 CMES, vol.104, no.3, pp.227-249, 2015

A Parallel Boundary Element Formulation for Tracking Multiple Particle Trajectories in Stoke's Flow for Microfluidic Applications

Z. Karakaya[1], B. Baranoğlu[2], B. Çetin[3] and A. Yazici[4]

Abstract: A new formulation for tracking multiple particles in slow viscous flow for microfluidic applications is presented. The method employs the manipulation of the boundary element matrices so that finally a system of equations is obtained relating the rigid body velocities of the particle to the forces applied on the particle. The formulation is specially designed for particle trajectory tracking and involves successive matrix multiplications for which SMP (Symmetric multiprocessing) parallelisation is applied. It is observed that present formulation offers an efficient numerical model to be used for particle tracking and can easily be extended for multiphysics simulations in which several physics involved.

Keywords: Boundary Element Method, particle tracking, Stoke's flow, parallel computing.

1 Introduction

Particle tracking problems have many important applications. Especially, with the recent developments in microfluidics technology, tracking several particles, especially with external forces acting (such as forces arising from the application of an electric, acoustic or optic field) gained special importance [Cetin, Ozer, and Solmaz (2014)]. For an efficient design of microfluidic systems, the prediction of the particle tracks is crucial. Particle trajectory is the result of the interaction of the particle with the external fields present. One approach to model the particle trajectory within the microchannel is the stress tensor approach [Cetin and Li (2011)]. In this approach, the field variables are solved with the presence of the finite-sized particle. The resultant force on the particle can be obtained by integrating the appropriate stress tensor on the particle surface. In each incremental movement of the

[1] Computer Engng. Dept., Atilim University, Turkey.
[2] Manufacturing Engng. Dept., Atilim University, Turkey.
[3] Microfluidics & Lab-on-a-chip Research Group, Mech. Engng. Dept., Bilkent University, Turkey.
[4] Software Engng. Dept., Atilim University, Turkey.

particle, the field variables need to be resolved. In many studies, this approach has also been successfully implemented [Ai, Joo, Jiang, Xuan, and Qian (2009); Ai, Mauroy, Sharma, and Qian (2011)] to explore the nature of the particle flow within a microchannel. A rigorous simulation of the particle motion utilising tensor approach requires massive remeshing. For methods involving domain discretisation, such as finite element method (FEM) or finite volume method (FVM) not only the remeshing process is computationally expensive, but also at each remeshing step, some interpolating algorithms relating the field variables in the new mesh in terms of the variables of the old mesh are required which cause some loss in the accuracy. Moreover, the determination of the forces induced on the particles requires the calculation of gradient of the field variables. Therefore, for an accurate calculation of gradient of field variables, fine mesh is required on and within the close neighbourhood of the particle surface. Due to the computationally expensive nature, only 2D models with relatively coarse mesh and the motion of single particle have been worked on using FEM. To overcome the remeshing problem for the simulation of particulate flow at macroscale, immersed boundary method [Fogelson and Peskin (1988)] and fictitious domain method [Glowinski, Pan, Hesla, Joseph, and Periaux (2001)] have been proposed and implemented. Although these methods are computationally very efficient, to model the particle-particle interaction, some contact modelling is required which has a resolution that cannot be accepted for the simulation at microscale. Moreover, these methods are well established for flow simulations, but very rare studies exist for the coupling of flow with the electrical, magnetic and/or acoustic fields [Langthjem and Nakano (2013)].

Boundary element method (or Boundary integral method) is a numerical tool that is well applied to many problems in engineering. The boundary element method (BEM) possesses several advantages over other numerical methods (such as the finite element method, FEM, or the finite volume method, FVM) most important of which can be listed as: (i) BEM has boundary only discretisation, which results in less computational effort in meshing and remeshing, (ii) In BEM formulations, continuity and compatibility conditions for the governing equations are satisfied exactly (not approximately) within the solution domain (iii) In BEM, the derivative quantities e.g. the flux or stress/strain within the solution domain is determined by analytical differentiation; therefore, such quantities can be determined more accurately in the BEM when compared with the numerical methods that involve domain discretisation. (iv) When the solution domain extends to infinity in one or more directions, such as the case in external problems or half-space problems, the BEM is readily applicable - there is no need for truncation of boundaries.

Considering the microchannel networks within the LOC devices, typical flow speed is low (resulting in very low Reynolds number) and the inertia forces are negligible

(in magnitude) when compared with the pressure or the viscous forces. The flow is Stoke's flow which governs by a set of linear partial differential equations. Linear equations are suitable for Boundary Element Method (BEM). Since the BEM does not require meshing within the flow region and the exact calculation of the gradient of the field variables, it is preferable for particle tracking in a microchannel. Referring these advantages, recently, Dustin and Luo [House and Luo (2010, 2011)] implemented the BEM to simulate the particle trajectory within a microchannel under the action of electrophoretic and electro-osmotically driven flow field. With the BEM, Stoke's flow over a slowly moving particle [Sellier (2012)] and later over a fixed or freely suspended particle [Sellier (2013)] is studied with the fast calculation of the hydrodynamic net force and moment over the arbitrary shaped particle.

In this study, a formulation to track the particles in a microchannel under the action of pressure driven flow is presented. The method involves the reorganisation of the BE matrices that are evaluated for the flow problem, and through several manipulations, reducing the problem to a linear system of equations where the only unknowns are the motion parameters (e.g. the translational and rotational velocities of the centers of gravity) of the particles. With such manipulation, the dimension of the linear system of equations to be solved is reduced drastically, resulting in a comparably fast solution. Also, this formulation makes it very simple to apply external forces (of any nature) to the particle(s) in motion. Inversely, the computation of the drag force when the velocity components are known is also obtained easily. Since the formulation depends on excessive matrix multiplications (instead of solution of the total system), it provides a promising future in parallelisation, especially with the new developments in CPU and GPU architectures. In this study, the parallelisation of the algorithm is performed using SMP (Symmetric multiprocessing) [Wilkinson and Allen (2005)] parallelisation techniques applied by using OpenMP [OpenMP (2014)] preprocessor directives, running on NUMA (Non-uniform Memory Access) [NUMA (2014)] architecture computer.

In an overall look, the present formulation offers an efficient numerical model to be used for the simulation of the particle trajectory for microfluidic applications and can easily be extended for multiphysics simulations.

2 Boundary element formulation for Stoke's flow and boundary conditions for moving particle

The governing equations (GE) for the Stoke's flow is given as [Wrobel (2002)]

$$-\nabla P + \mu \nabla^2 u = 0 \tag{1}$$

along with the continuity condition

$$u_{i,i} = 0 \tag{2}$$

where P is the modified pressure defined in terms of the pressure, p, the gravitational acceleration, $\mathbf{g}$, the density ρ and the viscosity μ of the fluid, and the cartesian coordinates $\mathbf{x}$ as

$$P = p - \rho \mathbf{g} \cdot \mathbf{x} \tag{3}$$

It should be noted that in all equations presented in this study, unless otherwise is explicitly stated, summation convention is in place requiring a summation over a repeated index in the range of that index. The boundary element formulation of the Stoke's flow is given in several references [Pozrikidis (1999), Wrobel (2002)]

$$C_{ij}(A)u_j(A) = \int_S G_{ij}(A,P)t_j(P)\mathrm{d}S - \int H_{ij}(A,P)u_j(P)\mathrm{d}S \tag{4}$$

In this equation, A represents the fixed (evaluation) point and P represents the varied (integration) point. Note that P is on the boundary of the solution region, whereas A can be in the solution domain, on the boundary, or outside the domain. C_{ij} takes values 1, if A is in the solution domain, $\frac{1}{2}$ if it is on a smooth boundary or 0 if it is outside the solution domain. The field variables of Equation (4) are the velocity components, u_i and the traction components t_i and G_{ij} and H_{ij} are the first and second fundamental solutions of Stoke's equation. In the context of this study, since it can be thought of as a post-processing for the problems in hand, the integral equation for pressure is not considered.

