A domain renumbering algorithm for multi-domain boundary face method

Jianming Zhang[*], Chenjun Lu, Yuan Li, Lei Han, Pan Wang, Guangyao Li

State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, China

*Correspondence to: Jianming Zhang
College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China
Telephone: +86-731-88823061
E-mail: zhangjianm@gmail.com

Abstract: In this paper, a domain number optimization algorithm for multi-domain boundary face method is proposed. The advantage of the algorithm is to make nonzero blocks of the overall assembled matrix are as close to the main diagonal as possible. This will minimize the block fill-in effect that occurs during the solution process. Consequently, the time used for LU-decomposition and the memory requirement of the matrix will be reduced significantly. In this algorithm, one or more level structures are generated by considering the freedom degrees and the connectivity of the domains. Then we renumber the domains according to the level structure of the smallest bandwidth. Four steady-state heat conduction problems of multi-domain are solved to test the algorithm, and high efficiency is observed.

Keywords: multi-domain formulation, boundary face method, domain renumbering

1. Introduction

The boundary element method (BEM) is an efficient numerical technique for solving engineering problems, such as Laplace equation, Navier's equation, Helmholtz equation and linear diffusion-reaction equation [1-6]. In the BEM, partial differential equations are converted to an equivalent boundary integral equation by Green's theorem and a fundamental solution. Thus, only boundary discretization can lead to an accurate result together with a high rate of convergence. This is the main advantage over the classic domain methods such as finite element method (FEM) and finite difference method (FDM).

Based on the BEM, Zhang et al. have proposed the boundary face method (BFM) in recent years [7, 8]. The BFM is implemented directly based on the boundary representation data structure (B-rep) that is used in most CAD packages for geometry modeling. Each bounding surface of geometry model is represented as parametric form by the geometric map between the parametric space and the physical space. Both boundary integration and variable approximation are performed in the parametric space. The integrand quantities are calculated directly from the faces rather than from elements, and thus no geometric error will be introduced. The BFM has been applied in analyses of various structural problems with complicated geometries [9-15]. In this paper, we employ the BFM to solve 3D potential problems of multi-domain.

Multi-domain formulations are employed when the entire domain under consideration is governed by individual differential equations in different parts and/or constructed of different materials [16-21]. Besides, in the case of a domain with complicated boundary profile or parallel computation, the domain may be decomposed for better computational efficiency. In a multi-domain solver, the original domain is divided into a finite number of sub-domains, and in each of them the full integral representation formula is applied. At the common interfaces between the adjacent sub-domains, the corresponding full matching conditions are enforced. How to satisfy the continuity and equilibrium conditions at the interfaces is one of the important aspects of implementation for a multi-domain algorithm. There are mainly two methods in the literature: the standard multi-domain method [22] and the domain decomposition method [23]. In the standard multi-domain method, the discretized equations corresponding to the sub-domains are assembled into a system of equation according the boundary and interface conditions. While the matrices that arise in the single-domain formulation are fully populated, the multi-domain formulation leads to overall matrix equations with a sparse blocked structure. In the domain decomposition method, the interface conditions are assumed and then the sub-domain problems are solved independently. The modification of the interface condition is usually iterative using different methodologies, as the Schwarz Neumann-Neumann and Schwarz Dirichlet-Neumann methods. Repetition of the iteration process is continued until convergence. The domain decomposition method allows different type of discretization methods (e.g. BEM and FEM) to be used for a numerical solution of the individual sub-domains and coupling between them without accessing to the source codes of the methods. However, it has some relevant parameters to be chosen and the optimal values for these parameters are usually problem-dependent. This arbitrariness represents a disadvantage of the method.

In the present paper, we adopt the standard multi-domain method, and make full use of the resultant sparsity of the coefficient matrix in the equations during the solution process. The sparsity pattern (population of the nonzero blocks) of the matrix has a severe impact on the condition number of the matrix, and thus on the solution procedure. If all of the nonzero blocks are clustered near the main diagonal, we call the matrix is banded. Research indicates that reducing the bandwidth of the matrix will reduce both the memory requirement and the computation time. Many algorithms have been proposed for the problem of matrix bandwidth reduction, such as the Cuthill-McKee (CM) algorithm [24] and the Gibbs-Poole-Stockmeyer (GPS) algorithm [25]. For these algorithms, it is assumed that the connection of the nonzero elements of the matrix is continuous, and is feasible for FEM, only.

On the implementation of the multi-domain BFM, we reduce the bandwidth of the overall assembled matrix from the following two aspects: (1) As the sparse structure of the matrix is directly related to the ordering of unknowns in the overall system of equations, we use the ordering strategy suggested by J.H. Kane [22] to obtain an optimal blocks structure. (2) A domain number optimization algorithm is proposed for the first time in this paper. The algorithm will make the nonzero blocks of the matrix are as close to the main diagonal as possible. The number of the block fill-ins (to be discussed in Section 3) will be reduced correspondingly, which should make the equation solving task run much quicker. In this algorithm, firstly, one or more level structures are generated by considering the freedom degrees and the connectivity of the domains. Then the bandwidths of the level structures are computed by a formulation we defined. Finally, we renumber the domains according to the level structure of the smallest bandwidth. Four examples of three-dimensional steady-state heat conduction problems, including a practical engineering problem of gravity dams, are investigated to test the optimization algorithm. Results show that the time used for LU-decomposition of the overall matrix is reduced significantly, as well as the memory requirement.

This paper is organized as follows: We introduce the formulation of the multi-domain BFM in Section 2. The domain number sequence optimization algorithm is described in detail in Section 3. In Section 4 we present the numerical examples and finally, the paper ends with conclusions in Section 5.

2. Multi-domain formulations of BFM

In this section, we will derive a multi-domain formulation to solve 3D potential problems. The formulation is obtained by assembling the equations for each single domain into an overall system of equations using the continuous and equilibrium relations on the interfaces between the sub-domains.

The 3D potential problem can be mathematically described as

$$\begin{aligned} u,_{ii} &= 0, \quad \forall x \in \Omega \\ u &= \bar{u}, \quad \forall x \in \Gamma_u \\ ku,_i n_i &\equiv q = \bar{q}, \quad \forall x \in \Gamma_q \\ ku,_i n_i &= h(u_\infty - u), \quad \forall x \in \Gamma_c \end{aligned} \tag{1}$$

where *Ω* is the corresponding domain which is enclosed by $\Gamma=\Gamma_u \cup \Gamma_q \cup \Gamma_c$. On the essential boundary Γ_u and the flux boundary Γ_q, we impose the boundary condition $\bar{u}$ and $\bar{q}$, which denote the prescribed temperature and the normal flux, respectively. The boundary Γ_c represent that have convective conditions. u_∞ is the temperature in the surrounding medium, *k* and *h* stand for the conductivity and convective heat transfer coefficient, respectively. *n* with components n_i, $i=1,2,3$ is the outward normal on the boundary *Γ*.

