Predição de Consumo de Energia de Aplicações OpenMP Multithreads

Fellipe Queiroz[1], **Erick Damasceno**[1], **Marcos Amaris**[1]

[1]Universidade Federal do Pará
Faculdade de Engenharia de Computação, Tucuruí - Pará

{fellipe.queiroz,erick.silva}@tucurui.ufpa.br, amaris@ufpa.br

Resumo. *Investigamos o consumo de energia em benchmarks OpenMP com o objetivo de otimizar o desempenho em hardware de alto desempenho. Utilizamos técnicas de regressão linear e polinomial para prever o consumo de energia. Os resultados mais promissores incluem um erro de previsão de apenas 0.85% para o benchmark LavaMD no conjunto de benchmarks Rodinia e 0.94% para o benchmark Cholesky no conjunto Polybench.*

Abstract. *We investigate energy consumption in OpenMP benchmarks with the aim of optimizing performance on high-performance hardware. We employ linear and polynomial regression techniques to predict energy consumption. The most promising results include a prediction error of only 0.85% for the LavaMD benchmark in the Rodinia benchmark suite and 0.94% for the Cholesky benchmark in the Polybench suite.*

1. Introdução

Green Computing é uma área de pesquisa que se concentra em tornar a computação mais sustentável e amigável ao meio ambiente. Atualmente, o consumo de energia em sistemas computacionais é um assunto ativo e de importante discussão nas comunidades de pesquisa [Cordeiro et al. 2023]. A chegada das máquinas multi-core pós-era de *Moore* revolucionou o cenário da computação, proporcionando um aumento significativo no desempenho e eficiência energética.

As plataformas paralelas disponíveis atualmente estão se tornando cada vez mais paralelas e heterogêneas. Uma ferramenta essencial nesse cenário é o padrão de programação "OpenMP", uma extensão para programação paralela em linguagens como C, C++, e Fortran, amplamente utilizada para tirar proveito de sistemas com memória compartilhada. O OpenMP permite a criação de programas paralelos, nos quais uma tarefa é dividida em várias *threads* de execução que podem ser simultaneamente processadas por múltiplos núcleos de CPUs ou GPUs.

Esse trabalho tem como objetivo aplicar técnicas de análise de dados e regressão para estimar o consumo de energia em ambientes que fazem uso do padrão **OpenMP** para programação *multithread*. Assim, este projeto busca fornecer *insights* valiosos para otimização de recursos e economia de energia, tornando os sistemas paralelos mais eficientes e sustentáveis.

A seguir, a Seção 2 apresenta alguns trabalhos relevantes para a compreensão desse trabalho. a Seção 3 expõe a metodologia. Seção 4 explica os resultados e finalmente na Seção 5 são apresentadas as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

[Krause et al. 2017] analisaram o impacto das diferentes versões de compiladores no consumo de energia de aplicações paralelas, com foco nas aplicações NQueens e Graph500. O estudo destacou que a escolha do compilador deve ser baseada no tipo de aplicação, e o GCC 6.2.0 foi identificado como a escolha mais vantajosa de forma geral. Esses resultados podem servir como base para futuros estudos que explorem a relação entre o consumo de energia e a seleção do compilador em aplicações paralelas.

[Amaris et al. 2023] discutem várias abordagens para predição de tempo de consumo de funções CUDA executadas em placas gráficas de propósito geral (GPGPU). Eles explicam diferentes que há aplicações regulares onde modelos analíticos podem ser usados para prever desempenho, e outras mais irregulares onde técnicas de aprendizado de máquina obtém melhores predições.

[Anjos and Fazenda 2023] fazem uma análise objetiva do consumo de energia de padrões de programação comuns em aplicativos científicos que exigem alto processamento. O estudo também envolveu a variação na quantidade de threads (linhas de execução) usadas e nas opções de compilação, com o propósito de identificar os impactos que essas mudanças têm na eficiência energética.

