Proceedings of ENCIT 2010
Copyright © 2010 by ABCM

13th Brazilian Congress of Thermal Sciences and Engineering
December 05-10, 2010, Uberlândia, MG, Brazil

A COMPUTATIONAL STUDY OF WENO SCHEMES USING A TOPOLOGICAL DATA STRUCTURE

Fernanda Paula Barbosa, fernandapaulab@gmail.com
Antonio Castelo Filho, castelo@icmc.usp.br
José Alberto Cuminato, jacumina@icmc.usp.br
Instituto de Ciências Matemáticas e de Computação, USP - ICMC, São Carlos, SP, 13566-590, Brazil

João Luiz F. Azevedo, azevedo@iae.cta.br
Instituto de Aeronáutica e Espaço, CTA - IAE - ALA, São José dos Campos, SP, 12228-903, Brazil
Carlos Breviglieri, carbrevi@yahoo.com.br
Instituto Tecnológico de Aeronáutica, CTA - ITA - IEC, São José dos Campos, SP, 12228-900, Brazil

Abstract. *The present paper addresses the issue of computational flexibility when using weighted essentially nonoscillatory (WENO) schemes in unstructured meshes. The WENO schemes are indeed able to provide high order of accuracy on general unstructured grids. However, computational costs tend to rise quite sharply as the order of accuracy is increased due to the obvious increase in the size of the computational molecule and, at least partially, due to the inherent additional search work required to perform calculations with such increased molecule in unstructured grids. In such context, the work will explore the possibility of isolate WENO schemes by coupling a standard cell centered, unstructured grid, finite volume method with an implicit topological data structure, the Mate Face, which will handle all the grid-related functions and operations. The computations are performed considering compressible flow of a perfect gas, modeled by the 2-D Euler equations, written in conservation-law form. Spatial discretization uses a WENO scheme, implemented for a cell centered, finite volume method on general unstructured grids. Time march uses explicit Runge-Kutta schemes for solution advancement in time.*

Keywords: *WENO Schemes, Topological Data Struture, Finite Volume Method*

1. INTRODUCTION

The advances in information technology and experimental methods have allowed the numerical and experimental study of more complex phenomena in Computational Fluid Dynamics (CFD), particularly for unsteady problems. The need for numerical simulation in CFD is justified by the lack of analytical solutions of the Navier-Stokes equations for most practical cases. One of the most common aspects in problems treated through CFD simulations is the domain geometrical complexity. The computational domain representation becomes the one of the first issues to be solved. Therefore, the geometrical modeling represents an important step, because it has as one of its main goals the adequate computational representation of spatial decompositions.

The motivation for the present paper addresses the issue of computational flexibility when using high order of accuracy weighted essentially nonoscillatory (WENO) schemes in unstructured meshes. Previous work by some of the present authors (Wolf and Azevedo, 2006) has indicated that WENO schemes are indeed able to provide high order of accuracy on general unstructured grids. However, computational costs tend to rise quite sharply as the order of accuracy is increased due to the obvious increase in the size of the computational molecule and, at least partially, due to the inherent additional search work required to perform calculations with such increased molecule in unstructured grids. Moreover, one should remember that WENO schemes are characterized by adaptive computational molecules (Wolf, 2006), which may change from one iteration to the next, for the same grid point or control volume. Therefore, such identification and search operations have to be performed at every iteration, or time step, of the time marching scheme, even for steady state problems (Wolf, 2006).

In such context, the work will explore the possibility of improving the computational scheme of high-order WENO schemes by coupling a standard cell centered, unstructured grid, finite volume method with an implicit topological data structure, the Mate Face (Cunha, 2009), which will, then, handle all the grid-related functions and operations. The computations are performed considering compressible flow of a perfect gas, modeled by the 2-D Euler equations, written in conservation-law form. Spatial discretization uses a WENO scheme (Wolf and Azevedo, 2006), implemented for a cell centered, finite volume method on general unstructured grids. Time march uses explicit Runge-Kutta schemes for solution advancement in time.