In this study, the boundary discretisation is performed using constant triangular elements. After discretisation, a matrix relation as

$$\mathbf{H} \cdot \mathbf{u} = \mathbf{G} \cdot \mathbf{t} \tag{5}$$

is obtained, where the elements of the matrices $\mathbf{G}$ and $\mathbf{H}$ are evaluated through the integral relations

$$\begin{aligned} G_{ij}^{lk} &= \int_{C_k} G_{ij}(A_l, P_k)\mathrm{d}S \\ H_{ij}^{lk} &= \int_{C_k} H_{ij}(A_l, P_k)\mathrm{d}S \end{aligned} \tag{6}$$

where l refers to the element that the fixed point A is on, and k refers to the element being integrated. Note that, for 3D discretisation for a fixed l and k, G_{ij}^{lk} and H_{ij}^{lk} refers to the components of 3×3 matrices. The matrix relation in Equation (5) constitutes to a $3N$ equations of $6N$ unknowns, $3N$ of which is to be determined by given boundary conditions.

The imposition of boundary conditions requires the definition of one and only one of the couples (u_i^n, t_i^n) or a combination of these two at all points of the defined boundary, where n represents the node number and i represents the direction. In the context of this study, we will consider the motion of undeformable particles, therefore we employ the rigid-body motion conditions. Thus, for any point on the particle, the velocity vector can be obtained through

$$\mathbf{u} = \mathbf{u}^B + \omega \times \mathbf{r} \tag{7}$$

where $\mathbf{u}$ is the velocity of the point on the particle, $\mathbf{u}^B$ is the velocity of the selected center of the particle, ω is the rotational velocity vector and $\mathbf{r}$ is the relative position vector of the particle point to the selected center of the particle. The imposition of Equation (7) relates all velocity components to six new parameters (per particle): three components of the translational velocity of the center of the particle and three components of the rotational velocity vector. Therefore, this condition increases the size of the matrix system by six-per-particle. Therefore, six new equations per particle are needed to make the system of equations solvable. These six equations come from the force equilibrium on the particle

$$f_i^B = \sum_{n=1}^{M} f_n = \sum_{n=1}^{M} \int_{C_n} t_i(P_n) \mathrm{d}S = \sum_{n=1}^{M} t_i^n A_n \tag{8}$$

and the moment equilibrium

$$\mathbf{m}^B = \sum_{n=1}^{M} \mathbf{r} \times \mathbf{f}^n \tag{9}$$

where M is the number of elements on the particle.

2.1 Impedance formulation

The major drawback of the conventional formulation (which is presented above) is the repeated solution of a considerably large linear system of equations. Although a very large part of the coefficient matrices which corresponds to non-moving boundary does not need to be re-evaluated, as the particle moves. Parts of the coefficient matrices that are related with particle has to be updated and the system is to be re-solved. In majority of the applications regarding particle tracking, the main focus

is on obtaining the trajectory of the particle(s). The determination of field variables on the boundary or in the solution domain does not have a significant importance. With this note in place, we present a new formulation for tracking multiple particles, the impedance formulation, which is based on a similar formulation derived by Mengi and co-workers in several studies [Mengi and Argeso (2006); Argeso and Mengi (2013); Yalcin and Mengi (2013)]. For this, we express the rigid body motion of a particle in matrix form as:

$$\mathbf{u}^P = \mathbf{M} \cdot \mathbf{u}^B \tag{10}$$

where $\mathbf{u}^P$ is a column vector containing the velocity values at the nodes of the particle which is organised as

$$\mathbf{u}^P = \left\{ \left\{ \begin{array}{ccc} u_1 & u_2 & u_3 \end{array} \right\}^1 \left\{ \begin{array}{ccc} u_1 & u_2 & u_3 \end{array} \right\}^2 \cdots \left\{ \begin{array}{ccc} u_1 & u_2 & u_3 \end{array} \right\}^N \right\}^{\mathbf{T}} \tag{11}$$

and $\mathbf{u}^B$ is the vector containing the center velocity of the particle:

$$\mathbf{u}^B = \left\{ \begin{array}{cccccc} u_1^B & u_2^B & u_3^B & \omega_1^B & \omega_2^B & \omega_3^B \end{array} \right\}^{\mathbf{T}} \tag{12}$$

and the coefficient matrix $\mathbf{M}$ is of size $(3N \times 6)$ whose elements are obtained through the Equation (7). Similarly, defining a combined resultant force-moment vector for the particle as:

$$\mathbf{f}^B = \left\{ \begin{array}{cccccc} f_1^B & f_2^B & f_3^B & m_1 & m_2 & m_3 \end{array} \right\}^{\mathbf{T}} \tag{13}$$

and the traction vector defined as:

$$\mathbf{t}^P = \left\{ \left\{ \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \right\}^1 \left\{ \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \right\}^2 \cdots \left\{ \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \right\}^N \right\}^{\mathbf{T}} \tag{14}$$

a matrix relation as:

$$\mathbf{f}^B = \mathbf{F} \cdot \mathbf{t}^P \tag{15}$$

can be obtained. Here, the $(6 \times 3N)$ matrix $\mathbf{F}$ is obtained through the relations given in Equation (8) and Equation (9). At this point, we make a partitioning in the system of equations such as:

$$\left[\begin{array}{cc} \mathbf{H}_{00} & \mathbf{H}_{0P} \\ \mathbf{H}_{P0} & \mathbf{H}_{PP} \end{array} \right] \left\{ \begin{array}{c} \mathbf{u}^0 \\ \mathbf{u}^P \end{array} \right\} = \left[\begin{array}{cc} \mathbf{G}_{00} & \mathbf{G}_{0P} \\ \mathbf{G}_{P0} & \mathbf{G}_{PP} \end{array} \right] \left\{ \begin{array}{c} \mathbf{t}^0 \\ \mathbf{t}^P \end{array} \right\} \tag{16}$$

where the index 0 refers to the non-moving boundary and P refers to moving boundary - thus $0P$ refers to components that are evaluated when the fixed point is on

non-moving boundary and the integration is done over the moving boundary, etc. We will assume that all components $\mathbf{u}^0$ are known and $\mathbf{t}^0$ are unknown, easily obtainable when necessary column changes are performed with given boundary conditions. The first line of Equation (16) requires:

$$\mathbf{H}_{00}\mathbf{u}^0 + \mathbf{H}_{0P}\mathbf{u}^P = \mathbf{G}_{00}\mathbf{t}^0 + \mathbf{G}_{0P}\mathbf{t}^P \tag{17}$$

which leads to

$$\mathbf{t}^0 = \mathbf{G}_{00}^{-1}\left(\mathbf{H}_{00}\mathbf{u}^0 + \mathbf{H}_{0P}\mathbf{u}^P - \mathbf{G}_{0P}\mathbf{t}^P\right) \tag{18}$$

