SMARTLB: Proposta de um balanceador de carga para redução de tempo de execução de aplicações em sistemas paralelos *

Vinicius R. S. dos Santos[1], **Edson L. Padoin**[1,2]

[1]Universidade Reg. do Noroeste do Estado do Rio G. do Sul (UNIJUI) - Ijuí - RS - Brasil

[2]Universidade Federal do Rio Grande do Sul (UFRGS) - Porto Alegre - RS - Brasil

{vinicius.ribas,padoin}@unijui.edu.br

Resumo. *Este artigo apresenta a proposta de um novo balanceador de carga que almeja a redução do tempo de execução de aplicações paralelas quando executadas em ambientes de memória compartilhada. O algoritmo proposto foi implementado no modelo de programação paralela* CHARM++. *Os resultados mensurados apresentaram speedup de* $1,22$ *à* $1,65$ *em relação ao tempo de execução com diferenças significativas no nível de desbalanceamento final na execução das aplicações.*

1. Introdução

O desenvolvimento de novos sistemas computacionais buscam suprir a demanda de alto desempenho proveniente do crescimento de simulações e pesquisas científicas. Atualmente, a maioria das aplicações paralelas apresentam comportamentos dinâmicos com cálculos baseados em fórmulas complexas. Por conta disso, empresas e instituições buscam adquirir uma infraestrutura suficiente para suportar a demanda computacional destas aplicações [Arruda et al. 2015]. Um problema surge devido ao fato que a maioria destas aplicações apresentam desbalanceamento de carga, impedindo um uso eficiente dos recursos computacionais das máquinas paralelas [Padoin et al. 2014]. Diante deste problema, balanceadores de carga são desenvolvidos almejando melhor distribuir de cargas das tarefas entre as unidades de processamento.

Nesse sentido, este artigo apresenta a proposta de um novo balanceador de carga (BC) denominado SMARTLB. Nossa proposta almeja a redução do tempo total de execução da aplicação por meio da migração de tarefas considerando a carga atual dos cores e das tarefas.

O restante do trabalho está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta a proposta do balanceador de carga SMARTLB. A Seção 4 descreve a metodologia utilizada na implementação e o ambiente de execução utilizado na realização dos testes. Resultados são discutidos na Seção 5, seguidos das Conclusões e Trabalhos Futuros.

2. Trabalhos Relacionados

Várias aplicações científicas adotam estratégia centralizada de balanceamento de carga. Neste modelo, as decisões de balanceamento de carga são realizadas em um processador específico, com base nos dados de carga tempo de execução [Zheng et al. 2010]. Outras, por sua vez recorrem a esquemas de balanceamento de carga com diferentes estratégias e constroem seu próprio modelo de carga de trabalho para orientar a atribuição de trabalho aos processos.

*Trabalho parcialmente apoiado por UNIJUI e CNPq. Pesquisa tem recebido recursos do edital da VRPGPE de bolsa e PIBIC/UNIJUI.

Balanceadores de carga tem sido desenvolvidos almejando uma melhor distribuição das tarefas entre as unidades de processamento. Do ambiente CHARM++, considerou-se os seguintes balanceadores de carga para implementação da nossa proposta:

- **GREEDYLB**: é um algoritmo guloso. Seu paradigma é frequentemente usado na teoria e na prática de otimização combinatória fazendo com que a grande maioria das tarefas sejam migradas a cada iteração [Bang-Jensen et al. 2004];
- **REFINELB**: éalgoritmo que move tarefas dos cores mais carregados para os menos carregados almejando atingir uma média limitando a quantidade de migrações; e
- **AVERAGELB**: constitui-se de uma melhoria da estratégia do algoritmo GREEDYLB. Seu algoritmo leva em consideração a média aritmética do processador para decidir quais tarefas serão migradas [Freytag et al. 2015].

3. SmartLB

A estratégia utilizada para implementação do balanceamento de carga proposto constitui-se de melhorias nas estratégias utilizadas nos algoritmos GREEDYLB e REFINELB. Nossas melhorias buscam equilibrar as cargas entre os processadores reduzindo o número de migrações. Para tanto, é adotado um valor de *threshold* que determina o desbalanceamento de carga aceitável.

Quando o balanceador é chamado, ele primeiramente verifica as cargas do cores mais e menos carregado e a diferença de carga (desbalanceamento de carga) entre eles. Caso essa diferença for maior que o *threshold* , o algoritmo seleciona tarefas com carga menor ou igual ao desbalanceamento para mover do core mais carregado para o menos carregado.

