Understanding Performance Bottleneck to Improve Parallel Efficiency of Louvain Algorithm

Naw Safrin Sattar and Shaikh Arifuzzaman
Department of Computer Science, University of New Orleans, New Orleans, LA-70148, USA.
Email: (nsattar, smarifuz)@uno.edu

***Abstract*—Detecting communities (clusters) in massive networks is a fundamental problem in network science. The Louvain algorithm is one of the fastest modularity-based algorithms, and works well with large graphs. It also reveals a hierarchy of communities at different scales, which can be useful for understanding the global functioning of a network. In current literature, there exists several shared memory parallel implementation with better scalability on large-scale graphs. Our focus is to analyze the performance bottlenecks in distributed environment. We also look for the scope of improvements in a hybrid (both shared and distributed memory) approach. Using profiling tools, we can identify the factors behind memory consumption, communication overheads. This understanding helps us to design better algorithms in distributed and hybrid environments. We are also working towards a gpu-based parallel implementation of Louvain algorithm utilizing the power of GPUs to parallelize massive data.**

***Keywords*-Louvain algorithm; parallel methods; large-scale dataset; community detection; GPU**

Motivation, Methods and Contribution

Community structures help us solve many real-world problems such as rumor propagation, epidemic spreading, recommender system in e-commerce, terrorist activities in social networks and many more. Louvain algorithm [1] is one of the efficient and well-known algorithms to detect communities considering the quality of its output communities and the computational time. Parallelizing Louvain algorithm in distributed memory environment is difficult considering the challenges of communication overhead among processors and a well-balanced partitioning technique of initial data. In our work we aim at pointing out the bottlenecks to improve the parallel efficiency of Louvain algorithm in distributed environment. A hybrid algorithm provides the flexibility to the users to balance between both shared and distributed memory system according to available resources. We analyze our distributed-memory Louvain algorithm [2] with MPI profiling tool TAU to understand the communication pattern among processors, to identify possibilities towards improvement. We observe that for MPI_Send and MPI_Recv functions in overall communication, 65% and 69% of the processors respectively, take less than average time to complete their tasks when the load is balanced (METIS graph-partitioner) shown in Figure 1. Again, only few processors have higher number of communications and larger message size compared to the larger threshold in the load-imbalanced environment.

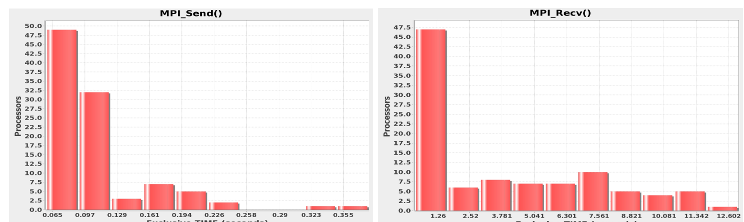

Figure 1: Runtime of MPI functions

We will look further into the memory consumption i.e. branching and cache access patterns, time stalled waiting for resources (such as in memory reads), etc. as well as communication time at different phases of the algorithm to identify performance bottlenecks. It will help us to identify whether communication time overweighs computation time and change the design of our algorithm accordingly. We will also experiment with other graph-partitioning techniques (i.e. hypergraph partitioning for social networks) for improved load-balancing and higher parallel efficiency. We are also working on designing a parallel implementation of Louvain algorithm with data parallel computations in GPUs.

References

[1] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[2] N. S. Sattar and S. Arifuzzaman, "Overcoming mpi communication overhead for distributed community detection," in *Workshop on Software Challenges to Exascale Computing*. Springer, 2018, pp. 77–90.