Avaliação de um Modelo de Consumo Energético Aplicado ao Mapeamento de Processos*

Edson L. Padoin, Eduardo H. M. Cruz, Francieli Z. Boito, Francis B. Moreira, Laércio L. Pilla, Rodrigo V. Kassick, Philippe O. A. Navaux

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

{elpadoin, ehmcruz, fzboito, fbmoreira,
llpilla, rvkassick, navaux}@inf.ufrgs.br

Abstract. *Nowadays, energy consumption arises as an essential aspect to the new generations of processor architectures and large-scale HPC systems. In such context, there is high demand for studies on how and where energy is spent on a computer system and which techniques can be used to increase it's efficiency. Energy-consumption models try to ease this analysis by providing theoretical values, but their precision is still not well studied. Therefore, this paper presents a study of the precision of an energy-consumption model for architecture with hierarchical cache and its appliance on the context ao process-mapping techniques on a real system.*

Resumo. *A redução do consumo energético é hoje um dos grandes desafios da computação, essencial para viabilizar as novas gerações de computadores e arquiteturas paralelas de grande escala. Nesse contexto, é fundamental o entendimento de onde e como a energia é aplicada, e quais técnicas podem ser empregadas para aumentar sua eficiência. A modelagem do consumo de energia visa facilitar esta análise, porém sua precisão com relação a arquiteturas reais ainda é pouco explorada. Sendo assim, este artigo visa estudar um modelo de consumo energético para arquiteturas com hierarquia de cache, de forma a aplicá-lo no contexto de mapeamento de processo em uma arquitetura real.*

1. Introdução

Sistemas de computação de alto desempenho são frequentemente avaliados por sua capacidade de computação – quantidades de operações por segundo, velocidade de comunicação, tempo para uma aplicação completar sua execução, dentre outros. No entanto, recentemente, há um interesse crescente no consumo energético destas arquiteturas e quão eficientes elas são nesse aspecto [Hsu and Feng 2005]. Isto se torna ainda mais evidente quando são levadas em consideração as características de novas arquiteturas de clusters que são propostas para os próximos anos, que têm por objetivo atingir o patamar de 10^{18} operações de ponto flutuante por segundo (exa-flop) [Donofrio et al. 2009].

Dadas as previsões de tecnologia disponível [Kogge et al. 2008] e a grande escala destes sistemas, os custos energéticos com processadores, armazenamento, rede de interconexão e refrigeração necessária são inviáveis. Portanto, torna-se essencial o

*Trabalho parcialmente apoiado por CNPq, CAPES, FAPERGS e FINEP

estudo do consumo energético de arquiteturas paralelas sob diferentes cargas de trabalho e, assim, investigar formas de aumentar sua eficiência energética. Nas arquiteturas de processadores *multicore*, por exemplo, o subsistema de memória se apresenta como o provedor de dados aos processadores, sendo um ponto essencial ao desempenho. Por esta razão, ele é constantemente adaptado às necessidades computacionais [Alves et al. 2009] para esconder a diferença de desempenho entre memórias e processadores [Hennessy and Patterson 2007].

A energia consumida pela hierarquia de memória é parte importante do consumo total do sistema. Portanto, uma otimização na hierarquia de memória e interconexão possui um impacto importante. Técnicas simples, como o ajuste de parâmetros das memória cache para uma aplicação específica, podem apresentar uma economia de até 60% do consumo de energia [Herranz and Moreno-Navarro 2003], podendo ser facilmente empregadas em processadores de uso específico. Entretanto, mecanismos mais sofisticados devem ser empregados em processadores de propósito geral, a fim de atingir um equilíbrio entre o desempenho e o custo energético.

O mapeamento de processos é uma técnica que pode ser aplicada de forma a aprimorar a afinidade de memória do sistema, adaptando o mapeamento da aplicação à arquitetura. Um dos parâmetros utilizados para o mapeamento de processos é o compartilhamento de dados entre as tarefas. Com isto, é possível guiar o posicionamento de tarefas para otimizar a utilização de caches compartilhadas e reduzir os sobrecustos das latências de comunicações remotas. Uma vez que o mapeamento melhora o desempenho de uma aplicação, o consumo de energia também sofrerá modificações. Surge então a necessidade de explorar como técnicas de otimização influenciam o consumo energético do sistema.

Porém, medições nem sempre podem ser feitas diretamente em sistemas reais, seja por falta de acesso físico ao recurso ou por este ainda estar em fase de projeto. Nestes casos, modelos de consumo energético podem ser aplicados para estimar o desempenho energético de aplicações, algoritmos e sistemas. Entretanto, estes modelos de consumo energético precisam ser avaliados para confirmar suas precisões e, consequentemente, aplicabilidade.

Neste contexto, este artigo apresenta um estudo de um modelo de consumo energético aplicado ao mapeamento de processos em processadores atuais, de forma a avaliar a precisão do modelo e o impacto do mapeamento de processos sobre o consumo energético da arquitetura. Por ter sido desenvolvido para uma arquitetura muito mais simples que as atuais, o modelo teve que ser adaptado para utilização em uma hierarquia de memória.

