LEANDRO BITENCOURT FERNANDES

TI VERDE: ESTUDO E IMPLEMENTAÇÃO DE UM AMBIENTE DE ALTO DESEMPENHO UTILIZANDO CLUSTER COMPUTACIONAL, COM COMPUTADORES RECICLADOS

Trabalho de Conclusão de Curso, apresentado para obtenção do Grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Paulo João Martins

CRICIÚMA
2012

LEANDRO BITENCOURT FERNANDES

TI VERDE: ESTUDO E IMPLEMENTAÇÃO DE UM AMBIENTE DE ALTO DESEMPENHO UTILIZANDO CLUSTER COMPUTACIONAL, COM COMPUTADORES RECICLADOS

Trabalho de Conclusão de Curso, apresentado para obtenção do Grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Criciúma, 29 de Dezembro de 2012.

BANCA EXAMINADORA

Prof. Paulo João Martins - MSc - (UNESC) - Orientador

Prof. Esp. Sérgio Coral – (UNESC)

Prof. MSc. Gustavo Bisognin – (UNESC)

Dedico este trabalho primeiramente a Deus, pela saúde, fé e perseverança que tem me dado.

A meus pais, a quem honro pelo esforço com o qual mantiveram seus filhos na escola. A meus irmãos, pelo incentivo na busca de novos conhecimentos, a minha namorada Juliana pela compreensão quando não pude estar presente e ao meu professor mestre Paulo João Martins pela sabedoria e dedicação com a qual orientou este trabalho.

AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus por ter me dado forças para não desanimar e saúde para continuar lutando por um futuro melhor.

Agradeço aos meus pais João e Salete, por me incentivarem a conquistar meus objetivos e me apoiarem nos momentos em que mais precisei.

Também não posso esquecer-me de meu irmão Luciano, por todo o incentivo e ensinamento no decorrer deste curso, o meu muito obrigado.

Meu agradecimento também a Patrícia, minha irmã querida, pela atenção e compreensão na ajuda que dedicou para correção deste trabalho, muito obrigado.

Aos meus cunhados Cristiane e Franco, pelo incentivo e apoio em toda essa caminhada.

Meu agradecimento muito especial, ao meu professor e orientador Msc Paulo João Martins, muito obrigado pela sua dedicação, paciência, pelos ensinamentos e pelo incentivo neste trabalho e ao aprendizado que levarei comigo por toda minha vida.

De forma especial, agradeço a minha namorada Juliana, que em tão pouco tempo, me disponibilizou muito de sua atenção e carinho para que este trabalho se concretizasse.

Não posso esquecer os (as) amigos (as), que sempre me deram palavras de ânimo nos momentos mais difíceis no decorrer do curso e torceram por mim, em especial Wellington e Robison.

Jamais poderia me esquecer de meu amigo inseparável Leandro Jerônimo, por caminhar junto comigo durante todos esses anos, incentivando apoiando e compartilhando alegrias.

Também aos meus professores, pelo carinho, apoio e ensinamentos que serão usados em todos os momentos de minha vida.

Por fim, gostaria de agradecer, mais uma vez aos meus amigos e familiares, pelo carinho e pela compreensão nos momentos em que a dedicação aos estudos foi exclusiva, a todos que contribuíram direta ou indiretamente para que esse trabalho fosse realizado, meu eterno AGRADECIMENTO.

“Criciúma, Criciúma
nosso clube de amor
alma, garra e coração.”
Carlos Ernesto Lacombe

RESUMO

Esta pesquisa foi desenvolvida através da necessidade de reaproveitar os computadores em desuso nas organizações. Para isso, observa-se na TI Verde uma metodologia importante para tratar os conceitos de descarte correto dos equipamentos e/ou a reutilização dos mesmos. A solução para reutilização dos computadores foi encontrada na computação distribuída, onde se pode implementar um *cluster* computacional reaproveitando o processamento dos computadores descartados. Procurando uma plataforma de *cluster* computacional de uso geral, encontra-se no Openmosix uma ferramenta que viabiliza este projeto. A partir da plataforma escolhida, executa-se a aplicação Omtest, objetivando testar o funcionamento do cluster. O Omtest é um software composto por várias aplicações que foram tratadas separadamente. Os resultados obtidos foram analisados e comparados conforme a quantidade de nós adicionados ao aglomerado.

Palavras-chave: TI Verde. Sistemas Distribuídos. Cluster.

ABSTRACT

This research was developed through the need to reclaim disused computers in organizations. For this, there is a methodology in Green IT important for treating the concepts of proper disposal of equipment and / or reuse them. The solution for reuse of computers in distributed computing found where one can implement a reusing computational processing cluster of computers discarded. Looking for a platform for general purpose computing cluster, the openMosix is a tool that makes possible this project. From the chosen platform, you run the application Omtest, aiming to test the functioning of the cluster. The Omtest is composed of several software applications that have been treated separately. The results were analyzed and compared according to number of nodes added to the cluster.

Keywords: Green IT. Distributed Systems. Cluster.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3D	Terceira Dimensão
CD	Compact disc
CPU	Unidade Central de Processamento
CRT	Cathodic Ray Tube
GPL	General Public Licence
HD	Hard Disc
HPC	High Performance Computing
IBM	International Business Machines
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MHZ	Mega Hertz
MIMD	Multiple Instruction, Multiple Data
MISD	Multiple Instruction Single Data
PAD	Processamento de Alto Desempenho
PC	Personal Computer
SIMD	Single Instruction Multiple Data
SISD	Single Instruction Single Data
SMP	Multiprocessamento simétrico
SO	Sistema Operacional
SSI	Sistema de Imagem Simples
TCC	Trabalho de Conclusão de Curso
TI	Tecnologia da Informação
TI Verde	Tecnologia da Informação Verde

SUMÁRIO

1 INTRODUÇÃO

Com o grande avanço tecnológico, tanto de hardware e software, as empresas cada vez mais precisam estar atualizadas com o seu poder computacional, e consequentemente descartar seu equipamento defasado para a utilização dessas novas tecnologias.

Segundo Silva (2009), as áreas que mais precisam estar atualizadas são as que exigem processamento contínuo e áreas que necessitam de precisão nos seus resultados.

O *cluster* computacional surgiu nas últimas décadas como alternativas ao alto custo dos supercomputadores. É um tipo de sistema de processamento paralelo que consiste na coleção de computadores independentes, interconectados através de uma rede, trabalhando cooperativamente como um único e integrado recurso computacional.

Ao contrário de se obter somente um supercomputador de custos altíssimos, pode-se criar um cluster computacional utilizando muitos computadores convencionais de baixo custo, interligados por uma rede estruturada, obtendo-se assim, uma alta capacidade de processamento.

Aumentar o desempenho é um dos principais objetivos dos *clusters* de computadores, entende-se desempenho como carga ou tempo de execução de tarefas (BACELLAR, 2010).

O projeto tem por características diferenciadas dos demais projetos de *cluster* de alto desempenho, procurou-se a reutilização dos computadores obsoletos, condicionados, comprovando a importância de uma política de TI Verde, onde se fortalece o reaproveitamento dos computadores. Outra característica do projeto é que todas as ferramentas e documentações foram de uso livre, ou seja, Open Source.

O objetivo foi criar um centro de processamento computacional de alto desempenho com todas as estações provenientes do descarte da organização, comprovando assim a importância do reaproveitamento dos computadores que possui certo processamento, mas que já foram dados como lixo eletrônico pela organização.

Tais estudos nos levaram a procura por computadores encontrados nos depósitos das organizações e que possam ser utilizados no cluster e assim, contribuir para a ampliação da capacidade de processamento.

Todo o desempenho do cluster computacional será avaliado por uma aplicação. Os relatórios gerados pela aplicação serão documentados. O objetivo dos testes é avaliar os ganhos de desempenho do *cluster* a cada nó (computador) adicionado ao aglomerado.

1.1 OBJETIVO GERAL

Implementar um cluster computacional de alto desempenho, utilizando computadores descartados pela organização.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste projeto são:

a) entender e aplicar os conceitos de computação distribuída em um *cluster* computacional;
b) utilizar os conceitos da área de TI Verde em um cluster computacional;
c) implementar o ambiente para um cluster de alto desempenho;
d) realizar testes de desempenho no ambiente do cluster com a utilização de uma aplicação;
e) documentar e descrever os testes realizados no ambiente.

1.3 JUSTIFICATIVA

Com a evolução das aplicações exigindo cada vez mais poder de processamento e devido ao custo elevado para se adquirir supercomputadores, surgiu o desafio de aproveitar o poder de processamento dos computadores descartados pela organização.

Segundo Lorenzato (2010), “Hoje, é muito comum fazermos a disposição de equipamentos antigos para quem compra, desmantela e revende peças. Mas esta não é a melhor solução.” A conquista de certificado oficial, as empresas precisam garantir o descarte correto dos computadores.

Um cluster computacional é um ambiente de computação paralela, formada por computadores com configurações heterogêneas. Limita-se a apenas utilizar a mesma plataforma de cluster para todos os nós, configurado todos em um mesmo domínio.

A abordagem cluster possui alta escalabilidade, pois tarefas de inclusão ou exclusão de nós escravos não exigem que sejam feitas modificações no ambiente. É realizado de forma isolada, partindo do nó mestre, por meio da execução de algum comando específico do software escolhido (CUNHA et al, 2005).

Normalmente se opta por um cluster quando surge a necessidade de segurança e desempenho em uma organização. Construímos o cluster quando os conteúdos são críticos ou quando os serviços precisam estar disponíveis e/ou processados mais rapidamente.

Logo, este trabalho tem como objetivo principal o reaproveitamento dos computadores em desusos pela organização, a fim de criar um cluster computacional de alto desempenho. Para isso será diagnosticado a política de descarte de computadores da organização, para então poder estar inserindo no aglomerado.

1.4 ESTRUTURA DO TRABALHO

Este TCC está organizado em seis capítulos onde descrevemos o desenvolvimento do projeto de criação de cluster computacional com computadores descartados por uma organização. No primeiro capítulo há uma introdução do assunto tratado neste projeto, apresentando os objetivos específicos e gerais, como também a justificativa do mesmo.

No capítulo dois, intitulado “TI Verde”, é direcionado a esclarecer os conceitos básicos de TI Verde, para demonstrar o quanto é importante a implantação das políticas verdes em uma organização, contribuindo com o meio ambiente.

O capítulo três, intitulado “Computação paralela e distribuída” tem por objetivo esclarecer ao leitor os conceitos de computação paralela e sistemas distribuídos com o propósito de construir um ambiente de cluster computacional com computadores reciclados, resultando na união dos conceitos de computação paralela e distribuída e TI Verde.

O capítulo quatro trata do maior foco do trabalho que é o Cluster, explicando detalhes de suas características, vantagens, benefícios, tais como, os tipos e aplicações que podem ser utilizadas.

No quinto capítulo, temos o material descritivo referente aos trabalhos correlatos que foram usados para embasamento desse projeto tais como: Práticas da TI Verde que contribuem para o desenvolvimento sustentável: Um estudo de caso nas indústrias do RN, por Glauber Ruan Barbosa Pereira de 2009.

No sexto capítulo, intitulado “Projeto do Cluster Computacional”, inicia-se a documentação de todo trabalho desenvolvido do projeto,começando a documentar os tipos clusters que iremos testar, estrutura física do aglomerado, ferramentas de análise e gerenciamento. Por fim, terminamos o capítulo documentando os “Resultados obtidos com o projeto”.

No sétimo e ultimo capitulo, temos as conclusões que obtivemos com esse projeto.

2 TI VERDE

A Tecnologia da Informação (TI), que se define como um conjunto de todas as soluções e atividades providas por recurso de computação, ganha importância nas organizações quando elas mesmas percebem que a informação que detém faz parte de seu patrimônio.

Segundo Takahashi (2009), a tecnologia quando bem inserida nas organizações independente de seu tamanho ou porte, traz vantagens para as empresas com a diminuição de custos, aumento da produtividade dos seus colaboradores assim melhorando a qualidade dos serviços.

As organizações sofrem constante pressão para reduzir custos em investimentos e despesas de TI. Esses fatores devem ser cuidadosamente analisados na perspectiva do valor da empresa, pensando na redução de custos ou da não continuidade de algum projeto pode prejudicar o desempenho da área comercial ou um serviço de atendimento ao cliente e também uma ineficiência operacional, afetando diretamente na linha de receita e a imagem da empresa perante o mercado (BOTTO, 2010).

A TI Verde tem ampliado os campos de pesquisas na área de tecnologia nos últimos tempos, tem sido muito discutido e inserido nas campanhas publicitárias e em mídias visuais. Nada mais é do que um conjunto de práticas que visa tornar mais sustentável e menos prejudicial o nosso uso da computação. Geralmente, alguns administradores de TI dedicam seu tempo de estudo na economia de recursos computacionais e principalmente na economia de energia.

Figura 1 - Símbolo da TI Verde

Fonte: http://computerworld.uol.com.br

2.1 TENDÊNCIAS DA TI VERDE NAS ORGANIZAÇÕES

Devido ao grande foco da mídia referente à problemática ambiental, há uma grande mobilização da indústria de informática no mundo e principalmente nas organizações a fim de estabelecer normativas de estruturação e organização com propósito de estabelecer um desenvolvimento sustentável.

> A TI tem gerado varias oportunidades para o crescimento do consumo de diferentes produtos e serviços, dentre elas o desenvolvimento de novas alternativas nas organizações em utilizar efetivamente os benefícios tecnológicos a seu favor, agilizando as atividades organizacionais e promovendo o bem estar a sociedade ao uso de TI. Dessa maneira em função do crescente ritmo de demanda das empresas, novos produtos acabam se aglomerando junto a outros que se tornam defasados devido ao consumo crescente e as constantes atualizações tecnológicas, gerando o que se conhece como lixo eletrônico (PEREIRA, 2009, p.12).

A TI VERDE e o lixo eletrônico atualmente têm sido um dos temas de constante discussão na sociedade tecnológica. O constante aumento das vendas de equipamentos eletrônicos tais como: computadores, celulares, notebooks, monitores, impressoras entre outros aumentam gradativamente no Brasil e no mundo. A tecnologia está cada vez mais próxima das classes econômicas inferiores e as classes altas trocam cada vez mais rapidamente seus equipamentos por modelos mais atualizados e modernos, o que torna mais preocupante, dado a evolução das tecnologias.

Muitos monitores antigos do modelo Cathodic Ray Tube (CRT), em inglês, sigla de (tubo de raios catódicos), são encontrados em depósitos de empresas ou nos lixões urbanos, formando milhares de toneladas de lixo eletrônico, que irão levar milhares de anos para se decompor, lembrando também do risco de contaminação do meio ambiente, pois os equipamentos eletrônicos concentram sua fabricação em metais pesados.

Apenas trocar todos os monitores CRT da organização por monitores de cristal líquido (LCD) ou os de diodo emissor de luz (LED), simplesmente não é uma atitude verde de economizar o consumo de energia, e sim tornar um ato selvagem contra a natureza, pois os mesmo monitores serão descartados incorretamente.

2.2 PRÁTICAS DE TI VERDE

Segundo Takahashi et al (2009) as práticas de TI Verde pode ser dividida em três subníveis:

a) TI Verde de incrementação tática: onde não muda a infraestrutura de TI da organização e apenas criam-se políticas de contensão de gastos elétricos excessivos. Como exemplo o monitoramento de energia disponível nos equipamentos. O desligamento dos mesmos no momento do não uso, otimização da temperatura no setor e salas. Tais medidas são simples e não geram custos adicionais as organizações;
b) TI Verde estratégico: necessita-se de uma auditoria sobre a infraestrutura de TI e seu uso relacionado ao meio ambiente, desenvolvendo meios mais viáveis e ecologicamente corretos de produção de bens ou serviços. Como exemplo a implantação de uma nova infraestrutura elétrica visando maior eficiência de consumo elétrico;
c) Deep IT: Muito mais amplo que os demais níveis, criam-se então um projeto de implantação estrutural de um parque tecnológico otimizando o desempenho com o mínimo de gastos elétricos. Como exemplo a implantação de sistemas de refrigeração o que dispõem de equipamentos no local gerando maiores gastos que os demais níveis.

No nível de incrementação tática de TI Verde sua implantação se torna muito simples, onde apenas algumas configurações de energia no sistema operacional do computador reduzem significantemente o consumo de energia.

A TI Verde estratégica se torna um pouco mais complexa e necessita de especialista de TI para sua implantação. Medidas como a virtualização de Datacenters, uma prática destinada à redução de custos de energia e centralização dos serviços tecnológicos, onde poderíamos ter vários servidores em várias máquinas ocupando muito espaço físico e também gerando alto consumo de energia, pode-se então montar apenas um servidor onde todos os outros serviços fiquem virtualizados em vários outros sistemas operacionais na mesma máquina.

Outra medida computacional relacionada a TI Verde dentro de uma organização é o serviço de Terminal Server, onde a ideia é centralizar o processamento de muitas estações de trabalho em apenas um servidor. No local das estações de trabalho comum, pode-se então

trocar por ThinClients (Cliente Magro) ou “terminais burros”, que são equipamentos que funcionam em rede com consumo mínimo de energia. Funciona como estação de trabalho cliente/servidor, todo o processamento fica por conta do servidor.

2.3 TI VERDE NAS ORGANIZAÇÕES

A preocupação com o destino dado aos resíduos eletrônicos é um assunto que está em debate entre os grandes empresários do mundo. Vários setores estão em alerta perante o meio ambiente, e não é diferente nos departamentos de TI, já que praticamente tudo depende desse setor.

Segundo Bill Joy, co-fundador da SUN Microsystem, declarou que a onda da TI Verde será uma revolução ainda maior do que foi a Internet, há quinze anos. E conta ainda que as empresas que não se adequarem a estes projetos ficarão para trás por conta de um mercado cada vez mais exigente.

Uma pesquisa realizada pela Symantec Corporation apontou que a maioria dos gestores de TI no mundo, demonstrou interesse por estratégias e soluções de TI Verde, por conta da redução de custos em energia, também como parte das iniciativas de responsabilidade sócio-ambiental da organização. Conclui ainda que 97% dos participantes afirmaram estar pelo menos discutindo estratégias de Verdes, enquanto 45% já implementou alguma das características na organização. No Brasil, 73% afirmam ter um projeto de implantação das políticas Verdes (TOTTALMARKETING, 2009).

Cresce a pressão para que as empresas sejam auto-sustentáveis. No entanto tem-se dificuldade de encontrar profissionais especializados em questões ambientais e sustentabilidade. O que abre portas para os profissionais de TI os quais tomam proveito do que já sabem sobre TI Verde.

A tecnologia está presente em todos os processos da organização, no entanto as principais ações para corte do consumo excessivo de energia elétrica vêm dos departamentos de TI.

Já é muito claro que ser sustentável não é mais opção e sim essencial em uma organização. Ser responsável pelo impacto produzido pelas suas operações no meio ambiente e também trabalhar para minimizá-las é obrigação, cobrada pelos consumidores e pela sociedade (RIBEIRO, 2009).

O meio ambiente já é uma preocupação de todos, de todas as áreas, setores, organizações do mundo, e a tecnologia já vem se adequando a essa nova realidade. Os

profissionais da área de TI, já estão se adequando uma nova política “verde”, onde o ser sustentável é fator importante no desenvolvimento das organizações, assim gerando maior credibilidade e reconhecimento de todos.

