Comparative Study of Sparse Matrix Storage Format in the Finite Element Analysis of Thermal-Structure Coupling Problems

Abul Mukid Mohammad Mukaddes[1], Ryuji Shioya[2] and Masao Ogino[3]

[1]Department of Industrial and Production Engineering
Shahjalal University of Science and Technology, Bangladesh
Email: mukaddes1975@gmail.com
[2]Faculty of Information Sciences and Arts
Toyo University, Japan
Email: shioya@toyo.jp
[3]Information Technology Center, Nagoya University, Japan
Email: masao.ogino@cc.nagoya–u.ac.jp

Corresponding author: mukaddes1975@gmail.com

Abstract

The aim of this research is to compare different sparse matrix storages schemes in the finite element analysis of thermal-structure coupling problems. Thermal-structure coupling approach has been developed using the ADVENTURE System. The approach relies upon the existing module ADVENTURE Thermal, which handles the heat conduction problems to have the temperature distribution in the solid model and a module named ADVENTURE Solid which takes care of the stress analysis. Quite commonly, the matrix that participates in the finite element computation of thermal and structural problem is sparse. The present adventure modules which are based on the domain decomposition method use the sparse matrix-vector multiplication (SpMxV) as their basic operation. Sparse matrix by definition, are populated primarily with zeroes and thus special storage formats are used to enable efficient storage and computational operations. These representations usually store the non-zero values of the matrix with additional indexing information about the position of these values. A memory efficient storage format is proposed in this research. In the proposed technique, the inherent block sizes present in the sparse matrix are exploited to reduce the memory requirement as well as computation time. A SpMxV library has been developed that could be used both thermal and structural problems. Impacts of sparse matrix storages formats on the total execution time of the solver are evaluated. A 3D blast furnace cooling stave is analysed efficiently in terms of computation time and memory using the developed approach.

Keywords: thermal-structure coupling; sparse matrix; indexing; cooling stave; ADVENTURE System

Introduction

With the advent of high performance computer, efficient modules for the finite element analysis (FEA) to solve large scale problems are the present demand of FEM users. The ADVENTURE [adventure] is open source CAE software that has been developed for large scale analysis and design of computational mechanics system. This software is able to analyze three-dimensional (3-D) finite element models of arbitrary shape with 10-100 million degrees of freedom (dof). We have been developing the ADVENTURE system for the future high performance computer as a memory and time efficient FEA module. In this research a thermal-structure coupling system has been developed using two of ADVENTURE modules, ADVENTURE_Thermal and ADVENTURE_Solid. The

developed system could be used to analyze heat transfer problems that have complex geometries for the temperature distribution. The predicted temperature combined with the applied external load is then used to compute the deformation and thermal stresses of the model. Both thermal and solid modules use the parallel finite element method called domain decomposition method to solve problems in parallel computers.

The performance of the domain decomposition method is dominated by the sparse matrix-vector multiplication (SpMxV), $y \rightarrow y + Ax$ where, A is a sparse matrix and x, y are dense vectors [Mukaddes (2014)]. The method is also influenced by the preconditioning techniques [Ogino (2011)]. Sparse matrix by definition, are populated primarily with zeroes and thus special storage schemes are used to enable efficient storage and computational operations. These representations usually store the non-zero elements of the matrix with additional indexing information about the position of these values. A variety of compressed sparse row (CSR) format is used to store and manipulate the sparse matrix for the thermal problem [Mukaddes (2014)]. The matrix in the structural problems has inherent block shape. In order to reduce the memory requirement, exploiting the block shape of the matrix could be a beneficial choice. In this research, a diagonal block compressed sparse row (DBCSR) format for the structural problem are proposed and compared with other formats. Here instead of entire rows or columns of a matrix, block algorithms operate sub-blocks or data blocks.

The proposed storage formats are evaluated considering two models: High Temperature Test Reactor (HTTR) and 3D cooling stave [Kumar et al. (2012), Lijun et al. (2006)]. The numerical results obtained are presented and discussed.

2. Thermal-Solid coupling analysis

The developed system is conducted to predict temperature distribution in solid models and then to investigate the thermal expansion or deformation due to the temperature change. Analysis steps are as follows.

