CLUSTER DE ALTO DESEMPENHO PARA PROCESSAMENTO DE IMAGENS

Nome do(s) Autor(es)

AQUINO, Juliano T. G.

Angelo, Jefferson D.

REIS, Matheus F. B.

SANTOS, Willian C.

CARDOSO, Tomás H. C.

Nome do Orientador

Brezolin, Ligia M. T. F.

Resumo

O presente artigo tem como objetivo aprofundar os conhecimentos em computação paralela a partir do desenvolvimento de um cluster de alto desempenho, sendo assim, capaz de compreender o funcionamento do mesmo e de outros modelos de cluster. Também foi estudado a importância de um cluster com tolerância a falhas e porque ele é adotado em empresas como NASA e IBM. Além disso, como forma de demonstrar o funcionamento do cluster, foi estudado o processamento de imagens em JAVA e realizado execução em cluster.

Palavras-chave: cluster; máquinas virtuais; processamento de imagem.

Abstract

This article aims to deepen the knowledge in parallel computing from the development of a high-performance cluster, thus being able to understand the functioning of the same and other cluster models. The importance of a fault-tolerant cluster was also studied and why it is adopted in companies like NASA and IBM. In addition, as a way to demonstrate the functioning of the cluster, JAVA image processing was studied and executed in the cluster.

Keywords: *cluster; virtual machines; image processing.*

I. INTRODUÇÃO

Atualmente, vemos que a necessidade de poder computacional para executar determinadas tarefas vem se mostrando alta, visto que tecnologias principalmente gráficas e que envolvem cálculos complexos tem avançado cada vez mais. Sendo assim, o desenvolvimento de hardwares mais potentes tem aumentado bastante visando especialmente a execução destas tarefas, no entanto, seu custo é o principal limitador na hora da venda.

Sendo assim, diversas empresas vêm buscando outras formas de obter desempenho e agilidade na computação gráfica, uma das maneiras encontradas foi a utilização de clusters de computadores, onde é possível, de forma acessível financeiramente, juntar o poder computacional de diversas máquinas que talvez não teriam mais finalidade.

Objetivos Gerais

Em meio a limitação física dos componentes computacionais, pensou-se na elaboração de uma solução prática e eficiente para otimização de processamento de imagens. Sendo conseguida a partir da criação de um Cluster (Alto Desempenho) focado na otimização de processamento de imagens.

Objetivos Específicos:

- Criação de quatro máquinas virtuais em rede com o sistema operacional Linux;
- Configuração das VM’s em Cluster;
- Desenvolvimento do software orquestrador (master) utilizando a linguagem Golang.
- Desenvolvimento do software de processamento de imagens (workers) utilizando a linguagem Java.
- Automatização utilizando a ferramenta Vagrant.

II. REFERENCIAL TEÓRICO

Nos dias atuais, a preocupação em se obter resultados precisos cada vez mais rápidos, oriundos de um grande volume de instruções e de algoritmos complexos, demandam de recursos computacionais cada vez mais eficazes. A computação paralela é uma alternativa aos exorbitantes supercomputadores encontrados em laboratórios de grandes instituições científicas, militares e industriais anos atrás. De acordo com (Hwang, 1993), o processamento ou computação paralela pode ser definida como: Forma eficiente do processamento de dados com ênfase na exploração de eventos concorrentes no processo computacional. A principal ideia da computação paralela é reduzir o tempo computacional para a resolução de um problema (Grama et al., 2003), (Hariri e Parashar, 2004). É fundamentada no fato de que a execução das instruções de um problema pode ser dividida em um conjunto de instruções menores, que podem ser executadas simultaneamente por meio de algum tipo de coordenação.

As arquiteturas paralelas possuem várias formas de organização de fluxos em sistemas computacionais. (Flynn e Rudd, 1996) definiu a relação entre fluxos de instruções e fluxos de dados dentro do processo computacional. Um fluxo de instruções corresponde a uma sequência de instruções executadas (em um processador) sobre um fluxo de dados aos quais estas instruções estão relacionadas (Tanenbaum, 2001), (Flynn e Rudd, 1996).

Uma forma de aplicação da computação paralela são os Clusters. São grupos de computadores configurados para trabalhar em conjunto com aplicações específicas. O modo como são configurados dá a impressão de serem como um único computador (FERREIRA, p. 2008). A estrutura e agrupamento de computadores definida como:

> Estrutura de agrupamento de computadores ou clusters apresenta vantagens competitivas em relação aos ambientes multiprocessados de memória compartilhada (computadores com diversos processadores em uma placa-mãe), permitindo que o acréscimo de computadores torne o sistema mais rápido, possuindo componentes de fácil disponibilidade e manutenção (PITANGA, 2008 p.10).

