MULTICORE PARALLELIZATION OF MESHLESS PDE SOLUTION BY OPENMP

Gregor Kosec, Roman Trobec, Matjaž Depolli, Aleksandra Rashkovska,
Department of Communication Systems, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

Abstract

The application of the local meshless numerical method (LRBFCM) in solving partial differential equations (PDE) is explored with the respect to a parallel implementation on multicore computers. The numerical approach is tested on the natural convection problem governed by the fluid flow and energy transport. To solve such kind of problems a pressure-velocity coupling is required, which in general pose a global problem and therefore introduces global communication and consequently smaller efficiency of the parallel implementation. The coupling is performed iteratively, with a local pressure correction, predicted from the violation of the mass continuity. The analysis of OpenMP based parallelization of the proposed numerical approach is presented. The meshless local approach is evaluated through various tests, which are gradually complicated from the simplest independent calculations to the solution of a coupled system of PDEs describing the unsteady natural convection problem. The results show superlinear speedup for larger problem domains and smaller number of cores with gradually decreasing efficiency for larger number of cores.

1 Introduction

Meshless methods, sometimes named also meshfree or mesh reduction methods, represent a class of numerical methods with spatial discretization based on an arbitrarily distributed set of nodes without any topological relations between them. Several meshless methods have been proposed such as the Element Free Galerkin method (EFG) [1], the Meshless Petrov-Galerkin method (MPG) [2], the Point Interpolation Method (PIM) [3], the Smoothed Particle Hydrodynamics method (SPH) [4], the Reproducing Kernel Particle Method (RKPM) [5], the Kansa Method (KM) [6], etc. However, this work is focused on one of the simplest class of meshless methods in development today, the point interpolation Radial Basis Function Collocation Method (RBFCM) [3].

The main drawback of the global approaches is in the necessity of solving a final linear system represented by dense matrices. The conditioning of such systems is generally sensitive to the distribution of the computation nodes as well as to the parameters of the numerical methods itself. The problem becomes important even with a relatively small number of nodes, e.g., 1000. From the computational point of view, the global approach is unwanted as it introduces the global communication between computing nodes and therefore complicates the parallelization. The mitigation of the related problem has been attempted by domain decomposition, multi-grid approach, and compactly supported basis functions, which all represent a substantial complication of the original simple method.

It was demonstrated [7] that the local formulation does not substantially degrade the accuracy with respect to the global one. The local formulation is much less sensitive and

effective regarding the computational time. The local variant of RBFCM (LRBFCM) has been used for the first time in a convective-diffusive problems [8] and since then successfully applied in several other problems. In this paper the simplest local meshless (LRBFCM) technique is combined with the simplest parallelization standard (OpenMP) in order to solve, in parallel, a system of partial differential equations (PDE) for the thermo-fluid problem of natural convection.

Computational time is an important factor in numerical methods and it is often not addressed adequately. Parallel computers can compensate for the lack of single computer performance, but only in cases if an efficient parallelization method is known. Many such algorithms are known for solving PDEs and have been implemented on parallel computers for different applications [9, 10, 11, 12].

The parallelization of a direct solver within the EFG method has been published in [13] with measured speedup and efficiency, which were 7.1 and 0.89, respectively, for four dual processors and systems with up to 2000 equations. Large sparse linear systems obtained by Free Mesh Method, which is a virtually meshless method based on FEM [14], have been parallelized on up to 64 processors [15]. Construction of the linear system was ideally parallelizable, while the parallel efficiency for solving the system was 0.6 on all available processors. Parallel RKPM in 3-D was implemented with efficiency of 0.6 on 64 processors [16]. The system of equations arising from 3-D meshless methods for crack propagations have been solved in parallel on 16 processors [17, 18]. The listed examples indicate that several demanding numerical applications need parallel implementations of meshless methods.

The parallel efficiency of the global methods degrade significantly if the problem is to be solved on a greater number of processors, because a dense global linear system must be solved. If the system was previously factorized, the solution implies the computational complexity of order $\mathcal{O}(N_D^2)$, N_D being the number of spatial discretization nodes. The global approach requires complex parallelization code and a significant amount of interprocessor communication. We have shown by theoretical and experimental results that the local MPG has a potential for parallelization similar to those of FDM and FEM [19].

