ISSN: 2319-0124
josif 2023
15ª JORNADA CIENTÍFICA E TECNOLÓGICA
12º SIMPÓSIO DE PÓS-GRADUAÇÃO
DO IFSULDEMINAS
REALIZAÇÃO INSTITUTO FEDERAL Sul de Minas Gerais
APOIO FAPEMIG CNPq

COMPARAÇÃO DE DESEMPENHO DOS ALGORITMOS DE ORDENAÇÃO: quicksort, mergesort e bucketsort

Gabriel ANGELO[1]; Kevin CAMARGO[2]; Milnner K.T. ANDRADE [3]; Pedro H. BARBOSA[4]; João P. de T. GOMES[5]

RESUMO

Este trabalho apresenta uma análise comparativa de desempenho dos algoritmos de ordenação: *Quicksort, Mergesort* e *Bucketsort* entre um *cluster* com 8 *Tv's box* (processador Arm Cortex-A53 *quad-core*, de 1.8 GHz, e 2GB de memória RAM) e um um *notebook* Acer Aspire 5 (processador AMD Ryzen 7-5700U de oito núcleos (16 *threads*), de 4.3 GHz, e 8GB de memória RAM). As simulações dos algoritmos utilizam a biblioteca OpenMPI para implementar uma execução paralela aplicada em problemas de diferentes tamanhos com objetivo de analisar o desempenho computacional obtido e verificar a capacidade operacional de utilizar um *cluster* embarcado de baixo custo através de *tv 's box*.

Palavras-chave: *OpenMPI*; *Tv box*; *Cluster*; Ordenação de vetores.

1. INTRODUÇÃO

A evolução tecnológica contínua tem impulsionado o aumento exponencial do poder de processamento dos computadores nos últimos anos. No entanto, essa melhoria tem sido acompanhada por um considerável aumento na quantidade de dados a serem processados. Diante desse panorama, a comunidade científica da computação está fortemente empenhada na busca por sistemas cada vez mais rápidos e eficientes (ULLER et al, 2020).

Em áreas como Ciências Naturais e Engenharia, a demanda por poder computacional é ainda maior. Simulações de problemas científicos que envolvem grandes quantidades de dados requerem um esforço computacional considerável. Um desempenho insuficiente dos sistemas pode resultar em restrições nas simulações e resultados imprecisos (TEZDUYAR et al, 1996). Portanto, manter o poder de computação em constante crescimento, seguindo os princípios da Lei de Moore, tem sido um desafio para engenheiros e cientistas da computação (MOORE, 1965).

Para enfrentar problemas arquiteturais específicos, como a limitação na redução do tamanho dos transistores, o aumento da densidade de transistores em um único processador e questões de dissipação de energia e resfriamento, os designers de processadores começaram a explorar a criação

[1] Discente do curso de Bacharelado em Ciência da Computação, IFSULDEMINAS – *Campus* Passos. E-mail: gabriel.angelo@alunos.ifsuldeminas.edu.br.
[2] Discente do curso de Bacharelado em Ciência da Computação, IFSULDEMINAS – *Campus* Passos. E-mail: kevin.camargo@alunos.ifsuldeminas.edu.br.
[3] Discente do curso de Bacharelado em Ciência da Computação, IFSULDEMINAS – *Campus* Passos. E-mail: milnner.andrade@alunos.ifsuldeminas.edu.br.
[4] Discente do curso de Bacharelado em Ciência da Computação, IFSULDEMINAS – *Campus* Passos. E-mail: pedro6.silva@alunos.ifsuldeminas.edu.br.
[5] Professor, IFSULDEMINAS – *Campus* Passos. E-mail: joao.gomes@ifsuldeminas.edu.br.

de sistemas paralelos (PITANGA, 2008). Entretanto, ao invés de simplesmente aumentar o poder de processamento de um único núcleo, eles passaram a considerar a criação de processadores multicores, que integram dois ou mais núcleos em um único circuito integrado. Além disso, a utilização de *clusters*, onde dois ou mais computadores trabalham juntos em paralelo para executar tarefas, tem se mostrado uma solução promissora para reduzir o tempo de processamento total de aplicações computacionais complexas (PITANGA, 2008). Felizmente, existem atualmente várias APIs e bibliotecas disponíveis para auxiliar no desenvolvimento de programas paralelos, como o *Message Passing Interface* (MPI) e o *Open Multi-Processing* (OpenMP). Essas ferramentas permitem um gerenciamento eficiente de memória compartilhada, garantindo a consistência dos cálculos realizados pelos dispositivos (MALLÓN et al, 2009).

