Annexe F : Les jeux d'essai (benchmark)

Les performances théoriques maximales annoncées des constructeurs ne sont que rarement atteintes par des applications réelles. Ces différences sont principalement dues aux compilateurs, aux systèmes d'exploitation et aux congestions des réseaux d'interconnexions entre les ressources. Afin d'évaluer les performances d'une machine, des applications appelées jeux d'essai ont été développées. Ces jeux d'essai sont en fait des applications réelles auxquelles ont été ajoutées des instructions de mesure de temps et d'espionnage de l'utilisation des ressources.

les jeux d'essai sont censés [Damm 96] :

- être portables sur différentes types de machines, ce qui implique le plus souvent une programmation dans un langage de haut niveau ;
- être représentatifs de classes d'applications réelles ;
- supporter le changement d'échelles (*scalability*) pour qu'il soit possible de choisir la taille du problème sans en changer la nature ;
- fournir des métriques utiles et précises, tout en conservant des notions générales telles que le gain (*speed-up*) ou l'efficacité ;
- fournir des mesures de paramètres systèmes ;
- être fiables et exempts de bogues ;
- être accessibles à tous (mise dans le domaine public) ;
- être bien documentés.

Les différents types de jeux d'essai

Les applications inclues dans les jeux d'essai sont divisibles en trois classes : les applications réelles, les noyaux de calcul et les applications synthétiques.

Les applications réelles, comme leur nom l'indique, ont été développées à des fins de calcul et ensuite modifiée pour devenir des jeux d'essai. D'une part, ces applications comportent de nombreuses lignes de codes (ce qui compliquent leur compréhension et leur portabilité) et d'autre part, elles fournissent des mesures difficilement représentatives du comportement général d'une machine (car trop liées au problème). Dans cette catégorie de jeu d'essai, on peut citer entre autres PERFECT [Cybenko 92] et SPEC [Spec 90].

Les noyaux de calcul (*kernel*) sont des parties de calcul qui sont très souvent utilisées dans des application numériques réelles telles que des fonctions de résolution de systèmes linéaires, de calcul matriciel (inversion, transposition, résolution). Les noyaux de calcul donnent des résultats plus facilement interprétables car ils sont liées à un type d'application donnée. L'exemple le plus connu est le Livermore Fortran Kernel (aussi appelé Lawrence Livermore Loops) [MacMahon 86] qui est un jeu de test basé sur 24 boucles Fortran typiques extraites d'applications numériques dans le domaine des sciences physiques.

Enfin, **les applications synthétiques** sont des programmes qui simulent l'exécution d'une application réelle en reproduisant son utilisation des ressources (telles que la mémoire, les processeurs, les gestionnaires d'entrées-sorties, etc.). Ainsi, l'environnement Alpes [Kitajima & al. 93] dispose d'un modèle de programme appelé ANDES (*Algorithms aNd DEScription*) [Kitajima 94] à partir duquel, il est possible de générer un programme synthétique. Le programme synthétique est ensuite exécuté sur une véritable

machine parallèle. On retrouve la même approche avec OLGA [Waser & al. 93], qui produit des programmes synthétiques à partir de programmes Occam [Hoare 88].

Les jeux d'essai séquentiels

Les premiers jeux d'essai ont été développés pour mesurer les performances d'applications de calcul séquentiel et vectoriel [Weicker 91]. Parmi ces jeux d'essai on peut citer entre autres :

- **Linpack/Lapack** [Dongarra & al. 76 et Dongarra 89] était au départ une bibliothèque de routines d'algèbre linéaire (d'où son nom *Linear Algebra PACKage*). C'est désormais un jeu de test qui effectue des résolutions de systèmes denses d'équations linéaires. Les résultats de l'exécution des jeux d'essai Linpack sont exprimés en MFLOPS (*Million Floating-point Instructions Per Second*). Linpack est utilisé régulièrement pour établir le Top 500 des super-ordinateurs [Dongarra & al. 96] et produire les résultats de mesures en utilisant les paramètres définis par [Hockney 91]. Enfin, il existe une version de LinPack en langage Java [LinpackJava 97].
- **Whetstone** [Curnow & al. 76] est un jeu d'essai qui a d'abord été conçu en langage Algol, puis porté en Fortran, en Pascal et en C. Il réalise des opérations sur des nombres entiers, sur des nombres flottants et effectue des tests et des appels de fonctions. Les résultats sont exprimés en MWIPS (*Mega Whetstone Instructions Per Second*).
- **Dhrystone** [Weicker & al. 84] est un jeu de tests orienté système. Il contient douze procédures et une boucle avec 94 instructions. L'exécution d'une boucle est un drhrystone et l'unité de mesure est le drhrystone par seconde.
- **Livermore Fortran Kernel** [MacMahon 86] fournit des résultats exprimés en MFLOPS pour chaque boucle.

les jeux d'essai parallèles

les jeux d'essai présentés ci-dessus ont été conçus pour des machines séquentielles, c'est pourquoi, ces dernières années de nouveaux jeux d'essai dédiés aux machines parallèles ont fait leur apparition. Parmi les plus connus on trouve :

- **Scalapack** [Blackford & al. 97] est l'équivalent de Lapack, mais pour des machines distantes qui communiquent par passage de messages ou pour des réseaux de stations de travail utilisant les bibliothèques de communication PVM ou MPI. Les résultats sont exprimés en MFLOPS.
- **Parkbench** (*PARallel Kernel and BENCHmarks*) [Parkbench 97] est un jeu d'essai qui a été défini par un comité fondé en 1992. Ce jeu d'essai est écrit en Fortran 77 (pour les applications séquentielles) et avec PVM (pour les applications parallèles). Dans Parkbench, les applications sont classés en trois groupes, en fonction du niveau des paramètres qu'elles mesurent. On distingue alors les applications de bas niveaux, les noyaux de calcul et les applications réelles.

