Un « Beowulf » par des physiciens

Choix, problèmes, performances

Florent Calvayrac [*] — **Yvan Labaye** [*] — **Jean-Christophe Gimel** [**]

[*] *Florent.Calvayrac@univ-lemans.fr*
Laboratoire de Physique de l'État Condensé - UMR-CNRS 6087
Faculté des Sciences
Université du Maine
F-72085 Le Mans Cedex 9
[**] *UMR 5626 (même adresse)*

RÉSUMÉ. *Une équipe de physico-chimistes décrit l'installation d'un calculateur parallèle de type Beowulf (Linux sur PC, réseau Fast Ethernet en agrégation de canaux – « Channel Bonding » – 70 processeurs, 18 Go de mémoire vive totale, 240 Go de disque), toute l'intégration logicielle étant faite par les futurs utilisateurs, en utilisant le plus possible d'outils du domaine public, pour un coût total de 700 000 FF fin 1999. Les choix, les problèmes rencontrés, ainsi que les performances (ScaLapack et de logiciels spécifiques de modélisation) sont discutés.*

ABSTRACT. *A team of physicists and physico-chemists describes the installation of a Beowulf-class parallel computer (Linux on PCs, Fast Ethernet Channel Bonding type interconnect, 70 processors, 18 Go total RAM, 240 Go disk space), all the software installation being done by the future users, using as much free (or opensource) software as possible, for a total cost of FF 700 000 end of 1999. Choices, encontered problems, as well as performance (ScaLapack and custom physics codes) are discussed.*

MOTS-CLÉS : *parallélisme, Beowulf, grappe, MPI, PVM, DFT, Heisenberg, Linux, Hoshen- Kopelman, agrégation, Scalapack, Linpack*

KEYWORDS: *parallelism, Beowulf, cluster, MPI, PVM, DFT, Heisenberg, Linux, Hoshen- Kopelman, aggregation, channel bonding, Scalapack, Linpack*

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

1. Introduction

La majorité des équipes de recherche, quelle que soit la discipline, sont désormais en train de ranger au nombre de leurs outils, à côté de la théorie et de l'expérience, la simulation numérique. Les chercheurs et ingénieurs français bénéficient certes d'un atout majeur dans ce domaine grâce aux centres nationaux et régionaux de calcul, tels que l'IDRIS ou le CINES. Ces derniers mettent à leur disposition des ressources permettant d'aborder des systèmes de taille considérable, grâce à des super-ordinateurs parallèles comme le T3E ou le SP3.

Il émerge cependant le besoin de calculateurs de puissance intermédiaire entre ces gros équipements et l'ordinateur personnel. Un ordinateur de taille moyenne permet de développer et de mettre au point de façon indépendante de nouveaux programmes, et de ne réserver aux super-ordinateurs nationaux que les calculs réellement inabordables sur d'autres machines.

Afin de répondre à de nouveaux besoins en simulation numérique intensive, deux unités de l'université du Maine au Mans, le laboratoire de Physique de l'État Condensé (PEC) et le laboratoire Polymères, Colloïdes et Interfaces, ont décidé de s'équiper en commun d'un calculateur parallèle de taille moyenne. La gestion en est assurée par les utilisateurs. Ainsi, pour un coût relativement modéré par rapport à des solutions « clefs en main » il a été possible d'installer un calculateur réellement polyvalent, pouvant servir à l'enseignement et à la formation aussi bien qu'à la recherche.

Dans cet article, nous décrivons et tentons de justifier nos choix matériels et logiciels, et discutons des performances obtenues avec ScaLapack comme avec certains programmes parallèles de simulation numérique.

2. Choix de l'architecture

Les ordinateurs traditionnels, quel que soit leur prix (déterminé par exemple par la quantité de mémoire disponible ou la qualité des entrées/sorties), n'offrent pas une puissance de calcul très élevée en termes de millions d'opérations en virgule flottante par secondes (MFlops). Les simulations numériques intensives requièrent cependant de très bonnes performances dans ce domaine.

Le seul remède à ce problème semble venir de la parallélisation des calculs, en distribuant des opérations non interdépendantes sur différentes unités élémentaires. Différentes approches sont alors possibles : sur les ordinateurs de type vectoriel (CRAY J94 par exemple), le « grain » du parallélisme est de l'ordre de l'opération élémentaire ; sur les ordinateurs de type massivement parallèle, le grain peut être plus gros. Sur ces dernières machines, des processeurs de type courant travaillent simultanément, et mettent en commun les résultats intermédiaires soit par un partage de la mémoire (ORIGIN 2000 de Silicon Graphics par exemple), soit par un réseau de communications interprocesseurs. Le CRAY T3E de l'IDRIS et l'IBM SP3 du CINES appar-

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

tiennent à cette dernière catégorie. Le T3E dispose par exemple de 256 processeurs Alpha.

Il est à notre avis inapproprié d'envoyer sur de tels ordinateurs, de coût de l'ordre de plusieurs MF, des calculs longs, certes accélérables par la parallélisation, mais dont la nature séquentielle et les besoins intermédiaires en mémoire (moins de 1 Go) font qu'il est peu pertinent de les faire tourner sur plus de 16 ou 32 processeurs.

