INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **53**:455–469
Published online 6 July 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1288

Hierarchical distributed metamodel-assisted evolutionary algorithms in shape optimization

Marios K. Karakasis[‡], Dimitrios G. Koubogiannis[§,¶]
and Kyriakos C. Giannakoglou[*,†,‖]

National Technical University of Athens, School of Mechanical Engineering,
Laboratory of Thermal Turbomachines, Athens, Greece

SUMMARY

In aerodynamic shape optimization, the availability of multiple evaluation models of different precision and hence computational cost can be efficiently exploited in a hierarchical evolutionary algorithm. Thus, in this work the demes of a distributed evolutionary algorithm are ordered in levels, with each level employing a different flow analysis method, giving rise to a hierarchical distributed scheme. The arduous task of exploring the design space is undertaken by demes consisting the lower hierarchy level, which use a low-cost flow analysis tool, namely a viscous–inviscid flow interaction method. Promising solutions are directed towards the higher level, where these are further evolved based on a high precision/cost evaluation tool, viz. a Navier–Stokes equations solver. The final, optimal solution is obtained from the highest hierarchy level. At each level, metamodels, trained on-line on the outcome of evaluations with the level's analysis tool, are used. The role of metamodels is to allow a parsimonious use of computational resources by filtering the poorly performing individuals in each deme. The entire algorithm has been implemented so as to take advantage of a parallel computing system. The efficiency and effectiveness of the proposed hierarchical distributed evolutionary algorithm have been assessed in the design of a transonic isolated airfoil and a compressor cascade. Remarkable superiority over the conventional evolutionary algorithms has been monitored. Copyright © 2006 John Wiley & Sons, Ltd.

Received 7 November 2005; Revised 16 March 2006; Accepted 20 April 2006

KEY WORDS: distributed evolutionary algorithms; hierarchical optimization; metamodels

*Correspondence to: Kyriakos C. Giannakoglou, P.O. Box: 64069, Athens 157 10, Greece.
†E-mail: kgianna@central.ntua.gr
‡E-mail: mkk@mail.ntua.gr
§E-mail: dkoumbog@central.ntua.gr
¶Assistant Professor, Technological Educational Institute of Athens.
‖Associate Professor.

Contract/grant sponsor: Dassault Aviation
Contract/grant sponsor: A.S. Onassis Public Benefit Foundation

1. INTRODUCTION

Evolutionary algrorithms (EAs) are capable of providing a way to locate global extrema in difficult multimodal optimization problems. However, in their conventional form this is achieved after a high number of calls to the objective function has been performed. When the evaluation of the objective function requires the numerical computation of fluid flow fields, especially by solving the Navier–Stokes equations, the computational cost becomes prohibitive for industrial use.

The inherent ability of EAs to concurrently evaluate their population members can be exploited in order to reduce the wall-clock time of the optimization process, provided a parallel computing system is available. In addition, the search for the optimal solution becomes more effective, if distributed EAs (DEAs) are employed. In DEAs, semi-isolated demes evolve concurrently and regularly exchange information among each other via properly scheduled migrations [1–3].

A step forward in the direction of reducing the computational cost is to substitute approximate models (metamodels) for the costly objective functions—which will be referred to as the exact evaluation models, so as to make a clear distinction from the metamodels—as frequently as possible during the evolution. Polynomial-based response surfaces [4, 5], statistical methods [6, 7], artificial neural networks [8], including back-propagating multi-layer perceptrons [9, 10], radial-basis function networks (RBFNs) [11, 12] and support vector machines [13], have proved to be effective metamodels. Even though several comparisons of different metamodels are available [11, 14–16], a clear conclusion is difficult to draw. This is due to several parameters that affect their performance; among them, the most important are the number of design variables, the availability of information to build the metamodel with and the noise residing in it.

A major distinction concerning the EA-metamodel combination is whether the construction of the latter takes place off- or on-line with respect to the evolution. In the first case, a global metamodel is constructed before the EA is launched and is exclusively used to evaluate the candidate solutions generated by the EA [9, 17]. The 'optimal' solution may be fed back to increase the metamodel's resolution in the search space areas where the EA converges [10, 18–21]. A closer interaction between the EA and the metamodel is obtained when the latter is constructed or updated during the evolution. The metamodel construction is based on the outcome of previous evaluations using the exact objective function. Either entire generations are regularly evaluated using the exact model [7, 22], in order to control and improve the quality of the metamodel, or selected individuals in each generation are exactly re-evaluated [23–26].

The present authors have proposed the on-line use of local metamodels as a filter to single out the most promising individuals within each generation, through the so-called inexact pre-evaluation (IPE) phase [27]. A dedicated local metamodel is constructed (trained) and then used to predict the fitness of each newly appearing individual. Only for the individuals with the highest fitness, as estimated by the metamodel, the exact objective function value needs to be computed. The replacement of the exact evaluation tool by low-cost metamodels for the majority of the population members in each generation reduces the total CPU cost of the resulting EA–IPE algorithm by roughly one order of magnitude.

The metamodels and the exact model, combined as described above, can be perceived as forming a two-level evaluation hierarchy, where only promising solutions propagate from the lower—associated with the metamodel—to the higher and more precise analysis level. This hierarchy can be further extended in cases where a physical process can be described by more than one analysis models of different precision and consequently of different computing cost. This is the case of non-(massively) separated flows in shape design problems. A viscous–inviscid interaction (V–II) model (e.g. an integral boundary-layer method coupled with an Euler equation solver for the external flow through a suitable interaction scheme) gives reasonably accurate results at only a fraction of time required for solving the Navier–Stokes (N–S) equations. For the final design, however, one cannot trust but the N–S analysis. The purpose of this work is to combine multiple analysis tools, when available, in a hierarchical, distributed optimization scheme.

In the proposed method, multiple EAs, each employing IPE to filter their proper individuals, are combined to form the demes of a distributed EA, henceforth denoted by D(EA–IPE) [28]. Each D(EA–IPE) forms an optimization level. Here, the lower level employs a V–II analysis method as its 'exact' model. The higher one employs a N–S equation solver. The two levels regularly exchange their best individuals, with the lower level exploring the design space and the higher one mainly searching in the promising regions indicated by the former. The resulting hierarchical distributed metamodel-assisted EA, HD(EA–IPE), markedly outperforms, in both efficiency and effectiveness, even the EA–IPE scheme that uses exclusively the N–S equation solver.

