Modalidade do trabalho: Relatório técnico-científico
Evento: XXIV Seminário de Iniciação Científica

ANÁLISE DE DESEMPENHO DA APLICAÇÃO DE BALANCEAMENTO DE CARGA EM BENCHMARK SINTÉTICOS[1]

Bruna Schneider Padilha[2], Edson Luiz Padoin[3].

[1] Resumo expandido resultado do Projeto de Pesquisa Utilização de Balanceamento de Carga e DVFS na Paralelização de Aplicações Almejando Aumento da Eficiência Energética em Sistema Paralelos e Heterogêneos da Unijuí.

[2] Acadêmica do 7º semestre do Curso de Ciências da Computação da Unijuí; Bolsista PIBIC Unijuí, email: brunaspadilha0@gmail.com
[3] Professor Orientador do Departamento de Ciências Exatas e Engenharias. E-mail: padoin@unijui.edu.br.

Introdução

Aplicações complexas com excessiva comunicação entre tarefas e desbalanceamento de carga impedem o uso eficiente do potencial existente nos processadores. Dessa maneira, algumas unidades de processamento podem permanecer ociosas enquanto outras executam partes da aplicação. Como consequência disso, determinadas aplicaçõGráfico 2. Resultados dos testes utilizando benchmark lb_test com 200 chareses levam mais tempo para executar, consumindo mais energia.

Para resolver tais problemas, balanceadores de carga têm sido desenvolvidos, almejando utilizar todo o potencial dos sistemas computacionais e aumentar a eficiência em nível de processamento e tempo de execução. Nesse contexto, muitas aplicações paralelas que envolvem simulação com comportamentos dinâmicos ou cálculos baseados em diversas fórmulas complexas tem utilizado tais recursos devido ao desbalanceamento de carga e a quantidade de comunicações (Pilla and Meneses 2015).

Nesta área de pesquisa foi desenvolvida a plataforma Charm++. Ela visa equilibrar a carga computacional entre os processadores, podendo assim reduzir o tempo total de execução e por consequência reduz o consumo de energia do sistema. As tarefas paralelas, denominadas de chares, comunicam-se entre si através de chamadas assíncronas, usando um sistema orientado a objeto simultâneo, mas com clara separação de objetos paralelos. Sendo que esses objetos são mapeados pelo sistema para execução em determinado elemento processador, podendo ser migradas para outro processador para equilibrar a carga do sistema.

Nesta análise de desempenho foi utilizado o benchmarks lb_test desenvolvido na plataforma Charm++. Selecionou-se os balanceadores de carga DummyLB, RefineLB, GreedyLB, e AverageLB, uma vez que cada balanceador possui um método específicos para tomada de decisões e migração de tarefas entre os processadores, permitindo assim dividir uma aplicação em objetos migráveis, tentando manter os núcleos ocupados pelo maior tempo possível, causando a perda de eficiência.

Metodologia

Modalidade do trabalho: Relatório técnico-científico
Evento: XXIV Seminário de Iniciação Científica

Para execução dos testes foram realizados a instalação da plataforma Charm++ em um equipamento com processador Intel Core i7, com 4 núcleos, 16GB de memória RAM, placa de vídeo AMD 2GB e disco de 1TB. O sistema operacional instalado foi Ubuntu 14.04 LTS 64-bits.
O benchmark o lb_test foi escolhido por ser iterativo podendo ser configurado para apresentar diferentes níveis de desbalanceamento de carga. Assim, permitindo que a carga computacional de cada tarefa seja configurada em diferentes padrões de carga tanto de irregularidade quanto de comunicação.
Devido aos estudos realizados foram selecionados quatro balanceadores, cada um com características específicas dos algoritmos, que cada um utiliza no tratamento para migração de tarefas. Os quais são:
DummyLB: é um algoritmo que não realiza migração de tarefas, sendo apenas utilizado para comparação de desempenho;
RefineLB: é um algoritmo refinado de busca, acionando assim uma estratégia que pode determinar quando e cujas tarefas podem ser migradas usando limite de migração (Yagoubi end Slimani, 2006).
GreedyLB: é um algoritmo guloso, a grande maioria das tarefas são migradas com base nas informações disponíveis na iteração corrente (Bang-Jensen et al, 2004).
AverageLB: é uma variação do algoritmo GreedyLB, pois é a média aritmética de cada carga que decide quais objetos devem ser migrados, usando uma abordagem centralizada, assim evitando migrações desnecessárias (Arruda, 2015).
Para aumentar o desempenho da aplicação Charm++, não é necessário apenas melhorar as memórias e escolher o melhor tipo de comunicação, é preciso fazer uso de balanceadores de carga que conheçam, a arquitetura da máquina, considerem o mapeamento inicial dos chares e que evitem migrações excessivas, assim aumentando a eficiência do sistema computacional.

