International Journal of Computational Intelligence Research.
ISSN 0973-1873 Vol.2, No.1 (2006), pp. 169-180
© Research India Publications http://www.ijcir.info

Evolutionary Algorithm as a Tool for Advanced Designing of Diesel Engines

Teresa Donateo[1], Domenico Laforgia[1], Giovanni Aloisio[2] and Silvia Mocavero[2]

[1]Research Center for Energy and Environment – University of Lecce
via per Monteroni - Lecce, Italy
{teresa.donateo, domenico.laforgia}@unile.it

[2] CACT/ISUFI & NNL/INFM&CNR – University of Lecce,
via per Arnesano - Lecce, Italy
***{giovanni.aloisio**, silvia.mocavero}@unile.it*

***Abstract*: An evolutionary algorithm has been developed for the design of a diesel engine combustion chamber in order to fulfill present day and future regulations about pollutant emissions and greenhouse gases. The competitive goals to be achieved in engine optimization are the reduction of emission levels (soot, NOx and HC) and the improvement of specific fuel consumption. They have been taken into account by using a multi-objective approach implemented in an optimization tool called HiPerGEO, which is characterized by a very small population and a mechanism of reinizialization, combined with an external memory to store non-dominated solutions.**

The method was applied to the design of the combustion chamber profile and numerical simulations were performed with a modified version of the KIVA3V code to evaluate the fitness values of the solutions. The chamber profile was defined according to five geometrical parameters used as inputs to the optimization method. The output of the simulations in terms of emissions and IMEP were used to define four different objective functions. The search for the optimum was performed by applying the Pareto optimality criterion so that it is not bounded to arbitrary weights assigned to each objective. At the end of the simulation, the user can choose from the final Pareto set the best compromise solution for different applications.

The method allows the optimization with respect to different engine operating conditions, i.e. load and speed values. In the present investigation, four operating modes were considered and weights were assigned to them according to their importance in the reduction of emissions and fuel consumption. The use of a 3D simulation code to simulate the behavior of the engine with respect to four operating modes is a very time expensive approach. To reduce the required computational time, which is prohibitive on a sequential machine, grid technologies were implemented in a grid portal named DESGrid.

Keywords: Grid technologies, industrial problem, micro genetic algorithm, multi-objective optimization, Diesel engines, clustering**.**

I. Introduction

The optimization of an internal combustion engine is a challenging task for the following reasons.

- The potentially large number of options to be tested due to the many possible choices in both design and operating conditions
- The necessity to deal with both continuous and discrete variables
- The multiple and competitive goals to be achieved. A typical case is the contemporary reduction of *soot* and *NOx* emissions in a diesel engine. Most measures taken to reduce particulate will increase *NOx*, and vice versa This is due to the competitive mechanisms of formation of these pollutions which both depend on local temperature and air fuel mixing. This problem is called the “Diesel dilemma”
- The nonlinearities and the complex interactions among design variables and optimization goals
- The presence of constraints, restrictions and limits that the designer must meet due to regulations and performance requirements
- The vague distinction between constraints and objective functions
- The necessity of using a predictive simulation code to model the thermo and fluid dynamic processes that take place in an internal combustion engine
- The high computational time due to the execution of a large number of simulations.

Genetic algorithms (GAs) are suitable for engine optimization thanks to their high robustness and their capability to deal with multi-objective optimization. Moreover, they are simple to use and to combine with existing simulation code without significant modifications. The implicit parallel nature of GAs make it easy to exploit the growing parallel computing power. In fact, they work with a population of solutions, then multiple optimal individuals can be captured in a single run. This is another reason for using GAs to solve a multi-objective problem. On the other hand, the convergence rate of a GA can be low if high accuracy is required. A possible way to achieve a faster convergence rate is the use of a micro-GA approach [3], which reduces the amount of comparisons required to generate the Pareto front.

The combination of GAs and numerical simulations for engine optimization has been considered by Reitz and his research group who applied a computer code (KIVA-GA) to optimize the combustion chamber geometry together with several engine input parameters (e.g. EGR, injection profile, etc.) for a single operating mode [15]-[16]-[17]. De Risi et al. [7]-[4]-[5] found that the effectiveness of a particular combustion chamber in reducing emissions depends on engine load and operating conditions, which means that engines need to be optimized for different operating conditions. Senecal et al. [14] applied the KIVA-GA method to optimize the chamber design for two operating modes. The mesh generator used by Senecal permits a large variety of shapes, but the results presented in [14] are unsuitable for practical application.

The method used by Reitz is based on the definition of a merit function to take into account several objectives, i.e. NOx and soot emissions, specific consumption, etc. This approach allows the use of a single-objective micro genetic algorithm but the search for solutions is bounded to the fixed weight assigned to each objective during all the optimization process.

On the other hand, a multi-objective approach has been considered by Hiroyasu et al. [11] to optimize the shape of the injection rate with a phenomenological model to reduce the computational time. However, the capability of phenomenological models to simulate engine behavior is limited. In fact, phenomenological models are calibrated according to specific conditions and cannot be expected to perform reasonably well in simulation of different kinds of engines and operating conditions. This reduces the degrees of freedom in the use of GAs.

In the present investigation, an innovative optimization tool named HiPerGEO (High Performance Genetic algorithm for Engine Optimization) is illustrated. HiPerGEO differs from KIVA-GA because it uses a multi-objective approach and allows engines to be optimized with respect to several operating modes. Moreover, unlike the approach of Hiroyasu, the computational time has been reduced not by reducing the confidence in engine simulations, but with the use of a micro-GA and advanced computational technologies (grid technologies.) Engine simulations were performed with a modified version of the KIVA3V code with improved models for spray and combustion. The HiPerGEO uses a micro-GA model where the rank method is applied to compare the individuals and the Pareto front is uniformly defined with the use of clustering. Moreover, the choice of a limited number of final optimal solutions is also performed in a completely automatic fashion by using a clustering algorithm.

The use of a simple web interface allows a trusted user to execute a HiPerGEO run by selecting the algorithm parameters.

II. The Optimization Method

The proposed design method consists of ten optimization steps:

STEP 1. Identification of the geometrical parameters to be optimized, e.g. combustion chamber shape, compression ratio, injector position, inlet valve closing, etc.

STEP 2. Choosing a model to predict the engine behavior once the geometrical parameters are changed: If multi-dimensional simulation codes are used, this step requires the definition of a parametric computational mesh to divide the geometrical domain into a sufficiently high number of cells.

