Aplicação das Políticas de Balanceamento de Carga do *Tree Load Balancing Algorithm* em Simulações

Evandro Raphaloski[1], Maria Stela Veludo de Paiva[2]

[1]Divisão de Desenvolvimento Eletrônico – SMAR Equipamentos Industriais Ltda
Rua Dr. Antônio Furlan Jr., 1028 – 14170-480 – Sertãozinho – SP – Brasil

[2]Escola de Engenharia de São Carlos – Universidade de São Paulo (EESC-USP)
Av. Trabalhador Sãocarlense, 400 – 13566-590 – São Carlos – SP – Brasil

[1]evandro@smar.com.br, [2]mstela@sel.eesc.usp.br

Resumo. Os algoritmos de balanceamento de carga são aplicados em ambientes homogêneos, heterogêneos, distribuídos localmente e *grids*, visando um melhor desempenho do sistema, a fim de distribuir de forma eqüitativa a carga de processos entre todos os computadores envolvidos na rede. Para que isto seja possível esses algoritmos são dotados de políticas de balanceamento de carga que gerenciam e coordenam desde a localização dos computadores mais ociosos até a transferência do processo propriamente dito. Para melhorar o desempenho de um sistema, baseado na aplicação de políticas de balanceamento de carga, recentemente foi proposto um algoritmo de balanceamento de carga denominado *Tree Load Balancing Algorithm*. Para avaliá-lo foram desenvolvidos um simulador e um protótipo que validaram seus resultados. Este trabalho apresenta a implementação de um novo simulador, que permite o estudo e a aplicação das políticas de balanceamento de carga deste algoritmo, em busca de um melhor desempenho de sistemas distribuídos.
Palavras-chave: Políticas de Balanceamento de Carga, Alto Desempenho e Simuladores.

Application of the Tree Load Balancing Algorithm Policies in Simulations

Abstract. Load balancing algorithms are used on homogeneous, heterogeneous systems, locally distributed and grids to improve de performance of the systems, sharing the processes load between the computers on a network. These algorithms are adopted of load balancing policies that manage and coordinate the resources on network. To improve the performance of a system, based on the application load balancing polices, a new load balancing algorithm called Tree Load Balancing Algorithm has been recently proposed. The evaluation of this algorithm was made by a simulator and a prototype which were developed for this purpose. This work shows the implementation of a new simulator which allows studies and application of the TLBA load balancing policies, in order to improve the performance of distributed systems .
Keywords: Load Balancing Policies, High Performance and Simulators.

(Received November 29, 2005 / Accepted April 06, 2006)

1. Introdução

Algoritmos de balanceamento de carga buscam atingir um conjunto de objetivos relacionados com medidas de desempenho, como por exemplo: otimizar a ocupação de recursos e diminuir o tempo médio de resposta dos processos [7].

Suas aplicações variam desde sistemas localmente distribuídos a *grids* para ambientes heterogêneos ou homogêneos. A heterogeneidade dos recursos existentes nas diversas máquinas de um sistema computacional leva à possibilidade de se ter uma grande variação em suas potências computacionais. Essa variação, aliada à possibilidade de utilização de diferentes arquiteturas introduz dificuldades quer precisam ser gerenciadas quando se tem o balanceamento de cargas, visando a obtenção de um melhor desempenho [1].

Esse gerenciamento de recursos de uma determinada rede de computador é possível através da implementação de mecanismos e políticas de balanceamento de cargas [8][9][10].

Essas políticas são classificadas da seguinte forma:

1) **Política de Transferência**: determina se um computador é capaz de participar do processo de transferência de tarefas. Ela identifica computadores emissores e receptores de processos, transferindo-os em seguida. São normalmente consideradas políticas de *Thresholds* [3].
2) **Política de Seleção**: viabilizada a transferência de um processo, ela tem como objetivo selecionar o processo com as características de maior ocupação da CPU (*Central Unit Processing*) e longo tempo de execução, que ocasionará a mínima sobrecarga no sistema em virtude da transferência e com poucas dependências e chamadas relacionadas ao sistema local [9];
3) **Política de Localização**: visa encontrar o melhor computador receptor de processos para o melhor emissor e vice-versa. O melhor emissor e o melhor receptor são, respectivamente, o computador mais sobrecarregado e mais ocioso do sistema [5][6][9];
4) **Política de Informação**: define quando, de onde e que tipo de informações relacionadas aos demais computadores da rede devem ser coletadas. Essas informações são utilizadas para o cálculo do índice de carga, que define a ocupação de cada processador ou computador no ambiente[5][9]. Existem três tipos de políticas de informação:
   a) *Política Orientada à demanda*: uma máquina coleta o estado das outras máquinas somente quando ela se torna emissora ou receptora;
   b) *Política Periódica*: as informações são coletadas de tempos em tempos;
   c) *Política Orientada à Mudança de Estado*: as informações das máquinas são divulgadas conforme muda o grau de seu estado.

Já os mecanismos empregados no balanceamento de cargas, por outro lado, podem ser divididos em:

1) **Mecanismo de métrica da carga**: define o método utilizado para medir a carga dos computadores;
2) **Mecanismo de comunicação da carga**: define o método que será efetuado a comunicação das informações de carga entre os computadores; e
3) **Mecanismo de migração**: define o protocolo a ser utilizado quando da ocorrência de migração de processos entre os computadores.