The rest of the paper is organized as follows. In the next section a short background is given on the LRBFCM solution method. In Section 3, the OpenMP is described in short followed by the description of the multicore test computer architecture. The classical De Vahl Davis benchmark problem [20] with the proposed local solution procedure is described in Section 4. Section 5 is devoted to the presentation of the obtained results. The paper concludes with the summary and directions for future work.

2 Local Radial Basis Function Collocation Method (LRBFCM)

The general idea behind the local meshless numerical approach – in the present case the LRBFCM – is the use of a local influence domain for the approximation of a discretized field u, to evaluate the differential operators ($\mathcal{L}$) needed to solve a PDE. In this paper, five node influence domains are used, i.e., the affected node where the current computations take place and its four closest neighboring nodes. Each node uses its own influence domain for the evaluation of the spatial differential operations, consequently the domain is discretized with

overlapping influence domains. The described principle is depicted in Figure 1.

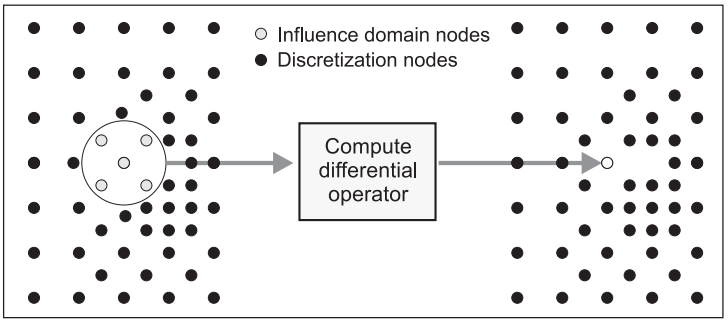

Figure 1: Local meshless principle.

The approximation function is introduced as

$$\hat{u}(\mathbf{p}) = \sum_{n=1}^{N} \alpha_n \Psi_n(\mathbf{p}), \tag{1}$$

where $\hat{u}$, N, α_n, $\mathbf{p}(p_x, p_y)$ and Ψ_n stand for the approximation function, the number of basis functions, the approximation coefficients, position vector and the basis functions, respectively. The basis could be selected arbitrarily, however in this paper only Hardy's Multiquadrics (MQs) $\Psi_n(\mathbf{p}) = \sqrt{(\mathbf{p}-\mathbf{p}^n)\cdot(\mathbf{p}-\mathbf{p}^n)/\sigma_C^2 + 1}$ are used, where σ_C stands for the free shape parameter and $\mathbf{p}^n$ denotes n-th influence domain node. Taking into account all influence domain nodes and Equation (1) the approximation system is constructed. The number of basis functions is taken the same as the number of influence domain nodes $N = 5$. The local approximation simplifies to collocation, which results in a linear systems of N equations in each computational node. The matrix formulation of the collocation is thus

$$\boldsymbol{\Psi}\boldsymbol{\alpha} = \boldsymbol{u} \tag{2}$$

The coefficients $\boldsymbol{\alpha}$ are obtained by solving Equation (2). The system (2) has to be solved only when the topology of influence domain changes; accordingly, the computation can be optimized by computing $\boldsymbol{\Psi}^{-1}$ in a pre-process phase. With the constructed collocation function an $\mathcal{L}$ can be applied on (1)

$$\mathcal{L}\hat{u}(\mathbf{p}) = \sum_{n=1}^{N} \alpha_n \mathcal{L}\Psi_n(\mathbf{p}), \tag{3}$$

which provides a numerical evaluation of $\mathcal{L}u(\mathbf{p})$.

3 Multicore parallelization

3.1 OpenMp

The OpenMP (Open Multi-Processing) is an Application Programming Interface (API) that supports multi-platform shared-memory multiprocessing programming in C, C^{++}, and For-

tran, on most processor architectures and operating systems, including Unix, Mac OS, Microsoft Windows and others. The OpenMP consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior.

The OpenMP uses a portable, scalable model and a simple and flexible interface for the development of parallel applications on different platforms ranging from desktop computers to supercomputers. A combination of the shared-memory and message passing interprocess communication is also possible either by using both the OpenMP and the Message Passing Interface (MPI), or through the use of an OpenMP extension for cluster systems.

