Modified Binary Tree in the Fast PIES for 2D Problems with Complex Shapes

Andrzej Kużelewski(✉), Eugeniusz Zieniuk, Agnieszka Bołtuć, and Krzystof Szerszeń

Institute of Informatics, University of Bialystok,
Ciolkowskiego 1M, 15-245 Bialystok, Poland
{akuzel,ezieniuk,aboltuc,kszerszen}@ii.uwb.edu.pl

Abstract. The paper presents a modified binary tree in the fast multipole method (FMM) included into the modified parametric integral equations system (PIES), called the fast PIES, in solving potential 2D boundary value problems with complex shapes. The modified binary tree proposed in this paper is built based on a one-dimensional reference system contrary to a quad-tree (based on a two-dimensional reference system) which is applied in the fast multipole boundary element method (FM-BEM). Application of the proposed tree allows reducing the number of numerical computations performed during its construction and fast multipole calculations in the fast PIES. The proposed modification of the tree in the fast PIES allows obtaining accurate solutions in engineering problems with complex shapes on a standard personal computer in a short time.

Keywords: Parametric integral equations system · Fast multipole method · Boundary value problems

1 Introduction

The fast multipole method (FMM) was initially proposed by Rokhlin [1] to accelerate the solution of 2D potential problems using boundary integral equations (BIE). Its main advantage is the reduction of the computational complexity of the matrix-vector multiplication from $O(N^2)$ to $O(N)$ (where N is the size of the matrix) combined with an iterative solver. In papers [2,3], Greengard refined the algorithm by adding decomposition of the domain using the hierarchical tree structure. It significantly reduces the utilization of random access memory (RAM) in computer. The use of the FMM for solving 2D and 3D boundary value problems (BVPs) is well-documented [4–6], also in application to the boundary element method (BEM) [7].

Mentioned above BEM together with the finite element method (FEM) [8] are well-established (might be called classic or conventional) methods of modelling and solving boundary problems. However, new approaches are also being developed to eliminate the disadvantages of classical methods (the time-consuming

© Springer Nature Switzerland AG 2020
V. V. Krzhizhanovskaya et al. (Eds.): ICCS 2020, LNCS 12138, pp. 1–14, 2020.
https://doi.org/10.1007/978-3-030-50417-5_1

discretization of a boundary or a domain resulting in a large number of finite or boundary elements). That group includes, among others, meshless methods [9] and still being developed the parametric integral equations system (PIES) [10].

The authors of this paper still working on development and application of the PIES in modelling and solving BVPs. The PIES has been used to solve potential [11], elasticity [12] or acoustics problems [13]. The PIES includes in its mathematical formalism the shape of the boundary of considered problem [10], therefore it does not require discretization of the domain or the boundary, contrary to element methods. The shape of the boundary is directly included into the PIES kernels using well-known functions from computer graphics - curves for 2D and surface patches for 3D problems. The accuracy of solutions can be improved by changing the number of collocation points only, without any interference in the shape of the modelled boundary. The efficiency of modelling and high accuracy of solutions obtained using the PIES has been confirmed in previous studies (e.g. [11–13]). The authors of this paper also proposed extensions of the PIES method for uncertainly defined [14,15] or transient [16] problems.

Conventional PIES, similarly to the BEM, produces non-symmetrical dense matrices using $O(N^2)$ operations and to solve the problem using direct solver, it needs another $O(N^3)$ operations (where N is the number of system equations). Therefore, solving engineering problems with complex shapes requires a lot of RAM and time-consuming computations. Application of OpenMP [17] or acceleration of solving the PIES using CUDA [18,19] allows for a significant reduction of time of computations. Unfortunately, the problem of limited resources of RAM in a personal computer (PC) still exists. Therefore, mentioned above parallelization techniques do not allow for efficient solving of problems on a PC.

The authors of this paper presented a way of including the FMM, based on a binary tree, into modified PIES in [20]. Application of the FMM increased the difficulty of implementation of so-called the fast PIES. However, the proposed approach gives accurate solutions in a short time for examples with a quite simple shape of the boundary. Some assumptions of the FMM may result in obtaining incorrect solutions, especially for complex shapes of a boundary. Our research has shown the need to modify the binary tree and to change the FMM algorithm to consider assumptions mentioned above for complex shapes of a boundary.

The main goal of this paper is to present the modified binary tree in the FMM used to accelerate numerical calculations in the PIES and to reduce utilization of RAM for BVPs with complex shapes of a boundary. The efficiency and accuracy of the proposed fast PIES are tested on 2D potential BVPs.

2 Formulation of the Fast PIES Method

Conventional PIES for 2D potential problems is presented by the following formula [10]:

$$\frac{1}{2}u_l(\overline{s}) = \sum_{j=1}^{n} \int_{s_{j-1}}^{s_j} \overline{U}^*_{lj}(\overline{s}, s) p_j(s) J_j(s) ds - \sum_{j=1}^{n} \int_{s_{j-1}}^{s_j} \overline{P}^*_{lj}(\overline{s}, s) u_j(s) J_j(s) ds, \quad (1)$$

where: $l = 1, 2, ..., n$, $s_{l-1} \leq \overline{s} \leq s_l$, $s_{j-1} \leq s \leq s_j$, s_{l-1} and s_{j-1} correspond to the beginning of l-th and j-th segment, while s_l and s_j to their ends, $J_j(s)$ is the Jacobian, n is the number of parametric segments that creates boundary of domain in parametric reference system s and $\overline{s}$ (presented in Fig. 1).