1) Read the input data for the heat conductive analysis and decompose the model by ADVENTURE_Metis.

2) Analyze heat conduction problems using ADVENTURE_Thermal.

3) Gather temperature of all nodes of the model from outputs of heat conduction problems.

4) Read temperature of all nodes and other input data for structural analysis and then decompose the model by ADVENTURE_Metis again.

Figure 1 shows the flow chart of thermal-solid coupling analysis with the developed system. The name of the ADVENTURE module used in each analysis is shown in parentheses.

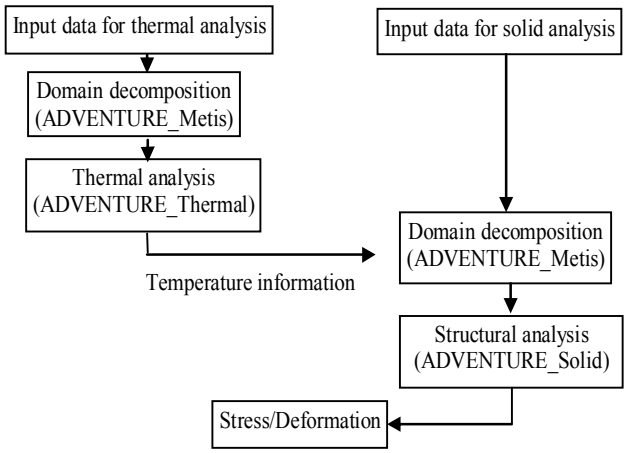


Figure 1 Flow chart of thermal-solid coupling system.

3. Sparse storage formats and their implementations

Sparse matrix storage schemes are described in this section. Consider the lower part of a typical symmetric matrix A. This matrix can be stored using different storage schemes. Several sparse matrix storage schemes are studied in this research and evaluated in both thermal and structural analysis. They are Compressed Sparse Row (CSR), Compressed Sparse Column (CSC), Variable Block Compressed Sparse Row (VBCSR) and Diagonal Block Compressed Sparse Row (DBCSR). The DBCSR are proposed in this research and compared with other storage format. For simplicity Skyline storage, CSR storage and DBCSR storage formats are explained for the example matrix A.

$$
\begin{bmatrix}
1 & & & & & & & & & & & \\
2 & 3 & & & & & & & & & & \\
4 & 5 & 6 & & & & & & & & & \\
7 & 8 & 9 & 10 & & & & & & & & \\
11 & 12 & 13 & 14 & 15 & & & & & & & \\
16 & 17 & 18 & 19 & 20 & 21 & & & & & & \\
 & & & 22 & 23 & 24 & 25 & & & & & \\
 & & & 26 & 27 & 28 & 29 & 30 & & & & \\
 & & & 31 & 32 & 33 & 34 & 35 & 36 & & & \\
37 & 38 & 39 & & & & & & & 40 & & \\
41 & 42 & 43 & & & & & & & 44 & 45 & \\
46 & 47 & 48 & & & & & & & 49 & 50 & 51
\end{bmatrix}
$$

Matrix A

4.1 Skyline or Variable Band (SKY)

The Skyline representation becomes popular for direct solvers especially when pivoting is not necessary. This is the most common matrix storage format. The matrix elements are

stored using three single arrays: *data, row_index and row_pattern.* The array *data* stores the elements of the matrix *A* row by row, *row_index* contains column number of first element of each row and *row_pattern* array points to the start of every row.

Table 1: Skyline format

data:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	**0**	**0**	**0**	**0**	**0**	**0**	40	41	
42	43	**0**	**0**	**0**	**0**	**0**	**0**	44	45	46	47	48	**0**	**0**	**0**	**0**	**0**	**0**	49	50	51		

row_index:

1	1	1	1	1	1	3	3	3	1	1	1

row_pattern:

1	2	4	7	11	16	22	26	31	37	47	58	69

4.2 Compressed Sparse Row (CSR)

Compressed sparse row format [Saad (1994)] is popular and the most general purpose storage format for the sparse matrix. The elements are stored using three arrays: *data*, *row_pattern* and *col_index*. The single array *data* of length number of nonzero (*nnz)* contains the non-zero elements of *A* row by row, *col_index* of length *nnz* contains the column number which correspond to the non-zero elements in the array *data*. The integer vector *row_pattern* of length *nrow+1* contains the pointers to the beginning of each row in the array *data* and *col_index*. With the *row_pattern* array we can easily compute the number of non-zero elements in the i^{th} row as *row_pattern*[i+1] - *row_pattern*[i]. The last element of *row_pattern* is *nnz.* The CSR representation of the symmetric matrix *A*:

Table -2 CSR format

data:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41							
42	43	44	45	46	47	48	49	50	51														

col_index:

1	1	2	1	2	3	1	2	3	4	1	2	3	4	5	1	2	3	4	5	6	4	5	6
7	4	5	6	7	8	4	5	6	7	8	9	1	2	3	10	1							
2	3	10	11	1	2	3	10	11	12														

row_pattern:

1	2	4	7	11	16	22	26	31	37	41	46	51

4.3 Diagonal Block Compressed Sparse Row (DBCSR)

The DBCSR exploit the 3x3 block shape of the matrix. In this format the diagonal block can be stored separately in a *diag* array which does not require the indexing. The off-

diagonal elements are stored in the data array. *index_brow* represents the position of first element of the first block in the data array. The last element of the *index_brow* is the number of elements in the off-diagonal blocks. *Index_bcol* stores the column number of the first elements of each off-diagonal blocks. Such indexing reduces the working sets as well as memory requirements.

Table-3 DBCSR

diag:

1	2	3	4	5	6	10	14	15	19	20	21	25	29	30	34	35	36	40	44	45	49	50	51

data:

7	8	9	11	12	13	16	17	18	22	23	24	26	27	28	31	32	33	37	38	39	41	42	43	4

index_brow:

1	10	19	27

index_bcol:

1	4	1

4.4 Working sets of sparse matrix storage formats

The skyline format takes more memory than others as it needs to store some unnecessary zero elements. The CSR requires less memory than skyline, as it does not store zero elements of the matrix. DBCSR reduces the memory requirement of the indexing part of CSR. The working sets of three storage formats are given below.

Table-4 Working sets

SKY	CSR	DBCSR
elm + 4(row+1+row)	8nnz + 4(row+1+nz)	8nnz + 4(brow +1 + brow)

elm: num. of matrix elements; nnz: num. of non-zero; row: num. of rows; brow: num. of block rows

5 Numerical results and discussions

Evaluation of sparse matrix storage schemes:

A large scale High Temperature Test Reactor (HTTR) model is used to evaluate the sparse matrix storage schemes. The computational environment is Intel Corei7-960 (3.20GHz/L2 256KB/L3 8MB/QuadCore). The DBCSR are evaluated and compared with other sparse matrix storage fromats. Using the CSR type storage format computation time is reduced to around 50% and required memory is reduced to around 45% comapared to the skyline storage format. Again, DBCSR shows better performance over the CSR format. It reduces 21% computation time and 15% required memory compared to the CSR format.

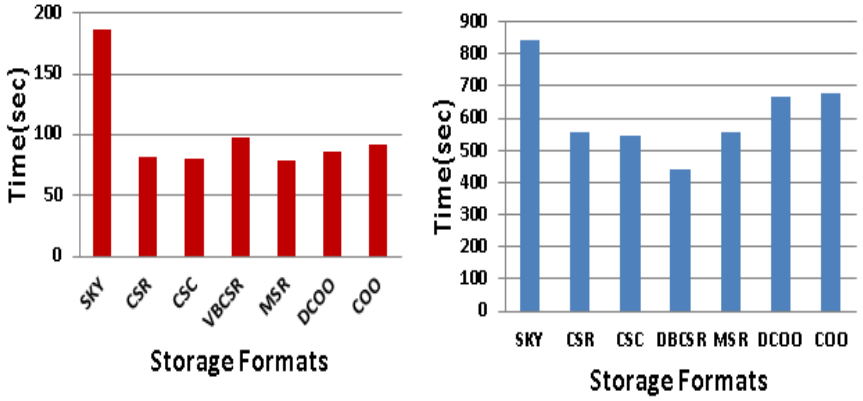


Figure-2 Computation time for thermal (left) and sturcture (right)