Conclusion

Le lecteur intéressé par les jeux d'essai trouvera des compléments d'informations sur le site internet BenchWeb [Benchweb 97].

D'après des études récentes, il existe un corrélation de plus de 95% entre les vitesses de crête annoncée par les constructeurs et le jeu d'essai Linpack [Damm 96]. Il est donc difficile dans ces conditions de se fier à l'un ou à l'autre. Ces jeux d'essai permettent néanmoins, pour des classes d'applications données, d'avoir des comparatifs de puissance entre super-calculateurs ou réseaux de stations de travail. On ne peut pas par contre

négliger les différences de qualité des outils logiciels de développement disponibles sur ces machines qui ont été utilisés lors des tests. Certains compilateurs optimisés pour des machines spécifiques produisent un code de bas niveau de très bonne qualité et offre de meilleurs temps d'exécution. Ceci complique donc la tâche d'évaluation.

Références

[Benchweb 97] BenchWeb : point de départ sur le Web de recherche d'informations concernant les jeux d'essai (benchmark), <http://www.netlib.org/benchweb>.

[Blackford & al. 97] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon,J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, & R. C. Whaley, «ScaLAPACK: A Linear Algebra Library for Message-Passing Computers», SIAM Conference on Parallel Processing, March 1997, <http://www.netlib.org/utk/papers/siam397-scalapack/siam397-scalapack.ps>.

[Curnow & al. 76] H.J. Curnow & B.A. Wichmann, «A synthetic Benchmark», Computer Journal, Vol. 19 (1), 1976, pp. 43-49, <http://www.netlib.org/benchmark/whetstones>.

[Cybenko 92] G. Cybenko, «Supercomputer Performance Trends and the PERFECT Benchmarks», Supercomputing Review, 1992, pp. 53-60.

[Damm 96] G. Damm, «Conception et Réalisation d'une Bibliothèque de Modèles d'Architectures Parallèles», Thèse de doctorat de l'Université Pierre et Marie Curie, 13 Décembre 1996.

[Dongarra & al. 76] J.J. Dongara & al., «LINPACK User's Guide», SIAM Publications, Philadelphia PA, 1976.

[Dongarra 89] J.J. Dongara, «Performance of Various Computers Using Standard Linear Equations Software (LinpackBenchmark Report)», Computer Science Department, University of Tennessee, Technical Report UT-CS-89-95, **1997** *(regulary updated)*, <http://www.netlib.org/benchmark/performance.ps>.

[Dongarra & al. 96] Jack Dongarra, Hans Meuer & Erich Strohmaier, «Top500 Report», November 1996, <http://www.netlib.org/benchmark/top500/lists/top500_9611.ps>.

[Hoare 88] C. Hoare, «Occam 2 Reference Manual», Prentice Hall, 1988.

[Hockney 91] R. Hockney, «Performance Parameters and Benchmarking of Supercomputers», North Holland Eds., Parallel Computing, Vol 17, pp. 1111-1130, December 1991.

[Kitajima & al. 93] J.P. Kitajima, C. Tron & B. Plateau, «ALPES: a tool for the performance evaluation of parallel programs», in Environment and Tools for Parallel Scientific Computing, J.J. dongarra & B. Tourancheau Eds., Amsterdam, The Nederlands, 1993, North Holland, pp. 213-228.

[Kitajima 94] J.P. Kitajima, «Modèles quantitatifs d'algorithmes parallèles», PhD Thesis, Institut Polytechnique de Grenoble, 1994.

[LinpackJava 97] <http://www.netlib.org/benchmark/linpackjava/>.

[MacMahon 86] F.H. MacMahon, «The Livermore Fortran Kernels: a Computer Test of the Numerical Performance Range», Lawrence Livermore National Laboratory, Technical Report UCRL-53745, December 1986.

[Parkbench 97] <http://www.netlib.org/parkbench/>

[Spec 90] System Performance Evaluation Cooperative, «SPEC Benchmark Release 1.0», Supercomputing Review, 1990, pp. 48-57.

[Waser & al. 93] S. Waser & H. Burkhart, «Olga: a Modelling Tool for Algorithmic Skeletons», Reinhart Grebe & al. Eds, Transputer Applications and Systems '93, Vol. 1, IOS Press, Amsterdam, The Nederlands, 1993, pp. 395-409.

[Weicker & al. 84] R.P. Weicker & R. Dhrystone, «A Synthetic Systems Programming Benchmark», Communication of the ACM, Vol. 27 (10), pp. 1013-1030, October 1984, <http://www.netlib.org//benchmark/dhry-c>.

[Weicker 91] R.P. Weicker, «A Detailled Look at Some Popular Benchmarks», North Holland Eds., Parallel Computing, Vol. 17, pp. 1153-1172, December 1991.