A similar multi-level approach, though considerably simpler in implementation, has also been presented in Reference [29]. In a completely different domain of application (flywheel design optimization using a finite-element (FEM) evaluation model), Eby *et al.* [30] presents a method with quite a few similarities to the present one. In Reference [30], an island genetic algorithm is proposed, where demes are associated with different levels of resolution (plane stress or 3D FEM or the same FEM with different number of degrees of freedom), hybridized also with local search operators (simulated annealing and threshold accepting). An important difference, however, between References [29, 30] and the present method is the additional use of metamodels, according to the IPE concept, by the latter.

With respect to the method proposed in Reference [30] as well as other published works in aerodynamics [31, 32], it should become clear that the hybridization with hill-climbing, i.e. any optimization method, which could further improve the current best individual at each generation, is beyond the scope of this paper. It is expected that such a hybridization may further improve the demonstrated performance of HD(EA–IPE) but the latter is where this paper focuses on.

The hierarchical optimization algorithm is described in Section 2. The evaluation models and metamodels used are presented in brief in Section 3. Finally, in Section 4, the proposed hierarchical, distributed metamodel-assisted EA is applied to the design of an isolated transonic airfoil and a compressor cascade.

2. HIERARCHICAL DISTRIBUTED OPTIMIZATION ALGORITHM

The proposed hierarchical optimization algorithm is formed by multiple levels of distributed EAs. More information about the latter can be found in Reference [28]. At each level, the flow quantities of interest are computed with a different tool, whose accuracy and computational cost increase from the lowest to the highest level. The role of the lower levels is to explore the design space with the minimum possible CPU cost and guide the higher ones to scrutinize particular areas by modelling additional flow features, which cannot be described by the lower-level tools. The optimal solution is obtained from the highest level, which employs the most accurate evaluation tool. Apparently, at each level a single deme may equally be used.

Within each level, local metamodels are trained on the outcomes of evaluations carried out with the level's flow analysis tool (Figure 1). Metamodels are used for the IPE of the population members, to predict the performance of new individuals appearing in each deme. The approximate fitness

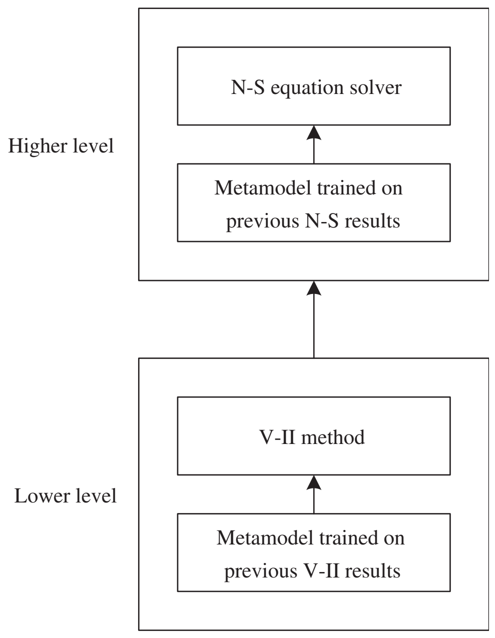

Higher level
N-S equation solver
Metamodel trained on previous N-S results
Lower level
V-II method
Metamodel trained on previous V-II results

Figure 1. Models used at each of a two-level hierarchical optimization method.

values provided by the metamodels allow the selection of the most well performing individuals, to be re-evaluated with the level's exact flow analysis method. Therefore, the main task of the optimization algorithm is the EA–IPE scheme [27, 33] (Figure 2), which is applied to each deme. For the lower levels, the use of metamodels may be redundant, if the cost of training the metamodel is comparable to that of performing the evaluation with the level's software. This is not the case, however, for the higher levels, where the cost of the flow analysis is expected to be high.

The distributed scheme within each level acts as if it were a normal D(EA–IPE) algorithm [28] and the most important addition to HD(EA–IPE) is the inter-level communication. A level agent coordinates the communication between demes, gathers the elite individuals from all of them and distributes the individuals imported from adjacent levels to the demes. In particular, after a predefined number of migrations between the level's demes has been performed, the best individuals from all of them are gathered and made available to the adjacent levels. The adjacent levels are requested to provide their own elite members. For each level of the HD(EA–IPE) to incorporate immigrants originating from its adjacent ones, these individuals must be evaluated with the level's proper flow analysis tool. It is important not to merge objective function values computed through different flow evaluation tools, otherwise the selection process, which is crucial to evolution, can easily be misled. An incoming individual from a lower level replaces an existing one in a deme, only if it performs better than that. If a level consecutively fails to provide its higher one with useful individuals, its evolution is terminated as well as that of its lower levels. As soon as the inter-level communication has been accomplished, the demes' evolution resumes. The main communication steps are given in the block diagram of Figure 3.

The HD(EA–IPE) algorithm is implemented so as to take advantage of a multi-processor system. The requests for evaluation with a specific flow analysis tool are addressed from all levels to a single