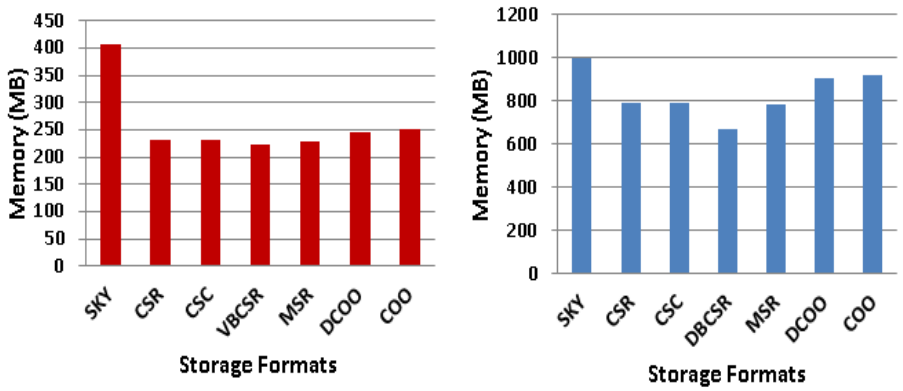


Figure-3 Required Memory for thermal (left) and structural (Right)

Cooling stave analysis:

In the present paper, a cooling stave of a blast furnace has been modeled and analyzed using the developed system. Parts name are shown in figure 4a. For the finite element analysis, the 3D geometry of the cooling stave is made using the commercial CAD software, Meshman. After that the model is exported to .iges file. Then the file is imported to the ADVENTURE Systems. For simplicity, a part model of a cooling stave (figure 4b) is analysed. Results of full model are given later. The design parameters and material properties are taken from [Kumar et al. (2012)]. The boundary conditions for thermal and solid are set up as follows:

- Air temperature is 323 K, water temperature is 303 K.

- Heat convection coefficients: between furnace shell and atmosphere-12 W/(m^2 K), between water and inner sides of the furnace shell -8000 W/(m^2K).
- The upper and lower surface are fixed.

The cooling channels are shown in figure 4b, where convection boundary conditions are considered. The figure 3a shows the temperature distribution after the thermal analysis using the ADVENTURE_Thermal. The ADVENTURE_PostTool is used to visualize the temperature, displacement and stress. The temperature information is used in ADVENTURE_Solid as loads to measure the thermal expansion and stresses. The thermal expansion (X 100) in the y direction is shown figure 3d and corresponding nodal equivalent stress is shown in figure 4a. Numerical results depict that the stress intensity on the cold surfaces is mainly affected by the cooling water and much higher on the hot sides.

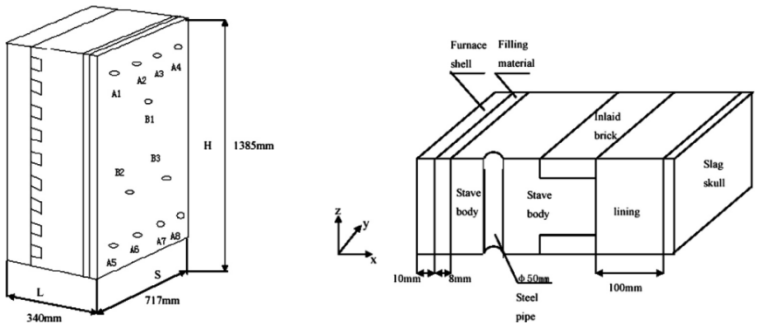


Figure-4 Cooling stave full (left) and part (right)

