AVALIAÇÃO DE ESTRATÉGIAS DE MAPEAMENTO DE TAREFAS PARA CLUSTERS DE COMPUTADORES HETEROGÊNEOS

Vitor C. F. Gomes, Vinicius V. Cogo, Andrea S. Charão
Universidade Federal de Santa Maria
Laboratório de Sistemas de Computação
{vconrado, vielmo, andrea}@inf.ufsm.br

Resumo. *Clusters* de computadores compostos por nós heterogêneos e multiprocessados representam um desafio para desenvolvedores de programas paralelos, pois requerem uma abordagem dinâmica na busca por ganho de desempenho. Este artigo apresenta uma avaliação de duas estratégias de mapeamento de tarefas para *clusters* heterogêneos com o objetivo de obter um perfil da simulação de incêndios florestais neste tipo de arquitetura em relação a estas estratégias. Os resultados mostram que o ganho de desempenho com uma abordagem dinâmica em relação a uma abordagem estática depende da quantidade de nós utilizados na computação. Este fato é devido ao custo de manter um nó gerenciando o mapeamento dinâmico.

1. Introdução

O acesso a equipamentos com mais de uma unidade ativa de processamento está se tornando cada vez mais comum. Além disso, estão cada vez mais disponíveis, em meios acadêmicos e empresariais, redes de estações de trabalho, também conhecidas como NOW (*Network Of Workstation*). A infraestrutura existente nestes ambientes pode ser utilizada como sistemas para a execução de programas paralelos e/ou distribuídos [1].

Estes sistemas representam uma boa oportunidade para a execução de aplicações paralelas, entretanto, os equipamentos disponíveis nesses ambientes normalmente possuem recursos heterogêneos, o que pode dificultar a utilização eficiente destes sistemas. Esta dificuldade aparece na distribuição de tarefas entre esses equipamentos com diferentes capacidades computacionais, a qual deve ser planejada para alocar uma carga de trabalho compatível com o poder de computação de cada nó.

Seguindo esta tendência, este trabalho implementa duas abordagens de mapeamento de tarefas em um cluster heterogêneo para obter o perfil de desempenho de um simulador de incêndios florestais neste tipo de arquitetura paralela. No restante deste artigo, são apresentadas técnicas de mapeamento de tarefas e a aplicação testada. Na seção 4 é apresentado o ambiente de teste, enquanto nas seções 5 e 6 são apresentados respectivamente os detalhes de implementação e os resultados . Na seção 7 são feitas as considerações finais.

2. Mapeamento de tarefas

No desenvolvimento de aplicações paralelas indica-se o uso de uma metodologia de projeto que possui quatro etapas [2]. Es-

tes estágios conhecidos como Particionamento, Comunicação, Aglomeração e Mapeamento (PCAM) servem para orientar o desenvolvimento de aplicações desta natureza. O Mapeamento consiste na etapa onde se indica em qual nó do *cluster* cada tarefa será executada [2]. A correta distribuição de tarefas evita que processadores tornem-se ociosos durante o processamento da aplicação, o que reduziria o desempenho global da aplicação.

De maneira geral, podemos identificar duas abordagens para realizar o mapeamento das tarefas entre os nós de um sistema paralelo. Com uma estratégia estática pode-se fixar o número de tarefas por nó, minimizando a comunicação e facilitando a divisão das tarefas, que pode ser feita antes da execução. Outra, visando maior flexibilidade, realiza a distribuição de tarefas em tempo de execução. Esta estratégia é chamada de mapeamento dinâmico e pode utilizar algoritmos de balanceamento de carga para realizar uma melhor distribuição de trabalho entre os nós. Esta última solução é normalmente utilizada para problemas onde o custo computacional de cada tarefa do problema é variável, e não é possível quantificá-lo previamente [2]. Além disso, pode ser usada para a distribuição de tarefas em um sistema heterogêneo, onde não é possível determinar o tempo de processamento de cada tarefa em cada nó do *cluster*.

Na Figura 1 é possível observar uma forma de realizar o mapeamento estático de tarefas. Nesta solução, cada processo (numerados de 0 a M) recebe uma porção igual de trabalho, independente do seu poder de processamento e do custo de computação da tarefa. Esta solução é ideal para problemas em que cada tarefa possui custo de processamento igual e as máquinas são computacionalmente equivalentes.