2.4 TI VERDE E O SOFTWARE LIVRE

Quando analisamos e discutimos TI Verde não podemos esquecer-nos da primeira e pioneira tendência verde presente na grande maioria das empresas que são os softwares livres. O software livre pode ser usado, copiado, modificado, estudado e redistribuído sem nenhuma restrição. Sua distribuição é livre vem acompanhado de uma licença chamada General Public Licence (GPL), em inglês, com a disponibilização de seu código fonte.

A maior vantagem do uso de um software livre é na redução de gastos com licença, pois não teríamos custos nenhum de investimento.

Segundo Ribeiro (2009), quando realizamos um projeto sem software livre e sem políticas verdes, além do alto custo com licenças e energia, a imagem do projeto fica manchada e os clientes insatisfeitos.

2.5 TI VERDE E O LIXO ELETRÔNICO

Um dos tipos de lixo que mais crescem nas indústrias de reciclagem em todo mundo é o do lixo eletrônico. Tudo isso por conta do grande crescimento das tecnologias, onde cada vez mais rápido são lançados novos produtos.

O consumismo da sociedade é um fator de alerta, onde a própria mídia de peso na decisão de compra dos consumidores, tendo fator direto no aumento de produção do lixo eletrônico.

Atualmente é designado o lixo eletrônico como *e-lixo*, termo utilizado para denominar como qualquer equipamento que perdeu sua vida útil e que já foi descartado, como exemplo, computadores desktop, monitores, celulares, entre outros.

Os equipamentos eletrônicos possuem normalmente em sua estrutura metais e compostos químicos tais como: mercúrio, chumbo, cádmio entre outros que podem afetar o meio ambiente onde é descartado e também causando diversos males ao ser humano.

Para tal problema, existe uma série de soluções que podem ser realizadas para o tratamento correto do lixo eletrônico, como o descarte do material em institutos e

organizações responsáveis pela reciclagem. Outra forma é a doação para alguma organização de incentivo da inclusão digital tais como ONGs.

Mediante a tais fatos analisados, o descarte desses equipamentos ditos obsoletos para uso em uma organização, podem sim serem reaproveitados por algumas soluções desenvolvidas no mundo da tecnologia da informação. Em nosso caso, diante do descarte do lixo eletrônico podemos utilizar o processamento obsoleto dos equipamentos, para então contribuir com o poder de processamento de um cluster computacional.

3 COMPUTAÇÃO PARALELA E DISTRIBUIDA

Nos últimos tempos o poder de processamento dos computadores tem aumentado em torno de 20% ao ano, enquanto que os microprocessadores têm sido em torno de 40% ao ano (PITANGA, 2008).

A evolução está acima da média devido ao fato de os microprocessadores estarem evoluindo tanto em arquitetura como também em tecnologia. Esse avanço está servindo para diminuir o esforço necessário para processar um ciclo de instrução.

Outro grande passo do avanço tecnológico foi a criação das redes locais (LAN), onde grande quantidade de dados podem ser compartilhados entre os computadores conectados a rede.

A capacidade de interligar os diversos computadores através de um barramento externo de alta taxa de transferência é o início para podermos criar os Sistemas Distribuídos.

> Esta constante demanda por poder computacional vem gerando a necessidade de processadores cada vez mais rápidos para viabilizar aplicações com elevada taxa de computações e diminuir o tempo de resposta desejado pelos usuários. Na computação de alto desempenho, utilizada para programação científica, multimídia, gerenciamento de grandes volumes de dados etc., a solução passa por máquinas com múltiplos processadores ou ainda clusters proprietários fornecidos por grandes empresas. Ambas as soluções são custosas e de pouca escalabilidade. (PITANGA, 2008, p. 10)

A arquitetura de cluster apresenta vantagens em relação aos ambientes multiprocessados onde possuem vários processadores na mesma placa-mãe, pois permitem que o acréscimo de computadores na rede aumente o desempenho do sistema.

3.1 SISTEMAS DISTRIBUÍDOS

Grande parte dos sistemas distribuídos tem por objetivo prover maior desempenho computacional.

Os sistemas distribuídos, sob o aspecto de arquitetura de máquinas para a execução de aplicativos, devem ser vistos como configurações com grande poder de escala pela agregação dos computadores existentes nas redes convencionais (DANTAS, 2005).

O autor Pitanga (2008) definiu os sistemas distribuídos com um conjunto de elementos que se comunicam por meio de uma rede interconexão, utilizando software de sistema distribuído.

O objetivo dos sistemas distribuídos está ligado na melhoria da comunicação entre os computadores, este é o ponto crítico no incremento de desempenho da comunicação entre os processos (PITANGA, 2008).

Umas das grandes dificuldades encontradas nos sistemas distribuídos estão relacionadas ao software, tudo isto devido à elevada complexidade no seu desenvolvimento onde exige um vasto conhecimento do desenvolvedor. Nos ambientes distribuídos a dificuldade encontrada está na heterogeneidade de um conjunto de máquinas, onde cada computador possui uma arquitetura diferente de hardware executando seu próprio sistema operacional.

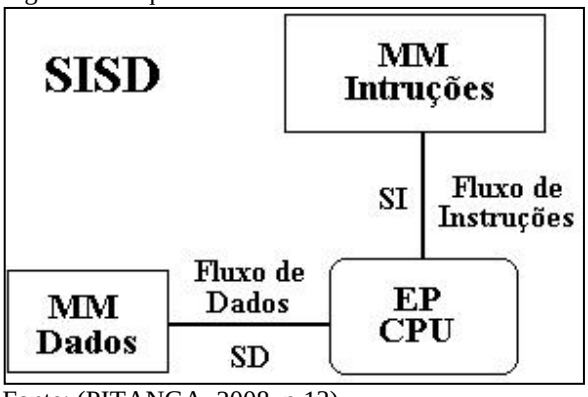
3.2 TAXONOMIA DE FLYNN

A grande diversidade de tipos de arquiteturas de computadores, uma grande quantidade de taxonomias já foram propostas, para tentar obter um padrão coerente de características dos diferentes sistemas computacionais. Uma das classificações de taxonomias mais aceitas na área de arquitetura de computadores é a Taxonomia de Flynn (DANTAS, 2005).

A classificação de Flynn desenvolvida a mais de trinta anos e valida até hoje considera o total de instruções executadas em paralelo versus o conjunto de dados em que as instruções são dominadas. A taxonomia de Flynn estabeleceu as seguintes classificações de computadores:

> Arquitetura SISD - (SingleInstruction Single Data) é a identificação mais simples, onde o equipamento é considerado seqüencial, pois só é executada uma instrução por vez para cada dado enviado. São as maquinas *Von Neuman* tradicionais, sem qualquer paralelismo. São exemplos nessa categoria as máquinas monoprocessadas (PC, MAC, Workstation Sun) (PITANGA, 2008, p. 12).

Figura 2 - Arquitetura SISD



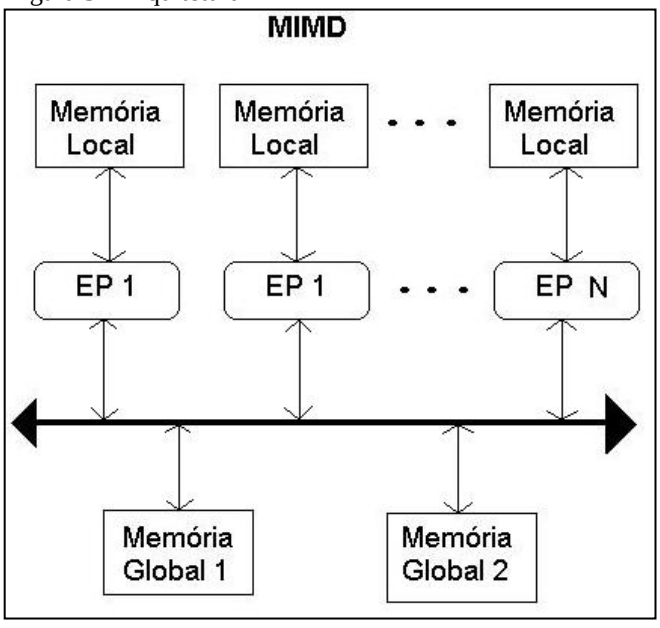
Fonte: (PITANGA, 2008, p.13)

> Arquitetura MISD – (MultipleInstruction Single Data) seriam máquinas que executam várias instruções ao mesmo tempo sobre um único dado. Torna-se até difícil imaginar uma máquina nestas condições. Este tipo de classificação não possui representante e até mesmo Flynn duvidou que algum dia isso pudesse existir. Mas poderíamos criar por essa teoria uma máquina com vários processadores tentando quebrar um código de criptografia, mas com a arquitetura MIMD poderíamos resolver este problema sem precisar construir uma maquina para este uso especifico (PITANGA, 2008, p. 12).

> Arquitetura SIMD – (Single InstructionMultiple Data) é o equivalente ao paralelismo de dados, onde uma simples instrução é executada paralelamente utilizando vários dados de forma síncrona, em que se executa um único programa ao mesmo tempo. Observe que neste contexto também se encaixa, pela característica, a tecnologia MMX (MultiMediaeXtension) existente dentro dos processadores atuais e nos computadores vetoriais com Cray 1, CM-2. Nas máquinas SIMD temos menos hardware, menos memória e um hardware muito mais especifico (PITANGA, 2008, p. 13).

> Arquitetura MIMD – (MultipleInstruction, Multiple Data) refere-se ao modelo de execução paralela na qual cada processador está essencialmente agindo independentemente, havendo, portanto, múltiplos fluxos de instruções e múltiplos dados com se fossem um conjunto de máquinas SISD em que cada processador é capaz de executar um programa diferente. São exemplos nessa categoria servidores com múltiplos processadores, sistemas MPP e os clusters de computadores. Aparentemente nos da uma visão de maior custo e pode-se usar processadores de uso geral. Há forte necessidade de um hardware a parte para obtenção de uma sincronização rápida (PITANGA, 2008, p. 13).

Figura 3 - Arquitetura MIMD



Fonte: (PITANGA, 2008, p.12)

Classificação de Flynn torna-se bastante imprecisa para classificar as variedades de multiprocessadores existentes. A categoria MIMD engloba uma grande diversidade de máquinas e com dificuldade para a classificação das mesmas. A forma mais usada é uma

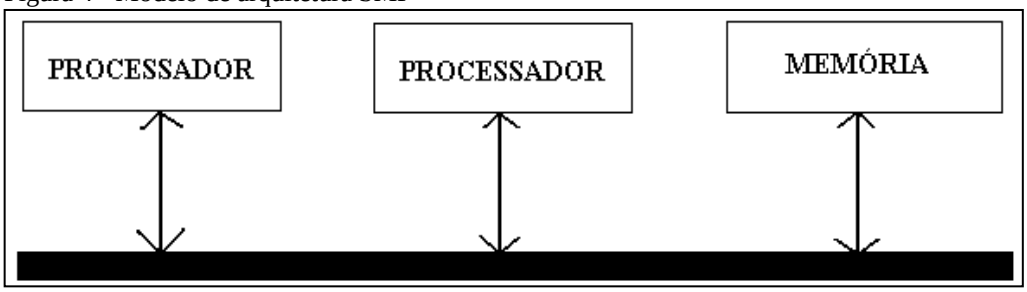
classificação que usa como critério a forma como a memória central esta fisicamente organizada e o tipo de acesso que cada processador tem a totalidade da memória, classificando, assim, esta arquitetura em computadores MIMD de memória compartilhada e memória distribuída (PITANGA, 2008).

3.2.1 Arquitetura de MIMD de memória compartilhada

Nesta classe de arquitetura incluem-se todas as máquinas de múltiplos processadores que compartilham um espaço de endereço de memória (PITANGA, 2008).

Os processadores e a memória interligam-se por meio de seu sistema local de interconexão, provendo por um intermédio de uma barra a facilidade de configuração compartilhada (DANTAS, 2005).

Figura 4 - Modelo de arquitetura SMP



Fonte: (PITANGA, 2008, p.15)

A vantagem desta arquitetura está no compartilhamento de dados entre os processos.

E como desvantagem pode-se citar os computadores de altíssimos custos. A existência de uma limitação física para quantidade de processadores e também na necessidade de técnicas de sincronização e leitura e escrita.

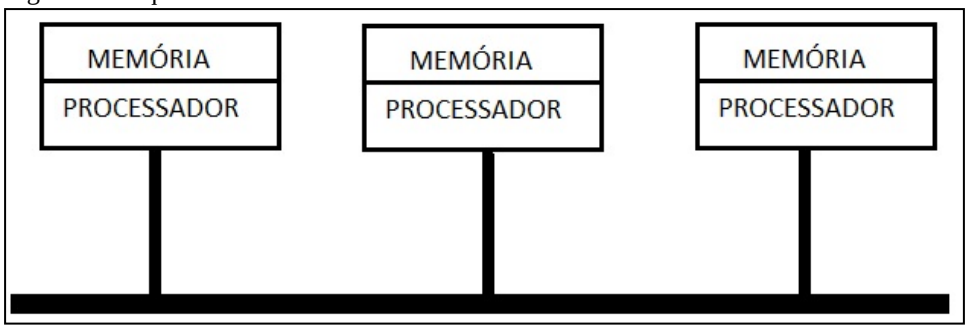
3.2.2 Arquitetura MIMD de memória distribuída

Esta classe de arquitetura inclui-se as máquinas formadas por várias unidades processadoras cada com a sua própria memória, chamada de *multicomputadores* (PITANGA, 2008).

Nesta situação não existe qualquer tipo de memória comum. A comunicação entre os nós diferentes do sistema é feita através de dispositivos físicos de entrada e saída (PITANGA, 2008).

A desvantagem da memória distribuída está na dificuldade de programação por ser muito complexa e complicada. Isto porque existe a necessidade que as tarefas enviem mensagens de forma explicita enquanto outras tarefas estão em execução em outro processador. Ainda possui outra desvantagem referente a exigir muita comunicação, o desempenho fica comprometido no alto custo envolvido na troca de mensagens (PITANGA, 2008).

Figura 5 - Arquitetura de Memória Distribuída



Fonte: (PITANGA, 2008, p. 16)

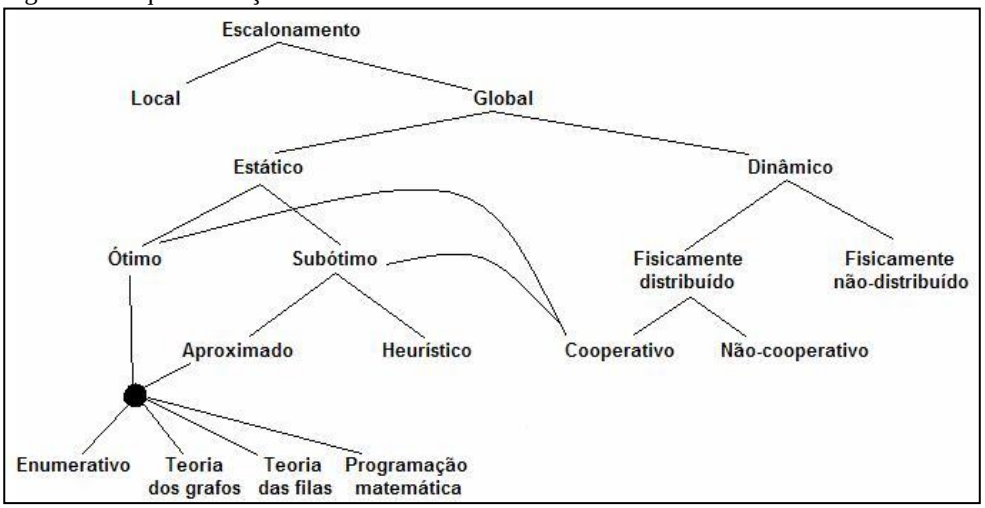
3.3 TAXONOMIA DE CASAVANT

Uma das taxonomias que mais se destacam é a de Casavant, por ser muito abrangente e de grande aceitação.

A taxonomia busca meios de explorar da melhor forma possível os recursos de um sistema distribuído em um tempo viável, um grande número de técnicas e metodologias foram desenvolvidas, juntamente com diversas formas de classificá-las e organizá-las (SILVA, 2006).

Essa classificação computacional visa representar em uma árvore hierárquica de algoritmos distribuídos. O esquema dessa taxonomia é apresentado conforme figura 6.

Figura 6 - Esquematização de Taxonomia de Casavant



Fonte: (SILVA, 2006, p. 3)

A hierarquia começa pela divisão de local e global, mais a divisão mais significativa fica por conta da estática e dinâmica, onde se define quais as decisões de escalonamento serão tomadas. Torna-se mais importante, pois há situações onde os algoritmos dinâmicos são mais efetivos do que os estáticos (SILVA, 2006).

Nos dois casos o processo a ser executado é divido em tarefas, e a precedência entre as tarefas são representadas por grafos. Os algoritmos estáticos assumem em que todas as informações sobre as tarefas são apresentadas antes do tempo de execução, enquanto que nos algoritmos dinâmicos é assumido sobre as necessidades das tarefas, iterações sobre um laço de execução e também no ambiente em que ela será executada (SILVA, 2006).

> Na divisão entre ótimo e subótimo, no primeiro caso se considera que todo o conhecimento relacionado ao estado do sistema são conhecidos quando algum destes problemas é computacionalmente intratável ou não pode ser medido com certa exatidão, uma abordagem subótima é estimada. Seguindo-se por este caminho, chega-se à divisão entre aproximada e heurística. Numa solução aproximada, utiliza-se o modelo formal padrão do processo e se busca a primeira solução considerada razoável no espaço de soluções possíveis, enquanto que uma solução heurística faz assunções realistas sobre o conhecimento a priori das tarefas baseadas em parâmetros que afetam o sistema como um todo, obtendo em um tempo menor do que o exponencial uma solução próxima à ótima. Em seguida, podem-se ver as categorias básicas de algoritmos usados nesses tipos de escalonamento, isto é, algoritmos baseados em teoria dos grafos ou teoria das filas, em programação matemática e em enumeração no espaço de soluções, que são comuns tanto à abordagem ótima quanto à subótima aproximada (SILVA, 2006, p.3).

Voltando-se na hierarquia, chega-se à divisão entre distribuído e não-distribuído, reside em um único processador, que escalona as tarefas e direciona-as para os demais processadores. A última divisão diz respeito a trabalho cooperativo ou não-cooperativo. No primeiro, as decisões são tomadas cooperativamente entre os processadores; no último, cada

processador toma uma decisão independentemente dos demais, baseado apenas em parâmetros locais, ou seja, a melhor decisão local. Note que o nodo cooperativo da árvore também se subdivide em ótimo e subótimo, tratados anteriormente (SILVA, 2006).

3.4 SPEEDUP

Seja P um problema computacional e seja n o tamanho da entrada. Denote-se a complexidade sequencial de P por T(tempo)*(n). Pode-se então desenvolver um algoritmo sequencial que resolve P com esse tempo mínimo, podendo comprovar que nenhum algoritmo sequencial pode resolver P de forma mais rápida. Seja A um algoritmo paralelo que resolve P num tempo Tp(n) em um computador paralelo com p processadores (YANO, 2010).

