CANDY: An Efficient Framework for Updating Properties on Large Dynamic Networks

Aashish Pandey
University of North Texas
Denton, USA
aashishpandey@my.unt.edu

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

Sriram Srinivasan
University of Oregon
Eugene, USA
sriram.srinivasan@vcuhealth.org

Sanjukta Bhowmick
University of North Texas
Denton, USA
sanjukta.bhowmick@unt.edu

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

Boyana Norris
University of Oregon
Eugene, USA
norris@cs.uoregon.edu

***Abstract*—Queries on large graphs use the stored graph properties to generate responses. As most of the real-world graphs are dynamic, i.e., the graph topology changes with time, and hence the related graph properties are also time-varying. In such cases, maintaining correctness in stored graph properties requires recomputation or update on previous properties. Here, we present an efficient framework, CANDY for updating the properties in large dynamic networks. We prove the efficacy of our general framework by applying it to update graph properties such as Single Source Shortest Path (SSSP), Vertex Coloring, and PageRank. Empirically we show that our shared-memory parallel and NVIDIA GPU-based data-parallel implementations perform better than the state-of-the-art implementations.**

***Index Terms*—Large Dynamic Network, Parallel Graph Property update, Scalable Algorithms**

I. Introduction

The systems of interacting entities are often modeled as networks consisting of nodes representing the entities, and edges representing the interactions. Analysis of such networks leads to interesting insights about the system. Network analysis has applications in various domains including bioinformatics, drug discovery, internet routing, and recommendation systems. For large networks, computing and storing graph properties efficiently is itself an expensive operation due to the irregular memory access during graph traversal. The real-world networks, which are dynamic in nature, possess additional challenges of maintaining the correct property values with the change in network structure.

Most of the existing algorithms perform the ad-hoc computation of properties on network snapshots at different time instances. This technique of applying static graph algorithms on dynamic networks is experimentally expensive, involving redundant operations. Furthermore, the challenge increases with the increase in the size of the network [1]–[4].

We introduce our work-in-progress software platform CANDY (Cyberinfrastructure for Accelerating Innovation in Network Dynamics) for managing and analyzing large dynamic networks. We show that our framework can be used to produce efficient parallel algorithms on shared memory, and GPU platforms for updating different graph properties such as Single Source Shortest Path (SSSP), vertex coloring, and PageRank in large networks. We compare our implementation with the state-of-the-art techniques to compute those properties on a set of real-world and synthesized networks.

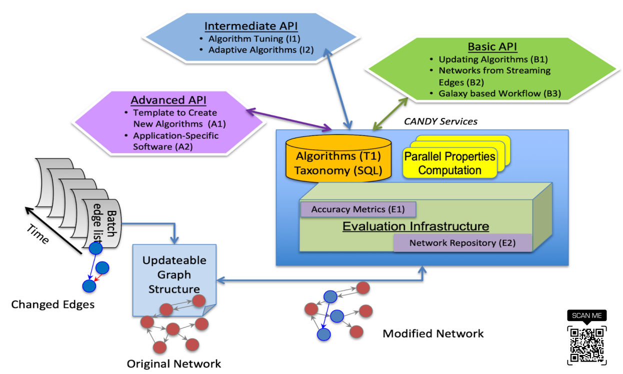


Fig. 1: Overview of CANDY

II. CANDY: Cyberinfrastructure for Accelerating Innovation in Network Dynamics

In this section we introduce CANDY, a software framework envisioned for dynamic network computations that aims to:

- Create a novel hierarchical taxonomy of network analysis algorithms that allows for layered specification of parallel algorithms based on multiple parameters.
- Include templates for creating new scalable, parallel algorithms for dynamic network analysis.
- Provide algorithms to partition streaming sets of nodes and edges into network snapshots at changing points.

We intend to apply CANDY to numerous areas, including evolutionary genomic computations and cost-effective operation of complex mining applications. The functional diagram of CANDY is given in Figure 1.

III. Graph Property Update Framework

Figure 2 shows our parallel framework for updating properties in large dynamic networks. Here we describe it with an example of updating the Page Rank (PR) algorithm. The framework has mainly three steps. **Step 1. Preprocessing:** Change edges not affecting the graph property are discarded, dangling vertices are removed; **Step 2. Adaptive Switch:** It evaluates the total percentage of affected and high PR vertices. If more than 50% vertices are affected then the adaptive switch uses a recomputation algorithm to compute PR from scratch. Else an update algorithm is used in next step. **Step 3. Update affected Nodes:** It is an iterative process of updating the property. In each iteration, the property is updated in parallel for the affected vertices in the current frontier. The next frontier is generated by visiting the neighbors and selecting the possible set of affected vertices. We introduce an Early Termination Condition (ETC) to keep track of vertices that are active during a given iteration. ETC helps the framework avoid redundancy and makes the update algorithm scalable.

