ParComb

Parallel Algorithms for Combinatorial Scientific Computing

Department of Informatics and Department of Mathematics,
University of Bergen

1 Project Overview

Although scientific computing is traditionally viewed as the province of continuous mathematics, of differential equations and linear algebra, there are many combinatorial subproblems that arise in the solution of scientific computing problems.

Research on these types of problems has been carried out for several years but it is only recently that it has been recognized as a field of its own, now known as "Combinatorial scientific computing" (CSC). This is an interdisciplinary research area involving discrete mathematics in scientific computing and refers to the development, analysis, and application of combinatorial algorithms to solve problems in computational science and engineering. The importance and interest in the field is mirrored by the number of conferences and workshops dedicated to CSC that has been arranged over the last years.

As an example of the role that CSC plays in scientific computing consider the problem of computing a matrix-vector product $y = Ax$ where A is large and sparse. This is the core operation of almost every iterative equation solver. When this operation is carried out on a parallel computer, as is often the case for large applications, it has long been recognized that one should partition A so that each processor gets an almost equal share of the data elements while at the same time minimizing the amount of required communication. Achieving such a partitioning is a pure combinatorial problem.

By viewing the matrix A as either a graph or a hypergraph researchers have developed a number of software packages for graph partitioning that can solve this task [13, 15, 16, 35, 36, 37, 38]. Although these software packages are sophisticated and complex they are still easy to use, runs fast, and will in most cases produce partitions of high quality. The best way to see the impact of this software is in the way the issue of partitioning is treated in scientific publications. Whereas researchers would previously have to develop and explain their own partitioning software to great length and depth, now there is usually just a comment that says "the matrix was partitioned with package xx".

It is the intention of the current applicant group to study other such common problems motivated from scientific computations and to develop algorithms and software libraries that offer solutions that can easily be used by other researchers. Since large scale applications typically run on parallel computers it is our intention that the developed software should itself be parallel, otherwise the combinatorial parts might become bottlenecks. The specific problems that we will be investigating stem from sparse linear algebra and are motivated and described in more detail in Section 2.

The applicant group consists of researchers from both mathematics and informatics and has a long and successful record within scientific computing in general but also more specifically in working with both parallel and CSC type problems. The main part of this project will be research and software developed in close collaboration with two new Ph.D. candidates. Through our international partners we will be able to offer these candidates a unique possibility to participate in cutting-edge research that will have a direct impact on how scientific computing is performed.

Management, members, and partners

The applicant group consists of the following permanent faculty members from the Department of Infor-

matics and the Department of Mathematics, at the University of Bergen (UoB), here listed in alphabetical order:

- Professor Petter Bjørstad, Department of Mathematics, UoB
- Professor Pinar Heggernes, Department of Informatics, UoB
- Professor Fredrik Manne, Department of Informatics, UoB

The project asks for financing of two Ph.D. students for three years. As part of their training these students will spend 6 month at Old Dominion University in Virginia, USA, under the guidance of Professor Alex Pothen who is one of the leading researchers in the field of combinatorial scientific computing.

In addition to Dr. Pothen, three additional partners, who are also leading researchers in the area and of varying backgrounds, have been invited as active participants of the project. They have all enthusiastically accepted the invitation, specifically by accepting mutual visits between Bergen and their respective institutes, and to participate in advising and collaborating with the Ph.D. students in the project. As our partners and their involvement in numerical linear algebra and CSC are well known to the broad scientific community we find it sufficient to list their names alphabetically with a pointer to their respective web-pages. We also attach a letter of interest from each.