Resultados e Discussões
Nesta seção serão analisados tempos de execução mensurados nos testes realizados com o benchmark lb_test e com os balanceadores de carga DummyLB, RefineLB, GreedyLB, e AverageLB. O benchmark foi executado com cada um dos balanceadores, configurado para 100 chares e 150 iterações em cada tarefa, que variam de 5000ms e máximo de 80000ms. A cada 10 iterações o balanceador é chamado para analisar o desbalanceamento e migrar tarefas se necessário.

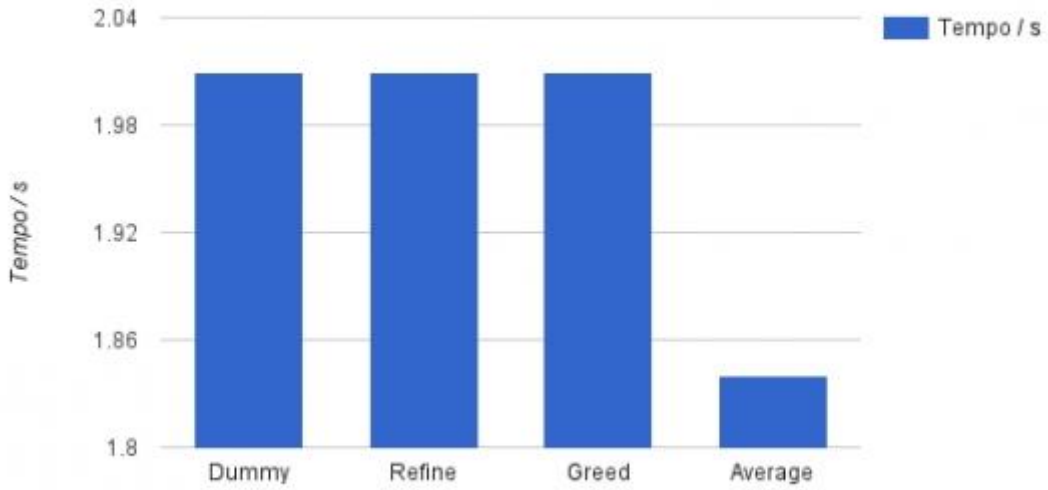

Gráfico 1. Resultados dos testes utilizando benchmark lb_test com 100 chares

Com base nos tempos mensurados, observa-se que com 100 chares, o benchmark lb_test, apresenta desempenho semelhante com 3 balanceadores de carga, sendo que semente o AverageLB reduziu o tempo de execução para 1,84 segundos, conforme apresentado no Gráfico 1.
Um segundo teste foi realizado com 200 chares e as mesmas configurações do primeiro teste.

Modalidade do trabalho: Relatório técnico-científico
Evento: XXIV Seminário de Iniciação Científica

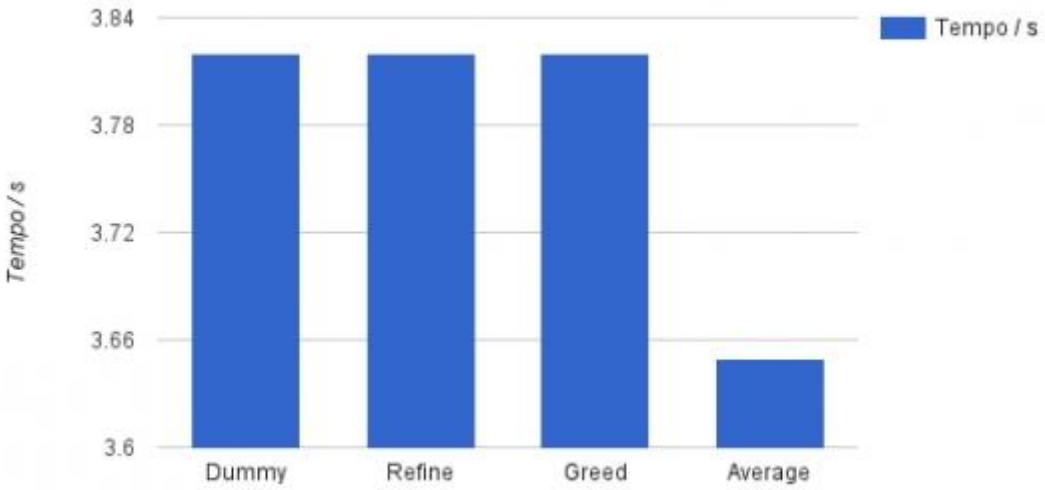

Gráfico 2. Resultados dos testes utilizando benchmark lb_test com 200 chares

Percebe-se um aumento no tempo total de execução do benchmark em função do aumento do número chares, entretanto, semelhante ao primeiro teste, percebe-se que somente o balanceador AverageLB reduz o tempo de execução, conforme apresentado no Gráfico 1.
Observa-se que os tempos de execução de cada balanceador de carga foram bastante semelhantes nos dois testes realizados. Isso se justifica devido ao pequeno desbalanceamento presente nestes testes.