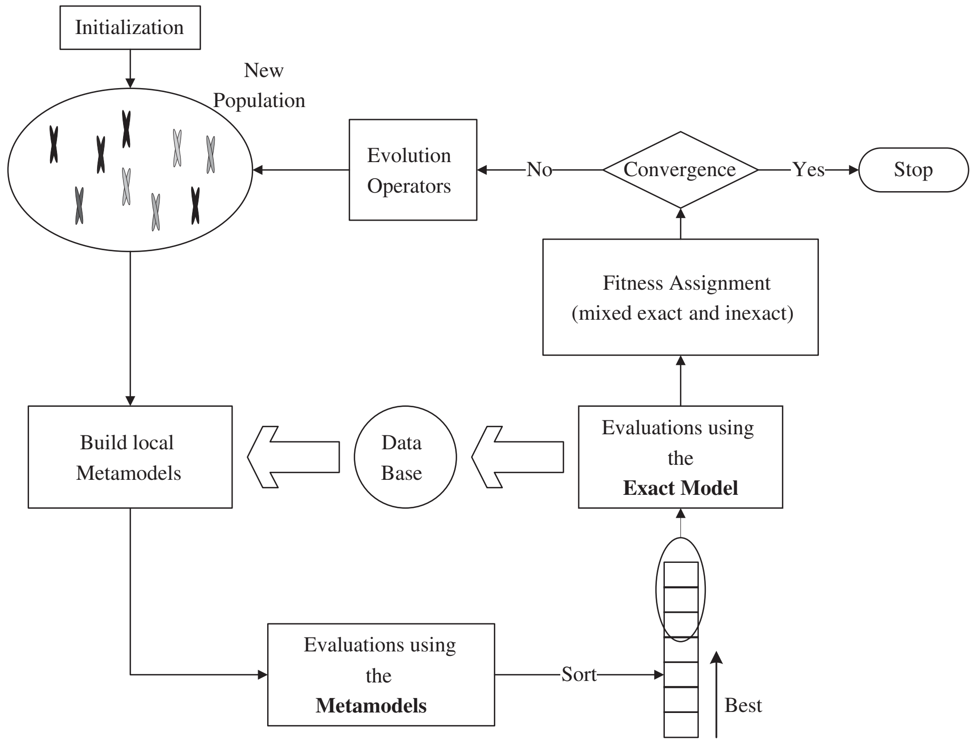

Initialization
New Population
Evolution Operators
No
Convergence
Yes
Stop
Fitness Assignment (mixed exact and inexact)
Build local Metamodels
Data Base
Evaluations using the Exact Model
Evaluations using the Metamodels
Sort
Best

Figure 2. Application of Inexact Pre-Evaluation (IPE) in a single-level, single-deme Evolutionary Algorithm. To distinguish between the evaluation model and the corresponding metamodel, the terms *exact* and *inexact* are used, respectively.

evaluation server, which assigns accordingly the available computing resources. The evaluation results are stored in appropriate databases (DBs), one for each level. The DB entries of each level are used for subsequent metamodel constructions (or updates). The operation of HD(EA–IPE) for two hierarchy levels is schematically depicted in Figure 4. Two important practicalities of the proposed HD(EA–IPE) are discussed below.

IPE for the immigrating individuals: In general, the number of individuals that emigrate from a lower to a higher level can be large and they have necessarily to be evaluated with the analysis tool of the host level. Assuming that the maximum number of calls to the high-fidelity flow analysis code is set by the user, this extra evaluation cost deprives the highest level of a considerable number of evolution generations. To alleviate this shortcoming, IPE can also be applied to the incoming individuals. They all get an approximate objective function value through metamodels. These values are sorted and only the best immigrants are re-evaluated with the host level's analysis tool.

Handling of constraints: The treatment of constraints in hierarchical optimization is particularly important. In this work, the constraints are imposed via penalty multipliers, which increase the original objective function value (assuming a minimization problem). There are two kinds of constraints: those being independent of the flow (e.g. geometrical constraints) and those that depend

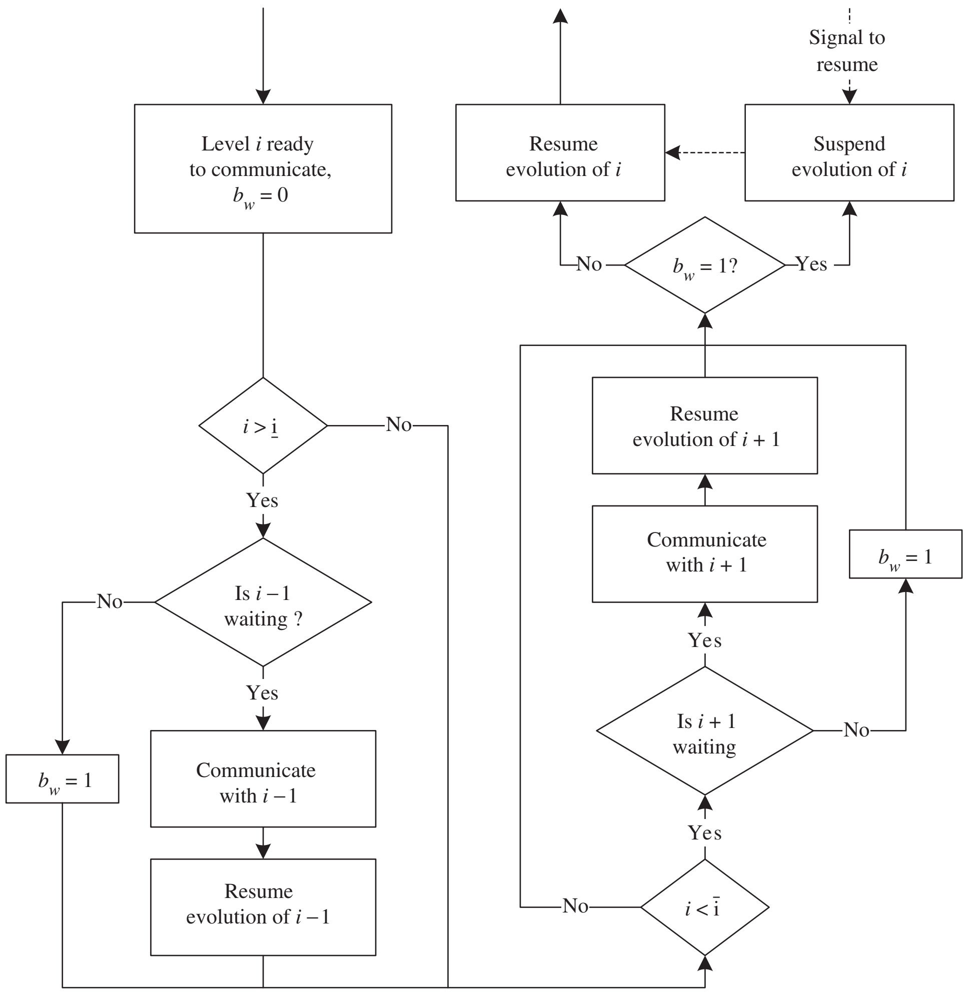


Figure 3. Inter-level communication in a HDEA. Every level i communicates with its lower $i - 1$ and upper one $i + 1$ in the hierarchy. The lowest level is denoted by $\underline{i}$, while the highest by $\bar{i}$; apparently, $\underline{i} \leqslant i \leqslant \bar{i}$. The boolean variable b_w controls if i has to wait any of its adjacent levels to communicate with. The signal to resume evolution comes from them, upon the end of communication.

on it (e.g. the flow exit angle from a compressor cascade or the lift coefficient of an isolated airfoil). For the former the constraint value is the same in all levels; for the latter, however, an individual that does not violate a constraint at a lower level may do so at a higher one. This happens quite often during the early stages of the optimization process. Even though an incoming individual violates a constraint, it certainly carries useful information. Therefore, at the beginning of the optimization process a host level should be indulgent in the constraint violation for the individuals emigrating from a lower level, either by ignoring the violation or by mildly penalizing it. This indulgence, however, should decrease as the evolution advances.

