On the Use of Fujitsu A64FX Processor for Genome Sequence Analysis Workloads

Julian Pavon, Ivan Vargas Valdivieso, Osman Ünsal and Adrian Cristal
Department of Computing Science
Barcelona Supercomputing Center
Barcelona, Spain
Email: {julian.pavon, ivan.vargas, osman.unsal, adrian.cristal}@bsc.es

***Abstract*—Fujitsu's A64FX has been the first Arm-based processor to power a top-ranked supercomputer, challenging the dominance that x86 processors have held in high-performance computing (HPC). While x86 processors and their software ecosystems have been extensively characterized, the performance and efficiency behavior of Arm-based alternatives, like the A64FX, remains less well explored, limiting broader adoption.**

This work methodically characterizes the performance and power efficiency of the A64FX processor running a variety of genome sequence analysis workloads compiled with the GNU Compiler Collection (GCC). We compare A64FX against an Intel Xeon Gold 5318N processor. While Skylake demonstrates higher single-thread performance, A64FX matches or exceeds performance in multi-threaded scenarios due to its high-bandwidth memory. Additionally, as genome sequence analysis workloads only target the Arm NEON ISA extension, there are unveiled opportunities to further improve performance using the Scalable Vector Extension (SVE) ISA extension. We implement in-house SVE-based algorithms for all the evaluated algorithms which demonstrate 2.1× better performance compared to their AVX512-based counter versions.

I. Introduction

The decreasing cost and increasing throughput of genome sequencing technologies, along with the availability of complete genomes for humans and other species, have enabled rapid advances in personalized medicine [1]–[3], evolutionary biology [4], [5], and forensics [6], [7]. At the core of these applications lies genome sequence analysis, where sequences are compared to extract key genetic insights —ranging from identifying disease-causing mutations [8], and computing similarity across large datasets for phylogenetic studies [9].

Modern genome analysis relies primarily on two algorithmic families: sequence alignment and edit distance approximation. These algorithms receive as input *reads* —short fragments of DNA or RNA sequences obtained from high-throughput sequencing machines. Reads are typically aligned or compared against reference genomes or other sequences to identify genetic variations, detect mutations, or reconstruct longer genomic regions, making their efficient execution critical for many genomic applications. These algorithms, particularly when applied to the increasing volume and length of sequencing data, pose significant computational challenges due to their intensive dynamic programming workloads.

To meet growing performance and energy-efficiency demands, recent genomic computing platforms are embracing hardware-software co-design and advanced instruction set architectures. One prominent example is Fujitsu's A64FX processor, the first Arm-based CPU to power a top-ranked supercomputer. Designed with high memory bandwidth and Scalable Vector Extension (SVE) support, A64FX offers a compelling platform for accelerating genomics workloads that exhibit high data-level parallelism.

In this paper, we explore the performance of the Arm ecosystem for genome sequence analysis by evaluating widely-used sequence alignment and edit distance algorithms on the A64FX processor. Using the GNU Compiler Collection (GCC), we investigate how effectively current toolchains support parallelism and vectorization, and compare A64FX's performance and efficiency to a baseline server class x86 system. Our analysis demonstrates that the A64FX system provides better performance scalability in multi-threaded scenarios thanks to its high-bandwidth memory (HBM). Additionally, the A64FX provides 2.1× better performance than AVX512 implementations when using the SVE ISA.

II. Experimental Environment

A. CPU Hardware Platforms

Our evaluation platforms consist of a Fujitsu A64FX processor with 48 cores and 32GB HBM2 [10] (referred as *a64fx* in Section III), and a server class Intel Xeon Gold 5318N with 24 (48) cores (threads) processor [11] (Cascade Lake architecture) with 192GB DDR4-2667 (referred as *Xeon* in Section III).

B. Benchmarks

Use case 1: Modern read aligners. For this use case, Wavefront Alignment (`WFA`) [12] and Bidirectional Wavefront Alignment (`BiWFA`) [13] algorithms, two recently proposed DP algorithms that run in $O(n * s)$ time, where n is the sequence length and s the error (or score) between the sequences. We use the best-performing configurations reported by Marco-Sola *et al.* [14], [15].

