COMMUNICATIONS IN NUMERICAL METHODS IN ENGINEERING
Commun. Numer. Meth. Engng 2008; **24**:1009–1017
Published online 11 June 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cnm.1007

Parallel Delaunay triangulation for particle finite element methods

Yannis Fragakis*, † and Eugenio Oñate

International Center for Numerical Methods in Engineering (*CIMNE*), *Technical University of Catalonia*, *Edificio C1*, *Campus Norte*, *Gran Capitan s/n*, *Barcelona 08034*, *Spain*

SUMMARY

Delaunay triangulation is a geometric problem that is relatively difficult to parallelize. Parallel algorithms are usually characterized by considerable interprocessor communication or important serialized parts. In this paper, we propose a method that achieves high speed-ups, but needs information regarding locally maximum element circumspheres prior to the beginning of the algorithm. Such information is directly available in iterative methods, like the particle finite element methods. The developed parallel Delaunay triangulation method, has minimum communication requirements, is quite simple and achieves high parallel efficiency. Copyright © 2007 John Wiley & Sons, Ltd.

Received 6 April 2006; Revised 17 February 2007; Accepted 22 March 2007

KEY WORDS: Delaunay triangulation; particle finite element method; parallel computing

1. INTRODUCTION

Delaunay triangulation is a geometric problem often encountered in a wide spectrum of scientific areas, like terrain modelling, robotics, finite element methods, etc. It is defined as the generation of elements connecting a set of points, in a way that the interior of the circumsphere of each element does not contain any of the points. In the 3D space the elements are tetrahedral, while in two dimensions they are triangular. Among various methods that have been proposed over the years for the Delaunay triangulation, many perform very efficiently in serial processing. However, the parallelization of Delaunay algorithms presents considerable difficulties. The algorithms cannot be easily divided in parts that can be run concurrently in parallel processors. Thus, parallel methods usually require a lot of interprocessor communication, which reduces their parallel efficiency.

*Correspondence to: Yannis Fragakis, International Center for Numerical Methods in Engineering (CIMNE), Technical University of Catalonia, Edificio C1, Campus Norte, Gran Capitan s/n, Barcelona 08034, Spain.
†E-mail: fragayan@cimne.upc.edu

Contract/grant sponsor: Spanish Ministry of Education and Science

The present paper describes a new parallel Delaunay triangulation method that requires very little interprocessor communication. This method is applicable to problems where the maximum circumsphere size of the elements can be bounded prior to the triangulation. Such problems are encountered in iterative methods that require a triangulation of a set of points in each iteration and thus allow to bound the circumsphere size from information available from the positions of the points at the previous iteration. In particular, the parallelization of the particle finite element method (PFEM), which is an iterative method that contains the above characteristics, has been the principal motivation for the present research. The sharp circumsphere bounds that are available in this method made the development of a parallel Delaunay algorithm possible, quite simple, efficient and applicable in any parallel computer architecture, thanks to its relatively small communication requirements.

The PFEM is a recently developed Lagrangian method for solving computational fluid dynamics and fluid structure interaction problems with moving free surfaces, flow separations, moving solids within liquid masses, etc. [1, 2]. It combines the ideas originating from particle methods, finite element methods and meshless methods. In more detail, the movement of the nodes of the fluid is monitored in time in the fashion of particle methods. At the end of each time step, the nodes are moved to their new position and the next time step begins with the Delaunay triangulation of the cloud of nodes, in order to form a finite element mesh for solving the fluid flow. The shape functions of the elements are formed in the fashion of the meshless methods and free surfaces are detected with the alpha shape method that discards oversize elements, following a local size criterion.

The present study is organized as follows. Section 2 overviews a previous research in parallel Delaunay triangulation, while Sections 3 and 4 present the new approach. Section 5 presents computational results and the concluding remarks of the paper are outlined in Section 6.

