A Multilevel Algorithm for Spectral Partitioning with Extended Eigen-Models

Suely Oliveira[1] and Takako Soma

[1] The Department of Computer Science,
The University of Iowa, Iowa City, IA 52242, USA,
oliveira@cs.uiowa.edu,
WWW home page: http://www.cs.uiowa.edu/~oliveira

Abstract. Parallel solution of irregular problems require solving the graph partitioning problem. The extended eigenproblem appears as the solution of some relaxed formulations of the graph partitioning problem. In this paper, a new subspace algorithm for the solving the extended eigenproblem is presented. The structure of this subspace method allows the incorporation of multigrid preconditioners. We numerically compare our new algorithm with a previous algorithm based on Lanczos iteration and show that our subspace algorithm performs better.

1 Introduction

One of the main problems encountered when dealing with irregular problems on parallel architectures is mapping the data into the various processors. Traditionally graph partitioning has been used to achieve this goal.

Kernighan and Lin developed an effective combinatorial method based on swapping vertices [10]. Multilevel extensions of the Kernighan-Lin algorithm have proven effective for graphs with large numbers of vertices [9]. Like combinatorial methods, spectral methods have proven effective for large graphs arising from FEM discretizations [12]. In fact, currently various software packages (METIS [8] and Chaco [4] and others) combine multilevel combinatorial algorithms and spectral algorithms. The spectral algorithms which are used in these packages are based on the models that partition a graph by finding the second smallest eigenvector of its graph Laplacian using an iterative method. This is in fact the model that we used in [7] and [6].

However, as the example below shows, there are number of reasons for modifying the traditional model and spectral heuristics for graph partitioning. Recently, Hendrickson et al. [5] pointed out the problems with traditional models. Consider Figure 1. Assume we have already partitioned the graph into two pieces, left and right halves, and that we have similarly divided the left half graph into top and bottom quadrants. When partitioning the right half graph between processors 3 and 4 we should like messages to travel short distances. The mapping shown in the left-hand figure is better since the total message distance is less than that for the right-hand figure. Note that even though in modern computers

the distance between processors may not imply very different timings, smaller distances traveled by the messages will decrease congestion problems. The basic idea is to associate with each vertex in the partitioned subgraph a value which reflects its net desire or preference to be in the bottom quadrant instead of the top quadrant. Note that this preference is a function only of edges that connect the vertex to vertices which are not in the current subgraph. The preferences need to be considered when subgraphs are partitioned. These preferences should be propagated through the recursive partitioning process. In order to to overcome this problem, Hendrickson et al. used an extended eigen-model in [5].

Another way in which extended eigen-models occur is when some nodes are pre-assigned to some partitions, and we wish to assign the remaining nodes to the remaining partitions. In this situation the extended eigen-model can also be used to assigning the remaining nodes. This can be a much smaller problem, and would be very useful for dynamic re-partitioning.

In other words, extended eigen-models can be used to develop more refined models of the true costs of a partition, and to develop algorithms which give more useful and effective partitionings.

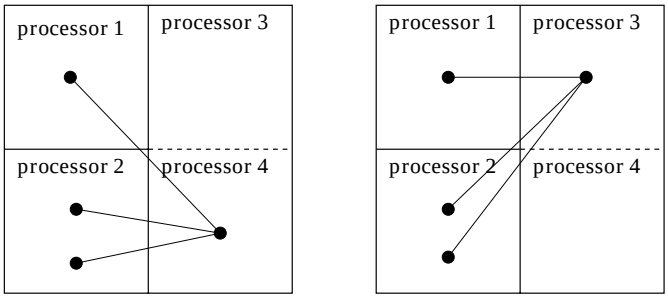

Fig. 1. Partition between the 4 processors.

2 Extended Eigen-Model

Here we describe the motivation given in [5], but notice that there are other applications in which similar problems to extended eigenproblems need to be solved [1]. Let $G = (V, E)$ be a graph with vertices $v \in V$ and edges $e_{ij} \in E$. We also allow either edges or vertices to have positive weights associated with them, which we denote by w_v and $w_{e_{ij}}$ respectively. Assume we want to divide V into two subsets V_1 and V_2, and that we have a vector d of preferences for the vertices of V to be in V_1. The cost associated with a partition now has two components. First, every edge $e_{ij} \in E$ crossing between V_1 and V_2 contributes a value of $w_e(e_{ij})$. Second, for each vertex in V_1 with a negative preference we add the magnitude of the preference to the cost, and similarly for each vertex in

V_2 with a positive preference. Our goal is to find a partition of the vertices into two sets of nearly equal size in which this combined cost function is minimized.

