The 10th International Scientific Conference
eLearning and software for Education
Bucharest, April 24-25, 2014
10.12753/2066-026X-14-081

DIFFERENT APPROACHES FOR SOLVING OPTIMIZATION PROBLEMS USING INTERACTIVE E-LEARNING TOOLS

Adrian FLOREA, Arpad GELLERT
"Lucian Blaga" University of Sibiu, Computer Science and Electrical Engineering Department, Emil Cioran Street, No. 4, 550025, Sibiu, Romania
adrian.florea@ulbsibiu.ro, arpad.gellert@ulbsibiu.ro

***Abstract**: Solving optimization problems, regardless of the scope, involves knowledge of mathematical apparatus based on the techniques and methods that are not always simple (differential calculus, operational research, etc.) and concepts of artificial intelligence, machine learning, evolutionary computing, graph theory. These problems are NP-complete and very often the optimization process targets more than one objective, at least two, and they can have an antagonist behavior. As an example, we can consider a simple car design: two objectives - cost (production cost or fuel consumption) that should be minimized and performance (speed limit or reliability) which are to be maximized. Or, if we talk about a microprocessor design the multi-criteria analysis must targets: high performance, small integration area, small energy consumption having also constraints about thermal dissipation. Given the above, becomes more difficult to teach optimization methods, to communicate new concepts and skills in an informative and formative manner, and in the same time to be attractive for students. Thus, developing effective e-learning tools targeting evolutionary algorithms in order to solve optimization problems is a continuous challenge. Moreover, technology applied to education became a key issue in nowadays knowledge society and education represents an essential element for knowledge improvement and economy growth. In this work we try to tackle the above challenges by developing the ETTOP tool (E-learning Tool for Teaching Optimization Problems) in order to gain a better understanding and familiarity of the students with new advanced learning methods and tools in the Evolutionary Computing domains, and especially in the optimization methods field. The main aim of our work consists in highlighting of different approaches for solving mono and multi-objective optimization problems using interactive e-learning tools (non-Pareto techniques, Pareto techniques and techniques based on swarm behavior). Although our software tool is designed and developed as an Application Programming Interface (API) that allows each user to select an existing problem or define a new one, to customize the solution algorithm based on problem specific constraints that the users can construct themselves or take over from sets of predetermined functions and rules, in this stage we present only three case studies that we implemented.*

***Keywords:** Evolutionary Computing; Multi-objective optimization; Design Space Exploration; E-learning; Education.*

I. INTRODUCTION

The motivation of the chosen topic for this article is two-fold. First, ICT students must be involved in the development of new software tools, combining efforts and expertise to address a rapidly changing education landscape, shifting from being simply consumers of the technology to being its creators. Second, the multimedia and communication technologies evolution facilitated the use of e-learning in education on a large scale. Furthermore, facilitating the public access to educational tools aimed to solve optimization problems in various domains became a necessity.

The classical educational approach in teaching evolutionary computing concepts is of limited applicability, being based largely on oral communication of professors. Although currently are available online many textbooks, scientific articles that describe and implement various evolutionary algorithms, however, a thorough understanding of the optimization methods based only on theoretical explanations or exemplified by pseudocode language, it is difficult from the perspective of a beginner student. Thus, teachers need to educate students so they gain technical and analytical skills for learning optimization methods based on evolutionary computation. Therefore, developing effective e-learning tools targeting evolutionary algorithms in order to solve optimization problems is a continuous challenge. Moreover, technology applied to education became a key issue in nowadays knowledge society and education represents an essential element for knowledge improvement and economy growth [12]. Modern students (also called "digital natives" [13] or "Net-generation") learn efficiently through discoveries individually or collaboratively with their colleagues. Rather they learn by doing and not by reading instructions from the manual or by listening to teachers [8].

Consistent with the previously expressed fundamental challenges, our developed application is designed to help not only in learning but also allows its extension by students, by proposing their own solutions to optimization problems in various fields (computer science, electronics, mechanical engineering, logistics, etc.). In this work we try to tackle the above challenges by developing the ETTOP tool (E-learning Tool for Teaching Optimization Problems) in order to acquire a better understanding and familiarity of the students with new learning methods and tools in the Evolutionary Computing domains, and especially in the optimization methods field. The main aim of our work consists in highlighting different approaches for solving single- and multi-objective optimization problems using interactive e-learning tools. The Net-generation students may benefit from our application since they can start from a certain type of optimization algorithm or from a certain type of problem and develop new solutions/ methods inside of the ETTOP tool, or define a new problem, customizing the solution algorithm based on problem specific constraints that the users can construct themselves or take over from sets of predetermined functions and rules. In this stage we present only three case studies that we implemented:

- Solving the Traveling Salesman Problem (TSP) (finding a shortest closed tour which visits all the cities in a given set).
- Optimizing geometric parameters for cylindrical machine parts.
- Developing an automatic Design Space Exploration (DSE) tool dedicated to the PowerPC superscalar architecture. In order to reduce the simulation time we selected the PSATSim simulator that offers as outputs both performance and energy dissipation statistics, allowing thus a multi-objective analysis [16].