3. Metodologia

3.1. Componentes de Hardware

Nesta pesquisa, dois sistemas diferentes foram usados para realizar os experimentos. Na primeira configuração, foi usado um processador Intel Core i7 5500u, com frequência de 2.4GHz (turbo até 3.00GHz), 2 núcleos físicos e 4 lógicos. O sistema tinha três níveis de cache: L1 de 64KB, L2 de 256KB e L3 compartilhado de 4MB. A memória do sistema era de 10GB DDR3L a 1600MHz.

Na segunda configuração, utilizou-se um processador Intel Core i5-7200U com 2 núcleos físicos e 4 lógicos, frequência base de 2.50 GHz (turbo até 3.10 GHz), e uma memória cache de 128 KB de cache L1, 512 KB de cache L2 e 3 MB de cache L3. A memória do sistema era de 12GB DDR4 a 2133 MHz.

3.2. Componentes de Software

Ambas as plataformas trabalham com o sistema operacional Linux Ubuntu versão 22.4 e utilizaram o compilador GCC na versão 11.4.0. Os benchmarks empregados incluíram o suíte Rodinia e o PolyBench ACC, proporcionando uma avaliação abrangente do desempenho em ambientes heterogêneos.

As aplicações selecionadas para aferir o consumo de energía da suíte Rodinia estão abaixo à esquerda e à direita as aplicações do PolyBench ACC.

Aplicações do Rodinia

- LU DECOMPOSITION
- BACK PROPAGATION
- STREAMCLUSTER
- LAVA MD
- MYOCYTE

Aplicações do PolyBench-ACC

- 3MM
- CHOLESKY
- FDTD-APML
- CONVOLUTION-2D
- CORRELATION
- COVARIANCE

A coleta de dados envolveu a execução desses *benchmarks* utilizando a api de monitoramento de consumo da **PowerApi** chamada *Jouleit*. Foram criados *scripts* específicos utilizando *bash* para cada *benchmark*, ajustando gradualmente os valores de entrada para aumentar a carga de trabalho de acordo com a capacidade do sistema.

Este estudo manteve aproximadamente 100 amostras por benchmark, buscando testar o sistema de forma significativa. Resultou na criação de um amplo conjunto de dados em arquivos CSV, com dados organizados em colunas distintas e categorizados com base em atributos como CORE, CPU, DRAM, DURATION, UNCORE e INPUT. Todas as colunas foram consideradas relevantes para análise, exceto EXIT_CODE, que sinalizava possíveis falhas na coleta. As métricas de consumo de energia em CORE, CPU, DRAM e UNCORE foram inicialmente registradas em micro joules, enquanto a métrica DURATION foi capturada em microssegundos. Posteriormente, essas métricas foram convertidas para joules e segundos, respectivamente, para facilitar a representação visual do sistema.

Um script em Python com regressão linear e polinomial utilizando a biblioteca scikit learning foi desenvolvido para prever amostras futuras, evitando a necessidade de coleta adicional de dados após o hardware ter atingido seu limite durante o estudo. Essa abordagem permitiu uma análise sólida do desempenho do sistema. Utilizamos 80% e 60% respectivamente dos dados coletados, para treinar os modelos, com o propósito de fazer previsões para as 10 amostras mais recentes. Em seguida, empregamos a métrica MAPE (Mean Absolute Percentage Error) para avaliar a taxa de erro das previsões.

4. Resultados

4.1. Rodinia

Ao analisar os dados, inicialmente dividimos os benchmarks com base em padrões observáveis nos perfis de consumo relacionados a diferentes entradas de dados. Identificamos três classes distintas. A primeira classe engloba benchmarks com gastos exponenciais, exemplificados por Lava MD e Lu Decomposition. A segunda classe mostra curvas de consumo semi-lineares, representadas por Myocyte e Back Propagation. Por fim, a terceira classe apresenta padrões irregulares de consumo, com Stream Cluster sendo o único representante. Esses padrões também são refletidos nas métricas relacionadas à memória RAM e ao tempo de execução, dependendo das entradas de dados utilizadas.