Segundo Yano(2010), *speedup* alcançado por A como:

$$Sp(n) = T^*(n) / Tp(n)$$

O valor Sp(n) mede o *speedup* obtido por um algoritmo A quando p processadores estão disponíveis para utilização. O objetivo é projetar algoritmos paralelos que alcancem Sp(n) melhores de acordo que se introduzem mais processadores p (YANO,2010).

Existem vários fatores que introduzem ineficiência nos algoritmos paralelos. Entre eles, destaca-se os atrasos na comunicação, a sobrecarga ocorrida na sincronização das atividades dos vários processadores e no controle do sistema (YANO, 2010).

Nota-se que T1(n), é o tempo do algoritmo paralelo A quando o número p de processadores é igual a 1, não necessariamente o mesmo que T*(n), portanto, o *speedup* é medido relativamente de acordo com o melhor algoritmo sequencial possível (YANO,2010).

4 CLUSTER

Quando se deseja um maior desempenho para o processamento de um conjunto de operações computacionais ou até mesmo uma única operação mais complexa, as opções seriam trabalhar mais intensamente, trabalhar com mais eficiência ou pedir ajuda (DANTAS, 2005).

Na utilização de processadores mais potentes para obter a execução mais rápida da operação, já tem sido limitada por uma série de fatores prevista por Gordon Moore em 1965, onde os limites da própria física façam que os processadores cheguem a seu ápice de processamento. Limitações como a miniaturização do processador chegando a um ponto onde não mais área existente para agregar elementos eletrônicos de processamento.

Outra forma de se obter melhores desempenhos é o desenvolvimento de algoritmos eficazes. Porém, a utilização de algoritmos mais eficientes não garante melhores resultados para inúmeras aplicações. A conclusão é que a grande quantidade de dados a serem processados pela operação seja o problema para obter processadores mais potentes.

A outra opção citada por Dantas (2005) pedir ajuda, resumindo para uso da computação é a utilização dos sistemas paralelos e distribuídos. A criação de um ambiente agregado de computadores processando de forma paralela os ambientes de cluster permitindo melhoria no desempenho das aplicações.

O *Cluster* (que do inglês significa agrupamento) em termos de arquiteturas de hardware, é uma unificação de vários computadores processando de forma dedicada ou não, para execução de aplicações especificas (DANTAS, 2005).

Normalmente são formados por computadores comuns do tipo PC Desktop, pertencendo a uma única organização ou unidade.

A estrutura do *cluster* pode ser classificada como dedicadas e não dedicadas. Uma arquitetura dedicada é quando criamos a exemplo *pilhas de PCs (Beowulf)* onde todos os nós são de uso exclusivo para o cluster. Nos cluster não dedicados, cada um dos nós são estações de trabalho na organização e em momentos de ociosidade e são utilizados para adicionar processamento.

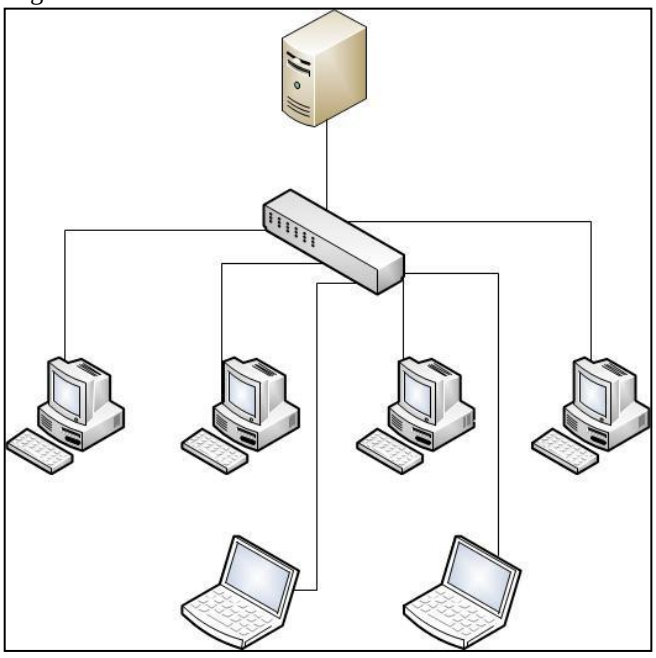
Figura 7 – Cluster dedicado Beowulf

Fonte: http://www.cse.mtu.edu/cseri.html

A escalabilidade é um fator diferencial do *cluster*, onde os recursos crescem a cada nó acrescentado a ele. (DANTAS, 2005).

Na criação de um ambiente não dedicado, as tarefas executadas são monoprocessadas, e pode ser utilizado para execução de aplicações que solicitem um grande poder computacional. Esses *clusters* não dedicados, normalmente são compostos por computadores com a disponibilidade de uma grande quantidade de recursos (memória, processadores e capacidade de armazenamento).

Figura 8 - Cluster não dedicado



Fonte: do autor.

Nos *cluster*s não dedicados cada computador ou nó adicionado deve-se ter instalado pacotes de software no sistema operacional.

4.1 BENEFÍCIOS

Dentre as inúmeras vantagens de se utilizar um cluster, Pitanga (2008) define as principais:

a) alto desempenho: Possibilidade de resolver problemas muito complexos exigindo alto poder de processamento e que por meio do processamento paralelo diminua o tempo de resolução do problema;
b) escalabilidade: Possibilidade de que novos componentes sejam adicionados à medida que cresce a carga de trabalho;
c) tolerância a falhas: O aumento da confiabilidade do sistema com um todo, caso alguma parte falhe;
d) baixo custo: a Redução de custo para se obter processamento de alto desempenho utilizando-se computadores simples ao contrário de se obter um supercomputador com custos altíssimos;
e) independência de fornecedores: Utilização de hardware aberto, software de uso livre e independência de fabricantes;
f) transparência: Capacidade de se apresentar como um sistema único.

4.2 TIPOS DE CLUSTER

Os *cluster* podem ser classificados em diversas categorias e funcionalidades, cada classificação com a suas próprias características especificas. Seja para aumentar o poder de processamento de algum serviço, manter um sistema de nível crítico 100% no ar, ou por fim juntar as duas características em um único projeto.

4.2.1 Alta disponibilidade

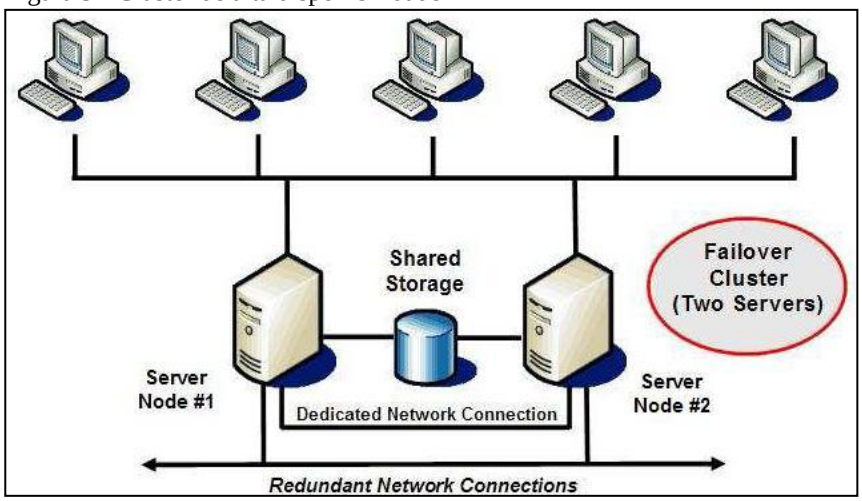
A alta disponibilidade vem para resolver problemas de missão crítica, onde o serviço nunca pode parar. Tem função principal deixar um sistema no ar vinte e quatro horas por dia, onde paradas não planejadas influenciam diretamente na qualidade do serviço e os prejuízos financeiros muito altos.

No *cluster* de alta disponibilidade os equipamentos funcionam paralelamente para manter o serviço sempre ativo, replicando serviços entre os nós do cluster, evitando paradas, ociosidades no desempenho, na espera que apenas um equipamento ou serviço deixe de funcionar e os demais servidores passem a responder por ele, porém poderá ter perda de

desempenho e poder de processamento, mas o principal objetivo será alcançado, ou seja, não interromper o serviço.

Segundo Pitanga (2008), a disponibilidade do sistema torna-se então uma questão vital para a sobrevivência de uma organização. Um sistema de comércio eletrônico, como por exemplo, vendas de livros, não admite indisponibilidade.

Figura 9 - Cluster de alta disponibilidade



Fonte: (SILVA, 2009, p.43)

4.2.2 Alto desempenho

Este tipo de cluster tem como foco principal o desenvolvimento de um supercomputador (que é nossa meta de estudo neste projeto).

Um cluster de computadores é uma das soluções alternativas em organizações que exigem processamento de alto desempenho na resolução de problemas por meio de aplicações paralelizáveis, com um custo razoavelmente baixo se comparado ao custo de investimento na compra de um supercomputador com o mesmo desempenho.

Com evolução dos softwares, exigindo cada vez mais poder de processamento dos computadores e também o elevado custo das arquiteturas de alto desempenho, exigiu a necessidade de criar um modelo de alto desempenho utilizando computadores comuns, mais que combinados em conjunto pode ser então atingir o desempenho esperado para execução de aplicações paralelas e distribuídas.

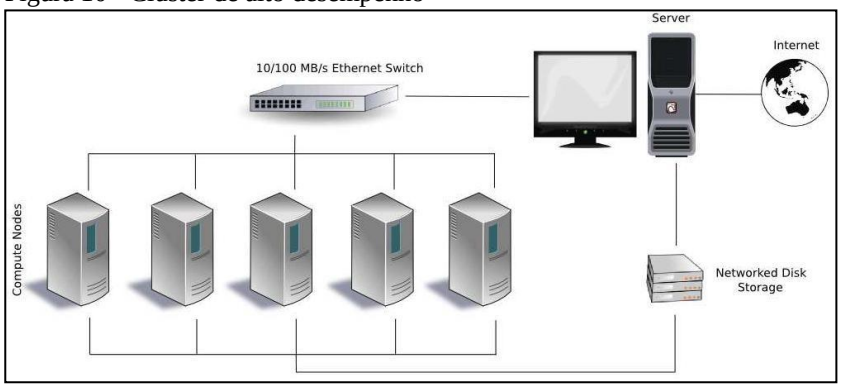
A vantagem deste tipo de *cluster* é muito lógica, pois ao somar as estações de trabalho comum para alcançar o mesmo desempenho de um supercomputador teremos um custo muito inferior.

A dificuldade então está no desenvolvimento de componentes e sistemas de propósito geral, tais como redes de estações de trabalho, em contraste com o uso de arquiteturas especializadas, como *Cray* e *SGI*, desta forma podendo obter taxas de processamento próximas a dos supercomputadores proprietários com um custo bem mais abaixo (PITANGA, 2008).

Essa arquitetura é muito interessante, pois para o aumento de desempenho sobre demanda, ou seja, de acordo com a necessidade de processamento exigido, podemos adicionar mais nós (PCs) ao cluster.

Ao longo do tempo algumas plataformas experimentais dentro de laboratórios de pesquisas e nas próprias universidades foram pesquisadas. No começo eram apenas pequenos protótipos de clusters com poucos nós, e então o desafio era o software a serem utilizados na arquitetura.

Figura 10 - Cluster de alto desempenho



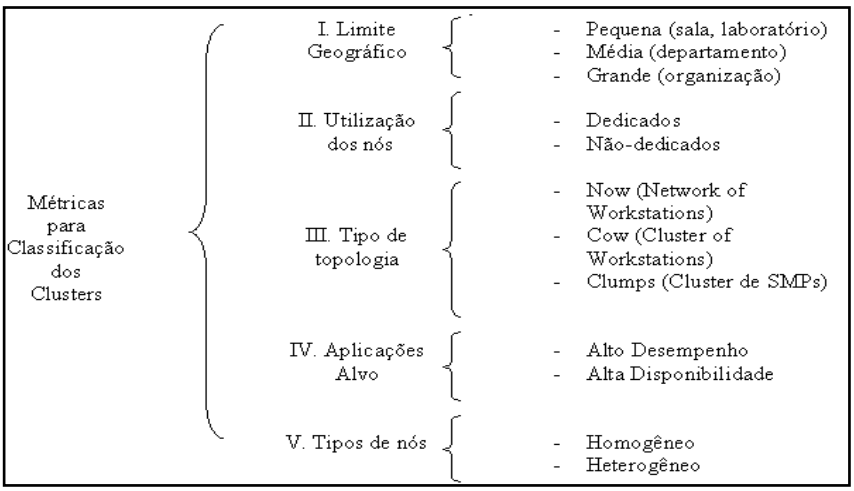
Fonte: (SILVA, 2009, p.47)

4.2.3 Métricas para classificação de cluster

Presentemente não se dispõem de uma taxonomia globalmente aceita para os ambientes de clusters Contudo, efetua-se uma classificação de tipos de clusters através da observação de algumas características.

Conforme a figura 11 classificou-se cinco tipos de métricas que podem ser empregadas para classificar os clusters.

Figura 11- Métricas para classificação do cluster



Fonte: (DANTAS, 2005. p. 148.)

4.3 APLICAÇÕES COM O CLUSTER

É muito diversificada a área de aplicações dos *clusters*. Em qualquer área ou organização que necessite de um grande poder de processamento ou ainda onde tenhamos a necessidade de criar um ambiente que seja tolerante a falhas, onde a disponibilidade seja mais importante que o desempenho.

Os computadores pessoais cada vez se tornam mais potentes, devido ao fato de que novas tecnologias estão sendo empregadas na fabricação de novos componentes eletrônicos.

Os clusters paralelos têm uma importante participação no desenvolvimento cinematográfico para renderização de imagens e animações tridimensionais. São também utilizados na ciência e engenharia, para processarem cálculos muito complexos de projetos de redes neurais, análise genética, estatísticas, astrofísica entre outros.

A utilização dos clusters tem aumentado nos últimos 20 anos devido ao avanço dos estudos e pesquisas nas áreas medicinas e engenharia. Tais pesquisadores e organizações estão utilizando os clusters, pois necessitam incrementar escalabilidade, disponibilidade, desempenho a um nível supercomputacional e a um preço acessível.

Alguns exemplos de áreas onde a utilização de *cluster* são indicadas:

a) servidores de Internet: Com o grande crescimento de sistemas de Internet, também revela algumas vantagens em utilizar cluster para poder distribuir a carga e aumentar a capacidade de resposta;

b) segurança: Com a grande capacidade de processamento paralelo, o cluster contribuirá para o processo de identificação de quebra de criptografia e verificação de possíveis soluções;
c) banco de dados: Algumas pesquisas em banco de dados podem demorar um elevado tempo em um sistema comum. Com cluster podemos reduzir o resultado de uma pesquisa. É muito utilizado também, em aplicações alvo que requerem alta disponibilidade para prover que o sistema fique 100% no ar;
d) computação gráfica: nessa área já é muito comum encontrar cluster de alto desempenho para prover resposta mais rápida na renderização de imagens e a elaboração de um filme com cenas 3D;
e) análise de elementos finitos: Cálculos de barragens, pontes, navios, aviões, grandes edifícios, veículos espaciais;
f) previsão do tempo: Muito utilizado para prover dados mais precisos referentes ao clima geográfico.

4.4 PLATAFORMAS DE CLUSTER COMPUTACIONAL DE ALTO DESEMPENHO E DE BAIXO CUSTO

Cluster de processamento de alto desempenho (HPC – high performance cluster), essa categoria de cluster tem como principal objetivo a alta performance. Esses clusters se tornam alternativas viáveis para universidades e empresas de qualquer porte, pois podem obter alto desempenho em aplicações, a um custo razoavelmente baixo se comparado a aquisição de um supercomputador.

Alguns exemplos de cluster de alto desempenho serão mostrados a seguir.

4.4.1 Beowulf

Um dos mais importantes avanços da computação tem sido a grande crescimento computacional dos computadores pessoais PCs. Isso devido ao mercado de PCs ser muito maior que o mercado das Workstations, fazendo que o valor dos computadores pessoais decresça popularizando no mundo todo (PITANGA, 2008).

O cluster Beowulf foi criado em 1994, quando a agência espacial (NASA) precisava de um computador que processasse na ordem de um gigaflops, o que significa um bilhão de operações em ponto flutuante por segundo. Todavia, um supercomputador com este

poder computacional custaria para época o valor de um milhão de dólares, o que era considerado um investimento enorme apenas para atender um grupo de pesquisadores (PITANGA, 2008).

Os pesquisadores conseguiram então, agrupar 16 computadores pessoais contendo o processador 486(100Mhz), usando o sistema operacional Linux interconectados em uma rede ethernet, alcançando um poder computacional de 70 megaflops (PITANGA, 2008).

A plataforma Beowulf é composta por várias aplicações que possuem desempenho elevado e são gratuitas na maioria de suas ferramentas. Como exemplo pode-se citar os sistemas operacionais GNU/Linux e FreeBSD sobre os quais estão instaladas as diversas ferramentas que viabilizam o processamento paralelo, como é o caso das API's(*ApliccationProgramming Interface),*MPI (*MessagePassing Interface*) e PVM (*Parallel Virtual Machine*). Isto permitiu fazer alterações no sistema operacional Linux para dotá-lo de novas características que facilitaram o desenvolvimento de aplicações paralelas (MATOS, 2006).

O cluster é dividido em um nó controlador principal (nó mestre), cuja função é controlar o cluster, monitorar e distribuir os processos das aplicações como também migrando os processos pelos demais nós do cluster. Os nós secundários são exclusivamente dedicados a processar as tarefas que são enviadas pelo nó principal.

Todos os nós são responsáveis por atender aos pedidos de processamento. Se um dos nós falhar, as tarefas são automaticamente redistribuídas pelos demais nós disponíveis no momento.

4.4.2 Mosix

Mosix é um sistema de cluster computacional desenvolvido para Linux que prove gerenciamento de clusters, multi-clusters ou clouds orientado a alto desempenho.

Consiste em uma plataforma de sistema distribuído que provê uma transparência do sistema chamado de Sistema de Imagem Simples (SSI). O cluster é constituído de vários nós (computadores), onde através da troca de mensagens entre os nós e a comunicação preemptiva, o sistema possui uma abstração para usuário parecendo que possui apenas um só computador (COSTA, 2009).

4.4.3 Openmosix

O Openmosix nasceu da ramificação do projeto Mosix. Foi criado porque foi identificado que o Mosix estava se tornando uma ferramenta muito comercial e alguns pesquisadores acreditavam que o projeto deveria ser de código aberto (COSTA, 2009).

O idealizador do projeto foi Ph.D Moshe Bar que iniciou o projeto em 2002.

O Openmosix é uma extensão do núcleo do kernel do sistema operacional Linux, pois faz com que o cluster computacional se comporte como se fosse um único supercomputador.

Ele vem implementado a migração preemptiva de processos que quer dizer que o processamento computacional onde o kernel tem controle do tempo que será usado para cada processo, e tem poder de tomar de volta este tempo e entregar a outro processo segundo seu esquema de prioridades.

Para melhorar o desempenho, ele é controlado por algoritmos de balanceamento dinâmico de carga e prevenção contra falta de memória. Estes algoritmos foram projetados para responder a variação de utilização dos demais nós do cluster, garantindo a confiabilidade seja com qualquer quantidade de computadores no aglomerado (COSTA, 2009).