- Iain Duff, Rutherford Appleton Laboratory, United Kingdom/CERFACS, France.
  `http://www.numerical.rl.ac.uk/people/isd/isd.html`
- John Gilbert, University of California, Santa Barbara, USA.
  `http://www.cs.ucsb.edu/~gilbert/`
- Bruce Hendrickson, Sandia National Laboratories, USA.
  `http://www.cs.sandia.gov/~bahendr/`
- Alex Pothen, Old Dominion University, USA.
  `http://www.cs.odu.edu/~pothen/`

Professor Fredrik Manne will function as the Principal Investigator (PI). His background both from applied industry as well as academia makes him well suited to lead the ParComb project. Professor Manne has been working in the field of combinatorial scientific computing for the last 15 years and has published a number of papers in the field with the bulk of his work being on designing parallel algorithms for CSC type problems [8, 24, 25, 26, 40, 42, 43, 44, 45].

After graduating from the University of Bergen he spent two years working as a senior system consultant for the Norwegian oil company Norsk Hydro with a main responsibility for large scale applications. Following this he worked as a senior scientist at the University of Bergen Supercomputing center Parallab before joining the Department of Informatics in 1998 as an associate professor. He also held an adjunct position at the University of Oslo in 2000. In 2002 he was promoted to full professor.

He spent the academic year 2004-2005 on sabbatical leave at Sandia National Laboratories, New Mexico, USA, working on the development of parallel coloring codes for the Zoltan library [10, 11] as well as developing new sequential coloring algorithms for computing sparse derivate matrices [28].

During this stay he strengthened and established new relations with several of the leading groups involved in combinatorial scientific computing in the USA. It is partly these contacts that has lead to the current application.

His main line of research has been focused on algorithms both sequential and parallel, with a preference for validating analytical results through practical simulations. Lately, his research focus has also expanded into distributed computing and various problems related to communication networks. He is the PI of the IKT-2010 funded project "Spectrum management in static and dynamic networks" with a total budget of 5.5M NOK. He has graduated 14 M.Sc. students and two Ph.D. student and is currently the supervisor of 1 doctoral student and 3 M.Sc students.

Professor Bjørstad has been the main architect in establishing a research program in computational science at the University of Bergen and has also been a driving force in establishing the Norwegian super computer infrastructure. He initiated Parallab, the super computing center at the University of Bergen, in 1990 and served as its operating director until 2002, when he became research director for the Bergen Center for Computation Science, which in addition to Parallab, also includes the externally funded research at UoB on computational science. His main line of research is in domain decomposition methods [3, 4, 5, 7] as well as parallel computing [6, 8, 12].

He has managed several large grants within computational science funded both from the Norwegian Research Council and from the EU. He has graduated 9 Ph.D. students and also serves on the advisory board of the eVITA program.

Professor Heggernes is a very active researcher and has served on the program committee of the Scandinavian Workshop on Algorithm Theory several times. Her research is to a large extend focused on computing orderings for sparse matrices [1, 2, 9, 32, 33, 34].

She holds several appointments related to the infrastructure for scientific computing in Norway, both as a member of the directing boards of the eVITA program and of UNINETT Sigma AS, which is responsible for the procurement and operation of national equipment for high performance computing infrastructure in Norway.

All members are internationally active researchers. Since 1998 the University of Bergen group has more than 50 papers either appearing in refereed journal publications or in conferences with LNCS-(type) proceedings.

In 2002 the Research Council of Norway presented an international evaluation of all academic research groups within the field of ICT. In this evaluation the Numerics group at the University of Bergen to which Dr Bjørstad belongs was given the highest score of "Excellent", while the Algorithms group to which the other applicants belong was given the second highest score of "Very Good".

Resources sought

The project is planned to start August 2006 and run for 4 years. We apply for funds for the following:

- 2 Ph.D. student positions of 3 years' duration each, with necessary equipment and a 6-month research stay at Old Dominion University.
- Funds to travel to conferences, and to visit our international partners, one trip per year per project member.
- Funds to invite our international partners and other research collaborators to Bergen.

The members of the applicant group, the 3 UoB professors will contribute with 33% of their work time each. The cost of the various parts are given in the following table (in 1000 NOK).