STEP 3. Choosing the strategic goals to reach: e.g. improvement of engine torque, reduction of pollutant emissions, control of combustion noise, etc. Note that only the most important output values have to be included in the fitness array, since the other ones can be used as penalty functions (see step 5.)

STEP 4. Selection of the operating conditions (modes) for the optimization and choice of their weight in the definition of the fitness array: Since engine behavior is strongly influenced by load and speed, as many modes as possible should be used and the weight should be assigned according to the importance of each mode for each objective, e.g. if the optimization process aims at reducing emissions, high-speed high-load modes are not important while low and medium values of load and speed could be weighted according to their occurrence in the driving tests.

STEP 5. Definition of penalty functions to penalize geometrical solutions that do not respect user defined requirements: e.g. noise emissions, structural constrains on maximum pressure, maximum discharge temperature, etc.

STEP 6. Setup of an interface between the genetic algorithm and the engine model.

STEP 7. Identification of the reference configuration for the optimization: e.g. a commercial chamber that represents the baseline configuration to improve or a target solution that indicates the best expected results obtainable by changing engine geometry. This step requires the definition of the range of variation for all parameters chosen at step 1.

STEP 8. Running the genetic algorithm with a selected number of individuals to be analyzed . The higher the number of individuals to be tested, the higher the required computational time and the higher the confidence in the optimization results.

STEP 9. Representation of the Pareto set in the hyperspace defined by the fitness array components and selection of few optimized chambers by means of a clustering algorithm.

STEP 10. Choice of the optimal configuration to be built and tested according to the actual user requirements.

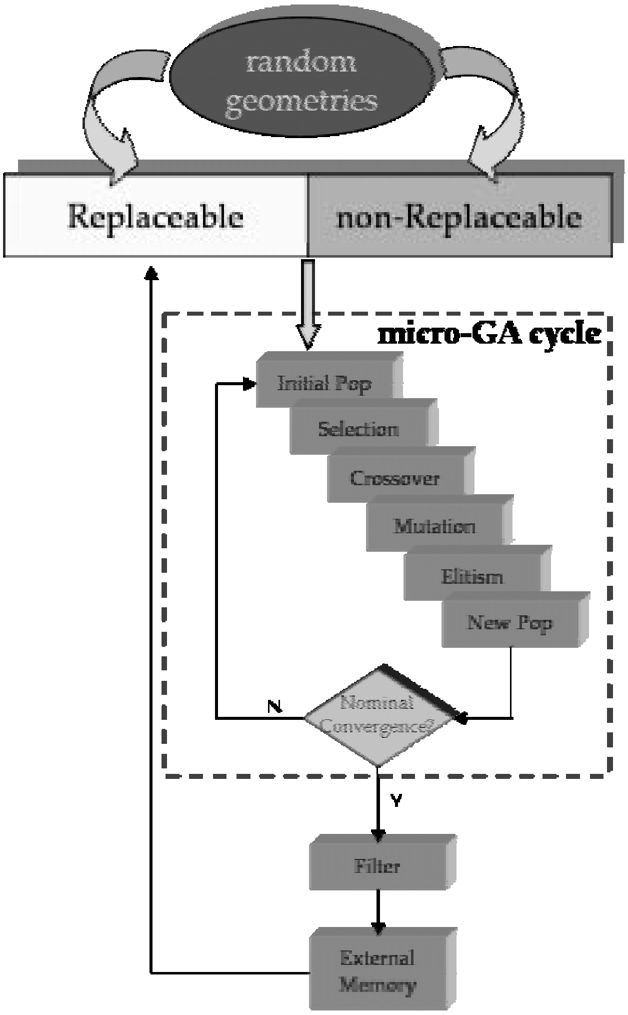

Figure 1. Micro-GA flow chart.

A. Micro-GA model

In the HiPerGEO algorithm the micro-GA technique is applied to engine optimization by using Coello's and Pulido's [3] approach which performs the multi-objective optimization on two levels (see Fig. 1.) Externally, a fixed number of iterations is executed. At each iteration, the micro-GA cycle is performed until the nominal convergence is reached. Nominal convergence may be defined in terms of a fixed (generally low) number of cycles to be executed or in terms of similarity among the solutions belonging to the micro-population.

At the beginning of the process, the algorithm randomly generates the chromosomes belonging to the population memory. Solutions are located into the two portions (replaceable and non-replaceable) of the population memory. This is performed only once because the former portion contains solutions that may be replaced during the optimization process while the latter never changes, and then represents a source of diversity for the algorithm.

The percentages of each memory portion can be regulated by the user, which allows the execution of different experiments to evaluate what is the best choice for a specific test case.

From both memory portions an initial small size population is selected for each micro-GA cycle.

The best individual belonging to the micro-population is passed unaltered to the next micro-generation (first form of elitism.)

The other solutions of the new micro-population are generated during the micro-GA cycle by applying standard genetic operators such as selection, crossover, and mutation.

At the end of the micro-GA cycle, the algorithm verifies if the nominal convergence has been achieved. In this case, any of the new population solutions can be considered representative and the algorithm randomly selects one of them. This individual is copied into a separate memory called "external memory", where all the non-dominated solutions are collected forming the Pareto front.

Coello's and Pulido's model suggests two other types of elitism:

1. The representative solution of the micro-GA cycle replaces a randomly selected individual of the replaceable portion if it wins the tournament.
2. At a fixed number of iterations some of the non-dominated solutions are used to update the replaceable portion.

B. HiPerGEO Algorithm

In this paragraph the specific features of HiPerGEO in the application of the micro-GA model to engine optimization are described in detail.

1) Population Memory generation

A fixed number of random solutions are located into the two portions belonging to the population memory according to a percentage defined by the user. The chromosomes are represented by geometric and control parameters of the engine. Parameters values are selected into ranges of allowable values fixed by the user (see paragraph III.A.)

2) Fitness evaluation

The fitness values of each chromosome, representing a possible engine configuration, are calculated via CFD simulations for each operating mode. Both memory portions contain a reference engine configuration (see paragraph III.E.)

3) Micro-GA cycle

At the micro-GA cycle level, the N individuals of initial micro-population are obtained by selecting the chromosomes from either the replaceable portion or the non-replaceable one with a probability specified by the user.

The three elitism methods suggested by Coello and Pulido

[3] were all implemented in HiPerGEO. To account for the multi-objective character of engine optimization, the approach developed by Fonseca [8] was followed to select the best individuals to apply elitism.

