Análise de Desempenho em Cluster *Beowulf* Usando MPI

Renê Mikami, Luciano Drosda M. dos Santos,
Ana Cristina B. Kochem Vendramin, Celso Antônio A. Kaestner

Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná (UTFPR) – Curitiba, PR – Brasil

{rene0907, lucianodrosda}@gmail.com , {cristina, kaestner}@dainf.ct.utfpr.edu.br

Abstract. *The aim of this paper is to present a performance evaluation of a Beowulf cluster in a matrix computational task. In order to accomplish this task we implement a matrix multiplication program using the MPI library. The studies were conducted in order to analyze performance benefits obtained when the number of cluster nodes is augmented, and also considering an increment in the involved matrix orders. Obtained results show that for larger matrix orders the computational gain is significant.*

Resumo. *O objetivo deste artigo é apresentar uma avaliação de desempenho de um cluster do tipo Beowulf numa tarefa que envolve matrizes. Para tanto foi desenvolvido um programa para multiplicação de matrizes que utiliza a biblioteca MPI. Os estudos foram conduzidos para analisar a melhoria obtida na performance em função do aumento do número de máquinas do cluster, considerando matrizes de ordens crescentes. Os resultados obtidos mostram que para matrizes de grandes ordens o ganho computacional é significativo.*

1. Introdução

Inúmeros problemas encontrados no mundo real exigem elevada capacidade computacional na sua solução. Com o avanço da tecnologia criaram-se computadores com altíssima velocidade de processamento e grande capacidade de memória, empregados em pesquisas científicas e militares. Estes equipamentos, porém, possuem custo elevado. Os avanços na área de sistemas computacionais distribuídos permitiram o surgimento de uma alternativa de baixo custo para estes problemas: os *clusters* de computadores [Pitanga 2002].

Um *cluster* é formado por um conjunto de computadores convencionais interconectados por uma rede de alta velocidade, sob ação de um sistema operacional distribuído, de forma a trabalhar como uma única máquina de grande porte. Para melhorar a velocidade de resposta de um determinado processamento, o paralelismo é amplamente utilizado. Tarefas grandes e complexas são divididas em sub-tarefas que são distribuídas entre os processadores do cluster, que as executam simultaneamente, com o intuito de obter rapidamente uma solução para o problema global.

Além do baixo custo e do alto poder de processamento, existem outras vantagens no uso de *clusters*, tais como: (a) alta escalabilidade, pois é possível adicionar máquinas ao *cluster*; (b) tolerância a falhas, pois o sistema não é totalmente comprometido quando algum computador falha; e (c) heterogeneidade, pois é possível a utilização de diferentes configurações de *hardware* nos componentes [Pitanga 2002].

Esta alternativa apresenta uma contrapartida: a necessidade de se adaptar os algoritmos, que antes funcionavam em sistemas monoprocessados, para os sistemas distribuídos e com vários processadores que caracterizam o *cluster* [Dietz 1998].

Este trabalho apresenta um estudo de avaliação de desempenho em um *cluster* que emprega ambiente de programação paralela, sobre o problema da multiplicação matricial. Resultados sobre o desempenho são obtidos a partir do incremento gradual do número de componentes do *cluster*, a partir de um programa desenvolvido para a distribuição da tarefa entre várias máquinas.

2. Programação Paralela

Algumas ferramentas surgiram para auxiliar na implementação de algoritmos e programas em sistemas paralelos. Dentre elas, podem-se citar as bibliotecas PVM (*Parallel Virtual Machine*) [PVM 1994] e MPI (*Message Passing Interface*) [MPI 2008]. Ambas possuem rotinas específicas para permitir a troca de mensagens entre os processadores em ambiente de memória distribuída [Cáceres, Mongelli e Song 2001].

A PVM permite a junção de várias estações de trabalho, que podem ser heterogêneas, formando uma única máquina paralela virtual. Ela trata de forma transparente o roteamento das mensagens, a conversão de dados e o escalonamento das tarefas em uma rede de computadores. Possui uma interface de programação onde é possível utilizar bibliotecas com rotinas que permitem a inicialização e o término de atividades na rede, além da comunicação e sincronização entre tarefas [PVM 1994].