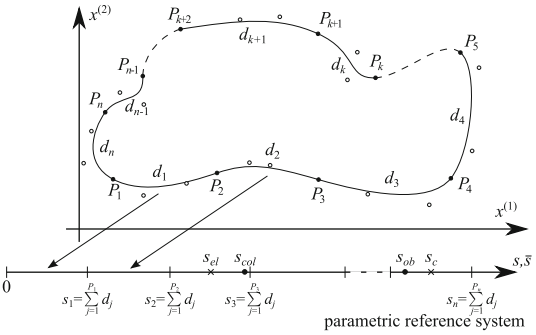

Fig. 1. Defining the shape of the boundary in the PIES on a straight line in parametric reference system $s, \overline{s}$

Integrands $\overline{U}^*_{lj}(\overline{s}, s)$ and $\overline{P}^*_{lj}(\overline{s}, s)$ in (1) are presented in the following form:

$$\begin{aligned} \overline{U}^*_{lj}(\overline{s}, s) &= \frac{1}{2\pi} \ln \frac{1}{\sqrt{\left(S^{(1)}\right)^2 + \left(S^{(2)}\right)^2}}, \\ \overline{P}^*_{lj}(\overline{s}, s) &= \frac{1}{2\pi} \frac{S^{(1)} n_j^{(1)}(s) + S^{(2)} \cdot n_j^{(2)}(s)}{\left(S^{(1)}\right)^2 + \left(S^{(2)}\right)^2}, \end{aligned} \tag{2}$$

where: $S^{(1)} = S_l^{(1)}(\overline{s}) - S_j^{(1)}(s)$ and $S^{(2)} = S_l^{(2)}(\overline{s}) - S_j^{(2)}(s)$, $n_j^{(k)}(s)$ $(k = \{1, 2\})$ are the components of normal vector to segment j. Expressions $S_k^{(i)}(s_n)$ $\{i = 1, 2\}, \{k = j, l\}, s_n = \{\overline{s}, s\}$ are parametric functions (curves or lines), which describe particular segments of a boundary (j or l), i is the number of coordinate in Cartesian reference system.

Boundary functions $u_j(s)$ and $p_j(s)$ in (1) are approximated by the following series:

$$u_j(s) = \sum_{k=0}^{N} u_j^{(k)} L_j^{(k)}(s), \quad p_j(s) = \sum_{k=0}^{N} p_j^{(k)} L_j^{(k)}(s), \tag{3}$$

where $u_j^{(k)}$ and $p_j^{(k)}$ are unknown or given values of boundary functions in defined points of the segment j, N - is the number of terms in series, $L_j^{(k)}(s)$ - are the base functions (Lagrange polynomials).

The pseudospectral method was applied to solve the PIES (1). Therefore, the PIES is transformed into the system of algebraic equations $\mathbf{Ax} = \mathbf{b}$. The oldest implementation of the PIES uses Gaussian elimination with pivot to solve

the system. Newer version, as well as the fast PIES, uses very fast and popular iterative solver based on the generalized minimal residual method (GMRES) [21]. Application of the FMM allows reducing computational time of matrix-vector multiplication (frequently used by GMRES solver) from order $O(N^2)$ to $O(N)$. There is also no need to store in the RAM entire matrix [6].

2.1 The Fast PIES Procedures

The FMM in the fast PIES is composed of two main steps: the upward and the downward pass (a full description is presented in [20]). The upward pass uses procedures of kernels expansion and moment-to-moment translation, while in the downward pass moment-to-local and local-to-local translations are applied. The most important information about these procedures are described below.

Kernels Expansion and Multipole Moments. The first procedure of the fast PIES is an expansion of kernels in Taylor series. The direct inclusion of the FMM [1] into the PIES is problematic due to the calculation of subsequent derivatives of kernels (2) for the Taylor series approximation. In the paper [20], the authors presented a modification of the PIES kernels by complex analysis. The PIES with modified kernels have the following form [20]:

$$\begin{aligned}\frac{1}{2}u_l(\overline{s}) = &\sum_{j=1}^{n} \Re\left\{ \int_{s_{j-1}}^{s_j} \overline{U}_{lj}^{*(c)}(\overline{\tau},\tau)\, p_j(s)\, J_j(s) ds \right\} \\ &- \sum_{j=1}^{n} \Re\left\{ \int_{s_{j-1}}^{s_j} \overline{P}_{lj}^{*(c)}(\overline{\tau},\tau)\, u_j(s)\, J_j(s) ds \right\},\end{aligned} \tag{4}$$