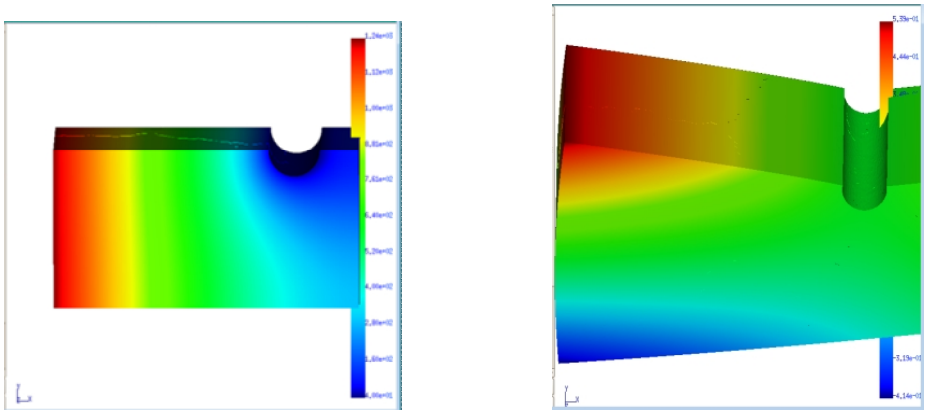
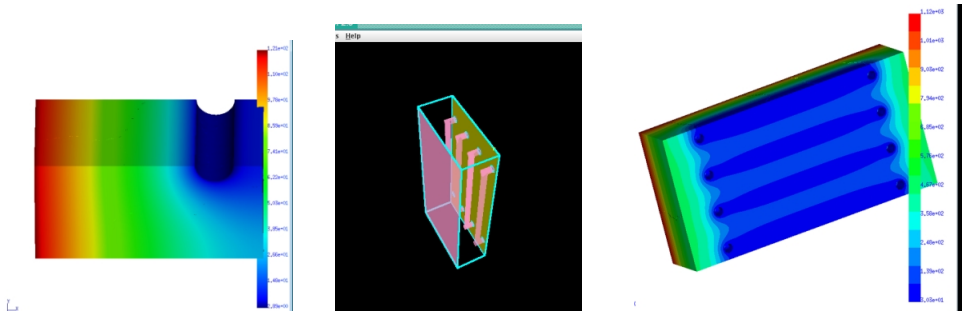


Figure 5: Temperature distribution (left) and expansion (x100, right)



(a) (b) (c)

Figure 5: (a) Equivalent stress (b) Cooling channel (c) Temperature distribution of full model

Finally, a full model of cooling stave is modelled and analysed as shown in figure 5b. After thermal analysis of the full model it is found that, the temperature on the top and bottom surfaces are higher than other regions and maximum values are on the hot sides (figure 5c).

6. Conclusion

In this research, the developed thermal-solid coupling system is successfully implemented to analyze the cooling stave of a blast furnace. The intensity of stress is reduced due to the cooling system. The computational cost of the developed system is improved dramatically by employing several sparse matrix storage formats. Several sparse matrix storage formats are implemented and compared. DBCSR type storage format shows better performance compare to the previous Skyline and CSR storage format. Future research is to compare different blocking strategies of DBCSR storage format.

Acknowledgements

This research is financially supported by one of Japan Science and Technology (JST), CREST projects named "Development of Numerical Library Based on Hierarchical Domain Decomposition for Postpeta Scale Simulation".

References

http://adventure.sys.t.u-tokyo.ac.jp/

Kumar, A., Bansal, S. N. and Chandraker, R., (2012), Computational modeling of blast furnace cooling stave based on heat transfer analysis, *Material Physics and Mechanics*, 15, 46-65.

Mukaddes, A M M., Ogino, M. and Shioya, R. (2014), Performance study of domain decomposition method with sparse matrix storage schemes in modern supercomputer, *International Journal of Computational Method (in press*).

Lijun, W, Weiguo, Z. Huier, C. Yunlong, S., Xiaojing Li and Canyang, S. (2006), The study of cooling channel optimization in blast furnace cast steel stave based on heat transfer analysis. *International Journal of Advanced Manufacturing Technology*, 29, 64-69.

Ogino, M., Shioya, R. and Kanayama, H. (2008) An inexact balancing preconditioner for large-scale structural analysis, *Journal of Computational Science and Technology*, 2- 1, 150-161.

Saad, Y., (1994) SPARSEKIT: A basic tool kit for sparse matrix computations. Technical report, Computer Science Department, University of Minnesota, Minneapolis, MN 55455, Version 2.

Shahnaz, R. and Usman, Anila (2011), Blocked-based sparse-matrix vector multiplication on distributed memory computer, *The International Arab Journal of Information Technology,* 8- 2, 130-136.