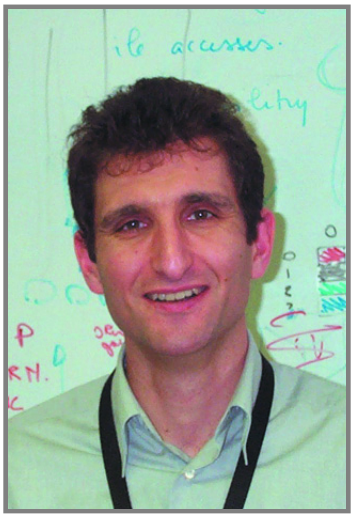
TECHNOLOGIE

Dr. Franck Cappello
Responsable du groupe
Clusters & Grilles
du LRI
Chargé de Recherche
au CNRS
Université Paris Sud
www.lri.fr/~ fci
fci@lri.fr



Calcul Global Pair à Pair : extension des systèmes Pair à Pair au calcul

▶ 1 – Introduction

La disponibilité simultanée de nombreuses ressources de calcul/stockage performantes, d'un réseau de communication mondial suffisamment efficace et stable pour véhiculer rapidement des grands volumes d'informations et d'applications nécessitant des très grandes capacités de calcul et de stockage a donné naissance au concept de Systèmes Distribués à Grande Échelle (SDGE). Les projets étudiés et les infrastructures proposées émanent de différentes communautés et peuvent être classés en trois catégories : les systèmes de GRID, les systèmes de Calcul Global et les systèmes de partage de ressources entre Pairs (*Peer-to-Peer*).

Le *Metacomputing* [Sma92] et les systèmes de GRID (Globalisation des Ressources Informatiques et des Données) [Fos98], activement développés pour les applications scientifiques de calcul (EuroGRID) ou de stockage massif (DataGRID) ont pour ambition d'établir une infrastructure permettant 1) de faire interopérer des ressources de calcul, des grands instruments de mesure, des grandes bases de données et des centres de visualisation et de réalité virtuelle et 2) de faciliter l'accès et l'organisation de ces ressources pour l'exécution d'applications scientifiques. Les systèmes de Calcul Global (*Global Computing*) concernent aussi les applications scientifiques mais impliquent la participation de volontaires particuliers trouvant un intérêt social dans une expérience spécifique (lutte contre le cancer, recherche pharmaceutique, recherche d'intelligence extraterrestre, généthon). Le principe du Calcul Global est différent de celui des GRID. Il s'agit d'utiliser un très grand nombre d'ordinateurs volontaires distribués géographiquement à l'échelle mondiale et communiquant uniquement par Internet pour exécuter des applications informatiques de très grande taille. Le modèle de calcul est un parallélisme massif nécessitant peu de communications entre les sites. Le modèle d'exploitation est l'utilisation des ordinateurs d'institutions ou d'individus volontaires, essentiellement pendant leurs périodes d'inactivité. Les systèmes de partage de ressources Pair à Pair ont, jusqu'à maintenant, concerné principalement la communauté des utilisateurs d'Internet cherchant à échanger des documents multimédias (musiques, films, articles). Dans ces systèmes, toutes les machines peuvent remplir le rôle de client et de serveur ; d'où le terme de ServEnt dans Gnutella [Kan01]. Les ressources partagées peuvent être de différentes natures : données, espaces de stockage, puissance CPU. Le rôle de l'infrastructure système est de mettre en relation directe les machines recherchant des ressources (les clients) et celles disposant des ressources recherchées (les serveurs), à un instant donné.

Cet article examine plus particulièrement la problématique des systèmes de partage de ressources Pair à Pair appliqués au calcul. Comme dans le cas de partage de données, toutes les machines du système peu-

vent remplir les rôles de client et de serveur et il s'agit de mettre en relation des machines clientes désirant exécuter des calculs et des machines serveurs.

▶ 2 – Classification

Il n'existe pas de consensus sur la définition des systèmes Pair à Pair. Aussi, pour mieux identifier et comprendre les propriétés de ces systèmes, il est préférable de les classer suivant différents paramètres discriminants. Il est intéressant de constater dans la littérature que les systèmes de Calcul Global sont considérés comme une forme de système Pair à Pair [Bar01]. Pour présenter les distinctions entre ces systèmes, nous intégrons dans notre classification les systèmes de Calcul Global.

Dans tous les cas, un système Pair à Pair suppose que les machines collaborent à un but commun qui peut être un calcul et/ou le stockage/partage d'informations.

Le premier paramètre discriminant concerne la propriété pour une même machine de fonctionner alternativement ou simultanément comme un client et/ou un serveur du système. Les systèmes de Calcul Global comme SETI@home, Entropia, UnitedDevice ne respectent pas cette propriété contrairement à XtremWeb (http://www.xtremweb.net/), à ActiveCluster de Platform et à tous les systèmes d'échange de fichiers Pair à Pair connus.