where: $\Re$ - is the real part of complex number, the parametric functions, which describe the shape of the boundary in the PIES, can be defined in complex notation as $\overline{\tau} = S_l^{(c)}(\overline{s}) = S_l^{(1)}(\overline{s}) + iS_l^{(2)}(\overline{s})$, $\tau = S_j^{(c)}(s) = S_j^{(1)}(s) + iS_j^{(2)}(s)$, (c) - means complex variable and an indeterminate (the imaginary unit) $i = \sqrt{-1}$. The complex form of kernels is as follows [20]:

$$\overline{U}_{lj}^{*(c)}(\overline{\tau},\tau) = -\frac{1}{2\pi}\ln(\overline{\tau}-\tau), \quad \overline{P}_{lj}^{*(c)}(\overline{\tau},\tau) = \frac{1}{2\pi}\frac{n^{(c)}}{\overline{\tau}-\tau}, \tag{5}$$

where $n^{(c)} = n^{(1)} + in^{(2)}$ - the complex notation of normal vector to the curve created segment j.

We assume that the point s_c (corresponding to the complex point τ_c) is close to the observation point s_{ob} and the point s_{el} (corresponding to the complex point τ_{el}) is close to the collocation point s_{col} (presented in Fig. 2). If $|s_{ob} - s_c| \ll |s_{col} - s_c|$ and $|\tau_{ob} - \tau_c| \ll |\tau_{col} - \tau_c|$, then kernels (5) can be expanded about the point τ_c using the Taylor series expansion. Therefore, we obtained the multipole moments $M_k(\tau_c)$ and $N_k(\tau_c)$ [20]:

$$M_k(\tau_c) = \int_{s_{j-1}}^{s_j} \frac{(\tau - \tau_c)^k}{k!} p_j(s)\, J_j(s) ds,$$
$$N_k(\tau_c) = \int_{s_{j-1}}^{s_j} \frac{(\tau - \tau_c)^{k-1}}{(k-1)!} n^{(c)} u_j(s)\, J_j(s)\, ds \tag{6}$$

and the following form of approximated Eq. (4) [20]:

$$\frac{1}{2} u_l(\overline{s}) = \sum_{j=1}^{n} \Re\left\{ \frac{1}{2\pi} \sum_{k=0}^{\infty} U_k(\overline{\tau}, \tau_c) M_k(\tau_c) \right\} - \sum_{j=1}^{n} \Re\left\{ \frac{1}{2\pi} \sum_{k=1}^{\infty} P_k(\overline{\tau}, \tau_c) N_k(\tau_c) \right\}, \tag{7}$$

where:

$$U_k(\overline{\tau}, \tau_c) = \begin{cases} -\ln(\overline{\tau} - \tau_c) & \text{for } k = 0 \\ \frac{(k-1)!}{(\overline{\tau} - \tau_c)^k} & \text{for } k \geq 1 \end{cases}, \quad P_k(\overline{\tau}, \tau_c) = \frac{(k-1)!}{(\overline{\tau} - \tau_c)^k} \quad \text{for} \quad k \geq 1.$$

Moments are calculated once only and they are independent of $\overline{\tau}$ (that is also from $\overline{s}$).

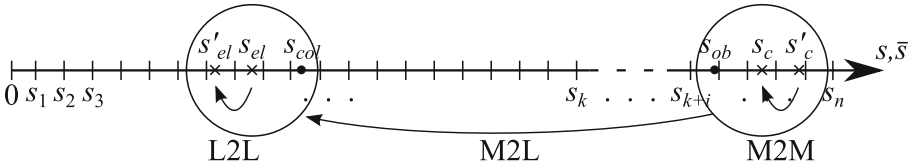

Fig. 2. Location of specific FMM points in the parametric reference system

Moment-to-Moment Translation (M2M). Moments (6) can be very efficiently recalculated for new point s_c' (corresponding to the complex point τ_c') close to the point s_c (presented in Fig. 2) without reuse of integration. For this purpose, moment-to-moment translation was used [20]:

$$M_k(\tau_c') = \sum_{m=0}^{k} \frac{(\tau_c - \tau_c')^{(k-m)}}{(k-m)!} M_m(\tau_c),$$
$$N_k(\tau_c') = \sum_{m=0}^{k} \frac{(\tau_c - \tau_c')^{(k-m)}}{(k-m)!} N_m(\tau_c). \tag{8}$$

Moment-to-Local Translation (M2L) and Local Expansion. The next step of the fast PIES is local expansion around the point s_{el} (corresponding

to the complex point τ_{el}) (presented in Fig. 2). If $|s_{col} - s_{el}| \ll |s_c - s_{el}|$ and $|\tau_{col} - \tau_{el}| \ll |\tau_c - \tau_{el}|$, then U_k and P_k in (7) can be expanded about the point τ_{el} using the Taylor series expansion, hence [20]:

$$\begin{aligned}\frac{1}{2}u_l(\bar{s}) = &\sum_{j=1}^{n} \Re\left\{\frac{1}{2\pi}\sum_{l=0}^{\infty} L_l^U(\tau_{el},\tau_c)\frac{(\tau_{col}-\tau_{el})^l}{l!}\right\} \\ &- \sum_{j=1}^{n} \Re\left\{\frac{1}{2\pi}\sum_{l=0}^{\infty} L_l^P(\tau_{el},\tau_c)\frac{(\tau_{col}-\tau_{el})^l}{l!}\right\},\end{aligned} \tag{9}$$