A transmissão de dados é o principal gargalo nas operações. Isso ocorre devido à latência que há nas informações que são trocadas entre os nós. Hoje o principal meio de comunicação de dados é a rede, sendo composta por meios físicos e mecanismos de controle para o transporte, além de uma política de sincronização de dados.

Apesar deste gargalo natural, aumentar o desempenho é um dos principais objetivos de clusters de computadores, entende-se desempenho como carga ou tempo de execução de tarefas. Os clusters exploram explicitamente o paralelismo como principal ferramenta, e para comprovar ganhos existem métricas que são utilizadas a fim de estruturar uma melhor solução para os problemas paralelizados. Entre essas métricas podemos citar o speedup. O speedup é uma equação que possibilita medir a razão entre o tempo gasto para executar uma aplicação em um único processador, pelo tempo de execução em paralelo, conforme a seguir:

Sun e Ni (1990), Hwang (1993), Sahni e Thanvantri (1996) definem duas conotações para o speedup: absoluto e relativo. O speedup absoluto é a relação entre o tempo do melhor algoritmo sequencial pelo tempo gasto com uma versão do algoritmo paralelo em P processadores. O speedup absoluto pode levar em consideração os recursos da máquina ou não. No primeiro caso, o speedup é definido como a relação entre o tempo gasto com o melhor algoritmo sequencial em um processador e o tempo gasto pelo algoritmo paralelo. No segundo caso, o speedup é definido pelo tempo do melhor algoritmo sequencial executado na máquina mais rápida em relação ao tempo do algoritmo paralelo na máquina paralela. O speedup relativo considera o paralelismo de maneira igualitária. É definido pela relação entre o tempo gasto pelo algoritmo paralelo em 1 processador e o tempo gasto pelo algoritmo paralelo em P processadores. O speedup relativo permite avaliar como o desempenho do algoritmo varia de acordo com o número de processadores, já que é comparado consigo mesmo. Ao utilizar dez processadores em um ambiente paralelo, por exemplo, logo vem a ideia de que as aplicações executadas neste ambiente serão processadas dez vezes mais rápidas. Este desempenho considerado linear, raramente é alcançado na prática (Patterson e Hennessy, 2005). Amdahl (1967) observa que se a parcela sequencial de uma aplicação consome 1/s do tempo de execução total, logo o

speeedup máximo possível por um computador paralelo é 1/s. Por exemplo, se um programa precisa de 20 horas para ser executada em um ambiente com apenas 1 processador, e contém uma parcela de 1 hora que não pode ser paralelizada, restando 19 horas (95%) de porção paralelizável, então mesmo utilizando centenas de processadores na execução desta aplicação, o tempo mínimo de execução não pode ser menor do que 1 hora. Assim, o speedup máximo é 20x.

Em virtude dos fatos mencionados, a NASA idealizou, em meados da década de 90, o primeiro projeto de um típico cluster chamado de Beowulf, com o objetivo de processar as informações espaciais. Além de oferecer baixo custo de implementação, apresenta alta disponibilidade e é tolerante a falhas, pois os recursos podem ser duplicados de maneira redundante, considerados fatores importantes em sistemas de alto desempenho. Também oferece alta disponibilidade, sendo necessário apenas adicionar uma nova máquina para aumentar o poder de processamento. Porém, devem ser considerados alguns aspectos quanto ao overhead causado pela latência de comunicação de rede entre os processos. Um cluster pode ser homogêneo ou heterogêneo: homogêneo quando todos os equipamentos em todos os nós são iguais e, heterogêneo, caso contrário. Além dos clusters, existe outra categoria denominada por Network of Workstation (NOW) em que computadores que não são exclusivamente dedicados à computação paralela também são aproveitados para efetuar o processamento de aplicações paralelas. Contudo, os clusters não são utilizados apenas em sistemas em que se busca o desempenho. (FERREIRA 2008, p.683) estende sua aplicabilidade em outras áreas, de acordo com a sua funcionalidade em duas categorias básicas: High Availabitity (HA) ou Alta Disponibilidade, e clusters de High-Performance (HP) ou Alto Desempenho, sendo possível a combinação entre esses, conforme a necessidade.