O restante do artigo está divido da seguinte maneira: a Seção 2 apresenta o modelo de consumo energético considerado. A Seção 3 introduz conceitos de mapeamento de processos. O método de avaliação utilizado é apresentado na Seção 4 e os resultados obtidos são abordados na Seção 5. Trabalhos relacionados são tratados na Seção 6. Por fim, a Seção 7 apresenta as principais contribuições deste artigo e trabalhos futuros.

2. Modelo de Consumo Energético da Hierarquia de Memória

Modelos especializados para diferentes componentes do sistema são necessários para a manutenção da precisão. Um exemplo disso é o modelo de consumo energético por

memória cache apresentado em [Shiue and Chakrabarti 1999]. Neste modelo, o consumo é diferenciado pela sua fonte, podendo esta ser das células de memória (E_{cell}), do decodificador (E_{dec}) ou da entrada e saída de dados (E_{io}). Assim, o consumo gerado por um dado encontrado na cache (*hit*) depende de E_{cell} e E_{io}, enquanto o de um dado não encontrado (*miss*) depende do custo de um acesso à memória principal.

Um novo modelo foi refinado em [Kim et al. 2000] considerando também os acessos de escrita. Essa nova versão do modelo é utilizada para comparação do consumo energético da hierarquia de memória de uma arquitetura multicore:

$$
\begin{aligned}
Energy &= E_{bus} + E_{cell} + E_{pad} + E_{main} \\
E_{bus} &= E_{add\_bus} + E_{data\_bus} \\
E_{cell} &= \beta \times (Word\_line\_size) \times (Bit\_line\_size + 4,8) \times (N_{hit} + 2 \times N_{miss}) \\
E_{pad} &= E_{add\_pad} + E_{data\_pad} \\
E_{main} &= Em \times 8L \times N_{miss} \times (1 + dirty\_r) \\
E_{add\_bus} &= 0,5 \times 10^{-12} \times Pr1 + V^2 \times (N_{hit} + N_{miss}) \times W_{add} \\
E_{data\_bus} &= 0,5 \times 10^{-12} \times Pr2 + V^2 \times (N_{hit} + N_{miss}) \times 32 \\
E_{add\_pad} &= 20 \times 10^{-12} \times Pr3 + V^2 \times N_{miss} \times W_{add} \\
E_{data\_pad} &= 20 \times 10^{-12} \times Pr4 + V^2 \times (1 + dirty\_r) \times N_{miss} \times 64 \\
Word\_line\_size &= m \times (8L + T + St) \\
Bit\_line\_size &= C/(m \times L)
\end{aligned}
$$

onde C = tamanho da cache; L = tamanho da linha da cache; m = associatividade; T = *tag size* em bits; St = número de bits de *status* por bloco; N_{hit} = número de *hits*; N_{miss} = número de *misses*; W_{add} = largura do barramento de endereço; $dirty\_r$ = porcentagem de blocos escritos de volta à memória na substituição; Em = energia consumida por um acesso à memória principal; V = *Voltage level*; *Pr1, Pr2, Pr3 e Pr4* = taxa de troca de bits nos barramentos; e β é uma constante que depende da tecnologia empregada.

Esse modelo não considera a existência de mais de um nível de cache, relacionando o gasto gerado pelas faltas na cache com o acesso à memória principal. Adicionalmente, adaptações são necessárias para a existência de mais de um banco de memória cache para um determinado nível (por exemplo, cada núcleo do processador pode ter uma cache L1 privada e compartilhar a L2 com os demais). Assim, nesse trabalho, é utilizado o seguinte modelo adaptado:

$$
\begin{aligned}
Energy &= Energy'(L_1) \\
Energy'(L_n) &= Energy'(L_{n+1}) + \sum_{i=0}^{i<qt(L_n)} (E_{bus}(L_{ni}) + E_{cell}(L_{ni}) + E_{pad}(L_{ni})) \\
Energy'(L_m) &= \sum_{i=0}^{i<qt(L_{m-1})} E_{main}(L_{m-1,i})
\end{aligned}
$$

Existem $(m+1)$ níveis de cache (L_1, L_2, ..., L_{m-1}); $qt(L_n)$ significa o número de caches no nível n; L_{ni} é a i-ésima memória cache de nível n; e L_m é a memória principal.

3. Mapeamento de Processos

O mapeamento de processos é uma técnica que possibilita o aumento de desempenho em aplicações paralelas através de uma alocação mais eficiente dos recursos. Uma característica importante a ser explorada nos sistemas multiprocessados é o tempo de comunicação entre núcleos – o tempo necessário para que um processador escreva em uma variável compartilhada e a alteração ser propagada para outro núcleo. Esse tempo de comunicação é frequentemente heterogêneo, pois depende da localização dos núcleos envolvidos. Este tempo é também influenciado pela hierarquia de memória cache, já que núcleos podem possuir compartilhamento de diferentes níveis de memória cache ou podem estar localizados em diferentes processadores [Cruz et al. 2011b].

Uma maneira de diminuir o tempo de execução de uma aplicação paralela é mapear threads que compartilham memória em núcleos que compartilham cache [Alves et al. 2009], reduzindo a latência de acesso à dados compartilhados. Além disso, é mais vantajoso manter tarefas comunicantes em núcleos em um mesmo socket do que em processadores distintos. Isso se deve não somente às latências já mencionadas, mas também porque isso reduz a sobrecarga imposta por protocolos de coerência.