No Openmosix não existe a necessidade de configuração do nó mestre com o nó escravo com é feito no Cluster Beowulf. Cada nó do aglomerado é nó mestre para os processos que são criados localmente no seu sistema operacional é também um servidor de processos remotos onde migra para o demais nós do cluster. Isto confirma que podemos estar adicionando ou removendo nós do cluster sem nenhuma modificação de configuração ou parada no sistema.

O Openmosix possui também algoritmos de identificação do poder de processamento, a carga da CPU e a memória livre disponível para receber processos.

Como o Openmosix funciona de forma silenciosa e as operações são transparentes para as aplicações, podemos então executar aplicações sequenciais e paralelas como se fosse um único computador de multiprocessamento simétrico (SMP). Não é preciso saber onde os processos estão sendo executados, nem se preocupar com que os outros usuários estão fazendo na rede, por isso ele usa o acrônimo "forkandforget". O que ele faz é, pouco tempo depois de iniciar os processos, enviá-los para um melhor computador da rede, continuar a monitorar os novos processos e os demais, e movimentá-los pelos computadores com pouca carga de trabalho maximizando o trabalho e melhorando o desempenho (COSTA, 2009).

5 TRABALHOS CORRELATOS

Existem vários trabalhos relacionados a TI Verde, tais como técnicas de como aplicá-la, tudo isso devido aos sérios problemas causados pela tecnologia para com o meio ambiente. Para reduzir ao máximo os efeitos problemáticos que a tecnologia traz, muitos pesquisadores tem trabalhado em soluções mais sustentáveis.

5.1 PRÁTICAS DA TI VERDE QUE CONTRIBUEM PARA O DESENVOLVIMENTO SUSTENTÁVEL: UM ESTUDO DE CASO EM INDÚSTRIAS DO RN

Essa dissertação de mestrado por Glauber Ruan Barbosa Pereira em 2009, apresentada ao programa de Pós-graduação da Universidade do Rio Grande do Norte, que pesquisa técnicas que contribuem para o desenvolvimento sustentável nas indústrias. Segundo (PEREIRA, 2009) a TI Verde, dispõe de técnicas apropriadas, ligadas as atividades organizacionais tais como a reciclagem, reutilização, descarte correto do lixo eletrônico, racionalização da energia e também por fim a reeducação ambiental dos colaboradores na organização.

Com relação às técnicas de TI Verde Pereira (2009), observou um comparativo entre duas empresas onde ambas buscaram na TI Verde, fator de gerar sustentabilidade organizacional. Foi analisado que uma das empresas encontrou na TI Verde práticas que favoreceram o processo produtivo, mantendo uma postura proativa referente as questões sustentáveis, sempre na busca de certificações ambientais, para melhor competitividade no mercado. Embora a outra empresa analisado no estudo de caso buscou na TI Verde um caráter mais econômico.

Segundo Pereira (2009), mesmo com muitos estudos e incentivos, observa-se que as ações estão distantes de serem efetivamente executadas, pois ainda existem lacunas estruturais do próprio uso da TI, o que acaba desfocando o real objetivo da TI Verde que é o desenvolvimento sustentável.

5.2 LIXO ELETRÔNICO E A SOCIEDADE

Artigo desenvolvido pelo acadêmico Eduardo Ceretta Dalla Favera em 2008, apresentada a disciplina de Computadores e Sociedade do curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM), que se sensibiliza com a problemática ambiental e relata com clareza e objetividade um artigo sobre lixo eletrônico e a sociedade.

Segundo Favera (2008), o consumismo da sociedade é um dos fatores que mais fazem crescer a produção de lixo eletrônico. O autor desenvolve um artigo científico embasado em todo conteúdo de lixo eletrônico. Subdivide-se em vários capítulos: Introdução, consumismo, descarte, composição, componentes tóxicos, estatísticas, riscos a saúde, problemas sociais, Gana, iniciativas, Convenção da Basiléia e por fim completa com as diretivas para o lixo elétrico e equipamentos eletrônicos. Tal pesquisa científica foi muito importante para diagnosticar a importância das políticas de TI Verde para com as organizações e a sociedade.

6 PROJETO DO CLUSTER COMPUTACIONAL

Foram pesquisados dois modelos principais para o desenvolvimento do cluster de alto desempenho: Beowulf e o OpenMosix. A primeira plataforma a ser testada foi o cluster Beowulf, onde observou-se na pesquisa que a plataforma necessitava que as aplicações fossem projetadas para sistemas distribuídos, que não é o nosso caso, onde testaremos aplicações simuladas para desenvolvimento científico acadêmico. Já o Openmosix, possui ótimas ferramentas de visualização e gerenciamento do desempenho do sistema, com uma interface gráfica bastante atrativa possibilitando uma análise total dos processos dentro do cluster.

A partir da pesquisa procura-se encontrar uma solução eficiente e de fácil configuração, a fim de se obter resultados no cluster com o Openmosix.

6.1 DYNEBOLIC

O Dynebolic foi à primeira solução estudada, uma distribuição Linux que roda via live-cd, o que significa que o sistema operacional executa a partir do CD, bastando apenas dar *boot* pelo mesmo.

A distribuição traz nativa a plataforma Openmosix, aplicação responsável por distribuir os processos entre os nós encontrados na rede.

O objetivo do Dynebolic é proporcionar aos usuários ferramentas de manipulação e criação de multimídia, para isso ele vem repleto de aplicativos específicos para produção. O sistema operacional foi otimizado para trabalhar em computadores antigos de baixo processamento, tornando máquinas obsoletas em estações de produção multimídia. Um detalhe a ser mencionado, é que por ser uma distribuição projetada para computadores antigos, as aplicações nativas na distribuição estão em versões desatualizadas.

A última versão da distribuição foi lançada em 8 de setembro de 2011, com o codinome MUNIR.

Por ser uma distribuição com foco em produção multimídia, descarta-se a sua utilização no teste do cluster, pois há necessidade de um cluster para uso geral.

6.2 CLUSTER KNOPPIX

Essa distribuição Linux é baseada na distribuição Knoppix e readaptada para *cluster* incluindo a plataforma Openmosix, pronta para utilização.

O Cluster Knoppix é uma distribuição live-cd e praticamente tudo roda automaticamente, bastando apenas dar boot pelo CD que o sistema operacional inicializa.

As vantagens são as seguintes (PEREIRA, 2010):

a) é um sistema LIVE-CD, o que dispensa a utilização de disco rígido;
b) contém no seu kernel 2.4 o módulo Openmosix compilado;
c) vem com todas as ferramentas para a implementação de um cluster Openmosix;
d) inicializa automaticamente todos os serviços necessários;
e) possui *auto discovery*, que insere automaticamente novos nós no cluster, sem precisar reiniciar os serviços;
f) possui diversos aplicativos para uso cotidiano;
g) auto instalação de *drivers* da maioria dos hardwares.

6.3 CONSTITUIÇÃO FÍSICA DO CLUSTER

A necessidade de darmos alguma funcionalidade para os computadores descartados pela organização se teve a ideia de construir um aglomerado de computadores capazes de contribuir no alto processamento de alguma aplicação ou serviço. As máquinas utilizadas no projeto já estavam descartadas pela organização e estavam em depósitos como máquinas reservas.

Os critérios para o descarte das máquinas foram:

a) aplicações, serviços, sistemas, Internet já exigiam maior poder computacional;
b) alguma falha por hardware específico sem reposição no mercado local (placa USB, Portas de Comunicação).

Depois de estudarmos as plataformas de cluster computacional, o sistema operacional escolhido foi o Cluster Knoppix.

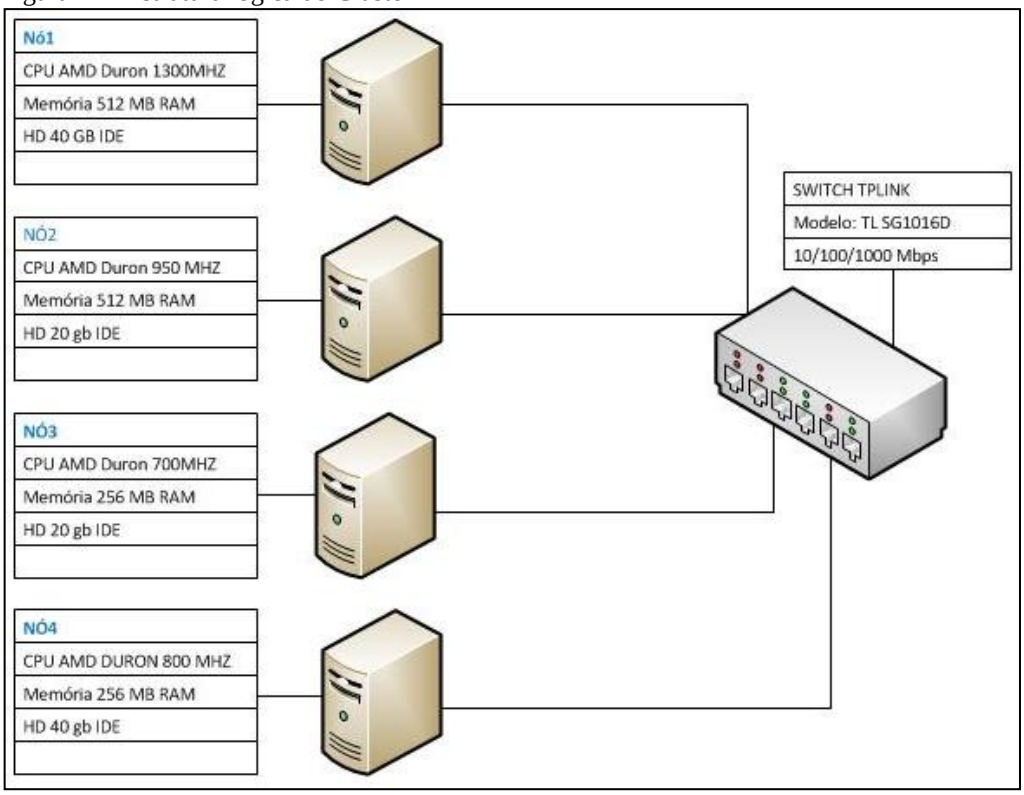
A estrutura física do aglomerado foi determinada pelos componentes já existentes pela organização.

O cluster é constituído por:

a) 01 switch 16 portas 10/100/1000Mbps TP-LINK para interligar 4 computadores;

b) nó 01 Principal: AMD Duron 1300mhz – 512mb SDR 133mhz – Placa de Vídeo Onboard 32mb – HD 20gb;
c) nó 02: AMD Duron 950Mhz – 512mb SDR 133Mhz – Placa de Vídeo Onboard 32mb – HD 20gb;
d) nó 03: AMD Duron 700Mhz – 256mb SDR 133Mhz – Placa de Vídeo onboard 8mb – HD 20 Gb IDE;
e) nó 04: AMD Duron 800Mhz - 256 SDR 133Mhz - Placa de Vídeo onboard 16mb – HD 40Gb IDE.

Figura 12 - Estrutura lógica do Cluster



Fonte: do autor.

No projeto do *cluster* foram inseridos no aglomerado máquinas mais homogêneas possíveis para garantir a confiabilidade dos dados.

Nós com configurações iguais não são obrigatórios, mas contribuem para diminuir problemas decorrentes de compilação de um código para arquiteturas diferentes, ou por terem também uma carga de trabalho desbalanceada, onde os nós mais rápidos irão terminar primeiro que os mais lentos e ficar esperando. Assim, a homogeneidade não é uma condição necessária para um cluster, mas é um fator que irá reduzir bastante a quantidade de problemas (PITANGA, 2002).

6.4 APLICAÇÃO DE TESTES NO AGLOMERADO

A partir da instalação do Cluster Knoppix em todos os nós do aglomerado (de acordo com manual de instalação apêndice - A), foram efetuados testes de desempenho do sistema rodando tarefas comuns de sistema operacional, a fim de verificar a migração de processo entre os nós do aglomerado.

Não foram feitas nenhuma alteração nas configurações do Openmosix, deixando as mesmas configurações nativas do sistema operacional.

Nos testes de migração de processo foi avaliado que as migrações entre os processos não eram suficientes para atender as necessidades do projeto, devido à exigência de processamento de aplicações básicas de Internet e editores de textos serem mínimas, fazendo que o Openmosix calcula-se a viabilidade de migração de processos, decidindo por manter os processos no nó principal.

Avaliando a necessidade de um algoritmo de alto processamento, foi pesquisada aplicação chamada Omtest disponível no link: http://www.openmosixview.com/omtest/#down

Segundo Rechenburg (2005), o teste de esforço é feito para testar o OpenMosix mais o kernel cluster. O Omtest irá executar diversas aplicações paralelamente, para testar o kernel, verificar a estabilidade e outras características do OpenMosix (por exemplo,a migraçãode processos, mfs). Durante este teste, o cluster será carregado principalmente por isso, deve-se parar de executar outros aplicativos antes de iniciá-lo.

O Omtest é constituído por várias aplicações, segue descrição:

a) Distkeygen: Esta aplicação é utilizada para gerar pares de chaves RSA 4000 com 1024 bits de comprimento da chave. Ele é distribuído em vários processos como processadores em seu cluster Openmosix via Fork;
b) Portfolio: Programa perl que simula uma carteira de diversas ações para um determinado período de tempo. Este método é a base para o livro "The intelligent asset allocator", que do inglês siginifica “Ativador de ativos inteligentes” de William Bernstein;
c) Eatmen: Simplesmente calcula o seno+Raiz quadrada de um valor 1.000.000 vezes e exibe na saída a contagem de loop para um arquivo. Este teste é iniciado em diversos períodos de uma só vez pelos vários processadores encontrados no cluster Openmosix automaticamente;
d) Forkit: Teste de forkit é semelhante ao teste de eatmem, mas usa fork para criar múltiplos processos (3 * [processors_in_your_openMosix_cluster]);

e) Timewaster: Isso irá criar um arquivo de 10 MB e copiá-lo para todos nós e para trás. É para verificar as capacidades do MFS.

O Projeto de Teste Linux é um projeto conjunto a SGI, IBM, OSDL, e Bull, com o objetivo de entregar conjuntos de testes para a comunidade de código aberto que irá validar a confiabilidade, robustez e estabilidade do Linux. O objetivo é melhorar o kernel do Linux por trazer a automação de teste para o esforço de teste do kernel (RECHENBURG, 2005).

O 'moving.sh' vai passar o 'start_openMosix_test.sh " em torno de cada nó no cluster openMosix durante a execução do teste de estresse. Então 'start_openMosix_test.sh' migrará a cada minuto para outro nó durante o teste de corrida. Independente de quanto tempo o teste será executado no cluster que será migrado 20 a 40 vezes (RECHENBURG, 2005).

6.4.1 Instalando o Omtest

Após o download da aplicação precisamos descompactar o arquivo. Para isso usamos o shell de comando. Acesse a pasta onde foi colocado o download do Omtest e execute os comandos:

gunzip omtest.tar.gz

tar -xvf omtest.gz

Após a descompactação precisamos acessar a pasta criada e compilar o programa. Para compilar precisamos ter previlégio de Administrador (root).

./compile_tests.sh

A compilação irá instalar os módulos necessários do Perl. Após compilação podemos executar o Omtest.

./start_openMosix_test.sh

6.4.2 Executando o Omtest

Durante o processo, a aplicação informa o tempo de início e o tempo final, sendo primordial para avaliar a evolução dos tempos conforme são adicionados mais nós ao aglomerado. Ao final, ele gera um relatório detalhado sobre cada componente que foi testado.

Com o cluster já configurado, os testes realizados foram os seguintes:

a) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) e documentar o desempenho;

b) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais o nó secundário (NÓ2) e documentar o a evolução de desempenho;
c) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais os nós auxiliares (NÓ2 e NÓ3) e documentar a evolução de desempenho;
d) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais os nós auxiliares (NÓ2, NÓ3 e NÓ4) e documentar a evolução do desempenho;

Identificamos em alguns testes anteriores que a quantidade de vezes para testar cada nó seria suficiente para definirmos uma média de resultados. Isso devido aos testes terem sido muito estáveis, não tendo muitas oscilações de tempo.

6.4.3 Ferramentas de análise

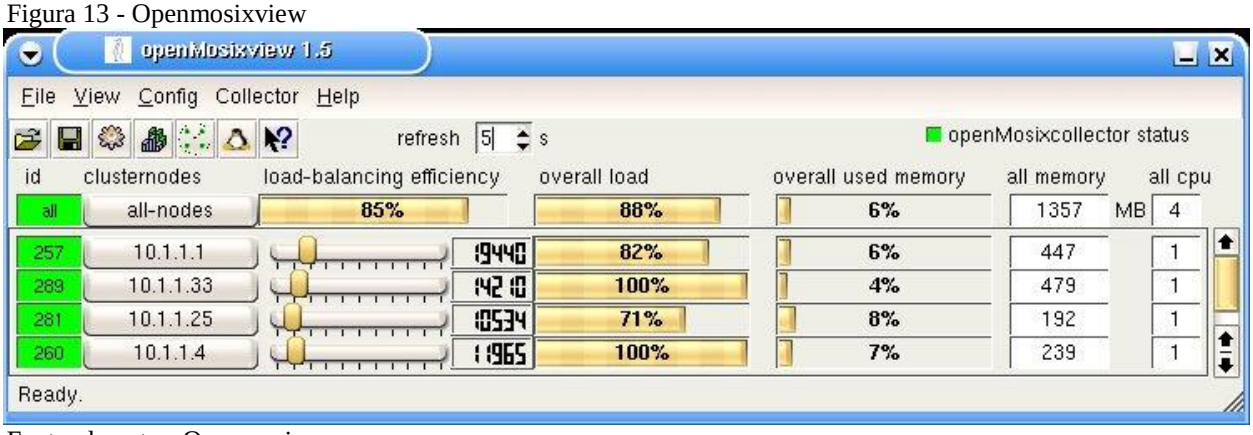
Com o cluster Knoppix instalado podemos visualizar o funcionamento do Openmosix com algumas ferramentas: Openmosixview, Openmosixmigmon e o Openmosixhistory.

Essas ferramentas ajudam a acompanhar melhor o andamento do aglomerado.

A ferramenta Openmosixview é uma ótima ferramenta de gerenciamento e visualização, possibilitando controle sobre a migração de processos.