O MPI é uma biblioteca que possui diversas rotinas para a troca de mensagens entre processadores de um *cluster*. As funções da biblioteca podem ser utilizadas a partir das linguagens C, C++ e Fortran. As tarefas não são gerenciadas em relação aos processadores. Cabe ao programador definir qual tarefa deve ser executada em cada processador [MPI 2008].

Neste trabalho utilizou-se um *cluster* que funciona sob ambiente operacional obtido com o uso da biblioteca MPI. Esta opção foi adotada pois as funções em MPI para troca de mensagens possuem melhor desempenho que as equivalentes em PVM, quando o *cluster* possui máquinas semelhantes em *hardware* e *software* [Elts 2004] [Geist, Kohl e Papadopoulos 1996].

3. O *Cluster Beowulf*

Para a obtenção de um ambiente computacional de alto desempenho neste trabalho empregou-se um *cluster* do tipo *Beowulf*. *Clusters Beowulf* são amplamente empregados por serem formados a partir de computadores comuns – e muitas vezes obsoletos – interligados por tecnologias de redes simples e instrumentalizados com sistemas operacionais *open-sources* [Beowulf 2008] [Gropp, Lusk e Sterling 2003].

O *cluster* é composto por onze máquinas com processador Intel Pentium D com 2.8 GHz de clock e 1GB de memória RAM, sob sistema operacional Ubuntu Server 8.04. Um procedimento mestre, alocado em uma das máquinas, atua como responsável por coordenar todas as funções do *cluster* tais como a distribuição das tarefas entre o restante dos nós.

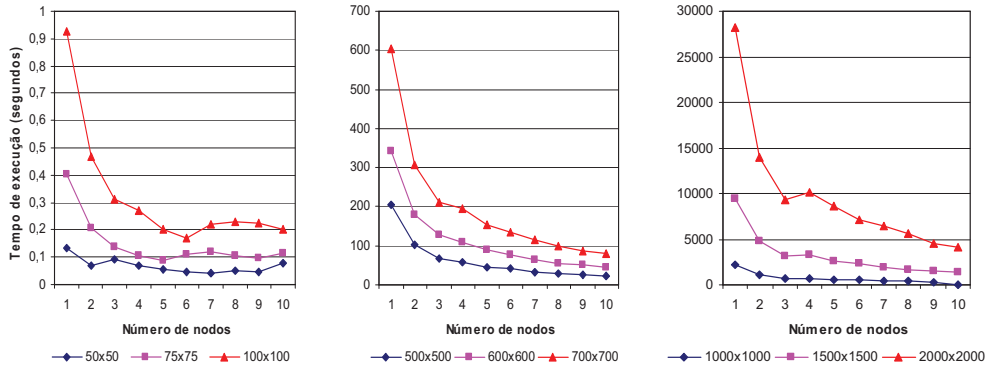
4. Desenvolvimento do Programa

Para realizar a análise de desempenho do cluster foi desenvolvido um programa que realiza a multiplicação de duas matrizes e armazena o respectivo tempo de execução. Neste programa cada posição da matriz resultante é obtida em um nó do *cluster*, sendo que os resultados são centralizados no mestre para a apresentação do resultado final.

O programa desenvolvido segue os seguintes passos: (1) A decomposição é feita seguindo-se as linhas da primeira matriz e as colunas da segunda matriz; (2) Cada elemento a calcular da matriz resposta está associado a um nó específico; (3) Em cada nó do *cluster* todas as posições da matriz são percorridas; caso a posição do elemento a calcular corresponda ao número do nó, o elemento correspondente é calculado pela fórmula usual da multiplicação matricial; caso não haja correspondência, o nó simplesmente ignora esta posição; (4) Realizada a multiplicação, o nó envia o resultado do elemento correspondente ao mestre; (5) O mestre recebe os resultados dos elementos individuais e os insere na devida posição da matriz resultante, de forma a obter o resultado final.

