Hindawi
Science and Technology of Nuclear Installations
Volume 2017, Article ID 7072197, 10 pages
https://doi.org/10.1155/2017/7072197

Research Article

Development and Application of a New High-Efficiency Sparse Linear System Solver in the Thermal-Hydraulic System Analysis Code

Li Ge,[1] Wei Liu,[2] and Jianqiang Shan[1]

[1]*Xi'an Jiaotong University, 28 W. Xianning Rd, Xi'an, China*
[2]*Science and Technology on Reactor System Design Technology Laboratory, Nuclear Power Institute of China, Chengdu, China*

Correspondence should be addressed to Jianqiang Shan; jqshan@mail.xjtu.edu.cn

Received 7 April 2017; Revised 3 July 2017; Accepted 22 August 2017; Published 19 September 2017

Academic Editor: Manmohan Pandey

This paper presents a faster solver named NRLU (Node Reordering Lower Upper) factorization solver to improve the solution speed for the pressure equations, which are formed by RELAP5/MOD3.3. The NRLU solver uses the oriented graph method and minimal fill-ins rule to reorder the structure of the nonsymmetry sparse pressure matrix. It solves the pressure matrix by LU factorization. Then the solver is embedded into the large scale advanced thermal-hydraulic system analysis program RELAP5/MOD3.3. The comparisons of the original solver and the NRLU solver show that the NRLU solver is faster than the original solver in RELAP5/MOD3.3, and the rate enhancement can be 44.44%. The results also show that the NRLU solver can reduce the number of fill-ins effectively. This can improve the calculation speed.

1. Introduction

After entering the 21st century, the requirement of the safety and economy performance for nuclear power plant has been raised. Researchers proposed different novel concepts of advanced nuclear power systems, such as the small module reactor and the generation IV reactors. Accurate and fast simulation of these systems' detailed behavior under transient conditions has become the key issue.

For simulating the large or complex systems using a personal computer, the existing nuclear system analysis code would spend several days to get the results. And it will take more time to modify or design a modeling system repeatedly. The future nuclear system power programs require the more fine or complex modeling, which could cause a long calculation time. So it is necessary to develop a more efficient numerical technique to improve the calculation efficiency of the nuclear system analysis code. On the other hand, the faster calculation speed is important to achieve the real-time requirement for the nuclear power plant simulators.

To improve the calculation speed, the nuclear system analysis code must be equipped with a faster unsymmetrical sparse matrix solver for the system equations, which could cost nearly half the time during the total calculation. However, the research in improving the matrix solver for nuclear system analysis code is rare. At present, the existing nuclear system analysis codes are all equipped with the matrix solvers developed at least ten years ago. RELAP5/MOD3.3 code is equipped with the matrix solver developed by Curtis and Reid in 1971. CATHARE code uses the normal Newton iteration method. These matrix solvers are universal and are not specially developed for the nuclear system analysis code. Therefore, a more efficient matrix solver for the nuclear system analysis code needs to be developed.

Over the last several decades, computer speed and memory have increased dramatically. Many methods have been developed for solving the large unsymmetric, sparse linear matrices. Standard direct methods based on the Gaussian elimination require more work than iterative schemes. As a result, such system matrices are typically solved by iterative methods with fast algorithms, such as GMRES [1] and Bi-CGSTAB [2]. For the integral equations of classical physics, iterative scheme has led to some of the fastest solvers known today. However, iterative methods still have some significant

disadvantages compared with direct methods: (1) The iteration number for an iterative solver is highly sensitive to the conditioning of the system matrix. (2) Iterative methods cannot solve a linear system governed by a fixed matrix with multiple right hand sides. While direct methods can be applied to solve this kind of system at a much lower cost [3], iterative methods are very useful for a large sparse system. But for small or medium sparse matrix, iterative methods cannot do better than direct methods. In the thermal-hydraulic system analysis code, the matrices are mostly small or medium. So, to solve the system matrix of the thermal-hydraulic system analysis code, direct methods are better than interactive methods.

Standard direct methods based on Gaussian elimination require the computation work of $o(N^3)$, where N is the system size. This means that this method will become infeasible while N is increasing. Therefore, the coefficient matrix needs to be reordered to reduce the new nonzero elements storage and floating point operation. So far, the matrix reorder algorithms can be classified into two groups by their optimization goals. One group is to reduce the bandwidth and profile of matrix, such as the RCM algorithm, the GPS algorithm [4], the Schonauer algorithm, the King algorithm, and the Sloan algorithm. The other is to reduce the fill-ins [5], such as the MD algorithm [6], the AMD algorithm [7], and the Nested Dissection algorithm [8].

The original matrix solver in RELAP5/MOD 3.3 is a direct method based on Markowitz algorithm. The concept of the Markowitz algorithm [9] is choosing the element with minimum Markowitz factor as the pivot at each step of the factorization. If the number of nonzero elements in row i is r and the number of nonzero elements in column j is c, the Markowitz factor of element (i, j) is as follows:

$$\text{Markowitz factor} = (r-1)(c-1). \tag{1}$$

The solution process in RELAP5/MOD3.3 is performed in three steps:

(1) First, symbolic factorization to determine the maximum memory requirements using Markowitz ordering

(2) Second, numeric factorization using Markowitz ordering with threshold pivoting

(3) Third, solution of the LU systems.

A faster direct solver with the minimal fill-ins preprocessing will be presented in this paper, which is developed to solve sparse linear systems resulting from the application of the discretized two phase hydrodynamics equations for nuclear reactor transient problems. This solver is a direct method based on lower upper factorization with a structure reordering processing. Before numerical LU factorization, a nodes reordering is processed during symbolic factorization according to its data structure to reduce the fill-ins. This solver, named NRLU, will improve the calculation speed compared with the standard direct method and hold the advantages of the direct methods. The NRLU solver is then embedded into the large scale advanced thermal-hydraulic system analysis program RELAP5/MOD3.3. Some transients for reactor power plant systems were analyzed using RELAP5/MOD3.3 program with its original matrix solver and the NRLU solver, respectively.