Políticas consideradas como viáveis para utilização são aquelas que proporcionam um melhor desempenho das aplicações e não influenciam ou pouco influenciam na carga do sistema.

Já bons algoritmos são aqueles que utilizam poucos recursos computacionais do ambiente e mesmo nestas condições conseguem alocar recursos de forma otimizada e homogênea oferecendo, conseqüentemente, um alto desempenho para as aplicações.

Este trabalho é composto pela Seção 2 com uma breve explicação do TLBA e suas políticas de balanceamento; na Seção 3 são apresentados resultados para determinados casos de estudo; a Seção 4 contém as conclusões; a Seção 5 apresenta os agradecimentos dos autores; a Seção 6 as referências.

2. *Tree Load Balancing Algorithm*

Proposto por [5] o TLBA (*Tree Load Balancing Algorithm*) é um algoritmo de balanceamento de cargas para ambientes distribuídos heterogêneos escaláveis, projetado para executar de forma modular sobre cada um dos computadores do sistema. A origem do nome deve-se devido à criação da topologia lógica no formato de árvore que interconecta os computadores do sistema . Para a aplicação desse algoritmo dois componentes são fundamentais, o *Broker* e o *Peer*, e são descritos a seguir [5].

1) ***Broker***: é o componente responsável pela montagem de toda a árvore de computadores interconectados. O *Broker* define em que nível da árvore um computador será inserido, qual seu antecessor e quais seus sucessores. Essa inserção é dada de tal forma a manter a árvore balanceada. Para isso, um *benchmark* é executado pelo computador durante a etapa que requer sua participação no sistema. O *Broker* contém toda a estrutura hierárquica dessa árvore lógica e pode ser replicado em outros computadores para garantir alta disponibilidade [6].
2) ***Peer***: esse componente é responsável por requisitar a adição de um computador na árvore e implementa as políticas de balanceamento de carga do algoritmo TLBA. O *Peer*, periodicamente, analisa a ocupação do recurso de CPU e memória principal e em seguida calcula o índice de carga. Esse evento periódico tem intervalos definidos pelo administrador do sistema.

O novo índice de carga proposto por [5], baseia-se na ocupação percentual da CPU e da memória. Esse índice contribui para o aumento de desempenho e estabilidade no balanceamento de carga, comprovado através de sua aplicação no *Lowest Modificado* [7] e comparado a Algoritmos Simetricamente Iniciados [9].

A política de informação implementada pelo TLBA utiliza-se desse índice para determinar os computadores sobrecarregados e ociosos no sistema de forma a distribuir eqüitativamente a carga dos computadores do sistema.

A Figura 1 indica que os índices de carga calculados por um computador α localizado no último nível ***Nk*** de uma árvore, são propagados para seu antecessor β localizado no nível ***Nk-1***. O computador β armazena em memória seu índice de carga e de seus sucessores. O computador β, devido ao fato de apresentar sucessores, realiza um somatório de seu índice e dos sucessores, submetendo o resultado ao seu antecessor na árvore. Essa operação é propagada até chegar à raiz da árvore, atualizando as informações do sistema [5].

A política de informação do TLBA utiliza-se de índices de cargas calculados em cada computador do sistema e que são propagados para seus respectivos antecessores até chegar no nível do computador raiz. Para evitar que essa transferência de informações sobrecarregue o sistema, a propagação de dados é feita se houver mudanças significativas na ocupação dos recursos. Essas mudanças são baseadas em um parâmetro denominado *Threshold* de Atualização de Cargas que é adicionado ou removido dos índices de cargas e cuja variação superior ou inferior definirá se é necessária a propagação dos índices de carga para os antecessores.

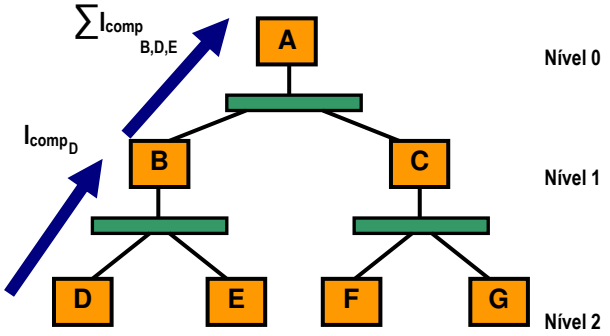

Figura 1: Política de Informação do TLBA.

A segunda política do TLBA é a política de localização que visa encontrar somente por computadores receptores. Essa realiza uma busca por computadores ociosos da forma *top-down*, iniciando-se pelo computador raiz e propagando-se até o último nível da árvore computacional (Figura 2).

A política de localização do algoritmo TLBA privilegia localizar computadores mais próximos das "folhas", ou seja, computadores presentes no último nível ***Nr*** da árvore. Assim sendo, para casos onde o sistema apresenta baixa carga, as mensagens de busca em profundidade tendem a percorrer todos os níveis e localizar computadores receptores próximos do nível *Nr*. Já em altas cargas, os receptores de processos são localizados mais próximos do primeiro nível da árvore, portanto, nesses casos, há menor consumo de recursos do ambiente [5].

