Interfaces para Programação Paralela, uma Alternativa para Anahy[1]

Deives Mesquita Kist[2], Gerson Geraldo H. Cavalheiro, André Rauber Du Bois

Programa de Pós-Graduação em Computação
Centro de Desenvolvimento Tecnológico – Universidade Federal de Pelotas (UFPel)
Campus Porto – Rua Gomes Carneiro, 1 – 96010-610 – Pelotas – RS – Brazil
{dmkist, gerson.cavalheiro, dubois}@inf.ufpel.edu.br

Resumo. *Neste trabalho serão estudados esqueletos de algoritmos paralelos presentes na literatura e estruturas algorítmicas recorrentes em programas paralelos, permitindo construir uma proposta para uma interface de programação (API) para Anahy utilizando esqueletos.*

1. Introdução

A crescente utilização das arquiteturas de computadores com mais de uma unidade de processamento motivaram o crescimento da demanda por software paralelo. No entanto, o desenvolvimento desse tipo de software tem alto grau de dificuldade, pois, dependendo da ferramenta de programação utilizada, o programador deve não apenas decompor explicitamente a concorrência da aplicação, mas também indicar no código de forma igualmente explícita o mapeamento das atividades concorrentes nos recursos de processamento disponíveis em uma máquina paralela.

Para que um programa em execução utilize de maneira eficiente uma determinada arquitetura, é necessário que as suas atividades paralelas sejam mapeadas sobre os recursos de processamento disponíveis de forma a minimizar algum índice de desempenho, por exemplo, o tempo de processamento total. Alguns ambientes de execução, como os providos por Anahy [Cavalheiro et al., 2007], Cilk [Blumofe et al.,1996] e OpenMP [Chandra et al.,2001], são dotados de mecanismos de escalonamento que permitem ao programador abstrair este mapeamento, cabendo ao programador apenas descrever a concorrência de sua aplicação. Estas ferramentas oferecem bons índices de desempenho de execução e a solução é atrativa pelos bons resultados de desempenho que oferecem. No entanto, o estado da arte de tais ambientes ainda carece de estudos sobre interfaces de programação.

2. Esqueletos para programação Paralela

Um esqueleto é uma função de alta ordem que encapsula padrões recorrentes de paralelismo [Gonzalez-Velez and Leyton, 2010]. Na computação paralela, os esqueletos capturam, organizam e mascaram do programador todos os detalhes envolvidos na estrutura da computação paralela que não são relevantes para o programador. Eles são utilizados para resolver problemas rotineiramente encontrados quando se desenvolve programas para execução paralela. Utilizando esqueletos, a natureza concorrente da

1 FAPERGS/PqG (11/1065-1) e PRONEX/FAPERGS/CNPq (10/0042-8).
[2] Bolsista de Mestrado CAPES

aplicação pode ser expressa sem que seja necessário informar nenhum atributo de como a execução paralela destas aplicações deve ser feita.

Esqueletos de algoritmos aproveitam padrões de programação comuns para ocultar a complexidade de aplicações paralelas e distribuídas. A partir de um conjunto básico de padrões (esqueletos), padrões mais complexos podem ser construídos através da combinação dos básicos. Assim, a programação é basicamente reduzida à habilidade de capturar e compor corretamente o conjunto de esqueletos para a aplicação.

3. Proposta

A atual API de Anahy é voltada ao desenvolvimento de aplicações no estilo de paralelismo de tarefas e sua implementação se dá na forma de uma biblioteca de primitivas que respondem a um subconjunto de serviços definidos pelo padrão *POSIX Threads* [Agrawal et al., 2010] , possuindo, portanto, um grau de abstração semelhante às bibliotecas que seguem este padrão. O trabalho em curso propõe o desenvolvimento de uma nova interface que contemple primitivas que implementem esqueletos para a programação concorrente, com maior capacidade de expressão e maior vocação para explorar arquiteturas altamente paralelas. No presente estudo esta proposta é abordada considerando que novas aplicações devam ser desenvolvidas para arquiteturas com um grande número de unidades de processamento.

Logo, serão estudados esqueletos de algoritmos paralelos presentes na literatura [Gonzalez-Velez and Leyton, 2010] e estruturas algorítmicas recorrentes em programas paralelos, permitindo construir uma proposta para uma interface de programação (API) para Anahy utilizando esqueletos. A presente proposta busca oferecer uma interface de programação com maior poder de expressão, permitindo ao programador obter todos os benefícios do ambiente de execução modelado por Anahy.

Referências

Agrawal, K.; Leiserson, C. E.; Sukha, J. Executing task graphs using work-stealing. 2010 IEEE International Symposium on Parallel Distributed Processing IPDPS, [S.l.], p.1–12, 2010.

Blumofe, R. D.; Joerg, C. F.; Kuszmaul, B. C.; Leiserson, C. E.; Randall, K. H.; Zhou, Y. Cilk: An Efficient Multithreaded Runtime System. Cambridge, MA, USA: [s.n.], 1996.

Cavalheiro, G. G. H.; Gaspary, L. P.; Cardozo, M. A.; Cordeiro,O. C. Anahy: a programming environment for cluster computing. In: VII High Performance Computing for Computational Science, 2007, Berlin. Anais. Springer-Verlag, 2007. (LNCS 4395).

Chandra, R.; Dagum, L.; Kohr, D.; Maydan, D.; MCDonald, J.; Menon,R. Parallel programming in OpenMP. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001.

Gonzalez-Velez, H., Leyton, M.(2010b). A Survey of Algorithmic Skeleton Frameworks: High-Level Structured Parallel Programming Enablers. In Journal of Software: Practice and Experience. 2010 Published online in Wiley InterScience, 40(12):pages1135-1160.