Accelerating Linear Programming (LP) algorithms on AMD GPUs

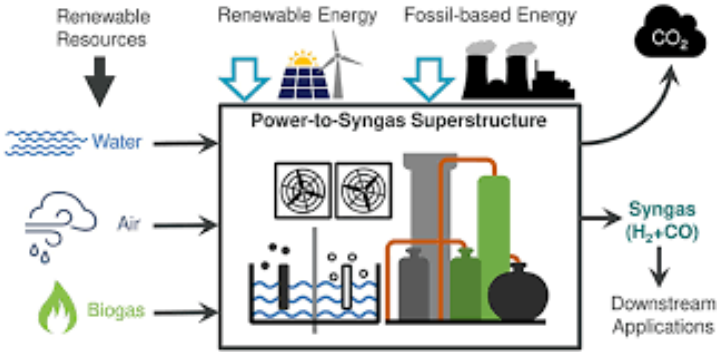
Industry Sponsor: AMD

Industry Mentor: Alireza Kaviani, PhD, Senior Fellow of Engineering

Academic Mentor: Dr. Minxin Zhang, Department of Mathematics, UCLA

Abstract

Linear Programming (LP) is a foundational optimization technique with widespread applications in finance, energy trading, supply chain logistics, and control systems [1]. However, traditional CPU-based LP solvers often struggle to meet the latency and requirements of dynamic, high-dimensional environments [2]. Efficiently solving large-scale LP problems has thus emerged as a significant computational challenge.



This project investigates the acceleration of LP optimization on AMD GPUs using the open-source ROCm (Radeon Open Compute) platform and HIP (Heterogeneous-compute Interface for Portability) to exploit fine-grained parallelism and high memory bandwidth [3]. The core focus is the development of a robust, high-performance, open-source implementation of the Primal-Dual Hybrid Gradient (PDHG) algorithm tailored for general LP problems on AMD hardware. To assess its effectiveness, the implementation is benchmarked against standard LP test sets and established CPU-based solvers. Experiments on both synthetic and real-world datasets—spanning domains such as supply chain optimization and resource scheduling—demonstrate speedups of 6× to 20× over conventional solvers for problems with more than 10,000 variables. These performance gains enable near real-time optimization and support deployment in cloud and edge computing environments. The results establish AMD GPUs as a competitive, cost-effective, and open-source platform for high-performance mathematical programming, particularly in applications where traditional solvers fall short due to latency or scalability constraints [1].

Project background and description

Linear Programming (LP) offers a robust framework for optimizing linear objectives under linear constraints and remains a foundational tool across numerous scientific and industrial domains. While classical solvers—such as the Simplex Method and Interior Point Methods (IPMs) —are well-established, first-order methods like the Primal-Dual Hybrid Gradient (PDHG) algorithm have gained significant traction for large-scale LP problems, particularly those suited for GPU acceleration [4, 5]. PDHG algorithms rely on iterative operations dominated by sparse matrix-vector multiplications (SpMV), dense vector updates, and simple projection steps, making them naturally compatible with parallel computing architectures. Recent efforts, such as cuPDLP.jl [6], have demonstrated that advanced PDHG variants—incorporating adaptive step sizes, restarts, and preconditioning—can outperform traditional CPU-based solvers for certain LP problem classes when deployed on GPU platforms.

However, robust and high-performance open-source PDHG implementations targeting AMD GPUs remain scarce. Addressing this gap is essential for enabling scalable LP optimization on accessible and open

hardware. This is particularly relevant for computationally intensive applications such as logistics, resource scheduling, and power systems optimization—notably Security-Constrained Economic Dispatch (SCED) [7]. The vast scale of modern instances, such as those in the ACTIVSg70k dataset [8], highlights the need for efficient solvers capable of real-time performance on large problem instances.

The objective of this project is to design, implement, and evaluate a Restarted PDHG solver for general LPs, accelerated using AMD GPUs via the ROCm platform. The solver will be built upon the formulation proposed by Lu & Yang (2024) [4], integrating adaptive parameter tuning and restart strategies to enhance convergence. Implementation will be carried out in Python, using the PyTorch framework for GPU parallelism and ROCm compatibility. While designed for general LPs, structural properties of SCED-like problems will guide algorithmic tuning and performance benchmarks. The project aims to deliver an open-source Python module showcasing ROCm-based PDHG performance on large-scale LP benchmarks. A work statement with tentative milestones is attached.

Required skills:

- Solid understanding of Linear Algebra and LP principles
- Strong programming skills in Python.
- Experience with numerical computing libraries (NumPy/SciPy).
- Data structures and algorithms.

