INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet ReMaP

Régularité et parallélisme massif

Rhône-Alpes

THÈME 1A

Table des matières

1. Composition de l'équipe **1**

2. Présentation et objectifs généraux **2**

2.1.1. Évolution des architectures 2

2.1.2. Évolution des logiciels 2

2.1.3. Plates-formes de métacomputing (ou grilles) 3

2.1.4. Positionnement du projet 4

3. Fondements scientifiques **4**

3.1. Ordonnancement, algorithmique hétérogène 4

3.1.1. Nos défis de recherche : 5

3.2. Outils pour la mise en place de serveurs de calcul sur la grille 5

3.2.1. RPC pour la grille 5

3.2.2. Simulation de plates-formes hétérogènes et de grilles 6

3.2.3. Algorithmique parallèle mixte 6

3.2.4. Nos défis de recherche : 7

3.3. Algorithmique numérique parallèle pour le creux 8

3.3.1. Solveurs creux directs et méthode multifrontale. 8

3.3.2. La plate-forme logicielle MUMPS. 8

3.3.3. Nos défis de recherche : 9

4. Domaines d'application **9**

4.1.1. Contexte 9

4.1.2. Applications des serveurs de calcul 10

4.1.3. Applications des solveurs numériques creux parallèles 10

4.1.4. Application de l'algorithmique hétérogène 11

5. Logiciels **11**

5.1. DIET 11

5.1.1. Scilab$_{//}$ 11

5.1.2. DIET 11

5.1.3. FAST 11

5.2. MUMPS 11

5.2.1. Description. 12

5.2.2. Diffusion. 12

5.3. SimGrid 12

6. Résultats nouveaux **12**

6.1. Algorithmique et ordonnancement 12

6.2. Outils pour la mise en place de serveurs de calcul sur la grille 14

6.3. Algorithmique numérique parallèle 15

6.3.1. Extensions de la plate-forme logicielle MUMPS. 15

6.3.2. Ordonnancement sur grands nombres de processeurs et grappes de SMP. 15

6.3.3. Études de la topologie des arbres d'assemblage issus des techniques de renumérotation modernes. 16

6.3.4. Optimisation de la consommation mémoire dans l'approche multifrontale. 16

6.3.5. Expérimentation sur des problèmes industriels. 16

6.4. Stockage distribué et entrées/sorties parallèles 17

7. Contrats industriels **18**

7.1. Contrat LaRIA/CETMEF 18

8. Actions régionales, nationales et internationales **19**

8.1. Actions nationales 19

8.2. Relations bilatérales internationales 20
9. Diffusion des résultats 20
9.1. Animation de la communauté scientifique 20
9.1.1. Responsabilité d'animation 20
9.1.2. Comités de rédaction, de pilotage et de programme 20
9.2. Enseignement universitaire 21
9.2.1. Responsabilités d'organisation 21
9.2.2. Enseignement 21
9.3. Autres enseignements 21
9.4. Participation à des colloques, séminaires, invitations 21
10. Bibliographie 22

1. Composition de l'équipe

Le projet ReMaP est un projet commun CNRS/ENS Lyon/INRIA du Laboratoire de l'Informatique du Parallélisme (LIP), UMR CNRS/ENS Lyon/INRIA 5668. Ce projet est localisé à Lyon dans les locaux de l'École normale supérieure de Lyon.

Responsable scientifique

Frédéric Desprez [DR INRIA]

Assistantes de projet

Sylvie Boyer [Adjointe INRIA, 30% sur le projet]
Anne-Pascale Botonnet [Adjointe administrative ENS Lyon, 30% sur le projet]

Personnel Inria

Frédéric Desprez [DR]
Isabelle Guérin-Lassous [CR, départ le 22/03/02]
Jean-Yves L'Excellent [CR]
Tanguy Risset [CR, départ le 31/12/01]
Gil Utard [CR (détachement)]
Frédéric Vivien [CR, arrivée le 01/09/02]

Personnel CNRS

Alain Darte [CR, départ le 31/12/01]
Jean-Christophe Mignot [IR, départ le 01/09/02]
Loïc Prylli [CR, départ le 01/09/02]
Nicolas Schabanel [CR, départ le 01/03/02]

Personnel ENS Lyon

Eddy Caron [Maître de conférences, arrivée le 01/09/02]
Raymond Namyst [Maître de conférences, départ le 31/08/02]
Yves Robert [Professeur]

Ingénieurs experts

Ludovic Bertsch [Ingénieur contractuel INRIA, arrivée le 01/10/02]
Eddy Caron [Ingénieur contractuel INRIA, départ le 01/09/02]
Philippe Combes [Ingénieur contractuel INRIA]
Christophe Péra [Ingénieur contractuel ENS, arrivée le 01/09/02]

Chercheurs doctorants

Gabriel Antoniu [Allocataire MENRT, départ le 31/08/02]
Olivier Aumage [Allocataire MENRT, départ le 01/10/02]
Alice Bonhomme [Boursière Cifre *SAM*, départ le 31/12/01]
Vincent Boudet [Allocataire moniteur normalien, départ le 31/08/02]
Claude Chaudet [Allocataire MENRT, départ le 22/03/02]
Frédérique Chaussumier [Boursière Cifre *SAM*, départ le 31/12/01]
Vincent Danjean [Allocataire moniteur normalien, départ le 31/08/02]
Dominique Douthaut [Allocataire MENRT, départ le 22/03/02]
Abdou Guermouche [Allocataire MENRT]
Arnaud Legrand [Allocataire moniteur normalien]
Guillaume Mercier [Allocataire MENRT, départ le 31/08/02]
Martin Quinson [Allocataire MENRT]
Hélène Renard [Allocataire MENRT (ACI GRID), arrivée le 01/09/02]
Frédéric Suter [Allocataire MENRT, départ le 15/11/02]
Antoine Vernois [Allocataire MENRT (ACI GRID), arrivée le 01/09/02]

2. Présentation et objectifs généraux

Mots clés : *Environnement de programmation*, *bibliothèques*, *application répartie*, *algorithmique hétérogène*, *calcul sur la grille*.

Le parallélisme est maintenant solidement entré dans le monde industriel. Certes on n'a pas assisté à l'explosion promise au début des années 80 mais on retrouve des architectures parallèles dans tous les domaines, du niveau des processeurs jusqu'aux machines gigantesques du programme nucléaire américain ASCI, en passant par les réseaux de PCs. Le Téraflops a été obtenu en 1996 et on parle maintenant de Pétaflops.

Le domaine du calcul numérique, bien qu'il ne représente certainement qu'une infime portion du marché potentiel du parallélisme, est celui qui a reçu jusqu'à maintenant le plus d'attention. La simulation, qu'elle soit utilisée dans le cadre de la construction mécanique, de la mécanique des fluides, de la simulation d'explosions nucléaires, de la météo, ou dans bien d'autres domaines, utilise le parallélisme de manière intensive depuis plusieurs années. Les besoins en puissance de calcul et en taille mémoire sont cependant toujours plus importants et la technologie n'avance pas assez vite pour accéder à cette demande.

2.1.1. Évolution des architectures

Les caractéristiques des processeurs évoluent chaque année et l'on atteint aujourd'hui pour des stations de travail de taille moyenne des performances dignes de supercalculateurs des années quatre-vingt. Les processeurs actuels possèdent des unités fonctionnelles multiples et donc un parallélisme interne au processeur, des pipelines superscalaires, des hiérarchies mémoire importantes et des tailles de caches internes dignes des mémoires des stations de travail du passé.

On constate actuellement une évolution franche vers les grappes de machines SMP. Un bon exemple est la nouvelle machine SP3 d'IBM qui est constituée de nœuds puissants à 8 processeurs partageant une même mémoire reliée par un réseau rapide. Par ailleurs, on a également vu la disparition totale des processeurs spécialisés développés par les constructeurs au profit de processeurs « on the shelf » utilisés plutôt dans les PCs puissants ou les stations de travail que dans les machines parallèles. Les dernières versions de ces cartes peuvent abriter jusqu'à 8 processeurs pour donner des puissances jusqu'au Gigaflops.

Du côté des réseaux également, on a vu la prolifération de réseaux à bas prix et aux caractéristiques tout à fait impressionnantes comme les réseaux Myrinet, SCI, Fast Ethernet ou GigaEthernet. L'agglomération de ces processeurs et ces réseaux bon marché nous a conduit à construire des grappes de PCs comme on joue au Lego. Ces prix compétitifs ont permis la mise en place de plusieurs machines de petites tailles (jusqu'à 200 processeurs toutefois pour la grappe de l'UR Rhône-Alpes !) et ainsi d'effectuer les tests de diverses configurations matérielles et logicielles.

2.1.2. Évolution des logiciels

Les logiciels, comme dans tous les domaines de l'informatique mais encore plus sur les machines parallèles, ont eu une évolution plus lente comparée à celle du matériel.

La vitesse des processeurs progresse mais la complexité et la puissance des compilateurs également. Les recherches sur la compilation pour les architectures superscalaires ont donné de très bons résultats et les compilateurs actuels peuvent obtenir des performances proches des performances en crête sur des benchmarks qui donnaient auparavant de bien piètres résultats. Les compilateurs d'aujourd'hui restructurent le code de manière importante et savent générer des opérations duales pour tirer parti au mieux des processeurs à plusieurs unités d'exécution.

Si l'on regarde plus spécifiquement le domaine du calcul numérique parallèle, on peut dégager trois tendances.

Tout d'abord, la complexité des architectures parallèles a vite ralenti le développement de compilateurs paralléliseurs efficaces [101][106].

Les bibliothèques numériques rassemblant des noyaux de base ont eu un grand impact sur le développement des applications. Ces bibliothèques permettent de remplacer certains noyaux numériques par des versions optimisées et de pouvoir tester simplement de nouveaux algorithmes (comme par exemple dans le cas des solveurs itératifs).

Le passage de messages est donc resté le paradigme de programmation privilégié pour le développement d'applications sur machines parallèles à mémoire distribuée mais aussi sur machines à mémoire partagée. La complexité de sa programmation et de sa recherche d'erreur le rend souvent relativement inaccessible au béotien. Il reste toutefois le moyen le plus « facile » d'obtenir les meilleures performances et ceci pour un grand nombre d'applications. Le développement récent d'une interface standard comme MPI[1] [116][113] l'a propulsé au premier rang.

Il y a bien sûr d'autres outils et langages de parallélisation d'applications comme les threads, les langages data-parallèles basés sur C ou Fortran, les outils graphiques (Hence, Grade), les mémoires partagées virtuelles, la programmation par composants logiciels et même Java.

2.1.3. Plates-formes de métacomputing (ou grilles)

Après avoir tenté de mettre au point des machines de tailles toujours plus importantes (jusqu'à 9000 Pentiums pour la machine du programme américain ASCI), on agrège maintenant la puissance de nombreuses machines de tailles moins importantes pour former des grappes. Une nouvelle tendance consiste même à tenter d'utiliser des machines disponibles dans le monde entier, et ceci avec une relative transparence. Les plates-formes ainsi mises en place ont un potentiel immense car la plupart des machines utilisées interactivement sont le plus souvent sous-employées, et parce que l'on peut déporter les calculs vers la machine la plus adaptée pour les résoudre. Malheureusement, de telles solutions sont extrêmement difficiles à mettre en place de manière efficace. Le principe général est donc de tenter d'agréger un certain nombre de ressources disponibles comme un *méta-ordinateur*. À mi-chemin entre le concept architectural (on agrège des machines accessibles sur la toile) et le logiciel (on utilise la puissance des machines sans les connaître comme on utiliserait l'électricité), le métacomputing ou *grid computing* constituera certainement l'une des évolutions majeures de l'informatique de demain.

Au niveau de l'architecture matérielle, le métacomputing peut prendre plusieurs formes, mais il est toujours fortement hétérogène et hiérarchisé. Le métacomputing, dans sa forme la plus large, est l'agrégation de milliers de machines accessibles sur l'Internet pour résoudre des applications à grande échelle. Il s'agit déjà d'une réalité puisque des projets comme SETI@home permettent d'effectuer des calculs sur des PCs de par le monde. Il faut bien avouer que vue la bande passante actuelle du réseau, il s'agit plutôt d'applications « *embarrassingly parallel* » qui ne communiquent qu'épisodiquement avec une machine maître. Un autre exemple est certainement la grille TERAGRID qui interconnecte plusieurs sites aux États-Unis pour une puissance cumulée de 13.6 Téraflops par seconde en crête ! Toujours à grande échelle mais avec des performances en communications plus importantes, on peut citer le projet VTHD[2] du RNRT qui relie les centres de recherche de France Telecom, les unités de recherche INRIA et d'autres laboratoires par un réseau à 2.5 Gb/s. Sur une telle plate-forme, le réseau inter-centres est même plus rapide que les réseaux internes des grappes qu'il relie ! Plusieurs projets de ce type existent à travers le monde qui connectent généralement un petit ensemble de machines par un réseau rapide. Enfin, au niveau d'un laboratoire, on peut monter une plate-forme hétérogène en connectant plusieurs grappes entre elles par des réseaux plus ou moins rapides.

Le problème commun à toutes ces architectures n'est en fait pas le matériel (ces machines sont rarement isolées du monde) mais plutôt le logiciel (ce qui va naturellement du système à l'algorithmique). En effet, il y a peu de chance que ces machines soient totalement homogènes. Au mieux, on agrège des machines SMP de même type par un réseau rapide et au pire, on prend la puissance de calcul disponible sur la toile et, des réseaux aux processeurs en passant par les systèmes d'exploitation, l'ensemble de l'architecture est hétérogène.

Il y a deux challenges principaux pour le développement des plates-formes de métacomputing et pour une utilisation plus sérieuse que la recherche de vie extra-terrestre dans l'univers : le développement d'environnements rendant « transparente » l'utilisation de la grille et des résultats sur l'algorithmique des applications utilisant de telles plates-formes. Dans « environnement », on peut bien sûr placer le système d'exploitation mais aussi les langages, les bibliothèques et le middleware [84][87][96]. Actuellement, la plupart des environnements existants reprennent les approches choisies pour les machines parallèles « classiques »

[1] Message Passing Interface.
[2] Réseau à Vraiment Très Haut Débit.

en tentant de les adapter à la grille. Les résultats sont bien entendu inégaux. Les recherches autour de l'ordonnancement des tâches en milieux hétérogènes sont également de première importance.

