ISFV15-053-S20

15th International Symposium on Flow Visualization
June 25-28, 2012, Minsk, Belarus

ACCELERATION OF SOME PRE- AND POSTPROCESSING ALGORITHMS FOR CFD DATA ON 3-D UNSTRUCTURED GRIDS

PETER VOINOVICH

The 3-D unstructured grids based on tetrahedral grid elements are widely used in CFD simulations for decades. The modern computer hardware offers resources sufficient to deal with the grids containing many tens or even hundreds of millions of the grid objects N (grid nodes, tetrahedra, triangular faces, edges) using common personal computers, and several orders of magnitude more using powerful computer installations. Some intuitive algorithms for pre- and/or postprocessing of the grid and CFD data result in computational complexity of $O(N^2)$, which makes impractical their application to the large grids. Here belongs creation of data sets for grid edges or faces from the data base for tetrahedra referring to the grid nodes as their vertices. Extraction of triangular domain boundary faces (surface triangulation) from a nodal-tetrahedral data base represents another typical example. This can be done using a temporary data set for all the triangular faces of the grid tetrahedra. For every tetrahedron, each of its 4 faces is compared to the existing faces in the current data set: if the face already exists, it is marked as an internal face of the computational domain, if not – the face is added to the data set. Finally, all the unmarked faces belong to the domain boundary, as they were encountered only once, in contrast to the internal faces which separate two adjacent tetrahedra. The above algorithm has a complexity of O(Ntetrahedra*Nfaces), i.e. $O(N^2)$. This makes it extremely inefficient in postprocessing, especially when an animated image is created from the data on a grid which varies in time, as in the case of adaptively refined grids applied to a transient problem. Hash-indexing the faces can essentially reduce the number of comparisons for each face to be tested. Several types of hash functions and hash tables were implemented. A drastic improvement in computational efficiency was achieved using a three-dimensional hash table and indexing the faces according to the Cartesian coordinates of their centers. However, even better result was obtained using a one-dimensional hash table and a hash function based on the numbers of the grid nodes corresponding to the face vertices. This reduced the computational complexity of the above algorithm to O(Ntetrahedra).