Análise da Integração entre as Bibliotecas C-XSC e MPICH

Paulo Sérgio Morandi Júnior*, Carlos Amaral Hölbig,
Tiaraju Asmuz Diverio

Universidade Federal do Rio Grande do Sul
Instituto de Informática e PPGC
Av. Bento Gonçalves, 9500 - Caixa Postal 15064 - CEP. 90501-970
Porto Alegre, RS

Universidade de Passo Fundo
Curso de Ciência da Computação - ICEG
Campus 1 - BR-285 - CEP. 99001-970
Passo Fundo, RS

{sergio,holbig,diverio}@inf.ufrgs.br

Introdução

É crescente a utilização de agregados de computadores(*Cluster*). Porém a preocupação com a qualidade do resultado nem sempre é observada com a devida atenção. Na preocupação em prover uma ferramenta para auxiliar nos cálculos e na exatidão dos mesmos, criou-se uma biblioteca chamada *C-XSC* ([KLA 93]). O C-XSC mostrou-se uma excelente ferramenta de suporte à Computação Verificada ([HÖL 2002] e [HÖL 2004]). No cenário dos *Clusters* encontra-se a biblioteca de troca de mensagens *MPICH* ([FOR 94]), que implementa o padrão *MPI* e fornece todo um conjuto de primitivas de suporte a comunicação inter-nodos. Logo, de posse dessas duas ferramentas e de um ambiente de alto desempenho, procurou-se aliar alta exatidão (fornecido pelo C-XSC) e o alto desempenho (com MPICH). Após comprovada essa viabilidade ([MOR 2004]), partiu-se para o desenvolvimento de testes que procurassem validar essa integração. O objetivo deste artigo é mostrar os testes realizados após a integração, destacando os resultados comparativos entre a computação de alto desempenho com e sem a alta exatidão do C-XSC.

Descrição dos Testes

O objetivo dos testes implementados nesta pesquisa foi de verificar e exatidão e o desempenho acarretados com a inclusão da alta exatidão nos programas (seqüênciais e paralelos) executados no cluster labtec da UFRGS. Com isso pode-se analisar a viabilidade de se utilizar a biblioteca C-XSC, de maneira eficiente, neste tipo de ambiente. Os testes realizados abordaram a multiplicação de matrizes e o cálculo do produto escalar.

*CNPq-UFRGS/PIBIC

Dimensão	Programa	$A \times B$
256	C/C++	-1.10000000000000008881784197001252323389053
	C-XSC	9.59999999999994315658113919198513031005859
512	C/C++	-1.10000000000000008881784197001252323389053
	C-XSC	19.19999999999998863131622783839702606201172
1024	C/C++	-1.10000000000000008881784197001252323389053
	C-XSC	38.39999999999997726263245567679405212402344

Tabela 1: Mutliplicação de Matrizes – resultados

Multiplicação de Matrizes

Nos testes foi implementado a multiplicação convencional em versão seqüêncial e paralela. A versão seqüêncial trata-se da implementação tradicional dos três laços e o resultado (em segundos) pode ser verificado na tabela 2. A versão paralela seguiu um algoritmo de mestre e escravo, onde o mestre divide e distribui a matriz para os escravos (em linha). Por exemplo, se deseja-se multiplicar A por B, a matriz A é divida entre os escravos e a matriz B é enviada a todos os processos por *broadcast*. Foram utilizados 1 mestre e 7 escravos. Caso a divisão da matriz não seja exata, o mestre fica com a sobra. Após o cálculo de cada produto parcial, o mesmo é enviado para o mestre que monta a matriz resultante. As matrizes foram inicializadas da seguinte forma (considerando matrizes de dimensão $4x4$):

$$C = \begin{pmatrix} 10^{50} & 1.25 & 10^{50} & 1.1 \\ 10^{50} & 1.25 & 10^{50} & 1.1 \\ 10^{50} & 1.25 & 10^{50} & 1.1 \\ 10^{50} & 1.25 & 10^{50} & 1.1 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{pmatrix}$$

Analisando a operação acima, percebe-se que, na matriz resultante (C), cada um dos elementos será $4 \times (10^{50} + 1.25 - 10^{50} - 1.1) = 4 \times 0,15 = 0,6$. Generalizando esse cálculo, como 10^{50} aparece duas vezes, tem-se que $1.25 - 1.1$ ocorre em $1/4$ de N nos cálculos, então o resultado esperado para uma multiplicação de uma matriz $N \times N$ é

$$\frac{N}{4} \times 0.15 \tag{1}$$

considerando N múltiplo de 4, ou seja, cada um dos elementos da matriz resultade será $\frac{N}{4} \times 0.15$. Portanto, segundo 1, o resultado exato esperado para 256 é uma matriz preenchida com 9.6, para 512 é 19.2, para 1024 é 38.4 e para 2048 é 76.8. Os resultados encontrados nos cálculos podem ser encontrados na tabela 1. Tanto a versão paralela quanto a seqüêncial apresentaram a mesma exatidão, como era de se esperar. Para esses testes foram utilizadas matrizes quadradas de dimensões 512, 1024 e 2048. Foram realizados 50 execuções da versão seqüêncial e 10 execuções das versões paralelas, e o resultado, em relação à média do tempo de execução, é apresentado nas tabelas 2 e 3, respectivamente.