In HiPerGEO the individuals are ranked according to the Pareto criterion of dominance. If *F(x)* is the fitness vector associated to solution *x*, *F(y)* is the fitness vector of individual *y*, and the goal is the maximization of all the fitness component of *F*, than *F(y)* is said to dominate *F(x)* if the condition (1) is verified:

$$\forall\, i\, (F_i(x) \leq F_i(y)) \wedge \exists\, i\, (F(x_i) < F(y_i)) \qquad (1)$$

If a vector isn't dominated by any other, it is called non-dominated or not inferior. The goal of a multiobjective optimization is to find all the Pareto points, i.e the set of individuals whose fitness vectors are non dominated.

Thus, the overall fitness of an individual *j* is assigned according to its rank *r(j)*, which is defined from the number of fitness vectors by which *F(j)* is dominated, increased by 1. Thus, the Pareto individuals have rank equal to one and the maximum value of the rank is equal to the population size.

Rank method pseudo-code is the following:

```
For  i=1 to N
  Rank(i)=1
    For m=1 to N
        If (($F_1$(i) - $F_1$(m)<=0) and ($F_2$(i) - $F_2$(m)<=0) and
        ($F_3$(i) - $F_3$(m)<=0) and ($F_4$(i) - $F_4$(m)<=0))
            If (($F_1$(i) - $F_1$(m)<0) or ($F_2$(i) - $F_2$(m)<0) or
            ($F_3$(i) - $F_3$(m)<0) or ($F_4$(i) - $F_4$(m)<0))
        Individual is dominated
        Rank(i) = Rank(i) + 1
    next m
next i
```

After computing the rank of each solution, a portion of the roulette in inverse proportion to its rank is assigned to the individual. In this way, selection is inclined to prefer non-dominated chromosomes (belonging to the Pareto front.)

After selection, the crossover technique is applied over the selected geometries. The uniform crossover has been chosen since experience [12] showed its stability compared with other crossover methods and its efficiency if applied on simple individuals (vectors with few parameters.)

The next step is the application of the uniform mutation with a low probability. This kind of mutation allows new values of the engine parameters to be considered with respect to the micro-population. The application of the mutation operator is driven by the necessity to explore the search space obtaining new building blocks. On the other hand, a high mutation probability could slow down the convergence rate.

4) Nominal convergence

In the present work, the micro-GA cycle stops when nominal convergence is reached in terms of similarities among the chromosomes.

For each geometric parameter, a range of similarity is fixed; the nominal convergence is reached when the difference in terms of chromosomes is within the range of similarity for all of the parameters values. Once the similarity criterion is satisfied, a representative individual is selected as a nominal solution. However, if convergence is not reached within a fixed number of cycles, the micro-GA execution stops and the individuals are ranked to select the nominal solution.

5) External Memory

The external memory contains the Pareto front and its size can be specified by the user. The nominal solution of the micro-GA cycle is copied in the external memory. However, this happens only when the nominal solution is non-dominated with respect to the individuals already belonging to the external memory.

Sometimes, the Pareto front contains a high number of solutions. The selection of the final good individuals, according to the user experience, can represent a problem when an extremely large Pareto front has to be considered. For this reason, the HiPerGEO algorithm allows the user to limit the Pareto front dimension specifying the external memory size. After each iteration, the algorithm verifies if the number of solutions stored into external memory exceeds the maximum size defined by the user during the submission of the optimization process. When this happens, it is necessary to exclude some of the individuals, preserving a uniform distribution of solutions. In order to do this, a clustering algorithm is generally applied [18]. Cluster analysis partitions a collection of solutions into groups of relatively homogeneous elements, then selects just a representative individual for each cluster. The average linkage method [13] suggests, for example, to select the centroid (the point with minimal average distance to all other points in the cluster) as representative solution. In the present work, a new method to reduce the external memory size has been defined. The HiPerGEO procedure to perform the uniformity of the Pareto front is describing in the following steps:

1. Definition of a matrix *D* of size *m* x *m*, where the generic element d^{ij} is the distance between i^{th} and j^{th} individuals.
2. Search of the couple of points which have the minimum distance.
3. Deleting of the point in the couple selected in previous step which have minimum distance from a third point.
4. If the number of excluded points is lower than *m* – *n*, go to step 2.

The selection of the centroid as representative solution, without knowing a priori the features of the Pareto front, does not guarantee a final spatial uniform distribution. Sometimes a cluster can be very near to another, then the selection of the opposed extreme instead of the centroid could be the better choice. Fig. 2 shows the comparison of the two methods if a front of ten solutions must be reduced to five individuals. The centroid of C2 is not a good choice due to the minor distance of C2 from C3 compared with C2 from C1. The application of HiPerGEO clustering method allows balancing every case the solutions belonging to the front.

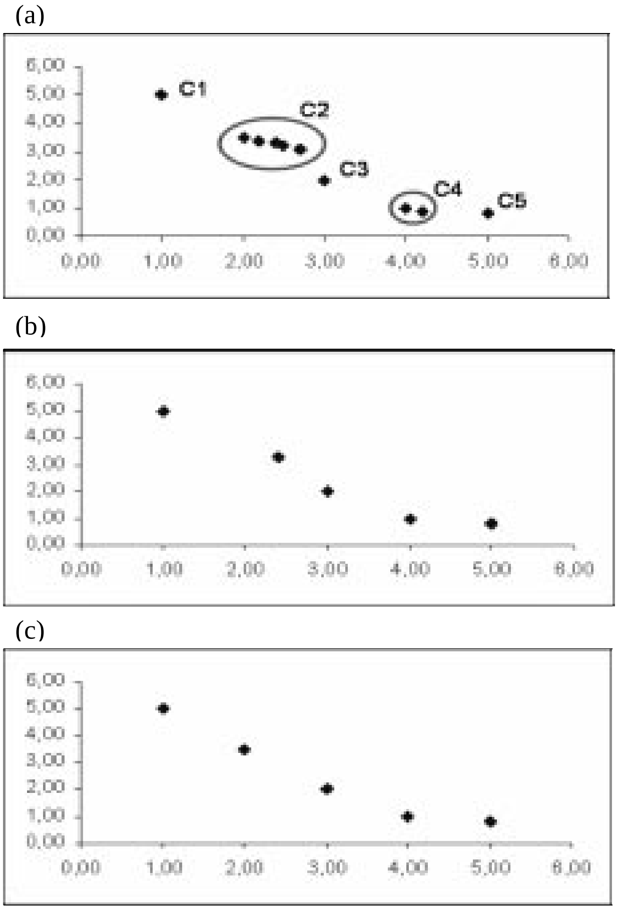

Figure 2. (a) Definition of clusters. (b) Pareto front provided by selection of the centroid. (c) Pareto front provided by HiPerGEO clustering.