2. The NRLU Solver Algorithm

2.1. Pressure Matrix in RELAP5/MOD3.3. In RELAP5/MOD3.3, the equations of the noncondensable density equation, the vapor/gas energy equation, the liquid energy equation, the difference density equation, and the sum density equation are firstly eliminated to obtain an equation that involves only the unknown variables $(P_L^{n+1} - P_L^n)$, $v_{g,j+1}^{n+1}$, $v_{g,j}^{n+1}$, $v_{f,j+1}^{n+1}$, and $v_{f,j}^{n+1}$. Then substituting the velocity equations into this equation, this will result in a single equation involving only the pressure parameter. This is done for each volume, giving rise to an $N \times N$ system of linear equations for the new pressures in a system containing N control volumes.

The linear equation is formed as

$$Ax = b. \tag{2}$$

The coefficient matrix A will be larger as the number of the control volumes N increases. And the sparsity of A will increase because the problems of the nuclear system are almost 1D or 2D in RELAP5/MOD3.3.

During the long-term transient analyses of the nuclear systems, (2) will be solved more than a million times. So, every unnecessary nonzero operation and nonzero storage need to be avoided. Therefore the coefficient matrix A needs to be reordered before factoring (1) to reduce the operation times and element storages during the matrix factorization. So far, there are two basic principles for reorder: (1) minimal fill-ins and (2) minimal long operations. In this paper, the minimal fill-ins will be the basic principle to renumber the nodes to improve the calculation speed of the sparse linear matrix.

Because of the existence of the time dependent volumes, the pressure coefficient matrix A is almost nonsymmetric. At present, one of the popular methods is to regard the nonsymmetric structure as symmetric. However, this method needs more computer memory and wastes calculation time. In this paper, a method aimed at solving the nonsymmetric pressure matrix formed from the RELAP5/MOD3.3 code is proposed based on the oriented graph method. The key technology of this method is the node ordering optimization for the control volumes of the simulation system [10].

2.2. Matrix Represented by Oriented Graph. Graph theory is a fundamental tool in sparse matrix techniques. The nonzero pattern and the relationship between the nonzero elements can be represented by the oriented graph.

Assuming that the order of the coefficient matrix A is N and the elements of main diagonal are nonzero, there are n nodes to represent the row numbers (or column numbers) of the matrix A. The nonzero elements of the matrix A can be represented by the directed line between two nodes. For example, for $a_{ij} \neq 0$ $(i \neq j)$, a directed line can be used from j node to i node to represent the nonzero element a_{ij} in row

i and column j. The nonzero elements of the main diagonal can be represented by each node itself [11].

For example, the oriented graph for the nonsymmetric matrix (3) is shown in Figure 1.

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 & a_{25} \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & 0 \\ 0 & a_{52} & 0 & 0 & a_{55} \end{pmatrix}. \tag{3}$$

If there are two directed lines between two nodes, these two directed lines can be replaced by an undirected line. Figure 1 can be simplified as Figure 2. Obviously, the graph is undirected if matrix A has symmetric structure.

2.3. Elimination Rules Using Oriented Graph. In order to introduce the reordering method using oriented graph briefly, eliminating one node from the oriented graph is discussed firstly. Based on the Gaussian elimination method, the coefficient matrix A can eliminate unknown x_i from the ith equation by substituting (4) into (2) [12]

$$x_i = -\frac{1}{a_{ii}} \left[\sum_{k=1}^{n} a_{ik} x_k - b_i \right] \quad (k \neq i). \tag{4}$$

For the oriented graph, the ith node and the lines connected to this node need to be eliminated. The rules of eliminating one node from the oriented graph are as follows:

(1) If the ith node has only one line, eliminating it and its line will not produce new line.

(2) If the ith node has two lines and the two connected nodes are j and k, (a) if there is a directed path from j to k via i, eliminating the ith node and its lines will produce a directed line from j to k, (b) if there is an undirected path from j to k via i, it will produce an undirected line between j and k, and (c) if there is no path, it will not produce a line.

(3) If the ith node has more than two lines, then every two lines need to be processed as rule (2).

(4) If the new line between j and k did not exist before elimination, the new line means a fill-in.

Figure 3 shows an example of eliminating the ith node from the oriented graph, while Figure 3(b) shows the production of three new elements a_{lj}, a_{mk}, and a_{mj}.

2.4. The Minimal Fill-Ins Reordering Algorithm. As we know, the LU direct factorization method is the deformation algorithm of Gaussian elimination method. So the rules can also apply to the LU factorization method. According to the minimal fill-ins principle, the node with no fill-in will be processed as first priority at each step of the factorization. Next, the node with minimal fill-ins will be considered.

Taking (3) as an example, Figure 4 shows the whole elimination process using oriented graph method. It shows that the new order is 1, 2, 5, 3, and 4. There is only one new nonzero during the whole process: $a_{45}^{(2)}$ at the second step. Equation (5) shows that the new matrix forms at each step after permutation. The symbol "X" represents the fill-in after elimination according to Figure 4. If (3) is factorized directly, it would produce two new nonzeros. So, the minimal fill-ins algorithm can reduce the new nonzeros during elimination effectively. Because all nonzeros need to be stored in the computer memory during solving the matrices, reducing the new nonzeros can reduce the requirement of the computer memory. As the order of the matrix becomes larger, the efficiency of reducing the memory by using the minimal fill-ins algorithm will be higher and higher.

