A Parallel Graph Laplacian Solver

Tristan Konolige Jed Brown

Abstract

Problems from graph drawing, spectral clustering, network flow and graph partitioning all can be expressed as Laplacian matrices. Theoretically fast approaches to solving these problems exist, but in practice these techniques are slow. Two practical approaches have been proposed and work well in serial. However, as problem sizes increase and single core speeds stagnate, parallelism is essential to solve problems quickly. We present a modified version of Livne and Brandt's Lean Algebraic Multigrid algorithm with good parallel performance. We are esspecially interested in performance on scale-free graphs.

Theoretical Solvers

A variety of theoretical Laplacian solvers have been proposed in literature starting with Spielman and Teng's 2003 paper [1]. To our knowledge, no working implementation of this algorithm exists. Kelner et al. later proposed a simple and novel technique with a complexity bound of $O(m \log n^2 \log \log n \log (\epsilon^{-1}))$ [2]. In practice this algorithm appears slower than the preconditioned conjugate gradient method [3].

Practical Serial Solvers

Two practical Laplacian solvers have been proposed: Koutis and Miller's Combinatorial Multigrid (CMG) [4] and Livne and Brandt's Lean Algebraic Multigrid (LAMG) [5]. Both use multigrid techniques to solve the laplacian problem. CMG, like much of the theoretical literature takes a graph theoretic approach. It constructs a multilevel preconditioner using a modified spanning tree [4]. LAMG uses a more standard AMG approach with modifications suited for Laplacian matrices. Notably it employs a specialized distance function, a clustering algorithm suited to scale free graphs, and a Krylov method to accelerate solutions on each of the multigrid levels. These changes are not rooted in theory but produce good empirical results. LAMG is slightly slower than CMG but more robust [5]. Both CMG and LAMG work serially, they do not lend themselves to a clear parallel implementation. In CMG, the spanning tree splitting and clustering steps are not necessarily easy to do in parallel. Furthermore, our implementation of CMG did not achieve the performance results of Koutis and Miller. This leads us to believe there are some key parts of the solver that are not discussed in the paper. LAMG's partial elimination procedure and clustering process both are inherently serial.

Our Parallel Solver

Our solver is a modification of LAMG suited to parallel execution. Notable changes are 1. a different strength of connection metric, 2. a parallel partial elimination algorithm, and 3. a new parallel clustering algorithm. We use a block-wise distribution of matrix entries because our experiments with row-wise distributions failed scale well on scale-free graphs. We use the CombBLAS library as it provides a 2D matrix entry distribution [6]. Although we achieve better scaling, the 2D entry distribution has a more complicated communication pattern and higher constant factors.

The strength of connection metric indirectly determines how likely any two nodes will be clustered together. Our choice of metric is motivated by empirical tests. We ran LAMG on a large set of problems from the University of Florida Sparse Matrix Collection [7] with affinity strength of connection (proposed in the LAMG paper) and algebraic distance (proposed in [8]). In our tests, algebraic distance preformed better than affinity a majority of the time. Changing the metric used for strength of connection has no effect on parallel performance.

Both affinity and algebraic distance are easily parallelizable.

The algorithm proposed by Livne and Brandt for partial elimination in sequential. It follows a three step process: 1. choose all low degree nodes as elimination candidates. 2. Remove nodes from the candidate set if they neighbor another candidate. 3. Eliminate remaining candidates. Step 1 and 3 are simple to make parallel. Step 2, proposed by Livne and Brandt is a sequential process in which candidates nodes are considered for removal in order. To make step 2 parallel, we apply a hashing scheme to locally make decisions on which candidate in a pair of neighbors we should eliminate. With this step, the elimination phase still requires some communication to communicate candidates to neighbors.

Our modification of Livne and Brandt's LAMG clustering algorithm is a work in progress. We hope to adapt one of the many parallel multigrid clustering algorithms in the literature. We will maintain the goal of Livne and Brandt's clustering algorithm to avoid clustering high degree nodes together.

Expected Results

As our solver is entirely practical, we hope to produce good empirical results. We plan to test our implementation against a large selection of graphs from the University of Florida Sparse Matrix collection. We will compare our results against Jacobi preconditioned conjugate gradient, pcg after low-degree elimination, and the SuperLU direct solver [9]. We will also provide a weak and strong scaling analysis.

References

[1] Spielman, D A and Teng, S (2003). Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. *CoRR*. **cs.DS/0310051**. http://arxiv.org/abs/cs.DS/0310051

[2] Kelner, J A, Orecchia, L, Sidford, A and Zhu, Z A (2013). A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. *CoRR*. **abs/1301.6628**. http://arxiv.org/abs/1301.6628

[3] Boman, E G, Deweese, K and Gilbert, J R (2015). Evaluating the potential of a laplacian linear solver. *CoRR*. **abs/1505.00875**. http://arxiv.org/abs/1505.00875

[4] Koutis, I, Miller, G L and Tolliver, D (2011). Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding*. **115** 1638–46. http://www.sciencedirect.com/science/article/pii/S1077314211001627

[5] Livne, O E and Brandt, A (2011). Lean Algebraic Multigrid (LAMG): Fast Graph Laplacian Linear Solver. *arXiv.org*. http://arxiv.org/abs/1108.0123v1

[6] Buluç, A and Gilbert, J R (2011). The Combinatorial BLAS: Design, implementation, and applications. *The International Journal of High Performance Computing Applications*. **25** 496–509

[7] Davis, T A and Hu, Y (2011). The university of florida sparse matrix collection. *ACM Trans. Math. Softw.* ACM, New York, NY, USA. **38** 1:1–1:25. http://doi.acm.org/10.1145/2049662.2049663

[8] Ron, D, Safro, I and Brandt, A (2011). Relaxation-based coarsening and multiscale graph organization. *Multiscale Modeling & Simulation*. Society for Industrial & Applied Mathematics (SIAM). **9** 407–23. http://dx.doi.org/10.1137/100791142

[9] Li, X S (2005). An overview of SuperLU: Algorithms, implementation, and user interface. *toms*. **31** 302–25