A diminuição no tempo de execução das aplicações tende a resultar na diminuição do consumo energético do sistema. Adicionalmente, com um menor número de dados e mensagens de protocolos de coerência trafegando nas interconexões, bem como menos linhas duplicadas em caches distintas, uma diminuição no consumo de potência também é esperada. Isso pode ser obtido através do mapeamento eficiente de processos, tendo em vista que tais componentes são responsáveis por grande parte do consumo energético dos processadores atuais [Kumar et al. 2005].

O mapeamento pode ser feito de maneira dinâmica ou estática. No mapeamento dinâmico [Tam et al. 2007], a cada etapa de processamento as threads devem ser remapeadas para se adequarem ao novo padrão de compartilhamento de dados. Diferentemente, o mapeamento é predefinido no caso estático [Cruz et al. 2010], ou seja, uma vez iniciada a computação, o mapeamento não será mais modificado. Em ambos os casos, a decisão sobre o mapeamento deve ser tomada com base nas informações fornecidas pelo programador, pelo compilador ou outras técnicas que forneçam informações sobre acessos à memória. Neste trabalho, é adotada a política de mapeamento estático baseada em traços de memória capturados em execuções anteriores.

4. Método de Avaliação

Para analisar o consumo energético tanto no modelo apresentado na Seção 2 quanto em um caso real, foram utilizados os benchmarks da família NAS-NPB [Jin et al. 1999] versão 3.3.1 paralelizados com a interface OpenMP. Estes benchmarks foram escolhidos por representarem diversas famílias de aplicações paralelas. Foram executadas as aplicações CG, MG, EP, SP, LU, IS, FT e UA em suas instâncias de tamanho A e B.

Para cada teste, foram avaliadas as estratégias de mapeamento do sistema operacional (**Padrão**) e dois mapeamentos heurísticos baseados em quantidade de acessos a regiões de memória compartilhada (**Quantidade de Acessos**) e em quantidade de memória compartilhada (**Quantidade de Memória**). Para os mapeamentos heurísticos, foram avaliados os melhores e piores mapeamentos.

4.1. Consumo Energético na Arquitetura Real

Para a avaliação em um ambiente real foi utilizado um nó de processamento Dell Poweredge R710 com dois processadores Xeon E5530 de 2.40GHz. Cada processador contém quatro núcleos capazes de execução simultânea de duas threads (HT). A arquitetura desta máquina é apresentada na Figura 1.

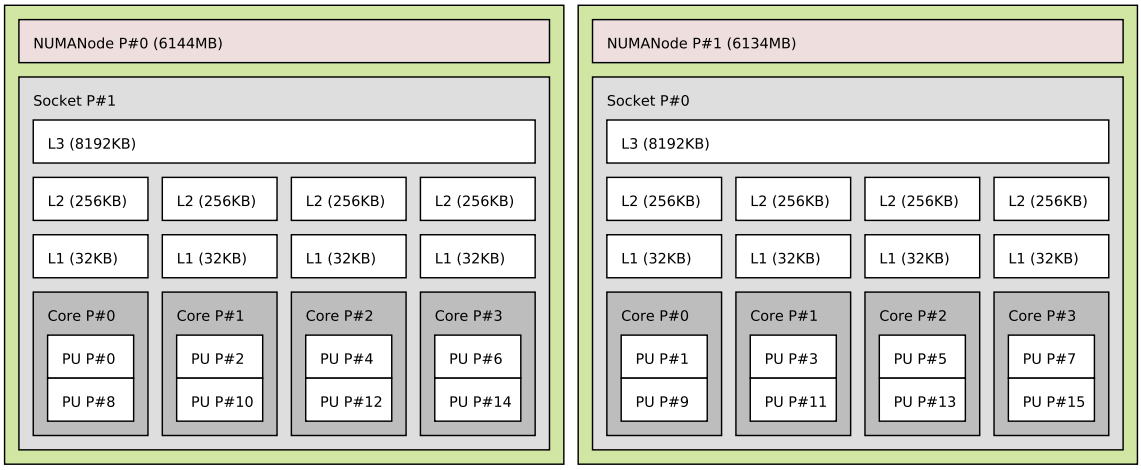


Figura 1. Arquitetura da máquina de teste

A medição de consumo foi feito com a interface IPMI (*Intelligent Platform Management Interface*) através da ferramenta FreeIPMI[1]. Foram feitas quatro medições por segundo durante 10 execuções de cada benchmark.

4.2. Alimentação do Modelo de Consumo Energético

Para obter os dados necessários ao modelo, os benchmarks foram novamente executados no ambiente previamente descrito e foram capturadas informações do processador com o auxílio da ferramenta Vtune[2], extraindo valores para contadores de *hits* e *misses* de cache, entre outros. Os dados restantes foram obtidos através da especificação da arquitetura e da ferramenta CACTI[3] versão 5.3.

Dado que não foi utilizado nenhum simulador para a execução, as taxas de troca de bits (Pr1, Pr2, Pr3 e Pr4 no modelo, como descrito na Seção 2) não puderam ser determinadas. Dessa forma, foi assumido um valor de 25% para elas, de forma semelhante ao feito em [Kim et al. 2000] [Hicks et al. 1997].