Analisando a figura 13 podem-se identificar os seguintes campos do Openmosixview:

Figura 13 - Openmosixview



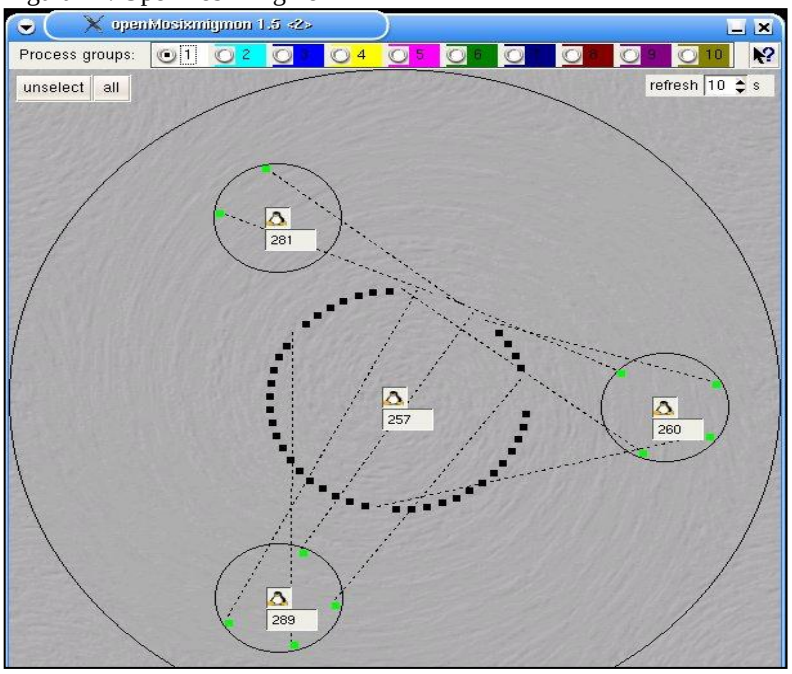
Fonte: do autor, Openmosix.

a) id: nome de cada nó. Esse campo podemos alterar o nome dos nós para uma melhor identificação no *cluster*;
b) clusternodes: endereço de IP de cada nó;

c) load-balancing efficiency: eficiência do balanceamento de carga do *cluster*;
d) overall load: carga de processamento de cada nó;
e) overall used memory: quantidade de memória utilizada no *cluster* e em cada nó;
f) all memory: total de memória do cluster e de cada nó;
g) all CPU: total de CPUs do *cluster* e de cada nó.

O Openmosixmigmon é uma ferramenta de análise de migração de processos do nó mestre para os demais nós do aglomerado. Identifica-se a migração de processos do nó 257 (mestre) para os nós secundários 260, 281 e 289. Essa ferramenta, apenas conseguimos visualizar o tráfego, não tendo opção de manipulação.

Figura 14: Openmosixmigmon

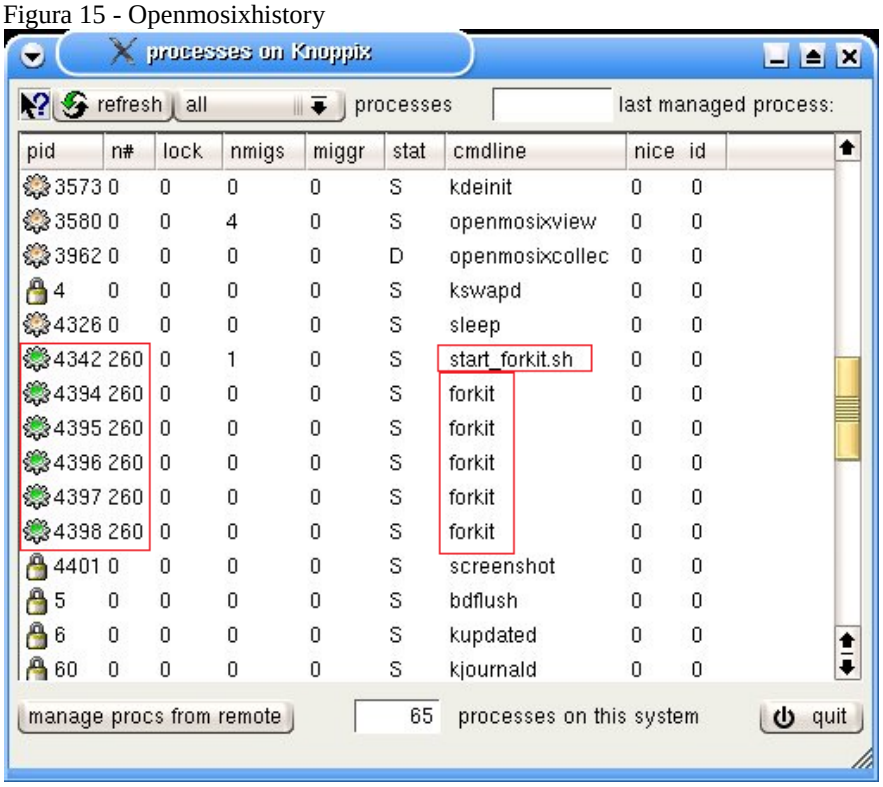


Fonte: do autor, Openmosix.

Outra ferramenta de análise é o Openmosixhistory, que contém informações dos processos. Notasse na figura 15 que os processos 4342, 4394, 4395, 4396, 4397, 4398 foram migrados para o nó 260. Na linha *cmdline* identificamos o nome do processo que foi migrado.

O Openmosixhistory é uma ferramenta muito importante, pois conseguimos aprofundar minuciosamente o estado em que processo se encontra, comprovando a migração de processos entre o aglomerado.

Figura 15 - Openmosixhistory



Fonte: do autor, Openmosixhistory.

O Openmosix é muito interessante, pois oferece aos administradores de TI ferramentas muito intuitivas que possibilitam um maior controle sobre aglomerado. Qualquer usuário está apto a usar o sistema sem ao menos perceber que as tarefas estão sendo distribuídas entre os nós (COSTA, 2009).

6.5 RESULTADOS OBTIDOS

Na execução deste projeto encontrou-se algumas dificuldades.

Inicialmente, realizou-se todos os 15 testes com a aplicação Omtest em cada nó e observou-se o desempenho de cada um, para então realizarmos a escolha do nó principal e os nós secundários.

Apartir da definição da sequência dos nós iniciou-se os testes no aglomerado.

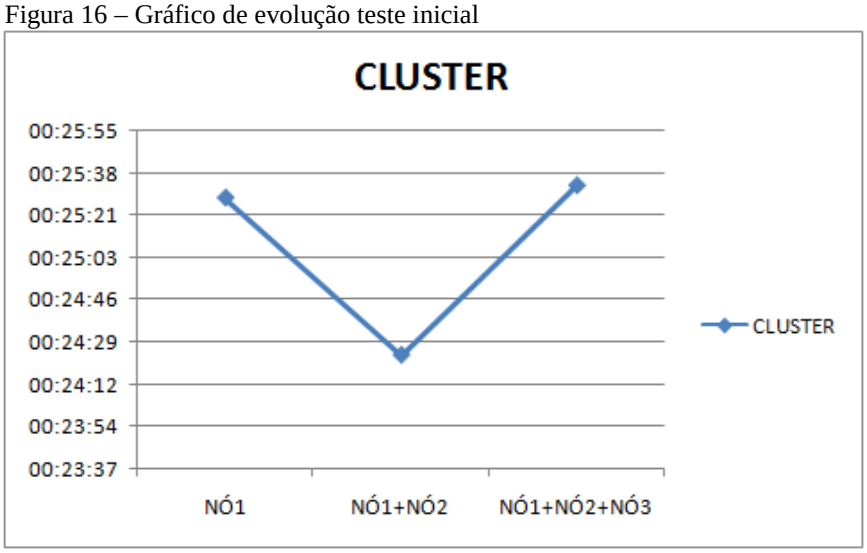
Os resultados obtidos seguindo as informações fornecidas pela própria aplicação, onde na amostragem informava o tempo inicial e o tempo final. O cálculo a ser feito era então diminuir o tempo final pelo tempo inicial.

Avaliou-se que ao rodar a primeiro nó obtivemos uma média de 25 minutos e 28 segundos. Ao rodar o primeiro nó juntamente com o segundo nó obteve-se uma média de 24

minutos e 23 segundos. O problema encontrado foi ao rodar juntamente com o terceiro nó que a média foi de 25 minutos e 33 segundos.

O gráfico a seguir demonstra de forma mais clara os resultados obtidos.

Figura 16 – Gráfico de evolução teste inicial



Fonte: do autor.

Ao verificarmos o problema acreditávamos que o gargalo estaria no tráfego de rede entre o nó. Para isso criamos uma rede dedicada Gigabit. Trocou-se as placas de rede de todos os computadores do aglomerado por placa de rede gigabit 10/100/1000Mbps, como também foi adquirido um switch TP-Link 16 portas gigabit 10/100/1000Mbps.

Novamente rodou-se os testes no aglomerado e os resultados não foram satisfatórios, pois ainda não conseguíamos ter uma melhora de tempo com a adição de mais nós ao aglomerado.

Paralelamente foi também testado a possibilidade de criar o cluster virtualizado com objetivo de remover trafego de rede, pois os nós estariam criados na mesma máquina. Para isso foi criado três maquinas virtuais utilizando o software VirtualBox 4.2.2 rodando em computador Intel Pentium Dual Core 1.73GHz com 2 GB de memória RAM.

Os resultados da virtualização também não foram satisfatórios, pois se verificou que os testes não obtinham melhores desempenhos, também acarretando problemas de travamento no computador virtualizado, devido ao excesso de carga de CPU.

A solução encontrada foi então estarmos executando o Omtest dividindo os resultados conforme as aplicações executem paralelamente, para verificarmos o ganho de desempenho conforme é adicionado nós ao aglomerado.

Os tempos foram calculados conforme resultados que a aplicação gerou. Documentou-se o tempo dos cálculos e ao final em azul temos uma média dos resultados obtidos.

6.5.1 Resultados Distkeygen

A primeira aplicação a rodar é o Distkeygen, e após rodar a aplicação 15 vezes os resultados foram os seguintes.

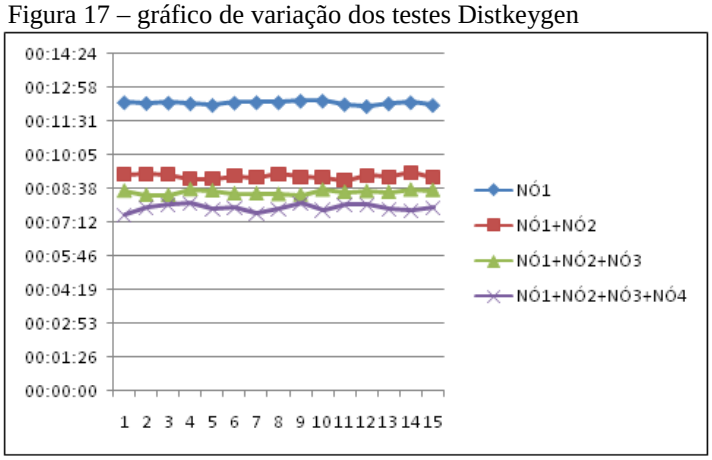
Tabela 1 – Resultados do Distkeygen

DISTKEYGEN			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:12:19	00:09:14	00:08:33	00:07:30
00:12:17	00:09:15	00:08:22	00:07:48
00:12:18	00:09:14	00:08:22	00:07:57
00:12:16	00:09:04	00:08:37	00:08:00
00:12:12	00:09:03	00:08:34	00:07:45
00:12:18	00:09:10	00:08:26	00:07:50
00:12:19	00:09:06	00:08:26	00:07:35
00:12:19	00:09:15	00:08:24	00:07:44
00:12:23	00:09:09	00:08:23	00:07:59
00:12:23	00:09:07	00:08:36	00:07:43
00:12:14	00:09:00	00:08:30	00:07:56
00:12:09	00:09:13	00:08:32	00:07:56
00:12:16	00:09:09	00:08:30	00:07:46
00:12:19	00:09:19	00:08:36	00:07:41
00:12:12	00:09:07	00:08:35	00:07:49
00:12:17	**00:09:10**	**00:08:30**	**00:07:48**

Fonte: do autor.

Criou-se um gráfico onde pode-se visualizar a variação de tempos dos testes:

Figura 17 – gráfico de variação dos testes Distkeygen

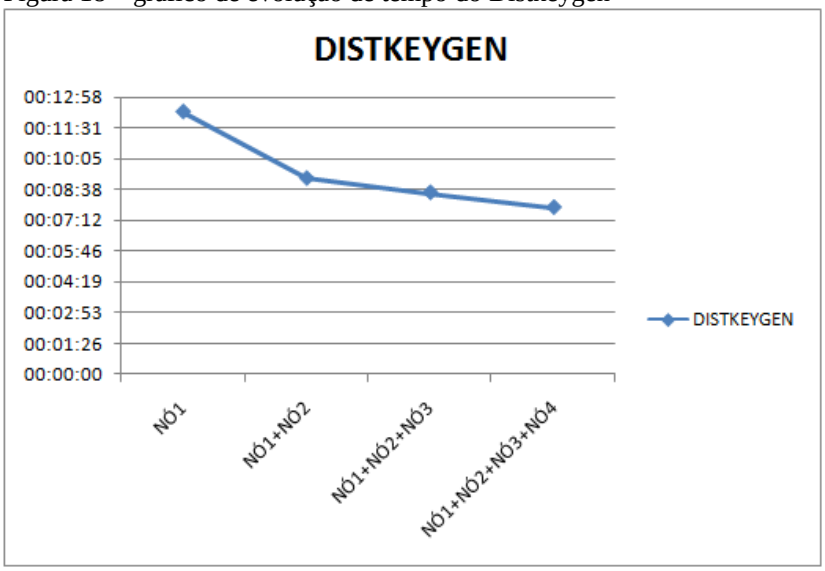


Fonte: do autor.

Nota-se uma variação muito pequena em todos os resultados obtidos do Distkeygen, garantindo uma estabilidade excelente.

Neste gráfico a seguir temos a evolução dos tempos.

Figura 18 – gráfico de evolução de tempo do Distkeygen



Fonte: do autor.

Conforme descrito anteriormente o teste Distkeygen gera pares de chaves RSA 4000 de 1024 bits de comprimento, concluindo com os resultados obtidos um melhor desempenho conforme adiciona-se nós ao aglomerado. Observa-se que aplicações de criptografia o cluster sejam bem viáveis para implementação. Os tempos foram diminuindo conforme aumentou-se o poder computacional do *cluster*.

6.5.2 Resultados Portfolio

A segunda aplicação que o Omtest executa é o *Portfolio.* A seguir temos a tabela de resultados obtidos em nosso aglomerado.

Tabela 2 – resultados do *Portfolio*

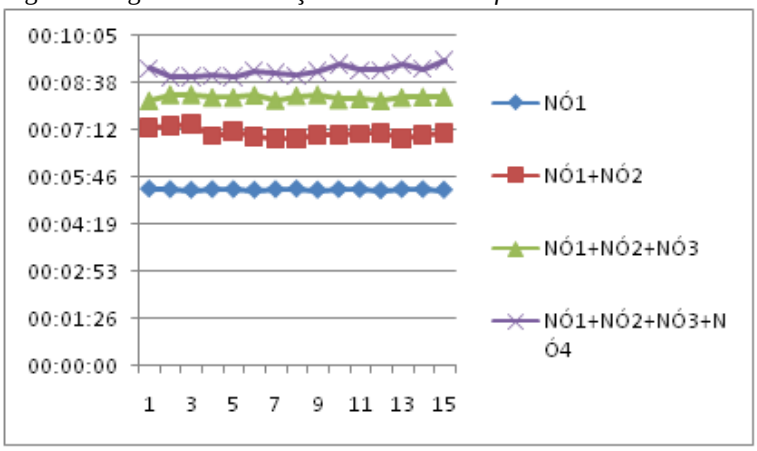
PORTFOLIO			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:05:24	00:07:15	00:08:05	00:09:05
00:05:23	00:07:19	00:08:16	00:08:50
00:05:22	00:07:23	00:08:16	00:08:50
00:05:23	00:07:01	00:08:11	00:08:51
00:05:23	00:07:10	00:08:11	00:08:49
00:05:22	00:07:00	00:08:15	00:09:01
00:05:23	00:06:57	00:08:06	00:08:56
00:05:24	00:06:56	00:08:14	00:08:52
00:05:22	00:07:03	00:08:16	00:08:59
00:05:23	00:07:04	00:08:07	00:09:12
00:05:23	00:07:05	00:08:09	00:09:02
00:05:21	00:07:07	00:08:05	00:09:02
00:05:23	00:06:57	00:08:11	00:09:13
00:05:23	00:07:04	00:08:12	00:09:04
00:05:22	00:07:06	00:08:12	00:09:20
00:05:23	**00:07:06**	**00:08:11**	**00:09:00**

Fonte: do autor.

O gráfico de variação figura 18, se mostrou também muito estável onde não tivemos muitas variações nos valores obtidos com os testes.

Valores muito próximos demonstram-se a estabilidade no desempenho dos testes, onde se consegue observar que os resultados não destoam tanto, assim não apresentando picos muito diferentes, esta homogeneidade, é importante em função de perceber que não existem flutuações de tempo, percebendo que o retardo de transferência existente na rede, não interfere significativamente nos testes.

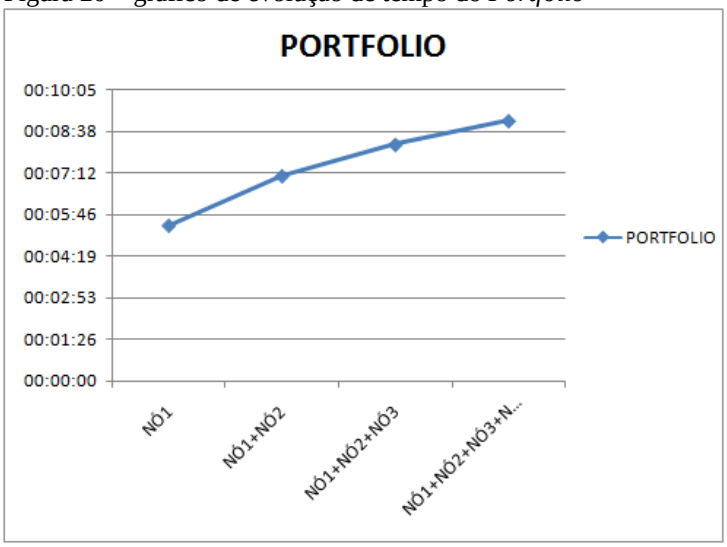
Figura 19 - gráfico de variação dos testes *Portfolio*



Fonte: do autor.

Neste gráfico a seguir temos a evolução dos tempos da aplicação do Portfolio, conforme avaliado verificou que o desempenho não foi satisfatório, mas comprovou-se a eficácia da aplicação, pois conforme análise da variação dos resultados o Openmosix se mostrou muito estável.

Figura 20 – gráfico de evolução de tempo do *Portfolio*



Fonte: do autor.

6.5.3 Resultados Eatmen

A terceira aplicação que o Omtest executa é o Eatmen. Conforme já havíamos explicado anteriormente o *Eatmen* é um aplicativo que efetua um cálculo de seno mais raiz quadrada de um número, calculando 1000000 vezes este valor distribuindo o cálculo por todos os nós automaticamente. Logicamente então se ele repete o mesmo cálculo por todos os nós não teríamos melhoras de tempo com o aglomerado.

A seguir temos a tabela de resultados obtidos em nosso aglomerado.