The problem can be converted into an equivalent BIE [1] which is described as the following formulation:

$$\int_\Gamma (u(\mathbf{s}) - u(\mathbf{y}))q^s(\mathbf{s},\mathbf{y})d\Gamma = \int_\Gamma q(\mathbf{s})u^s(\mathbf{s},\mathbf{y})d\Gamma \tag{2}$$

In this formula, ***y*** and ***s*** respectively stand for the field point and source point. $u^S(\boldsymbol{s}, \boldsymbol{y})$ and $q^S(\boldsymbol{s}, \boldsymbol{y})$ are the corresponding fundamental solutions which satisfies

$$u^s,_{ii} = \delta(y,s), \quad \forall s \in \Omega \tag{3}$$

and

$$q^s = \partial u^s / \partial n(s) \tag{4}$$

For 3D potential problems, the fundamental solution can be written as

$$u^s(\mathbf{s},\mathbf{y}) = \frac{1}{4\pi}\frac{1}{r(\mathbf{s},\mathbf{y})} \tag{5}$$

where *r* is the distance between field point ***y*** and source point ***s***.

Dividing *Γ* into *M* elements and approximating *u* and *q* with Lagrange interpolation functions, the BIE (2) is discretized into

$$\sum_{j=1}^{M}\int_{\Gamma_j} u^s(\mathbf{s},\mathbf{y})\sum_{k=1}^{N} N_k(\mathbf{s})q_k d\Gamma(\mathbf{s}) - \sum_{j=1}^{M}\int_{\Gamma_j} q^s(\mathbf{s},\mathbf{y})\sum_{k=1}^{N}(N_k(\mathbf{s}) - N_k(\mathbf{y}))u_k d\Gamma(\mathbf{s}) = 0 \tag{6}$$

where u_k and q_k are the value of temperature and the normal flux on boundary nodes, $N_k(\boldsymbol{y})$ and

*N_k(**s**)* are the contributions from the k-th interpolation node to the collocation point ***y*** and field point ***s***, respectively. After collocating the field point at every interpolation point, we get the following system of linear equations which can be solved efficiently：

$$\mathbf{Gq} - \mathbf{Hu} = \mathbf{0} \tag{7}$$

The full details of the BFM formulation can refer to [8]. In order to simplify the process and present the multi-domain formulation more clearly, here we consider three sub-domains as an example. The model is shown in Fig. 1, in which cube 1 and cube 2 intersect in Γ^{12}, cube 1 and cube 3 intersect in Γ^{13} and cube 2 and cube 3 intersect in Γ^{23}.

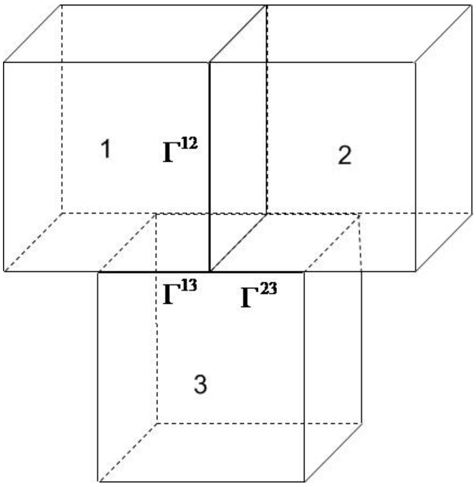

Figure 1: The sketch map of 3 cubes.

On the intersection boundaries, we have the conditions:

$$\hat{u}_i = \hat{u}_j \tag{8}$$

$$\hat{q}_i = -\hat{q}_j \tag{9}$$

where *i* and *j* stand for the number of domain, *i, j =1,2,3*.

The matrix equation of multi-domain BFM for sub-domain 1 can be written as following:

$$\begin{bmatrix} H_{dd} & H_{dn} & H_{dr} & H_{d2} & H_{d3} \\ H_{nd} & H_{nn} & H_{nr} & H_{n2} & H_{n3} \\ H_{rd} & H_{rn} & H_{rr} & H_{r2} & H_{r3} \\ H_{2d} & H_{2n} & H_{2r} & H_{22} & H_{23} \\ H_{3d} & H_{3n} & H_{3r} & H_{32} & H_{33} \end{bmatrix} \begin{Bmatrix} \bar{u} \\ u \\ \tilde{u} \\ \hat{u}_2 \\ \hat{u}_3 \end{Bmatrix} = \begin{bmatrix} G_{dd} & G_{dn} & G_{dr} & G_{d2} & G_{d3} \\ G_{nd} & G_{nn} & G_{nr} & G_{n2} & G_{n3} \\ G_{rd} & G_{rn} & G_{rr} & G_{r2} & G_{r3} \\ G_{2d} & G_{2n} & G_{2r} & G_{22} & G_{23} \\ G_{3d} & G_{3n} & G_{3r} & G_{32} & G_{33} \end{bmatrix} \begin{Bmatrix} q \\ \bar{q} \\ \tilde{q} \\ \hat{q}_2 \\ \hat{q}_3 \end{Bmatrix} \tag{10}$$

in which subscripts *d, n, r* stand for the boundary conditions of Dirichlet type, Neumann type and Robin type, respectively. Subscripts 2 and 3 denote for the areas where the boundary of sub-domain 1 intersects with sub-domain 2 or sub-domain 3. By considering the boundary condition of sub-domain 1, we denote

$$\{\tilde{q}\} = \boldsymbol{\beta} - \boldsymbol{\alpha}\{\tilde{u}\} \tag{11}$$

where

$$\boldsymbol{\beta} = \begin{Bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{Bmatrix}, \quad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_n \end{bmatrix} \tag{12}$$

$$\beta_i = -hu_\infty \,, \quad \alpha_i = -h \tag{13}$$

Then we can reassemble the above boundary integral systems into

$$\begin{bmatrix} G^1_{dd} & -H^1_{dn} & (-H^1_{dr} - G^1_{dr}\alpha) & -H^1_{d2} & -H^1_{d3} & G^1_{d2} & G^1_{d3} \\ G^1_{nd} & -H^1_{nn} & (-H^1_{nr} - G^1_{nr}\alpha) & -H^1_{n2} & -H^1_{n3} & G^1_{n2} & G^1_{n3} \\ G^1_{rd} & -H^1_{rn} & (-H^1_{rr} - G^1_{rr}\alpha) & -H^1_{r2} & -H^1_{r3} & G^1_{r2} & G^1_{r3} \\ G^1_{2d} & -H^1_{2n} & (-H^1_{2r} - G^1_{2r}\alpha) & -H^1_{22} & -H^1_{23} & G^1_{22} & G^1_{23} \\ G^1_{3d} & -H^1_{3n} & (-H^1_{3r} - G^1_{3r}\alpha) & -H^1_{32} & -H^1_{33} & G^1_{32} & G^1_{33} \end{bmatrix} \begin{Bmatrix} q^1_d \\ u^1_n \\ \tilde{u}^1_r \\ \widehat{u}^1_2 \\ \widehat{u}^1_3 \\ \widehat{q}^1_2 \\ \widehat{q}^1_3 \end{Bmatrix} = \begin{bmatrix} H^1_{dd} & -G^1_{dn} & -G^1_{dr} \\ H^1_{nd} & -G^1_{nn} & -G^1_{nr} \\ H^1_{rd} & -G^1_{rn} & -G^1_{rr} \\ H^1_{2d} & -G^1_{2n} & -G^1_{2r} \\ H^1_{3d} & -G^1_{3n} & -G^1_{3r} \end{bmatrix} \begin{Bmatrix} \overline{u}^1_d \\ \overline{q}^1_n \\ \beta^1 \end{Bmatrix} = \overline{y}^1 \tag{14}$$