New matrices during elimination:

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 & a_{25} \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & 0 \\ 0 & a_{52} & 0 & 0 & a_{55} \end{pmatrix} \longrightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 & a_{25} \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & 0 \\ 0 & a_{52} & 0 & 0 & a_{55} \end{pmatrix} \longrightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & 0 & a_{43} & a_{44} & \mathrm{X} \\ 0 & 0 & 0 & 0 & a_{55} \end{pmatrix} \longrightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \longrightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{5}$$

The matrix reordering algorithm using oriented graph method is shown in Figure 5. The order of the elements in source vector b must be changed with the order of nodes in the coefficient matrix A.

This reordering method with its numerical factorization and back substitution code is named NRLU. The NRLU

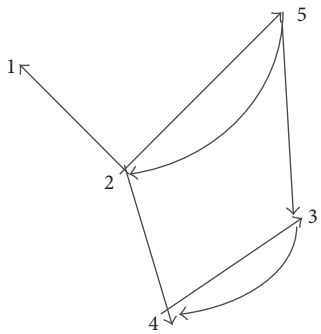

Figure 1: The oriented graph of matrix A.

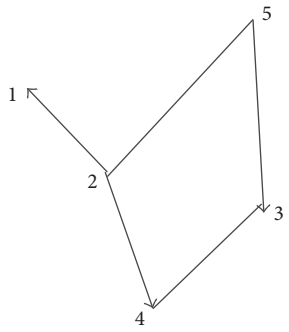

Figure 2: The simplified oriented graph of matrix A.

solver code will be implemented using the FORTRAN programming language and then embedded into the large scale advanced thermal-hydraulic system analysis program RELAP5/MOD3.3. Some problems of the reactor power plant systems are analyzed using RELAP5/MOD3.3 program with its original matrix solver and the NRLU solver, respectively, in the next section.

3. The Sensitivity Analysis of Node Number

The physical system needs to be divided into control volumes which are modeled as nodes using the lumped parameter method in RELAP5 code. For nuclear power system analysis codes, increasing the node number can improve the simulation accuracy. However, increasing the number of nodes in the simulation system will increase the order of the system matrix. The calculation speed of the system matrix is seriously affected. On the other hand, the performance of the matrix solver algorithm is also closely related to the order of the matrix. Therefore, it is necessary to analyze the performances of the NRLU solver under the conditions of different node number.

It is necessary to know that the system matrix formation is only related to the number of control volumes and the linking information between them, and it has nothing to do with the geometry size of the system. The primary side of the normal PWR system with three loops can be simulated simply as a system with 4 branch loops. Its nodalization modeled by RELAP5 is shown in Figure 6. The branch loop with five parallel pipes is used to simulate the reactor vessel with

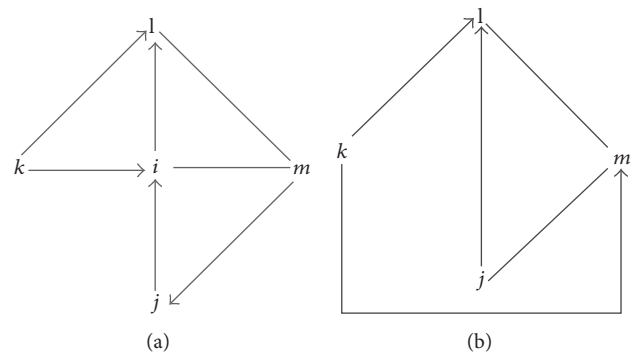

Figure 3: The result of eliminating the ith node.

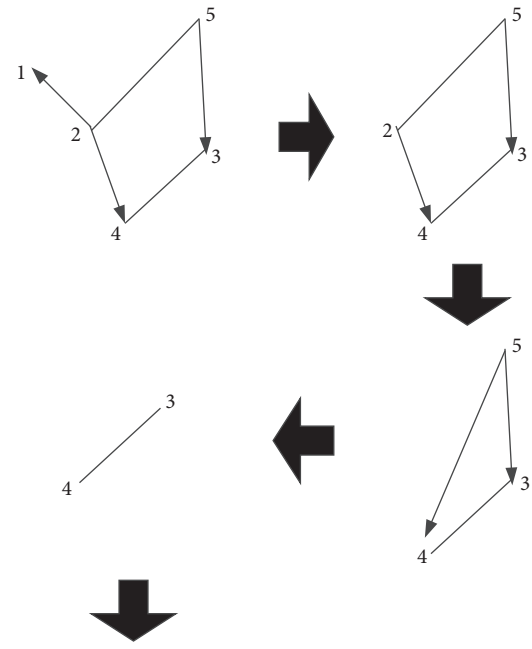

Figure 4: Elimination process using oriented graph method.

various core cooling channels (shown in Figure 7(a)), and the other three branch loops are used to simulate the three cooling loops connected to the vessel (shown in Figure 7(b)).

To simplify simulation, all pipes have the same number n of the control volumes, and the total number of the control volumes or the order of the simulation system's pressure matrix is nz = 12×n+9. Comparing the running performance of solving the pressure matrix with the NRLU with the original matrix solver is made under the conditions of n = 5, 10, 20, 40, and 80. The pressure matrix is produced by RELAP5 according to the nodalization in Figure 6. The time step is 0.1 s, and the end time is 5000 s.