Overall costs (in 1000 NOK)	**2006**	**2007**	**2008**	**2009**	**Sum**
Doctoral student	296	592	592	296	1776
+ Equipment	50				50
+ Visit abroad		60			60
Doctoral student		592	592	592	1776
+ Equipment		50			50
+ Visit abroad			60		60
Visits to Bergen	50	50	50	50	200
Conference travel	100	100	100	100	400
Equipment	30	30	30	30	120
Total applied for from NFR	526	1474	1424	1068	4472
33% time of 3 UoB professors (funded by UoB)	800	800	800	800	3200
Project total	1326	2274	2224	1868	7692

Gender issues

The current applications asks for financing for two PhD positions. As it is not permissible by Norwegian law to only allow one gender to apply for a position we cannot guarantee that we will be able to fill one or both of the positions with a female applicant. But what we can say is that we will use all available means that we have at our disposal to do so. This means that we will actively encourage females to apply for the positions and we will give preference to a female applicant if she is deemed to be of the same quality as a male applicant.

The applicant group has a track record for attracting very good female candidates, Heggernes is currently supervising three PhD students, one of which is female, and in the project "Spectrum management in static and dynamic networks" being led by Manne one out of the two PhD students is also a female. Both of these were chosen from a highly competitive field, mainly due to their high academic quality but we also gave preference to them because of gender issues. At the moment our group is supervising several good female Master students who might be possible applicants for the positions.

It should also be noted that Pinar Heggernes is the only female full professor at the Department of Informatics and the Department of Mathematics at the University of Bergen. Although she would have been a natural project leader she has chosen to not to be so in order to avoid any conflict of interest with her involvement in the eVITA program.

2 Focus areas

Graphs are a unifying abstraction for many fundamental problems in scientific computing as they are a natural language for expressing data dependencies or sparse matrices. Algorithms on graphs are critical steps for improving memory and computational performance of key scientific operations. Unfortunately, the language of graph theory is unfamiliar to most computational scientists, and the needs of scientific computations are unfamiliar to most experts in graph algorithms. To help bridge this chasm, the discipline of *Combinatorial Scientific Computing (CSC)* has recently been founded. Although this formal labeling of the field has increased the awareness of issues related to CSC, the problems which it addresses are not new. We point to a few of the areas where combinatorial algorithms play an important role.

Data movement With increased cpu clock rates the ratio of compute speed versus memory speed has gotten consistently worse. Thus it is often the case that it is not the speed of the cpu that is the limiting factor of an application but instead it is the movement of data between main memory and the cpu. Data movement is also an issue when using parallel computers. Then one would like to move as little data as possible between the processors to reduce communication. However, for adaptive and dynamic systems this is unavoidable and one must focus on how to do this in an efficient and cost effective manner.

Ordering of computations In many applications one has a certain freedom in how one orders the underlying matrix. This can for instance be in the direct solution of sparse system of equations where one can reorder the matrix to reduce the amount of introduced fill elements. Another example is in the computation of sparse derivate matrices in optimization problems where the grouping of columns can be used to reduce the number of subsequent function evaluations needed to estimate the matrix. By using combinatorial algorithms one can reduce the subsequent amount of work and memory usage.

Advanced data structures There exist a number of examples including multi level algorithms, domain decomposition methods and adaptive mesh refinements, that all require the use of advanced data structures and algorithms to organize the computations. Algorithm theory has a long history of designing and analyzing the use of advanced data structures. However there is often a problem that the produced solutions are to theoretically oriented to be easily used in scientific computing where the actual running time is of major importance.

Scientific applications typically involve the solution of partial differential equations through simulations that involve large data sets. It has long been realized that the only way to get high performance on these type of problems is through the use of parallel computers. Once the data has been partitioned on to the processors the core numerical operations such as matrix-vector multiplication are typically trivially parallellizable. However, as discussed above, if the underlying model is unstructured and sparse there can be a substantial overhead in managing the data and just getting to the point where one can perform the numerical operations. Thus if these combinatorial parts are not parallelized themselves they might become bottlenecks.

