Sub-population policies for a parallel multiobjective genetic algorithm with applications to wing design

D. Quagliarella, A. Vicini
C.I.R.A., Centro Italiano Ricerche Aerospaziali,
Via Maiorise – 81043 Capua (Italy)

ABSTRACT

In this work a parallel multi-objective genetic algorithm is presented. The population selection and mating phase is kept distinct from the population fitness evaluation loop, that is implemented in parallel. The population can be logically split in sub-populations which number does not depend on the number of processors available for computation. Different sub-population topologies and migration rates among sub-populations can be used. Applications are illustrated both for single and multi-objective mathematical test cases, and aerodynamic transonic wing design.

1. INTRODUCTION

Multiobjective optimization through genetic algorithms is currently applied to a growing set of aerospace design tasks, ranging from multipoint aerodynamic design to multidisciplinary optimization problems. However, the heavier constraint to a widespread application of such a technique is the high computational cost of the numerical programs used to evaluate the functions to be optimized.

Parallelization of genetic optimizers is one of the solution often proposed to this problem. To this purpose, various *ad hoc* algorithms have been developed for parallel machines, most of which take advantage from splitting the population into multiple sub-populations, showing results that are, on average, better than those of their serial, one-population based, counterparts [1, 2]. Sub-populations, in fact, may allow sometimes a better control on population diversity, reducing the appearance of premature convergence and genetic drift. However, although this approach can be easily and naturally implemented on parallel computers, it is not exclusive domain of such kind of machines.

This work describes a parallel genetic algorithm in which parallel evaluation of population elements and population splitting into sub-populations are kept conceptually and physically separated. Therefore, when the problem at hand can favorably be solved using different sub-populations evolving at the same time, this can be obtained independently of the number of available processing elements.

The application examples reported are related both to mathematical function optimization and aerodynamic wing design; for each of the cases considered, the possible advantages of adopting sub-populations are discussed.

2. GENETIC ALGORITHM WITH SUB-POPULATIONS

The algorithm here described, that will be called Virtual Subpopulation Genetic Algorithm (VSGA), is an extension of the multi-objective genetic algorithm described in [3]. The two main new features that distinguish this new algorithm with respect to the previous one are the *sub-population model* and the *parallel evaluation loop*. The first point will be described here, and the second in the following paragraph.

In the present context, a genetic algorithm is meant to follow a sub-population model when different sub-populations are let evolve at the same time with possible communication strategies among them.

The model adopted here distributes all the population elements on a single toroidal landscape [4], and the subpopulations are obtained introducing logical boundaries that define the domain assigned to each sub-population.

The selection scheme adopted is the random walk: for each element in the current generation, two random walks of assigned number of steps are carried out using its position as starting point; the non dominated individuals among those encountered in each of these random walks are selected as parents, and the starting element is replaced in the new generation by one of the offsprings obtained by their recombination.

The migration strategy, i.e. the way the sub-populations exchange individuals, is defined by letting the random walk path to cross the boundaries between subpopulations on the basis of a given probability function. In the presented applications this function is a simple switch that allows the boundary crossing with an assigned probability, equal for each boundary and chosen between 0 and 1, that will be called *flip rate*. Thus, a flip rate of 0 corresponds to not crossable boundaries, while a flip rate of 1 is equivalent to the lack of boundaries.

The Pareto front of the current generation is obtained extracting the non dominated individuals with respect to the whole population and the Pareto front of the previous generation. However, for each element belonging to the front, information is retained on the sub-population of provenance.

The elitism strategy is implemented by randomly selecting an assigned percentage of parents from the current set of non dominated solutions. In this case two possible mechanisms can be adopted:

1. the offspring is assigned to the same sub-population of the Pareto front parent selected;
2. the offspring is assigned to a sub-population chosen at ran-

dom.

The latter mechanism is, from another point ov view, another communication policy, thus, for example, if the elitism strategy is active and the flip rate is set equal to 0 only the elements belonging to the current front can migrate between sub-populations.