The OpenMP implements multithreading where the master thread forks off a specified number of slave threads and divides the task among them. The threads run then concurrently, also on the shared data, with the runtime environment allocating threads to different processors. Each section of the code that must run in parallel is marked with a preprocessor directive. After the execution of the parallelized code, the slave threads join back into the master thread, which continues toward the end of the program.

The OpenMP can implement the task parallelism and the data parallelism at the same time. It runs particularly effective on multicore computer architectures based on the multilevel memory hierarchy and fast shared-memory caches. For the purpose of the present analysis we use `#pragma omp parallel for` directive with static scheduling.

3.2 Architecture of the test computer

All tests have been performed on a computer system with four Intel Xeon E7450 processors, each with six cores, system clock of 2.40 GHz, 1066 MHz front side bus (FSB), 64 GB of shared main memory and without hyperthreading technology. The system has three levels of cache hierarchy:

- Each core has L1 execution cache (32 KB) to store micro-operations (shortest decode time on cache hits) and data cache (32 KB) to improve data tracing. Typical L1 latency is 2 clock cycles.
- Each pair of cores shares 3 MB of L2 cache, for a total of 9 MB of shared L2 cache per processor Typical L2 latency in the case of L2 cache hit is 6 clock cycles.
- Each processors has 12 MB of shared L3 cache. Typical L3 latency in the case of L3 cache hit is 60 cycles or 120 clock cycles in the case of L3 cache miss and main memory hit. The 6-core modules communicate through PCIe bus, which implements 64 GB of shared main memory. The architecture of the whole system is shown in Figure 2.

The computational performance is analyzed through the speedup (S) defined as:

$$S = \frac{t_1}{t_{N_C}}, \tag{4}$$

where t_1 and t_{N_C} stand for computation time on a fastest single core system and on an N_C-core system, respectively. Because of the hierarchical shared cache/memory architecture we can expect a significant impact of the problem size (N_D) and the number of cores (N_C) on the speedup (S) of parallel applications.

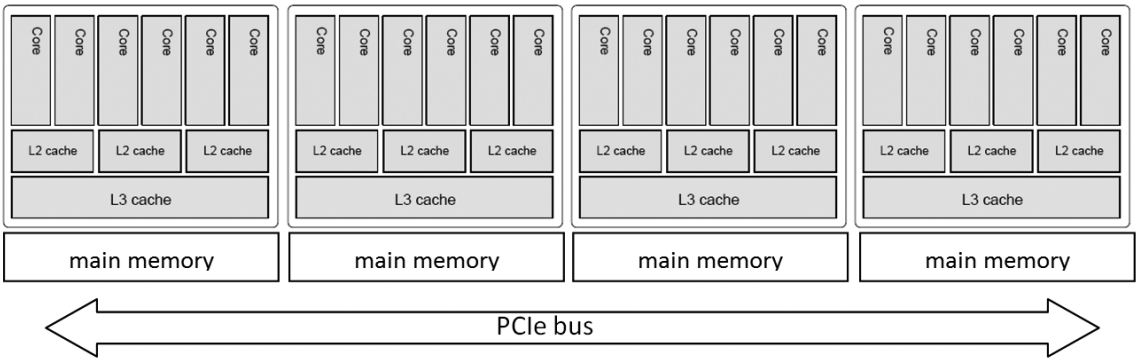

Figure 2: Block diagram of the test computer based on four Intel Xeon E7450 processors connected through PCIe bus.

.

4 Test problems

4.1 Synthetic test case

For the better insight into the performance of the OpenMP parallelization a synthetic test is designed with a single spatial loop in a temporal loop. To control the amount of calculation complexity N_F floating point operations (FP) can be requested in each spatial iteration. Additionally, a single float variable is rewritten in the system memory N_F times in each spatial iteration. This simple test case is referred to in the following as SL.