It is these types of combinatorial problems that we will be investigating in the ParComb project. They are characterized by *i*) They occur frequently, *ii*) They are done in order to speedup/enhance the ensuing numerical computations, and *iii*) One cannot spend to much time on them, otherwise they might become bottlenecks themselves.

In sequential applications these types of problems are typically solved using some kind of greedy strategy. This ensures that one only has to look at each data element once and thus results in a linear or close to linear running time. The reason for this is that anything more sophisticated would be to costly and the gained subsequent savings from using a more elaborate scheme would be outweighed by the time needed to compute it. These types of algorithms might be used even for problems where there exist exact solutions that run in polynomial time since these might be to slow for practical purposes.

Greedy algorithms are inherently sequential since each decision depends on the previous choices made so far. This is a particular problem when running on a parallel computer. If one has to resort to a sequential algorithm for a specific combinatorial problem then this would require that one first gathers the data on one processor where one solves the problem and then broadcast the solution. So even if the sequential algorithm is quite fast this operation can still become a bottleneck due to data traffic.

In the following we motivate three types of such problems and outline the work we intend to do. It should be noted that all of these problems arise in applications where one typically needs to solve the same type of problem a large number of times.

Matchings

Motivation: A matching in a graph is a pairing of adjacent vertices such that each vertex is matched with at most one other vertex, the objective being to match as many vertices as possible or that the sum of the weights of the matched edges is maximized.

One application of matchings in scientific computations is when using pivoting in the direct solution of a system of equations $Ax = b$. Once a pivot column i has been chosen no other element in column i can be used as a pivot again. The typical strategy is then to choose pivots in a greedy fashion. But as was shown by Duff and Koster [20, 21] this is can lead to sub-optimal results and they demonstrate that better results can be achieved by modeling the pivoting problem as computing a matching in a bipartite graph. This is done by viewing A as a bi-partite graph $G(V_1, V_2, E)$ where there is one vertex in V_1 for each row of A, one vertex in V_2 for each column of A, and the weight of edge (i, j) is equal to $|a_{ij}|$. Then any selection of pivots is equivalent to computing a matching in G.

Another application in scientific computing where a matching is needed is in software for graph partitioning where the graph in question represents the structure of a sparse matrix A. All such software programs have in common that the partitioning is carried out by first shrinking A by (structurally) collapsing similar columns or rows until A reaches a moderate size on which one can perform a crude partitioning. This partitioning is then expanded in a number of levels until one has a complete partitioning of A, at each level performing local refinements of the current partitioning. The shrinking phase can be modeled as performing multiple rounds of matchings on a series of graph where there initially is on vertex for each column and the weight of an edge is some measurement of similarity between the two associated columns.

One example of such a system is the Chaco package developed by Hendrickson [35]. This work has since then been extended into the Zoltan package [15, 16] which also includes hypergraph partitioning.

Our contribution: Even though a maximum matching can be computed in polynomial time the time

complexity of doing this is still high. However, the last years has seen a number of new fast approximation algorithms for the weighted matching problem [18, 19, 46, 49]. We believe that these offer the possibility to design new efficient parallel algorithms where one exploits the structure of the matrix to extract parallelism or where this is achieved through randomization in the selection of edges. We intend to explore both of these venues in the current project.

There has also been very recent developments by Pothen and his group in producing fast matching algorithms for the vertex weighted matching problems [17]. We believe that it should also be possible to parallelize these algorithms using similar types of techniques as for the edge weighted problem.

Coloring

Motivation: Our next area of interest stems from computing sparse Jacobian and Hessian matrices in optimization applications. Here it is of vital importance to partition the matrix to exploit structural independence between the rows and columns in order to reduce the number of function evaluations needed to estimate the matrix in question. Since the pioneering work of Coleman and More in the early 1980s, graph coloring has been found to be a powerful tool for analyzing the complexity of and designing effective algorithms for these partitioning problems. Together with Pothen and Gebremedhin we published a 77-pages article in the December 2005 issue of SIAM Review where we have synthesized the research around these problems of the past two decades and developed a unifying graph-theoretic and algorithmic framework [25]. Depending on the type of matrix (Jacobian vs. Hessian), the type of partitioning used (1d vs. 2d), and how the entries are computed, a number of different coloring problems are introduced.

