Parallel computations of one-phase and two-phase flows using the MecaGRID

Stephen F. Wornom

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

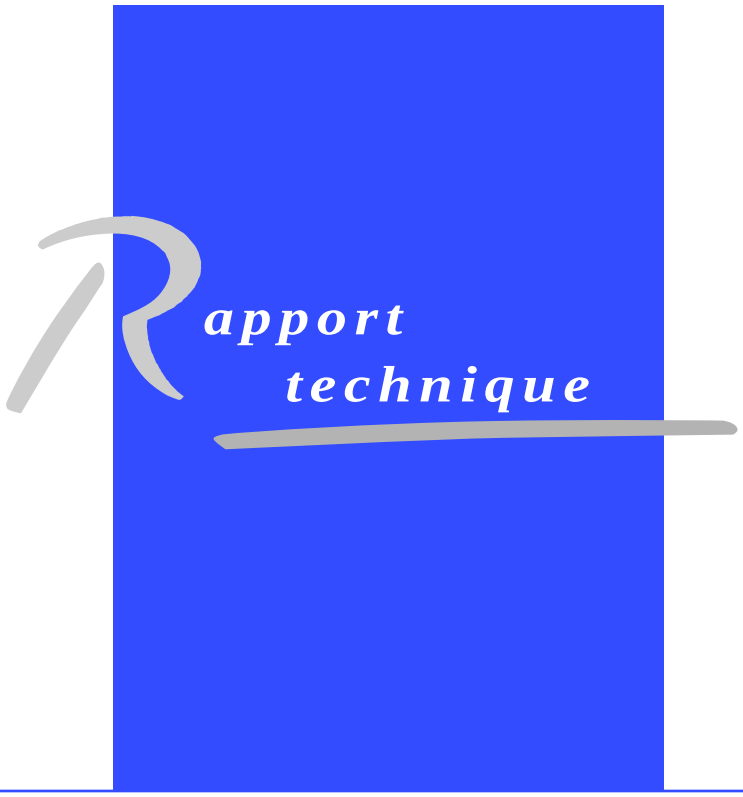
Parallel computations of one-phase and two-phase flows using the MecaGRID

Stephen Wornom

N° 0297

August 2004

Thème NUM



ISSN 0249-0803

Parallel computations of one-phase and two-phase flows using the MecaGRID

Stephen Wornom*

Thème NUM — Systèmes numériques
Projet Smash

Rapport technique n° 0297 — August 2004 — 51 pages

Abstract: The present report examines the application to Grid Computing of two fluid simulation codes. The first code, AERO-F, simulates aerodynamic flows for external and internal non-reacting flows. The second code, AEDIF, simulates two-phase compressible flows. The work examines the application of these codes executing on parallel processors using the Message Passing Interface (MPI) on the MecaGRID that connects the clusters of INRIA Sophia Antipolis, Ecole des Mines de Paris at Sophia Antipolis, and the IUSTI in Marseille. Methods to optimize the MecaGRID applications accounting for the different processor speeds at the different sites and different RAM sizes are presented. The Globus Alliance software is used for the Grid applications. Suggestions for future research are given

Key-words: Grid Computing, Globus, MPICH-G2, computational fluid dynamics, two-phase flow

* INRIA, 2004 Route des Lucioles, BP. 93, 06902 Sophia-Antipolis, France

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Calculs parallèles sur MecaGRID d'écoulements mono et diphasiques

Résumé : Ce rapport étudie l'application du calcul sur grille de deux logiciels de Mécanique des fluides. Le premier, AERO-F, traite d'écoulements monophasiques d'aerodynamiques externe et interne. Le second logiciel simule des écoulements diphasiques. Ce travail examine l'application de ces logiciels utilisant MPI sur les installations de MecaGRID, connectant les clusters de l'INRIA à Sophia-Antipolis, de l'Ecole des Mines de Paris à Sophia-Antipolis, et de l'IUSTI à Marseille. Plusieurs méthodes visant à optimiser ces applications MecaGRID par la prise en compte des vitesses de processeurs et des mémoires disponibles sur les différents sites sont propoées. Le logiciel de grille de Globus Alliance est utilisé dans ces applications MPI. Le rapport se termine avec des suggestions d'investigations futures.

Mots-clés : Grille de calcul, Globus, MPICH-G2, Mécanique des fluides numérique, écoulement multi-phase

1 Introduction

The development of Grid Computing began in the mid 1990s. The idea is to make available different computing resources, data resources, and other computer-related expertise on a computational Grid in order to realize scientific projects that would otherwise be impossible or not practical at a single computing site. One could envision, for example, thousands of scientists in many different locations in the world pooling their resources to analysis the data of a major experiment at the European high energy physics laboratory in CERN, Switzerland (The European Data Grid or EDG[1]). Another scenario would involve the coupling of codes developed at different sites each with specific expertises.

Grid Computing is a difficult challenge because the needed technology was not initially available. Progress continues as technology improves and more developers work on the project. The wide-spread acceptance of Grid computing was evident at the Globus World 2003 Conference attended by more than 500 Grid engineers representing 25 countries; a two-fold increase from the same conference held in 2002.

The MecaGRID project started in the fall 2002 is sponsered by the French Ministry of Research through the ACI-Grid program[2]. The purpose of the MecaGRID project is to provide the Province-Alpes-Côte d'Azur (PACA) region with a means to make large parallel computations in fluid mechanics using hundreds of processors. This possibility can be realized by creating a large computational Grid comprised of clusters at the different sites in the PACA region. The initial phase of the MecaGRID brings together the clusters at INRIA Sophia Antipolis, CEMEF (Centre de mise en forme des matériaux de l'Ecole des Mines de Paris-Sophia Antipolis), and the IUSTI (Institut Universitaire des Systèmes Thermiques et Industriels) in Marseille - see Guillard [7]. In the preliminary stage of development of the MecaGRID project, the i-cluster of INRIA Rhone Alpes was included in the pool of machines available to the MecaGRID community. Each partner has their own different scenarios as to how they would use the increased capability provided by the MecaGRID.

Even though Grid Computing has been under development for almost a decade, the challenges involved in creating computational Grids are many and today near plug-and-play software exists only for special types of Grid Com-

[1] http://www.eu-datagrid.org
[2] http://www.recherche.gouv.fr/recherche/aci/grid.htm

puting like Desktop Computing and Enterprise Grids which are simplified by being located at a single site with local networks and local administrators. Plug-and-play software does not yet exist for what are known as Virtual Organizations (VO) Grids like the MecaGRID. VOs are institutions or groups that are totally independent entities. However, Grid software like the Globus Alliance software (http://www.globus.org) is sufficiently developed to permit the creation of VO Grids over a 1-2 month period under proper conditions. The proper conditions are on the human side rather than the software side and can be the most difficult to resolve even among partners.

VO Grids are difficult for many reasons. First, the computing resources are located at different institutions usually in different geographical locations. The institutions are independent of each other, having their own priorities, system administrators, batch systems, queue priorities, security procedures, hardware and user constituency. Changes at any particular VO site are made in a timely manner as the changes are under control of a local system administrator. The major reason why creating VO Grids is difficult is the absence of a Grid administrator to plan, establish benchmarks, and coordinate the local systems administrators (LSAs). See Wornom [14] for further discussion of the challenges encountered in creating the MecaGRID.

2 MecaGRID resources

The computer resources for the MecaGRID are the clusters at the different members of the MecaGRID to be combined to function as one large parallel computer giving MecaGRID users the capability to perform computations that are not possible at any member site (the maximum processors available at any member site in the present MecaGRID configuration is 70 at INRIA Sophia Antipolis).

Clusters have either public or private IP addresses. Table 1 shows that 679 processors would be available if both private and public IP addresses were included. The local area network (LAN) speed is in Giga bits per second (Gbps). p/N is the number of processors per Node.

At the present time, users at the individual member sites may experience long wait times when more than 8-16 processors are requested. Depending on the work load, the wait times may extend from several hours to a week. One could imagine that with 679 processors available, computations requiring

Site	cluster	IPA	CPUs	GHz	RAM/Node	LAN	p/N
INRIA Alpes	icluster1	public	216	0.7	3.00 GB	0.100	1
INRIA Alpes	icluster2	public	200	1.0	3.00 GB	1.000	2
INRIA sophia	INRIA-pf	public	38	1.0	0.50 GB	0.100	2
INRIA sophia	INRIA-nina	public	32	2.0	1.00 GB	1.000	2
IUSTI	m3h-cluster	private	30	2.0	1.00 GB	0.100	1
CEMEF	sarek-cluster	private	63	1.0	0.50 GB	0.100	2
Total CPUs		pub+pri	679				

Table 1: Total MecaGRID processors (public + private)

32-64 processors (less than 10 percent of the available processors) would not require long wait periods using the MecaGRID. A recent example experienced by the author was a request for 48 processors on the INRIA cluster to evaluate performance that remained in the queue three days before obtaining the 48 processors. The same run was submitted to the MecaGRID where 48 processors were immediately available permitting performance evaluation of the inter-clusters used in the run.

In the summer 2003 INRIA the Rhone-Alpes cluster underwent an upgrade lasting several months and became an inactive member of the MecaGRID. This upgrade is now complete and it is in the interest of the MecaGRID that INRIA Rhone-Alpes be integrated as an active member as its two clusters have three times as many available processors than the other three members combined!

2.1 Grid computations using private IP addresses

Table 2 shows what the MecaGRID configuration would resemble using only the clusters with public IP addresses (486 processors). Shown in Table 3 is the present MecaGRID configuration that uses private IP addresses. The clusters at CEMEF and the IUSTI have private IP addresses which necessitated the creation of a Virtual Private Network (VPN) or tunnel to pass messages between the different clusters. The INRIA clusters have public IP addresses but the VPN treats them as if they were private.

The maximum number of processors available at the CEMEF and the IUSTI is 92 (each site uses two processors for system management).

The Random Access Memory (RAM) is the total RAM available to the Node thus, approximately 1/2 is available to each CPU of the Node for Nodes

Site	cluster	IPA type	CPUs	GHz	RAM/Node	LAN speed
INRIA Alpes	icluster1	public	216	1	3.00 GB	1.000 Gbps
INRIA Alpes	icluster2	public	200	1	3.00 GB	1.000 Gbps
INRIA sophia	INRIA-nina	public	32	2	1.00 GB	1.000 Gbps
INRIA sophia	INRIA-pf	public	38	1	0.50 GB	0.100 Gbps
Total CPUs			486			

Table 2: MecaGRID clusters composed of public IP addresses

Site	cluster	IPA type	CPUs	GHz	RAM/Node	LAN speed
CEMEF	sarek-cluster	private	62	1	0.50 GB	0.100 Gbps
INRIA sophia	INRIA-pf	public	38	1	0.50 GB	0.100 Gbps
INRIA sophia	INRIA-nina	public	32	2	1.00 GB	1.000 Gbps
IUSTI	m3h-cluster	private	30	2	1.00 GB	0.100 Gbps
Total			162			

Table 3: Current MecaGRID configuration composed of private IP addresses

with bi-processors. Note that with the exception of the IUSTI, all the clusters have two CPUs/Node. Thus the INRIA-nina (or simply nina) and the IUSTI clusters both have 2 GHz processors, nina with 1/2 GB RAM/CPU and the IUSTI with 1 GB/CPU.