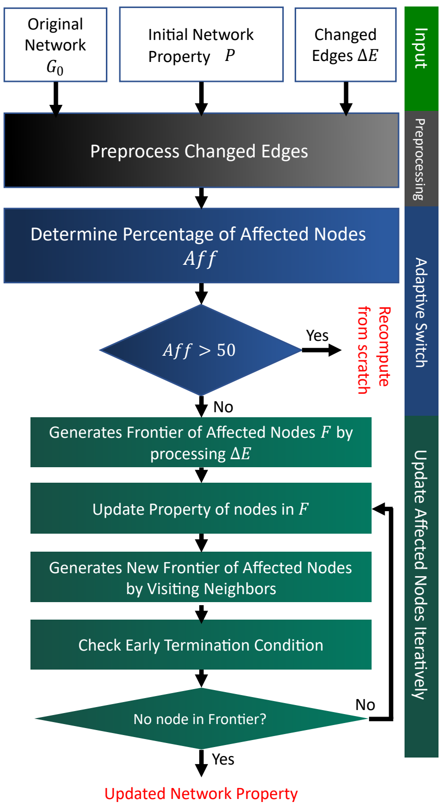


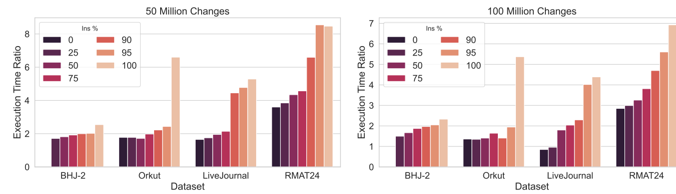
Fig. 2: Adaptive Framework for Property Update.

IV. Experimental Results

We used our framework to develop and implement parallel algorithms to update SSSP, Vertex Coloring, and PageRank in large dynamic networks. In our experiment, $x\%$ Ins in ΔE indicates total $\Delta E * x/100$ edge insertion and $\Delta E * (1 - x/100)$ edge deletion.

A. Single Source Shortest Path Update

Our NVIDIA CUDA-based SSSP update implementation [1] outperforms state-of-the-art Gunrock's [5] GPU-based SSSP implementation (recomputation approach) in most cases. Figure 3a, 3b shows the ratio of SSSP recomputation [5] time, and SSSP update time on GPU

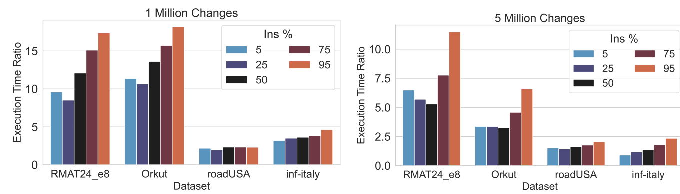


(a) 50 million change edges (b) 100 million change edges

Fig. 3: SSSP update: Comparison with Gunrock shortest path.

B. Vertex Color Update

Figure 4a, and 4b shows the ratio of color recomputation [6] time (using GPU-based Kokkos coloring), and vertex color update time on GPU. Experimental result shows that the execution ratio is more than 1 in most cases, i.e., the update algorithm takes less time than recomputation.



(a) 1 million change edges (b) 5 million change edges

Fig. 4: Color Update: Comparison with static coloring [6]

C. Page Rank Update

Figure 5 demonstrates how our shared-memory-based PR update algorithm scales well on real-world networks.

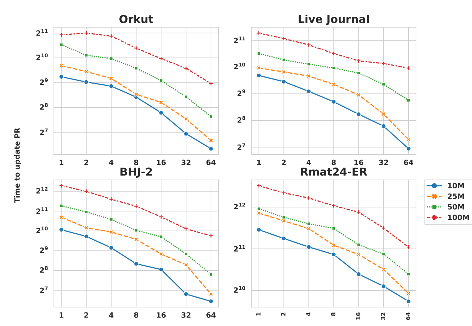


Fig. 5: Page Rank update scalability

V. Conclusion

We introduce the CANDY platform, a parallel framework to develop algorithms for updating properties of large dynamic networks. We demonstrate our framework with an example of PageRank update algorithm. We implement our algorithms on shared memory and GPU architecture and show that our implementations update different graph properties faster than the current state-of-the-art methods. We plan to publish the basic CANDY APIs soon for the end users.

Acknowledgments

This work was partially supported by the NSF projects SANDY (Award # OAC-1725755) and CANDY (Award # OAC-2104115, 2104076, 2104078)

REFERENCES

[1] A. Khanda, S. Srinivasan, S. Bhowmick, B. Norris, and S. K. Das, "A parallel algorithm template for updating single-source shortest paths in large-scale dynamic networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 4, pp. 929–940, 2022.

[2] A. Khanda, F. Corò, F. B. Sorbelli, C. M. Pinotti, and S. K. Das, "Efficient route selection for drone-based delivery under time-varying dynamics," in *IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, 2021, pp. 437–445.

[3] A. Khanda, F. Corò, and S. K. Das, "Drone-truck cooperated delivery under time varying dynamics," in *Proceedings of the 2022 Workshop on Advanced Tools, Programming Languages, and PLatforms for Implementing and Evaluating Algorithms for Distributed Systems*, ser. ApPLIED '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 24–29. [Online]. Available: https://doi.org/10.1145/3524053.3542743

[4] S. Srinivasan, S. Pollard, S. K. Das, B. Norris, and S. Bhowmick, "A shared-memory algorithm for updating tree-based properties of large dynamic networks," *IEEE Transactions on Big Data*, vol. 8, no. 2, pp. 302–317, 2022.

[5] Y. Wang, A. Davidson, Y. Pan, Y. Wu, A. Riffel, and J. D. Owens, "Gunrock: A high-performance graph processing library on the gpu," in *Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2016, pp. 1–12.

[6] S. Rajamanickam, S. Acer, L. Berger-Vergiat, V. Dang, N. Ellingwood, E. Harvey, B. Kelley, C. R. Trott, J. Wilke, and I. Yamazaki, "Kokkos kernels: Performance portable sparse/dense linear algebra and graph kernels," *arXiv preprint arXiv:2103.11991*, 2021.