2. PARALLEL DELAUNAY TRIANGULATION ALGORITHMS

The proposed method is quite distinct from previous lines of research in parallel Delaunay triangulation. A brief outline of previous research will however be useful. Both serial and parallel algorithms in this scientific area are usually based either on the principle of 'incremental construction' or on 'incremental insertion', thus allowing the distinction of two general categories of methods. Methods based on 'incremental construction' start from one element and gradually form the adjacent elements, satisfying the circumsphere criterion. Examples of parallel methods based on this strategy can be found in [3–7]. These methods are usually complemented with the divide and conquer strategy, forming concurrently partial triangulations that are merged together in a final phase. Methods following 'incremental insertion' start with a virtual element that contains all points. Each point is inserted consecutively and the triangulation is locally updated to satisfy the circumsphere criterion. Parallel methods using this principle can be found in [8–11].

The methods proposed in the above papers are characterized of considerable variety, because of the difficulty in parallelizing the Delaunay algorithms. They vary, for instance, in the range of application of the algorithms. Some are applied only in the 2D case [5, 6], while others are designed specifically for shared memory parallel computers [11]. Achieved speed-ups and level of complexity also vary.

DOI: 10.1002/cnm

3. PROPOSED STRATEGY

The basic idea for the proposed algorithm is borrowed from overlapping domain decomposition methods. Let the computational domain be divided in overlapping subdomains and each subdomain assigned to a separate processor. In this paragraph, it will be explained that if the overlap exceeds a certain length, the correct Delaunay triangulation will be obtained in parallel for the whole domain with no need of merging the partial triangulations. This argument will be explained with the help of Figure 1 that shows the Delaunay triangulation of a small set of nodes in the area near the boundary between two subdomains. Figure 1(a) shows the triangulation that is the target of the parallel simulation (this example has been obtained from a larger problem and only the area near the interface between two subdomains is shown). The area between axes a–a and c–c is considered to be the overlapping area between the subdomains, while axis b–b lies in equal distance between a–a and c–c. The distance d is equal to the diameter of the maximum element circumcircle in this area of the mesh. We name 'primary area' of each subdomain, the area of each subdomain that lies within axis b–b. The rest is named 'secondary area'. Accordingly, subdomain nodes are distinguished in primary and secondary.