Programa/Dimensão	256	512	1024
CXSC	9.717248	88.819302	708.453258
C	1.567539	18.743103	150.464266

Tabela 2: Multiplicação de Matrizes em Seqüêncial (tempos em segundos)

Programa/Dimensão	512	1024	2048
CXSC	121.58	960.34	7684.27
C	4.59	34.08	265.32

Tabela 3: Multiplicação de Matrizes em Paralelo (tempos em segundos)

Produto Escalar

O produto escalar seqüêncial foi idealizado da mesma forma que a multiplicação de matrizes. Logo, os resultados obtidos, por exemplo, para um produto escalar entre dois vetores de dimensão 20000 é $(20000/4) \times 0,15 = 750$, e assim por diante. Por exemplo, considerando dois vetores de dimensão 4:

$$\left[10^{50} \;\; 1.25 \;\; 10^{50} \;\; 1.1\right] . [1 \;\; 1 \;\; -1 \;\; -1] = 0.15$$

Os resultados obtidos são apresentados na tabela 4.

No caso da versão paralela também foi utilizado o modelo mestre-escravo, onde o mestre inicializa e distribui os vetores de forma igual a todos os escravos. Os escravos, por sua vez, recebem esses pedaços de vetores, calculam os produtos parcias e devolvem o resultado parcial para o mestre. O mestre junta esses pedaços e devolve o resultado. Vale lembrar que, no caso do C-XSC, o que ocorre é um produto escalar ótimo, ou seja, ocorre apenas um arredondamento em todo o cálculo realizado. Foram utilizados 1 mestre e 3 escravos, e os testes foram executados 50 vezes (tanto seqüêncial quanto parelelo). A tabela 5 apresenta as médias dos tempos (em segundos) da versão paralela.

Conclusões e Trabalhos Futuros

A análise dessas tabelas mostram que o C-XSC não foi muito eficiente em relação ao desempenho, principalmente em relação à multiplicação de matrizes. Porém, os resul-

Dimensão	Linguagem	$a \times b$
1000	C/C++	-1.10000000000000008881784197001252323389053 3
	C-XSC	3.74999999999999777955395074968691915273666 4
20000	C/C++	-1.10000000000000008881784197001252323389053 3
	C-XSC	749.99999999999954525264911353588104248046 88
30000	C/C++	-1.10000000000000008881784197001252323389053 3
	C-XSC	1124.99999999999931787897367030382156372070 3

Tabela 4: Produto Escalar seqüêncial – resultados

Programa/Dimensão	30000	90000	180000
C-XSC	0.041015	0.113847	0.225083
C/C++	0.031205	0.096727	0.188670

Tabela 5: Produto Escalar em Paralelo (tempos em segundos)

Programa/Dimensão	1000	20000	30000
C-XSC	4613.44	9165.5	13742.04
C/C++	153.26	314.72	480.32

Tabela 6: Produto Escalar em Seqüêncial (tempos em μsegundos)

tados obtidos em termos de exatidão foram totalmente superiores ao C padrão, mostrando a qualidade do C-XSC em manipular dados numéricos. Considerando que o objetivo principal da biblioteca é a qualidade numérica do resultado e analisando a tabela 5, percebe-se que o C-XSC não ficou em muita desvantagem. Como trabalhos futuros está a finalização da implementação do método Gradiente Conjugado em paralelo e com alta exatidão, para uma futura integração do mesmo com o modelo *HIDRA*, um modelo computacional que simula a vazão de águas e a propagação da poluição no lago Rio Guaíba da cidade de Porto Alegre (RS). Também como trabalhos futuros, para uma outra etapa dessa pesquisa, a realização da otimização das rotinas do C-XSC visando tornar a biblioteca mais eficiente (em termos de desempenho) para esse tipo de ambiente computacional.

Referências

[FOR 94] FORUM, M. **The mpi message passing interface standard**. Knoxville: University of Tennesse, 1994.

[HÖL 2002] HÖLBIG, C. A. et al. Automatic result verification in the environment of high performance computing. In: IMACS/GAMM INTERNATIONAL SYMPOSIUM ON SCIENTIFIC COMPUTING, COMPUTER ARITHMETIC AND VALIDATED NUMERICS, 2002. **Anais...** Extended abstracts, 2002. p.54–55.

[HÖL 2004] HÖLBIG, C. A. et al. Solving linear systems on cluster computers with high accuracy. In: WORKSHOP ON STATE-OF-THE-ART IN SCIENTIFIC COMPUTING, PARA´2004, 2004, Copenhagem. **Anais...** [S.l.: s.n.], 2004.

[KLA 93] KLATTE, R. et al. **C-xsc - a c++ class library for extended scientific computing**. New York: Springer-Verlag, 1993.

[MOR 2004] MORANDI JÚNIOR, P. S. et al. A integração da biblioteca de alta exatidão c-xsc em agregados de computadores. **Revista Eletrônica de Informática**, http://www.sbc.org.br, v.Ano IV, n.II, 2004.