5. Resultados

O programa que multiplica matrizes foi executado utilizando matrizes de diversas dimensões, com elementos de valores aleatórios.



(a) **(b)** **(c)**

Figura 1: Desempenho obtido ao multiplicar matrizes de pequenas (a), médias (b) e grande (c) dimensões.

Nota-se que a utilização de mais de uma máquina torna o processamento significativamente mais rápido. No entanto, com matrizes de dimensões iguais ou menores que 100x100, não há ganho significativo em relação ao desempenho a partir do uso de 5 nodos, conforme ilustra a Figura 1(a), pois o tempo de execução se estabiliza em torno de 0,2 segundos. Isso ocorre devido à limitação por parte da comunicação de dados, que passa a ser especialmente relevante quando o tempo de processamento se torna muito baixo.

Nos testes realizados com matrizes quadradas de dimensões médias e grandes, como ilustram as Figuras 1(b) e 1(c), houve um ganho considerável de desempenho. A cada nó adicionado, obteve-se a redução do tempo de processamento. Nota-se que para

processamentos mais pesados, o custo da limitação por parte da comunicação torna-se irrelevante.

6. Conclusões e Trabalhos Futuros

Este trabalho teve como principal objetivo avaliar o desempenho de um *cluster* sobre a tarefa de multiplicação de matrizes. Utilizou-se um *cluster Beowulf* formado por onze computadores convencionais. Para realizar os testes foi desenvolvido um programa em linguagem C que emprega chamadas à biblioteca MPI para efetuar a troca de mensagens entre os nós do *cluster*. Este programa retorna o tempo utilizado para sua execução, de forma a se avaliar o tempo necessário para efetuar a multiplicação de matrizes.

Foram efetuados testes com matrizes de diversas dimensões. Notou-se que com matrizes menores houve um pequeno ganho de desempenho no tempo de processamento para obtenção da resposta global; no entanto, este ganho foi limitado pelo *overhead* de comunicação. Para multiplicação entre matrizes com dimensões médias e grandes sempre houve ganho de desempenho, diminuindo o tempo de processamento à medida que se adicionavam nós ao *cluster*. Assim, conclui-se que o uso de *clusters* é uma alternativa interessante e eficaz em aplicações computacionalmente intensivas.

Agradecimentos

Este trabalho foi desenvolvido no âmbito de convênio entre a empresa Nova Soluções em Informática S/A e a Fundação de Apoio à Educação, Pesquisa e Desenvolvimento Científico e Tecnológico da Universidade Tecnológica Federal do Paraná.

Referências

Beowulf (2008) "Beowulf Project", http://www.beowulf.org. Acesso em 21/Nov/2008.

Cáceres, Edson Norberto, Mogelli, Henrique, Song, Siang Wun (2001) "Algoritmos Paralelos usando CGM/PVM/MPI: uma Introdução", XXI Congresso da Sociedade Brasileira da Computação, Jornada de Atualização em Informática, pp. 219-278.

Dietz, Hank (1998) "Linux Parallel Processing HOWTO", http://cobweb.ecn.purdue.edu/~pplinux/PPHOWTO/pphowto.html. Acesso em 17/nov/2008.

Elts, Ekaterina (2004) "Comparative Analysis and PVM and MPI for the development of physical applications on parallel clusters", Saint-Petersburg State University.

Geist, G. A., Kohl, J. A., Papadopoulos, P. M. (1996) "PVM and MPI: a Comparison of Features", Calculateurs Paralleles, 1996, vol. 8, no 2, pp. 137-150.

Gropp, William, Lusk, Ewing, Sterling, Thomas. (2003) "Beowulf Cluster Computing with Linux", 2 ed., Cambridge, MA: MIT Press.

MPI (2008) "MPI: A Message-Passing Interface Standard Version 2.1", http://www-unix.ms.anl.gov/mpi, Acesso em 21/nov/2008.

Pitanga, Marcos José (2002) "Construindo supercomputadores com Linux", Rio de Janeiro: Brasport.

PVM (2007) "PVM: Parallel Virtual Machine", http://www.csm.ornl.gov/pvm, Acesso em 21/nov/2008.