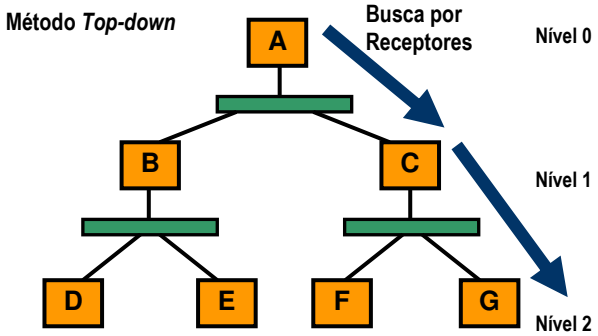

Figura 2: Política de Localização do TLBA.

Uma vez localizado o melhor receptor de processos, seu endereço é enviado ao elemento responsável pela ativação da política de transferência, que pode ser o *broker* ou um computador sobrecarregado, conforme os casos a seguir.

- **é enviado para o *Broker***, caso esse tenha recebido uma requisição para executar um novo processo no ambiente e tenha ativado a política de localização; e
- **é enviado para o computador sobrecarregado**, caso esse tenha ativado a política de localização para transferir parte de sua carga.

A política de seleção de um algoritmo de balanceamento de carga é responsável por selecionar processos para transferência. No algoritmo TLBA essa política busca pelo processo que ocupa maior quantidade dos recursos de CPU e memória em um computador sobrecarregado α.

Uma vez localizado o processo de maior ocupação, seu identificador é submetido para a política de transferência que cuida das etapas de salvar seu contexto, transferi-lo para seu destino e reiniciá-lo.

Após o recebimento do endereço do receptor, o novo processo é transferido. Esse contexto de transferência de processos envolve a localização do computador mais

ocioso, seleção de processo e uma etapa denominada *checkpointing* [12].

O *checkpointing* salva o contexto de um determinado processo, para que seja reiniciado em um outro computador [2]. A última etapa é a da transferência propriamente dita.

No entanto, todas as etapas geram um grande custo computacional e são realizadas nos computadores que apresentarem processos ocupando a maior quantidade de CPU e memória.

A validação desse algoritmo foi realizada através de um simulador e um protótipo e seus resultados um baixo número de mensagens geradas em operações de balanceamento de carga, maior estabilidade em altas cargas e baixos tempos médios de resposta em comparação ao algoritmo *Lowest* modificado, exceto em sistemas de baixas cargas para os tempos de resposta médio e mediano.

3. Resultados e Simulações

Como resultados finais deste trabalho teve-se a implementação de um novo simulador abrangendo um número maior de parâmetros de balanceamento de carga para o estudo do TLBA e a análise da variação, propriamente dita, desses parâmetros aplicada em diferentes casos de estudo.

3.1 Novo Simulador e Seus Recursos

Ao simulador apresentado nesse trabalho foi dado o nome de TLBASim (*Tree Load Balancing Algorithm Simulator*) e foi desenvolvido em C#, o que possibilitou a implementação dos diversos recursos aqui apresentando devido a praticidade e facilidade de utilização da linguagem.

A sua interface é composta de uma tela principal, onde são visualizados os principais resultados de cada simulação e possibilita a elaboração de gráficos, montagem em tempo real da árvore computacional e, ainda, uma janela para edição de anotações. Além disso, existe uma outra janela disposta ao lado da tela principal que contém os parâmetros de configuração das simulações (Figura 3).

Para melhor separação desses parâmetros, eles foram divididos em quatro classes descritas a seguir.

- **Classe Rede (*Networks*):** apresenta parâmetros de uma rede distribuída e interconectada e da árvore a ser elaborada pelo TLBA no processo de simulação. Dentre esses parâmetros destacam-se os tipos de computadores presentes na rede, suas capacidades computacionais e memória, características do *switch* de rede, número de níveis e número de computadores da árvore computacional;
- **Classe Estatístisca (*Statistical*):** apresenta os parâmetros relacionados aos dados estatísticos a serem considerados como, por exemplo, o intervalo de confiança ser utilizado.
- **Classe Sistêmica (*Systemical*):** apresenta parâmetros mais gerais do sistema referentes ao número de aplicações a serem considerados no sistema, custos percentuais de cada migração, entre outros.
- **Classe Balanceamento (*Balancing*):** aqui estão presentes os parâmetros relacionados aos algoritmos de balanceamento de carga como, por exemplo, algoritmo de balanceamento de carga, número máximo de migrações permitidas, *Thresholds* etc.

3.2 Validação e Casos Estudados

Já com o simulador pronto e analisado a coerência de seus resultados iniciais, os passos seguintes foram o de validação do simulador através da comparação dos resultados obtidos do novo simulador (TLBASim) com os do antigo simulador (TLBA V1.0) o qual foi validado através de um protótipo implementado por [5]. Por esse motivos levou-se em consideração a validação do TLBASim através do simulador TLBA V1.0, implementado por [5].

As simulações apresentadas neste trabalho foram geradas com intervalos de confiança de 95%, excetuando-se, as simulações para comparação com o simulador TLBA V1.0 de forma a manter o código original desse simulador e que não continha tal recurso.

