**Abstract**

Sparse matrix computations arise in numerous computational science and engineering computations as well as in network analysis and data-based simulations. On parallel computing platforms, however, sparse matrix computations represent a major impediment to realizing high performance. Our project aims at designing and implementing solvers for: (i) large sparse linear systems, and (ii) large sparse symmetric eigenvalue problems that achieve high performance on a single multicore node and clusters of many multicore nodes. Further, we demonstrate both the superior robustness and parallel scalability of our solvers compared to other publicly available solvers.
ABSTRACT

Sparse matrix computations arise in numerous computational science and engineering computations as well as in network analysis and data-based simulations. On parallel computing platforms, however, sparse matrix computations represent a major impediment to realizing high performance. Our project aims at designing and implementing solvers for: (i) large sparse linear systems, and (ii) large sparse symmetric eigenvalue problems that achieve high performance on a single multicore node and clusters of many multicore nodes. Further, we demonstrate both the superior robustness and parallel scalability of our solvers compared to other publicly available parallel solvers for these two fundamental problems.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received  Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received  Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations


Number of Presentations: 2.00

---

**Non Peer-Reviewed Conference Proceeding publications (other than abstracts):**

<table>
<thead>
<tr>
<th>Received</th>
<th>Paper</th>
</tr>
</thead>
</table>

**TOTAL:**

---

**Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):**

---

**Peer-Reviewed Conference Proceeding publications (other than abstracts):**

<table>
<thead>
<tr>
<th>Received</th>
<th>Paper</th>
</tr>
</thead>
</table>

**TOTAL:**

---

**Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):**

---

**(d) Manuscripts**

<table>
<thead>
<tr>
<th>Received</th>
<th>Paper</th>
</tr>
</thead>
</table>

05/29/2015  4.00  Yao Zhu, Ahmed H Sameh. PSPIKE+: a family of parallel hybrid sparse linear system solvers, ( )

**TOTAL:**  1
### Books

<table>
<thead>
<tr>
<th>Received</th>
<th>Book</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL:**

<table>
<thead>
<tr>
<th>Received</th>
<th>Book Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL:**

### Patents Submitted

### Patents Awarded

### Awards

### Graduate Students

<table>
<thead>
<tr>
<th>NAME</th>
<th>PERCENT_SUPPORTED</th>
<th>Discipline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yao Zhu</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td><strong>FTE Equivalent:</strong></td>
<td><strong>0.50</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Total Number:</strong></td>
<td><strong>1</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Names of Post Doctorates

<table>
<thead>
<tr>
<th>NAME</th>
<th>PERCENT_SUPPORTED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FTE Equivalent:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Total Number:</strong></td>
<td></td>
</tr>
</tbody>
</table>
Names of Faculty Supported

<table>
<thead>
<tr>
<th>NAME</th>
<th>PERCENT_SUPPORTED</th>
<th>National Academy Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ahmed H Sameh</td>
<td>0.10</td>
<td></td>
</tr>
</tbody>
</table>

FTE Equivalent: 0.10
Total Number: 1

Names of Under Graduate students supported

| NAME
| PERCENT_SUPPORTED
| FTE Equivalent:
| Total Number:

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period.

- The number of undergraduates funded by this agreement who graduated during this period: 0.00
- The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields: 0.00
- The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields: 0.00
- Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale): 0.00
- Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering: 0.00
- The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense: 0.00
- The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields: 0.00

Names of Personnel receiving masters degrees

| NAME
| Total Number:

Names of personnel receiving PHDs

<table>
<thead>
<tr>
<th>NAME</th>
<th>Total Number:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alicia Marie Klinvex</td>
<td>1</td>
</tr>
</tbody>
</table>

Names of other research staff

<table>
<thead>
<tr>
<th>NAME</th>
<th>PERCENT_SUPPORTED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FTE Equivalent: 
Total Number: 

Sub Contractors (DD882)
The only significant technology transfer involving an Army Research Lab took place when I was contacted by Dr. Betsy Rice to help in speeding up the parallel implementation of the following computation (sparse matrix – sparse matrix multiplication) in a loop:

```plaintext
for i = 1: max_iter
    if (trace(A) > threshold)
        A = A*A
    else
        A = 2*A – A*A
    end
end
```

Analyzing the graphs represented by the matrices A in the above loop, we observed that all the matrices A of order n can be reordered by the same permutation matrix P such that PT A P = E, where E is all zero except of a first dense diagonal block C of order r much less than n. This allowed us to perform all the multiplications in the loop using the high data-locality dense matrix multiplications involving the matrix C, and retrieving A via the reverse ordering: A = P E PT.