2.1.4. Positionnement du projet

À sa création, l'objectif du projet *ReMaP* était de contribuer à l'élaboration des connaissances dans le domaine du calcul massivement parallèle régulier. Les évolutions de nos axes de recherches et les recrutements et départs divers et variés nous ont conduit à faire évoluer nos thématiques avec les axes suivants :

- algorithmique et ordonnancement pour les architectures hétérogènes et pour la grille,
- environnements pour la mise en place d'applications en mode client-serveur sur la grille,
- algorithmique pour la résolution de grands systèmes creux sur architectures hétérogènes.

Deux points forts du projet ont toujours été ses activités de transfert et ses collaborations internationales. Citons principalement les collaborations actuelles :

- collaboration avec Sun Labs Europe pour la mise en place d'environnements de type ASP (Application Service Provider) sur la grille,
- coopération avec l'Université de Californie à San Diego autour de l'ordonnancement sur la grille et le développement d'un simulateur d'ordonnanceurs sur architecture hétérogène.
- coopération avec l'université du Tennessee, Knoxville, dans le cadre de la bibliothèque d'algèbre linéaire parallèle *ScaLAPACK* et du logiciel NetSolve qui sont mis à la disposition de la communauté internationale.

Les mots-clés de *ReMaP* sont donc :

Algorithmique + Bibliothèques + Applications

sur architectures hétérogènes et sur la grille

3. Fondements scientifiques

3.1. Ordonnancement, algorithmique hétérogène

Participants : Vincent Boudet, Arnaud Legrand, Hélène Renard, Yves Robert, Frédéric Vivien.

L'ordonnancement statique de graphes de tâches est un problème important mais difficile pour l'algorithmique parallèle et distribuée. Traditionnellement, on s'intéresse à la minimisation du temps total d'exécution [111][89][93]. Pour être réaliste, il faut alors utiliser un modèle d'exécution qui prenne en compte les contraintes sur les ressources de calcul (nombre borné, vitesses différentes), et aussi sur les communications (contraintes d'émission et de réception sur un seul port à la fois, partage des liens, routage point à point ou pipeliné, etc.). Cette nouvelle approche est explorée par de nombreuses équipes [112][99][100][107][108]. Hélas, on aboutit inexorablement à des résultats de NP-complétude, ou pire, d'inapproximabilité : par exemple, il est impossible de garantir la performance des heuristiques polynomiales pour des architectures hiérarchiques en grappe de grappes.

Pour les plate-formes hétérogènes et distribuées, nous proposons d'abandonner le problème de la minimisation du temps d'exécution total. En effet, quand on traite des problèmes de grande taille, chercher à minimiser précisément le temps total d'exécution n'est pas vraiment nécessaire : l'obtention du régime permanent optimal suffit largement à assurer une bonne utilisation des ressources de calcul. On s'intéresse alors à l'optimalité asymptotique, en relaxant les contraintes du problème :

1. On néglige les phases d'initialisation et de terminaison.
2. On détermine le meilleur régime permanent, par exemple par résolution d'un programme linéaire en nombres rationnels.
3. On prouve enfin l'optimalité asymptotique d'un ordonnancement périodique construit à partir de la solution du programme linéaire.

Cette approche, originellement proposée par Bertsimas et Gamarnik [86] pour le routage de paquets, semble extrêmement prometteuse. Nous l'avons utilisée avec succès, en collaboration avec des chercheurs de San Diego, pour la distributions d'applications sur une plate-forme distribuée avec un schéma de type maître-esclaves. Chaque application est décrite sous la forme d'un graphe de tâches quelconque, de sorte que les différentes tâches composant l'application seront *a priori* exécutées sur des nœuds différents de la plate-forme. Nous supposons ici que toutes les applications ont le même graphe de tâches, c'est-à-dire qu'elles correspondent à l'exécution indépendante du même code sur des jeux de données différents. L'application s'exécute selon un schéma maître-esclaves : un processeur particulier de la plate-forme détient (ou produit) initialement tous les jeux de données à exécuter (par exemple sous forme de fichiers), et les distribue aux autres processeurs (les esclaves).

Il semble désespéré de chercher à minimiser le temps d'exécution pour un problème aussi difficile. Par contre, déterminer un algorithme de distribution et d'ordonnancement des tâches qui ait un bon comportement asymptotique quand le nombre de jeux de données à traiter devient grand semble un objectif plus raisonnable : on s'intéresse au comportement asymptotique d'un algorithme d'ordonnancement, et on introduit pour le résoudre un problème relaxé, dans lequel le traitement des tâches et les communications sont infiniment divisibles. Ce problème relaxé nous permet de déterminer le meilleur régime permanent, i.e. le débit maximal (fractionnaire) d'applications qui peuvent être traitées par unité de temps.

3.1.1. Nos défis de recherche :

3.1.1.1. Régime permanent pour des tâches avec fichiers

La mise en œuvre d'une application maître-esclave peut nécessiter la distribution de fichiers, éventuellement partagés entre plusieurs tâches. Nous avons déjà abordé ce problème dans un travail commun avec l'équipe GRAIL [88], mais nos résultats doivent être étendus au calcul du meilleur régime permanent pour un ensemble de fichiers distribués.

3.1.1.2. Macro-communications pipelinées

Dans un pool de ressources interconnectées, toutes les ressources ont des puissances de calcul différentes. De même, les vitesses des liens de communication diffèrent. Il peut y avoir plusieurs routes d'une machine à une autre. On s'intéresse aux primitives de macro-communications classiques (routage, diffusion, scatter, gather, polling) dans un tel environnement. Ici encore, la minimisation absolue du temps d'exécution pour une seule macro-communication n'est pas l'objectif : on aborde au contraire le problème d'une suite de macro-communications, dont on veut pipeliner l'exécution : on cherche alors le meilleur régime permanent.

3.1.1.3. Algorithmes multi-phases pour la distribution de tâches divisibles

Le modèle *divisible load* a reçu beaucoup d'attention, mais les seuls résultats d'optimalité connus s'appliquent au cas où on distribue le travail en une seule étape. Bien sûr, distribuer le travail en plusieurs phases serait une meilleure stratégie, pour bénéficier d'un meilleur recouvrement du calcul et des communications. L'intuition montre qu'il faut préférer des envois courts dans les premières étapes (pour que toutes les ressources démarrent l'exécution au plus vite), puis recourir à des messages plus longs dans les phases ultérieures. Les chercheurs de GRAIL ont proposé un algorithme où la taille des messages suit une progression géométrique [115], alors que nous avons conçu un algorithme périodique asymptotiquement optimal [85]. Nous allons comparer ces deux approches, et (tenter d') automatiser la phase de sélection des ressources.

3.2. Outils pour la mise en place de serveurs de calcul sur la grille

Participants : Ludovic Bertsch, Vincent Boudet, Eddy Caron, Philippe Combes, Frédéric Desprez, Arnaud Legrand, Christophe Pera, Martin Quinson, Frédéric Suter, Gil Utard, Antoine Vernois.

3.2.1. RPC pour la grille

Partant du constat que la majeure partie des applications gravitant autour de la grille sont de type numériques, les PSE (*Problem Solving Environments*) se sont tournés vers l'approche RPC [103]. Plusieurs environnements appelés NES (*Network Enabled Server*) ont implémenté ce paradigme comme NetSolve [83], Ninf [105], NEOS [95] et enfin DIET développé dans le projet. Le point commun entre ces environnements réside dans

les cinq éléments qui les composent : les clients, les serveurs, les bases de données (pour la gestion des ressources logicielles et hardware), les moniteurs et enfin les ordonnanceurs.

ReMaP a développé un NES basé sur une approche hiérarchique. Une telle structure est adaptée à la structure hiérarchique des réseaux et permet d'éviter le goulot d'étranglement provoqué par l'unicité de l'agent (comme dans NetSolve développé à l'Université du Tennessee à Knoxville). Conscient de ce problème, les PSE s'orientent vers des systèmes avec des agents distribués sur le réseau et chargés d'une partie de la plate-forme. Cependant afin de proposer des solutions efficaces et performantes il est indispensable d'intégrer des mécanismes d'ordonnancement distribués adaptés.

Les concepts de DIET (Distributed Interactive Engineering Toolbox) ont été développés dans un logiciel dont une première version a été présentée dans le cadre de SuperComputing 2002 (Baltimore). La version actuelle permet à un client d'obtenir le meilleur serveur pour le problème soumis. DIET est construit sur les SeD (démons pour la gestion des serveurs : *Server Daemons*). L'ordonnanceur est distribué le long de la hiérarchie sur les LA (*Local Agents*) et les MA (*Master Agents*). Cette approche a pu être également validée par simulation à l'aide de développements basés sur le logiciel SimGrid (développé dans l'équipe GRAIL de l'Université de Californie à San Diego).

Notre approche a été validée expérimentalement et par simulation, et les performances de la plate-forme sont fortement dépendantes du déploiement des agents maîtres et des agents locaux. Cependant, deux contraintes fortes sont à prendre en compte pour utiliser efficacement DIET : (i) l'architecture hiérarchique doit correspondre au réseau physique du système ce qui offre une diffusion des requêtes plus performante ; (ii) l'utilisation d'une hiérarchie d'agents peut avoir un impact sur les performances, surtout dans le cas d'un grand nombre de serveurs, en introduisant une gestion distribuée et parallèle des requêtes.

Nos développements sont également validés sur de nombreuses applications dans plusieurs domaines (géologie, chimie, physique, électronique, ...). Certaines d'entre elles sont décrites dans la section « Domaines d'applications ».

3.2.2. Simulation de plates-formes hétérogènes et de grilles

La plupart des résultats d'ordonnancement que l'on peut trouver dans la littérature sont obtenus grâce à des hypothèses assez restrictives et à des modèles assez simples. Les réseaux d'interconnexion sont souvent très simplistes (étoile, graphe complet, arbre, ...). Les ressources de calcul et de communication sont souvent supposées ne pas avoir de variation de performances et même si certaines heuristiques d'ordonnancement prennent en compte leur hétérogénéité, l'hypothèse que ces heuristiques sont capables d'obtenir des prédictions parfaites des différences de vitesses de leurs ressources est quasiment toujours faite. Il est nécessaire de faire ce type d'hypothèse simplificatrice pour comprendre certains phénomènes et réussir à concevoir des heuristiques efficaces. Cependant, ces hypothèses ne sont jamais vérifiées en pratique et le retour à la réalité est rarement effectué. L'impact de ces hypothèses simplificatrice (les variations de charge des ressource, l'impact de l'imprécision des performances, la contention qui peut survenir sur certains liens du réseau, ...) sur une application réelle est très rarement étudié.

La validation des résultats théoriques en ordonnancement pour les grilles s'effectue soit par des expériences grandeur nature soit par des simulations. Cependant, dans le cas d'une plate-forme de calcul hétérogène et distribuée, de telles expériences sont très délicates à mener en raison de l'instabilité latente de telles plates-formes. Il est en effet impossible de garantir que l'état d'une plate-forme de calcul qui n'est pas entièrement dédiée à l'expérimentation, va rester le même entre deux expériences, ce qui empêche donc toute comparaison rigoureuse. On utilise donc des simulations afin d'assurer la reproductibilité des expériences, toute la difficulté étant alors d'arriver à simuler un environnement réaliste. Une approche efficace consiste à effectuer des simulations utilisant des traces, c'est-à-dire utilisant des enregistrements de différents paramètres de la plate-forme pour obtenir un comportement réaliste.

3.2.3. Algorithmique parallèle mixte

Les algorithmes parallèles peuvent être séparés en deux catégories : les algorithmes utilisant un paradigme de programmation *data-parallèle* et ceux utilisant le *parallélisme de tâches*. Dans le premier modèle, les données utiles au calcul sont découpées selon la grille virtuelle de processeurs et ceux-ci effectuent les

mêmes opérations sur des données différentes ; il s'agit donc d'une approche SPMD. Dans le second modèle, le programme parallèle est découpé en tâches et ces tâches sont réparties sur les processeurs selon les dépendances qui les lient. Nous travaillons donc sur l'algorithmique parallèle mixte. Dans ce modèle, nos programmes sont constitués de tâches qui sont elles-mêmes data-parallèles. Pour cela, l'application est décrite par un graphe de tâches dont chacun des nœuds représente un calcul potentiellement parallèle et dont les arêtes identifient les relations de précédence et les éventuelles redistributions de données.

Nos travaux consistent à trouver le meilleur placement des tâches sur l'architecture cible ainsi que la meilleure allocation des processeurs. Pour cela, nous donnons des modèles précis des routines effectuant les calculs et des redistributions de données entre les groupes de processeurs.

3.2.4. Nos défis de recherche :

3.2.4.1. Simuler des plates-formes hétérogènes et des grilles.

SIMGRID est un outil de simulation développé conjointement avec Henri Casanova de l'Université de Californie à San Diego. Il permet l'évaluation d'algorithmes d'ordonnancement sur des plates-formes de métacomputing. Il est actuellement possible d'observer des architectures « réelles » et d'utiliser le résultat de ces observations au sein du simulateur pour évaluer les algorithmes sur des plate-formes réalistes. Ce type d'architecture est évidemment défini par un nombre de paramètres très important, ce qui rend la simulation extrêmement délicate. La détermination et la mesure de ces paramètres est donc au cœur de nos travaux actuels. Nos simulations doivent être réalistes, et nous devons donc utiliser des « images » des architectures réelles les plus réalistes possibles.

3.2.4.2. Prédiction de performances.

L'objectif de la bibliothèque FAST (*Fast's Agent System Timer*) est de fournir à un ordonnanceur les informations nécessaires à la fois en terme de besoin des routines (temps d'exécution et espace mémoire) et en terme de disponibilités des ressources de calcul et de communication. Nos travaux en la matière se dirigent maintenant vers la prédiction des besoins de routines parallèles en plus des routines séquentielles actuellement traitées. Pour cela, il semble nécessaire d'offrir des outils automatiques d'analyse structurelle du code source pour comprendre le schéma de communication, et tenir compte des éventuels recouvrement calcul/communication réalisés par ces routines.

