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

Aashish Pandey[1], Arindam Khanda[2], Sriram Srinivasan[3], Sanjukta Bhowmick[1], Sajal K. Das[2],and Boyana Norris[3]

[1]University of North Texas, [2]Missouri University of Science and Technology, [3]University of Oregon

Parallel Algorithms for Updating Large Dynamic Networks

- Analysis of dynamic networks has become an important tool for studying large-scale systems of interacting entities that change over time.

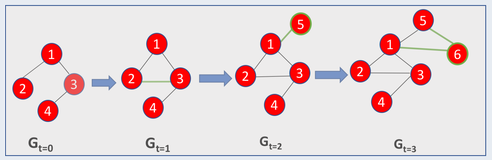


Figure: Changing of graph over time

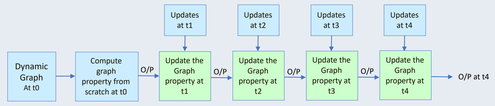


Figure: General approach for dynamic network algorithms

CANDY Architecture

- **CANDY**: a parallel, scalable, extendable, and user-friendly software platform for updating important properties of dynamic networks
- Support parallel dynamic network algorithm development on distributed memory, shared memory, and GPUs, and their use through user-friendly interfaces
- Comprehensive cyberinfrastructure supporting innovative research challenges in large-scale, complex, dynamic networks

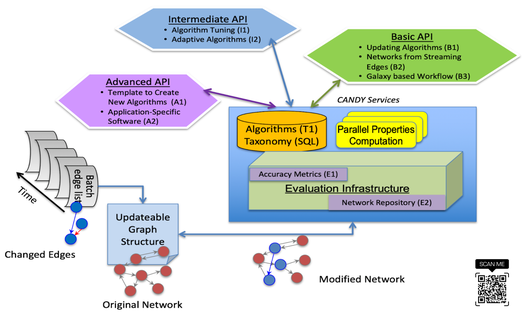


Figure: Overview of CANDY(Cyberinfrastructure for Accelerating Innovation in Network Dynamics)

Challenges and Intellectual Merits

Challenges

- Designing an efficient algorithm to update with minimal graph traversal.
- Determining the impact of changes and consider which approach is best re-computation or update algorithm.
- Creating parallel graph computation software infrastructure for modern heterogeneous architectures.

Intellectual Merits

- Includes templates for creating new scalable, parallel algorithms for dynamic network analysis
- Provides functionality and tools to create new algorithms or modify existing ones, catering to users with varying expertise
- Provides algorithms to partition streaming sets of nodes and edges into network snapshots at changing points

Graph Properties

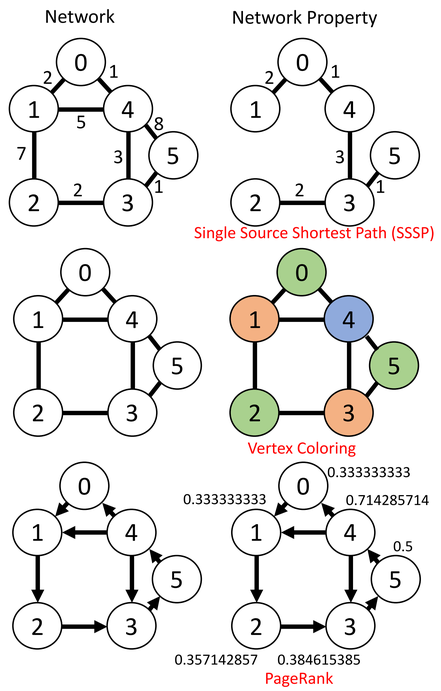


Figure: Example of common graph properties

Adaptive PageRank

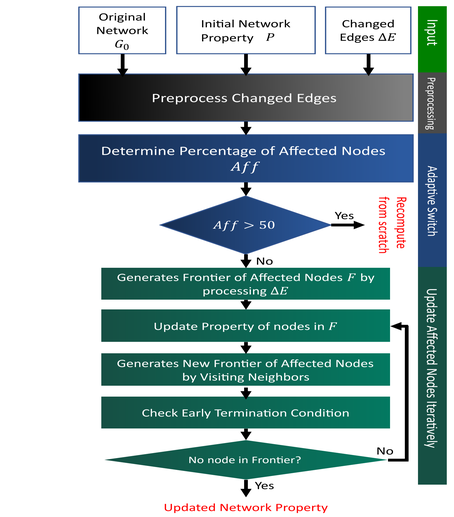


Figure: Adaptive framework template

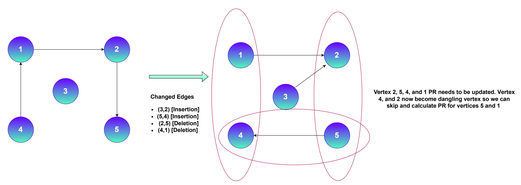
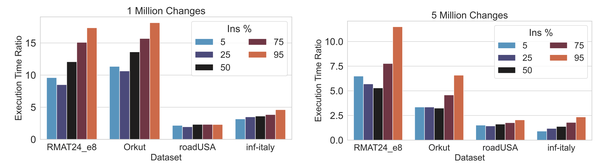


Figure: PageRank update in dynamic network

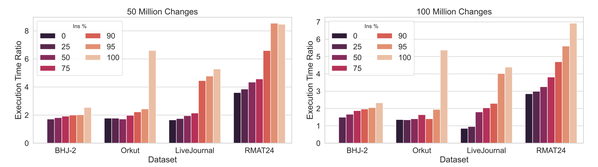
- High PageRank (PR) nodes have a cascading effect on a large number of remaining nodes.
- Adaptive Switch evaluates the total percentage of affected and high PR nodes.
- Adaptive Switch decides if updating property will be more efficient than property recomputation.
- Our update algorithm uses the previous property and iteratively computes the updated property in the affected subgraph.
- We empirically tested our adaptive implementation on real-world and synthetic networks and based on our observation we saw if the changes impact above 50% of nodes with high PR, the overall nodes affected are always above 95% of the entire network.
- When we compare our implementation with the recomputation approach if the changes impact less than 50 % of nodes with high PR values update algorithm outperforms the recomputation approach.

Results



(a) Speedup 1M (b) Speedup 5M

Figure: Kokkos color recomputation vs our Color update.



(a) Speedup 50M (b) Speedup 100M

Figure: Gunrock's SSSP recomputation vs our SSSP update.

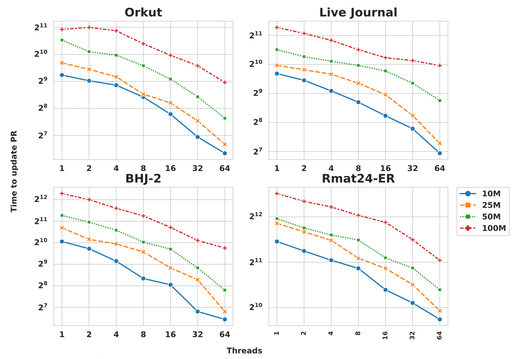


Figure: Shared Memory implementation of PR update

Conclusion

- Introduced CANDY: a high-performance network analysis software with performance optimization.
- Presented an adaptive template for creating new scalable, parallel algorithms for dynamic network property updates.

References

- 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.
- 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.
- 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

Acknowledgment

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