A constante β, que depende da tecnologia de integração, também não pode ser determinada. Por isso, seu valor foi escolhido conforme o especificado em [Kim et al. 2000], valendo 1.44×10^{-14}. Portanto, foi assumida uma previsão teórica pessimista em seu valor bruto. Porém, o erro adicionado por isso age de forma uniforme em todas as medidas e não compromete as comparações.

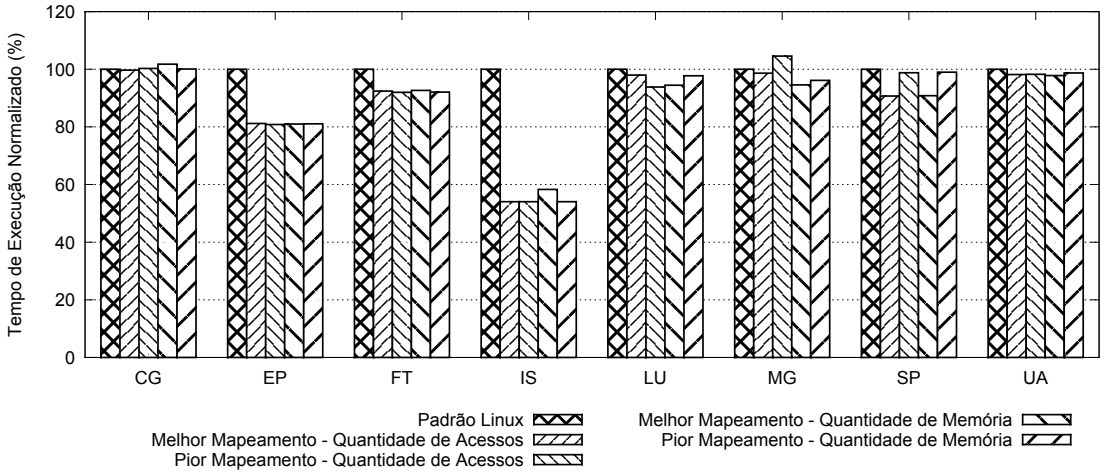
5. Resultados

A Figura 2 contém os tempos de execução normalizados dos benchmarks NPB para os tamanhos de entrada A e B. A normalização dos tempos foi feita em relação ao tempo

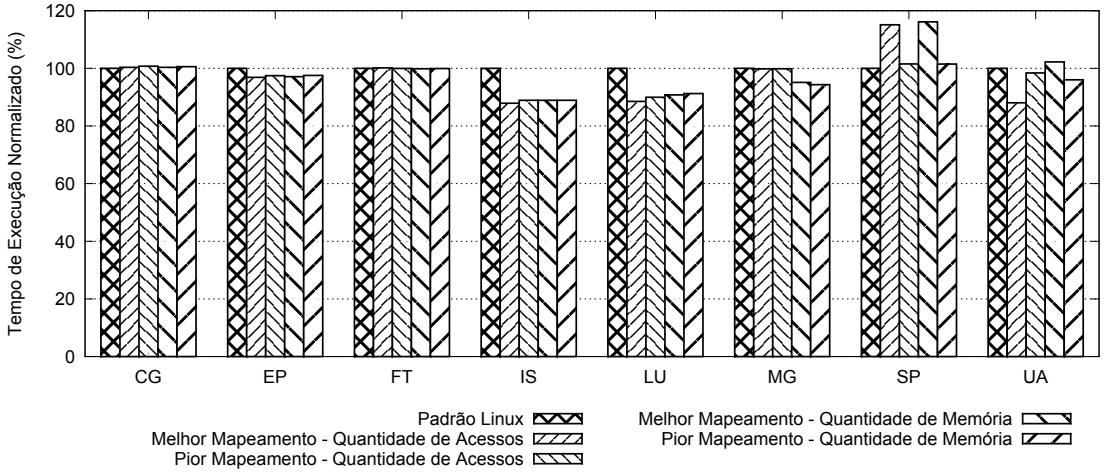
[1]http://www.gnu.org/software/freeipmi/

[2]http://software.intel.com/en-us/articles/intel-vtune-amplifier-xe/

[3]http://quid.hpl.hp.com:9081/cacti/



(a) Tamanho A.



(b) Tamanho B.

Figura 2. Tempos de execução dos benchmarks NPB normalizados.

de execução original de cada benchmark. Duas diferentes heurísticas de mapeamento foram usadas, onde a diferença entre elas está na métrica utilizada na tomada de decisões: uma é baseada na quantidade de acessos realizados à memória compartilhada enquanto a outra é baseada na quantidade de memória compartilhada. Além disso, na Figura 2, são apresentados o melhor e o pior mapeamento estático feito com cada heurística.

Os resultados mostram que o escalonador original do sistemas operacional Linux possui desempenho inferior às políticas de mapeamento experimentadas para todos os benchmarks, com exceção do benchmark SP com tamanho B. Isto acontece porque o escalonador do Linux realiza diversas migrações de tarefas entre os núcleos disponíveis a cada região paralela da aplicação, o que resulta em sobrecustos ligados a uma maior taxa de faltas na cache, já que os dados necessitam ser movimentados entre as caches. O mesmo comportamento pode ser observado em [Cruz et al. 2010].