Similarly, for sub-domain 2 and sub-domain 3, we also have the following reassembled boundary integral systems:

$$\begin{bmatrix} G^2_{11} & G^2_{1d} & -H^2_{1n} & (-H^2_{1r} - G^2_{1r}\alpha) & -H^2_{13} & -H^2_{11} & G^2_{13} \\ G^2_{d1} & G^2_{dd} & -H^2_{dn} & (-H^2_{dr} - G^2_{dr}\alpha) & -H^2_{d3} & -H^2_{d1} & G^2_{d3} \\ G^2_{n1} & G^2_{nd} & -H^2_{nn} & (-H^2_{nr} - G^2_{nr}\alpha) & -H^2_{n3} & -H^2_{n1} & G^2_{n3} \\ G^2_{r1} & G^2_{rd} & -H^2_{rn} & (-H^2_{rr} - G^2_{rr}\alpha) & -H^2_{r3} & -H^2_{r1} & G^2_{r3} \\ G^2_{31} & G^2_{3d} & -H^2_{3n} & (-H^2_{3r} - G^2_{3r}\alpha) & -H^2_{33} & -H^2_{31} & G^2_{33} \end{bmatrix} \begin{Bmatrix} \widehat{q}^2_1 \\ q^2_d \\ u^2_n \\ \tilde{u}^2_r \\ \widehat{u}^2_3 \\ \widehat{u}^2_1 \\ \widehat{q}^2_3 \end{Bmatrix} = \begin{bmatrix} H^2_{1d} & -G^2_{1n} & -G^2_{1r} \\ H^2_{dd} & -G^2_{dn} & -G^2_{dr} \\ H^2_{nd} & -G^2_{nn} & -G^2_{nr} \\ H^2_{rd} & -G^2_{rn} & -G^2_{rr} \\ H^2_{3d} & -G^2_{3n} & -G^2_{3r} \end{bmatrix} \begin{Bmatrix} \overline{u}^2_d \\ \overline{q}^2_n \\ \beta^2 \end{Bmatrix} = \overline{y}^2 \tag{15}$$

$$\begin{bmatrix} G^3_{11} & G^3_{12} & G^3_{1d} & -H^3_{1n} & (-H^3_{1r} - G^3_{1r}\alpha) & -H^3_{11} & -H^3_{12} \\ G^3_{21} & G^3_{22} & G^3_{2d} & -H^3_{2n} & (-H^3_{2r} - G^3_{2r}\alpha) & -H^3_{21} & -H^3_{22} \\ G^3_{d1} & G^3_{d2} & G^3_{dd} & -H^3_{dn} & (-H^3_{dr} - G^3_{dr}\alpha) & -H^3_{d1} & -H^3_{d2} \\ G^3_{n1} & G^3_{n2} & G^3_{nd} & -H^3_{nn} & (-H^3_{nr} - G^3_{nr}\alpha) & -H^3_{n1} & -H^3_{n2} \\ G^3_{r1} & G^3_{r2} & G^3_{rd} & -H^3_{rn} & (-H^3_{rr} - G^3_{rr}\alpha) & -H^3_{r1} & -H^3_{r2} \end{bmatrix} \begin{Bmatrix} \widehat{q}^3_1 \\ \widehat{q}^3_2 \\ q^3_d \\ u^3_n \\ \tilde{u}^3_r \\ \widehat{u}^3_1 \\ \widehat{u}^3_2 \end{Bmatrix} = \begin{bmatrix} H^3_{1d} & -G^3_{1n} & -G^3_{1r} \\ H^3_{2d} & -G^3_{2n} & -G^3_{2r} \\ H^3_{dd} & -G^3_{dn} & -G^3_{dr} \\ H^3_{nd} & -G^3_{nn} & -G^3_{nr} \\ H^3_{rd} & -G^3_{rn} & -G^3_{rr} \end{bmatrix} \begin{Bmatrix} \overline{u}^3_d \\ \overline{q}^3_n \\ \beta^3 \end{Bmatrix} = \overline{y}^3 \tag{16}$$

Considering the interface conditions and denoting $F^k_{ir} = H^k_{ir} + G^k_{ir}\alpha$, we have the final systems of equation:

$$\left[\begin{array}{ccccccccccccccc}
G^1_{dd} & -H^1_{dn} & -F^1_{dr} & -H^1_{d2} & -H^1_{d3} & -G^1_{d2} & & & & & -G^1_{d3} & & & & \\
G^1_{nd} & -H^1_{nn} & -F^1_{nr} & -H^1_{n2} & -H^1_{n3} & -G^1_{n2} & & & & & -G^1_{n3} & & & & \\
G^1_{rd} & -H^1_{rn} & -F^1_{rr} & -H^1_{r2} & -H^1_{r3} & -G^1_{r2} & & & & & -G^1_{r3} & & & & \\
G^1_{2d} & -H^1_{2n} & -F^1_{2r} & -H^1_{22} & -H^1_{23} & -G^1_{22} & & & & & -G^1_{23} & & & & \\
G^1_{3d} & -H^1_{3n} & -F^1_{3r} & -H^1_{32} & -H^1_{33} & -G^1_{32} & & & & & -G^1_{33} & & & & \\
 & & & -H^2_{11} & & G^2_{11} & G^2_{1d} & -H^2_{1n} & -F^2_{1r} & -H^2_{13} & & -G^2_{13} & & & \\
 & & & -H^2_{d1} & & G^2_{d1} & G^2_{dd} & -H^2_{dn} & -F^2_{dr} & -H^2_{d3} & & -G^2_{d3} & & & \\
 & & & -H^2_{n1} & & G^2_{n1} & G^2_{nd} & -H^2_{nn} & -F^2_{nr} & -H^2_{n3} & & -G^2_{n3} & & & \\
 & & & -H^2_{r1} & & G^2_{r1} & G^2_{rd} & -H^2_{rn} & -F^2_{rr} & -H^2_{r3} & & -G^2_{r3} & & & \\
 & & & -H^2_{31} & & G^2_{31} & G^2_{3d} & -H^2_{3n} & -F^2_{3r} & -H^2_{33} & & -G^2_{33} & & & \\
 & & & & -H^3_{11} & & & & & -H^3_{12} & G^3_{11} & G^3_{12} & G^3_{1d} & -H^3_{1n} & -F^3_{1r} \\
 & & & & -H^3_{21} & & & & & -H^3_{22} & G^3_{21} & G^3_{22} & G^3_{2d} & -H^3_{2n} & -F^3_{2r} \\
 & & & & -H^3_{d1} & & & & & -H^3_{d2} & G^3_{d1} & G^3_{d2} & G^3_{dd} & -H^3_{dn} & -F^3_{dr} \\
 & & & & -H^3_{n1} & & & & & -H^3_{n2} & G^3_{n1} & G^3_{n2} & G^3_{nd} & -H^3_{nn} & -F^3_{nr} \\
 & & & & -H^3_{r1} & & & & & -H^3_{r2} & G^3_{r1} & G^3_{r2} & G^3_{rd} & -H^3_{rn} & -F^3_{rr}
\end{array}\right]
\left\{\begin{array}{c}
q^1_d \\ u^1_n \\ \tilde{u}^1_r \\ \hat{u}^1_2 \\ \hat{u}^1_3 \\ \hat{q}^2_1 \\ q^2_d \\ u^2_n \\ \tilde{u}^2_r \\ \hat{u}^2_3 \\ \hat{q}^3_1 \\ \hat{q}^3_2 \\ q^3_d \\ u^3_n \\ \tilde{u}^3_r
\end{array}\right\}
=\left\{\begin{array}{c}
\bar{y}^1 \\ \bar{y}^2 \\ \bar{y}^3
\end{array}\right\} \tag{17}$$