4.2 Fluid flow test case

The classical De Vahl Davis [20] 2-D natural convection problem (NC) is considered for benchmarking purposes. The problem domain is a closed air-filled square-shaped cavity with differentially heated vertical walls with temperature difference ΔT and insulated horizontal walls. There are several numerical solutions published in the literature [21, 22, 23] that make the test convenient for benchmarking purposes. The NC problem is described by three coupled PDE equations: mass (5), momentum (6) and energy conservation (7) equations, where all material properties are considered to be constant. The Boussinesq approximation defined in Equation (8) is used for the treatment of body force in the momentum equation. The natural convection is thus described by the following system of equations

$$\nabla \cdot \mathbf{v} = 0, \tag{5}$$

$$\rho \frac{\partial \mathbf{v}}{\partial t} + \rho \nabla \cdot (\mathbf{v}\mathbf{v}) = -\nabla P + \nabla \cdot (\mu \nabla \mathbf{v}) + \mathbf{b}, \tag{6}$$

$$\rho \frac{\partial \left(c_p T\right)}{\partial t} + \rho \nabla \cdot \left(c_p T \mathbf{v}\right) = \nabla \cdot \left(\lambda \nabla T\right), \tag{7}$$

$$\mathbf{b} = \rho\left[1 - \beta_T(T - T_{\text{ref}})\right]\mathbf{g}, \tag{8}$$

where λ, $\mathbf{v}(u,v)$, t, c_p, ρ, P, μ, $\mathbf{b}$, T, β_T, T_{ref} and $\mathbf{g}$ stand for thermal conductivity, velocity, time, specific heat, density, pressure, viscosity, body force, temperature, thermal expansion coefficient, reference temperature and gravitational acceleration, respectively. Problem is fully characterized by two dimensionless numbers, the Prandtl number ($\text{Pr} = \mu c_p/\lambda$) and the Rayleigh number ($\text{Ra}= |\mathbf{g}|\,\beta_T\,(\Delta T)\,\Omega^3\rho^2 c_p/\lambda\mu$). The present test is performed for $\text{Pr} = 0.71$ and $\text{Ra} = 10^8$. More details on the subject can be found in [24].

4.3 Solution procedure

The problem is solved numerically with the LRBFCM for spatial discretization on N_D regularly distributed nodes and explicit time stepping employed for temporal discretization. An important part of the solution procedure is the treatment of the mass and momentum equations, which are not explicitly coupled. The most widely known remedy for this problem is Semi-Implicit Method for Pressure Linked Equations (SIMPLE) [25] where the problem is translated to the pressure correction Poisson equation. Such an approach poses a global problem.

To maintain the locality of computations, an alternative approach is used [24], similar to the artificial compressibility method [26]. Equations (5) and (6) are solved iteratively through the pressure correction based on the local mass continuity (5) violation. In each time step an additional internal iteration takes place, where the pressure is corrected with the divergence of the intermediate velocity.

The basic elements of the complete solution procedure in each time step are as follows:

- In the first step the energy transport (7) is considered. The new temperature is computed as

$$T_1 = T_0 + \frac{\Delta t}{\rho c_p}\left[\nabla\cdot(\lambda\nabla T_0) - \nabla\cdot(\rho c_p T_0 \mathbf{v}_0)\right], \tag{9}$$

  where index 0 denotes values at current time t and index 1 denotes values at time $t+\Delta t$. The Δt stands for the time-step length.

- In the second step, the new velocity is estimated from the discretized transient form of Equation (6)

$$\hat{\mathbf{v}}_1 = \mathbf{v}_0 + \frac{\Delta t}{\rho}\left[-\nabla P_0 + \nabla\cdot(\mu\nabla\mathbf{v}_0) + \mathbf{f}_0 - \nabla\cdot(\rho\mathbf{v}_0\mathbf{v}_0)\right]. \tag{10}$$

- In the third step the pressure-velocity coupling is performed. The calculated velocity $\hat{\mathbf{v}}_1$ does not satisfy the mass continuity Equation (5) in general. The internal iteration algorithm is used to couple Equations (5) and (6). In the first internal iteration, the pressure and the velocity are set to the values from the previous time step. In the next internal iteration, P^{m+1} and $\mathbf{v}_1^{m+1}$ (m stands for internal iteration index) are estimated from the mass continuity violation as follows

$$\bar{P} = {\Omega_{ref}}^2\frac{\rho}{\Delta t}\nabla\cdot\mathbf{v}^m, \tag{11}$$

$$P^{m+1} = P^m + \gamma \bar{P}, \tag{12}$$

$$\mathbf{v}_1^{m+1} = \mathbf{v}_1^m - \gamma \frac{\Delta t}{\rho} \nabla \bar{P}. \tag{13}$$

γ stands for the relaxation parameter and Ω_{ref} for the characteristic length. The internal iteration stops when the convergence criteria ($\nabla \cdot \mathbf{v}^{m+1} < \epsilon$) is met in all computational nodes.

