Computação Verificada em Agregados de Computadores

Carlos Amaral Hölbig[1]
Dalcidio Moraes Claudio[2]
Tiaraju Asmuz Diverio[3]

Resumo

Até os dias de hoje tem-se buscado uma combinação de software (métodos de inclusão monotônica) com o hardware (aritmética de alta exatidão, matemática intervalar, arredondamentos direcionados, produto escalar ótimo, etc.) para que a tarefa de decidir se o resultado é ou não satisfatório seja transferido para o computador, ou seja, a Computação Verificada. Agora, deseja-se capacitar os novos ambientes, propícios ao processamento paralelo e distribuído, com esta técnica, para que eles resolvam problemas com alta exatidão e alto desempenho. O trabalho abordado nesse artigo visa o desenvolvimento de bibliotecas para a resolução de sistemas de equações lineares densos e esparsos, utilizando a biblioteca C-XSC. Além disso, o trabalho visa a otimização da biblioteca C-XSC para ser utilizada em agregados de computadores.

1 Introdução

Computação Verificada significa o processamento numérico de problemas, utilizando a aritmética de alta exatidão, os métodos intervalares de inclusão e a convergência garantida pelo Teorema de Ponto Fixo de Brouwer (KULISCH; MIRANKER, 1981). Entende-se por aritmética de alta exatidão a aritmética computacional de ponto flutuante baseada no padrão IEEE-754 (IEEE, 1985), acrescida de arredondamentos direcionados, da matemática intervalar e do cálculo do produto escalar e somatórios em registradores especiais, que permitem que valores parciais sejam armazenados sem arredondamentos, resultando que o valor final dessas operações difira do valor real por apenas um arredondamento, vindo daí a máxima exatidão. Os métodos de inclusão trabalham com intervalos. Cada aproximação é um intervalo que contém a solução do problema. As aproximações sucessivas são resultantes da intersecção de

[1]holbig@inf.ufrgs.br
[2]dalcidio@inf.pucrs.br
[3]Apoio: FAPERGS(Cooperação Internacional), CNPq e LabTeC UFRGS/Dell

intervalos, o que garante que tenham um diâmetro menor (ou igual) à aproximação anterior. Esses métodos também identificam a não existência de solução, através de uma aproximação vazia (o resultado não é um intervalo). Esse ramo da Computação Científica ou da Matemática Computacional busca melhorar a qualidade numérica dos cálculos em ponto flutuante em computadores. Como é ilustrado em (1), tem-se um somatório que depende da ordem da soma ou de um produto escalar (o resultado deste somatório é igual a 1).

$$\sum_{i=N}^{-N}(16^i - 16^i) + \sum_{i=N}^{-N}(16^i - 16^i) + 1 \tag{1}$$

Tomando como exemplo os dados apresentados na tabela 1, as aplicações foram resolvidas em máquinas vetoriais que executavam milhares de operações em ponto flutuante por segundo (MFlops). Entretanto, a qualidade numérica foi questionável, como demonstrado em (DIVERIO; FERNANDES; CLAUDIO, 1996; DIVERIO, 1995; ADAMS; KULISCH, 1993).

Tabela 1: Resultados para N = 29

Somatório	**Modo Escalar**	**Modo Vetorial**
S0	0.000000E+00	0.2951479E+21
S23	0.000000E+00	0.1152922E+19
S119	0.000000E+00	0.0000000E+00
S122	0.000000E+00	- 0.2951479E+21
S153	0.000000E+00	0.1000000E+01
S154	0.100000E+01	0.1000000E+01
S236	0.100000E+01	0.2951479E+21

Essa busca da qualidade numérica dos cálculos torna-se mais crucial em aplicações de larga escala de computação, as quais necessitam da realização de uma grande quantidade de operações em ponto flutuante (DIVERIO; FERNANDES; CLAUDIO, 1996). Com as novas tecnologias de intercomunicação de redes, foi possível a construção de máquinas baratas mas com grande poder computacional, os agregados de computadores, também conhecidos como Clusters. Qual a qualidade numérica dos cálculos executados nessas máquinas? Será isso importante para a área de Processamento de Alto Desempenho? Essas são questões que esta pesquisa procurará responder no transcorrer de seu desenvolvimento.

