Scientia Iranica D (2015) 22(6), 2379–2388

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
www.scientiairanica.com

A hierarchical parallel strategy for aerodynamic shape optimization with genetic algorithm

M. Ebrahimi and A. Jahangirian*

Department of Aerospace Engineering, Amirkabir University of Technology, Tehran, Iran.

Received 1 June 2014; received in revised form 6 September 2015; accepted 3 November 2015

KEYWORDS
Parallelization;
Aerodynamic shape optimization;
Computational fluid dynamics;
Genetic algorithm.

Abstract. An efficient parallel strategy is presented for optimization of the aerodynamic shapes using Genetic Algorithm (GA). The method is a hybrid Parallel Genetic Algorithm (PGA) that combines a multi-population PGA and master-slave PGA using Message Passing Interface. GA parameters are firstly tuned according to the fact that sub-populations evolve independently. The effect of the number of sub-population on the computational time is investigated. Finally, a new strategy is presented based on the load balancing that aims to decrease the idle time of the processors. The algorithm is used for optimization of a transonic airfoil. An unstructured grid finite volume flow solver is utilized for objective function evaluations. For the considered class of problems, the suggested Hierarchical Parallel Genetic Algorithm (HPGA) results in more than 30% reduction in optimization time in comparison to regular master-slave PGA. A semi-liner speed-up is also obtained which indicates that the model is suited for modern cluster work stations.

1. Introduction

Genetic Algorithms (GAs), as an evolutionary method, have now introduced themselves as a powerful tool for various design optimization problems. One of the key features of GA is that it searches the design space in a population of points, resulting in a greater likelihood of finding the global optimized point [1]. Additionally, it only needs the objective function and does not require its derivatives. Such features make GA attractive for practical engineering applications like aerodynamic shape optimization [2-4]. However, GA has the key disadvantage of being computationally time-consuming in aerodynamic optimization problems, where CFD methods are used for objective function computation. Therefore, reducing the computational time of CFD simulations problems is a prominent area of research, whereas high fidelity of the analysis is retained [5,6]. Fortunately, another well-known merit of GAs is their capability to partition the population of individuals within multiple computing clusters and nodes. It is widely accepted that optimization with GAs principally can take full benefit of massively parallel computer architectures [7,8]. This point is motivated by the fact that the objective values, associated with each member of the population among each generation of the algorithm, can be evaluated in parallel. Thus, the clock-time needed to reach an acceptable solution is reduced [9]. In addition, to improve the convergence rate of classical GAs migration operators as well as fine and coarse grain PGAs-based subpopulations were introduced [10,11]. With the development of multi-core computers and growth of the problem's scale, PGA has been widely used in many fields of research and communication efficiency and parallel computing speed-up have been enhanced rapidly [12,13].

HPGAs are inspired by PGAs, which are, in turn, based on several GAs being run in parallel [14,15].

*. *Corresponding author. Tel.: +98 21 64543223*
E-mail addresses: Mebrahimi@aut.ac.ir (M. Ebrahimi); Ajahan@aut.ac.ir (A. Jahangirian)

Each model in hierarchical algorithms could have an algorithm with various control parameters, such as probability of reproduction, crossover, mutation and population size, etc. For an overview of HPGA applications, one can see references [7,16].

Although numerous studies on CFD optimizations with parallel GAs have been reported, most of them consider homogeneous and dedicated computing resources, which are not easily extendable towards harnessing computing nodes [17,18]. In addition, optimizations with PGAs are mostly constrained by the limited commercial licenses, especially due to the high costs of commercial analysis packages. This is one of the main reasons that the recent advent of grid computing has gained widespread attention, as it establishes the concept of generating a set of open standards for distributed computational resources.

The present paper evaluates a hierarchical parallel GA strategy for optimum shape design applications that is well suited to the problems with high CPU costs and large memory requirements. The algorithm is not sensitive to the structure of computing resources and is easily implemented in harnessing or homogeneous computing resources. Furthermore, crucial GA parameters are tuned and efficiency of the proposed method under various numbers of sub-populations as well as individuals is assessed. Finally, a new strategy based on load balancing is presented, which can effectively lead to computational time-saving.

2. Genetic algorithm

Genetic Algorithm is a search algorithm that is based on the natural selection and genetics. It uses three operators of crossover, reproduction, and mutation [1]. In the current study, a real coded GA is applied and chromosomes, genes, and fitness are corresponding to the design candidates, design variables, and objective function, respectively [19]. According to the nature of the problem and considering the state-of-art, an elitist strategy for the tournament operator is applied, where the two best chromosomes in each generation are transferred into the next generation without any change [20]. Selected airfoil shapes comprise the initial population for comparison purposes. The objective function is evaluated using the numerical solution of governing flow equations. Then, the population is optimized according to the objective function value (fitness) through the GA. The crossover operator exchanges the chromosomes of the selected parents, randomly. A simple one-point crossover operator [20] is utilized in this paper with 75% probability of combination, as the use of smaller values was observed to deteriorate the GA performance [19].

To provide a better diversity in the design space, a uniform mutation rate of 7% with variable boundary is applied to randomly select genes of each chromosome. Unlike other well-known strategies, based on the feedback obtained by monitoring fitness value evaluations of individuals, the boundary is updated in each generation. Mutation is applied to the parent genes up to the cross-over point, where the genes of the parent are exchanged to produce the offspring.