The VPN establishes routes through the Grid and tunnels between frontends in order to provide connectivity between the frontend nodes; all the frontends have public IP address. Each pair of frontends is connected via a tunnel in which crypted and encapsulated packets are transmitted. The packets are compressed to improve the flow rate through the tunnel. The VPN is completed by the addition of routes so that each processor can send a packet to any other processor on the Grid and see each frontend as a default gateway for external addresses (i.e. not on the same LAN). A packet sent from an INRIA processor to a CEMEF processor is first sent to the INRIA's frontend where special routes have been set up to send it in the appropriate tunnel. The VPN functions as if all processors were on a WAN (wide area network) - see Nivet [11]. Figure 3 (from Nivet [11]) shows a schematic of the VPN for the present MecaGRID configuration. For additional technical details on the MecaGRID the reader is referred to Nivet's report.

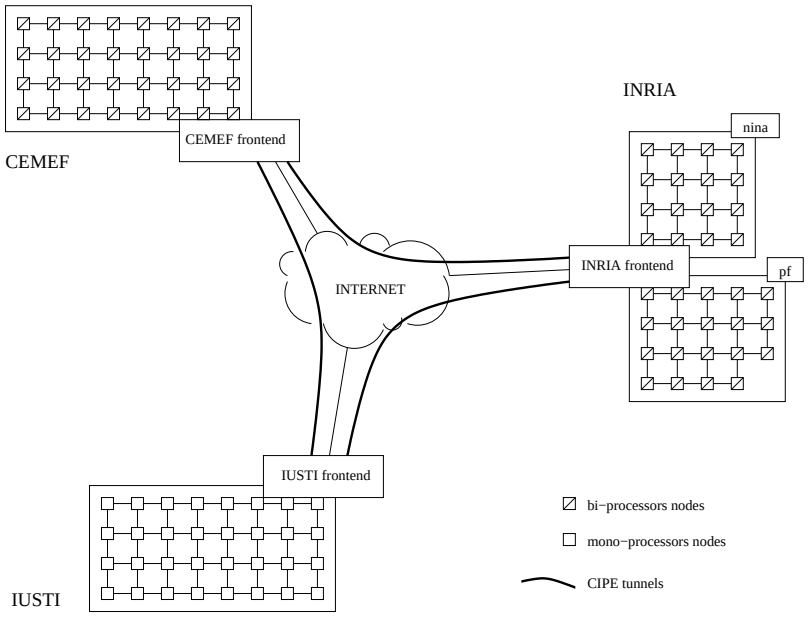


Figure 1: Schematic of MecaGRID VPN

3 One-phase flow experiments

3.1 Numerical Algorithm

The first kernel considered in this study is the software AERO developed at the University of Colorado with the collaboration of INRIA - see Dervieux [3], Konga and Guillard [1], Farhat [5], and Martin and Guillard [9] for details. AERO relies on an unsteady three-field model consisting of a structural model (AERO-S), a fluid model (AERO-F), and a pseudo-elasticity model (AERO-E) for the dynamical fluid mesh. It is useful for the sequel to give a few equations describing the coupled model:

$$\begin{aligned} \frac{\partial}{\partial t}(V(x,t)w(t)) + F^c(w(t),x,\dot{x}) &= R(w(t),x) \\ M\frac{\partial^2 q}{\partial t^2} + f^{int}(q) &= f^{ext}(w(t),x) \\ \tilde{M}\frac{\partial^2 x}{\partial t^2} + \tilde{D}\frac{\partial x}{\partial t} + \tilde{K}x &= K_c q \end{aligned} \tag{1}$$

where t designates time, x the position of a moving fluid grid point, w is the fluid state vector, V results from the finite-element/volume discretization of the fluid equations, F^c is the vector of convective ALE fluxes. R is the vector of diffusive fluxes, q is the structural displacement vector, f^{int} denotes the vector of internal forces in the structure, and f^{ext} the vector of external forces. M is the finite-element mass matrix of the structure, $\tilde{M}$, $\tilde{D}$ and $\tilde{K}$ are fictitious mass, damping and stiffness matrices associated with the moving fluid grid and K_c is a transfer matrix that describes the action of the motion of the structural side of the fluid/structure interface on the fluid dynamic mesh.

An implicit finite-element time scheme is used for the structural model and an implicit time-staggered scheme for the structure. A vertex-centered upwind finite-volume scheme is employed when AERO-F is used in the fluid-only mode. Numerical options address second-order accuracy both in space and time - see Dervieux [3], Konga and Guillard [1], Farhat [5], and Martin and Guillard [9].

The goals of this study were: 1) Creation of the MecaGrid, 2) Studing the efficiency of the one-phase AERO-F code using the MecaGRID, and 3) Developing and examining the efficiency of a three-dimensional two-phase version of the AERO-F code for Grid applications. The results of these experiments are reported in the sections that follow.

4 Test case: AGARD swept wing

The AGARD test case computes three-dimensional flow around a swept wing at low Mach number - see Yates [15]. The mesh used here contains 22K vertices. The MecaGRID is designed for large problems containing thousands or millions of vertices thus this mesh is relatively small for Grid calculations. However, it will give an idea of how small meshes perform on the MecaGRID and permit a comparison between the performance of the implicit and explicit

solver options in the AERO-F code. The implicit runs required 128 time steps to obtain a residual convergence of 10^{-6}. Using the explicit scheme would require several thousands of time steps and a very large CPU time to achieve the same level of convergence. The explicit solver uses a 4-stage Runge-Kutta time scheme. This stringent convergence is not necessary to establish Grid performance. For the Grid performance study, roughly equivalent times for both the explicit and implicit methods were used (on the order of 150 seconds, 256 explicit solver time steps).

Shown in Figure 2 is the non-Globus speedup versus the number of processors on the INRIA-nina and INRIA-pf clusters at INRIA Sophia Antipolis. Even though the mesh is small, a reasonable speedup is obtained with 8-processors (a factor of approximately 7). Mach contours at zero incidence for the AGARD mesh are shown in Figure 3. The view gives the false impression that the wing is at an angle of attack.

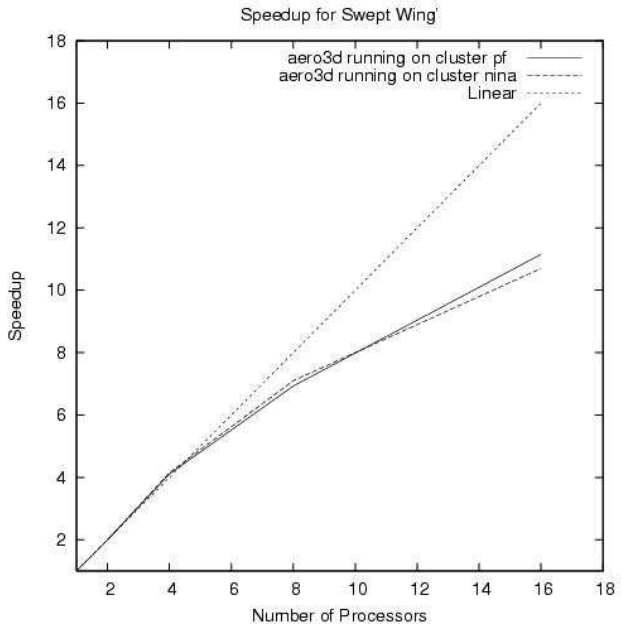


Figure 2: MPI speedup on the INRIA clusters using the implicit solver

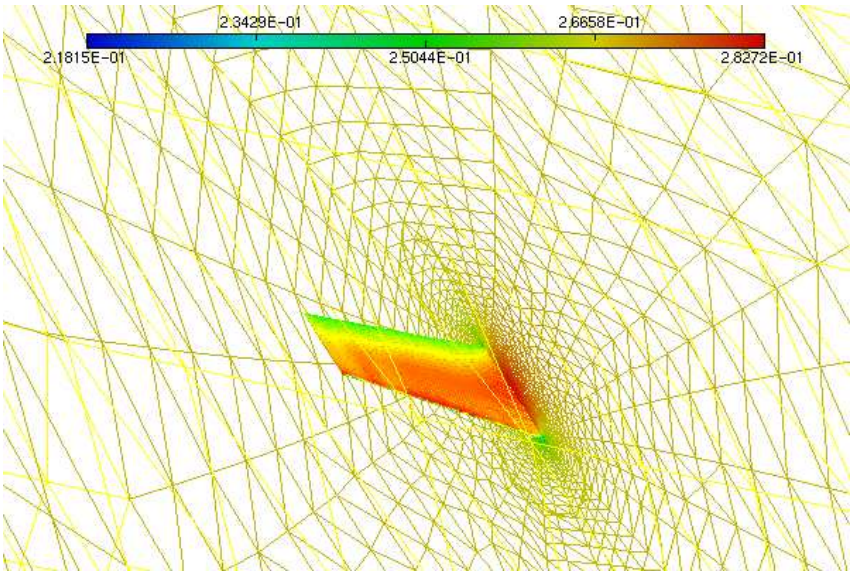


Figure 3: AGARD wing mesh: Mach contours

In the next sections, the MecaGRID performance is examined for 8 processors and combinations of 8 processors involving the different MecaGRID clusters.

4.1 Globus versus non-Globus performance

Shown in Table 4 is a comparison between non-Globus and Globus performance for the AGARD swept wing test case using the INRIA-nina cluster. Table 4 shows the Globus performance to be slightly better than the non-Globus performance. The non-Globus MPI uses the MPICH p4_ch device whereas the Globus MPI uses the globus2 device. The small differences in performance are due to slightly different configure options. Global inter-communication occurs, for example, when the maximum, minimum, or sum of the values of a variable are computed over all the processors. Local inter-communication occurs when messages are passed between two processors. The total computation time includes the inter/intra-communication times but not setup times (reading data, meshes, initialization, ...etc). The times shown in all tables are in seconds and the maximum values for all the processors. Times to save

intermediate solutions are not taken into account. The Communication/Work ratio is the sum of the local and global communication time divided by the total computational time - communication time (Work)[3] . The minimum and average Communication/Work ratios are much smaller.

The non-Globus and Globus computational times are approximately the same on the nina cluster. Without testing each individual cluster, we hypothesize that the non-Globus and Globus times will be approximately the same on the other clusters as well.