2 O Ambiente Computacional

Este trabalho está desenvolvendo ferramentas computacionais (software) utilizando o Cluster LabTeC do II-UFRGS e a biblioteca C-XSC (descrita em detalhes em (HAMMER, 1995; HOFSCHUSTER; KRÄMER, 2001)).

O C-XSC é baseado na linguagem C-ANSI e é implementado como uma biblioteca numérica da linguagem C++. O C-XSC torna o computador mais poderoso aritmeticamente e simplifica significantemente a programação. Ele consiste de um sistema *runtime* escrito em C, incluindo produto escalar ótimo e muitos tipos de dados pré-definidos para elementos mais comumente usados em espaços vetoriais, tais como números reais e complexos, vetores e matrizes. Operadores para elementos desses tipos são pré-definidos e podem ser chamados pelos seus símbolos de operadores usuais. Assim, expressões aritméticas e algoritmos numéricos são expressos em uma notação que é muito similar à notação matemática usual. Todos os operadores numéricos pré-definidos são de alta exatidão, ou seja, o resultado computado difere do resultado correto por apenas um arredondamento. A ênfase do C-XSC é mais na exatidão e na confiabilidade do resultado do que na velocidade da obtenção do mesmo. O ambiente de programação do C-XSC é facilmente portável para qualquer computador que suporte um Compilador C++ padrão. Além do C-XSC, existem versões "XSC"para o Pascal e o Fortran. O C-XSC possui ainda, entre suas características, aritmética intervalar, aritmética complexa, aritmética intervalar complexa e as correspondentes aritméticas de vetores e matrizes. Encontra-se também no C-XSC, módulos para a resolução de problemas numéricos, tais como: sistemas de equações lineares e não lineares, inversão de matrizes, autovalores e autovetores, avaliação de expressões aritméticas e muitos outros.

3 Ferramentas Computacionais em Desenvolvimento

Como os objetivos desta pesquisa são o desenvolvimento de *solvers* com alta exatidão para a resolução de sistemas de equações lineares em agregados de computadores e a otimização da biblioteca C-XSC nestes ambientes, primeiramente foi necessário realizar a integração da biblioteca C-XSC com a biblioteca MPICH 1.2.2 (o cluster utilizado nesta integracao foi o cluster LabTeC do II-UFRGS). Com essa integração, buscou-se reunir alta exatidão com a paralelização resultante do uso da divisão de tarefas entre os diversos nodos disponíveis no cluster, sendo que todos executaram as mesmas tarefas e a comunicação entre os nodos e entre nodos e o servidor se deu através de troca de mensagens. Medições e testes foram efetuados para comparar o tempo de execução de rotinas em C, em C com MPI, em C-XSC e em C-XSC com MPI. Nos testes realizados pode-se observar que pequenas e simples modificações em algoritmos tradicionais podem trazer bom ganho de desempenho e que a forma de uso do pipeline do processador é definitivo para o resultado obtido. Nestes testes preliminares foi notado que a biblioteca C-XSC necessita ser otimizada para tornar-se eficiente em um ambiente de alto desempenho (até o momento, o principal objetivo do C-XSC foi a funcionalidade e portabilidade e não a velocidade).