3. Domain number sequence optimization algorithm

A system of linear equations is called sparse if only a relatively small number of its coefficient matrix elements are nonzero. From Eq. (17), it can be seen that a multi-domain formulation leads to an overall assembled matrix with a sparse blocked structure. The sparsity pattern of the coefficient matrix has a severe impact on the solution procedure especially when a direct equation solver (e.g. inversion or LU-decomposition) is employed. During the solution process, there are some initially zero elements will become nonzero which is called fill-ins, just like the blank blocks which are numbered 4, 5, 8 respectively shown in Fig. 2, for which storage must be reserved. It is wasteful to use general methods of linear algebra on such problems, because most of the O(N^3) arithmetic operations devoted to solving the set of equations or inverting the matrix involve zero operands. Furthermore, the memory requirement for storing the matrix for large scale problems may beyond available memory space, and it is wasteful to reserve storage for unfruitful zero elements. Therefore, carefully optimized to minimize the number of fill-ins or reduce the bandwidth of the matrix is necessary and significant.

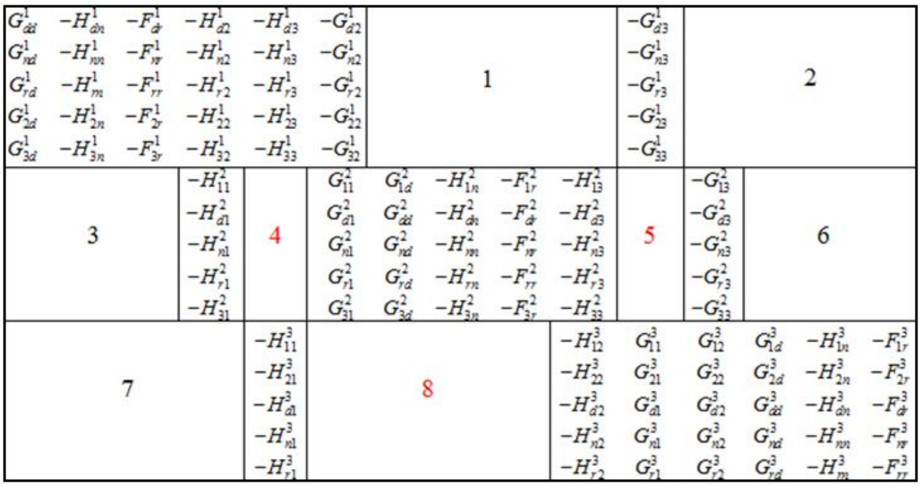

Figure 2: Matrix with a sparse blocked structure.

On the implementation of minimizing the number of fill-ins, we remark the following two aspects:

1. The sparsity pattern of the coefficient matrix in system equation (17) is related to the ordering of unknowns listed in the second column matrix in the left of the equation. In order that the nonzero blocks in the overall system are as close to the main diagonal as possible, we use the particular ordering suggested by J.H. Kane [22]. The order is determined by listing all permutations of two sub-domains as shown below:

11 12 13 21* 22 23 31* 32* 33

where the nodes exclusively belong to one domain has been grouped into a single cluster denoted by the number of the domain, e.g. the permutation “ii” represents “id in ir”. For permutations where the first digit is less than the second digit, blocks of potential are generated; otherwise, blocks of normal flux are generated. The permutations associated with blocks of normal flux are shown with an asterisk in the above list.

2. In our work, it is found that the domain number sequence has a severe impact on the population of the nonzero blocks. However, the domains number obtained from UG model are usually in random order. In order to obtain an overall assembled matrix of minimal bandwidth, a domain number optimization algorithm is proposed in this section.

Some important concept should be introduced before describing the procedure of the algorithm:

Degree of a domain: the degree of a domain is the amount of domains adjacent to it.

Level structure: If D is a finite nonempty set of the domains, then a level structure is defined as a partition of the set into levels $L_1, L_2, \ldots, L_k$ such that

1. all domains adjacent to domains in level L_1 are in either level L_1 or L_2,

2. all domains adjacent to domains in level L_k are in either level L_k or L_{k-1}, and

3. for $1 < i < k$, all domains adjacent to domains in level L_i are in either level L_{i-1}, L_i, or L_{i+1}.

Then we illustrate the optimization algorithm by an example. In this example, the geometry of the model which consists of nine domains is shown in Fig. 3. The original domain numbers obtained from UG model are labeled in this picture. We will renumber these domains according to the following algorithm:

A. Generate the level structures rooted at each domain of the minimum degree. Its levels are determined by

1. L_1 = {the root domain}, and

2. for $i > 1$, L_i is the set of all those domains adjacent to domains of level L_{i-1} and not yet assigned to a level.

For the model in Fig. 3, the domain numbered 5 and 9 will be selected to be the roots. The corresponding level structures are shown in Fig. 4 and Fig. 5 respectively.

B. For each rooted level structure generated in step A, number the domains level by level with consecutive positive integers according to the following procedure:

1. The root domain is assigned the number 1.

2. For each successive level, beginning with level 2, first renumber the domains adjacent to the lowest renumbered domain of the preceding level, in order of decreasing original number. The remaining domains adjacent to the next lowest renumbered domain of the preceding level are renumbered next, again in order of decreasing original number. Continue the process until all domains of the current level are renumbered, then begin again on the next level. The procedure

terminates when the domains of all levels have been renumbered.

The new number of the domains are labeled in bracket “()” with red color in Fig. 4 and Fig. 5, the optimized domain number sequence are 5, 8, 4, 7, 3, 6, 2, 9, 1, and 9, 6, 1, 7, 4, 2, 8, 3, 5 respectively. The straight lines in this figure represent that two domains located at the ends of the line are adjacent. The value of D_i *(i=1,2,...,14)* is the difference between the optimized numbers of the adjacent domains, e.g. $D_3 = 5\text{-}2 = 3$ and $D_{12} = 8\text{-}6 = 2$ in Fig. 4.

C. For each numbering produced in step B, compute the corresponding bandwidth, which is defined by

$$B = \sum_{i=1}^{N} (D_i - 1) \tag{18}$$

where, N represents the amount of the straight lines. Select the numbering which produces the smallest bandwidth as the finally optimized domain number sequence.

The bandwidth of the level structures in Fig. 4 and Fig. 5 are 12 and 18 respectively. Therefore, the domain number sequence as 5, 8, 4, 7, 3, 6, 2, 9, 1 will be selected. The sparsity patterns of the overall matrices before and after optimization are shown in Fig. 6. The green areas in Fig. 6 represent the zero elements which will be filled in during the solution process. It can be seen that the number of the fill-ins is reduced obviously after optimization, as well as the bandwidth of the matrix.