This approach resulted in significant savings. For example, for a loop of 17 iterations, the speed improvements realized by our scheme over the sparse matrix-sparse matrix multiplication kernel in the DOE Trilinos project for a matrix A of small size n = 23,552 was 2.4 if we use a single node of 80 cores. However, for a matrix A of a modest size of 565,238, we realized a speed improvement of 24 if we use the same single node with 80 cores, and a speed improvement of 10.4 if we use a cluster of 8 nodes in which each node contains 24 cores. The advantage of our approach would yield much higher speed improvements for matrices with much larger size.

Dr. Rice was pleased with the outcome of this collaboration and stated:

“This will help to enable a critical capability within the enterprise for multiscale material research at arl

Thanks to everyone!

Betsy”
Project title: Parallel Sparse Linear System and Eigenvalue Problem Solvers – from multicore to petascale computing

Principal Investigator: Ahmed H. Sameh, Department of Computer Science, Purdue University

Grant number: 7W911NF-11-1-0401

I. Introduction

Sparse matrix computations arise in numerous computational science and engineering applications as well as in network analysis and data-based simulations. Further, sparse matrix computations represent a major impediment to realizing high performance on parallel computing platforms. Designing sparse matrix primitives and algorithms capable of achieving high parallel scalability on platforms consisting of a single node with many cores, or thousands of multicore nodes is the main objective of our research effort. In this project, we addressed designing:

(1) Tools for sparse matrix primitives such as sparse matrix-vector multiplication (and sparse matrix-multivector multiplication) – “matvecs”, and sparse matrix reordering designed to enhance both the realization of effective preconditioning techniques for solving large linear systems, as well as high performance “matvecs”;
(2) A hybrid parallel sparse linear system solver that is much more robust than current preconditioned iterative solvers, and much more scalable than currently available sparse direct linear system solvers.

(3) A parallel sparse symmetric eigenvalue problem solver for obtaining either extreme or interior eigenpairs.

We believe that our kernels and solvers fill a critical need for researchers and developers of engineering applications in which robustness and speed are vital for the large-scale simulations to be conducted.

II. PSPIKE: a parallel hybrid sparse linear system solver

This algorithm was motivated by some of the early work of the PI, e.g. see [SaKu78], [PoSa06], and [MaSS09]. This solver consists of three critical primitives/algorithms: (a) sparse matrix reordering, (b) determination of an effective preconditioner for Krylov subspace methods, and (c) designing and implementing parallel schemes for solving systems involving the preconditioner in each outer Krylov iteration.

(a) Sparse matrix reordering: this reordering consists of two steps – the first is nonsymmetric reordering that removes zeros from the diagonal and maximizes the magnitude of the product of the diagonal elements; the second is a symmetric reordering that brings as many of the heaviest elements (i.e. off-diagonal elements of largest magnitudes) as close to the diagonal as possible. Our nonsymmetric reordering is similar to subroutine MC64 of the Harwell Subroutine Library (HSL), while our symmetric reordering is based on the Fiedler vector of the corresponding weighted Laplacian. In this sense it is similar to subroutine MC73 of HSL except that we use our own parallel TraceMIN eigensolver (perfected and extended through this grant) for obtaining the Fiedler vector rather than the multilevel eigensolver used in MC73 which yields low performance on a variety of parallel architectures. Currently, however, we are also experimenting with another graph partitioning scheme that enhances the success of block Jacobi preconditioners.

(b) Extraction of the preconditioner: the above reordering creates a central “generalized band” that can be used as a preconditioner for an outer Krylov subspace iteration. We use the term “generalized band” so as to allow a central band that consists of overlapped diagonal blocks in which each block is a sparse matrix. Such a construct allows the encapsulation of many off-diagonal element so that the reordered sparse coefficient matrix $A'$ can be expressed as $A' = M + E$, in which $M$ is the preconditioner consisting of overlapped diagonal blocks, and $E$ is the low-rank sparse matrix that lies
outside M. In this case, the number of outer Krylov subspace iterations will be proportional to the rank of E.