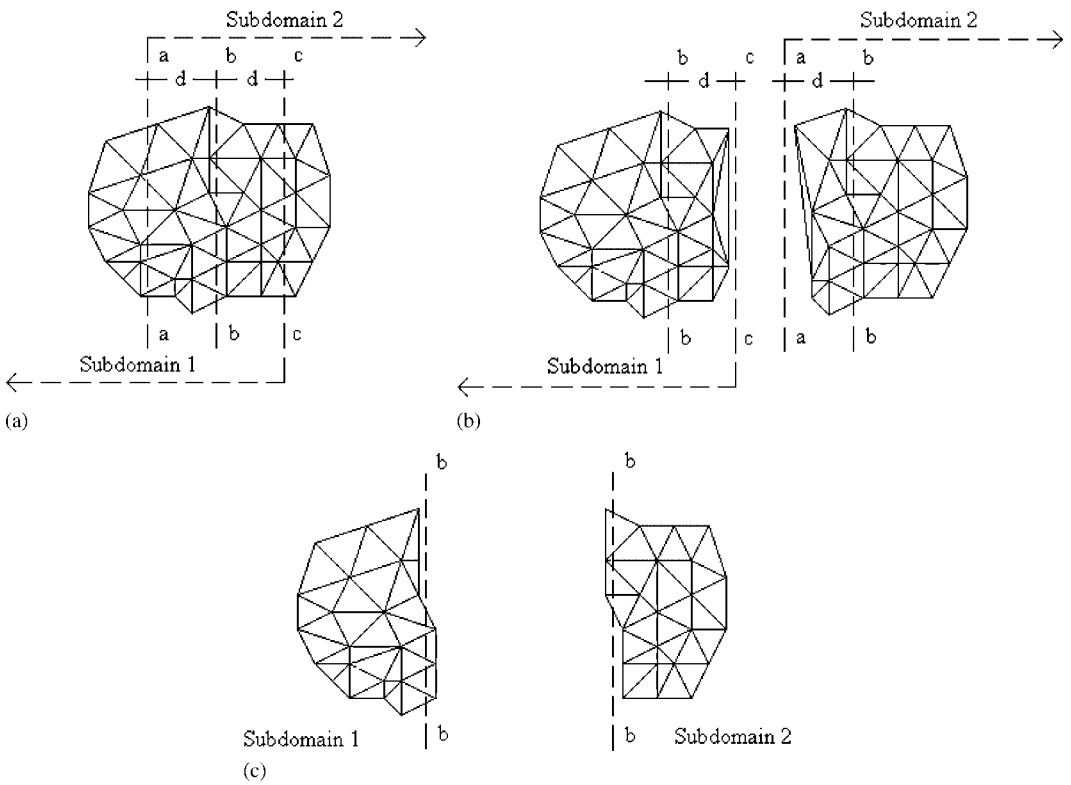

Figure 1. (a) A part of the Delaunay triangulation of a set of nodes, in the vicinity of the interface between two subdomains; (b) independent Delaunay triangulation of the subdomains; and (c) final mesh of the subdomains.

DOI: 10.1002/cnm

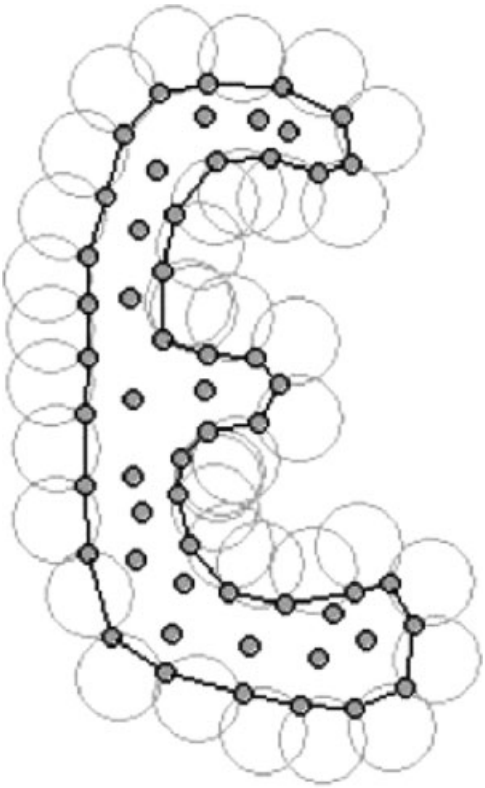

Figure 2. Contour recognition.

At the beginning of the parallel algorithm, each subdomain is assigned the nodes that lie within its limits. Then, the independent Delaunay triangulation of each subdomain is obtained (Figure 2(b)). It is observed that near the limit of each subdomain, non-Delaunay elements are obtained, if the entire node set is considered. However, the elements with at least one primary node are correct, because it is guaranteed that their circumcircle does not cross the limits of the subdomain. Based on the above remarks, in order to obtain the full Delaunay triangulation of this set of nodes, we only need to define a criterion that decides which elements of each subdomain should be discarded. In Figure 2(c), for instance, we have chosen to discard from each subdomain the elements whose centroid lies within the secondary area of the subdomain.

4. APPLICATION OF THE PROPOSED METHOD IN THE PARTICLE FINITE ELEMENT METHOD

The proposed strategy is tested for obtaining Delaunay triangulations in the analyses of PFEM. Here follow some details of the implementation of the above parallel triangulation method for problems of the PFEM.

DOI: 10.1002/cnm

As it was noted in the introduction, at the beginning of each time step of the PFEM, a Delaunay triangulation of the cloud of nodes of the analysis is performed. In parallel analysis, we start with a partitioning of the nodes in overlapping subdomains. We have chosen to partition the nodes in regions of orthogonal parallelepiped shape with approximately equal number of nodes, as it will be shown in the numerical results of the paper. The generated regions have an as much as possible cubic shape, in order to obtain a relatively small amount of overlap. After partitioning, the nodes are communicated to the processors and subdomain triangulations are performed in parallel.