3.2.1 Análise entre os Simuladores em Função da Variação do Número de Processos Submetidos ao Sistema (Validação)

No gráfico da Figura 4 são apresentados os tempos médios de resposta obtidos para esse caso de estudo. O TLBA V1.0 e TLBASim(1) apresentam os resultados do antigo e novo simulador e levam em consideração apenas as mensagens de balanceamento do algoritmo para cada nível dos processos submetidos ao sistema. Já o TLBASim(2) considera o número total de mensagens geradas no sistema. Essa última análise só é possível porque o novo simulador faz a análise de mensagens adicionais que estão envolvidas nas políticas de balanceamento do TLBA.

Essas mensagens adicionais são geradas quando um computador sobrecarregado solicita ao computador raiz, informações do sistema para transferência de seus processos, mensagens de atualização de carga,

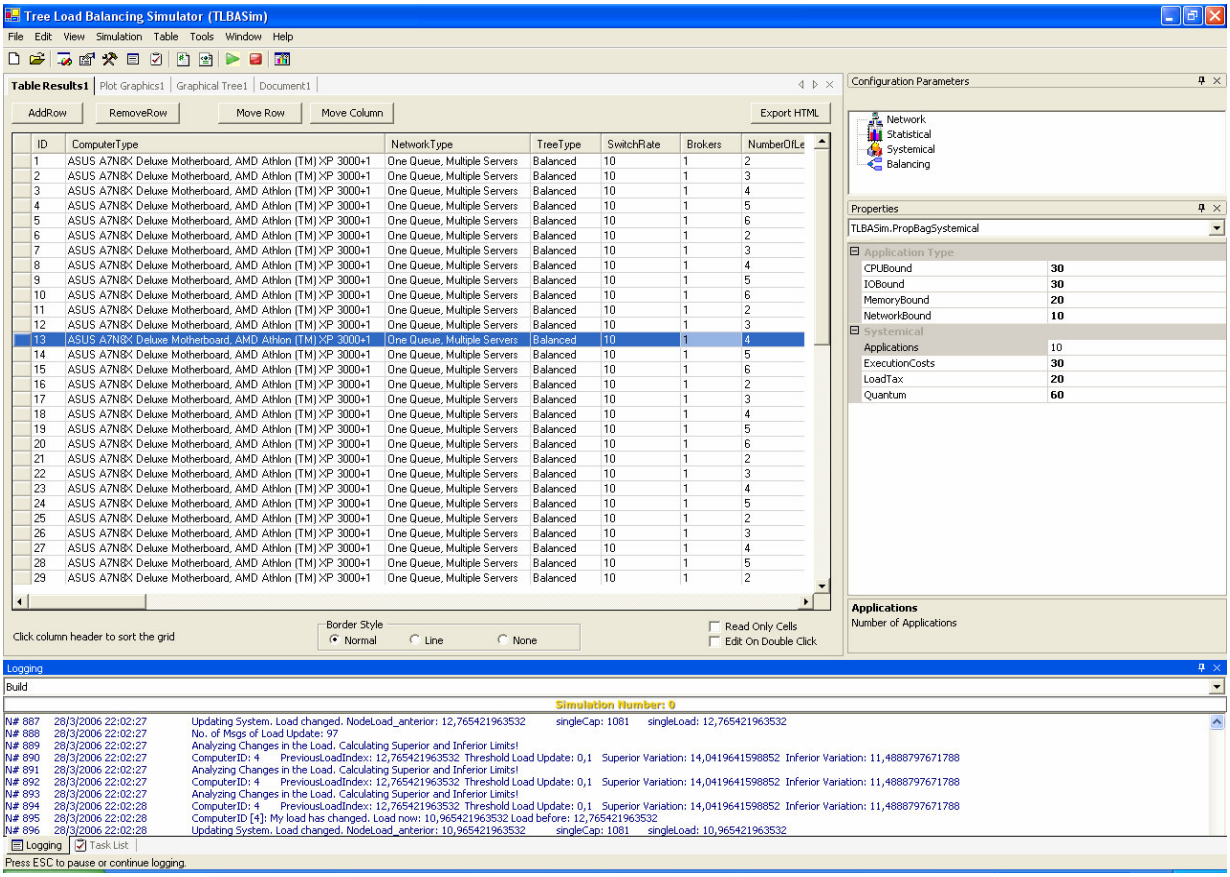

Figura 3: Tela Principal do Simulador.

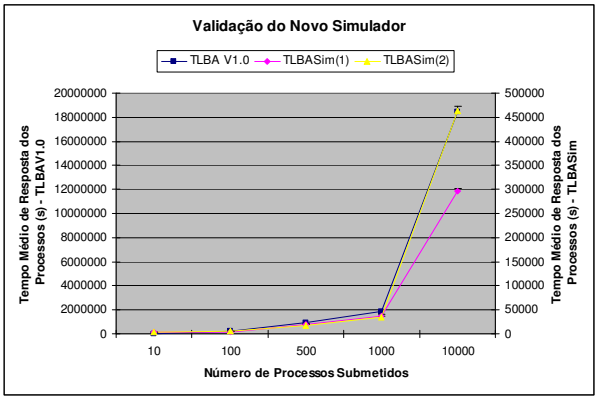

Figura 4: Tempo médio de resposta (s) dos processos em função do número de processos enviados

mensagens de balanceamento de carga (encontrar o computador mais ocioso e envio do processo), mensagem requisitando execução de processos no sistema e mensagens para inserir um novo computador na árvore.