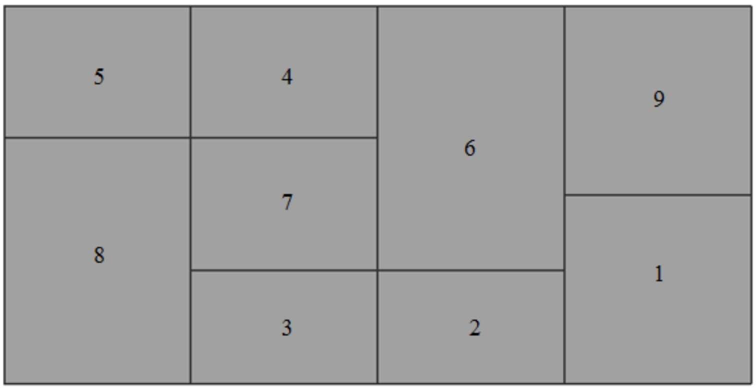

Figure 3: Geometry and the original number of the domains.

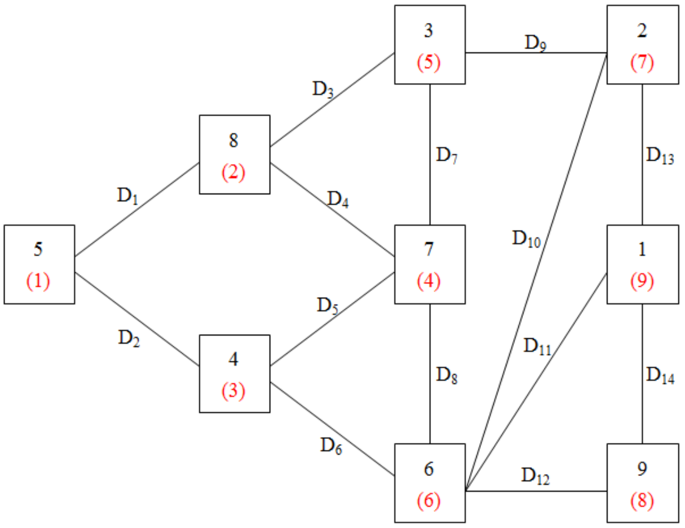

Figure 4: Level structure rooted at domain 5.

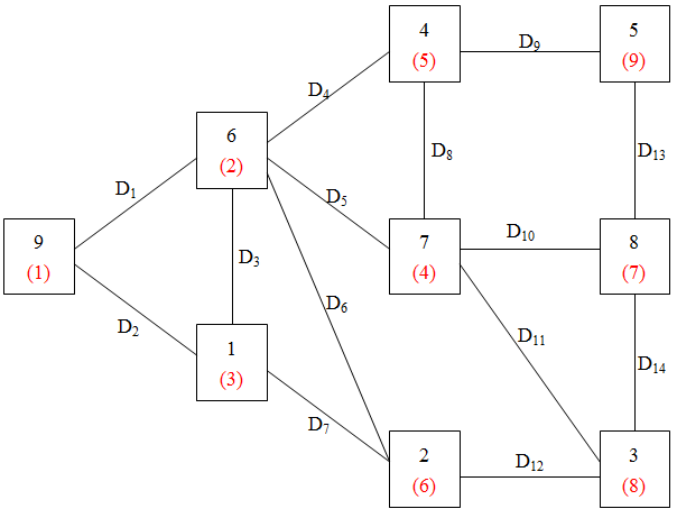

Figure 5: Level structure rooted at domain 9.

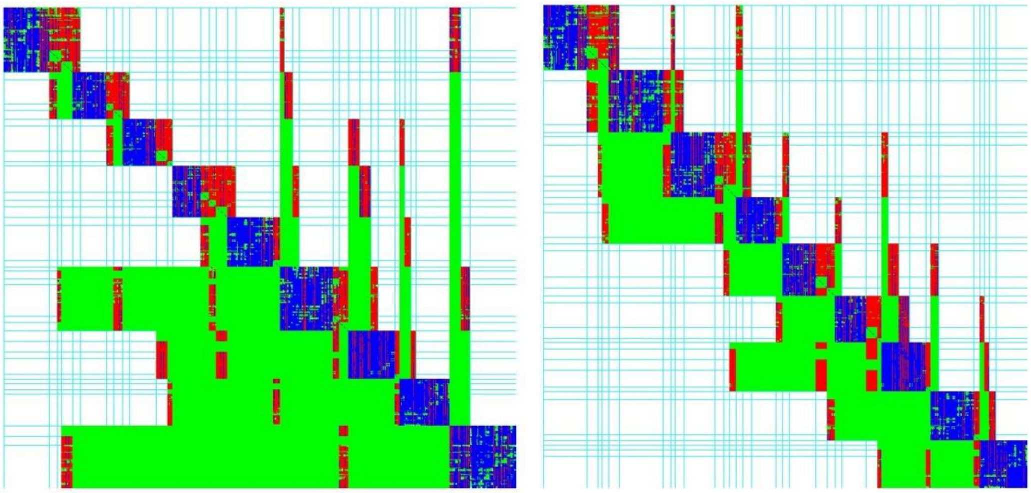

Figure 6: The assembled overall matrices before and after optimization.

4. Test examples

In this section, we will solve four steady-state heat conduction problems of different models with the multi-domain BFM to verify the efficiency of the domain number sequence optimization algorithm. All computations of these examples are carried out on the same desktop computer with an Intel(R) Core(TM) i7-3700K CPU (3.5GHZ) and 16GB RAM.

4.1 Example 1: a cuboid consists of six cubes

A simple model is first considered, which domain consists of six cubes. The side length of cubes is 100mm. Each cube is treated as a sub-domain naturally. The UG model and the original domain numbers are shown in Fig. 7. The level structure of the minimal bandwidth generated according to the proposed algorithm is shown in Fig. 8, in which the domain number after optimization is labeled in bracket "()" with red color. Actually, the optimized domain number sequence as 6, 5, 2, 4, 1, 3 and 3, 1, 4, 2, 5, 6 will result in overall assembled matrices with the same bandwidth. In this case we can select either of them as the final sequence. The sparsity patterns of the matrices before and after optimization are shown in Fig. 9. The bandwidth of the matrix is reduced as we expected.

We discretized the model with 46116 nodes. The computation costs of these four examples are summarized in Table1, in which the first column lists the example number; the second column lists the total number of discrete nodes; the third and fourth columns demonstrate the time used for LU-decomposition of the overall matrix before and after optimization respectively; the fifth and sixth columns demonstrate the memory required for the matrix storage before and after optimization respectively. It can be seen that a big improvement of efficiency has been achieved for example 1.

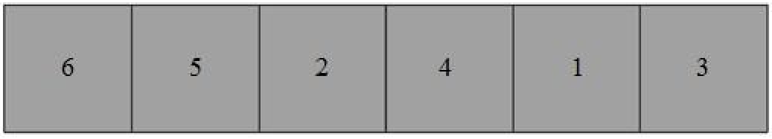

Figure 7: Geometry and the original numbers of the domains in example 1.

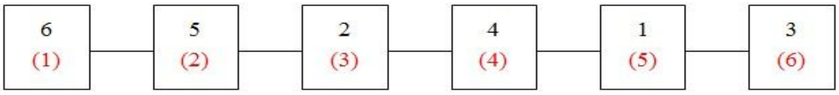

Figure 8: The level structure of the minimal bandwidth for example 1.