3. CFD evaluation of fitness function

Since most of the computational time required for the airfoil shape optimization process is consumed by CFD solver, it must possess high efficiency and convergence rate. In this work, the turbulent flow equations are solved using a finite volume cell-centered implicit scheme that follows the work of Jahangirian and Hadidoolabi on unstructured grids [21]. A two-equation $k-\varepsilon$ turbulence model is implemented together with the wall function near wall treatment for computation of Reynolds Averaged Navier-Stokes (RANS) equations [22]. The generation of high-quality grids is essential in this work, because the CFD solver performs several hundreds of times in a single optimization cycle. Therefore, the successive refinement approach [23] is used in the current research. The method is capable of producing high-quality (regular) stretched cells inside the boundary and shear layers as well as isotropic cells outside these regions. During the optimization process, the airfoil boundaries are changing; therefore, the existing grid is modified in an automatic manner using tension-spring analogy in order to be adapted to the changing domain [24].

4. Parallelization strategy

Several possible parallelization strategies can be taken into account for the problems related to the time-consuming CFD simulations [25,26]. It is only during the recent years that efforts have been made to propose methodologies for designing PGAs in the field of the aerodynamic shape optimization [16]. One of the well-known strengths of PGAs is their capability to facilitate speciation, a process by which various subpopulations evolve simultaneously in diverse directions. Panmictic GAs (a GA without any structure) can be parallelized readily by using master/slave model, which works well for a small number of individuals. However, as the number of nodes increases, Panmictic GAs becomes inefficient by excessive communications [27]. In addition to the parallel panmictic GA, the island and cellular GAs are two other popular parallel structured GAs [11]. Figure 1 shows such basic models of PGA.

In master/slave PGAs, only a single panmictic population, i.e. a Canonical GA (CGA), is assumed to exist. However, on the contrary to CGS, individuals' evaluations are distributed among the processing slave

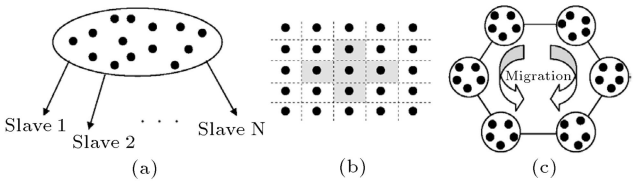

Figure 1. Different basic PGAs: (a) Master/slave; (b) cellular; and (c) distributed.

nodes by scheduling fractions of the population. This model can be implemented easily and does not alter the search behaviour of a CGA.

Cellular PGA consists of only a spatially structured single population, which is designed to run on a closely linked massively parallel processing system. In such an algorithm, selection and mating are limited to small groups that overlap to permit some interactions among all members. Hence, good solutions might be disseminated across the entire populations. Sometimes, the Cellular parallel GA is also termed as the Fine-grained PGA.

A distributed PGA may sound more complicated, as it consists of several subpopulations that exchange members occasionally. This exchange of members is called migration that is controlled by several parameters. Distributed PGAs are also known as the multi-deme or island model PGA.

Different PGA models may be used together to form other new Hierarchical PGA (HPGA) models. For example, one may produce a hierarchical PGA that combines a distributed PGA (at the upper level) and a Cellular PGA or master-slave PGA, which we consider in this paper, or even another level of island PGAs (at lower levels). Basically, HPGA is any combination of two or more of the three basic forms of PGA.

Despite the wide range of PGA applications in different fields of optimization, many important parameters need to be tuned when it is applied in the field of aerodynamic shape optimization. The aim of the parallel strategy described so far is to reduce the cost of objective function evaluations. While, another level of parallelism, which is directly related to the subpopulation evaluation, can be exploited at the cluster level; this applies to the GA operators, airfoil shape generation, and grid movement.

In the present work, a two-level HPGA, including island model for the first level and master/slave for the second one, is applied. Here, one of the main remarks is that after each chromosome and subpopulation evaluation, the information exchange shall occur. In addition, to minimize the computational time, it is important to send only the expensive part of the program to the lower level of parallelization. In fact, by applying more CPUs, the program should be able to compensate the added time due to the communication of information among computational nodes. The main steps of the proposed HPGA for aerodynamic shape optimization can be outlined as follow (Figure 2):

