Implementação de um Cluster Embarcado usando a Plataforma Raspberry Pi

Felipe dos Anjos Lima[1], Wanderson Roger Azevedo Dias[2], Edward David Moreno[3]

[1]Coordenadoria de Tecnologia da Informação – Centro Universitário Estácio de Sergipe
Aracaju – SE – Brasil

[2]Coordenadoria de Computação – Instituto Federal de Rondônia (IFRO)
Ji-Paraná – RO – Brasil

[3]Departamento de Computação – Universidade Federal de Sergipe (UFS)
São Cristóvão – SE – Brasil

{felipes1474, wradias, edwdavid}@gmail.com

Resumo. *Este artigo apresenta os resultados obtidos nas simulações que mensuraram o desempenho computacional e o consumo energético de um cluster embarcado de baixo custo implementado com a plataforma Raspberry Pi. Nas simulações, foram utilizado os algoritmos matemáticos de multiplicação de matrizes e produto escalar, além dos benchmarks HPL e HPCC, onde foi possível constatar um aumento considerável de desempenho obtido com o uso cluster, quando comparado com soluções sequenciais. Também pode-se concluir que o aumento no poder de processamento do cluster com a biblioteca OpenMPI ocasionou um consumo de 5,7% a mais energia durante o seu processamento.*

1. Introdução

Com o desenvolvimento tecnológico, o poder de processamento dos computadores aumentou consideravelmente nos últimos anos. No entanto, a quantidade de dados processados também aumentou. Por isso, a busca por processadores e sistemas cada vez mais velozes e que atendam as novas demandas continua sendo prioridade para a comunidade cientifica de computação (Lima *et al*, 2016).

Segundo Rauber e Rünger (2013), várias áreas das Ciências Naturais e da Engenharia estão cada vez mais necessitando de poder computacional, para realizar simulações de problemas científicos que manipulam grandes quantidades de dados, demandando assim grande esforço computacional, visto que um baixo desempenho dos sistemas pode levar a uma restrição das simulações e uma imprecisão nos resultados obtidos. Desta forma, a tarefa de manter o poder de computação dos processadores aumentando continuamente, seguindo a *Lei de Moore* (Moore, 1965), tem sido um grande desafio para os engenheiros e cientistas da computação.

Para revolver problemas arquiteturais específicos como: limitação na redução do tamanho dos transistores, aumento no quantitativo de transistores inseridos no núcleo do processador, dissipação de energia, resfriamento, dentre outros, os *designers* de processadores passaram a considerar a criação de sistemas paralelos, ou seja, ao invés de

tentar aumentar o poder de processamento de um único núcleo, passaram a considerar a criação de processadores com dois ou mais núcleos em um único circuito integrado, chamados de processadores *multicores*. Além de associar dois ou mais computadores (*clusters*), para que juntos executem uma mesma tarefa em paralelo, reduzindo assim o tempo de processamento total da aplicação (Costa, 2007).

Existem atualmente várias APIs e bibliotecas que auxiliam o desenvolvimento de programas paralelos, tais como: MPI (*Message Passing Interface*) (MPI, 2020) e OpenMP (*Open Multi-Processing*) (OpenMP, 2020) e outras. Dessa forma, todo o gerenciamento de memória compartilhada necessário para manter a consistência da computação realizada pelos dispositivos, pode ser feito de forma segura.

Além do poder de processamento, outro fator importante a ser considerado em um *cluster* é o consumo de energia. Os *Clusters* com baixa eficiência energética podem apresentar problemas de desempenho computacional. No entanto, atualmente, os processadores ARM (ARM, 2020) apresentam um bom desempenho energético devido ao conjunto de instruções simplificado, o que é relevante no projeto de sistemas embarcados. Portanto, neste trabalho, analisamos o desempenho computacional e o consumo energético de um *cluster* embarcado utilizando a plataforma *Raspberry Pi* (RPi) que possui em sua arquitetura o processador ARM.

O artigo está organizado em quatro seções, a Seção 2 apresenta o ambiente e as simulações; já a Seção 3 faz a análise dos resultados obtidos nas simulações, e por fim, a Seção 4 apresenta as conclusões e ideias para trabalhos futuros.

2. Ambiente de Simulação

Para as simulações, construímos um *cluster* embarcado usando quatro nós com a plataforma ***Raspberry Pi 2 modelo B*** (Raspberry Pi, 2020), que possui processador ARM, produzido pela empresa *Broadcom*, modelo BCM2836, ARMv7, sendo *quad-core* de 900 MHz e também 1GB de memória interna (RAM - *Random Access Memory*) (Broadcom, 2020). Após a montagem e configuração do *cluster*, realizamos as simulações a fim de mensurar o desempenho e consumo energético do mesmo.