Além do poder de processamento, outro fator crítico a ser considerado em um *cluster* é o consumo de energia. *Clusters* com baixa eficiência energética podem comprometer o desempenho computacional, tornando-se um obstáculo para aplicações intensivas. Nesse contexto, os processadores ARM têm se destacado, graças ao seu conjunto de instruções simplificado, resultando em um desempenho energético superior, especialmente em projetos de sistemas embarcados (Z. Ou et al, 2012). Motivado por esses desafios e oportunidades, este artigo apresenta um *cluster* de *TV Box*, desenvolvido com base nessas premissas. Serão discutidos o ambiente de desenvolvimento, as simulações realizadas e a análise dos resultados obtidos. O objetivo principal é avaliar o desempenho computacional e o consumo de energia desse *cluster*, contribuindo para a compreensão e a aplicação eficiente dessas tecnologias.

2. MATERIAL E MÉTODOS

Para a análise, foi montado um *cluster* embarcado composto por oito nós de processamento. Cada um desses nós consiste em uma *TV Box* equipada com um processador Arm Cortex-A53 *quad-core*, com frequência máxima de 1.8 GHz, e 2GB de memória RAM (*Random Access Memory*). O Sistema Operacional original das *TVs Box* foi substituído pela distribuição Linux Armbian, possibilitando a conexão entre os dispositivos. Todas as requisições de dados entre os nós do *cluster* foram realizadas por meio do protocolo SSH (*Secure Shell*). A rede interna estabelecida utiliza um *switch* modelo DGS-1210-10, fabricado pela D-Link, que permite a transmissão de dados a uma velocidade de 100 Mbits/s.

O repositório orientativo para a formatação e montagem do *cluster* com *Tv's box* pode ser obtido no endereço: https://www.espacomaker-passos.com/laboratorios-e-materias-de-apoio.

Além do *cluster*, também foi utilizado um *notebook* Acer Aspire 5, equipado com um processador AMD Ryzen 7-5700U de oito núcleos (16 *threads*), com frequência máxima de 4.3 GHz, e 8GB de memória RAM. O *notebook* possui o sistema operacional Ubuntu 22.04.2 LTS

x86_64, com kernel 5.19.0-46-generic. Sendo assim, essas duas plataformas distintas foram utilizadas para possibilitar análises comparativas do desempenho computacional do *cluster* construído. Para os testes, foram implementados os algoritmos de ordenação *Bucket Sort, Merge Sort e Quick Sort*, aplicados à ordenação de vetores.

Para a paralelização das operações, foi utilizada a biblioteca OpenMPI. Os algoritmos foram submetidos à ordenação de vetores de tamanhos variando de 25.000 a 100.000.000, e seus tempos de execução foram comparados.

3. RESULTADOS E DISCUSSÃO

A execução dos algoritmos de ordenação foram submetidos a uma variação de tamanho do vetor e registrado o tempo de execução em cada ambiente de simulação da análise. Embora o *notebook* possua um processador mais poderoso (Ryzen 7-5700U), os tempos de execução não mostraram diferenças significativas em várias faixas de tamanhos de vetores, como mostrado através das Figuras 1, 2 e 3.

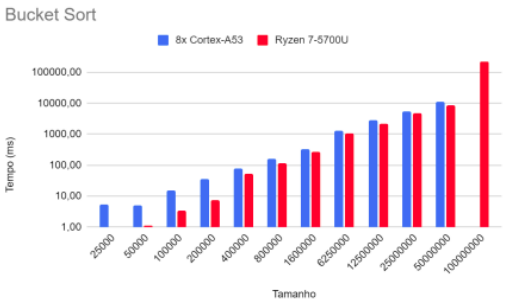

Figura 1 - Gráfico de tempo de execução do *Bucket Sort* nas diferentes plataformas