A graph partition can be described by assigning a value of $+1$ to all vertices in one set and a value of -1 to all vertices in the other. If we denote the value assigned to a vertex i by $x(i)$, then the simple function $(x(i) - x(j))^2/4$ is equal to 1 if the vertices i and j are in different partitions and equal to 0 otherwise. If $d(i)$ is the preference for a vertex to be in the set denoted by $+1$ which we will define to be V_1, then the new problem we want to solve is

$$\begin{array}{l} \text{Minimize } f(x) = \frac{1}{4} \sum\limits_{e_{ij} \in E} w_e(e_{ij})(x(i) - x(j))^2 - \frac{1}{2} \sum\limits_{i \in V} d(i)x(i) \\ \text{Subject to} \\ \text{(a) } \sum\limits_{i \in V} w_v(i)x(i) \approx 0 \\ \text{(b) } x(i) = \pm 1. \end{array} \tag{1}$$

After some transformations and relaxing the integrality constraint that $x(i) = \pm 1$, this model can be expressed as

$$\begin{array}{l} \text{Minimize } f(x) = y^T A y - 2h^T y \\ \text{Subject to} \\ \text{(a) } s^T y = 0 \\ \text{(b) } y^T y = \omega_v \end{array} \tag{2}$$

where A is the graph Laplacian matrix for the edge weights w_e, $\omega_v = |V|$ the number of vertices, $s_i = w_v(i)$, and $h_i = d(i)$ for all i.

We introduce Lagrange multipliers λ and μ, and look for stationary points of the Lagrangian function

$$F(y, \lambda, \mu) = y^T A y - 2h^T y + \lambda(s^T y) + \mu(\omega_v - y^T y) = 0. \tag{3}$$

satisfying (a) and (b).
Taking derivatives with respect to the components of y, we obtain

$$\nabla_y F(y, \lambda, \mu) = 2Ay - 2h + \lambda s - 2\mu y = 0. \tag{4}$$

We can calculate λ by left multiplying Equation 4 by s^T. Since s is orthogonal to y and s is a zero eigenvector of A, we have $\lambda = 2s^T h/\omega_v$. We now define

$$g = h - \frac{s^T h}{\omega_v} s, \tag{5}$$

which allows us to rewrite Equation 4 as

$$Ay = \mu y + g. \tag{6}$$

This extended eigenproblem must be solved subject to the constraints in (2). In fact, we need to use the smallest μ for which a solution of (2, 6) can be found. A proof of this fact can be found using [11, Thm. 4.3, p. 78], for example.

In [5], Hendrickson used a numerical method for solving this extended eigenproblem based on the Lanczos algorithm, which will be described in section 4. Next section we present our theoretical motivation for studying subspace algorithms for the extended eigenproblem. In section 5 we compare our subspace algorithms with the method of [5].

3 Subspace Algorithms for Solving Extended Eigenproblems

Generalized Davidson algorithms are subspace methods for large symmetric eigenvalue problem. They solve the eigenvalue problem $Ay = \mu y$ by constructing an orthonormal basis $V_k = \{v_1 \ldots, v_k\}$ at each k^{th} iteration step and then finding an approximation for the eigenvector y of A by using a vector y_k from the subspace spanned by V_k. Previously, we have developed a Davidson-type algorithm for solving the $Ay = \lambda y$ model where A is the graph Laplacian matrix [6, 7]. Now, we present Davidson-type algorithms to solve the extended eigenproblems.

Our problem is to find the smallest μ and the corresponding y such that $Ay = \mu y + g$. The corresponding optimization problem being

$$\begin{array}{l} \text{Minimize } f(y) = y^T Ay - 2h^T y \\ \text{Subject to} \\ \text{(a) } s^T y = 0 \\ \text{(b) } y^T y = \omega_v. \end{array} \tag{7}$$

The solution of this problem is given by the solution of Equation 4. Notice that if we represent y in a subspace V with $y = Vz$, we can rewrite (7) as

$$\begin{array}{l} \text{Minimize } f(z) = z^T V^T AVz - 2h^T Vz \\ \text{Subject to} \\ \text{(a) } s^T Vz = 0 \\ \text{(b) } z^T V^T Vz = \omega_v. \end{array} \tag{8}$$

Again, the solution of this problem can be obtained from the Lagrangian

$$\tilde{F}(z, \lambda, \mu) = z^T V^T AVz - 2h^T Vz + \lambda s^T Vz - \mu(\omega_v - z^T V^T Vz) = 0. \tag{9}$$

This leads to

$$\begin{aligned} \nabla_z \tilde{F}(z, \lambda, \mu) &= V^T 2AVz - V^T 2h + \lambda V^T s - 2\mu V^T Vz \\ &= V^T [2AVz - 2h + \lambda s - 2\mu Vz] \\ &= V^T [2Ay - 2h + \lambda s - 2\mu y] \\ &= V^T \nabla_y F(y, \lambda, \mu). \end{aligned} \tag{10}$$

Consequently, The subspace problem (8) can be used to estimate the solution of the original problem (7). That gives us an opportunity to develop Davidson-type algorithms for the extended eigenproblem.