(c) Solving systems of the form $M z = r$: in order to realize maximum parallel scalability of our hybrid solver PSPIKE, we need to solve systems involving the preconditioner with maximum parallel efficiency. Here, we used our “tearing” method, e.g. see [NaMS10]. This algorithm proved to be very effective on parallel architectures. It gives rise to solving a “balance system” whose size is equal to the sum of the overlaps in M. If the balance system is formed explicitly and solved using a direct method, then this part will be the only impediment to high parallel scalability on large-scale parallel computing platforms. On a cluster of few multicore nodes (e.g. 4 or 8 nodes), however, the impediment to high parallel performance is minimized. On large clusters of many multicore nodes, the balance system is not formed explicitly and is solved using a preconditioned Krylov subspace method. In such iterative schemes, all that is needed is the result of multiplying the coefficient matrix of the balance system with a vector as well as computing residuals corresponding to different iterates. Both of these are derived from the direct solutions of the sparse systems that constitute the overlapped diagonal blocks of the preconditioner M. The last critical component is solving sparse linear systems (one per overlapped diagonal block) efficiently so as to obtain only certain components of the solution taking advantage of the sparsity of the right hand-side.

Extensive numerical experiments have shown that on large clusters of multicore nodes, our parallel solver PSPIKE is as robust as sparse direct solvers, and more robust as well as much more scalable on large number of multicore nodes than LU-and algebraic multigrid-preconditioned Krylov subspace methods. This has been demonstrated in previous annual reports of this grant. Further, we have demonstrated that our hybrid solver PSPIKE can be much faster than direct sparse solvers like Pardiso, MUMPS and WSMP if we need only to achieve solutions with modest relative residuals (e.g. in the range of $10^{-2}$ to $10^{-5}$).

As mentioned above, we have created a version of PSPIKE (PSPIKE+) in which: (i) we use a reordering scheme that enhances the effectiveness of block Jacobi preconditioners and simultaneously providing us with the benefits of weighted spectral reordering, and (ii) form the balance system explicitly and solve it directly. Appendix 1 contains a draft of the paper to be submitted soon to the Journal of Computational and Applied Mathematics.

III. TraceMIN: a parallel sparse symmetric eigenvalue problem solver

This algorithm was developed by the PI in 1982, e.g. see [SaWi82], based on the trace minimization property that given the generalized symmetric eigenvalue
problem \( Ax = \mu B x \) where \( A \) is symmetric and \( B \) is symmetric positive definite, then minimizing the trace of \( (Y^T A Y) \) subject to the constraint that \( Y^T B Y = I_p \), where \( Y \) is a block of \( p \) independent columns, and \( I_p \) is the identity of order \( p \), results in the following: \( \min [\text{tr}(Y^T A Y)] = \sum \mu_k \), the sum of the \( p \) smallest eigenvalues near zero. After almost a decade and half, the Jacobi-Davidson algorithm [SlVo96] was introduced for the nonsymmetric eigenvalue problem without a proof of convergence but which is based on the trace minimization property (used by the PI) when the eigenvalue problem is symmetric. Later, the PI extended TraceMIN to make use of the expanding subspace strategy adopted in Jacobi-Davidson, resulting in the solver TraceMIN-Davidson, e.g. see [SaTo00]. More recently, investigators at Sandia National Labs launched the Trilinos project aimed at implementation of sparse linear system and sparse eigenvalue problem solvers including: (i) LOBPCG [Knya01, Knya07], (ii) BKS [Stew00, ZhSa08], and (iii) RTR [AbBG07, ABGS04], which is based on our TraceMIN scheme. In this project, we implemented both TraceMIN and TraceMIN-Davidson on a cluster of multicore nodes and compared them with those of the Trilinos project (Anasazi library [Anas15]). The robustness exhibited by TraceMIN and TraceMIN-Davidson is due to the fact that unlike Lanczos- or Arnoldi-based eigensolver, our algorithms do not require solutions of linear systems that arise in each outer eigensolver iteration to have very low relative residuals. Further, the robustness and superior parallel scalability of TraceMIN and TraceMIN-Davidson rely on efficient algorithms for solving the saddle-point problems that arise from the above constrained minimization of the trace of a section of the matrix \( A \). In addition to comparing the performance of our algorithms with those currently in the Trilinos project, we also compare our solver with an eigensolver adopted recently by Intel’s Math Kernel Library (MKL) – a solver that relies on contour integration, e.g. see [Poli09, and Poli14]. Further, we also provide comparisons with the Jacobi-Davidson algorithm implemented in the SLEPc library [SLEP14].