where:

$$L_l^U(\tau_{el},\tau_c) = \begin{cases} -\ln(\tau_{el}-\tau_c)M_0(\tau_c) + \sum_{k=1}^{\infty}\frac{(k-1)!\cdot M_k(\tau_c)}{(\tau_{el}-\tau_c)^k} & \text{for } l=0 \\ (-1)^l\sum_{k=0}^{\infty}\frac{(k+l-1)!\cdot M_k(\tau_c)}{(\tau_{el}-\tau_c)^{k+l}} & \text{for } l\geq 1, \end{cases}$$

$$L_l^P(\tau_{el},\tau_c) = (-1)^l\sum_{k=1}^{\infty}\frac{(k+l-1)!\cdot N_k(\tau_c)}{(\tau_{el}-\tau_c)^{k+l}} \quad \text{for } l\geq 1.$$

This procedure allows the transformation of moments collected at a point s_c (τ_c) to a local expansion point s_{el} (τ_{el}) and is called moment-to-local translation (M2L) [20].

During modelling complex shapes of the boundary, fulfilment of the condition $|\tau_{col} - \tau_{el}| \ll |\tau_c - \tau_{el}|$ might be not possible to meet for some cells. The results obtained in this case are subject to large errors. The authors of this paper proposed the way of elimination such cases - in the FMM algorithm they are considered as neighbouring cells (a more detailed description of the neighbourhood of the cell is described in Sect. 2.3).

Local-to-Local Translation (L2L). Similarly to the M2M, moments at a point s_{el} (τ_{el}) can be efficiently recalculated for a nearby point s'_{el} (corresponding to the complex point τ'_{el}) (presented in Fig. 2). For this purpose, a transformation called local-to-local (L2L) translation (similar to M2M) is used [20]:

$$\begin{aligned} L_l^U(\tau'_{el},\tau_c) &= (-1)^l\left\{\sum_{k=0}^{\infty}\sum_{m=l}^{\infty}\frac{(k+m-1)!\cdot M_k(\tau_c)}{(\tau_{el}-\tau_c)^{k+m}}\cdot\frac{(\tau'_{el}-\tau_{el})^{m-l}}{(m-l)!}\right\}, \\ L_l^P(\tau'_{el},\tau_c) &= (-1)^l\cdot\left\{\sum_{k=1}^{\infty}\sum_{m=l}^{\infty}\frac{(k+m-1)!\cdot N_k(\tau_c)}{(\tau_{el}-\tau_c)^{k+m}}\cdot\frac{(\tau'_{el}-\tau_{el})^{m-l}}{(m-l)!}\right\}. \end{aligned} \tag{10}$$

2.2 Tree Structure in the Fast PIES

In classic FMM, the binary tree for 1D, quad-tree for 2D and octa-tree for 3D problems are applied. In this paper, 2D problems are discussed, hence in classic FMM implementation quad-tree presented in Fig. 3 [6] is used.

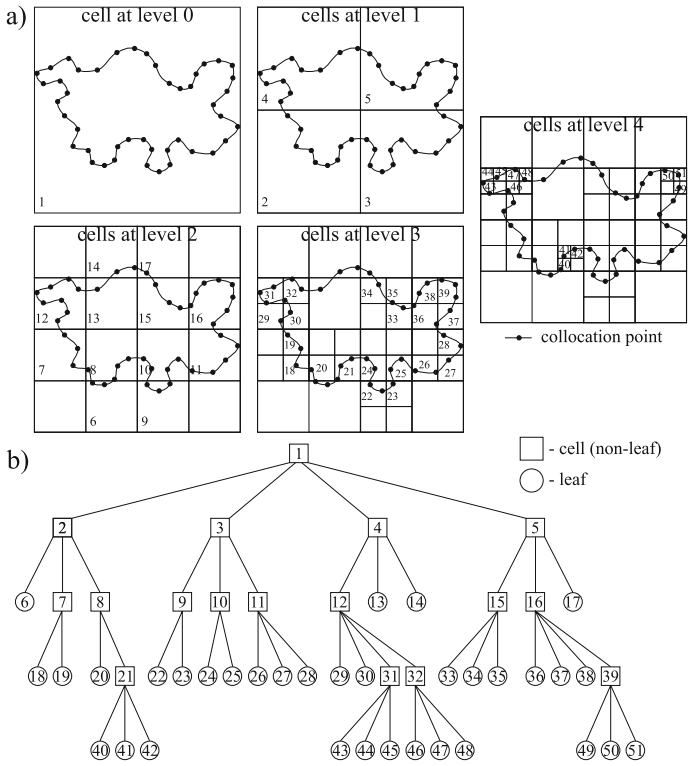

Fig. 3. a) Constructing the FMM tree in the FM-BEM, b) structure of obtained quad-tree for 2D problem

A square surrounding the entire domain of the problem (Fig. 3a) is called level 0 cell. This cell is the parent of four cells of level 1 (so-called children cells) obtained as a result of dividing the parent cell into four identical squares. Only the cell crossing the boundary of the problem is considered to be a child cell. The parent cell of level l ($l \geq 0$) is divided into children cells of a level $l + 1$. The division is performed until a predetermined number of elements are inside a cell (in the example in Fig. 3a - max. 2 elements whose centre is marked as a collocation point) or the assumed maximum level l has been reached. A cell without a child is called a leaf. The quad-tree presented in Fig. 3b is obtained in the described way. The presented method of the FMM tree construction is applied, among others, in the FM-BEM [6].