Cependant, la baisse très rapide des coûts du matériel informatique, accompagnée d'un progrès rapide de la puissance des processeurs grand public, phénomènes rendus possibles par les importantes économies d'échelle provoquées par l'explosion du marché du multimédia depuis quelques années, rend possible l'assemblage d'une machine parallèle disposant justement de 32 ou 64 processeurs donnant au total une puissance supérieure à un super-ordinateur de 1994 pour un coût de l'ordre du million de francs.

Plusieurs universités ou laboratoires de par le monde ont eu recours à cette solution ces dernières années. Il existe par exemple au LANL de Los Alamos, au Goddard Flight Center, entre autres, des réalisations regroupées sous le nom de Beowulf, d'après le nom de la première machine ainsi assemblée à partir d'éléments dits « off the shelf » (pris directement sur les étagères des distributeurs grand public). On peut en trouver une liste sur le « Cluster Top 500 ». En France, l'Action transversale thématique « GRAPPES » du GdR ARP (Architecture, Réseaux et système, Parallélisme) est très active autour de ce genre de systèmes, dont il existe un nombre croissant de réalisations.

3. Choix du matériel

La définition qui nous semble communément admise d'un ordinateur de type Beowulf [STE 99] est une grappe de stations de travail, d'un type orienté vers le calcul numérique choisi parmi ceux de la plus grande diffusion, et utilisées uniquement pour le calcul parallèle (par opposition à un « cluster » traditionnel où les nœuds peuvent être utilisés individuellement comme poste de travail, par exemple). Le réseau de communications doit être local, rapide, dédié au parallélisme et découplé du « campus » par l'intermédiaire d'une machine frontale, et utilisant dans la mesure du possible des logiciels libres, en particulier le système d'exploitation Linux (des exceptions étant possibles pour les compilateurs ou les logiciels ou codes de calcul eux-mêmes).

4. Communications

La machine parallèle que nous avons construite est ainsi du type « mémoire distribuée », et les communications interprocesseurs se font explicitement par passage de messages en utilisant les bibliothèques MPI ou PVM. Le choix du réseau de communications est donc crucial, car le temps perdu à échanger les messages entre processeurs ne doit pas pénaliser d'un facteur trop important les gains de vitesse obtenus par la parallélisation des codes.

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

La solution minimale nous a semblé être l'utilisation du protocole TCP/IP standard, sur un réseau local de type Fast Ethernet (100 Mbit/s) géré par un commutateur réservé au calculateur parallèle. Nous avons de plus utilisé l'agrégation de canaux (« channel bonding »), pour quasiment doubler le débit point à point par rapport à Fast Ethernet, en équipant chaque nœud de deux cartes réseau.

Cette solution a l'avantage d'être bon marché et assez rapide sur le papier, mais au vu de l'expérience accumulée dans le domaine, il est clair que le protocole TCP/IP n'est pas la meilleure solution de passage de messages pour le calcul numérique intensif parallèle. Il peut par exemple y avoir accumulation d'un grand nombre de copies des messages dans le cas d'une diffusion d'un même message à un ensemble de processeurs, et la charge de travail imposée au processeur de chaque nœud pour gérer les communications peut rapidement devenir limitante.

Les solutions plus performantes (Myrinet, Gigabit Ethernet, SCI ou autres) sont évidemment beaucoup plus chères. Les performances que nous discutons dans la partie 7 montrent cependant que le choix de Fast Ethernet/TCP-IP ne pénalise pas particulièrement la majorité des codes que nous avions l'intention de porter sur le calculateur.

5. Configuration matérielle

Les machines du meilleur rapport performance/prix en septembre 1999 nous ont semblé être soit les PC dits « d'assembleur », soit les stations de travail à base de processeur Alpha. Au vu de l'expérience accumulée en matière de maintenance des premiers, c'est la solution pour laquelle nous avons opté, malgré le plus grand nombre de machines et donc les plus grands encombrement et dissipation de chaleur correspondants, à prix total égal. Vu le relativement faible nombre de machines (34) que notre budget permettait d'acquérir, nous avons également opté pour des PC en boîtier standard par opposition aux machines dites en châssis, ou « rack », certes moins encombrantes et plus faciles à gérer et alimenter électriquement, mais beaucoup plus dispendieuses.

Nous avons d'autre part choisi des machines biprocesseurs, afin de réduire l'encombrement total et d'économiser moitié d'alimentations et de boîtiers, malgré le coût plus élevé des cartes mères et la légère diminution de performance correspondante, dans certains cas du moins. Le choix des biprocesseurs nous semble optimal, les cartes mères à plus de processeurs étant à notre avis de très mauvais rapport performance/prix par suite sans doute de leur faible diffusion et de la complexité de leur conception en ce qui concerne la gestion de la mémoire et du cache.

La mémoire vive totale installée sur chaque machine a donc dû être choisie assez grande (512 Mo ici), valeur standard adoptée par la majorité des clusters essentiellement à cause de la taille des problèmes traités.