Em conjunto com essa integração foram desenvolvidos as versões iniciais de *solvers* para a resolução de sistemas de equações lineares com matrizes densas e esparsas. A descrição completa dos algoritmos implementados nestes solvers pode ser encontrada em (KRÄMER; KULISCH; LOHNER, 1994). O *solver* para sistemas de equações lineares $Ax = b$, com matrizes densas $n \times m$, soluciona sistemas quadrados ($m = n$), sobre-determinados ($m > n$) e sub-determinados ($m < n$). Já o algoritmo implementado no *solver* para sistemas esparsos (em especial para matrizes do tipo banda) foi baseado no estreito relacionamento existente entre as matrizes com estrutura banda e equações diferenciais. De acordo com esse relacionamento, as equações diferenciais podem ser reescritas equivalentemente como um sistema linear triangular com matrizes bandas. Similarmente, pode-se reescrever um sistema triangular com matrizes banda como equações diferenciais. O sistema (2) é equivalente à equação diferencial $a_{i,i-m+1}x_{i-m+1} + \cdots + a_{i,i}x_i = b_i$, $i = m, \ldots, n$ de ordem $m-1$ com os valores iniciais $x_i = (b_i - a_{i,i-1}x_{i-1} - \cdots - a_{i,1}x_1)/a_{i,i}$, $i = 1, \ldots, m-1$.

$$Ax = \begin{pmatrix} a_{1,1} & & & & 0 \\ \vdots & \ddots & & & \\ a_{m,1} & & \ddots & & \\ & \ddots & & \ddots & \\ 0 & & a_{n,n-m+1} & \cdots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = b \tag{2}$$

Como apresentado em (KRÄMER; KULISCH; LOHNER, 1994), a solução de sistemas triangulares por retro-substituição intervalar pode resultar em sobre-determinações devido ao "*wrapping effect*"(NEUMAIER, 1993). Com o objetivo de minimizar esse efeito, usou-se a resolução de sistemas triangulares com matrizes banda através da resolução de equações diferenciais. Para sistemas com matrizes bandas em geral aplica-se a decomposição *LU* sem pivotamento para a matriz de coeficientes *A*, gerando uma iteração intervalar similar a (3).

$$[y]_{k+1} = R\Diamond(b - A\tilde{x}) + \Diamond(I - RA)[y]_k \tag{3}$$

Aqui, entretanto, não usou-se a completa aproximação da inversa R mas sim, essa aproximação foi realizada pela solução de dois sistemas lineares com matrizes triangulares bandas (L and U). Diferentes abordagens para soluções verificadas de sistemas lineares com grandes matrizes bandas ou com matrizes esparsas com coeficientes arbitrários poderão ser encontradas em (RUMP, 1993).

4 Testes

Para ilustrar a qualidade numérica dos *solvers* que estão em desenvolvimento neste trabalho e, que são utilizados para a resolução de sistemas de equações lineares densas e esparsas, apresenta-se aqui um exemplo onde calculou-se a inclusão para um sistema linear esparso de grande porte (ordem $n = 200.000$). Pegou-se a matriz simétrica Toeplitz com 5 bandas com os valores $1, 2, 4, 2, 1$ e definiu-se todos componentes de b iguais a 1. Então o programa produziu a seguinte saída para esse sistema (somente os dez primeiros e os dez últimos componentes da solução foram apresentados):

```
Dimensão  n = 200000
tamanho bandas l,k : 2 2
A = 1 2 4 2 1
troca elementos ? (s/n) n
b = =1
troca elementos ? (s/n) n
x =
  1: [  1.860146067479180E-001,  1.860146067479181E-001 ]
  2: [  9.037859550210300E-002,  9.037859550210302E-002 ]
  3: [  7.518438200412189E-002,  7.518438200412191E-002 ]
  4: [  1.160876404875081E-001,  1.160876404875082E-001 ]
  5: [  1.003153932563721E-001,  1.003153932563722E-001 ]
  6: [  9.427129202687645E-002,  9.427129202687647E-002 ]
  7: [  1.028361799416204E-001,  1.028361799416205E-001 ]
  8: [  1.005240450090008E-001,  1.005240450090009E-001 ]
  9: [  9.874921290539136E-002,  9.874921290539138E-002 ]
 10: [  1.004617422430963E-001,  1.004617422430964E-001 ]

199990: [  1.001953939326196E-001,  1.001953939326197E-001 ]
199991: [  1.004617422430963E-001,  1.004617422430964E-001 ]
199992: [  9.874921290539136E-002,  9.874921290539138E-002 ]
199993: [  1.005240450090008E-001,  1.005240450090009E-001 ]
199994: [  1.028361799416204E-001,  1.028361799416205E-001 ]
199995: [  9.427129202687645E-002,  9.427129202687647E-002 ]
199996: [  1.003153932563721E-001,  1.003153932563722E-001 ]
199997: [  1.160876404875081E-001,  1.160876404875082E-001 ]
199998: [  7.518438200412189E-002,  7.518438200412191E-002 ]
199999: [  9.037859550210300E-002,  9.037859550210302E-002 ]
200000: [  1.860146067479180E-001,  1.860146067479181E-001 ]

max. rel. error =  1.845833860422451E-016   em i = 3
max. abs. error =  2.775557561562891E-017   em i = 1
min. abs. x[3] = [ 7.518438200412189E-002,  7.518438200412191E-002 ]
max. abs. x[1] = [ 1.860146067479180E-001,  1.860146067479181E-001 ]
```