L'une des caractéristiques de certains systèmes Pair à Pair est, qu'une fois l'étape de mise en relation terminée, la transmission de données est réalisée sans intermédiaire, entre pairs. Ce n'est pas le cas pour FreeNet [Lan01], où la transmission de l'information est réalisée par l'infrastructure de mise en relation. Cette propriété est aussi absente dans les systèmes de Calcul Global. L'une des questions ouvertes sur les systèmes de Calcul Global Pair à Pair concerne justement l'intérêt d'établir une connexion directe entre client et serveur de calcul.

Certains systèmes comme SETI@home, Napster et XtremWeb utilisent une architecture centralisée. Dans Napster, le répertoire qui permet à un client d'identifier les serveurs potentiels est centralisé. Dans SETI@home et XtremWeb, l'ordonnanceur de calcul et le collecteur de résultats sont centralisés. À l'inverse, les architectures Pair à Pair récentes comme Gnutella, Freenet et Fastrack utilisent une infrastructure (*backbone*) distribuée pour remplir la fonction de mise en relation des clients et des serveurs.

La capacité d'auto-organisation du système est quelque fois associée au principe de communauté dynamique des systèmes de Calcul Global et des systèmes Pair à Pair. Cette propriété fait référence à la possibilité pour tous les nœuds de rejoindre ou de quitter à tout moment le système. Du point de vue infrastructure système, l'auto-organisation signifie la propriété de reconfiguration dynamique et totalement distribuée

du système en fonction du nombre et des caractéristiques des nœuds qui composent le système. Par exemple, FastTrack et Gnutella permettent à certains nœuds de jouer le rôle de super-nœuds qui serviront de répertoires de recherche pour améliorer les performances du système. L'introduction de caches dans le système permet aussi d'exploiter les propriétés de localité spatiale et temporelle des accès aux documents.

Le tableau suivant résume la classification des systèmes de Calcul Global et Pair à Pair. Le type d'opérations réalisées est mentionné par un C pour calcul et par un D pour données.

	Collaboration à un but commun	Rôles client et serveur possible	Connexion directe client/serveur	Infrastructure distribuée	Système auto organisé
Calcul Global	Oui / C				
XtremWeb I	Oui / C	Oui			
Napster (P2P)	Oui / D	Oui	Oui		
FreeNet (P2P)	Oui / D	Oui	Non	Oui	Oui
FastTrack (P2P)	Oui / D	Oui	Oui	Hiérarchique	Super-nœud
Gnutella (P2P)	Oui / D	Oui	Oui	Oui	Oui
P2P calcul	Oui / C + D	Oui	?	?	?

▶ 3 – Éléments fondamentaux d'architecture par l'exemple

À la base, un système Pair à Pair est composé d'un système de recherche de ressources (ou système de mise en relation) et d'un système de transport de documents (lorsque l'application vise à communiquer des documents). Des éléments supplémentaires interviennent dans les systèmes récents comme les nœuds d'indexation. Enfin, un système Pair à Pair doit fonctionner même lorsque des machines du système sont protégées par des pare-feux. Nous présentons ici ces éléments fondamentaux à partir d'exemples choisis d'architectures.

Le système de recherche de ressources (ou système de mise en relation)

Le système de recherche de ressources (ou de mise en relation) sert de mécanisme de recherche et d'identification d'un pair capable de fournir la ressource recherchée. La figure 1 présente les systèmes de recherche de ressources de Napster et de Gnutella dans sa version initiale.

Dans Napster, le système de recherche de ressources est centralisé et repose sur un répertoire stockant des relations ressource-pair. Lorsqu'un pair recherche une ressource, il obtient auprès du répertoire les adresses IP de pairs disposant de la ressource.

Le système de recherche de ressources de Gnutella fonctionne d'une façon totalement distribuée. Un pair n'est connecté qu'à un ensemble limité de pairs voisins. Il s'agit de voisins logiques identifiés unique-