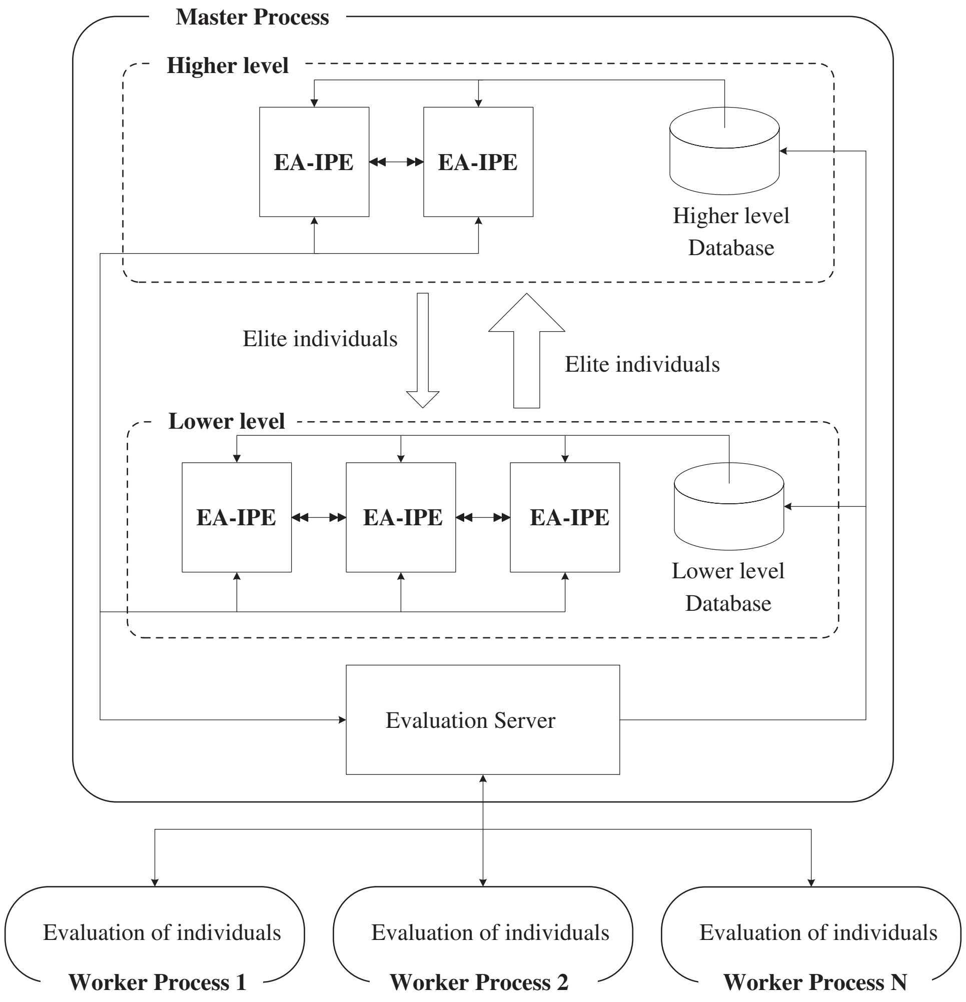


Figure 4. A two-level Hierarchical Distributed Evolutionary Algorithm implemented for parallel computing systems. Each level has its own database and metamodels, which approximate the objective function as modelled by the level's 'exact' analysis tool. The metamodels are used in the context of IPE (Figure 2).

3. EVALUATION MODELS

In the applications that follow, the hierarchical optimization algorithm comprises two levels. In the low level, a V–II analysis method is used as the exact evaluation tool. In the high level, a N–S equation solver is employed. In both levels RBFNs are used as metamodels. A brief description of these evaluation models is given below.

Viscous–inviscid flow interaction method: This method utilizes the MSES/MISES code for external/internal aerodynamics. The underlying theory is thoroughly presented in References [34, 35]. The Euler equations are discretized on a conservative streamline grid and are coupled to a two-equation integral boundary-layer formulation. Transition prediction is incorporated into the viscous flow model. The entire discrete equation set is solved as a fully coupled nonlinear system of equations, resulting in a particularly efficient code. Strong viscous–inviscid interactions and limited flow separation are successfully dealt with.

Navier–Stokes equation solver: The N–S equation solver is based on a time-marching, vertex-centred, finite volume method on unstructured grids with triangular elements. The two-dimensional N–S equations or the quasi-three-dimensional ones with variable streamtube thickness for turbomachinery cascades are solved. The convective terms of the equations are discretized by means of an upwind scheme (Roe flux difference splitting), while the computation of the diffusive ones is based on the assumption of a linear distribution of the primitive variables within each triangular element. Second-order accuracy in space is obtained through MUSCL extrapolation, while monotonicity is guaranteed by means of appropriate limiters. Turbulence modelling is accomplished by means of one- and two-equation models. More details on the description of the numerical method and the implementation of turbulence models can be found in Reference [36].

Radial-basis function networks: A RBFN uses a single hidden-neuron layer to perform a two-stage mapping: a nonlinear one from the design space to the hidden layer's space and a second linear one to the objective space. With appropriate selection of the RBF centres, the simplicity of its architecture reduces its training to the solution of a linear algebraic system of equations or a linear least-squares problem [8, 37]. RBFNs possess valuable properties for function approximation [38] and, in the context of IPE, are used as local metamodels trained on a small subset of the DB entries—those being adjacent to each new individual. This makes their training cost negligible compared even to the cost of the V–II method. The training patterns can either be interpolated or approximated, depending on the required network's generalization capability [33].