- With known new temperature, pressure and velocity the simulation proceed to the next time step.

The flowchart of the above solution procedure is shown in Figure 3.

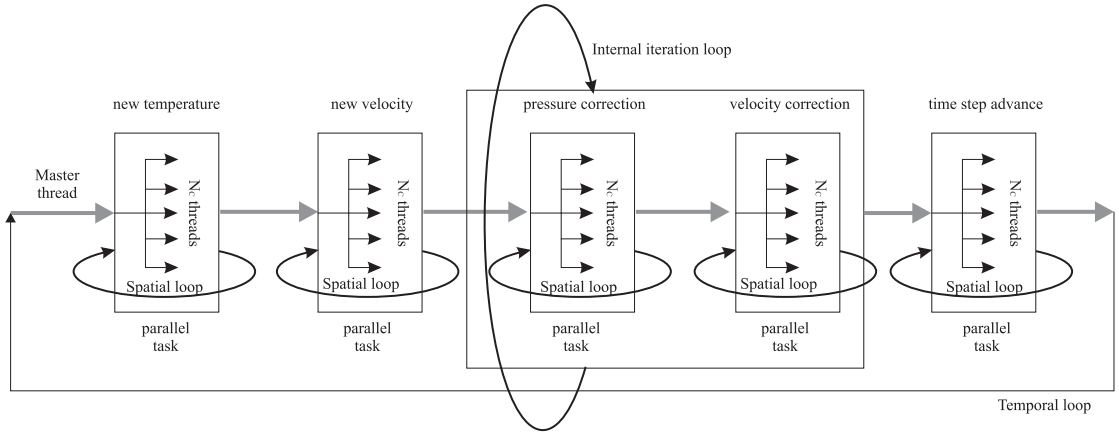

Figure 3: Scheme of the fluid flow test case solution procedure.

5 Results

The results of the NC solution are presented in terms of cavity stream function and temperature contours in Figure 4 (left), and temporal development of Nusselt number ($\mathrm{Nu} = -\partial T/\partial \mathbf{p}_x + Tv$) in Figure 4 (right). The detailed comparison of results computed with the proposed solution procedure against the data from previous publications can be found in [24].

The speedup analysis for the SL test case with $N_F = 1$ is shown in Figure 5 (left). It demonstrates a super linear behavior on larger systems and limited speedups on smaller systems. The super linear speedup originates from L1 cache exploitation. For the smaller systems all data can be stored in a single L1 cache. Memory demands become higher as the system size N_D increases and hence only a single L1 cache is not enough to store all the application data. The application has to access data from higher level of memory hierarchy. Increasing N_C provides more L1 caches, which results in super linear speedups for greater systems ($N_D > 17424$).