A divergência entre os valores apresentados do novo simulador e do simulador de [5] deve-se ao fato de que esse último considera fixa a ocupação percentual de cada processo no sistema (que é da ordem de no mínimo 10^7) o que nem sempre representa o comportamento dos sistemas reais e eleva, então, o valor final do tempo de resposta de cada ciclo de execução.

Já o novo simulador assume valores de tempos de chegadas de processos de acordo com o modelo de [4], capacidades computacionais obtidas de especificações de sistemas reais e sua ocupação percentual varia para cada computador.

Pelo gráfico pode-se notar uma grande variação do tempo médio de resposta entre 1000 e 10000 processos. No entanto, se for calculada a variação entre os tempos médios de resposta em função da variação do número de processos submetidos, pode-se notar que o aumento do tempo final de execução dos processos é gradual e seu comportamento é equivalente ao do antigo simulador, mesmo para sistema com um elevado número de computadores.

3.2.2 Análise da Variação do Número de Computadores por Nível e de Níveis da Árvore Computacional (Validação)

Nos gráficos das Figuras 5 e 6 é apresentado o número de mensagens por nível geradas em função da variação

do número de níveis e de computadores por nível para o TLBA V1.0 e TLBASim,, respectivamente. Esses resultados confirmam os apresentados por [5], onde árvores com muitos níveis e, conseqüentemente, poucos computadores por nível, geram uma maior sobrecarga nas buscas em profundidade utilizadas para localizar os computadores ociosos e árvores com poucos níveis geram maior número de mensagens entre níveis e diminuem a escalabilidade do ambiente.

Nas simulações do TLBA em dois níveis, o algoritmo tem estrutura semelhante ao Central [11]. Essa estrutura apresenta baixa escalabilidade para o sistema, pois não há suportes físicos ilimitados para interconectar computadores em uma mesma rede fisicamente única. Outra limitação técnica é que, quanto maior o número de computadores, maior a cargas do computador central [5][9][11].

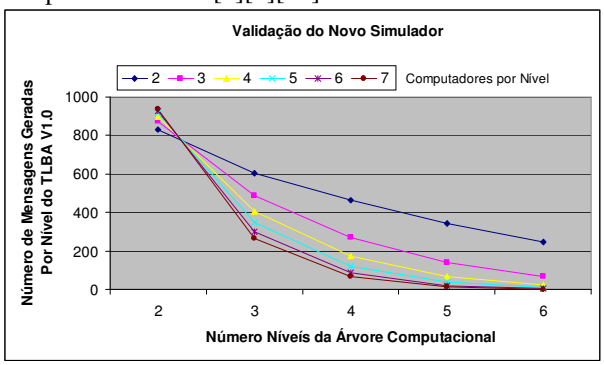

Figura 5: Número de mensagens por nível geradas em função da variação do número de níveis e computadores por nível para o TLBA V1.0

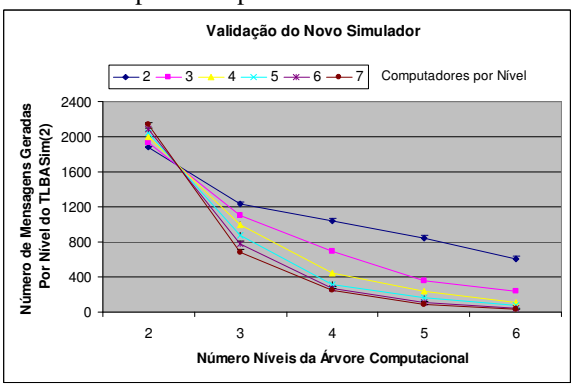

Figura 6: Número de mensagens por nível geradas em função da variação do número de níveis e computadores por nível para o TLBASim[2]

Uma das características do TLBA é sua escalabilidade, conseqüência da hierarquia em forma de árvore, que cria grupos de comunicação entre um antecessor e seus sucessores [5].

3.2.3 Análise da Variação dos *Thresholds* de Viabilidade de Transferência e Atualização de Carga

Como pode ser notada através da Tabela 1, a variação do *Thresholds* relacionada ao TLBA, não trouxeram impactos significativos no tempo médio de resposta para cada simulação.

Com relação as mensagens geradas no sistema as que possuem relação direta com os *Thresholds* de viabilidade de transferência e de atualização de carga são as mensagens de balanceamento de carga, computadores sobrecarregados (que também podem ser consideradas como mensagens geradas devido ao balanceamento de cargas) e as de atualização de carga.

Desta forma, pode-se notar que a variação desses *Thresholds* não proporcionou uma melhora significativa tanto no tempo médio de resposta quanto na redução do número de mensagens de balanceamento de cargas geradas pelo TLBA.

