Linear Algebra Based Graph Analysis on RISC-V GPGPU Vortex

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

Semyon Grigorev
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—In this work we evaluate Spla—sparse linear algebra based library for graph analysis—on RISC-V ISA based open source GPGPU Vortex. We show that !!!**

***Index Terms*—GraphBLAS, Sparse Linear Algebra, Graph Analysis, GPGPU, RISC-V**

I. Introduction

Sparse linear algebra has emerged as a powerful paradigm for high-performance graph analysis. A wide range of problems—from graph traversing to clustering—can be reduced to efficient algebraic operations over matrices and vectors. GraphBLAS API [1] follows this idea and defines a standardized set of building blocks: sparse matrices and vectors, algebraic structures like monoids and semirings, and fundamental operations such as matrix-matrix multiplication. GraphBLAS is specifically designed to serve as a foundational layer for the development of scalable, linear-algebra-based graph algorithms.

While highly tuned CPU implementations of GraphBLAS—most notably SuiteSparse:GraphBLAS[1] [2]—deliver strong performance on multi-core systems, implementing the GraphBLAS API efficiently on general-purpose graphics processing units (GPGPUs) remains a significant challenge. While GPGPUs is a promising platform for linear algebra based computations, they introduce well-known obstacles for sparse workloads, including irregular memory access patterns and load imbalance. Additionally, creating generalized kernels capable of operating not only on primitive data types like floats or integers but also on user-defined custom types presents a nontrivial engineering task.

Despite these challenges, several efforts have been made to create GPU-accelerated libraries for linear-algebra-based graph analysis, such as GraphBLAST[2] [3] which uses CUDA and the portable Spla[3] [4] library which uses OpenCL.

In parallel, the rise of open instruction set architectures (ISAs), most notably RISC-V, is expanding the hardware landscape. Recent work has explored the potential of RISC-V-based CPUs for graph analysis [5]–[7], including designs leveraging vector extensions [8]. Beyond CPUs, specialized accelerators—including RISC-V-based GPGPUs—are now emerging. One actively developed example is the Vortex platform, a RISC-V-based GPGPU that has been evaluated not only for graphics but also for scientific computing [10] and graph analysis [9], [11].

In this paper, we evaluate the suitability of the Vortex architecture for linear-algebra-based graph analysis. Specifically, we examine the performance scaling of the Spla library on this platform. Our evaluation using a cycle-approximate simulator shows that [Results to be inserted here].

II. Spla Graph Analysis Library

Spla [4] is a GPGPU-accelerated, GraphBLAS-inspired library for graph analysis. It is based on sparse linear algebra and uses OpenCL to offload linear algebra kernels to appropriate devices, including GPGPUs. Using OpenCL makes the library vendor-agnostic: it has been shown in [4] that Spla performs and scales well across GPUs from different vendors including AMD, Intel, and Nvida.

The library implements several classical graph analysis algorithms, including canonical single-source level BFS, triangle counting (TC), single-source shortest path (SSSP), and PageRank.

III. RISC-V GPGPU Vortex

Vortex[4] [12] is an open-source RISC-V-based GPGPU. It supports OpenCL programming via the POCL compiler[5] [13]. Additionally, it is designed for FPGAs equipped with high-bandwidth memory (HBM), which is advantageous for graph processing.

The high-level architecture of the Vortex processor[6] is shown in Fig. 1. The processor consists of *clusters*, which may share an optional L_3 cache. Each *cluster* contains multiple *sockets*, which may share an optional L_2 cache. *Sockets* consist of cores with shared L_1 cache, and each core hosts multiple threads. Threads share local memory and are logically grouped into warps.

Identify applicable funding agency here. If none, delete this.

[1] Source code of SuiteSparse:GraphBLAS on GitHub: https://github.com/DrTimothyAldenDavis/GraphBLAS

[2] GraphBLAST project page: https://github.com/gunrock/graphblast

[3] Spla project page: https://github.com/SparseLinearAlgebra/spla

[4] https://github.com/vortexgpgpu/vortex

[5] Portable Computing Language project: https://portablecl.org/

[6] Detailed architectural information is available at https://github.com/vortexgpgpu/vortex/blob/master/docs/microarchitecture.md.

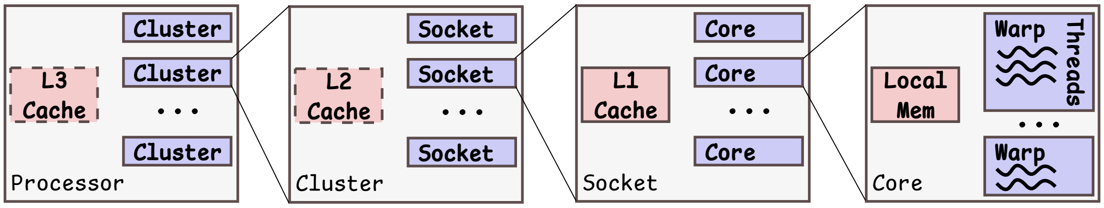


Fig. 1. Vortex architecture

The design is flexibly configurable: the numbers of clusters, cores, threads, and warps in the target processor can be specified, and the L_3 and L_2 caches can be independently enabled or disabled. Number of sockets calculated automatically such that socket size is a minimum of 4 and number of cores.