Nous avons choisi d'installer des disques durs dans chacun des nœuds de calcul, afin d'une part de disposer de stockage local pour certains codes, d'un espace disque

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

total considérable grâce à PVFS [CAR 00], et d'autre part d'un démarrage et d'une recherche de pannes facilitées par rapport à une solution « tout réseau » grâce à la présence d'un système local. L'apparition d'une distribution telle que « Beowulf 2 » est cependant en train de remettre en question ce dernier point.

Nous avons choisi de ne pas installer de lecteurs de CD-ROM dans les nœuds de calcul, l'installation du logiciel étant faite par le réseau, mais, bien que des solutions de démarrage initial automatique sur le réseau et de détachement de la console sur le port série soient possibles et très économiques, nous avons préféré pour la souplesse de fonctionnement et la recherche de pannes installer un lecteur de disquettes, une carte vidéo très bon marché et un commutateur clavier-écran-souris pour interagir facilement avec la console de chacun des nœuds et suivre le processus d'installation. Ce choix s'est avéré très pratique lors de la recherche des problèmes matériels, configuration des BIOS, etc.

La configuration matérielle que nous avons choisie après appel d'offres se compose donc de :

– 34 nœuds de calcul, comprenant chacun :

- boîtier moyen-tour Tellus TK 216 à démontage et ouvertures faciles, avec étiquette d'individualisation
- lecteur de disquette pour un démarrage initial facile
- carte mère ASUS P2B-D
- 2 processeurs Pentium III 600 Mhz (Katmaï, boxed)
- 4 bancs de mémoire 128 Mo PC100 SDRAM
- 2 cartes réseau Fast Ethernet EEPro 100 Management
- 1 disque dur de 6 (ou 8) Go IDE
- 1 carte écran Trio3D SVGA sur bus AGP
- pas de lecteur de CDROM (installation automatique de Linux kickstart sur NFS)
- pas de clavier ou de moniteur

– 1 nœud « maître » avec

- boîtier grande-tour
- lecteur de disquettes
- carte mère ASUS P2B-DS (SCSI intégré)
- 2 processeurs Pentium III 550 Mhz (Katmai, boxed)
- 4 bancs de mémoire 128 Mo PC100 SDRAM
- 2 disques durs IBM Ultrawide SCSI 18 Go
- lecteur de CDROM SCSI 40x
- Graveur Yamaha CD-RW 2 4 6
- 3 cartes réseau Fast Ethernet EEPro 100 Management (2 pour le cluster (channel bonding)- 1 pour l'extérieur reliée au réseau du campus)

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

- carte vidéo Matrox G200
- moniteur 21"
- onduleur APC-1000 Smart UPS relié par le port série
- sauvegarde sur bande HP Surestore DAT 24

– Commutateur (« Switch ») Cisco Catalyst 4000 Fast Ethernet avec 2*48 ports, contrôlé par le port série (ce commutateur permet l'agrégation de canaux par le partage en deux VLANs et l'utilisation du « Fast Etherchannel »)

– 3 commutateurs 16 ports Maxxtro KVM (clavier, écran, souris) avec un clavier et un moniteur basique de 15"

– climatiseur 7 kW Technibel

– régulateur de tension 6kW Salicru

– 3 étagères d'archivage standard de bureau

– 71 câbles Ethernet CAT-5, 34 câbles KVM étiquetés et tenus par des colliers plastiques

6. Configuration logicielle

Nous avons choisi d'installer la distribution Linux Redhat 6.1 (Cartman release) - avec le noyau 2.2.18 au moment où nous écrivons ces lignes - à cause de la facilité que procure la procédure d'installation automatique dite « Kickstart », qui permet d'obtenir rapidement une installation homogène sur tous les nœuds même si le matériel diffère légèrement (en particulier, les adresses IP sont attribuées automatiquement et les partitions sont faites de façon homogène même si les disques durs sont de modèles différents, ce qui était le cas dans notre configration).

Une installation restreinte au calculateur de NIS (pages jaunes) a été essayée dans un premier temps, mais des problèmes de stabilité et de surcharge réseau récurrents, d'ailleurs observés par d'autres groupes, nous ont conduits à nous limiter à distribuer des copies des fichiers d'authentification sur les nœuds de calcul.

Comme sur la plupart des Beowulf, les fichiers des utilisateurs résident sur la machine frontale et sont exportés par NFS sur les nœuds de calcul. Cette configuration n'est pas optimale pour le parallélisme car les codes de calcul ont souvent tendance à accéder aux mêmes portions de gros fichiers au même instant, ce pour quoi NFS n'a pas été conçu. Là encore, « Beowulf 2 » semble apporter une réponse à ce problème.

Quelques utilitaires du monde Beowulf ont également été installés, comme `prsh` qui permet l'exécution simultanée de commandes UNIX sur toute la grappe, et `Bwatch`, qui permet de surveiller l'état du calculateur (charge de chaque nœud, et moyennant une petite modification, température des boîtiers et des processeurs grâce aux capteurs de type LM78 sur les cartes mères).