The Mate Face data structure, (Cunha, 2009), Cunha *et. all.*, 2008) is used to represent the mesh, controlling every single access to the mesh points by providing a complete set of iterators and operators. The Mate Face is an implicit topological data structure specialized in representing meshes formed by different elements, such as triangles, quadrilaterals, or even mixed mesh elements. Its internal representation of adjacent elements provides fast searching of elements and data points. As each technology is currently built using different code implementations, an integration of the simulator (Fortran) and the data structure (C++) is made at compilation time. This integration is performed by using an interface that translates and controls each call to methods in the C++ code back to the Fortran code. The interface code is a set of functions prepared to provide easy access by the simulator code. All modules are compiled together in memory in order to provide best performance when passing information through them.

2. THEORICAL FORMULATION

The 2D Euler equations in conservative form (Anderson, 1991), can be written as

- Continuity

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0, \tag{1}$$

- Momentum in the x direction

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} + \frac{\partial p}{\partial x} = 0, \tag{2}$$

- Momentum in the y direction

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2)}{\partial y} + \frac{\partial p}{\partial y} = 0, \tag{3}$$

- Energy

$$\frac{\partial E}{\partial t} + \frac{\partial[u(E+p)]}{\partial x} + \frac{\partial[v(E+p)]}{\partial y} = 0, \tag{4}$$

where u and v are, respectively, the flow velocity in the x and y directions, p is pressure, ρ is density and E is total energy, given by:

$$E = \rho \left[e + \left(\frac{u^2 + v^2}{2} \right) \right], \tag{5}$$

and e is internal energy which, for ideal gases is given by:

$$e = \frac{R}{\gamma - 1} T, \tag{6}$$

where T is absolute temperature and is the specific heat ratio equal 1.4. In order to allow a more stable and direct solution procedure, especially for flows with supersonic regions, which is the case of transonic flows, Equations (1) to (4) were rewritten in compact form as:

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0, \tag{7}$$

where Q is the vector of conserved variables, F and G are the convective flux vectors given by

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \quad F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{bmatrix}, \quad G = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ v(E+p) \end{bmatrix}, \tag{8}$$

where vector Q represents the desired result. The flux variables allow the calculation of other variables of interest, such as pressure.

3. DISCRETIZATION

The two-dimensional domain is represented by a mesh, manipulated by a topologic data structure, formed with cells defining linear triangles elements. The finite volume method was used to discretize the equations, as showed in the following.

4. NUMERICAL FORMULATION

In the present work, the 2D Euler equations are solved in their integral form as

$$\frac{\partial}{\partial t}\int_V \mathbf{Q} dV + \int_V (\nabla \cdot \mathbf{P})\, dV = 0, \tag{9}$$

where $\mathbf{P} = \mathbf{F}\vec{i} + \mathbf{G}\vec{j}$. Appying the Gauss theorem in Equation (9), we have

$$\frac{\partial}{\partial t}\int_V \mathbf{Q} dV + \int_V (\mathbf{P} \cdot \mathbf{n})\, dS = 0, \tag{10}$$

where V represents the volume of the cell, S represents the surface of the cell and $\mathbf{n}$ is the unit vector normal to the surface S. The Equation (10), discretized in a finite volume context centered in cells, can be rewritten for a control volume i as

$$\frac{\partial \mathbf{Q_i}}{\partial t} = -\frac{1}{V_i}\int_{S_i} (\mathbf{P} \cdot \mathbf{n})\, dS, \tag{11}$$

where $\mathbf{Q_i}$ is the mean value of $\mathbf{Q}$ at time t, in the control volume V_i, defined as

$$\mathbf{Q_i} = \frac{1}{V_i}\int_{V_i} \mathbf{Q} dV. \tag{12}$$

Therefore, a final formulation for a discretization of Euler equations in two dimensions inside a finite volume context is of form

$$\frac{\partial \mathbf{Q_i}}{\partial t} = -\frac{1}{V_i}\sum_{k=1}^{nf}\left(\mathbf{F_k}\vec{\mathbf{i}} + \mathbf{G_k}\vec{j}\right) \cdot \mathbf{S_k}, \tag{13}$$

where nf is the number of faces, or edges in the two dimensions space, of the volume control.