Fig. 1 reports an example of virtual subpopulation organization. For the sake of simplicity, the toroidal landscape las been represented as a rectangle, but the elements can pass through the external sides of the rectangle too.

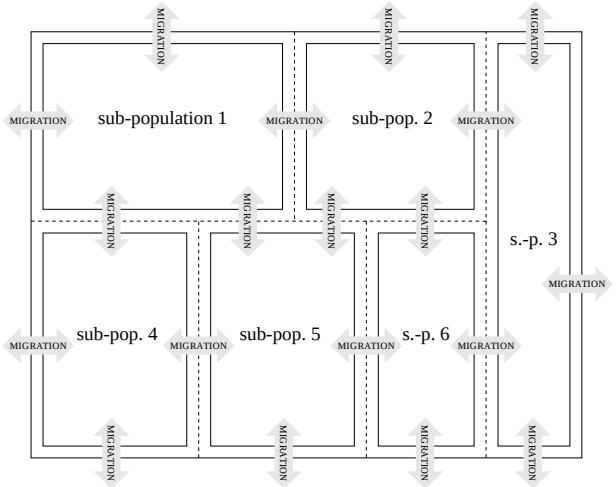

Figure 1: Virtual subpopulation organization

The evaluation of the offsprings obtained using the above procedure is postponed to a second phase of the computation, that is the only one that is actually carried out in parallel. VSGA is therefore able to take the maximum advantage from parallel computation even when different sub-population sizes are used at the same time.

3. PARALLEL EVALUATION LOOP

The genetic algorithm evaluates the new population members in a single loop that follows the recombination phase and that can be carried out in parallel. The parallel programming model adopted relies on shared memory multiprocessing and the parallelism is implemented at the process level.

The parallel machine adopted is a SGI POWER CHALLENGE system with 16 R-10000 processors. The parallel code has been implemented using the lightweight UNIX process primitives available on this machine [5].

The software is organized following the master-slave paradigm. Figure 2 shows the architecture of the parallel evaluation loop.

In the initialization phase that precedes the first execution of the evaluation loop, the master process creates a pool containing a number of processes, equal to the maximum number of processors available for the computation (NPROC). The child processes created are immediately put in a wait state, and they will remain in such a state until they receive a “go ahead” signal from the master to start the computation. This architecture has been

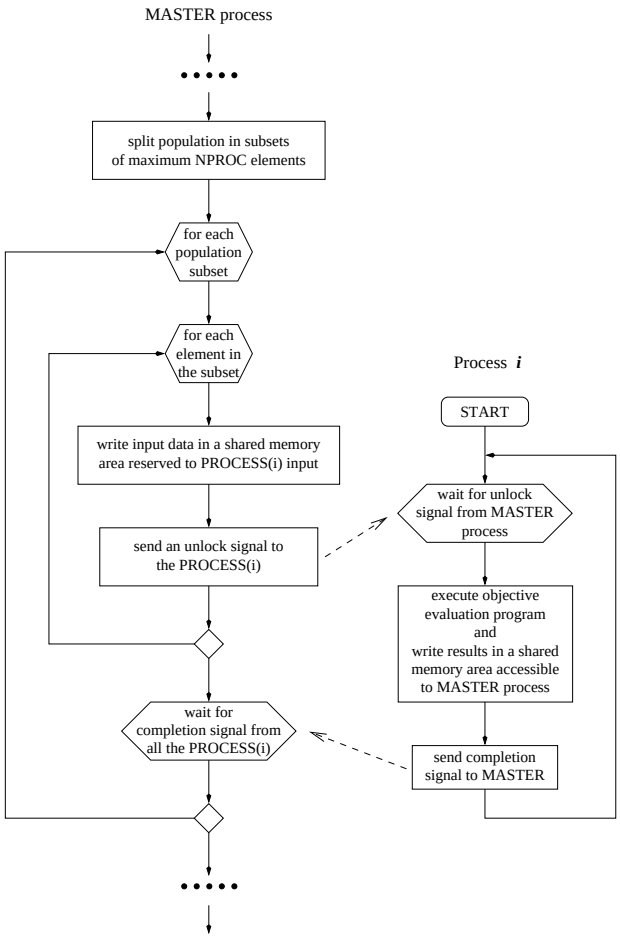

Figure 2: Parallel evaluation loop

chosen to avoid the inefficiency of creating a child process every time a computation is needed and killing it at the end.