It is also worth describing how the maximum circumsphere diameter is defined *a priori* in the PFEM. In order to explain this issue, we need to describe the free-surface detection mechanism: considering that points follow a variable $h(x)$ distribution, where $h(x)$ is the minimum distance between two points and x is the position vector, the following criterion is used for boundary surface detection: all points on an empty sphere with a radius bigger than $\alpha h(x)$ are considered as boundary points (Figure 2). α is a parameter close to, but greater than one. It is noted that this criterion is coincident with the alpha shape concept [12].

Hence, when a Delaunay triangulation creates elements with a circumsphere radius higher than $\alpha h(x)$, these elements are removed in order to form the boundary surface. Therefore, we can define the minimum required overlap in the parallel triangulation method, setting $d \geqslant 2\alpha h(x)$ (where d has been defined in Figure 1). It is also worth noting that since $h(x)$ depends on the position x, the bound d can also vary in space, thus taking into account heterogeneous densities in the point distribution. In our implementations, we set $\alpha = 1.2$ and $d = 2\alpha h(x)$. It now remains to explain how $h(x)$ is estimated before the actual triangulation. From the triangulation of the previous time step, we know for each point its nearest point. We also know the new positions of the points, after their displacement during the previous time step. Hence, we obtain an upper limit for $h(x)$ taking the new distance of each point with the point which was closest to it at the previous time step. Given that there are relatively small displacement of the points in consecutive time steps, this upper limit will be quite sharp and overlapping will be just little higher than the absolutely necessary.

Finally, it is worth noting that the parallelization process allows for the use of any serial triangulation method for the subdomains, provided that any degenerate cases of the Delaunay triangulation encountered in the overlapping region are treated by the involved subdomains in a similar way. Degenerate are considered the cases where four or more points (three or more points in 2D) are encountered on the surface of a circumsphere, thus allowing more than one viable triangulations. To be precise, an incompatibility in the meshes that are generated by neighbouring subdomains can occur only when there is a degenerate case that is composed of both primary and secondary nodes of the subdomains (in Figure 1 this would mean that the circumsphere of this degenerate case crosses axis b–b). In order to obtain the same solution for this degeneracy in the involved subdomains, either the Delaunay triangulation method that is used for obtaining the mesh of the subdomains has to produce directly the same solution of this degenerate case in the involved subdomains, or the subdomains that share this degenerate case have to communicate at the end of the triangulation, so that the same solution is used by all the subdomains.

In our implementation, the subdomains are treated by a method using randomized incremental insertion and at the end of the algorithm if degenerate cases composed of both primary and secondary nodes of the subdomains have been detected, the involved subdomains communicate in order to use the same solution. Usually such degenerate cases are very few, but even in the unlikely case that there are many, they can be grouped in a single message that is exchanged between neighbouring subdomains. In general, the communication cost would thus be negligible.

DOI: 10.1002/cnm

We believe that a deterministic way of producing the same solution for shared degeneracies between subdomains could also be built, thus giving a more elegant solution to this problem, but we have not looked into this issue yet.

5. NUMERICAL TESTS

For the numerical tests we use a 2D and a 3D problem of the PFEM. We choose a time step and perform the triangulation in various numbers of processors. The chosen problems have a relatively small size (smaller than 100k nodes), because of two reasons: (i) smaller size problems pose more difficulties in achieving high speed-ups and (ii) the most time-consuming PFEM analyses that we perform usually do not reach more than 100k nodes, because the required large number of time steps increases excessively the computational cost. The tests are performed on a cluster of 8 AMD Athlon 1700 computers interconnected with 100 Mbit Ethernet. Interprocessor communication costs are practically insignificant in our analyses, because of very low communication requirements of the employed method.