Foi utilizado o Sistema Operacional (SO) *Raspbian* (Raspberry Pi, 2020) que é otimizado especificamente para executar na plataforma *Rapsberry Pi*, sendo o mesmo derivado da distribuição do SO Debian. Todos os nós do *cluster* foram conectados a um *switch* na rede local (Gebali, 2011), (Jin *et al*, 2011). Durante a execução das aplicações, todas as requisições de dados entre os nós do *cluster* se deram através do protocolo SSH (*Secure Shell*).

Como algoritmos para a carga de testes no *cluster*, foram utilizadas duas aplicações matemáticas, sendo uma que realiza a multiplicação de matrizes e outra que faz o cálculo do produto escalar. Além disso, com o auxílio dos *benchmarks* HPL e HPCC, também foi possível extraímos métricas de desempenho, como: tempo de execução, *GFlops* e o consumo energético. Com o *framework* MPI foi possível paralelizar e dividir a carga de dados entre os nós do *cluster*.

Nas simulações, utilizamos diferentes implementações usando as bibliotecas MPICH-2 e OpenMPI. Além da métrica do tempo de execução do *cluster*, também apresentamos o *speedup*, que permitiu medir o desempenho das soluções paralelas

quando comparadas com as respectivas soluções sequenciais. Também foi possível mensurar o consumo energético do *cluster* executando o *benchmark* HPCC usando as bibliotecas MPICH-2 e OpenMPI.

3. Análise dos Resultados

Após os testes, foi possível verificar o comportamento do *cluster* durante a resolução de problemas de diferentes tamanhos e em diferentes ambientes de programação. Levando em consideração a resolução de sistemas lineares com cinco mil e dez mil variáveis, pode-se observar que o *cluster* implementado com a biblioteca OpenMPI obteve um melhor desempenho computacional, quando comparado ao *cluster* implementado com a biblioteca MPICH-2, tanto para cinco e dez mil variáveis (ver Figura 1).

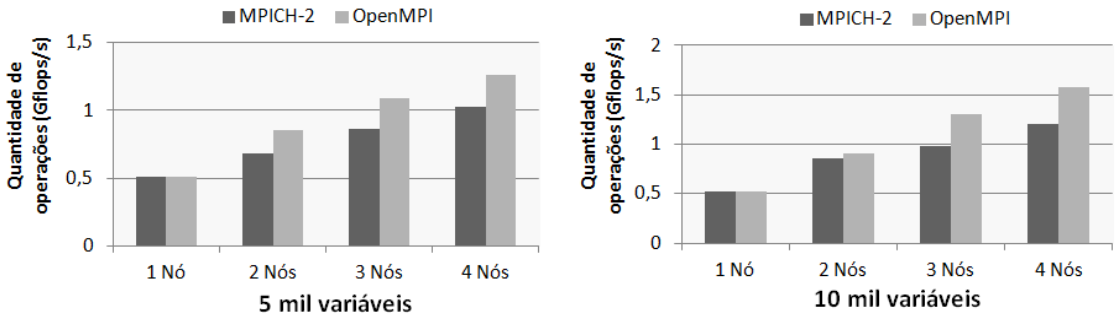

Figura 1. Desempenho do *Cluster* com HLP

A análise dos dados apresentados nos gráficos da Figura 2 mostram que para a solução de sistemas lineares com cinco mil variáveis, o desempenho do *cluster* composto por quatro nós e com a biblioteca OpenMPI foi em média 18% mais rápido na execução dos algoritmos matemáticos (multiplicação de matrizes e cálculo do produto escalar), quando comparado ao *cluster* usando a biblioteca MPICH-2 com os mesmos quatro nós. Já para a resolução de sistemas lineares com dez mil variáveis, o *cluster* implementado com a biblioteca OpenMPI também obteve melhor desempenho, quando comparado ao *cluster* implementado com a biblioteca MPICH-2, ou seja, 24% a mais de desempenho computacional, analisando o *cluster* com quatro nós. Além disso, quando comparado o *cluster* com apenas um nó e quatro nós, os *speedups* foram de 2,5x para a implementação com OpenMPI e 2x para a implementação com a biblioteca MPICH-2 (ver Figura 3).

