Bibliotecas e Ferramentas para Computação Numérica de Alto Desempenho

Bernardo Goerl[1], Cleber Roberto Milani[1],
Mateus Raeder[1], Mariana Kolberg[2]
Luiz Gustavo Fernandes[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
[2]Universidade Luterana do Brasil
{bernardo.goerl, cleber.milani, mateus.raeder}@acad.pucrs.br,
luiz.fernandes@pucrs.br, mariana.kolberg@ulbra.br

Resumo

A resolução de Sistemas de Equações Lineares é um problema de grande importância em diversas áreas e no qual cada vez mais tem-se aplicado a Computação Verificada, uma vez que os algoritmos tradicionais não garantem a corretude da solução. Entretanto, a Computação Verificada aumenta o custo computacional e, em alguns casos, impossibilita a resolução do problema em um tempo aceitável. Ferramentas de computação paralela de alto desempenho surgem como alternativas para minimizar este custo. Diversos trabalhos têm focado em otimizar a Computação Verificada para execução em agregados de computadores. Nos últimos anos, a computação de alto desempenho tem sido explorada em processadores multicore e nos processadores das placas de vídeo (GPUs), dada a maior facilidade de acesso a esses do que aos agregados. Este trabalho apresenta um estudo sobre a Computação Numérica de Alto Desempenho como suporte à Computação Verificada.

Introdução

A Computação Verificada (CV) é essencial em diversas áreas do conhecimento nas quais o rigor dos resultados computados é um requisito indispensável. Em Ciência da Computação a garantia de que um algoritmo está correto não implica, necessariamente, que o resultado computado por ele será correto [HAM 97]. Frequentemente, um computador digital produz resultados incorretos para problemas numéricos, não devido a erros de programação ou ao uso de hardware não confiável, mas por ser uma máquina discreta e finita que não consegue tratar alguns dos aspectos contínuos e infinitos da Matemática. Por esse motivo, faz-se necessária a utilização da Aritmética Intervalar a qual, além de ser um requisito para a CV amplia o domínio dos problemas numéricos computáveis, pois permite trabalhar com dados imprecisos, o que não é possível através de sistemas formados por números pontuais.

Bibliotecas puramente de Computação Verificada, embora permitam a resolução verificada e deêm suporte aos sistemas intervalares, transformam-se em um gargalo na aplicação por requererem uma grande quantidade de operações adicionais. Por isso, é comum a utilização de bibliotecas numéricas de alto desempenho para otimização de trechos dos algoritmos da Computação Verificada.

O casamento entre a Computação Verificada e a Computação de Alto Desempenho surge como consequência natural desse contexto, pois, faz-se necessário computar sistemas cada vez mais complexos com maior exatidão. Ao longo dos anos,

trabalhos têm sido desenvolvidos combinando algoritmos de Computação Verificada com técnicas da Computação Paralela em *clusters*. Entretanto, nos dias de hoje é possível observar a maciça utilização de processadores *multicore* e processamento numérico beneficiando-se das GPUs cada vez mais potentes pela popularização de jogos e ferramentas gráficas. Ademais, mesmo os *clusters* modernos têm sido construídos utilizando processadores *multicore* e, em alguns casos, nós compostos por máquinas NUMA com múltiplos nós *multicore*. Assim, a adaptação das soluções da Computação Verificada para esse novo nicho de arquiteturas é de grande interesse.

Computação Numérica de Alto Desempenho

A compreensão do inter-relacionamento entre as características de um algoritmo paralelo e as da arquitetura em que ele irá executar é fundamental para o desempenho. Em geral, os trabalhos de Computação Verificada que aplicam técnicas de computação paralela em agregados de computadores (*clusters*) fazem uso combinado das bibliotecas MPI *(Message Passing Interface*) e o ScaLAPACK [CHO 96]. Alguns deles, como [KOL 07] e [KOL 08a], operam sistemas lineares do tipo pontual enquanto outros, como [KOL 08b], resolvem sistemas lineares intervalares. Bibliotecas como o ScaLAPACK são construídas com base na BLAS (*Basic Linear Algebra Subprograms*) [LAW 79] e empregam a Computação Paralela para otimizar a resolução dos mais diversos problemas de álgebra linear. A seguir, são descritas algumas dessas bibliotecas. Cabe ainda observar que, mesmo com a utilização dessas bibliotecas de *software* otimizadas, a resolução de SELAs continua apresentando grande custo computacional quando se tratando de sistemas de grande porte. O custo torna-se ainda maior ao considerar sistemas lineares intervalares.

