World Applied Sciences Journal 27 (9): 1119-1125, 2013
ISSN 1818-4952

DOI: 10.5829/idosi.wasj.2013.27.09.15177

A Parallel 2D Stabilized Finite Element Method for Darcy flow on Distributed Systems

[1]Masroor Hussain, [2]Muhammad Abid, [1]Mushtaq Ahmad and [1]Fawad Hussain

[1]Faculty of Computer Science and Engineering,
GIK Institute of Engineering Sciences and Technology, Topi, KPK, Pakistan
[2]Faculty of Mechanical Engineering,
GIK Institute of Engineering Sciences and Technology, Topi, KPK, Pakistan

Abstract: This paper discusses the high performance implementation of a 2D stabilized Mixed Finite Element Method (MFEM) for Darcy flow in a distributed memory system, using Message Passing Interface (MPI) specification. Darcy flow is involved in a wide range of problems related to earth sciences, such as groundwater flow, earth filled dams, soil liquefaction, petroleum engineering, etc. These problems involve huge amount of data for analysis. We propose a mesh partitioning algorithm based on reordered quadtree to solve the problem in parallel. The proposed algorithm first partitions the mesh over all processing elements and then parallelizes the system using the MPI specification. The conjugate gradient method augmented with preconditioning is used for the solution of the resulting system of stabilized mixed finite element equations for Darcy flow. Numerical experiments are conducted on a quad core Intel Xeon based computer cluster. Overall, a maximum of 12.24 speedup factor is achieved during the numerical simulations on 28 processors.

Key words: Darcy flow · Matrix bandwidth · Mesh partitioning · MPI · Parallel · Quadtree

INTRODUCTION

Darcy's law mainly focuses on the flow of a viscous fluid in a Permeable Porous Media (PPM), based on the conservation of mass balance equation [1]. It covers the fluid flow with a Reynolds number less than 10. For example, most of the groundwater flow cases come into this category. This fluid flow is termed as Darcy flow. This law is based on the simultaneous computation of pressure and velocity values into PPM. It has many practical applications in the field of petroleum engineering, chemical engineering and earth sciences [2, 3].

Equations of Darcy's law can be solved numerically using existing numerical methods (such as finite difference and element methodologies) with adequate accuracy [4, 5, 6, 7]. However, mass conservation is not guaranteed due to the loss of accuracy in the processes. Masud *et al.* [7] proposed a new stabilized MFEM for Darcy flow, which stabilizes the weak formulation of governing equations using a priori error estimation in the stability norm. This method has also been used as an extension for Darcy-Stokes [6] and discontinuous Galerkin methods by Masud *et al.* [8]. This method works for non-symmetrical stabilized mixed finite element formulations for Darcy flow by adding the adjoint residual form. Loula *et al.* [4, 9] presented symmetrical and stable MFEMs for Darcy flow by combining least-squares residual forms.

The classical mixed variational formulation can be represented in terms of the function spaces L2 (Ω) and H(div, Ω) for pressure and velocity values, respectively. L2 (Ω) is the space of Lebesgue square-integrable functions defined on the domain Ω and H(div, Ω) is the space of Lebesgue square-integrable vector fields, its divergence is also the space of Lebesgue square-integrable. Masud *et al.* [7] proposed the stabilized MFEMs for Darcy using the said classical formulation. Their method accommodates continuous velocity with continuous pressure interpolations and continuous velocity with discontinuous pressure values.

Corresponding Author: Masroor Hussain, Faculty of Computer Science and Engineering,
GIK Institute of Engineering Sciences and Technology, Topi, KPK, Pakistan.

The wide range of applications can be seen in oil and gas exploration, seismic analysis, earth filled dams, soil liquefaction, etc. These problems span over several miles and involve Giga/Tera bytes of data for simulation and analysis. This huge data needs basic infrastructure to solve and store the data structures of meshes for massively parallel computers. Most of the related work for parallel finite element methods has been done using graph based partitioning techniques, based on minimum cut set [10, 11, 12] and octree methods for distributed memory systems [13, 14, 15]. Mitchell [15] has discussed that better partitions are produced using octree based methods over graph based partitioning, Hilbert Space Filling Curve (HSFC), Recursive Coordinate Bisection (RCB) and Recursive Inertial Bisection (RIB).