Un système de gestion de files d'attente a également été installé. Les deux principaux logiciels libres sont à notre connaissance DQS [DQS] et PBS. Nous avons choisi

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

DQS, mais avons rencontré quelques manques dans la documentation au sujet de la soumission de travaux parallèles, qui est néanmoins possible et implémentée.

Le Portland Cluster Development Kit a par la suite été installé afin de bénéficier de f90, HPF, OpenMP, qui n'ont pas à l'heure actuelle d'équivalent disponible dans le domaine du logiciel libre.

7. Performances

Dans cette partie, après avoir discuté les performances de `MPI` sur le calculateur, nous présenterons les résultats du test standard `Scalapack`, puis les résultats de programmes spécifiques de modélisation numérique, de plus en plus exigeants du point de vue de la bande passante ou de la latence du réseau de communications.

7.1. *Réseau*

Les performances du réseau local dédié au calculateur ont été mesurées à l'aide du paquetage "SkaMPI" [REU 97], qui mesure de façon statistique les caractéristiques des appels MPI ; le résultat dépend donc aussi bien des performances de l'implémentation de MPI elle-même (ici MPIch 1.2), que des performances du système et du réseau. Par rapport aux performances brutes du réseau, proches de la norme Fast Ethernet de toute façon quel que soit le choix du système, des cartes réseau ou du commutateur, cet ensemble de tests nous semble plus représentatif pour l'usage ultérieur en calcul numérique, et nous semble plus dépendre de l'adéquation mutuelle des différents composants du Beowulf et donc des choix discutés plus haut.

Les résultats sont, pour la première fonction testée (`MPI_Bsend` point à point, moyenne sur 64 processeurs du cluster

– Fast Ethernet normal

- Latence à 0 octets : 773,5 μs
- Latence à 62,9844 ko : 14239,6 μs
- Bande passante à 62,9844 ko : 8,846 Mo/s

– Fast Ethernet, agrégation de deux canaux (« Channel Bonding »)

- Latence à 0 octets : 842,6 μs
- Latence à 62,9844 ko : 9601,8 μs
- Bande passante à 62,9844 ko : 13,319 Mo/s

On peut constater que la bande passante est augmentée d'à peu près 50 % par l'agrégation de canaux. On rejoint ici l'idéal des promoteurs du concept Beowulf [STE 99], à savoir une augmentation des performances à peu près proportionnelle

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

au coût du matériel ajouté. Les valeurs absolues, quoique relativement faibles par rapport aux performances des ordinateurs parallèles traditionnels, ou à celles de solutions moins « grand public», comme Myrinet, sont néanmoins acceptables pour un grand nombre de codes de physique/chimie computationnelles, comme la partie suivante le montre. Le goulot d'étranglement pour certains codes serait plutôt à trouver au niveau de la valeur très élevée de la latence.

L'agrégation de canaux comme l'usage de cartes mères biprocesseurs semblent avoir un impact négatif à ce niveau.

7.2. *Linpack*

Nous avons ensuite tenté d'évaluer les performances en calcul en utilisant un code standard tel que le test de virgule flottante de Linpack [DON 90], pour la résolution d'un système linéaire plein en double précision. Ce code est utilisé pour l'établissement bisannuel du « TOP 500 », [MT], dont nous avons suivi la procédure.

Cependant, la liberté laissée de choix de taille de la matrice la plus favorable permet à notre avis d'orienter ce benchmark parallèle vers un problème « embarassingly parallel », vu les mémoires vives énormes qui sont aujourd'hui monnaie courante.

Néanmoins, cette application authentiquement parallèle démontre la fonctionnalité du calculateur sur un vrai problème (quoique peu gourmand en communications), par rapport à un supercalculateur parallèle âgé de seulement quelques années, en particulier en ce qui concerne la capacité à traiter un calcul exigeant en mémoire vive (12 Go pour le problème de taille 40000×40000, qui est donc traité à moindre coût de façon répartie sur un système 32 bits comme celui ici présenté, au lieu de recourir à un système à 64 bits beaucoup plus dispendieux).

De plus, Linpack, qui génère une grande quantité de calculs, qui accède à une grande quantité de mémoire pour les grandes matrices de façon plus ou moins prévisible pour le système de cache, et qui induit une quantité proportionnellement faible mais non négligeable de communications, nous semble représentatif d'une bonne partie des codes de physico/chimie computationnelle. Les résultats pour différentes tailles de matrices sont présentés sur la figure 1.

On voit sur cette figure que pour la taille de matrice la plus favorable, les performances atteignent pour 64 processeurs environ 40 % du maximum théorique (300 MFlops par processeur Pentium III, obtenus grâce aux routines auto-ajustantes de ATLAS [WHA 99, WHA 01], qui choisissent une taille de vecteur optimale par rapport aux performances et à la taille du cache lors de l'installation pour les routines du BLAS, déroulent les boucles, découpent en blocs optimaux, etc...).