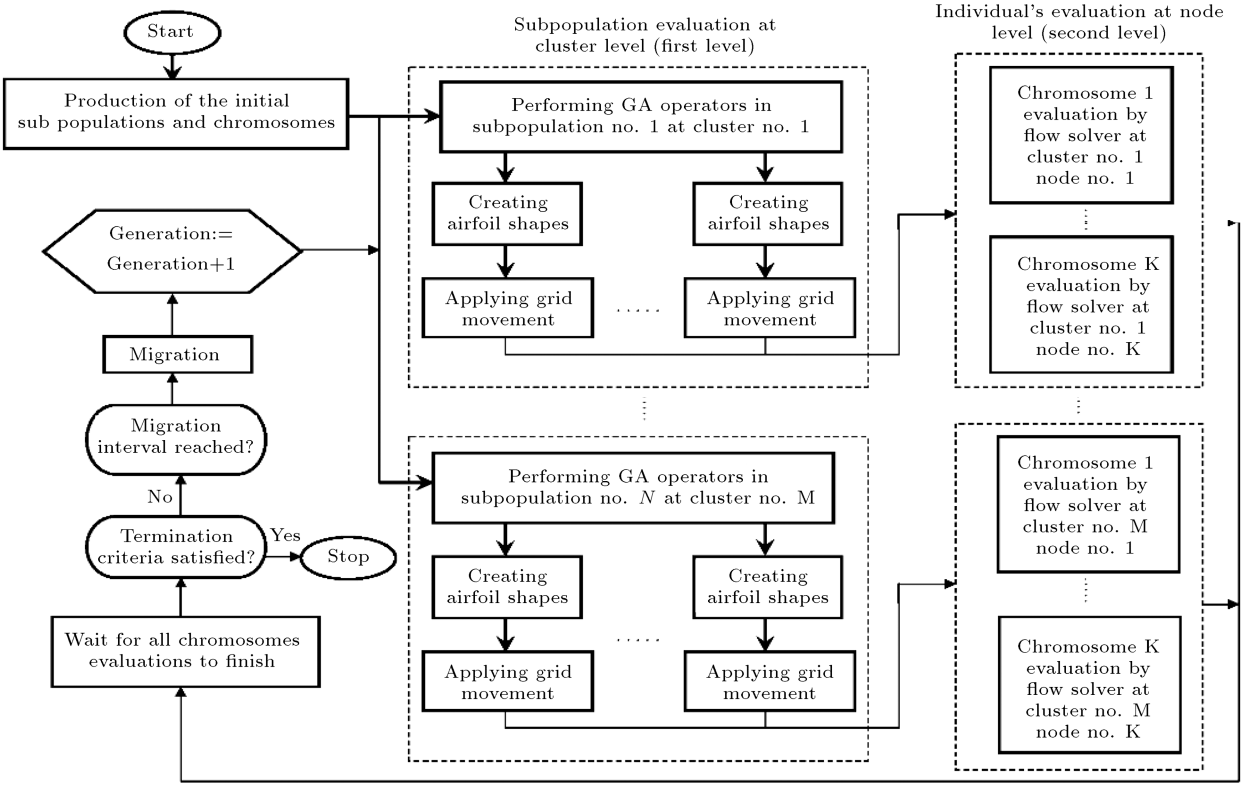

Figure 2. The flowchart of the proposed HPGA for aerodynamic shape optimization.

1. The HPGA subpopulations are transferred onto the computing nodes, which after being contacted by the main program offered, require subpopulation and individual evaluation services.
2. Parallel subpopulations' evolutions then begin at the selected computing clusters. Whenever they receive a launch request of the subpopulation evolution service, job submission protocol is represented at the master node of the respective clusters. The main tasks include GA operations, airfoil shapes generation, and grid movement.
3. Then, at each cluster, scheduling and resource discovering are conducted to farm the field of available processing nodes for chromosome evaluations.
4. Once all chromosomes of each subpopulation are evaluated by CFD solver, the obtained objective function is marshalled back to the master node to undergo parallel algorithm.
5. Finally, the developed subpopulations are leaded back to the HPGA master to proceed with the migration operation. Such a process repeats until the optimization criteria are met.

5. Results

This section is mainly devoted to a detailed evaluation of the proposed strategy (HPGA) for aerodynamic shape optimization. The proposed method is applied to the problem of airfoil shape optimization. A typical test case is chosen with extensive and reliable computational data available. The parallel computer system at the Laboratory of High Performance Computing in Amirkabir University of Technology (LHPC) is used for parallel implementation with the configuration as summarized in Table 1.

Table 1. Configuration of the parallel computer of CALA.

Unit	Configuration
CPU	AMD Opteron 2.54 GHz
Memory	200 Gbytes
Network	Myrinet
Operating system	Linux (64-bit version)

5.1. Aerodynamic shape optimization

Although our main goal in this paper is to obtain an efficient parallelization strategy for aerodynamic shape optimization, in this section, the efficiency of the base optimization process is demonstrated; in addition, a comparison between the parallel and serial outcomes are carried out. More details of the serial method may be found in [27]. A transonic flow is considered with the Mach number of 0.74, Reynolds number of 6.5 million, and incidence angle of 2.8 degrees. The RAE-2822 airfoil is considered as the initial airfoil and the objective function is the lift coefficient (Cl) to the drag coefficient (Cd) which is computed by solving the Reynolds-averaged Navier-Stokes equations. The computational field is discretized using triangular unstructured grids. The unstructured grid generated around the initial airfoil containing 10,651 cells is shown in Figure 3. Also, the modified grid around design airfoil using spring analogy is illustrated in this figure.

The distributions of surface pressure coefficient (C_p) and obtained airfoil shapes for parallel and serial optimization algorithms (considering imposed physical constrains) are plotted in Figure 4. According to this figure, there is a rather strong shock wave near the middle part of the initial airfoil's upper surface that is weakened in the optimum shape.

The values of lift and drag coefficients and the objective functions for the initial and optimum shapes are also shown in Table 2. According to this table, as well as Figure 4, no significant divergence is observed for the parallel and serial solutions. The limited differences between serial and parallel results can be assumed due to the random nature of GA. For instance,

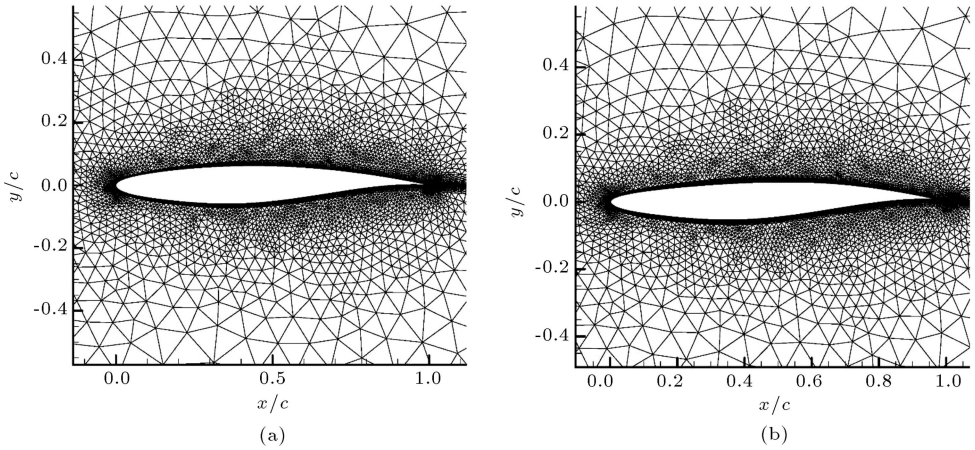

Figure 3. Unstructured grids around (a) initial airfoil, and (b) optimum airfoil.