Clusters High-Availability - HA

Alta disponibilidade é uma técnica que consiste na configuração de dois ou mais computadores para que passem a trabalhar em conjunto. Dessa forma cada computador monitora os demais e quando ocorrer falhas assume o serviço que fica indisponível. Para este subgrupo de computadores atribui-se o nome de clusters de alta disponibilidade (FERREIRA 2008, p.683).

Clusters High-Performance Beowulf

Beowulf é uma categoria de multicomputador que pode ser utilizada para computações paralelas, normalmente é composto por um nó servidor e um ou mais nós clientes conectados via rede. É construído com hardware comum, como qualquer computador capaz de operar com Linux, placa de rede padrão Ethernet e switches. Para esta classe não é utilizado nenhum hardware feito sobre encomenda e é facilmente reproduzível (FERREIRA 2008, p. 691).

Beowulf usa software comum como o sistema operacional Linux, Paralell Virtual Machine (PVM), e Message Passing Interface (MPI). O nó servidor é encarregado de controlar todo o grupo de computadores e serve arquivos para os nós clientes, também é a porta de saída para o mundo exterior (FERREIRA 2008, p. 691).

Pitanga (2008) observa que, dentre as diversas áreas de aplicação, os diversos ramos da computação gráfica têm usufruído muito dos *clusters* de alta performance, principalmente pelo seu vantajoso equilíbrio entre custo e benefício. O uso de agrupamentos de computadores tem se destacado principalmente no uso de atividades relacionadas com a renderização de imagens e cenas artificiais realistas.

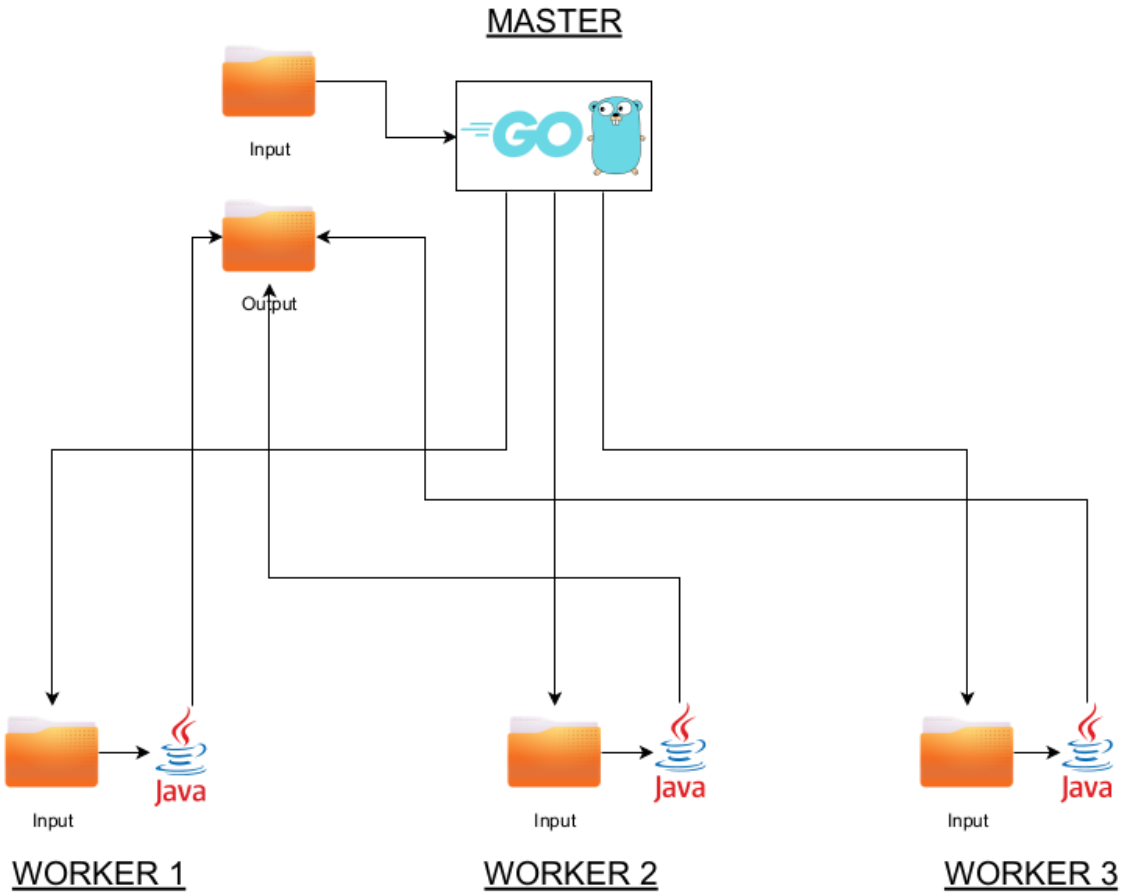
Com a geração de imagens e cenas cinematográficas artificiais cada vez mais próximas da realidade, o processo de renderização tem exigido, de um modo crescente, maior poder computacional. Como as cenas artificiais realistas são muito complexas, com inúmeros parâmetros e detalhes, o processo de renderização exige um conjunto de cálculos extremamente massivos. Se justificando, assim, o uso de Cluster para otimização do processamento.