La proportionnalité est donc acceptable même pour 64 processeurs, mais la pente de la courbe de performances fait soupçonner que pour plus de 100 processeurs, une solution réseau plus performante que Fast Ethernet (même avec agrégation de canaux) serait à envisager. L'agrégation de canaux apporte un gain de performances de 10 à

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

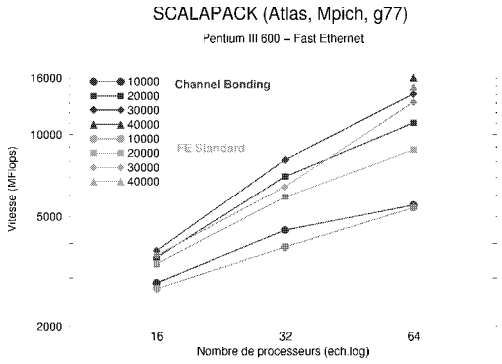

FIG. 1. *Performances de Linpack - comparaison Fast Ethernet simple / Agrégation de canaux en fonction du nombre de processeurs*

20 %, là encore à peu près proportionnel aux coûts supplémentaires. Cette augmentation de performance montre que Linpack, même pour les tailles de matrices relativement grandes que nous avons employées (le rapport quantité de calculs/quantité de communications étant donc favorable), présente tout de même une certaine sensibilité à la qualité et à la bande passante du réseau de communications reliant les nœuds de calcul. La latence semble n'avoir aucun impact, ce qui est raisonnable compte tenu des grandes tailles de messages générées par les grands systèmes que nous avons employés. Ce ne serait certainement pas le cas pour des systèmes de taille inférieure par un ordre de grandeur, par exemple.

7.3. *Codes de calcul utilisés et développés localement*

7.3.1. *Fonctionnelle de la densité sur grille régulière en espace direct*

Nous présentons d'abord les résultats de performance pour le code dont le comportement est le plus proche de celui de Linpack, qui vient juste d'être discuté. Il s'agit d'un logiciel utilisé en physique des molécules, des agrégats, des solides et des plasmas, inspiré de codes de physique nucléaire (les fonctions d'ondes sont discrétisées sur grille régulière en espace tridimensionnel). Le cadre théorique pour les électrons de valence du système est la fonctionnelle de la densité avec ses extensions dépendant du temps, restreinte le plus souvent à des versions locales en temps et en espace, jointe à l'usage de pseudopotentiels locaux ou non locaux et à une dynamique moléculaire classique pour les noyaux. La parallélisation est effectuée pour la majorité des calculs en distribuant les fonctions d'onde sur les différents processeurs. Le calcul des potentiels est, lui, parallélisé en distribuant des tranches de la grille tridimensionnelle à chaque processeur. Ce code a été développé par un des auteurs en collaboration [CAL 98], [CAL 00]. Les performances pour 64 fonctions d'onde discrétisées sur une grille 64^3, pour 200 itérations dynamiques, sont illustrées figure 2. Sur un seul

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

processeur, le code séquentiel correspondant (facilement généré d'ailleurs à partir de la source du programme parallèle grâce à l'emploi de directives de précompilation autour des parties spécifiquement parallèles) va environ à la même vitesse que le code parallèle sur deux processeurs.

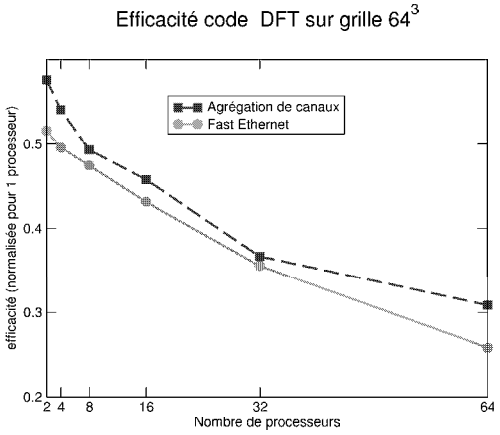

FIG. 2. *Comparaisons Fast Ethernet simple/ agrégation de canaux de l'efficacité parallèle d'un code de fonctionnelle de la densité sur réseau, en fonction du nombre de processeurs.*

On observe là encore une mise à l'échelle relativement bonne, jusqu'à 64 processeurs et sans doute au delà, et une amélioration de performances comprise entre 10 et 20 % grâce à l'agrégation de canaux. Les communications dans ce code sont en effet, comme dans le cas de Linpack, relativement peu fréquentes, mais impliquent, environ 1 fois par seconde, des messages de quelques Mo (des tranches entières de tableaux tridimensionnels de réels ou de complexes). La latence n'a donc aucun impact, et la bande passante de Fast Ethernet avec agrégation de canaux n'est pas pénalisante. Il est cependant à noter que le code en question, développé par des physiciens en nombre beaucoup plus restreint que les contributeurs à Linpack, n'est sans doute pas optimal. Il est, par exemple, exclusivement fait usage de communications bloquantes, et vu la complexité des équations et du processus itératif, l'accès à la mémoire et la gestion du cache ne sont sans doute pas les meilleures possibles. La parallélisation apporte cependant un gain considérable en vitesse grâce à la scalabilité relativement acceptable.