The control volumes considered in this work are triangles and they can be decomposed into a finite number of line segments Γ_j. One should observe that the control volumes could be composed by any type of polygon, because the really important aspect is that its bounding contour could be decomposed into a finite number of line segments. The surface integral from Equation (10) can be discretized using N -point Gaussian integration formulae

$$\int_{S_i} (\mathbf{P} \cdot \mathbf{n})\, dS \approx \sum_j |\Gamma_j| \sum_{l=1}^{N} w_l \mathbf{P}\left(\mathbf{Q_j}(\mathbf{G_l}), t\right) \cdot \mathbf{n}, \tag{14}$$

where $\mathbf{G_l}$ and w_l are, respectively, the Gaussian points and the weights on the Γ_j line segment. For the second-order accuracy scheme just one Gauss point is necessary for the integration. Given the coordinates of the points in the corner of the control volume face, z_1 and z_2, it can be obtained the centroid point of the face,

$$\mathbf{G_1} = \frac{\mathbf{z_1} + \mathbf{z_2}}{2}. \tag{15}$$

In the case, the weight, w_1, is chosen as $w_1 = 1$. For the third-order schemes, two Gaussian points are necessary along each line segment. Their values are given by

$$G_1 = \frac{\sqrt{3}+1}{2\sqrt{3}} z_1 + \left(1 - \frac{\sqrt{3}+1}{2\sqrt{3}}\right) z_2 \qquad \text{and} \qquad G_2 = \frac{\sqrt{3}+1}{2\sqrt{3}} z_2 + \left(1 - \frac{\sqrt{3}+1}{2\sqrt{3}}\right) z_1, \tag{16}$$

where the respective weights, w_1 and w_2 are given by $w_1 = w_2 = 1/2$

Using the method described above, one can compute values of Q_i in some instant, t, and then, from these mean values, one can reconstruct polynomials that represent the primitive variables ρ, u, v and p. Finally, it is possible to compute values of the conserved variables in the Gaussian points. Due to the discontinuity of the reconstructed values of the conserved variables over the cell boundaries, one must use a numerical flux function to approximate these flux values on the cell boundaries. In this paper, we have used the Roe flux difference splitting method (Roe, 1981) to compute such approximations. An explicit Runge-Kutta scheme of five stages with second order of accuracy was used to obtain the time solution of the governing equations, proposed by Mavriplis (Mavriplis, 1988).

5. RECONSTRUCTIONS OF ENO AND WENO SCHEMES

The process of reconstruction of ENO schemes relies on the approximation of mean values of the primitive variables for cells in the mesh. The approximation is performed by using polynomials of one order less than the expected spatial order of accuracy. In order to construct polynomials of η-th order, $N(\eta)$ cells must be used, and $N(\eta) = (\eta+1)(\eta+2)/2$. For obtaining the polynomial reconstruction for each cell, first one has to define the possible set of cells, named a stencil, in which will be used. The stencils can be selected in a von Neumann neighborhood, in a finite volume cell centered scheme, for a linear polynomial reconstruction (second-order accuracy). This approach can be extended to higher orders through the use of von Neumann neighborhoods of the primary neighbors already selected for the second-order reconstruction. In this paper, the cells will be represented by triangles and the polynomials $p(x, y)$ are calculated as

$$p(x, y) = \sum_{|\beta|\leq\eta} r\beta_1\beta_2 \left(x - x_c\right)^{\beta_1} \left(y - y_c\right)^{\beta_2}, \tag{17}$$

where $|\beta| = \beta_1 + \beta_2$, with $\beta_i \in 0, 1, 2, \ldots, x_c$ and y_c are the Cartesian coordinates of the barycenter of the control volumes and $r\beta_1\beta_2$ are unknown coeficients which are some approximations to the derivatives of the primitive variables.