Fig. 1

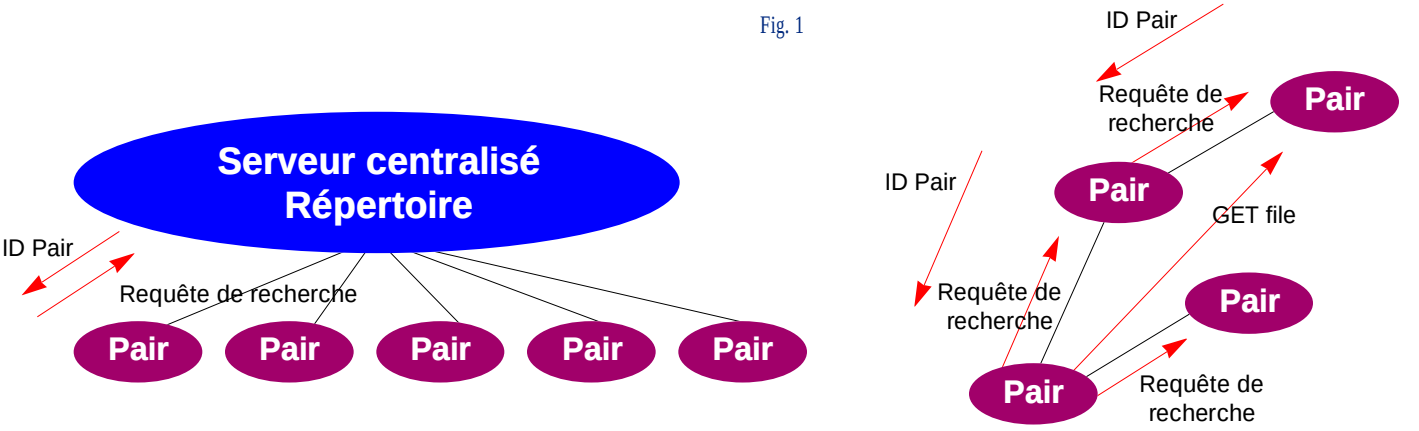


Fig. 1 – À gauche, l'architecture de Napster et à droite celle du système Gnutella (dans sa version initiale).

ment par leur adresse IP. L'ensemble des connexions forme une topologie logique qui n'a pas de correspondance avec la topologie physique d'Internet. Le système de mise en relation de Gnutella est construit par-dessus cette topologie logique et fonctionne par diffusion en utilisant le protocole TCP. Un pair qui recherche un document (ou un service) diffuse une requête à ses voisins dans la topologie logique. La requête est propagée par les voisins à leurs propres voisins. Le protocole agit par inondation des nœuds (pairs) présents dans le système. Si un pair qui reçoit la requête possède le document recherché, il ne propage plus la requête et envoie une réponse à destination de l'émetteur de la requête, en indiquant son adresse IP. La réponse suit le parcours inverse de la requête en se propageant de Pair en Pair jusqu'au destinataire. Cette méthode fonctionne car tous les pairs propageant une requête retiennent l'origine et la destination (voisins immédiats) par lesquels la requête est passée. Gnutella possède un mécanisme pour limiter l'inondation : chaque requête possède une durée de vie (un compteur de sauts) qui est décrémentée à chaque fois qu'un pair propage la requête. Un mécanisme permet aussi d'éviter les boucles dans le protocole d'inondation.

Le mécanisme de recherche de ressources de FreeNet fonctionne aussi de manière totalement distribuée, selon une organisation différente de celle de Gnutella. Comme dans Gnutella, un nœud FreeNet ne connaît que des voisins logiques. Cependant, dans FreeNet toute ressource est cryptée et identifiable à partir d'une clé unique. Un nœud FreeNet stocke des relations clés-adresses et le système de propagation de requêtes ne transmet la requête qu'au voisin possédant une clé proche de la clé de la ressource recherchée. Une clé peut représenter le cryptage du contenu de la ressource entière ou d'un ensemble de mots-clés identifiant la ressource.

Le système de transport de ressources

Dans Napster comme dans Gnutella, lorsque l'émetteur de la requête reçoit une réponse, il contacte directement le pair possédant le document recherché en utilisant l'adresse IP contenue dans la réponse. Dans Gnutella, le protocole de transfert consiste en une requête http (*get*) émise par le pair recherchant le document. Le pair qui possède le document reçoit la requête http sur un numéro de port différent de 80

et y répond en transmettant le document. FreeNet a été conçu pour minimiser la possibilité d'identifier les machines clientes et serveurs de documents. Il n'utilise pas de système de connexion directe pour éviter que le pair émetteur de la requête n'obtienne l'adresse IP du serveur. Le document est transmis au destinataire par le système de recherche de ressources lui-même qui devient aussi le système de transport. En fait, chaque nœud FreeNet fonctionne comme un cache pouvant stocker dans chaque relation clé-adresse la ressource correspondante. Le système de cache ne retient que les ressources les plus recherchées si bien que le temps de recherche d'une ressource souvent référencée est très rapide. En revanche, les ressources les moins recherchées ont peu de chance de se trouver dans le système de caches et demandent plus de temps de recherche. Des mécanismes de transports plus sophistiqués peuvent être mis en œuvre pour faciliter l'échange de fichiers très volumineux comme des films. Par exemple, dans Edonkey2000 l'émetteur de la requête peut solliciter plusieurs machines possédant un document (ou un segment du document) pour paralléliser le téléchargement. Il demande à chaque machine un segment différent du document. Dès qu'il a lui-même reçu un premier segment, il peut servir de serveur pour d'autres machines recherchant ce segment du document.

Les nœuds d'indexation

Pour limiter le nombre de messages nécessaires à la découverte d'une ressource, la notion de super-Pair (ou *reflector*) a été introduite. La figure 2 présente les systèmes d'indexation de Clip2 pour Gnutella et de FastTrach avec les super-nœuds.