This paper discusses a novel approach of mesh partitioning to solve the stabilized MFEM for Darcy flow using the MPI specification. The underlying method partitions the mesh based on quadtree and quicksort method as discussed in Hussain *et al.* [16, 17]. Furthermore, we provide a comparative analysis using the proposed partitioning method with the already implemented mesh partitioning algorithm using Parmetis library based on k-way graph partitioning algorithm [10, 18, 19, 20, 21, 22, 23, 24]. The proposed tree based approach method shows better performance in terms of time as compared to Parmetis. Finally, Parmetis is combined with a tree based partitioning method to produce better results in terms of speedup. Numerical experimentations are conducted on a seven node computer cluster based on quad core Intel Xeon processors. A maximum of 12.24 speedup factor is achieved during our numerical experiments.

An outline of this paper is as follows. Section 2 outlines the underlying stabilized mixed finite element form for Darcy flow. Section 3 presents a mesh partitioning algorithms based on quadtree and quicksort methods. Communication of shared nodes is discussed in Section 4. Numerical experiments are presented in Section 5, discussion on numerical results is presented in Section 6 and conclusion is drawn in Section 7.

Problem Formulation: The Boundary Value Problem (BVP) for Darcy flow using stabilized MFEM is taken from Masud *et al.* [7]. Here, the domain $\Omega \subset \Re^{nsd}$ of the BVP is defined by an open bounded region with piecewise smooth boundary (Γ), where n_{sd} denotes the number of spatial dimensions. Mathematically, Darcy's law for the flow of viscous fluid in PPM and conservation of mass are written as:

$$v = - k/\mu \, (\Delta p + \rho/gc \; g) \text{ on } \Omega \quad (1)$$

$$\text{div } v = \varphi \text{ on } \Omega \quad (2)$$

$$v \bullet n = \psi \text{ on } \Gamma \quad (3)$$

where v is the velocity vector, p is the pressure field, g is the gravity vector, φ is the flow rate of sink, ψ is the normal component of the velocity field on the boundary, k is the positive permeability, μ is the positive viscosity, ρ is the positive density, gc is the conversion constant and n is the normal unit vector outwards to boundary (Γ) [7].

The stabilized finite element form can be written as (see [24] for detailed discussion): find $V^h = \{v^h, p^h\} \in L^h$, such that, for all $W^h = \{w^h, q^h\} \in L^h$,

$$B_{stab}(W^h, V^h) = L_{stab}(W^h) \quad (4)$$

where

$$B_{stab}(W^h, V^h) = B(W^h, V^h) + 1/2 \; ((-\mu/k \; w^h + \Delta q^h), k/\mu \; (-\mu/k v^h + \Delta^{@} p^h)) \quad (5)$$

$$L_{stab}(W^h) = L(W^h) - 1/2 \; ((-\mu/k \; w^h + \Delta q^h), k/\mu \; (\rho/g_c \; g)) \quad (6)$$

where $B(W^h, V^h) = L(W^h)$ is the classical finite element formulation of the governing equations (1, 2 and 3) and is defined as:

$$B(W^h, V^h) = (w^h, \mu/ kv^h) - (\text{div} w^h, p^h) + (q^h, \text{div } v) \quad (7)$$

and,

$$L(W^h) = (w^h, \rho/g_c g) + (q^h, \varphi) \quad (8)$$

$(\bullet, \bullet)$ denotes L_2 inner product, which is bilinear, continuous and symmetric. Pre-conditioned Conjugate Gradient (PCG) solver (see for instance [16, 17, 25, 26]) is used to solve the resulting linear discretized system of equations.

Reordered Quadtree Based Mesh Partitioning: Mesh partitioning is an important task for dynamic load balancing of parallel algorithms in distributed memory systems. The partitioning of mesh or graph is an NP-complete problem. Therefore, different heuristic partitioning algorithms have been proposed including those that use minimum cut set of graphs to produce solution of different sizes [10, 19, 22, 23, 24] (Metis and Parmetis libraries) and spatial data structures [13, 4, 15].

