Computação Verde na Camada de Aplicação

Thales Areco Bandiera Paiva
Orientador: Alfredo Goldman vel Lejbman
Instituto de Matemática e Estatística da Universidade de São Paulo

Introdução

Computação Verde é o campo que estuda meios para minimizar o impacto ambiental de sistemas computacionais. Um dos focos importantes deste campo é o uso eficiente de recursos computacionais. [1]

Grande parte do trabalho em eficiência energética aposta em técnicas para hardware, sistema operacional e virtualização. Porém, há poucos trabalhos sobre computação verde na camada de aplicação e em geral são baseados em otimizações para performance, dada a alta correlação entre performance e consumo energético.

Fizemos 4 simples experimentos na camada de aplicação com o objetivo de encontrar oportunidades de economia de energia ou validar resultados obtidos por outros pesquisadores para nossa plataforma.

Contribuições por Experimento

1 Mostra que o perfil de consumo de potência de uma aplicação varia de acordo com o uso que esta faz de recursos do sistema.

2 Mostra que um algoritmo pode ter menor consumo energético que outro apesar de ter pior performance. Como [2], sugere que aplicações adaptáveis que utilizem diferentes algoritmos dependendo do contexto podem ter maior eficiência energética.

3 Valida seis *code smells* como sendo responsáveis por aumentar o consumo energético de aplicações. Porém, diferente de [3], o aumento é causado principalmente pelo aumento do tempo de execução e não pelo aumento do consumo de potência.

4 Explora oportunidades de economia energética no agendamento paralelo de processos. Sugere ser possível economizar energia pela utilização de um agendador de processos alternativo.

Referências

[1] Murugesan, San. "Harnessing green IT: Principles and practices."IT professional 10.1 (2008): 24-33.

[2] Bunse, Christian, et al. "Choosing the"Best"Sorting Algorithm for Optimal Energy Consumption."ICSOFT (2). 2009.

[3] Vetro, Antonio, et al. "Definition, Implementation and Validation of Energy Code Smells: an Exploratory Study on an Embedded System."ENERGY 2013, The Third International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies. 2013.

Mais informações

- E-mail: thalespaiva@gmail.com
- Repositório Git: github.com/thalespaiva/mac0499

Agradecimentos

- Ao Professor Hermes Senger por ter gentilmente emprestado a máquina em que fiz todos os experimentos.
- Ao seu aluno Denis Oliveira por ter solucionado minhas dúvidas sobre o sistema.

Medindo o Consumo Energético

Utilizamos uma máquina com um sensor de potência emprestada pelo Prof. Dr. Hermes Senger da UFSCar. Desta máquina, pode-se fazer requisições Telnet ao servidor MW100 Yokogawa de dados do sensor. Nosso acesso ao MW100 era restrito e não podíamos requerer gravações. Assim, escrevemos uma simples biblioteca estática `mw100_recorder` para encapsular as requisições Telnet e fazer gravações sobre o consumo de potência.

Utilizando esta biblioteca, implementamos duas ferramentas:

- **`powerdump`**: faz amostras do consumo de potência por um intervalo de tempo determinado pelo usuário. Útil para estudar o perfil de consumo tanto de aplicações quanto do sistema.
- **`energyanalyser`**: executa um comando certo número de vezes gravando informações sobre tempo decorrido, potência média consumida, energia total do sistema e energia consumida pela aplicação. Tanto o comando quanto o número de amostras são passados pelo usuário. Calcula a energia consumida pela aplicação subtraindo o consumo base do sistema do consumo total.

Descrição dos Experimentos

Para encontrar oportunidades de melhoria na camada de aplicação, fizemos quatro experimentos.

1 Perfis de consumo de Potência: Gravamos o consumo de potência para aplicações CPU Bound, Memory Bound e I/O Bound experimento e comparamos seus perfis. Este experimento testa nosso sistema de medição e oferece *insights* sobre o consumo de potência de aplicações.

2 Escolha do Algoritmo: Implementamos uma variação esparsa do algoritmo de ordenação BucketSort e comparamos com o QuickSort. Escolhemos o BucketSort esparso por ele ter consumido menos potência que o QuickSort em todos os casos. Experimento baseado em [2].

3 Eliminação de *Code Smells*: Em [3], os autores identificam alguns padrões ruins de código em C++ que aumentam a energia consumida pela aplicação sem impacto na performance. Nosso experimento testou o impacto destes e outros *code smells* em código C.

4 Agendamento de Processos: Queremos estudar como o agendamento paralelo de processos impacta no consumo energético. Este experimento testa diferentes números de processos paralelos por vez para realizar uma determinada tarefa e grava o consumo de potência, performance temporal, e consumo de energia, para cada valor.

Resultados