When the master process enters the evaluation loop, it splits the population in subsets of maximum NPROC elements and then, for each subset, it copies the data relevant to a single computation in a shared memory area that is accessed by one of the child processes; afterwards, a signal is sent to the child, through a standard POSIX semaphore, so that the computation can begin. When the child terminates its computation, it copies the results in a memory region accessible to the master, but not overwritable by the other children. Thereafter, it sends a completion signal to the master (using another POSIX semaphore). The master waits for the completion of all child processes in a synchronization point. When this happens, the master passes to the next subset of population elements. The child processes are terminated at the very end of the program, when all the evaluation loops related to each generation have been completed.

It can be observed that this architecture is efficient only when each sub-process has an even computational charge. If this is not the case, the computation process can loose efficiency because the master has to wait at the synchronization point for the completion of the slowest process. Fortunately, this is not the case for the kind of application considered so that the efficiency decay can be neglected. However it is not difficult to modify the

master so that it can assign a new task to a child as soon as this has completed the previous one.

4. SINGLE-OBJECTIVE MATHEMATICAL PROBLEM

The sub-population algorithm has been applied to the well known Rastringin's function:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right] \tag{1}$$

with: $n = 16$; $x_i \in [-5.12, 5.12]$. The main characteristic of this function is that, within the assigned variation range for the variables, it is characterized by 11^n local minima, and by only one global minimum $f(\mathbf{0}) = 0$. It is therefore very easy for an optimizer to be trapped in a local minima, which makes this function a suitable candidate to investigate the robustness of an optimization procedure. The results reported are related to six different population sizes (64, 144, 256, 400, 576, 900), and for each population size 10 runs have been done with different starting populations; the maximum number of generations has been set to 2000, and the best minimum value for the objective function to 1e-6. The elitist strategy has been adopted, and the variables have been discretized with a precision of 30 bits; the mutation rate has been set to 8%, and a three step length has been adopted for the random walk; the one-point crossover has been chosen with an application rate of 85%. The toroidal landscape has been split in 4 regions of equal dimension and shape. Each set of runs has been repeated for the following flip rates:

0.0000	0.0005	0.0010	0.0100	0.0500	1.0000

The averaged objective function values resulting from each set of runs are reported in fig. 3, while the figs. 4 and 5 report the averaged convergence histories related to populations of 400 and 900 elements. Finally, in fig. 6 the whole set of best function values obtained for each run is reported for population of 144, 400 and 900 elements. The presented data show that the sub-population model can improve the robustness of the genetic algorithm in this particular case. However, the number of runs carried out is not enough to settle a clear correlation between the optimal flip rate and the population size.

5. MULTI-OBJECTIVE MATHEMATICAL PROBLEM

The multi-objective problem here considered is to find a suitable approximation of the Pareto front of the following two-objective problem:

$$\min_{\mathbf{x} \in X} \boldsymbol{F}(\mathbf{x}) \tag{2}$$

where

$$\boldsymbol{F} = (F_1, \ldots, F_k, \ldots, F_m),$$

$$F_k(\mathbf{x}) = \sqrt[4]{\frac{\sum_{i=1}^{n} \left[(x_i - a_k)^2 - 10cos[2\pi(x_i - a_k)] + 10 \right]}{n}},$$

$$\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_n) \in X \subseteq \mathbf{R}^n,$$

when

$$m = 2,$$

$$n = 16,$$

$$x_i \in [-5.12, 5.12],$$

and

$$a_1 = 0, a_2 = 1.5.$$

The same parameters of the previous case were adopted for setting up the genetic algorithm, with the exception of the elitist strategy that was not adopted here. Two different population sizes have been considered (64, 576), and the results related to three flip rates (0.0, 0.2, 1.0) are reported. Five runs for each flip rate have been executed with different starting population, and the figs. 7 and 8 report the front obtained combining those related to each single run with a given flip rate. How it can be observed, the sub-population model does not seem useful in this case. However the performance of the genetic algorithm is in any case poor, because it fails to locate the global optima for both functions.