A heuristic tree based mesh partitioning algorithm is proposed in this section using greedy approach. Our algorithm uses the following steps to partition the mesh:

- Generate a reordered quadtree.
- Assign the leaves of the quadtree to each processor.

We now introduce a quadtree creation algorithm that minimizes the bandwidth of the matrix and communication during the computation phase. Our algorithm constructs a quadtreeusing positive aspects of quicksort algorithm. It first reorders both the elements and the nodes of the given mesh using the quicksort partitioning algorithm and then recursively partitions the mesh into four sub-meshes using the quadtree approach. The recursive algorithm keeps re-ordering and dividing sub-meshes into further sub-meshes, until the number of elements of a sub-mesh becomes less than or equal to a user defined threshold value (t). The value of t should be equal to or a multiple of the size of the main memory of the distributed memory system. Consequently, the mesh reordering also minimizes the bandwidth of the element stiffness matrix used in the system of linear equations for the numerical solution. On the other hand, a simple quadtree based algorithm only partitions the mesh based on element positions without ordering of the elements.

As a byproduct of the mesh reordering technique using quadtree and quicksort methods, the mesh is partitioned into small areas using a top-down approach. The leaves of quadtree are used to distribute n/p elements of the mesh over all the PEs using first-fit in an anti-clock wise fashion. In the rest of this paper, this partitioning approach will be termed as Quadtree with Quicksort Partitioning (QQP).

Optimization of QQP: QQP may also produce a good initial distribution for Parmetis library to optimize the partitioning using k-way graph partitioning algorithm. Hence, we can combine Parmetis with the proposed QQP partitioning technique to obtain optimized PQQP partitioning scheme.

Algorithm 1: Optimization of QQP partitioning using Parmetis

1. procedure Distribute(Mesh)
2. if *process_id*≥*master then*
3. call Distribute_using_QQP(Mesh)
4. call Send Initial_Distribution_to_All_Slaves (Mesh)
5. {Initialize some of Parmetis input parameters}
6. Partition ← Par Metis_V3_Par Mesh Kway(....)
7. call Redistribute Mesh (Mesh, partition)
8. end if
9. if *process_id* ≥ slave then
10. call Initialize (Mesh)
11. call Receive_Initial_Distribution_from_Master (Mesh)
12. {Initialize some of Parmetis input parameters}
13. Partition ← Par Metis_V3_Par Mesh Kway (....)
14. call Redistribute Mesh (Mesh, Partition)
15. end if
16. end procedure

Algorithm 1 depicts the detailed working of PQQP method. Firstly, an initial mesh is partitioned and distributed using QQP method, which can be seen in line 3. Line 4 distributes the initial mesh (only element information) over all PEs from the master node. This initial partition is, however, optimized in parallel on all the PEs using k-way graph partitioning algorithm in lines 6 and 13 for both the master and slave nodes, respectively. This function generates a new optimized partition for all the PEs in parallel. The mesh information along with the boundary conditions is redistributed over all the PEs in lines 7 and 14.

Communication of Shared Nodes: In an un-structured mesh, there is no deterministic way to find out a shared region. A shared graph is constructed for the shared nodes of a mesh for all the PEs and tasks are assigned using both the first-fit and round robin fashion, as discussed elsewhere [18]. Most of the time, vector reduction function is called for the shared region using MPI_ALLREDUCE function with O(lg(*p*)) time complexity. But only one nodal resultant value is used for computation of the final resultant value. In the worst case, the maximum number of shared nodes n_{sn} can be *n/p*, those are involved in the communication using MPI_ALLREDUCE function, so the total communication overhead is $O(n\lg p/p)$.

Numerical Experimentation: Numerical experiments are conducted to calculate the pressure inside a biunit square as discussed in [24]. The velocity is calculated by the stabilized MFEM of Darcy's law. The values of additional employed parameters are shown in Table 1. Here, the fluid under study is water and porous medium is clay as discussed in [7].

