèle complémentaire basé sur LogP[2].

[1]Et incluant des systèmes comme les grappes analysées et qui sont complètement différentes.

[2]Ou non.

Remerciements

Je le remercié à **l'Ambassade de France en Colombie** et au programme de bourses de la Commision d'Etudes en France qu'a permis de poursuivre mes etudes de Master 2 Recherche à Grenoble. Au Dr. **Yves Denneulin** pour donner-moi l'opportunité de travailler sous son direction dans le laboratoire ID-IMAG. Aussi à sa femme, Claudia pour faire le contact avec lui. Au Dr. **Guillermo Uribe** et **Amelita Castellanos**, mes *parents adoptives* en France pendant cette année. À **Luiz-Angelo Stephanel** pour le code de *LogP_multitest* et ses recommendations, aussi au Dr. **Gregory Mounie** pour ses importants points de vue et des resolutions à mes questions. Je le remercie à mes amis colombiens et latinos en Grenoble : Lina, John, Jimmy, Mar, Evelio, Mario, Santiago, Javier..., et au personel et mes copains du **Laboratoire ID-IMAG**, en spécial à mes camarades de boureau Yannis, Thrung et Erick pour l'excelente ambiance de travail.

Je le dédie ce rapport à mon grand pêre, *Luis Gustavo*, par s'exemple de vie à suivri.

Bibliographie

[1] Alexandrov, A., Ionescu, M., Schauser, K. et Scheiman, C. *LogGP : Incorporating Long Messages into the LogP Model*. Proceedings in 7th Annual ACM Syposium on Parallel Algorithms and Architectures, Santa Barbara, CA, E.U.A. 1995.

[2] Casanova, H. *SIMGRID : A Toolkit for the Simulation of Application Scheduling*. Document d'Internet. http ://grail.sdsc.edu/papers/ . Université de San Diego, CA, E. U. A. 2001.

[3] Casanova, H. *Network Modeling Issues for Grid Application Scheduling*. San Diego Computer Center, Department of Computer Science and Engineering, University of California at San Diego, E.U.A., 2003.

[4] Culler, D., Karp, Patterson, D., Sahay, A., Schauser, K., Santos, E., Subramonian, R., Von Eicken, T. *LogP : Towards a Realistic Model of Parallel Computation.* Proceedings in Four ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, San Diego, C.A. E.U.A. 1993.

[5] Foster, I. et Kesselman, C. *The Grid : Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann Publishers, E.U.A. 1998.

[6] GGF Network Measurements Working Group. *A Hierarchy of Network Performance Characteristics for Grid Applications and Services*. Document d'Internet, http ://nmwg.internet2.edu/docs/nmwg-measurements-v14.pdf . 2003.

[7] INRIA Rhône - Alpes *La Grappe I-Cluster2 : Presentation Générale.* Document d'Internet, http ://www.inrialpes.fr/sed/i-cluster2/welcome.html, Institute National de Recherche en Informatique et en Automatique, Region Rhône - Alpes, France, 2005.

[8] ISO/IEC 7498-1. *Information Technology– Open Systems Interconnection– Basic Reference Model : The Basic Model*. Document d'Internet, http ://www.iso.org .

[9] Joseph, J., Ernest, M. et Fellenstein, C. *Evolution of Grid Computing : Architecture and Grid Adoption Models*. IBM Systems Journal, Vol. 43. No. 4, E.U.A., 2004.

[10] Kielmann, T., Bal, H., et Verstoep, K. *Fast Measurement of LogP Parameters for Message Passing Platforms*. Lecture Notes In Computer Science ; Vol. 1800, Proceedings of the 15 IPDPS 2000 Workshops on Parallel and Distributed Processing, Springer - Verlag, U.K., 2000.

[11] Lai, K. et Baker, M. *Measuring Link Bandwidths Using a Deterministic Model of Packet Delay*. Proceedings in Special Interest Group on Data Communications, Stockholm, Sweden. 2000.

[12] Legrand, A. et Quinson, M. *Automatic Deployment of the Network Weather Service using the Effective Network View*. Rapport Technique du Laboratoire Informatique et Distribution "ID-IMAG", France, 2003.

[13] Leduc, J. *La Grappe IDPOT : Presentation Générale.* Document d'Internet, http ://idpot.imag.fr . Laboratoire de Informatique et Distribution"ID-IMAG", France, 2004.

[14] Lombard, P. *NFSP : Une Solution de Stockage Distribué pour Architectures Grande Échelle*. Thése, Institut National Polytechnique de Grenoble. Laboratoire Informatique et Distribution "ID-IMAG", France, 2003.

[15] Marchal, L. *A Network Model for Simulation of GRID Application*. Rapport de Stage MIM-2. ENS-Lyon, France, 2002.

[16] Moritz, C. et Frank, M. *LogGPC : Modeling Network Contention in Message-Passing Programs*. IEE Transactions on Parallel and Distributed Systems. Vol. 12. No. 4, E.U.A. 2001.

[17] Németh, Z. et Sunderam, V. *Characterizing Grids : Attributes, Definitions, and Formalisms*. Journal of Grid Computing 1 ; Klewer Academic Publishers, Hollande, 2003.

[18] Ould-Khaoua, M., Sarbazi-Azad, H. et Obaidat M. *Performance Modeling and Evaluation of High Performance Parallel and Distributed Systems*. Performance Evaluation Journal No. 60, Editorial. E.U.A. 2004.

[19] Schoonderwoerd, R. *Network Performance Measurement Tools : A Comprehensive Comparison*. Thése de Master, Vrije Universiteit, Hollande, 2002.

[20] Statistics Canada. *Les Statistiques : Le Pouvoir des Données* Document d'Internet, http ://www.statcan.ca/francais/edu/power/toc/contents_f.htm . Agence Statistics Canada, Canada, 2005.

[21] Tracy, G. *LogGP : Analysis of Parallel Sorting Algorithms.* Document d'Internet. http ://www.cs.wisc.edu/ gtracy/classes/747/project.htm . Université de Winsconsin. E.U.A. 2000.

[22] Vincent, J.M. et Chassin de Kergommeaux, J. *Notes du Cours Module MD : Mesure et Analyse de Données pour l'évaluation de Performances de Réseaux et de Systèmes.* Documents d'Internet. http ://www-apache.imag.fr/ jvincent/dea/ . Laboratoire de Informatique et Distribution "ID-IMAG". France. 2005.