In the PIES applied for the 2D problems, it is possible to implement a suitably modified and improved binary tree. The improvement is related to the way of defining problems in the PIES - the boundary of the problem is defined in the 1D parametric reference system (presented in Fig. 1). It is a different way of modelling the shape of a boundary than in the BEM. Also, the number of segments describing the boundary in the PIES is smaller than the number of

elements in the BEM (see Fig. 4). It is connected with the way of defining the shape of the boundary in the PIES [10].

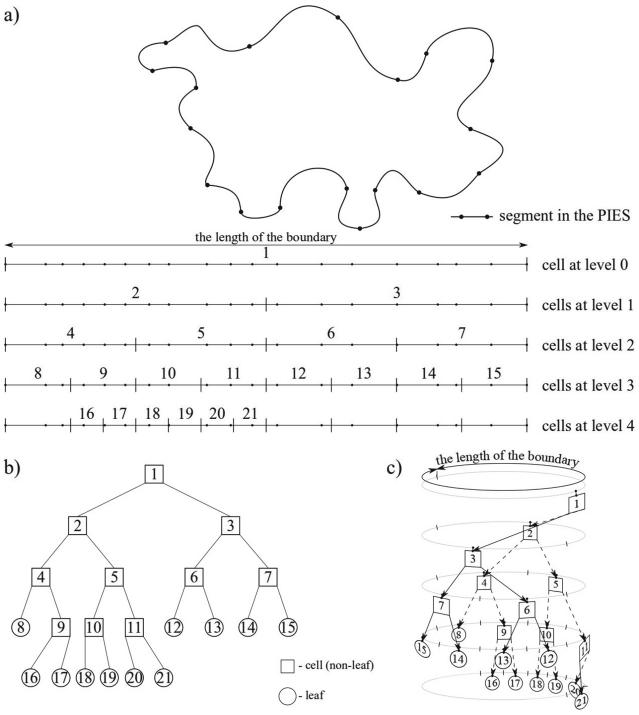

Fig. 4. a) Constructing the FMM tree in the PIES, b) classic binary tree for the 1D problem, c) modified binary tree in the PIES for the 2D problem

The proposed binary tree structure assumes that the boundary is described in the 1D reference system. A level 0 cell covers the entire boundary of the problem (presented in Fig. 4a). This cell is the parent of two children cells of level 1 obtained as a result of dividing the parent cell into two identical segments. The parent cell of level l ($l \geq 0$) is divided into children cells of a level $l + 1$. The division is performed until a predetermined number of segments are inside a cell (in the example in Fig. 4a - max. 2 segments) or the assumed maximum level l has been reached. In the classic FMM, the binary tree for 1D problems has the form presented in Fig. 4b. However, in the PIES 2D problem is reduced to the 1D parametric reference system, hence the first and the last boundary segment are very close to each other. Therefore, we propose to close the binary tree in the form presented in Fig. 4c. For all levels, the first and the last cells are treated as adjacent.

2.3 Algorithm of Solving the Fast PIES

The algorithm of the PIES with modified kernels proceeds in several steps (flow chart presented in Fig. 5). The first step after initialization of the algorithm is to determine the structure of the modified binary tree. Then right-hand sided vector $\mathbf{b}$ is computed and GMRES is called to find solution of $\mathbf{Ax} = \mathbf{b}$. These procedures uses the modified fast multipole algorithm presented in Algorithm 1. At the end of algorithm results are presented on the screen and written to the output file.

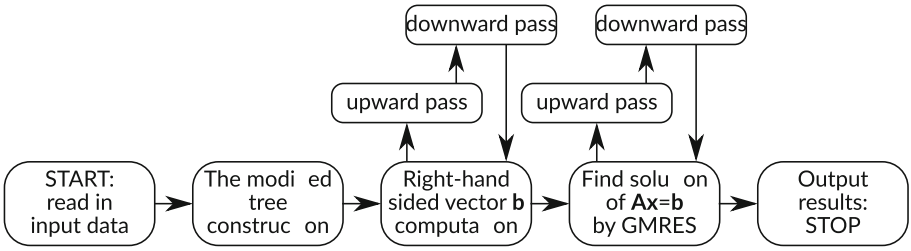

Fig. 5. Flow chart for the fast PIES

The first run of the fast multipole procedure is used for calculating the right-hand sided vector $\mathbf{b}$. The fast multipole procedure is composed of two steps: the upward pass and the downward pass. At the upward pass, all moments in leaves are calculated (line 4 in Algorithm 1) for the level *lev*. Then, tracing the tree structure upward, moments in all parent cells are calculated up to level 2 using M2M (line 6).