Use case 2: Edit distance approximation. We use the state-of-the-art edit distance approximation technique SneakySnake (`SS`) [16]. `SS` filters the input reads to skip the alignment of those inputs that exceed a defined edit distance threshold parameter. We used the best performing configurations reported by Alser *et al.* [17]

Use case 3: End-to-end framework. In this use case, we use minimap2 [18], [19], an end-to-end tool designed to map reads to large reference genomes. Minimap2 includes multiple algorithms such as *seeding*, *chaining* and *alignment*.

C. Datasets

For use cases 1 and 2, the evaluated datasets range from 100 base pairs (*bp*) to 30K base pairs. We use two real datasets (*100bp_1*, *250bp_1*) and two simulated datasets (*10Kbp* and *30Kbp*). Table I summarizes the main characteristics of the evaluated datasets. For the *100bp_1*, *250bp_1* and *10Kbp*, we use the datasets available in the SneakySnake repository [17]. We generate the *30Kbp* dataset following the same methodology as SneakySnake [16]. The *100bp_1* and *250bp_1* datasets are representative of the newest short-read technologies available in the market, ranging from the Ilumina iSeq100 which generates 100bp to the Illumina Next Generation Sequencing (NGS) that generates 300bp [20]–[22]. The *10Kbp* and *30Kbp* datasets are representative of long-read technologies such as PacBio that released a new HiFi technology that generates long-read in the range of 10K - 30K base pairs [22]–[24].

For use case 3, we use the reference and long input sequences provided by the minimap cookbook repository [25]. We evaluate the end-to-end execution of minimap2 going from index creation, through seeding and sequence alignment.

TABLE I
INPUT DATASET CHARACTERISTICS.

Dataset	Read Length	No. of Pairs	Dataset Size
100bp_1	100	30M	6GB
250bp_1	250	30M	14GB
10Kbp	10,000	100K	2GB
30Kbp	30,000	100K	6GB

D. Experiments

Scalability. In each experiment, we utilize up to 24 threads, pinning each thread to a dedicated core. To evaluate scalability, we begin with a single thread and increment the thread count in steps of four, up to the maximum of 24 threads —corresponding to the total number of physical cores available on the Xeon Gold 5318N CPU.

Vectorization. All evaluated benchmarks include optimized implementations that leverage the SIMD capabilities of both x86 [26] and Arm CPUs [27]. For x86, all benchmarks are optimized using AVX-512 instructions. For Arm, NEON-based implementations are included in the evaluated benchmarks. However, the A64FX processor supports the SVE ISA, which introduces advanced features such as predication —particularly well-suited for genome sequence analysis and capable of delivering further performance gains. In Section III-B, we evaluate the performance impact of incorporating SVE optimizations across all benchmarks.

III. EVALUATION

A. Performance comparison and scalability analysis

Modern sequence aligners: Fig. 1 depicts de performance results for modern read aligners. We make four observations: (i) At 1 thread, the Xeon Gold 5318N outperforms the A64FX across all configurations. On average, *Xeon* outperforms to *a64fx* by 2.2× and 2.1× for `WFA` and `BiWFA`, respectively. This performance results are mainly by Xeon's more aggressive out-of-order execution. (ii) Although *Xeon* initially scales well, its throughput gains diminish beyond 16 threads. For instance, in WFA-short, the jump from 16 to 24 threads yields only 25% improvement due to the system becoming bottlenecked by memory bandwidth and NUMA overheads. (iii) *a64fx* scales almost linearly across all thread counts, particularly in BiWFA-long. From 1 to 24 threads, throughput increases by over 22×. This behavior is driven by its high-bandwidth HBM2 memory and highly parallelize cache hierarchy. (iv) In long-read scenarios, *a64fx* outperforms *Xeon* at high thread counts. On average, *a64fx* outperforms *Xeon* by 1.2× and 1.1× for `WFA` and `BiWFA`, respectively, for the 24 threads experiments. The memory and cache-bound nature of long-read alignments benefit from *a64fx*'s superior memory bandwidth and parallelism.