Inserting this to the second row equation of Equation (16), we get

$$\mathbf{B}\cdot\mathbf{t}^P = \mathbf{A}\cdot\mathbf{u}^P + \mathbf{C}\cdot\mathbf{u}^0 \tag{19}$$

where

$$\begin{aligned} \mathbf{A} &= \left[\mathbf{H}_{PP} - \mathbf{G}_{P0}\mathbf{G}_{00}^{-1}\mathbf{H}_{0P}\right] \\ \mathbf{B} &= \left[\mathbf{G}_{PP} - \mathbf{G}_{P0}\mathbf{G}_{00}^{-1}\mathbf{G}_{0P}\right] \\ \mathbf{C} &= \left[\mathbf{H}_{P0} - \mathbf{G}_{P0}\mathbf{G}_{00}^{-1}\mathbf{H}_{00}\right] \end{aligned} \tag{20}$$

Inserting equations (10) and (15) to Equation (19) and defining

$$\begin{aligned} \mathbf{K} &= \mathbf{F}\mathbf{B}^{-1}\mathbf{A}\mathbf{M} \\ \mathbf{b} &= \mathbf{F}\mathbf{B}^{-1}\mathbf{C}\mathbf{u}^0 \end{aligned} \tag{21}$$

we obtain

$$\mathbf{K}\mathbf{u}^B = \mathbf{f}^B - \mathbf{b} \tag{22}$$

It is important to note here:

- Equation (22) is a system of linear equations where $\mathbf{K}$ is a square matrix of size $(6P \times 6P)$, and $\mathbf{u}^P$, $\mathbf{f}^B$ and $\mathbf{b}$ are column vectors of size $6P$ where P is the number of particles in the system.
- Inverting a considerably large matrix, $\mathbf{G}_{00}$ to obtain $\mathbf{G}_{00}^{-1}$, just once (at the beginning of the analysis), the rest of the formulation contains matrix multiplications and an inversion of the considerably very small matrix $\mathbf{B}$ which is of size $(3M \times 3M)$ (where M is the total number of nodes in all particles) to get $\mathbf{B}^{-1}$.

- The formulation readily involves the external forces to the system. This allows direct imposition of external forces found from different analyses involving different type of problems (electromagnetic, acoustic, etc.).

- The final form is obtained with matrix multiplications, for which parallelisation is not only simple, but also effective.

- A special case of Equation (22) is when there is no non-moving boundary, the case that arises in infinite medium. In such case, $\mathbf{b} = \mathbf{0}$ and $\mathbf{A} = \mathbf{H}_{PP} = \mathbf{H}$, $\mathbf{B} = \mathbf{G}_{PP} = \mathbf{G}$. The solution can be evaluated through

$$\begin{aligned} &\mathbf{K}\mathbf{u}^B = \mathbf{f}^B \\ &\mathbf{K} = \mathbf{F}\mathbf{G}^{-1}\mathbf{H}\mathbf{M} \end{aligned} \tag{23}$$

- Another comment can be placed concerning the deformable particles. The determination of the deformation mainly would require accurate evaluation of the surface tractions on the boundary of the particle(s). The formulation in hand, however, presents only the body displacements, but is is possible to obtain the boundary tractions with small modifications on the formulation. Recall that, it was assumed that all $\mathbf{u}^0$ are known (if not, necessary column changes between the system matrices $\mathbf{G}$ and $\mathbf{H}$ are performed to obtain this). The following steps would be recommended for a fast evaluation of the boundary tractions on the particles(s) after the evaluation of the rigid body velocities, $\mathbf{u}^B$:

  - Find $\mathbf{u}^P$ using Equation (10)
  - With the previously calculated $\mathbf{B}^{-1}$, $\mathbf{A}$ and $\mathbf{C}$, re-write Equation (19) to obtain the boundary tractions on particle(s) as

$$\mathbf{t}^P = \mathbf{B}^{-1} \cdot (\mathbf{A} \cdot \mathbf{u}^P + \mathbf{C} \cdot \mathbf{u}^0) \tag{24}$$

In the present study, time integration is performed in an explicit sense, using forward Euler difference, for simplicity. For more rigorous analysis, other time integration schemes, both explicit and implicit can be adopted.

3 Verification of the formulation

The developed formulation has been tested for some 2D benchmark problems, and good agreement has been achieved by analytical and FEM solutions. The details of 2D verification can be found elsewhere [Baranoglu and Cetin (2014)]. For the verification of the 3D formulation, the flow in a square channel with width of W

Table 1: BEM results compared with analytical solution for channel flow.

x	$Analytical$	BEM 4400 elem.	$\%Error$	BEM 11264 elem.	$\%Error$
0	0.0	0.0		0.0	
10	82.6	84.1	1.76	83.5	1.09
20	141.4	143.8	1.70	142.9	1.06
30	180.3	183.4	1.71	182.2	1.07
40	202.5	205.9	1.66	204.6	1.03
50	209.6	213.2	1.70	211.8	1.06
60	202.5	205.9	1.66	204.6	1.03
70	180.3	183.4	1.71	182.2	1.07
80	141.4	143.8	1.70	142.9	1.06
90	82.6	84.1	1.76	83.5	1.09
100	0.0	0.0		0.0	

and length of L is analysed as a first benchmark. The analytical solution can be obtained by using integral transform techniques as:

$$u(x,y) = \frac{16}{W^2}\frac{\Delta P}{\mu L}\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\frac{\sin(\beta_m x/W)\sin(\lambda_n y/W)}{(\beta_m^2+\lambda_n^2)\beta_m\lambda_n} \tag{25}$$

where β_m and λ_n's are the eigenvalues defined as:

$$\beta_m = (2m-1)\pi/W, \quad \lambda_n = (2n-1)\pi/W \tag{26}$$

In the analysis, the square channel is taken to be $100\mu m \times 100\mu m$ with length being $L = 500\mu m$. The viscosity of the fluid is taken to be $\mu = 0.001 Pa \cdot s$. The inlet velocity is taken to be constant at $100\mu m/s$.

Table 1 presents the BEM results along with the analytical solution for the given benchmark problem. It can be seen that BEM gives sufficiently accurate results for this benchmark.

As a second benchmark, the motion of a spherical particle over a stationary particle in an infinite medium is analysed. The schematic drawing of this problem is given in Figure (1). This problem has an analytical solution for $\xi \gg 1$ (far-field solution) and for $\xi \ll 1$ (near-field solution) [Risbud and Drazer (2013)]. The comparison of the far-field is given in Figure (2). For this example, the flow speed, U_∞ is selected to be $100\mu m/s$ in the stated direction of the figure, the dimensions are, $a = b = 10\mu m$, $b_{\min} = 60\mu m$ and horizontal distance between the particles is $100\mu m$. In

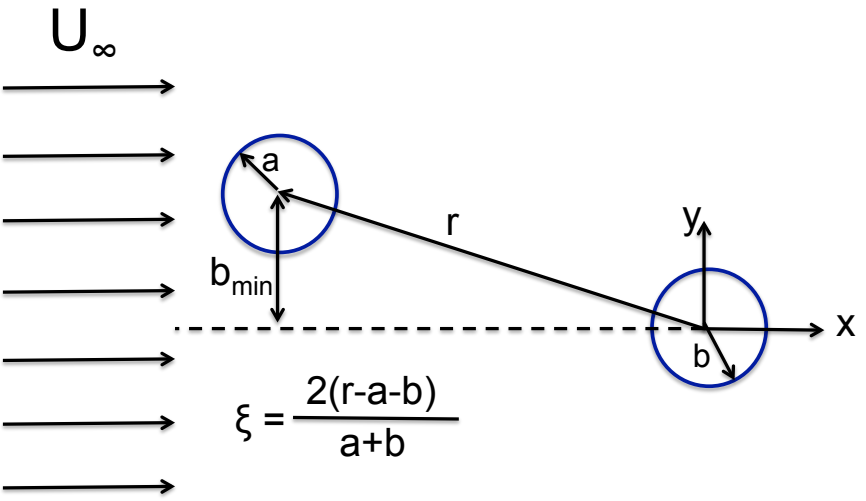

Figure 1: Schematic drawing of the second benchmark problem.