Desired skills:

- Experience with PyTorch or another deep learning framework (TensorFlow, JAX).
- Familiarity with sparse matrix formats (CSR) and operations.
- Source code revision control with git.
- Parallel programming concepts (multi-threading/processing).
- GPU programming concepts (HIP is a plus, but focus is on using PyTorch's GPU capabilities).
- Familiarity with power system modeling (Economic Dispatch, OPF, SCED, stretch goal only).

About AMD

Advanced Micro Devices, Inc. (AMD) is a top semiconductor company worldwide. It operates through Data Center, Client, Gaming, and Embedded segments. The company offers x86 microprocessors and graphics processing units (GPUs) as an accelerated processing unit, field programmable gate arrays (FPGA), and adaptive SoC products. AMD is founded more than 55 years ago and is currently valued at roughly $180B.

References

1. Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.
2. Verscheure, D., De Laet, Q., et al. (2009). Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control*, 54(10), 2318–2327.
3. ROCm Developer Hub. https://rocm.docs.amd.com
4. Lu, H., & Yang, J. (2024). cuPDLP.jl: A GPU Implementation of Restarted Primal-Dual Hybrid Gradient for Linear Programming in Julia. arXiv preprint arXiv:2311.12180v4.
5. Chambolle, A., & Pock, T. (2011). A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. Journal of Mathematical Imaging and Vision, 40(1), 120-145.
6. NVIDIA cuOpt Whitepaper. (2022). https://developer.nvidia.com/cuopt
7. Wood, A. J., & Wollenberg, B. F. (1996). Power Generation, Operation, and Control. John Wiley & Sons. (SCED context)
8. National Renewable Energy Laboratory (NREL). ACTIVSg large-scale synthetic power system datasets. [Placeholder: e.g., https://github.com/NREL-Sienna/PowerSystemsTestData/tree/master/ACTIVSg70k (Dataset for stretch goal)
9. Gill, P. E., Murray, W., & Wright, M. H. (2021). *Numerical linear algebra and optimization*. Society for Industrial and Applied Mathematics.
10. Applegate, D., Díaz, M., Hinder, O., Lu, H., Lubin, M., O'Donoghue, B., & Schudy, W. (2025). PDLP: A Practical First-Order Method for Large-Scale Linear Programming. *arXiv preprint arXiv:2501.07018*.
11. Mittelmann, H. D., & Spellucci, P. (2005). *Decision tree for optimization software*.
12. Gay, D. M. (2013, August 22). *NETLIB LP test problems*. Retrieved May 27, 2025, from http://www.netlib.org/lp/ (netlib.org, netlib.org)

Tentative Project Milestones

Week 1 (June 29): Background Learning curve

- Read about LP [9, Chapter 7].
- Read about the PDHG algorithm for LP [5, 10] and its GPU implementation [4].
- Familiarize with ROCm & HIP [3].

Week 2 (July 6): Environment setup

- Configure AMD ROCm and HIP to enable GPU-accelerated computing with PyTorch.
- Build and run ROCm-PyTorch examples to confirm end-to-end compatibility.
- Develop a baseline PDHG solver in PyTorch following [5].
- Validate convergence on the LP benchmark dataset NETLIB/LP [12] ($n < 1000$).

Week 4 (July 20): Integration of advanced techniques

- Integrate diagonal preconditioning, resolving, adaptive restarting, and feasibility polishing [10, 4].
- Implement adaptive step sizes and backtracking line search.
- Conduct parameter-sensitivity studies on random LPs ($n = 5{,}000$–$10{,}000$).

Week 5 (July 27): Benchmarking

- Benchmark against standard LP datasets (NETLIB/LP [12]; Mittelmann's LP [11]).
- Compare iteration counts and runtimes vs. CPU linprog/IPM solvers.
- Midterm presentation

Week 7 (August 10): Profiling, tuning, and extended benchmarking

- Profile GPU kernels and memory flows with rocProfiler and PyTorch Profiler.
- Optimize sparse kernels and data layouts.
- Test on the new LP benchmark dataset in [10].
- Benchmark on ACTIVSg70k Security-Constrained Economic Dispatch instances [7,8].

Week 8 (August 17): Robustness and scalability analysis

- Conduct robust checks on ill-conditioned or degenerate LPs.
- Implement fallback strategies for convergence failure and poor scaling behavior.
- Analyze throughput vs. batch size and document GPU utilization.

Week 9 (August 22): Project wrap-up

- Finalize code release with documentation and Jupyter notebooks.
- Compile results and draft the final report.
- Final presentation.