Entretanto, as diferenças entre os tempos de execução entre o pior e melhor mapeamento apresentam um padrão diferente do esperado. Em várias situações, o pior mapeamento obtém um ganho de desempenho superior ao do melhor mapeamento. A razão para estes resultados divergentes é em função de a máquina utilizada para os experimentos possui apenas um nível na hierarquia de memória cache. Quando mais níveis compartilhados estão presentes, maior é o ganho de desempenho, como pode ser verificado em [Cruz et al. 2011a], onde ganhos de até 75% foram obtidos ao se utilizar técnicas de mapeamento em máquinas NUMA.

	A					B				
	Padrão	**Quantidade de Acessos**		**Quantidade de Memória**		**Padrão**	**Quantidade de Acessos**		**Quantidade de Memória**	
		melhor	**pior**	**melhor**	**pior**		**melhor**	**pior**	**melhor**	**pior**
cg	0,75	0,27	0,20	0,19	0,27	14,07	14,10	14,12	13,95	14,01
ep	1,48	1,50	1,50	1,49	1,50	6,01	6,00	5,94	6,01	5,94
ft	1,02	0,92	0,92	0,93	0,84	11,06	11,04	11,05	11,05	10,97
is	0,32	0,12	0,19	0,19	0,12	0,65	0,65	0,65	0,66	0,65
lu	9,78	6,15	6,23	6,13	6,30	29,08	27,34	27,64	27,62	27,76
mg	0,61	0,56	0,49	0,57	0,49	2,85	1,77	1,76	1,76	1,84
sp	8,74	7,88	7,95	7,89	8,39	48,96	47,65	47,78	49,76	52,28
ua	9,88	6,22	6,37	6,21	6,44	30,90	30,35	30,56	30,27	30,58

Tabela 1. Consumo (Watts-hora) para execução do NPB com mapeamento em máquina real

As Tabelas 1 e 2 apresentam os valores de consumo de energia e de potência média para a execução dos testes cujos resultados foram apresentados na Figura 2. Para os testes com tamanho A, houve em alguns casos uma diminuição no consumo, apesar de os resultados de desempenho apresentarem pouca diferença em tempo de execução.

Nos testes com tamanho B, no entanto, onde a variação de consumo não foi desprezível, há pouca correlação com as diferenças nos tempos de execução. O teste IS, por exemplo, teve seu tempo de execução diminuído para aproximadamente 90% do tempo obtido com o escalonador padrão do SO. Seu consumo de energia, no entanto, manteve-se idêntico para todos os casos. Já o teste SP apresentou diferenças em seu consumo energético em função do maior ou menor tempo de execução.

Estes resultados levam a crer que, apesar da pouca diferença em desempenho ob-

	A					B				
	Padrão	**Quantidade de Acessos**		**Quantidade de Memória**		**Padrão**	**Quantidade de Acessos**		**Quantidade de Memória**	
		melhor	**pior**	**melhor**	**pior**		**melhor**	**pior**	**melhor**	**pior**
cg	226,49	192,03	176,13	170,00	240,69	249,58	252,58	252,88	248,53	252,20
ep	265,92	269,33	269,46	268,03	269,71	267,19	269,79	270,79	270,59	270,59
ft	245,83	253,93	253,56	258,00	253,16	256,84	258,11	258,33	258,30	256,52
is	228,94	141,17	225,87	228,11	211,44	233,78	233,60	235,73	237,49	235,64
lu	255,15	283,83	283,91	282,85	283,59	276,18	283,66	282,66	281,73	282,30
mg	243,26	252,67	250,08	258,06	221,98	244,70	264,98	253,50	254,16	264,33
sp	262,26	267,46	267,57	268,03	264,99	256,95	257,94	257,86	255,93	255,05
ua	254,11	269,75	269,77	269,55	269,45	266,81	268,49	268,32	268,39	268,47

Tabela 2. Potência Média para execução do NPB com mapeamento em máquina real

tida com o mapeamento, o consumo de energia pode ser significativamente afetado por outras variáveis externas ao nó de processamento, sendo necessárias maneiras mais refinadas para avaliar o impacto de otimizações no consumo de energia.

Os valores obtidos com o modelo teórico no tamanho B do benchmark são apresentados na Figura 3. Nota-se que os resultados obtidos possuem escala de grandeza maior do que foi observado com IPMI. Conforme discutido anteriormente, isso era esperado devido aos parâmetros do modelo que foram estimados usando valores adequados à situação tecnológica de mais de 10 anos atrás.

A Figura 4 apresenta os mesmos dados que a 3(b), porém normalizados a partir do valor real de consumo do benchmark SP para facilitar a comparação. O modelo mostra, assim como nas medidas reais, pouca diferença entre os consumos das estratégias de mapeamento.

No entanto, é possível notar que as relações entre o consumo de diferentes benchmarks são quase totalmente mantidas, sendo SP o teste com maior consumo e IS o de menor valor. A única exceção observada foi para o caso do teste MG que, no modelo teórico apresentou consumo mais elevado que o EP quando o inverso foi observado no caso real.

