Science and Technology Highlights from Published Papers or Reports

Title: Distributed Louvain Algorithm for Graph Community Detection

Achievement: We presented a distributed memory implementation of the Louvain algorithm for parallel graph community detection. We introduced several heuristics, the inclusion of which was found crucial for improving performance and scalability. Our parallel implementation demonstrated speedups of 1.8x-23x (using up to 4K processes) relative to the Baseline version, for a wide variety of real-world networks.

Significance and Impact: —In most real-world networks, the nodes/vertices tend to be organized into tightly-knit modules known as communities or clusters, such that nodes within a community are more likely to be "related" to one another than they are to the rest of the network. Identifying such clusters efficiently reveal the non-trivial structures of the networks, and there are various use cases in the domain of social science, biology, and machine learning.

Research Details:

- Parallel solution for community detection that scale.
- Different heuristic for community detection to improve the run-time and scalability without compromise the quality.
- Complete study of tradeoff between each heuristic.

Citation and DOI: Sayah Ghosh, Mahantesh Halappananvar, Antonino Tumeo, Ananth Kalyanaraman, **Hao Lu**, Daniel Chavarria-Miranda, Ariful Khan, Assefaw H. Gebremedhin. "Distributed Louvain Algorithm for Graph Community Detection." *Parallel and Distributed Processing Symposium (IPDPS), 2018 IEEE International*. (Accepted)

Summary: In this paper, we present the design of a distributed memory implementation of the Louvain algorithm for parallel community detection. Our approach begins with an arbitrarily partitioned distributed graph input, and employs several heuristics to speed up the computation of the different steps of the Louvain algorithm. We evaluate our implementation and its different variants using real-world networks from various application domains (including internet, biology, social networks). Our MPI+OpenMP implementation yields about 7x speedup (on 4K processes) for soc-friendster network (1.8B edges) over a state-of-the-art shared memory multicore implementation (on 64 threads), without compromising output quality. Furthermore, our distributed implementation was able to process a larger graph (uk-2007; 3.3B edges) in 32 seconds on 1K cores (64 nodes) of NERSC Cori, when the state-of-the-art shared memory implementation failed to run due to insufficient memory on a single Cori node containing 128 GB of memory.