Para os perfis exponenciais, aplicamos um polinômio de grau 3, obtendo uma taxa de erro de 0.85% para ”LavaMD”e 3.62% para ”LUD”. Já para os perfis lineares, utilizamos um polinômio de grau 1, alcançando um erro de 0.97% para ”Myocyte”e 2.64% para ”BackProp”, respectivamente. No caso do perfil irregular do ”stream Cluster”, aplicamos um polinômio de grau 1, resultando em um erro de 11.22%, e um polinômio de grau 3, que gerou um erro de 48.65%.

4.2. Polybench

Os resultados obtidos, revelaram que entradas divisíveis por 128 resultam em maior consumo de energia. Isso pode ser atribuído ao uso de arrays no código das aplicações, afetando o gerenciamento de memória e aumentando o consumo de energia. A CPU foi identificada como o principal consumidor de energia, enquanto o componente DRAM contribuiu com menos consumo. A variação no crescimento do consumo de energia entre

diferentes aplicações também foi observada. Esses resultados enfatizam a necessidade de otimizar o tratamento de arrays e implementar estratégias de gerenciamento de energia focadas na CPU.

Realizando previsões com regressões polinomiais de grau 3 e utilizando 60% dos dados disponíveis, obtivemos os seguintes resultados: o benchmark "3mm"apresentou um erro percentual médio absoluto (MAPE) de 7,8%, indicando boa precisão; o "Correlation"registrou 5,1%, sugerindo previsão precisa com baixo erro; "Covariance"teve 10,3%, enquanto "Cholesky"obteve o menor erro, apenas 0,94%. O benchmark "Fdtd-apml"registrou um MAPE de 3,2%. Adicionalmente, aplicando regressão linear para prever o consumo de energia em "Convolution-2d", inicialmente obteve-se um MAPE de 15,1%. Porém, ao utilizar 80% dos dados, esse valor foi reduzido para 8,5%.

5. Conclusão e Trabalho futuro

A pesquisa forneceu informações valiosas sobre o consumo de energia em aplicações de alto desempenho. Observou-se padrões que resultam num maior consumo de energia, provavelmente devido à ineficiência da gestão da memória e processador, ocasionando o aumento do consumo de energia. Além disso, os benchmarks apresentaram padrões de consumo distintos, exigindo abordagens de otimização específicas.

Em futuros trabalhos, recomenda-se explorar a relação entre consumo de energia e uso de memória RAM e CPU, desenvolvendo estratégias de otimização para reduzir o consumo energético. Também é importante focar em estratégias para melhorar a eficiência da CPU e continuar analisando o comportamento das aplicações para otimização específica. Essas implementações podem reduzir custos e promover uma computação de alto desempenho mais sustentável, alinhada à eficiência energética e conservação de recursos.

Referências

[Amaris et al. 2023] Amaris, M., Camargo, R., Cordeiro, D., Goldman, A., and Trystram, D. (2023). Evaluating execution time predictions on gpu kernels using an analytical model and machine learning techniques. *JPDC*, 171:66–78.

[Anjos and Fazenda 2023] Anjos, P. and Fazenda, A. (2023). Custo energético em computação de alto desempenho. In *Anais da XIV Escola Regional de Alto Desempenho de São Paulo*, pages 45–48, Porto Alegre, RS, Brasil. SBC.

[Cordeiro et al. 2023] Cordeiro, D., Francesquini, E., Amarís, M., Castro, M., Baldassin, A., and Lima, J. (2023). Green cloud computing: Challenges and opportunities. In *Anais do XIX SBSI*, pages 129–131, Maceió/AL. SBC.

[Krause et al. 2017] Krause, A. M., Moro, G. B., and Schnorr, L. M. (2017). Análise do consumo energético de aplicações paralelas com diferentes versões de compiladores. In *Anais da XVII ERAD-RS*. SBC.