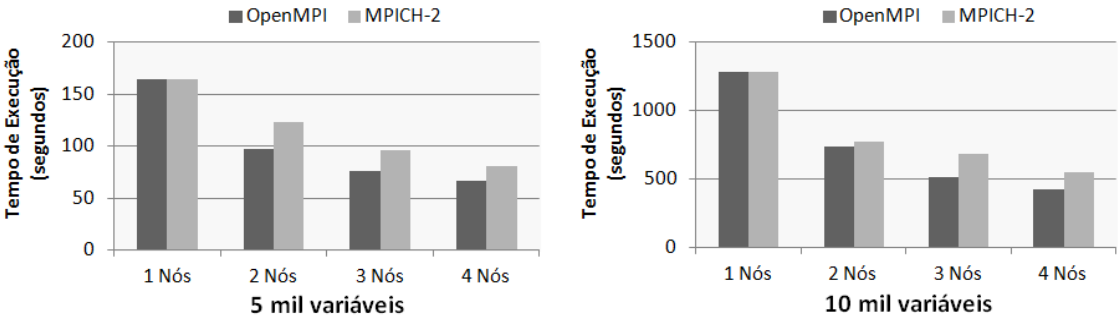

Figura 2. Tempos das bibliotecas OpenMPI e MPICH-2

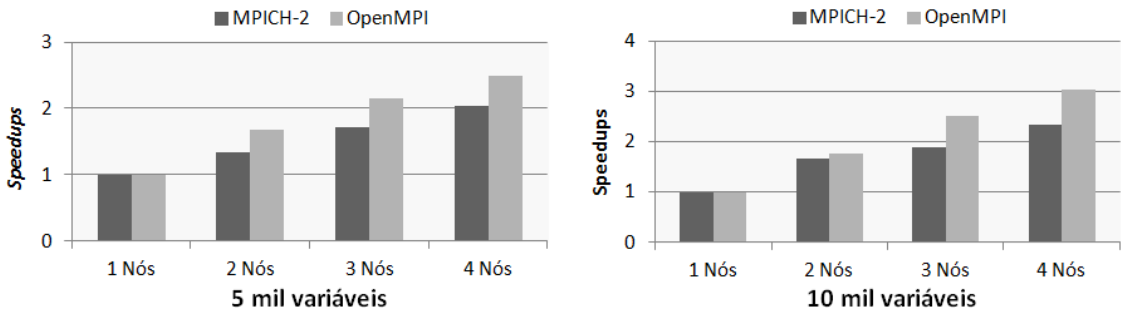

Figura 3. *Speedups* das bibliotecas MPICH-2 e OpenMPI

3.1. Consumo Energético do *Cluster*

Além da capacidade de processamento e do tempo de execução, também foi mensurado o consumo de energia (tensão, corrente e potência) do *cluster* com o uso das bibliotecas OpenMPI e MPICH-2 executando o *benchmark* HPCC. Analisando o *cluster* quando executando a resolução do sistema linear com cinco mil variáveis, os dados apresentados nos gráficos da Figura 4 mostram que a potência máxima alcançada pelo *cluster* com a biblioteca OpenMPI foi 5,7% maior (média para 1, 2, 3 e 4 nós), quando comparada ao *cluster* implementado com a biblioteca MPICH-2. Assim, pode-se concluir que o aumento no poder de processamento do *cluster* com a biblioteca OpenMPI fez com que o *cluster* consumisse mais energia durante o processamento (ver Figura 5 e Tabela 1).

