Efficient Parallel Algorithm for Shortest Path Updates in Dynamic Networks at Scale

ARINDAM KHANDA,

akkcm@mst.edu, Missouri University of Science and Technology, Rolla, USA

SAJAL K. DAS (ADVISOR),

sdas@mst.edu, Missouri University of Science and Technology, Rolla, USA

CCS Concepts: • **Computing methodologies** → **Parallel algorithms**.

Additional Key Words and Phrases: Shortest Path Algorithm, Dynamic Network, GPU

ACM Reference Format:

Arindam Khanda and Sajal K. Das (Advisor). 2021. Efficient Parallel Algorithm for Shortest Path Updates in Dynamic Networks at Scale.

1 INTRODUCTION

The application of graphs (Networks) are versatile, and they are applied to various fields, including but not limited to social network analysis, transportation logistics, biological and genetic interaction study, IP traffic routing, and resource allocation in IoT networks.

All these domains deal with a massive amount of data and require large-scale graphs to model them. These graphs are often dynamic in nature, i.e., the structure of the graph changes with time.

The Single Source Shortest Path (SSSP) problem, which finds out the shortest distance of all vertices from a source vertex, often appears in different scenarios. In a large-scale dynamic network, finding SSSP becomes challenging due to the rapid structural changes in the graph and scalability issues. Although numerous parallel shortest path algorithms on static networks have been proposed in the literature, the SSSP problem in large-scale dynamic networks require more attention due to its continual appearances in modern scenarios.

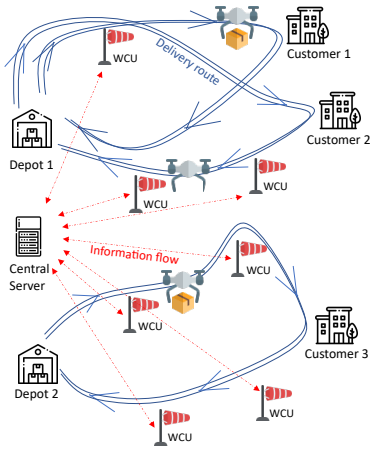


Fig. 1. A DBDS scenario.

1.1 Motivational Scenario —- Dynamic Route Selection for Drone Based Delivery System in Varying Wind Condition

An efficient drone-based delivery system (DBDS) aims to complete a route from the depot to the customer and go back to the depot with minimum energy usage. However, the simple shortest delivery route is not always an optimal choice

Manuscript submitted to ACM

in varying wind conditions, as the wind direction and speed can significantly affect the energy usage for flying [3]. In a DBDS, wind control units (WCU) collect the wind-related data and send them to a central server periodically for analysis [4]. The complexity and scale of the problem increase when multiple depots are involved, and drones with less computation power are used. In such scenarios, the central server becomes responsible for computing the updated delivery paths simultaneously for each drone [1]. This scenario resembles SSSP problem in a large-scale dynamic network.

2 PARALLEL SHORTEST PATH UPDATE

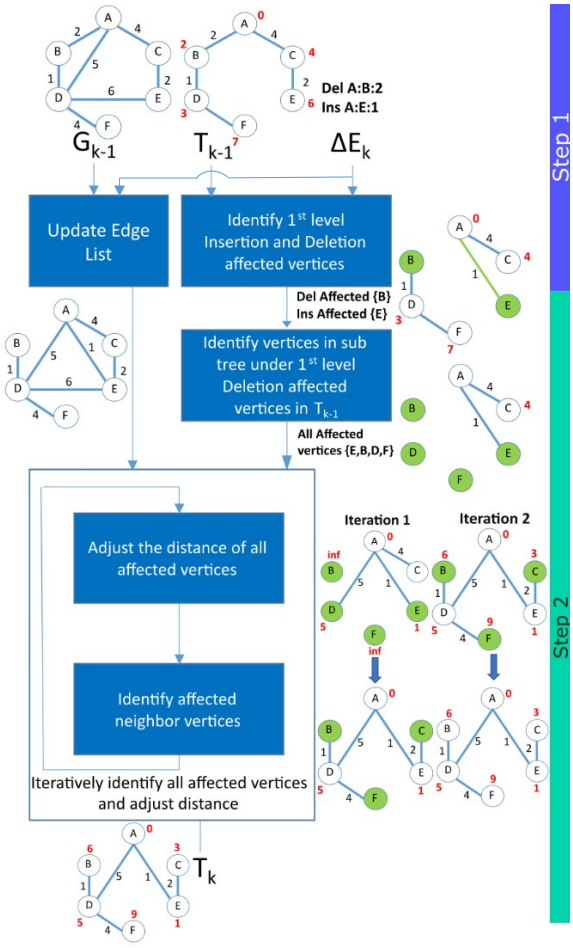


Fig. 2. SSSP update steps with example.