Once a good approximation to the mean values of primitive variables of each cell is established on $p(x, y)$, it is possible to write a linear system, $[R]r = \bar{u}$, of $N(\eta)$ equations for the $N(\eta)$ unknowns, $r\beta_1\beta_2$. Notice that $[R]$ is the matrix of control volume moments (the same as in Gooch (Gooch, 1997)), calculated using the scaling technique proposed by Friedrich (Friedrich,1998). Beyond that, r is the vector of unknown coeficients that must be found and $\bar{u}$ is the vector composed by the mean values for each primitive variable. Moreover, if $[R]$ matrix is invertible, the stencil is considered admissible. The control volume moments that compose the $[R]$ matrix are defined as

$$x^{\varphi}\bar{y}^{\zeta} \equiv \frac{1}{V_i}\int_{S_i} (x - x_c)^{\varphi}(y - y_c)^{\zeta} dS, \tag{18}$$

and are evaluated using Gauss quadrature formulae following the same procedure as the one used in the flux computation.

For each cell, after the polynomial reconstruction is done, the following step is to verify which polynomial is the least oscillatory to use in the ENO scheme. The oscillation is computed by using an indicator that assesses the smoothness of $p(x, y)$. Following the results presented in the literature (Friedrich, 1998, Wolf and Azevedo, 2005), the oscillation indicator used is the one proposed by Jiang and Shu (Jiang and Shu, 1996), which was later modified by (Friedrich, 1998). The formulation for this oscillation indicator is

$$OI_{JS}(p(x, y)) = \left[\sum_{1\leq|\beta|\leq\eta} \int_{V_i} h^{2|\beta|-4} \left(\frac{\partial^{|\beta|}p(x, y)}{\partial x^{\beta_1}\partial y^{\beta_2}}\right)^2 dxdy\right]^{\frac{1}{2}}, \tag{19}$$

where h is the mesh width.

Unlike ENO schemes, WENO schemes use all the calculated polynomials. These polynomials are added together through the use of weights which are computed for each one of the polynomials proportional to its respective oscillation indicator. The main idea in the WENO reconstruction is to set the computed weights for each polynomial aiming at reconstructing a new polynomial to $p(x, y) = \sum_{k=1}^{m} w_k p_k(x, y)$. The weights are of order one in the smooth regions of the flow and the order of the desired accuracy in the solution in the regions with discontinuities. The weights are computed as

$$w_k = \frac{\left[\epsilon + OI(p_k(x, y))\right]^{-\theta}}{\sum_{k'=1}^{m} \left[\epsilon + OI(p_{k'}(x, y))\right]^{-\theta}}, \tag{20}$$

where ϵ is a small real number used to avoid division by zero and θ is a positive integer. The WENO schemes have the property of being very smooth and stable in smooth regions of the flow, but this property is lost if θ is chosen as a large value.

6. THE MATE FACE DATA STRUCTURE - MF

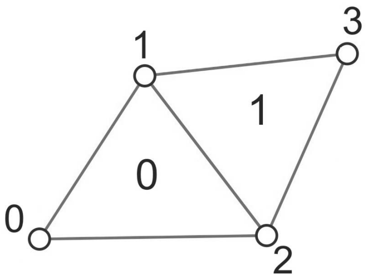
The Mate Face (MF) is an implicit topological data structure to storage unstructured meshes. The MF implements operators and iterators to manipulate meshes. The MF data structure allows the representation of two-dimensional meshes (2D or superficial) and three-dimensional meshes. For two-dimensional meshes, they can be simple or hybrid (triangle and/or quadrilateral), and for volumetric meshes they are simple (tetrahedron, prism, pyramid or hexahedron).

The MF is a flexible data structure that can represent different mesh types, including meshes with different element types, as triangles and quadrilaterals. The MF has a technique of partial allocations of vectors in order to represent vector of elements. In this way, memory blocks are dynamically set, as new elements are added in the mesh. This strategy eases the problem of memory space for vector of elements.

The vertexes store geometric information about the mesh, and each one is associated to position of the space, base for the most of geometric operation. Each vertex in the MF also store a reference to its mate cell through an identification number. The singularity of vertex are also implemented, facilitating diverse query types as star queries and operations on non convex meshes.

The edges in MF are represented explicitly, and also stores the adjacent cells that share it. This representation has the advantage that information can be directly stored in the edge, as for example the id of adjacent cells of the edge.