6. APPLICATIONS TO WING DESIGN

Wing design is a highly multidisciplinary task; the use of designer expertise is therefore necessary to obtain realistic results, unless the various design criteria and off-design considerations can be included in the formulation of the optimization problem. The multiobjective optimization approach offers great advantages for these kind of problems, avoiding the need of arbitrarily interrelating the different design criteria into a single scalar objective function.

The parallel genetic algorithm has been here applied to the optimization of the shape of a wing for transonic flow conditions. This has been done in two separate steps: first, the wing planform has been optimized taking into account both aerodynamic and structural requirements; afterwards, the wing section has been modified to further reduce aerodynamic drag. The genetic algorithm has been coupled with a finite-difference full potential flow solver [6]. The design point chosen is set at Mach number $M = 0.85$ and lift coefficient $c_L = 0.5$. The starting point chosen is a straight, untwisted and untapered wing of aspect ratio *AR*=7, with a RAE 2822 airfoil; for simplicity, the wing planform is maintained trapezoidal, so that all geometric characteristics vary linearly from the root section to the tip. A total of 5 design variables have been used: 4 of these act directly on the wing planform, namely the taper ratio λ, the sweep angle at 25% of the chord Λ, the aspect ratio *AR* and the twist angle θ; moreover, the thickness at the wing root has also been included among the design parameters, while the thickness at the wing tip has been fixed at $t/c \mid_t = 10\%$. The wing surface is kept constant, so that the average wing loading is not changed during optimization. In table 1 the initial values of the design parameters are reported together with the allowed variation ranges. The wing twist is distributed symmetrically between the root and the tip; in other words, a twist angle θ corresponds to an increase of local incidence of $\theta/2$ at the tip, and a decrease of $\theta/2$ at the root. The wing weight is computed using the algebraic equation of [7] which combines analytical and empirical methods, and shows design sensitivity and prediction accuracy that make it possible to use it with success for preliminary design.

The selection has been carried out through a 2 steps random walk, with one-point crossover (rate = 100%) and bit mutation (rate =10%); a population of 64 individuals was let evolve for 50 generations, and several flip rates have been used, ranging from 0 to 1. In fig. 9 the Pareto fronts obtained corresponding to flip rates of 0 and 1 are illustrated; it can be seen how for this problem splitting the population into sub-populations doesn't lead to significant changes in the results.

As anticipated, after optimization of the planform a further improvement of the aerodynamic characteristics has been obtained by modifying the shape of the wing section. One of the solutions belonging to the Pareto front has been selected as starting point; the geometry chosen lies approximately at the center of the front, being characterized by $c_D/c_L^2 = 0.776$ and $W/W_o = 0.65$. The wing section has been modified using the shape functions technique described in [8]; 12 design variables have been used, and for simplicity the wing profile has been maintained constant in the spanwise direction. The optimization problem in this case has been formulated so as to reduce wave drag, at the same design point previously considered; like in the previous case, the lift coefficient has been fixed to $c_L = 0.5$, and the maximum thickness has been maintained at the value obtained by the previous run at each spanwise station. The same genetic algorithm parameters used for the wing planform optimization have been adopted, except for the mutation rate which has been reduced to 4%, and for the population size which has been reduced to 64. In fig. 10 the convergence histories obtained for various flip rates are illustrated. It is apparent how the subpopulations strategy seems to have a negative effect in this case, as the best performance is obtained when no boundaries are established. In fig. 11 the Mach number distribution over the wing before and after the airfoil optimization is shown; it is possible to observe how the intensity of the shock wave has been substantially reduced all along the span.