Variação Percentual	TMR para Variação *Threshold* de Transferência	TMR para Variação *Threshold* de Atualização
5	3442,55 ± 87,77	3412,72 ± 86,97
10	3634,56 ± 92,62	3580,83 ± 91,30
15	3609,72 ± 92,04	3552,74 ± 90,59
25	3545,19 ± 90,39	3754,04 ± 95,67
45	3559,99 ± 90,79	3539,05 ± 90,24
55	3459,36 ± 88,20	3368,81 ± 85,92
65	3662,47 ± 93,41	3518,60 ± 89,72
75	3440,59 ± 87,75	3378,56 ± 86,15
85	3397,59 ± 86,57	3465,48 ± 88,37
95	3515,77 ± 89,63	3473,53 ± 88,58

Tabela 1 – Tempo Médio de Resposta (TMR, s) em função da variação dos *Thresholds* de Viabilidade de Transferência e Atualização de Carga.

3.2.4 Análise da Variação de Computadores no Sistema

Essas simulações foram realizadas em função da variação do número de computadores da árvore computacional e, também, da variação do número máximo de computadores por nível.

Pela Figura 7 é possível notar que o número de mensagens geradas por nível no sistema tende a diminuir à medida que novos computadores são adicionados ao sistema. Esse resultado comprova novamente que árvore com muitos níveis e poucos computadores por nível, geram sobrecarga nas buscas em profundidade utilizadas para localizar os computadores ociosos [5].

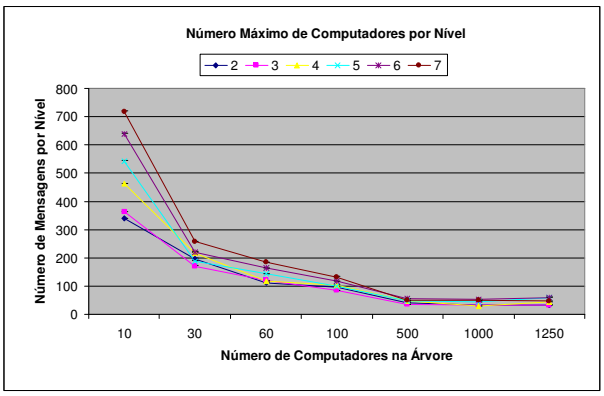

Figura 7: Número de mensagens por nível geradas em função da variação do número máximo de computadores por nível e da árvore computacional.

Na Figura 8 é apresentado o número de mensagens geradas por computadores sobrecarregados tende a aumentar à medida que se aumenta o número de computadores na árvore lógica e quanto menor for o número de computadores por nível maior será o número de mensagens de computadores sobrecarregados que buscarão compartilhar sua carga com os outros computadores do sistema.

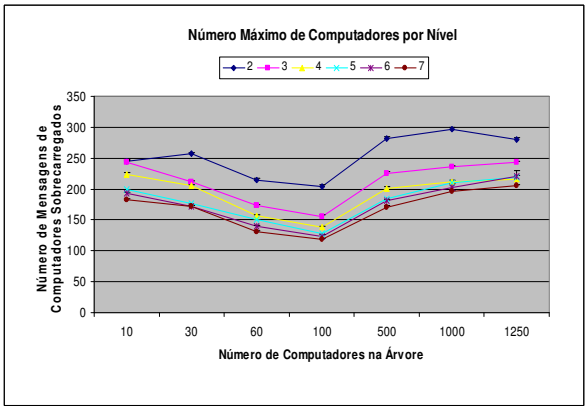

Figura 8: Número de mensagens de computadores sobrecarregados em função da variação do número máximo de computadores por nível e na árvore

Com esses resultados também é possível de se observar que quando o número de processos no sistema está próximo ao número de computadores na árvore lógica há uma diminuição do tráfego de mensagens de computadores sobrecarregados. Assim sendo, o sistema apresenta um melhor desempenho se o número de processos for equivalente ao número de computadores, ou melhor, se todos os computadores possuírem uma carga igualmente distribuída ou balanceada.

Na Figura 9 é apresentado o tempo médio de resposta para um sistema onde se tem um aumento do número de computadores por nível e no número de computadores da árvore lógica. Através desse gráfico é possível concluir que o TLBA não é recomendado para sistemas de pequeno porte, ou seja, redes formadas por poucos computadores ou que tenham uma grande concentração de computadores por nível.

A aplicação do TLBA para esse tipo de sistema irá piorar o seu desempenho aumentando o número de mensagens de balanceamento de carga e terá um aumento no tempo médio de resposta dos processos.

Pelas Figuras 10 e 11, pode-se notar uma estabilização do número de mensagens de atualização e de balanceamento de cargas a partir do momento em que o número de computadores é maior que o número de processos submetidos ao sistema.

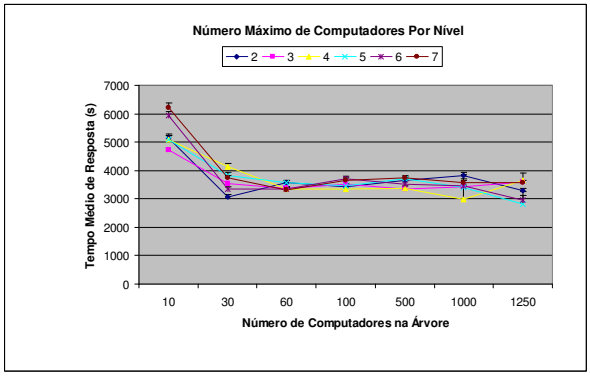

Figura 9: Tempo médio de resposta (ms) em função da variação do número máximo de computadores por nível e na árvore.