The MF also represents cells, storing references to vertexes, edges (and faces in the three-dimensional case). Beyond that, mate cells can be retrieved through incident edges or faces. The Figure 1 shows an example of the representation of two cells in a two-dimensional space.



Cell 0 :

Vertexes:	0	1	2
Edges :	01	12	20
Neighbors:	-	1	-

Cell 1:

Vertexes:	1	3	2
Edges :	13	32	21
Neighbors:	-	-	0

Figure 1. Representation of triangle cell in MF.

One consideration about the organization in memory of mesh elements in two-dimensional meshes is that for each cell, the number of vertexes is equal to the number of edges and neighbors. This characteristic eases the implementation and representation of relationships of neighborhood. By using a counter clock orientation, every neighbor cells are indexed in the mesh.

Another advantage of the MF is the support to the loading of diverse file formats that defines meshes. The advantage is that the simulator code can load any mesh of any format available. Currently, the following types are allowed: VTK [1] , OFF [2], VRML, PLY [3] e HEX [4].

Beyond storage and representation, the MF also provides a set of traversing operators. The traversing operators are methods that traverses the mesh from a start element and visit other elements following a specific criteria. These operators are implemented in the form of iterators. Simple operators visit sequentially every vertex, cell or edge in the mesh while complex iterators visit cells around a vertex or cell, for example.

The iterator that visits around a vertex (called star query) has different behavior depending on the type of the query center. The different types of traversing routes can be seen in Figure 2, which consider an internal vertex or a boundary vertex.

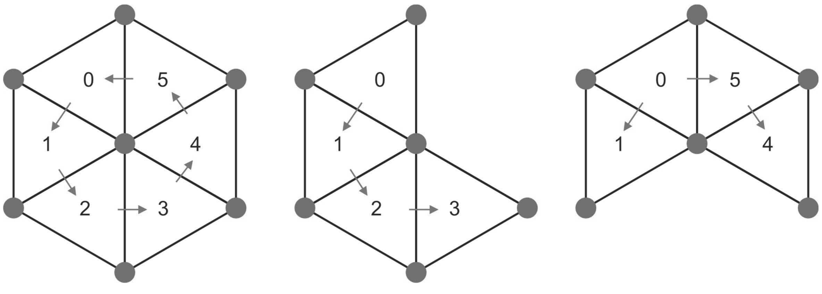


Figure 2. Star query around a vertex with initial cell zero.

Besides traversing queries, geometric operators are also available, and are very used in the simulation code, such the distance between two points, the area of a cell, the localization of a given query point (like inside a triangle or on the edge of it), the coordinates of a centroid of a cell, among others.

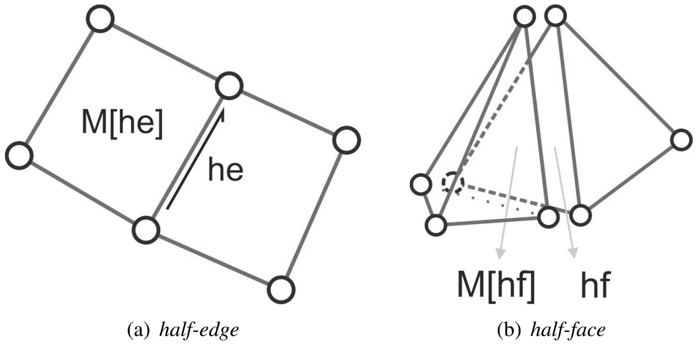
The representation of neighborhood of cells in the MF is performed by using half-edges in superficial meshes and half-faces in volumetric meshes. Figure 3 shows the approach of using half-edges in quadrilateral cells (a) and by using half-faces in tetraedric cells (b).

[1] VTK - Site: http://www.vtk.org

[2] Geomview - Site : http://www.geom.uiuc.edu/software/geomview

[3] PLY - Polygon File Format - Site: http://local.wasp.uwa.edu.au/∼pbourke/dataformats/ply

[4] Unstructured Hexahedral Meshes - Site: http://www.cc.gatech.edu/∼lindstro/data/hexzip



(a) *half-edge* (b) *half-face*

Figure 3. Neighborhood cells in MF.

