Luiz Henrique Bincoletto Tomazella

Um Estudo Experimental de Utilização de Cluster na Aplicação WRF de Previsão de Tempo

Florianópolis, Santa Catarina

10 de Maio de 2009

Luiz Henrique Bincoletto Tomazella

Um Estudo Experimental de Utilização de Cluster na Aplicação WRF de Previsão de Tempo

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Ciências da Computação

Orientador:
Mario A. R. Dantas

UNIVERSIDADE FEDERAL DE SANTA CATARINA

Florianópolis, Santa Catarina

10 de Maio de 2009

Monografia de graduação sob o título ”Um Estudo Experimental de Utilização de Cluster na Aplicação WRF de Previsão de Tempo”, defendida por Luiz Henrique Bincoletto Tomazella, aprovada em 10 de junho de 2009, em Florianópolis, Santa Catarina, pela banca examinadora constituida pelos doutores:

Prof. Dr. Mario A. R. Dantas
Orientador

José Mazzucco Jr.
Universidade Federal de Santa Catarina

Luís Fernando Friedrich
Universidade Federal de Santa Catarina

Resumo

Os ambientes de supercomputação, também chamados de supercomputadores ou computadores de alto desempenho, estão mais acessíveis e cada vez mais presentes como ferramentas para resolução de problemas, da área científica e de engenharia, que necessitam de resutados em um intervalo de tempo aceitável. Normalmente esses problemas são classificados como *grand challanges* (grandes desafios, em português) e são fortemente embasados em modelagem numérica e simulações. Dessa forma, os ambientes de supercomputação podem auxiliar a resolução desses problemas de grande impacto econômico e científico, através das técnicas e recursos computacionais de alto desempenho.

Dada essa necessidade ambientes altamente especializados são necessários para o processamento dos dados oriundos das aplicação. Bem como a grande variedade de problemas a serem resolvidos, os ambientes computacionais de alto desempenho propiciam inúmeras arquiteturas que em conjunto com o problema constitem uma ambiente computacional paralelo, como exemplos de arquiteturas, pode ser citado as MPP (*Massive Parallel Processing*) e os *clusters*.

Os ambientes de *cluster* atualmente estão altamente difundidos como ambientes de computação de alto desempenho. Essa popularidade se deve falta de complexidade estrutural para a sua composição, além da opção de serem construídos com equipamentos *off-the-shelf*. Dada essa vantagem e a características desse ambiente implementar uma arquitetura de memória distribuida, muitas aplicação que implementam esse paradigima fazem uso do ambiente, nesse estudo especificamente, a aplicação WRF(*Weather Research Forecasting*) faz o uso desse poder computacional.

Com o assunto contextualizado, esse trabalho apresenta um estudo empírico realizado no ambiente da EPAGRI S.A. (Empresa de Pesquisa Agropecuária e Extensão Rural de Santa Catarina). Através da abordagem de diminuir o custo de processamento relativo aos fatores resultantes da comunicação entre os processos parelelos, simplesmente, mantendo um núcleo ocioso por nó em relação ao processamento da aplicação e dedicado ao processamento da comunicação.

Utilizando todo conheciemnto teórico adquirido duranto esse trabalho e ulizando a aboradagem proposta foi possível prover uma melhora no desempenho da aplicaçãoe WRF, que em algumas configurações alcançou um ganho de mais de 25% sem o acréscimo de *hardware*.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Os ambientes de supercomputação, também chamados de supercomputadores ou computadores de alto desempenho, estão mais acessíveis e cada vez mais presentes como ferramentas para resolução de problemas, da área científica e de engenharia, que necessitam de resutados em um intervalo de tempo aceitável(KUMAR et al., 1994). Normalmente esses problemas são classificados como *grand challanges* (grandes desafios, em português) e são fortemente embasados em modelagem numérica e simulações. De acordo com (KUMAR et al., 1994), os ambientes de supercomputação podem auxiliar a resolução desses problemas de grande impacto econômico e científico, através das técnicas e recursos computacionais de alto desempenho.

Dada essa necessidade ambientes altamente especializados são necessários para o processamento dos dados oriundos das aplicação. Bem como a grande variedade de problemas a serem resolvidos, os ambientes computacionais de alto desempenho propiciam inúmeras arquiteturas que em conjunto com o problema constitem uma ambiente computacional paralelo, como exemplos de arquiteturas, pode ser citado as MPP (*Massive Parallel Processing*) e os *clusters*.

Os ambientes de *cluster* atualmente estão altamente difundidos como ambientes de computação de alto desempenho. Essa popularidade se deve falta de complexidade estrutural para a sua composição, além da opção de serem construídos com equipamentos *off-the-shelf*. Dada essa vantagem e a características desse ambiente implementar uma arquitetura de memória distribuida, muitas aplicação que implementam esse paradigima fazem uso do ambiente, nesse estudo especificamente, a aplicação WRF(*Weather Research Forecasting*) faz o uso desse poder computacional.

Essa monografia tem seu foco no entendimento dos aspéctos que constituem a base do conhecimento no âmbito da computação paralela, distribuída e de alto desempenho, bem como no estudo do impacto da comunicação TCP/IP utilizada para troca de mensagens entre os processos, nos avanços da tecnologia VIA e na realização de experimentos que demonstram que certas características da comunicação podem influênciar no processo de otimização da aplicação.

Dessa forma o Capítulo 2, é apresentado uma revisão bibliográfica acerca dos conhecimen-

tos básicos da computção paralela e de alto desempenho. Neste Capítulo, são apresentados alguns conceitos fundamentais, assim como a descrição das arquiteturas de *hardware* e os paradigmas que envolvem a programação paralela.

O estudo sobre a comunicação TCP/IP, apresentado no Capítulo 3, investiga o impacto da utilização do protocolo TCP/IP para troca de mensagens entre os processos. Nesse Capítulo é questionado o *overhead* inerente do protocolo e também o tempo de CPU utilizado para o processamento das mensagens. Ainda nesse Capítulo é analisado um modelo abstrato chamado VIA (*Virtual Interface Architecture*) que como alternativa a utlização do protocolo TCP/IP apresenta características importantes para prover um ganho de desempenho

Por fim o Capítulo 4, mostra que algumas aplicações possuem características de comunicação que possibilitam, utilizando o mesmo equipamento (*hardware*), prover um ganho de desempenho otimizando a distribuição dos processos paralelos. Para isso, são utilizadas as aplicções WRF e NAS, ambas bem difundidas no cenário científico.

1.1 Motivação

A tecnologia *multi-core* se desenvolveu com a constatação que o aumento do *clock* de processamento não era mais viável dado o seu alto consumo energético e a alta dissipação de calor gerada. Para contornar esses problemas técnico ao invés do aumento do *clock* foi proposto a utilização de mais de um núcleo (*core* em Inglês) de processamento com *clock* menores, assim como o aumento da memória *cache* .

A evolução dos microprocessadores indica que a tecnologia *multi-core* está se consolidando como a atual tendência do mercado, tanto para computares pessoais como para servidores. Diferentemente dos processadores sequênciais, que já vinham sendo estudados há algum tempo, os processadores *multi-core* abrem um novo ramo para uma detalhada pesquisa do seu pontencial.

Como o termo *multi-core* sugere, muitos *cores* podem ser adicionados a um único *chip*, por exemplo a fabricante intel já produziu um *chip* que possui 80 núcles (INTEL, 2007), e com esse aumento a importância de sistemas de comunicação como arquiteturas *network-on-chip* aumenta (FLICH et al., 2008), contudo, não para interligar computadores, mas sim os núcleos presentes em um único processador e essa disposição pode impactar no desempenho das aplicações.

Tendo a disposição a oportunidade de realizar um trabalho de vanguarda a motivação desse trabalho se baseia na necessidade de um melhor entendimento dessa nova tecnologia, assim

como uma avaliação do seu potencial como ferramenta, para auxiliar na otimização do desempenho dos componentes de comunicação de aplicações científicas.

1.2 Objetivos Gerais

O primeiro objetivo desse trabalho é o de realizar um estudo bibliográfico e científico aplicando o conhecimento adquirido durante o decorrer curso. Porém, de uma forma mais aplicada esse trabaho tem como objetivo geral o entendimento e o estudo aprofundado de alguns componentes envolvidos na comunicação de ambientes de computação paralela e de alto desempenho, como por exemplo, o *overhead* encontrado na comunicação entre os processos e também compreender algumas tecnologias alternativas ao uso de redes *ethernet* para a interconexão desses ambientes.

E por fim, considerando o estudo desenvolvido e uma aplicação paralela difundida no meio científico, através de métodos empíricos prover uma otimização do ambiente computacional auxiliando os componentes de comunicação envolvidos no processo.

1.3 Objetivos Específicos

- Desenvolver uma visão geral dos ambientes de computação paralela e de alto desempenho, proporcionando conhecimento necessário para classificá-los de acordo com suas aplicações;
- Entender a comunicação realizada entre os processos durante a execução de aplicações paralelas.
- Conhecer os motivos do *overhead* existente na comunicação entre processos utilizando o protocolo *TCP/IP* e analisar as suas consequências em um ambiente de cluster;
- Aprofundar o entendimento das redes de interconexão que utilizam a tecnologia *VIA* a fim de compreender como seu desenvolvimento proporciona melhor desempenho comparativo a redes que utilização o protocolo *TCP/IP*;
- Utilizar alguma aplicação científica e/ou *grand challange* em um ambiente de cluster.

1.4 Metodologia

Essa monografia tem seu foco no entendimento dos aspéctos que constituem a base do conhecimento no âmbito da computação paralela, distribuída e de alto desempenho, bem como no estudo do impacto da comunicação TCP/IP utilizada para troca de mensagens entre os processos, nos avanços da tecnologia VIA e na realização de experimentos que demonstram que certas características da comunicação podem influênciar no processo de otimização da aplicação.

No Capítulo 2, é apresentado uma revisão bibliográfica acerca dos conhecimentos básicos da computção paralela e de alto desempenho. Neste Capítulo, são apresentados alguns conceitos fundamentais, assim como a descrição das arquiteturas de *hardware* e os paradigmas que envolvem a programação paralela.