Os comportamentos apresentados nos gráficos das Figuras 10 e 11 são em virtude de que a partir do momento que o número de computadores supera o número de processos presentes no sistema não há mais necessidade de migração de processos e, por conseguinte, não há mais necessidade de balanceamento de carga no sistema.

Por outro lado, é por esse motivo também que se deve fixar o número de migração de processos entre os computadores, pois neste caso novas migrações somente viriam a prejudicar o sistema e não agregariam benefícios de desempenho em sua execução. Além disso, sistemas desse tipo não apresentam uma sobrecarga significante para que as políticas de

balanceamento atuem de forma a melhor distribuir os recursos entre os computadores.

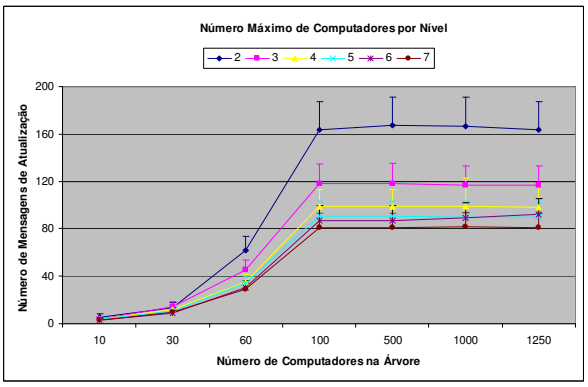

Figura 10: Número de mensagens de atualização geradas em função da variação do número máximo de computadores por nível e da árvore.

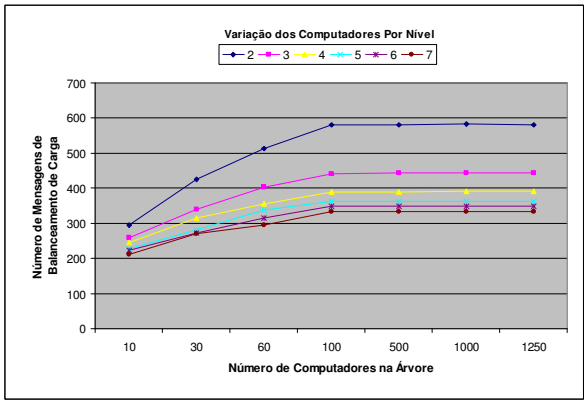

Figura 11: Número de mensagens de balanceamento de carga em função da variação do número máximo de computadores por nível e da árvore.

3.2.5 Análise de Sistemas Homogêneos e Heterogêneos

As simulações dessa seção permitem analisar os impactos apresentados a um sistema composto por computadores homogêneos e heterogêneos.

Para sistemas homogêneos foi adotada a configuração de um computador **ASUS A7N8X Deluxe Motherboard, AMD Athlon (TM) XP 3000+1 da Advanced Micro Devices**, com aproximadamente 960 MHz e memória de 1024 Mbytes.

Já para os sistemas heterogêneos a capacidade computacional desses computadores variou aleatoriamente em função do número de computadores.

Atualmente, o simulador possui 52 computadores disponíveis para simulação e apresentam em 80% dos casos capacidades computacionais e memórias diferentes um dos outros. No entanto, o usuário está livre para adicionar qualquer nova configuração que se faça necessário, pois existe uma ferramenta no simulador que auxilia na adição de novos computadores e, conseqüentemente, novas configurações.

A Tabela 2 apresenta os resultados dos tempos médios de resposta para simulações em ambientes homogêneos e heterogêneos.

Pode-se notar que com o aumento do número de computadores no sistema existe uma melhora no tempo médio de resposta para ambos ambientes. Essa é uma das características importantes do TLBA que é recomendado para ambientes com um elevado número de computadores.

Nº Comp.	AMBIENTES HOMOGÊNEOS		
	Tempo Médio de Resposta (s)	Variância (s)	Desvio Padrão (s)
10	7543,02	67593419,38	192,35
100	2919,33	3307481,91	74,43
500	2779,10	2653977,81	70,83
1000	2778,10	2853977,81	71,13
5000	2775,27	2744443,73	72,76

(a)

Nº Comp.	AMBIENTES HETEROGÊNEOS		
	Tempo Médio de Resposta (s)	Variância (s)	Desvio Padrão (s)
10	7732,20	78047973,66	197,20
100	3440,82	7280377,93	87,77
500	3495,75	4285556,38	89,08
1000	3538,24	4372803,35	90,16
5000	3586,27	4412478,38	79,31

(b)

Tabela 2 – Tempo médio de resposta , variância e desvio padrão em (s) para ambientes homogêneos (a) e heterogêneos (b).

4. Conclusões

O processo de simulação foi fundamental para a obtenção dos resultados, otimizando o tempo de execução de todo o processo e flexibilizando a variação dos parâmetros do projeto.

Uma vez que a simulação é extremamente útil para uma primeira visão do sistema foi implementado um simulador para análise do TLBA (TLBASim).