7. TECHNOLOGIES INTEGRATION

As separated research results, the simulator program and the data structure library were developed at different programming languages. In order to the simulation program (FORTRAN code) makes use of the data structure library (C++ code), an integration scheme was performed at compilation time. The proposed solution for the integration of technologies was the development of a mapping layer, acting as an interface between technologies, which maps operations of the Mate Face into functions that can be used by the simulator program. Figure 4 shows the integration scheme used to map and link both programs. The interface translate and controls every access to the mesh by the simulator program. All information exchanged is made at runtime between both codes using main memory for sharing data.

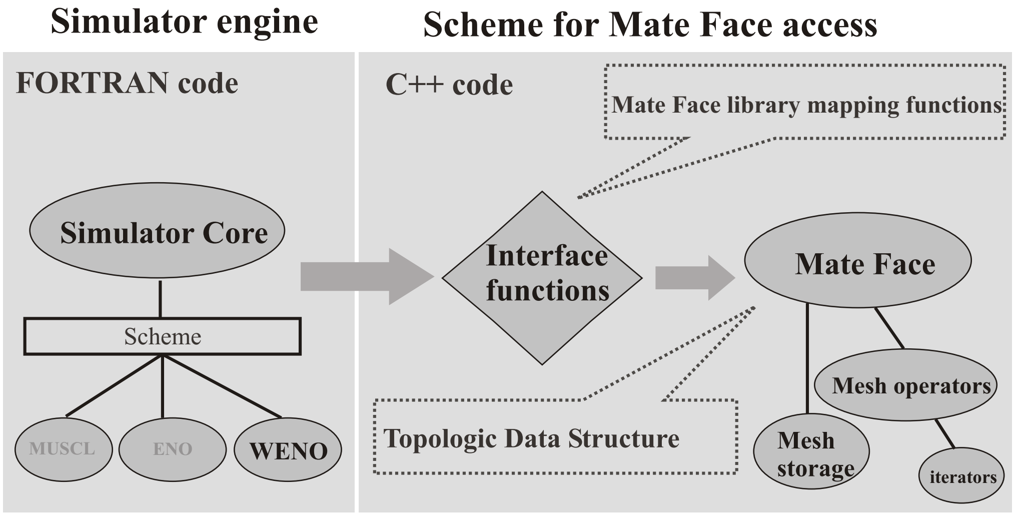


Figure 4. Integration scheme of simulator program (Fortran 77 (WENO)) and Mate Face structure library (C++).

The interface is defined as a set of functions in such a format that can be called by the simulator program. Every call of functions in the interface share a controlled portion of main memory to pass information in and out. The processed information is formatted by the compiler, which performs a correct alignment of bytes in memory. By calling a set of functions in the interface, the simulator program can access the information of mesh, either performing all operations available in the Mate Face structure.

There are many advantages of centering all the mesh information and access in a data structure. One advantage is that the data structure is specialized in mesh control and consumes less memory to represent the same information previously stored inside the simulator program. Instead of wasting memory and time searching lots of tables of redundant mesh information, a single data structure optimizes and centralizes the mesh usage. The new scheme separates the mesh representation from the simulator core code.

The interface layer maps all the iterators available in the Mate Face structure, organized in sets of functions to start, accesses current information, moves to the next item, and verifies if the iterator has ended. Examples would be simple queries of the type “loop over all cells (edges, vertexes)” to more complex queries such “retrieve all cells in the star (neighborhood) of a given vertex”. Beyond such queries, the interface provides common mesh information such as area value of the triangles of the mesh.

Another advantage of using a data structure is that Mate Face supports hybrid cells, such as triangles together with quadrilaterals, in the same mesh. All elements of the mesh will be automatically managed along with the proper iterators. Future functionalities include moving meshes controlled by the data structure triggered by simulator program data.

8. NUMERICAL RESULTS

The test case is the computation of the supersonic flow field past a wedge with half-angle $\theta = 10^o$. The initial mesh has 816 nodes and 1504 volumes, as can be seen in Figure 5. The leading edge of the wedge is located at coordinates $x = 0.25$ and $y = 0.0$. The computational domain is bounded along the bottom by the wedge surface and by an outflow section before the leading edge. The inflow boundary is located at the left and top of the domain, while the outflow boundary is located ahead of the wedge and at the right of the domain.