6) Replaceable portion

The nominal solution of the micro-GA cycle is also compared with a randomly selected individual of the replaceable memory according to the non dominance criterion.

The winner of the match will remain in the memory portion, while the loser will be excluded (second form of elitism.)

Finally, after a fixed number of iterations, some solutions from the external memory (Pareto front) are used to match the same number of solutions belonging to the replaceable portion. Also in this case, the winners of the match will fill the memory portion, driving the evolution of the algorithm, while old chromosomes will be excluded (third form of elitism.)

III. Design problem: the combustion chamber of a Diesel Engine

A. Identification of the input parameters

In direct injection engines, the combustion chamber is represented by the space defined, at each time, by the cylinder head and walls and the piston crown. A symmetrical cavity, called the bowl, is usually present in the piston to allow fuel to be injected, mixed with air and burned. Since the combustion and emissions mechanisms of formation are strongly affected by the flow field produced by the chamber shape, the optimization of the bowl profile is a strategic way to fulfill present day and future regulations about pollutant emissions and greenhouse gases, which depend on fuel consumption.

To optimize the bowl profile, a simple parametric schematization has been considered by selecting the five geometrical features of Fig. 3, where a half cross section of the piston is shown. Instead of binary coding, a representation with real numbers was used in HiPerGEO since the variables were in continuous domain.

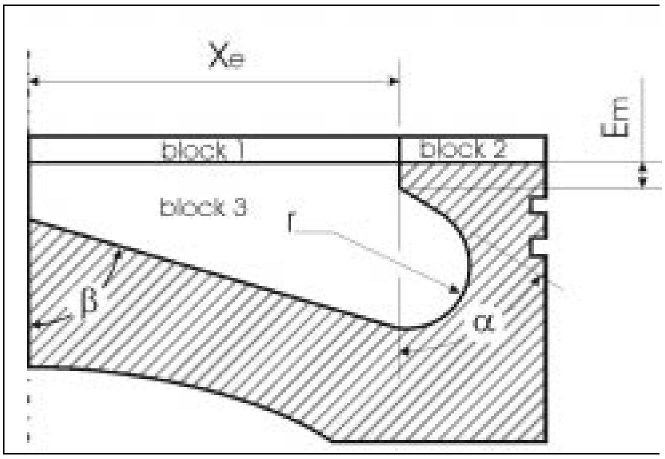

Figure 3. Parameterization of the bowl mesh

The application required the development of the “Meshmaker” tool which automatically generates a structured computational mesh with the following maximum cell size: $\Delta x =$ 2.2 mm, $\Delta z =$ 1.4 mm and $\Delta \vartheta =$ 2°.

The generation process is automated: the Meshmaker spends few seconds to produce the computational mesh selecting the values of geometric parameters of Fig. 3 within the ranges of allowable values fixed by developers (see paragraph III.E.) The Meshmaker outputs the input text file for the K3PREP mesh pre-processor and a second file containing the parameters vector which are used during the crossover and mutation phases.

B. Choice of the optimization goals

A modified version of the KIVA3V code has been used in the present investigation to evaluate the behavior of the combustion chambers selected by the genetic algorithm [4]. Thanks to improved models for spray and combustion, the new version of the KIVA3V code is able to accurately predict the behavior of modern direct injection diesel engines when changing geometrical and control parameters like injection strategy and Exhaust Gas Recirculation (EGR.)

The optimization has been aimed at minimizing pollutant emissions (*soot*, NO_x and Unburned Hydrocarbons (*HC*)) at low speed and load and maximizing engine performance at high load. The effect of the combustion chamber on engine performance was evaluated by considering the Indicated Mean Effective Pressure. *IMEP* is a internal torque indicator that measure the capability of a combustion chamber to convert chemical energy in mechanical work at a specific operating condition and is independent of engine size, speed and mechanical losses. Since HiPerGEO was developed for maximization problems, the fitness functions were defined as follows:

$$F_1 = \sum_{i=1}^{Nmodes} F_{1p}(i) \cdot w_1(i) \left(\frac{(NO_x)_0}{NO_x} \right)_i \qquad (2)$$

$$F_2 = \sum_{i=1}^{Nmodes} F_{2p}(i) \cdot w_2(i) \left(\frac{(soot)_0}{soot} \right)_i \qquad (3)$$

$$F_3 = \sum_{i=1}^{Nmodes} F_{3p}(i) \cdot w_3(i) \left(\frac{(HC)_0}{HC} \right)_i \qquad (4)$$

$$F_4 = \sum_{i=1}^{Nmodes} F_{4p}(i) \cdot w_4(i) \left(\frac{IMEP}{(IMEP)_0} \right)_i \qquad (5)$$

Mode	**1**	**2**	**3**	**4**
Engine speed [rpm]	1500	2000	3000	5300
Indicated Mean Pressure (IMEP)	4.3	8.0	25.0	20.5
Weights for emissions, w_1=w_2=w_3	0.5	0.5	0	0
Weights for IMEP, w_4	0	0	0.5	0.5

Table 1. Operating conditions for the optimization

The behavior of each configuration was evaluated by comparing its predicted emission levels and *IMEP* value with those produced by a baseline chamber for the same operating mode, marked with subscript 0.

For the optimization process described in the present investigation, the four modes of Table 1 have been used.

In the calculation of the fitness functions, the four modes were taken into account by defining a weight on the basis of their influence on the definition of engine emissions and performance. These weights were named $w_j(i)$, j=1,4 , where 1=*NOx*, 2=*soot*, 3=*HC* and 4=*IMEP*. Note that the same weight was assigned to mode 1 and 2 in the calculation of the fitness components F_1, F_2 and F_3. In the same way, a weight equal to 0.5 was assigned to mode 3 and 4 for IMEP maximization.