4. APPLICATIONS

The following aerodynamic applications intend to compare the performance of the hierarchical optimization algorithm not only with a conventional EA (single-level, single-deme) based exclusively on the N–S solver but also with an EA assisted by metamodels to reduce the number of calls to the N–S solver. Performance is measured by monitoring the fitness of the best solution at the end of each high-level generation in terms of the total computational cost. The cost unit stands for the CPU cost of a call to the software that solves the N–S equations to compute the objective function value. It should be noted that, in all applications, the SST variant of the k–ω turbulence model has been used along with the Venkatakrishnan limiter to enforce monotonicity in the second-order accurate Roe scheme of the N–S solver. The CPU cost of the calls to the V–II code, measured in cost units, is also taken into account. The RBFN training cost is negligible compared to the aforementioned and, thus, ignored.

The airfoil shapes are parameterized with two NURBS curves, separately for the pressure (PS) and suction sides (SS). For all test-cases, geometrical and aerodynamic constraints are imposed via penalty multipliers. The same geometrical constraints apply to both test-cases and are concerned with: (a) the maximum airfoil thickness, (b) the airfoil thickness at 80% of the chord, (c) the curvature of PS/SS at the leading edge (LE), (d) the angle between the first PS/SS control points and the LE, (e) the angle between the first PS control point and the chord. Constraints (c)–(e) aim at ensuring acceptable performance at off-design flow conditions. Robust design methods would allow to dispense with them but this is out of the scope of the present paper. Due allowance must be

Table I. A well performing configuration of the Hierarchical Distributed Metamodel-Assisted EA, common to all applications. In a generalized (μ, λ)-EA, μ and λ stand for the parent and offspring population sizes, respectively.

	High level (N–S solver)	Low level (V–II method)
Intra-level communication $(D(EA-IPE))$		
Number of demes	2	3
Size of each deme $(\mu=\lambda=)$	40	30
Migration frequency (generations)	4	4
Migration rate (individuals)	2	2
Exact re-evaluations after IPE	10%	10%
Inter-level communication		
Migration frequency (total generations)	8	120
Elite individuals imported for the first time	30	6
Elite individuals imported otherwise	10	6
Exactly evaluated immigrants	50%	100%

made for the aerodynamic constraints, which depend on the outcome of the analysis software. At the beginning of evolution and until about half of the projected maximum number of generations, the high level is indulgent in them for the individuals coming from the lower level.

The number of elite individuals imported to the N–S equation solving level for the first time is set to a high value in order to quickly take advantage of the progress made at the low level. The basic HD(EA–IPE) configuration, common to all applications, is shown in Table I. The tabulated values stand for a well performing configuration for aerodynamic shape optimization problems. It is by no means unique but it proved to be adequate, even with some modifications, in a series of similar studies undertaken by the authors. It is reasonable that the user's experience on tuning single-deme, single-level EAs (i.e. setting the parent and offspring population sizes, etc.) can be transferred to HDEAs.

4.1. Design of a transonic isolated airfoil

The first case aims at the design of an isolated airfoil with minimum drag, operating at transonic flow conditions and yielding a prescribed lift coefficient. The flow conditions and the lift coefficient are (after Reference [39]): $Re=6.5\times10^6$, $M_\infty=0.73$, $\alpha_\infty=3.19^\circ$ and $c_l=0.764$. At these flow conditions, the drag coefficient of the reference airfoil (RAE 2822) is $c_d=0.0128$, as computed by the N–S code. The targeted c_l is set as aerodynamic constraint. The maximum airfoil thickness should exceed 11% of the chord.

The convergence history is plotted in Figure 5. The superiority of the HD(EA–IPE) scheme is clear over conventional and single-level, single-deme metamodel-assisted EAs. The solution obtained with a CPU cost equivalent to approximately 1000 calls to the N–S equation solver is illustrated in Figure 6. In this problem, the CPU cost of evaluating a candidate solution with the V–II method is $\sim 1/10$ of that of the N–S solver.

The conventional EA requires approximately 6160 min of CPU time (on an Intel Pentium 4 processor) to reduce c_d by 22.5%. With HD(EA–IPE) the same reduction in c_d is achieved in 1540 min and the drag of the optimal airfoil is reduced by 34.2% compared to the reference one.

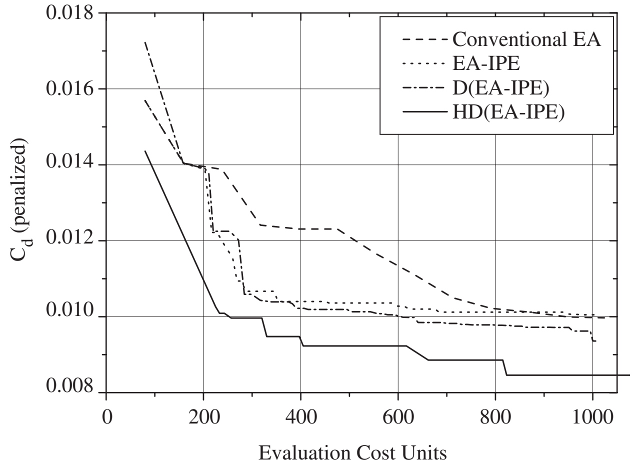


Figure 5. Transonic airfoil: Convergence history. The evaluation cost unit is the equivalent of the CPU cost of solving the Navier–Stokes equations. The non-hierarchical variants employ exclusively a Navier–Stokes equation solver.