Uma forma de realizar o mapeamento dinâmico de tarefas é mostrado na Figura 2. A abordagem apresentada usa a técnica mestre-trabalhador para efetuar a distribuição de tarefas. Um processo Trabalhador (**T**) solicita ao processo Mestre (**M**) uma nova tarefa toda vez que se encontrar ocioso. Desta forma, um processo executado em uma máquina com maior poder de processamento requisitará mais trabalho e irá contribuir mais para a solução do problema. Esta técnica pode apresentar problemas quanto à escalabilidade pois a concentração da divisão de trabalho em um único processo pode tornar-se gargalo no sistema [2].

Figura 1. Mapeamento estático.

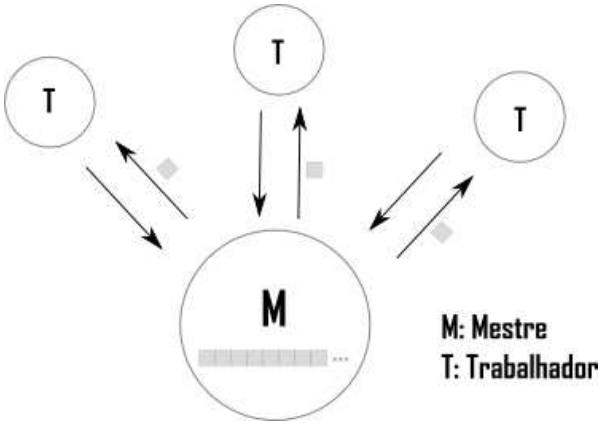

Figura 2. Mapeamento Mestre-Trabalhador

3. Aplicação

A aplicação escolhida para testar estas duas abordagens de mapeamento em um *cluster* heterogêneo foi um **simulador de incêndio florestal** baseado no Método de Monte Carlo. O Método de Monte Carlo é um método estatístico que utiliza amostras obtidas em distribuições observadas a fim de aproximar a função de interesse [4]. É comumente utilizado para obter aproximações numéricas de funções complexas, como por exemplo, o cálculo de integrais.

Na aplicação teste deste trabalho, a floresta é representada por uma matriz onde cada elemento desta é uma árvore. As implementações são baseadas no código de David Joiner [5] e adaptadas para o uso das abordagens em estudo.

4. Ambiente de testes

Para a execução dos testes deste trabalho foi utilizado um *cluster* heterogêneo com 6 máquinas interligadas por uma rede *Gigabit* dedicada. Este agregado, chamado Paple, está disponível no Laboratório de Sistemas de Computação da UFSM. As configurações das máquinas são apresentadas na Tabela 1. Os equipamentos executam sistema GNU/Linux Debian Lenny com Kernel 2.2.26. Observa-se que há nós com memória compartilha.

Tabela 1. Nós do cluster Paple

Nó	Processador	RAM
paple01	2 x Pentium III 1GHz	1GB
paple02	2 x Pentium III 1GHz	1GB
paple03	2 x Athlon 2GHz	1GB
paple04	1 x Pentium IV 2.4GHz	1GB
paple05	1 x Pentium IV 2.4GHz	384MB
paple06	2 x Athlon 2GHz	1GB

5. Implementação

Com o objetivo de utilizar de maneira eficiente os recursos do *cluster* heterogêneo foram utilizados dois níveis de paralelismo. O primeiro nível considera um sistema de memória distribuída e, para tanto, utiliza troca de mensagens através de MPI (MPICH) [3]. O segundo nível de paralelismo ocorre nas máquinas com múltiplos núcleos e utiliza-se para isso OpenMP. Com a utilização de diretivas OpenMP é possível tornar dinâmica a criação de fluxos de processamento (*threads*), o que facilmente se adapta à quantidade de unidades ativas do nó do sistema paralelo.

Para a implementação da aplicação que utiliza **mapeamento estático**, cada processo identifica as tarefas que deve processar através do uso de seu identificador e da quantidade de processos. A computação das tarefas inicia sem que sejam trocadas mensagens entre os processos. Ao fim da computação, o resultado é enviado para o nó 0 do *cluster* para poder ser impresso em tela.

Na implementação que utiliza **mapeamento dinâmico**, o processo de índice 0 é considerado Mestre e realiza o registro do recebimento dos resultados através de uma chamada MPI não bloqueante (*MPI_Irecv*). Esta técnica evita que ocorra a sincronização entre os processos no momento de envio dos resultados para o processo Mestre.

Cada processo Trabalhador inicia requisitando uma tarefa, realiza sua computação e envia o resultado para o processo Mestre. Na sequência requisita uma nova tarefa até que receba um trabalho com identificador inválido (-1), indicando que deve finalizar sua execução.

Após entregar todas as tarefas, o processo Mestre aguarda o recebimento de todos os resultados (*MPI_Waitall*) para imprimir os resultados e finalizar sua execução.