5.1. 2D example

The first numerical test consists of the Delaunay triangulation of a 2D point set obtained in a time step of the analysis shown in Figure 3. At the beginning of this analysis, a water mass is supported by a vertical barrier. The barrier is removed and water is let to fall. When water reaches the box, this starts moving under the pressure forces. The box reaches an obstacle and overturns. We present the computational time required for the triangulation performed at the time step that

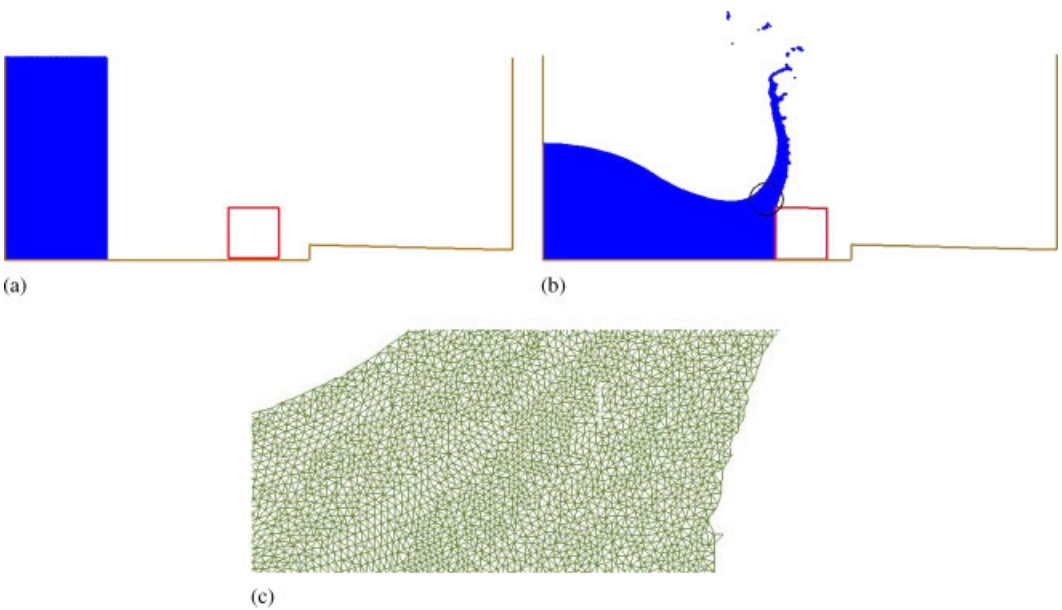

Figure 3. (a) A box and a water mass supported by a vertical barrier within a container; (b) water falls on the box and pushes it; and (c) detail of the Delaunay triangulation in the area above and to the left of the box (see the black circle in (b)). The mesh boundaries are defined by the alpha shape method.

DOI: 10.1002/cnm

Table I. CPU time and parallel efficiency for the 2D example analysis.

Number of subdomains	Time (s)	Overlap index (%)	Parallel efficiency (%)
2	1.84	95	95
4	0.96	92	91
8	0.53	84	82

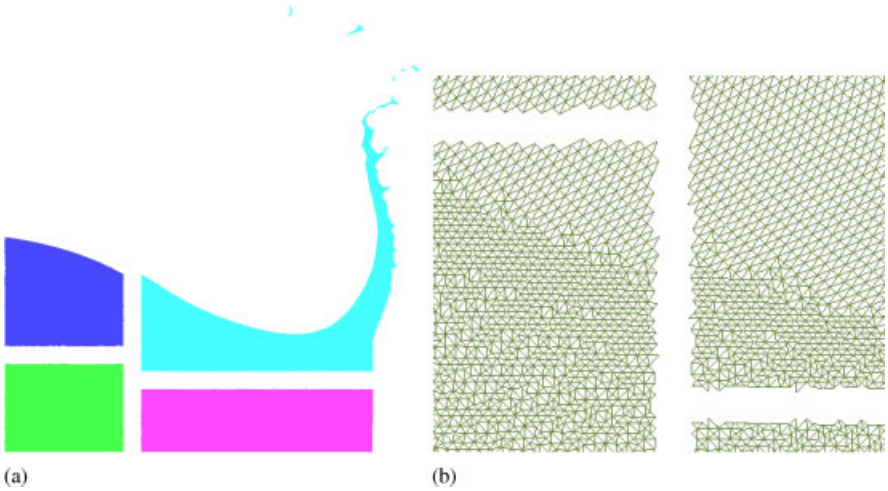

Figure 4. (a) Decomposed triangulation of the 2D example in four subdomains and (b) detail of the decomposed triangulation.

begins at the time instant shown in Figure 3(b). At this instant, water has just reached the box and splashes, while the box is moving to the right. The problem is discretized with approximately 80k nodes and its serial triangulation takes 3.48 s.