Fig. 2 –

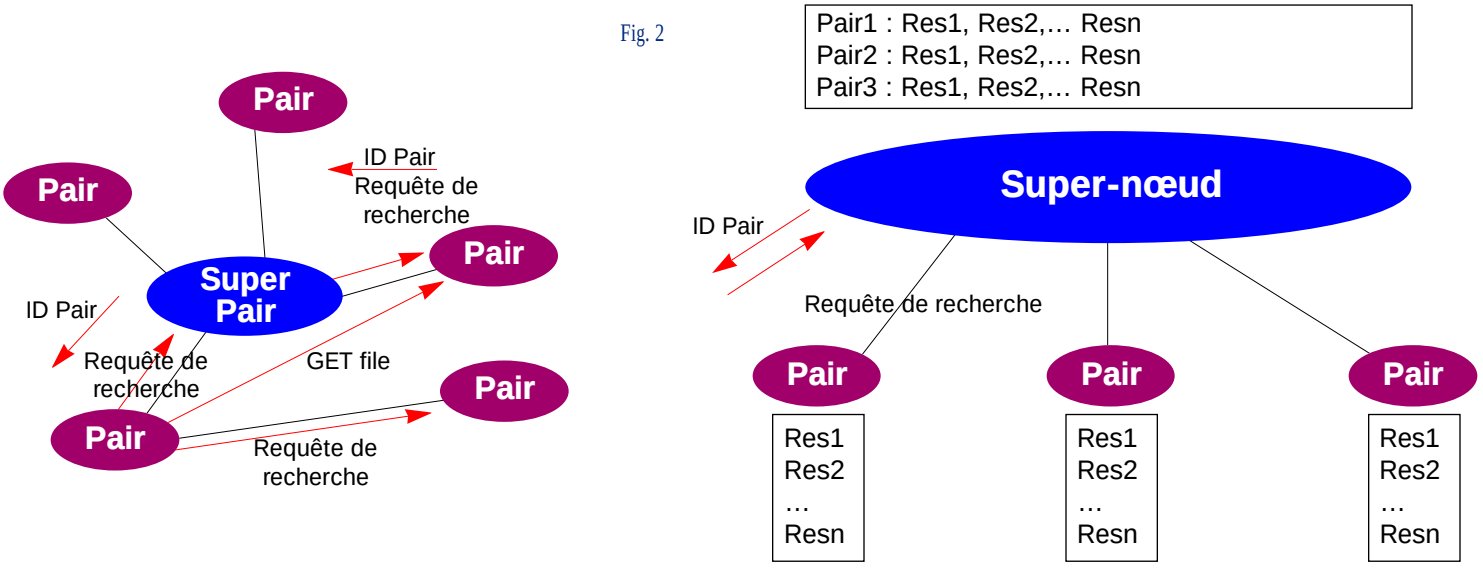


Fig. 2 – À gauche un super-Pair du système Clip2 et à droite un super-nœud du système FastTrack.

Dans le protocole compatible Gnutella proposé par la société Clip2, chaque super-Pair agit comme un mini serveur Napster ; c'est-à-dire qu'il stocke un index des ressources et des pairs possédant ces ressources. Ainsi, au lieu de propager une requête, le super-Pair, s'il trouve une référence de la ressource recherchée dans son index, répond

directement à l'émetteur de la requête. Les super-Pairs modifient uniquement le système de mise en relation. Le mécanisme de transport reste le même. Le système FasTrack fonctionne de manière analogue. L'un des problèmes liés à l'introduction de super-Pairs dans un système comme Gnutella est la cohérence avec la réalité des informations stockées dans l'index. En principe, un super-Pair doit mettre à jour son index lorsqu'une machine disposant d'une ressource se retire du système et lorsqu'une machine ne dispose plus d'une ressource.

Le problème des pare-feux

Dans tous les systèmes Pair à Pair, se pose le problème des pare-feux qui protègent les pairs connectés au système. Ce problème concerne d'abord le système de mise en relation. En effet, pour mettre en œuvre le système de mise en relation, il est nécessaire que des pairs acceptent des connexions externes. Si aucun pair n'accepte de connexion externe ou si leur nombre est insuffisant, le système ne pourra pas être mis en œuvre ou sera très limité en performance. Le problème des pare-feux se pose aussi pour le transport du document. Il arrive très souvent que le possesseur du document soit protégé par un pare-feu. Dans ce cas, la requête http de l'émetteur de la requête sera bloquée (port fermé). Pour contourner le problème Gnutella prévoit un mécanisme qui permet à l'émetteur de la requête de recontacter le possesseur du document par le système de mise en relation en lui demandant d'exécuter lui-même une requête http (*put*) en direction du destinataire. Ce mécanisme appelé « *push* » inverse les rôles pour la transmission du document. Bien évidemment si l'émetteur de la requête de document est lui-même protégé par un pare-feu, la transmission ne sera pas possible. C'est là une des limites intrinsèques des systèmes de transport de documents par connexion directe. La technique de transport utilisée dans FreeNet permet d'éviter, en partie, le problème des pare-feux puisque les pairs ne communiquent pas directement mais par l'intermédiaire du système de mise en relation.