O SMARTLB foi desenvolvido utilizando o framework de balanceamento de cargas disponibilizado pelo CHARM++. Este framework de balanceamento de carga foi escolhido uma vez que permite tanto a criação de novos BC, quanto a utilização dos BCs disponibilizados pelo ambiente para comparações de resultados.

4. Metodologia

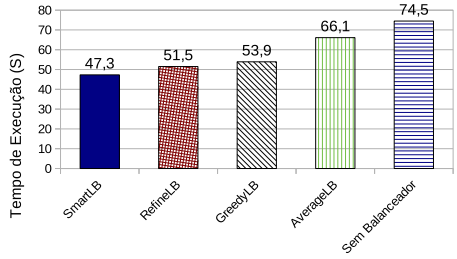
Para analisar os resultados alcançados com o balanceador de carga proposto, testes foram realizados utilizando-se 2 benchmarks disponibilizados pelo ambiente de programação CHARM++. O primeiro, *lb_test*, foi configurado com carga computacional que varia entre 1500ms e 1500000ms. O segundo, *kNeighbor* foi configurado para realizar testes com 100000 mensagens. Ambos os benchmarks foram executados com 150 iterações por tarefa e com sincronização do balanceador a cada 10 iterações. Em ambos os testes foi adotado um *threshold* de valor igual a 5%.

O desempenho alcançado com o BC proposto foi comparado com dois balanceadores de carga disponíveis no CHARM++, o GREEDYLB e o REFINELB; e com o proposta do AVERAGELB. O equipamento utilizado nos testes possui um processador Intel Core i7-4790 de 8 cores. Possui sistema operacional Linux Ubuntu 16.04 com kernel versão 4.4.33. A versão do CHARM++ utilizada para implementação foi a 6.5.1 e compilador g++ de versão 5.4.1.

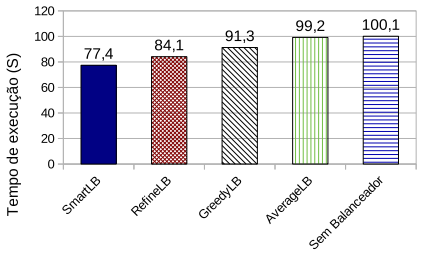
5. Resultados

Na Figura 1 são apresentados os tempos de execução dos testes realizados com o benchmark *lb_test* e *kNeighbor* para diferentes quantidades de tarefas.

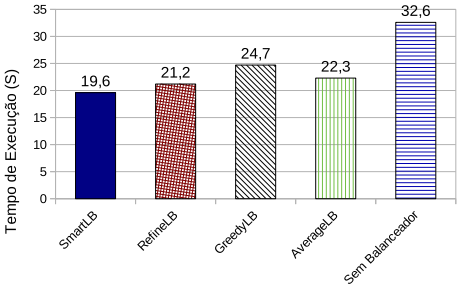
Figura 1. Tempos de execução mensurados durante a execução



