Towards Scalable Identification of Motifs Representing Non-Determinism in HPC Simulations

Ali Khan
alikhan@my.unt.edu
University of North Texas
Denton, Texas, USA

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

Michela Taufer
taufer@utk.edu
University of Tennessee, Knoxville
Knoxville, Tennessee, USA

ABSTRACT

We present a scalable algorithm for aligning large networks representing executions of HPC simulations through efficient computation of multiple execution motifs. Our algorithm is one of the first to identify the subtle differences between nearly similar execution networks, and thus can be used pinpoint the structural non-determinism across a set of event graphs arising from multiple runs of the same HPC simulations. These differences can help identify regions of non-determinism in large-scale asynchronous executions. The results are instrumental in enhancing reproducibility and reliability of exascale simulations.

CCS CONCEPTS

• **Mathematics of computing → Graph theory**; **Graph algorithms**; **Graph enumeration**; **Graph theory**; **Graph algorithms**; • **Computing methodologies → Parallel algorithms**; • **Theory of computation → Parallel algorithms**; **Parallel algorithms**.

KEYWORDS

graph comparison, graph alignment, parallel, shared memory

ACM Reference Format:
Ali Khan, Sanjukta Bhowmick, and Michela Taufer. 2022. Towards Scalable Identification of Motifs Representing Non-Determinism in HPC Simulations. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages.

1 INTRODUCTION

Network alignment (NA) is a fundamental problem in network analysis, with applications in varied disciplines including biology and social sciences. Given two graphs, network alignment matches vertices that have similar structure across the graph pairs.

Current NA algorithms cannot effectively identify differences in nearly similar graphs, such as a set of event graphs from the same high performance computing (HPC) simulation. Subtle differences in event graphs pinpoint the regions of non-determinism in repeated executions of the same simulation. Identifying these regions can help developers improve the reproducibility and reliability of large-scale, asynchronous application on exascale systems.

This observation motivated us to develop an *algorithm for large scale network alignment that is scalable, tunable, and sensitive to small changes*. We present an innovative algorithm that is based on comparing the graphlet degree vector (GDV) [2] of vertices. GDV measures the number of times a vertex belongs to particular graphlet (where a graphlet represents an execution motif).

Conference'17, July 2017, Washington, DC, USA
2022.

2 OVERVIEW OF OUR ALGORITHM

Similar to GRAAL [2], a NA tool for biological networks, our alignment matches vertices that have similar GDVs. As a by product, our algorithm can also be used for finding motifs in large graphs. Compared to GRAAL or other motif finding tools such as GraphPI [4] and CuTS [5], our method has these unique features;

Scalability: Our method can find multiple motifs in a single run while also maintaining scalability. Current scalable motif finding tools only find one motif per execution, and those that find multiple motifs in one execution are not scalable.

Tunability: Our method adapts to known topological properties, such as whether the graph is tree-like or planar, to speed the computation. Our algorithm can also satisfy user inputs regarding which nodes to include in motifs.

Sensitivity: Our method is designed to be sensitive to small differences in the input graphs. This is critical in identifying subtle structural changes across nearly similar networks.

2.1 GDV Computation

Given a set of motifs, we compute the automorphism orbits of each vertex (i.e., position in the motif). Because the number of motifs and orbits increases exponentially with size (a set of five or less vertices creates 30 unique motifs and 72 different orbits), computing each set of motif separately for each vertex is very computationally expensive. Using our algorithm, multiple motifs, belonging to multiple sets of vertices, are computed simultaneously.

Figure 1 gives an overview of the steps, including the creating initial roots ofsubtrees (Step 1), the subtree numeration (Step 2), the motif generation (Step 3), and the calculation of obits by using distance and degree vectors (Step 4). Starting from a root vertex, we grow trees to the maximum motif size (Steps 1 and 2). We then add back-edges to the trees to create the non-tree motifs (Step 3). We determine the orbit of each vertex by comparing the degree vector (the degree distribution of the motif) and the distance vector (the distance of the vertex from other vertices in the motif). These two vectors can uniquely identify most orbits in motifs with 5 or less vertices. The GDV is updated according to the motif (Step 4).

2.2 Scalable Implementation

Our algorithm is inherently scalable as each vertex can be processed in parallel. We address the challenges in reducing redundancy and resource usage by leveraging three features.

(i) *Redundant Computation*: If a motif has N vertices, the motif is created N times, generating redundancy. To eliminate this redundancy, we add two constraints. *First*, a tree is considered a motif only if the leaf nodes have a lower *id* than the root. *Second*, a back edge is added only if it connects the

two lowest *id* vertices in the cycle. It can be proven that by maintaining these two constraints each motif is processed (i.e., updated to GDV), exactly once.

(ii) *Memory Usage*: By creating trees from each vertex as root, the storage costs can increase exponentially. To reduce the memory usage, we store the trees as Prufer sequences[3]. We also use a limited storage queue, where if the queue is filled, new tree generation stops until the queue empty.

(iii) *Load Balancing*: Not all root vertices perform equal amount of computation, leading to computational load imbalance. We apply dynamic scheduling, and vertex splitting, to process high degree vertices to reduce the imbalance.