O estudo sobre a comunicação TCP/IP, apresentado no Capítulo 3, investiga o impacto da utilização do protocolo TCP/IP para troca de mensagens entre os processos. Nesse Capítulo é questionado o *overhead* inerente do protocolo e também o tempo de CPU utilizado para o processamento das mensagens. Ainda nesse Capítulo é analisado um modelo abstrato chamado VIA (*Virtual Interface Architecture*) que como alternativa a utlização do protocolo TCP/IP apresenta características importantes para prover um ganho de desempenho

Por fim o Capítulo 4, mostra que algumas aplicações possuem características de comunicação que possibilitam, utilizando o mesmo equipamento (*hardware*), prover um ganho de desempenho otimizando a distribuição dos processos paralelos. Para isso, são utilizadas as aplicções WRF e NAS, ambas bem difundidas no cenário científico.

1.5 Trabalhos Relacionados

Seguindo a organização lógica da monografia os trabalhos relacionados estão dispostos em função de suas contribuições a esse estudo.

Para a caracterização da comunicação MPI referente ao NAS Parallel Benhchmarks os trabalhos de (KIM; LILJA, 1998), (TABE; STOUT, 1999), (FARAJ; YUAN, 2002) e (SOKOLOWSKI; GROSU, 2004) demonstram os tipos de comunicação, quantida, frequência e tamanho das mensagens transmitidas. Já o trabalho de (SUBHLOK; VENKATARAMAIAH; SINGH, 2002) foca seus estudos na quantificação da utilização do procesador e memória, além de considerarem o volume de dados transmitidos.

Os trabalhos a seguir fornceneram o embasamento teórico para o entendimento do modelo

do aplicativos WRF: (KERBYSON; BARKER; DAVIS, 2007), (SKAMAROCK et al., 2008), (AMSTRONG et al., 2006), (MICHALAKES et al., 1999), (MICHALAKES et al., 2004), (ZAMANI; AFSAHI, 2005), e para a caracterização da comunicação os trabalhos (ARMSTRONG et al., 2006) e (ZAMANI; AFSAHI, 2005)

Apersar de ser uma tecnologia totalmente consolidada, tanto em ambientes científicos com comerciais, o protocolo TCP/IP apresenta alguns questionamentos sobre o seu rendimento e desempenho, tendo isso, os estudos (CLARK et al., 1989) (GALLATIN; CHASE; YOCUM, 1999) quantificam o overhead e tempo de CPU gastos com o processamento da comunicação inerente da troca de mensagens de computadores de um cluster.

No Capítulo 4 a tecnologia VIA (YU; LEE; MAENG, 2001) é estudada e para o entendimento do seu modelo foram utlizados os trablahos de (DUNNING et al., 1998), (EICKEN; VOGELS, 1998), (COMPAQ; INTEL; MICROSOFT, 1997). Para o fornecimento de dados realisticos sobre implementações comerciais do model VIA os trabahos (YU; LEE; MAENG, 2001) e (PENTAKALOS, 2002) foram utlizados.

2 *Computação Paralela e de Alto Desempenho*

O computador surgiu da necessidade do ser humano por uma ferramenta que lhe ajudasse a resolver problemas de origem matemática. Desde então, essa ferramenta se mostrou cada vez mais útil em diversas áreas do conhecimento humano e assim, gradativavente, problemas mais complexos começaram a ser modelados para serem processados por computadores.

Contudo, esses problemas consumiam cada vez mais recursos computacionais, o que por sua vez, exigia que os recursos evoluissem para atender a crescente demanda e em meados de 1950 a Lei de Grosh (PARHAMI, 2002) dizia que a melhor forma de obter melhor desempenho era aumentando o poder de processamento do processador. Essa lei foi apoiada pelos grandes *mainframes* e minicompudadores até a década de 70, quando foi evidenciada as dificuldades de produzir processadores sequências cada vez mais rápidos e também quando um conceito, igualmente da época de 1950 começou a ganhar força, o conceito de que para obter um aumento de desempenho era necessário a utlização de computadores paralelos que poderiam operar tarefas simultaneamente (ALMASI; GOTTLIEB, 1994).

A lei de Moore (QUINN, 1994; KUMAR et al., 1994) de 1965, afirma que o número de transistores de um circuito integrado, tem seu crescimento de acordo com uma função exponencial e que a cada dezoito a vinte e quatro meses, aproximandamente, o número de transistores dobra. Essa tendência de crescimento vem sendo apioada pelo crescimento histórico dos processadores, como mostra a figura 2.

O poder computacional de um processador não cresce linearmente com o aumento do número de transsistores e com a grande taxa de crescimento dos *chips* alguns problemas físicos, como por exemplo, temperaturas relativamente altas podem causar um consumo exessivo de energia, acarretando em custos para o resfriamento do processador e perigo de interferências entre componentes o que afetaria a confiabilidade do produto(CHAPARRO et al., 2007).

Devido a essas limitações físicas a evolução dos processadores sequenciais foi abalada.

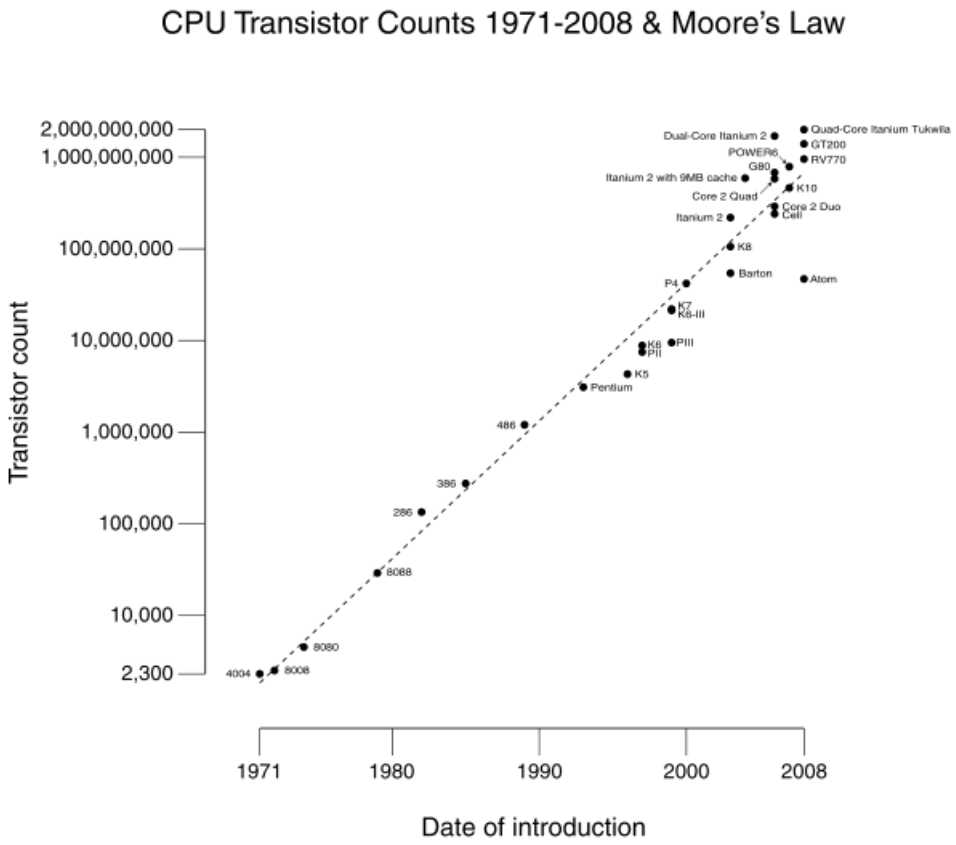


Figura 2.1: Lei de Moore. (WGSIMON, 2008)

Como uma primeira solução, processadores, ainda sequenciais, porém energeticamente mais econômicos foram desenvolvidos e posteriormente, com o intuido de continuar o ciclo de aumento do poder computacional, processadores paralelos, *multi-core*, começaram a ser desenvolvidos.

2.1 Taxonomias e Arquiteturas Paralelas

Data a ampla variedade de cassificações possíveis para ambientes computacionais paralelos e de alto desempenho esta seção concentra seus esforços em explicar algumas delas. Primeiro uma classificação formal proposta por Flynn, depois a lei de Amdahl que faz menção ao *speedup* de aplicação paralelas e por último uma classificação perante a organização do *hardware* dos ambientes.

	Single Instruction stream	*Mutiple Instruction streams*
Single Instruction stream	SISD	MISD
Mutiple Instruction streams	SIMD	MIMD

Tabela 2.1: Taxonomia de Flynn

2.1.1 Taxonomia de flynn

Taxonomia de Flynn (FLYNN, 1972) é uma classificação de arquiteturas computacionais porposta por Michael J. Flynn em 1966. Essa classificação se baseia no número de fluxos de instruções(*Instruction Streams*) e no fluxo de fluxos de dados (*Data Streams*) como demonstrado na tabela 2.1.

A classificação SISD (*Single Instruction, Single Data*) refere-se aos computadores sequênciais, até tempos atrás os mais comuns, baseados na arquitetura de von Neumann (FOSTER, 1995), aonde a cada ciclo do processador uma instrução opera, de maneira sequêncial, sobre apenas um conjunto de dados (memória).

Sendo um tipo de computador paralelo, o modelo SIMD (*Single Instruction, Multiple Data*) executa a mesma instrução sobre um conjunto de dados diferentes, pode ser encontrado duas variações: *Processor Arrays* e *Vector Pipelines* . Dessa forma, essa arquitetura é indicada para resolução de problemas específicos, como por exemplo, processadores gráficos.

A classe MISD (*Mutiple Instruction, Single Data*) aparece de maneira coadjuvante no desenvolvimento da taxonomia. Uma das poucas aplicações para essa classe são ambientes que necessitam de tolerancia a falha, dado que o mesmo dado é processado por vários processadores.

Por fim, a classificação MIMD (*Multiple Instruction, Multiple Data*) apresenta-se como a mais aproveitada em ambientes de processamento paralelo, pois, um fluxo de dados podem ser processados por um conjunto de processadores distintos, processadores esses que podem estar no mesmo *chip* (*multi-core*), no mesmo computador ou em computadores distribuídos.

