Towards Efficient Methods for Higher Quality Weighted Matching Applications

Michael Mandulak* Sayan Ghosh† S M Ferdous† Mahantesh Halappanavar†
George M. Slota *

1 Introduction

As a classical graph problem prevalent across a variety of fields, matching, or the linear assignment problem, seeks to map one set of discrete entities to another. This mapping is generated within a wide range of contexts, whether that be instances of residents to hospitals, computer vision pattern recognition, or numerical linear algebra applications, among countless others. Specifically, we consider maximum weighted matching (MWM) instances.

In modern applications as input graph sizes become increasingly large, demands for efficient, HPC-enabled approximate matching solutions are high. These solutions often face inefficiencies through irregularity in real-world graph data, high data movement costs and memory restrictions on device, among others. Despite this, there has been significant work using CPU parallel platforms, GPU hardware, or some combination of the two [1, 2, 3].

Analyzing these, we take notice of an approaching "performance wall" in traditional $\frac{1}{2}$-approximate MWM approaches. Thus, we aim to further advancements through alternative platforms and methods for application. Our goals here are two-fold: (1) improve communication costs through a block-partition-based 2D sparse-communication grid; (2) explore parallel implementations of higher quality-yielding matching methods. This work discusses each contribution towards these goals.

2 Prior Works

Background: While numerous approximation methods have been explored in the past decades, a majority of parallel MWM methods are based on the $\frac{1}{2}$-approximate locally-dominant framework [4]. Modern parallel methods take advantage of the vertex-centric nature of this computation, or the similar proposal-based system known as the Suitor method [5]. Both systems have seen work in distributed and shared memory parallel domains with similar efficiencies.

Performance Wall: Detailing implementation progression, we provide a timeline of performance improvements in Figure 1. The decreasing trends of performance gains from SR-OMP to the GPU instances of SR-GPU and LD-GPU, respectively, highlight the performance wall present in parallel implementations of traditional $\frac{1}{2}$-approximate MWM solutions. This is largely due to the tradeoffs of communication costs with parallel performance and the requirements of synchronization within computation rounds. These complexities are furthered by limited device memory, imposing data movement overheads that offset the performance gains yielded by GPUs. These concepts are discussed in detail in [3].

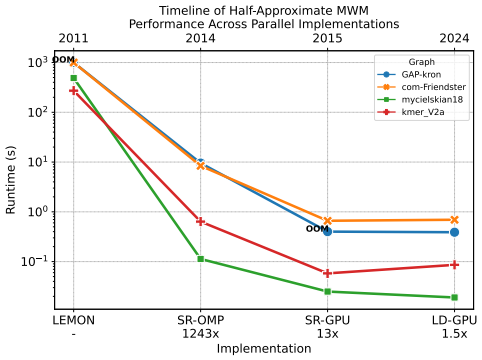

Figure 1: Implementation timeline from the sequential LEMON optimal implementation [6] to the OpenMP Suitor (SR-OMP) [2], GPU Suitor (SR-GPU) [1] and multi-GPU locally-dominant (LD-GPU) [3] implementations and their average runtime improvements upon the previous. OOM denotes out-of-memory constraints due to hardware/input size.

Higher Quality Applications: Towards matching solutions of higher quality, we first discuss applicability as demonstrated in prior works. There have been recent works detailing the usage of $\frac{2}{3} - \epsilon$-approximate methods [7] within algebraic multigrid (AMG) [8] and in the security domain towards data privacy on general graphs [9]. In the AMG instance, the authors show improvements to convergence and runtimes by augmenting the coarsening procedure with a method based on the $\frac{2}{3} - \epsilon$-approximate algorithm. Similarly, in the data privacy instance, the authors develop the first secure protocol for matching on general graphs, requiring $O(N\ log\ N)$ rounds with N vertices. Considering this growing usage of higher quality-yielding ap-

*Rensselaer Polytechnic Institute
†Pacific Northwest National Laboratory

proximate methods in modern work, we target efficient implementations for direct application.

3 Results

2D Communications: To address growing dependencies on efficient communication, we experiment with a 2D graph distribution. Such a distribution creates a 2D block partitioning of the adjacency matrix [10], with each ranking 'owning' the edges and relevant vertices within a given block. 2D distributions and associated communication patterns have been widely used to enable scalability in benchmark problems such as triangle counting and breadth-first search, but have not yet been widely adopted for algorithms as complex as MWM.