The Vortex design is distributed with SimX, a cycle-approximate functional simulator. A cycle-accurate RTL simulation is also available. Although the A extension (atomics instructions)[7] is declared, atomic operations are currently supported only in the SimX simulator and not in the RTL implementation.

IV. Evaluation

The goal of the evaluation is to estimate Spla performance scaling on Vortex. To do it perform several experiments utilizing SimX because it is faster.

A. Environment

Problems with floats. So BFS and TC only. Single one graph: !!! name, vertices, edges.

Two experiments Fixed number of clusters (2) and cores (4) to estimate caches. Iterates throw warps and threads (threads per warp). For the best configuration from the previous step: iterates throw clusters and cores (core per cluster).

B. Results

Edges per core on cycle. Compare with Spla on other GPUs.

C. Scaling limitations analysis

LSU, Graphics

V. Conclusion

In this work we evaluated Spla—linear algebra based graph analysis library—on RISC-V IAS based GPGPU Vortex. We show that Spla is portable enough to be run on Vortex. Vortex ready to run. Scaling.

Future work. Floats. More experiments in SimX. On different graphs. On other algorithms. Limits of scaling with clusters. FPGA resources. Performance on FPGA.

Evaluate Ventus [8] [14] GPU, compare with Vortex. !!!

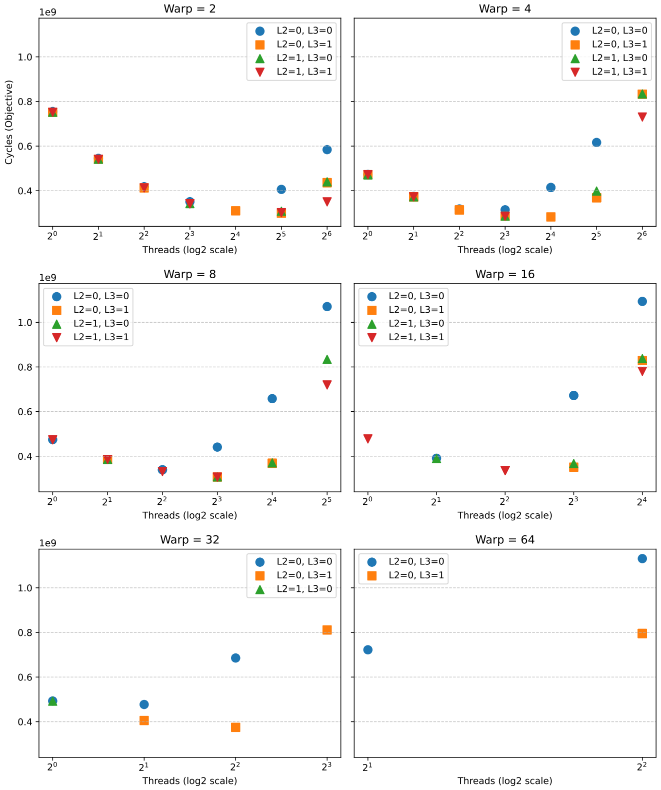
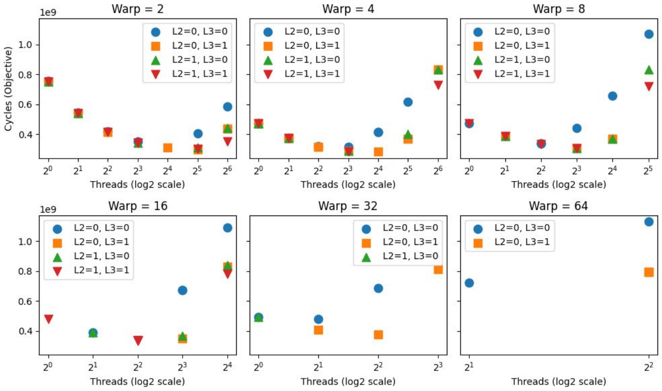
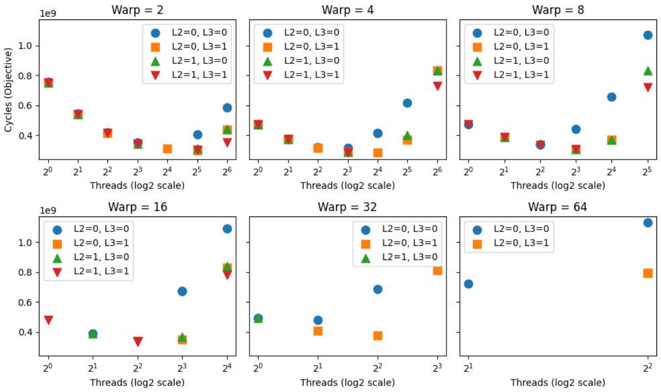