Nous souhaiterions aussi traiter les routines, comme celles de calcul numérique creux, où les performances dépendent non seulement de la taille des données en entrée, mais aussi de leur contenu. Nous pensons qu'une façon d'y parvenir est de découper les routines en sous-étapes. Ainsi par exemple, la résolution d'un système linéaire creux se décompose en une phase d'analyse symbolique de la matrice (renumérotation, etc.), dont le temps d'exécution semble impossible à prédire, et en la résolution proprement dite, dont les performances ne dépendent pas du contenu des données, mais de caractéristiques calculées lors de l'analyse.

Enfin, d'autres travaux devront étendre la portée des outils de surveillance des ressources utilisées, par exemple pour mesurer efficacement les réseaux rapides comme SCI ou myrinet. Nous souhaiterions aussi étudier les prédictions à moyen ou même long terme grâce à des méthodes comme les prédictions d'ensembles utilisées en météorologie.

3.2.4.3. Découverte de topologie.

La connaissance de la topologie réseau est indispensable dans divers domaines, comme la simulation de plates-formes de calcul, la prédiction de communications *all to all* ou le placement automatique de serveurs. Cependant, les outils et méthodologies existantes nous semblent plus adaptés à l'administration des réseaux qu'à l'ordonnancement dans un contexte de Grid computing. Pour corriger ceci, nous pensons qu'il est nécessaire d'améliorer le passage à l'échelle des outils tout en mettant l'accent non pas sur l'interconnexion physique des éléments, mais sur les effets de la topologie sur les communications point-à-point.

3.2.4.4. Déploiement d'une plate-forme de type client-serveur sur la grille.

Notre architecture est déployée statiquement par les administrateurs des domaines en fonction du nombre de serveurs de calcul disponibles et du nombre potentiel de clients et de requêtes. Nous devons étudier l'ajout ou le retrait de nouveaux agents en fonction des scénarios d'utilisation et des modèles de calcul.

De plus, dans la version courante de DIET, les agents sont connectés de manière statique en fonction de leur localisation sur le réseau. Cependant, la charge de ce même réseau varie de manière significative durant l'exécution. Des routes moins chargées peuvent être trouvées afin de réduire le temps de propagation des requêtes. Une approche paire-à-paire peut être utilisée entre les agents en utilisant la surveillance du réseau et des heuristiques de placement de ces agents. Nous étudierons la faisabilité et l'intérêt d'utiliser ces technologies dans un cadre hiérarchique (mise en place d'un réseau pair-à-pair entre les différents éléments de l'architecture).

3.2.4.5. Heuristiques d'ordonnancement pour les NES.

Les environnements comme Ninf ou NetSolve utilisent des heuristiques d'ordonnancement basiques. Nous souhaitons mettre en place des heuristiques évoluées et adaptées à la hiérarchie et aux modèles de tâches envoyées par les clients (avec ou sans dépendances, grain fin ou gros grain). Nos travaux sur l'ordonnancement mixte auront bien évidement des implications pour l'ordonnancement de tâches data-parallèles sur les serveurs de calcul.

3.3. Algorithmique numérique parallèle pour le creux

Participants : Abdou Guermouche, Jean-Yves L'Excellent, Gil Utard.

3.3.1. Solveurs creux directs et méthode multifrontale.

La résolution de systèmes linéaires creux de grande taille se trouve au cœur de la plupart des techniques de simulation : la chimie moléculaire, le calcul de structures, la mécanique des fluides, ...L'importance et la variété des domaines d'application constituent la motivation principale dans la recherche de méthodes de résolution performantes des systèmes linéaires creux (symétriques ou non symétriques, réguliers ou irréguliers, ...). Ainsi, et afin d'accroître toujours la taille des simulations, il est important de pouvoir traiter ce type de problèmes le plus efficacement possible à la fois d'un point de vue temps de calcul et d'un point de vue utilisation mémoire. Pour ce faire, on utilise au mieux les caractéristiques des calculateurs parallèles actuels, dont l'utilisation est rendue nécessaire de part le volume important des calculs induits. Il est important aussi de rester à l'écoute des besoins en fonctionnalités des applications et des utilisateurs, afin d'offrir des solutions robustes pour de larges gammes de problèmes.

De part leurs performances et leur robustesse, les méthodes directes (basées sur la factorisation de Gauss) restent des méthodes de choix pour résoudre ce type de problèmes. Dans ce cadre, nous nous intéressons en particulier à l'approche multifrontale [91][92], pour des problèmes symétriques, symétriques définis positifs ou non symétriques, avec pivotage numérique pour assurer la stabilité, ce qui induit des structures de données dynamiques et imprévisibles de façon symbolique.

Cette méthode est basée sur un arbre d'élimination [102] qui résulte de la structure du graphe correspondant à la matrice creuse du problème à résoudre et de l'ordre d'élimination des variables. Cet arbre constitue le graphe de dépendance des calculs et est exploité pour définir les tâches qui peuvent être exécutées en parallèle. D'autre part la méthode multifrontale permet de se ramener pour chaque nœud de l'arbre à des factorisations partielles sur des matrices denses. Cette approche permet une bonne localité et une bonne utilisation des caches.

Afin de gérer le pivotage numérique et de garder une approche adaptable aux évolutions des différentes architectures des calculateurs parallèles existants, nous nous intéressons plus spécialement à des approches intrinsèquement dynamiques et asynchrones [82][81]. Les algorithmes retenus, outre leurs qualités numériques, s'appuient ainsi sur une approche originale basée sur la gestion dynamique et distribuée des tâches de calcul. Ils sont très intéressants d'un point de vue étude d'ordonnancement et parallélisme distribué. En effet les graphes de tâches associés sont très irréguliers et dynamiques.

3.3.2. La plate-forme logicielle MUMPS.

L'environnement logiciel MUMPS [104] est une bibliothèque pour la résolution de systèmes linéaires creux de grande taille sur machines à mémoire distribuée. Il est issu de longues années de collaborations multi-institutions. Il a débuté avec le projet européen PARASOL et son développement a été poursuivi ensuite, pour

y intégrer les avancées résultant de nos recherches dans le domaine. Cet environnement est d'autre part notre plate-forme d'expérimentation privilégiée pour valider de nouvelles idées et recherches. Plus de détails sont disponibles dans la partie 5.1.

3.3.3. Nos défis de recherche :

3.3.3.1. Optimisation mémoire et solveurs creux parallèles out-of-core.

Les solveurs creux directs, et en particulier l'approche multifrontale, exhibent une grosse consommation mémoire pour les problèmes 3D de grande taille. Ainsi, pour certaines classes de problèmes, seule une approche out-of-core pourra permettre de faire face aux défis des simulations, pour lesquels les volumes de calcul et la taille des problèmes rencontrés sont de plus en plus grands.

Dans la factorisation directe de matrices creuses par une méthode multifrontale, on peut imaginer deux approches : (i) la première consiste à stocker les facteurs systématiquement sur disque au fur et à mesure du calcul, et à minimiser la mémoire active ou mémoire incore, (ii) la seconde, nécessaire pour les problèmes très critiques, consistera à mettre aussi sur disque une partie de la mémoire active, nécessitant des techniques de *prefetch* appropriées, surtout dans un cadre parallèle où l'ordonnancement est dynamique.

3.3.3.2. Calcul creux sur plate-formes hétérogènes.

L'évolution des architectures des machines parallèles actuelles, et l'apparition de ressources de calcul de plus en plus hétérogènes dans un contexte metacomputing et grid fait que les applications existantes doivent s'adapter et prendre en compte l'accès de plus en plus dynamique aux ressources, les approches purement statiques s'adaptant difficilement. Ainsi il apparaît intéressant de pousser nos techniques d'ordonnancement pour la résolution de systèmes linéaires creux sur des machines du type grappes de SMP. Ensuite, nous souhaitons être capables d'utiliser des machines encore plus hétérogènes où les processeurs pourront être utilisés en mode multi-utilisateur, sachant que l'adaptation à la charge est théoriquement permise par nos types d'approches, moyennant certaines extensions. Cela nécessitera la prise en compte d'informations statiques et dynamiques sur la vitesse et la charge des processeurs, et l'utilisation d'informations statiques permettant elles aussi de guider les choix dynamiques pour l'affectation d'une partie des tâches de calcul.

3.3.3.3. Utiliser la grille pour démocratiser l'accès à ce type de solveurs.

Une autre utilisation intéressante de la grille est de permettre l'accès à nos méthodes de résolution de manière relativement transparente. Dans un environnement client-serveur, des utilisateurs distants pourront utiliser des ressources de calcul qui devront être présentes côté serveur. Ainsi, ces utilisateurs n'auraient pas besoin d'installer une série de logiciels compliqués mais pourraient avoir accès, par exemple directement depuis leur application, à des ressources de calcul et à des solveurs leur permettant de résoudre leurs problèmes. Ces perspectives rejoignent alors les objectifs de DIET, et leur mise en place en vrai grandeur permettrait dans le même temps de valoriser nos travaux de recherche et nos logiciels auprès des utilisateurs.

4. Domaines d'application

4.1.1. Contexte

Le parallélisme évolue et les domaines d'application se multiplient. Avec l'apparition de réseaux de stations de travail ou de piles de PC au bon rapport performance/prix, de nombreux utilisateurs se tournent vers des solutions parallèles. Outre les domaines d'application (on dépasse le traditionnel calcul scientifique pour aborder le secteur médical, bancaire, la sidérurgie, le textile, la publicité, la géographie, etc.), la nature des partenaires évolue (aux centres de recherche et développement des grands groupes publics ou privés s'ajoutent désormais les petites et moyennes entreprises). Ces nouveaux utilisateurs ont un besoin crucial d'environnements de programmation performants.

Par ailleurs, on constate que la popularité des environnements de type « grilles » est principalement due au nombre important d'applications de grandes tailles qui y sont déployées. Grâce à ces environnements, de nouvelles applications ont vu le jour.

Nous allons décrire rapidement des applications de nos travaux de recherche :

4.1.2. Applications des serveurs de calcul

Modèles numériques de terrain (LST ENS Lyon) Le LST (Laboratoire des Sciences de la Terre) dispose d'un programme parallèle pour le calcul de modèles numériques de terrains (MNT) à partir de deux vues stéréoscopiques. Le principe est de mettre en correspondance un maximum de points effectivement présents dans les deux images. Les différentes altitudes sont alors calculées en utilisant les propriétés géométriques des deux prises de vues ainsi que la disparité de position entre les pixels représentant le même point. L'utilisateur désireux d'effectuer un calcul de MNT va d'abord se connecter au serveur Web de la plate-forme pour communiquer la localisation sur le réseau des images stéréoscopiques et des fichiers de paramètres ainsi que l'endroit où renvoyer le résultat. Le résultat de cette application est une reconstruction pyramidale du maillage complet minimisant les erreurs à chaque niveau de résolution. Par ailleurs, on pourra envisager d'effectuer un calcul sur une zone précise en ayant recours à une base de donnée externe à l'application.

Simulation de circuits électroniques (IRCOM) Cette application concerne la simulation de composants électroniques. Le simulateur de circuit en équilibrage harmonique utilise le domaine fréquentiel pour décrire la partie (ou sous-circuit) linéaire du système à analyser, et n'utilise le domaine temporel que pour la description des éléments (ou sous-circuits) non-linéaires. Le temps de calcul croît de manière quasi linéaire avec le nombre de transistors du circuit. Étant donné le résultat de cette constatation, une première parallélisation « gros grain » peut être mise en place et ainsi diminuer le temps de calcul quasiment par le nombre de transistors du circuit pour autant que l'on dispose d'autant de processeurs que de transistors.

Dynamique moléculaire (Physique Lyon 1 et ENS, MAPLI Lyon 1) Cette application concerne la dynamique moléculaire parallèle de grands systèmes (laboratoire de physique de l'université de Lyon I, laboratoire de mathématiques appliquées de Lyon I, laboratoire de physique de l'ENS Lyon). Le but de l'application est de simuler des très grands systèmes (de l'ordre du million de particules) par dynamique moléculaire (résolution microscopique des équations du mouvement des atomes) en vue d'une comparaison avec les équations continues (Navier Stokes) dans des conditions où la validité de celles-ci n'est pas triviale. Ce traitement parallèle est basé sur un code de décomposition spatiale du système (testé et utilisé sur de nombreuses machines parallèles). L'utilisateur spécifie la configuration de départ et les paramètres d'interaction, et le programme intègre les équations du mouvement pendant un temps prédéterminé. Les données recueillies à la fin de calcul sont les trajectoires des particules ou les champs de vitesse à des instants sélectionnés.

Autres applications D'autres applications sont également en cours d'étude comme notamment une application de chimie (Hyper Surface d'Énergie Potentielle, SRSMC Nancy), la bioinformatique (recherche de protéines, ACI GRID GRIPPS, IBCP Lyon), etc.

4.1.3. Applications des solveurs numériques creux parallèles

Nos travaux sur les solveurs directs et en particulier sur la méthode multifrontale en environnement distribué permettent de résoudre des systèmes linéaires creux de grande taille, réels ou complexes. Ce type de systèmes est au cœur de la plupart des techniques de simulation : que l'on utilise des approches par éléments ou différences finis pour des équations aux dérivées partielles, ou que l'on s'intéresse à de l'optimisation numérique, que les problèmes soient linéaires ou non linéaires, on est presque toujours amené à résoudre des systèmes impliquant des matrices creuses. Les domaines d'applications sont donc très variés. Nous citons ici à titre d'exemple quelques unes des applications pour lesquelles notre logiciel MUMPS a été utilisé (voir aussi la section Logiciels) : analyse/calcul de structures (voitures, pièces de moteurs, bateaux, plate-formes offshore, ...), chimie moléculaire, dynamique moléculaire, dynamique des fluides, modélisation de l'océan, magnéto-hydrodynamique, discrétisation des équations de Navier-Stokes, modélisation de la structure du nerf optique, des tissus autour du cœur, simulation de semi-conducteurs, de réservoirs d'huile, de la houle dans des bassins marins, de l'effet Hall quantique, problèmes d'acoustique dans les avions civils, tomographie sismique, propagation d'ondes pour des problèmes de géophysique, de sismologie, d'optique, propagation de neutrons,

problèmes de contrôle optimal, électromagnétique, aéronautique, ...

4.1.4. Application de l'algorithmique hétérogène

D'une manière générale, la recherche d'ordonnancements en régime périodique, sur des plate-formes distribuées à grande échelle, a de nombreuses applications : citons le calcul collectif comme SETI@home [109] ou la factorisation des grands nombres [90], les problèmes de calcul distribué traités par des compagnies comme Entropia [94], et les prototypes logiciels de calcul maître-esclave hétérogène [110][98][97][114].