Here we present a parallel algorithm for updating SSSP in dynamic networks. Instead of recomputing SSSP for any graph structural changes, our algorithm uses the previous shortest path parameters and updates them to find the current shortest path [2, 5]. We use a tree structure, called SSSP-tree to maintain the distance from the source-vertex and the parent of each vertex in the graph. Let $G_{k-1}(V_{k-1}, E_{k-1})$ is a graph at time step $k-1$, and one SSSP-tree of G_{k-1} is T_{k-1}. Let the structural change of the graph between time step $k-1$ and k is captured by a set of changed edges ΔE_k (includes inserted and deleted edges). Our goal is to compute the SSSP tree T_k at time step k, based on the structure and edge weights of the tree T_{k-1} and the changed edge set ΔE_k.

2.1 SSSP update steps

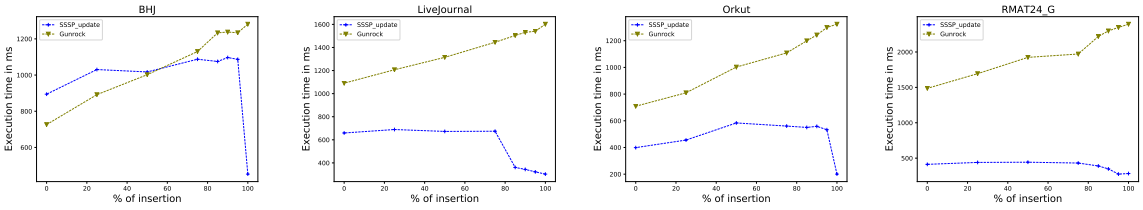
Step 1: Changed edges are processed in parallel and checked if the deletion/insertion is affecting T_{k-1}. A vertex is marked affected if a deletion/insertion of an edge changes the distance of the vertex from the source.

Step 2: If an edge deletion disconnects a vertex from the SSSP tree, the descendants of the disconnected vertex would also get disconnected. Therefore, the algorithm traverse through the subtree rooted at the vertices affected by any edge deletion and disconnects the descendant vertices (also marked affected).

Next, for each affected vertex v, the distance from the source $Dist[v]$ is adjusted by visiting their neighbors. For $e(v,n) \in E_k$ with edge weight $W(v,n)$, if $Dist[v] > Dist[n] + W(v,n)$, then distance of v is updated to $Dist[n] + W(v,n)$ and marked as affected. On the other hand, if $Dist[n] > Dist[v] + W(v,n)$, then the distance of n is updated and marked affected. This process is continued iteratively until there is no affected vertex left.

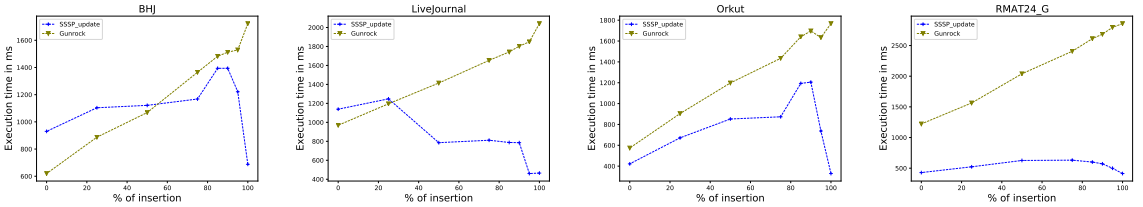
In our GPU implementation, at step 1, all changed edges are distributed among the CUDA threads in a grid-stride-loop manner. At step 2, the affected vertices in each iteration are distributed similarly. In our implementation we use a CUDA __ballot_sync based filter method to select the unique affected vertices in each iteration. This filter method decreases overlapping or redundant work.

3 RESULTS AND ANALYSIS



(a) BHJ (b) LiveJournal (c) Orkut (d) RMAT24

Fig. 3. $\Delta E = 50$ millions.