In the downward pass, we trace the tree structure downward, and previously calculated moments are used in calculations. First of all, we should remind cells neighbourhood to clarify this step [20]. Two cells are *adjacent* at level i, if they have a common end at level i. Two cells are $well-separated$ at level i, if they are not adjacent at level i, but their parent cells are adjacent at level $i-1$. The *interaction list* of cell $K-th$ is the list of cells $well-separated$ from cell $K-th$ at level i. At last, two cells are far cells, if their parent cells are not adjacent.

In modelling complex shapes of the boundary, we should modify the FMM algorithm described in [20] due to the possibility of too small distance between cells with τ_c and τ_{el} points, hence the assumption of $|\tau_{col} - \tau_{el}| \ll |\tau_c - \tau_{el}|$ used for the M2L translation is not fulfilled. The authors of this work propose to modify the algorithm by marking such cells as *adjacent* and not to enter them on *interaction list* (calculations are performed as in the case of *adjacent* cells). Including this modification to the fast PIES, solutions with the same accuracy as in conventional PIES are obtained.

Starting from level 2 and tracing the tree structure downward to all leaves coefficients of local expansion (the line 24 in the Algorithm 1) are computed. Coefficients at cell $K-th$ at level i are computed as the sum of two elements:

Algorithm 1. Modified fast multipole procedure

Require:
lev - the number of tree levels
$\min c_i$ - the lowest number of a cell on the level i
$\max c_i$ - the highest number of a cell on the level i
//upward pass
1: **for** $i \leftarrow lev$ to 2 **do**
2: **for** $icell \leftarrow \min c_i$ to $\max c_i$ **do**
3: **if** $i == lev$ **then**
4: $multipole\_moments(icell)$
5: **else**
6: $moment\_expansion(icell)$
7: **end if**
8: **end for**
9: **end for**
//downward pass
10: **for** $i \leftarrow lev$ to 2 **do**
11: **for** $icell \leftarrow \min c_i$ to $\max c_i$ **do**
12: **if** $i \neq 2$ **then**
13: $local\_to\_local(icell)$
14: **end if**
15: **for** $jcell \leftarrow \min c_i$ to $\max c_i$ **do**
16: **if** $parent(icell)$ & $parent(jcell)$ are neighbours **then**
17: **if** ($cell(icell)$ & $cell(jcell)$ are neighbours) || !($|\tau_{col} - \tau_{jcell}| \ll |\tau_{icell} - \tau_{jcell}|$) **then**
18: $direct(icell, jcell)$
19: **else**
20: $moment\_to\_local(icell, jcell)$
21: **end if**
22: **end if**
23: **end for**
24: $local\_expansion(icell)$
25: **end for**
26: **end for**

contributions from all far cells (computed using L2L - the line 13) and from the cells in the interaction list of cell $K - th$ (computed using M2L - the line 20). There are no far cells to a cell K at level 2, therefore only M2L is used to compute coefficients. Contributions from adjacent cells of leaf K at the lowest level are computed directly (line 18), as in conventional PIES. Finally, the FMM procedure produces a right-hand vector $\mathbf{b}$.

To solve the system of algebraic equations $\mathbf{Ax} = \mathbf{b}$, iterative GMRES solver is used. The method requires the application of multiplication of the matrix $\mathbf{A}$ by the vector of unknowns $\mathbf{x}$, therefore the solver can be directly integrated with the fast PIES. The FMM in GMRES is performed in the same way as for vector $\mathbf{b}$.

3 Tests of the Fast PIES with Modified Binary Tree

In the paper [20], the authors presented preliminary results of the fast PIES solutions. However, solving the problem presented in this paper by the fast PIES without modification of the binary tree gives unsatisfactory results. Obtained solutions were far from expected.

The problem of temperature distribution (modelled by Laplace's equation) in heat-sink is considered. The shape of the boundary and boundary conditions are shown in Fig. 6. The boundary is composed of 716 linear segments. The same number of collocation points (from 4 to 8) is defined on each segment, and finally, we should solve the system of 2864 to 5728 algebraic equations. The problem is solved by conventional and fast PIES. PC based on Intel Core i5-4590S with 8 GB RAM and g++ 7.4.0 compiler with -O2 optimization on 64-bit Linux operation system (Ubuntu, kernel 5.0.0) and LAPACK 3.10.3 library [22] is used during tests.

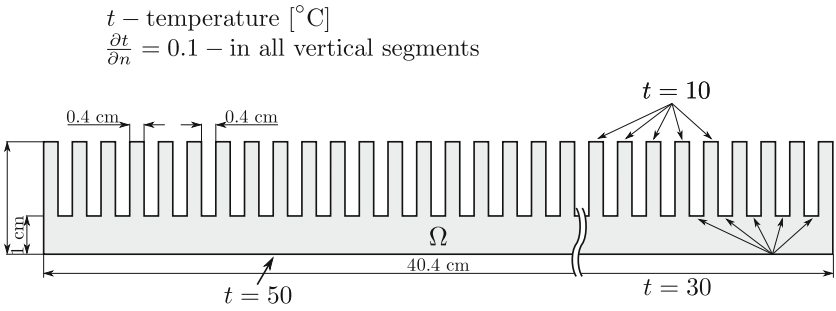

Fig. 6. The shape of considered heat-sink

First of all, a comparison of CPU time and RAM utilization between a different number of the modified tree levels in the fast PIES is performed to find the optimal value of tree levels. Taylor series composed of 25 terms approximated the fast PIES kernels. The value of tolerance (convergence criterion) of the GMRES was equal to 10^{-8}.