5. Logiciels

5.1. DIET

Mots clés : *serveurs de calcul*, *application service provider*, *évaluation de performances*.

Participants : Ludovic Bertsch, Eddy Caron, Philippe Combes, Frédéric Desprez [correspondant], Christophe Pera, Martin Quinson, Frédéric Suter, Antoine Vernois.

5.1.1. Scilab$_{//}$

Scilab est un logiciel de calcul scientifique de type Matlab développé par le projet Métalau à l'INRIA. Nous avons parallélisé ce logiciel dans le cadre de l'ARC INRIA OURAGAN.

Nos développements concernent la mise en place de serveurs de calcul efficaces accessibles depuis l'interface Scilab et l'interfaçage directement dans Scilab de bibliothèques numériques telles que ScaLAPACK ou PETSc. Pour plus d'informations voir le site Web : http://www.ens-lyon.fr/~desprez/OURAGAN/.

5.1.2. DIET

L'objectif du logiciel DIET (*Distributed Interactive Engineering Toolbox*) est de développer un ensemble d'outils pour construire des environnements à base de serveurs de calcul sur la grille. Il offre la possibilité de traiter les problèmes de grande taille en utilisant les ressources disponibles à travers Internet au même titre que Globus ou Legion. Les applications sont généralement basées sur le calcul numérique et l'utilisation de librairies comme les BLAS, LAPACK, ScaLAPACK ou PetSc est incontournable. L'intégration de telles librairies dans des langages haut niveau tel que Fortran ou C est loin d'être facile. De plus, la puissance de calcul et la mémoire nécessaire n'est pas à coup sûr disponible sur tous les serveurs. L'approche RPC semble être une bonne solution pour la mise en œuvre de PSE (*Problem Solving Environments*) sur la Grille. Plusieurs outils suivant cette approche existent comme Netsolve, NINF, NEOS ou RCS. DIET reprend ces objectifs mais offre de nouvelles fonctionnalités comme la hiérarchisation de la structure pour une meilleure extensibilité, la persistance des données, la tolérance aux pannes. Pour plus d'informations voir le site Web : http://graal.ens-lyon.fr/~diet/.

5.1.3. FAST

FAST (*Fast Agent's System Timers*) est un outil dynamique pour la prédiction de performances sur la grille. FAST est composé de plusieurs couches logicielles. Dans un premier temps, il utilise un logiciel de monitoring réseau et CPU pour prendre en compte dynamiquement les changements de ressources, comme la charge CPU ou la bande passante. FAST utilise NWS (*Network Weather Service*), un système distribué qui surveille périodiquement et prédit dynamiquement les performances de différents réseaux et de différentes ressources. Le module d'acquisition dynamique de FAST utilise et améliore NWS. FAST inclut également des routines pour modéliser le temps et l'espace nécessaire à chaque triplet de la forme (problème ; machine ; paramètres). Les données sont recueillies par des tests effectuées, lors de l'installation de FAST sur la machine, d'après un panel de tests représentatifs. Pour stocker ces statistiques, FAST utilise un arbre LDAP. Pour plus d'informations voir le site Web : http://www.ens-lyon.fr/~mquinson/fast.html.

5.2. MUMPS

Mots clés : *systèmes linéaires*, *matrices creuses*, *méthode multifrontale*, *calcul parallèle*, *calcul scientifique*, *mémoire distribuée*.

Participants : Jean-Yves L'Excellent [correspondant], Abdou Guermouche.

5.2.1. Description.

MUMPS (*MUltifrontal Massively Parallel Solver*) est un logiciel de résolution de systèmes linéaires creux par une méthode directe (méthode multifrontale) développé en collaboration avec l'IRIT et le CERFACS (Toulouse, France), et PARALLAB (Bergen, Norvège). C'est un code parallèle distribué (Fortran 90, MPI), unique par les performances atteintes et le nombre de fonctionnalités disponibles, parmi lesquelles :

- type de systèmes : symétriques définis positifs, symétriques généraux ou non symétriques,
- format d'entrée : matrice assemblée ou par éléments, centralisée sur un seul processeur ou distribuée,
- détermination du rang et calcul d'une base du noyau pour les problèmes singuliers,
- factorisation partielle avec calcul du complément de Schur,
- approche complètement asynchrone avec répartition dynamique de certaines tâches de calcul, recouvrement des calculs et des communications et pivotage numérique dynamique pour assurer la stabilité numérique,
- interface C ou Fortran 90,
- arithmétique réelle ou complexe.

5.2.2. Diffusion.

Le logiciel *MUMPS* est dans la continuité de développements effectués initialement au sein du projet européen PARASOL dont les résultats sont du domaine public. *MUMPS* est distribué gratuitement. De plus amples informations ainsi que la licence précise sont disponibles à l'adresse http://www.enseeiht.fr/apo/MUMPS/ ou http://www.ens-lyon.fr/~jylexcel/MUMPS/ (sites en mirroir).

Cette année, nous avons continué à mettre à disposition la version publique de *MUMPS* à un certain nombre d'utilisateurs du monde académique et du monde industriel. Nous comptons parmi les personnes qui ont utilisé *MUMPS* :

- des étudiants et des utilisateurs académiques du monde entier : Europe, US (45 .edu ou .gov), Japon, Corée, Inde, Argentine, Brésil, ...
- des développeurs de solveurs de type éléments finis,
- des industriels qui souhaitent expérimenter ou bien qui utilisent déjà *MUMPS*, comme par exemple Boeing, le CEA, Dassault, EADS, Shell ou les anciens partenaires de Parasol.

5.3. SimGrid

Mots clés : *simulation*, *ordonnancement*, *modélisation*.

Participant : Arnaud Legrand [correspondant].

SimGrid est un logiciel développé en collaboration avec Henri CASANOVA de l'Université de Californie, San Diego. *SimGrid* est un simulateur modulaire écrit en C permettant de simuler une application distribuée où les décisions d'ordonnancement peuvent être prises par différentes entités. La force de ce simulateur réside dans sa capacité à importer et simuler aisément des plate-formes réalistes (de la grille de métacomputing au réseau de stations de travail). Pour y parvenir, les simulations utilisent des traces, c'est-à-dire des enregistrements de différents paramètres d'une plate-forme pour obtenir un comportement réaliste.

Les sources et la documentation de *SimGrid* sont disponibles à l'adresse suivante

6. Résultats nouveaux

6.1. Algorithmique et ordonnancement

Participants : Vincent Boudet, Arnaud Legrand, Hélène Renard, Yves Robert, Frédéric Vivien.

Mots clés : *Algorithmique hétérogène*, *ordonnancement*, *parallélisme*.

Cette partie résume nos travaux récents concernant l'algorithmique. Nous développons de nouvelles méthodes d'ordonnancement et d'allocation en algèbre linéaire hétérogène et de nouvelles techniques de distribution des données pour des bibliothèques de calcul distribué.

Pour les plate-formes hétérogènes et distribuées, nous proposons donc d'abandonner le problème de la minimisation du temps d'exécution total, l'obtention du régime permanent optimal suffisant largement à assurer une bonne utilisation des ressources de calcul. On s'intéresse alors à l'optimalité asymptotique, en relaxant les contraintes du problème:

1. On néglige les phases d'initialisation et de terminaison.
2. On détermine le meilleur régime permanent.
3. On prouve enfin l'optimalité asymptotique d'un ordonnancement périodique construit à partir du meilleur régime permanent.

Cette approche originale semble extrêmement prometteuse. Nous l'avons utilisée avec succès dans les trois exemples que nous détaillons ci-dessous.

6.1.1. Paradigme maître-esclave pour des tâches indépendantes.

Nous nous sommes intéressés au problème de l'allocation d'un grand nombre de taches indépendantes et de taille identique sur des plate-formes de calcul hétérogènes. Nous utilisons des graphes non-orientés pour modéliser les plate-formes dont les ressources peuvent avoir des vitesses de calcul ou de communication différentes les unes des autres ainsi que diverses possibilités de recouvrement. Nous montrons comment déterminer le régime permanent optimal pour chaque processeur (c'est-à-dire la fraction de temps passée à calculer et celles passées à communiquer avec chacun de ses voisins). Ces résultats restent valables dans un cadre plus général où l'on autorise les cycles et les chemins multiples dans le graphe d'interconnexion, ainsi que l'existence de plusieurs maîtres.

Les arbres étant cependant plus facile à utiliser en pratique (il n'y a qu'un seul chemin du maître à un autre processeur), il est naturel de se demander comment extraire du réseau d'interconnexion le meilleur arbre couvrant, c'est-à-dire celui ayant le meilleur rendement en régime permanent. Nous démontrons que ce problème est NP-difficile et qu'il n'est pas approximable à un facteur constant près : il existe une famille de graphes d'interconnexion dont l'arbre couvrant optimal a un rendement arbitrairement éloigné de celui qu'une solution utilisant plusieurs chemins peut obtenir. Nous introduisons et comparons cependant un certain nombre d'heuristiques de faibles complexités et déterminant un arbre couvrant sous-optimal. Les meilleures heuristiques donnent d'excellents résultats dans la plupart des expériences.

6.1.2. Parallélisme mixte.

Nous avons voulu étendre les résultats précédents au cas de tâches avec dépendances: nous nous sommes intéressés à l'ordonnancement d'une application complexe sur une plate-forme de calcul hétérogène. L'application consiste en une suite de problèmes identiques et indépendants à résoudre. Chaque problème correspond à un graphe de tâches, avec contraintes de précédence. Un exemple typique de cette situation est l'exécution répétée d'un même algorithme sur des données différentes. La plate-forme de calcul est modélisée par un graphe non-orienté, où les ressources ont des vitesses de calcul et de communication différentes. Nous montrons comment déterminer le régime permanent optimal pour chaque processeur (c'est-à-dire la fraction de temps passée à calculer et celles passées à communiquer avec chacun de ses voisins).

Il faut souligner la difficulté de ce problème. Notre solution autorise que les différentes copies d'un même type de tâches soient exécutés par plusieurs processeurs, alors que chercher à allouer un processeur unique à chaque type de tâches rend le problème NP-difficile. À notre connaissance, c'est le premier résultat d'optimalité en parallélisme mixte pour une architecture hétérogène.

6.1.3. Algorithmes multi-étapes pour les tâches divisibles.

Nous avons comparé plusieurs algorithmes d'ordonnancement de tâches divisibles (*divisible workload*) sur une plate-forme hétérogène. Nous avons obtenu de nouveaux résultats d'optimalité pour les algorithmes à

une étape. Mais surtout, nous avons conçu un algorithme multi-étapes asymptotiquement optimal. Ce dernier algorithme effectue automatiquement la sélection des ressources à utiliser, tâche délicate généralement laissée à l'utilisateur. En raison de sa périodicité, il est plus facile à mettre en œuvre et plus robuste aux variations de charge des processeurs ou des liens de communications. D'un point de vue théorique, c'est, à notre connaissance, le premier résultat garanti sur les performances d'un algorithmes multi-étapes. D'un point de vue plus appliqué, les simulations que nous avons menées montrent que cet algorithme est meilleur que les autres algorithmes sur une large variété de plate-formes, tout particulièrement quand le rapport entre les communications et le calcul est élevé (le cas le plus délicat).

6.2. Outils pour la mise en place de serveurs de calcul sur la grille

Participants : Ludovic Bertsch, Vincent Boudet, Eddy Caron, Philippe Combes, Frédéric Desprez, Christophe Pera, Martin Quinson, Frédéric Suter, Gil Utard, Antoine Vernois.

Mots clés : *Calcul numérique*, *serveurs de calculs*, *évaluation de performances*, *grille de calcul*.

Cette partie résume nos travaux récents concernant la mise en place d'une plate-forme de type ASP (*Applications Service Provider*) DIET. Ces travaux ont bénéficié des recherches et des développements effectuées sur FAST dans le cadre de la prédiction de performance. À la fois client de FAST et composant algorithmique de DIET, le parallélisme mixte a également été au centre de nos recherches.

6.2.1. Développement de la plate-forme DIET.

Nos travaux autour de DIET ont consisté à le rendre utilisable. Cela est passé par une implémentation de toute la phase de connexion du client au serveur et de calcul sur ce dernier. Nous avons également modifié le typage de données DIET, qui constitue le langage de communication entre client et serveur. Il conditionne aussi toute l'évaluation des temps de transfert et de calcul. Nous avons amélioré l'interface du serveur, pour prendre en compte ce nouveau typage et surtout pour intégrer le lancement du calcul. Nous y avons inséré l'interfaçage avec FAST pour que l'évaluation des temps de calcul et de transfert devienne une réalité.

Côté client, nous avons développé l'interface en deux temps: adaptation au nouveau typage de données puis mise en conformité avec le standard naissant GridRPC proposé au Global Grid Forum.

Nous avons rendu l'utilisation de DIET plus conviviale, en lui donnant une configuration automatique et un manuel de l'utilisateur, ce qui lui faisait grandement défaut.

Enfin, nous avons dû grandement modifier l'implémentation interne de la hiérarchie et les communications entre entités pour améliorer les performances de la plate-forme.

Nous avons joint aux sources des exemples d'applications portées sur DIET et directement utilisables, comme les bibliothèques BLAS et SCALAPACK. Des applications plus complexes ont également été portées sur DIET pour la démonstration faite à SuperComputing 2002 (Baltimore): les modèles Numériques de Terrain et SciLab$_{//}$.

6.2.2. Évaluation de performances.

Cette année a été l'occasion de généraliser la bibliothèque FAST à des routines autres que celles de calcul matriciel, ainsi qu'à d'autres architectures matérielles. Le but était est de simplifier l'installation et l'usage de FAST sur toutes les machines cibles de DIET.

Dans le même temps, nous avons débuté des travaux pour offrir des informations sur la topologie réseau en plus des informations de bande passante et latence *end to end* déjà fournies par FAST.

Nous avons également développé une extension de la bibliothèque FAST pour la gestion des routines parallèles. Nous proposons ici de coupler les estimations données par FAST concernant les routines séquentielles et les disponibilités réseau avec des modèles de routines parallèles obtenus par analyse de code.

