VII CONNEPI
Congresso Norte Nordeste de Pesquisa e Inovação
PALMAS . TOCANTINS . 2012
19 a 21 de outubro - Ciência, Tecnologia e Inovação: Ações Sustentáveis para o desenvolvimento regional

Implementação de um *Cluster* Computacional para Resolução de Algoritmos Paralelizáveis

Dhiogo de Sá Barreto Morais[1], Bruno Silvério Costa[2], Camilo Alves Carvalho[2]

[1]Graduando do Curso de Bacharelado em Sistemas de Informação do Instituto Fedral da Bahia – IFBA / Campus Vitória da Conquista – Bolsista do IFBA – e-mail: dhiogo-morais@hotmail.com
[2]Professores do Instituto Federal da Bahia – IFBA – Campus Vitória da Conquista – e-mail: {brunosilverio, camilocarvalho}@ifba.edu.br

Resumo: *Cluster* Computacional é um grupamento de micro-computadores individuais que tem suas capacidades de processamento combinadas em função da resolução de um problema algorítmico com características paralelizáveis. Estes problemas são implementados por meio de alguma linguagem de programação se utilizando de bibliotecas específicas, tais como as derivadas da MPI (*Message Passing Interface*), que permitem que os mesmos sejam fragmentados em partes e executados em separado (paralelamente) em diferentes processadores. Este trabalho visa implementar um *cluster* computacional para viabilizar a solução deste tipo de problema algorítmico.

Palavras-chave: cluster, paralelismo, mpi, mpich-2

1. INTRODUÇÃO

Avanços científicos e inovações em hardware e software permitiram aumento exponencial no desempenho dos sistemas computacionais nos últimos 40 anos. Em grande parte, esta melhoria no desempenho foi propiciada pelo que se convencionou denominar de "Lei de Moore" (MOORE, 1965), ou a capacidade da indústria de dobrar a cada dois anos (aproximadamente)número de dispositivos (transistores) que podem ser colocados em um circuito integrado (CI) (HENESSY & PATTERSON, 2006).

Até recentemente, a tendência na indústria de processadores era empregar transistores adicionais na implementação de CIs contendo sistemas (processador, seus caches, etc.) com um único processador cada vez mais poderoso. No entanto, três obstáculos se tornaram aparentemente grandes demais para a continuidade desta tendência: (i) o consumo de energia e a conseqüente necessidade da dissipação do calor proveniente do chaveamento em alta freqüência de um número cada vez maior de transistores (*the Power Wall* (ASANOVIC et alii, 2006)); (ii) a crescente latência da hierarquia de memória (*the Memory Wall* (WULF & McKEE, 1995)); e (iii) as dificuldades associadas à uma maior exploração do paralelismo no nível de instrução (*the ILP Wall* (ASANOVIC et alii, 2006)). David Patterson sumarizou estes três problemas na expressão: "*the Power Wall*" + "*the Memory Wall*" + "*the ILP Wall*" = "*the Brick Wall for serial performance*" (ASANOVIC et alii, 2006).

Em face dos obstáculos mencionados, uma das soluções propostas consistia em explorar as possibilidades oferecidas por uma tecnologia que se popularizou a partir da década de 1970: as redes de computadores. Uma infra-estrutura de rede permite a comunicação entre diversos computadores autônomos, compartilhando recursos (TANENBAUM, 2003). Um dos mais nobres recursos que podem ser disponibilizados é justamente o processador. Aplicações que fossem desenvolvidas para dividir sua carga de processamento e depois combinar seus resultados teria um uso muito mais diversificado de processamento disponível na rede (PITANGA, 2004).

Desta forma, agrupamentos (*clusters*) de computadores passaram a ser montados de forma a oferecer demanda computacional para aplicações que exigissem mais que uma plataforma serial – ou mesmo uma plataforma com múltiplos processadores, levando em consideração a vantagem da escalabilidade (PITANGA, 2004).

De posse de um sistema computacional que atenda a uma elevada demanda de processamento, os algoritmos mais complexos em termos de uso deste recurso ganharam mais possibilidades de execução. Os mais comuns são os cálculos realizados sobre matrizes, presentes em qualquer tipo de trabalho sobre Computação Gráfica.

ISBN 978-85-62830-10-5

2. MATERIAL E MÉTODOS

A implementação de um *cluster* computacional faz uso de micro-computadores individuais conectados por uma rede de comunicação. O arranjo deste grupamento obedece uma hierarquia onde haverão os nós de execução (*nodes* ou *slaves*) e um nó de controle (*master* ou *controller*). O nó de controle é o responsável por controlar a presença dos nós de execução, localizando-os neste segmento da rede. Também é por meio dele que os processos (*jobs*) que serão executados serão submetidos, indicando como parâmetro a quantidade de nós de execução serão empregados nesta trabalho. Por fim, ele possui a responsabilidade de oferecer uma interface de comunicação entre o *cluster* e o "mundo exterior", ou seja, é por meio dele que os usuários interagem com o grupamento, não necessariamente em contato direto – a utilização de terminais remotos é recomendada.