Figures 8 and 9 show the calculation time and memory requirements of the NRLU solver and the original matrix solver in RELAP5/MOD3.3 with the different node number. The results show that the calculation time and memory requirements increase with the node number using both matrix solvers. The NRLU solver calculated three times faster than the original solver. And the memory requirements of the

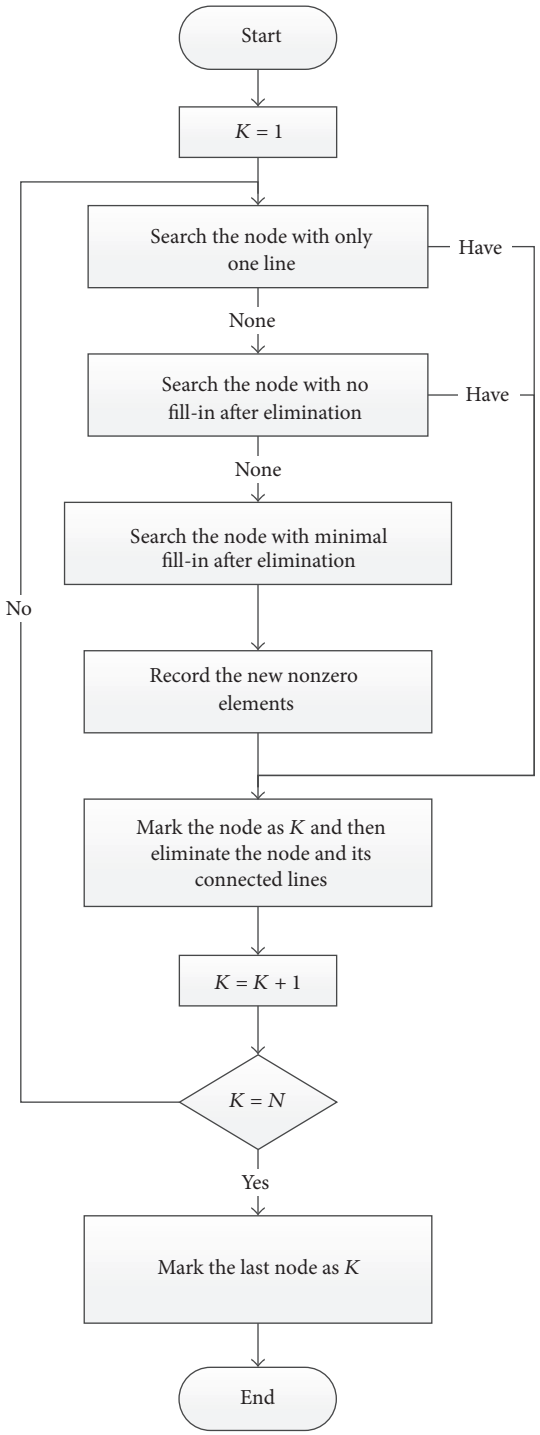

Figure 5: The matrix reordering algorithm.

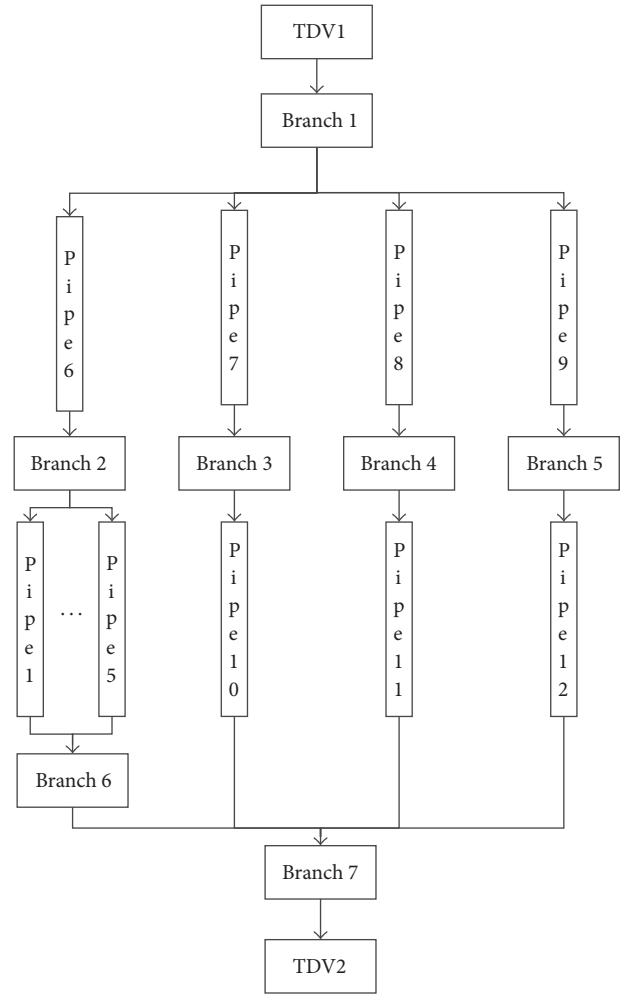

Figure 6: Nodalization of the 4-branch system.

NRLU solver are less than half of the memory requirements of the original solver. In general, the performance of the NRLU solver is much better than the original solver in this system.

4. Numerical Experiments

We take two problems of the reactor power plant systems as examples to evaluate the performance of NRLU solver compared with the original solver in RELAP5/MOD3.3. Figures 10 and 11 show the RELAP5/MOD3.3 nodalizations of the simple parallel pipes system and the small modular reactor system. These two systems are typical in nuclear power plant simulation using RELAP5. The simple parallel pipes system shows the typical secondary cycle in nuclear power plants, and the small modular reactor system shows the typical primary system in nuclear power plants.

The nonzero numbers nz and structures of the two system matrices are shown in Figures 12 and 13, respectively. In the first case, the order of the coefficient matrix is 14 and the nonzero number nz is 40. In the second case, the order of the coefficient matrix is 424 and the nonzero number is 1317.