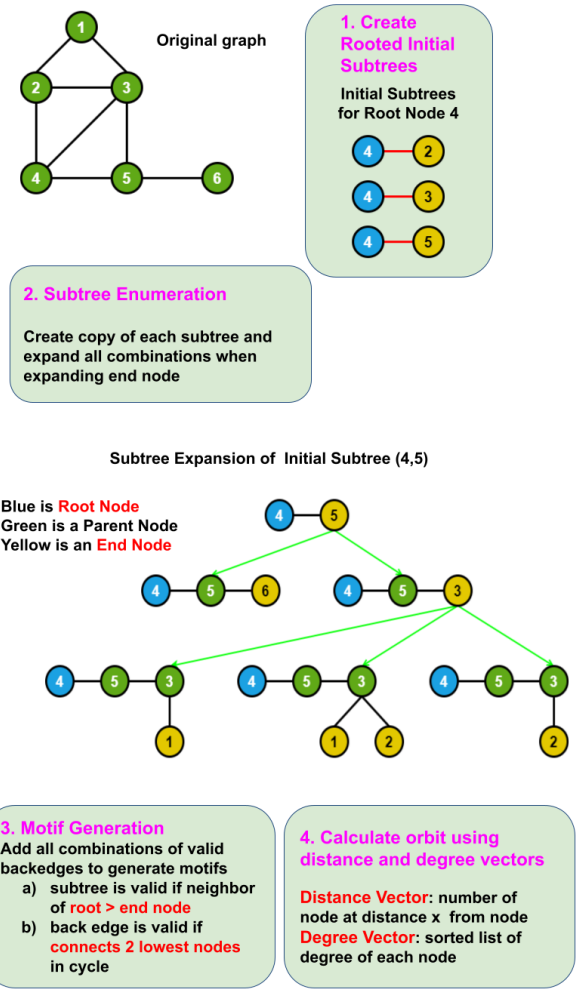

Figure 1: Workflow of GDV Computation

3 RESULTS AND CONCLUSION

The algorithm is evaluated on two event graphs generated by the ANACIN-X tool[1] from the AMG2013, an algebraic multigrid application, and the Message Race, N-1 processes sending a message to a single node. The AMG2013 event graph of 1392 nodes and 2056 edges took 1.51 seconds to execute, and the MessageRace event graph of 132 nodes and 166 edges took 11 ms to execute. The node level results for each graph are shown in Figure 2.

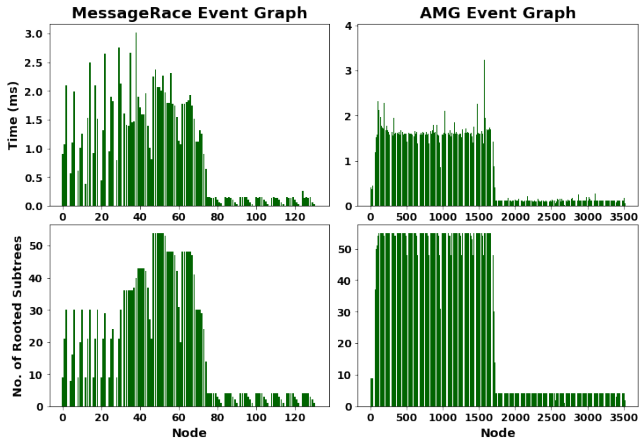

Figure 2: Comparing subtree generation for AMG2013 and Message Race. Top: time elapsed per vertex. Bottom: number of subtrees per vertex.

Our algorithm advances motif-based network alignment through its scalability, tunability and sensitivity to changes. In the poster, we will present results to highlight (i) its strong scalability on shared memory; (ii) time for finding motifs compared to GraphPI; and (iii) its ability leverage graph properties and user specifications.

ACKNOWLEDGMENTS

We are grateful for the assistance from Krishna Sai Ujwal Kambhumpati, Nigel Tan, and Patrick Bell. This work is supported by NSF grants #1900888 and #1900765. Support of XSEDE and IBM through a Shared University Research Award is also acknowledged.

REFERENCES

[1] Patrick Bell, Kae Suarez, Dylan Chapp, Nigel Tan, Sanjukta Bhowmick, and Michela Taufer. 2021. ANACIN-X: A software framework for studying non-determinism in MPI applications. *Software Impacts* 10 (2021), 100151. https://doi.org/10.1016/j.simpa.2021.100151

[2] Oleksii Kuchaiev, Tijana Milenković, Vesna Memišević, Wayne Hayes, and Nataša Pržulj. 2010. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface* 7, 50 (2010), 1341–1354.

[3] Heinz Prüfer. 1918. Neuer beweis eines satzes über permutationen. *Arch. Math. Phys* 27, 1918 (1918), 742–744.

[4] Tianhui Shi, Mingshu Zhai, Yi Xu, and Jidong Zhai. 2020. Graphpi: High performance graph pattern matching through effective redundancy elimination. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–14.

[5] Lizhi Xiang, Arif Khan, Edoardo Serra, Mahantesh Halappanavar, and Aravind Sukumaran-Rajam. 2021. CuTS: Scaling Subgraph Isomorphism on Distributed Multi-GPU Systems Using Trie Based Data Structure. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 69, 14 pages. https://doi.org/10.1145/3458817.3476214