Our contribution: Over the last years we have been heavily involved in developing fast parallel graph coloring algorithms [10, 11, 23, 24, 26]. Recently we developed new scalable parallel coloring algorithms for distance-1 [10] and distance-2 [11] coloring of graphs. These algorithms are based on sequential greedy coloring algorithms and use partitioning of the vertices and randomization in the coloring to extract parallelism.

Although some of the instances for computing Jacobians and Hessians are covered by our existing work there are still a number of unsolved problems motivated from [25]. For instance, the algorithms do not cover the important case of coloring hypergraphs. This should be possible to achieve using similar types of techniques as those previously used and is something that we intend to develop in the current project.

Also, in [28] we have recently developed two new efficient sequential star and acyclic coloring algorithms for computing Hessians. These algorithms are based on maintaining and exploiting the structure of the two colored induced subgraphs in the partially colored graph. Preliminary experimental studies using the ADOL-C package [31] have shown that these algorithms render dramatic savings in overall runtime for computing Hessians [27]. Although these algorithms are greedy they are still more complicated than regular graph coloring algorithms as they require maintaining global data structures. As part of our work we intend to investigate how these can be parallelized.

Finally, we mention that due to memory constraints it is not always suitable for an application to have a vertex and its adjacent edges stored on a single processor [15] or even to have have an explicit representation of the underlying graph [42]. These settings require additional effort to produce good colorings.

Orderings

Motivation: A direct solution of a sparse positive definite system $Ax = b$ involves first factoring A into LL^T using Cholesky factorization and then solving the associated triangular systems. Here $L + L^T$ is a superset of A with the extra elements referred to as *fill-elements*. The factorization is usually carried out in three separate stages, first an appropriate fill reducing ordering of A is computed. From this, the structure of $L + L^T$ can then be computed before one performs the numerical factorization.

In a parallel application it is also important in addition to reducing fill, to order the matrix to ensure that one obtains maximum parallelism in the ensuing numerical factorization. This is modeled by the data structure known as the *elimination tree* [39].

Typical algorithms that are used for computing the fill reducing ordering are Minimum Degree [30]

and Nested Dissection [29]. Today it is fairly well understood how the ensuing numerical factorization can be carried out in parallel. However, less is known about how to compute the orderings themselves and to predict the structure of intermediate fill elements in parallel. Thus this stage is typically still carried out sequentially. On massively parallel computers it is therefore possible that the ordering step might become a bottleneck and it is thus important to be able to do this step in parallel.

Another important issue is to take a given ordering and to reorder this in order to make it minimal. This involves finding an equivalent ordering as the given one but that does not introduce any unnecessary fill elements. As has been shown this can lead to substantial reductions in both time and space usage [9, 32].

Our contribution: We have a very good theoretical understanding of how the Minimum Degree algorithms works [2, 33, 34] as well as a experience on on how to produce and exploit orderings that give low elimination trees [1, 41, 44]. Combining this with the knowledge of our partners [14, 47, 48] on Minimum Degree implementations we are in a very good position to start designing parallel Minimum Degree algorithms.

Project member Heggernes is one of the leading experts on computing minimal fill orderings, something that is underscored by her recent survey article on the subject [32]. We will design parallel algorithms for these problems by exploiting the inherent parallelism that exist in several of the proposed algorithms for computing minimal fill orderings

We also note the recent article by Eisenstat and Liu on how to extend the theory of elimination trees to sparse unsymmetric matrices [22]. Using this we believe that it should be possible to carry over some of our work to the case where A is unsymmetric.