Some comparisons of the original and the NRLU solver calculated results of case 1 and case 2 are provided in Figures 14–19. The parallel pipes system has a valve between control volumes 300 and 126, and the valve is opened between 2 and 500s. Figures 14–17 show the comparisons between the original solver and the NRLU solver calculated mass flows of pipes 125 and 126 and the inlet temperature and pressure of pipe 125, respectively, for case 1. Figures 17–19 show the

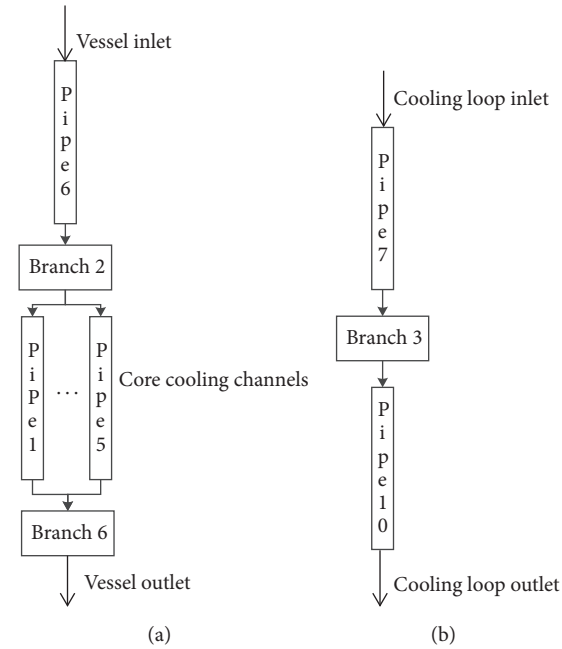

Figure 7: The branch loops for the vessel and cooling loop.

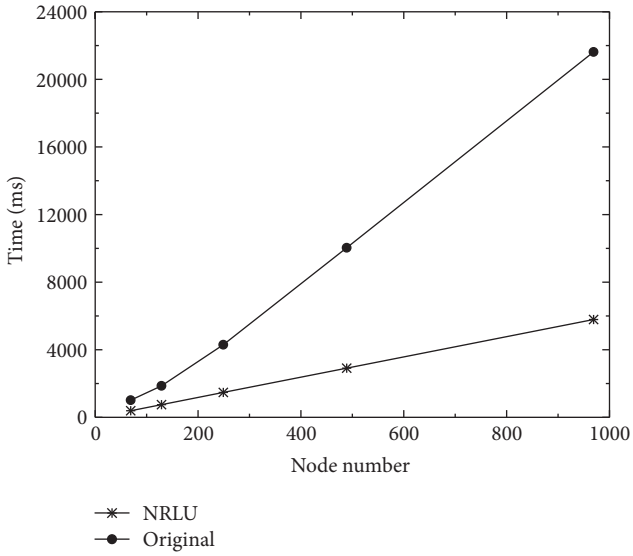

Figure 8: The calculation time of NRLU and original solver with node number.

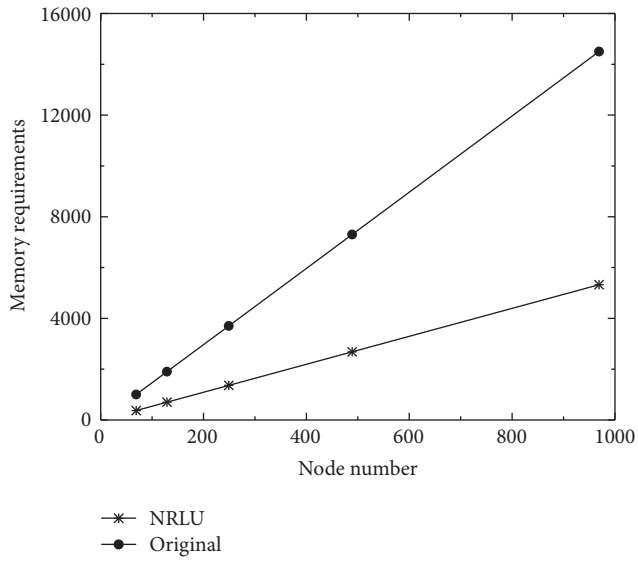

Figure 9: The memory requirements of NRLU and original solver with node number.

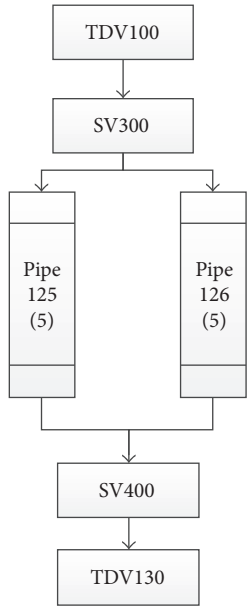

Figure 10: RELAP5 nodalization for the simple parallel pipes system.

comparisons between the original solver and the NRLU solver calculated mass flows of the core channels, inlet temperature of the core, and pressure of the core inlet and outlet, respectively, for case 2. The comparisons indicate that the results of cases 1 and 2 calculated by the original solver and the NRLU solver are almost the same. Therefore, the NRLU solver is feasible and valid.

The speed results of these two cases are shown in Table 1. The problem time for case 1 is 1.0×10^5 s, and for case 2 it is 300 s. The results show that the NRLU solver can reduce the fill-ins during LU factorization process effectively compared with the original matrix solver. Moreover, the matrix calculation speed of the NRLU solver is obviously higher than that of the original solver in RELAP5/MOD3.3. For case 1 the speed ratio is 1.5, and for case 2 the speed ratio is 1.8. The results indicate that effectively reducing the number of fill-ins can improve the solver speed of pressure matrix. For the more refined or complex systems taking several days, the time-saving will be in the order of days. Also, it is very useful to modify or design a modeling system repeatedly.