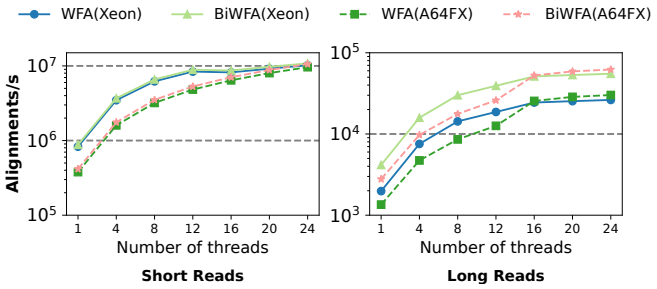

Fig. 1. Read aligners throughput results using short and long reads input datasets. Results are reported on a logarithmic scale.

Edit distance approximation: Fig. 2 shows the throughput results for edit distance experiments. We make two observations: (i) `SS` shows similar architectural trends as WFA and BiWFA. *Xeon* leads in single-thread performance for both short and long reads due to its stronger per-core throughput, but *a64fx* scales more effectively across threads with comparable throughput at 20 and 24 threads. (ii) The number of input sequences filtered by each thread can vary significantly, leading to workload imbalance. Since `SS` does not incorporate any software-level workload balancing mechanisms, the performance scalability on both hardware platforms tends to plateau beyond 12 threads. Nevertheless, as shown in Section III-B, fine-tuning `SS` for the SVE ISA yields substantial performance improvements despite this limitation.

End-to-end scalability analysis: Fig 3.a depicts the performance results for minimap2. Results are normalized to the *a64fx* single thread performance. We make two observations: (i) Similar to use cases 1 and 2, *Xeon* exhibits limited performance scalability as the number of threads increases. (ii)

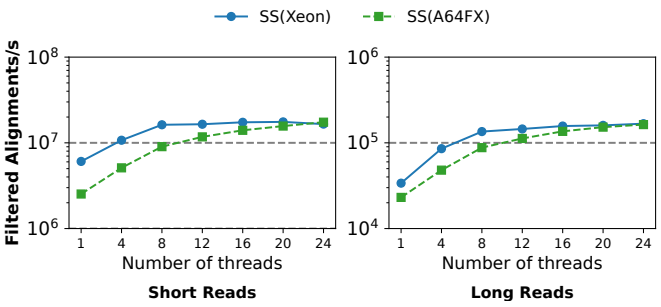

Fig. 2. Edit distance throughput results using short and long reads input datasets. Results are reported on a logarithmic scale.

Minimap2 is an end-to-end genome sequence analysis tool that integrates multiple algorithms, among which seeding is one of the most critical, as it dominates overall execution time [28], [29]. Seeding involves mapping input reads to a reference genome —a highly parallelizable operation. Consequently, a highly parallel system like the *a64fx* delivers notable performance advantages. On average, *a64fx* outperforms *Xeon* by 1.1×, 1.2×, and 1.2× for 16, 20, and 24 threads, respectively. These results emphasize the *a64fx*'s effectiveness for parallel and memory-intensive genome sequence analysis workloads.

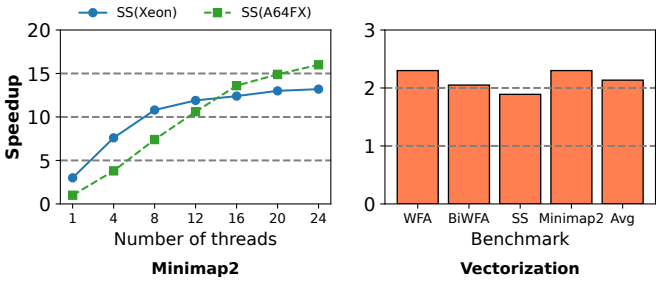

Fig. 3. Performance results for minimap2 [18] and SVE-based vectorization. Results for minimap2 are normalized to the single-thread A64FX performance. For SVE-based vectorization, results are normalized to the A64FX 24 threads performance of each algorithm.

B. Impact of vectorization

As mentioned in Section II-D, the evaluated benchmarks currently do not support the Arm SVE ISA, which offers several key features that could significantly enhance the performance of these workloads.