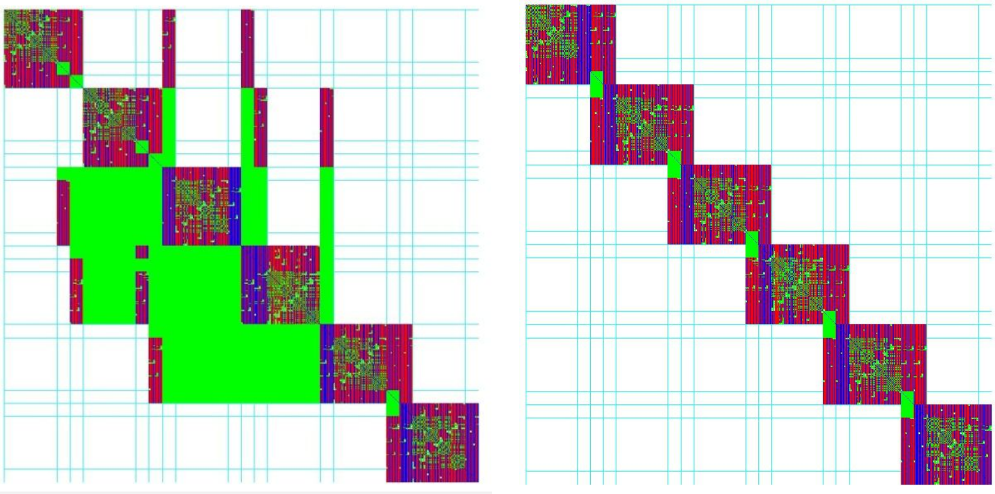

Figure 9: The assembled overall matrices before and after optimization for example 1.

Table 1: The time used for LU-decomposition and the memory requirement of matrix

	The total number of nodes	Used Time/s		Required Memory/MB	
		Before optimization	After optimization	Before optimization	After optimization
Example 1	46116	2374	481	7992.085	5222.721
Example 2	52992	2350	951	9219.367	5689.879
Example 3	4368	3	2	74.946	53.281
Example 4	21734	115	45	1067.237	773.126

4.2 Example 2: a crisscross model consisting of twelve cubes

For the second test example, we consider a crisscross model consisting of twelve cubes. The shape of the model and the original domain numbers are shown in Fig. 10. The side length of cubes is 100mm. We treat each cube as a sub-domain as usual. There are four domains numbered 4, 6, 8, and 12 respectively with the lowest degree. By comparing the bandwidth of the level structures rooted at them, the finally optimized domain number sequence 12, 1, 11, 10, 9, 7, 5, 8, 2, 3, 6, 4 is obtained just as shown in Fig. 11.

The model is discretized with 52992 nodes as illustrated in the second column of Table 1. The sparsity patterns of the overall matrices before and after optimization are shown in Fig. 12. The time used for LU-decomposition of the overall matrix and the memory requirement are illustrated in the last four columns of Table 1. It can be found that the efficiency is improved. For problems of complicated structures, more efficient computation will be obtained if we employ another novel linear equation solution method, such as a row elimination back-substitution method (REBSM) proposed by Gao [26], to replace LU-Decomposition solver.

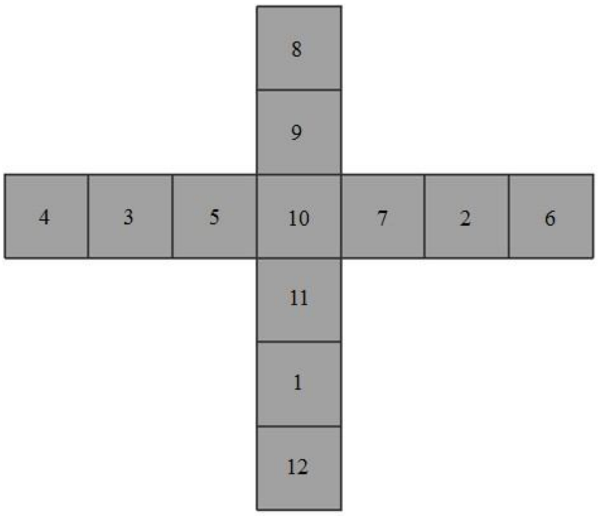

Figure 10: Geometry and the original numbers of the domains in example 2.

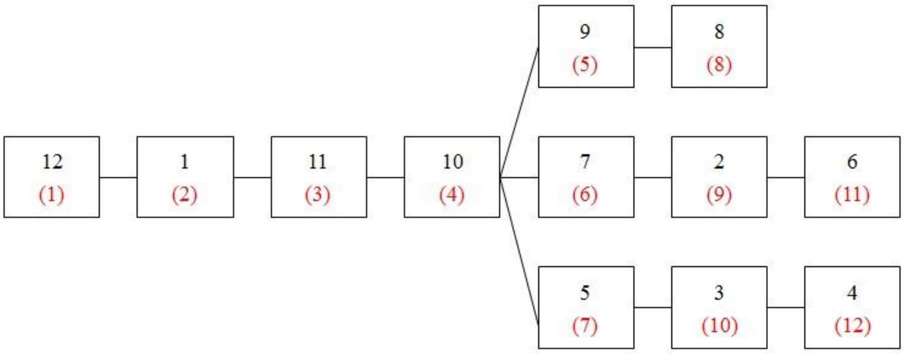

Figure 11: The level structure of the minimal bandwidth for example 2.

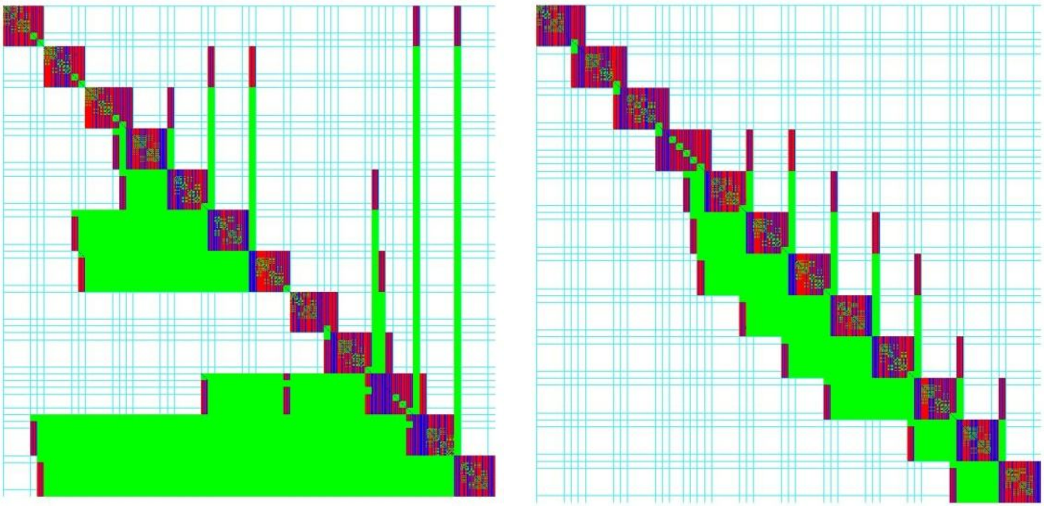

Figure 12: The assembled overall matrices before and after optimization for example 2.