Numerical experiments are conducted on a seven node computer cluster. A gigabit Ethernet switch is used as an interconnection network. Each node contains

Table 1: Values of the parameters for Darcy flow

Serial No.	Parameter	Symbol	Valueused	Units
1.	Dynamic viscosity	μ	3.732×10^{-5}	$kg/(m–s)m^2$
2.	Permeability	k	8.0×10^{-6}	m^2
3.	Density	ρ	1.0	kg/m^3
4.	Gravity along x-axis	gx	0.4	m/s^2
5.	Gravity along y-axis	gx	9.8	m/s^2
6.	Conversion constant	gc	1.0	nil

Table 2: Elements and nodes for the different meshes

Serial No.	Mesh Name	No. of Elements	No. of Nodes
1.	Hollow Square	4342	2389
2.	Small Solid Square	22910	11656
3.	Large Solid Square	92006	46404

a quad core Intel Xeon processor (Thus resulting in total of 28 PEs). All methods are implemented using Fortran programming language and MPI environment. Finite element meshes are generated in ANSYS software.

In order to analyze the speedup behavior with respect to the mesh size, three different meshes are created using unstructured triangular elements (Table 2 for the detail parameters of these meshes). Here, the first mesh contains a hollow circle inside the biunit square and the rest of them are solid biunit squares.

Four noded bi-linear quadrilateral shape functions are employed for the unstructured triangular element, where 3rd and 4th nodes are the same. Full quadrature isused for numerical integration, which is also known as 2 × 2 integration rule [26]. All numerical simulations are conducted 10 times to get anaverage speedup factor using QQP, Parmetis and PQQP techniques.

RESULTS AND DISCUSSION

Speedup factor for the solution of the different 2D biunit square of parallel Darcy flow using QQP, Parmetis and PQQP partitioning schemes is calculated using the serial time of Darcy flow without calling the reordering code (as a base line). Speedup formula is given as:

$$\text{Speedup}=\frac{\text{Serial time without using reordering}}{\text{Parallel time}}$$

Figure 1 shows the growth of speedup factor for the hollow biunit square. In this problem PQQP does not show good results as compared to QQP and Parmetis. Detailed growth of the speedup factor during the solution of the biunit solid square for the parallel stabilized MFEM for Darcy flow is given in Figure 2. The experiments are conducted on 28 PEs of the computer cluster. The speedup factor decreases after 26 PEs for both the small and the large meshes of the biunit solid square. This might be due to the overhead involved in computer cluster due to scheduling and running of other services.

We observe that Parmetis produces 2-8% better speedup for the hollow biunit square mesh as compared to other meshes. However in other meshes, QQ Pproduces 2-6% better speedup as compared to the result produced by both Parmetis and PQQP techniques. We also observe a saw behavior of speedup factor in Figure 2 of the small mesh on the result using both the Parmetis and PQQP approach. This behavior was introduced by Parmetis due to unequal distribution of elements over all the PEs. This effect can't be seen in the large mesh due to its smaller effect on the results.

Table 3 shows the maximum speedup factor of all the three meshes. These results clearly suggest that the better speedup is produced by employing QQP partitioning as compared to Parmetis and PQQP partitioning. This follows from the fact that QQP decreases both the cache miss rate and the bandwidth of matrix using its novel reordering technique discussed in [16, 17]. The minimized bandwidth also reduces the communication time by decreasing the number of shared nodes of boundary elements. It is also observed that speedup factor is increased by increasing the size of a mesh, as it reduces the communication to computation ratio with fixed number of PEs.

The average bandwidth (B) measure is also used to analyze the results of employed QQP, Parmetis and PQQP partitioning methods. Table 4 shows the average bandwidth of all the distributed meshes using eight PEs. The table clearly shows that overall QQP produces less bandwidth and leads to better results as compared to Parmetis and PQQP methods for the current case study. Non-uniform distribution of the bandwidth values is also observed for QQP and PQQP partitioning schemes due to underlying un-structured triangular elements of the mesh. Both the QQP and the PQQP methods use the spatial information. However, Parmetis exhibits a linear behavior for average bandwidth values, as it does not account for the spatial information.

The isoefficiency function [27] of the parallel stabilized MFEMs for Darcy flow is $\Omega(n_{pe}^{m})$ due to the PCG solver and usage of the MPI_ALLREDUCE function for the communication. This function helps to analyze the scalability of a parallel algorithm. Thus, the problem size should be at least of the order of $\Omega(n_{pe}^{m})$, in order to maintain a fixed efficiency.