6. Trabalhos Relacionados

O consumo energético de sistemas computacionais tem sido alvo de pesquisa, principalmente em ambientes de alto desempenho. Dentre as diferentes abordagens para o aumento da eficiência energética, estão o controle de voltagem e frequência (*Dynamic Voltage and Frequency Scaling*, DVFS) dos processadores e a aplicação de otimizações nas hierarquias de memória. Para estimar o consumo desses sistemas, usualmente são empregados contadores de hardware [Bellosa 2000, Bircher et al. 2005].

Em [Gordon-Ross et al. 2005] é discutida a redução do espaço de busca para otimização da melhor configuração de cache em relação ao consumo de energia. Já em [Silva-Filho et al. 2006], onde é utilizado o mesmo modelo, é analisado, além do consumo de energia, o desempenho computacional obtido com as novas configurações de memória. Abordagem semelhante é encontrada em [Kim et al. 2000], onde são relacionados trabalhos que apresentam técnicas para diminuição do consumo de energia em

memória cache.

No trabalho de [Hicks et al. 1997] é analisado como o aumento da complexidade das caches, com configurações diferentes de associatividade, tamanho do bloco e tamanho de cache apresentam impacto na energia consumida. A taxa de acerto também é analisada frente a energia consumida. Os autores concluem que as taxas de acerto também desempenham um papel chave na avaliação da eficiência energética. Neste trabalho também é apresentado equações de um modelo de consumo energético que leva em consideração as células de memória, o decodificador e a entrada e saída.

Os autores de [Zhang and Vahid 2003] buscam estimar a energia consumida para uma determinada configuração de cache, envolvendo diferentes tamanhos de blocos e associatividade. Exploram, a partir de equações o consumo de energia, para um conjunto de configurações que impactam no desempenho e tem influência no consumo.

No artigo [Zarandi and Miremadi 2005] novas caches são propostas para melhorar as taxas de sucesso em relação aos regimes convencionais. O mapeamento proposto é simulado tendo como base alguns padrões de rastreamento. Os resultados apontam um aumanto na quantidade de *hits* em relação aos esquemas de mapeamentos atuais, minimizando assim o consumo de energia.

7. Conclusão

Devido às características e tamanho das arquiteturas propostas para os próximos anos, há um crescente interesse na avaliação e controle do consumo energético dos sistemas computacionais. Uma forma de fazê-lo acontece através da modelagem dos fenômenos elétricos que acontecem nestas máquinas. Porém, pouco esforço tem sido colocado na validação destes modelos.

Nesse contexto, este trabalho apresentou os resultados da avaliação do consumo energético da hierarquia de memória de uma arquitetura multicore e sua comparação com um modelo de consumo energético teórico. Para tanto, alguns parâmetros do modelo precisaram ser adaptados para o mesmo ser aplicável às hierarquias de memórias atuais.

Para realizar a avaliação, foi empregado uma técnica de mapeamento de processos utilizando duas métricas para decisão: quantidade de acessos e quantidade de memória. Esta técnica de mapeamento utilizada não apresentou ganhos de desempenho para as aplicações do NPB na arquitetura utilizada. O principal motivo para tal comportamento é que a máquina utilizada nos experimentos possui apenas um nível de hierarquia de memória a ser explorada.

Os resultados obtidos pelas medições de consumo de energia com o IPMI não apresentaram ganhos com o mapeamento. Além do mesmo motivo apontado para o desempenho, outro fator que contribui para tais resultados é o fato de que o IPMI mede o consumo de todo o sistema não permitindo um monitoramento do gasto de energia das memória cache, interconexões e outros componentes de forma isolada.

Adicionalmente, pôde-se observar que o modelo teórico é capaz de captar o comportamento geral de gasto de energia na situação de mapeamento, ainda que a escala não corresponda ao caso real. O erro na escala se deve à constante β do modelo, que foi estimada com o mesmo valor que o utilizado no artigo original. No entanto, o artigo original

se baseia em uma tecnologia de $800nm$, enquanto a tecnologia presente no ambiente de testes é de $45nm$.

Como trabalhos futuros, planeja-se monitorar o consumo de forma isolada de cada componente da arquitetura para melhor compreender o impacto de técnicas de otimizações nos diferentes níveis do processador. O comportamento divergente do benchmark MG também deverá ser estudado para identificar que parâmetros que influenciaram em seu consumo de energia que não foram captados pelo modelo.

Referências

Alves, M. A. Z., Freitas, H. C., and 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.

Bellosa, F. (2000). The benefits of event: driven energy accounting in power-sensitive systems. In *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*, pages 37–42. ACM.

Bircher, W., Valluri, M., Law, J., and John, L. (2005). Runtime identification of microprocessor energy saving opportunities. In *Proceedings of the 2005 international symposium on Low power electronics and design*, pages 275–280. ACM.

Cruz, E. H., Alves, M. A., and Navaux, P. O. (2010). Process mapping based on memory access traces. In *Computing Systems (WSCAD-SCC), 2010 11th Symposium on*, pages 72 –79.

Cruz, E. H., Ribeiro, C. P., Alves, M. A., Carissimi, A., Mehaut, J., and Navaux, P. O. (2011a). Using memory access traces to map threads and data on hierarchical multicore platforms. In *25th IPDPS 13th Workshop on Advances in Parallel and Distributed Computational Models*.