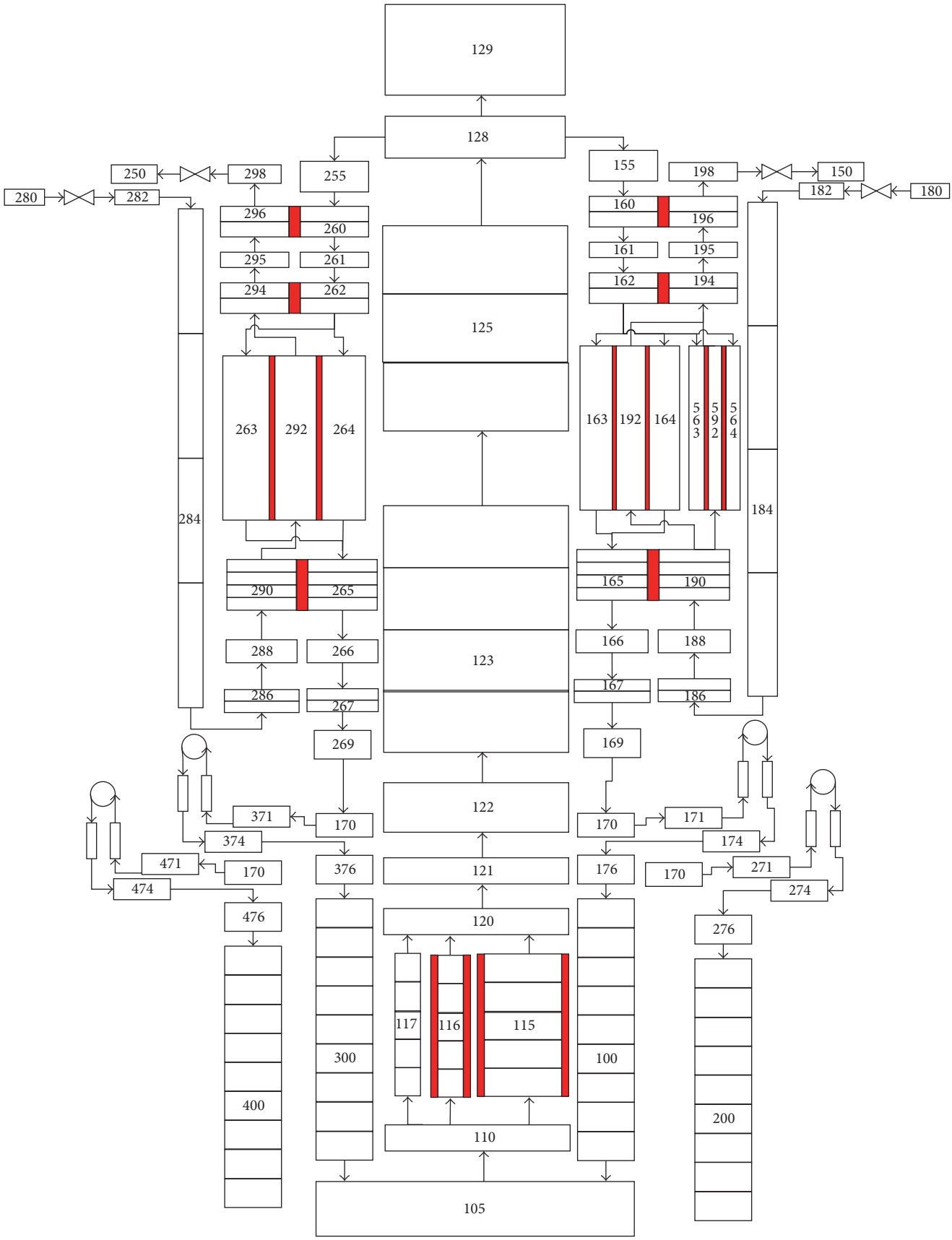

Figure 11: RELAP5 nodalization for the small modular power system.

Table 1: The comparisons of the two matrix solvers for cases 1 and 2.

Case number	1	2
Description	Simple parallel pipes	Small modular power plant
Order	14	424
Nonzero number	40	1317
Original matrix solver's fill-ins	48	1635
NRLU solver's fill-ins	18	642
Original matrix solver's matrix calculation time (s)	6	18
NRLU solver's matrix calculation time (s)	4	10
Speed ratio	1.5	1.8

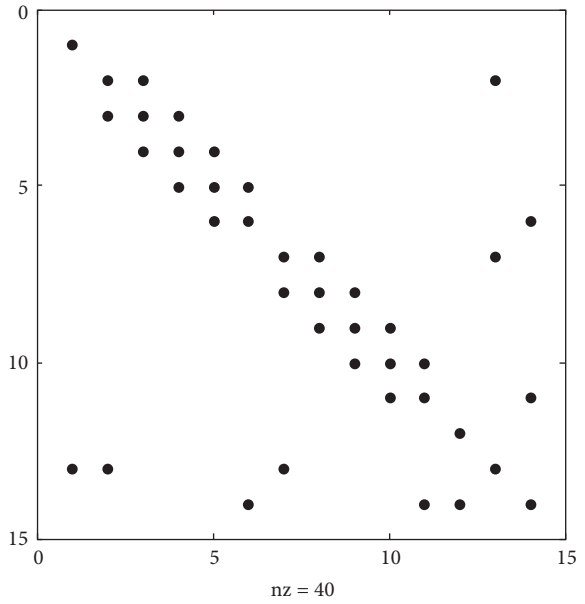

Figure 12: The nonzero structure of matrix A for case 1.