Run type	non-Globus	Globus
Name of cluster	INRIA nina	INRIA nina
Processor speed	2 GHz	2 GHz
LAN speed	1 Gbps	1 Gbps
cache	512K	512K
RAM/CPU	1/2 GB	1/2 GB
Executable size	236 MB	236 MB
Number of processors	8	8
Total computational time	103.6	96.7
Local inter-comm. time	15.5	13.2
Global inter-comm. time	16.0	8.7
Communication/Work	0.4	0.3

Table 4: Globus versus non-Globus performance implicit performance

4.2 Individual cluster performance

Shown in Table 5 are the Globus performances on the different individual clusters of the MecaGRID and the communication times relative to the nina values. The computational ratios shown in the tables that follow are based on the total computational times relative to the INRIA-nina cluster; Examining Table 5 one can imagine the difficulty in optimizing Grid computations due to the different processors speeds, RAM, cache, and LAN speeds on the different clusters. For the AGARD swept wing case the size of the executable is 236 MB

[3]The maximum local and global communication times may be on different processors. Therefore the Communication/Work ratios shown here computed with the mentioned ratios are only an upper estimate and are larger than the actual values. The most recent version of the AERO-F code compute these ratios correctly-see APPENDIX F.

Run type	Globus	Globus	Globus	Globus
Name of cluster	INRIA-nina	IUSTI	CEMEF	INRIA-pf
Processor speed	2 GHz	2 GHz	1 GHz	1 GHz
LAN speed	1.00 Gbps	100 Mbps	100 Mbps	100 Mbps
cache	512K	512K	256K	256K
RAM/CPU	1/2 GB	1 GB	1/4 GB	1/4 GB
Executable size	236 MB	236 MB	236 MB	236 MB
Number of processors	8	8	8	8
Total computational time	87.6	148.0	200.2	264.1
Local inter-comm. time	11.3	48.8	49.4	56.6
Global inter-comm. time	1.5	3.3	7.9	8.0
Computation ratio	1.0	1.5	2.3	3.0
Communication/Work	0.4	0.5	0.4	0.3

Table 5: Globus performances on the individual clusters

and is less than the available RAM/processors on the different clusters. Thus one would expect that the INRIA-nina and the IUSTI clusters to show about the same performance as both have 2 GHz processors. Table 5 shows the IUSTI cluster to be slightly slower than the nina cluster. A possible explanation for this difference may the difference in LAN speeds, 1 Gbps for nina and 0.1 Gbps at the IUSTI. Likewise one might expect the CEMEF cluster to be approximately twice as slow as the nina cluster rather than a factor of 2.3 that again may be related to the different LAN speeds (1 versus 0.1 Gbps). One would also expect that the INRIA-pf cluster would be a factor of two slower than nina instead of a factor of 3.0. In general, it is difficult to evaluate the relative importance of the different cache sizes, LAN speeds, and the RAM available to a processor particularily for a small mesh.

4.3 Grid performance summary

Shown in Table 6 are the performances on the different clusters of the MecaGRID for the AGARD test case for different combinations of the clusters for both the implicit and the explicit solvers relative to the nina times. For both the implicit and explicit solvers the performance degrades significantly for inter-cluster computations. Inter/intra-cluster computations are indicated by 4-4 thus, for example, nina-cemef means 4 processors on nina and 4 processors on the CEMEF cluster.

Name of cluster(s)	CPUs	implicit	explicit
nina	8	1.0	1.0
iusti	8	1.4	1.7
cemef	8	3.9	2.3
pf	8	4.2	3.0
nina-pf	4-4	4.1	2.4
inter cluster			
nina-iusti	4-4	21.6	16.4
nina-cemef	4-4	24.6	17.9
pf-iusti	4-4	23.4	16.3
pf-cemef	4-4	27.2	17.0
iusti-cemef	4-4	-	22.7

Table 6: AGARD Globus performance summary

Run type Name of cluster	Globus nina-pf	Globus nina-cemef	Globus nina-iusti	Globus iusti-cemef
Processor speed	2/1 GHz	2/1 GHz	2/2 GHz	2/1 Ghz
cache (K)	512/256	512/256	512/512	512/256
Executable size	236 MB	236 MB	236 MB	236 MB
Number of processors	4-4	4-4	4-4	4-4
Total computational time	208.6	1570.6	1438.4	1990.3
Local inter-comm. time	50.4	1137.9	1178.0	1188.0
Global inter-comm. time	6.3	158.9	159.3	146.0
Computational ratio	2.4	17.9	16.4	22.7
communication/work	0.37	4.73	13.23	2.03

Table 7: AGARD: Inter/intra-cluster Explicit solver performance

Table 7 shows some of the inter-cluster and the intra-cluster (nina-pf) performances using the explicit solver where it is seen that the inter-cluster performances are degraded due to large local inter-communication and global communication times. This may in part due to the large physical distance between Sophia Antipolis and Marseille where different routers and networks may be involved. The problem is certainly agravitated by the small mesh where most of the time is used in message passing rather than processor work. Lastly, another factor impossible to evaluate in the present MecaGRID configuration is the efficiency of the VPN.

Regarding the AGARD test case with the 22K vertices mesh, the following observations are made:

1. In general, inter-cluster Grid performance was poor for both the implicit and the explicit solver indicating that small meshes are not suitable for inter-cluster GRID applications. For small meshes the communication time between processors is larger than total processor work time. Therefore one must be careful in extrapolating these results to larger meshes.
2. The explicit time scheme shows better inter-cluster Grid performance than the implicit time scheme. This is perhaps a result of the vector matrix product iteration in the implicit solver that may add more communication time than processor work on small meshes; application on larger meshes may result in the contrary. However, for computing steady-state flows like the AGARD swept wing test case, the total CPU time for the implicit scheme is much less than the explicit solver[4] .
3. Inter-cluster Grid computations involving combination pf-CEMEF clusters and the combination nina-CEMEF give approximately the same performance.
4. Using the explicit solver, the CEMEF cluster showed a slightly better performance than the INRIA-pf cluster. Equivalent performance was expected.
5. In general, inter-cluster Grid computations involving the CEMEF cluster were the least efficient of the inter-cluster computations. The reasons for the poor performance will be discussed in section 9.1.

5 Two-phase flow experiments

5.1 Numerical Algorithm

The MecaGRID project was created specifically for MPI codes applied to fluid dynamics problems. Therefore a natural candidate for the MecaGRID was the AERO-F software developed at the University of Colorado with the

[4]The explict solver was approximately a factor of 38 slower for this problem. However, the explict solver can be optimized for steady-state problems, this was not done in this study as it was beyond the scope of this investigation.

collaboration of INRIA that executes on parallel computers using MPI. A second important reason for selecting the AERO code is the ease of integrating the recent two-phase advancements of the SMASH team (see Murrone and Guillard [6], Murrone [10]) into a three-dimensional code. The integration is facilitated by the fact that the origins of the two-dimensional code of Murrone and Guillard are an earlier version of the AERO code therefore the structure of the two codes is very similar.

Therefore in the context of the MecaGRID, the fluid part of the software AERO-F has been extended to the calculation of two-phase flows. The Euler equations for a unique perfect gas has been replaced by the following model:

Seven-equation quasi conservative reduced model

$$\begin{aligned}
\frac{\partial}{\partial t}\alpha_k\rho_k + \mathrm{div}(\alpha_k\rho_k u) &= 0\ ; \quad k = 1,2 \\
\frac{\partial}{\partial t}\rho u + \mathrm{div}(\rho u \otimes u) + \nabla p &= 0 \\
\frac{\partial}{\partial t}\rho e + \mathrm{div}(\rho e + p)u &= 0 \\
\frac{\partial}{\partial t}\alpha_2 + u.\nabla\alpha_2 &= \alpha_1\alpha_2\frac{\rho_1 a_1^2 - \rho_2 a_2^2}{\sum_{k=1}^{2}\alpha_{k'}\rho_k a_k^2}\mathrm{div}u
\end{aligned}$$

with $e = \varepsilon + u^2/2$ and $\rho\varepsilon = \sum_{k=1}^{2}\alpha_k\rho_k\varepsilon_k(p,\rho_k)$.

where α_k are the mass fractions of the two fluids. To distinguish the different versions of the AERO codes, the two-phase version is named AEDIF, only the explicit solver is used. Details on the development of AEDIF are given in [13]

6 Shockwave-bubble test cases

Shown in Table 8 are the mesh sizes for the three-dimensional shock-bubble interaction test cases. The number of mesh vertices were 262K, 568K, and 1.03M. This test case computes the interaction of a shockwave moving through a low density fluid and interacting with a bubble of high density fluid. The

Test cases	No. vertices	Code
3d Shock-bubble interaction	262K	AEDIF
3d Shock-bubble interaction	568K	AEDIF
3d Shock-bubble interaction	1.03M	AEDIF

Table 8: Shockwave-bubble test Cases

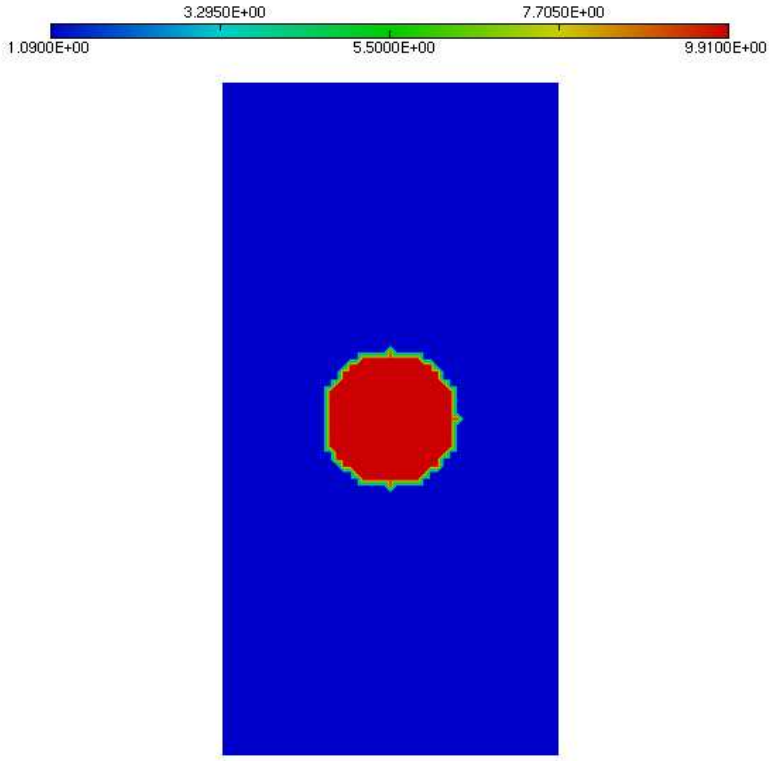


Figure 4: Test case: Shockwave-bubble symmetry plane density profile

explicit three-stage time scheme of Shu and Osher [12] was used to advance the solution in time.