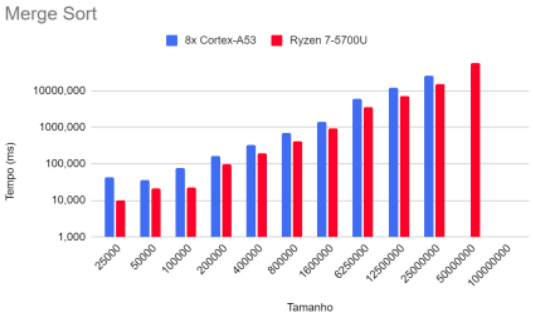

Figura 2 - Gráfico de tempo de execução do *Merge Sort* nas diferentes plataformas

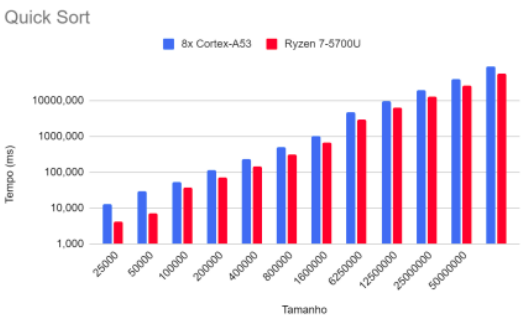

Figura 3 - Gráfico de tempo de execução do *Quick Sort* nas diferentes plataformas

Os resultados apontam que o desempenho do *cluster* é promissor e capaz de lidar eficientemente com esse tipo de algoritmo. Alguns casos de teste foram invalidados durante a execução pela falta de memória disponível nas plataformas, é interessante notar que a diferença de memória total disponível no *Cluster* (16GB) contra a memória do notebook (8GB) não teve impacto significativo para problemas dos tamanhos testados.

4. CONCLUSÃO

Com base nos resultados obtidos, pode-se concluir que o *cluster* de *TV Box* é uma alternativa viável para realizar tarefas de processamento intensivo, como a ordenação de grandes conjuntos de dados. Embora o *notebook* Acer Aspire 5 possua um processador mais poderoso, o *cluster* apresentou desempenho competitivo em relação ao tempo de execução dos algoritmos de ordenação, demonstrando sua capacidade de processamento paralelo.

Entre os algoritmos analisados, constata-se: o *Bucket Sort* pode ser vantajoso quando há mais núcleos, aliando-o a outro algoritmo de ordenação que também seja eficiente, fazendo uma divisão uniforme da carga de trabalho entre eles. Para o *Quicksort* e *Mergesort*, por serem algoritmos de divisão e conquista, são mais simples de serem paralelizados, sendo vantajoso para o aprendizado. Portanto, para os devidos fins, este trabalho, da disciplina de Computação de Alto Desempenho, permitiu a associação da teoria e prática.

REFERÊNCIAS

ULLER, João Fellipe et al. **É possível buscar supercomputadores energeticamente eficientes?** 2020. Disponível em: <https://repositorio.ufsc.br/handle/123456789/214019>. Acesso em: 10/07/2023.

TEZDUYAR, T., ALIABADI, S., BEHR, M. *et al.* **Simulação de fluxo e computação de alto desempenho**. *Computational Mechanics* 18 , 397–412 (1996). https://doi.org/10.1007/BF00350249.

PITANGA, Marcos. **Construindo supercomputadores com Linux.** Brasport; 3ª edição, 2008. ISBN-10: 8574523720.

MALLÓN, Damián A. et al. **Performance evaluation of MPI, UPC and OpenMP on multicore architectures**. In: Recent Advances in Parallel Virtual Machine and Message Passing Interface: 16th European PVM/MPI Users' Group Meeting, Espoo, Finland, September 7-10, 2009. Proceedings 16. Springer Berlin Heidelberg, 2009. p. 174-184. Disponível em: <http://jfs.des.udc.es/papers/mallon_pvmmpi09.pdf>. Acesso em: 10/07/2023.

Z. Ou, B. Pang, Y. Deng, JK Nurminen, A. Ylä-Jääski e P. Hui, "**Energy- and Cost-Efficiency Analysis of ARM-Based Clusters**," 2012 12º IEEE/ACM International Symposium on Cluster , *Cloud and Grid Computing (ccgrid 2012)* , Ottawa, ON, Canadá, 2012, pp. 115-123, doi: 10.1109/CCGrid.2012.84.