III. MATERIAIS E MÉTODOS

O projeto foi dividido em três etapas:

1. Criação e configuração das máquinas virtuais;

2. Desenvolvimento do algoritmo orquestrador do cluster;
3. Desenvolvimento do algoritmo de processamento de imagens.



Etapa 1

Para a criação das máquinas virtuais (VM) foi utilizada a ferramenta Vagrant, que consiste em um um software de código aberto para criar e manter ambientes de desenvolvimento virtuais portáteis, utilizando VirtualBox, KVM, Hyper-V, Docker containers, VMware, e AWS. Ele tenta simplificar a gerência de configuração de software das virtualizações para aumentar a produtividade do desenvolvimento. Neste caso o hypervisor utilizado foi o VirtualBox.

Através do arquivo Vagrantfile foi possível especificar as configurações de cada VM, como quantidade de processadores e memória principal; endereço IP; imagem do sistema operacional etc. Além de executar scripts de instalação de

dependências e configuração de chaves de acesso remoto, uma vez que o cluster opera em rede.

Configuração das VMs

VM	SO	CPU	RAM	IP	Dependências	
Master	ubuntu/xenial64	2	1024 MB	172.42.42.100	Golang	sshpass
Worker1	ubuntu/xenial64	1	1024 MB	172.42.42.101	JAVA 8	sshpass
Worker2	ubuntu/xenial64	1	1024 MB	172.42.42.102	JAVA 8	sshpass
Worker3	ubuntu/xenial64	1	1024 MB	172.42.42.103	JAVA 8	sshpass

Etapa 2

Para o desenvolvimento do algoritmo orquestrador do cluster foi escolhida a tecnologia Golang, uma linguagem de programação criada pela Google e lançada em código livre em novembro de 2009. É uma linguagem compilada e focada em produtividade e programação concorrente.

Esta etapa do projeto foi dividida em outras quatro:

1. **Função que divide uma base de imagens entre os workers**

   Através de *Goroutines* e *Buffered Channels*, criou-se uma fila de execução, na qual cada “thread” envia uma mensagem de liberação no canal ligado à pilha de execução principal. Desta forma, cada imagem é enviada intercaladamente entre os “nós trabalhadores”, sem que haja colisões e perda de arquivos na rede.

2. **Função que executa o programa de processamento de imagens em cada worker**

   Após cada worker possuir uma fração da base de imagens original, é disparada três *Goroutines,* uma para cada nó escravo, responsáveis por executar remotamente o programa de processamento de imagens, que será detalhado mais adiante.

3. **Função que copia de volta para o master as imagens processadas em cada worker**

   Como este procedimento exige acesso de várias VMs ao mesmo diretório, foi necessário utilizar uma execução linear, garantindo assim o correto funcionamento do algoritmo.

4. **Função que prepara os workers para uma nova execução de processamento**

   Processo que consiste em executar um script em cada worker a fim de apagar as imagens do processamento anterior.

Toda comunicação em rede foi feita com *sshpass*, uma forma de passar a autenticação *ssh* como parâmetro, trazendo fluidez ao processo. Além disso, foram também utilizados *WaitGroups* para a gerência das *Goroutines.*

Etapa 3

Para o programa de processamento de imagens foi escolhida a tecnologia JAVA, uma linguagem de programação orientada a objetos desenvolvida na década de 90 por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems.

O programa, chamado de *workerApp.jar*, basicamente consiste em ler um banco de imagens, guardando-o em uma lista, e através da função *Convert* realizar a mudança de cada imagem para 8 bits em escala de cinza (*TYPE_BYTE_BINARY*).

Além disso, também foi desenvolvido o programa *workerCopy.jar,* que realiza a cópia remota das imagens processadas de volta para a VM Master.