Conclusões

Atualmente existe um grande número de pesquisas que almejam aumento da eficiência na utilização dos sistemas computacionais de alto desempenho. Isto justificado pela construção de máquinas com centenas de milhares de processadores, as quais buscam obter resultados mais precisos no menor tempo possível. Entretanto, um dos principais problemas é o desbalanceamento de carga presente nas aplicações executadas nestes sistemas. Tal desbalanceamento é responsável por impedir que as máquinas paralelas alcancem ótimos desempenhos.
Neste trabalho apresentou-se uma análise de desempenho do benchmark lb_test com os balanceadores de carga DummyLB, RefineLB, GreedyLB, e AverageLB. Ambos que utilizam um abordagem centralizada para tomada de decisão de balanceamento de carga. Entretanto, dado o baixo desbalanceamento dos testes conseguiu-se desempenhos muito semelhantes, para todos balanceadores utilizados.
Como trabalhos futuros pretende-se utilizar outros benchmarks bem como realizar testes com maiores desbalanceamentos de carga. Também espera-se fazer um monitoramento dos testes, para analisar os tempos de cada passo temporal bem como o overhead de balanceamento. Podendo

assim, a partir dos resultados obtidos, analisar simulação com os mesmos balanceadores comparando o total de objetos migrados de um processador para o outro.
Além de benchmarks, pretende-se utilizar aplicações reais como por exemplo, Stencil3D, na analise de desempenho e consumo de energia com diferentes balanceadores.

Palavras-Chave: Balanceamento de Carga; Benchmarks;

Agradecimentos: Agradeço pelo auxílio do professor orientador no desenvolvimento deste trabalho vinculado a minha bolsa de pesquisa "Utilização de Balanceamento de Carga e DVFS na Paralelização de Aplicações Almejando Aumento da Eficiência Energética em Sistema Paralelos e Heterogêneos".

Referências Bibliográficas

ARRUDA, Guilherme Henrique Schiefelbein; PADOIN, Edson Luiz. Balanceamento de carga em sistemas multiprocessadores utilizando o modelo de programação CHARM++. Salão do Conhecimento, v. 1, n. 1, 2015.
BANG-JENSEN, Jorgen; GUTIN, Gregory; YEO, Anders. When the greedy algorithm fails. Discrete Optimization, v. 1, n. 2, p. 121-127, 2004.
KALE, L. Parallel programming with charm: An overview. Dept. of Computer Science, University of Illinois at Urbana-Champaign, Tech. Rep, p. 93-8, 1993.
KALE, L. The Charm++ parallel programming system manual. Version 6.5.0. Dept. of Computer Science, University of Illinois at Urbana-Champaign.
MELO, Andreia; SKLYAROV, Valery; FERRARI, António. HiParaGraphs, uma Linguagem de Especificação de Algoritmos de Controlo Paralelos e Hierárquicos. Electrónica e Telecomunicações, v. 3, n. 3, p. 214-221, 2001.
NAVAUX, Philippe Olivier Alexandre. Dynamic Load-balancing: A New Strategy for Weather Forecast Models. 2011. Tese de Doutorado. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL.
PADOIN, Edson Luiz et al. Balanceamento de carga visando reduçao do consumo de energia para o modelo de programaçao Charm++. XIV Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul, Alegrete, RS, Brasil, 2014.
PILLA, Laércio L. et al. A hierarchical approach for load balancing on parallel multi-core systems. In: Parallel Processing (ICPP), 2012 41st International Conference on. IEEE, 2012. p. 118-127.
PILLA, Laércio L. et al. Avaliação da Adequação da Plataforma Charm++ para Arquiteturas Multicore com Memória Hierárquica.
PILLA, Laércio Lima; MENESES–ESTEBAN, Esteban. Programação Paralela em CHARM++.
PINHO, Eduardo G. "Uma Linguagem de Programação Paralela Orientada a Objetos para Arquiteturas de Memória Distribuída". Ceará, Fortaleza, 2012.
PINT0, Vinícius Garcia. Charm++: programação paralela orientada a objetos.
YAGOUBI, Belabbas; SLIMANI, Yahya. Dynamic load balancing strategy for grid computing. Transactions on Engineering, Computing and Technology, v. 13, p. 260-265, 2006.