design variable	initial value	allowable range	selected wing
λ	0.0	[0.1 , 1.0]	0.10
Λ	1.0	[0.0 , 50]	39.6
θ	0.0	[-10 , 10]	-5.2
AR	7.0	[6.0 , 8.0]	6.41
$t/c\mid_r$ %	12	[12 , 15]	12.1

Table 1: Design parameters for the wing planform optimization

Finally, fig. 12 shows the parallel genetic algorithm execution times compared to the scalar version. The comparison has been made running for 10 generations the same wing section optimization problem described above, but with 32 population elements. The speed-up obtained is generally good, but it should be observed that the scalar version is faster than the parallel one when only one slave processor is activated. This behavior is due to synchronization and context switching overheads.

7. CONCLUSIONS

The sub-population model may be useful, in some cases, to improve the performance of a genetic algorithm. However it must be observed that the advantages obtainable are strongly dependent on the problem at hand. In the case of the single objective Rastringin's function, for example, the introduction of the sub-population model substantially improved the average performance of the genetic optimizer, but for the multiobjective test case there were not advantages with respect to the single population model. Probably, in this case, the advantage offered by the sub-populations, in terms of avoiding premature convergence, was not needed because the already good performance of the random walk selection scheme. The aerodynamic design problem showed, instead, how it can be even detrimental to have a sub-population organization when there are limits to the maximum number of objective function evaluations allowed. In such a case it is far more important to have an efficient and flexible implementation of a parallel genetic algorithm.

References

[1] D. J. Doorly, J. Peiró, J.-P. Oesterle, "Optimisation of Aerodynamic and Coupled Aerodynamic-Structural Design Using Parallel Genetic Algorithms," in *6th AIAA/NASA/USAF Multidisciplinary Analysis & Optimization Symposium*, AIAA Paper 96-4027, Bellevue, Seattle, WA, U.S.A., Sep. 1996.

[2] D. J. Doorly, J. Peiró, "Supervised parallel genetic algorithms in Aerodynamic optimisation," in *13th AIAA CFD Conference*, AIAA paper 97-1852, American Institute of Aeronautics and Astronautics (AIAA), Snowmass Co., U.S.A., June 1997.

[3] D. Quagliarella, A. Vicini, "Coupling Genetic Algorithms and Gradient Based Optimization Techniques," in Quagliarella D. et al., editors, *Genetic Algorithms and Evolution Strategies in Engineering and Computer Science*, John Wiley & Sons Ltd., England, Nov. 1997, pp. 289–309.

[4] A. Vicini, D. Quagliarella, "Inverse and Direct Airfoil Design Using a Multiobjective Genetic Algorithm," *AIAA Journal*, Vol. 35, No. 9, Sep. 1997, pp. 1499–1505.

[5] D. Cortesi, A. Evans, W. Ferguson, J. Hartman, *Topics in IRIX™ Programming — Chapter 8, Models of Parallel Computation*, Silicon Graphicsi Inc., 1996, Document Number 007-2478-004.

[6] A. Jameson, D. A. Caughey, "Numerical Calculation of the Transonic Flow Past a Swept Wing," Technical Report NASA-CR-153297, NASA, June 1988.

[7] E. Torenbeek, "Development and Application of a Comprehensive, Design-Sensitive Weight Prediction Method for Wing Structures of Transport Categoty Aircraft," Technical Report LR-693, TU Delft, Sep. 1992.

[8] A. Vicini, D. Quagliarella, "Airfoil and Wing Design Through Hybrid Optimization Strategies," in *16th Applied Aerodynamics Conference*, AIAA Paper 98-2729, American Institute of Aeronautics and Astronautics (AIAA), Albuquerque, New Mexico, June 1998.

