PARTITIONING FOR GRAPH ANALYTICS

Partitioning large-scale real-world graphs for parallel analysis is challenging. We ideally want a partitioning scheme that satisfies the following:

- Balanced number of vertices and edges per partition to evenly distribute computation among tasks.
- Small maximal per-part edge cut to balance communication requirements among tasks.
- Small total edge cut to minimize total communication requirements.
- Fast partitioning with minimal memory and computational overhead.

**PULP, PULP-M, PULP-MM**

To satisfy the above objectives, we employ an efficient iterative label propagation-based [1] partitioning scheme to exploit the community structures inherent in most small-world graphs. More algorithmic detail is available in [4].

1. Initialize p random partitions.
2. Execute degree-weighted label propagation.
3. for $k_1$ iterations do
4. for $k_2$ iterations do
5. Balance partitions to satisfy constraint 1.
6. Refine partitions to minimize objective 1.
7. for $k_3$ iterations do
9. and minimize objective 2.
10. Refine partitions to minimize objective 1.

We call the above algorithm PULP-MM. By running parts of the above we also have PULP (minimize edge cut and balance vertices) and PULP-M (minimize edge cut and balance vertices and edges).

**RESULTS: SUPERIOR SCALABILITY AND PERFORMANCE COMPARED TO MULTILEVEL PARTITIONERS**

**Visualization of PULP:** Partitioning Using Label Propagation

![Visualization of PULP](image)

1.) Randomly initialize the partition. Solid lines indicate cut edges and dotted lines indicate intra-part edges.

2.) Perform label propagation to create clusters and minimize edge cut.

3.) Balance the partition for vertices while additionally refining to minimize edge cut.

4.) Further balance the partition for edges and minimize maximal per-part edge cut.

**Execution Time:** PULP gives up to an order of magnitude speedup in both serial and parallel partitioning relative to KaFFPa [3] and METIS/ParMETIS [2].

**Memory Savings:** PULP’s direct single-level approach avoids the considerable memory overheads of multi-level schemes.

**CONCLUSIONS**

Utilizing an iterative label propagation-based approach to small-world graph partitioning can offer orders of magnitude improvement in both running time and memory utilization, while producing better cut quality under the complex objectives modern graph analytics requires.

**REFERENCES**


**ACKNOWLEDGEMENTS**

Thanks to Siva Rajamanickam for assistance and guidance. This work is supported by NSF grant ACI-1253881 and the DOE Office of Science through the PAR surprise and 2nd place honorable mention, Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-AC04-94AL85000.