Figure 4 and Figure 5 show the initial symmetry plane density contours (10:1 density ratio) and the three-dimensional density contours after 720 time

steps at which time the shockwave has passed through the bubble reflected off the top boundary and passed through the bubble a second time.

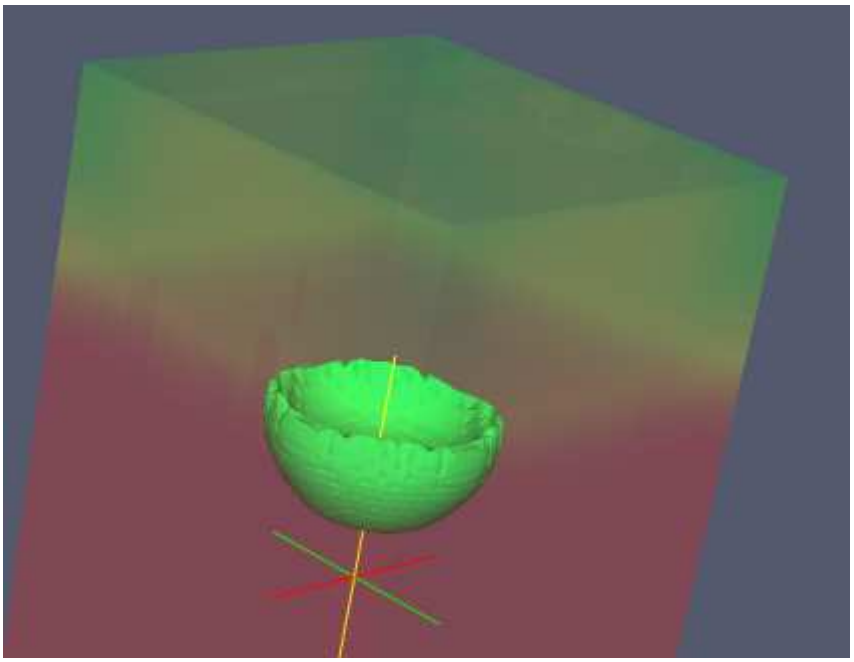


Figure 5: Shockwave-bubble density contours after 720 time steps

7 262K mesh

7.1 262K mesh: Globus performance on individual clusters

Shown in Table 9 are the performances on the different individual clusters of the MecaGRID. For this relatively large mesh, the executable size is 871 MB well within the 1 GB of RAM at the IUSTI but greater than the available RAM on the other clusters. This fact may account for the increase in performance of the IUSTI cluster relative to the other clusters. The performances on the individual clusters are approximately as one might expect based on the processor speed. Note that the 2 GHz clusters (nina and the IUSTI) show the same performance. The INRIA-pf cluster performs better than expected with a ratio of 1.2 as opposed to 2. One would expect that the INRIA-pf

Run type Name of cluster	Globus INRIA-nina	Globus IUSTI	Globus INRIA-pf	Globus CEMEF
Processor speed	2 GHz	2 GHz	1 GHz	1 GHz
LAN speed	1.00 Gbps	100 Mbps	100 Mbps	100 Mbps
cache	512K	512K	256K	256K
RAM/Node	1 GB	1 GB	1/2 GB	1/2 GB
Executable size	871 MB	871 MB	871 MB	871 MB
Number of processors	8	8	8	8
Total computational time	331.4	341.2	564.7	759.6
Local inter-comm. time	2.0	7.2	9.1	12.6
Global inter-comm. time	8.6	9.5	16.6	35.3
Computation ratio	1.00	1.03	1.7	2.3
Communication/Work	0.03	0.05	0.05	0.07

Table 9: 262K mesh: Comparison of Globus performances on individual clusters

and CEMEF clusters to perform approximately the same rather than 1.7 and 2.3. In spite of the fact that the executable size of 871 MB would seem to be larger than the INRIA-pf and the CEMEF available RAM, the performances of the INRIA-pf and the CEMEF are notable. Note also that all the Communication/Work[5] ratios are in the range 3-7 percent thus processor time is dominated by work, a desirable characteristic in parallel codes.

Run type Name of cluster	Globus nina-pf	Globus nina-cemef	Globus nina-iusti	Globus iusti-cemef
Processor speed	2/1 GHz	2/1 GHz	2/2 GHz	2/1
cache (K)	512/256	512/256	512/512	512/256
Executable size	871 MB	871 MB	871 MB	871 MB
Number of processors	4-4	4-4	4-4	4-4
Total computational time	401.8	550.8	640.7	970.6
Local inter-comm. time	7.8	120.5	187.8	207.0
Global inter-comm. time	17.8	52.6	65.7	108.0
Computational ratio	1.2	1.7	1.9	2.9
Communication/Work	0.07	0.46	0.65	0.48

Table 10: 262K mesh: Inter/intra-cluster Explicit solver performance

[5]The Communication/Work ratio is the total communication time divided by the Work. (Work = total computational time - total communication time.

Table 10 shows some of the inter-cluster performances using the explicit solver where it is seen that the inter-cluster performances are quite good, the Communication/Work ratios are < 0.65.

7.2 262K mesh: Grid performance summary

Name of cluster(s)	CPUs	262K	22K AGARD
nina	8	1.0	1.0
iusti	8	1.0	1.6
pf	8	1.7	2.9
cemef	8	2.3	2.2
nina-pf	4-4	1.2	2.3
inter cluster			
nina-iusti	4-4	1.9	15.7
nina-cemef	4-4	1.7	17.1
pf-iusti	4-4	2.5	15.6
pf-cemef	4-4	2.2	16.7
iusti-cemef	4-4	2.9	21.7

Table 11: 262K mesh: Globus performance summary

Shown in Table 11 are the performances for different combinations of the MecaGRID cluster for the 262K mesh. Also shown for comparison are the AGARD results obtained with the explicit solver. Table 11 shows better inter-cluster and intra-cluster performances for the 262K mesh than the 22K AGARD swept wing test case.

Regarding the shockwave-bubble test case with the 262K vertices mesh, the following observations are made:

1. Based on the inter-cluster Grid performance, 262K mesh seems suitable for Grid applications. The individual clusters Communication/Work ratios are in the range 3-7 percent. The inter-cluster Communication/Work ratios are less than 0.7.

2. The INRIA-pf cluster gives a better performance than the CEMEF cluster. Equivalent performance was expected.

3. In general, inter-cluster Grid computations involving the CEMEF cluster were the least efficient with the exception of the nina-cemef (1.7).

8 568K mesh

8.1 568K mesh: Globus performance on individual clusters

After the 262K study was completed, the AEDIF code was restructured to remove all unnecessary tables and subroutines to permit larger meshes with smaller executables than otherwise possible.

Run type	Globus	Globus	Globus	Globus
Name of cluster	INRIA-nina	IUSTI	INRIA-pf	CEMEF
Processor speed	2 GHz	2 GHz	1 GHz	1 GHz
LAN speed	1 Gbps	100 Mbps	100 Mbps	100 Mbps
cache	512K	512K	256K	256K
RAM/CPUe	1/2 GB	1 GB	1/4 GB	1/4 GB
Executable size	237 MB	237 MB	237 MB	237 MB
Number of processors	16	16	16	16
Total computational time	104.0	94.4	195.6	189.7
Local inter-comm. time	1.8	13.3	12.2	13.4
Global inter-comm. time	51.5	39.1	55.9	81.5
Computational ratio	1.0	0.91	1.9	1.9
Communication/Work	1.0	1.2	0.5	1.0

Table 12: 568K mesh: Globus performances on individual clusters - 16 CPUs -O3 option

Shown in Table 12 are the performances on the different individual clusters using 16 processors. The performances on the INRIA-pf and the CEMEF clusters are quite good (compute ratio < 2). Note that the IUSTI cluster performance is 20 percent faster than the INRIA-nina cluster, an unexpected result. However, the Communication/Work ratios for the 568K mesh with 16 processors are much larger than for the 262K mesh using 8 processors[6] . In examining the computational times for the the 262K and 568K runs, it was found that different compile options were used[7] and explains the differences in the Communication/Work ratios. Therefore for the same mesh, the more efficient the code (less Work per processor) the larger the Communication/Work

[6]This abnormality was noted during the writing of the report. Since the 568K mesh is two times larger than the 262K mesh, and 16 processors were used rather than 8, one would expect approximately the same Communication/Work ratios. For both the 262K and 568K meshes, 10 time steps were used for the comparisons.

[7]The Makefile shows that the -O1 option was used to compile AERO-F for the 262K mesh and the -O3 option for the 568K mesh. The -O1 option increases the processor work by a factor of approximately 4 relative to the -O3 option.

ratios as the communication times depend on the LAN speeds that remain unchanged[8] !

Run type	Globus	Globus	Globus	Globus
Name of cluster	INRIA-nina	iusti	INRIA-pf	cemef
Processor speed	2 GHz	2 GHz	1 Ghz	1 GHz
LAN speed	1 Gbps	100 Mbps	100 Mbps	100 Mbps
cache	512K	512K	256K	256K
RAM/CPUe	1/2 GB	1 GB	1/4 Gb	1/4 GB
Executable size	871 MB	871 MB	871	871 MB
Number of processors	8-8	8-8	8-8	8-8
Total computational time	547.3	449.9	740.9	1039.8
Local inter-comm. time	2.3	12.8	9.4	12.5
Global inter-comm. time	280.8	178.9	288.7	277.3
Computational ratio	1.00	0.82	1.35	1.90
Communication/Work	1.07	0.74	0.67	0.39

Table 13: 568K mesh: Globus performances on individual clusters - 16 CPUs -O1 option

Table 13 shows the performances on the individual clusters using the -O1 option. Comparison of Table 12 (-O3 option) with Table 13 (-O1 option) shows that compiling with the -O1 option reduces the Communication/Work time ratios[9].

Shown in Table 14 are some of the inter-cluster performances with 16 processors. It is noted that the local communication times for nina-cemef and pf-cemef are approximately two times smaller than the other inter-cluster combinations. This astonishing observation cannot be explained.

8.2 Influence of the "0" processor location

Table 15 shows the influence of the location of the "0" processor on the performance. The inter-cluster performance is best when the "0" processor is located on the fastest processor. This is expected as the "0" processor initiates the global communication and writes data.

[8] In hindsight, for comparison purposes, the -O1 compile option should have been retained for the 568K and the 1.03M meshes. The -O1 compile option was used in the AGARD test case.

[9] Examining Tables 12-13 shows that the local communication times are approximately the same for both the -O1 and -O3 compile options whereas the global communication times for the -O1 option are on the order of 3-6 times larger than the -O3 times.