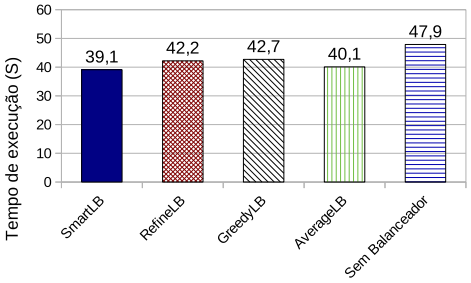
(a) *lb_test*- 100 tarefas



(b) *lb_test*- 200 tarefas



(c) *kNeighbor*- 100 tarefas

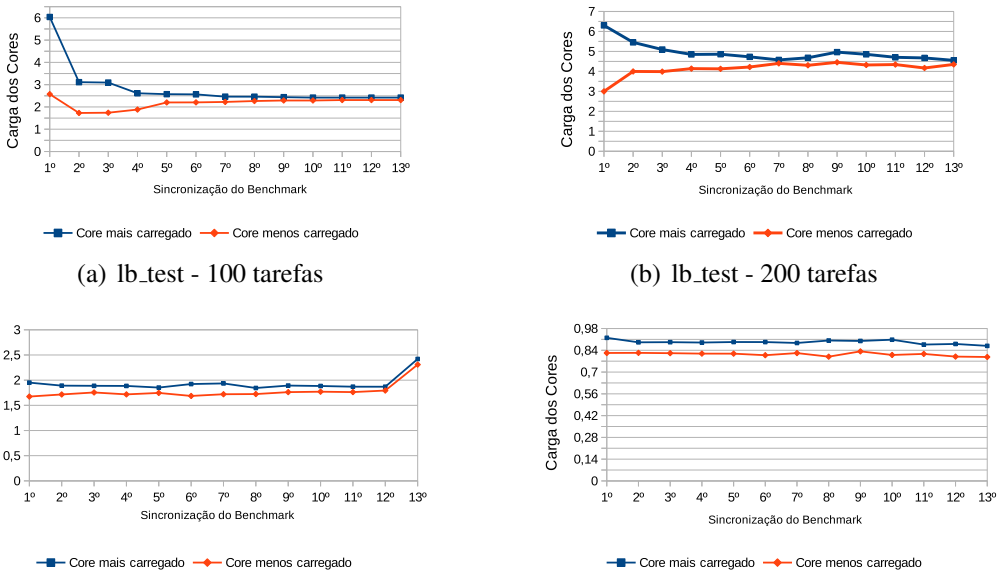


(d) *kNeighbor*- 200 tarefas

O BC SMARTLB apresentou melhor desempenho para ambos os benchmarks testados. Para lb_test com 100 tarefas o SMARTLB conseguiu reduzir o tempo para 47,3 segundos, o que representa uma redução de $36,5\%$ em relação a execução sem balanceador e $28,52\%$ em relação ao AVERAGELB. Com 200 tarefas o SMARTLB também apresentou redução de $22,64\%$ em relação a execução sem balanceador e, $7,9\%$ menor que o REFINELB. Quando utilizado com o benchmark *kNeighbor*, o SMARTLB também obteve os menores tempos de execução. Ele foi $20,41\%$ e $8,45\%$ menor que GREEDYLB para 100 e 200 tarefas respectivamente.

Na Figura 2 são apresentados os desbalanceamentos de carga mensurados a cada chamada do balanceador de carga proposto. Em todos os testes, o SMARTLB conseguiu concluir a execução dos benchmarks apresentando o menor de desbalanceamento. Nos testes com o lb_test com 100 e 200 tarefas, na primeira chamada do balanceador a diferenças entre o core mais carregado e o core menos carregado era de $3,46$ e $3,30$ unidades respectivamente. O SMARTLB conseguiu reduzir, até a última sincronização, essas diferenças para $0,12$ e $0,20$ respectivamente. Nos testes realizados com o benchmark Kneighbor com 100 e 200 tarefas, as diferenças entre os cores mais carregados e menos carregados eram de $0,10$ e $0,28$. Com a utilização do SMARTLB, essas diferenças diminuíram para $0,07$ e $0,11$ respectivamente.

Figura 2. Desbalanceamento mensurado durante a execução dos benchmarks



(a) lb_test - 100 tarefas

(b) lb_test - 200 tarefas

(c) *kNeighbor*- 100 tarefas

(d) *kNeighbor*- 200 tarefas

6. Conclusões e trabalhos futuros

Este trabalho apresentou a proposta de um novo balanceador de carga denominado SMARTLB. Os resultados alcançados com o SMARTLB mostraram-se muito consistentes. Conseguiu-se reduzir o nível de desbalanceamento de carga e consequentemente o tempo de execução quando aplicado nos dois benchmarks selecionados.

Como futuros trabalhos, pretende-se realizar melhorias no algoritmo de tomada de decisão do SMARTLB de modo a melhorar o controle de migrações de tarefas. Pretende-se também realizar testes em sistemas paralelos maiores utilizando problemas reais de computação científica bem como comparar com outros balanceadores de carga do estado da arte.

Referências

Arruda, G., Padoin, E. L., Pilla, L. L., Navaux, P. O. A., and Mehaut, J.-F. (2015). Proposta de balanceamento de carga para redução de migração de processos em ambientes multiprogramados. In *XVI Simpósio de Sistemas Computacionais (WSCAD-WIC)*, pages 1–8.

Bang-Jensen, J., Gutin, G., and Yeo, A. (2004). When the greedy algorithm fails. *Discrete Optimization*, 1(2):121–127.

Freytag, G., Arruda, G., Martins, R. S. M., and Padoin, E. L. (2015). Análise de desempenho da paralelização do problema de caixeiro viajante. In *XV Escola Regional de Alto Desempenho (ERAD)*, pages 1–4, Gramado, RS. SBC.

Padoin, E., Castro, M., Pilla, L., Navaux, P., and Mehaut, J.-F. (2014). Saving energy by exploiting residual imbalances on iterative applications. In *High Performance Computing (HiPC), 2014 21st International Conference on*, pages 1–10.

Zheng, G., Meneses, E., Bhatele, A., and Kale, L. V. (2010). Hierarchical load balancing for charm++ applications on large supercomputers. In *2010 39th International Conference on Parallel Processing Workshops*, pages 436–444. IEEE.