La version actuelle de l'extension traite uniquement certaines routines parallèles d'algèbre linéaire dense de la bibliothèque ScaLAPACK. Dans le cas de telles routines, l'étape de description consiste à déterminer quels sont les équivalents BLAS appelés, les paramètres d'appels de ces routines (taille des données, coefficients multiplicateurs, transpositions, etc.), les schémas de communication ainsi que les volumes de données

échangées. FAST pouvant estimer tous les équivalents séquentiels, des interrogations à FAST suffisent alors pour prédire leurs temps d'exécution.

Différentes expérimentations nous ont permis de valider la précision de cette extension que ce soit pour des formes de grilles ou des tailles de matrices différentes dans un cadre homogène.

Nous envisageons d'étendre ces travaux à l'ensemble des routines de la bibliothèque ScaLAPACK afin de fournir un outil d'évaluation de performances pour un noyau d'algèbre linéaire dense complet.

6.2.3. Algorithmique parallèle mixte.

Une validation de l'utilisation du parallélisme mixte dans le cadre du *grid computing* a tout d'abord été proposée. Pour cela, une étude théorique et expérimentale de différentes implantations mixtes des algorithmes de produit de matrices de Strassen et Winograd a été réalisée. De plus, un algorithme original d'ordonnancement utilisant le parallélisme mixte a été proposé. Cet algorithme, ciblant des plates-formes homogènes et/ou hétérogènes, utilise l'outil FAST pour acquérir une connaissance précise de chacune des tâches de l'application. De plus, chaque tâche se voit associée à une liste de plates-formes d'exécution potentielles. Les processus d'allocation et d'ordonnancement sont ensuite effectués simultanément pour gérer au mieux les coûts de redistribution inter-tâches.

6.3. Algorithmique numérique parallèle

Participants : Eddy Caron, Abdou Guermouche, Jean-Yves L'Excellent, Gil Utard.

Mots clés : *Calcul numérique*, *calcul scientifique*, *matrices creuses*, *méthode directe*, *parallélisme*, *out-of-core*.

Les activités dans ce thème sont divisées en deux parties, d'une part l'algorithmique numérique pour le creux (plate-forme logicielle MUMPS, aspects expérimentaux et validation sur problèmes réels, ordonnancements sur grappes de SMP, impact des renumérotations sur l'arbre de dépendance et optimisation mémoire) et d'autre part l'algorithmique numérique out-of-core (factorisation et inversion de matrices denses).

6.3.1. Extensions de la plate-forme logicielle MUMPS.

Des travaux logiciels incluant de la validation ont été nécessaires dans un premier temps à la fois pour servir de base (plate-forme d'expérimentation) à nos recherches futures, mais aussi pour rester ouvert au monde des utilisateurs, en particulier de l'industrie, et valoriser ainsi nos recherches sous forme de logiciel, en tenant alors compte des besoins des utilisateurs. Nous nous sommes ainsi intéressés aux extensions suivantes: version pour arithmétique complexe, version mono-processeur (utilisation simplifiée), interface C, et couplage avec des techniques modernes de renumérotation comme PORD (Université de Paderborn, Allemagne), METIS (Université du Minnesota), SCOTCH (projet ScAlApplix, Bordeaux), et AMF (Approximate Minimum Fill), en plus de AMD (Approximate Minimum Degree).

Ces travaux, auxquels nous allons intégrer certains des travaux de recherche mentionnés plus bas, ont donné lieu en décembre 2002 à une nouvelle version de MUMPS (MUMPS 4.2 bêta) qui est, tout comme la précédente version, mise à disposition gratuitement.

6.3.2. Ordonnancement sur grands nombres de processeurs et grappes de SMP.

Nous avons collaboré avec le CERFACS (C. Vömel et S. Pralet) et l'IRIT (P. Amestoy) sur les aspects ordonnancement pour plate-formes à grands nombres de processeurs et pour les plate-formes organisées en grappes de SMP. Dans l'ordonnancement que nous utilisons, les tâches de travail sont allouées, en priorité, aux processeurs les moins chargés. En donnant dynamiquement une pénalité aux processeurs les plus éloignés (c'est-à-dire sur un nœud distant), on a tendance à effectuer les tâches parallèles impliquant beaucoup de communications sur un même nœud et les performances sont améliorées de façon très significative. D'autre part, même sur des machines homogènes, l'approche complètement dynamique utilisée ne se comporte bien que jusqu'à une soixantaine de processeurs. Afin d'arriver à une mise à l'échelle jusqu'à 256 voire 512 processeurs, il est nécessaire d'injecter de l'information statique, qui limite les choix dynamiques locaux. Ces approches donnent de bons résultats, voire très bons dans le cas de problèmes réguliers. L'évolution actuelle des architectures matérielles avec des environnements de type metacomputing et grid font que les

aspects ordonnancement deviennent de plus en plus importants, et nous pensons qu'une approche mélangeant informations statiques et dynamiques est nécessaire pour le passage à l'échelle sur des architectures qui sont de plus en plus hétérogènes.

6.3.3. Études de la topologie des arbres d'assemblage issus des techniques de renumérotation modernes.

Les algorithmes de renumérotations sont des heuristiques de permutations de lignes (resp. colonnes) visant essentiellement la réduction du nombre d'opérations et la minimisation de la taille des facteurs après factorisation. Cependant, ces heuristiques modifient aussi la topologie du graphe de dépendance lors de la factorisation d'une matrice creuse. Dans le cas qui nous intéresse ce graphe se réduit à un arbre, et nous avons donc étudié l'impact des techniques de renumérotation sur sa topologie. Nous avons pu constater que selon la famille de méthodes utilisée, l'arbre pouvait avoir des caractéristiques très différentes impliquant aussi un comportement différent lors de la factorisation. Notre étude comparative des arbres générés a été faite en se basant sur des critères tels que la forme de l'arbre (largeur, profondeur, équilibre,...) et la taille des nœuds. Ces études ont été effectuées sur un grand nombre de problèmes issus d'applications variées en utilisant les techniques de renumérotation les plus modernes disponibles actuellement, telles que AMD, AMF, PORD, SCOTCH et METIS et elles ont été à la base des travaux sur la mémoire mentionnés dans le paragraphe ci-dessous.

6.3.4. Optimisation de la consommation mémoire dans l'approche multifrontale.

Afin de résoudre des problèmes de taille toujours croissante, les aspects consommation mémoire sont très importants et ne doivent pas être négligés par rapport aux aspects de performance pure ou de nombre d'opérations. Dans l'approche multifrontale, la mémoire est divisée en deux parties: une partie statique qui correspond au résultat de la factorisation (partie facteurs), et une partie dynamique, qui correspond à l'espace de travail du processus de factorisation. La taille de la mémoire dynamique dépend fortement de la topologie de l'arbre d'assemblage qui guide le processus de factorisation et la distribution des calculs sur les processeurs. Comme l'arbre résulte des techniques de renumérotations, nous avons montré comment celles-ci influencent la mémoire dynamique. Il apparaît que les approches qui donnent un arbre équilibré, large et donc avec beaucoup de parallélisme apparaissent grosses consommatrices de mémoire alors que celles qui donnent des arbres profonds et déséquilibrés le sont moins. Un compromis est donc nécessaire.

D'autre part, nous avons étudié l'influence de l'ordre de parcours de l'arbre d'élimination et avons proposé un algorithme pour minimiser l'espace mémoire d'une part dans le cadre de la factorisation in-core (minimisation de la mémoire totale), d'autre part dans le cadre de la factorisation out-of-core (minimisation de la mémoire active, considérant que les facteurs sont systématiquement stockés sur disque une fois calculés). Cet algorithme original a été expérimenté au sein de la plate-forme MUMPS ; la table ci-dessous donne les meilleurs gains en mémoire active que nous avons obtenus en utilisant notre algorithme pour différentes heuristiques de renumérotation:

Matrice	thermal	THREAD	af23560	xenon2	rma10
Heuristique de renumérotation	PORD	METIS	AMF	SCOTCH	AMD
Gain mémoire (%)	73.5	30.4	32.2	24.7	17.6

Enfin, nous nous sommes intéressés aux aspects mémoire dans le cas parallèle avec ordonnancement dynamique où notre algorithme ne s'applique plus que sur les sous-arbres traités en séquentiel. En parallèle, il apparaît que la mémoire active passe mal à l'échelle si l'ordonnancement est basé uniquement sur la charge de travail et non sur la mémoire. Par conséquent, nous nous intéressons actuellement à des approches d'ordonnancements qui prennent en compte la mémoire dans les décisions d'affectations des tâches au processeurs.

6.3.5. Expérimentation sur des problèmes industriels.

Les travaux mentionnés au paragraphe « Extension de la plate-forme logicielle MUMPS »3.3.2 nous ont permis de répondre à des besoins nouveaux d'utilisateurs. Citons en particulier EADS (Toulouse, France) qui

a financé un stagiaire (S. Pralet) au CERFACS avec qui nous collaborons. L'interface C et l'étude d'un solveur creux direct pour arithmétique complexe a permis d'expérimenter notre approche au sein de codes de calcul d'EADS et le résultat est que MUMPS est aujourd'hui opérationnel dans la chaîne de production de EADS.

Citons aussi le CETMEF (Centre d'Études Maritimes et Fluviales), qui s'intéresse à la simulation de la houle dans les grands bassins marins. Les travaux réalisés pour arithmétique complexe ont permis de montrer que notre approche était nettement plus efficace que les alternatives disponibles pour le CETMEF (travaux réalisés par O. Soyez du LaRIA et G.~Utard).

Parallèlement à cela, nous mettons à disposition du public une version incorporant nos derniers travaux et dernières recherches (environ une à deux demandes par semaine). Cela permet des discussions et retours d'utilisateurs, ce qui nous permet d'être très au fait du domaine applicatif et - dans une certaine mesure - d'orienter nos recherches en conséquence.

Enfin, notons que toutes nos recherches et études algorithmiques sont validées sur des problèmes industriels de grande taille, par exemple les collections de matrices creuses Rutherford-Boeing ou issues du projet européen PARASOL, aujourd'hui dans le domaine public.

6.3.5.1. Algorithmique numérique parallèle out-of-core.

Suite à une étude préliminaire des algorithmes parallèles *out-of-cores* de ScaLAPACK, nous avons défini un modèle de prédiction de performance pour les algorithmes parallèles *out-of-core*s de factorisation matricielle dans le cas dense. Grâce à ce modèle, nous avons démontré un résultat assez remarquable : en choisissant une distribution de la matrice adéquate, et en modifiant l'algorithme initial (introduction d'un schéma de recouvrement du temps d'entrées/sorties par du calcul), le temps d'exécution de l'algorithme *out-of-core* sur une machine disposant d'une mémoire proportionnelle à l'*ordre* de la matrice ($O(n)$) **est identique** au temps d'exécution de l'algorithme *in-core* disposant d'une mémoire proportionnelle à la *taille* de la matrice ($O(n^2)$). Par exemple, nous pouvons calculer la factorisation LU d'une matrice d'ordre 100~000 (en réels double précision) de 80 Gigaoctets en moins d'une journée avec 16 stations possédant seulement 4 Gigaoctets de mémoire au total. L'algorithme parallèle classique (*left-looking*) nécessiterait une mémoire totale de 80 Gigaoctets pour un même temps d'exécution. Ce travail clos notre étude sur le cas dense. Aujourd'hui nous travaillons sur des extensions *out-of-cores* pour le cas creux, dont le travail préliminaire précédemment décrit consiste à étudier le comportement mémoire des solveurs creux.

6.4. Stockage distribué et entrées/sorties parallèles

Participant : Gil Utard.

Mots clés : *I/O parallèle*, *MPI-IO*, *Systèmes de fichiers distribués*, *stockage pair à pair*.

Nous nous intéressons aux problèmes des entrées/sorties parallèles et au stockage distribué à différentes échelles. À l'échelle des clusters haut débit, nous étudions les mécanismes de base qui permettent d'obtenir des entrées/sorties parallèles à haute performance. Nous définissons et développons une nouvelle bibliothèque d'entrées/sorties appelée READ^2 où le flot de données et de contrôles des entrées/sorties n'interfère pas avec l'activité des nœuds hôtes et permet d'exploiter au mieux l'architecture. À l'autre bout de l'échelle, nous nous intéressons au problème du stockage dans les architectures de type pair à pair. Notre objectif est d'étudier dans quelles mesures ce type de système peut permettre de garantir la pérennité des données. Nous avons analysé les différents schémas de redondance ainsi que les mécanismes d'auto-réparations distribuées qui permettent de garantir la disponibilité des données. Ce travail fait partie de l'ACI GRID CGP2P et est effectué en collaboration avec le LaRIA (Amiens).

6.4.1. Support exécutif pour les entrées-sorties haute performance.

L'utilisation des grappes pour des applications traitant de grandes masses de données est une approche séduisante. Par exemple, les meilleures performances pour les tests de tri (Datamation, MinuteSort) sont aujourd'hui obtenues avec ce type d'architecture. Les dernières évolutions des périphériques d'entrées/sorties (disques, contrôleur, réseau) laissent espérer des améliorations sensibles des performances. La contrepartie de cette évolution est qu'une pression de plus en plus forte est mise sur le maillon faible des nœuds : les bus.

Nous avons donc étudié et développé une technique pour réduire cette pression et repousser les limites de performances que nous appelons READ^2 (*Remote Efficient Access to Distant Device*): nous employons les nouvelles capacités d'accès des cartes réseaux à l'espace d'adressage IO des bus d'entrées/sorties pour piloter directement les disques distants sans interférer avec le nœud hôte. Une grappe peut alors être considérée comme une architecture où les disques sont entièrement partagés par l'ensemble des nœuds. Dans une étude théorique préliminaire, nous avons estimé les principaux bénéfices que permet une telle approche : une meilleur utilisation des ressources d'entrées/sorties et une amélioration des performances globales des applications traitant de grands flots de données. Nous avons effectué une implémentation de cette technique sur un cluster à base de Myrinet et de disques à haut débit. Les premiers résultats ont conforté notre étude préliminaire. Actuellement, nous étudions l'intégration de READ^2 dans un système de fichiers distribué pour cluster et son impact sur les performances.