Run type	Globus	Globus	Globus	Globus	Globus	Globus
Name of cluster	nina	nina-pf	nina-iusti	nina-cemef	pf-cemef	iusti-cemef
Processor speed	2 GHz	2/1 Ghz	2/2 GHz	2/1 Ghz	1/1 Ghz	2/1 Ghz
Executable size	343 MB	343 MB	343 MB	343 MB	343 MB	343 MB
Number of processors	16	8-8	8-8	8-8	8-8	8-8
Total computational time	547.3	702.3	1207.9	1322.4	1323.4	2041.6
Local inter-comm. time	2.3	10.5	496.3	190.8	181.9	554.0
Global inter-comm. time	280.8	279.5	449.2	411.4	411.4	449.4
Computational ratio	1.00	1.28	2.21	2.42	2.41	3.73
Communication/Work	1.07	0.70	3.61	0.83	0.81	0.97

Table 14: 568K mesh: inter-cluster performance - 16 CPUs -O1 option

Name of cluster(s)	CPUs	262K	568K
nina-pf	4-4	1.2	-
pf-nina	4-4	1.7	-
nina-cemef	4-4	1.7	-
cemef-nina	4-4	4.6	-
iusti-cemef	4-4	2.5	1.8
cemef-iusti	4-4	2.9	2.4

Table 15: Influence of the "0" processor location

8.3 568K mesh: Grid performance summary

Shown in Table 16 are the performances for different combinations of Meca-GRID clusters using the 568K mesh relative to the nina times with the -O1 and -O3 options. Also shown for comparison are the performances for the 262K mesh.

Regarding the shock-bubble test case with the 568K vertices mesh, the following observations are made:

1. The IUSTI cluster's performed 20 percent faster than the INRIA-nina cluster.

2. The INRIA-pf cluster gives a slightly better performance than the CE-MEF cluster (1.4 and 1.9), showing better a than expected performance relative to the INRIA-nina cluster.

	262K	568K	
Name of cluster(s)	-O1	-O1	-O3
nina	1.0	1.0	1.0
iusti	1.0	0.8	0.9
pf	1.7	1.4	1.9
cemef	2.3	1.9	1.9
nina-pf	1.2	1.3	1.6
inter cluster			
nina-iusti	1.9	2.2	5.8
nina-cemef	1.7	2.4	3.7
pf-iusti	2.5	2.4	4.0
pf-cemef	2.2	2.6	7.3
iusti-cemef	2.9	3.8	6.8

Table 16: Globus performance summary: 262K versus 568K

3. Inter-cluster Grid computations involving the CEMEF cluster tend to be the least efficient.

4. The inter-cluster performances relative to nina using the -O1 compile option are better than with the -O3 compile option. This occurs due to a larger Work/processor when compiled with the -O1 option and therefore a smaller Communication/Work time ratio. A way of decreasing this ratio using the -O3 compile option is to increase the mesh size holding the number of CPUs fixed.

9 Globus performance on a 1.03M vertices mesh

This section examines the MecaGRID performance for a large number of processors with a mesh containing 1.03M vertices.

Shown in Tables 17-18 are MecaGRID performances using 60-64 CPUs for the 1.03M mesh where Ts/Hr = time steps per hour, CRate[10] = Number of vertices * nstages/(computational time)/(number of time steps). The performances are for 10 time steps. The first column indicates the type of run (g=Globus, ng=non-Globus). The second column gives the number of partitions (P) and the number of processors used. T1 is the computational time

[10]CRate should not be confused with megaflops.

and T2 the communication time. W is the work = T1 - T2 and includes the time to write the solution files. Sav = the number of times the solution files were written. The times shown in Tables 17-18 are the average CPU times. For 64/48, a total of 48 processors are used for the 64 partition mesh(see Load balancing by processor speed (LB-2) in section 12. Table 17 shows that for non-Globus computations on the INRIA clusters one can compute at a rate of approximately 200 time steps/hour with the 1.03M vertices mesh writing solution files every 10th time step. When the solution files are written every two time steps using nina-pf processors, Table 18 shows a Globus computational rate on the order of 50. When inter-clusters are used the rate is on the order of 10 time steps per hour.

		Processor distribution								
Typ	P/CPU	nina	pf	cemef	iusti	Sav	T1/T2	CRate	Ts/Hr	T/W
ng	64/64	32	32	-	-	1	165/ 108	1870	218	1.9
ng	64/48	16	32	-	-	1	185/ 113	1674	195	1.6
inter cluster										
g	60/60	32	-	-	28	1	3523/ 2452	88	10	2.3

Table 17: 1.03M results Sav = 1

		Processor distribution								
Typ	P/CPU	nina	pf	cemef	iusti	Sav	T1/T2	CRate	Ts/Hr	T/W
ng	64/64	32	32	-	-	5	681/ 470	454	53	2.2
g	64/64	32	32	-	-	5	655/ 442	472	55	2.1
g	62/62	32	-	-	-	5	637/ 417	485	57	1.9
inter cluster										
g	64/64	32	16	-	16	5	2796/ 1950	111	13	2.3
g	64/64	16	4	8	24	5	3520/ 2537	88	10	2.6

Table 18: 1.03M results Sav = 5

Four full Globus production runs (800 time steps) with the 1.03M mesh were attempted using 62 processors (32 nina CPUs + 30 iusti CPUs) and 60 processors (32 nina CPUs + 28 iusti CPUs). Three of the four runs failed when one of the requested CPUs failed to start execution. The problem of failing CPUs has existed for at least six months. It occurs at random, the job remains active blocking the CPUs until the job is killed. Two of the failed runs

blocked the system for nina and IUSTI users for five and eight hours before being killed. This failing due to dying CPUs has been noted by other Globus users on the Globus users E-mail list to which all Globus users can subscribe. The Globus software is an evolutionary software, open source and free. Users download the software, install it, and test it. Bugs are found and usually reported on the Globus users E-mail lists often with fixes that they have found or simply bring the bugs to the attention of the Globus Alliance gurus who seek to fix the problems that occur in an evolutionary software. It is possible that this problem is solved in the newer versions of the Globus software.

9.1 Analysis of Grid performance

The extremely poor performance for the Globus inter cluster runs shown in Tables 17-18 is hardware related. Take for example the mismatch in the hardware characteristics of the different frontend machines shown below[11]:

IUSTI has a Pentium IV processor at 2Ghz with 1 GB of RAM
CEMEF has a Pentium IV processor at 400Mhz with 256 MB of RAM
INRIA has a dual-Pentium III processor at 1.2 GHz with 1 GB of RAM

The frontends use three different generations of the Pentium processors. One can immediately see a probable reason why the inter-cluster performances involving the CEMEF are poor. Recall that with the VPN approach all message passing is through the frontend machines. The CEMEF frontend can only receive and send messages at 400 Mhz compared to 2 Ghz at the IUSTI and dual 1.2 Ghz processors at INRIA. Additionally the available RAM at the CEMEF is 256 MB compared to 1 GB at both INRIA and the IUSTI. In theory these reasons result in network jams whenever inter-cluster applications involve the CEMEF cluster.

The poor performance using 60-64 nina-iusti processors cannot be totally attributed to the frontend hardware characteristics as the frontends at INRIA and the IUSTI are roughly equivalent. A possible reason for the poor performance may be that the VPN becomes saturated as the number of processors increases.

[11] Noted by Basset [2]

Tests using more than 24 processors were limited since the IUSTI has only 30 processors available and 24 at the CEMEF[12] . Therefore it was not possible to perform numerical experiments varing the number of nina/iusti processors for the 64-partitions mesh.

To evaluate the MecaGRID performance for a fixed number of processors, numerical experiments were performed using the 32 partition mesh. These results of these experiments are shown in Table 19 varying the number of nina-iusti processors from 8 to 32. The total number of nina-iusti processors for each run was 32. Ideally one would like to see the CRate and Ts/Hr constant for the different combinations of nina-iusti processors. However the performance degrades with the number of iusti processors increases due to larger communication times (T2).

CPUs		Performance using 32 CPUs			
nina	iusti	T1/T2	T/W	CRate	Ts/Hr
32	0	129/ 80	1.65	796.89	278
30	2	381/ 226	3.29	349.56	122
28	4	650/ 447	4.02	184.80	65
24	8	1201/ 924	5.44	94.21	33
20	12	1288/1025	7.22	88.32	31
16	16	1590/1281	7.17	70.59	25
12	20	1492/1184	7.21	76.45	27
8	24	952/ 781	54.20	129.54	45
4	28	631/ 441	22.48	223.86	78

Table 19: 32 partitions: Performance vs. nina-iusti processors

[12] The CEMEF cluster has 62 CPUs available but is configured so that the maximum CPUs available to Globus users is 24. Note that users of the INRIA and IUSTI clusters request the number of CPUs whereas users of the CEMEF cluster request the number of Nodes by queue submission q2, q4, q6, q8, q16, q24 and q32. Each Node has 2 CPUs, the default ppn (processors per Node) is 1. q32 requests 32 Nodes but as only 31 Nodes are available q32 jobs never run. Thus the maximum available Nodes is q24 with default ppn=1, therefore 24 maximum processors! Hopefully this abnormally will be corrected in the near future. For Globus users, the globusrun script must be modified by the local system administrator to permit the user to set the ppn parameter in the OpenPBS script written and submitted by globusrun.

The 1.03M vertices mesh can be computer with 24 processors. Attempts using 16 processors resulted in a buffer size too large[13] . The buffer size can be changed in the AERO-F parameter statements but this was not tried.[14]

At this point in the study the capability to compute the MPI transfer rates between the processors was added. Each partition sends/receives data from its neighboring partitions. The transfer rate is computed by multiplying the total number of data sent/received [15] divided by the time between the sends and receives. Table 20 shows the performances using 24 CPUs[16] compiled with the -O3 option. Also shown are the transfer rates computed by Basset [2] and some ping test results-see APPENDIX G. Ping tests from cemef to the INRIA cluster have three decimal place time accuracy and the computed transfer rates are reasonable. The rates shown are for 100 ping tests. Ping tests from INRIA have only 1 decimal place accuracy which is not sufficent to compute transfer rates. The reader is referred to the report of Basset to better understand the effect of hardware on Grid performance. Table 20 shows a significant loss in performance when inter-clusters are used.

Processor distribution				Mbps					
nina	pf	cemef	iusti	AEDIF	Basset [2]	ping	CRate	Ts/Hr	Tcom/Twork
24	-	-	-	206.1	509.7	-	1920.7	223	1.46
12	12	-	-	40.1	89.3	-	1309.2	152	1.00
-	24	-	-	37.4	86.3	-	1094.8	127	1.03
-	-	24	-	36.6	84.1	-	1062.3	123	0.63
inter cluster									
12	-	12	-	1.9	7.2	60.3	464.2	54	2.12
-	12	12	-	1.9	7.2	60.3	492.8	57	2.40
-	-	12	12	0.6	3.7	-	224.4	26	3.71
12	-	-	12	0.7	5.0	-	263.1	30	5.17
8	-	8	8	0.5	-	-	203.1	23	4.47

Table 20: 1.03M mesh: Globus MPI Transfer rates using 24 CPUs

[13]The AERO-F code prints the following error message is the buffer size is too small

"MESSAGE_IS_TOO_LONG_FOR_BUFFER."