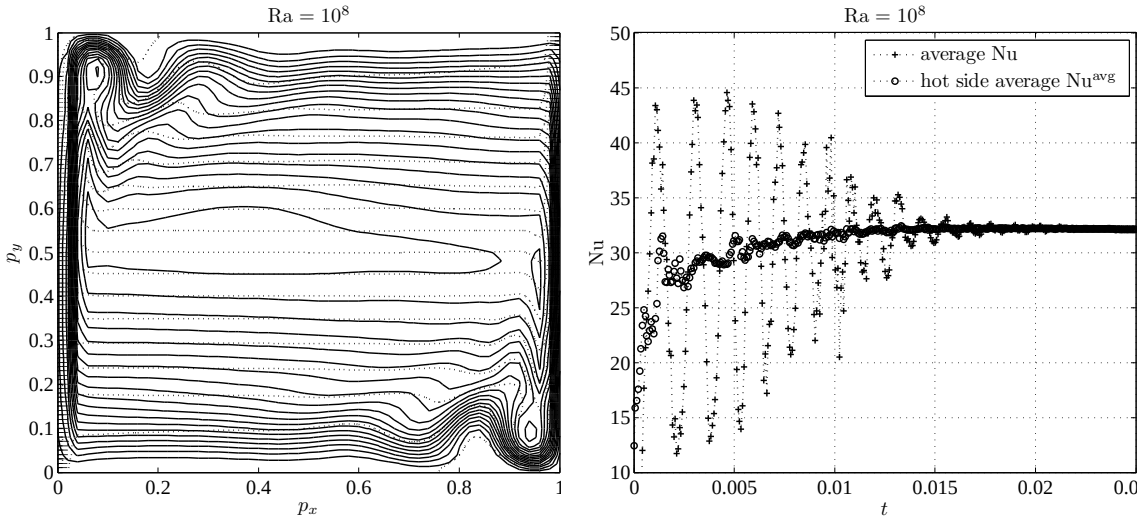

Figure 4: Cavity streamlines (solid line) and temperature contour plot (dotted line) (left) and Nusselt temporal development (right).

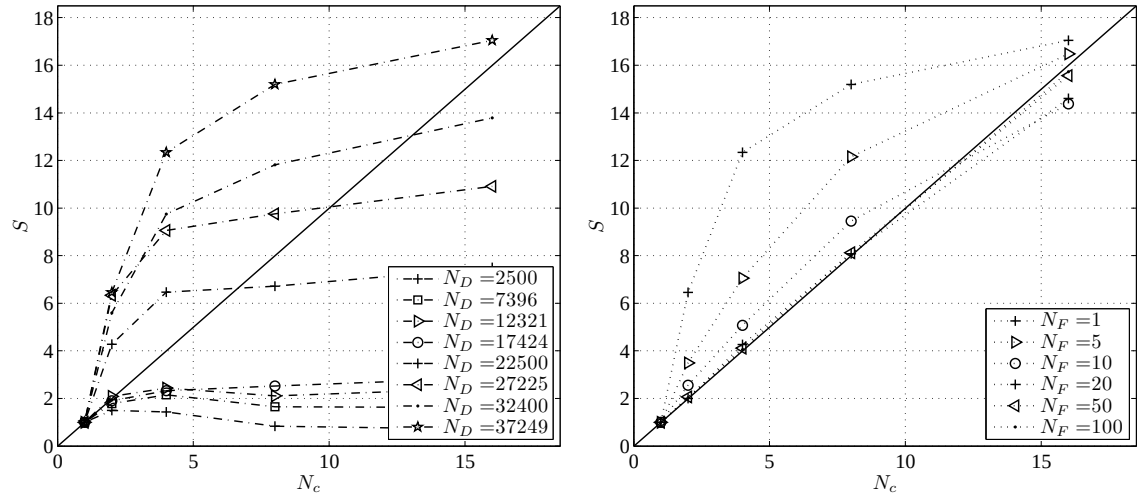

Figure 5: Speedup of the SL case for $N_F = 1$ as a function of number of cores (N_C) for different problem sizes (N_D) (left), and speedup for the SL case for different computational complexities (N_F) with $N_D = 37249$ (right).

In Figure 5 (right) the speedup on the largest system ($N_D = 37249$) is tested for different N_F. For the lowest computational cost ($N_F = 1$) the speedup is strongly super linear as the data manipulation presents the bulk proportion of the execution time. With the increasing N_F the super linearity of speedup diminishes, because the floating point calculation prevails. On the other hand, the cases with higher N_F can be solved faster only with higher N_C.

The computation time of the SL for N_F as a function of the domain size is shown in Figure 6 (left). If the computation is carried out on a single core, the computation time t_C increases