7.3.2. *Algorithme de Hoshen-Kopelman (HK)*

En physique, chimie, biologie, géologie il existe de nombreux processus plus ou moins aléatoires dans lesquels de petites particules s'agrègent pour former des structures complexes de grande taille appelées agrégats, amas ou flocs. Dans certains cas, ces structures peuvent s'étendre dans tout le système et former un gel. Le point de gel est l'instant où apparaît cet amas « infini ». Elle peut être décrite par la théorie de la percolation [STA 92, GIM 99]. Malheureusement, cette théorie ne peut pas être

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

résolue analytiquement dans des espaces tridimensionnels et il faut faire appel à des simulations de type Monte-Carlo. Ces dernières consistent à représenter l'espace à 3 dimensions par un réseau discret de sites (réseau cubique par exemple) puis à distribuer aléatoirement de la matière en occupant une fraction donnée des sites. Pour chaque fraction de sites occupés, il faut ensuite identifier et dénombrer les amas formés (un amas est constitué d'un ensemble de sites occupés proche-voisins). L'algorithme de Hoshen-Kopelman (HK) [HOS 76] est couramment utilisé pour effectuer ce travail. Au voisinage d'une concentration critique de sites (31.16 % sur un réseau cubique), le rayon moyen des amas diverge et un amas de taille infini apparaît. L'étude de ce point critique nécessite de faire varier la taille du réseau utilisé afin de s'affranchir des effets de taille finie. Pour chaque concentration proche du point de gel, il faut de plus réaliser plusieurs dizaines de milliers d'essais indépendants afin de calculer avec précision la distribution des amas.

Afin d'étudier la transition de percolation sur des réseaux cubiques dont la taille est comprise entre 200^3 et 1000^3 sites, nous avons dû développer une version parallèle de l'algorithme HK [GIM 00]. En résumé, pour n processeurs, le réseau de taille L^3 est découpé en n tranches (sous- réseau), réparties chacune sur un processeur et analysées indépendamment les unes des autres avec l'algorithme séquentiel HK. Ensuite il convient de reconstituer les amas qui ont été artificiellement fractionnés par le découpage. Pour cela, nous avons utilisé les instructions de la bibliothèque `MPI` afin que les processeurs échangent les informations pertinentes stockées aux frontières de leurs sous-réseaux.

Cet algorithme a été implémenté en `C` et en `FORTRAN90` et a été testé sur deux architectures : le Cray T3E de l'IDRIS (256 processeurs alpha EV5 cadencés à 300 MHz et disposant chacun de 128 Mo de RAM) et sur le cluster de PC ici discuté (baptisé Lotus). Rappelons que le Cray T3E dispose d'un réseau ultra rapide de communication entre processeurs alors que les communications de Lotus sont basées sur des interfaces « Fast Ethernet » 100Mb/s en agrégation de canaux.

Pour la concentration critique de 31,16 % sur un réseau cubique de taille L^3, nous avons mesuré le temps de calcul nécessaire au dénombrement des amas en utilisant n processeurs.

La figure 3 montre l'évolution du temps de calcul en fonction du nombre de processeurs pour différentes tailles de réseaux. On constate que pour Lotus, les communications deviennent très rapidement l'étape limitante du calcul (au delà de 16 processeurs, le gain devient négligeable, voire même le calcul nécessite plus de temps). Cela n'est pas le cas sur le Cray T3E.

Les performances médiocres du réseau « Fast Ethernet » de Lotus ne permettent donc pas dans ce cas d'atteindre la bonne mise à l'échelle observée sur le T3E, d'autant plus que les processeurs Pentium 600 de Lotus étant nettement plus rapides dans la plupart des cas que les Alpha EV5 à 300 MHz , le rapport bande passante/vitesse de calcul y est plus défavorable.

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

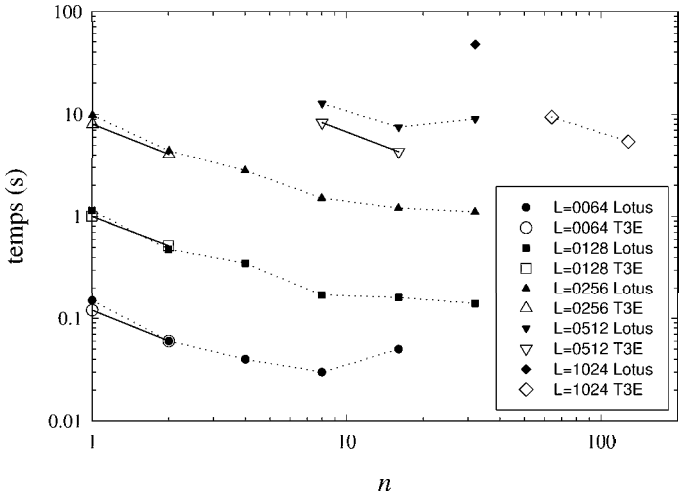

FIG. 3. *Évolution du temps de calcul du code HK en fonction de la taille du sous-réseau L^3/n et de l'architecture*

Ainsi, le cluster de PC permet de tester localement des réseaux de grande taille (plus de 2 Go) dans un temps raisonnable (47 secondes). Cela nous permet de mettre au point les codes et de résoudre des problèmes de taille intermédiaire.