► 4 – Technologies P2P

Pour construire un système Pair à Pair, il existe plusieurs points de départ suivant le type de réalisations visées. Pour simplifier, nous pouvons considérer trois types de réalisations possibles : créer un protocole de système Pair à Pair, créer un système distribué Pair à Pair, créer une application reposant sur une infrastructure Pair à Pair.

Pour créer un protocole de système Pair à Pair, les technologies logicielles sont à la base celles associées à l'Internet : protocoles TCP, http. Certains systèmes ont été construits par-dessus le protocole de messagerie classique SMTP. Rapidement, le mode d'interaction par appel de procédures distantes s'est avéré comme particulièrement adapté et des projets comme XtremWeb ont été construits par-dessus Java RMI ou JINI. Depuis les premières expériences et les premiers systèmes, d'autres protocoles ont été proposés comme SOAP et XML RPC. SOAP (*Simple Object Access Protocol*) qui encapsule des RPC encodés

au format XML dans des messages http (notons que SOAP n'est pas limité au protocole http) est retenu par beaucoup de projets comme protocole de base pour le développement. Indépendamment du protocole et du mode d'interaction, le système de mise en relation avec les mécanismes évoqués comme la durée de vie d'une requête, l'évitement de boucle dans le protocole de diffusion, le cache, etc. doivent être construits de toute pièce par le développeur. Il est à noter que le Peer-to-Peer Working Group, notamment soutenu par Intel, entend développer des technologies de base pour faciliter le développement de systèmes Pair à Pair en abordant, par exemple, les problèmes de la traduction des adresses IP (NAT), des pare-feux et de la sécurité. Actuellement, une bibliothèque de sécurisation des communications fondée sur OpenSSL est mise à disposition.

Pour simplifier la tâche du développeur, des environnements sont proposés permettant de construire des systèmes Pair à Pair plus facilement. Ainsi Jxta de SUN propose de structurer un système autour de quatre éléments : 1) les « *Peer pipes* », mécanismes de connexion d'un pair à un autre et de partage d'informations sur le réseau, de manière distribuée, 2) les « *Peer groups* » qui permettent la création de groupes de façon dynamique et le regroupement logique et cohérent de contenu, 3) la possibilité de surveiller et de mesurer les interactions et de définir des politiques de contrôle entre pairs (*Peer monitoring*) et, 4) des mécanismes de sécurité permettant de garantir la confidentialité, l'identité et l'accès contrôlé aux services.

Le projet COSM propose aussi de construire des environnements Pair à Pair plus simplement qu'en utilisant les protocoles de communication. Les projets Folding@home et Genome@home utilisent COSM comme plate-forme de base.

Le niveau supérieur dans la hiérarchie des environnements pour le développement de systèmes Pair à Pair consiste en des plates-formes complètes et opérationnelles dans lesquelles il « suffit » d'intégrer l'application désirée. FastTrack et XtremWeb proposent des plates-formes adaptées respectivement pour les applications d'échanges de documents et de calcul distribué. FastTrack est une plate-forme industrielle payante alors qu'XtremWeb est une plate-forme de recherche à code source ouvert. Tous les deux proposent des outils de configuration et d'administration permettant à l'administrateur d'installer simplement, de mettre en œuvre et de contrôler le système. Ils incluent aussi les mécanismes de diffusion des programmes clients et offrent des interfaces utilisateurs.

► 5 – Besoin d'outils pour la validation

L'engouement actuel pour les systèmes Pair à Pair se traduit par la naissance à un rythme élevé de protocoles, d'environnements de développement et de projets se référant aux systèmes Pair à Pair. Par exemple, le site SUN consacré à Jxta voit chaque mois augmenter le nombre de projets utilisant cette technologie comme base de dévelop-

pement. L'annonce de Plateform concernant son logiciel Active Cluster va sans doute mettre d'avantage l'accent sur ces systèmes et renforcer l'intérêt des industriels envers ceux-ci.