In equations (2)-(5), N_{modes} is the total number of modes considered in the application and $F_{jp}(i)$, j=1,2,3 is the value of the penalty function for the fitness component *j* calculated on mode *i.*

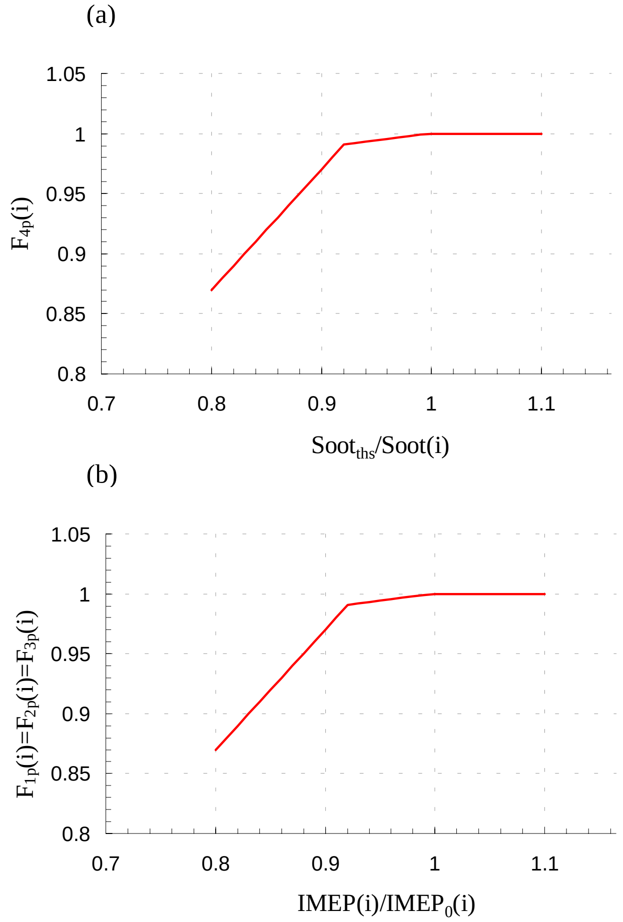

Figure 4. Penalty functions used in the optimization.

C. Choice of the penalty functions

Two penalty functions were considered to take into account *IMEP* at low speed and smoke at full load. *IMEP* is considered the main objective at full load but it has to be taken into account at low load and speed too. In fact, *IMEP* values can be very low, fuel injected being the same, if the completeness of the combustion processes is somehow prevented. In spite of their low performance, these chambers could be considered good solutions because they produce very low levels of *NOx*. For this reason, the penalty function plotted in Fig. 4.a was used to penalize the chamber configurations with low *IMEP* values at low speed and load. If the current chamber gives an *IMEP* value higher than the baseline configuration, the penalty function is set equal to 1 and no penalization is given to the chamber. Otherwise, the chamber is slightly penalized if the reduction of *IMEP* is inferior to 8% while penalization is much higher when the reduction is greater than 8% with respect to the baseline case. The same criterion was applied at full load to penalize chamber configurations with soot emissions higher than a prefixed threshold value. In the present investigation, the value $Soot_{ths}$=0.78 g/kgf was considered (baseline value at operating mode 4.)

The multi-objective nature of the problem could make some solutions located at the extremity of the front to be unallowable due to the poor value of a specific fitness function. However, the definition of two penalty functions provides all the solutions of the Pareto front to be allowable.

D. *Reduction of the computational time*

For each individual evaluated in the HiPerGEO process, the same number of KIVA3V runs as the number of modes has to be performed. To reduce the required computational time, which is prohibitive on a sequential machine, grid technologies [10] were implemented in a grid portal named DESGrid (Fig. 5.) The grid portal architecture, showed in Fig. 6, consists of three essential modules: a web interface to access the system transparently; a second module for the management and the execution of the HiPerGEO, and a grid resource manager to optimize the use of the available computational resources.

The HiPerGEO manager is a demon, the task of which is the iterative control of the job requests made by the user: each request corresponds to the generation of a HiPerGEO optimization process with its own evolution. User requests are queued and will be asynchronously processed by the manager, which has the responsibility to coordinate and execute the optimization. However, each HiPerGEO process requires a large number of KIVA3V simulations.

Each request for a simulation is submitted to a scheduler, the DES Manager, responsible for the scheduling of all the simulation requests on the available grid resources. The scheduler selects the machine for the execution using information provided by an information system. The GRB [1] and GRB-GSIFTP libraries [2] have been used to provide secure file transfer and batch and interactive jobs submission.

The typology of the present scientific problem calls for the implementation of a simple Web interface that allows users to start a distributed simulation and monitor the status of the execution through the following implemented services.

1. HiPerGEO parameters definition: the user can modify HiPerGEO parameters before starting the optimization process. The application stores these new values that will be read at run time.
2. Optimization process submission: the user submits the job by clicking on the submit button. Since the system is totally transparent, the user doesn't know what the system will do from that moment to the end of the process.
3. Monitoring of the process status: when a user asks for the process status to the system, a web page containing the fitness values of the current best solutions (chamber geometries) is provided; the page also display a 3D model of the half piston with the current bowl and a text file with the bowl profile.

Welcome to DESGrid web site

Job submission

Population memory size

Initial population size

External memory size

Replaceable portion (%)

Mutation probability

Iterations number

Replaceable portion probability

Replacement percentage

Replacement cycle

Iteration	Fitness	Link	Chamber

Figure 5. DESGrid portal

However, only a trusted user can access these functionalities. The Grid Security Infrastructure (GSI) protocol, as provided by the Globus Toolkit [9], provides single sign-on authentication, communication protection, and support for delegation of credentials. GSI uses X.509 certificates as the basis for the user authentication.

E. Choice of the reference conditions

The test case described in the present investigation refers to a single-cylinder research engine whose specifications are reported in Table 2. A commercial chamber, representing the starting point of the optimization, has been considered as the baseline case.

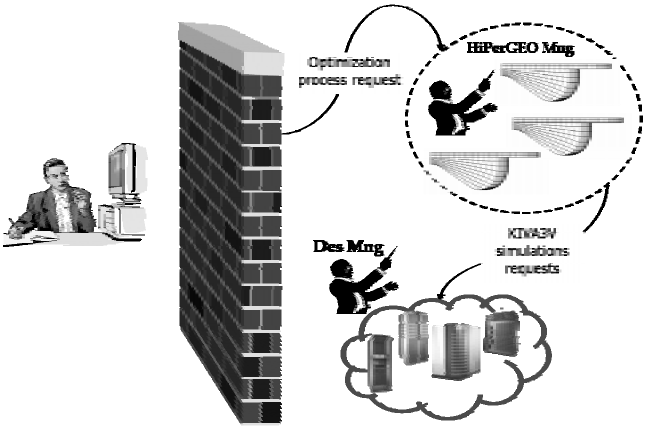

Figure 6. DESGrid architecture

The ranges of variation of parameters X_e, r and E_m, reported in Table 3 were chosen on the basis of technological and geometrical restrictions. The injection strategies adopted for mode 1 and 2 in order to obtain a partially homogeneous charge diesel engine are reported in Table 4.