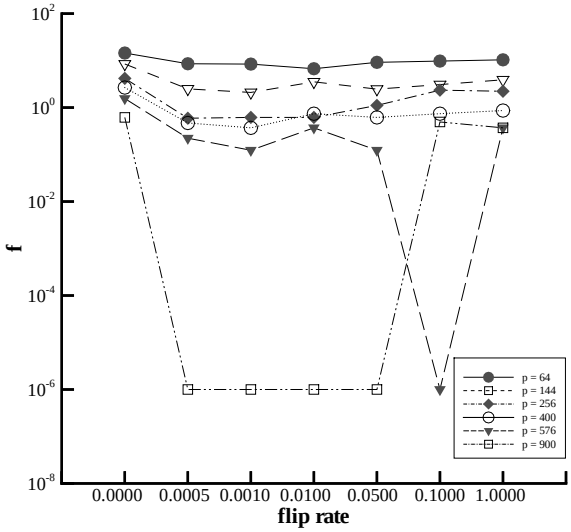

Figure 3: Rastingin's function: averaged results

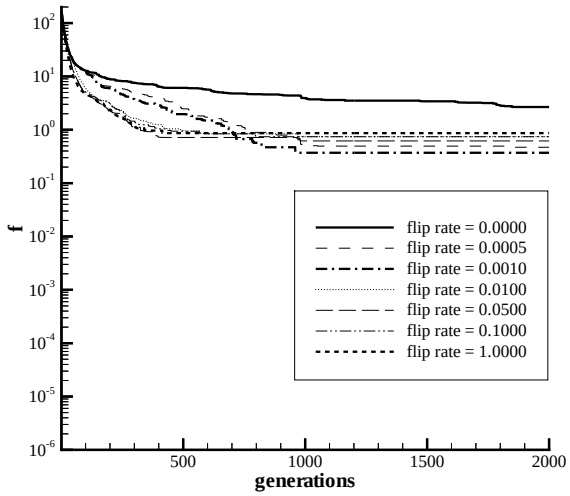

Figure 4: Rastringin's function, averaged convergence histories for a population size of 400 elements

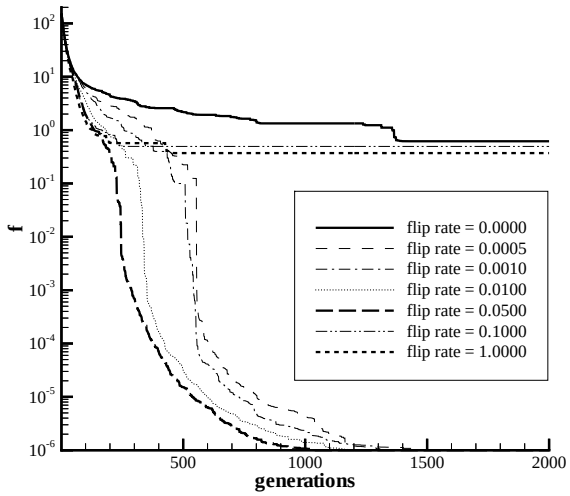

Figure 5: Rastringin's function, averaged convergence histories for a population size of 900 elements