L'une des questions fondamentales actuelles, du point de vue scientifique mais aussi pratique, est l'efficacité de ces systèmes dans l'utilisation des ressources qu'ils mettent en œuvre et leur résistance aux défaillances les plus diverses. En plus des pairs qui participent par principe à un système Pair à Pair, d'autres ressources, comme les réseaux locaux, les réseaux métropolitains et les réseaux longue distance, sont utilisées non plus seulement pour transporter des documents mais aussi pour mettre en relation les pairs d'un système. Des études sur le système Gnutella ont démontré l'utilisation excessive des réseaux par le système de mise en relation [Rip01]. D'autres études ont montré la vulnérabilité de systèmes comme FreeNet pour les cas d'attaques ciblées sur les nœuds présentant le plus de connexions [Hon01]. Le projet SETI@home a rencontré plusieurs problèmes de défaillance. Le plus important a été la coupure accidentelle d'une partie du réseau de l'Université de Berkeley lors de travaux. Le serveur est resté déconnecté de ses millions de participants pendant quelques heures. Un autre type de défaillance a été relevé lorsque plusieurs participants ont commencé à retourner des résultats faux. L'optimisation de l'utilisation des ressources, de l'extensibilité du système, du dimensionnement de l'infrastructure, de la tolérance aux défaillances et de la résistance aux attaques ciblées du système Pair à Pair passe par une compréhension fine des phénomènes intervenant dans ces plates-formes. La modélisation des systèmes, leur simulation (ou leur émulation) sont des outils indispensables dans cette perspective.

Le problème majeur que rencontre la modélisation, la simulation et l'émulation des systèmes Pair à Pair est le nombre et la complexité des ressources à considérer. La dimension typique d'un système Pair à Pair est de plusieurs centaines de milliers de ressources. Est-il nécessaire de construire des modèles, des simulateurs et des émulateurs capables de prendre en compte les interactions de plusieurs centaines de milliers de ressources pour faire émerger les problèmes et trouver des solutions intéressantes ? En attendant de connaître un élément de réponse à cette question, beaucoup de chercheurs se posent la question de la construction de tels modèles, simulateurs ou émulateurs. Les tous prochains mois verront très certainement fleurir nombre de ces outils. Il existe déjà des prototypes ad hoc, basés sur une reproduction très fidèle du comportement des réseaux sous-jacents. Ces simulateurs utilisent généralement NS (*Network Simulator*) et un générateur de topologies. D'autres simulateurs ne prennent pas en compte le réseau et s'intéressent à des phénomènes de seuils apparaissant dans les protocoles Pair à Pair, quelle que soit la topologie réelle. D'autres simulateurs s'intéressent à l'ordonnancement des tâches en prenant en compte des temps de calcul et de communication [Cas01].

▶ 6 – P2P et calcul

Les problèmes que nous avons évoqués jusqu'à ce stade concernent globalement les applications d'échange de documents et de calcul Pair à Pair. Le calcul Pair à Pair pose cependant des problèmes spécifiques que nous évoquons ici. Le projet XtremWeb du groupe Cluster et Grille du LRI et le projet d'ACI GRID CGP2P (www.lri.fr/~ fci/CGP2P.html) abordent précisément les problèmes liés au calcul Pair à Pair. Nous ne pouvons en présenter ici que quelques-uns. D'autres problèmes comme l'ordonnancement, la tolérance à la volatilité, les limites du domaine d'application, l'interface utilisateur et l'aide à la décision sont tout aussi importants et ardus. Ceci souligne la richesse scientifique de ce thème de recherche.

Sécurité des participants

La recherche de performances conduit à exécuter l'application sur les machines participantes dans le jeu d'instructions natif de la machine. Le problème de cette approche est la sécurité des participants. L'exécution d'un code binaire ne doit en aucun cas pouvoir conduire à une corruption de la machine ou de ses données.

Java apporte une solution de compromis à cette question. Sans compilation à la volée, le *byte code* Java est interprété par une machine virtuelle qui respecte la politique de sécurité par défaut ou imposée par le propriétaire de la machine. Pour améliorer les performances, certains environnements Java traduisent à la volée le *byte code* Java dans le format binaire de la machine d'exécution. Il est alors de la responsabilité du propriétaire de la machine de vérifier quelle politique de sécurité respecte ces environnements. L'un des problèmes liés à l'utilisation de Java réside dans la diversité des machines virtuelles existantes, de leur performance et de leur sécurité. En effet, il paraît difficile d'imposer à un participant une machine virtuelle précise. L'autre problème lié à l'utilisation de Java est que de nombreuses applications scientifiques ne sont pas écrites en Java.

Ces deux problèmes soulignent la nécessité de trouver une solution générale pour la protection des participants dans le cas de l'exécution de codes binaires résultats de la compilation de langages variés. La protection de la machine repose dans ce cas sur le système qui fonctionne sur cette machine. La problématique sous jacente est celle du « *sandboxing* » (enveloppement) de codes exécutables.

Il existe différentes voies pour parvenir à cet objectif. La première consiste à exécuter l'application en sécurisant le noyau du système d'exploitation de la machine d'exécution. Les systèmes Janus [Gol96] et Subterfugue reposent sur cette approche. Le système intercepte tous les appels systèmes réalisés par l'application, avant leur exécution. Ces appels sont analysés dynamiquement pour évaluer leur agressivité. Par exemple, on cherche à savoir si les fichiers visés sont bien dans l'espace de travail de l'application. Si ce n'est pas le cas l'application est stoppée.