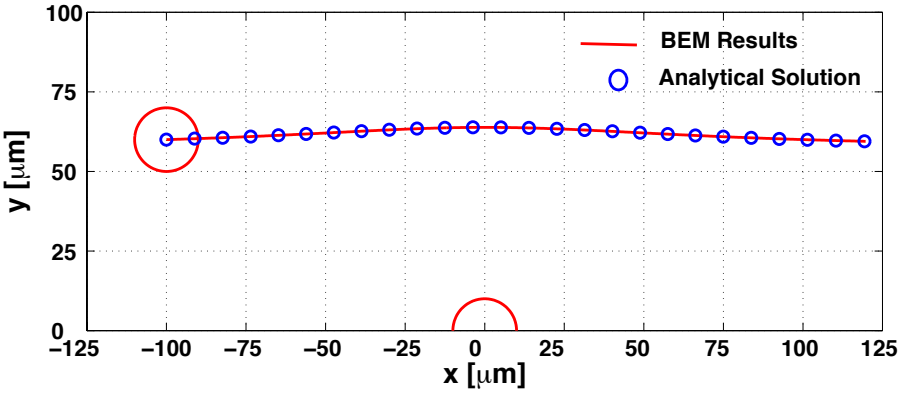

Figure 2: Comparison of BEM and analytical solution for far-field two sphere problem.

the simulations, 6144 elements per particle are used on the particles. As seen from the figure, BE formulation recovers the analytical solution pretty well.

As a third benchmark problem, a three sphere (with identical radius) problem is considered (schematic drawing of the problem can be seen in Figure 3). The fluid medium where the spheres are placed is steady. To the center of gravity of the middle sphere a predefined force with magnitude of $6\pi\mu a$ is exerted in horizontal direction (to the right), which moves this and the other two particles to the right.

In the analysis, selected parameters can be nondimensionalised with a (where in the numerical analysis, nondimensional value for a is taken to be 1) and μ (where in the numerical studies nondimensional value for μ is taken as 1). The nondimensional force, therefore, is obtained to be 6π in numerical value. Recently, a numerical technique has been proposed and compared against the Stokian Dynamics solution, and found that this technique is accurate up to particle separation of 0.1 radius [Wilson (2013)]. The result of the study is compared with the present BEM formulation and the results are tabulated in Table 2 (the results from [Wilson (2013)] was digitised for the comparison) . As seen from the results, a very good agreement has been achieved.

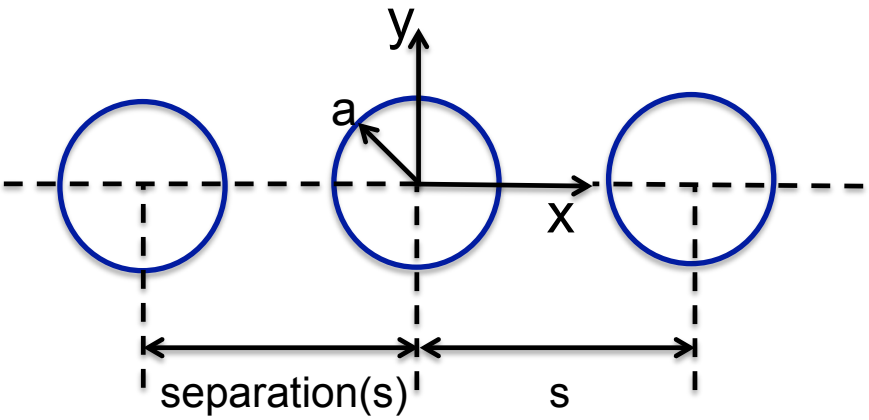

Figure 3: Schematic drawing of the benchmark problem.

Table 2: Comparison of the drag forces on the spheres for three sphere problem.

s/a	Middle Sphere			Side Sphere		
	BEM	Wilson (2013)	% Error	BEM	Wilson (2013)	% Error
2.01	0.666	0.659	1.08	0.644	0.645	0.21
2.05	0.703	0.694	1.28	0.629	0.630	0.10
2.10	0.733	0.727	0.89	0.615	0.614	0.10
2.20	0.780	0.775	0.69	0.589	0.589	0.01
2.40	0.847	0.838	1.02	0.548	0.547	0.13
2.60	0.886	0.878	0.89	0.513	0.512	0.25
2.80	0.913	0.906	0.80	0.483	0.483	0.06
3.00	0.933	0.925	0.90	0.457	0.456	0.15

4 Parallellization

The BE problem to be solved involves a large asymmetric dense and highly ill-conditioned coefficient matrix. Therefore, instead of an iterative solver, a direct solver more specifically LU-Factorisation is preferred [Engeln-Müllges and Uhlig (2014); Chan, van de Geijn, and Chapman (2010)]. To show the comparison of the conventional formulation and impedance formulation, both formulation is implemented using both sequential and parallel algorithms. The activity diagram of both formulations are given in Figures (4) and (5).

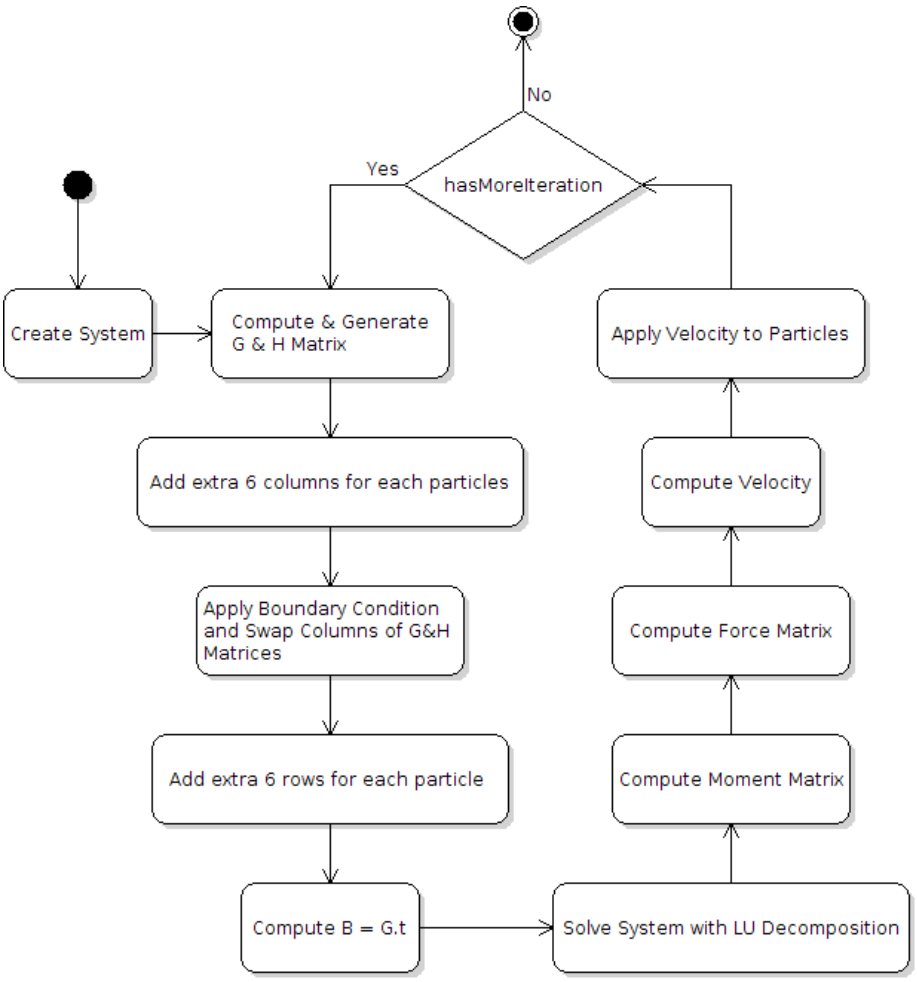

Figure 4: General activity diagram of conventional formulation.