BLAS

A BLAS (*Basic Linear Algebra Subprograms*) é utilizada como núcleo comum para o desenvolvimento de diversos pacotes de software para álgebra linear. As versões de referência as rotinas da BLAS são disponibilizadas livremente na Internet. Entretanto, os fabricantes de processadores oferecem implementações proprietárias otimizadas para suas arquiteturas. Além disso, existem diversas implementações alternativas da BLAS disponíveis gratuitamente na Internet construídas sob diferentes abordagens. Dentre essas, destacam-se a ATLAS (*Automatically Tuned Linear Algebra Software*) e a Goto BLAS. Além das otimizações em nível de instrução, as versões otimizadas da BLAS fazem uso maciço dos recursos de *multithread*. Cabe ainda observar a existência de uma versão paralela da BLAS para multicomputadores disponível para sistemas com suporte a MPI ou PVM, a PBLAS (*Parallel Basic Linear Algebra Subprograms*) [CHO 95].

LINPACK, LAPACK e ScaLAPACK

Os pacotes LINPACK, LAPACK e ScaLAPACK são conjuntos de rotinas de software para álgebra linear implementados com base na BLAS. Basicamente, a diferença entre eles é o tipo de arquitetura em que estão focados. O LINPACK é baseado nas operações de vetores (*Level* 1 BLAS), e foi desenvolvido para os supercomputadores existentes nos anos 70 e 80. Por volta dos anos 80, percebeu-se a existência de uma lacuna entre o desempenho dos processadores e memórias e, então, surgiu a necessidade de reformular os algoritmos do pacote de modo que es-

ses operassem matrizes com matrizes, ou seja, operações da BLAS *Level* 3. Nesse contexto, surgiu o LAPACK como sucessor do LINPACK.

O desenvolvimento do LAPACK teve como objetivo fazer as já famosas bibliotecas EISPACK e LINPACK executarem eficientemente em supercomputadores vetoriais de memória compartilhada. Nessas arquiteturas o LINPACK e EISPACK eram ineficientes, pois seus padrões de acesso à memória desrespeitavam a hierarquia de memória com múltiplas camadas dessas máquinas. Nos anos 90, com o advento e popularização dos aglomerados de computadores, foi desenvolvida uma versão do LAPACK para arquiteturas de memória distribuída, o ScaLAPACK.

O ScaLAPACK utiliza o paradigma *Single-Program-Multiple-Data* (SPMD) com troca de mensagens explícita para comunicação entre processadores heterogêneos que suportem MPI ou PVM. Da mesma forma que no LAPACK, as rotinas do ScaLAPACK são baseadas no particionamento em blocos dos algoritmos e assume-se que as matrizes são armazenadas em blocos bi-dimensionais. Os níveis 1, 2 e 3 da implementação paralela da BLAS, PBLAS, são utilizados no ScaLAPACK em conjunto com a BLACS (*Basic Linear Algebra Communication Subprograms*) [DON 97]. Para atingir escalabilidade, além do particionamento em blocos de tamanhos ajustáveis dos algoritmos, o ScaLAPACK possui diversos algoritmos equivalentes para um mesmo cálculo permitindo assim a escolha, em tempo de execução, do melhor algoritmo para uma dada entrada ou arquitetura em execução.

Computação Numérica em Arquiteturas Híbridas

Nos dias de hoje, a Computação de Alto Desempenho tem se voltado fortemente à exploração de computadores com processadores *multicore* e processamento em GPUs, bem como *clusters* combinando os mesmos em arquiteturas híbridas. Essa tendência deve se manter pelos próximos anos devido ao menor custo e maior facilidade de acesso a essas máquinas. Por serem arquiteturas bastante recentes, essas não se encaixam em nenhuma das classificações tradicionais utilizadas para máquinas paralelas. Cabe também observar que, de acordo com [BUT 08], processadores *multicore* não podem ser vistos como sendo um novo tipo de SMP e nem mesmo como semelhantes aos computadores com múltiplos processadores.

Projetos em andamento tem desenvolvido diversas rotinas da álgebra linear otimizadas para esse novo paradigma da computação paralela. No entanto, não se econtram, hoje, disponíveis pacotes de *software* numéricos otimizados tão completos quanto o LAPACK e ScaLAPACK para as novas arquiteturas. No caso dos processadores *multicore*, destaca-se o projeto PLASMA (*Parallel Linear Algebra for Scalable Multi-core Architectures*) [BUT 08] o qual visa suceder o LAPACK e ScaLAPACK. A biblioteca PLASMA emprega a BLAS na implementação de operações com fluxo único de execução, também conhecidas como núcleos (*kernels*). Com isso, as otimizações em nível de instrução (dependentes de máquina) são exploradas por implementações eficientes da BLAS ao passo que o paralelismo das operações é explorado em um nível algorítmico acima, pela PLASMA.

Já no caso das arquiteturas híbridas, cabe destacar o projeto MAGMA (*Matrix Algebra on GPU and Multicore Architectures*) [TOM 10]. A filosofia neste caso é de que, para atacar problemas complexos, as arquiteturas híbridas emergentes precisam de soluções de *software* também híbridas as quais combinem os benefícios de diferentes algoritmos em um mesmo *framework*. Dessa forma, uma mesma rotina pode ser adaptativa à arquiteutra em que estiver executando. No domínio das

implementações proprietárias cabe destacar o CUDA *Toolkit*, *framework* criado para permitir que desenvolvedores obtenham o máximo dos recursos de processamento dos GPUs de placas NVidia.