[14]Based on item 4 of the last section, it would have been prudent to increase the buffer size so as to compare the Communication/Work time ratios for the 568K and the 1.03M meshes compiled with the -O3 option for a fixed number of CPUs.

[15]by 64 as AEDIF is compiled with the -r8 option therefore 64 bits for real data

[16]Basset used 2 CPUs for his inter cluster tests and 4 CPUs for the intra cluster tests

9.2 MecaGRID efficiency using 64 CPUs

Table 21 shows the efficiency of the MecaGRID using 64 processors. In Table 21, T1 is the computational time, T2 the local communication time, and T3 the global communication time. Eff, R2, and R3 are the computational, local communication, and global communication times relative to the globus run using 32 nina and 32 pf processors. As seen in Table 21, applications using 64 CPUs are in the range 4-5 times slower than the same run on the INRIA cluster. The reader is referred to APPENDIX G for additional discussion.

Processor distribution				MecaGRID efficiency 64 CPUs							
nina	pf	cemef	iusti	T1	T2	T3	CRate	Ts/Hr	Eff	R2	R3
32	32	-	-	655	8	442	471	54	1.0	1.0	1.0
32	16	-	16	2795	1146	1289	111	12	4.3	133.3	2.9
16	4	8	24	3520	1727	1448	88	10	5.4	200.9	3.3

Table 21: 1.03M mesh: Efficiency using 64 CPUs with Sav = 5

10 Mesh partitioners for Grid calculations

The AERO-F and the AEDIF codes both use MPI to parallelize the calculation over partitions (or domains), one processor for each partition. Therefore the mesh partitioner is an integral part of the parallel method. This study uses the mesh partitioner developed at the CEMEF by Digonnet [4] with recent developments by Lanrivain [8] accomplished during a 2003 internship in the SMASH project at INRIA. These developments included an heterogeneous partitioning option and file formats directly readable by the AEDIF and AERO-F codes. The CEMEF Mesh Partitioner is executed on the INRIA cluster with the following command:

./MeshPartitionerSTD.lsf sinus.mtc ncpu

where sinus.mtc is the mesh to be partitioned and ncpu the number of processors (equals the number of partitions). STD denotes the standard script (1 CPU/Partition). The mesh partitioner input file is of the type ".mtc." The sinus.mtc is created using a SINUS2MTC interface (./SINUS2MTC fluid.sinus) where fluid.sinus is the mesh to be partitioned written in the "sinus" format.

Partitioning using the MeshPartitionerSTD.lsf script is satisfactory as long as the number of partitions is small. If the number of partitions is large, long wait times may occur before obtaining say 64 processors. For partitions $>$ 70 this form of the mesh partitioner cannot be used as the maximum number of processors available at INRIA is 70 (1 CPU/partition). Two forms of the standard MeshPartitionerSTD.lsf script are shown below:

```
bsub -n $2 -m "linux-nina linux-pf" mpijob meshmig_homo $1
bsub -n $2 -m "linux-nina linux-pf" mpijob meshmig_grille $1
```

where $1 is the sinus.mtc file and $2 the number of processors. The script using the meshmig_homo executable returns partitions of equal size (homogeneous) whereas the script using the meshmig_grille creates partitions based on the speed of the processors (heterogeneous), thus partitions using nina processors would be two times larger than INRIA-pf partitions reflecting the fact that the processor speeds for nina and INRIA-pf are 2 Ghz and 1 Ghz respectively. The heterogeneous partitions result in load balancing with respect to processor speed and should increase the overall performance.

In order to create a large number of partitions (superior $>$ 70 processors) and/or not encounter large wait times, an alternative script was written.

```
./MeshPartitioner_II.lsf ncpu npartitions
```

where ncpu is the number of processors to be used, and npartitions the number of partitions. The MeshPartitioner_II.lsf submit script looks like:

```
bsub -m "linux-nina linux-pf" -n $1 mpijob MeshInterface.x $2
```

where $1 is the ncpus to be used and $2 the number of partitions. The MeshPartition_II.lsf script submits a job using ncpu processors. MeshInterface.x determines the hostnames of the processors, writes a mymachinefile.LINUX file to be used by the mesh partitioner and executes the actual partitioning script. Each processor appears in the mymachinefile.LINUX file npartitions/ncpu times (thus it is important that npartitions be a multiple of the ncpus). The partitioning script is:

```
mpirun -machinefile mymachinefile.LINUX -np $1 meshmig_homo fluid.mtc
```

Table 22 shows some of the Mesh Partitioner results using both methods to partition meshes. The mesh partitioner performs iterations until equal size partitions (homogeneous) or heterogeneous (optimized) partitions are obtained.

Partitions	script	Machines	CPUs	time	Iterations
16	MeshPartioner_II.lsf	4-nina	4	0:43	13
24	MeshPartionerSTD.lsf	24-pf	24	1:03	18
48	MeshPartioner_II.lsf	16-nina	16	0:43	20
60	MeshPartioner_II.lsf	18-nina 2-pf	20	0:53	22
62	MeshPartioner_II.lsf	20-nina 11-pf	31	0:59	28
64	MeshPartioner_II.lsf	32-nina	32	1:02	30

Table 22: 1.03M mesh: Mesh Partitioner homogeneous results

11 Submitting jobs to the MecaGRID

The long-term goal is to have the MecaGRID clusters to function as a single computer similar to how the INRIA clusters function. INRIA users can submit a job to one or both the nina and the pf clusters. Four possible scripts for heterogeneous partitioning are:

```
bsub -n $2 -m linux-nina              mpijob meshmig_grille $1
bsub -n $2 -m linux-pf                mpijob meshmig_grille $1
bsub -n $2 -m "linux-nina linux-pf"   mpijob meshmig_grille $1
or
bsub -n $2 -m MyMachinefile.LINUX     mpijob meshmig_grille $1
where
MyMachinefile.LINUX is
linux-nina
linux-pf
```

Carrying this idea over to the MecaGRID, the user would submit his/her job similar to:

```
bsub -n $2 -machinefile MyMachinefile.LINUX     mpijob aedif.x $1
where MyMachinefile.LINUX would look something like
```

```
linux-nina
linux-pf
m3h-cluster
sarek-cluster
```

The globus job manager would query the MecaGRID cluster's jobmanagers until the requested number of processors are available and then submit the job.

At the present time the MecaGRID does not function as described above. Presently, one must specify, in advance, the individual clusters to be used and the number of processors to be used on each cluster. This is done with a RSL (Resource Specification Language). This approach has the obvious disadvantage that global availability of processors is not taken into account. Therefore processors may be requested on a cluster that is fully saturated while another cluster with many processors available goes unused. An example RSL script using 64 processors is shown below requesting 40 processors on the INRIA clusters, 8 processors at the CEMEF and 16 processors on the IUSTI cluster:

```
+
( &(resourceManagerContact="cluster.inria.fr")
   (label="subjob 0")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 0)
      (LD_LIBRARY_PATH /usr/local/globus/lib)
      (PGI /usr/local/pgi))
   (directory =/net/home/swornom/Bubble_3d/64-proc)
   (executable=/net/home/swornom/Bubble_3d/64-proc/aerodia_globus_wornom.x)
   (stdout=globus_sophia_40_cemef_8_iusti_16.out)
   (stderr=globus_sophia_40_cemef_8_iusti_16.err)
   (count=40)
   (jobtype=multiple)
   (MaxWallTime=15)
)
( &(resourceManagerContact="m3h-cluster.polytech.univ-mrs.fr")
   (label="subjob 1")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 1)
      (LD_LIBRARY_PATH /usr/lib/:/home/swornom/pgi/:/usr/local/globus/lib/))
```

```
   (directory =/home/swornom/Bubble_3d/64-proc)
   (executable=/home/swornom/Bubble_3d/64-proc/aerodia_globus_wornom.x)
   (stdout=globus_sophia_40_cemef_8_iusti_16.out)
   (stderr=globus_sophia_40_cemef_8_iusti_16.err)
   (count=16)
   (MaxWallTime=15)
)
( &(resourceManagerContact="sarek-cluster.cma.fr")
   (queue=q8)
   (label="subjob 2")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 2)
      (LD_LIBRARY_PATH /mecagrid/tmp/packages_globus/globus_RH7.1/lib:
         /mecagrid/nivet/pgi:/mecagrid/wornom/pgi))
   (directory =/mecagrid/wornom/Bubble_3d/64-proc)
   (executable=/mecagrid/wornom/Bubble_3d/64-proc/aerodia_globus_wornom.x)
   (stdout=globus_sophia_40_cemef_8_iusti_16.out)
   (stderr=globus_sophia_40_cemef_8_iusti_16.err)
   (count=8)
   (MaxTime=15)
)
where count = the number of processors (Nodes at the CEMEF)
```

12 Optimizing MecaGRID calculations

Methods to optimize MecaGRID computations can be described as follows:

1. Global submittal scripts- Creating a submittal script that accounts for global processor availability should be a priority. This was discussed previously. This approach is not currently being pursued at INRIA but is essential before users will accept the MecaGRID. It is a topic of current interest on the Globus users E-mail list and is part of the Globus Tool Kit (Metacomputing Directory Service or MDS).

2. Load balancing by processor speed (LB-1)- The recent improvements by Lanrivain [8] to the mesh partitioner developed by Digonnet [4] at the CEMEF is notable. Lanrivain and Digonnet created a heterogeneous version that partitions a mesh accounting for the different processor speeds

on the different clusters. This approach would work well in the present MecaGRID configuration where the clusters and the number of processors to be used on each cluster must be specified in advance. Basset [2] obtained mixed results using this approach, very good or very poor.

For the global processor availability method, there are several disadvantages to this innovative approach, they are: 1) the clusters that will be used and the number of processors on the cluster are not known in advance. Therefore the partitioner must be executed in the same run as AEDIF run to create the partitions that AEDIF will use on the same processors and, 2) Table 22 shows that as much as an hour is required to partition a large mesh, thus a major overhead for each AEDIF run.

3. Load balancing by processor speed (LB-2)- A simpler load balancing approach was suggested by Alain Dervieux and tested in this study. The idea is that rather than partition the mesh according to the processor speed to obtain partitions of different sizes (LB-1), create homogeneous mesh partitions (equal sizes) and give more partitions to the faster processors at execution. This avoids the necessity to run the mesh partitioner before executing the AERO-F and AEDIF codes. The MecaGRID clusters have either 1 Ghz or 2 Ghz processors. Therefore the INRIA-nina and the IUSTI processors would get two partitions and the INRIA-pf and the CEMEF clusters one partition. It does require an Interface to write the MyMachine.LINUX file. Another advantage of this approach is that the homogeneous partitions can be configured to fit the minimum RAM available (256 MB on the INRIA-pf) thus avoiding swapping in/out of RAM.