6.4.2. Stockage pair à pair.

Dans le cadre de l'ACI Grid CGP2P (*Calcul Global et Pair à Pair*), nous nous intéressons à la problématique du stockage pair à pair. Aujourd'hui il apparaît de nouveaux protocoles de type pair à pair pour la gestion de fichiers sur Internet. Parmi les plus connus on peut citer Napster, Freenet, Gnutella où les fichiers sont distribués sur l'ensemble des PCs. Ces systèmes sont des alternatives à l'approche client/serveur du WEB. Alors que Napster est centralisé en ce qui concerne l'indexation des données, Gnutella et Freenet sont entièrement distribués. En fait ces systèmes sont orientés essentiellement dans la mise en commun et dans la diffusion des données et des informations (surtout les fichiers mp3). Les objectifs avoués étant essentiellement celui de l'anonymat des sources d'informations (celui-ci est particulièrement poussé sur Freenet où il est impossible de connaître la machine d'origine de l'information), ce qui impose un surcoût non négligeable dans l'accès aux données.

Nous étudions actuellement les techniques pour garantir une meilleure pérennité des données dans les systèmes de fichier pair à pair. Des mécanismes de redondance et de reconstructions automatiques doivent être introduits. Nous avons analysé les propriétés de pérennité ainsi que les coûts induits de reconstruction pour les différentes techniques de redondance. Un prototype de stockage pair à pair intégrant ces mécanismes est en cours de développement. Il sera interopérable avec la plate-forme pair à pair XtremWeb du LRI.

7. Contrats industriels

7.1. Contrat LaRIA/CETMEF

Participants : Abou Guermouche, Jean-Yves L'Excellent, Gil Utard.

Dans le cadre des activités sur le calcul parallèle *out-of-core*, un contrat d'étude est en cours avec le CETMEF (Centre d'Études Techniques Maritimes et Fluviales) pour définir et intégrer un solveur out-of-core dans un code de simulation de la houle dans les grands bassins maritimes. Le code de houle REFONDE est un code d'agitation de houle qui résout l'équation de réfraction-diffraction de Berkhoff par une méthode de calcul d'éléments finis. Il peut prendre en compte des ouvrages réfléchissants. Il a été développé au sein du CETMEF.

Dans une modélisation par éléments finis, on se ramène généralement à la résolution d'un système linéaire creux. Certains problèmes traités par le CETMEF (par exemple l'aménagement du port du Havre), peuvent induire la résolution de systèmes creux de plusieurs dizaines de Gigaoctets. Malheureusement, dans certains cas, les systèmes à résoudre sont extrêmement coûteux en terme de temps de calcul et/ou d'espace mémoire. Le solveur employé par le CETMEF jusqu'à présent ne permet pas de traiter de tels problèmes.

Dans une étude préliminaire, nous avons étudié la structure des systèmes à résoudre dans Refonde à partir des matrices exemples fournies par le CETMEF. La caractéristique principale de ces systèmes est qu'ils sont composés de matrices creuses bandes irrégulières. Une étude plus fine des matrices considérées a montré une irrégularité très forte. Il est connu que l'ordre dans lequel un système creux est résolu peut réduire significativement le nombre de calculs et les besoins en mémoire. Nous avons donc étudié les différentes techniques d'ordonnancement des calculs (*ordering*) et leur influence sur les performances pour les systèmes

considérés. Lors de cette analyse, nous avons constaté que l'irrégularité des matrices considérées exhibe un fort degré de parallélisme à grain fin.

Il s'avère que la méthode de résolution directe la plus adaptée pour les matrices considérées semble être la méthode multifrontale. Nous avons donc couplé MUMPS avec le logiciel REFONDE. L'ensemble de ce travail à fait l'objet d'un stage de DEA (Olivier Soyez, LaRIA) financé par le CETMEF.

En termes de capacité de traitement, i.e. la taille maximale des problèmes qui peuvent être résolus, l'intégration d'un nouveau format de stockage des systèmes et l'utilisation de renumérotations ou d'*ordering*s adéquats, permettent d'envisager des problèmes qui étaient hors de portée de la version initiale de Refonde. Le parallélisme permet de repousser encore plus loin les limites. Pour de nouveaux systèmes qui dépasseraient ces nouvelles limites, nous comptons intégrer l'extension out-of-core de MUMPS en cours d'études (thèse d'Abdou Guermouche).

8. Actions régionales, nationales et internationales

8.1. Actions nationales

GDR ARP, thème iHPerf. Le projet ReMaP participe activement aux activités du thème iHPerf du GDR ARP (*Architectures, réseaux et systèmes, parallélisme*) sur l'algorithmique et les outils pour le parallélisme dans les applications régulières et irrégulières: E.~Caron, J.-Y.~L'Excellent, F.~Desprez, Y.~Robert, G.~Utard et F.~Vivien.

RNRT VTHD, 2 ans, 2000-2001. L.~Bertsch, E.~Caron, P.~Combes et F.~Desprez participent au projet RNTL VTHD rassemblant plusieurs équipes de recherche françaises autour de l'exploitation du réseau à « Vraiment Très Haut Débit » (2.5 Gb/s) reliant plusieurs centres INRIA et centres de recherche de France Telecom. Les recherches s'articulent autour de l'exploitation de ce réseau à plusieurs niveaux: protocoles, middlewares, applications. URL: http://www.vthd.org.

RNRT VTHD++, 2 ans, 2001-2003. L.~Bertsch, E.~Caron et F.~Desprez participent au projet RNTL VTHD++ qui fait suite au projet VTHD. Dans ce projet, notre but est de valider les divers protocoles de qualité de service sur le réseau en fonction des contraintes de nos applications. Nous souhaitons également étudier les aspects sécurité liés au déploiement d'une telle infrastructure logicielle ainsi que l'amélioration de l'extensibilité de la suite logicielle DIET.

RNTL GASP, 2 ans, 2001-2003 F.~Desprez est coordinateur du projet RNTL GASP (Grid Application Service Provider)[3] qui a été labelisé par ministère de l'industrie en 2001. Ce projet a pour but de développer une infrastructure logicielle permettant la mise en place simple et performante d'applications de type ASP sur une plate-forme de métacomputing. Les partenaires sont le projet ReMaP, le projet Résédas de l'INRIA Lorraine, l'équipe SDRP du LIFC, le laboratoire IRCOM, le laboratoire des Sciences de la terre de l'ENS Lyon et Sun Labs.

ACI Grid ASP, 3 ans, 2002-2005. F.~Desprez est le coordinateur de l'ACI Grid GridASP (Grid Application Service Provider) à laquelle participent également E.~Caron, J.-Y.~L'Excellent et G.~Utard. Il s'agit d'un projet pluridisciplinaire dont le but est de fournir des services de calcul à haute performance à des chercheurs d'autres discipline (physiciens, chimistes, mathématiciens appliqués, géologues, électroniciens, etc.).

ACI Grid CGP2P, 3 ans, 2002-2005 : *Calcul Global peer-to-peer*. Il s'agit d'un projet logiciel dont l'objectif est de définir une plate-forme de calcul global sur internet de type `SETI@home` pair à pair, i.e. que tous les participants ont accès aux ressources de calcul et de stockage. Il s'agit d'un projet multi-sites (LRI+LAL+ASCI (Orsay), LIFL (Lille), LaRIA (Amiens), IMAG (Grenoble) et LIP (Lyon)). Ce projet est décomposé en 5 sous-projets (applications et interfaces utilisateurs, sécurité des ressources et des applications, stockage et fouille de données, communications et ordonnancements, interopérabilité et vérification théorique). Le coordonnateur est Franck Cappello (LRI). Gil Utard est responsable du sous-projet stockage et fouille de données.

[3] http://www.ens-lyon.fr/~desprez/GASP/

ACI Grid Grid2, 3 ans, 2002-2005 Les membres de ReMaP participent au projet d'animation GRID2 (Groupe de Rencontres, d'Information et de Discussion sur la Globalisation des Ressources Informatiques et des Données) piloté par Jean-Louis Pazat. Yves Robert est responsable du thème « Modèles et algorithmique » qui fédère cinq équipes.

ACI Grid TLSE, 3 ans, 2002-2005. Nous participons à l'ACI GRID TLSE. Il s'agit d'un projet dont font aussi partie le CERFACS, l'IRIT, le LABRI et qui vise à mettre en place une base de données et un site expert sur les matrices creuses. En s'appuyant sur le logiciel DIET et sur les techniques de metacomputing développées par le projet REMAP, ce projet permettra à des utilisateurs de soumettre des requêtes d'expertise pour la résolution de systèmes linéaires creux, ces expertises s'appuyant sur des logiciels sur le creux développés par les partenaires de cette ACI et par la communauté matrices creuses.

ACI GRID GRIPPS, 2 ans, 2002-2004. Le projet GriPPS (Grid Protein Pattern Scanning) est un projet de recherche à l'interface de la Biologie et de l'Informatique. Les domaines de recherche concernés sont respectivement la génomique et les architectures informatiques distribuées. Les partenaires sont l'Institut de Biologie et Chimie des Protéines (IBCP), l'équipe RESO de l'INRIA et l'IN2P3.

Réseau d'excellence *CoreGrid* Yves Robert et Brigitte Plateau coordonnent la participation des équipes CNRS du département STIC dans le projet de réseau d'excellence *CoreGrid* en cours de montage.

8.2. Relations bilatérales internationales

Contrat NSF-INRIA, USA. Nous collaborons avec le projet Aladin de l'Irisa, le laboratoire LIMA-IRIT à Toulouse, le LaBRI à Bordeaux, l'Université du Minnesota, l'Université de l'Indiana, le Lawrence Berkeley laboratory. Ce projet a pour but de développer des préconditionneurs robustes et parallèles pour la résolution de grands sytèmes numériques. Nous sommes fournisseurs de méthodes directes pour la communauté méthodes itératives et nous nous intéressons plus particulièrement aux aspects algorithmiques de la parallélisation et aux aspects réduction mémoire de ces solveurs.

9. Diffusion des résultats

9.1. Animation de la communauté scientifique

9.1.1. Responsabilité d'animation

Département STIC du CNRS. Yves Robert est co-responsable du Réseau Thématique Pluridisciplinaire *Calcul à hautes performances et calcul réparti*.

Working Group ERCIM. Jean-Yves L'Excellent est membre du working group ERCIM intitulé "Application of numerical mathematics in science".

9.1.2. Comités de rédaction, de pilotage et de programme

Yves Robert est éditeur associé du journal *IEEE Transactions on Parallel and Distributed Systems*. Il est membre de l'Editorial Board de *International Journal of High Performance Computing Applications* (Sage Press).

Yves Robert a été membre des comités de programme suivants: Applied Informatics 2002 (Parallel and Distributed Computing and Networks), Vienne, Autriche; EuroPDP'03 (European Symposium on Parallel and Distributed Processing), Gênes, Italie; HCW'03 (IEEE Heterogeneous Computing Workshop), Nice; Euro PVM-MPI 2003, Venezia, Italie.

Yves Robert a été vice-président du comité de programme, responsable du thème *Algorithmes*, de IPDPS'03 (IEEE International Parallel and Distributed Processing Symposium), Nice . Enfin, il est président du thème *Scheduling and load balancing* de EuroPar'03, Klagenfurt, Autriche (2003)

Frédéric Desprez fait partie du comité de programme du journal *Parallel and Distributed Computing Practices* (http://www.cs.okstate.edu/~pdcp) dont l'éditeur en chef est M.~Paprzycki.
Frédéric Desprez fait partie du comité de programme du journal Computing Letters (COMPULETT) édité par Cambridge International Science Publishing Ltd.
Frédéric Desprez a été membre des comités de programmes suivants: EuroPAR 2002, ASIAN 2002, Euro PVM-MPI 2002, PMAA 2002, EuroPVM2003.

Gil Utard est membre du comité de pilotage de la conférence RenPar (Rencontre Francophone du Parallélisme).

9.2. Enseignement universitaire

9.2.1. Responsabilités d'organisation

Concours d'entrée à l'ENS Lyon. Yves Robert a été co-responsable de la nouvelle épreuve pratique "Algorithmique et programmtion" (épreuve sur machine) à l'oral du concours d'entrée informatique de l'ENS Lyon.

9.2.2. Enseignement

DEA d'informatique de Lyon. En 2002-2003, plusieurs membres du projet enseignent au DEA d'informatique fondamentale (DIF): G.~Utard (*Calcul Out-of-Core et Entrées/Sorties Parallèles* et Y.~Robert (*Algorithmique avancée*). URL: http://www.ens-lyon.fr/DIF/.

ENSEIRB, Bordeaux. Frédéric Desprez a donné des cours sur les environnements de type grilles de calcul dans l'option parallélisme de la 3ème année d'Ecole d'ingénieur de l'ENSEIRB.

9.3. Autres enseignements

Frédéric Desprez a été invité à donner un cours à Supelec dans le cadre de la formation continue sur les environnements pour la parallélisation d'applications numériques.

9.4. Participation à des colloques, séminaires, invitations

Yves Robert a été invité à donner deux séminaires aux Etats-Unis: University of Central Florida, Orlando, Avril 2002 et University of Illinois at Urbana-Champaign, Décembre 2002.

Yves Robert a donné les trois conférences invitées suivantes:

- Scheduling techniques for heterogeneous platforms, « Clusters and Grids for Parallel Scientific Computing », Faverges, Septembre 2002
- Static scheduling strategies for large-scale distributed systems, « Int. Symposium on Computer and Information Sciences », Orlando, Octobre 2002
- Static scheduling strategies for distributed systems, « Paralllel Matrix Algorithms and Applications 2002 », Neuchâtel, Novembre 2002

Frédéric Desprez a donné une conférence invitée à RenPAR 2002 (Hammamet, Tunisie) autour des serveurs de calcul sur la grille.

Eddy Caron et Frédéric Desprez ont donné un tutorial sur les systèmes de RPC sur la grille à l'Ecole GRID 2002 (Aussois, 2-6 Dec).

Frédéric Desprez a donné un séminaire sur le logiciel DIET à Grenoble (séminaires CIMENT) devant le PSMN (ENS Lyon).

Frédéric Desprez a organisé une journée GRID@INRIA à l'ENS de Lyon le 31 Janvier 2002 (http://www.ens-lyon.fr/~desprez/GRID-INRIA/). Cette journée a rassemblé 50 chercheurs de projets et d'action INRIA intéressés par les travaux autour des grilles de calcul. Une deuxième journée a été organisée à Nice en juillet 2002.

Frédéric Desprez et Franck Cappello ont organisé une journée de présentations autour des recherches sur les serveurs de calcul et sur les systèmes pair-à-pair les 4 et 5 juillet 2002. Il s'agissait de faire un point sur nos recherches et les éventuelles collaborations qui pourraient être mises en place.