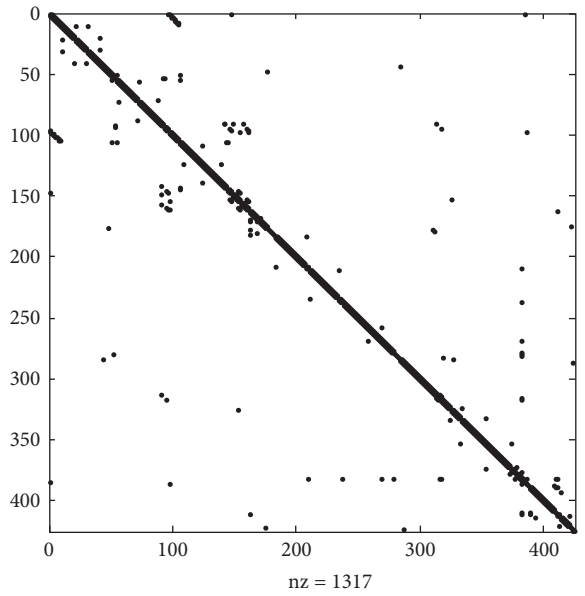

Figure 13: The nonzero structure of matrix A for case 2.

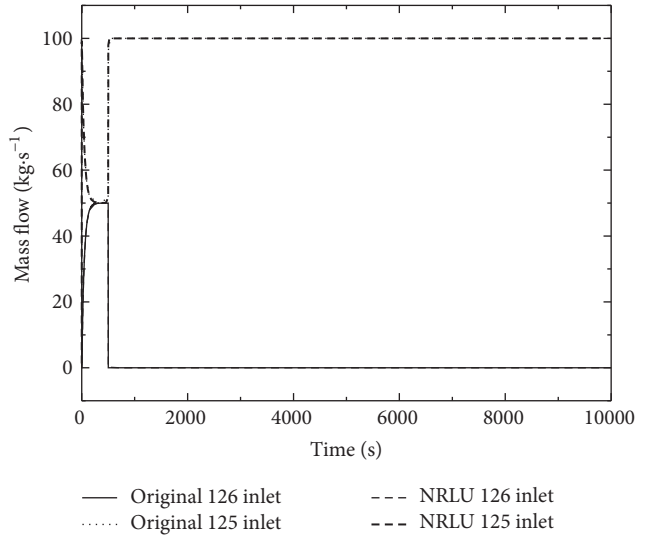

Figure 14: Original and NRLU calculated inlet mass flows of pipes 125 and 126 for case 1.

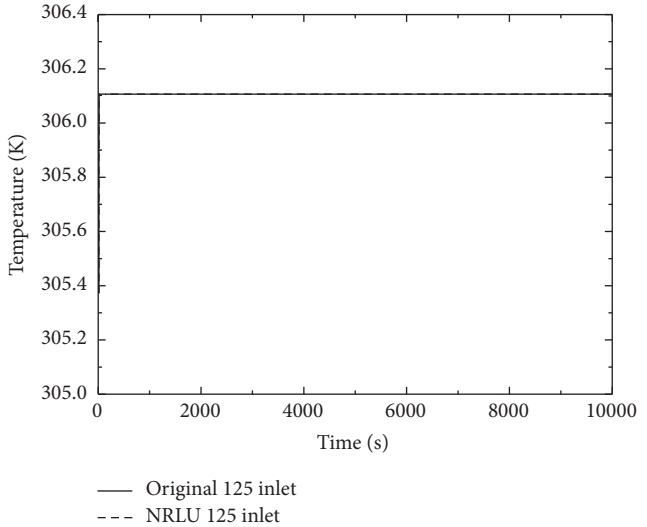

Figure 15: Original and NRLU calculated inlet temperature of pipe 125 for case 1.

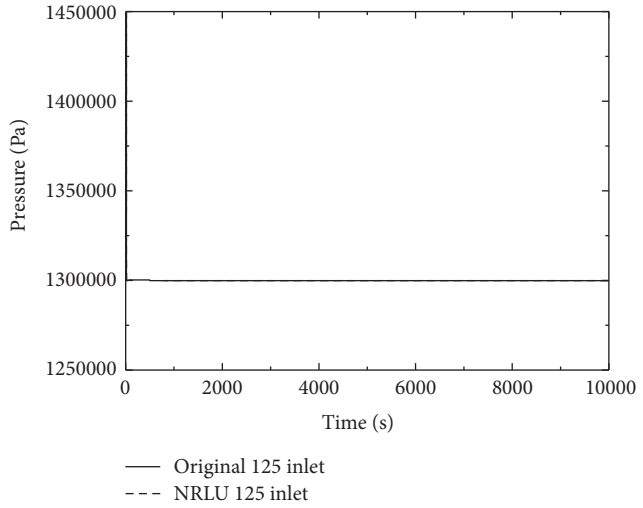

Figure 16: Original and NRLU calculated inlet pressure of pipe 125 for case 1.

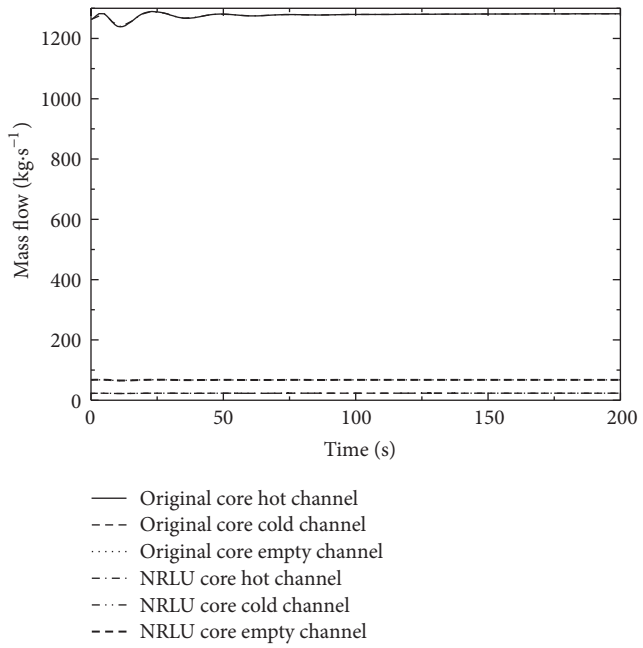

Figure 17: Original and NRLU calculated core channel mass flows for case 2.