IV. ANÁLISE E DISCUSSÃO DOS RESULTADOS

A elaboração de cluster de computadores com propósito de utilizá-lo para processamento de imagens é sensacional quando feita com uma grande quantidade de computadores. Foi notado que o funcionamento do cluster em máquinas virtuais também é funcional, no entanto, o fato do poder computacional das máquinas virtuais serem provenientes da máquina física é um grande limitador no projeto. Sendo assim, cluster de alto desempenho para ter uma boa redundância e alta performance necessita de uma variedade de máquinas, mesmo sendo ultrapassadas, sua utilização no balanceamento dos nós é fundamental, assim não sobrecarregando um nó.

No entanto, foram obtidos bons resultados, visto que o objetivo do projeto foi a execução de um processamento de uma imagem utilizando o poder computacional de outras máquinas, estes resultados foram alcançados, obtendo até mesmo o processamento de várias imagens ao mesmo tempo.

V. CONSIDERAÇÕES FINAIS

A utilização de cluster atualmente é uma ótima solução para executar tarefas que demandam alto poder computacional, como exemplo a renderização de uma imagem.
Como foi visto, a variedade de tipos de cluster consegue oferecer diversas arquiteturas computacionais se tornando aplicável em qualquer área ou segmento.

O desenvolvimento deste projeto mostrou que através de uma aplicação de Cluster Beowulf para processamento e renderização de uma imagem, foi possível unir poder computacional de várias máquinas, no caso virtuais, a fim de executar uma tarefa, no qual trouxe resultado satisfatório, constatando assim que clusters de computadores é uma solução eficiente e acessível para quem busca alta performance em processamento.

Vale também ressaltar que o desempenho do cluster de computadores é proporcional a quantidade de computadores envolvidos, o poder computacional de cada computador e também a estruturação e quantidade ideal de nós para

determinada tarefa, no entanto, a utilização de muitos nós desnecessários pode impactar no desempenho.

REFERÊNCIAS

AMDAHL, G. M. *Validity of the single processor approach to achieving large scale computing capabilities.* In: AFIPS '67 (Spring): Proceedings of the April 18-20, 1967, spring joint computer conference, New York, NY, USA: ACM, 1967, p. 483485.

BACELLAR, Hilário. Cluster: *Computação de Alto Desempenho*. Instituto de Computação, Universidade Estadual de Campinas. 2012

FERREIRA, Rubem E. Linux Guia do Administrador do Sistema-2ª Edição. Novatec Editora,2008.

FLYNN, M. J.; Rudd, K. W. *Parallel architectures*. ACM Comput. Surv., v. 28, n. 1, p. 6770, 1996.

GRAMA, A.; Gupta, A.; Karypis, G.; Kumar, V. *Introduction to parallel computing, second edition. Second ed.* Addison Wesley, 856 p., 2003.

HARIRI, S.; Parashar, *M. Tools and environments for parallel and distributing computing*. Wiley Interscience, 2004.

HWANG, K. *Advanced Computer Architecture: Parallelism. Scalability*, Programmability (McGraw-Hill, New York, 1993), 1993.

MOURA, B. M. Paz; Junior, R. R. M. Vieira. *Arquitetura de Sistemas para Cluster e Grades Computacionais: Uma solução Independente de Fabricantes Baseada em Clusters Beowulf.* Revista do CCEI. V. 19, N. 34 (2015).

PATTERSON, D. A.; Hennessy, J. L. *Organização e projeto de computadores: A interface hardware/software.* 3 ed. Rio de Janeiro: Editora Campus, 2005.

PITANGA, Marcos. Construindo Supercomputadores com Linux. 3ª e.d, Brasport, Rio de Janeiro, 2008.

SAHNI, S.; Thanvantri, V. *Performance metrics: Keeping the focus on runtime*. IEEE Parallel and Distributed Technology, v. 4, n. 1, 1996.

SHISHIDO, HENRIQUE. *Paralelização de algoritmo de processamento de imagens digitais*. Dissertação (Mestrado). Universidade Estadual de Maringá. 2010.

SUN, X.-H.; Ni, L. M. *Another view on parallel speedup*. In: Supercomputing '90: Proceedings of the 1990 ACM/IEEE conference on Supercomputing, Los Alamitos, CA, USA: IEEE Computer Society Press, 1990.

TANENBAUM, A. *Organização estruturada de computadores*. Rio de Janeiro: LTC, v. 9, 2001.