Une autre approche, certainement la plus prometteuse, consiste à intégrer les dispositifs de sécurité dans les fonctions du système d'exploitation. C'est ce que proposent les « *Linux Security Modules* » qui intègrent des points de vérification avant les sections sensibles des fonctions systèmes. Ces points de vérification prennent la forme de branchements, pris ou non selon la politique de sécurité, qui déroutent l'exécution vers un module de sécurité que l'administrateur ajoute au noyau standard. Sans vérification, le coût des points de vérification est nul. Si un module de sécurité est ajouté, les performances des appels systèmes deviennent dépendants de la politique de sécurité et de son implémentation.

Il est important de noter que ces systèmes sont encore incomplètement maîtrisés.

Certification de résultats

L'une des problématiques complexes des systèmes de GRID (et en particulier des systèmes de calcul Pair à Pair) est la certification de résultats. Le problème provient de la possibilité qu'a un participant d'envoyer des résultats qui ne sont pas conformes à ceux prévus. Ces retours de résultats erronés peuvent être volontaires comme involontaires. Par exemple, dans le projet SETI@home, certains participants ont optimisé le calcul de la FFT qui constitue le cœur du calcul réalisé sur les machines participantes. Comme cette optimisation a été effectuée sans chercher à suivre des spécifications, le code optimisé ne répondait pas aux critères de qualité admissibles pour la FFT calculée sur les clients. Les codes optimisés ont donc envoyé des résultats faux.

Se prémunir contre des attaques volontaires ou involontaires sur les résultats de calcul est une question complexe. L'application ne peut pas compter sur les protocoles et la sécurisation des communications pour certifier un résultat. C'est le système ou l'application elle-même qui doivent offrir une parade à cette possibilité de corruption.

Il existe essentiellement deux approches possibles : l'approche système et l'approche application. L'approche système repose sur des techniques comme les exécutions redondantes avec vote et le test ponctuel. Une technique plus subtile qui, en fait, lie les deux précédentes, consiste à attribuer une crédibilité à chaque résultat et aux machines fournissant les résultats. Selon cette technique, les ressources acquièrent une crédibilité de plus en plus grande à mesure que le nombre de vérifications ponctuelles validées augmente. De même, les résultats acquièrent une crédibilité de plus en plus grande à mesure que le nombre de résultats concordants pour une même tâche augmente.

En complément de l'approche système, la certification de résultats peut être effectuée par l'application après réception du résultat. Plusieurs approches sont possibles suivant le type d'applications. Dans le cas où le code source de l'application n'est pas ouvert, le cryptage des résultats au moment du calcul par l'application et avant leur stockage ou leur transmission permet de résoudre le problème simple-

ment. Dans le cas général, où l'on admet que la machine effectuant le calcul dispose sous une forme ou sous une autre des sources de l'application et de la méthode de cryptage, le problème est beaucoup plus complexe et reste ouvert scientifiquement.

Applications parallèles

L'exécution d'applications parallèles sur Internet est tentante si l'on considère le nombre de ressources connectées. Plusieurs projets dont Javelin [Nea99] et Bayanihan [Sar01] ont examiné la possibilité de faire communiquer des applets Java et de construire des modèles de programmation par-dessus cette fonctionnalité. Notons que dans les deux cas, les échanges d'informations entre les applets passent systématiquement par un intermédiaire centralisé (le *broker*), ce qui n'est pas souhaitable dans le cas général.

Une des caractéristiques d'une organisation de type Calcul Global est l'extrême volatilité des PC participants. Cette caractéristique semble s'opposer aux principes qui sont à la base du modèle de programmation parallèle le plus utilisé pour les architectures parallèles à mémoire distribuée : le passage de messages explicites avec les fonctions d'envoi et réception classiques. En effet, dans un système de Calcul Global, il est difficile de garantir que les deux partenaires d'une communication seront encore actifs lorsque celle-ci sera exécutée.

Le principe de sites intermédiaires fonctionnant comme des mémoires associatives permet de concevoir un mécanisme de communication entre les machines participantes adapté à l'extrême volatilité des machines d'un système Pair à Pair. Une machine intermédiaire fiable joue le rôle de mémoire associative. Un élément de la mémoire associative est composé d'un identifiant et d'une donnée. Pour communiquer l'émetteur dépose un message dans la mémoire associative en utilisant un identifiant. Le récepteur s'adresse à la mémoire associative pour lire le message en utilisant le même identifiant. Ce principe permet aux tâches communicantes d'être totalement asynchrones (le récepteur pouvant lire un message envoyé par l'émetteur ayant depuis cessé de participer). Dans le principe, l'émetteur et le récepteur peuvent être remplacés par d'autres machines en cas de défaillance puisque leurs actions sur l'environnement extérieur se résument aux accès en écriture ou en lecture à une mémoire associative qui, elle, est fiable. Des machines de remplacement accèderaient directement à la mémoire associative pour continuer le calcul.