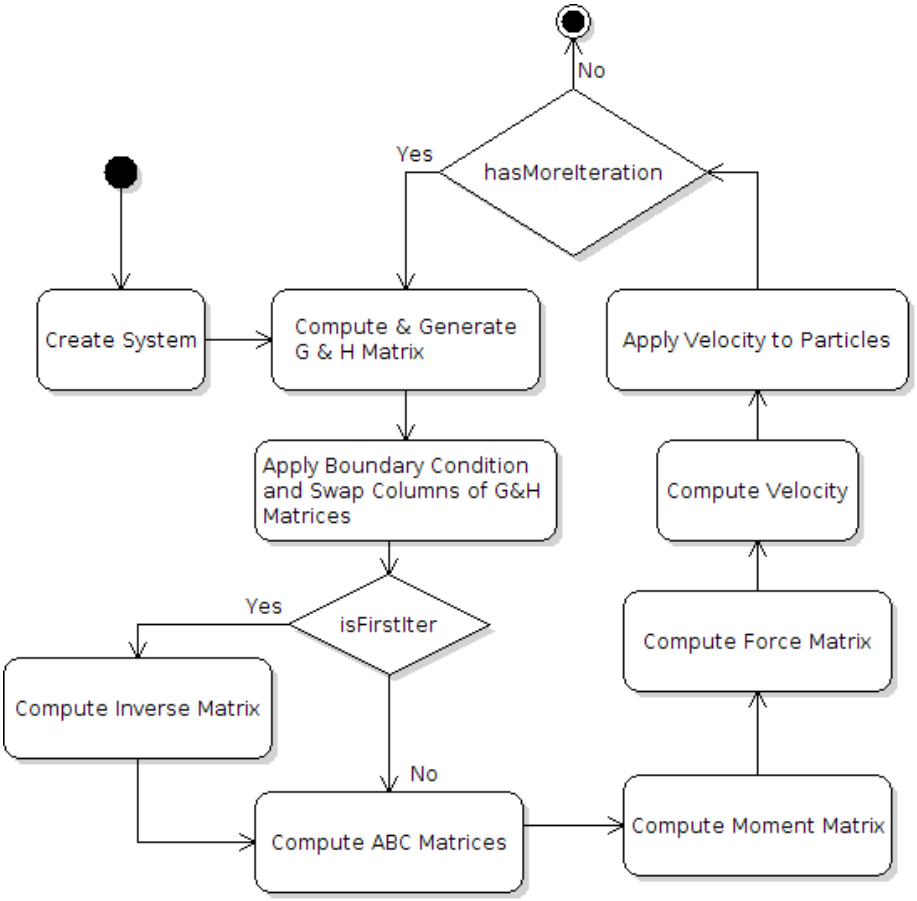

Figure 5: General activity diagram of impedance formulation.

Before introducing the parallelisation into the codes, some performance optimisation within the sequential and parallel solutions using reordering loop variables to make more cache hits [Kowarschik and Weiß (2003)] are applied. As another trivial optimisation, the repeated calculations of the unchanged part of matrices, such as the computation of $\mathbf{G}_{00}$ and $\mathbf{H}_{00}$ given in Equation (16) are avoided after the initial step. The activity diagram of creating **G** and **H** matrices are given in Figure (6).

For the high-performance basic linear algebra operations, many library frameworks such as BLAS (Basic Linear Algebra Subprograms)[Lawson, Hanson, Kincaid, and Krogh (1979); Dongarra, Croz, Hammarling, and Hanson (1998); Dongarra, Croz, Hammarling, and Duf (1990)], ATLAS [Whaley and Dongarra (1998)], Intel's MKL [Intel (2014)], AMD's ACML [AMD (2014)], and IBM's ESSL [IBM

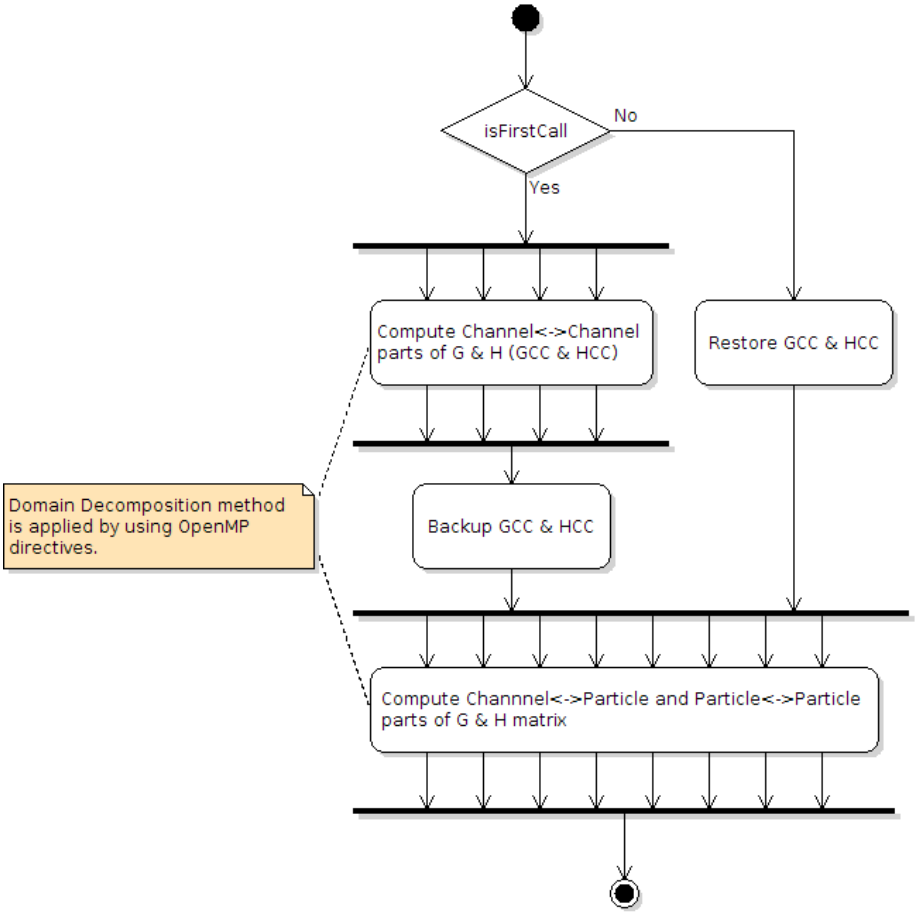

Figure 6: Computation of **G** and **H** matrices.