We developed the following multi-objective optimization techniques:

1. The first approach uses *non-Pareto* optimization techniques: Weighted-Sum approach [5], the VEGA algorithm (Vector Evaluated Genetic Algorithm) [15].
2. The second approach uses Pareto optimization techniques: NSGA-II (*Non-dominated Sorting Genetic Algorithm*) [1].
3. The third approach uses bio-inspired optimization techniques based on swarm behavior: PSO (*Particle Swarm Optimization*) [4], SMPSO (*Speed-constrained Multi- objective PSO*) [10] and *ant colony optimization* (ACO) [2].

From a didactical point of view, the ETTOP tool has benefits in the learning process because it helps students to observe the influence of many parameters on the evaluated model. Some important parameters that may be varied are: the optimization technique, the features of the genetic algorithm (GA) such as the length and type of solution, the population size, the different variation operators – crossover, mutation – according to the representation of solution, the selection of parents and the survival selection, the termination condition, etc. The students can observe the algorithm which converges quicker to the optimal solution as well as the advantage introduced by Pareto optimization techniques opposite non-Pareto methods. They can also observe the influence of the number of simulated generations on the evaluated metrics. Because the evolutionary algorithms are stochastic, even the starting individuals may influence the obtained results and sometimes the convergence speed.

The organization of the rest of this paper is as follows. In section 2 we shortly review the related work in the field of software tools used in the teaching process of optimization techniques. Section 3 describes a short theoretical background related to multi-objective optimization algorithms,

whereas section 4 presents the application's software design. Section 5 illustrates some simulation results. Finally, section 6 suggests directions for future work and concludes the paper.

II. RELATED WORK

In this section we present some well-known e-learning systems that exploit the multi-objective optimization algorithms. In [6] the author presents a system for the resolution of combinatorial optimization problems under multiple objectives. It allows the user to define metaheuristics and test them on machine scheduling problems. The interactions of the user with the system are supported by a graphical interface. The results can be easily visualized and compared using different plots. In [14] the authors provide a methodology for teaching structural design optimization. A computer program that integrates a genetic algorithm based optimal design methodology, with structural analysis and design capabilities, is made available to the students. They solve the design problem using a combination of manual iterations and the semi-automated design features available in the program. In [9] the authors present a tool which enables the users to apply a certain nature analogous technique to a certain problem without constructing an own program. The tool consists of shells evolutionary algorithms, cellular automata, Boolean networks, artificial neural networks and fuzzy-expert-system. The shells are parameterized and allow introducing even problem specific rules. Unlike these works, our ETTOP tool can be used both in terms of teaching and research. It can run various optimization applications by simply mapping problem to a corresponding form of solution's representation, selecting then single- or multi-objective optimization algorithms. We consider that our work is in the European trend of Framework projects that target the research's extension in advanced education field using modern learning technologies [12].

III. MULTI-OBJECTIVE OPTIMIZATION ALGORITHMS

Usually, real-world problems have more than one objective function. Since these functions are often conflicting to each other, there are different optimal solutions corresponding to different objectives. Therefore, there is not only one optimal solution, rather a trade-off optimal solutions set, generally known as "*Pareto-Optimal*" solutions (named after the Italian economist Vilfredo Pareto [11]). None of these solutions can be considered to be better than any other with respect to all objective functions. Based on how the objectives are integrated within, the multi-objective algorithms may be classified as: non-Pareto, Pareto and Bio-inspired techniques. Since nearly all the algorithms are detailed in the literature we will only illustrate their role and also the differences between them.

3.1. Non-Pareto Techniques

In order to support the students and make the transition easier from single-objective optimization genetic algorithms to the multi-objectives, first we considered traditional techniques (non-Pareto). Among these, we have implemented: Aggregating approaches and VEGA algorithm.