Na próxima seção será abordado a questão das arquiteturas que se classficam como MIMD.

2.1.2 Ambientes de memória compartilhada: Multiprocessadores

Nos ambientes multiprocessadores de memória compartilhada existe apenas um espaço de endereçamento de memória, sendo esse, global e comum a todos os processadores. Dessa forma não existe um enderaçamento reservador para um processador específico o que facilita a programação de sistemas que utlizam esses ambientes, pois pode-se fazer uso de variáveis

globais de status para sincronização dos processos.

A figura 2.1.2 representa a organização básica de um ambiente de memória global compartilha.

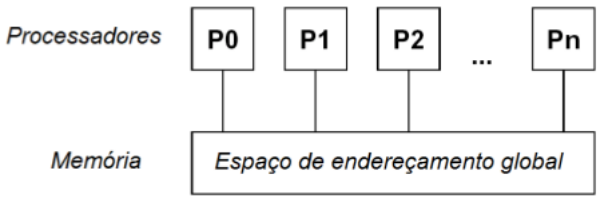


Figura 2.2: Ambiente de memória compartilhada

A seguir são apresentado duas formas de acesso a memória compartilhada.

2.1.2.1 Memória Compartilhada Centralizada (UMA)

Essa arquitetura é referênciada por UMA(*Uniform Memory Access*) como também SMP (*Symmetric Multiprocessor*)(HENNESSY; PATTERSON, 2003) e possui um número limitado de processadores, dificilmente passando de 32. Esse limitação se existe pois todos os processadores acessam a mesma memória física e para isso utlizam o mesmo barramento de acesso, o que causa um gargalho com o aumento do número de processasores. Outro ponto que importante a ressaltar é que o sistema operacional não faz nenhuma distinção entre os processadores, assim todos possuem o mesmo tempo de acesso a memória principal.

2.1.2.2 Memória Compartilhada Distribuída (NUMA)

Chamada de NUMA (*Non-Uniform Memory Access*), essa arquitetura constitui-se de vários ambientes UMA conectados por uma rede de interconexão. Como a figura 2.1.2.2 mostra, todos os processadores tem acesso a qualquer endereçamento da memória, porém esse acesso não é mais realizado de forma uniforme. Outra característica é que um número limitado de processadores concorrem pela utlização do barramento diretamente conectado, assim como os aplicativos tem como característica uma certa localidade espacial (HENNESSY; PATTERSON, 2003) a frequência com que um processador necessite acessar uma endereço de memória muito distante do endereço atual diminui e consequentemente o gargalo no barramento diminuirá.

Com essa váriação no modo de acesso a memória, não é difícil de encontrar ambientes NUMA com 512, 1024 ou mais processadores

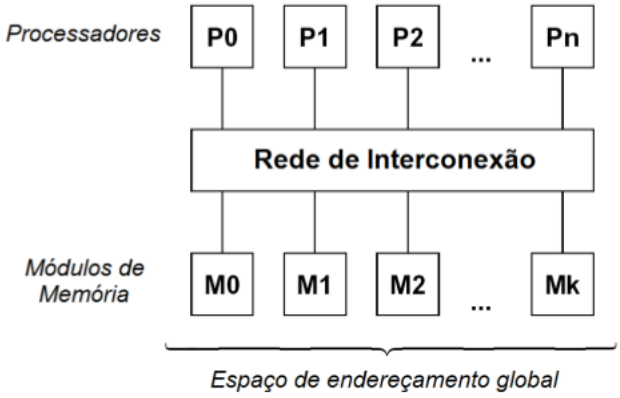


Figura 2.3: Ambiente de memória compartilhada NUMA

2.1.3 Ambientes de memória distribuída: Multicomputadores

Como o nome sugere, este ambiente não possui uma única memória global acessível a todos os processadores do ambiente, ao contrário, cada processador possui a sua própria memória tanto a nível físico quanto operacional, por isso o nome de multicomputadores. Dessa forma, quando um processador necessita de dados que não foram encontrados em sua memória é necessário utilizar uma rede de interconexão para se comunicar com outro computador e assim trocar mensagens para que seja possível a sincronização dos processos (FOSTER, 1995).

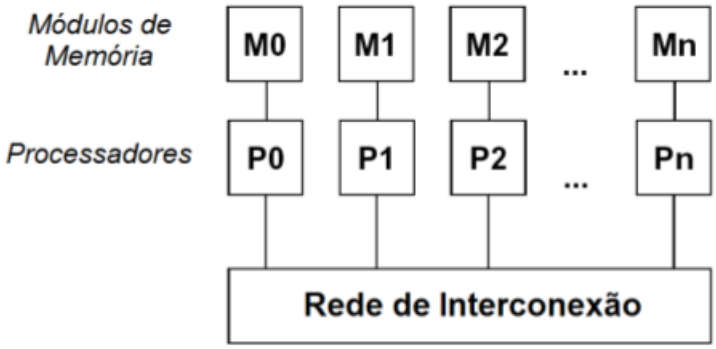


Figura 2.4: Ambiente de memória distribuída

A figura 2.1.3 demonstra a organização de um ambiente de memória distribuída, observando a figura fica claro que a parte de programação dos sistemas que utlizaram essa arquitetura sofrem um ganho de complexidade, uma vez que não é mais possível acessar diretamente a memória desejada.

Devido a sua organização esse ambiente possui um escalabilidade muito boa podendo che-

gar até a centena de milhares de processadores e diferentemente, dos ambientes de memória compartilhada que são fortemente acoplados, esses são fracamente aclopados. Como exemplo, pode-se citar os ambientes de *cluster* que estão sendo cada vez mais adotados para a resolução de problemas que necessitam de ambientes de alto desempenho.

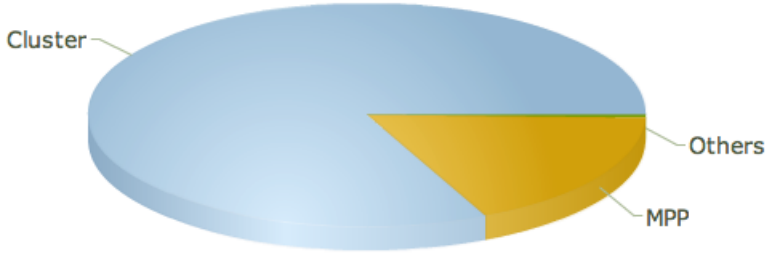


Figura 2.5: Relação entre ambientes de memória compartilhada e memória distribuída no ranking Top 500

A figura 2.1.3 mostra, a proporção do ambientes computacionais de memória distribuída (*cluster*) e memória compartilhada (MPP) presentes no site Top 500, que relaciona os 500 maiores ambientes computacionais de alto desempenho do mundo.

2.1.4 Lei de Amdahl

A lei de Amdahl (JORDAN; ALAGHBAND, 2003) é uma lei que quantifica o *speedup* de um aplicação quando essa é executada de forma paralela. De acordo com (KUMAR et al., 1994) a existência de um componente sequencial C_{seq} no algoritmo pode resultar em um gargalo e reduzir o desempenho

Segundo essa lei, o máximo de *speedup* que pode ser alcançável por ambiente computacional paralelo é dado pela seguinte fórmula, com ,$0 \leq C_{seq} \leq 1$:

$$S_{max}(n) <= \frac{1}{C_{seq} + \frac{1-C_{seq}}{n}} = \frac{n}{1+(n-1)C_{seq}} \tag{2.1}$$

, onde S_{max} é o *speedup* máximo que pode ser atingido utilizando n processadores paralelos e o mesmo conjuto de dados de entrada, também chamado de *workload*.

Analisando a fórmula 2.1 é possível afirmar que o *speedup* de uma aplicação paralela nunca poderá aumentar linearmente com o aumento do número de processadores, pois, sempre existirá uma porção C_{seq} que influenciará negativamente no aumento do *speedup*, porém, se o *workload*

não for constante e também crescer, então o *speedup* também crescerá, pois o *overhead* total é independente do tamanho do *workload* e assim terá menos impacto na parte, uma vez que a parte C_{seq} ficará cada vez menor proporcionalmente ao conjunto de entrada de dados, esse comportamento é chamado de efeito Amdahl (KUMAR et al., 1994).

2.2 Paradigimas de Programação Paralela

Com o foco da tecnologia de processadores voltado aos *multi-cores*, uma questão muito importante vem a tona, os aplicativos comuns estão preparados para extrair o verdadeiro poder de processamento que essa nova tecnologia está oferecendo ? A resposta é não, pois os a maioria dos aplicativos não estão modelados sobre um paradigima paralelo, mas sim em um paradigima sequencial.

Essa seção explora exatamente os paradigimas de programação paralelo existentes. Dado a variedade de arquiteturas paralelas também existem vários paradigimas de programação paralela que não necessariamente estão vinculados com a arquitetura dos ambientes, porém, para que se obtenha um melhor desempenho é aconselhável que o modelo de programação escolhido seja condizente com a arquitetura projetada. Outra questão relevante sobre os paradigimas refere-se a dificuldade de programação que esses modelos apresentam. Por exemplo, pode-se optar por uma linguagem de desempenhho inferior em detrimento de uma programação mais facíl.

2.2.1 Granularidade

O paralelismo de um ambiente está diretamente ligado a iteração entre os processadores, tanto a nível arquitetural (*hardware*) como a nível da aplicação que possui as características do algoritmo e o paradigima de programação escolhido.

Essas iterações representam um tempo significativo na execução do aplicativo, uma vez que exista uma dependencia entre os dados, processos ao mesmo tempo podem necessitar dados processados por outro processador e também fornecer dados para que um teceiro processador consiga executar sua tarefa.