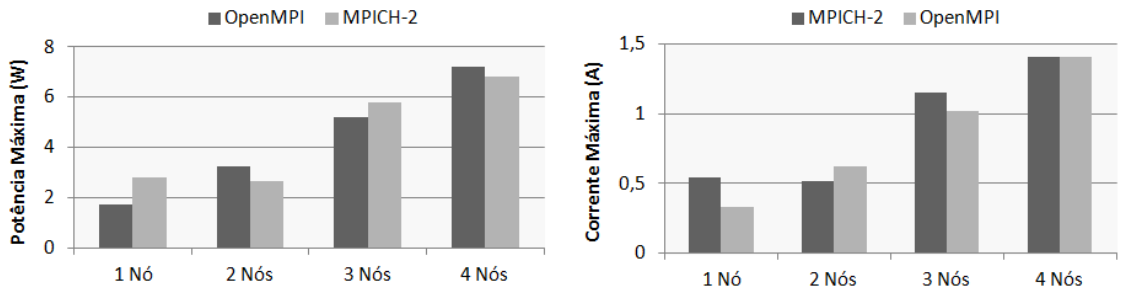

Figura 4. Potência e Corrente Máxima do *Cluster*

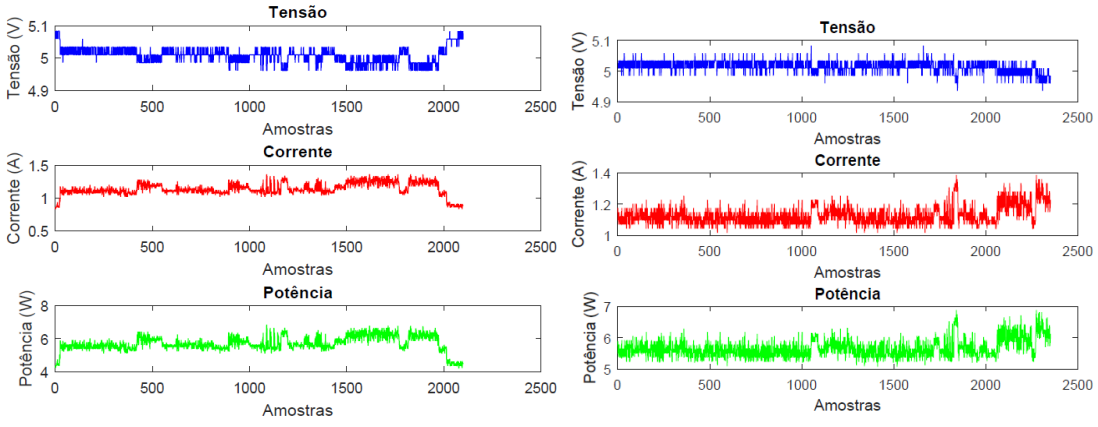

Figura 5. Consumo Energético do *Cluster* com 4 nós executando o *Benchmark* HPCC

Tabela 1. Consumo Energético do *Cluster* com 4 nós executando o *Benchmark* HPCC

Biblioteca	Potência (W)		Corrente (A)	
	Máxima	Média	Máxima	Média
MPICH-2	6,8495	5,6991	1,3606	1,1383
OpenMPI	6,8809	5,6557	1,3870	1,1276

4. Conclusões e Trabalhos Futuros

Após realizar este trabalho, fazemos as seguintes observações: (i) o aumento no número de nós do *cluster* provocou uma redução considerável do tempo de execução das aplicações; (ii) um aumento na quantidade de dados processados provou uma melhora no desempenho do *cluster*. Isso se deve ao fato de que para pequenas quantidades de dados, o tempo gasto na comunicação entre os nós supera o tempo de processamento, fazendo com que a solução sequencial seja mais eficiente; (iii) após a análise das aplicações executadas e dos resultados obtidos com a execução dos *benchmarks HPL* e *HPCC*, concluímos que o *cluster* implementado com *Raspberry Pi*, usando a biblioteca OpenMPI obteve um melhor desempenho em relação ao mesmo *cluster* com a biblioteca MPICH-2. Como trabalhos futuros, pretendemos mensurar o desempenho computacional e o consumo energético do *cluster*, executando outros *benchmarks* e aplicações, além de verificar o impacto provocado por outros sistemas operacionais.

Referências

ARM. (2020) "ARM Company Profile", Disponível em: <http://www.arm.com/>, Acessado em: 01 de Maio de 2020.

Broadcom (2020) "Broadcom", Disponível em: <https://www.broadcom.com/>, Acessado em: 02 de Maio de 2020.

Costa, R. A. G. (2007) "Desempenho e Consumo de Energia de Algoritmos Criptográficos do MiBench em Sistemas Móveis", UEA - Amazonas, Novembro de 2007.

Gebali, F. (2011) "Algorithms and Parallel Computing". – New Jersey: Wiley, 1st edition, 2011, 341p.

Jin, H.; Jespersen, D.; Mehrotra, P.; Biswas, R.; Huang, L. and Chapman, B. (2011) "High Performance Computing Using MPI and OpenMP on Multi-core Parallel Systems", In *Journal Parallel Computing*, 37(9):562-575, September, 2011.

Lima, F. A.; Moreno, E. D.; Dias, W. R. A. (2016) "Performance Analysis of a Low Cost Cluster with Parallel Applications and ARM Processors", In *IEEE Latin America Transactions*, 14(11):4591-4596, December, 2016.

Moore, G. E. (1965) "Cramming Moore Components onto Integrated Circuits", In *Reprinted from Electronics*, 38(8):114-114, April, 1965.

MPI. (2020) "Open MPI: Open Source High Performance Computing", Disponível em: <http://www.open-mpi.org/>, Acessado em: 30 de Maio de 2020.

OpenMP. (2020) "The OpenMP API Specification for Parallel Programming", Disponível em: <http://openmp.org/>, Acessado em: 10 de Junho de 2020.

Raspberry Pi. (2020) "Raspberry Pi Foundation", Disponível em: <https://www.raspberrypi.org/>, Acessado em: 15 de Março de 2020.

Rauber, T.; Rünger, G. (2013) "Parallel Programming: for Multicore and Cluster Systems". – New York: Springer, 2nd edition, 2013, 532p.