Paper Selection Proposal
Algorithms: Design and Analysis

Hammad Malik (hm08298)
Mahrukh Yousuf (my08055)

March 29th, 2025

- **Title:** New Algorithms for All Pairs Approximate Shortest Paths
- **Authors:** Liam Roditty
- **Conference:** 55th Annual ACM Symposium on Theory of Computing (STOC 2023)
- **Year:** 2023
- **DOI/Link:** `https://doi.org/10.1145/3564246.3585197`

  `https://dl.acm.org/doi/pdf/10.1145/3564246.3585197`

Paper Overview

This paper by Roditty explores new algorithms for all pairs approximate shortest paths (APASP) in unweighted undirected graphs. The goal is to improve the approximation time for these paths without relying on Fast Matrix Multiplication (FMM). The paper discusses algorithms that compute additive and multiplicative approximations with faster time complexity than prior works, specifically targeting dense graphs.

Key Definitions

- *All-Pairs Shortest Paths (APSP)*: The problem of finding the shortest paths between every pair of vertices in a graph.

- *Approximate Shortest Paths*: An approximation where the computed shortest paths are within a certain error bound (additive or multiplicative).
- *Multiplicative 2 Approximation*: An approximation where the computed distance between two vertices is at most twice the true shortest path distance.
- *Additive Approximation*: An approximation where the computed distance differs from the true shortest path by no more than a constant.

Major Results and Proofs

- **Improved Time Complexity for Dense Graphs:**
  - The paper introduces algorithms that improve the running time to $(n^{2.29})$ for dense graphs, outperforming previous methods.
- **Multiplicative Approximation:**
  - New algorithms compute a multiplicative 2 approximation in $(minn^{1/2}m, n^{9/4})$ time.
- **Additive Approximation:**
  - Algorithms that compute additive 2 approximations with improved bounds for graphs with distances greater than 2.

Summary

This paper presents algorithms for computing all-pairs approximate shortest paths with improved time complexity. The main improvements focus on dense graphs and provide both additive and multiplicative approximations that perform better than previous algorithms without the need for Fast Matrix Multiplication (FMM). These results have significant implications for optimizing graph algorithms and applications like routing, network analysis, and transportation planning.

Justification

This paper is highly relevant to our project as it addresses the challenge of optimizing shortest path algorithms for large-scale graphs. The improved

approximation techniques discussed in the paper could be applied to real-world graph problems, such as network routing or transport logistics, where exact shortest paths are computationally expensive. By implementing and testing these algorithms, we aim to contribute to the optimization of such systems.

Implementation Feasibility

Implementation of New Algorithms

We plan to implement the new algorithms for all pairs approximate shortest paths. The focus will be on both additive and multiplicative 2 approximations. We will test these implementations on various graph types, including sparse and dense graphs, to evaluate the algorithm's performance in real-world scenarios.

Experimentation with Baseline Algorithms

For comparison, we will also implement baseline algorithms such as the all-pairs shortest path (APSP) and single-pair shortest path algorithms. We will analyze and compare the time complexity and approximation accuracy of the new algorithms against these standard methods.

Implementation Considerations

- **APASP Algorithm:** The core algorithm will be implemented following the pseudocode in the paper, with adjustments to handle different graph types.
- **Verification:** We will implement tests to validate the correctness of the shortest path approximations and compare the results with exact computations.
- **Baseline Comparisons:** We will use algorithms like Single Pair Shortest Path and All Pair Shortest paths which we are already studying for comparison to assess the improvements in efficiency and accuracy.

Feasibility of Implementation

The implementation of these algorithms is feasible using existing graph libraries such as NetworkX in Python. The algorithms rely on well-understood

graph traversal methods, and we can leverage existing implementations to optimize the experiments.

Team Responsibilities

- **Hammad Malik:** Implementing the new algorithms and performing time complexity analysis.
- **Mahrukh Yousuf:** Conducting experiments to compare the new algorithms with baseline algorithms, preparing the final report, and presenting the results.