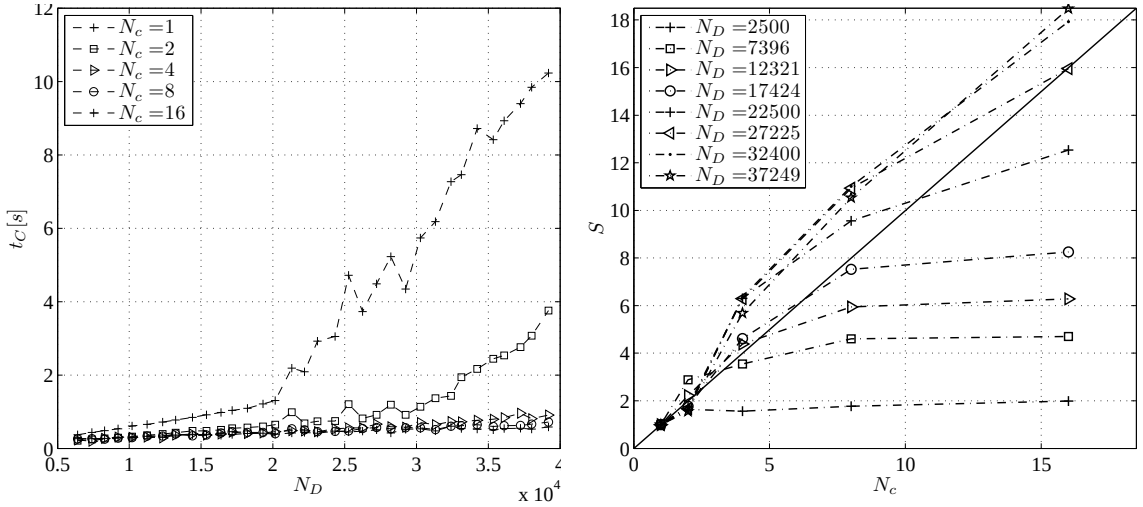

Figure 6: Computation time for the SL case as a function of problem size (N_D) for different number of cores (N_C) (left), and speedup for the NC case as a function of N_C for different N_D (right).

proportional to N_D up to roughly $N_D = 2 \cdot 10^4$. From that point on the slope of t_C increases and destabilizes since higher level memories have to be accessed. The behavior for $N_C = 2$ is similar, however the slope destabilizes at the greater system ($N_D = 3.5 \cdot 10^4$), because two L1 caches are available. For higher N_C the accumulated caches provide enough memory for all tested cases.

Finally, the speedup of a real application (NC) is shown in Figure 6 (right). The speedup is still super linear for larger systems but it converges to linearity because the NC computational complexity is relatively large regarding the time needed for the problem data access.

6 Conclusions

We demonstrate the application of the local meshless numerical technique in solving the natural convection problem. To treat a fluid flow problem one needs to solve pressure-correction, which is a global problem as it is governed by an elliptic equation. Instead of solving the global problem we approach the subject with a simple iterative correction algorithm, similar to the artificial compressibility method. Such a localized approach makes the parallelization of the code straightforward and appropriate for the OpenMP parallelization.

Two different test cases are analysed. First, the pure synthetic case, where the number of FPs can be controlled and second, the natural convection case which requires a PDE system solution. The results are presented in terms of speedups and computational times for different scenarios. We show that as a result of accumulating L1 cache, multiple cores might produce super linear speedup, which is reduced as the complexity of the spatial iteration is increased.

It is demonstrated that the system of parabolic and elliptic PDEs can be effectively solved with a local solution procedure. The local behavior of the presented solution procedure shows convenient advantages like an ease of implementation, straightforward parallelization, simple consideration of complex physical models and processors' effectiveness. Nevertheless, the proposed methodology can be directly applied in solving more complex problems like an unsteady fluid flow dynamics [27] or even more complex macrosegregation problem [28].

In future work the presented analysis will be implemented for computing clusters with much more interconnected multicore processors. The current implementation will be upgraded by MPI communication.

References

[1] X. H. Zhang, J. Ouyang, L. Zhang, Element-free characteristic galerkin method for burgers' equation, Engineering Analysis with Boundary Elements 33 (3) (2009) 356–362.

[2] S. N. Atluri, The Meshless Method (MLPG) for Domain and BIE Discretization, Tech Science Press, Forsyth, 2004.

[3] J. G. Wang, G. R. Liu, A point interpolation meshless method based on radial basis functions, International Journal for Numerical Methods in Engineering 54 (11) (2002) 1623–1648.

[4] J. J. Monaghan, An introduction to sph, Computer Physics Communications 48 (1) (1988) 89–96.