Considerações Finais

A adaptação e desenvolvimento de novas ferramentas da Computação Numérica para computadores híbridos surge, assim, como consequência natural da mudança de paradigmas. Alguns esforços iniciais nesse contexto são os trabalhos desenvolvidos em [KOL 08] e [MIL 10] os quais se propõem a resolver sistemas intervalares em processadores *dualcore* e *multicore*, respectivamente.

Dando continuidade aos estudos e pesquisas, está sendo avaliado o desenvolvimento de estratégias que permitam obter ganhos de desempenho em computadores heterogêneos/híbridos, ou seja, naqueles em que é possível explorar o paralelismo não apenas em CPUs com múltiplos *cores* mas também em GPUs. Uma possibilidade nesse sentido é a utilização de resultados do projeto MAGMA bem como uma possível integração quando houver bibliotecas disponibilizadas por ele.

Referências

[HAM 97] R. Hammer, D. Ratz, U. W. Kulisch e M. Hocks. C++ Toolbox for Verified Scientific Computing I: Basic Numerical Problems. Secaucus:Springer-Verlag, 1997, 400p.

[KOL 07] M. L. Kolberg, L. Baldo, P. Velho, L. G. Fernandes, and D. M. Claudio. Optimizing a parallel self-verified method for solving linear systems. In Applied Parallel Computing. State of the Art in Scientific Computing, volume 4699, pages 949-955, Berlin, 2007. Springer Lecture Notes in Computer Science.

[KOL 08a] M. L. Kolberg, L. G. Fernandes, and D. M. Claudio. Dense linear system: A parallel selfverified solver. International Journal of Parallel Programming, 36(4):412-425, 2008.

[KOL 08b] M. L. Kolberg, M. Dorn, G. Bohlender, and L. G. Fernandes. Parallel verified linear system solver for uncertain input data. In Proceedings of 20th International Symposium on Computer Architecture and High Performance Computing, pages 89-96, 2008.

[BUT 08] A. Buttari, J. J. Dongarra, J. Kurzak, J. Langou, P. Luszczek, and S. Tomov. The impact of multicore on math software. In Applied Parallel Computing. State of the Art in Scientific Computing, volume 4699, pages 1-10, Berlin, 2008. Springer Lecture Notes in Computer Science.

[CHO 96] J. Choi, J. W. Demmel, I. Dhillon, J. J. Dongarra, S. Ostrouchov, A. P. Petitet, K. Stanley, D. W. Walker, and R. C. Whaley. Scalapack: a portable linear algebra library for distributed memory computers - design issues and performance. Computer Physics Communications, 97(1-2):1-15, 1996.

[LAW 79] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic linear algebra subprograms for fortran usage. ACM Trans. Math. Softw., 5(3):308-323, 1979.

[CHO 95] J. Choi, J. J. Dongarra, S. Ostrouchov, A. P. Petitet, and D. M. H.Walker. Lapack working note 100: A proposal for a set of parallel basic linear algebra subprograms. Technical report, Knoxville, TN, USA, 1995.

[DON 97] J. J. Dongarra and R. C. Whaley. Lapack working note 94: A user's guide to the blacs v1.1. Technical report, Knoxville, TN, USA, 1997.

[BUT 08] A. Buttari, J. J. Dongarra, J. Kurzak, J. Langou, P. Luszczek, and S. Tomov. The impact of multicore on math software. In Applied Parallel Computing. State of the Art in Scientific Computing, volume 4699, pages 1-10, Berlin, 2008. Springer Lecture Notes in Computer Science.

[TOM 10] S. Tomov, R. Nath, H. Ltaief e J. J. Dongarra. Dense Linear Algebra Solvers for Multicore With GPU Accelerators. In: Aceito para o 15th International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS), 2010, 8p.

[KOL 08] M. Kolberg, D. Cordeiro, G. Bohlender, L. G. Fernandes and A. Goldman. A Multithreaded Verified Method for Solving Linear Systems in Dual-Core Processors. In: 9th PARA - International Workshop on Applied Parallel Computing. State of the Art in Scientific Computing, a ser publicado no LNCS, Trondheim (Noruega). PARA 2008 - Revised Selected Papers (LNCS). Heidelberg : Springer Berlin, 2010. v. 6126.

[MIL 10] C. R. Milani, M. L. Kolberg e L. G. Fernandes. Solving Dense Interval Linear Systems With Verified Computing on Multicore Architectures. In: Aceito para o 9th International Meeting of High Performance Computing for Computational Science (VECPAR), 2010, 14p.