In this section, we evaluate the performance impact of the Arm SVE ISA on genome sequence analysis. To this end, we implemented in-house versions of the benchmarks using SVE intrinsics, using the NEON-based algorithms as a baseline. We (i) replaced NEON instructions with their corresponding SVE instructions and (ii) improved vectorization by leveraging the predication capabilities of the SVE ISA. Fig. 3.b shows the performance results obtained. In these experiments, we used 24 threads on the *a64fx* for both NEON and SVE-based implementations.

On average, SVE-based implementations outperform NEON-based ones by 2.1×. This significant performance improvement is primarily due to the advanced features of SVE, especially its predication capabilities. Unlike NEON, which requires separate conditional instructions to handle masking and branching, SVE supports predicated execution directly within vector operations. This reduces the need for additional control flow instructions, simplifying the code and minimizing costly branching. Furthermore, the larger vector length available in the *a64fx* architecture allows for better memory bandwidth utilization and more efficient data processing. The combination of these features —predication, reduced code complexity and larger vector length —enables SVE-based algorithms to outperform NEON-based implementations, leading to more efficient execution overall.

IV. Conclusion

This work evaluates the performance and scalability of the Arm A64FX processors for bioinformatics workloads compared to a server class Intel Xeon processor. Our findings highlight that while the Xeon processor excels in single-thread performance due to higher per-core power, it encounters limitations in multi-threaded scenarios due to memory bandwidth constraints. In contrast, the A64FX outperforms the Xeon in multi-threaded performance, showcasing superior scalability and efficiency, especially in memory-intensive tasks. Notably, the A64FX's HBM2, its highly-parallel cache hierarchy and its ability to scale effectively across multiple threads make it ideal for large-scale genomic analysis tasks, as evidenced in the minimap2 pipeline.

The study also emphasizes the importance of optimizing code for the SVE ISA, which significantly boosts performance in bioinformatics applications. Optimizations using SVE, demonstrated an average improvement of 2.1×, underscoring the potential of SVE to enhance processing efficiency in data-intensive applications. As bioinformatics workloads increasingly rely on parallelism and memory efficiency, the A64FX, with its advanced vectorization and parallel processing capabilities, presents a promising platform for the future. Future work will explore further optimizations of bioinformatics pipelines and the potential of Arm's Neoverse processors.

References

[1] C. Alkan, J. M. Kidd, T. Marques-Bonet, G. Aksay, F. Antonacci, F. Hormozdiari, J. O. Kitzman, C. Baker, M. Malig, O. Mutlu *et al.*, "Personalized Copy Number and Segmental Duplication Maps Using Next-Generation Sequencing," *Nature Genetics*, 2009.

[2] E. A. Ashley, "Towards Precision Medicine," *Nature Reviews Genetics*, 2016.

[3] L. Chin, J. N. Andersen, and P. A. Futreal, "Cancer Genomics: From Discovery Science to Personalized Medicine," *Nature Medicine*, 2011.

[4] H. Ellegren, "Genome Sequencing and Population Genomics in Non-Model Organisms," *Trends in Ecology & Evolution*, 2014.

[5] J. Prado-Martinez, P. H. Sudmant, J. M. Kidd, H. Li, J. L. Kelley, B. Lorente-Galdos, K. R. Veeramah, A. E. Woerner, T. D. O'connor, G. Santpere *et al.*, "Great Ape Genetic Diversity and Population History," *Nature*, 2013.

[6] M. J. Alvarez-Cubero, M. Saiz, B. Martínez-García, S. M. Sayalero, C. Entrala, J. A. Lorente, and L. J. Martinez-Gonzalez, "Next Generation Sequencing: An Application in Forensic Sciences?" *Annals of Human Biology*, 2017.

[7] C. Børsting and N. Morling, "Next Generation Sequencing and Its Applications in Forensic Genetics," *Forensic Science International: Genetics*, 2015.

[8] W.-W. Liao, M. Asri, J. Ebler, D. Doerr, M. Haukness, G. Hickey, S. Lu, J. K. Lucas, J. Monlong, H. J. Abel *et al.*, "A Draft Human Pangenome Reference," *Nature*, 2023.