Performance results of our sparse nonsymmetric linear systems and symmetric eigenvalue problem solvers are contained in Appendix 2.

IV. Education and training

Two PhD graduate students associated with project, one (Alicia Klinvex) supported by a federal fellowship and one (Yao Zhu) supported via this grant, have gained extensive experience in designing parallel algorithms for sparse matrix computation. Alicia Klinvex will graduate later this month (May 2015). She has already accepted a postdoctoral fellowship from Sandia’s Trilinos project. Based on the work she has done with me in developing the parallel TraceMIN and TraceMIN-Davidson, the Trilinos project has recently adopted both of them as eigensolvers in the Anasazi library. Yao Zhu, has been working on the PSPIKE hybrid linear system solver, and my co-author of the paper draft attached to this report. Yao Zhu will graduate this August and has already an offer as a technical analyst from a Wall Street investment firm.
V. Technology Transfer

The only significant technology transfer involving an Army Research Lab took place when I was contacted by Dr. Betsy Rice to help in speeding up the parallel implementation of the following computation (sparse matrix – sparse matrix multiplication) in a loop:

```matlab
for i = 1: max_iter
    if (trace(A) > threshold)
        A = A*A
    else
        A = 2*A - A*A
    end
end
```

Analyzing the graphs represented by the matrices A in the above loop, we observed that all the matrices A of order n can be reordered by the same permutation matrix P such that \( P^T A P = E \), where E is all zero except of a first dense diagonal block C of order r much less than n. This allowed us to perform all the multiplications in the loop using the high data-locality dense matrix multiplications involving the matrix C, and retrieving A via the reverse ordering: \( A = P E P^T \).

This approach resulted in significant savings. For example, for a loop of 17 iterations, the speed improvements realized by our scheme over the sparse matrix-sparse matrix multiplication kernel in the DOE Trilinos project for a matrix A of small size \( n = 23,552 \) was 2.4 if we use a single node of 80 cores. However, for a matrix A of a modest size of 565,238, we realized a speed improvement of 24 if we use the same single node with 80 cores, and a speed improvement of 10.4 if we use a cluster of 8 nodes in which each node contains 24 cores. The advantage of our approach would yield much higher speed improvements for matrices with much larger size.

Dr. Rice was pleased with the outcome of this collaboration and stated:

“This will help to enable a critical capability within the enterprise for multiscale material research at arl!

Thanks to everyone!
VI. Publications

- Books

- Journal Papers
  - “A direct tridiagonal solver based on Givens rotations for GPU architectures”, I. Venetis, A. Kouris, A. Sobczyk, E. Gallopoulos, and A. H. Sameh, Parallel Computing, 2015, pp. 1-16 (in press). (my co-authors forgot to include my ARO grant in the acknowledgement)

VII. Conclusion

The development of reliable high-quality software containing the above solvers has been an important goal to enable the realization of various simulations in computational mechanics in much shorter times. Codes for the above solvers are readily available. We plan to complete the documentation and user manuals of two codes for PSPIKE in Fall 2015 – one in which the balance system is not formed explicitly and solved using a preconditioned iterative scheme aimed at large-scale parallel architecture, and another aimed at cluster of few multicore nodes in which the balance system is formed explicitly and solved directly. *The TraceMIN and TraceMIN-Davidson eigensolvers have already been adopted by the Trilinos project and the complete documentation and user manuals of these two codes will be available this coming Fall semester.*

VIII. References


[AbBG07]

[ABGS04]

[Anas15]
Anasazi examples.

[Poli09]
E. Polizzi. “Density‐matrix‐based algorithms for solving eigenvalue problems”. 

[Poli14]
Peter Tang and E. Polizzi. “FEAST as a subspace iteration eigensolver accelerated by 
approximate spectral projection”. *SIAM Journal on Matrix Analysis and Applications*, 

[SLEP14]
Report DSIC-II/24/02 – Revision 3.5, D. Sistemes Informatics i Computacio, Universitat 