Software libraries

Although there is an abundance of useful computer software and libraries available in computer science and scientific computing, still only a small portion of this gets significant use by others than the developers. We believe that there are several reasons for this. One is that the interface to the software is too complicated. It might also be that there are too many options that the user has to specify or that the software is not well known.

In our previous work we have demonstrated that we are able to produce high quality software and integrate it with other libraries. As an example, Heggernes, Manne and Duff participated in the EU funded Parasol project that developed different parallel solvers, all with a common front end. We intend to continue this in the current project. To make usage of the libraries both easy and flexible we will produce both a generic interface as is used by Zoltan [16] where no assumptions are made about the underlying data structure, and also a version where only a very simple data structure is assumed and the routine can be invoked through a single subroutine call. This last version is similar to what is used by Metis [37].

Through extensive testing we will supply recommended values for system parameters but still offer the option for the user to tune these as she sees fit. To disseminate knowledge about our work we will present our results at appropriate conferences as well as in scientific journals. Also, our international partners are an insurance that our work will be known within the scientific community.

3 Impact

The ParComb project will produce a number of key enabling technologies that rely on graph algorithms. As described, these include tools that will have a wide user base in scientific computing. In addition to the produced software and scientific results the project will also have a lasting impact through the doctoral students who once graduated will bring with them their training and knowledge into future jobs in industry and academia.

The project will increase the visibility and awareness of combinatorial abstractions and their power in scientific applications. The contribution of graph algorithms is often hidden inside tools that are opaque to

application developers. But further progress in scientific computing will rely crucially on greater attention being paid to these capabilities.

Project members Pothen and Hendrickson are currently involved in a bid under the US funded SciDAC program to establish an institute on combinatorial scientific computing and petascale simulations. This proposal also involves scientists who are developing applications in areas such as biomolecular and circuit simulations. Thus collaboration with this project will give us direct contact with an even larger base of relevant application developers.

Right now we have a window of opportunity to achieve the goals set forth in our proposal. We have a very strong international project team, there is a demand for our solutions, and we have the ideas and knowledge needed to produce them. But in order to do so we need financing for our doctoral students as well as travel funds to present our results at international conferences, and also to meet with our partners face-to-face. If we can obtain this we will be able to exploit and maintain our competitive edge while achieving a substantial impact on scientific computing.

References

[1] B. Aspvall and P. Heggernes, *Finding minimum height elimination trees for interval graphs in polynomial time*, BIT, 34 (1994), pp. 484 – 509.

[2] A. Berry, P. Heggernes, and G. Simonet, *The minimum degree heuristic and the minimal triangulation process*, Proceedings of the 29^{th} Workshop on Graph Theoretic Concepts in Computer Science, (2003), pp. 58 – 70. LNCS 2880.

[3] P. Bjørstad, M. Dryja, and T. Rahman, *Efficient schwarz methods for elliptic mortar finite element problems*, 2002, pp. 303–310.

[4] P. Bjørstad, M. Dryja, and R. Talal, *Additive schwarz methods for anisotropic elliptic problems*, IMA Volumes in Mathematics and its Applications, (2000), pp. 279–294.

[5] ——, *Additive schwarz methods for elliptic mortar finite element problems*, Numerische Mathematik, 95 (2003), pp. 427–457.

[6] P. Bjørstad and J. Koster, *Balancing domain decomposition applied to structural analysis problems*, Advances in Parallel Computing 2003, 13 (2003), pp. 845–852.

[7] P. Bjørstad, J. Koster, and P. Krzyzanowski, *Domain decomposition solvers for large scale industrial finite element problems*, 2000, pp. 374–384.

[8] P. Bjørstad, F. Manne, T. Sørevik, and M. Vajteršic, *Efficient matrix multiplication on simd computers*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 386–401.

[9] J. R. S. Blair, P. Heggernes, and J. A. Telle, *A practical algorithm for making filled graphs minimal*, Theor. Comput. Sci., 250 (2001), pp. 125–141.