[5] C. Lee, C. W. Im, H. K. Jung, H. K. Kim, D. W. Kim, A posteriori error estimation and adaptive node refinement for fast moving least square reproducing kernel (fmlsrk) method, CMES: Computer Modeling in Engineering and Sciences 20 (2007) 35–41.

[6] E. J. Kansa, Multiquadrics - a scattered data approximation scheme with application to computational fluid dynamics, part i, Computers and Mathematics with Applications 19 (1990) 127–145.

[7] C. K. Lee, X. Liu, S. Fan, Local muliquadric approximation for solving boundary value problems, Computational Mechanics 30 (2003) 395–409.

[8] B. Šarler, From global to local radial basis function collocation method for transport phenomena, Springer, Berlin, 2007, pp. 257–282.

[9] G. Golub, J. M. Ortega, Scientific Computing - An Introduction with Parallel Computing, Academic Press Inc., Boston, 1993.

[10] I. Foster, Designing and building parallel programs, Addison-Wesley, New York, 1996.

[11] B. Urban, D. Janezic, Symplectic molecular dynamics simulations on specially designed parallel computers, Journal of Chemical Information and Modelling 45 (2005) 1600–1604.

[12] S. H. Paik, J. J. Moon, S. J. Kim, , M. Lee, Parallel performance of large scale impact simulations on linux cluster super computer, Computers and Structures 84 (2006) 732–741.

[13] V. Singh, Parallel implementation of the EFG method for heat transfer and fluid flow problems, Computational Mechanics 34 (2004) 453–463.

[14] G. Yagawa, T. Yamada, Free mesh method: A new meshless method, Computational Mechanics 18 (1996) 383–386.

[15] M. Shirazaki, G. Yagawa, Large-scale parallel flow analysis based on free mesh method: a virtually meshless method, Computer Methods in Applied Mechanics and Engineering 174 (1999) 419–431.

[16] L. Zhang, G. J. Wagner, W. K. Liu, A parallelized meshfree method with boundary enrichment for large-scale CFD, Journal of Computational Physics 176 (2) (2002) 483–506.

[17] T. Rabczuk, S. Bordas, G. Zi, A three-dimensional meshfree method for continuous crack initiation, nucleation and propagation in statics and dynamics, Computational Mechanics 40 (3) (2007) 473–495.

[18] S. Bordas, T. Rabczuk, G. Zi, Three-dimensional crack initiation, propagation, branching and junction in non-linear materials by an extended meshfree method without asymptotic enrichment, Engineering Fracture Mechanics 75 (5) (2007) 943–960.

[19] R. Trobec, M. Šterk, B. Robič, Computational complexity and parallelization of the meshless local Petrov-Galerkin method, Computers and Structures 87 (1-2) (2009) 81–90.

[20] G. de Vahl Davis, Natural convection of air in a square cavity: a bench mark numerical solution, International Journal of Numerical Methods in Fluids 3 (1983) 249–264.

[21] M. Hortmann, M. Perić, G. Scheuerer, Finite volume multigrid prediction of laminar natural convection - bench-mark solutions, International Journal for Numerical Methods in Fluids 11 (1990) 189–207.

[22] C. Prax, H. Sadat, P. Salagnac, Diffuse approximation method for solving natural convection in porous media, Transport in Porous Media 22 (1996) 215–223.

[23] H. Sadat, S. Couturier, Performance and accuracy of a meshless method for laminar natural convection, Numerical Heat Transfer B37 (2000) 455–467.

[24] G. Kosec, B. Šarler, Solution of thermo-fluid problems by collocation with local pressure correction, International Journal of Numerical Methods for Heat and Fluid Flow 18 (2008) 868–882.

[25] J. H. Ferziger, M. Perić, Computational methods for fluid dynamics.

[26] A. G. Malan, R. W. Lewis, An artificial compressibility cbs method for modelling heat transfer and fluid flow in heterogeneous porous materials (2011).

[27] G. Kosec, B. Šarler, Solution of phase change problems by collocation with local pressure correction, CMES: Computer Modeling in Engineering and Sciences 47 (2009) 191–216.

[28] G. Kosec, M. Zaloznik, B. Šarler, H. Combeau, A meshless approach towards solution of macrosegregation phenomena, Computers, materials and continua 580 (2011) 1–27.