Dessa forma, A granularidade de uma aplicação paralela pode ser definida como a razão do tempo de computação em relação ao ao tempo gasto com a comunicação (e sincronização) (KUMAR et al., 1994).

$$Granularidade = \frac{T_{comp}}{T_{com}} \tag{2.2}$$

De acordo com (PINTO, 2008), a granularidade da comunicação de uma aplicação pode ser classificada como:

- grossa: necessidade de comunicação infrequente, com longos períodos de computação idependente de comunicação, e volume total de comunicação desconsiderável.
- média: necessidade de comunicação infrequente, com períodos médios a longos de computação idependente de comunicação, e volume total de comunicação considerável.
- fina: necessidade de comunicação frequente, com curtos períodos de computação seguidos de comunicação, e volume total de comunicação considerável.

2.2.2 Memória Compartilhada e Passagem de Mensagens

Esses dois modelos de programação paralela podem ser relacionados com as arquiteturas acima descritas. Para o modelo de memória compartilhada fica explicita a sua relação com a arquitetura de memória compartilhada, tanto UMA quanto UMA, e por sua vez o modelo de passagem de mensagens está mais intimamente ligado a arquitetura de memória distribuída, porém, esses modelos de programação não são exclusivos para as arquiteturas, é perfeitamente possível utlizar um modelo de programação em uma arquitetura não tão relacionada com a sua origem, mas para isso algumas adaptações se fazem necessárias.

No paradigima de memória compartilhada os processos utlizam a memória para comunicarem uns com os outros. Como, não existe espaco de memória privada para um processo a utlização de semáforos ou bloqueis, por exemplo, tem a função de garantir a exclusão mútua aos recursos do sistema (KUMAR et al., 1994). Com as informações acima fica claro que esse paradigima apresenta facilidades na sua programação.

Como pode ser verificado na figura 2.1.3 os ambientes de memória distribuída constituem a maioria dos ambientes de computação de alto desempenho, assim é de se esperar que o modelo de programação inerente dessa arquitetura seja o mais utlizado também, como citado por (FOSTER, 1995). Como nesse paradigima não existe o conceito de memória compartilhada, então, mensagens são trocadas entre os processes a fim de prover a sincronização e a troca de dados. Devida a sua organiação esse paradigima apresenta mais dificuldade de programação se comparado ao modelo acima.

Em (PINTO, 2008) são encontrados exemplos de API (*Application programming interface*) para os paradigimas citados acima, são eles:

- OpenMP: para memória compartilhada.
- MPI (*Message-Passing Interface*): para memória distribuída

3 *Redes de Interconexão*

Anteriormente os conceitos essenciais acerca da computação paralela e de alto desempenho foram inseridos nesse estudo, porém, desde ponto em diante o foco desse trabalho se volta para os ambientes de memória distribuída, mais especificamente para os ambientes denominados *clusters* e a nível de paradigima de programação o foco recai ao modelo de programação distribuída MPI. Essa especialização do estudo será concluída no Capítulo 4 onde o experimento realizado utliza todos os conceitos teóricos apresentados nesse estudo.

Dada as conciderações acima, esse capítulo irá analisar a o impacto da rede de interconexão utlizada para conectar os processos de um ambiente de memória distribuída, entenda-se *cluster*. A necessidade dessa análise pode ser justificada por (JORDAN; ALAGHBAND, 2003) que atribui a rede de interconexão a responsabilidade de conectar fisicamente todos o núcleos de processamento e suportar de forma confiável, a transmissão e recebimentos dos dados trocados entre os processos.

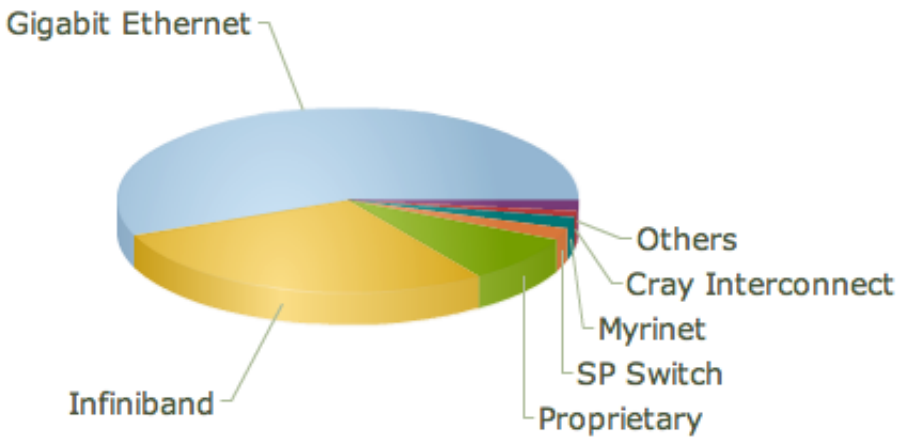


Figura 3.1: Relação entre as redes de interconexão utilizadas nos ambientes do ranking Top 500

No decorer desse capítulo a primeira seção estuda o custo oriundo do processamento da troca de mensagens entre os processos e o *overhead* presente quando uma rede de interconexão

baseada no protocolo TCP é utlizada e a última seção discute as alternativas apresentadas pelo modelo VIA(*Virtual Interface Architecture*) (DUNNING et al., 1998) para prover um ganho de performance considerável quando comparado com as redes TCP.

3.1 Protocolo TCP

O protocolo de transporte TCP juntamente com o protrocolo de rede IP vem sendo largamanete usados desde a sua criação na rede ARPANET (TANEMBAUM, 1997). Tamanha é sua importância que foi criado um modelo de referência de redes baseado em sua arquitetura, como mostra a figura 3.1.

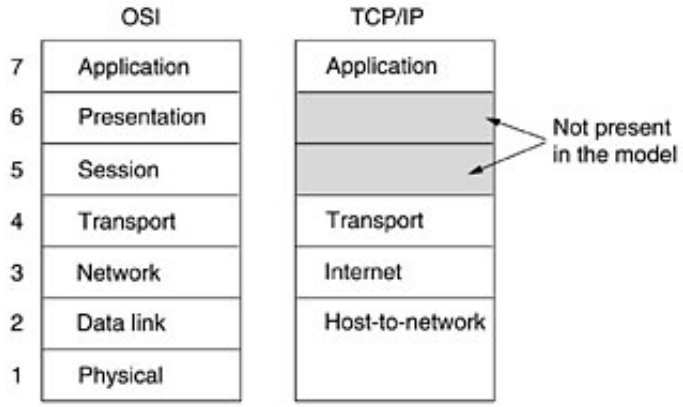


Figura 3.2: O modelo de referência TCP/IP

Acoplado com a tecnologia *Ethernet*, o protocolo TCP/IP vem sendo usado como protocolo para vários ambientes computacionais, desde uma LAN(*Local Area Network*), a Internet até ambientes de computação de alta desempenho. Porém, para poder proporcionar essa flexibilidade a norma (RFC793, 1981), que regulamentou o protocolo TCP/IP, vem sendo alterada constante para para correção de erros, incosistências e mudanças de requisitos (TANEMBAUM, 1997).

Essas alteraçãos fundidas com a generalidade que o protocolo TCP/IP adquiriu, faz com que essa solução não seja a mais interessante para alguns ambientes computacionais, por exemplo um *cluster*, uma vez que o motivo de sua criação foi de oferecer um fluxo de bytes, fim a fim confiável em um rede não confiável além de se adptar dinamicamente a rede e ser tolerante a alguns tipos de falha (TANEMBAUM, 1997).

Apesar de um ambiente de cluster de alto desempenho requisitar algumas das características presentes no protocolo TCP/IP, por exemplo, a confiabilidade de entrega dos dados, outras

caraciterísticas não são bem vindas, uma vez que o foco é um ambiante otimizado de alto desempenho.

Vamos conciderar uma LAN utlizando utlizando uma rede *Ethernet* TCP/IP, dedicada para um ambiente de cluster, onde todos os nós de processamento estão conectados através de um *switch*, dada as condições normais de funcionamento da mesma é comprovado que o taxa de perda de TPDUs (*Transport Protocol Data Unit*) nessa rede é muito baixa, consequentemente, a necessidade de retransmissão dos mesmo TPDUs também é baixa, porém a funcionalidade de retransmissão do protocolo TCP/IP está presente em todas as mensagens, indiferente da sua necessidade ou não (RFC793, 1981). O Exemplo acima questiona apenas um dos *overheads* presentes no protocolo TCP/IP que degrada a otimização de uma ambiente de computação de alto desempenho, outros exemplos que podem ser citados são o *handshake* inicial de conexão e o encerramento de uma conexão, que necessitam da troca várias mensagens para que sejam efetuadas.

Per byte:	
User-system copy	200 µs
TCP checksum	185 µs
Network-memory copy	386 µs
Per packet:	
Ethernet driver	100 µs
TCP + IP + ARP protocols	100 µs
Operating system overhead	240 µs

Figura 3.3: Tempos de atraso inerentes da comunicação TCP/IP. (CLARK et al., 1989)

Analisando um pouco melhor a questão *overhead* da comunicação TCP/IP, é possível verificar que não apenas o *overhead* causado pela traca de mensagens impacta no desempenho do ambiente, mas também a frequência e o tamnho dos TPDUs transmitidos. Isso se deve ao processamento do cabeçalho de cada TPDU, esse impacto pode ficar claro observando o tempo e processamento para calcular o total de verificação (*checksum*) nas figuras 3.1 e 3.1. A título de exemplo, quando se transmite um milhão de bytes, o *overhead* por byte é o mesmo, não importando o tamanho da TPDU, porém, TPDUs de 128 bytes significa cirar um *overhead* 32 vezes maior por TPDU em relação ao uso de TPDUs de 4Kb (TANEMBAUM, 1997).

Não distante das questãos apresentadas acima esta outro ponto que merece uma observação mais detalhada, a questão do custo de processamento resultande da comunicação TCP/IP. De acordo com (TANEMBAUM, 1997) a ação mais importante para prover ganho de performance é aumentando o poder de processamento dos nós. As figuras 3.1 e 3.1 ajudam a confirmar essa afimação, pois respectivamente monstram os tempos de atraso inerentes da comunicação

TCP/IP e a utlização da CPU durante o recebimento de TPDUs de diferentes MTUs (*Maximum Transmissino Unit*) em diferentes larguras de banda, vale ressaltar que em alguns casos o processador se ocupa mais de 80% apenas para o processamento da TPDU.