7.3.3. *Modèle de Heisenberg classique sur réseau*

Le modèle de Heisenberg classique sur réseau [HER 68] est utilisé pour simuler les propriétés de différents nanosystèmes magnétiques. Ici, à l'aide de codes de type Monte-Carlo/Metropolis, développés au laboratoire PEC, il est possible d'évaluer des grandeurs thermodynamiques usuelles telles que la chaleur spécifique, sur des systèmes de spins classiques. Dans ces simulations nous avons utilisé une grille de $L \times L \times H$ sites et un Hamiltonien d'interaction qui donne l'énergie sur chaque site. Le calcul de cet Hamiltonien fait intervenir quatre termes dont les trois premiers sont locaux, et le dernier (terme dipolaire) nécessite un calcul sur l'ensemble de la grille.

La parallélisation de l'algorithme permet de distribuer le calcul du terme non local (le plus coûteux en temps de calcul) sur N processeurs. L'accélération obtenue (Temps de calcul sur 1 processeur/ Temps de calcul sur N processeurs) est dépendante du rapport temps de calcul sur temps de communication, lui même dépendant de la taille du système simulé et des performances du réseau de communication. La taille des

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

données à échanger est faible (une dizaine d'octets), c'est donc la latence du système de passage de messages qui sera limitante lors d'une communication.

L'accélération obtenue pour le même calcul, distribué sur un nombre croissant de processeurs, est représentée figure 4. La taille de la grille utilisée est de $11 \times 11 \times 96$. Comme on peut le voir, on est loin d'une loi de mise à l'échelle linéaire. Le temps de calcul de chaque tranche de grille diminue comme $96/N$ mais la latence reste au mieux constante.

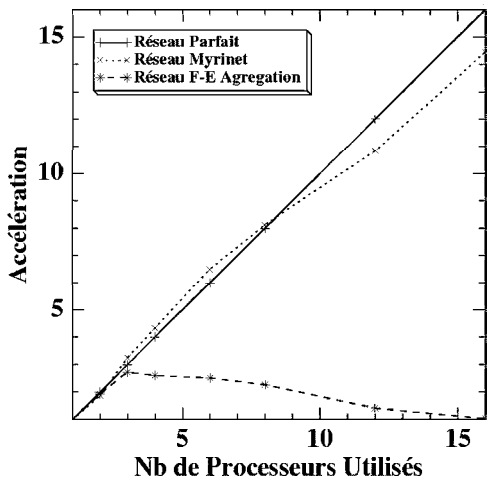

FIG. 4. *Performances du code simulant le modèle de Heisenberg classique sur réseau tridimensionnel en fonction du nombre de processeurs*

Cet exemple montre la difficulté d'obtenir une mise à l'échelle de ce type de calcul dit à faible grain. Afin d'obtenir une meilleure mise à l'échelle nous présentons les résultats obtenus pour le même programme sur une grappe de PC de vitesse comparable mais équipés d'un réseau de type Myrinet 2000 à faible latence (de l'ordre de quelques microsecondes). Les résultats d'accélération obtenus pour le même calcul mais avec ce nouveau réseau de communication sont tracés sur le même graphique (courbe en pointillés). Ce type d'algorithme nécessite une communication (de quelques octets) par étape et tire donc profit des performances d'un réseau de communication performant.

8. Conclusion

Le rapport performance/prix des calculateurs parallèles de type Beowulf est désormais suffisamment bon pour que ce choix soit à considérer pour les besoins intermédiaires en calcul. En effet, une partie importante des codes de calcul parallèle se

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

satisfont, pour un nombre de processeurs inférieur à la centaine, d'un réseau de communications aux performances moyennes voire médiocres (comme Fast Ethernet sur TCP/IP) par rapport à la quantité de données que peut générer une station de travail contemporaine. De plus, une équipe de non spécialistes en parallélisme et informatique a pu mener à bien la configuration matérielle et logicielle d'un tel calculateur, en utilisant quasi-exclusivement des logiciels libres et le savoir-faire accumulé depuis 1994 par le groupe original créateur du concept de Beowulf, ainsi que le nombre croissant d'utilisateurs dans le monde y ayant eu recours. Les économies ainsi réalisées permettent d'avoir un système de très forte puissance de calcul par rapport au nombre d'utilisateurs, et adapté sur mesure aux besoins : nous aurions pu par exemple consacrer une partie plus importante du budget à la qualité du réseau de communications, ou aux mémoires de masse, si la majorité de nos logiciels l'avaient exigé. Les performances ici discutées, mesurées sur des codes parallèles courants, montrent que le système est fonctionnel, fiable et performant, sauf évidemment dans les problèmes les plus exigeants en termes de performances de réseau, comme les deux derniers exemples discutés.

Remerciements

Nous remercions A.Gibaud, A.Bulou, A. Khater, G.-R. Perrin, le MENRT, la région Pays de la Loire, le CETIC de l'UdM, J.-L. Pazat, J.-J. Rousseau, N.Etourneau, J.-M. Grenèche, T.Davis, et les contributeurs au logiciel libre, GNU/Linux et le projet Beowulf.

