Combinatorial Relations and Transformations between "Graph Partitioning by Vertex Separator" and "Hypergraph Partitioning" Problems

Over the last four decades, graph partitioning with vertex separator (GPVS) and hypergraph partitioning (HP) problems have been two most commonly used combinatorial models in sparse matrix and VLSI communities, respectively. More recently, the modeling flexibility provided by hypergraphs has drawn a lot of interest from the combinatorial scientific community, leading to novel models and algorithms. The popularity of these two problems has been accompanied with the development of effective GPVS and HP tools, all of which adopt the multilevel framework successfully. This talk discusses the combinatorial relations between these two problems, presents how each problem can be formulated in terms of the other problem, and comments on the trade-offs of these two cross-formulations.

Algorithms on hypergraphs are inherently more complicated than those on graphs both in terms of computational complexity and runtime performance, because operations on nets are performed on sets of vertices as opposed to pairs of vertices. How can we solve problems that are most accurately modeled with hypergraphs using graph algorithms for attaining their better runtime performance without sacrificing too much from what is really important for the application? This question has been asked before in the VLSI community and the associated work investigated the relationship between HP and graph partitioning by edge separator (GPES) but unfortunately it achieved little success. In [1] we show how to solve the HP problem by finding vertex separators on the net intersection graph (NIG) of the hypergraph. In the NIG representation of a hypergraph, each net is represented by a vertex, and each vertex of the hypergraph is replaced with a clique of the nets connecting that vertex. A vertex separator on this graph defines a net separator for the hypergraph. Once vertices of the hypergraphs are replaced with cliques, it will be impossible to preserve the vertex weight information accurately. Therefore, we can view the NIG model as a way to trade computational efficiency for exact modeling power. Nevertheless, we propose a weighting scheme in NIG, which is quite effective in attaining fairly vertex-balanced partitions of the hypergraph. We also propose an implementation for our GPVS-based HP formulation by adopting and modifying a state-of-the-art GPVS tool.

GPVS-based HP formulation can be used for faster partitioning in scientific computing applications where balance cannot be defined precisely; or there are additional constraints that cannot be easily incorporated into partitioning algorithms. For instance, hypergraph models can be used to permute a linear programming (LP) constraint matrix to a block angular form for parallel solution with decomposition methods. Load balance can be achieved by balancing subproblems during partitioning. However, it is not possible to accurately predict solution time of an LP, and equal-sized subproblems only increase the likelihood of computational balance.

Both GPES and HP problems are well suited for the successful multilevel framework, because the following nice property holds for the edge and net separators in multilevel GPES and HP. Any edge/net separator at a given level of uncoarsening forms a valid narrow edge/net separator of all the finer graphs/hypergraphs, including the original graph/hypergraph. However, this property does not hold for the GPVS problem. In [2] we show a novel formulation of the GPVS problem as an HP problem that is immune to this deficiency of GPVS, and thus producing better quality solutions. The formulation relies on finding an edge clique cover of the graph. The edge clique cover is used to construct a hypergraph, which is referred to here as the clique-node hypergraph. In this hypergraph, the vertices correspond to the cliques of the edge clique cover, and the nets correspond to the vertices of the graph. In matrix terms, this formulation corresponds to finding a structural factorization of a given matrix M in the form of $M = AA^T$. In applications like the solution of LP problems using an interior point method, such a matrix A is actually given as a part of the problem. For other problems, we present efficient methods to find such a structural factorization.

HP-based GPVS formulation can be effectively used in fill-reducing ordering of sparse symmetric matrices. In [2] we developed a new incomplete nested disection based fill reducing ordering that utilizes our HP-based GPVS formulation. The combinatorial transformations between these two well-known problems can also be utilized in practical cases where existing partitioning tools for the natural representation lack features available in the tools for the transformed problem. For example, our recent work [3] on permuting sparse square matrices into block diagonal form with overlap adopts a GPVS formulation with fixed vertices. As current GPVS tools do not support fixed vertices and there exists HP tools supporting fixed vertices, we resort to the proposed HP-based GPVS formulations to solve this problem. As an another example, GPVS formulation proposed and used for parallel frequent itemset mining with selective item replication [4] necessitates the use of two vertex weights to encode the partitioning constraint of computational load balancing and the partitioning objective of minimizing the amount of data replication. Since the existing GPVS tools do not support this feature, the HP-based GPVS formulation is successfully used in this application since in HP vertices and nets can easily be associated with different weight functions.

References

[1] Enver Kayaaslan, Ali Pinar, Umit V. Catalyurek and Cevdet Aykanat, "Partitioning Hypergraphs through Vertex Seperators on Graphs," SIAM Journal on Scientific Computing, vol. 34(2), pp. A970–A992, 2012.

[2] Umit V. Catalyurek, Cevdet Aykanat and Enver Kayaaslan, "Hypergraph-Partitioning-Based Fill-Reducing Ordering for Symmetric Matrices," SIAM Journal on Scientific Computing (SISC), vol. 33(4), pp. 1996–2023, 2011.

[3] Seher Acer, Enver Kayaaslan and Cevdet Aykanat, "A Recursive Bipartitioning Algorithm for Permuting Sparse Square Matrices into Block Diagonal Form with Overlap," SIAM Journal on Scientific Computing, vol. 35(1), pp. C99–C121, 2013.

[4] Eray Özkural, Bora Ucar, and Cevdet Aykanat, "Parallel Frequent Item Set Mining with Selective Item Replication," IEEE Transactions on Parallel and Distributed Systems, vol. 22(10), pp. 1632–1640, 2011.