Tabela 3 – Resultados do *Eatmen*

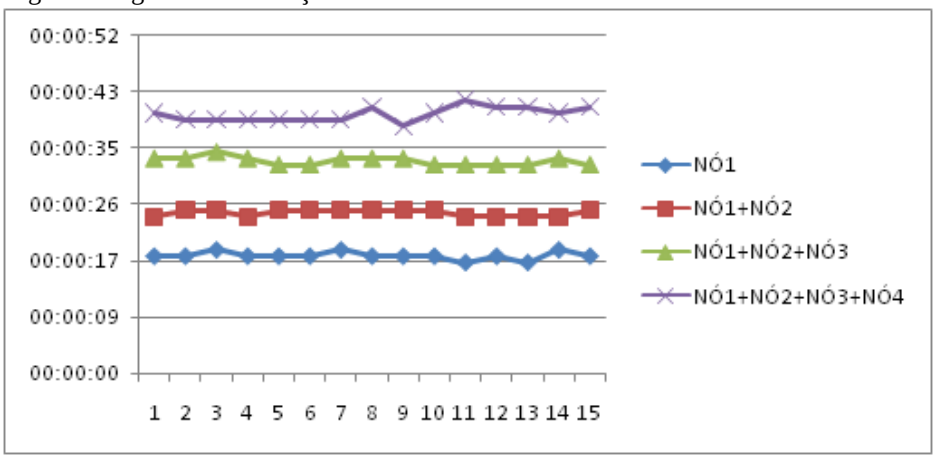
EATMEN			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:00:18	00:00:24	00:00:33	00:00:40
00:00:18	00:00:25	00:00:33	00:00:39
00:00:19	00:00:25	00:00:34	00:00:39
00:00:18	00:00:24	00:00:33	00:00:39
00:00:18	00:00:25	00:00:32	00:00:39
00:00:18	00:00:25	00:00:32	00:00:39
00:00:19	00:00:25	00:00:33	00:00:39
00:00:18	00:00:25	00:00:33	00:00:41
00:00:18	00:00:25	00:00:33	00:00:38
00:00:18	00:00:25	00:00:32	00:00:40
00:00:17	00:00:24	00:00:32	00:00:42
00:00:18	00:00:24	00:00:32	00:00:41
00:00:17	00:00:24	00:00:32	00:00:41
00:00:19	00:00:24	00:00:33	00:00:40
00:00:18	00:00:25	00:00:32	00:00:41
00:00:18	**00:00:25**	**00:00:33**	**00:00:40**

Fonte: do autor.

O gráfico da figura 20 se mostra estável, pois praticamente não obtivemos variação nos resultados.

A avaliação deste teste de esforço computacional se mostrou seguras em todos os nós do aglomerado.

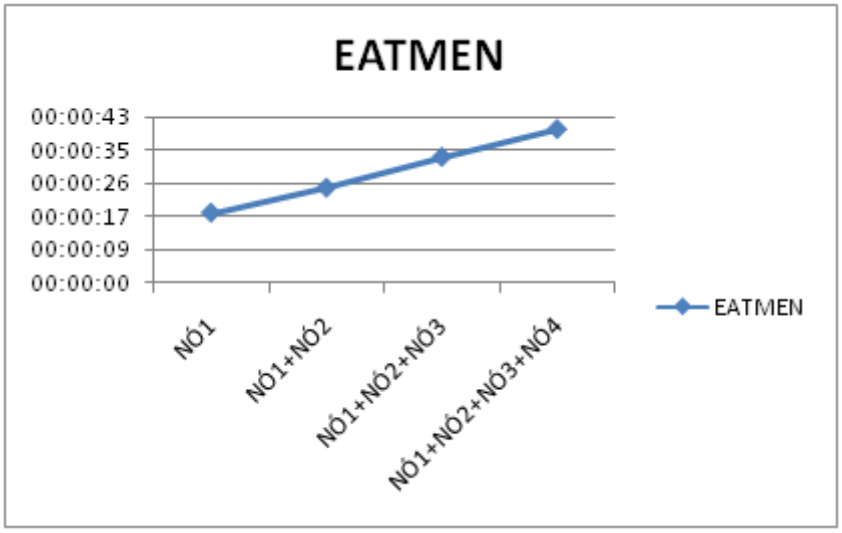
Figura 21 - gráfico de variação dos testes *Eatmen*



Fonte: do autor.

Conforme figura 21 verifica-se no gráfico que os tempos não tiveram melhores desempenho, por se tratar de uma aplicação de teste de esforço e não uma aplicação de simulação de desempenho de cluster computacional.

Figura 22 - gráfico de evolução de tempo do Eatmen



Fonte: do autor.

6.5.4 Resultados Forkit

A quarta aplicação a ser testada pelo Omtest é o Forkit. Uma aplicação semelhante a anterior Eatmen, mas que executa métodos Fork para criar múltiplos processos.

O Fork é um método de replicação de processos abertos na memória RAM. O método cria um novo processo (filho) a partir do processo original (pai). A diferença entre os dois processos que são criados diferentes identificação para cada processo. Quando o Fork é executado retorna-se 0 para processo filho e o id do processo filho para o pai, com isso conseguese identificar quem é o processo "filho" e quem é o processo original "pai" (SOUZA, 2007).

O método Fork é muito encontrado em banco de dados, quando ele é executado gera um processo filho responsável pelo processamento da conexão.

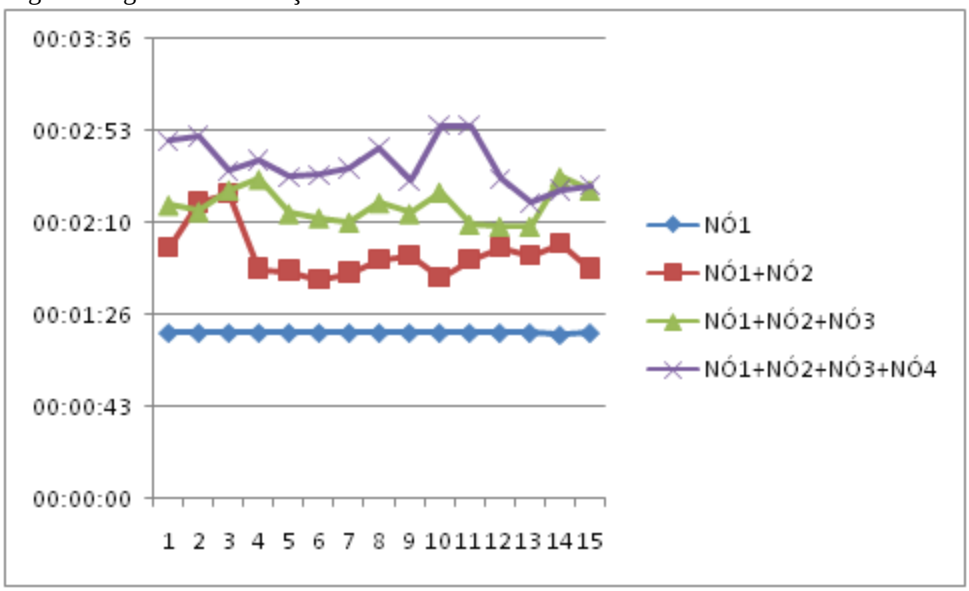
Tabela 4 – Resultados do *Forkit*

FORKIT			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:01:18	00:01:58	00:02:18	00:02:48
00:01:18	00:02:19	00:02:15	00:02:50
00:01:18	00:02:23	00:02:25	00:02:34
00:01:18	00:01:48	00:02:30	00:02:39
00:01:18	00:01:47	00:02:14	00:02:31
00:01:18	00:01:43	00:02:12	00:02:32
00:01:18	00:01:46	00:02:10	00:02:35
00:01:18	00:01:52	00:02:19	00:02:45
00:01:18	00:01:54	00:02:14	00:02:29
00:01:18	00:01:44	00:02:24	00:02:55
00:01:18	00:01:52	00:02:09	00:02:55
00:01:18	00:01:58	00:02:08	00:02:30
00:01:18	00:01:54	00:02:08	00:02:19
00:01:17	00:02:00	00:02:31	00:02:25
00:01:18	00:01:48	00:02:25	00:02:27
00:01:18	**00:01:55**	**00:02:17**	**00:02:37**

Fonte: do autor.

Na figura 22 a seguir, temos o gráfico de variação dos testes do *Forkit.* Nota-se que o gráfico se mostrou instável, comprovando certa ineficiência ao se utilizar métodos Fork em ambientes de *cluster.*

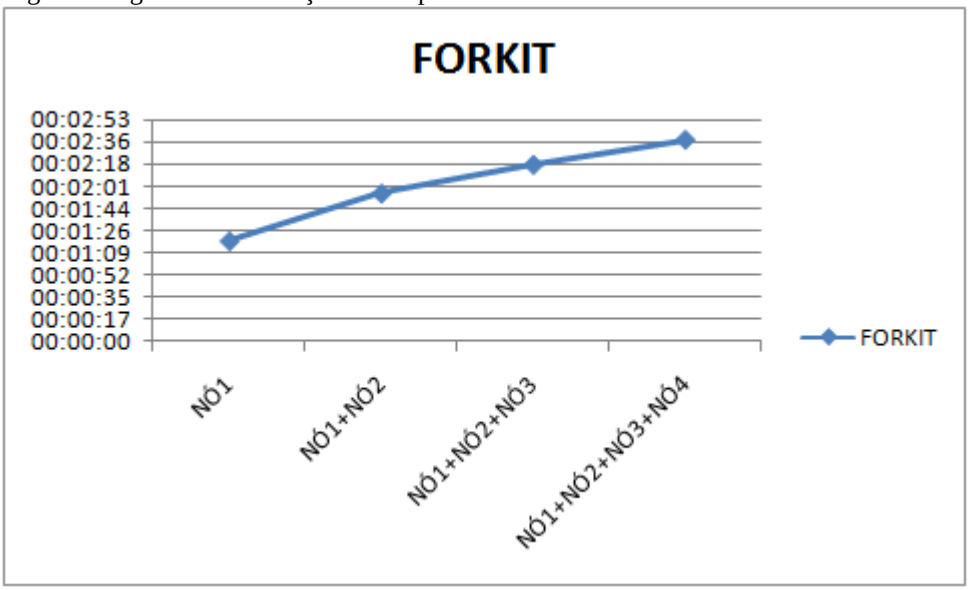
Figura 23: gráfico de variação dos testes *Forkit*



Fonte: do autor.

O gráfico da figura 24 demonstra a evolução dos tempos, na medida que se adicionam mais computadores no aglomerado.

Figura 24 - gráfico de evolução de tempo do *Forkit*



Fonte: do autor.

A aplicação *Forkit* não se mostrou eficiente para rodar em um ambiente cluster, mas apesar de não obter melhor desempenho vimos que o balanceamento de carga do método fork se mostrou estável e coerente. Isto devido ao Openmosix possuir ferramentas de análise que compravam a eficiência da migração de processos.

6.5.5 Resultados Timewaster

A última aplicação a ser testada pelo Omtest é o *Timewaster*. Uma ferramenta de teste de migração de dados entre os nós do aglomerado.

O *Timewaster* cria um arquivo de 10 MB e copia para todos os nós do cluster e os mesmos nós repete novos arquivos para o nó principal. Este teste foi criado para verificar o desempenho do MFS.

Teoricamente quanto mais nós no cluster, mais tempo leva-se para rodar o teste.

Tabela 5 - Resultados do *Timewaster*

TIMEWASTER			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:06:13	00:05:44	00:06:01	00:06:16
00:06:12	00:05:47	00:06:02	00:06:14
00:06:12	00:05:47	00:06:01	00:06:14
00:06:12	00:05:48	00:05:58	00:06:09
00:06:11	00:05:47	00:06:02	00:06:08
00:06:12	00:05:51	00:06:02	00:06:17
00:06:12	00:05:54	00:06:00	00:06:12
00:06:12	00:05:49	00:06:02	00:06:10
00:06:12	00:05:46	00:06:02	00:06:16
00:06:12	00:05:50	00:05:59	00:06:12
00:06:12	00:05:48	00:06:13	00:06:16
00:06:12	00:05:50	00:05:58	00:06:19
00:06:12	00:05:49	00:06:06	00:06:16
00:06:13	00:05:45	00:06:01	00:06:08
00:06:13	00:05:50	00:06:02	00:06:04
00:06:12	**00:05:48**	**00:06:02**	**00:06:13**

Fonte: do autor.

Na figura 24 temos o gráfico de variação dos testes da aplicação *Timewaster*, verifica-se que os testes não foram precisos, sofrendo variação entre os nós do cluster.

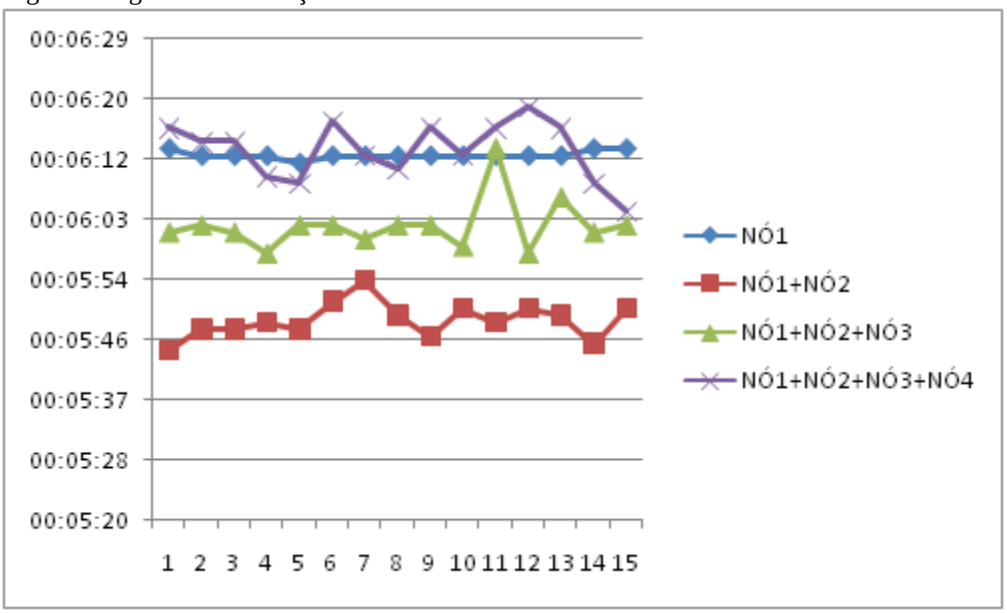
Podemos checar alguns problemas desta instabilidade:

a) tráfego de rede intenso;

b) problemas de discos para alocar os dados;

c) uso de CPU intenso, não tendo processamento para demais processos.

Figura 25 - gráfico de variação dos testes *Timewaster*

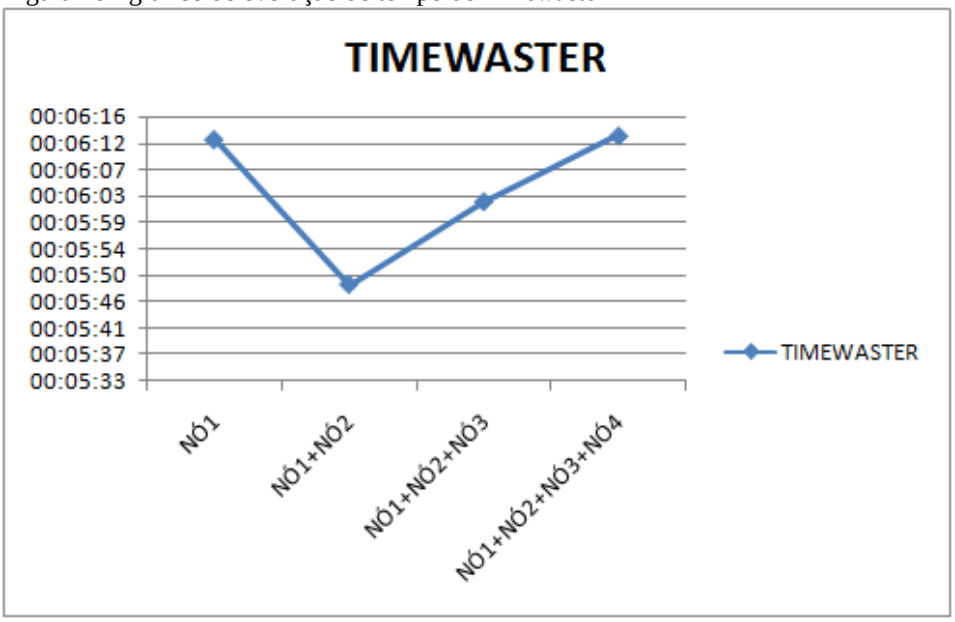


Fonte: do autor.

No gráfico da Figura 25, temos a evolução dos tempos no *Timewaster*.

Observa-se que os tempos foram aumentando a partir da adição de nós ao aglomerado.

Figura 26 - gráfico de evolução de tempo do *Timewaster*



Fonte: do autor.

Conforme figura 25 verificasse que o tempo entre o nó1 e o nó1+nó2 diminuíram, acredita-se a estar relacionado ao tempo de criação do arquivo. Com mais nós o arquivo cria-se mais rápido, no entanto demorasse mais tempo para replicar os dados na rede.

CONCLUSÃO

Este trabalho de conclusão de curso apresentou uma solução de TI Verde muito viável para qualquer tipo e tamanho de organização, podendo reutilizar as máquinas antigas já descartadas.

Implementou-se uma solução de cluster para as máquinas em desuso, de forma a se obter um melhor desempenho em alguma aplicação específica e que possamos aplicar em um mundo real. Tudo isso através de uma ferramenta muito intuitiva que é o Openmosix. Consegue-se então, aplicar alguns conceitos de computação paralela e distribuída que são o balanceamento de carga e a migração preemptiva de processos.

Os resultados obtidos conclui-se que algumas aplicações não são viáveis para um ambiente de cluster, pois não conseguiríamos melhorar o desempenho do processo realizado.

Concluí-se com todos os testes feitos no aglomerado, que os objetivos propostos foram satisfatórios e que apesar de cinco aplicações em apenas uma obtivemos uma melhora de desempenho. Com estes testes conseguiu-se visualizar quais tipos de aplicações teríamos uma performance melhor para aplicar-se em um mundo real.

Para trabalhos futuros pode-se implementar o Openmosix em outros sistemas operacionais de uso mais cotidiano, assim conseguindo criar um *cluster* de outro modelo, o de nós não dedicados, aproveitando apenas o processamento ocioso das estações de uma organização.

Um dos pontos críticos de desempenho nas organizações está relacionado a banco de dados, haja visto o gargalo de trafego de rede e processamento de disco dos servidores, então para trabalhos futuros poderíamos implementar um cluster de dados compartilhados.

Poderíamos implementar também para trabalhos futuros um tratamento por escalabilidade, podendo assim identificar até que ponto conseguiríamos obter melhores desempenhos de acordo que se adicione mais nós no aglomerado.

Com este projeto conseguimos identificar algumas aplicações que o aglomerado obteve melhores desempenhos, mas que mesmo assim poderíamos criar um projeto mais especifico e solucionar algumas dúvidas quanto à viabilidade de algumas aplicações.

REFERÊNCIAS

BACELLAR, Hilário Viana. (2010). **Cluster computação de alto desempenho.** Disponível em: <http://www.ic.unicamp.br/~ducatte/mo401/1s2010/T2/107077-t2.pdf> Acesso em: 08 do Dez. de 2010.

COLVERO. Taís Appel, MAR. Dantas, CUNHA. Daniel Pezzi da (2005). **Ambientes de Clusters e Grids Computacionais: Características, Facilidades e Desafios.** Artigo 1º Congresso Sul Catarinense de Computação. Disponível em: <http://www.dcc.unesc.net/sulcomp/05/Art114SulComp2005.pdf> Acesso em: 09 de Dez. de 2010.