(2014)] are available in the literature. The Open Source implementation OpenBLAS [OpenBLAS (2014)] is an optimised version of the BLAS library. The libFLAME [libFLAME (2014)], High Performance Dense Linear Algebra Library underlined with OpenBLAS, is preferred in this study because of its easy notation and high performance implementation of Dense Linear Algebra operations on Shared Memory Architecture (SMA). At this point, it should be emphasised that the major objective of this study is not to implement the most efficient application, but to develop a parallel application which illustrates the efficiency of the proposed algorithm. SMP parallelisation techniques are used for making efficient use of computation power in solving the problem. In order to accomplish this, OpenMP preprocessor directives are being utilised in parallel code generation. OpenMP is

an API which supports shared memory multiprocessing programming. OpenMP's SMA (Shared Memory Architecture) is based on *threading*, and gives programmer an API to use threading easily and safely. In the present problem, there is a huge amount of data to be processed and all the data is used in the subsequent steps of the computation. An advantage of SMA is that, there is no data transfer between threads, since all threads are executed on the same computer, but on different CPU cores, sharing the same memory. When developing parallel version of program, concentration is given on the parts where sequential code consumes much of the computing time.

As can be seen from general activity diagram given in Figure (4), after the system is created, all other operations are repeated in every iteration. Generating the system and triangulation of the mesh is performed in sequential codes. Figure (6) shows how the generation of **G** and **H** matrices in sequential codes is optimised, and where domain decomposition method is applied and the parallel computation is performed. As can be seen in Equation (16) **G** and **H** matrices are decomposed into four sub-matrices. Since the channel does not change its shape, $\mathbf{G}_{00}$ and $\mathbf{H}_{00}$ are needed to be computed only once. In the first iteration, they are computed and stored once in the memory and retrieved many times in the subsequent iterations. Also the loop reorganisation is applied to use more cache hits in order to increase the performance. In the analysis, DELL R720 computer with 32 Core CPU, having NUMA Architecture with 384 GB of RAM is used to measure and compare the efficiency of both applications.

5 Case studies

As stated above, the verification of the code is made for several problems, comparing the results with the analytical formulations or results from the previous studies. In this section, the case studies measuring the performance of conventional method and presented algorithm are given. For this, a very simple microchannel flow problem is defined: a channel with dimensions 100x100x500 μm with one or more particles located near to the centroid of the channel. Each particle is assumed to be spheres of 10 μm radius. A representative sketch with one particle is given in Figure (7).

The boundary conditions can be stated as follows: at the inlet, constant velocity profile with 100 $\mu m/s$ along the longer channel axis and zero velocity along the other two directions; at the exit no viscous forces; and on the channel walls zero velocity in all directions. From the analysis the particle velocity for single particle is found to be approximately 250 $\mu m/s$, which means, with a time step size of $0.001s$, the particle will move 250 μm in 1000 time steps, for which the performance calculations are based on. Note that, for different problems which have different di-

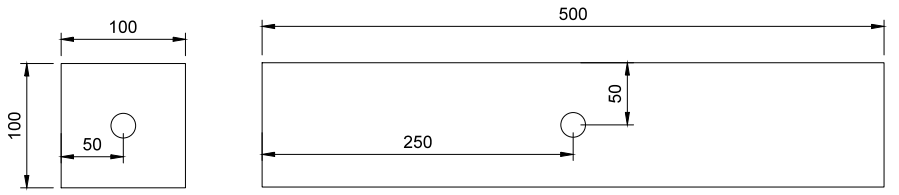

Figure 7: Example problem for performance analyses.

mensions and boundary conditions, the number of steps to consider would differ, which would affect the following discussions. It is clear that the performance of both method would depend on the problem size. The main difference between the conventional and impedance is that in the former method, the computation of LU decomposition is performed on **G** and **H** matrices as a whole, but in latter method, the inverse of $\mathbf{G}_{00}$ is computed once and used in the subsequent operations.

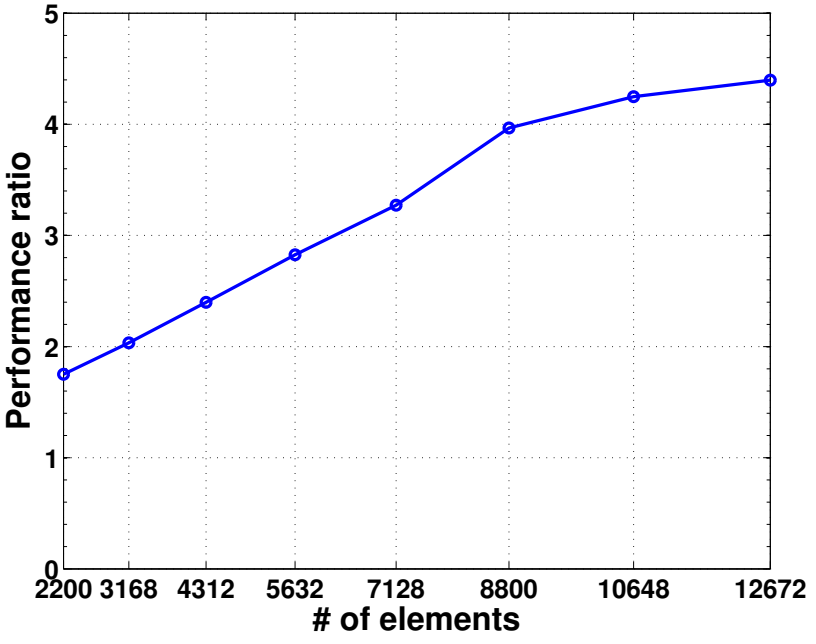

Figure 8: Performance ratio depending on the number of elements on channel.

We begin case studies with a fixed number of elements on the particle, which is selected to be 384, and investigate the effect of changing number of elements on the channel. It can be easily seen in Figure (8) that, for a fixed particle element count, the increasing number of elements on the channel increases the performance ratio of the impedance formulation. We use the following speed-up and performance ratio formulas:

$$
\begin{array}{rcl}
t_s & : & \text{total time spent in sequential algorithm} \\
t_p & : & \text{total time spent in parallel algorithm} \\
s & : & \text{speed-up} \\
ct & : & \text{total time spent in conventional method} \\
pt & : & \text{total time spent in proposed method} \\
pr & : & \text{performance ratio} \\
s & = & \frac{t_s}{t_p} \\
pr & = & \frac{ct}{pt}
\end{array}
$$

A second analysis is performed comparing the different numbers of particles in flow with the number of elements on the channel unchanged (Figure (9)). Each particle is of the same size and discretised with the same number of elements. Three different channel discretisations are performed: **(i)** 3168, **(ii)** 5652, and **(iii)** 8800 elements on the channel. Results are displayed in Figure(9) where it can be observed that, the performance of the impedance formulation depends on the ratio of number of elements on the channel to the total number of elements on the particles. Close inspection reveals that, when this ratio is increased, the performance of the presented method increases. At this point, a short note is in place: in most particle tracking problems in microfluidic applications, the total number of elements on the channel is comparably much more than the total number of elements on the particles - this introduces a very important advantage for the presented formulation.

A third analysis is on the parallelisation of the procedure. For a fixed number of elements on the channel and particle (5632 elements on channel and 768 elements on particle), the applications run on 1 to 32 threads (Figure (10)). It can be seen that the parallel speed-up of the impedance formulation is better than that of the conventional formulation.

To assess the proposed method further, a practical microfluidics application, namely the hydrodynamic separation process is considered. This process is widely used when particles of different sizes are needed to be separated. A major need for such a case would be the separation of the red blood cells from the bacteria in a blood sample, where the former is larger than the latter. In this example flow of two spherical particles are analysed. The radius of the particles are $10\mu m$ and $4\mu m$.