Gil Utard a donné un séminaire intitulé « Les entrées/sorties parallèles : état de l'art » dans le cadre d'un groupe de travail ARISTOTE en juin 2002 (`http://acanthe.saclay.cea.fr/calculdistribue/`).

Gil Utard a donné un séminaire intitulé « Les systèmes de stockage pair à pair » dans le cadre des séminaire du projet SOR.

Gil Utard a donné un séminaire intitulé « La méthode multifrontale pour résoudre les systèmes creux » dans le cadre des séminaires du LAMFA (Laboratoire Amienois de Mathématiques Fondamentales et Appliquées).

10. Bibliographie

Bibliographie de référence

[1] R. B. (ED.). *High Performance Cluster Computing.* volume 2 : Programming and Applications, Prentice Hall, 1999, ISBN 0-13-013784-7.

[2] P. R. AMESTOY, I. S. DUFF, J. KOSTER, J.-Y. L'EXCELLENT. *A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling.* in « SIAM Journal on Matrix Analysis and Applications », numéro 1, volume 23, 2001, pages 15-41.

[3] P. R. AMESTOY, I. S. DUFF, J.-Y. L'EXCELLENT. *Multifrontal parallel distributed symmetric and unsymmetric solvers.* in « Comput. Methods Appl. Mech. Eng. », volume 184, 2000, pages 501-520.

[4] D. ARNOLD, S. AGRAWAL, S. BLACKFORD, J. DONGARRA, M. MILLER, K. SAGI, Z. SHI, S. VADHIYAR. *Users' Guide to NetSolve V1.4.* Computer Science Dept. Technical Report, numéro CS-01-467, University of Tennessee, Knoxville, TN, July, 2001, http://www.cs.utk.edu/netsolve/.

[5] O. AUMAGE, L. BOUGÉ, A. DENIS, J.-F. MÉHAUT, G. MERCIER, L. PRYLLI. *A Portable and Efficient Communication Library for High-Performance Cluster Computing.* in « Proceedings of the IEEE Int. Conf. on Cluster Computing (CLUSTER 2000) », 2000.

[6] M. BAKER. *Cluster Computing White Paper.* 2000.

[7] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Optimal algorithms for scheduling divisible workloads on heterogeneous systems.* rapport technique, numéro 2002-36, LIP, ENS Lyon, France, octobre, 2002.

[8] D. BERTSIMAS, D. GAMARNIK. *Asymptotically optimal algorithm for job shop scheduling and packet routing.* in « Journal of Algorithms », numéro 2, volume 33, 1999, pages 296-318.

[9] H. CASANOVA, A. LEGRAND, D. ZAGORODNOV, F. BERMAN. *Heuristics for scheduling parameter sweep applications in grid environments.* in « Ninth Heterogeneous Computing Workshop », IEEE Computer Society Press, pages 349-363, 2000.

[10] *Scheduling Theory and its Applications.* éditeurs P. CHRÉTIENNE, E. G. C. JR., J. K. LENSTRA, Z. LIU., John Wiley and Sons, 1995.

[11] J. COWIE, B. DODSON, R.-M. ELKENBRACHT-HUIZING, A. K. LENSTRA, P. L. MONTGOMERY, J. ZAYER. *A world wide number field sieve factoring record : on to 512 bits.* in « Advances in Cryptology - Asiacrypt '96 », série LNCS, volume 1163, Springer Verlag, éditeurs K. KIM, T. MATSUMOTO., pages 382-394, 1996.

[12] I. S. DUFF, J. K. REID. *The multifrontal solution of indefinite sparse symmetric linear systems.* in « ACM Transactions on Mathematical Software », volume 9, 1983, pages 302-325.

[13] I. S. DUFF, J. K. REID. *The multifrontal solution of unsymmetric sets of linear systems.* in « SIAM Journal on Scientific and Statistical Computing », volume 5, 1984, pages 633-641.

[14] H. EL-REWINI, H. H. ALI, T. G. LEWIS. *Task scheduling in multiprocessing systems.* in « Computer », numéro 12, volume 28, 1995, pages 27-37.

[15] ENTROPIA. URL : http://www.entropia.com.

[16] M. FERRIS, M. MESNIER, J. MORI. *NEOS and Condor : Solving Optimization Problems Over the Internet.* in « ACM Transaction on Mathematical Sofware », numéro 1, volume 26, 2000, pages 1-18, http://www-unix.mcs.anl.gov/metaneos/publications/index.html.

[17] I. FOSTER, C. K. (EDS.). *The Grid : Blueprint for a New Computing Infrastructure.* Morgan-Kaufmann, 1998.

[18] J. P. GOUX, S. KULKARNI, J. LINDEROTH, M. YODER. *An enabling framework for master-worker applications on the computational grid.* in « Ninth IEEE International Symposium on High Performance Distributed Computing (HPDC'00) », IEEE Computer Society Press, 2000.

[19] E. HEYMANN, M. A. SENAR, E. LUQUE, M. LIVNY. *Adaptive scheduling for master-worker applications on the computational grid.* in « Grid Computing - GRID 2000 », Springer-Verlag LNCS 1971, éditeurs R. BUYYA, M. BAKER., pages 214-227, 2000.

[20] L. HOLLERMANN, T. S. HSU, D. R. LOPEZ, K. VERTANEN. *Scheduling problems in a practical allocation model.* in « J. Combinatorial Optimization », numéro 2, volume 1, 1997, pages 129-149.

[21] T. S. HSU, J. C. LEE, D. R. LOPEZ, W. A. ROYCE. *Task allocation on a network of processors.* in « IEEE Trans. Computers », numéro 12, volume 49, 2000, pages 1339-1353.

[22] C. KOEBEL, D. LOVEMAN, R. SCHREIBER, G. S. JR., M. ZOSEL. *The High Performance Fortran Handbook.* The MIT Press, 1994, ISBN 0-262-11185-3.

[23] J. W. H. LIU. *The Role of Elimination Trees in Sparse Factorization.* in « SIAM Journal on Matrix Analysis and Applications », volume 11, 1990, pages 134-172.

[24] S. MATSUOKA, H. NAKADA, M. SATO, S. SEKIGUCHI. *Design Issues of Network Enabled Server Systems for the Grid.* http://www.eece.unm.edu/~dbader/grid/WhitePapers/satoshi.pdf, 2000, Grid Forum, Advanced Programming Models Working Group whitepaper.

[25] MUMPS. http://www.ens-lyon.fr/~jylexcel/MUMPS.

[26] H. NAKADA, M. SATO, S. SEKIGUCHI. *Design and Implementations of Ninf : towards a Global Computing Infrastructure.* in « Future Generation Computing Systems, Metacomputing Issue », numéro 5-6, volume 15, 1999, pages 649-658, http://ninf.apgrid.org/papers/papers.shtml.

[27] OPENMP. http://www.openmp.org/.

[28] J. M. ORDUNA, F. SILLA, J. DUATO. *A new task mapping technique for communication-aware scheduling strategies.* in « Workshop for Scheduling and Resource Management for Cluster Computing (ICPP'01) », IEEE Computer Society Press, éditeurs T. M. PINKSTON., pages 349-354, 2001.

[29] C. ROIG, A. RIPOLL, M. A. SENAR, F. GUIRADO, E. LUQUE. *Improving static scheduling using inter-task concurrency measures.* in « Workshop for Scheduling and Resource Management for Cluster Computing (ICPP'01) », IEEE Computer Society Press, éditeurs T. M. PINKSTON., pages 375-381, 2001.

[30] SETI. URL : http://setiathome.ssl.berkeley.edu.

[31] G. SHAO, F. BERMAN, R. WOLSKI. *Master/slave computing on the grid.* in « Heterogeneous Computing Workshop HCW'00 », IEEE Computer Society Press, 2000.

[32] B. A. SHIRAZI, A. R. HURSON, K. M. KAVI. *Scheduling and load balancing in parallel and distributed systems.* IEEE Computer Science Press, 1995.

[33] O. SINNEN, L. SOUSA. *Exploiting unused time-slots in list scheduling considering communication contention.* in « EuroPar'2001 Parallel Processing », Springer-Verlag LNCS 2150, éditeurs R. SAKELLARIOU, J. KEANE, J. GURD, L. FREEMAN., pages 166-170, 2001.

[34] M. SNIR, S. OTTO, S. HUSS-LEDERMAN, D. WALKER, J. DONGARRA. *MPI : The Complete Reference.* The MIT Press, 1996, http://www.netlib.org/utk/papers/mpi-book/mpi-book.html.

[35] J. B. WEISSMAN. *Scheduling multi-component applications in heterogeneous wide-area networks.* in « Heterogeneous Computing Workshop HCW'00 », IEEE Computer Society Press, 2000.

[36] Y. YANG, H. CASANOVA. *Multi-round algorithm for scheduling divisible workload applications : analysis and experimental evaluation.* rapport technique, numéro CS2002-0721, Dept. of Computer Science and Engineering, University of California, San Diego, 2002.

[37] MPI FORUM. *http://www.mpi-forum.org/.*

Livres et monographies

[38] éditeurs C. CÉRIN, G. UTARD., *Parallélisme et Systèmes Distribués.* TSI, 2002, Numéro spécial de TSI sur RenPar 2001..

[39] A. LEGRAND, Y. ROBERT. *Algorithmique Parallèle - Cours et exercices corrigés.* Dunod, 2002.

Thèses et habilitations à diriger des recherche

[40] F. SUTER. *Parallélisme mixte et prédiction de performances sur réseaux hétérogènes de machines parallèles.* thèse de doctorat, École normale supérieure de Lyon, novembre, 2002.

Articles et chapitres de livre

[41] P. R. AMESTOY, I. S. DUFF, J.-Y. L'EXCELLENT, X. S. LI. *Analysis and Comparison of Two General Sparse Solvers for Distributed Memory Computers.* in « ACM Transactions on Mathematical Software », numéro 4, volume 27, 2001, pages 388-421.

[42] O. BEAUMONT, V. BOUDET, A. LEGRAND, F. RASTELLO, Y. ROBERT. *Static Data Allocation and Load Balancing Techniques for Heterogeneous Systems.* éditeurs C. YUEN., in « Annual Review of Scalable Computing », volume 4, World Scientific, 2002, chapitre 1, pages 1-37.

[43] O. BEAUMONT, V. BOUDET, F. RASTELLO, Y. ROBERT. *Partitioning a square into rectangles : NP-completeness and approximation algorithms.* in « Algorithmica », volume 34, 2002, pages 217-239.

[44] O. BEAUMONT, A. LEGRAND, F. RASTELLO, Y. ROBERT. *Dense linear algebra kernels on heterogeneous platforms : Redistribution issues.* in « Parallel Computing », volume 28, 2002, pages 155-185.

[45] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Static scheduling strategies for heterogeneous systems.* in « Computing and Informatics », 2003, à paraître.

[46] E. CARON, F. DESPREZ, E. FLEURY, F. LOMBARD, J.-M. NICOD, M. QUINSON, F. SUTER. *Une approche hiérarchique des serveurs de calculs.* éditeurs F. BAUDE., in « Calcul réparti à grande échelle », Hermès Science Paris, 2002, ISBN 2-7462-0472-X.

[47] A. DARTE, R. SCHREIBER, B. R. RAU, F. VIVIEN. *Constructing and exploiting linear schedules with prescribed parallelism.* in « ACM Transactions on Design Automation of Electronic Systems », numéro 1, volume 7, janvier, 2002, pages 159-172.

[48] A. LEGRAND. *Équilibrage de charge statique pour noyaux d'algèbre linéaire sur plateforme hétérogène.* in « Technique et Science Informatique », numéro 5, volume 21, 2002, pages 711-734, Numéro spécial RenPar'13.

[49] M. QUINSON. *Un outil de prédiction dynamique de performances dans un environnement de metacomputing.* in « Technique et Science Informatique », numéro 5, volume 21, 2002, pages 685-710, Numéro spécial RenPar'13.

[50] F. RASTELLO, Y. ROBERT. *Automatic partitioning of parallel loops with parallelepiped-shaped tiles.* in « IEEE Trans. Parallel Distributed Systems », numéro 5, volume 13, 2002, pages 460-470.

Communications à des congrès, colloques, etc.

[51] C. BANINO, O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Scheduling strategies for master-slave tasking on heterogeneous processor grids.* in « PARA'02 : International Conference on Applied Parallel Computing »,

série LNCS, numéro 2367, Springer Verlag, pages 423-432, 2002.

[52] O. BEAUMONT, V. BOUDET, Y. ROBERT. *A realistic model and an efficient heuristic for scheduling with heterogeneous processors.* in « HCW'2002, the 11th Heterogeneous Computing Workshop », IEEE Computer Society Press, 2002.

[53] O. BEAUMONT, V. BOUDET, Y. ROBERT. *The iso-level scheduling heuristic for heterogeneous processors.* in « PDP'2002, 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing », IEEE Computer Society Press, 2002.

[54] O. BEAUMONT, L. CARTER, J. FERRANTE, A. LEGRAND, Y. ROBERT. *Bandwidth-centric allocation of independent tasks on heterogeneous platforms.* in « International Parallel and Distributed Processing Symposium IPDPS'2002 », IEEE Computer Society Press, 2002.

[55] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *A polynomial-time algorithm for allocating independent tasks on heterogeneous fork-graphs.* in « ISCIS XVII, Seventeenth International Symposium On Computer and Information Sciences », CRC Press, 2002.

[56] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Mixed task and data parallelism.* in « Parallel Matrix Algorithms and Applications », Université de Neuchâtel, 2002.

[57] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Static scheduling strategies for heterogeneous clusters.* in « Parallel Matrix Algorithms and Applications », Université de Neuchâtel, 2002.

[58] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Static scheduling strategies for heterogeneous systems.* in « ISCIS XVII, Seventeenth International Symposium On Computer and Information Sciences », CRC Press, 2002.

[59] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Scheduling strategies for mixed data and task parallelism on heterogeneous clusters and grids.* in « PDP'2003, 11th Euromicro Workshop on Parallel, Distributed and Network-based Processing », IEEE Computer Society Press, 2003.