Découpler les machines communicantes et sauvegarder les communications en cours dans la mémoire associative ne suffit pas. Il faut aussi mettre en œuvre un mécanisme de reprise pour relancer globalement le calcul. Cette problématique est très complexe car les machines considérées ne sont connectées qu'à travers Internet ce qui rend en pratique déraisonnable la sauvegarde à distance de contexte d'exécution de plusieurs Mo. Sauvegarder localement (sur la machine elle-même) le contexte d'exécution aurait aussi peu d'intérêt puisque qu'il est peu probable qu'une machine se déconnectant du système se

Références :

[Bar01] D. Barkai, « Peer-to-Peer Computing », Intel press, 2001, octobre 2001.

[Cas01] H. Casanova, Al. Legrand, D. Zagorodnov, et F. Berman, « Heuristics for Scheduling Parameter Sweep Applications in Grid Environments », recueil d'articles du « 9th Heterogeneous Computing Workwhop (HCW'00) », IEEE Computer Society Press, mai 2000.

[Fos98] I. Foster et C. Kesselman, « The Grid : Blueprint for a new Computing Infrastreucture », Morgan-Kaufmann, 1998.

[Gol96] I. Goldberg, D. Wagner, R. Thomas, E. A. Brewer, « A Secure Environment for Untrusted Helper Applications », dans le recueil d'articles de Usenix'96, Usenix, http://www.usenix.org/publications/ordering, 1996.

[Hon01] T. Hong, « Performance », dans « Peer-to-Peer: harnessing the power of disruptive technologies », A. Oram éditeur, édition O'Reilly, mars 2001.

[Kan01] G. Kan, « Gnutella », dans « Peer-to-Peer: harnessing the power of disruptive technologies », A. Oram éditeur, édition O'Reilly, mars 2001.

reconnecte suffisamment rapidement pour que le calcul parallèle progresse à une vitesse raisonnable. Bref, le calcul parallèle dans les systèmes Pair à Pair pose des questions intéressantes et difficiles qui restent à l'heure actuelle des sujets de recherche.

▶ 7 – Conclusion

Les systèmes de calcul Pair à Pair représentent une formidable opportunité pour la globalisation et le partage des ressources de calcul et des données. Le projet XtremWeb a constitué une des premières tentatives dans ce sens en construisant une plate-forme verticale complète. Cette plate-forme est déjà utilisée en expérimentation dans différents sites. Des protocoles comme JXTA, COSM ou .NET permettent maintenant de construire ce type de systèmes mais n'offrent pas l'ensemble des mécanismes nécessaires pour construire XtremWeb, par exemple.

L'architecture du système lui-même reste un problème ouvert. Le choix entre architecture centralisée comme pour SETI@home, hiérarchisée comme pour FastTrack ou complètement distribuée comme pour Gnutella et FreeNet n'est pas établi. Une modélisation, une simulation ou une émulation des mécanismes fondamentaux (découverte de ressources, ordonnancement, routage) est nécessaire pour déterminer scientifiquement l'approche la plus pertinente. Ceci dit, les paramètres de ces modélisations et simulations sont encore à préciser. De nombreuses recherches sont aussi nécessaires sur la sécurité, la tolérance à la volatilité des nœuds, les applications parallèles, l'ordonnancement et les outils d'aide à la programmation. Le projet d'ACI GRID CGP2P (Calcul Global Pair à Pair) aborde ces questions et explore des pistes de solutions. Il faudra trouver des solutions efficaces à tous ces problèmes pour permettre, un jour, de globaliser plus complètement et à grande échelle les ressources informatiques, à partir de systèmes Pair à Pair.

[Lan01] A. Langley, « Frenet », dans « Peer-to-Peer: harnessing the power of disruptive technologies », A. Oram éditeur, édition O'Reilly, mars 2001.

[Nea99] M. O. Neary, S. P. Brydon, P. Kmiec, S. Rollins et P. Cappello, « Javelin + + : Scalability Issues in Global Computing », recueil d'articles de « ACM Java Grande 1999 Conference », ACM, 1999

[Rip01] M. Ripeanu, « Peer-to-Peer Architecture Case Study: Gnutella », recueil d'articles de « International Conference on Peer-to-peer Computing (P2P2001) », IEEE Computer Society Press, août 2001

[Sar01] L. F. G. Sarmenta. « Sabotage-Tolerance Mechanisms for Volunteer Computing Systems », recueil d'articles de « ACM/IEEE International Symposium on Cluster Computing and the Grid (CCGrid'01) », IEEE Computer Society Press, 2001

[Sma92] L. Smarr et C. E. Catlett, « Metacomputing », Communication of the ACM, Volume 35, N°6, pages 45-52, juin 1992.