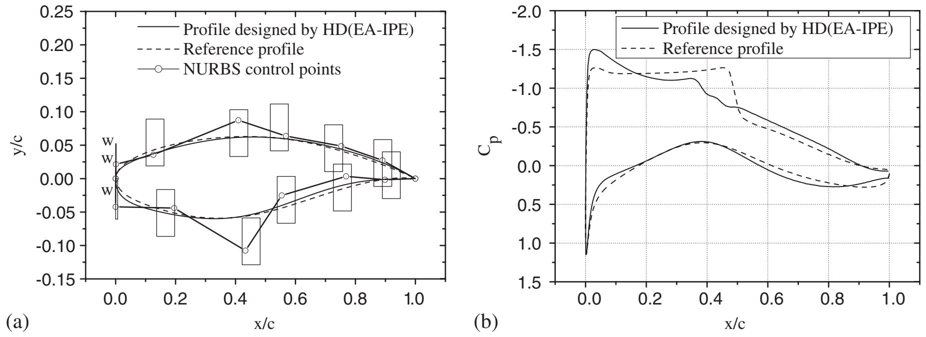
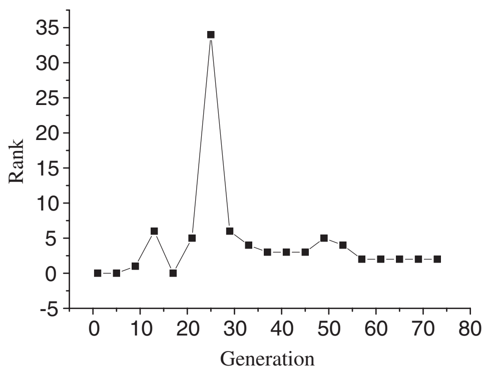
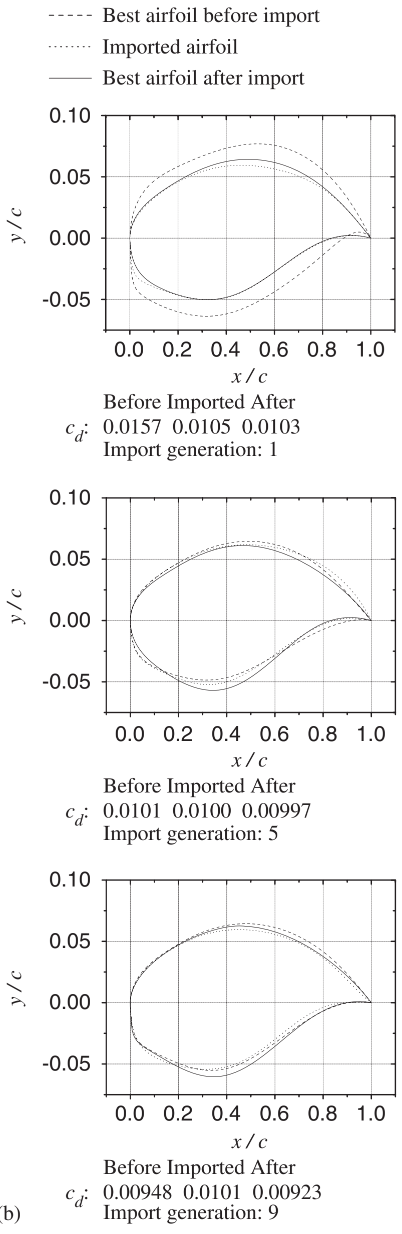


Figure 6. Transonic airfoil: The optimal shape obtained by HD(EA–IPE) at a cost equivalent to 1000 calls to the N–S equation solver compared to the reference airfoil: (a) optimal and reference airfoil profiles, NURBS control points and their bound boxes. The control points, whose weights were free to vary, are marked with a 'w'; and (b) C_p distribution of the optimal ($c_d = 0.00846$, $c_l = 0.765$) and reference ($c_d = 0.0128$, $c_l = 0.764$) profiles.

The contribution of the low level can be quantified by the rank the best individual immigrating to the high level obtains in its population, after having been evaluated with the N–S solver. This rank is plotted in terms of the number of generations, at which the inter-level communication occurs, in Figure 7(a) for the first deme. The contribution is more important (e.g. rank 0 means that the immigrant is better than the best in the host population) at the early generations. As the evolution advances, this contribution decreases, since the additional flow phenomena captured by the N–S solver yield an optimal solution different than that of the low level, based on the V–II solver.



(a)



(b)

Figure 7. Transonic airfoil: The contribution of the low level, as monitored by the rank of the best immigrating individual to the high level with respect to the host population, and the impact of imported individuals on the high-level solution: (a) rank 0 signifies that the fittest imported individual is better than the host population's best. The lower the rank, the higher the contribution is; and (b) best imported individuals from the high level and their impact on its fittest solution a few generations later. The tabulated c_d values are penalized with the violated constraints.

4.2. Losses minimization of a compressor cascade

The second case aims at the design of a controlled diffusion compressor cascade, operating at $M_1 = 0.618$, $Re = 8.41 \times 10^5$, $\alpha_1 = 47.0°$ (after Reference [40]). The objective is to minimize the mass-averaged total pressure losses between inlet and outlet, $\omega = (p_{01} - p_{02})/(p_{01} - p_1)$, while preserving flow turning (note that the flow exit angle is $\alpha_2 = 20.2°$). The losses of the reference cascade airfoil, computed by the N–S code, are $\omega = 0.0187$. The outlet flow angle is imposed as an aerodynamic constraint. The minimum maximum airfoil thickness is constrained to 10% of the chord.

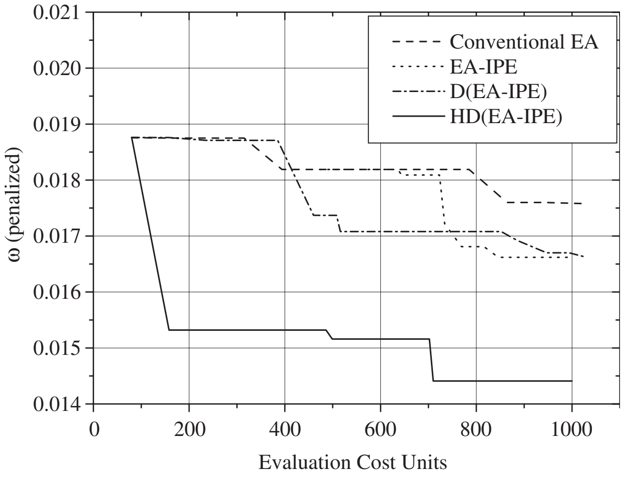


Figure 8. Compressor cascade: Convergence history. The evaluation cost unit is the equivalent of the cost of solving the Navier–Stokes equations. The non-hierarchical variants employ exclusively a Navier–Stokes equation solver.