Cruz, E. H. M., Padoin, E. L., Boito, F. Z., and Navaux, P. O. A. (2011b). Avaliando consumo de energia no mapeamento de processos. In *XI Escola Regional de Alto Desempenho (ERAD 2011)*, volume IX, pages 93–94. SBC.

Donofrio, D., Oliker, L., Shalf, J., Wehner, M. F., Rowen, C., Krueger, J., Kamil, S., and Mohiyuddin, M. (2009). Energy-efficient computing for extreme-scale science. *Computer*, 42:62–71.

Gordon-Ross, A., Vahid, F., and Dutt, N. (2005). Fast configurable-cache tuning with a unified second-level cache. In *Proceedings of the 2005 international symposium on Low power electronics and design*, pages 323–326. ACM.

Hennessy, J. L. and Patterson, D. A. (2007). *Computer Architecture: A Quantitative Approach*. Elsevier, USA, 4th edition.

Herranz, A. and Moreno-Navarro, J. (2003). Cache Configuration Exploration on Prototyping Platforms. In *Proceedings of the 14th IEEE International Workshop on Rapid System Prototyping (RSP'03)*, page 164. IEEE Computer Society.

Hicks, P., Walnock, M., and Owens, R. (1997). Analysis of power consumption in memory hierarchies. In *Proceedings of the 1997 international symposium on Low power electronics and design*, pages 239–242. ACM.

Hsu, C.-h. and Feng, W.-c. (2005). A power-aware run-time system for high-performance computing. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, SC '05, pages 1–, Washington, DC, USA. IEEE Computer Society.

Jin, H., Frumkin, M., and Yan, J. (1999). The openmp implementation of nas parallel benchmarks and its performance. In *Technical Report: NAS-99-011*.

Kim, H. S., Narayanan, V., Kandemir, M., and Irwin, M. J. (2000). Multiple access caches: Energy implications. In *Proceedings of the IEEE Computer Society Annual Workshop on VLSI (WVLSI'00)*, WVLSI '00, pages 53–, Washington, DC, USA. IEEE Computer Society.

Kogge, P., Bergman, K., Borkar, S., Campbell, D., Carson, W., Dally, W., Denneau, M., Franzon, P., Harrod, W., Hill, K., and Others (2008). Exascale computing study: Technology challenges in achieving exascale systems. Technical report, University of Notre Dame, CSE Dept.

Kumar, R., Zyuban, V., and Tullsen, D. M. (2005). Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In *Proceedings of the 32nd annual international symposium on Computer Architecture*, ISCA '05, pages 408–419, Washington, DC, USA. IEEE Computer Society.

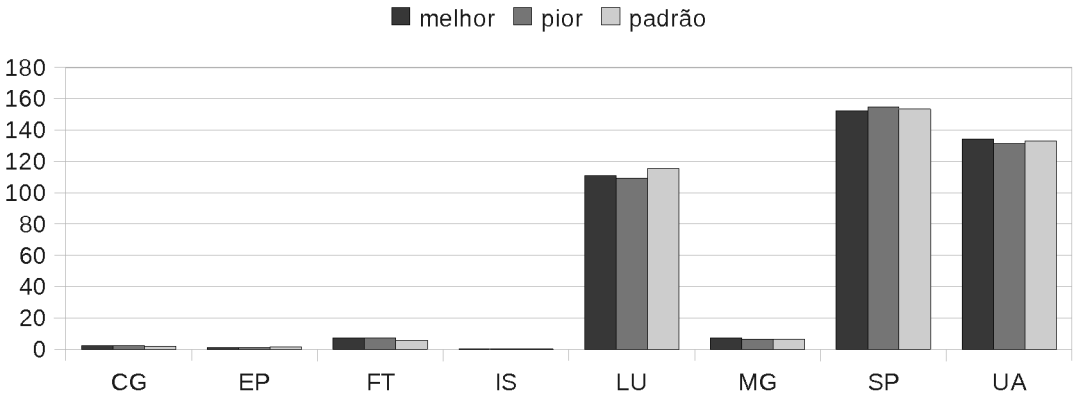
Shiue, W. and Chakrabarti, C. (1999). Memory exploration for low power, embedded systems.

Silva-Filho, A., Cordeiro, F., Sant Anna, R., and Lima, M. (2006). Heuristic for two-level cache hierarchy exploration considering energy consumption and performance. *Lecture notes in computer science*, 4148:75.

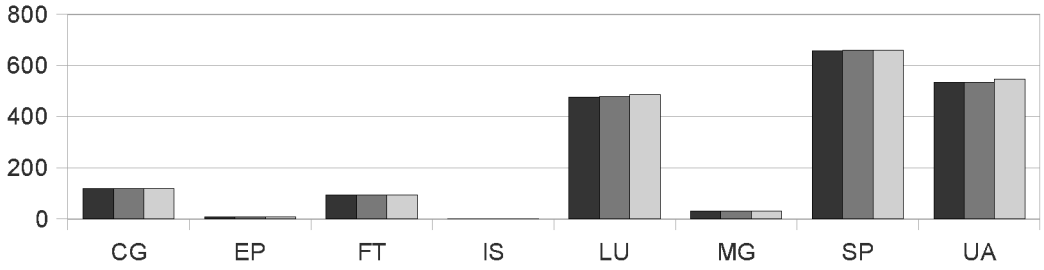
Tam, D., Azimi, R., and Stumm, M. (2007). Thread clustering: sharing-aware scheduling on smp-cmp-smt multiprocessors. *SIGOPS Oper. Syst. Rev.*, 41(3):47–58.