   Table 23 and Table 24 shows some non-Globus AEDIF results using the LB-2 method for the 1.03M mesh with 32 and 64 partitions. Table 23 shows that LB-2 using 24 processors (8-nina and 16-pf) is approximately as efficient as using 16-nina and 16-pf processors (a saving of 8 processors). Table 24 shows that LB-2 using 48 processors (16-nina and 32-pf) is approximately as efficient as using 64 processors (32-nina 32-pf) with 1 processor/partition. The LB-2 method run with 48 processors requires 25 percent fewer processors (48 instead of 64) to achieve the same result in approximately the same run time.

CPUs/Parts	Method	INRIA-nina		INRIA-pf		Time
		CPUs	Partitions	CPUs	Partitions	
32/32	STD	16	16	16	16	205 sec
24/32	LB-2	8	16	16	16	226 sec

Table 23: Load balancing method LB-2 using 32 partitions

CPUs/Parts	Method	INRIA-nina		INRIA-pf		Time
		CPUs	Partitions	CPUs	Partitions	
64/64	STD	32	32	32	32	165 sec
48/64	LB-2	16	32	32	32	185 sec

Table 24: Load balancing method LB-2 using 64 partitions

4. Dynamic memory allocation (DMA)- The current version of AEDIF is compiled with F77. The executable size is based on the maximum partition size. Therefore load balancing by processor speed using heterogeneous partitions requires the same RAM for both large and small partitions! To avoid this, Dynamic Memory Allocation should be introduced in future versions of the AEDIF code so that smaller partitions would require smaller RAM. This is most easily accomplished using F90 and would not (based on the author's experience in F90 programming) be difficult to implement.

5. Optimizing RAM- For the same processor speed, different sizes of RAM available may degrade performance in the sense that an IUSTI processor has 1 GB RAM available and INRIA-nina less 1/2 GB, yet both have have 2 Ghz processors. A method to equalize the RAM is as follows. For each nina Node requested, use only one CPU of that Node therefore the nina processor used will have 1 GB RAM available.

13 Summary

Progress has been achieved to realizing the goal of creating a large MPI computer to use hundreds of processors by pooling the cluster resources of the MecaGRID members. This report summarizes experiences in developing and performing computations on the MecaGRID project for 8-64 processors. The MecaGRID connects the INRIA clusters (70 CPUs) with the IUSTI (155 km, 30 CPUs) and CEMEF Ecole des Mines de Paris at Sophia Antipolis (1 km, 24 CPUs) [17]. The AERO-F (mono phase) and AEDIF (two-phase flow) codes have been successfully adapted to GRID Computing. Implicit and explicit solvers were evaluated using the AERO-F mono-phase code and two-phase flows using the AEDIF code.

The largest test case contained more than 1,000,000 million vertices with 7 million unknowns. The MecaGRID runs with 64 CPUs were 4-5 times slower than the same application on the INRIA clusters. The loss in efficiency using the MecaGRID is due to increased communication times needed to transfer data between the three MecaGRID clusters. Local communication times increased by factors of 100-200 over the same run on the INRIA clusters.

Several experiments suggest that the domain decomposition may have an impact on the efficiency of the Grid application[18] The role of mesh decomposition on optimizing commnications needs to be examined and will be an important subject to creating more efficient Grid Computing.

Advances in network transfer technology are needed before an efficient MecaGRID is possible. In 3-5 years high-speed optical networks should be a reality and Grid Computing will become as efficient as cluster computing today.

An interesting option available to create an efficient MecaGRID is to take advantage of the existing 2.5 Gbps VTHD highspeed network connecting INRIA Sophia Antipolis and INRIA Rhone Alpes. This would require a partnership between the two INRIAs sites as INRIA Rhone Alpes is not at present a member of the MecaGRID. Other options would be to create high-speed networks between the existing MecaGRID member sites.

[17] The CEMEF cluster has 62 CPUs but only 24 are available to Globus users

[18] The Communication/Work times for the 262K mesh using 8 CPUs were much smaller than for the 568K mesh with 16 CPUs. This may, in part, be due to a decomposition that resulted in a more optimal message passing and should be evaluated in future studies.

14 Suggested future work

1. The performance of Grids like the MecaGRID is believed to be best when all the processors have public IP addresses avoiding the need for VPNs and tunneling. Thus another extremely important motivation for integrating INRIA Rhone-Alpes as an active member is to permit evaluate the tunneling VPN and the public IP address approaches to determine relative performance and to establish at which point the efficiency of tunneling starts to deteriorate.
2. Another reason for integrating the INRIA Rhone Alpes cluster(s) into the MecaGRID is to take advantage of the existing 2.5 Gbps VTHD high-speed network connecting INRIA Sophia Antipolis and INRIA Rhone Alpes that should result in an efficient Grid.
3. The shockwave-bubble test case is time evolutionary and one is obliged to save the solution profiles at certain intervals in order to create animations. Saving solutions increases the global communication times if MPI sends/receives are used to gather the subdomain profiles on processor "0". This should be avoided whenever possible. A suggestion to minimize global communication times would be to write the profiles locally on each processor and recover them after the run has been completed using scp or sftp.
4. Create global submittal scripts using the Globus ToolKit MetaComputing Directory Service.
5. Upgrade the Globus software at regular intervals 8-12 months minimum.
6. Hardware upgrades at the different sites should be made taking into account Grid computing.
7. Global communication times increase significantly for inter-cluster executions. AEDIF is a research code with many global communications that could be eliminated to improve inter-cluster performances. For example, one could eliminate computing the maximum and minimum of density, pressure, u, v, w.
8. The role of mesh decomposition needs to be investigated.

15 APPENDICES

The user is referred to APPENDICES A-G for additional information useful in using AEDIF.

16 Acknowledgements

The research has been made possible by the ACI-GRID 2002 Project of the French Ministry of Research[19].

The author would like to thank Herve Guillard for the opportunity to work on the MecaGRID project. Thanks to Herve Guillard and Angelo Murrone for their important contributions to two-phase flow advancement that are the basis for the AEDIF code. The collaboration with Patrick Nivet has greatly added to the advancement of the project. A special thanks to the MecaGRID local system managers David Geldreich (INRIA), Carole Torrin (CEMEF), and Jacques Massoni (IUSTI) for their support and patience. A special thanks you to Alain Dervieux for his collaboration and insight in the development of the AEDIF code developed specifically for MecaGRID two-phase flow applications and Bruno Koobus a valuable member of the AEDIF development team and one of the major contributors to the AERO-F development. Lastly, the contributions of Hugues Digonnet and Rodolphe Lanrivain to the mesh partitioner adapted to the AERO-F and AEDIF code are gratefully acknowledged.

[19] http://www.recherche.gouv.fr/recherche/aci/grid.htm

References

[1] B. N'Konga and H. Guillard. Godunov type method on non-structured meshes for three dimensional moving boundary problems. *Comput. Methods Appl. Mech. Eng*, 113:183–204, 1994.

[2] Olivier Basset. Proposed title: Analysis of the mecagrid hardware using the performance utility. Technical report, Ecole des Mines de Paris at Sophia Antipolis, 2004, not released.

[3] A. Dervieux. Steady euler simulations using unstructured meshes. Lecture series 1985-04[20], Von Karman Institute for Fluid Dynamics, 1985.

[4] Hugues Digonnet. *Repartitionnement dynamique, mailleur paralléle et leur applications à la simulations numérique en mise en forme des matériaux.* PhD thesis, Ecole Nationnale Supérieure des Mines de Paris, 2001.

[5] C. Farhat. High performance simulation of coupled nonlinear transient aeroelastic problems. Special course on parallel computing in CFD. R-807, NATO AGARD Report, October 1995.

[6] H. Guillard and A. Murrone. A five equation reduced model for compressible two phase flow problems. Technical Report RR-4778, INRIA - Sophia Antipolis, March 2003, accepted for publication in Journal of Computational Physics, available at http://www.inria.fr/rrrt/rr-4778.html.

[7] Herve Guillard. Journée de travail entre informaticiens développeurs, July 18, 2002.

[8] Rodolphe Lanrivain. Partitionnement de mailliages sur une grille de calcul. Technical report, INRIA Sophia Antipolis, 2004.

[9] R. Martin and H. Guillard. Second-order defect-correction scheme for unsteady problems. *Computer & Fluids.*, 25(1):9–27, 1996.

[10] A. Murrone. *Modèles bi-fluides à six et sept équations pour les écoulement diphasiques à faible nombre de Mach.* PhD thesis, Universite de Provence (Aix-Marseille I), 2004, available at http://www.ccsd.crns.fr.

[20] Published in Partial Differential Equations of hyperbolique type and Applications, Geymonat Ed., World Scientific (1987)

[11] Patrick Nivet. A computational grid devoted to fluid mechanics. Technical report, INRIA Sophia Antipolis, May 2004, in progress.

[12] C.W. Shu and S. Osher. Efficient implementation of essential non-oscillatory shock capturing schemes. *Journal of Computational Physics*, 77(439-471), 1988.

[13] S. Wornom, H. Guillard, A. Murrone, B. Koobus, and A. Dervieux. Two-phase flow calculations based on the mixed-element-volume method. Technical report, INRIA Sophia Antipolis, 2004 in progress.

[14] Stephen Wornom. Rapport d'avancement pour la période du 01/02/03 au 30/07/03. Technical report, INRIA Sophia Antipolis, July 30, 2003.

[15] E. C. Yates. Agard standard aeroelastic configuration for dynamic response i -wing 445.6. Technical Report 765, AGARD, 1985.

APPENDIX A

USEFUL NOTES

1. The partitions shown in Table 22 were made with the version of the CEMEF mesh partitioner created by Lanrivain [8] (CMP_Lanrivain) during his 2003 Internship in the SMASH project directed by Herve Guillard. This version, CMP_Lanrivain, writes the partitions and the flu.glob files in the format readable by AERO-F and AEDIF, and includes both homogeneous and heterogeneous (optimized) options. The mesh partitioner of CEMEF is a very sophisticated code written in C++. Each system upgrade on the cluster has had a dramatic effect on the ability to compile the code even with the aid of the developers. Recent attempts by Patrick Nivet and the author to compile the mesh partitioner were unsuccessful, therefore as this report is written, the mesh partitioner is not available to partition meshes.

2. The maximun number of partitions show in Table 22 is 64. Partitioning for 96 partitions were tried but the iteration scheme in the code did not converged after 24 hours using 32 nina processors to create 96 partitions. The developer, Hugues Digonnet, suggested to limit the number of iterations to 30 maximum rather than the default 3000. This change was made but could not be tested due to 1., this explains why there are no partitions > 64.

3. The CMP_Lanrivain version Makefile to create heterogeneous partitions needs updating.

4. The statement to write the partitions and the flu.glob files in the format readable by AERO-F and AEDIF has been transfer to a more recent version of the mesh partitioner by Hugues Digonnet and Youssef Mesri(2004 SMASH intern), however, the code has not been validated.