[10] E. Boman, D. Bozdag, U. Catalyurek, A. H. Gebremedhin, and F. Manne, *A scalable parallel graph coloring algorithm for distributed memory computers*, in Proceedings of EuroPar 2005, vol. 3646, Lecture Notes in Computer Science, Springer, 2005, pp. 241–251.

[11] D. Bozdag, U. Catalyurek, A. H. Gebremedhin, F. Manne, E. Boman, and F. Ozguner, *A parallel distance-2 graph coloring algorithm for distributed memory computers*, in Proceedings of HPCC-05, vol. 3726, Lecture Notes in Computer Science, Springer, 2005, pp. 796–806.

[12] J. Brækhus, P. Bjørstad, and J. Cook, *Multilevel parallel solution of large, sparse finite element equations from structural analysis*, 1996.

[13] U. V. Catalyurek and C. Aykanat, *PaToH: A multilevel hypergraph partitioning tool, Version 3.0.* PaToH is available at http://bmi.osu.edu/~umit/software.htm, 1999.

[14] T. A. Davis, J. R. Gilbert, S. I. Larimore, and E. G. Ng, *A column approximate minimum degree ordering algorithm*, ACM Trans. Math. Softw., 30 (2004), pp. 353–376.

[15] K. Devine, E. Boman, R. Heaphy, R. Bisseling, and U. V. Catalyurek, *Parallel hypergraph partitioning for scientific comuting*, 2006. To appear.

[16] K. Devine, E. Boman, R. Heaphy, B. Hendrickson, and C. Vaughan, *Zoltan data management services for parallel dynamic applications*, Computing in science and engineering, 4 (2002), pp. 90–97.

[17] F. Dobrian, M. Halappanavar, and A. Pothen, *Fast approximation algorithms for bipartite vertex-weighted matching*. In preparation, 2006.

[18] D. E. Drake and S. Hougardy, *A simple approximation algorithm for the weighted matching problem*, Information Processing Letters, 85 (2003), pp. 211–213.

[19] ——, *A linear time approximation algorithm for weighted matchings in graphs*, ACM Transactions on Algorithms, 1 (2005), pp. 107–122.

[20] I. S. Duff and J. Koster, *The design and use of algorithms for permuting large entriees to the diagonal of sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 889–901.

[21] ——, *On algorithms for permuting large entries to the diagonal of a sparse matrix*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 973–996.

[22] S. C. Eisenstat and J. W. H. Liu, *The theory of elimination trees for sparse unsymmetric matrices*, SIAM Rev., 26 (2005), pp. 686–705.

[23] A. Gebremedhin, F. Manne, and A. Pothen, *Parallel distance-k coloring algorithms for numerical optimization*, in proceedings of Euro-Par 2002, vol. 2400, Lecture Notes in Computer Science, Springer, 2002, pp. 912–921.

[24] A. H. Gebremedhin and F. Manne, *Scalable parallel graph coloring algorithms*, Concurrency: Practice and Experience, 12 (2000), pp. 1131–1146.

[25] A. H. Gebremedhin, F. Manne, and A. Pothen, *What color is your Jacobian? Graph coloring for computing derivatives*, SIAM Rev., 47 (2005), pp. 629–705.

[26] A. H. Gebremedhin, F. Manne, and T. Woods, *Speeding up parallel graph coloring*, in Proceedings of PARA 2004, vol. 3732, Lecture Notes in Computer Science, Springer, 2005, pp. 1079–1088.

[27] A. H. Gebremedhin, A. Pothen, A. Tarafdar, and A. Walther, *Efficient computation of sparse hessians: An experimental study using ADOL-C*. In prep., 2006.

[28] A. H. Gebremedhin, A. Tarafdar, F. Manne, and A. Pothen, *New acyclic and star coloring algorithms with applications to computing hessians*. submitted, 2005.