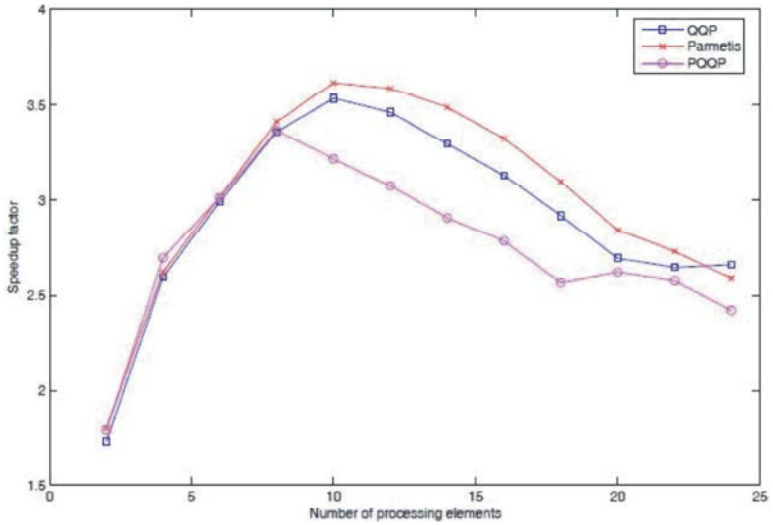
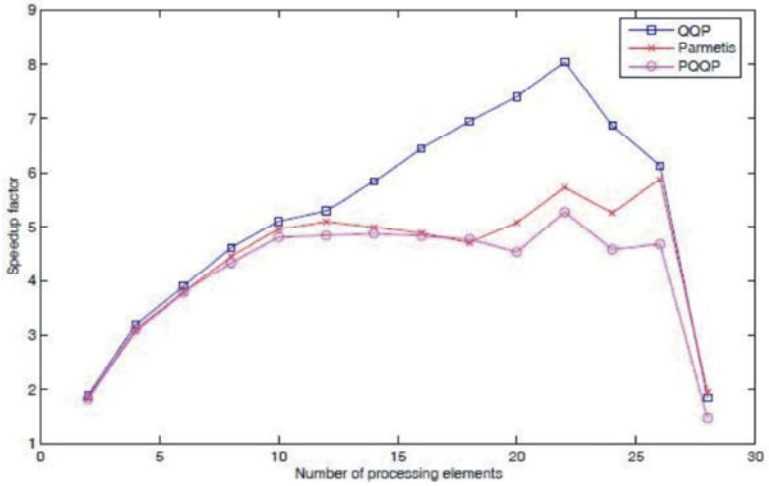
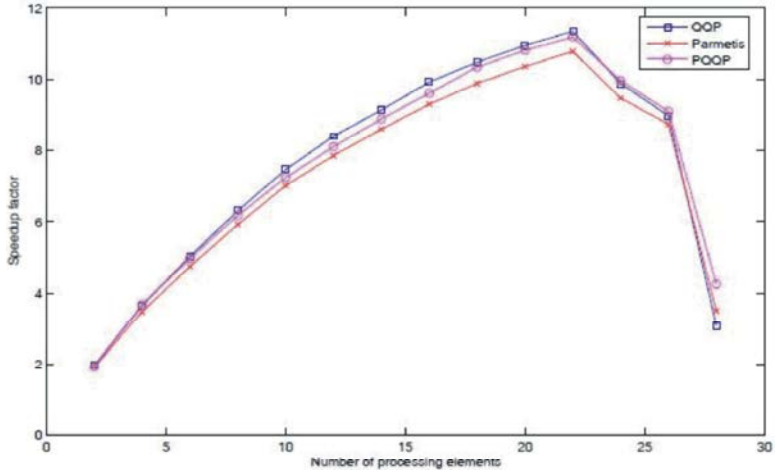


Fig. 1: Increase in speedup factor for 2D square containing hollow circle inside



(a) Small Mesh.



(b) Large Mesh.