[60] E. CARON, F. DESPREZ, F. LOMBARD, J.-M. NICOD, M. QUINSON, F. SUTER. *A Scalable Approach to Network Enabled Servers.* in « Proceedings of the 8th International Euro-Par Conference », série Lecture Notes in Computer Science, volume 2400, Springer-Verlag, éditeurs B. MONIEN, R. FELDMANN., pages 907-910, Paderborn, Germany, août, 2002.

[61] E. CARON, F. SUTER. *Extension parallèle d'un outil de prédiction dynamique de performances.* in « Quatorzièmes Rencontres Francophones du Parallélisme », pages 69-74, Hammamet, Tunisie, 10-13 avril, 2002.

[62] E. CARON, F. SUTER. *Parallel Extension of a Dynamic Performance Forecasting Tool.* in « Proceedings of the International Symposium on Parallel and Distributed Computing », pages 80-93, Iasi, Romania, Jul, 2002.

[63] E. CARON, G. UTARD. *Parallel Out-of-Core Matrix Inversion.* in « IPDPS'02. The 16th International Parallel and Distributed Processing Symposium », Fort Lauderdale, avril, 2002.

[64] P. CLAUSS, F. VIVIEN. *Schedules minimizing utility spans (to reduce memory requirements).* in « Parallel Matrix Algorithms and Applications », Université de Neuchâtel, 2002.

[65] P. COMBES, F. LOMBARD, M. QUINSON, F. SUTER. *A Scalable Approach to Network Enabled Servers.* in « Advances in Computing Science - ASIAN 2002. Internet Computing and Modeling, Grid Computing, Peer-to-Peer Computing, and Cluster Computing. Seventh Asian Computing Science Conference », série LNCS, volume 2550, Springer-Verlag, éditeurs A. JEAN-MARIE., pages 110-124, Hanoï, Vietnam, décembre, 2002.

[66] O. COZETTE, C. RANDRIAMARO, G. UTARD. *Improving Cluster IO Performance with Remote Efficient Access to Distant Devices.* in « Proc. Workshop on High-Speed Local Networks », IEEE, Tampa, Florida, US, novembre, 2002.

[67] A. GUERMOUCHE, J.-Y. L'EXCELLENT, G. UTARD. *Impact of sparse matrix reordering techniques on the memory usage of a parallel multifrontal solver.* in « Proceedings of the 2nd International workshop on Parallel Matrix Algorithms and Applications (PMMA'02) », 9-10 novembre, 2002.

[68] A. GUERMOUCHE, J.-Y. L'EXCELLENT, G. UTARD. *On the memory Usage of a Parallel Multifrontal Solver.* in « Proceedings of IPDPS'03 », 2003, à paraître.

[69] M. QUINSON. *Dynamic Performance Forecasting for Network-Enabled Servers in a Metacomputing Environment.* in « International Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems (PMEO-PDS'02) », 15-19 avril, 2002.

[70] F. VIVIEN. *On the Optimality of Feautrier's Scheduling Algorithm.* in « Proceedings of the 8th International Euro-Par Conference », série LNCS, volume 2400, Springer-Verlag, éditeurs B. MONIEN, R. FELDMANN., pages 299-308, Paderborn, RFA, août, 2002.

Rapports de recherche et publications internes

[71] P. R. AMESTOY, I. S. DUFF, J. KOSTER, J.-Y. L'EXCELLENT. *MUltifrontal Massively Parallel Solver (MUMPS Version 4.2 beta) Users' guide.* rapport technique, numéro TR2002-02, LIP, Lyon, France, December, 2002.

[72] P. R. AMESTOY, I. S. DUFF, J.-Y. L'EXCELLENT, X. S. LI. *Impact of the Implementation of MPI Point-to-Point Communications on the Performance of Two General Sparse Solvers.* rapport technique, numéro RR-4372, INRIA, 2002, http://www.inria.fr/rrrt/rr-4372.html, Submitted to *Parallel Computing*.

[73] V. BOUDET, F. DESPREZ, F. SUTER. *One-Step Algorithm for Mixed Data and Task Parallel Scheduling Without Data Replication.* rapport technique, numéro RR-4591, Institut National de Recherche en Informatique et en Automatique (INRIA), octobre, 2002, http://www.inria.fr/rrrt/rr-4591.html, Also available as LIP Research Report RR2002-34.

[74] E. CARON, P. COMBES, S. CONTASSOT-VIVIER, F. DESPREZ, F. LOMBARD, J.-M. NICOD, M. QUINSON, F. SUTER. *A Scalable Approach to Network Enabled Servers.* rapport technique, numéro RR-4501, Institut National de Recherche en Informatique et en Automatique (INRIA), juin, 2002, http://www.inria.fr/rrrt/rr-4501.html, Also available as LIP Research Report RR2002-21.

[75] E. CARON, F. SUTER. *Parallel Extension of a Dynamic Performance Forecasting Tool.* rapport technique, numéro RR-4470, Institut National de Recherche en Informatique et en Automatique (INRIA), juin, 2002, http://www.inria.fr/rrrt/rr-4470.html, Also available as LIP Research Report RR2002-19.

[76] E. CARON, G. UTARD. *Parallel Out-of-Core Matrix Inversion.*. rapport technique, numéro RR2002-04, Laboratoire de l'Informatique du Parallélisme (LIP), janvier, 2002, Submitted to *Parallel Computing*.

[77] F. DESPREZ, F. SUTER. *Impact of Mixed-Parallelism on Parallel Implementations of Strassen and Winograd Matrix Multiplication Algorithms.* rapport technique, numéro RR-4482, Institut National de Recherche en Informatique et en Automatique (INRIA), juin, 2002, http://www.inria.fr/rrrt/rr-4482.html, Also available as LIP Research Report RR2002-24.

[78] A. GUERMOUCHE, J.-Y. L'EXCELLENT, G. UTARD. *Analysis and Improvements of the Memory Usage in a Multifrontal Solver.* rapport technique, INRIA, 2003, http://www.inria.fr/rrrt/rr-4617.html, à paraître.

[79] A. LEGRAND, J. LEROUGE. *MetaSimGrid : Towards realistic scheduling simulation of distributed applications.* rapport technique, numéro 2002-28, LIP, juillet, 2002.

[80] F. VIVIEN, N. WICKER. *Minimal enclosing parallelepiped in 3D.* rapport technique, INRIA, 2002, http://www.inria.fr/rrrt/rr-4685.html, à paraître, also available as LIP Research Report RR2002-49..

Bibliographie générale

[81] P. R. AMESTOY, I. S. DUFF, J. KOSTER, J.-Y. L'EXCELLENT. *A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling.* in « SIAM Journal on Matrix Analysis and Applications », numéro 1, volume 23, 2001, pages 15-41.

[82] P. R. AMESTOY, I. S. DUFF, J.-Y. L'EXCELLENT. *Multifrontal parallel distributed symmetric and unsymmetric solvers.* in « Comput. Methods Appl. Mech. Eng. », volume 184, 2000, pages 501-520.

[83] D. ARNOLD, S. AGRAWAL, S. BLACKFORD, J. DONGARRA, M. MILLER, K. SAGI, Z. SHI, S. VADHIYAR. *Users' Guide to NetSolve V1.4.* Computer Science Dept. Technical Report, numéro CS-01-467, University of Tennessee, Knoxville, TN, July, 2001, http://www.cs.utk.edu/netsolve/.

[84] M. BAKER. *Cluster Computing White Paper.* 2000.

[85] O. BEAUMONT, A. LEGRAND, Y. ROBERT. *Optimal algorithms for scheduling divisible workloads on heterogeneous systems.* rapport technique, numéro 2002-36, LIP, ENS Lyon, France, octobre, 2002.

[86] D. BERTSIMAS, D. GAMARNIK. *Asymptotically optimal algorithm for job shop scheduling and packet routing.* in « Journal of Algorithms », numéro 2, volume 33, 1999, pages 296-318.

[87] éditeurs R. BUYYA., *High Performance Cluster Computing.* volume 2 : Programming and Applications, Prentice Hall, 1999, ISBN 0-13-013784-7.

[88] H. CASANOVA, A. LEGRAND, D. ZAGORODNOV, F. BERMAN. *Heuristics for scheduling parameter sweep*

applications in grid environments. in « Ninth Heterogeneous Computing Workshop », IEEE Computer Society Press, pages 349-363, 2000.

[89] *Scheduling Theory and its Applications.* éditeurs P. CHRÉTIENNE, E. G. C. JR., J. K. LENSTRA, Z. LIU., John Wiley and Sons, 1995.

[90] J. COWIE, B. DODSON, R.-M. ELKENBRACHT-HUIZING, A. K. LENSTRA, P. L. MONTGOMERY, J. ZAYER. *A world wide number field sieve factoring record : on to 512 bits.* in « Advances in Cryptology - Asiacrypt '96 », série LNCS, volume 1163, Springer Verlag, éditeurs K. KIM, T. MATSUMOTO., pages 382-394, 1996.

[91] I. S. DUFF, J. K. REID. *The multifrontal solution of indefinite sparse symmetric linear systems.* in « ACM Transactions on Mathematical Software », volume 9, 1983, pages 302-325.

[92] I. S. DUFF, J. K. REID. *The multifrontal solution of unsymmetric sets of linear systems.* in « SIAM Journal on Scientific and Statistical Computing », volume 5, 1984, pages 633-641.

[93] H. EL-REWINI, H. H. ALI, T. G. LEWIS. *Task scheduling in multiprocessing systems.* in « Computer », numéro 12, volume 28, 1995, pages 27-37.

[94] ENTROPIA. URL : http://www.entropia.com.

[95] M. FERRIS, M. MESNIER, J. MORI. *NEOS and Condor : Solving Optimization Problems Over the Internet.* in « ACM Transaction on Mathematical Sofware », numéro 1, volume 26, 2000, pages 1-18, http://www-unix.mcs.anl.gov/metaneos/publications/index.html.

[96] éditeurs I. FOSTER, C. KESSELMAN., *The Grid : Blueprint for a New Computing Infrastructure.* Morgan-Kaufmann, 1998.

[97] J. P. GOUX, S. KULKARNI, J. LINDEROTH, M. YODER. *An enabling framework for master-worker applications on the computational grid.* in « Ninth IEEE International Symposium on High Performance Distributed Computing (HPDC'00) », IEEE Computer Society Press, 2000.

[98] E. HEYMANN, M. A. SENAR, E. LUQUE, M. LIVNY. *Adaptive scheduling for master-worker applications on the computational grid.* in « Grid Computing - GRID 2000 », Springer-Verlag LNCS 1971, éditeurs R. BUYYA, M. BAKER., pages 214-227, 2000.

[99] L. HOLLERMANN, T. S. HSU, D. R. LOPEZ, K. VERTANEN. *Scheduling problems in a practical allocation model.* in « J. Combinatorial Optimization », numéro 2, volume 1, 1997, pages 129-149.

[100] T. S. HSU, J. C. LEE, D. R. LOPEZ, W. A. ROYCE. *Task allocation on a network of processors.* in « IEEE Trans. Computers », numéro 12, volume 49, 2000, pages 1339-1353.

[101] C. KOEBEL, D. LOVEMAN, R. SCHREIBER, G. STEELE, M. ZOSEL. *The High Performance Fortran Handbook.* The MIT Press, 1994, ISBN 0-262-11185-3.

[102] J. W. H. LIU. *The Role of Elimination Trees in Sparse Factorization.* in « SIAM Journal on Matrix Analysis and Applications », volume 11, 1990, pages 134-172.

[103] S. MATSUOKA, H. NAKADA, M. SATO, S. SEKIGUCHI. *Design Issues of Network Enabled Server Systems for the Grid.* http://www.eece.unm.edu/~dbader/grid/WhitePapers/satoshi.pdf, 2000, Grid Forum, Advanced Programming Models Working Group whitepaper.

[104] MUMPS. http://www.ens-lyon.fr/~jylexcel/MUMPS.

[105] H. NAKADA, M. SATO, S. SEKIGUCHI. *Design and Implementations of Ninf : towards a Global Computing Infrastructure.* in « Future Generation Computing Systems, Metacomputing Issue », numéro 5-6, volume 15, 1999, pages 649-658, http://ninf.apgrid.org/papers/papers.shtml.

[106] OPENMP. http://www.openmp.org/.

[107] J. M. ORDUNA, F. SILLA, J. DUATO. *A new task mapping technique for communication-aware scheduling strategies.* in « Workshop for Scheduling and Resource Management for Cluster Computing (ICPP'01) », IEEE Computer Society Press, éditeurs T. M. PINKSTON., pages 349-354, 2001.

[108] C. ROIG, A. RIPOLL, M. A. SENAR, F. GUIRADO, E. LUQUE. *Improving static scheduling using inter-task concurrency measures.* in « Workshop for Scheduling and Resource Management for Cluster Computing (ICPP'01) », IEEE Computer Society Press, éditeurs T. M. PINKSTON., pages 375-381, 2001.

[109] SETI. URL : http://setiathome.ssl.berkeley.edu.

[110] G. SHAO, F. BERMAN, R. WOLSKI. *Master/slave computing on the grid.* in « Heterogeneous Computing Workshop HCW'00 », IEEE Computer Society Press, 2000.

[111] B. A. SHIRAZI, A. R. HURSON, K. M. KAVI. *Scheduling and load balancing in parallel and distributed systems.* IEEE Computer Science Press, 1995.

[112] O. SINNEN, L. SOUSA. *Exploiting unused time-slots in list scheduling considering communication contention.* in « EuroPar'2001 Parallel Processing », Springer-Verlag LNCS 2150, éditeurs R. SAKELLARIOU, J. KEANE, J. GURD, L. FREEMAN., pages 166-170, 2001.

[113] M. SNIR, S. OTTO, S. HUSS-LEDERMAN, D. WALKER, J. DONGARRA. *MPI : The Complete Reference.* The MIT Press, 1996, http://www.netlib.org/utk/papers/mpi-book/mpi-book.html.

[114] J. B. WEISSMAN. *Scheduling multi-component applications in heterogeneous wide-area networks.* in « Heterogeneous Computing Workshop HCW'00 », IEEE Computer Society Press, 2000.

[115] Y. YANG, H. CASANOVA. *Multi-round algorithm for scheduling divisible workload applications : analysis and experimental evaluation.* rapport technique, numéro CS2002-0721, Dept. of Computer Science and Engineering, University of California, San Diego, 2002.

[116] MPI FORUM. *http://www.mpi-forum.org/.*