The analytical solution gives the change in properties across the oblique shock as a function of the free stream Mach number and shock angle, which is obtained from the $\theta - \beta - Mach$ relation. For this case, a free stream Mach number of $M_1 = 5.0$ was used, and the oblique shock angle β is obtained as 19.5^o. For the analytical solution, the pressure ratio is $p2/p1 \approx 3.083$ and the Mach number past the shock wave of $M_2 \approx 3.939$. The results from the computational calculus were $p_2/p1 \approx 3.04$ and $M_2 \approx 3.92$. The result shown in Figure 6 was obtained by the WENO scheme of second order of accuracy using the Roe method, scheme of second order of precision of Runge-Kutta.

The same information of mesh inside the simulator before the usage of the data structure is represented in the Mate Face. In this way, as the mesh information is the same, the results of simulation will be also the same. The change is where the mesh information is stored. Although there is an overhead of mixing both technologies, there is a trade off between flexibility and performance. The next experiment magnifies this trade off, where the performance was measured comparing the new strategy with the previous one.

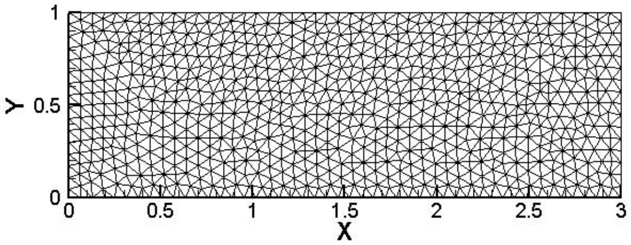
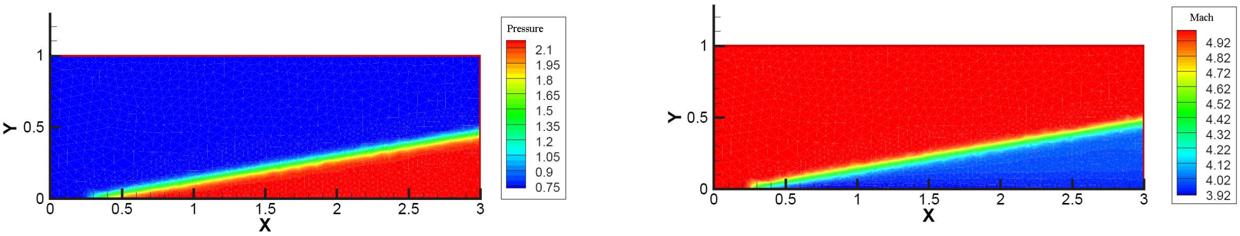


Figure 5. Initial mesh used in simulation.



(a)

(b)

Figure 6. Pressure (a) and Mach number contours, (b) for supersonic wedge flow.

Figure 7 shows the time spent comparing the use of the data structure with the mesh represented in tables inside simulator, analyzing the WENO routine. At each step of the simulation the time spent for processing the routine was measured, considering the first 50 steps of the simulation. It is very important to notice that a data structure provides mesh independence and guarantee no redundant information. Beyond that, a lot of operations are supported in the data structure, including future simulations of moving meshes. By analyzing the figure, we can see that the overhead of having an interface between the two technologies is just a few milisseconds per step. But the gain in the flexibility of having a separated data structure is acceptable, once many running researches in grid generation and optimization can be applied directly in the data structure, including support for moving meshes methods, without any intervention to the numerical code.

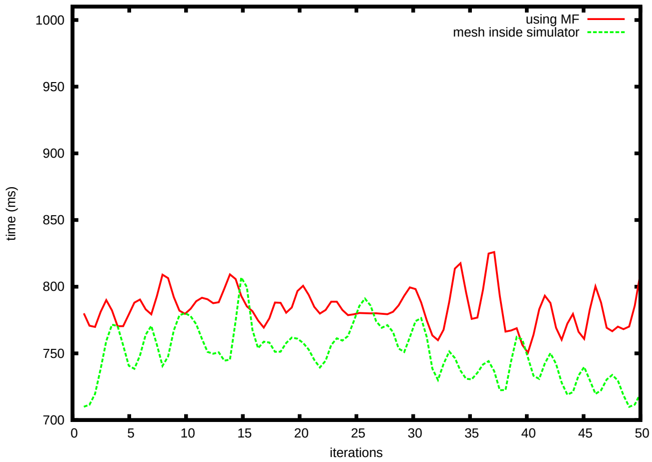


Figure 7. Time spent to compute WENO scheme at each step.