Neste projeto foi utilizado um total de 20 micros, sendo destes 16 nós de execução e 1 nó de controle, restando 3 micros sobressalentes, para substituição em caso de falha de *hardware*. Estes computadores foram proveniente de renovação dos equipamentos de outros laboratórios de informática do Instituto, por isso tratam-se de máquinas mais antigas, e com poder de processamento mais modesto que as atuais disponíveis no mercado. Para interligá-las em rede foi utilizado um *switch* 10/100 Mbps de 24 portas, também proveniente de outro laboratório. Em termos de alimentação, foram empregados dois *no-breaks* com

capacidade para 3KVA, e todo o conjunto foi alojado em um anexo subutilizado do prédio, devidamente isolado em seu acesso e refrigerado por meio de condicionador de ar dedicado.

Figura 1 – *Cluster* montado.

No que diz respeito a *software*, as necessidades giram em torno de um sistema operacional que seja robusto o suficiente para explorar bem as condições do *hardware* modesto, consumindo a menor quantidade de recursos e disponibilizando o restante para os processos. Também é importante um estável controle da comunicação em rede, além de ter um bom histórico de implementações das linguagens de programação mais utilizadas e suas bibliotecas de programação paralelas. Descritos todos estes critérios, o sistema operacional selecionado foi o GNU/Linux, em sua distribuição Ubuntu versão *server* 12.04[1] que acompanhado de sua publicação de documentação (UBUNTU-SG, 2012), ofereceu adequado suporte para a instalação e configuração dos equipamentos.

Uma das principais vantagens na utilização do *software* livre é o fato de que não se tem gastos em relação a suas licenças de utilização, bem como sua relação próxima com a produção de código programável

1 http://www.ubuntu.com/

ISBN 978-85-62830-10-5

aberto, mantido por comunidades de utilizadores. Neste tocante, as bibliotecas de programação mais conhecidas e empregadas são as derivadas da MPI (*Messagem Passing Interface*)[2]. A seguir, todos os recursos empregados no *cluster*.

- *Hardware*:
    * 20 micro-computadores Intel Pentium 4 Celeron 3.2 Ghz 512MB 20GB HD Rede 10/100Mbps;
    * 1 *switch* 24 portas 10/100Mbps *auto-sense*;
    * 2 *no-break* 3KVA 16 tomadas;
- *Software*:
    * GNU/Linux Ubuntu Server 12.04;
    * Bibliotecas MPICH-2;

3. RESULTADOS E DISCUSSÃO

A instalação e configuração do *cluster* passou pela primeira etapa de treinamento em manutenção de *hardware*, que visou a verificação de todo o conjunto de micro-computadores em sua integridade de funcionamento e, principalmente, uniformização de componentes – conexão dos *slots* de memória, dos discos rígidos, e o funcionamento do correto dos mesmos, incluindo as fontes de alimentação, tão sensíveis em relação a oscilação de tensão na rede elétrica. Também nesta etapa houve o treinamento em redes de computadores, desde a infra-estrutura de cabeamento – conectorização e organização de equipamentos – até a configuração dos sistemas operacionais em seus endereços numéricos e literais.

A segunda etapa veio como um treinamento em relação ao *software* básico, ou seja, o sistema operacional. O GNU/Linux tem sido difundido no cotidiano de boa parte da comunidade principalmente em razão de sua versão para dispositivos móveis de comunicação, o Android[3], sistema da empresa Google Inc. Porém, sua maior parcela de utilizadores ainda encontra-se nos ambientes de ensino, por se tratar de uma ferramenta muito adequada aos ambientes de desenvolvimento de tecnologia. Estável e robusto, ele consome poucos recursos, tem suporte a uma variedade muito extensa de *drivers* e dispositivos, e suas bibliotecas de programação são muito acessíveis. A fluência no referido sistema foi o objetivo desta etapa, permitindo que sua instalação e configuração, bem como a operação, ficasse mais dinâmica para os envolvidos no projeto. Por exemplo, o consumo de memória por parte desta versão do sistema ficou por volta de 70MB, bastante razoável em relação aos 512MB disponíveis.