Algorithms that fall in the *Aggregating approaches* category combine all the objectives of the problem into one single objective. The effect is that the multi-objective problem is transformed into a single-objective one and single-objective algorithms can be used. In order to achieve a set of solutions approximating the Pareto-optimal set it requires performing several runs with different parameter settings. Under this category we implemented the *Weighted-Sum* method [5]. The main disadvantages of this technique are the difficulty of choosing the weights assigned to objectives, and the impossibility of finding all solutions for problems having non-convex Pareto-optimal front.

The *Vector Evaluated Genetic Algorithm* (VEGA) [Sch84] represents the first multi-objective genetic algorithm built as an extension of the simple genetic algorithm with Roulette Wheel Selection mechanism. The whole population is randomly and equally divided to the number of objectives, each sub-population being subject to selection mechanism according with the assigned objective. After selection, these sub-populations would be shuffled together to obtain a new population of original size, on which the GA would apply the crossover and mutation operators in the usual way. The main drawback is that the population tends to converge towards solutions that are top in one objective, but very weak in others.

3.2. Pareto Techniques

The *NSGA-II* is a fast multi-objective algorithm with elitism introduced in [1] that removes some disadvantages of previous version (NSGA) such as high computational complexity for large populations and the diversity loss. Starting from a parent population, the algorithm performs crossover and mutation to obtain the offspring population. The two populations are combined into a single one on which is performed a non-dominated sorting to identify different Pareto fronts (elitism). The ranks and the crowding distance (density of individuals surrounding a particular one) guide further the selection process in order to form the next population. The binary tournament crowding selection mechanism considers an individual to be better than another if and only if it has a lower rank or, having the same rank, if it has a higher crowding distance. The diversity among non-dominated solutions is introduced by using the crowding comparison procedure during the population reduction phase. As disadvantage, the crowded comparison can restrict the convergence and also the non-dominated sorting is performed on double population size.

3.3. Bio-Inspired Techniques

The *PSO* technique is inspired by social behavior (particularly related to the transmission and sharing of information) of living beings such as flocks of birds, swarms of bees or birds, schools of fish [4]. The artificial search process is provided by a set of "particles" (similar with the chromosomes in GA) whose movement is characterized by a "velocities" that change over time depending on the characteristics of the entire system. These algorithms enable quick finding optimum but have difficulty in avoiding local minima. Population is called "swarm". These particles fly through the search space following the best particle at that time. The position of a particle is given by the current values of its parameters. Each particle tries to approach the best particles. To achieve this, its parameters are modified. In order to change the position, it is considering both the global best particle (leader) of the moment and the best local position it had in its history. After this change, the particle will have a new position and should be evaluated again. After all particles have been evaluated, a new leader is selected, each particle's personal best is updated and the process is restarted. In multi-objective algorithms can be more leaders at some point of time. Unlike GA, PSO has no variation operators such as crossover and mutation; through combination of local search methods with global search methods, PSO attempts to balance exploration and exploitation.

The *SMPSO* algorithm [10] allows producing new effective particle positions in those cases in which the velocity becomes too high using a velocity constriction mechanism. Somewhat similar to genetic algorithms, SMPSO uses an external archive to store the non-dominated solutions (leaders) found during the search and a density estimator (crowding distance) in the leader selection stage: binary tournament that choose the leaders taking into account the crowding distance. Other features of SMPSO include the use of polynomial mutation as a turbulence factor.

ACO is a heuristic technique inspired by biological systems for solving computational problems which can be reduced to finding good paths through graphs [2]. The ants are simple agents that, in TSP problem, build solutions moving from one city to another according to the graph problem, using artificial pheromones. It performs better against other global optimization techniques such as neural net, simulated annealing and can be used in dynamic applications.

IV. APPLICATION SOFTWARE DESIGN

4.1. ETTOP Software Architecture

Through this paper we intend to expand the jMetal.NET platform with new optimization techniques (non-Pareto, Pareto and bio-inspired approaches) highlighting the differences between them. The focus is not set on using the ETTOP tool but rather on understanding the software implementation details and the possibility of extending the framework with new problems to be solved, having different number of objectives, with new metaheuristics.

The software solution consists in two basic projects: *GAFramework* and *jMetal.NET*. The first contains the application graphical user interface (GUI) which allows selecting the algorithm's configuration class, the optimization problem, and facilitates viewing and comparing the obtained results and viewing the values of individuals and of objectives. The second project's goal was to provide C# implementation of jMetal (a Java framework for multi-objective optimization with

metaheuristics) [3] by porting this library on .NET platform. It contains a subset of algorithms and the way they are setup, a number of problems and their subsets of solution representations, and the genetic operators from original library.