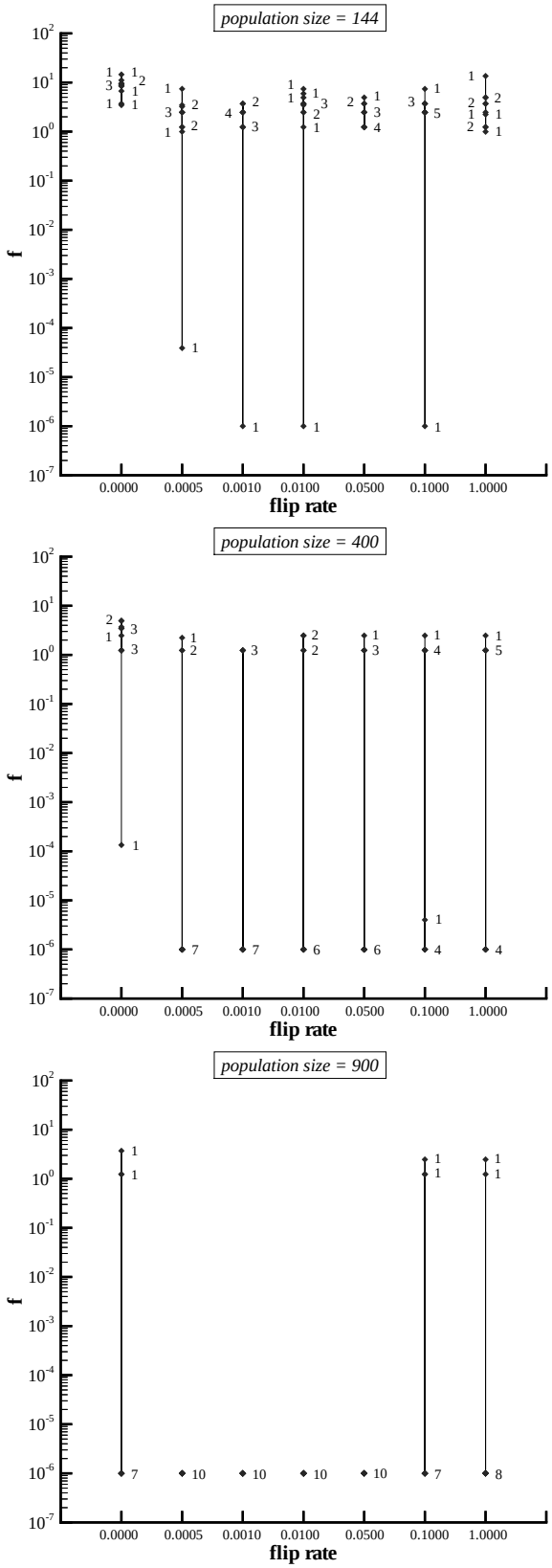

Figure 6: Rastringin's function, best values for each run; the number of runs that produced the same solution is reported on the sides of the relative point

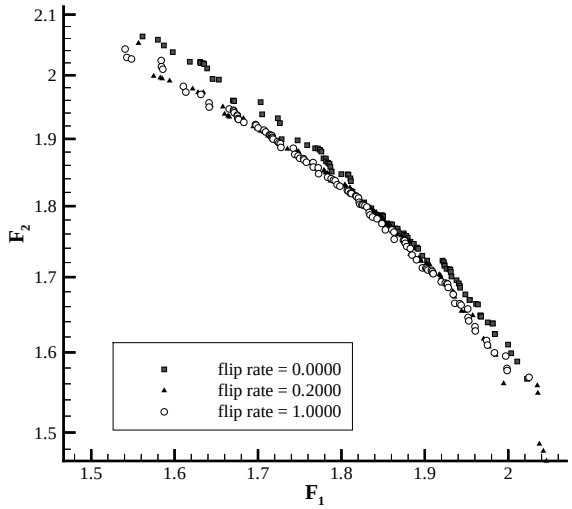

Figure 7: multi-objective mathematical problem, Pareto fronts obtained with a population of 64 elements, after 100 generations

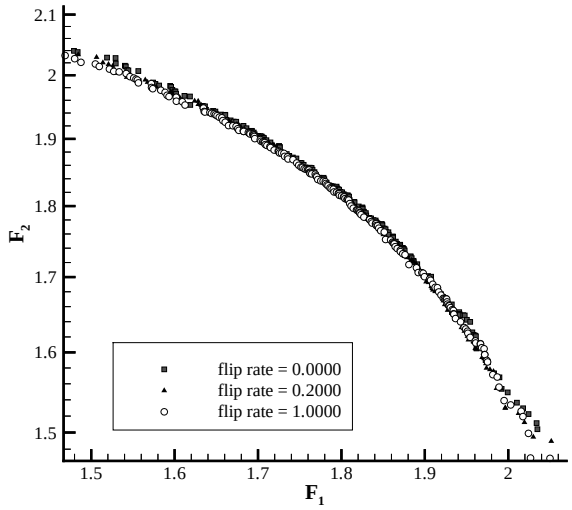

Figure 8: multi-objective mathematical problem, Pareto fronts obtained with a population of 576 elements, after 100 generations