Zarandi, H. and Miremadi, S. (2005). Hierarchical multiple associative mapping in cache memories.

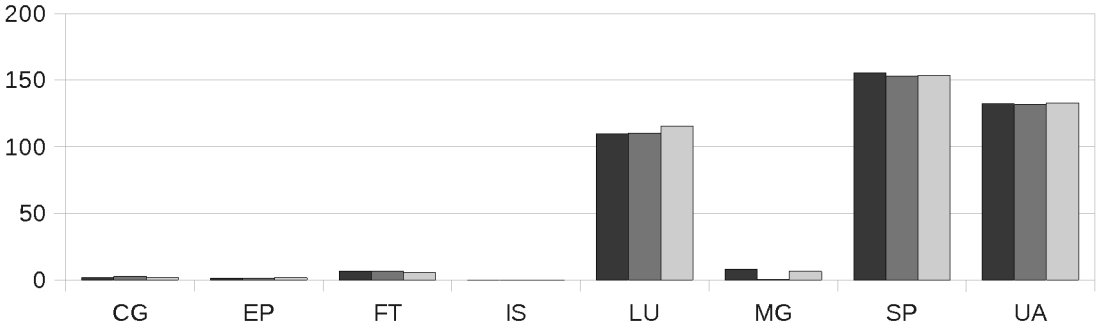
Zhang, C. and Vahid, F. (2003). Cache configuration exploration on prototyping platforms.



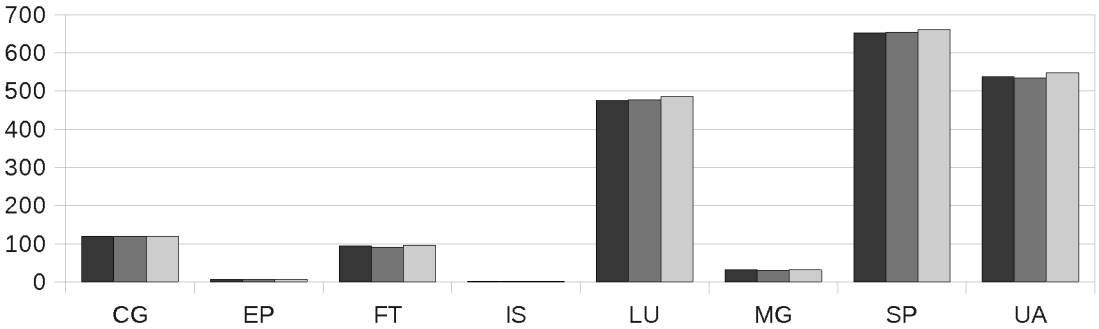
(a) Tamanho A, mapeamento por quantidade de acessos



(b) Tamanho B, mapeamento por quantidade de acessos



(c) Tamanho A, mapeamento por quantidade de memória



(d) Tamanho B, mapeamento por quantidade de memória

Figura 3. Consumo de energia teórico em watts-hora

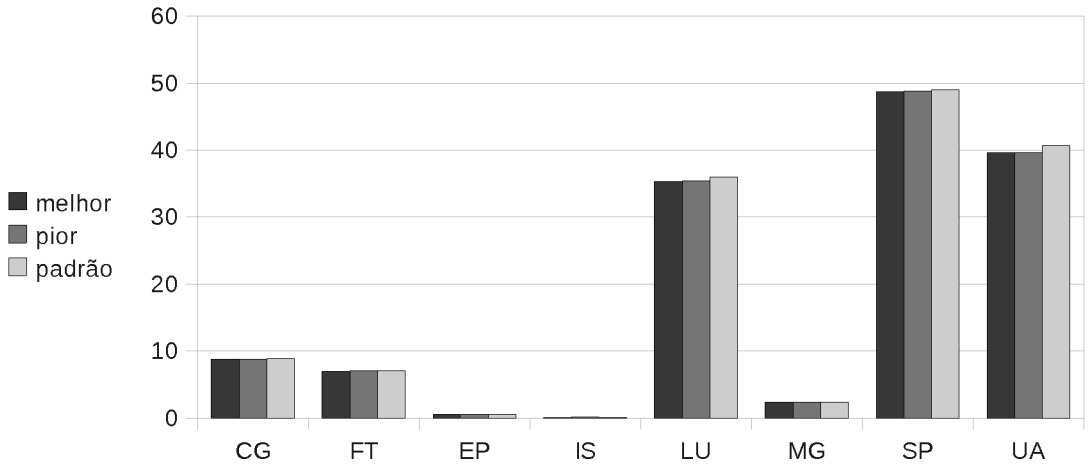


Figura 4. Consumo de Energia (Wh) de acordo com o modelo teórico – Normalizado com o caso real. Tamanho B, Mapeamento por Quantidade de Acessos