Displacement [cm^3]	420
Compression ratio	17.2
Intake valve closing [crank angle, CA]	134 BTDC
Injection system	Common Rail
Holes diameter [mm]	0.145
Number of holes	7

Table 2. Engine specifications

Parameter		**Range of variation**
X_e	[mm]	15.0-34.0
α		-90° – 90°
β		45° – 90°
r	[mm]	2.0-14.0
E_m	[mm]	1.0-5.0

Table 3. Input parameters

Mode	1	2	3	4
Engine speed [rpm]	1500	2000	3000	5300
Injection pressure [MPa]	70	90	160	180
Advance of Pilot Injection [CA BTDC]	-60	-60	-	-
Advance of Main Injection [CA BTDC]	-5	-1	-11	-25
Total injected mass [mg]	11	19	59	49

Table 4. Injection specifications

F. Run of the genetic algorithm

The HiPerGEO was run with the parameters of Table 5 and several fully optimized combustion chambers with respect to the four fitness functions were obtained.

Maximum Number of Iterations	100
Population Memory Size	50
Replaceable Portion (%)	70
Initial Population Size	5
Mutation Probability	0.04
External Memory Size (%)	100

Table 5. HiPerGEO parameters

Due to the complexity of the optimization, the analysis of the fitness functions could not be sufficient to underline the effect of the different operating modes and the comparison with the baseline chamber. For this reason, the plots of Fig. 7

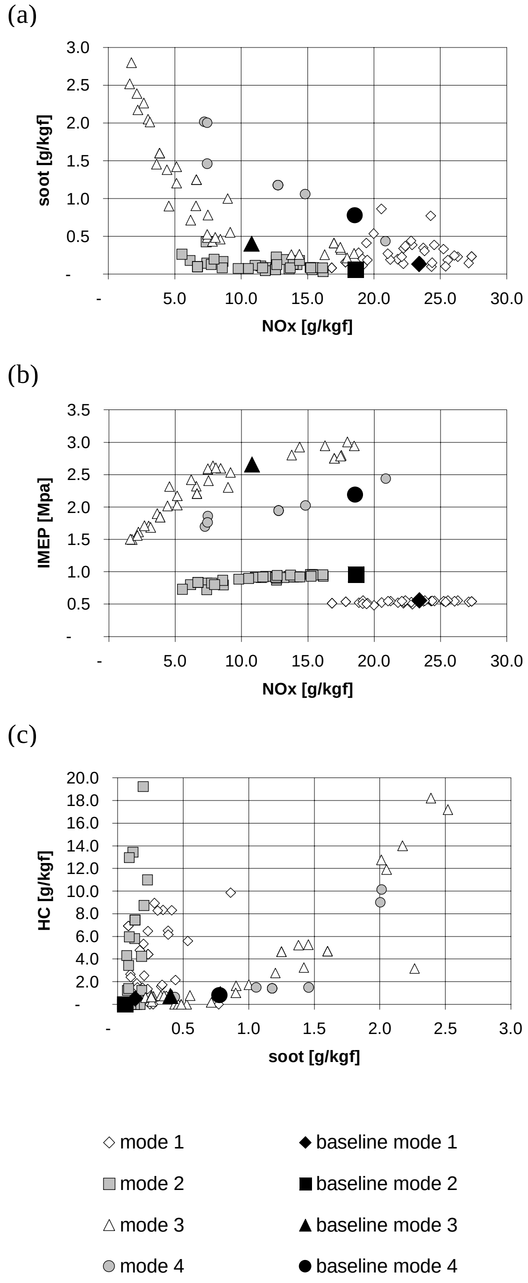

Figure 7. Population of the end of the HiPerGEO run

Fig. 7c where the points representing the baseline configuration are very close to the origin of the plot. shows the population at the last generation in terms of emissions and performance instead of showing the corresponding fitness values.

This choice allowed the following analysis of the optimization results.

- The baseline chamber appears to be optimized for *soot* and *HC* reduction for all modes. In fact, very small improvements on *soot* and *HC* were obtained with the HiPerGEO. This can be easily assessed by considering
- On the contrary, a strong reduction of *NOx* can be obtained with almost all the optimized chambers. The improvement in *IMEP* is up to 15% for all modes and corresponds to a reduction of fuel consumption by the same amount.

G. Clustering of the solutions

The clustering algorithm was used not only during the HiPerGEO run to uniformly define the Pareto front, but also at the end of the optimization process to reduce the number of chambers to be analyzed.

The non dominated chambers found by the HiPerGEO were grouped in clusters according to their shape in order to obtain five chambers which differ significantly. In this way, chambers very similar were excluded by the analysis of the optimized configurations. The outcome of the clustering process is shown in Fig. 8.

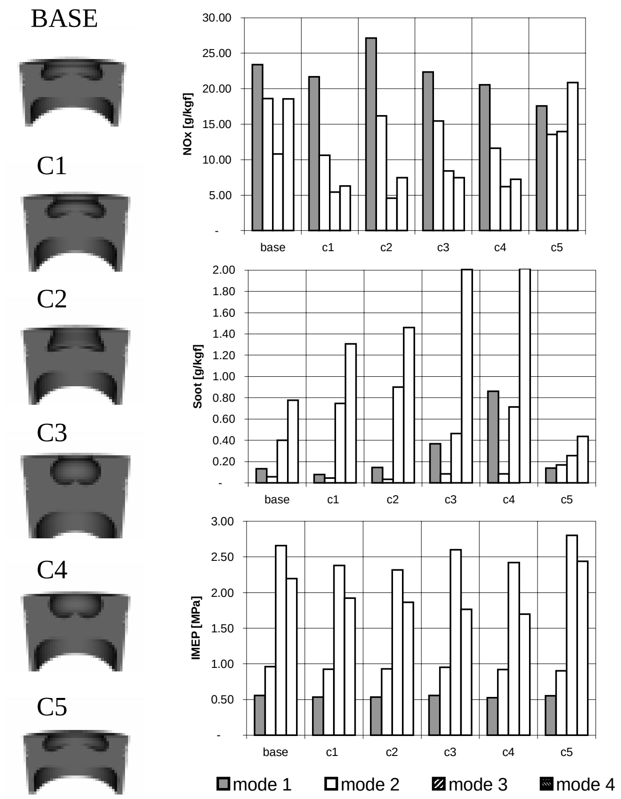

Figure 8. Optimized combustion chambers selected by means of the clustering algorithm

H. Choice of the "best" chambers

Even if the five chambers of Fig. 8, selected in a completely automatic fashion, are all good compromises in the achievement of the optimization goals, the final choice is left to the designer according to his experience, prior knowledge and other constraints.