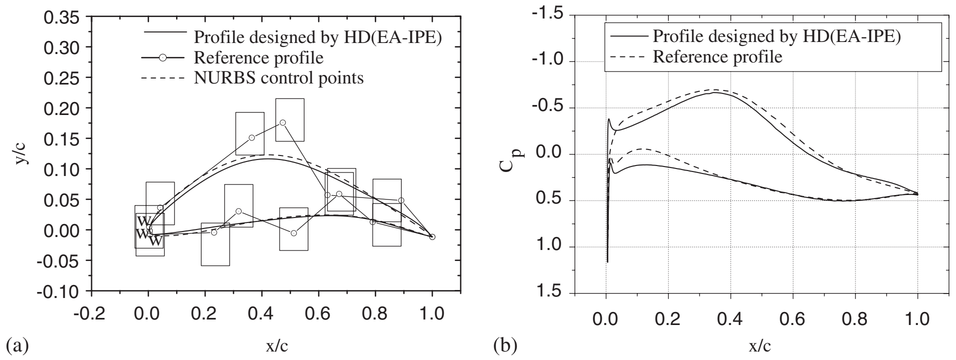


Figure 9. Compressor cascade: The design obtained by HD(EA–IPE) at a cost equivalent to 1000 calls to the N–S equation solver compared to the reference airfoil cascade: (a) airfoil profile, NURBS control points and their bound boxes. The control points, whose weights were free to vary, are marked with a 'w'; and (b) C_p distribution of the optimal design ($\omega = 0.0144$) superimposed on the reference one ($\omega = 0.0187$).

The convergence history is plotted in Figure 8. In this case, the superiority of the HD(EA–IPE) algorithm is even more pronounced since there is a better agreement between the N–S solver and the V–II method results. The solution obtained with a computational cost equivalent to approximately 1000 calls to the N–S equation solver is illustrated in Figure 9. Here, the CPU cost of evaluating a candidate solution with the V–II method is $\sim 1/80$ of the N–S cost.

The conventional EA requires 9200 CPU min to reduce ω by 5.9%. With HD(EA–IPE) a better solution is obtained after only 1420 min and the total pressure losses of the final airfoil cascade are reduced by 23.0%. A considerable difficulty in this test case is the particularly high number of failed evaluations. A failure may be due to the inability to automatically generate the grid because of a non-realistic geometry, inability of the flow analysis code to converge with the user-defined CFL number after a prescribed number of iterations, the need of a different limiter due to local grid peculiarities, etc.

5. CONCLUSIONS

When multiple models of different precision and hence computational cost are available, the organization of the demes of a DEA in hierarchy levels is beneficial in aerodynamic shape optimization problems. In this work, a bi-level hierarchical DEA, employing a viscous–inviscid flow interaction method in the low and a Navier–Stokes equation solver in the high level, has been applied to two aerodynamic shape optimization problems. By assigning the computationally hard task of exploring the design space to the lower level, considerable saving in computational resources is achieved. By allocating the saved resources to the high level, so as to thoroughly evaluate and further evolve the promising solutions coming from the low level, optimal solutions are obtained with a considerable economy in CPU cost.

The use of metamodels within each level, being on-line trained on the outcome of evaluations using the level's exact flow analysis tool, contributes to noticeable additional reduction in computational cost. Through local metamodels, poorly performing individuals in each deme are filtered out, without being further evaluated.

A marked improvement in both efficiency and effectiveness has been assessed in the design of a transonic airfoil and a compressor cascade under flow and geometrical constraints. The proposed hierarchical algorithm outperforms any other variant—including the single-level, single-deme metamodel-assisted EA—that solely uses the Navier–Stokes equation solver.

ACKNOWLEDGEMENTS

This research was part of a project funded by Dassault Aviation with Prof. J. Périaux, Pôle Scientifique Dassault Aviation/UPMC, as Project Monitor. The first author was supported by an A.S. Onassis Public Benefit Foundation Scholarship.

REFERENCES

1. Tanese R. Distributed genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, Schaffer JD (ed.). Morgan Kaufmann Publishers Inc.: San Francisco, CA, U.S.A., 1989; 434–439.
2. Doorly DJ, Peiró J, Spooner S. Design optimisation using distributed evolutionary methods. *37th Aerospace Sciences Meeting and Exhibit*, *AIAA-1999-111*, Reno, NV, U.S.A., January 1999.