Table 2. Lift and drag coefficients for the applied optimization method.

	C_l	C_d	C_l/C_d	Execution time (hr)
Initial shape	0.81	0.0261	31.09	-
Optimized airfoil-serial solution	0.88	0.0150	58.66	502.3
Optimized airfoil-parallel solution	0.89	0.0149	59.73	16.9

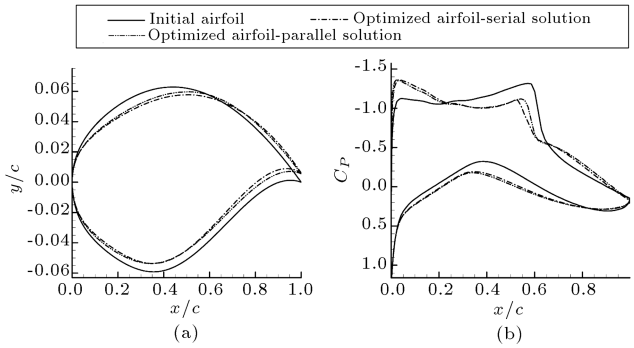

Figure 4. The obtained airfoil shapes (a) and surface pressure coefficients (b) for parallel and serial optimization algorithms.

the number of applied CPUs for this case is considered to be 60.

5.2. Parallel performance study

In this section, the performance of the parallel strategy is assessed. In particular, we are interested in how the proposed method for aerodynamic shape optimization performs under various numbers of individuals, subpopulations, computing nodes, and the cluster size. These objectives are studied in the following subsections. It should be noted that for all cases, the number of processors is considered the same as the number of individuals.

Optimizing the population size. When using a parallel processor for shape optimization with GA, optimum selection of the population size is highly demanded. In this section, the performance of a parallel optimization algorithm under different population sizes is investigated. Thus, the evolutions of PGA subpopulations are conducted only in a single level of parallelism, i.e. only level 2 HPGA, presented in Figure 2. Like the population size, which is equal to the increases in the number of processors, the clock time for evaluation of all individuals in each generation, for most of the cases, will rise. That is due to the fact that different airfoil shapes require different numbers of CFD iterations for evaluation. Higher population size, in turn, could lead to lower required numbers of generations in order to gain the same level of objective values. Therefore, to minimize the clock time of optimization process, a compromise between the population size and the required numbers of generations should be

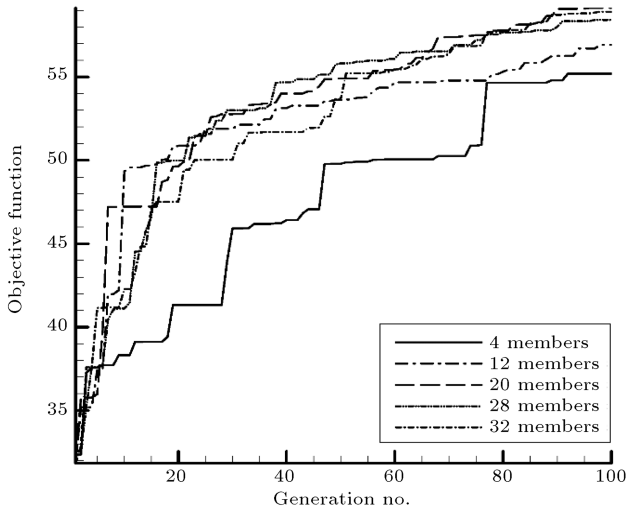

Figure 5. Convergence history of the maximum objective values after 100 generations.

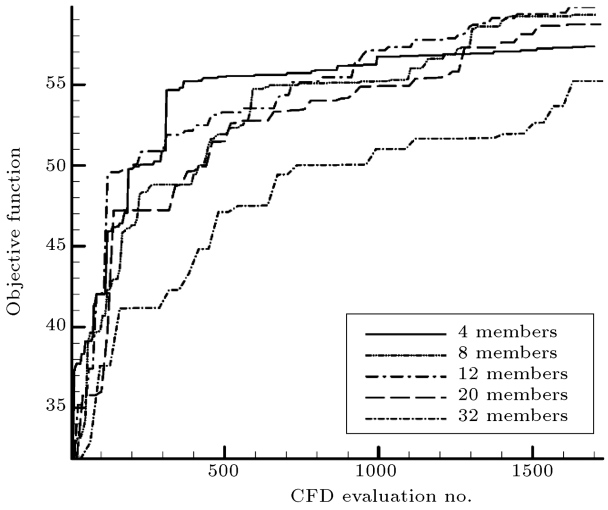

Figure 6. Convergence history of the maximum objective values after the evaluation of 1700 individuals.

applied. The convergence history of the maximum objective values for different population sizes (equal to the number of processors) after 100 generations is presented in Figure 5. To compare the required CFD evaluations, in Figure 6, the convergence history of the maximum objective values against the CFD evaluations is presented. Two distinctive outcomes can be concluded from these figures: (1) The optimum number of populations for airfoil shape optimization with a parallel GA is around 20; and (2) When using