As can be seen from Fig. 7, both CPU time and memory utilization decrease with the growing number of tree levels reaching the minimum value for a tree with 5–6 levels regardless of the number of collocation points. In our studies, the tree with 6 levels is adopted.

The research involved a comparison of the speed and RAM utilization between conventional, the fast PIES and the fast multipole BEM (application from [6]). In the fast multipole BEM two meshes are used: 2864 and 5728 elements. The value of tolerance (convergence criterion) of the GMRES and the number of terms in Taylor series are the same as previous, i.e. 10^{-8} and 25 respectively. Solutions are presented for two versions of conventional PIES

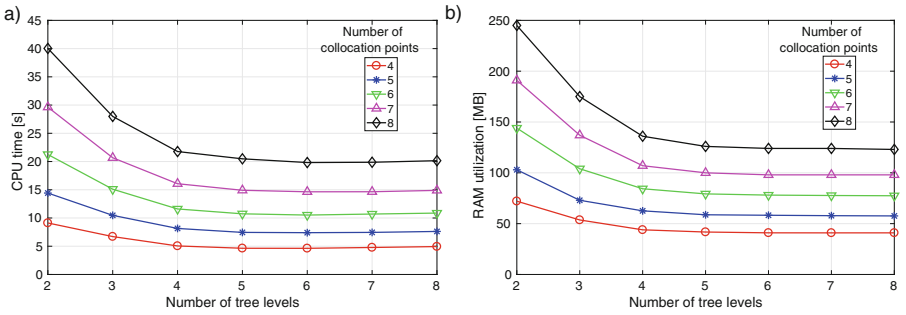

Fig. 7. Influence of the number of tree levels on a) computational time, b) RAM utilization of the fast PIES

(PIES_d with direct solver (Gaussian elimination method) and PIES_i with iterative solver GMRES), the fast PIES (fPIES) and the fast multipole BEM (the fmBEM). The accuracy of the solutions is calculated as the mean square error (MSE) between the results of conventional (PIES_d) and the fast PIES (P-P) and the fast PIES and the fast multipole BEM (P-B).

Table 1. Comparison of computational time and RAM utilization between the fast multipole BEM, conventional and the fast PIES with modified binary tree.

Number of		CPU time [s]				RAM utilization [MB]				MSE	
col. pts	eqs	fPIES	PIES_d	PIES_i	fmBEM	fPIES	PIES_d	PIES_i	fmBEM	P-P	P-B
4	2864	4.63	30.10	27.82	9.56	40.96	254	175	105.9	2.31e−10	0.051
5	3580	7.28	53.58	45.07	–	58.25	395	207	–	1.54e−10	–
6	4296	10.43	94.17	67.20	–	78.08	567	297	–	2.04e−10	–
7	5012	14.44	140.94	98.49	–	98	770	398	–	2.29e−10	–
8	5728	19.69	214.57	128.66	46.18	124	1005	517	107.5	2.18e−10	0.063

As can be seen from Table 1, the fast PIES is significantly faster than both versions of conventional PIES and about 2 times faster than the fast multipole BEM. The fast PIES also needs up to 4 times less RAM than the PIES_i and about 8 times less than the PIES_d. Obtained solutions are practically the same as in conventional versions of the PIES - MSE between the PIES_d and the fast PIES does not exceed $2.31 \cdot 10^{-10}$. The MSE between the fast PIES and the fast multipole BEM has higher value. However, our previous studies proved that the PIES is more accurate than the BEM.

Graphical comparison of CPU time and RAM utilization between all the PIES methods is presented in Fig. 8.

4 Conclusions

The paper presents the fast PIES based on the FMM with the modified binary tree in solving potential BVPs with complex shapes. The proposed modified

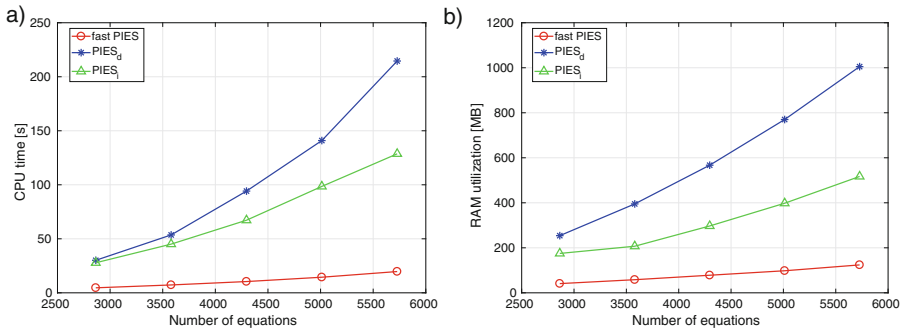

Fig. 8. Comparison of a) CPU time, b) RAM utilization between conventional and the fast PIES

binary tree is built based on the 1D parametric reference system. Such modification of the tree in the fast PIES allows obtaining very accurate solutions for engineering problems with complex shapes on standard PC in a short computational time. It also allows solving problems in which the condition of the appropriate distance between the centres of the leaves is not fulfilled.