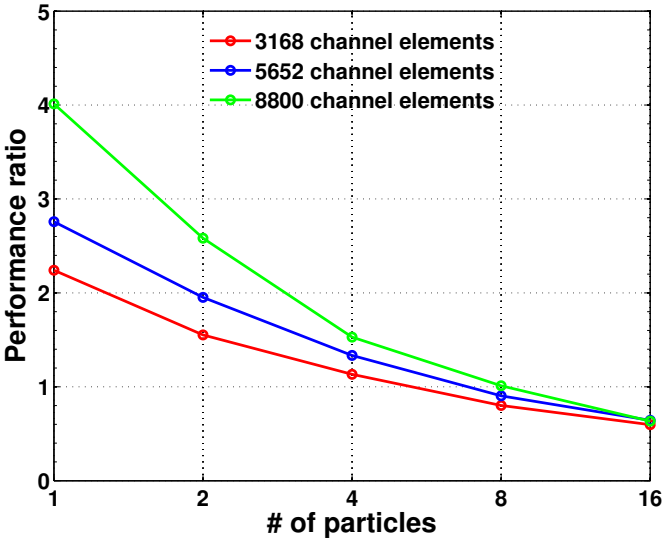

Figure 9: Performance ratio depending on the number of particle.

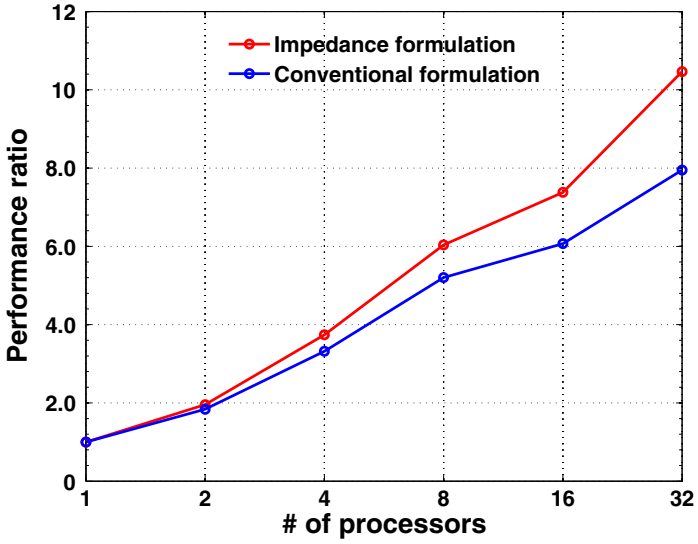

Figure 10: Speed-up depending on the number of processor.

The channel geometry is given in Figure (11). The boundary conditions are: constant velocity profile in the direction of the channel with magnitude of $100\mu m/s$, sticking condition at channel walls and free of tractions at the exit. Two particles are released at the same distance ($10\mu m$) from the channel side-walls.

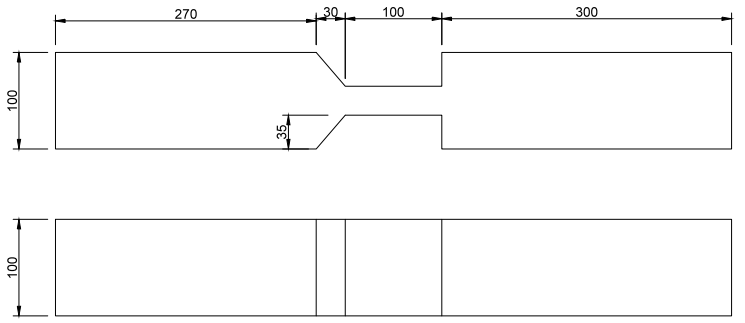

Figure 11: The channel geometry for separation problem.

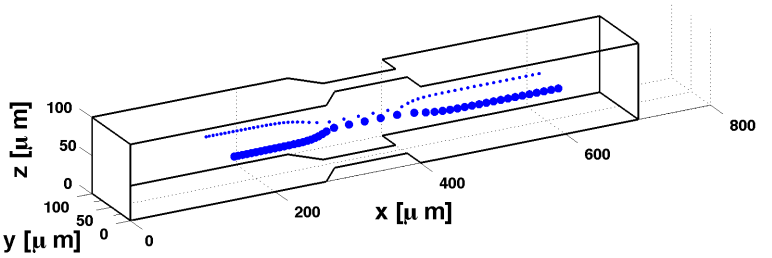

Figure 12: 3D visualisation of the particle trajectories.

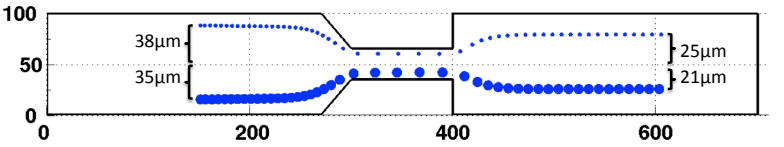

Figure 13: Top view of the channel presenting the trajectory of the particles.

The results of the analysis is presented, in 3D view, in Figure (12). The flow of the particles with the fluid can be visualised in this figure. Also, a top-view of the trajectories are presented in Figure (13). Here, the separation of the particles (distance from the center) can be seen.

6 Conclusions and Discussions

A new method for particle trajectory tracking is presented for Stokes' flow problems, especially applicable to the field of microfluidics. The method depends on matrix multiplications using the system matrices of the boundary element method, resulting into a system of equations relating the particle velocities (linear and angular) to the external forces and moments applied on the particle (in case of freely moving particle(s), these forces and moments are zero). The presented formulation would prove its use in multi-physics applications where the external force is due to another physical problem, such as electrical, magnetic and/or acoustic field etc. LU factorisation with partial pivoting, although being one of the most effective method for solving linear system of equations, is very difficult to parallelise, due to the pivoting step [Chan, van de Geijn, and Chapman (2010)]. Incremental pivoting is proposed [Chan, van de Geijn, and Chapman (2010)] for speeding up the LU factorisation. However, it was mentioned in the same article that, this approach may cause numerical instability.

Matrix multiplication operations, on the other hand, has been proven to have high parallel performance, especially on the new infrastructures (such as GPUs) are introduced. Also, with these new infrastructures, new algorithms that give higher speed-ups in parallel processing are being introduced [Alqadi, Aqel, and Emary (2008); Smith, Geijn, Smelyanskiy, Hammond, and Zee (2014)]. Accordingly, the formulation presented in this study is expected to be much more useful in the near future. Note that, the speed-up of the presented algorithm compared with the conventional solution is increasing when the number of CPU cores increase (recall Figure (10) for a demonstration up to 32 cores) - which is an indication of high-scalability of the proposed method.

In the presented algorithm, the matrices $\mathbf{G}_{00}$ and $\mathbf{H}_{00}$ are computed only once, and the inverse of the matrix $\mathbf{G}_{00}$ is obtained at the beginning of the process. A possible application to utilise this property of the present procedure would be as: computing the $\mathbf{G}_{00}$, $\mathbf{H}_{00}$ and $\mathbf{G}_{00}^{-1}$ in a powerful computer that has sufficient memory and storing it for use in a common computer where only the matrix multiplications are performed. Also, with the above matrices evaluated once, different problems (e.g. with different number of particles at different locations) can be solved simultaneously in different machines.