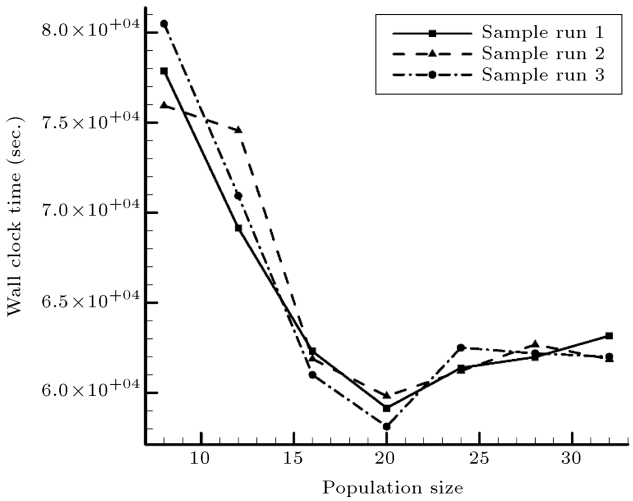

Figure 7. Total clock time for different population sizes.

single processing machine, the optimum number of individuals in this case is around 8 to 12.

Figure 7 shows the total clock time against the population size when the parallel processor is used. It should be noted that the calculated time, here, is the period when the program starts up to the time it reaches the objective value of 58.5. This figure also emphasizes the above conclusion that the optimum number of members is 20 in this case.

Subpopulation strategy. In this section, the first level of parallelization, shown in Figure 2, is studied, i.e. n subpopulations of the HPGA evolving across n number of parallel clusters are modelled. The computational efforts incurred by Single Cluster HPGA and HPGA for optimizing the airfoil shape under two cases, including n subpopulation in 1 cluster and n subpopulation in n clusters, are reported in Figure 8. It should be noted that the numerical results are reported for the averages of 5 independent runs for the case mentioned in Section 5.1 with population size of 20. Migration interval is considered MI = 4, since bigger values increase the idle time of processors, while the smaller ones influence the randomness of GA and decrease the positive effect of applying subpopulations.

Termination of the program occurs when the level of objective function reaches 58.5. This figure indicates that the HPGA is computationally more efficient than the Single Cluster HPGA. In addition, it shows that for HPGA, an optimum number of subpopulation exists in which by applying more subpopulations (here is equal to the number of clusters which is 3), not only does the computational cost increase, but also the clock time of the optimization process is risen.

Load balancing. In the parallel processing of airfoil shape optimization, many situations occur resulting in computational load imbalance. This is mainly due to the nonlinear behaviour of the CFD flow solution for different airfoil shapes of a generation. Figure 9 shows the evaluation time of 20 individuals from the 10th generation of an optimization problem with the flow conditions described in Section 5.1. According to this figure, about 20% imbalance is observed from the average evaluation time.

Our measurements showed that the aggregate time spent in the flow solver represents about 90% of the total elapsed time for a 20-population size problem. Therefore, when subpopulations are iterating along with each other, it is particularly important that approximately the same time be spent for total evaluation time of individuals (second level of parallelization). This problem can also be observed when using subpopulation strategy. The wall clock time of each generation evaluation for the mentioned optimization problem with three subpopulations is reported in

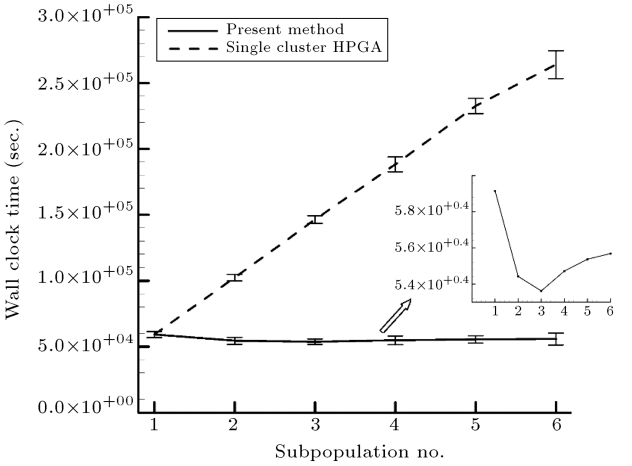

Figure 8. A comparison between optimization process wall clock times for the present method (HPGA) and single cluster HPGA.

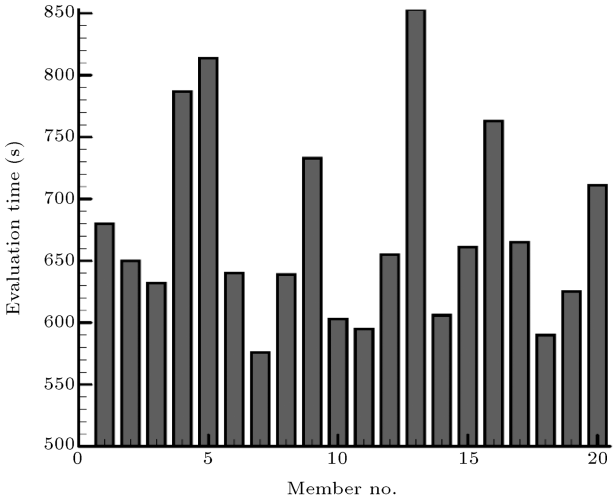

Figure 9. Computational time for evaluation of 20 individuals of the 10th generation.