The numerical test shows a reduction of the computation time and RAM utilization of the fast PIES compared to conventional ones as well as the fast multipole BEM. The speed-up of computations between the fast and conventional PIES increases with the size of solving problem, while the accuracy of solutions is almost the same.

Acknowledgments. The scientific work is founded by resources of Ministry of Science and Higher Education for research, granted to the Institute of Informatics, University of Bialystok.

References

1. Rokhlin, V.: Rapid solution of integral equations of classical potential theory. J. Comput. Phys. **60**(2), 187–207 (1985)
2. Greengard, L.F., Rokhlin, V.: A fast algorithm for particle simulations. J. Comput. Phys. **73**(2), 325–348 (1987)
3. Greengard, L.F.: The Rapid Evaluation of Potential Fields in Particle Systems. The MIT Press, Cambridge (1988)
4. Blankrot, B., Leviatan, Y.: FMM-accelerated source-model technique for many-scatterer problems. IEEE Trans. Antennas Propag. **65**(8), 4379–4384 (2017)
5. Nishimura, N.: Fast multipole accelerated boundary integral equation methods. Appl. Mech. Rev. **55**(4), 299–324 (2002)
6. Liu, Y.J., Nishimura, N.: The fast multipole boundary element method for potential problems: a tutorial. Eng. Anal. Boundary Elem. **30**(5), 371–381 (2006)
7. Brebbia, C.A., Telles, J.C.F., Wrobel, L.C.: Boundary Element Techniques, Theory and Applications in Engineering. Springer, New York (1984). https://doi.org/10.1007/978-3-642-48860-3

8. Zienkiewicz, O.C.: The Finite Element Method. McGraw-Hill, London (1977)
9. Perazzo, F., Lohner, R., Perez-Pozo, L.: Adaptive methodology for meshless finite point method. Adv. Eng. Softw. **39**(3), 156–166 (2008)
10. Zieniuk, E.: Hermite curves in the modification of integral equations for potential boundary-value problems. Eng. Comput. **20**(1–2), 112–128 (2003)
11. Zieniuk, E., Kapturczak, M.: Modeling the shape of boundary using NURBS curves directly in modified boundary integral equations for Laplace's equation. Comput. Appl. Math. **37**(4), 4835–4855 (2018)
12. Zieniuk, E., Bołtuć, A., Kużelewski, A.: Algorithms of identification of multi-connected boundary geometry and material parameters in problems described by Navier-Lamé equation using the PIES. In: Pejaś, J., Saeed, K. (eds.) Advances in Information Processing and Protection, pp. 409–418. Springer, Boston (2007). https://doi.org/10.1007/978-0-387-73137-7_37
13. Zieniuk, E., Szerszeń, K.: Triangular Bézier patches in modelling smooth boundary surface in exterior Helmholtz problems solved by PIES. Arch. Acoust. **34**(1), 51–61 (2009)
14. Zieniuk, E., Kużelewski, A., Kapturczak, M.: The influence of interval arithmetic on the shape of uncertainly defined domains modelled by closed curves. Comput. Appl. Math. **37**(2), 1027–1046 (2016). https://doi.org/10.1007/s40314-016-0382-0
15. Zieniuk, E., Kapturczak, M., Kużelewski, A.: Modification of interval arithmetic for modelling and solving uncertainly defined problems by interval parametric integral equations system. In: Shi, Y., et al. (eds.) ICCS 2018. LNCS, vol. 10862, pp. 231–240. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93713-7_19
16. Zieniuk, E., Sawicki, D., Bołtuć, A.: Parametric integral equations systems in 2D transient heat conduction analysis. Int. J. Heat Mass Transf. **78**, 571–587 (2014)
17. Kużelewski, A., Zieniuk, E.: OpenMP for 3D potential boundary value problems solved by PIES. In: 13th International Conference of Numerical Analysis and Applied Mathematics ICNAAM 2015, AIP Conference Proceedings, vol. 1738, Article number 480098. AIP Publishing LLC (2016)
18. Kuzelewski, A., Zieniuk, E., Boltuc, A.: Application of CUDA for acceleration of calculations in boundary value problems solving using PIES. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Waśniewski, J. (eds.) PPAM 2013. LNCS, vol. 8385, pp. 322–331. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55195-6_30
19. Kużelewski, A., Zieniuk, E., Kapturczak, M.: Acceleration of integration in parametric integral equations system using CUDA. Comput. Struct. **152**, 113–124 (2015)
20. Kużelewski, A., Zieniuk, E.: The fast parametric integral equations system in an acceleration of solving polygonal potential boundary value problems. Adv. Eng. Softw. **141**, 102770 (2020)
21. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. **7**, 856–869 (1986)
22. Anderson, E., et al.: LAPACK Users' Guide. Society for Industrial and Applied Mathematics, Philadelphia (1999)