Table I shows the CPU times and parallel efficiency obtained for parallel analysis, while Figure 4 shows the final subdomain meshes obtained in the four-processor case. Table I also presents the values of an overlap index that is defined as follows. If N denotes the total number of nodes of the given problem divided by the number of subdomains, while M denotes the number of nodes of the larger subdomain, then we define an overlap index as N/M. In fact, this index would not be a good measure of overlap, if the given problem was not partitioned in overlapping subdomains of almost equal number of nodes. However, in the tests that are presented in this paper, there are very small discrepancies in the number of nodes of the subdomains, so that any imbalances do not affect importantly the above index. In Table I, it is observed that the overlap index is closely related to the parallel efficiency that is measured in each analysis, which is quite reasonable to expect. Parallel efficiency is naturally diminished when we add more processors. Nevertheless, it stays above 80%, which suggests a relatively good use of the parallel processing resources.

5.2. 3D example

Our 3D test case is shown in Figure 5. A column of water is left to fall on a set of tetrapods. While water passes through, the tetrapods are displaced, but their relative positions remain more or less

DOI: 10.1002/cnm

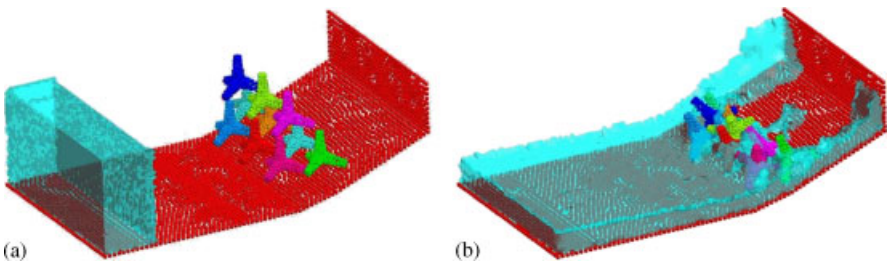

Figure 5. (a) A column of water and a set of tetrapods and (b) water splashes on the tetrapods.

Table II. CPU time and parallel efficiency for the 3D example analysis.

Number of subdomains	Time (s)	Overlap index (%)	Parallel efficiency (%)
2	3.32	94	104
4	1.76	87	98
8	0.97	79	89

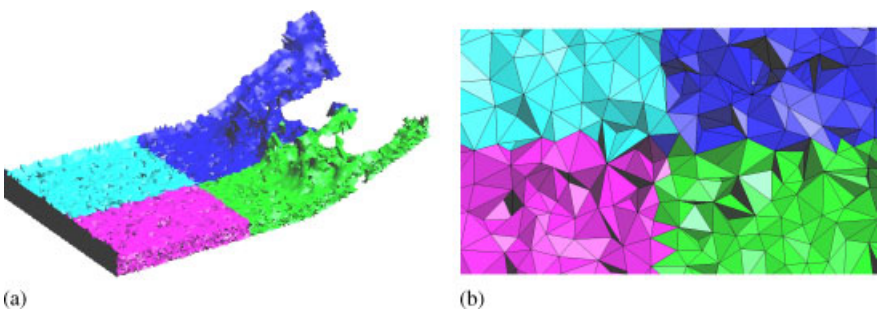

Figure 6. (a) Decomposed triangulation of the 3D example in four subdomains and (b) detail of the decomposed triangulation (top surface of the mesh in the intersection of the four subdomains).