For example, chamber C1 could be preferred to reduce emissions in urban driving cycle (mode 1 and 2) while C5 can be considered the best compromise at high speed and load. On the other hand chambers C3 and C4 could be chosen to reduce *NOx*, in spite of their high emissions of *soot*, according to the user's needs.

If this preference would change in the future for the use of gas after treatment devices or because of new tendencies in diesel engine development, the optimization process would not need to be repeated since the best configurations for the new goal would be already known.

IV. Computational time

The required computational time T_m to execute a HiPerGEO run is given by:

$$T_m = T_i + T_{ga} + T_{eval} \qquad (6)$$

where T_i is the time required for the initialization step, T_{ga} is the overall sequential time for the genetic algorithm operators (negligible) and T_{eval} is the time required to evaluate solutions during the micro-GA cycle.

T_i is the sum of T_{gen} (the time required by Meshmaker to generate the population memory) and the time to evaluate the population memory as reported in eq. (7.)

$$T_i = T_{gen} + \frac{T_{run} \times N_m \times P_s}{N_p} \qquad (7)$$

where:
T_{run} is the sequential time for each simulation of the engine cycle, depending on the server features;
N_m is the number of modes (load and speed values);
P_s is the population size;
N_p is the number of available processors;

The time required to evaluate solutions during the micro-GA cycle is given by:

$$T_{eval} = \frac{N_{it} \times N_c \times T_{run} \times N_{sim}}{N_{pc}} \qquad (8)$$

where:
N_{it} is the number of iterations;
N_c is the average number of cycles required to achieve nominal convergence in each iteration;
N_{pc} is the number of processors actually used per cycle, depending on the micro-population size and the number of modes, N_{pc}=min(N_p, N_{sim});

N_{sim} is the number of simulations per cycle, given by:

$$N_{sim} = N_m \text{ x } (MP_s\text{-}1) \qquad (9)$$

where MP_s is the Micro-population size.

Note that the best individual for each cycle is copied in the new micro-population. Since it has been already evaluated, the individuals to evaluate are (MP_s -1.)

A comparison among computational time needed on different architectures is reported in Table 6.

The first column in Table 6 shows the computational time required by the optimization process if it is executed on a single processor of an AlphaServer where the computational time of a KIVA3V simulation is equal to 35 minutes. In the second column the execution of the optimization process using the 16 processors of the same AlphaServer cluster provides a reduction of the computational time equal to 94%. The porting of the KIVA3V code on the Itanium clusters of the SPACI (Southern Partnership for Advanced Computational Infrastructure) sites allows reducing the computational time of each run (Trun) from 35 to 9.5 minutes. Note that the number of modes can be adjusted so that the number of simulations per cycle be equal to Np. For this reason, two columns are reported in Table 6 for the Itanium server. The first one illustrates the reduction of Tm due to the lower time required for each KIVA3V run (Trun), while the second shows that it is possible to increase the number of modes fully exploiting the available processors with the use of grid technologies. Due to the independence of needed simulations, the computational time linearly decreases where the number of processors increases.

	Sequential (Alpha Server)	**Alpha Server**	**Itanium Server**	**Itanium Server**
T_{gen}	6.25 (min)	6.25 (min)	6.25 (min)	6.25 (min)
T_{run}	35 (min)	35 (min)	9.5 (min)	9.5 (min)
N_m	4	4	4	32
P_s	50	50	50	50
N_p	1	16	128	128
$\mathbf{T_i}$	**7006.25 (min)**	**443.75 (min)**	**21.09 (min)**	**125 (min)**
$\mathbf{T_{ga}}$	**0.5 (min)**	**0.5 (min)**	**0.5 (min)**	**0.5 (min)**
N_{it}	100	100	100	100
N_c	7	7	7	7
MP_s	5	5	5	5
N_{sim}	16	16	16	128
N_{pc}	1	16	16	128
$\mathbf{T_{eval}}$	**392.000 (min)**	**24.500 (min)**	**6650(min)**	**6650 (min)**
$\mathbf{T_m}$	**277 (days)**	**17.32 (days)**	**4.63(days)**	**4.70 (days)**

Table 6. Required computational time

V. Conclusions

This investigation illustrates the application of an evolutionary algorithm to the design of diesel engine in order to reduce emissions and improve fuel consumption. In particular, the shape of a direct injection combustion chamber was optimized by means of a micro-GA (HiPerGEO) combined with a modified version of the KIVA3V code.

The main specification of HiPerGEO with respect to the particular application are:

- the representation with real numbers of the five input parameters;
- the consideration of four different fitness and two penalty functions;
- the use of weights to take into account different operating conditions;
- the rank method to deal with the multi-objective problem;
- the clustering method to uniformly define the Pareto front;
- the micro-GA approach performed on two optimization levels;
- the use of three forms of elitism.

The results of the optimization showed that:

- several configurations capable of improving engine behavior with respect to four optimization goals can be identified with the method;
- the choice between the selected chambers depends on user's need; if this preference or because of new tendencies in diesel engine development, the optimization process should not be repeated since the best configurations for the new goal would be already known;
- the design of a new optimized combustion chamber for a small bore direct injection diesel can be carried out in a completely automated fashion, with respect to 4 modes in about 18 days on AlphaServers;
- the use of advanced technologies not only reduces the required computational time but also allows the complexity of the system to be increased, e.g by increasing the number of modes
- since engine behavior is strongly influenced by load and speed, the possibility to include a large number of modes would improve the confidence in the results of the optimization. However, the validity of the method has been already validated by means of experimental tests [6].

References

[1] G. Aloisio, E. Blasi, M. Cafaro, I. Epicoco. "The GRBLibrary: Grid Programming with Globus in C". In *Proceedings of HPCN Europe*, pp. 133-140, 2001.