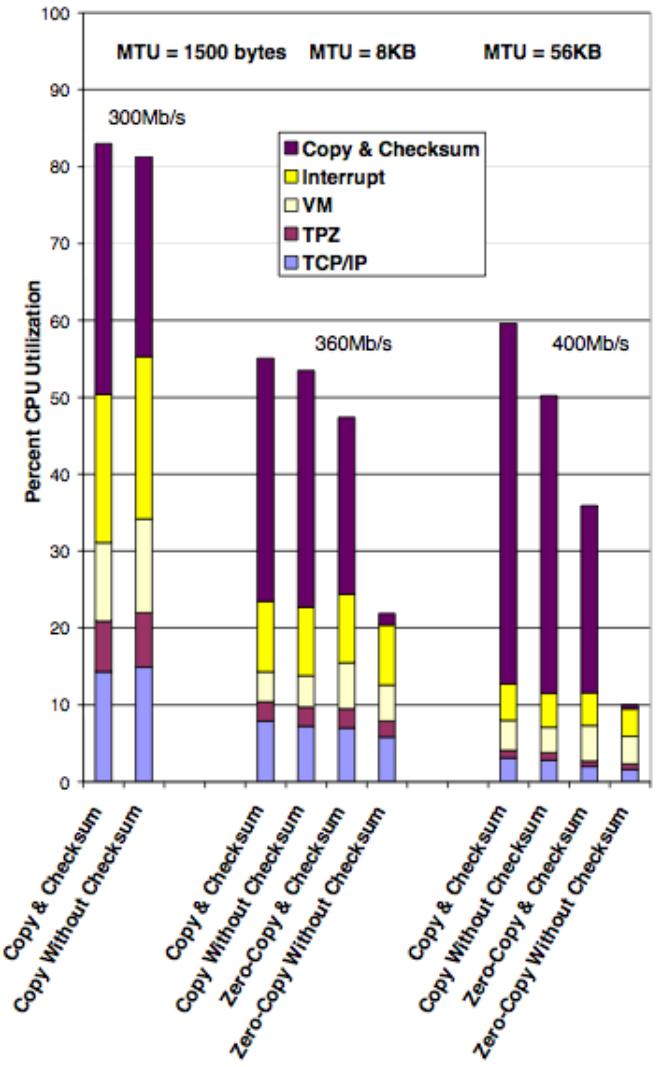


Figura 3.4: Utilização da CPU no recebimento de mensagens TCP. (GALLATIN; CHASE; YOCUM, 1999)

Tendo a mesma importância que os demais casos citados, a mudaça de contexto, tanto de modo usuário para modo *kernel* como vice-versa são críticos ao desempenho da rede. Conforme (HENNESSY; PATTERSON, 2003) há um grande custo em uma troca de contexto de um processo para outro tipicamente requer de centenas a milhares de ciclos de processamento, provocando um *overhead* considerável, além de na pior da hipóteses causar uma sequência de erros de *cache* e na melhor das hipóteses a mudança de contexto acontece do contexto atual para o contexto do *kernel* e depis novamente para o contexto do usuário do processor receptor,

no entanto, muitos sistemas operacionais executam várias outras mudanças nesse procedimento (TANEMBAUM, 1997).

Apesar de todos problemas apresentados pelo protocolo TCP/IP para a confecção de um cluster otimizado e eficiente, essa abordagem ainda é a mais utilizada, como pode ser verificado na figura 3. Essa liderança se deve principalmente a fato financeiro sendo que o custo para a implementação de uma rede *Ethernet* TCP/IP tem um custo irrisorio comparado com os equipamentos do *cluster*.

3.2 VIA (*Virtual Interface Architecture*)

A VIA (*Virtual Interface Architecture*) é um modelo abstrato desenvolvido pela Microsoft, Intel e Compaq em 1996 e surgiu da necessidade de criar uma interface de rede de alto desempenho para uma rede local conhecida como SAN (*System Area Networks*). Esse modelo é de propriedade livre, independente do sistema operacional e independente do meio físico de transporte e está demonstrada na figura 3.2.

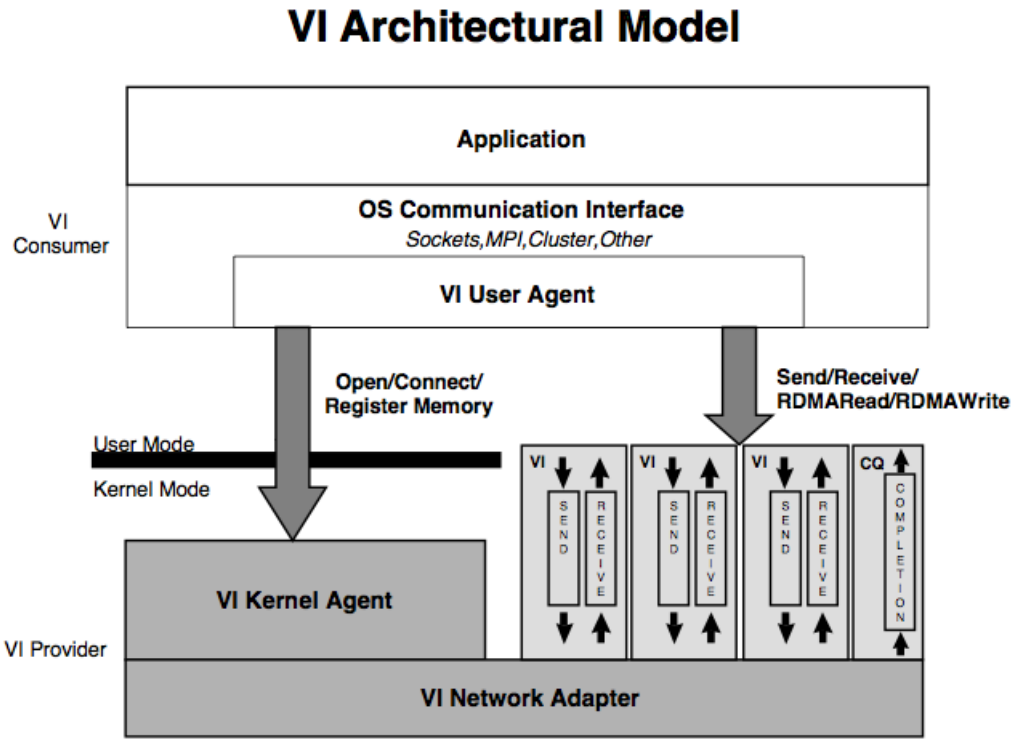
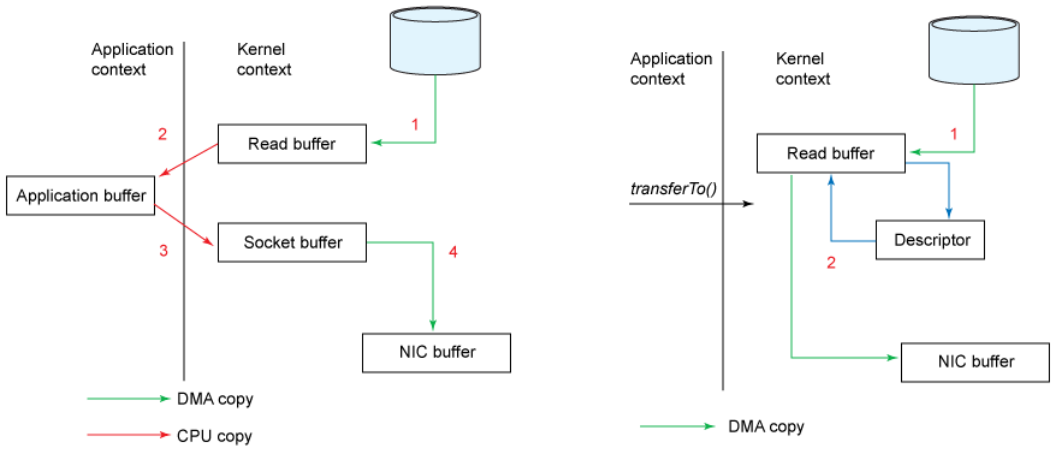


Figura 3.5: O modelo VIA. (COMPAQ; INTEL; MICROSOFT, 1997)

Uma das caracteristicas mais importante do modelo VIA é o seu conceito de *network-on-chip*, que significa que o HCA (*Host Channel Adapter*) possui um processador dedicado ao pro-

cessamento do fluxo de comunicação oriondo da troca de mensagens, dessa forma, o problema de mudança de contexto entre os processos, que era um dos gargalos mais críticos tratados na seção anterior, não existe mais, pois agora o processador principal do nó de processamento está disponível apenas para o processamento dos processos paralelos.



(a) Leitura dos dados sem *zero-copy* (b) Leitura dos dados com *zero-copy*

Figura 3.6: *Zero-Copy. (IBM, 2008)*

Uma vez que o fluxo da comunicação é processado um certo volume de dados é extraído desse fluxo, e outra propriedade do modelo VIA se torna importante o RDMA (*Remote Direct Memory Access*) (DUNNING et al., 1998), que como o próprio nome sugere, permite que um processo acesse a memória de outro nó de processamento, sem para isso, depender da CPU do processo de origem (HENNESSY; PATTERSON, 2003). Dessa forma, a entrega dos dados extraídos pode ser feita, sem a utilização da CPU e sem interferir no processamento em curso, envitanto assim, mais mudanças de contexto.

Para prover essas funcionalidades o RDMA faz uso do conceito de *zero-copy*, que consiste em transferir dados tento a memoria como destino ou origem sem passar pela aplicação, dessa forma, não é necessario a cópia dos dados para o nível de usuário, entenda-se aplicação, e depois a cópia para os *buffers* do sistema operacional (IBM, 2008). A figura 3.2 exemplifica esse conceito.