the same. As in the previous example, we choose a typical time step, whose initial node positions are shown in Figure 5(b). This problem is discretized with approximately 45k nodes and the time required for the triangulation of the given point set in serial computation is 6.88 s. The gains from parallel solutions are shown in Table II, while the final subdomain meshes for four processors are shown in Figure 6. Parallel efficiency measurements give values higher than 85%. In addition, it is worth noting that the two-processor solution results show that a serial solution of this problem using two subdomains would be faster than the normal process. This is due to the fact that the cost of the employed serial Delaunay triangulation algorithm is approximately proportionate to $n \log(n)$, where n is the number of points. Therefore, the total time required for the triangulation of the two overlapping subdomains in serial turns out to be less than the time required for the triangulation of the full problem. However, gains from using multi-subdomain solutions in serial processing are clearly limited.

DOI: 10.1002/cnm

6. CONCLUSION

The efficiency of parallel Delaunay triangulation algorithms is usually restricted from considerable interprocessor communication requirements or serialized computation costs. In this paper, we have proposed a method with minimum communication requirements, taking advantage of prior information regarding maximum element circumspheres. Our approach is directly applicable to iterative methods like the PFEM, where maximum element circumsphere diameters can be sharply bounded from previous iteration information. Achieved parallel efficiencies are high, because of small communication requirements and few serialized parts in the proposed algorithm.

ACKNOWLEDGEMENTS

The authors would like to thank Facundo Del Pin for his help in initial stages of this work and Miguel Angel Celigueta and Monica De Mier for their help in the numerical tests. During this project, the first author was supported from the program: 'Support for mobility of Spanish and foreign professors and researchers' of the Spanish Ministry of Education and Science.

REFERENCES

1. Idelsohn SR, Oñate E, Del Pin F. The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves. *International Journal for Numerical Methods in Engineering* 2004; **61**:964–989.
2. Idelsohn SR, Oñate E, Del Pin F, Calvo N. Fluid–structure interaction using the particle finite element method. *Computer Methods in Applied Mechanics and Engineering* 2005, in press (available online).
3. Aggarwal A, Chazelle B, Guibas L, O'Dunlaig C, Yap C. Parallel computational geometry. *Algorithmica* 1988; **3**:293–327.
4. Cignoni P, Montani C, Perego R, Scopigno R. Parallel 3D Delaunay triangulation. *Computer Graphics Forum* 1993; **12**:129–142.
5. Hardwick JC. Implementation sand evaluation of an efficient parallel Delaunay triangulation algorithm. *Proceedings of the 9th Annual ACM Symposium on Parallel Algorithms and Architectures* (*SPAA '97*), Newport, RI, U.S.A., 23–25 June 1997.
6. Chen MB, Chuang TR, Wu JJ. Efficient parallel implementations of 2D Delaunay triangulation with high performance Fortran. *Proceedings of 10th SIAM Conference on Parallel Processing for Scientific Computing*, Portsmouth, VI, U.S.A., 12–14 March 2001.
7. Lee S, Park CI, Park CM. An improved parallel algorithm for Delaunay triangulation on distributed memory parallel computers. *Parallel Processing Letters* 2001; **11**:341–352.
8. Chrisochoides N, Sukup F. Task parallel implementation of the Bowyer–Watson algorithm. *Proceedings of the 5th International Conference on Numerical Grid Generation in Computational Fluid Dynamic and Related Fields*, Mississippi State University, U.S.A., 1–5 April 1996.
9. Chrisochoides N, Nave D. Simultaneous mesh generation and partitioning for Delaunay meshes. *Proceedings of the 8th International Meshing Roundtable*, South Lake Tahoe, CA, U.S.A., 10–13 October 1999.
10. Okusanya T, Peraire J. 3D parallel unstructured mesh generation. *Trends in Unstructured Mesh Generation* (ASME) 1997; **AMD-220**:109–115.
11. Kohout J, Kolingerova I, Zara J. Parallel Delaunay triangulation in E2 and E3 for computers with shared memory. *Parallel Computing* 2005; **31**:491–522.
12. Edelsbrunner H, Mucke EP. Three-dimensional alpha-shape. *ACM Transactions on Graphics* 1994; **3**:43–72.

DOI: 10.1002/cnm