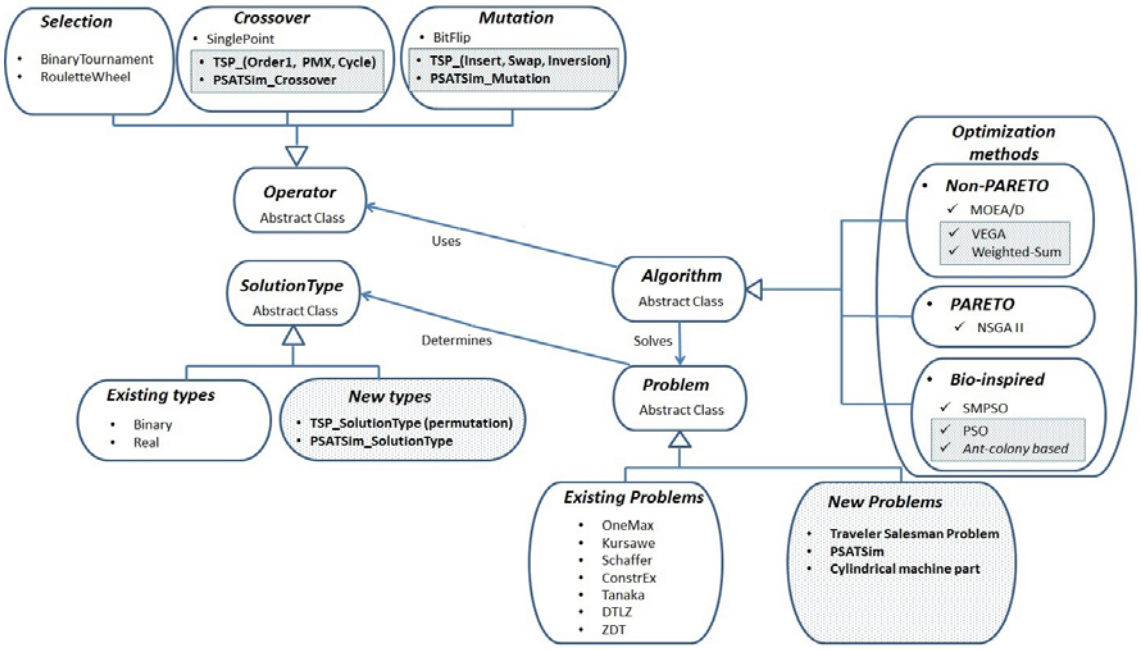

Figure 1. Application class diagram

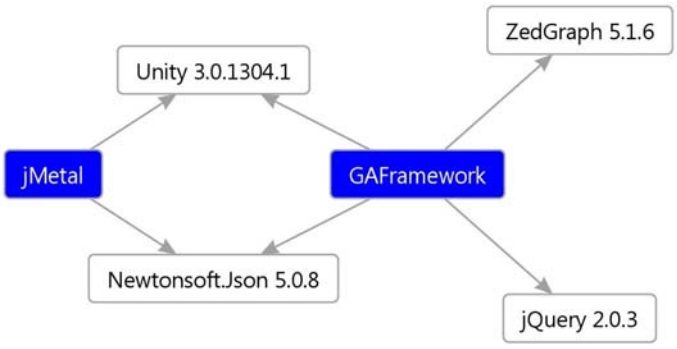

Figure 2. ETTOP Packages

In figure 1, in boxes, shaded, are illustrated the author's contributions in each class. First, the existing optimization algorithms NSGA-II and SMPSO were implemented in an efficient manner by parallel code sections (for simultaneous evaluation of the fitness function). Then, non-Pareto methods (VEGA and Weighted-Sum) and Bio-inspired (PSO and ACO) were developed. We proposed and solved new problems – TSP (single objective optimization), PSATSim and Cylindrical machine part (both of them being multi-objective optimization). For the first two, it was necessary to implement new representations for solutions that did not exist in jMetal.NET library (permutation and vector of integers). We implemented variation operators (crossover and mutation) specific for these two problems (see also figures 3a and 3b). Also, we implemented some quality metrics (Hypervolume, Epsilon and Spread [3]) that did not exist in jMetal.NET library.