Como pode ser verificado na figura 3.2, para o envio de dados utilizando o protocolo TCP/IP o fluxo da informação no modelo tradicional inclui a passagem da informação do nível de usuário para o nível de kernel, que possui várias outras etapas, para só então acessar o *hardware* e ser efetivamente transmitido. Porém, o modelo seguinte na figura 3.2 exemplifica o conceito de *kernel bypass*, que conciste no aplicativo transmitir diretamente os dados do nível de usuário para o hardware, sem passar pelo kernel do sistema (RISCA; MALIK; KESSLER, 2008). No-

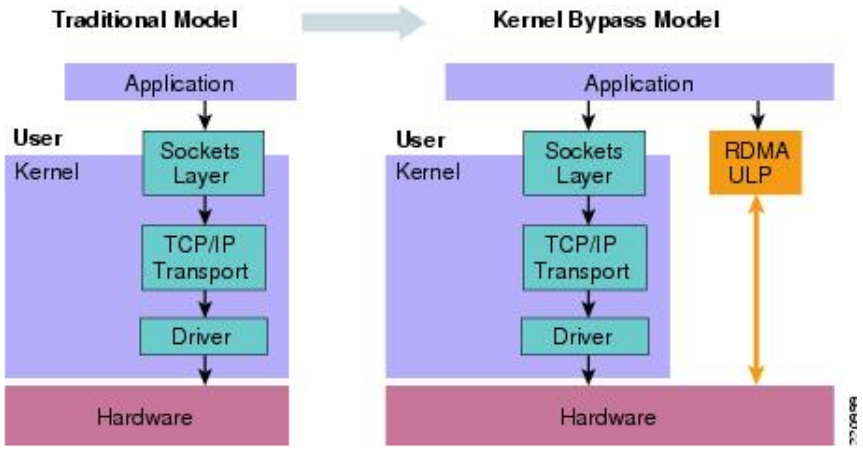


Figura 3.7: *Kernel bypass*. (RISCA; MALIK; KESSLER, 2008)

vamente, o principal intuito dessa ação é evitar as mudanças de contexto que existiriam caso os fluxo da informação utiliza-se o modelo tradicional.

Diferentemente do protocolo TCP/IP que possui a propriedade de se adaptar a várias aplicaçãoes e com isso acarreta um grande *overhead* de informação e processamento do nó da rede o modelo VIA tem sua aplicação bem definida, e utiliza várias técnicas para, principalmente, envitar a utilização do CPU para o processamento de comunicação como também minimizar as mudanças de contexto que geram um grande impacto em um ambiente de alto desempenho. Como o modelo VIA é um modelo abstrato, a título de curiosidade são apresentadas duas tecnologias que implementam esse modelo, a Infiniband (CASSIDAY, 2000) e a Myrinet (BODEN et al., 1995).

4 *Experimento*

Após o estudo teórico realizado, este capítulo apresenta um estudo empírico realizado no ambiente da EPAGRI S.A. (Empresa de Pesquisa Agropecuária e Extensão Rural de Santa Catarina). Através da abordagem de diminuir o custo de processamento relativo aos fatores resultantes da comunicação entre os processos parelelos, simplesmente mantendo um núcleo ocioso por nó em relação ao processamento da aplicação e dedicado ao processamento da comunicação, foi possível prover uma melhora no desempenho das aplicaçãoes NAS e WRF (*Weather Research Forecasting*) sem o acréscimo de *hardware*.

4.1 Abordagem Proposta

Devido à ausência de um processador de rede, todo processamento decorrente da comunicação via Ethernet é atribuída a um processador principal, isto é, concorrendo pelos mesmos processadores utilizados pela aplicação. No caso de um único processador por computador, a aplicação deve necessariamente parar sua execução para dar lugar ao processamento relativo à comunicação, decorrente da sua própria necessidade de interação entre os processos distribuídos, o que causa uma grande perda de desempenho.

Como citado no Capítulo 3, as redes de interconexão mais eficientes como Myrinet e Infiniband, baseadas no modelo VIA, são equipadas com processadores especializados e dedicados ao processamento da comunicaçã, localizados nas placas de rede em cada computador e no dispositivo de interconexão. Além disso, o fluxo de comunicacão é desviado do sistema operacional e a transmissão dos dados é feita via RDMA para a placa de rede, enquanto que na tecnologia *Ethernet* é necessário envolver o sistema operacional e algum processador principal no processo de comunicação.

Ainda mais com o advento dos processadores *multi-core*, computadores com múltiplos núcleos estão se consolidando como uma tecnologia de prateleira (commodity). Quando tais computadores são agregados via Ethernet para a composição de um cluster, surge a possibi-

lidade de melhorar o desempenho do sistema através da sobreposição do processamento da aplicação e do processamento decorrente da comunicação entre os processos da aplicação. Porém, quando todos os núcleos são alocados para a execução da aplicação, ocorrem inúmeras trocas de contexto entrem os processos dos diferentes níveis, já que a efetivação da comunicação não se dá em nível de aplicação. Dado o grande custo da uma troca de contexto de um processo para outro tipicamente requer de centenas a milhares de ciclos de processamento, provocando um overhead considerável (HENNESSY; PATTERSON, 2003).

Enfim, a abordagem proposta sugere que não sejam alocados aos processos da aplicação todos os núcleos de processamento disponíveis em cada computador, de modo que os núcleos então ao ociosos em relação à aplicação podem se ocupar com as tarefas decorrentes de sua execução, como o processamento de pacotes e protocolos de comunicação. Por- tanto, em função desse paralelismo, espera-se que o tempo gasto com o processamento decorrente da comunicação entre processos tenha seu impacto reduzido, traduzindo-se em um menor tempo de execução da aplicação. Esta é a idéia norteadora da abordagem proposta neste estudo de caso.

4.2 Ambiente Experimental

Três ambientes distintas de cluster serão avaliados no que diz respeito à distribuição dos processos e o número de núcleos de processamento utilizados por computador agregado. Além disso, experimentos em um único computador multiprocessado com até 8 núcleos de processamento serão apresentados a titulo de comparação, ampliando a análise em nível de arquiteturas paralelas e de processadores.

A nomenclatura dos ambients descritos na figura 4.2 seguem um padrão conforme o número de nós (N) e o número de núcleos de processamento ativos por nó (P). Por exemplo, o sistema N8xP1 é composto por 8 computadores (N = 8), cada um com 2 núcleos de processamento, porém apenas 1 núcleo é alocado para a aplicação (P = 1). O segundo núcleo de cada computador está, portanto, ocioso em relação à aplicação.

Os ambientes baseados em processadores Xeon (INTELÆ, 2004) rodam Linux kernel 2.6.8.1 e os ambientes baseados em processadores Opteron (AMD, 2007) rodam Linux kernel 2.6.22.8, todos com suporte a SMP ativado. Além disso, todos os sistemas estão isolados de interferencias externas e dedicados aos experimentos, isto é, não estão operando quaisquer outros serviços, exceto a configuração mínima necessária à execução dos experimentos com a biblioteca MPICH2.

INFO / SISTEMA	N8xP2	N8xP1	N4xP2	CMP-SMP
Interconexão	Gigabit Ethernet (GigE)			Crossbar
	3Com Switch 3812			HyperTransport
MTU	1500			N/A
Modelo do processador	32-bit Intel Xeon (DP)			64-bit AMD Opteron 2350
Veloc. do processador	2.66 GHz			2 Ghz
Tecnologia Manuf.	130nm			65nm
# Cores por soquete	1			4
# Cores por nó	2	2	2	8
# nós	8	8	4	1
# Cores Ativo/Ocioso	*16/0*	*8/8*	*8/0*	*variável*
	POR CORE			
Cache L1 (I/D)	12KB/8KB			64KB/64KB
Cache L2	512KB			512KB
Cache L3	-			2MB
DRAM	512MB	1GB	512MB	1GB
Velocidade DRAM	533MHz			1000Mhz
BogoMIPS	~ 5300			~ 4000

Figura 4.1: Ambientes de experimentação

É importante ressaltar que as aplicações usadas nestes experimentos são baseadas exclusivamente em MPI para a extrações do paralelismo, embora haja trabalhos (RABENSEIFNER; WELLEIN, 2003) (CAPPELLO; ETIEMBLE, 2000) que indiquem um modelo de programação híbrida, com OpenMP para paralelismo intra-nó e MPI entre nós, como o meio mais eficiente de utilizar agregados de computadores multiprocessados. Porém, como o foco deste trabalho está em analisar a sobreposição de computação e comunicação levou-se em consideração apenas a comunicação MPI por soquete entre os processos, seja entre nós ou intra-nó. Além disso, programação paralela com MPI deve continuar importante por razões de portabilidade e até mesmo pela enorme quantidade de aplicações baseadas em MPI.

4.3 NAS Parallel Benchmarks

O NPB é um conjunto de oito algoritmos de benchmark. Cinco deles são denominados kernel benchmarks (EP, FT, IS, CG e MG) e os outros três são considerados aplicações simuladas de benchmark (SP, BT e LU) (BAILEY et al., 1991). A compilação foi configurada igualmente para todos os sistemas, utilizando-se a diretiva de otimização O3. Este trabalho abrange apenas os algoritmos EP, FT e IS, pois estes algoritmos apresentam granularidades bem distintas. O conceito de granularidade está associado à razão do tempo gasto com computação em relação ao tempo gasto com comunicação entre os processos distribuídos.

O algoritmo EP (Embarrassingly Parallel) é um gerador de números aleatórios que apre-

NAS.EP.B	*N*8*xP*2	*N*8*xP*1	*N*4*xP*2
Temp de execução	25,9	47,7	47,3
Redução de tempo	0%	-84,2%	-82,7%
Speedup relativo	1	0,54	0,55
NAS.FT.B	*N*8*xP*2	*N*8*xP*1	*N*4*xP*2
Temp de execução	57,5	53,2	82,7
Redução de tempo	0%	7,48%	-43,8%
Speedup relativo	1	1,08	0,70
NAS.IS.B	*N*8*xP*2	*N*8*xP*1	*N*4*xP*2
Temp de execução	4,23	3,23	4,60
Redução de tempo	0%	23,6%	-8,75%
Speedup relativo	1	1,31	0,92

Tabela 4.1: Resultado do *benchmark* NAS

senta alta demanda por computação e necessidade de comunicação desprezível. Sua demanda por comunicação limita-se à distribuição de dados em um momento inicial e ao reagrupamento dos resultados parciais ao final de sua execução. Portanto, o algoritmo EP é de granularidade grossa.