4.3 Example 3: A cylinder model subdivided into eight sub-domains

In this section, we consider a cylinder model which is subdivided into eight sub-domains. The shape of the model and the original domain number are shown in Fig. 13. Dimensions are given as follows: the height is 200mm, the radius of the inner circle and the external circle are 200mm and 300mm, respectively. This model is special because that each domain has the same degree, and the level structures rooted at them have the same bandwidth. Thus, select one of them be the root, the corresponding level structure and the optimized domain number sequence are as shown in Fig. 14.

The model is discretized with 4368 nodes. The sparsity patterns of the overall matrices before and after optimization are shown in Fig. 15. Form Table 1, we can find that the time used for LU-decomposition of the overall matrix and the memory requirement are reduced.

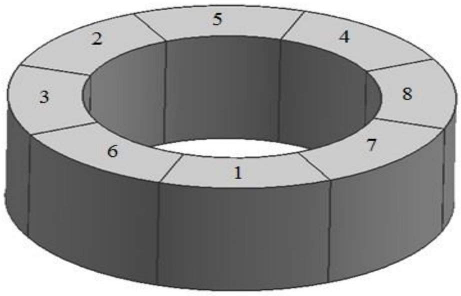

Figure 13: Geometry and the original numbers of the domains in example 3.

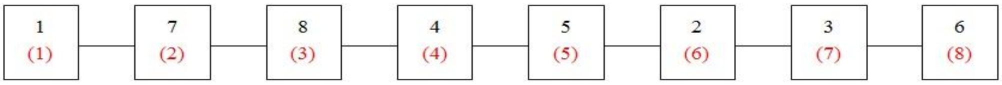

Figure 14: The level structure of the minimal bandwidth for example 3.

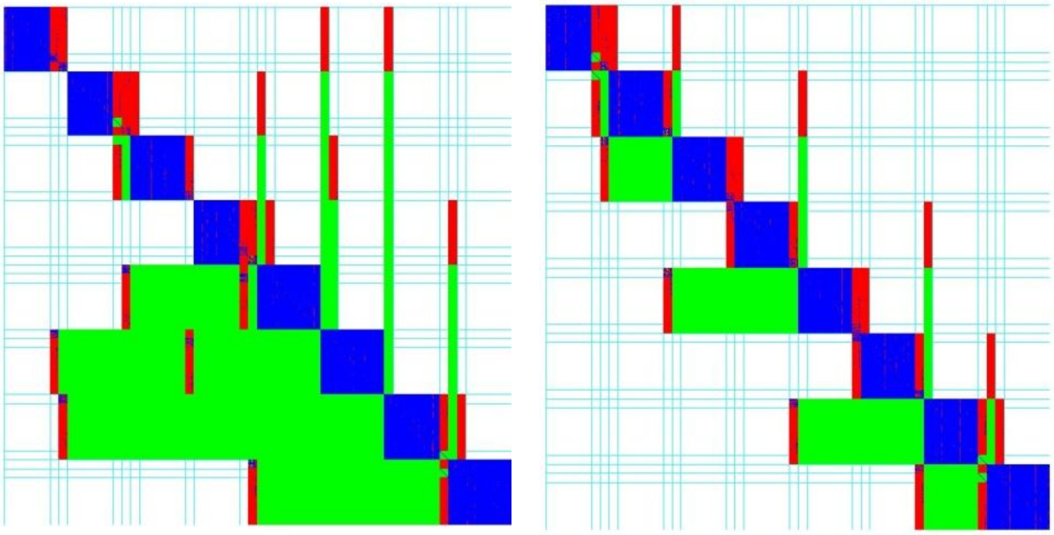

Figure 15: The assembled overall matrices before and after optimization for example 3.

4.4 Example 4: A real massive gravity dam

We take a real massive gravity dam as the last test example. In the process of the dam construction, it is divided into 13 layers to cast layer by layer at different times. Therefore, a multi-domain solver is a must for the simulation. The UG model and the original domain numbers are shown in Fig. 16. The optimized domain number sequence is 12, 13, 6, 11, 10, 9, 5, 4, 1, 3, 8, 2, 7, which is shown in Fig. 17.

The structure is discretized with 21734 nodes. The sparsity patterns of the overall matrices before and after optimization are shown in Fig. 18. The time used for LU-decomposition of the overall matrix is 115s and 45s, respectively, just as illustrated in the last row of Table 1. A big improvement of efficiency has been achieved. And it can be expected that with increasing the total number of nodes, the improvement will be more remarkable.

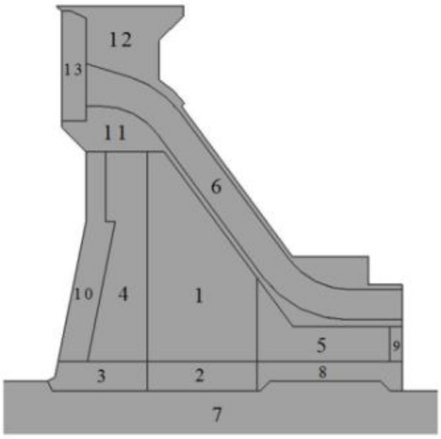

Figure 16: Domain numbers of a concrete dam in example 4.

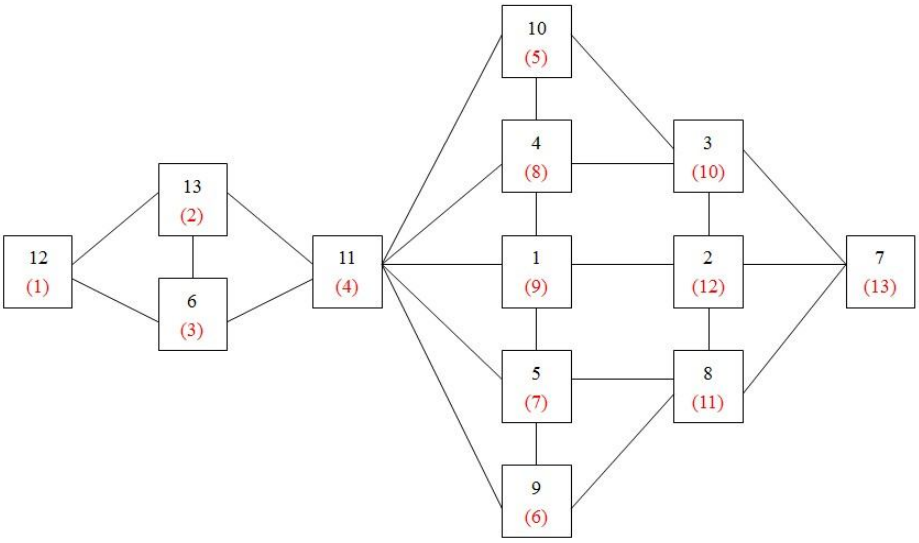

Figure 17: The level structure of the minimal bandwidth for example 4.