APPENDIX B

HOW TO Make and EXECUTE AEDIF

AEDIF can be obtained by contacting Herve Guillard at INRIA Sophia Antipolis.

```
mkdir AEDIF
cd AEDIF directory
cp source, makefiles, ... to AEDIF

cat README
Para13D.h
Param3D.h

In case the *.h gets deleted
cp -p Para13D.h.sav Para13D.h
cp -p Param3D.h.sav Param3D.h

ls Mak* shows
Makefile_aedif
Makefile_aedif_GLOBUS
Makefile_aedif_parameters
Makefile_aero_Interface

1) Create a working directory (WD)
mkdir AEDIF

cd WD
2) cp or link the flu-* and flu.glob files to the WD

3) cp -p AEDIF/Mak* .

4) Make new .h files based on the flu.glob for actual case
make -f Makefile_aedif_parameters
This need to be done only once.
```

```
5) Create the executables:
make -f Makefile_aedif_GLOBUS clean      Globus
make -f Makefile_aedif_GLOBUS aero       Globus
globusrun -f runscript.rsl
or
make -f Makefile_aedif clean         non-GLOBUS
make -f Makefile_aedif aero          non-GLOBUS

./nina_mv.lsf ncpus

nina_mv.lsf: non-globus run script
bsub -J aerodia16 -o test06.out -e test06.err -m "linux-nina linux-pf"  \
     -f "test06.hostnames      < hostnames.out" \
     -f "test06.flu.glbcpu     < flu.glbcpu" \
     -f "test06.flu.lclcpu     < flu.lclcpu" \
     -f "test06.cpu_times.dat  < times.dat" \
     -f "test06.rsitg.data     < rsitg.data"\
     -n $1 mpijob aerodia.x

The non-GLOBUS run script saves the files indicated. The globusrun script
has a FILE_STAGE_OUT option that could be used to save the files. Unless the
FILE_STAGE_OUT is used, the user must save the files manually before submitting
another job, otherwise they will be overwritten.
```

APPENDIX C

GRAPHICAL INTERFACE

The AEDIF graphical interface can be found at

dauphine.inria.fr: /net/home/swornom/AEDIF_Graphics.tar

AEDIF_Graphics contains these files:

```
Aerodia2d_to_Graphics.f
write_variables_VTK2d.c
Makefile_aedif2dg
aedif2dg.inp

Aerodia3d_to_Graphics_v2.f
write_variables_VTK.c
Makefile_aedif3dg
aedif3dg.inp
```

To create the executables:

```
make -f Makefile_aedif2dg (interface for Murrone-Guillard two dimensional code)
make -f Makefile_aedif3dg (interface for AEDIF)
```

cp either aedif2dg.inp and Makefile_aedif2dg (or aedif3dg.inp and Makefile_aedif3dg) to the directory containing the data.

The interface is excuted by:

```
./aedif2dg.x or
./aedif3d.x
```

and interactively answering the questions posed. In advance the aedif3d.inp or aedif3dg.inp files should be modified if necessary. The user has an option to write the graphic data for either the Medit code of INRIA or the ParaView graphics code of http://www.kitware.com. The advantage of ParaView over Medit is that multiple time step data can be processed in the batch mode. This is very useful for data sets of 50-100 to create video animations[21]

[21] The following link contains several animations created during this study. http://www-sop.inria.fr/smash/personnel/Stephen.Wornom/Stephen.Wornom-english.html

APPENDIX D

BATCH MESH PARTITIONER

```
AEDIF contains two directories
1) MeshPartitioner_Batch_script/
2) libs/

cd MeshPartitioner_Batch_script/
follow instructions in the README file.
```

APPENDIX E

PROCESSORS HOSTNAMES

The flu.data input file for the AEDIF code contains a hostname option (yes=1). Table 25 shows the neighbors of processor "0" and their hostnames printed in the file hostname.out file when this option is selected.

Processor	Hostname
0	nina08.inria.fr
1	nina08.inria.fr
2	nina06.inria.fr
3	nina06.inria.fr
4	node9.clustal.com
5	node8.clustal.com
6	node7.clustal.com
7	node6.clustal.com
8	node24.cemef
9	node23.cemef
10	node22.cemef
11	node20.cemef
12	pf8.inria.fr
13	pf8.inria.fr
14	pf3.inria.fr
15	pf3.inria.fr

Table 25: Processor hostname information

Table 26 shows the neighbors of processor "0" and their hostnames.

My Processor		My Neighbors	
CPU	Hostname	CPU	Hostname
0	nina08.inria.fr	1	nina08.inria.fr
0	nina08.inria.fr	8	node24.cemef
0	nina08.inria.fr	9	node23.cemef
0	nina08.inria.fr	10	node22.cemef
0	nina08.inria.fr	12	pf8.inria.fr

Table 26: Processor hostname information

Table 27 shows the neighbors of processor "0" and their hostnames. nprocs= 16 message passing between clusters at time step Total messages passed = 108

Cluster names	Number of messages
nina-nina	2
pf-pf	6
cemef-cemef	6
iusti-iusti	10
nina-pf	5
nina-cemef	7
nina-iusti	5
pf-cemef	7
pf-iusti	11
cemef-iusti	7

Table 27: Processor hostname information

APPENDIX F

TIME ANALYSIS-I

The time analysis is found in the flu.glbcpu and the flu.lclcpu files written at ktsav intervals. The non-GLOBUS run script saves these files. The globusrun script has a FILE_STAGE_OUT option that could be used to save the files but has not been used. Unless the FILE_STAGE_OUT is used, the user must save the files manually before submitting another job, otherwise they will be overwritten.

The minimum, maximum, and average times for all the processors are computed. The average time is the sum of the individual processors divide by the number of processors. MPI passes the fluxes, time step, and gradients (GRD) between partitions, the gradients contain the most data.

```
 ------------------------------------------------------------
globus_24_pf_12_cemef_12_run3.flu.glbcpu
 ------------------------------------------------------------
 Number of time steps            :    10
 Number of solution saves        :     1


 Values of local CPU times       :    1:MIN - 2:MAX - 3:AVRG
 ------------------------------------------------------------
 Wait time to get all needed CPUs:   7.058    22.328    15.205
 Total simulation time           : 795.915   811.187   804.063
 Problem setup time              : 163.527   178.799   171.675
 Total computational time        : 632.386   632.389   632.388
 Write local solution files      :   0.000     0.000     0.000
 Write global solution files     :   0.036   123.378     5.210
 ------------------------------------------------------------
 Total Computational time (Tcomp): 509.011   632.353   627.178
 Total Communication time (Tcomm): 244.473   476.911   442.616
 Twork = Tcomp - Tcomm           : 155.477   387.916   189.772
 Tcomm/Twork                     :   0.630     3.067     0.026
 ------------------------------------------------------------
```

```
Global intra-communication time :  68.562   234.007   164.888
Local  intra-communication time :  43.296   321.118   277.728
-----------------------------------------------------------------
Explicit convective fluxes      :  51.132    82.393    56.942
Explicit nodal gradients        :  25.299    43.051    34.956
-----------------------------------------------------------------
Local intra-comm. transfer rates
   Dt  transfer rate (Mbps)     :   0.535     6.303     1.363
   Grd transfer rate (Mbps)     :   0.907     8.054     2.098
   Flx transfer rate (Mbps)     :   0.857     8.167     1.972
-----------------------------------------------------------------
Local intra-communication times (sec)
   Dt  time                     :   0.651     7.650     5.174
   Grd time                     :  32.095   228.644   200.591
   Flx time                     :  10.550    86.602    71.963
-----------------------------------------------------------------
Other local comput. and I/O time:   3.560     5.234     4.453
Mesh motion and metrics update  :   0.000     0.000     0.000
KtGrd10                         :  10.000    10.000    10.000
-----------------------------------------------------------------
```

APPENDIX G

NEED for a COHERENT GRID COMPUTING POLICY

The need for a Grid coherent policy becomes evident when attempting to analyze Grid communication speeds using the LINUX ping tool. Table 28 shows the results of ping tests between the different MecaGRID member sites. Knowing the routes allows for additional ping tests to determine the transfer rates of the different routers involved.

From	To	Status	Routes
CEMEF	INRIA	successful	node10.cemef (192.168.8.110) 192.168.101.12 cluster.inria.fr (193.51.209.126) cluster.inria.fr (193.51.209.126) sarek.cemef (192.168.8.152) node10.cemef (192.168.8.110)
CEMEF	IUSTI	Failed	unknown
IUSTI	CEMEF	Failed	unknown
IUSTI	INRIA	Failed	unknown

Table 28: Ping tests between MecaGRID sites.

Additional information is given below:
Example 1: ping nina01 from the INRIA frontend cluster

IP address of nina01 is 193.51.209.36

ping -c 100 -R 193.51.209.36

Comment: The -R option shows the routes involved.

PING 193.51.209.36 (193.51.209.36) from 193.51.209.126 :
RR:
cluster.inria.fr (193.51.209.126)
nina01.inria.fr (193.51.209.36)

nina01.inria.fr (193.51.209.36)
cluster.inria.fr (193.51.209.126)

64 bytes from 193.51.209.36: icmp_seq=0 ttl=64 time=0.1 ms
64 bytes from 193.51.209.36: icmp_seq=1 ttl=64 time=0.1 ms
64 bytes from 193.51.209.36: icmp_seq=2 ttl=64 time=0.1 ms
64 bytes from 193.51.209.36: icmp_seq=3 ttl=64 time=0.1 ms

193.51.209.36 ping statistics

100 packets transmitted, 100 packets received, 0% packet loss

round-trip min/avg/max = 0.0/0.1/0.5 ms

Positive: The routes are printed
Negative: The time format is 1 decimal place.

Example 2: ping nina01 from the cemef node10

PING nina01.inria.fr (193.51.209.36) from 192.168.8.110

RR:
node10.cemef (192.168.8.110)
192.168.101.12
cluster.inria.fr (193.51.209.126)
nina01.inria.fr (193.51.209.36)
nina01.inria.fr (193.51.209.36)
192.168.101.21
sarek.cemef (192.168.8.152)
node10.cemef (192.168.8.110)

64 bytes from nina01.inria.fr (193.51.209.36): time=19.821 msec
64 bytes from nina01.inria.fr (193.51.209.36): time=10.425 msec
64 bytes from nina01.inria.fr (193.51.209.36): time=20.509 msec
64 bytes from nina01.inria.fr (193.51.209.36): time=20.590 msec
64 bytes from nina01.inria.fr (193.51.209.36): time=20.683 msec

64 bytes from nina01.inria.fr (193.51.209.36): time=20.747 msec
...
— nina01.inria.fr ping statistics —

100 packets transmitted, 100 packets received, 0% packet loss

round-trip min/avg/max/mdev = 5.087/14.752/22.962/4.909 ms

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-0803