De fato, os resultados reafirmam essas características do algoritmo EP, pois com 16 processadores alocados à aplicação, seu tempo de execução diminui quase pela metade, como mostra a Tabela 4.1, enquanto que não há uma diferença significativa entre os resultados com 8 processos, obtidos pelos sistemas N8xP1 e N4xP2.

Já o algoritmo FT (FFT 3D PDE), que resolve equações diferenciais parciais através de FFT's 1D em série, apresenta alta demanda por computação assim como por comunicação, segundo um padrão de todos para todos perfeitamente balanceado. Como os autores do NPB tomaram o cuidado de agregar as mensagens em nível de aplicação para minimizar o custo de comunicação, o resultado é um algoritmo de granularidade média, com um grande volume de dados transmitidos embora através de mensagens grandes, a maioria da ordem de megabytes, em uma frequência baixa.

Apesar dessa baixa frequência das operações de comunicação, pode-se apontar um ganho leve no desempenho do algoritmo FT quando o cluster é composto segundo a abordagem proposta. Já a configuração N4xP2 resultou em uma enorme perda de desempenho em relação às configurações N8xP2 e N8xP1, o que revela certa fragilidade do subsistema de memória e de E/S desses computadores. Vale ressaltar que, embora a necessidade de comunicação com maior taxa de transferência creça proporcionalmente ao desempenho do processador, esse aumento não é significativo com o crescimento do número de processadores usados na execução do algoritmo FT (SUN; WANG; XU, 1997).

O algoritmo IS (Integer Sort) é um ordenador de inteiros. Predominam comunicacões de redução e de todos para todos não-balanceadas com muitas mensagens pequenas (até poucos kilobytes) e grandes (da ordem de megabytes), mas poucas mensagens de tamanho mediano. A frequência das mensagens é alta e o volume total de dados transmitidos é considerável. Por isso, a granularidade do algoritmo IS é considerada menor do que dos algoritmos EP e FT, caracterizando-se como de granularidade fina.

Mais uma vez, o desempenho do sistema N4xP2 mostrou-se o pior dentre as trˆes configurações. Por outro lado, o cluster configurado segundo a abordagem proposta, o sistema N8xP1 com 8 processos executando a aplicação, obteve uma redução de mais de 20% em comparação com o tempo obtido pelo sistema N8xP2, com 16 processos, na execução do algoritmo de granularidade fina IS.

Portanto, com base nos resultados obtidos com o NPB, é possível apontar que o sistema N8xP1, configurado segundo a abordagem proposta, teve sucesso em minimizar o *overhead* decorrente do processo de comunicação entre os processos. Exceto para o algoritmo de granularidade grossa, houve ganho de desempenho em relação à configuração N8xP2, na qual todos os 16 núcleos de processamento disponíveis foram alocados a processos da aplicação.

4.4 WRF (*Weather Research Forecasting*)

A aplicação WRF (Weather Research and Forecasting Model) é um sistema de previsão meteorológica baseado em modelos numéricos, ativamente desenvolvida por um consórcio de agências governamentais, como o NCAR (National Center for Atmospheric Research), em parceria com a comunidade científica dos Estados Unidos e internacional (MICHALAKES et al., 1999). É amplamente utilizado tanto na previsão de tempo operacional como na pesquisa atmosférica.

O WRF utiliza uma matriz tri-dimensional para a representação da atmosfera, desde metros ate milhares de quilômetros, com diversas informações como de topografia e dados de observatórios para alimentar a simulação com uma condição inicial. As simulações aqui apresentadas tomam como entrada o conjunto de dados da forma que é utilizado no ambiente de produção da EPAGRI S.A., que realiza a previsão do tempo para o estado de Santa Catarina.

Na execução paralela do modelo WRF, cada processo recebe uma sub-matriz de tamanho aproximadamente igual, que diminui com o aumento do número de processos. Quanto à comunicação, a redistribuição dos dados laterais, nos quatro limites lógicos de cada sub-matriz, é a principal atividade, ocorrendo a cada iteração com mensagens entre 10 e 100 kilobytes

2horas	*N*8x*P*2	*N*8x*P*1	*N*4x*P*2
Temp de execução (min:seg)	24:12	18:03	24:50
Redução de tempo	0%	25,4%	-2,6%
Speedup relativo	1	1,34	0,98
12horas	*N*8x*P*2	*N*8x*P*1	*N*4x*P*2
Temp de execução (h:mm:ss)	2:01:07	1:33:49	2:01:54
Redução de tempo	0%	22.5%	-0,7%
Speedup relativo	1	1,29	0,99

Tabela 4.2: Resultado do WRF nos diferentes ambientes

(KERBYSON; BARKER; DAVIS, 2007). Cada iteração avança o tempo de simulação em 75 segundos, totalizando 96 e 576 iterações nas previsões para 2 e 12 horas, respectivamente. Além disso, a cada 20 iterações ocorre uma iteração de radiação física, que se soma ao tempo de processamento da iteração.

De acordo com a Tabela 4.2 as previsões de tempo para 2 e 12 horas, apresentaram uma redução maior do que 20% no tempo de execuçãao. Os resultados obtidos com o WRF mostram um ganho de desempenho quando o cluster é configurado segundo a proposta, atingindo um speedup relativo de 1,29 em comparação à previsão do tempo para 12 horas no ambiente de cluster original, com 16 processos.

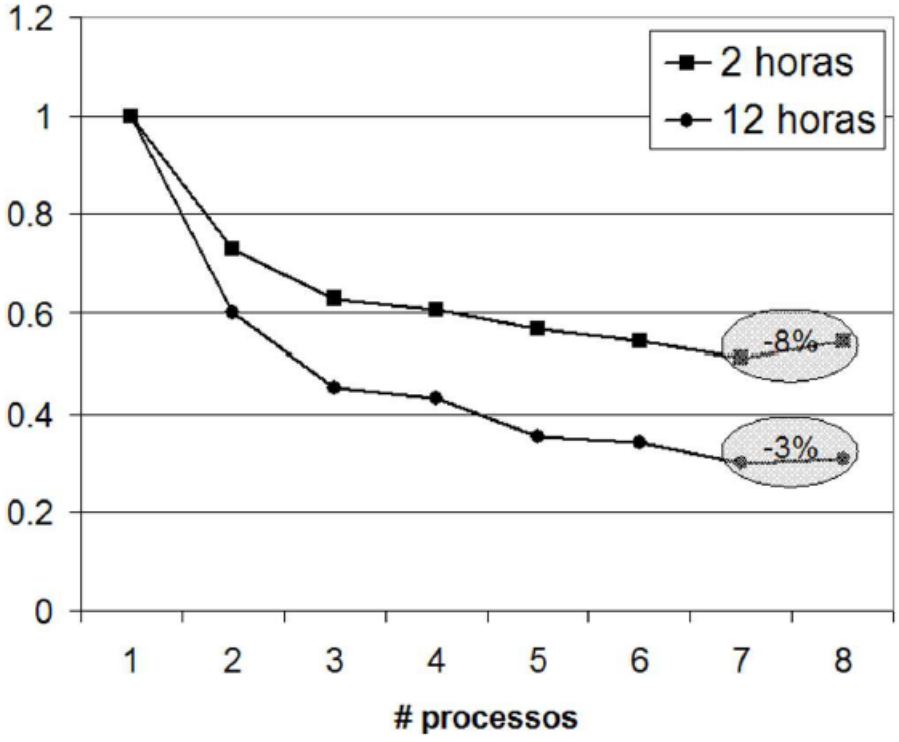


Figura 4.2: Speedup do WRF no ambiente CMP-SMP

Complementando aos experimentos em ambientes de cluster, a Figura 4.4 apresenta os resultados obtidos em uma única máquina multiprocessada, com dois processadores AMD quad-core. Pode-se observar que, embora a interação entre os processos se dê sem necessidade de acesso à rede, o tempo de execução aumenta quando todos os processadores disponíveis são alocados a processos da aplicação. Esta constatação vem para reforçar as indicações dos resultados anteriores no sentido de que a abordagem proposta é realmente pertinente, oferecendo melhor eficiência e desempenho, traduzidos em um menor tempo de execução.

4.5 Conclusão

Como indicação do estudo empírico desenvolvido, foi possível comprovar o sucesso da abordagem proposta para compor clusters eficientes para a execução do modelo numérico WRF de previsão de tempo, oferecendo uma alternativa para ganhar desempenho em clusters interconectados via Gigabit Ethernet sem a necessidade de adquirir, por exemplo, rede de interconexão mais eficiente. Vale ressaltar que a abordagem aproveita-se da tendência atual com relaçãoo a computadores multiprocessados.

Embora apenas 8 núcleos tenham sido alocados para a aplicação, houve ganho de desempenho em relação à configuração de cluster no qual todos os 16 núcleos de processamento disponíveis foram alocados a processos da aplicação. O ganho expressivo no desempenho do modelo WRF, com uma redução de mais de 20% no seu tempo de execução, também foi verificado para o algoritmo de granularidade fina IS do NAS Parallel Benchmark. Por outro lado, a abordagem se mostrou ineficiente para o algoritmo de granularidade grossa EP.

Ademais, reduzir o tempo de execução do modelo numérico torna-se ainda mais importante quando se tem em mente que todo o processo de previsão de tempo, que inclui o modelo WRF (MICHALAKES et al., 2004), é composto de outras etapas que não foram consideradas nesse trabalho. Há, por exemplo, a necessidade de um pós-processamento dos dados gerados pelo modelo para permitir a vizualização, que efetivamente auxilia os meteorologistas na previsão do tempo.

5 *Conclusão*

5.1 Conclusão

Com o estudo feito acerca da computação paralela e de alto desempenho foi possível constatar uma vasta gama de fatores que interferem diretamente no desempenho de um ambiente computacional de alto desempenho. Esses fatores podem ser agrupados em dois grandes grupos: fatores de *Software* e fatores de *Hardware*. Dessa formar, a composição de um ambiente computacional altamente especializado depende da harmonia desses dos grupos dadas as diversas arquiteturas de *hardware* e paradigimas de *software*.

