Considerações sobre o Mapeamento Estático de Processos em Agregados de Computadores Multicore

Vicente S. Cruz, Manuela K. Ferreira, Philippe O. A. Navaux

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{vscruz, mkferreira, navaux}@inf.ufrgs.br

1. Introdução

A demanda por poder computacional é uma constante e dentre as soluções para aumentar o desempenho no processamento de tarefas exigentes por grande capacidade de processamento [Kumar 2007][Brams 2009] destaca-se a classe dos agregados de computadores (*clusters*) [Navaux 2003]. Essas máquinas foram elaboradas com o objetivo de explorar o paralelismo intrínsico às aplicações de alto desempenho. Assim, é importante definir como as tarefas são atribuídas aos nodos dessas máquinas de modo a maximizar a utilização dos recursos oferecidos por elas. Tal questão é definida como mapeamento de processos. O objetivo deste trabalho consiste em analisar alguns aspectos do mapeamento de processos estáticos aplicados a *clusters multicore*.

O MPI é o padrão *de facto* para o desenvolvimento de aplicações de alto desempenho para *clusters* [Gropp 1999]. Ele é a especificação de um padrão para a criação de um *framework* de programação paralela em ambientes multiprocessados, de modo que ao usar uma ferramenta que implemente esse padrão (*e. g.* LAM MPI, MPICH), é possível alterar algumas bibliotecas de modo a considerar os aspectos abordados nesse trabalho.

2. Aspectos do Mapeamento Estático em *Clusters Multicore*

O mapeamento de processos é definido como um problema NP-Difícil, o que exige a definição de heurísticas para elaborar uma boa solução para casos específicos [Bokhari 1981]. As características abordadas neste trabalho possuem foco no mapeamento estático, embora elas também possam ser consideradas no mapeamento dinâmico. No mapeamento estático, cada processo é atribuído a um respectivo recurso no início do processamento, e permanece nesse local até a finalização da computação.

De um modo geral, ao efetuar o mapeamento de processos, deve-se levar em consideração a comunicação entre as tarefas. Em *clusters*, quando dois processos que possuem alto volume de comunicação são mapeados em nodos distantes, a rede de interconexão torna-se um gargalo [Bhatele 2009] porque além da rede ser o meio de tráfego de dados de maior latência [Chai 2007], o pacote deve efetuar diversos saltos (*hops*) para chegar ao seu destino. Somado a isso, quando existe muita demanda para o envio de mensagens, a disputa de acesso ao meio se torna maior, levando ao problema de contenção da rede. O ideal é que o algoritmo de mapeamento minimize esse gargalo, escalonando os processos que trocam muitas mensagens em nodos vizinhos, reduzindo,

dessa forma o número de *hops*. Se houver mais de um processador por nodo, então as tarefas devem ser alocadas no mesmo, de modo que explorem os meios de maior vazão. Por outro lado, ao alocar diversas tarefas em um mesmo nodo, inferioriza-se a exploração do paralelismo oferecido pelo *cluster*, implicando em perda de desempenho. Além disso, a memória transforma-se no gargalo da aplicação, uma vez que diversas tarefas buscarão instruções e dados para processar, implicando no aumento de faltas na cache (*cache miss*) [Alves 2009].

3. Objetivos

O objetivo de considerar tais fatores consiste em definir uma heurística que permita a elaboração de um bom mapeamento para obter maior desempenho na execução de aplicações paralelas em *clusters*. Algumas técnicas de mapeamento se aplicam à *threads* e processos [Rodrigues 2009]. Dessa forma, pretende-se obter uma boa heurística de mapeamento de processos em *clusters multicore* que minimize a latência de acesso à rede e maximize o compartilhamento de memória, sem que esta se torne um gargalo.

Referências

Alves, M. A. Z., Freitas, H. C., Navaux, P. O. A. (2009) Investigation of shared l2 cache on many-core processors. In Proceedings Workshop on Many-Core, pages 21–30, Berlin. VDE Verlag GMBH.

Bhatele, A., Kale, L. V. (2009) Quantifying Network Contention on Large Parallel Machines, Parallel Processing Letters (Special Issue on Large-Scale Parallel Processing), Vol. 19 Issue 4, Pages 553-572.

Bokhari, S. H. (1981) On the Mapping Problem. IEEE Trans. Computers, 30(3):207-214.

Brams (2009). Disponível em http://www.cptec.inpe.br/brams/.

Chai, L., Gao, Q. e Panda, D. (2007) Understanding the Impact of Multicore Architecture in Cluster Computing: A Case Study with Intel DualCore System, In Cluster Computing and the Grid, CCGRID, Seventh IEEE Internacional Sysmposium on, p. 471478.

Gropp, W.; Lusk, E.; Skjellum, A. (1999) “Using MPI – Portable Parallel Programming with Message-Passing Interface”. 2.ed. Massachusetts Institue of Technology – Cambridge, Massachusetts 02142: The MIT Press.

Navaux, P. O. A., DeRose, C. A. F. (2003) “Arquiteturas Paralelas”. 1.ed. Editora Sagra Luzzato, 2003. n.15.

Kumar, S., et al (2007). “Scalable Molecular Dynamics with NAMD on Blue Gene/L”. IBM Journal of Research and Development: Applications of Massively Parallel Systems, 52(1/2).

Rodrigues, E. R, Madruga, F. L. e Navaux, P. O. A., Multicore Aware Process Mapping and its Impact on Communication Overhead of Parallels Applications, IEEE Symposium on Computers and Communications (ISCC), 2009.