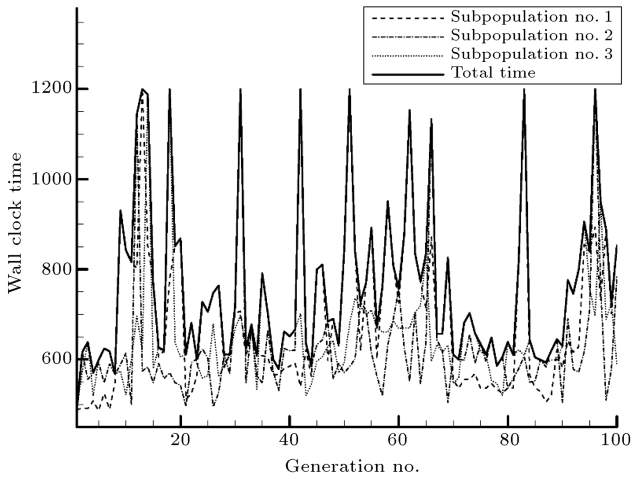

Figure 10. Wall clock time of each three sub-populations' evaluation as well as the total time.

Figure 10. According to the fact that the individuals are heterogeneous, the most time-consuming (solid line in Figure 10) evaluation has been the bottleneck of HPGA, since it waits for the completion of all the chromosome calculations before the migration and search operations may proceed.

To minimize the idling time of the processors, a simple strategy has been applied in which from the subpopulations with the highest time of evaluations individuals with the highest evaluating time migrate to subpopulations with lower times of evaluation. Migration interval is considered as 4, and according to the time of each generation evaluation, one or two members plus the best member are exchanged. Figure 11 compares the clock time of the previous problem with the one in which the Balanced HPGA (B-HPGA) is applied. In Figure 12, the effect of migration interval on the total clock time of optimization process is presented which indicates that the optimum value for the problem described in Section 5.1 is 4.

According to this figure, the Unbalancing Ratio

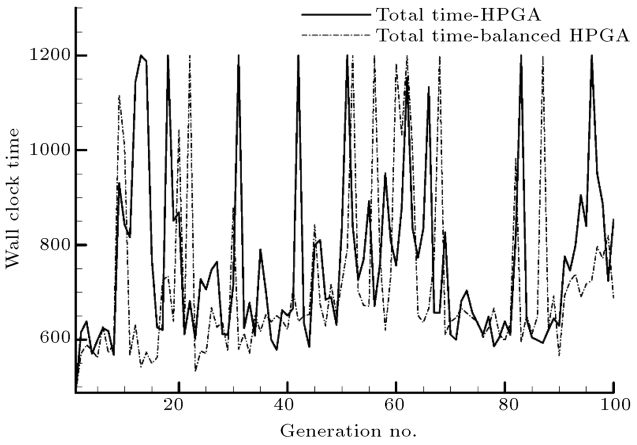

Figure 11. A comparison between the waiting times of HPGA and B-HPGA.

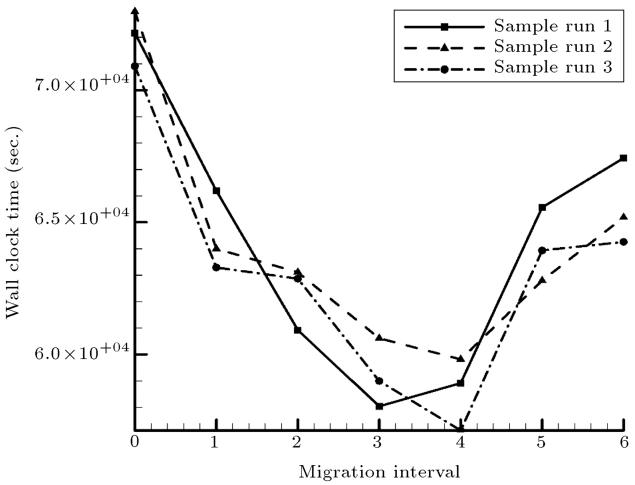

Figure 12. Effect of migration interval on the total clock time of optimization process.

Table 3. UBRs and total clock times (Tct) of HPGA and B-HPGA after 100 generations.

Population	HPGA		B-HPGA	
size	Tct (s)	UBR	Tct (s)	UBR
12	70150	0.16	66593	0.14
20	53616	0.24	49186	0.17
32	68160	0.36	62212	0.28

(UBR), which is defined as follows, is decreased from 35% to 27%:

$$\text{UBR} = \frac{\sum_{k=1}^{G} \frac{(T1_k - T2_k)}{T2_k}}{G}, \tag{1}$$

where $T1$ and $T2$ are the longest and shortest times of individual evaluations, respectively, and G is the generation number. To evaluate efficiency of the balancing method, the average clock time and UBR of the problems with population sizes of 12, 20, and 32 are compared with the normal HPGA in Table 3. Termination of the program occurs when the objective function reaches the level of 58.5. The results show about 30% reduction in UBR and 9% reduction in total clock time compared with the HPGA.

5.3. Scalability

Scalability is the ability of a parallel system to retain the performance levels when additional processors are utilized. In other words, it can refer to the capability of a system to decrease the computational time of a problem under an increased load when resources are added. This parameter, as a property of systems, is generally defined as:

$$\text{Scalability} = \frac{T_s}{T_{p\times N}}, \tag{2}$$

where T_s and T_p are the execution times of the sequential and parallel algorithms, respectively, and N is

Table 4. The effect of the number of sub-populations on the scalability of the problem.