9. Bibliographie

[CAL 98] CALVAYRAC F., « Dynamique non-linéaire des électrons de valence dans les agrégats métalliques », *Annales de Physique*, volume 23, 1998, pages 1-82.

[CAL 00] CALVAYRAC F., REINHARD P.-G., SURAUD E.etULLRICH C. A., « Nonlinear electron dynamics in metal clusters », *Physics Reports*, volume 337, 2000, pages 493-578.

[CAR 00] CARNS H., III W. B. L., ROSS R. B.etTHAKUR R., « 'PVFS: A Parallel File System For Linux Clusters », Dans *Proceedings of the 4th Annual Linux Showcase and Conference, Atlanta*, Octobre 2000, pages 317-427.

[DON 90] DONGARRA J. J., DUFF I. S., SORENSEN D. C.etVAN DER VORST H. A., *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM Publications, 1990.

[DQS] voir `www.scri.fsu.edu/~pasko/dqs.html`.

[GIM 99] GIMEL J.-C., NICOLAÏ T., DURAND D.etTEULER J.-M., « Structure and size distribution of percolating clusters. comparison with gelling systems », *Europhysics Journal B*, volume 12, 1999, pages 91-97.

[GIM 00] GIMEL J.-C.etTEULER J.-M., « A direct parallel implementation of the Hoshen-Kopelman algorithm for distributed memory architectures », *Computer Physics Communications*, volume 130, 2000, pages 118-129.

[HER 68] HERPIN A., *Théorie du Magnétisme*, P.U.F., Paris, 1968.

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

[HOS 76] HOSHEN J.etKOPELMAN R., « Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm », *Physical Review B*, volume 14, 1976, pages 3438-3445.

[MT] voir `www.top500.org`.

[REU 97] REUSSNER R., « Portable Leistungsmessung des Message-Passing-Interfaces », Diplomarbeit, Universität Karlsruhe, 1997, voir `wwwipd.ira.uka.de/~skampi/` .

[STA 92] STAUFFER D.etAHARONY A., *Introduction to percolation theory*, Taylor and Francis, London, 1992.

[STE 99] STERLING T. L., BECKER D. J., SALMON J. S.etSAVARESE D. F., *How to Build a Beowulf: A Guide to Implementation and Application of PC Clusters*, MIT Press, Cambridge, 1999.

[WHA 99] WHALEY R. C.etDONGARRA J. J., « Automatically Tuned Linear Algebra Software », Dans 9^{th} *SIAM Conference on Parallel Processing for Scientific Computing*, 1999, voir aussi `www.netlib.org/atlas` .

[WHA 01] WHALEY R. C., PETITET A.etDONGARRA J. J., « Automated Empirical Optimizations of Software and the ATLAS Project », *Parallel Computing*, volume 27, 2001, pages 3-25.

Article reçu le 9 avril 2001
Version révisée le 25 mars 2002
Rédacteur responsable : FRANCK CAPPELLO

Florent Calvayrac *est ancien élève de l'E.N.S. de Lyon, docteur en physique théorique. Il est actuellement Maître de Conférences à l'Université du Maine. Ses activités de recherche au sein du laboratoire de Physique de l'État Condensé (UMR 6087) portent sur la modélisation numérique* ab initio *et semi-empirique de systèmes nanoscopiques, le calcul parallèle et les développements méthodologiques reliés à ces questions.*

Yvan Labaye *est docteur en Sciences Physiques option physique et chimie des matériaux . Il est actuellement Maître de Conférences à l'Université du Maine. Ses activités de recherche au sein du laboratoire de Physique de l'État Condensé (UMR 6087) concernent les systèmes magnétiques nanostructurés et plus précisément la simulation numérique par méthode de Monte-Carlo.*

Jean-Christophe Gimel *est ingénieur de l'École Nationale Supérieure des Industries Alimentaires, docteur en Physico-Chimie des Polymères. Il est actuellement Chargé de Recherches au CNRS affecté au laboratoire Polymères, Colloïdes, Interfaces (UMR 6120) à l'Université du Maine. Ses activités de recherche concernent les phénomènes d'agrégation et de séparation de phase par les méthodes de Monte-Carlo.*

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com

Annexe pour le service de fabrication

Article pour la revue :
RSTI - TSI

Auteurs :
Florent Calvayrac [*] — *Yvan Labaye* [*] — *Jean-Christophe Gimel* [**]

Titre de l'article :
Un « Beowulf » par des physiciens

Titre abrégé :
Un « Beowulf » par des physiciens

Traduction du titre :
A « Beowulf » by physicists

Date de cette version :
2nd December 2002

Coordonnées des auteurs :

– téléphone : +33 2 43 83 26 26
– télécopie : +33 2 43 83 35 18
– Email : Florent.Calvayrac@univ-lemans.fr

Logiciel utilisé pour la préparation de cet article :
LaTeX, avec le fichier de style `article-hermes.cls`,
version 1.5 du 21.6.94.

Formulaire de copyright :
Joindre le formulaire de copyright signé, récupéré sur le web à l'adresse
`http://www.hermes-science.com`

Cet article des Editions Lavoisier est disponible en acces libre et gratuit sur tsi.revuesonline.com