[29] A. George and J. W. H. Liu, *Computer Solutions of Large Sparse Positive Definite Systems*, Prentice-Hall, 1981.

[30] ——, *The evolution of the minimum degree ordering algorithm*, SIAM Rev., 31 (1989), pp. 1–19.

[31] A. Griewank, D. Juedes, and J. Utke, *ADOL-C: A package for the automatic differentiation of algorithms written in c/c++*, ACM Trans. Math. Softw., 22 (1996), pp. 131–167.

[32] P. Heggernes, *Minimal triangulations of graphs - A survey*, Dicrete Math., 306 (2006), pp. 297–317.

[33] P. Heggernes, S. Eisenstat, G. Kumfer, and A. Pothen, *The computational complexity of the Minimum Degree algorithm*, in Proceedings of 14th Norwegian Computer Science Conference, University of Tromsø, Norway, 2001, pp. 98–109.

[34] P. Heggernes and Y. Villanger, *Modifying a given elimination ordering to reduce fill*, in Proceedings of PARA 2004 Workshop on State-of-the-Art in Scientific Computing, Springer Verlag, 2004, pp. 788 – 797. LNCS 3732.

[35] B. Hendrickson and R. Leland, *The Chaco user's guide, version 2.0*, 1995.

[36] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, *Multilevel hypergraph partitioning: application in VLSI domain*, 1997, pp. 526 – 529.

[37] G. Karypis and V. Kumar, *Metis a software package for partitioning unstructured graphs, partitioning meshe, and computing fill-reducing orderings of sparse matrices Version 4.0*, 1998.

[38] G. Karypis, K. Schloegel, and V. Kumar, *Parmetis: Parallel graph partitioning and sparse matrix ordering library, Version 3.1*, tech. rep., Dept. Computer Science, University of Minnesota, 2003.

[39] J. W. H. Liu, *The role of elimination trees in sparse factorization*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 134–172.

[40] F. Manne, *An algorithm for computing an elimination tree of minimum height for a tree*, in Presented at the Second Meeting of the International Linear Algebra Society, Lisbon, Portugal, 1992.

[41] F. Manne, *An algorithm for computing an elimination tree of minimum height for a tree*, Tech. Rep. CS-91-59, University of Bergen, Norway, 1992.

[42] F. Manne, *A parallel algorithm for computing the extremal eigenvalues of very large sparse matrices (extended abstract)*, in proceedings of Para98, Workshop on Applied Parallel Computing in Large scale scientific and Industrial Problems, vol. 1541, Lecture Notes in Computer Science, Springer, 1998, pp. 332–336.

[43] F. Manne and S. O. Andersen, *Automating the debugging of large numerical codes*, Birkhauser Verlag, 1997, ch. 16, pp. 339–352.

[44] F. Manne and H. Hafsteinsson, *Efficient sparse Cholesky factorization on a parallel SIMD computer*, SIAM J. Sci. Comput., 16 (1995), pp. 934–950.

[45] F. Manne and T. Sørevik, *Partitioning an array onto a mesh of processors*, in In proceedings of Para'96, Workshop on Applied Parallel Computing in Industrial Problems and Optimization, Lecture Notes in Computer Science 1184, Springer, 1996, pp. 467–477.

[46] S. Pettie and P. Sanders, *A simpler linear time* $2/3 - \epsilon$ *approximation for maximum weight matching*, Inf. Process. Lett., 91 (2004), pp. 271–276.

[47] T. A. D. P.R. Amestoy and I. S. Duff, *An approximate minimum degree ordering algorithm*, SIAM J. Matrix Anal. and Applics., 17 (1996), pp. 886 – 905.

[48] ——, *Amdm an approximate minimum degree ordering algorithm*, ACM Trans. Math. Softw., 30 (2001), pp. 381 – 388.

[49] R. Preis, *Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs*, in Proceedings of STACS 99, vol. 1563, Lecture Notes in Computer Science, Springer, 1999, pp. 259–269.