COMPUTERWORD, (2010). **O que é preciso saber para se adotar a TI VERDE.** Disponível em: <http://www.rs.sucesu.org.br/arquivos/TIVerde.pdf> Acesso em: 09 de Dez. de 2010.

COSTA, Carlos. **Construção e configuração de um aglomerado com computadores em laboratórios de informática.** Trabalho de Graduação. Universidade Federal de Santa Maria. (UFSM). Santa Maria, BR – RS, 2009, 48 f.

DADAMOS, Carla et al. **TECNOLOGIA VERDE – TI VERDE.** Santos, SP. 2008.

DANTAS, Mario. **Computação Distribuída de Alto Desempenho: redes, clusters e grids Computacionais.** Rio de Janeiro: Axcel, ebooks do Brasil, 2005.

FAVERA, Eduardo. **Lixo eletrônico e a Sociedade.** Artigo Cientifico. Universidade Federal de Santa Maria (UFSM).2008, 8 f.

GODOY, Rodrigo. (2009). **Saiba tudo sobre – TI VERDE – Parte I.** Disponível em: <http://geekstorm.com.br/tecnologia-empresarial/saiba-tudo-sobre-ti-verde-parte-i/> Acesso em: 10 de Nov. de 2010.

MATOS, Edval P. **Técnicas para construção de clusters de alto desempenho com baixo custo.** Monografia do curso de Ciência da Computação. Universidade de São Francisco (USF). Itatiba, BR – SP, 2006, 39 f.

PEREIRA, Felipe M. **Estudo de Performance em um Cluster OpenMosix.** Monografia - Faculdade Lourenço Filho – Fortoleza, BR – CE, 2010, 51 f.

PEREIRA, Glauber Ruan Barbosa. (2009).**Praticas da Ti verde que contribuem para o desenvolvimento sustentável: Um estudo de caso das indústrias RN**. 126 f. Dissertação de Mestrado, Universidade Federal do Rio Grande do Norte, Natal, BR - RN <http://bdtd.bczm.ufrn.br/tedesimplificado//tde_arquivos/24/TDE-2009-11-26T011517Z-2252/Publico/GlauberRBP.pdf> Acesso em: 02 de nov. de 2010.

PITANGA, Marcos. (2003). **Computação em Cluster.**
Disponível em:
<http://www.clubedohardware.com.br/artigos/Computacao-em-cluster/153>
Acesso em: 10 de Dez. de 2010.

PITANGA, Marcos. (2008). **Construindo supercomputadores com Linux.** 3 ed. Rio de Janeiro: Brasport. 375 p. Bibliografia: p. 361 – 368 ISBN 978-85-7452-372-9

PITANGA, Marcos. (2002). **Supercomputadores Caseiros: Construindo Clusters com o Linux - Parte 2.** Tutorial. Site Guia do Hardware.
Disponível em:
<http://www.clubedohardware.com.br/artigos/Supercomputadores-Caseiros-Construindo-Clusters-com-o-Linux-Parte-2/162/1>
Acesso em: 21 de Out. 2012

RECHENBURG, Matt. (2005). **The openMosix stress test.**
Disponivel em:
<http://www.openmosixview.com/omtest>
Acesso em: 01 novembro de 2012

SILVA, Tiago. (2006). **Escalonamento de Processos em Sistemas Distribuídos: Uma Visão Geral.**
Disponível em:
<http://www.inf.ufrgs.br/~asc/sodr/pdf2006-02/SODRTiagoSilva.pdf>
Acesso em: 10 de Jun. de 2011

SILVA, José Vanderlei. **Cluster – Possibilidades de eficiência e segurança.** Artigo Cientifico. Universidade Estadual de Maringá (UEM). Maringá, BR – PR, 2006, 7 f.

SILVA, José Mauricio Oliveira. **Estudo e construção de um ambiente de Alto Desempenho utilizando Cluster Computacional.** Trabalho Conclusão de Curso.Universidade Federal do Piauí (UFPI). Teresina, BR – PI, 2009, 106 f.

SOARES, Edileuza. (2005). **Reduza custos com a TI Verde.**
Disponível em:
<http://wnews.uol.com.br/site/noticias/materia_especial.php?id_secao=17&id_conteudo=579>
Acesso em: 03 de Nov. de 2010.

SOUZA, Orlando. (2007). **Processos.**
Disponível em:
<http://www.dei.isep.ipp.pt/~orlando/so2/processos.htm>
Acesso em: 11 de out. de 2012

TAKAHASHI, Arthur G. et al. **TI Verde conceitos e práticas.** Artigo Cientifico.
Guia do Hardware. 2009, 8 f.
Disponível em:
<http://www.hardware.com.br/arquivos/TI-Verde.pdf>
Acesso em: 27 de Out. de 2012

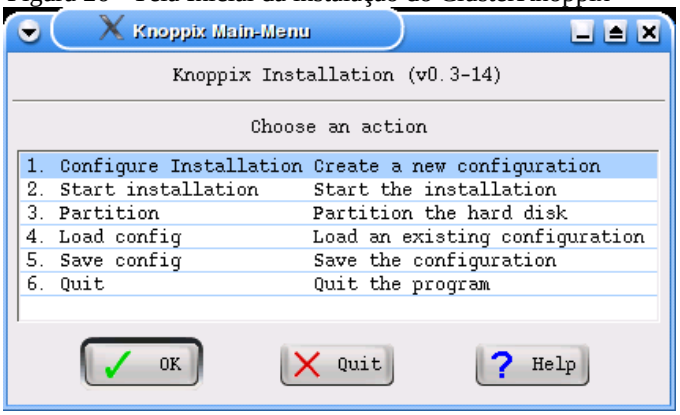
YANO, Luis Gustavo Abe. **Avaliação e comparação de desempenho utilizando tecnologia CUDA.** Monografia de Projeto Final. Universidade Estadual Paulista "Julio de Mesquita Filho". São José do Rio Preto, BR-SP, 2010, 83f.
Disponível em:
<http://rock.dcce.ibilce.unesp.br/spd/pubs/mono_Yano.pdf>
Acesso em: 10 de Nov. de 2012

APÊNDICES

APÊNDICE A: Instalação do Cluster Knoppix no HD

Se desejarmos instalar o ClusterKnoppix no HD sem a necessidade do CD os passos são muito simples. Abra o terminal Shell e dê o comando *sudo knoppix-installer* com permissões de administrador (root), a seguir o sistema precisa de instruções referente às partições de seu HD (para montagem desse cluster utilizaremos HD próprios exclusivos para o ClusterKnoppix). Preferencialmente criamos a partição *Swap* do tamanho de 520mb e criamos a partição com o resto do HD com o sistema de arquivos *reiservers*, (escolhemos reiservers por ser mais estável e compatível do que as demais versões de sistema de arquivos).

Figura 26 - Tela Inicial da instalação do ClusterKnoppix



Fonte: ISO de instalação ClusterKnoppix

Assim que o ClusterKnoppix estiver instalado no HD o próximo passo é configurar a rede. De padrão o sistema escolhe padrão DHCP, mas se você não tiver nenhum servidor instalado para configurar vá em *iniciar* > *Knoppix* > *network/internet* > *network card configuration* e configure seu IP fixo. Faça os mesmos passos nos demais nós do cluster.

APÊNDICE B: Artigo Científico

TI VERDE: ESTUDO E IMPLEMENTAÇÃO DE UM AMBIENTE DE ALTO DESEMPENHO UTILIZANDO CLUSTER COMPUTACIONAL, COM COMPUTADORES RECICLADOS

Leandro Bitencourt Fernandes[1], Paulo João Martins[2]

[1]Acadêmico de Ciência da Computação – Universidade do Extremo Sul Catarinense – (UNESC**),** Av. Universitária, 1105 - Bairro Universitário, C.P. 3167, CEP: 88806-000 Criciúma / Santa Catarina / Brasil

[2]Orientador Msc de Ciência da Computação – Universidade do Extremo Sul Catarinense – (UNESC**),** Criciúma / Santa Catarina / Brasil

leancri@gmail.com, pjm@unesc.net

Abstract: *This research was developed through the need to reclaim disused computers in organizations. For this, there is a methodology in Green IT important for treating the concepts of proper disposal of equipment and / or reuse them. The solution for reuse of computers in distributed computing found where one can implement a reusing computational processing cluster of computers discarded. Looking for a platform for general purpose computing cluster, the openMosix is a tool that makes possible this project. From the chosen platform, you run the application Omtest, aiming to test the functioning of the cluster. The Omtest is composed of several software applications that have been treated separately. The results were analyzed and compared according to number of nodes added to the cluster.*

Resumo: *Esta pesquisa foi desenvolvida através da necessidade de reaproveitar os computadores em desuso nas organizações. Para isso, observa-se na TI Verde uma metodologia importante para tratar os conceitos de descarte correto dos equipamentos e/ou a reutilização dos mesmos. A solução para reutilização dos computadores foi encontrada na computação distribuída, onde se pode implementar um cluster computacional reaproveitando o processamento dos computadores descartados. Procurando uma plataforma de cluster computacional de uso geral, encontra-se no Openmosix uma ferramenta que viabiliza este projeto. A partir da plataforma escolhida, executa-se a aplicação Omtest, objetivando testar o funcionamento do cluster. O Omtest é um software composto por várias aplicações que foram tratadas separadamente. Os resultados obtidos foram analisados e comparados conforme a quantidade de nós adicionados ao aglomerado.*

1 TI VERDE

Com o grande avanço tecnológico, tanto de hardware e software, as empresas cada vez mais precisam estar atualizadas com o seu poder computacional, e consequentemente descartar seu equipamento defasado para a utilização dessas novas tecnologias.

Segundo Silva (2009), as áreas que mais precisam estar atualizadas são as que exigem processamento contínuo e áreas que necessitam de precisão nos seus resultados.

O *cluster* computacional surgiu nas últimas décadas como alternativas ao alto custo dos supercomputadores. É um tipo de sistema de processamento paralelo que consiste na coleção de computadores independentes, interconectados através de uma rede, trabalhando cooperativamente como um único e integrado recurso computacional.

A Tecnologia da Informação (TI), que se define como um conjunto de todas as soluções e atividades providas por recurso de computação, ganha importância nas organizações quando elas mesmas percebem que a informação que detém faz parte de seu patrimônio.

As organizações sofrem constante pressão para reduzir custos em investimentos e despesas de TI. Esses fatores devem ser cuidadosamente analisados na perspectiva do valor da empresa, pensando na redução de custos ou da não continuidade de algum projeto pode prejudicar o desempenho da área comercial ou um serviço de atendimento ao cliente e também uma ineficiência operacional, afetando diretamente na linha de receita e a imagem da empresa perante o mercado (BOTTO, 2010).

A TI Verde tem ampliado os campos de pesquisas na área de tecnologia nos últimos tempos, tem sido muito discutido e inserido nas campanhas publicitárias e em mídias visuais. Nada mais é do que um conjunto de práticas que visa tornar mais sustentável e menos prejudicial o nosso uso da computação. Geralmente, alguns administradores de TI dedicam seu tempo de estudo na economia de recursos computacionais e principalmente na economia de energia.

Segundo Takahashi et al (2009) as práticas de TI Verde pode ser dividida em três subníveis:

d) TI Verde de incrementação tática: onde não muda a infraestrutura de TI da organização e apenas criam-se políticas de contensão de gastos elétricos excessivos. Como exemplo o monitoramento de energia disponível nos equipamentos. O desligamento dos mesmos no momento do não uso, otimização da temperatura no setor e salas. Tais medidas são simples e não geram custos adicionais as organizações;

e) TI Verde estratégico: necessita-se de uma auditoria sobre a infraestrutura de TI e seu uso relacionado ao meio ambiente, desenvolvendo meios mais viáveis e ecologicamente corretos de produção de bens ou serviços. Como exemplo a implantação de uma nova infraestrutura elétrica visando maior eficiência de consumo elétrico;

f) Deep IT: Muito mais amplo que os demais níveis, criam-se então um projeto de implantação estrutural de um parque tecnológico otimizando o desempenho com o mínimo de gastos elétricos. Como exemplo a implantação de sistemas de refrigeração o que dispõem de equipamentos no local gerando maiores gastos que os demais níveis.

O meio ambiente já é uma preocupação de todos, de todas as áreas, setores, organizações do mundo, e a tecnologia já vem se adequando a essa nova realidade. Os profissionais da área de TI, já estão se adequando uma nova política “verde”, onde o ser sustentável é fator importante no desenvolvimento das organizações, assim gerando maior credibilidade e reconhecimento de todos.

2. COMPUTAÇÃO PARALELA E DISTRIBUIDA

Nos últimos tempos o poder de processamento dos computadores tem aumentado em torno de 20% ao ano, enquanto que os microprocessadores têm sido em torno de 40% ao ano (PITANGA, 2008).

A evolução está acima da média devido ao fato de os microprocessadores estarem evoluindo tanto em arquitetura como também em tecnologia. Esse avanço está servindo para diminuir o esforço necessário para processar um ciclo de instrução.

Outro grande passo do avanço tecnológico foi a criação das redes locais (LAN), onde grande quantidade de dados podem ser compartilhados entre os computadores conectados a rede.
Grande parte dos sistemas distribuídos tem por objetivo prover maior desempenho computacional.

Os sistemas distribuídos, sob o aspecto de arquitetura de máquinas para a execução de aplicativos, devem ser vistos como configurações com grande poder de escala pela agregação dos computadores existentes nas redes convencionais (DANTAS, 2005).

Umas das grandes dificuldades encontradas nos sistemas distribuídos estão relacionadas ao software, tudo isto devido à elevada complexidade no seu desenvolvimento onde exige um vasto conhecimento do desenvolvedor. Nos ambientes distribuídos a dificuldade encontrada está na heterogeneidade de um conjunto de máquinas, onde cada computador possui uma arquitetura diferente de hardware executando seu próprio sistema operacional.

3. CLUSTER

Quando se deseja um maior desempenho para o processamento de um conjunto de operações computacionais ou até mesmo uma única operação mais complexa, as opções seriam trabalhar mais intensamente, trabalhar com mais eficiência ou pedir ajuda (DANTAS, 2005).

Na utilização de processadores mais potentes para obter a execução mais rápida da operação, já tem sido limitada por uma série de fatores prevista por Gordon Moore em 1965, onde os limites da própria física façam que os processadores cheguem a seu ápice de processamento. Limitações como a miniaturização do processador chegando a um ponto onde não mais área existente para agregar elementos eletrônicos de processamento.

Outra forma de se obter melhores desempenhos é o desenvolvimento de algoritmos eficazes. Porém, a utilização de algoritmos mais eficientes não garante melhores resultados para inúmeras aplicações. A conclusão é que a grande quantidade de dados a serem processados pela operação seja o problema para obter processadores mais potentes.

A outra opção citada por Dantas (2005) pedir ajuda, resumindo para uso da computação é a utilização dos sistemas paralelos e distribuídos. A criação de um ambiente agregado de computadores processando de forma paralela os ambientes de cluster permitindo melhoria no desempenho das aplicações.

O *Cluster* (que do inglês significa agrupamento) em termos de arquiteturas de hardware, é uma unificação de vários computadores processando de forma dedicada ou não, para execução de aplicações especificas (DANTAS, 2005).

A estrutura do *cluster* pode ser classificada como dedicadas e não dedicadas. Uma arquitetura dedicada é quando criamos a exemplo *pilhas de PCs (Beowulf)* onde todos os nós são de uso exclusivo para o cluster. Nos cluster não dedicados, cada um dos nós são estações de trabalho na organização e em momentos de ociosidade e são utilizados para adicionar processamento.

Figura 1 – Cluster dedicado Beowulf

Fonte: http://www.cse.mtu.edu/cseri.html

A escalabilidade é um fator diferencial do *cluster*, onde os recursos crescem a cada nó acrescentado a ele (DANTAS, 2005).

Dentre as inúmeras vantagens de se utilizar um cluster, Pitanga (2008) define as principais:

g) alto desempenho: Possibilidade de resolver problemas muito complexos exigindo alto poder de processamento e que por meio do processamento paralelo diminua o tempo de resolução do problema;
h) escalabilidade: Possibilidade de que novos componentes sejam adicionados à medida que cresce a carga de trabalho;
i) tolerância a falhas: O aumento da confiabilidade do sistema com um todo, caso alguma parte falhe;

Cluster de processamento de alto desempenho (HPC – high performance cluster), essa categoria de cluster tem como principal objetivo a alta performance. Esses clusters se tornam alternativas viáveis para universidades e empresas de qualquer porte, pois podem obter alto desempenho em aplicações, a um custo razoavelmente baixo se comparado a aquisição de um supercomputador.

O cluster escolhido para o desenvolvimento deste projeto foi o Openmosix.

O Openmosix é uma extensão do núcleo do kernel do sistema operacional Linux, pois faz com que o cluster computacional se comporte como se fosse um único supercomputador.

Ele vem implementado a migração preemptiva de processos que quer dizer que o processamento computacional onde o kernel tem controle do tempo que será usado para cada processo, e tem poder de tomar de volta este tempo e entregar a outro processo segundo seu esquema de prioridades.

No Openmosix não existe a necessidade de configuração do nó mestre com o nó escravo com é feito no cluster Beowulf. Cada nó do aglomerado é nó mestre para os processos que são criados localmente no seu sistema operacional é também um servidor de processos remotos onde migra para o demais nós do cluster. Isto confirma que podemos estar adicionando ou removendo nós do cluster sem nenhuma modificação de configuração ou parada no sistema.

4. PROJETO DO CLUSTER COMPUTACIONAL

A necessidade de darmos alguma funcionalidade para os computadores descartados pela organização se teve a ideia de construir um aglomerado de computadores capazes de contribuir no alto processamento de alguma aplicação ou serviço. As máquinas

utilizadas no projeto já estavam descartadas pela organização e estavam em depósitos como máquinas reservas.

O cluster será constituído por:

f) 01 switch 16 portas 10/100/1000Mbps TP-LINK para interligar 4 computadores;

g) nó 01 Principal: AMD Duron 1300mhz – 512mb SDR 133mhz – Placa de Vídeo Onboard 32mb – HD 20gb;

h) nó 02: AMD Duron 950Mhz – 512mb SDR 133Mhz – Placa de Vídeo Onboard 32mb – HD 20gb;

i) nó 03: AMD Duron 700Mhz – 256mb SDR 133Mhz – Placa de Vídeo onboard 8mb – HD 20 Gb IDE;

j) nó 04: AMD Duron 800Mhz - 256 SDR 133Mhz - Placa de Vídeo onboard 16mb – HD 40Gb IDE.

No projeto do *cluster* foram inseridos no aglomerado máquinas mais homogêneas possíveis para garantir a confiabilidade dos dados.

Nós com configurações iguais não são obrigatórios, mas contribuem para diminuir problemas decorrentes de compilação de um código para arquiteturas diferentes, ou por terem também uma carga de trabalho desbalanceada, onde os nós mais rápidos irão terminar primeiro que os mais lentos e ficar esperando. Assim, a homogeneidade não é uma condição necessária para um cluster, mas é um fator que irá reduzir bastante a quantidade de problemas (PITANGA, 2002).