Sub-population size	No. of CPU	Execution time		Scalability
		Parallel (Sec)$\times 10^4$	Serial (Sec)$\times 10^4$	
1	20	16.8	162.5	0.48
2	40	14.4	141.9	0.25
3	60	12.1	127.2	0.18
4	80	15.6	138.4	0.11

the number of processors. According to the mentioned definition, the proposed algorithm is said to scale if it is suitably efficient when a large number of CPUs are applied, e.g. a large number of subpopulations or a large number of participating nodes are applied. In Table 4, a sample problem with the conditions described in Section 5.1 is solved and the effect of increasing resources on the scalability and the execution time are presented. According to this table, with applying more CPUs, both the computational time and the scalability are decreased.

5.4. Parallel speed-up

When running a parallel algorithm, one of the main performance issues, in comparison to a sequential run of the same algorithm, is how much speed-up it can offer. Such a speed-up is defined by:

$$S_p = \frac{T_s}{T_p}, \tag{3}$$

where T_s and T_p are the same as those in Eq. (2). To compare the efficiency of the proposed method in terms of the actual clock time of optimization process, a parameter called Cost Function Efficiency (CFE) is introduced using Amdahl's law [28]. Three subpopulations and 20 members are considered here, since they are proved to be the optimum values. The performance of the proposed parallelization strategy for the above airfoil design problem is assessed and the results are presented in Table 5. Moreover, the influence of the proposed load balancing (B-HPGA) method is presented for the population sizes of 12, 20, and 32. For all cases, the calculation time defines the period when the program starts up to the time it reaches the objective value of 58.5.

Looking at Table 5, it is observed that by increasing the population size for both HPGA and B-HPGA, the speed-up is increased. This table also indicates that the obtained speed-up and CFE for the proposed B-HPGA are increased for all cases in comparison to HPGA. More importantly, it shows that how the use of a proper parallelization strategy could lead to more CFE, which means that applying more subpopulations as well as individuals does not always result in more efficiency. For instance, when the HPGA is applied and the numbers of subpopulations and population size are 32 and 4 respectively, the CFE is about 31% less than when 20 individuals and 3 subpopulations are utilized. The main reason is that by using more subpopulations, the idle time of processors increases. However, no significant different is observed in the number of generations in the optimization process.

6. Conclusions

A two-level Parallel Genetic strategy, including a master-slave PGA at the lower level and a distributed PGA at the upper one, was proposed. It showed to be very promising for aerodynamic shape optimization. Some crucial parameters in both levels were optimized and a new strategy for load balancing was proposed. The efficiency of the method was investigated through airfoil shape optimization. It was found that by using the proposed strategy for aerodynamic shape optimization, significant reduction in the computational time

Table 5. Speed-up and cost function efficiency of HPGA and B-HPGA.

Population size		One sub-population (PGA)			Two sub-population			Three sub-population			Four sub-population		
		12	20	32	12	20	32	12	20	32	12	20	32
S_p	B-HPGA	-	-	-	22.3	37.0	58.8	32.8	55.3	87.4	43.2	72.6	113.3
	HPGA	11.0	18.3	29.4	20.9	35.2	55.1	30.3	52.4	81.8	39.6	68.4	106.7
CFE	B-HPGA	-	-	-	72.1	97.8	77.6	74.7	100	80.1	70.5	97.6	76.9
	HPGA	64.3	88.4	67.7	68.4	91.1	71.9	69.8	92.3	72.7	65.3	89.2	69.2

is obtained. The semi-liner speed-up also indicates that the model is suited for modern cluster work stations.

Nomenclature

Symbols

T_1	The longest time of individual evaluations
T_2	The shortest time of individual evaluations
G	Generation number
Sp	Speed-up
T_s	The execution time of the sequential algorithms
T_p	The execution time of the parallel algorithms

Definitions, acronyms, and abbreviations

GAs	Genetic Algorithms
PGA	Parallel Genetic Algorithm
HPGA	Hierarchical Parallel Genetic Algorithm
CFD	Computational Fluid Dynamics
CGA	Canonical GA
LHPC	Laboratory of High Performance Computing
C_p	Pressure coefficient
C_l	Lift coefficient
C_d	Drag coefficient
MI	Migration Interval
UBR	Unbalancing Ratio
Tct	Total clock time
CFE	Cost Function Efficiency