Adiante, é possível afirmar que o trabalho realizado obteve exito ao conseguir compir os objetivos propóstos, de modo que a revisão teórica sobre computação paralela e de alto desempenho forneceu base para o experimento executado tão bem como auxiliou o estudo das redes de interconexão, onde foi possível compeender o impacto que a comunicação entre os processos, utlizando o protocolo TCP/IP, exerce no desempenho do ambiente e por fim estudar tecnologias que procuram minimizar esse impacto, nesse caso a tecnologia VIA.

E com o intuito de utilizar alguma aplicação científica *grand challange* juntamente com a tendência atual de processadores *multi-core*, o experimento realizado obteve exito ao, agregando os conhecimentos gerados anteriormente, prover uma alternativa de ganho de desempenho a um ambiente computacional já instalado, sem que para isso fosse necessário a aquisição de qualque equipamento. Essa afirmação fica evidenciada com a redução em mais de 25% do tempo de execução da aplicação WRF, quando o ambiente foi configurado para não utlizar todos os *cores* disponíveis no sistema para o processamento da aplicação, dessa forma, esses cores ditos ociosos foram capaz de auxiliar o processo de comunicação entre os processos proporcionando assim o ganho de perfomance dito.

5.2 Trabalhos futuros

Como trabalhos futuros pode-se citar:

- Utilizar arquiteturas vanguardistas, por exemplo, *Multi-cluster*, para futuras analises de desempenho do WRF.
- A análise da utlização de protocolos leves, como SCTP ou XTP, ao invés do protocolo TCP/IP, objetivamanto uma melhora de desempenho;
- A utlização de outras aplicações paralelas, a fim de, aprofundar o conhecimento dos fatores de *software* envolvidos na otimização de ambientes computacionais de alto desempenho;
- Extender o experimento para alguma rede de interconexão que implementa o modelo VIA, e inspecionar o desempenho oferecido por essa tecnologia;

Referências Bibliográficas

ALMASI, G. S.; GOTTLIEB, A. *Highly Parallel Computing*. 2^{TM}. ed. [S.l.]: The Benjamin/Cummings Publishing Company Inc., 1994.

AMD. *AMD Opteron(TM) Processor Product Data Sheet*. [S.l.], Publication 23932, 2007.

AMSTRONG, B. et al. Hpc benchmarking and performance evaluation with realistic applications. *SPEC Benchmark Workshop*, 2006.

ARMSTRONG, B. et al. Hpc benchmarking and performance evaluation with realistic applications. 2006.

BAILEY, D. H. et al. The nas parallel benchmarks. *International Journal of Supercomputer Applications*, v. 5, n. 3, p. 63–73, 1991.

BODEN, N. et al. Myrinet: A gigabit-per-second local area network. *IEEE Micro*, v. 15, n. 1, p. 29–36, 1995.

CAPPELLO, F.; ETIEMBLE, D. Mpi versus mpi+openmp on the ibm sp for the nas benchmarks. *Supercomputing*, 2000.

CASSIDAY, D. Infiniband architecture tutorial. *Hot Chips 12*, 2000.

CHAPARRO, P. et al. Understanding the thermal implications of multi-core architectures. *IEEE TPDS*, v. 18, n. 8, p. 1055–1065, 2007.

CLARK, D. D. et al. Ananalysisoftcp processingoverhead. *IEEE Communications Magazine*, 1989.

COMPAQ, C. C.; INTEL, C.; MICROSOFT, C. *Virtual Interface Architecture Specification*. [S.l.], 1997.

DUNNING, D. et al. The virtual interface architecture. *IEEE Micro*, v. 18, n. 2, p. 66–76, 1998.

EICKEN, T. von; VOGELS, W. Evolution of the virtual interface architecture. *IEEE Jornal*, 1998.

FARAJ, A.; YUAN, X. Communication characteristics in the nas parallel benchmarks. *Parallel and Distributed Computing and Systems*, 2002.

FLICH, J. et al. On the potential of noc virtualization for multicore chips. *International Workshop on Multi-Core Computing Systems (MuCoCoS)*, 2008.

FLYNN, M. J. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, v. 21, n. 9, p. 948–960, 1972.

FOSTER, I. *Designing and Building Parallel Programs*. 1TM. ed. [S.l.]: Addison-Wesley Publishing Company Inc., 1995.

GALLATIN, A.; CHASE, J.; YOCUM, K. Trapeze/ip: Tcp/ip at near-gigabit speeds. *USENIX Technical Conference*, 1999.

HENNESSY, J. L.; PATTERSON, D. A. *Computer Architecture - A Quantitative Approach*. 3TM. ed. [S.l.]: Morgan Kaufmann Publishers, 2003.

IBM. September 2008. Disponível em: <http://www.ibm.com/developerworks/linux/library/j-zerocopy/>.

INTEL. february 2007. Disponível em: <http://www.intel.com/pressroom/archive/releases/20070204comp.htm>.

INTELÆ. *IntelÆ XeonÆ Processor with 533 MHz FSB at 2GHz to 3.20GHz Datasheet*. [S.l.], Publication 252135, 2004.

JORDAN, H.; ALAGHBAND, G. *Fundamentals of Parallel Processing*. 1TM. ed. [S.l.]: Prentice Hall, 2003.

KERBYSON, D.; BARKER, K.; DAVIS, K. Analysis of the weather research and forcasting (wrf) model on large-scale systems. 2007.

KIM, J.; LILJA, D. Characterization of communication patterns in message-passing parallel scientific application programs. *Communication, Architecture, and Applications for Network-Based Parallel Computing*, p. 202–216, 1998.

KUMAR, V. et al. *Introduction to Parallel Computing*. 1TM. ed. [S.l.]: The Benjamin/Cummings Publishing Company Inc., 1994.

MICHALAKES, J. et al. Design of a next-generation regional weather research and forecast model. *Towards Teracomputing, World Scientific*, p. 117–124, 1999.

MICHALAKES, J. et al. The weather research and forecast model: Software architecture and performance. *ECMWF Workshop on the Use of High Performance Computing in Meteorology*, p. 156–168, 2004.

PARHAMI, B. *Introduction to Parallel Processing*. 1TM. ed. [S.l.]: Kluwer Academic Publishers, 2002.

PENTAKALOS, O. Via and infiniband: Interconnects for high-performance computing. 2002.

PINTO, L. C. *Estudo de Casos com Aplicações Científicas de Alto Desempenho em Agregados de Computadores Multi-core*. Dissertação (Mestrado) — Universidade Fuderal de Santa Catarina, 2008.

QUINN, M. J. *Parallel Computing*. 2TM. ed. [S.l.]: McGraw-Hill Inc., 1994.

RABENSEIFNER, R.; WELLEIN, G. Communication and optimization aspects of parallel programming models on hybrid architectures. *International Journal of High Performance Computing Applications*, v. 17, n. 1, p. 49–62, 2003.

RFC793. *Transmission Control Protocol*. [S.l.], 1981.

RISCA, M.; MALIK, D.; KESSLER, A. *Trading Floor Architecture*. 2008. Disponível em: <http://www.cisco.com/en/US/docs/solutions/Verticals/Trading_Floor_Architecture-E.html>.

SKAMAROCK, W. et al. *A Description of the Advanced Research WRF Version 3*. [S.l.], June 2008.

SOKOLOWSKI, P. J. S. P. J. S. P. J. S. P. J.; GROSU, D. Performance of the nas parallel benchmarks on grid enabled clusters. *Third IEEE International Symposium on Network Computing and Applications*, 2004.

SUBHLOK, J.; VENKATARAMAIAH, S.; SINGH, A. Characterizing nas benchmark performance on shared heterogeneous networks. *IEEE International Parallel and Distributed Processing Symposium*, p. 86–94, 2002.

SUN, Y.; WANG, J.; XU, Z. Architetural implications of the nas mg and ft parallel benchmarks. *Advances in Parallel and Distributed Computing*, p. 235–240, 1997.

TABE, T.; STOUT, Q. *The use of MPI communication library in the NAS parallel benchmarks*. [S.l.], 1999. Technical Report CSE-TR-386-99, University of Michigan, 1999.

TANEMBAUM, A. *Redes de Computadores*. 3^{TM}. ed. [S.l.]: Pearson, 1997.

YU, J.-L.; LEE, M.-S.; MAENG, S.-R. An efficient implementation of virtual interface architecture using adaptive transfer mechanism on myrinet. 2001.

ZAMANI, R.; AFSAHI, A. Communication characteristics of message-passing scientific and engineering applications. *International Conference on Parallel and Distributed Computing and Systems (PDCS)*, p. 644–649, 2005.

APÊNDICE A – Trabalhos Publicados

- **Uma Abordagem para Composição de Clusters Eficientes na Execução do Modelo Numérico WRF de Previsão do Tempo**
  Autores: Luiz Carlos Pinto, Luiz H. B. Tomazella, M. A. R. Dantas
  Evento: IX Workshop em Sistemas Computacionais de Alto Desempenho - Simpósio em Sistemas Computacionais (WSCAD-SSC)
  Local: Campo Grande, MS, Brasil
  Ano: 2008
  CAPES Qualis CC: Nacional C

- **An Experimental Study on How to Build Efficient Multi-Core Clusters for High Performance Computing**
  Autores: Luiz Carlos Pinto, Luiz H. B. Tomazella, M. A. R. Dantas
  Evento: IEEE 11th International Conference on Computational Science and Engineering (CSE)
  Local: São Paulo, SP, Brasil
  Ano: 2008
  CAPES Qualis CC: Não avaliado
  BEST PAPER AWARD: to appear in special issue of the International Journal of Computational Science and Engineering (IJCSE) and International Journal of High Performance Computing and Networking (IJHPCN)

- **Building Effiient Multi-Core Clusters for High Performance Computing**
  Autores: Luiz Carlos Pinto, Luiz H. B. Tomazella, M. A. R. Dantas
  Evento: IEEE 13th Symposium on Computers and Communications (ISCC)
  Local: Marrakech, Marrocos
  Ano: 2008
  CAPES Qualis CC: Internacional A