4 Algorithm

The problem is to find the smallest μ and the corresponding y such that

$$\begin{array}{l} Ay = \mu y + g \\ \text{Subject to} \\ \text{(a) } s^T y = 0 \\ \text{(b) } y^T y = \omega_v. \end{array} \tag{11}$$

The problem of finding the smallest μ and the corresponding y is not simply a matter of a linear system eigensolver because the norm constraint on y must also be satisfied. An iterative approach can be used. That is we guess a value of μ, solve for y and check whether $y^T y = \omega_v$ and adjust our guess for μ accordingly.

The existing approach of [5] is described here. We multiply $Ay = \mu y + g$ by $Q_j^T \in \mathbb{R}^{j \times n}$ (the transpose of the Lanczos basis at step j) and look for a solution of form $y = Q_j z$, obtaining

$$Q_j^T A Q_j z = \mu Q_j^T Q_j z + Q_j^T g. \tag{12}$$

From the standard Lanczos process we have after j steps that $Q_j^T A Q_j = T_j + Q_j^T r_j e_j^T$ where T_j is a tridiagonal matrix, Q_j is orthogonal, r_j is the residual vector and $e_j = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ where 1 is at the j^{th} position. Hence equation (12) becomes $(T_j + Q_j^T r_j e_j^T) z = \mu z + Q_j^T g$. Since in exact arithmetic $||Q_j^T r_j e_j^T||$ is zero, we can solve

$$(T_j - \mu I) z = Q_j^T g. \tag{13}$$

Substituting $\tilde{g} = Q_j^T g$ we obtain

$$(T_j - \mu I) z = \tilde{g}. \tag{14}$$

where we seek the pair (μ, v) corresponding to the left-most value of μ such that the applicable constraints are satisfied.

Now, the constraint equations are considered. Using $y = Q_j z$ the norm constraint on z becomes $\omega_v = y^T y = z^T Q_j^T Q_j z = z^T z = \omega_v$. Q_j can be constructed in such a way that all of its columns are orthogonal to s. The problem is then to find the smallest μ and the corresponding z

$$\begin{array}{l} (T_j - \mu I) z = \tilde{g} \\ \text{Subject to} \\ \text{(a) } ||z||_2^2 = \omega_v, \end{array} \tag{15}$$

which can be rewritten as

$$||(T_j - \mu I)^{-1} \tilde{g}||_2^2 = \omega_v. \tag{16}$$

since T_j is tridiagonal, the cost of calculating $(T_j - \mu I)^{-1} \tilde{g}$ is $O(k)$. In [5], equation (16) was solved using a bisection method.

However, there are a number of numerical difficulties with the Lanczos method, which are due to roundoff error [3, §9.2.2]. One of the main difficulties is that the Q_j matrices may become highly non-orthogonal for large j. Usually, some sort of re-orthogonalization scheme is needed [3, §§9.2.3–9.2.4], or else a method of removing "ghost" eigenvalues [3, §9.2.5]. Either approach results in code that is more reliable but less efficient.

We need to choose a method for solving the problem on a subspace. Instead of Lanczos we use an eigendecomposition approach which is described next. We consider the eigen decomposition of $A = Q\Lambda Q^T$, where Q is orthogonal. Substituting this into Equation (12), we have

$$(A - \mu I)^{-1} g = Q(\Lambda - \mu I)^{-1} Q^T g, \tag{17}$$

$$||(A - \mu I)^{-1} g||_2^2 = ||Q(\Lambda - \mu I)^{-1} Q^T g||_2^2 = ||(\Lambda - \mu I)^{-1} \hat{g}||_2^2, \tag{18}$$

where $\hat{g} = Q^T g$. The inversion of $(\Lambda - \mu I)$ has cost $O(k)$, where k is the dimension of the system. For each A only one eigendecomposition is needed, which costs $O(k^3)$. This makes it a competitive approach that is also very reliable. Since our minimization problem is

$$\begin{array}{l} z = (\Lambda - \mu I)^{-1} \hat{g} \\ \text{Subject to } ||z||_2^2 = \omega_v, \end{array} \tag{19}$$

we can rewrite it as

$$||(\Lambda - \mu I)^{-1} \hat{g}||_2^2 = \omega_v. \tag{20}$$

Equation (20) can be solved using a bisection method, but Brent's method gives better convergence rates [2].