(a) BHJ (b) LiveJournal (c) Orkut (d) RMAT24

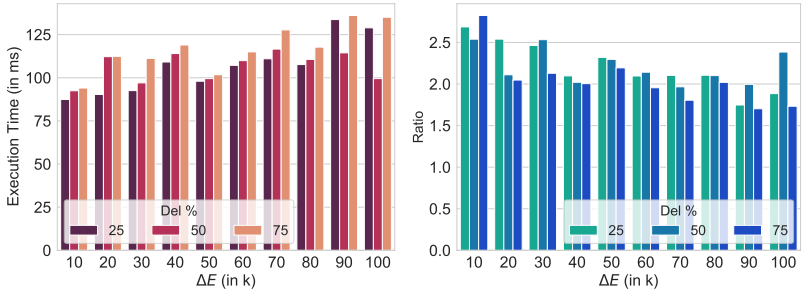
Fig. 4. $\Delta E = 100$ millions.

To prove the efficacy of our algorithm empirically, we compared the performance of the GPU implementation of our algorithm with Gunrock, the state-of-the-art GPU-based implementation for SSSP computation. We used 50-million and 100-million edge updates with different mixes of edge deletions and insertions. $p\%$ insertion in a set of edge updates implies $p\%$ of total changed edges are for insertion, and the rest of the edges are for deletion. Although, the execution time depends on the number of affected vertices, which again depends on graph architecture and the position of the change, in most cases, our algorithm outperforms Gunrock and shows significantly better performance when the percentage of insertion is high. As Gunrock computes shortest path from scratch in parallel, each time, we supplied Gunrock a processed graph with all inserted edges added and deleted edges removed from the original graph for computing the shortest path. Therefore, when the percentage of deletion was high, i.e. most edges were deleted, we supplied processed graph with very few edges to Gunrock. Hence, in case of the batches with high deletion percentage (or low insertion percentage), Gunrock executed faster than our algorithm.

3.1 Results from Drone Based Delivery Scenario

We modeled a DBDS scenario using *roadnet-PA* and applied our SSSP update algorithm to find shortest delivery route in presence of changed edges. We took different set of changed edges (10,000 changed edges to 100,000 changed edges

in each batch. Percentage of deleted edges were varied among 25%, 50% and 75%) and recorded the execution time. Figure 5a shows the execution time of our algorithm when applied in a DBDS scenario and Figure 5b shows the ratio of Gunrock's execution time and our algorithm's execution time. As this ratio is always more greater than 1, it indicates in DBDS scenario our algorithm always performs better than Gunrock.



(a) Execution Time (b) Execution Time Ratio

Fig. 5. Performance analysis in a DBDS scenario

4 CONCLUSIONS

We present a parallel shortest path update algorithm for large-scale dynamic networks. Our approach does not consider any domain constraints or dataset-specific optimization. However, domain knowledge can improve the solutions. E.g., in a DBDS , multiple customer vertices in the close proximity can be grouped and their centroid can be considered as single meta vertex for finding an approximated shortest path. In the future, we plan to use the domain knowledge and predictive algorithms for SSSP.

ACKNOWLEDGMENTS

This work is supported by the NSF OAC grants 2104078 and 1725755.

REFERENCES

[1] Arindam Khanda, Federico Coro, F Betti Sorbelli, Cristina M Pinotti, and Sajal K Das. 2021. Efficient Route Selection for Drone-based Delivery Under Time-varying Dynamics. In *18th International Conference on Mobile Ad-Hoc and Smart Systems (MASS). IEEE.*

[2] Arindam Khanda, Sriram Srinivasan, Sanjukta Bhowmick, Boyana Norris, and Sajal K Das. 2021. A Parallel Algorithm Template for Updating Single-Source Shortest Paths in Large-Scale Dynamic Networks. *IEEE Transactions on Parallel and Distributed Systems* (2021).

[3] Ty Nguyen and T. Au. 2017. Extending Range of Delivery Drones by Exploratory Learning of Energy Models.. In *AAMAS*. 1658–1660.

[4] Francesco Betti Sorbelli, Federico Corò, Sajal K Das, and Cristina M Pinotti. to appear, 2021. Energy-Constrained Delivery of Goods With Drones Under Varying Wind Conditions. *IEEE Transactions on Intelligent Transportation Systems* (to appear, 2021).

[5] Sriram Srinivasan, Sara Riazi, Boyana Norris, Sajal K Das, and Sanjukta Bhowmick. 2018. A shared-memory parallel algorithm for updating single-source shortest paths in large dynamic networks. In *IEEE 25th International Conference on High Performance Computing*. 245–254.