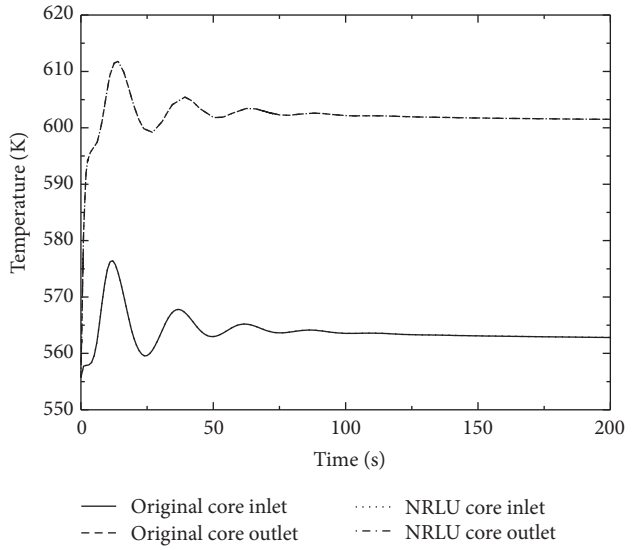

Figure 18: Original and NRLU calculated temperatures of core inlet and outlet for case 2.

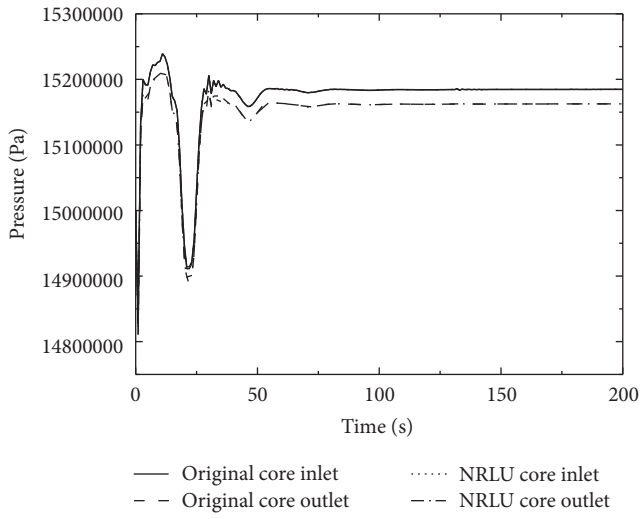

Figure 19: Original and NRLU calculated pressures of core inlet and outlet for case 2.

5. Conclusions

This paper presented a faster direct solver NRLU for solving the transient problems of the nuclear power plant system using the minimal fill-ins concept. After implementing the NRLU algorithm using the FORTRAN programming language and embedding it into RELAP5/MOD3.3, the matrix solver was tested using some sparse matrices formed from RELAP5 program. The calculation time and memory requirements during numerical factorization were tested compared with the original matrix solver in RELAP5/MOD3.3. The test results show that the NRLU solver can reduce the fill-ins during LU factorization process and the NRLU solver can achieve up to 1.8 speed ratios.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The study of this paper is financially supported by China Nuclear Power Simulation Technology Co. Ltd. that provided valuable comments and relevant information. The authors would like to express their heartily gratitude to its support and funding.

References

[1] Y. Saad, "A flexible inner-outer preconditioned GMRES algorithm," *SIAM Journal on Scientific Computing*, vol. 14, no. 2, pp. 461–469, 1993.

[2] H. A. van der Vorst, "BI-CGSTAB: a fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 2, pp. 631–644, 1992.

[3] K. L. Ho and L. Greengard, "A fast direct solver for structured linear systems by recursive skeletonization," *SIAM Journal on Scientific Computing*, vol. 34, no. 5, pp. A2507–A2532, 2012.

[4] N. E. Gibbs, W. G. Poole, and P. K. Stockmeyer, "A comparison of several bandwidth and profile reduction algorithms," *ACM Transactions on Mathematical Software (TOMS)*, vol. 2, no. 4, pp. 322–330, 1976.

[5] A. George and J. W. Liu, "Computer solution of large sparse positive definite systems," *Mathematics of Computation*, vol. 39, no. 159, 1981.

[6] P. R. Amestoy, T. A. Davis, and I. S. Duff, "Algorithm 837: AMD, an approximate minimum degree ordering algorithm," *ACM Transactions on Mathematical Software*, vol. 30, no. 3, pp. 381–388, 2004.

[7] A. George and J. W. Liu, "The evolution of the minimum degree ordering algorithm," *SIAM Review*, vol. 31, no. 1, pp. 1–19, 1989.

[8] J. R. Gilbert and R. E. Tarjan, "The analysis of a nested dissection algorithm," *Numerische Mathematik*, vol. 50, no. 4, pp. 377–404, 1987.

[9] P. R. Amestoy, X. S. Li, and S. Pralet, "Unsymmetric ordering using a constrained Markowitz scheme," *SIAM Journal on Matrix Analysis and Applications*, vol. 29, no. 1, pp. 302–327, 2006/07.

[10] L. Siefken, *SCDAP/RELAP5/MOD 3.3 Code Manual*, Division of Systems Technology, Office of Nuclear Regulatory Research, US Nuclear Regulatory Commission, 2001, Siefken L. SCDAP/RELAP5/MOD 3.3 Code Manual[M]: Division of Systems Technology, Office of Nuclear Regulatory Research, US Nuclear Regulatory Commission, 2001.

[11] J. Ebert, "A versatile data structure for edge-oriented graph algorithms," *Communications of the ACM*, vol. 30, no. 6, pp. 513–519, 1987.

[12] T. A. Davis, *Direct Methods for Sparse Linear Systems*, SIAM, Philadelphia, Pa, USA, 2006.