[2] G. Aloisio, M. Cafaro, I. Epicoco. "Early experiences with the GrifFTP protocol using the GRB-GSIFTP library". *Future Generation Computer Systems, Special Issue on Grid Computing: Towards a New Computing Infrastructure*, Volume 18, Number 8, pp. 1053-1059, 2002.

[3] C. Coello, A., G. Pulido, T. "Multiobjective Optimization using Micro-Genetic Algorithm". In *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO), pp. 274-282, 2001.

[4] A. De Risi, T. Donateo, D. Laforgia. "An Innovative Methodology to Improve the Design and the Performances of Direct Injection Diesel Engines". *International Journal of Engine Research*, Vol. 5 No 4, 2004.

[5] A. De Risi, T. Donateo, D. Laforgia. "CFD Modeling of Pilot Injection and EGR in DI Diesel Engines". In *Proceedings of Fall Technical Conference of the ASME Internal Combustion Engine Division*, ICEF2004-837, 2004

[6] A. De Risi, T. Donateo, D. Laforgia, G. Aloisio, E. Blasi, S. Mocavero. "An Evolutionary Methodology for the Design of a D.I. Combustion Chamber for Diesel Engines". In *Proceedings of THIESEL Conference on Thermo and Fluid-Dynamic Processes in Diesel Engines*, ISBN 84-9705-621-3, Volume I, pp. 545-559, 2004.

[7] A. De Risi, D. Manieri, D. Laforgia. "A Theoretical Investigation on the Effects of Combustion Chamber Geometry and Engine Speed on Soot and NOx Emissions". In *Proceedings of ASME-ICE*, vol. 33-1, pp. 51-59, Book No. G1127A, 1999.

[8] C. Fonseca, M., J. Fleming. "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization". In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 416-423, 1993.

[9] I. Foster, C. Kesselman. "GLOBUS: a Metacomputing Infrastructure Toolkit". *International Journal of Supercomputing Applications*, 11(2) pp. 115-28, 1997.

[10] I. Foster, C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*, Edited by I. Foster and K. Kesselman, Morgan Kaufmann Publishers, Inc., ISBN 1-55860-475-8, 1999.

[11] T. Hiroyasu, M. Miki, M. Kim, S. Watanabe, H. Hiroyasu, H. Miao. "Reduction of heavy duty diesel engine emission and fuel economy with multi-objective genetic algorithm and phenomenological model". *SAE Special Publication SP-1824 Diesel Fuel Injection and Sprays*, pp. 211-224, 2004.

[12] M. Mitchell. *An Introduction to Genetic Algorithms*, The MIT Press, Cambridge, MA, 1996.

[13] J. N. Morse. "Reducing the size of the nondominated set: Pruning by clustering". *Computers and Operations Research*, 7(1-2):55-66, 1980.

[14] P. Senecal, K., E. Pomraning, K. Richards. "Multi-Mode Genetic Algorithm Optimization of Combustion Chamber Geometry for Low Emissions". *SAE Paper* 2002-01-0958, 2002.

[15] P. Senecal, K., R. Reitz, D. “Simultaneous Reduction of Engine Emissions and Fuel Consumption using Genetic Algorithms and Multidimensional Spray and Combustion Modeling”. *SAE Paper*, 2000-01-1890, 2000.

[16] D. Wickman, D., P. Senecal, K., R. Reitz, D. “Diesel Engine Combustion Chamber Optimization Using Genetic Algorithms and Multi-Dimensional Spray and Combustion Modeling”. *SAE Paper*, 2001-01-0547, 2001.

[17] D. Wickman, D., H. Yun, R. Reitz, D. “Split-Spray Piston Geometry Optimized for HSDI Diesel Engine Combustion”. *SAE Paper*, 2003-01-0348, 2003.

[18] E. Zitzler, L. Thiele. “Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach”. *IEEE Transactions on Evolutionary Computation*, 3(4), 257-271, 1999.

Author Biographies

Teresa Donateo, Ph.D. was born on 29th June 1974. She attended the School of Material Engineering at the University of Lecce (Italy) and graduated in 1999 with first class honors. In June 2003, she received a Ph.D. degree from the School in Innovative Materials and Technologies at the ISUFI of Lecce. In November 2001, she joined the Faculty of the University of Lecce as Assistant Professor. From September to December 2005 visited the Ohio State University at Columbus OH and worked with Prof. Rizzoni at the Center for Automotive Research (CAR)in the field of HCCI engines and hybrid vehicles powertrain. Her research interests are: Numerical simulation of Internal Combustion Engines, Modeling of ignition and emissions mechanism of formation in HCCI engines, Evolutionary optimization techniques, Modeling and design of Hybrid Vehicles.

Prof. Domenico Laforgia

Full Professor of Energy Systems and Environment at the University of Lecce, Italy and Dean of the Engineering Faculty. Engineer at Ferrari, Maranello Modena, Italy from 1977 to 1978. Founder and senior partner at STIM Engineering Ldt. which provides consulting, engineering, and training services in industrial design, energy, ecology and environment, industrial property, technological innovation. Visiting fellow at the Mechanical and Aerospace Department at Princeton University (USA.) He collaborated with ELASIS, FIAT RESEARCH SYSTEM IN SOUTH ITALY, in the development of a highly innovative Diesel injection system, known today as Common Rail and manufactured by Bosch. His work is mainly on combustion of internal combustion engines and Diesel injection systems.

Prof. Giovanni Aloisio

Full Professor in Computer Engineering at the Engineering Faculty of the University of Lecce and Director of the ISUFI “Center for Advanced Computational Technologies” (ISUFI/CACT), University of Lecce. His research covers High Performance, Distributed and Grid Computing.

Prof Aloisio was a co-founder of the European Grid Forum (Egrid), now merged into the Global Grid Forum (GGF.) He is president of the SPACI consortium (www.spaci.it) , a research consortium on ICT among the University of Lecce, University of Calabria, Hewlett Packard Italy and Spaci l.t.d..

Silvia Mocavero, Ph.D student was born on 19th September of 1973.

She received a degree in Computer Engineering from the University of Lecce, Italy, on April 2002 and she is currently a PhD student in Innovative Materials and Technologies at the University of Lecce, Italy.

Since June 2002 to December 2004, she's involved in the WP10 of European Gridlab Project. Her research interests include high performance computing, distributed and grid computing, Grid Portals, Web and Grid Services, grid technologies for industrial problems solving, evolutionary optimization algorithms in grid environment.

Dr. Mocavero is currently a member of the ISUFI/Center for Advanced Computational Technologies and SPACI consortium.