Below we present the main steps of the Davidson algorithm for the extended eigenproblem.

1. Define V_1 as $V_1 = [v_1]$.
2. Find the extended eigenproblem solution (μ, y) on the subspace. The projected matrix on the subspace is $S_j = V_j^T A V_j$ where V_j is the current orthogonal basis.
3. Use Ritz values ($u_j = V_j y_j$) and vectors to estimate residual: $r_j = Au_j - \mu_j u_j - g = W_j y_j - \mu_j V_j y_j - g$, where $W_j = AV_j$.
4. Solve $t_j = M_j r_j$ approximately. (This corresponds to preconditioning the residual.)
5. Orthonormalize t_j against the current orthogonal basis V_j. Append the orthonormalized vector to V_j to give V_{j+1}.

5 Numerical Results

We compared our new subspace algorithm with the Lanczos based algorithm previously used for the extended eigenproblem. Our preconditioner was a multigrid

algorithm and research is under development exploring the use of other multilevel preconditioners for this algorithm. The two algorithms have been implemented and run on a HP VISUALIZE Model C240 workstation, with a 236MHz PA-8200 processor and 512MB RAM. Both algorithms were run with square matrices of various sizes. Figure 2 compares the observed running timings. From this graph we can see that the subspace algorithm takes less than the Lanczos based algorithm for all the test cases, specially as the problem sizes get bigger. In addition, there are other problems inherent to the Lanczos algorithm that will be avoided with our new preconditioned subspace algorithm.

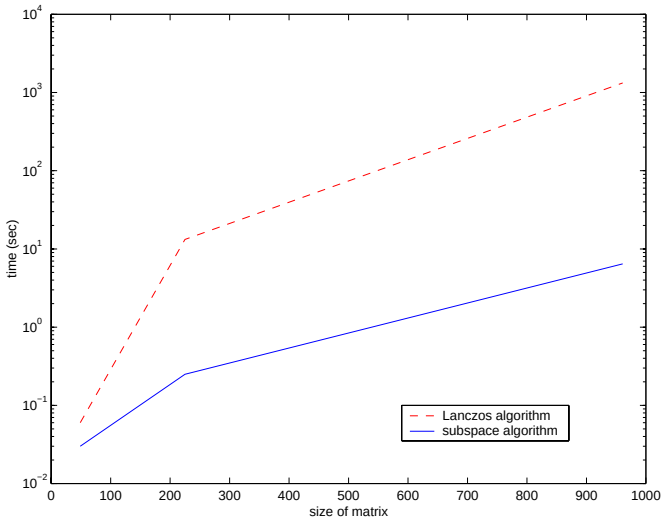

Fig. 2. Timings comparing the subspace implementation of extended eigenproblem against the Lanczos implementation of extended eigenproblem.

References

1. C. J. Alpert, T. F. Chan, D. J.-H. Huang, A. B. Kahng, I. L. Markov, P. Mulet, and K. Yan. Faster Minimization of Linear Wirelength for Global Placement. In *Proc. Intl. Symposium on Physical Design*, 1997.
2. R. P. Brent. *Algorithms for Minimization without Derivatives.* Automatic Computation. Prentice–Hall, Englewood Cliffs, NJ, 1973.
3. G. Golub and C. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, Maryland, 3rd edition, 1996.
4. B. Hendrickson and R. Leland. The Chaco user's guide, version 2.0. Technical Report SAND-95-2344, Sandia National Laboratories, 1995.
5. B. Hendrickson, R. Leland, and R. V. Driessche. Enhancing data locality by using terminal propagation. In *Proc. 29th Hawaii Intl. Conf. System Science*, volume 16, 1996.

6. M. Holzrichter and S. Oliveira. A graph based Davidson algorithm for the graph partitioning problem. *International Journal of Foundations of Computer Science*, 10:225–246, 1999.
7. M. Holzrichter and S. Oliveira. A graph based method for generating the Fiedler vector of irregular problems. In *Lecture Notes in Computer Science 1586, Parallel and Distributed Processing, 11 IPPS/SPDP'99*, pages 978–985, 1999.
8. G. Karypis and V. Kumar. METIS: Unstructured graph partitioning and sparse matrix ordering system Version 2.0. Technical report, Department of Computer Science, University of Minnesota, 1995.
9. George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392 (electronic), 1999.
10. B. Kerninghan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49:291–307, 1970.
11. J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, Berlin, Heidelberg, New York, 1999.
12. A. Pothen, H. Simon, and K. P. Liou. Partitioning sparse matrices with eigenvector of graphs. *SIAM J. Matrix. Anal. Appl.*, 11(3):430–452, 1990.