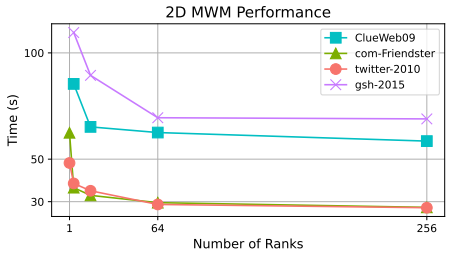

Figure 2: Scaling performance of large-scale graphs using the 2D MWM implementation.

As a test instance for MWM scalability, we implement a locally-dominant method using our 2D framework. We show preliminary results in Figure 2. From these results, we see high initial runtimes that scale to 256 GPUs, with the most significant improvements in the instances up to 64 ranks. We specifically note that the two instances small enough to fit on a single GPU (twitter-2010 and com-Friendster) strong scale all the way to 256 GPUs. These results are promising, and suggest an optimized implementation might be able to break the performance wall discussed above.

Higher Quality Matching: Towards higher quality matching solutions, we consider the work in [7], proposing two $\frac{2}{3} - \epsilon$-approximate methods. We draw from both, implementing the random order method for our sequential version and the deterministic method for an OpenMP implementation. For each, we devise optimizations to improve runtime performance relative to the greedy algorithm. Optimizations include early stoppage conditions to preemptively determine if an augmentation is viable within gain calculations and limiting neighborhood scans among two-hops by using the lesser degree vertex, to name a few. For the parallel implementation, we are experimenting with modifications to the greedy choice of 2-augmentations, using a variety of locally-dominant and maximal independent set-based methods to improve weight gains per iteration.

From experimental tests of the $\frac{2}{3} - \epsilon$-approximate method, we seek to characterize a baseline relative to standard $\frac{1}{2}$-approximate methods. For this, we refer to the greedy method, which simply sorts all edges and chooses the highest weight, non-overlapping edge to add into the matching until this is no longer possible. Relative to the greedy method on a set of small, real-world graph instances (<500K edges), we observe an average 3% improvement on total matching weight with comparable runtimes. Extending this to larger instances (<50M edges), we note that initial tests on dense cases that are typically problematic for $\frac{1}{2}$-approximate methods [3] yield near optimal weights with runtimes 4x-5x higher than the greedy method under a sequential implementation. Through our OpenMP implementation, we seek to improve this gap to yield higher weights with comparable runtimes to a $\frac{1}{2}$-approximate case.

4 Future Works

Towards each goal, we note the continual development and optimization of our 2D $\frac{1}{2}$-approximate and $\frac{2}{3} - \epsilon$ implementations. Our intent is to extend parallel implementations towards direct applications, with current experiments testing usability in more general AMG settings as well as within graph partitioning. Our preliminary results among modern works in the field demonstrate high potential for mass usage of $\frac{2}{3} - \epsilon$-approximate matching methods within respective applications given efficient parallel implementations, and we are currently working towards such a goal.

References

[1] M. Naim *et al.*, "Optimizing approximate weighted matching on Nvidia Kepler K40," *HiPC*, 2015.

[2] F. Manne *et al.*, "New effective multithreaded matching algorithms," in *IPDPS*, 2014.

[3] M. Mandulak, S. Ghosh, S. M. Ferdous, M. Halappanvar, and G. Slota, "Efficient weighted graph matching on GPUs," in *SC'24*, 2024.

[4] R. Preis, "Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs," in *STACS*, 1999.

[5] A. Pothen *et al.*, "Approximation algorithms in combinatorial scientific computing," *Acta Numerica*, 2019.

[6] LEMON Contributors, "LEMON: Library for efficient modeling and optimization in networks."

[7] S. Pettie *et al.*, "A simpler linear time $\frac{2}{3} - \epsilon$ approximation for maximum weight matching," *Information Processing Letters*, 2004.

[8] P. D'Ambra *et al.*, "AMG preconditioners based on parallel hybrid coarsening and multi-objective graph matching," in *PDP*, 2023.

[9] A. Brüggemann *et al.*, "Secure maximum weight matching approximation on general graphs," in *WPES'22*, 2022.

[10] E. G. Boman *et al.*, "Scalable matrix computations on large scale-free graphs using 2D graph partitioning," in *SC'13*, 2013.