Fig. 2: Increase in speed up factor of 2D solid squares

Table 3: Maximum speed up factor with the different partitioning algorithms

Mesh Type	QQP	Parmetis	PQQP
Hollow Square	3.78	3.86	3.56
Small Solid square	8.31	7.91	8.16
Large Solid Square	12.24	11.46	11.65

Table 4: The average bandwidth values using different partitioning schemes

	Hollow square			Small solid square			Large solid square		
PEs	QQP	Parmetis	PQQP	QQP	Parmetis	PQQP	QQP	Parmetis	PQQP
1	21.24	72.81	24.29	38.91	29.90	50.22	74.70	82.68	92.24
2	52.17	109.19	34.28	78.91	33.40	69.51	149.51	84.86	127.93
3	66.80	101.43	34.89	78.34	121.21	35.25	146.62	252.16	74.78
4	55.54	106.24	102.28	121.55	145.02	237.43	237.89	319.84	504.34
5	73.33	89.52	69.43	178.18	260.61	129.59	366.98	432.51	188.31
6	42.71	154.03	49.44	81.38	293.85	157.80	153.76	762.19	435.89
7	76.16	193.73	136.73	226.50	407.62	246.59	456.89	922.58	532.09
8	123.20	189.68	85.28	98.25	402.59	136.25	206.38	849.42	268.49

CONCLUSION

We have presented a novel parallel implementation of stabilized MFEM for Darcy flow using the MPI specification. Our algorithm uses a recursive tree based mesh partitioning based on reordering, recursive bisections and quicksort methods. The said mesh partitioning method produces better speedup factor as compared to other well known partitioning technique using multilevel k-way graph/mesh partitioning. A maximum 12.24 speedup for the large solid mesh is achieved using 28 PEs of a quad core Intel Xeon based cluster for the parallel stabilized mixed finite element method for Darcy flow. These results are verified using average bandwidth values.

The said partitioning scheme can be incorporated to any FEM solver to solve a variety of problems related to ALE moving mesh, analysis of earth filled dams and groundwater, oil and gas explorations, seismic analysis, soil liquefaction problem and other related FEM problems. This paper discusses the results using serial QQP for MPI environment, as mesh database is not distributed. Better scalability can be obtained if a parallel QQP will be implemented for distributed meshes, which will be addressed in a future work.

ACKNOWLEDGEMENTS

Support for this work is provided by the Pak-US project under HEC Pakistan and NSF USA (2006-2009). Support of School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, for experimental work is acknowledged for the numerical simulation over the computer cluster. Support of colleagues at FCSE, GIKI is also acknowledged in completing this work. Special thanks to Prof. Alex Kavokin, Prof. Arif Masud and Prof. Ashfaq Khokhar during understanding of the domain knowledge.

REFERENCES

1. Ertekin, T., J.H. Abou-Kassem and G.R. Kingm 2001. Basic Applied Reservoir Simulation, volume 7 of SPE monograph series. Society of Petroleum Engineers.
2. Ahmad, T.M., 2006. Pressure analysis of heterogeneties in oil reservoir system - A case study of lower Indus basin Sindh. Master's thesis, Petroleum and Natural Gas Engineering at Mehran University of Engineering and Technology, Jamshoro.
3. Hussain, M. and A. Kavokin, 2009. A 2D parallel algorithm using MPICH for calculation of ground water flux at evaporation from water table. In International Conference on Fron-tiers of Information Technology 2009, pages 68: 1-68: 4, New York, NY, USA, Dec. 2009. ACM.
4. Correa, M.R. and A.F.D. Loula, 2008. Unconditionally stable mixed finite element methods for Darcy flow. Computer Methods in Applied Mechanics and Engineering, 197(17-18): 1525-1540.
5. Kanschat, G. and B. Riviere, 2009. A strongly conservative finite element method for the coupling of stokes and darcy flow. Journal of Computational Physics, 229(17): 5933-5943.