O TLBASim apresentou uma interface gráfica coerente com as atuais práticas de desenvolvimento, além de diversos recursos adicionais para a introdução de conceitos básicos na área de balanceamento de cargas e sistemas distribuídos, tornando-o uma ferramenta adequada para fins didáticos.

Seus recursos de tabela e gráficos facilitam a interpretação de simulações e análise de resultados. O recurso de diagramação da árvore lógica auxilia a análise do algoritmo e a verificação da montagem da árvore computacional em tempo real de simulação.

O TLBASim comparado com o antigo simulador fornece uma diversidade de recursos adicionais, além de estar fundamentado em bases estatísticas mais sólidas e ter incorporado novas teorias e estudos apresentados na literatura. Graças a ele é possível o estudo de todos parâmetros do TLBA de acordo com suas políticas de balanceamento de carga.

Através das análises desses parâmetros, verificou-se que a alteração de alguns parâmetros não traz benefícios significativos ao sistema (alteração dos *Thresholds* e número de migrações), enquanto que em outros (número de níveis de computadores, número de computadores por nível, número de processos no sistema), influenciam diretamente no número de mensagens geradas e, conseqüentemente, no tempo de resposta final da execução dos processos no sistema.

Para os *Thresholds* foi possível verificar que eles pouco influenciam no tempo final de resposta dos processos e no número de mensagens geradas no sistema.

Para sistemas com um pequeno número de computadores na árvore lógica e com poucos níveis apresentaram um elevado número de mensagens e um aumento no tempo médio de resposta quando da aplicação do TLBA.

Os resultados confirmam a aplicação do TLBA para grandes sistemas homogêneos ou heterogêneos, uma vez que melhorou o tempo médio de resposta se comparado a sistemas sem migração.

Este estudo permitiu a análise do TLBA para um conjunto de parâmetros de balanceamento de carga e para sistemas homogêneos e heterogêneos e contribui de forma a priorizar os parâmetros a serem considerados para um melhor desempenho do TLBA para sistema distribuídos.

5. Agradecimentos

Os autores agradecem ao Prof.Dr. Rodrigo Fernandes de Mello do Instituto de Ciências da Computação e Matemática da Universidade de São Paulo (ICMC-USP), pelas explicações sobre o funcionamento do TLBA, compartilhamento do código fonte do TLBA V1.0, reuniões, dicas e comentários iniciais que contribuíram para a evolução do simulador; e ao Prof.Dr. Luis Carlos Trevelin do Departamento de Ciências da Computação da Universidade Federal de São Carlos (UFSCAR), pelos valiosos comentários e sugestões sobre o trabalho e o simulador.

6. Referências

[1] Branco, K.R.L.J. *Índices de carga e desempenho em ambientes paralelos/distribuídos – modelagem e métricas*. 260p. Tese (Doutorado em Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2002.

[2] Chanchio, K.; Sun, X.H. *A protocol design of comunication state transfer for distributed computing*. New York: IEEE Computer Society Press, p.715-718, 2001.

[3] Eager, D.L.; Lazowska, E.D.; Zahorjan, J. A comparison of receiver-initiated and sender-initiated adaptative load sharing. *Performance Evaluation*, v.6(1), p.53-68, 1986.

[4] Feitelson, D.G. *Packing schemes for gang scheduling*, 1996. Disponível em: <http://www.cs.huji.ac.il/labs/parallel/workload/models.html#feitelson96>. Acesso em: 20 mar. 2006.

[5] Mello, R.F. *Proposta e avaliação de desempenho de um algoritmo de balanceamento de carga para ambientes distribuídos heterogêneos escaláveis***.** 130p. Tese (Doutorado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2003.

[6] Mello, R.F.; Trevelin, L.C.; Paiva, M.S. *Comparative analysis of the prototype and the simulator of a new load balancing algorithm for heterogeneous computing environments.* Aceito para publicação no International Journal of High Performance Computing and Network, 2003. (No prelo).

[7] Mello, R.F.; Trevelin, L.C.; Paiva, M.S. *Comparative study of the server-initiated lowest algorithm using a load balancing index based on*

the process behavior for heterogeneous environment. Aceito para publicação no Journal of Networks, Software Tools and Application, 2003. (No prelo).

[8] Shirazi, B.A.; Hurson, A.R.; Kavi, K.M. *Scheduling and load balancing in parallel and distributed systems.* Los Alamitos: IEEE Computer Society, 1995.

[9] Shivaratri, N. G.; Krueger, P.; Singhal, M. Load distributing for locally distributed systems. *IEEE Computer*, v.25(12), p.33-44, Dec., 1992.

[10] Song, J.; Choo, H.K.; Lee, K.M. Application-level load migration and its implementation on top of PVM. *Concurrency: Practice and Experience*, v. 9(1), p.1-19, Jan., 1997

[11] Zhou, S.; Ferrari, D. *An Experimental Study of Load Balancing Performance,* Universidade da California/Computer Science Division (EECS), 1987. (Report No. UCB/CSD 87/336, PROGRES Report No. 86.8)

[12] *Website do Ckpt*. Disponível em: <http://www.cs.wisc.edu/~zandy/ckpt>. Acesso em: 17 abr. 2004.