9. CONCLUSION

This paper explored the possibility of implementing WENO schemes by coupling a standard cell centered, unstructured grid, finite volume method with an implicit topological data structure, the Mate Face, which handled grid-related functions and operations.

There are advantages of centering all the mesh information and access in a data structure, as the specialized controll and the minimized memory cost to represent the same information previously stored inside the simulator program by many tables and vectors. Instead of wasting memory and time searching lots of tables of redundant mesh information, a single data structure centralized the mesh usage. The new interface scheme separated the mesh representation from the simulator core code and managed all passing information.

With the simulation code independent of the mesh control, future work can explore new methodologies on meshes, including to support moving meshes with little adaptation on the simulation code.

10. ACKNOWLEDGEMENTS

The authors acknowledge the partial support provided to the present research by Fundação de Amparo à Pesquisa do Estado de São Paulo, FAPESP, under Grant No. 2008/01544-6 and partial support for the present work was also provided by Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, under the Research Project Grant No. 312064/2006-3 and 301408/2009-2 and it is greatly appreciated.

11. REFERENCES

Anderson, J. D., 1991, “Fundamentals of Aerodynamics”, 2nd edition, McGraw-Hill.

Cunha, I. L. L., 2009, “ Estrutura de dados Mate Face e aplicações em geração e movimento de malhas”. PhD thesis, Dissertação de Mestrado, Universidade de São Paulo.

Cunha, I. L. L., Lacassa, A. de ; Polizelli-Jr., V. ; Gois, J. P. ; Nonato, L. G. ; Castelo Filho, A, 2008, “Adaptive Algebraic Mesh Generation from Implicit Functions”, Iberian Latin American Congress on Computational Methods in Engineering, Maceió, Brazil.

Friedrich, O., 1998, “Weighted Essentially Non-Oscillatory Schemes for the Interpolation of Mean Values on Unstructured Grids”, Journal of Computational Physics, Vol. 144, No. 1, pp. 194-212.

Gooch, C. F. O., 1997, “High Order ENO Schemes for Unstructured Meshes Based on Least-Squares Reconstruction”, Argonne National Laboratory, Report No. P631-1296, Mathematics and Computer Science Division.

Jiang, G. S., and Shu, C. W., 1996, “Eficient Implementation of Weighted ENO Schemes”, Journal of Computational Physics, Vol. 126, No. 1, pp. 77-99.

Mavriplis, D. J., 1988, “Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes”, AIAA Journal, Vol.26, No.7, pp. 824–831.

Roe, P. L., 1981, “Approximate Riemann Solver, Parameter Vectors, and Difference Schemes, Journal of Computational Physics, Vol.43, No.2, pp. 200–212.

Wolf, W.R., and Azevedo, J.L.F., 2006, “High-Order Unstructured Essentially Nonoscillatory and Weighted Essentially Nonoscillatory Schemes for Aerodynamic Flows”, AIAA Journal, Vol. 44, No. 10, pp. 2295-2310.

Wolf, W.R., 2006, “Compressible Aerodynamic Flow Simulations Using High-Order Non-Oscillatory Schemes in Unstructured Grids”, Master Thesis, Instituto Tecnológico de Aeronáutica, São José dos Campos, SP, (in Portuguese).

Wolf, W. R., and Azevedo, J. L. F., 2005, .High-Order Unstructured Grid ENO and WENO Schemes Applied to Aerodynamic Flows,. AIAA Paper No. 2005-5115, Proceedings of the 17th AIAA Coputational Fluid Dynamics Conference (CD-ROM), Toronto, Canada.

12. Responsibility notice

The author(s) is (are) the only responsible for the printed material included in this paper