Georgia Tech Spring 2025 CS 3511: Honors Algorithms

Final Project

Authors: *Arjun Sheshadri, Vilohith Gokarakonda, Yiqi Sun, Tyler Huang*
Date: Wednesday 4/16/25

1 Introduction

Suppose if you have a graph $G = (V, E)$ with a given adjacency matrix, which represents where any two nodes are connected to each other. Suppose you wanted to find the shortest path to one node, using all nodes as a starting point.

Shortest distances between all pairs of nodes in a graph is important for its real world applications in communication networks, social graphs, and more. Edges in such graphs can be added and removed at any time, so being able to efficiently maintain all pairs distances is crucial. Using the Floyd-Warshall Algorithm, we are able to accomplish this in $\mathcal{O}(n^3)$ time. However, this becomes inefficient for larger or frequently changing graphs.

This report explores a *dynamic algorithm* described in Jan van den Brand's notes for maintaining All-Pairs-Shortest-Paths (`APSP`) in directed graphs with or without edge weights. Instead of recalculating shortest distances from scratch every update, it efficiently maintains a data structure where all distances can be updated in $\tilde{\mathcal{O}}(n^{2.5})$ time when an edge to a single vertex is added or removed. Specifically, it uses concepts from dynamic algebraic algorithms, involving polynomial matrix inverses to represent path information, and extending them to full distances with random sampling and Dijkstra's algorithm.

In this report, we present the problem statement, and the technical background relevant to the paper, including ring algebra, polynomial matrices, and how edges are updated. We also cover the steps to the solution, and connect this approach to the original groundbreaking paper by Sankowski, which can do updates in $\tilde{\mathcal{O}}(n^{1.932})$ randomized time and queries in $\tilde{\mathcal{O}}(n^{1.288})$ randomized time. Specifically, we compare their mathematical foundations, update operations, applications, and time complexities.

2 Problem Statement

Given a graph $G(V, E)$, we want to develop a data structre that can maintain `APSP` dynamically with an initial overhead $\tilde{\mathcal{O}}(n^{3.5})$ time and supports queries and updates in $\tilde{\mathcal{O}}(n^{2.5})$ time. Since this algorithm is optimized for maintaining `APSP` for changing graphs (hence requires a dynamic algorithm), having a time complexity of $\tilde{\mathcal{O}}(n^{2.5})$ is more efficient compared to the naive approach which takes $\mathcal{O}(n^3)$ time for queries and updates.

Specifically, we will be working towards proving the following theorem:

Theorem 1.0.0: *There exists a data structure that supports the following operations*:

1. INITIALIZE$(G = (V, E))$ *Initialize an* n*-node graph and return* ***APSP*** *in* $\tilde{\mathcal{O}}(n^{3.5})$ *time.*
2. UPDATE$(v,\ E^+,\ E^-)$ *Given a vertex* v *and two sets of edges* $E^+ \subseteq (\{v\} \times V \cup V \times \{v\})$ *to insert and* E^- *to delete (all incident to* v*), update* G *and return the new* ***APSP*** *in* $\tilde{\mathcal{O}}(n^{2.5})$ *time.*

Additionally, we will be extending this theorem using Sankowski's paper:

Theorem 1.0.1: *There exists a data structure that supports the following operations*:

1. INITIALIZE$(G = (V, E))$ *Preprocess in* $\tilde{\mathcal{O}}(n^3)$ *time.*
2. UPDATE(e) *Insert or delete a single edge* $e \in V \times V$ *in* $\mathcal{O}(n^{1.932})$ *randomized time.*
3. QUERY(s, t) *Return the current distance* $\mathrm{dist}_G(s, t)$ *in* $\mathcal{O}(n^{1.288})$ *randomized time.*

All operations succeed with high probability over the random choices.

3 Technical Background

3.1 Abstract Algebra Overview and Ring Operations

3.2 Naive APSP

3.3 Polynomial Matrices

3.4 Edge Updates

3.5 Hitting Sets

4 Solution

4.1 Algorithm

4.2 Time Complexity Analysis

5 Extension and Comparison

5.1 Broader Algebraic Structures

5.2 General Update Query

5.3 Multiple Matrix Functions

5.4 Better Complexity Bounds

References

1. Jan van den Brand's Notes
2. Sankowski's Paper