Avaliando a necessidade de um algoritmo de alto processamento, foi pesquisada aplicação chamada Omtest disponível no link: http://www.openmosixview.com/omtest/#down

Segundo Rechenburg (2005), o teste de esforço é feito para testar o OpenMosix mais o kernel cluster. O Omtest irá executar diversas aplicações paralelamente, para testar o kernel, verificar a estabilidade e outras características do OpenMosix (por exemplo,a migraçãode processos, mfs). Durante este teste, o cluster será carregado principalmente por isso, deve-se parar de executar outros aplicativos antes de iniciá-lo.

O Omtest é constituído por várias aplicações, segue descrição:

f) Distkeygen: Esta aplicação é utilizada para gerar pares de chaves RSA 4000 com 1024 bits de comprimento da chave. Ele é distribuído em vários processos como processadores em seu cluster Openmosix via Fork;

g) Portfolio: Programa perl que simula uma carteira de diversas ações para um determinado período de tempo. Este método é a base para o livro "The intelligent asset allocator", que do inglês siginifica “Ativador de ativos inteligentes” de William Bernstein;

h) Eatmen: Simplesmente calcula o seno+Raiz quadrada de um valor 1.000.000 vezes e exibe na saída a contagem de loop para um arquivo. Este teste é iniciado em diversos períodos de uma só vez pelos vários processadores encontrados no cluster Openmosix automaticamente ;

i) Forkit: Teste de forkit é semelhante ao teste de eatmem, mas usa fork para criar múltiplos processos (3 * [processors_in_your_openMosix_cluster]);

j) Timewaster: Isso irá criar um arquivo de 10 MB e copiá-lo para todos nós e para trás. É para verificar as capacidades do MFS.

Durante a execução a aplicação informa o tempo de início e o tempo final, sendo primordial para avaliar a evolução dos tempos conforme são adicionados mais nós ao aglomerado. Ao final, ele gera um relatório detalhado sobre cada componente que foi testado.

Com o cluster já configurado, os testes realizados foram os seguintes:

e) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) e documentar o desempenho;
f) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais o nó secundário (NÓ2) e documentar o a evolução de desempenho;
g) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais os nós auxiliares (NÓ2 e NÓ3) e documentar a evolução de desempenho;
h) rodar a aplicação Omtest 15 vezes no nó principal (NÓ1) mais os nós auxiliares (NÓ2, NÓ3 e NÓ4) e documentar a evolução do desempenho;

4. RESULTADOS OBTIDOS

Inicialmente executamos todos os 15 testes com a aplicação Omtest em cada nó e verificamos o desempenho de cada um, para então definirmos o nó principal e os nós secundários.

Apartir da definição da sequência dos nós, começamos a rodar os testes no aglomerado.

Os resultados foram obtidos seguindo os resultados fornecidos pela própria aplicação, onde na amostragem informava o tempo inicial e o tempo final. O cálculo a ser feito era então diminuir o tempo final pelo tempo inicial.

Executou-se Omtest dividindo os resultados conforme as aplicações executem paralelamente, para verificarmos o ganho de desempenho conforme é adicionado nós ao aglomerado.

Os tempos foram calculados conforme resultados que a aplicação gerou. Documentamos o tempo dos cálculos e ao final em azul temos uma média dos resultados obtidos.

A primeira aplicação a rodar é o Distkeygen, e após rodar a aplicação 15 vezes os resultados foram os seguintes.

Tabela 1 – Resultados do Distkeygen

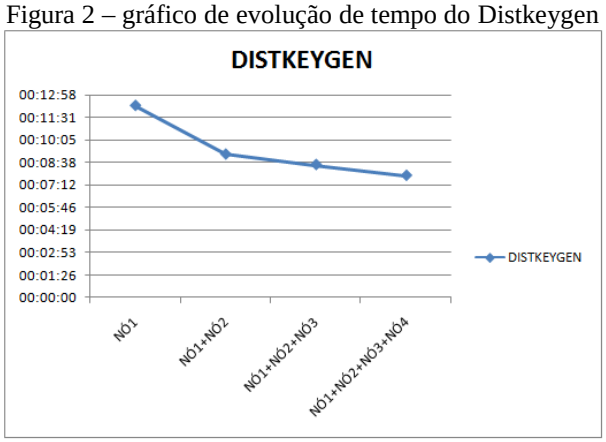
DISTKEYGEN			
NÓ1	NÓ1+NÓ2	NÓ1+NÓ2+NÓ3	NÓ1+NÓ2+NÓ3+NÓ4
00:12:19	00:09:14	00:08:33	00:07:30
00:12:17	00:09:15	00:08:22	00:07:48
00:12:18	00:09:14	00:08:22	00:07:57
00:12:16	00:09:04	00:08:37	00:08:00
00:12:12	00:09:03	00:08:34	00:07:45
00:12:18	00:09:10	00:08:26	00:07:50
00:12:19	00:09:06	00:08:26	00:07:35
00:12:19	00:09:15	00:08:24	00:07:44
00:12:23	00:09:09	00:08:23	00:07:59
00:12:23	00:09:07	00:08:36	00:07:43
00:12:14	00:09:00	00:08:30	00:07:56
00:12:09	00:09:13	00:08:32	00:07:56
00:12:16	00:09:09	00:08:30	00:07:46
00:12:19	00:09:19	00:08:36	00:07:41
00:12:12	00:09:07	00:08:35	00:07:49
00:12:17	**00:09:10**	**00:08:30**	**00:07:48**

Fonte: do autor.

Nota-se uma variação muito pequena em todos os resultados obtidos do Distkeygen, garantindo uma confiabilidade excelente.

Neste gráfico a seguir temos a evolução dos tempos.

Figura 2 – gráfico de evolução de tempo do Distkeygen



Fonte: do autor.

Conforme descrito anteriormente o teste Distkeygen gera pares de chaves RSA 4000 de 1024 bits de comprimento, concluindo com os resultados obtidos um melhor desempenho conforme foi adicionado nós ao nosso aglomerado. Comprovou-se que aplicações de criptografia o cluster seja bem viavel para implementação. Os tempos foram diminuindo conforme aumentou-se o poder computacional do *cluster*.

A segunda aplicação que o Omtest executa é o *Portfolio.* A seguir temos a tabela de resultados obtidos em nosso aglomerado.

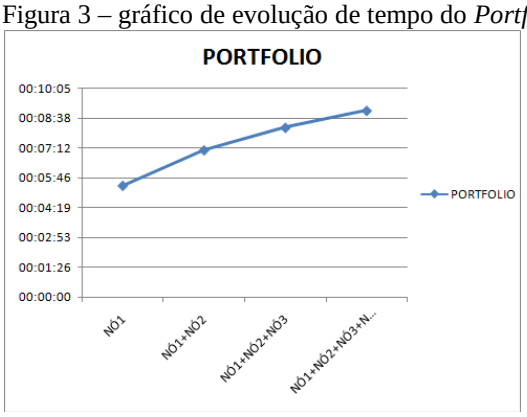
Tabela 2 – resultados do *Portfolio*

PORTFOLIO			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:05:24	00:07:15	00:08:05	00:09:05
00:05:23	00:07:19	00:08:16	00:08:50
00:05:22	00:07:23	00:08:16	00:08:50
00:05:23	00:07:01	00:08:11	00:08:51
00:05:23	00:07:10	00:08:11	00:08:49
00:05:22	00:07:00	00:08:15	00:09:01
00:05:23	00:06:57	00:08:06	00:08:56
00:05:24	00:06:56	00:08:14	00:08:52
00:05:22	00:07:03	00:08:16	00:08:59
00:05:23	00:07:04	00:08:07	00:09:12
00:05:23	00:07:05	00:08:09	00:09:02
00:05:21	00:07:07	00:08:05	00:09:02
00:05:23	00:06:57	00:08:11	00:09:13
00:05:23	00:07:04	00:08:12	00:09:04
00:05:22	00:07:06	00:08:12	00:09:20
00:05:23	**00:07:06**	**00:08:11**	**00:09:00**

Fonte: do autor.

Neste gráfico a seguir temos a evolução dos tempos da aplicação do Portfolio, conforme avaliado verificou que o desempenho não foi satisfatório, mas comprovou-se a eficácia da aplicação, pois conforme análise da variação dos resultados o Openmosix se mostrou muito estável.

Figura 3 – gráfico de evolução de tempo do *Portfolio*



Fonte: do autor.

A terceira aplicação que o Omtest executa é o Eatmen. O *eatmen* é um aplicativo que efetua um calculo de seno mais raiz quadrada de um numero 1000000 vezes este valor distribuindo o calculo por todos os nós automaticamente. Logicamente então se ele repeti o mesmo calculo por todos os nós não teríamos melhoras de tempo com o nosso aglomerado.

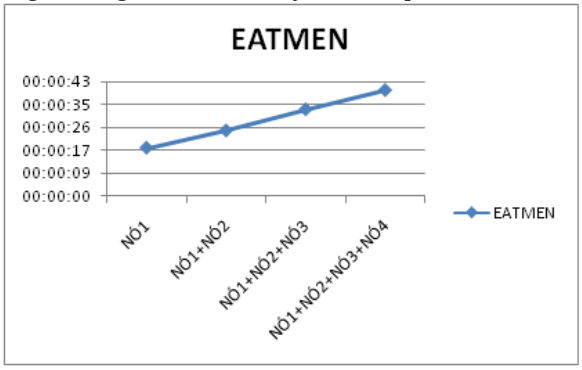
Tabela 3 – Resultados do *Eatmen*

EATMEN			
NÓ1	NÓ1+NÓ2	NÓ1+NÓ2+NÓ3	NÓ1+NÓ2+NÓ3+NÓ4
00:00:18	00:00:24	00:00:33	00:00:40
00:00:18	00:00:25	00:00:33	00:00:39
00:00:19	00:00:25	00:00:34	00:00:39
00:00:18	00:00:24	00:00:33	00:00:39
00:00:18	00:00:25	00:00:32	00:00:39
00:00:18	00:00:25	00:00:32	00:00:39
00:00:19	00:00:25	00:00:33	00:00:39
00:00:18	00:00:25	00:00:33	00:00:41
00:00:18	00:00:25	00:00:33	00:00:38
00:00:18	00:00:25	00:00:32	00:00:40
00:00:17	00:00:24	00:00:32	00:00:42
00:00:18	00:00:24	00:00:32	00:00:41
00:00:17	00:00:24	00:00:32	00:00:41
00:00:19	00:00:24	00:00:33	00:00:40
00:00:18	00:00:25	00:00:32	00:00:41
00:00:18	**00:00:25**	**00:00:33**	**00:00:40**

Fonte: do autor.

Conforme figura 4 verificasse no gráfico que os tempos não tiveram melhores desempenho, por se tratar de uma aplicação de teste de esforço e não uma aplicação de simulação de desempenho de cluster computacional.

Figura 4 - gráfico de evolução de tempo do Eatmen



Fonte: do autor.

A quarta aplicação a ser testada pelo Omtest é o Forkit. Uma aplicação semelhante a anterior Eatmen, mais que executa métodos Fork para criar múltiplos processos.

O Fork é um método de replicação de processos aberto da na memória RAM. O método cria um novo processo (filho) a partir do processo original (pai). A diferença entre os dois processos que são criados diferentes identificação para cada processo. Quando o Fork é executado retornasse 0 para processo filho e o numero do processo filho para o pai, com isso conseguisse identificar quem é o processo "filho" e quem é o processo original "pai" (SOUZA, 2007).

O método Fork é muito encontrado em banco de dados, quando o processo Fork é executado ele gera um processo filho responsável pelo processamento daquela conexão.

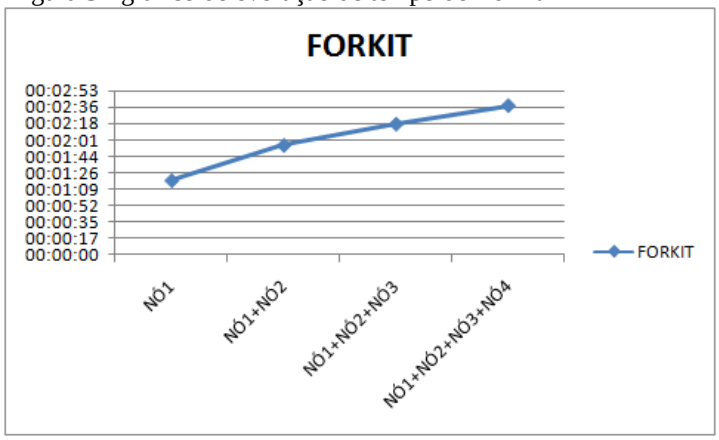
Tabela 4 – Resultados do *Forkit*

FORKIT			
NÓ1	**NÓ1+NÓ2**	**NÓ1+NÓ2+NÓ3**	**NÓ1+NÓ2+NÓ3+NÓ4**
00:01:18	00:01:58	00:02:18	00:02:48
00:01:18	00:02:19	00:02:15	00:02:50
00:01:18	00:02:23	00:02:25	00:02:34
00:01:18	00:01:48	00:02:30	00:02:39
00:01:18	00:01:47	00:02:14	00:02:31
00:01:18	00:01:43	00:02:12	00:02:32
00:01:18	00:01:46	00:02:10	00:02:35
00:01:18	00:01:52	00:02:19	00:02:45
00:01:18	00:01:54	00:02:14	00:02:29
00:01:18	00:01:44	00:02:24	00:02:55
00:01:18	00:01:52	00:02:09	00:02:55
00:01:18	00:01:58	00:02:08	00:02:30
00:01:18	00:01:54	00:02:08	00:02:19
00:01:17	00:02:00	00:02:31	00:02:25
00:01:18	00:01:48	00:02:25	00:02:27
00:01:18	**00:01:55**	**00:02:17**	**00:02:37**

Fonte: do autor.

O gráfico da figura 5 demonstra a evolução dos tempos de acordo que se adicionam mais computadores no aglomerado.

Figura 5 - gráfico de evolução de tempo do *Forkit*



Fonte: do autor.

A aplicação *Forkit* não se mostrou eficiente para estar rodando em um ambiente cluster, mais apesar de não obter melhor desempenho vimos que o balanceamento de carga do

método fork se mostrou estável e coerente. Isto devido ao Openmosix possuir ferramentas de analise que compravam a eficiência da migração de processos.

A ultima aplicação a ser testada pelo Omtest é o *Timerwaster*. Uma ferramenta de teste de migração de dados entre os nós do aglomerado.

O *Timewaster* cria um arquivo de 10 MB e copia para todos os nós do cluster e os mesmos nós repete novos arquivos para o nó principal. Este teste foi criado para verificar o desempenho do MFS.

Teoricamente quanto mais nós no cluster, mais tempo levaríamos para rodar o teste.

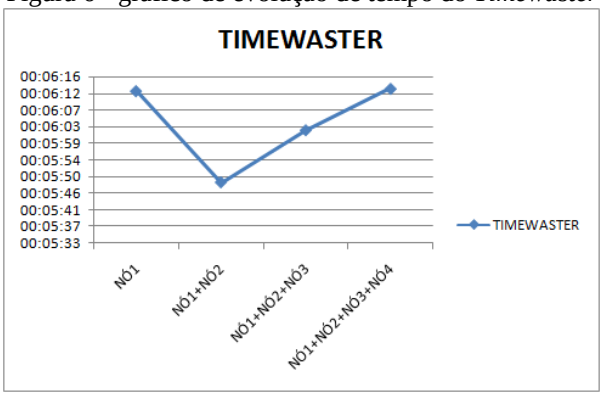
Tabela 5 - Resultados do *Timewaster*

TIMEWASTER			
NÓ1	NÓ1+NÓ2	NÓ1+NÓ2+NÓ3	NÓ1+NÓ2+NÓ3+NÓ4
00:06:13	00:05:44	00:06:01	00:06:16
00:06:12	00:05:47	00:06:02	00:06:14
00:06:12	00:05:47	00:06:01	00:06:14
00:06:12	00:05:48	00:05:58	00:06:09
00:06:11	00:05:47	00:06:02	00:06:08
00:06:12	00:05:51	00:06:02	00:06:17
00:06:12	00:05:54	00:06:00	00:06:12
00:06:12	00:05:49	00:06:02	00:06:10
00:06:12	00:05:46	00:06:02	00:06:16
00:06:12	00:05:50	00:05:59	00:06:12
00:06:12	00:05:48	00:06:13	00:06:16
00:06:12	00:05:50	00:05:58	00:06:19
00:06:12	00:05:49	00:06:06	00:06:16
00:06:13	00:05:45	00:06:01	00:06:08
00:06:13	00:05:50	00:06:02	00:06:04
00:06:12	**00:05:48**	**00:06:02**	**00:06:13**

Fonte: do autor.

No gráfico da Figura 6, temos a evolução dos tempos no *Timewaster*.
Verificasse que os tempos foram aumentando apartir da adição de nós ao aglomerado.

Figura 6 - gráfico de evolução de tempo do *Timewaster*



Fonte: do autor.

Este artigo cientifico, apresentou uma solução de TI Verde muito viável para qualquer tipo e tamanho de organização, podendo estar reutilizando as maquinas antigas já descartadas.

Implementamos uma solução de cluster para as maquinas descartadas, de forma a se obter um melhor desempenho em alguma aplicação especifica e que possamos estar aplicando em um mundo real. Tudo isso através de uma ferramenta muito intuitiva que é o Openmosix. Conseguimos então aplicar alguns conceitos de computação paralela e distribuída que são o balanceamento de carga e a migração preemptiva de processos.

Os resultados obtidos conclui-se que algumas aplicações não são viáveis para um ambiente de cluster, pois não conseguiríamos obter desempenho superior.

Concluímos com todos os teste feito no aglomerado, que os objetivos propostos foram satisfatórios e que apesar de cinco aplicações apenas uma obtivemos uma melhora de desempenho, com este teste conseguimos visualizar quais tipos de aplicações teríamos uma performance melhor para podermos aplicar em um mundo real.

5. REFERENCIAS

PITANGA, Marcos. (2003). **Computação em Cluster.**
Disponível em:
<http://www.clubedohardware.com.br/artigos/Computacao-em-cluster/153>
Acesso em: 10 de Dez. de 2010.

PITANGA, Marcos. (2008). **Construindo supercomputadores com Linux.** 3 ed. Rio de Janeiro: Brasport. 375 p. Bibliografia: p. 361 – 368 ISBN 978-85-7452-372-9

PITANGA, Marcos. (2002). **Supercomputadores Caseiros: Construindo Clusters com o Linux - Parte 2.** Tutorial. Site Guia do Hardware.
Disponível em:
<http://www.clubedohardware.com.br/artigos/Supercomputadores-Caseiros-Construindo-Clusters-com-o-Linux-Parte-2/162/1>
Acesso em: 21 de Out. 2012

RECHENBURG, Matt. (2005). **The openMosix stress test.**
Disponivel em:
<http://www.openmosixview.com/omtest>
Acesso em: 01 novembro de 2012

DANTAS, Mario. **Computação Distribuída de Alto Desempenho: redes, clusters e grids Computacionais.** Rio de Janeiro: Axcel, ebooks do Brasil, 2005.

SOUZA, Orlando. (2007). **Processos.**
Disponível em:
<http://www.dei.isep.ipp.pt/~orlando/so2/processos.htm>
Acesso em: 11 de out. de 2012