Since genetic algorithms exhibit the feature to be embarrassingly parallel problems, requiring little or no effort to separate the problem into a number of parallel tasks, we parallelized the population

evaluation in order to apply parents' selection mechanism. This is possible using the Parallel class from .NET framework (*System.Threading.Tasks* namespace) that contains specific implementations of *for* and *foreach* loops using multiple threads. A notable difference is that *Parallel.For* is just a normal static method with three arguments, where the last argument is a delegate expression (lambda function which is inline defined) whose input parameter (the *i* variable) will be incremented automatically by the framework. This delegate captures the unchanged loop body, which makes it particularly easy to experiment with introducing concurrency into a program. For a correct parallel evaluation, it requires the elimination of some specific dependencies. *Parallel.For* must be applied to variables that are not affected in loop body by critical sections.

```
#region PARALLEL
        object sync = new object();
        Parallel.For(0, populationSize, i => {
                Solution newSolution = new Solution(Problem);
                Problem.Evaluate(newSolution);
                Problem.EvaluateConstraints(newSolution);
                lock (sync)
                        population.Add(newSolution);
        });             //end of Parallel.For
        evaluations += populationSize;
#endregion
```

The simulation results illustrate the advantages of multicore architectures. The execution speed in case of parallel evaluation decreases linearly with the number of cores. Because our application aims the educational aspects and not necessarily the research aspects and due to the time consuming iterative methods as the generation number increases, we limited this statistic to 4 generations and all the results that we reported further were made using parallel evaluation. In the table below we reported results on PSATSim optimization problem and NSGA-II algorithms, using a hardware platform with AMD Athlon II X4 630 processor, 2.8 GHz, 4 GB RAM internal memory.

ITERATIVE vs. PARALLEL CHROMOSOME EVALUATION

Generation	**1**	**2**	**3**	**4**
Iterative execution time [s]	145	290	443	574
Parallel execution time [s]	47	92	133	190

During the implementation phase we encountered some issues. Invalid input combinations caused the PSATSim simulator to crash without giving some insights. Thus, we had to implement restrictions (validity checks) after applying the crossover and mutation operators and in the population initialization phase. Also, because the expression of results is very important in assessing the correctness of program implementation, or at least to see if the intuition is confirmed, and, due to the multitude of generated results difficult to be visualized and compared, another issue was to implement a simple visualizer for one or many results (Pareto Fronts) in case of multi-objective optimization problems. Based on Newtonsoft.JSON and ZedGraph packages (see figure 2) we scan the RESULTS folder, open the selected files and serialized information in human-readable format allowing further 2D graphical representation or creating spreadsheets such as Excel to plot or other operations. The zoom (in / out) operations can be done using the mouse wheel and the pan operation can be done by clicking the mouse wheel and moving it. Clicking a point on the plot will result in displaying a popup that will contain the objective values of that point (chromosome) but also the variables (genes) that have produced that output (figure 4a).

4.2. ETTOP: GUI and System Requirements

The ETTOP tool was implemented in C#, Visual Studio 2012 Express edition. In order to take advantage by parallel evaluation of chromosomes when NSGA-II, SMPSO and VEGA metaheuristics are applied, hardware systems with larger number of cores are recommended.

Before running, the user should configure ETTOP in order to establish the problem that will be solved (see Figure 3), the objectives (their number and what is to be maximized or minimized), if

there are some domain constraints (what non-negotiable conditions must be met – restrictions that if are not accomplished will lead to infeasible solutions). With respect to the DSE dedicated to the PowerPC superscalar architecture, one restriction refers to the size of the instruction window that should be greater than the instruction fetch rate. Other restrictions were implemented according with [7]. Then, the user selects the optimization methods (Pareto, non-Pareto, bio-inspired) and the specific parameters (population size, generation number or other stopping criteria, mutation and crossover type and mutation and crossover probability). After choosing the problem, the program extracts the number of variables that encoded each solution and the number of objectives. Then they are printed on the main form (figure 3a).