In the case study presented, it can be seen that, as the ratio of $\mathbf{G}_{00}$ over $\mathbf{G}_{PP}$ increases, the presented method gets better. When general applications of particle flow in microchannels is considered, this ratio is very high. This makes the presented algorithm a powerful tool for particle tracking problems in microfluidic applications.

Acknowledgement: The authors would like to thank to Prof. Dr. Yalçın Mengi for his support in the formulation.

References

Ai, Y.; Joo, S. W.; Jiang, Y.; Xuan, X.; Qian, S. (2009): Pressure-driven transport of particles through a converging-diverging microchannel. *Biomicrofluidics*, vol. 3, no. 022404, pp. 1–14.

Ai, Y.; Mauroy, B.; Sharma, A.; Qian, S. (2011): Electrokinetic motion of a deformable particle: Dielectrophoretic effect. *Electrophoresis*, vol. 32, pp. 2282–2291.

Alqadi, Z. A.; Aqel, M.; Emary, I. M. M. E. (2008): Performance analysis and evaluation of parallel matrix multiplication algorithms. *World Applied Sciences Journal*, vol. 5, no. 2, pp. 211–214.

AMD (2014): Amd core math library. http://developer.amd.com/tools-and-sdks/cpu-development/amd-core-math-library-acml/, 2014.

Argeso, H.; Mengi, Y. (2013): A frequency domain boundary element formulation for dynamic interaction problems in poroviscoelastic media. *Computational Mechanics*, vol. 1, pp. 1–3.

Baranoglu, B.; Cetin, B. (2014): A particle flow specific boundary element formulation for microfluidic applications. In Karayiannis, T.; Konig, C. S.; Balabani, S.(Eds): *4th Micro and Nano Flow Conference, 6–10 September 2014*, no. 206. Brunel University.

Cetin, B.; Li, D. (2011): Dielectrophoresis in microfluidics technology. *Electrophoresis, Special Issue on Dielectrophoresis*, vol. 32, pp. 2410–2427.

Cetin, B.; Ozer, M. B.; Solmaz, M. E. (2014): Microfluidic bio-particle manipulation for biotechnology. *Biochem. Eng. J.*, vol. 92, pp. 63–82.

Chan, E.; van de Geijn, R.; Chapman, A. (2010): Managing the complexity of lookahead for lu factorization with pivoting. In *Proceedings of the 22nd ACM symposium on Parallelism in algorithms and architectures*, pp. 200–208.

Dongarra, J. J.; Croz, J. D.; Hammarling, S.; Duf, I. (1990): A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Soft.*, vol. 16, no. 1, pp. 1–17.

Dongarra, J. J.; Croz, J. D.; Hammarling, S.; Hanson, R. J. (1998): An extended set of fortran basic linear algebra subprograms. *ACM Trans. Math. Soft.*, vol. 14, no. 1, pp. 1–17.

Engeln-Müllges, G.; Uhlig, F. (2014): *Numerical Algorithms with C*. Springer.

Fogelson, A. L.; Peskin, C. S. (1988): A fast numerical method for solving the three-dimensional stokes equations in the presence of suspended particle. *J. Comput. Phys.*, vol. 79, pp. 50–69.

Glowinski, R.; Pan, T. W.; Hesla, T. I.; Joseph, D. D.; Periaux, J. (2001): A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: Application to particulate flow. *J. Comput. Phys.*, vol. 169, pp. 363–426.

House, D. L.; Luo, H. (2010): Electrophoretic mobility of a colloidal cylinder between two parallel walls. *Engineering Analysis with Boundary Elements*, vol. 34, pp. 471–476.

House, D. L.; Luo, H. (2011): Effect of direct current dielectrophoresis on the trajectory of a non- conducting colloidal sphere in a bent pore. *Electrophoresis*, vol. 32, pp. 3277–3285.

IBM (2014): Engineering and scientific subroutine library. http://www.ibm.com/systems/power/software/essl/, 2014.

Intel (2014): Math kernel library. https://software.intel.com/en-us/intel-mkl, 2014.

Kowarschik, M.; Weiß, C. (2003): An overview of cache optimization techniques and cache-aware numerical algorithms. In Meyer, U.; Sanders, P.; Sibeyn, J.(Eds): *Algorithms for Memory Hierarchies*, volume 2625 of *Lecture Notes in Computer Science*, pp. 213–232. Springer Berlin Heidelberg.

Langthjem, M. A.; Nakano, M. (2013): Application of the Time-Domain Boundary Element Method to Analysis of Flow-Acoustic Interaction in a Hole-tone Feedback System with a Tailpipe. *CMES-COMPUTER MODELING IN ENGINEERING & SCIENCES*, vol. 96, no. 4, SI, pp. 227–241.

Lawson, C. L.; Hanson, R. J.; Kincaid, D. R.; Krogh, F. T. (1979): Basic linear algebra subprograms for fortran usage. *ACM Trans. Math. Soft.*, vol. 5, no. 3, pp. 308–323.

libFLAME (2014): High performance dense linaer algebra library. http://www.cs.utexas.edu/ flame/web/libFLAME.html, 2014.

Mengi, Y.; Argeso, H. (2006): A unified approach for the formulation of interaction problems by the boundary element method. *International Journal for Numerical Methods in Engineering*, vol. 66, pp. 816–842.

NUMA (2014): Non-uniform memory access. http://en.wikipedia.org/wiki/Non-uniform_memory_access, 2014.

OpenBLAS (2014): Optimized blas library based on gotoblas2. http://www.openblas.net/, 2014.

OpenMP (2014): The openmp api specification for parallel programming. http://openmp.org/wp/, 2014.

Pozrikidis, C. (1999): A spectral-element method for particulate stokes flow. *J. Comput. Phys.*, vol. 156, pp. 360–381.

Risbud, S. R.; Drazer, G. (2013): Trajectory and distribution of suspended non-brownian particles moving past a fixed spherical or cylindrical obstacle. *J. Fluid Mech.*, vol. 714, pp. 213–237.

Sellier, A. (2012): Stokes Flow about a Slip Arbitrary-Shaped Particle. *CMES-COMPUTER MODELING IN ENGINEERING & SCIENCES*, vol. 87, no. 2, pp. 157–176.

Sellier, A. (2013): Arbitrary Stokes Flow About A Fixed or Freely-Suspended Slip Particle. *CMES-COMPUTER MODELING IN ENGINEERING & SCIENCES*, vol. 96, no. 3, SI, pp. 159–176.

Smith, T. M.; Geijn, R. v. d.; Smelyanskiy, M.; Hammond, J. R.; Zee, F. G. V. (2014): Anatomy of high-performance many-threaded matrix multiplication. In *Proceedings of the 2014 IEEE 28th International Parallel and Distributed Processing Symposium*, IPDPS '14, pp. 1049–1059, Washington, DC, USA. IEEE Computer Society.

Whaley, R. C.; Dongarra, J. J. (1998): Automatically tuned linear algebra software. In *Proceedings of SC 98*.

Wilkinson, B.; Allen, M. (2005): *Parallel Programming : Techniques and Applications Using Networked Workstations and Parallel Computers*. Pearson Education, 2nd edition.

Wilson, H. J. (2013): Stokes flow past three sphere. *J. Comp. Phys.*, vol. 245, pp. 302–316.

Wrobel, L. C. (2002): *Boundary Element Method Volume 1: Applications in Thermo-Fluids and Acoustics*. Wiley, West Sussex, 2002.

Yalcin, O. F.; Mengi, Y. (2013): A new boundary element formulation for wave load analysis. *Computational Mechanics*, vol. 52, pp. 815–826.