3. Eklund SE. A massively parallel architecture for distributed genetic algorithms. *Parallel Computing* 2004; **30**(5–6):647–676.
4. Myers RH, Montgomery DC. *Response Surface Methodology*: *Process and Product Optimization Using Designed Experiments* (2nd edn). Probability and Statistics. Wiley: New York, U.S.A., 2002.
5. Engelund WC, Stanley DO, Lepsch RA, McMillian MM, Unal R. Aerodynamic configuration design using response surface methodology analysis. *Aircraft Design*, *Systems and Operations Meeting*, *AIAA-1993-3967*, Monterey, CA, U.S.A., August 1993; 11–13.
6. Santner TJ, Williams BJ, Notz WI. *The Design and Analysis of Computer Experiments*. *Statistics*. Springer: New York, U.S.A., 2003.
7. Ratle A. Optimal sampling strategies for learning a fitness model. *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 3. Washington DC, U.S.A., July 1999; 2078–2085.
8. Haykin S. *Neural Networks*: *A Comprehensive Foundation* (2nd edn). Prentice-Hall: Englewood Cliffs, NJ, U.S.A., 1999.
9. Greenman RM, Roth KR. Minimizing computational data requirements for multi-element airfoils using neural networks. *37th AIAA Aerospace Sciences Meeting and Exhibit*, *AIAA-1999-0258*, Reno, NV, U.S.A., January 1999.
10. Pierret S, Van den Braembussche RA. Turbomachinery blade design using a Navier–Stokes solver and artificial neural network (*ASME Paper 99-GT-4*). *Journal of Turbomachinery* 1999; **121**(2):326–332.
11. Papila N, Shyy W, Fitz-Coy N, Haftka RT. Assessment of neural net and polynomial-based techniques for aerodynamic applications. *17th AIAA Applied Aerodynamics Conference*, *AIAA-1999-3167*, Norfolk, VA, U.S.A., 1999.
12. Giotis AP, Giannakoglou KC, Périaux J. A reduced-cost multi-objective optimization method based on the Pareto front technique, neural networks and PVM. In *ECCOMAS 2000*, *Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering*, Oñate E, Bugeda G, Suárez B (eds). CIMNE: Barcelona, Spain, 2000.
13. Nakayama H, Arakawa M, Washino K. Using support vector machines in optimization for black-box objective functions. *Proceedings of the International Joint Conference on Neural Networks*, July 2003, vol. 2. 2003; 1617–1622.
14. Carpenter WC, Barthelemy J-FM. A comparison of polynomial approximations and artificial neural nets as response surfaces. *Structural and Multidisciplinary Optimization* 1993; **5**(3):166–174.
15. Simpson TW, Mauery TM, Korte JJ, Mistree F. Comparison of response surface and kriging models for multidisciplinary design optimization. *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, *AIAA-1998-4758*. St. Louis, MO, U.S.A., September 1998.
16. Shyy W, Papila N, Vaidyanathan R, Tucker K. Global design optimization for aerodynamics and rocket propulsion components. *Progress in Aerospace Sciences* 2001; **37**(1):59–118.
17. Papadrakakis M, Lagaros ND, Tsompanakis Y. Structural optimization using evolution strategies and neural networks. *Computer Methods in Applied Mechanics and Engineering* 1998; **156**(1–4):309–333.
18. Bull L. On model-based evolutionary computation. *Soft Computing—A Fusion of Foundations*, *Methodologies and Applications* 1999; **3**(2):76–82.
19. Nakayama H, Inoue K, Yoshimori Y. Approximate optimization using computational intelligence and its application to reinforcement of cable-stayed bridges. In *ECCOMAS 2004*, *Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering*, Neittaanmäki P, Rossi T, Korotov S, Oñate E, Périaux J, Knörzer D (eds). Jÿvaskÿla, Finland, July 2004.
20. Lagaros ND, Charmpis DC, Papadrakakis M. An adaptive neural network strategy for improving the computational performance of evolutionary structural optimization. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**(30–33):3374–3393.
21. Büche D, Schraudolph N, Koumoutsakos P. Accelerating evolutionary algorithms with Gaussian process fitness function models. *IEEE Transactions on Systems*, *Man*, *and Cybernetics—Part C*: *Applications and Reviews* 2005; **35**(2):183–194.
22. Jin Y, Olhofer M, Sendhoff B. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation* 2002; **6**(5):481–494.
23. Giannakoglou KC. Designing turbomachinery blades using evolutionary methods. *ASME Turbo Expo'99*, *ASME Paper 99-GT-181*. Indianapolis, U.S.A., June 1999.
24. Ong YS, Lum KY, Nair PB, Shi DM, Zhang ZK. Global convergence of unconstrained and bound constrained surrogate-assisted evolutionary search in aerodynamic shape design. *Proceedings of the 2003 Congress on Evolutionary Computation* (*CEC'03*), vol. 3. Canberra, Australia, December 2003; 1856–1863.

25. Ulmer H, Streichert F, Zell A. Evolution strategies assisted by Gaussian processes with improved pre-selection criterion. *Proceedings of the 2003 Congress on Evolutionary Computation* (*CEC'03*), vol. 1. Canberra, Australia, 2003; 692–699.
26. Branke J, Schmidt C. Faster convergence by means of fitness estimation. *Soft Computing—A Fusion of Foundations, Methodologies and Applications* 2005; **9**(1):13–20.
27. Giannakoglou KC, Giotis AP, Karakasis MK. Low-cost genetic optimization based on inexact pre-evaluations and the sensitivity analysis of design parameters. *Journal of Inverse Problems in Engineering* 2001; **9**(4):389–412.
28. Karakasis MK, Giotis AP, Giannakoglou KC. Inexact information aided, low-cost, distributed genetic algorithms for aerodynamic shape optimization. *International Journal for Numerical Methods in Fluids* 2003; **43**(10–11):1149–1166.
29. Sefrioui M, Périaux J. A hierarchical genetic algorithm using multiple models for optimization. In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature—PPSN VI*, Schoenauer M, Deb K, Rudolph G, Yao X, Lutton E, Guervós JJM, Schwefel H-P (eds), Lecture Notes in Computer Science, vol. 1917. Springer: New York, Paris, France, 2000; 879–888.
30. Eby D, Averill RC, Punch III WF, Goodman ED. Evaluation of injection island GA performance on flywheel design optimization. *Proceedings of the 3rd Conference on Adaptive Computing in Design and Manufacturing*, Plymouth, U.K., 1998; 121–136.
31. Dulikravich GS. Shape inverse design and optimization for three-dimensional aerodynamics. *33rd Aerospace Sciences Meeting and Exhibit*, *AIAA-1995-695*, Reno, NV, U.S.A., January 1995.
32. Foster NF, Dulikravich GS. Three-dimensional aerodynamic shape optimisation using genetic and gradient search algorithms. *Journal of Spacecraft and Rockets* 1997; **34**(1):36–42.
33. Karakasis MK, Giannakoglou KC. On the use of surrogate evaluation models in multi-objective evolutionary algorithms. In *ECCOMAS 2004*, *Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering*, Neittaanmäki P, Rossi T, Korotov S, Oñate E, Périaux J, Knörzer D (eds). Jÿvaskÿla, Finland, July 2004.
34. Giles MB, Drela M. A two-dimensional transonic aerodynamic design method. *AIAA Journal* 1987; **25**(9):1199–1206.
35. Drela M, Giles MB. Viscous-inviscid analysis of transonic and low Reynolds number airfoils. *AIAA Journal* 1987; **25**(10):1347–1355.
36. Koubogiannis DG, Athanassiadis AN, Giannakoglou KC. One- and two-equation turbulence models for the prediction of complex cascade flows using unstructured grids. *Computers & Fluids* 2003; **32**(3):403–430.
37. Poggio T, Girosi F. Networks for approximation and learning. *Proceedings of the IEEE* 1990; **78**(9):1481–1497.
38. Park J, Sandberg IW. Universal approximation using radial-basis-function networks. *Neural Computation* 1991; **3**(2):246–257.
39. Cook PH, McDonald MA, Firmin MC. Airfoil RAE 2822—pressure distributions and boundary layer and wake measurements. *AGARD-AR-138* (Experimental Data Base for Computer Program Assessment), May 1979.
40. Steinert W, Eisenberg B, Starken H. Design and testing of a controlled diffusion airfoil cascade for industrial axial flow compressor application. *Gas Turbine and Aeroengine Congress and Exposition*, *ASME Paper 90-GT-140*, Brussels, Belgium, June 1990.