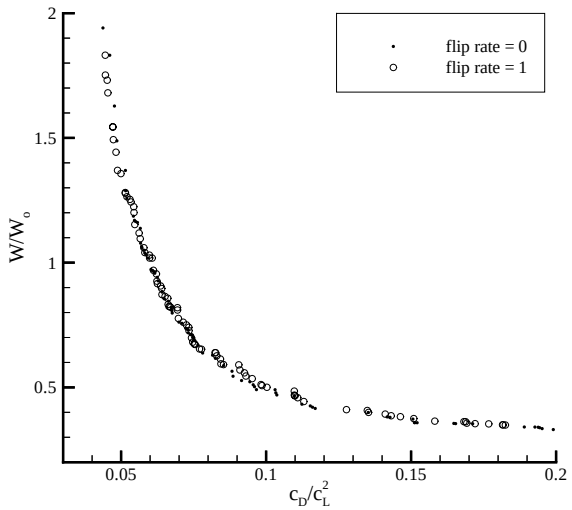

Figure 9: Pareto fronts of the wing planform design

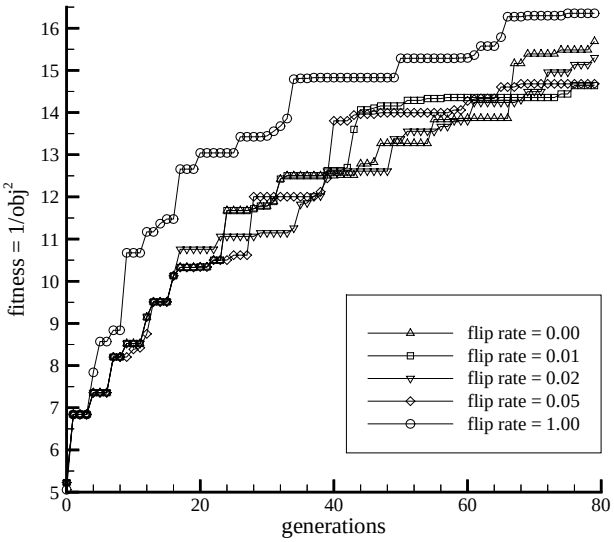

Figure 10: convergence histories of the wing section optimization

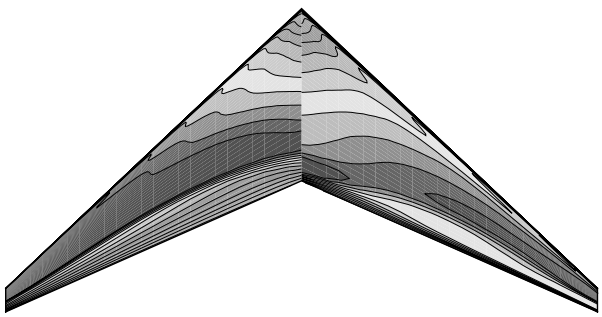

Figure 11: Mach number distribution; initial wing on the left, final wing on the right

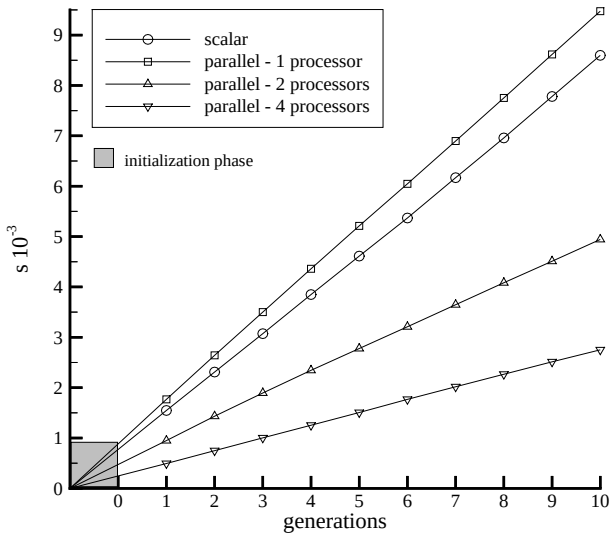

Figure 12: parallel genetic algorithm execution times