Em relação a suas bibliotecas de programação paralelas, as duas mais amplamente utilizadas na atualidade são as derivadas da MPI, principalmente sua implementação MPICH-2[4]. A primeira tem características mais relacionadas a programação paralela distribuível por meios de comunicação, ou seja, bastante adequada ao método de trabalho de um *cluster*. Sua instalação se dá por meio de pacotes que contém seus códigos e seu compilador. Funcionando como um complemento de linguagens de programação padronizadas, tais como C/C++ e Fortran, os parâmetros do MPICH-2 representam chamadas de sistema que, por exemplo, retornam tempos de execução e, principalmente, coordenam o envio dos fragmentos do processo para os nós de execução. Desta forma, os códigos são produzidos nestas linguagens mais amplamente difundidas, e contam com primitivas tais como *send* e *receive* (enviar e receber), que caracterizam de maneira inconfundível a passagem de mensagens (*message passing*). A seguir, comandos do código em dialeto MPI para as primitivas:

```
MPI_Send /* envia  mensagens com fragmentos de processos e instruções */
MPI_Recv /* recebe mensagens com resultados e tempos de processamento */
```

Uma dificuldade encontrada na utilização da estrutura já instalada foi a alimentação elétrica. Alguns eventos de instabilidade e oscilação provocaram um defeito em um dos *no-breaks* do conjunto, inviabilizando o uso de 8 dos 16 nós de execução, reduzindo assim pela metade o poder de processamento do grupamento, porém ainda em condições de funcionamento em sua outra metade. A detecção deste defeito foi

2 http://www.mcs.anl.gov/research/projects/mpi/
3 http://www.android.com/
4 http://www.mcs.anl.gov/research/projects/mpich2/

ISBN 978-85-62830-10-5

demorada por conta de sua intermitência de comportamento, inclusive no efeito sobre os micros, que eventualmente manifestavam erros e travamentos, o que prejudicou em algum nível o desenvolvimento de códigos mais aperfeiçoados de teste de carga.

4. CONCLUSÕES

A implementação de tal estrutura permite uma série de possíveis interações entre não somente as diversas modalidades de cursos de Informática implantados no IFBA/VC (integrado, subsequente, Proeja e superior), mas também com outros cursos, tais como Engenharia Elétrica e Ambiental. As possibilidades de modelagens que solucionem problemas computacionais de alta demanda por desempenho se apresentam em todos estes citados – balanceamento de carga em redes e serviços de rede, distribuição de serviços *web*, modelos de vasão de recursos hídricos, detecção de padrões e tratamento de sinais, entre outros. Ressalta-se o sucesso na preparação e disponibilização do espaço dedicado, bem como a concessão de uso dos equipamentos de Informática por parte da instituição, sem deixar de mencionar o enriquecimento do conteúdo dos envolvidos em tão próspero tema.

O recurso já encontra-se envolvido no suporte ao desenvolvimento de soluções para uma das orientações do Programa de Mestrado Interinstitucional entre o IFBA/VC e a Universidade Estadual de Santa Cruz – UESC, onde a proposta é modelar fenômenos hidrológicos por meio de estações de sensoriamento e tratar os dados utilizando o sistema de equações de Saint-Venant, adotando Método das Características (CHOW, 1959). Torna assim justificável e cheia de perspectivas a utilização da estrutura, confirmando o sucesso da proposta do projeto e sua execução.

REFERÊNCIAS

ASANOVIC, K.; BODIK, R.; CATANZARO, B. C.; GEBIS, J. J.; HUSBANDS, P.; KEUTZER, K.; PATTERSON, D. A.; PLISHKER, W. L.; SHALF, J.; WILLIAMS, S. W.; YELICK, K. A. **The Landscape of Parallel Computing Research: A View from Berkeley.** Technical Report No. UCB/EECS-2006-183, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, 2006.

CHOW, V. T. **Open Channel Hydraulics**. 1 Ed, New York, McGraw-Hill, 1959.

HENESSY, J. L.; PATTERSON, D. A. **Computer Architecture: A Quantitative Approach.** Fourth Edition, Morgan Kaufmann Publishers, Inc., 2006.

MOORE, G. E. **Cramming more components onto integrated circuits.** Electronics, Vol. 38, No. 8, pp. 114-117, 1965.

PITANGA, M. **Construindo Supercomputadores com Linux**, 2a. Edição, Editora Brasport, 2004.

TANENBAUM, A. **Redes de Computadores**, 4a. Edição, Editora Campus, 2003.

UBUNTU-SG. **Ubuntu Server Guide 12.04 LTS**. Acessado em 01/05/2012. Disponível em <https://help.ubuntu.com/12.04/serverguide/serverguide.pdf>, 2012.

WULF, W. A.; McKEE, S. A. **Hitting the Memory Wall: Implications of the Obvious, Computer Architecture News.** Vol. 23, no. 1, Mar. 1995, pp. 20–24.

ISBN 978-85-62830-10-5