6. Masud, A., 2007. A stabilized mixed finite element method for Darcy-Stokes flow. International Journal for Numerical Methods in Fluids, pp: 665-681.
7. Masud, A. and T.J.R. Hughes, 2002. A stabilized mixed finite element method for Darcy flow. Computer Methods in Applied Mechanics and Engineering, 191: 4341-4370.
8. Hughes, T.J.R., A. Masud and J. Wan, 2006. A stabilized mixed discontinuous Galerkin method for Darcy flow. Computer Methods in Applied Mechanics and Engineering, 195: 3347-3381.
9. Loula, A.F.D. and M.R. Correa, 2006. Numerical analysis of stabilized mixed finite element methods for Darcy flow. In III European Conference on Computational Mechanics Solids, Structures and Coupled Problems in Engineering.
10. Balman, M., 2006. Tetrahedral mesh refinement in distributed environments. In Proc. Supercomputing 93: 497-504. IEEE, Aug. 2006.
11. Walshaw, C. and M. Cross, 2000. Mesh partitioning: A multilevel balancing and refinement algorithm. SIAM J. Sci. Computing, 22(1): 63-80.
12. Karypis, G., 2003. Multi-constraint mesh partitioning for contact/impact computations. In Proc. Supercomputing '2003, pp: 497-504. IEEE Computer Society Washington, DC, USA.
13. Flaherty, J.E., R.M. Loy, M.S. Shephard, B.K. Szymanski, J.D. Teresco and L.H. Ziantz, 1997. Adaptive local refinement with octree load-balancing for the parallel solutionof three-dimensional conservation laws. Journal of Parallel Distributed Computing, 47: 139-152.
14. Flaherty, J.E., R.M. Loy, M.S. Shephard, B.K. Szymanski, J.D. Teresco and L.H. Ziantz, 1998. Parallel structures and dynamic load balancing for adaptive finite element computation.Applied Numerical Mathematics, 26(2): 241-263.
15. Mitchell, W.F., 2007. A refinement-tree based partitioning method for dynamic load balancing with adaptively refined grids. Journal of Parallel and Distributed Computing, 67(4): 417-429.
16. Hussain, M., M. Abid, M. Ahmad, A. Khokhar and A. Masud, 2011. A parallel implementation of ALE moving mesh technique for FSI problems using open MP. International Journal of Parallel Programming, 39: 717-745, 10.1007/s10766-011-0168-3.
17. Hussain, M., M. Ahmad, M. Abid and A. Khokhar, 2009. Implementation of 2D parallel ALE mesh generation technique in FSI problems using OpenMP. In International Conference on Frontiers of Information Technology, 18: 1-18: 6, New York, NY, USA, Dec.2009. ACM.
18. Gullerud, A.S. and R.H.D. Jr., 2001. MPI-based implementation of a PCG solver using an EBE architecture and preconditioner for implicit, 3D finite element analysis. Computer and Structures, 79: 553-575.
19. Gupta, A., 1996. Fast and effective algorithms for graph partitioning and sparse matrix reordering. IBM Journal of Research and Development, 41(1/2): 171-183.
20. Karypis, G. and V. Kumar, 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific Computing, 20: 359-392.
21. Karypis, G. and V. Kumar, 1998. Multilevel k-way partitioning scheme for irregular graphs. Journal of Parallel and Distributed Computing, 48: 96-129.
22. Karypis, G. and V. Kumar, 1999. Parallel multilevel k-way partitioning scheme for irregular graphs. SIAM Journal on Industrial and Applied Mathematics, 41(2): 278-300.
23. Schloegel, K., G. Karypis and V. Kumar, 2000. Parallel multilevel algorithms for multi-constraint graph partitioning. In Lecture Notes In Computer Science, 1900: 296-310. Springer-Verlag London, UK, 2000.
24. Schloegel, K., G. Karypis and V. Kumar, 2002. Parallel static and dynamic multi-constraint graph partitioning. Concurrency and Computation: Practice and Experience, 14: 219-240.
25. Hughes, T.J.R., R.M. Ferencz and J.O. Hallquist, 1987. Large-scale vectorized implicit calculations in solid mechanics on a cray X-MP/48 utilizing EBE preconditioned conjugate gradients. CMAME, 61(2): 215-248.
26. Masud, A., M. Bhanabhagwanwala and R.A. Khurram, 2007. An adaptive mesh rezoning scheme for moving boundary flows and fluid-structure interaction. Computer and Fluids, 36: 77-91.
27. Grama, A.Y., A. Gupta and V. Kumar, 1993. Isoefficiency: Measuring the scalability of parallel algorithms and architectures. IEEE Parallel and Distributed Technology, 1(2): 12-21.