Reduzindo o Consumo Energético por Controle Dinâmico de Paralelismo em Roubo de Tarefas

Luís Felipe Millani[1], **Lucas Mello Schnorr**[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

{lfgmillani,schnorr,nicolas}@inf.ufrgs.br

Resumo. *Este trabalho propõe um método de redução de consumo energético de aplicações em Cilk Plus por meio de dynamic concurrency throttling nas regiões onde o paralelismo não escala para o número de cores disponíveis.*

Os custos energéticos representam uma porção significativa dos custos totais de sistemas em larga escala. Apesar de esforços para amenizar os efeitos econômicos e ambientais do alto consumo energético este tende a continuar em ascenção [Kestor et al. 2013].

A lista Green500 [Subramaniam et al. 2014] ranqueia os supercomputadores da lista Top500 [Meuer et al. 2014] de acordo com sua eficiêcia energética. A lista mostra que o problema da baixa eficiência energética, se não resolvido, deve tornar-se um fator limitante na corrida em direção à computação exaescalar. 20 MW, ou 50GFLOPs/Watt, é considerado um limite de consumo viável para um sistema exaescalar [Torrellas 2009]. No entanto, mesmo o líder do Green500, o TSUBAME-KFC, possui uma eficiência de apenas 4.4GFLOPs/Watt. Este valor extrapoado para 1 exaflop resulta em 227 MW, o que é inviável economicamente. Não é mais viável considerar-se o desempenho de um sistema em larga escala de forma isolada, é necessário reduzir o consumo energético.

Neste trabalho é apresentada uma proposta para melhoria da eficiência energética por meio do controle por software do uso dos *cores* disponíveis em um sistema *multicore*.

Uma possibilidade para redução do consumo energético é o uso de *Dynamic Concurrency Throttling* (DCT) [Li et al. 2013]. Esta técnica consiste em controlar o número de *cores* em execução em um determinado momento. Com uma menor porção do chip em uso, o consumo de potência é reduzido. Seções do código com ampla escalabilidade utilizam um número maior de *cores*. Nas seções em que o ganho de desempenho advindo do uso de mais *cores* não compensa o custo energético utiliza-se menos *cores*. A escalabilidade de uma seção depende de gargalos como limite da banda de memória. Em determinados casos o uso de poucos *cores* pode resultar em maior desempenho que usar todos os *cores* disponíveis [Curtis-Maury et al. 2008].

Dynamic Voltage and Frequency Scaling (DVFS) envolve o controle da frequência na qual determinados *cores* operam [Dongarra et al. 2011]. *Cores* com frequências mais baixas possuem menor consumo de potência, bem como menor velocidade de execução. A redução de consumo de potência está atrelada a características da micro-arquitetura. Dependendo dessas características, pode não ser possível controlar as frequências dos *cores* de um processador de forma independente. Esse caso é ilustrado na Figura 1.

O trabalho a ser desenvolvido utilizará métricas de roubo de tarefas, como tamanho da fila e dependência entre as tarefas, aliadas a métricas de desempenho disponibilizadas pelo processador para controlar, através de DCT e DVFS, o nível de paralelismo

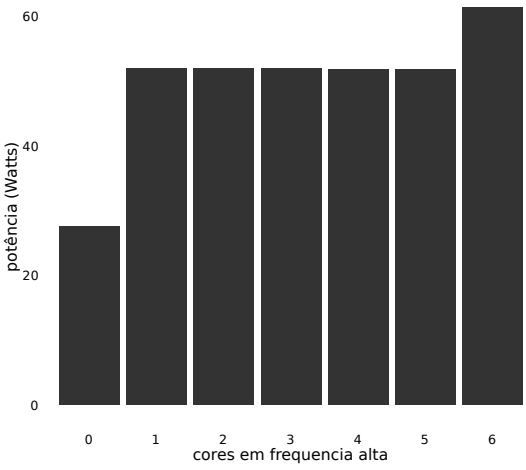

Figura 1. Consumo de potência em função de quantos dos 6 *cores* de um Intel Xeon E5-2630 estão em frequência alta (2.3GHz), enquanto os *cores* restantes estão em frequência baixa. Como as frequências não são independentes, a frequência utilizada é a mais alta das frequências requisitadas, limitando o uso de DVFS nessa micro-arquitetura.

em regiões paralelas mas com baixa escalabilidade. O contador da taxa de *cache misses*, por exemplo, permite verificar se a região atual possui um gargalo de memória que limita a sua escalabilidade. A implementação será feita sobre o Cilk Plus. Os resultados serão verificados por meio de conjuntos de *benchmarks* como o PBBS [Shun et al. 2012].

Referências

Curtis-Maury, M., Blagojevic, F., Antonopoulos, C., and Nikolopoulos, D. (2008). Prediction-based power-performance adaptation of multithreaded scientific codes. *Parallel and Distributed Systems, IEEE Transactions on*, 19(10):1396–1410.

Dongarra, J. et al. (2011). The international exascale software project roadmap. *International Journal of High Performance Computing Applications*, 25(1):3–60.

Kestor, G., Gioiosa, R., Kerbyson, D., and Hoisie, A. (2013). Quantifying the energy cost of data movement in scientific applications. In *Workload Characterization (IISWC), 2013 IEEE International Symposium on*, pages 56–65.

Li, D., de Supinski, B., Schulz, M., Nikolopoulos, D., and Cameron, K. (2013). Strategies for energy-efficient resource management of hybrid programming models. *Parallel and Distributed Systems, IEEE Transactions on*, 24(1):144–157.

Meuer, H., Strohmaier, E., Dongarra, J., and Simon, H. (2014). Top500. `http://www.top500.org`.

Shun, J., Blelloch, G. E., Fineman, J. T., Gibbons, P. B., Kyrola, A., Simhadri, H. V., and Tangwongsan, K. (2012). Brief announcement: The problem based benchmark suite. In *Proceedings of the Twenty-fourth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '12, pages 68–70, New York, NY, USA. ACM.

Subramaniam, B. et al. (2014). Green500. `http://www.green500.org`.

Torrellas, J. (2009). Architectures for extreme-scale computing. *Computer*, 42(11):28–35.