5 Conclusões e Trabalhos Futuros

Com o objetivo de tentar achar soluções para as perguntas feitas neste artigo, é que está em desenvolvimento essa pesquisa que objetiva disponibilizar ferramentas computacionais com alta exatidão em agregados de computadores. A integração inicial do C-XSC com o MPI já foi realizada, bem como o desenvolvimento de versões iniciais de *solvers* para a resolução de sistemas de equações lineares. Atualmente, o trabalho está voltado para o estudo de como realizar uma eficiente paralelização dos métodos intervalares já implementados, além do estudo, desenvolvimento e implementação de novos métodos. Juntamente com esses estudos, está se trabalhando nas tarefas de como realizar a otimização da biblioteca C-XSC para que ela possa ser utilizada, de maneita eficiente, em agregados de computadores.

Referências

ADAMS, E.; KULISCH, U. *Scientific Computing with Automatic Result Verification*. San Diego: Academic Press, 1993.

DIVERIO, T. A. *Uso Efetivo da Matemática Intervalar em Supercomputadores Vetoriais*. Tese (Doutorado) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 1995.

DIVERIO, T. A.; FERNANDES, Ú. A. L.; CLAUDIO, D. M. Errors in vector processing and the library libavi.a. *Reliable Computing*, Springer-Verlag, New York, v. 2, n. 2, p. 103–109, 1996.

HAMMER, R. et al. *C-XSC Toolbox for Verified Computing I: basic numerical problems*. New York: Springer-Verlag, 1995.

HOFSCHUSTER, W.; KRÄMER, W. *C-XSC 2.0: A C++ Class Library for Extended Scientific Computing*. Wuppertal, Germany, 2001.

HÖLBIG, C. et al. High performance with high accuracy laboratory. *Revista de Informatica Teórica e Aplicada*, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, v. 3, n. 2, p. 35–53, 1997.

IEEE. 1985. IEEE Standart 754-1985 for Binary Floating-Point Arithmetic, IEEE 754. New York, 1985.

KRÄMER, W.; GUDENBERG, J. Wolff v. *Scientific Computing, Validated Numerics, Interval Methods*. London: Kluwer Academic Publishers, 2001.

KRÄMER, W.; KULISCH, U.; LOHNER, R. *Numerical Toolbox for Verified Computing II - Advanced Numerical Problems*. Karlsruhe, Germany, 1994. Disponível em: <http://www.uni-karlsruhe.de/Rudolf.Lohner/papers/tb2.ps.gz>.

KULISCH, U.; MIRANKER, W. *Computer Arithmetc in Theory and Practice*. New York: Academic Press, 1981.

NEUMAIER, A. The wrapping effect, ellipsoid arithmetic, stability and confidence regions. *Computing Supplementum*, Springer-Verlag, New York, v. 9, p. 175–190, 1993.

RUMP, S. M. Validated solution of large linear systems. *Computing Supplementum*, Springer-Verlag, New York, v. 9, p. 191–212, 1993.