References

1. Goldberg, D.E., *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley, Reading (1989).
2. Shahrokhi, A. and Jahangirian, A. "Surrogate assisted evolutionary optimization method with application to the transonic airfoil design", *Eng Opt.*, **42**(6) pp. 497-515 (2010).
3. Pehlivanoglu, Y. and Yagiz, B. "Aerodynamic design prediction using surrogate-based modeling in genetic algorithm architecture", *Aero. SCI Tech.*, **23**(1), pp. 479-491 (2011).
4. Leifssona, L. and Kozielb, S. "Aerodynamic shape optimization by variable-fidelity computational fluid dynamics models: A review of recent progress", *Journal of Computational Science*, **10**(1), pp. 45-54 (2015).
5. Ebrahimi, M. and Jahangirian, A. "New analytical formulations for calculation of dispersion parameters of Gaussian model using parallel CFD", *Envi. Fluid Mech.*, **13**(2), pp. 125-144 (2013).
6. Tai, C.H., Liew, K.M. and Zhao, Y. "Numerical simulation of 3D fluid-structure interaction flow using an immersed object method with overlapping grids", *Comp. Struct.*, **85**(11), pp. 749-762 (2007).
7. Marco, N. and Lanteri, S. "A two-level parallelization strategy for genetic algorithm applied to optimum shape design", *Parallel Comp.*, **26**(4), pp. 377-397 (2000).
8. Oktay, E., Akay, H. and Merttopcuoğlu, O. "Parallelized structural topology optimization and CFD coupling for design of aircraft wing structures", *Comp. Fluids*, **49**(1), pp. 141-145 (2011).
9. Daneshtalab, M., Ebrahimi, M., Xu, T.C., Liljeberg, P. and Tenhunen, H. "A generic adaptive path-based routing method for MPSoCs", *J. Syst. Architect.*, **57**(1) pp. 109-120 (2011).
10. Cantu-Paz, E. "A summary of research on parallel genetic algorithms", Technical Report 95007, IlliGAL Report, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory (1995).
11. Nowostawski, M. and Poli, R. "Parallel genetic algorithm taxonomy", In: *Proceedings of the Third International Conference on Knowledge-Based Intelligent Information Engineering Systems*, KES'99, pp. 88-92, Adelaide (1999).
12. Daneshtalab, M., Palesi, M., Angiolini, J., Plosila, M. and Ebrahimi M. "Proceedings of the 2nd international workshop on many-core embedded systems (MES)", Held in conjunction with the *41st Annual IEEE/ACM International Symposium on Computer Architecture* (2014).
13. Rocha, I., Parente, E. and Melo, J. "A hybrid shared/distributed memory parallel genetic algorithm for optimization of laminate composites", *Compos Struct.*, **107**, pp. 288-297 (2014).
14. Periaux, J., Mantel, B., Sefrioui, M., Stouuet, B., Desideri, J., Lanteri, S. and Marco, N. "Evolutionary computational methods for complex design in aerodynamics", In: *96th American Institute for Aeronautics and Astronautics Conference*, AIAA-98-0222, Reno (1998).
15. Kim, J. and Zeigler, B.P. "A framework for multiresolution optimization in a parallel/distributed environment: Simulation of hierarchical gas", *J. Parallel Distr. Com.*, **32**(1), pp. 90-102 (1996).
16. Lim, D., Ong, Y., Jin, Y., Sendhoff, B. and Lee, B. "Efficient hierarchical parallel genetic algorithms using grid computing", *Future Gener Comp SY.*, **23**(4), pp. 658-670 (2007).
17. Menon, S., Mooney, K.G., Stapf, K.G. and Schmidt, D.P. "Parallel adaptive simplical re-meshing for deforming domain CFD computations", *J Comput Phys.*, **298**(1), pp. 62-78 (2015).

18. Ebrahimi, M., Daneshtalab, M., Liljeberg, P., Plosila, J. and Tenhunen, H. "Cluster-based topologies for 3D networks-on-chip using advanced inter-layer bus architecture", *Elsevier Journal of Computer and System Sciences*, **79**(4), pp. 475-491 (2013).

19. Tes, D. and Chan, Y.Y. "Multi-point design of airfoil by genetic algorithm", In: *8th Annual Conference of the CFD Society of Canada*, Montreal (2000).

20. Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*, Academic Press, New York (2001).

21. Jahangirian, A. and Hadidoolabi, M. "Unstructured moving grids for implicit calculation of unsteady compressible viscous flows", *Int. J. Numer. Meth. FL.*, **47**(10), pp. 1107-1113 (2005).

22. Launder, B.E. and Spalding, D.B. "The numerical computation of turbulent flows", *Comput. Method. Appl. M.*, **3**, pp. 269-289 (1974).

23. Jahangirian, A. and Johnston, L.J. "Automatic generation of adaptive unstructured grids for viscous flow applications", In: *5th International Conference on Numerical Grid Generation in CFD*, Mississippi State University (1996).

24. Shahrokhi, A., Jahangirian, A. and Fouladi, N. "Navier-Stokes optimization using genetic algorithm and a flexible parametric airfoil method", In: *ERCOFTAC Conference on Design Optimization: Methods and Application*, Spain, University of Las Palmas de Gran Canaria (2006).

25. Tai, C.H., Zhao, Y. and Liew, K.M. "Parallel computation of unsteady incompressible viscous flows around moving rigid bodies using an immersed object method with overlapping grids", *J. Comput. Phys.*, **207**(1), pp. 151-172 (2005).

26. Xia, G.H., Zhao, Y. and Yeo, J.H. "Parallel unstructured multigrid simulation of 3D unsteady flows and fluid-structure interaction in mechanical heart valve using immersed membrane method", *Comput Fluids.*, **38**(1), pp. 71-79 (2009).

27. Ebrahimi, M. and Jahangirian, A. "Aerodynamic optimization of airfoils using adaptive parameterization and genetic algorithm", *J. Optimiz. Theory App.*, **162**(1), pp. 257-271 (2014).

28. Ahmdal, G. "Validity of the single processor approach to achieving large scale computing capabilities", In: *AFIPS Conference Proceedings, Thompson Books*, Washington DC. **30**, pp. 483-485 (1967).

Biographies

Mehdi Ebrahimi was born in 1980. He received his BSc in Mechanical engineering from Ferdwosi University of Mashhad and his MSc and PhD in Aerospace Engineering from Amirkabir University of Technology. His research interests are mainly in the areas of computational fluid dynamics, parallelization, evolutionary optimization, environmental flow calculations, and neural networks.

Alireza Jahangirian was born in 1964. He is working as Associate Professor at the Faculty of Aerospace Engineering in Amirkabir University of Technology. He received his BSc, MSc, and PhD degree in Mechanical Engineering from Amirkabir University of Technology, Sharif University of Technology, and University of Manchester, respectively. His research interests are mainly in the areas of computational fluid dynamics, numerical grid generation, aerodynamics optimization, and turbulence modeling.