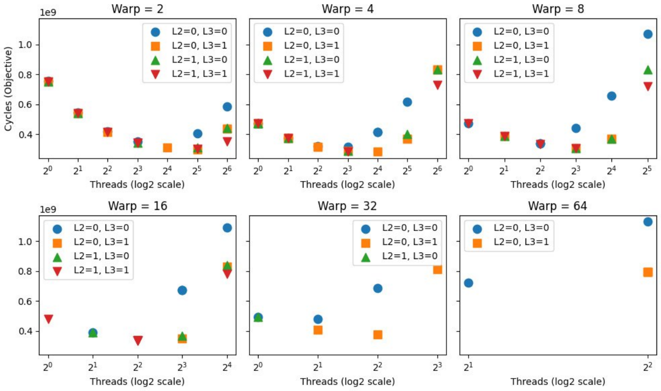
Fig. 2. Triangle counting performance threads and warps



[7] Supported RISC-V profiles are RV32IMAF and RV64IMAFD (https://github.com/vortexgpgpu/vortex?tab=readme-ov-file#specifications)

[8] https://github.com/THU-DSP-LAB/ventus-gpgpu





References

[1] B. Brock, A. Buluç, T. G. Mattson, S. McMillan, and J. E. Moreira, "Introduction to graphblas 2.0," in *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2021, pp. 253–262.

[2] T. A. Davis, "Algorithm 1037: Suitesparse:graphblas: Parallel graph algorithms in the language of sparse linear algebra," *ACM Trans. Math. Softw.*, vol. 49, no. 3, Sep. 2023. [Online]. Available: https://doi.org/10.1145/3577195

[3] C. Yang, A. Buluç, and J. D. Owens, "Graphblast: A high-performance linear algebra-based graph framework on the gpu," *ACM Trans. Math. Softw.*, vol. 48, no. 1, Feb. 2022. [Online]. Available: https://doi.org/10.1145/3466795

[4] E. Orachev, "Generalized sparse linear algebra library with vendor-agnostic gpus acceleration," Master's thesis, Saint Petersburg State University, 2023.

[5] A. M. Ravikumar, A. Vinay, K. K. Nagar, and M. Purnaprajna, "Parallel graph algorithms on a RISCV-based many-core," *Int. J. Reconfigurable Embed. Syst. (IJRES)*, vol. 14, no. 3, p. 843, Nov. 2025.

[6] K. Zhou, J. Deng, and Y. Zeng, "Design and memory access optimization of graph processing processor design based on risc-v," in *Proceedings of the 2023 6th International Conference on Artificial Intelligence and Pattern Recognition*, ser. AIPR '23. New York, NY, USA: Association for Computing Machinery, 2024, p. 576–583. [Online]. Available: https://doi.org/10.1145/3641584.3641670

[7] M. S. Yenimol, "Hardware/software co-design of domain-specific risc-v processor for graph applications," 2022. [Online]. Available: http://hdl.handle.net/11693/80659

[8] P. Vizcaino, J. Labarta, and F. Mantovani, "Graph computing on long vector architectures (yes, it works!)," in *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2024, pp. 986–995.

[9] N. Shah and M. Verhelst, "Graph analytics on risc-v gpu: Where are the bottlenecks?" in *Spring 2022 RISC-V Week, Location: Paris, France*, 2022.

[10] E. Guthmuller, J. Fereyre, and D. Herrera-Marti, "GPGPUs on FPGAs: A competitive approach for scientific computing ?" in *DATE 2025 - Design, Automation and Test in Europe Conference*, Lyon, France, Mar. 2025. [Online]. Available: https://cea.hal.science/cea-05043041

[11] S. Jeong, L. P. Cooper, J. M. Lee, H. Choi, N. Parnenzini, C. Ahn, Y. Lee, H. Kim, and H. Kim, "Sparseweaver: Converting sparse operations as dense operations on gpus for graph workloads," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2025, pp. 1437–1451.

[12] B. Tine, K. P. Yalamarthy, F. Elsabbagh, and K. Hyesoon, "Vortex: Extending the risc-v isa for gpgpu and 3d-graphics," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 754–766. [Online]. Available: https://doi.org/10.1145/3466752.3480128

[13] P. Jääskeläinen, C. S. Lama, E. Schnetter, K. Raiskila, J. Takala, and H. Berg, "pocl: A performance-portable opencl implementation," *Int. J. Parallel Program.*, vol. 43, no. 5, p. 752–785, Oct. 2015. [Online]. Available: https://doi.org/10.1007/s10766-014-0320-y

[14] J. Li, K. Yang, C. Jin, X. Liu, Z. Yang, F. Yu, Y. Shi, M. Ma, L. Kong, J. Zhou, H. Wu, and H. He, "Ventus: A high-performance open-source gpgpu based on risc-v and its vector extension," in *2024 IEEE 42nd International Conference on Computer Design (ICCD)*, 2024, pp. 276–279.