Nesta implementação, o processo Mestre não realiza a computação de tarefas, somente coordena a distribuição das mesmas.

Para a avaliação do desempenho das aplicações foram tomados os tempos sem considerar o tempo de impressão dos resultados, visto que podem haver variações significativas neste procedimento e que ele não representa parte do objetivo da avaliação.

6. Resultados

Os resultados obtidos com este trabalho referem-se ao cálculo da eficiência das abordagens paralelas, diferenciadas pela forma como são mapeadas as tarefas. É importante salientar que para os cálculos foi considerado

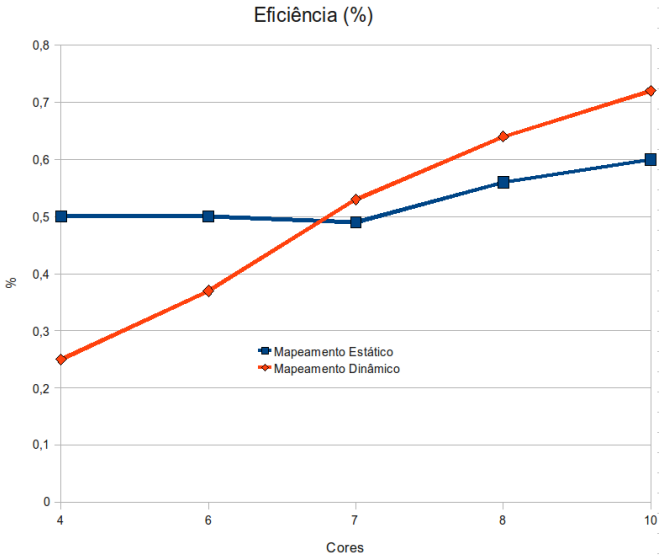

Figura 3. Eficiência

o número de núcleos envolvidos no processamento da aplicação e não o número de máquinas, visto que algumas máquinas possuem mais de um núcleo. Esta consideração é importante, pois pode-se ter uma visão errada da eficiência de uma aplicação paralela, quando não é considerada corretamente a quantidade de unidades funcionais. Além disso, o nó Mestre, apesar de não realizar a computação de tarefas é considerado no cálculo da eficiência neste trabalho, visto que este recurso foi alocado para a computação do mesmo. Para evitar variações nos dados coletados, são considerados os tempos médios tomados em 5 execuções de cada teste.

Na Figura 3 é apresentado o gráfico contendo a eficiência encontrada para cada uma das implementações. É possível observar que para até 6 núcleos o mapeamento estático foi mais eficaz. O mapeamento dinâmico apresenta resultados piores, neste caso, devido ao custo de utilizar 2 núcleos (paple01) somente para a distribuição de trabalho. A partir de 7 núcleos este custo acaba por ser pequeno em relação ao benefício de distribuir dinamicamente as tarefas entre os processos da computação paralela.

7. Considerações Finais

Neste trabalho foram apresentadas duas abordagens distintas quanto ao mapeamento de tarefas, analisando-as quanto ao desempenho em um sistema computacional heterogêneo, utilizando dois níveis de paralelismo. Para a avaliação das abordagens foram desenvolvidos dois programas que realizam a simulação de incêndios florestais usando o Método de Monte Carlo.

Como visto na Figura 3 a abordagem de distribuição dinâmica mostrou-se eficaz quando utilizados 7 ou mais núcleos. Conclui-se que o preço de ocupar um nó do *cluster* para a distribuição foi alto quando poucas (menos de 7) unidades atuam na computação do problema.

Uma possível solução para reduzir este efeito é a utilização de um dos núcleos do nó Mestre para a computação de tarefas. Com esta técnica, possivelmente haverá um aumento da eficiência na aplicação que distribui dinamicamente suas tarefas. Seguindo esta tendência, este trabalho segue na implementação deste recurso a fim de avaliar o comportamento desta possível configuração.

Referências

[1] CRAD-RS. *Caderno dos Cursos Permanentes.* SBC, 2006.

[2] I. Foster. *Designing and building parallel programs: concepts and tools for parallel software engineering.* Addison-Wesley, 1995.

[3] W. D. Gropp and E. Lusk. *User's Guide for* `mpich`*, a Portable Implementation of MPI.* Mathematics and Computer Science Division, Argonne National Laboratory, 1996.

[4] J. T. Halton. A Retrospective and Prospective Survey of the Monte Carlo Method. *SIAM Review*, 12, 1970.

[5] D. Joiner. Firestarter. http://njcstme.kean.edu:8080/EPortfolios/Members/joinerd, Acesso em 20/09/09.