[9] K. M. Swenson, M. Marron, J. V. Earnest-DeYoung, and B. M. Moret, "Approximating the True Evolutionary Distance Between Two Genomes," *JEA*, 2008.

[10] Fujitsu, "A64FX Microarchitecture Manual," Available at https://github.com/fujitsu/A64FX/blob/master/doc/A64FX_Microarchitecture_Manual_en_1.6.pdf.

[11] M. Arafa, B. Fahim, S. Kottapalli, A. Kumar, L. P. Looi, S. Mandava, A. Rudoff, I. M. Steiner, B. Valentine, G. Vedaraman *et al.*, "Cascade Lake: Next Generation Intel Xeon Scalable Processor," *IEEE Micro*, 2019.

[12] S. Marco-Sola, J. C. Moure, M. Moreto, and A. Espinosa, "Fast Gap-Affine Pairwise Alignment Using the Wavefront Algorithm," *Bioinformatics*, 2020.

[13] S. Marco-Sola, J. M. Eizenga, A. Guarracino, B. Paten, E. Garrison, and M. Moreto, "Optimal Gap-Affine Alignment in O (S) Space," *Bioinformatics*, 2023.

[14] S. Marco-Sola, "WFA2-lib," Available at https://github.com/smarco/WFA2-lib, accessed: 2023-03-23.

[15] Marco-Sola, "Optimal Gap-Affine Alignment," Available at https://github.com/smarco/BiWFA-paper, accessed: 2023-04-04.

[16] M. Alser, T. Shahroodi, J. Gómez-Luna, C. Alkan, and O. Mutlu, "Sneakysnake: A fast and accurate universal genome pre-alignment filter for cpus, gpus and fpgas," *Bioinformatics*, 2020.

[17] ——, "SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs and FPGAs," Available at https://github.com/CMU-SAFARI/SneakySnake, accessed: 2023-03-23.

[18] H. Li, "Minimap2: Pairwise Alignment for Nucleotide Sequences," *Bioinformatics*, 2018.

[19] ——, "Minimap2," Available at https://github.com/lh3/minimap2, accessed: 2023-04-04.

[20] Ilumina, "Ilumina Sequencing Platforms," Available at https://www.illumina.com/systems/sequencing-platforms.html, accessed: 2023-03-23.

[21] E. Mardis, "DNA Sequencing Technologies: 2006-2016," *Nature Protocols 12*, 2017.

[22] Y. Kurt, Matallana-RamirezLilian, W. Kohlway, R. Whetten, and J. Frampton, "A Fast, Flexible and Inexpensive Protocol for DNA and RNA Extraction for Forest Trees," *Forest Systems*, 2020.

[23] PacBio, "PacBio," Available at https://www.pacb.com/technology/hifi-sequencing/how-it-works/, accessed: 2023-03-23.

[24] T. Hon, K. Mars, and G. Young, "Highly Accurate Long-Read HiFi Sequencing Data for Five Complex Genomes," *Scientific Data 7*, 2020.

[25] H. Li, "Minimap2 Cookbook," Available at https://github.com/lh3/minimap2/blob/master/cookbook.md, accessed: 2025-04-04.

[26] Intel, "Intel Advance Vector Extension Programming Reference," Available at https://www.intel.com/content/dam/develop/external/us/en/documents/36945, accessed: 2023-04-04.

[27] ARM Ltd, "ARM," https://developer.arm.com/documentation/, accessed: 2022-01-15.

[28] M. Alser, J. Rotman, D. Deshpande, K. Taraszka, H. Shi, P. I. Baykal, H. T. Yang, V. Xue, S. Knyazev, B. D. Singer *et al.*, "Technology Dictates Algorithms: Recent Developments in Read Alignment," *Genome Biology*, 2021.

[29] M. Alser, Z. Bingöl, D. S. Cali, J. Kim, S. Ghose, C. Alkan, and O. Mutlu, "Accelerating Genome Analysis: A Primer on an Ongoing Journey," *IEEE Micro*, 2020.