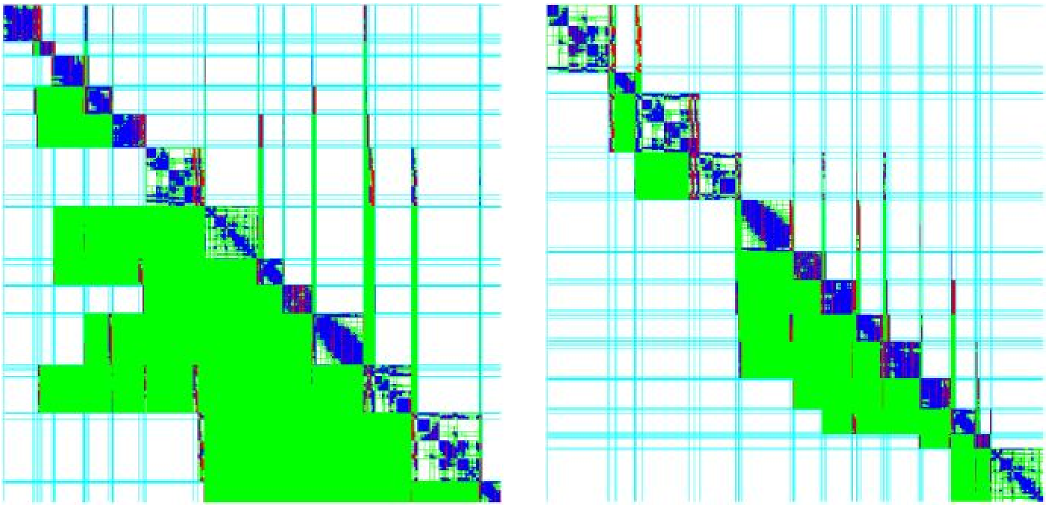

Figure 18: The assembled overall matrices before and after optimization for example 4.

5. Conclusion

In the implementation of multi-domain BFM, a domain number sequence optimization algorithm has been proposed to reduce the bandwidth of the overall assembled matrix. In this algorithm, one or more level structures are generated, which are rooted at the domains of the lowest degree. For each successive structure, a value representing the bandwidth is computed. Finally, the domains are renumbered according to the level structure of the smallest bandwidth. The root domain is assigned number 1. Then renumber the other domains level by level with consecutive positive integers by considering their original numbers and the connectivity with the preceding level domains. Four numerical examples are presented to study the performance of the proposed algorithm. Results demonstrate that the time used for LU-decomposition of the overall matrix and the memory requirement for storing the matrix have been reduced significantly.

Acknowledgement

This work was supported by National Science Foundation of China under grant number 11172098 and National 973 Project of China under grant number 2010CB328005.

References

[1] Brebbia CA. The Boundary Element Method for Engineers, Pentech Press London, 1978.

[2] Cheng AHD, Cheng DT. Heritage and early history of the boundary element method. Engineering Analysis with Boundary Elements 2005; 29: 268-302.

[3] Chen JT, Lin SR, Chen KH. Degenerate scale problem when solving Laplaces equation by BEM and its treatment. International Journal for Numerical Methods in Engineering 2005; 62(2): 233-261.

[4] Niu ZR, Zhou HL. The natural boundary integral equation in potential problems and regularization of the hypersingular integral. Computers & Structures 2004; 82(2,3): 315-323.

[5] Shen L, Liu YJ. An adaptive fast multipole boundary element method for three-dimensional acoustic wave problems based on the Burton–Miller formulation. Computational Mechanics 2007; 40: 461-72.

[6] Yang K, Gao XW. Radial integration BEM for transient heat conduction problems. Engineering Analysis with Boundary Elements 2010; 34: 557-563.

[7] Qin XY, Zhang JM., Li GY, Sheng XM. An element implementation of the boundary face method for 3D potential problems. Engineering Analysis with Boundary Elements 2010; 34: 934-943.

[8] Zhang JM, Qin XY, Han X, Li GY. A boundary face method for potential problems in three dimensions. International Journal for Numerical Methods in Engineering 2009; 80: 320-337.

[9] Zhou FL, Xie GZ, Zhang JM, Zheng XM. Transient heat conduction analysis of solids with small open-ended tubular cavities by boundary face method. Engineering Analysis with Boundary Elements 2013; 37: 542-550.

[10] Guo SP, Zhang JM, Li GY, Zhou FL. Three-dimensional transient heat conduction analysis by Laplace transformation and multiple reciprocity boundary face method. Engineering Analysis with Boundary Elements 2013; 37: 15-22.

[11] Wang XH, Zhang JM, Zhou FL, Zheng XM. An adaptive fast multipole boundary face method with higher order elements for acoustic problems in three-dimension. Engineering Analysis with Boundary Elements 2013; 37: 114-152.

[12] Qin XY, Zhang JM, Liu L, Li GY, Steady-state heat conduction analysis of solids with small open-ended tubular holes by BFM. International Journal of Heat and Mass Transfer 2012; 55: 6846-6853.

[13] Huang C, Zhang JM, Qin XY, Lu CJ, et al. Stress analysis of solids with open-ended tubular holes by BFM. Engineering Analysis with Boundary Elements 2012; 36: 1908-1916.

[14] Zhou FL, Zhang JM, Sheng XM, Li GY, A dual reciprocity boundary face method for 3D non-homogeneous elasticity problems. Engineering Analysis with Boundary Elements, 2012; 36: 1301-1310.

[15] Gu JL, Zhang JM, Sheng XM, The boundary face method with variable approximation by B-Spline basis function. International Journal of Computational Methods 2012; 9(1): 1240009/1-1240009/10.

[16] Ahmad S, Banerjee PK. Multi-domain BEM for two-dimensional problems of elastodynamics. International Journal for Numerical Methods in Engineering 1988; 26(4): 891-911.

[17] Yao ZH, Kong FZ, Wang HT, Wang PB. 2D simulation of composite materials using BEM. Engineering Analysis with Boundary Elements 2004; 28(8): 927-935.

[18] Liu YJ, A fast multipole boundary element method for 2D multi-domain elastostatic problems based on a dual BIE formulation. Computational Mechanics 2008; 42: 761-773.

[19] Gao XW, Guo L, Zhang C. Three-step multi-domain BEM solver for nonhomogeneous material problems. Engineering Analysis with Boundary Elements 2007; 31: 965-973.

[20] Zhang JM, Zhuang C, Qin XY, Li GY, Sheng XM. FMM-accelerated hybrid boundary node method for multi-domain problems. Engineering Analysis with Boundary Elements 2010; 34: 433-439.

[21] Zhang JM. Multi-domain Thermal Simulation of CNT Composites by Hybrid BNM. Numerical Heat Transfer 2008; 53: 246-258.

[22] Kane JH. Boundary Element Analysis in Engineering Continuum Mechanics. New Jersey: Prentice Hall Press; 1994.

[23] Meric RA. Domain decomposition methods for Laplace's equation by the BEM. Communications in Numerical Methods in Engineering 2000; 16: 545-57.

[24] Cuthill E, McKee J. Reducing the bandwidth of sparse symmetric matrices. Proceedings of the 1969 24th National Conference 1969; 157-172.

[25] Gibbs NE, Poole JWG, Stockmeyer PK. An algorithm for reducing the bandwidth and profile of a sparse matrix. SIAM Journal on Numerical Analysis 1976; 13(2): 236-250.

[26] Gao XW, Li LJ. A Solver of Linear Systems of Equations (REBSM) for Large-Scale Engineering Problems. International Journal of Computational Methods 2012; 9(1): 1240011(12 pages).