Auto-tuning de Regiões de Sobreposição Heterogêneas para Domínios Estruturados em Ambientes Paralelos*

Alexandre Almeida[1], **Nicolas Maillard**[1]

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

{avalmeida,nicolas}@inf.ufrgs.br

1. Introdução

Auto-tuners são sistemas que surgiram com o objetivo de adaptar automaticamente um determinado software a uma arquitetura alvo, de tal forma que o desempenho do software seja maximizado na plataforma em questão [Whaley et al. 2001, Sessão 2.1]. Originalmente, a técnica de *auto-tuning* surgiu para autoadaptar bibliotecas numéricas em máquinas sequenciais, podendo-se citar a biblioteca ATLAS [Whaley and Petitet 2005] como um exemplo. Neste contexto, *auto-tuning* de software é uma técnica já bem estabelecida. Entretanto, poucos são os trabalhos que apresentam *auto-tuning* aplicado à softwares paralelos para máquinas de memória distribuída, tais como ambientes de *cluster* de computadores.

Este trabalho propõe o estudo de *auto-tuning* aplicado à Decomposição de Domínios (DD) em máquinas paralelas de memória distribuída.

2. Contexto científico

O processo de DD, em computação paralela, refere-se às técnicas de distribuição dos dados de um modelo computacional entre os processadores de uma máquina de memória distribuída, processo este normalmente empregado para o processamento paralelo de Equações Diferenciais Parciais (EDPs) [Smith et al. 1996]. O processo de DD pode beneficiar-se das técnicas de *auto-tuning* para efetuar a adaptação automática do número de regiões de sobreposição entre subdomínios de um domínio estruturado.

Em uma máquina de memória distribuída, cada processo da aplicação paralela possui um pedaço do domínio, denominado subdomínio. Além dos dados do subdomínio em si, cada subdomínio estende-se em uma linha/coluna sobre seus vizinhos, a fim de satisfazer a dependência de dados dos cálculos. Essas linhas/colunas são denominadas *regiões de sobreposição* [Balay et al. 2008, Sessão 2.4], e a cada iteração do algoritmo paralelo seus valores devem ser recebidos dos subdomínios vizinhos. A Figura 1(a) apresenta, em destaque, um subdomínio alocado a um processo, onde os retângulos tracejados representam as regiões de sobreposição. O cálculo de cada ponto dos subdomínios depende dos quatro pontos vizinhos na horizontal e na vertical (geometria de 5 pontos).

O processo de DD pode beneficiar-se das técnicas de *auto-tuning* para aumentar automaticamente o número de regiões de sobreposição em cada subdomínio, definido como sendo igual a g. Aumentando-se g, cada processo tem condições de calcular os pontos sobrepostos de seus vizinhos ao invés de transferi-los pela rede de comunicação. Assim, posterga-se a comunicação entre subdomínios a cada g iterações do algoritmo.

*Trabalho financiado pela CAPES.

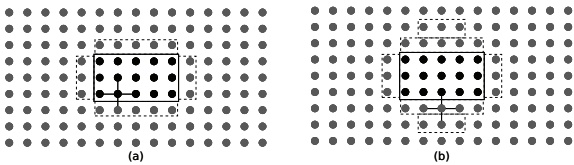

Figura 1. Subdomínios com regiões de sobreposição homogêneas (a) e regiões heterogêneas (b)

Embora existam trabalhos, como o de [Meng and Skadron 2009], que abordem a adaptação automática de g, a literatura não apresenta uma abordagem ainda mais maleável, que leve em consideração não apenas várias regiões sobrepostas, mas também regiões heterogêneas com relação ao seus tamanhos, assim como ilustra a Figura 1(b).

3. Objetivos do trabalho

Este artigo é uma sintetização de uma Proposta de Estudos e Pesquisa de Mestrado ainda a ser desenvolvida, e tem por objetivo propor um *auto-tuner* que adapte, em arquiteturas de *clusters*, o número de regiões de sobreposição bem como seus respectivos tamanhos.

A ideia por trás de ter regiões heterogêneas é buscar um algoritmo paralelo que esconda as comunicações entre os subdomínios, recebendo partes de suas regiões de sobreposição enquanto outros pontos são calculados. A EDP a ser adotada será a equação de Poisson por apresentar uma estrutura de computação simples, e também pelo fato de algoritmos mais complexos frequentemente assumirem a mesma estrutura de comunicação [Gropp et al. 1999, Sessão 4.1].

O *auto-tuning* será efetuado com base em uma modelagem teórica do algoritmo a ser desenvolvido, e levará em consideração as características da máquina paralela, assim como rede e poder de processamento dos nodos. Dessa forma, o *auto-tuner* buscará uma relação ótima (ou sub-ótima) entre o custo de recálculo das regiões, e a troca de dados pela rede de comunicação.

Referências

Balay, S., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H. (2008). PETSc Users Manual. Technical Report ANL-95/11 - Revision 3.0.0, Argonne National Laboratory.

Gropp, W., Lusk, E., and Skjellum, A. (1999). *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. The MIT Press, United States, 2nd edition.

Meng, J. and Skadron, K. (2009). Performance Modeling and Automatic Ghost Zone Optimization for Iterative Stencil Loops on GPUs. In *ICS '09*, pages 256–265, New York. ACM.

Smith, B., Bjørstad, P., and Gropp, W. (1996). *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University.

Whaley, R. C. and Petitet, A. (2005). Minimizing Development and Maintenance Costs in Supporting Persistently Optimized BLAS. *Softw. Pract. Exper.*, 35(2):101–121.

Whaley, R. C., Petitet, A., and Dongarra, J. J. (2001). Automated Empirical Optimizations of Software and the ATLAS Project. *Parallel Computing*, 27(1-2):3–35.