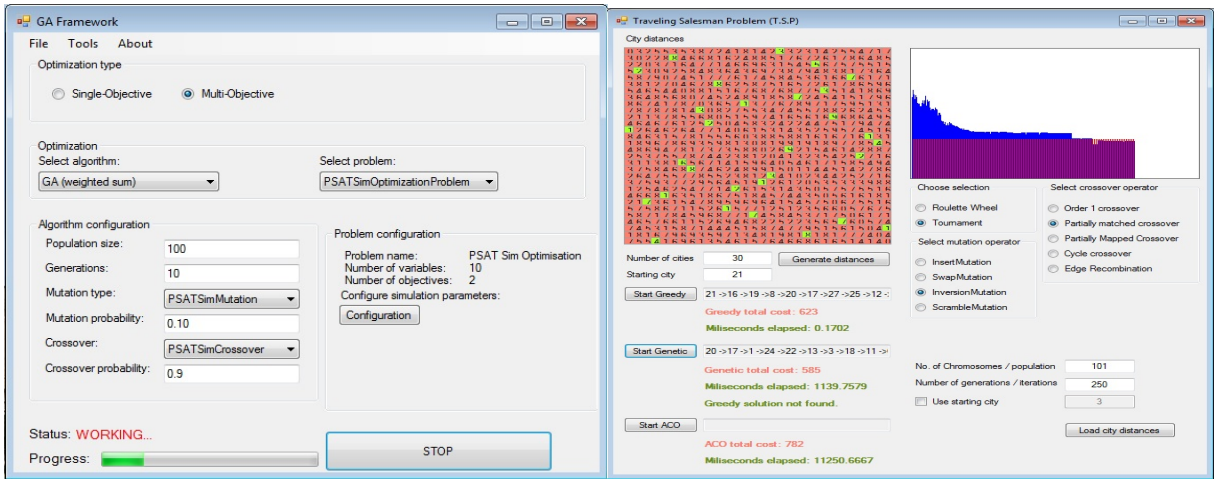

Figure 3. (a) User interface for configuring ETTOP and (b) Solving TSP through three different approaches (Greedy, GA and ACO)

Figure 3b illustrates the GUI that we obtain when from Figure 3a we select the radio button "*Single-Objective*" and we choose the "TSP – *Traveling Salesman Problem*" and then push the "*START*" button. In the upper left corner we have displayed the city distances. Under this panel, we have a section from where we can select the number of cities, a starting city and a button to generate random distances. Also, on the left side, we have 3 sections: Greedy, Genetic and ACO. In each one we have a button to start the algorithm, a list with the cities order, the cost and time. In the upper right corner we display the histogram with the evolution of Genetic/ACO algorithm against Greedy solution. On the right side, we have a section from where we can change the Selection method, Mutation and Crossover operators, the population size and the number of generations. In the bottom right corner, if we do not want to use random distances, we may load a file with predefined values.

V. SIMULATION RESULTS

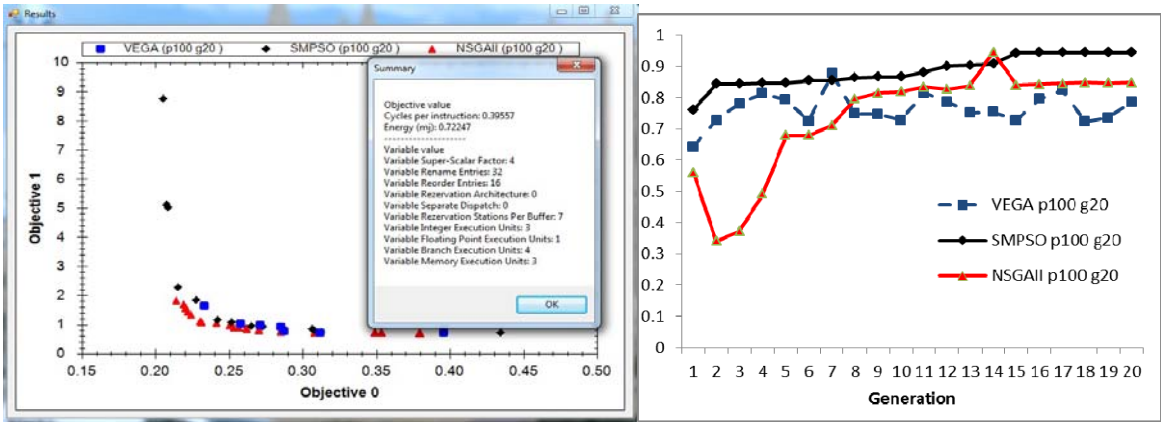

Figure 4. (a) Pareto fronts and (b) Hypervolume comparisons

In this section we present only few examples of results we may obtain with ETTOP. Figure 4a shows comparatively the obtained Pareto fronts after 20 generations by three runs (using NSGA-II, SMPSO and VEGA algorithms) of the PowerPC DSE problem. According to [17] the Pareto-optimal front should fulfill the following conditions: a good distribution of the obtained solutions and the size of the obtained non-dominated front should be maximized. As we can see, the less performing is the VEGA algorithm. The shuffling and merging of all subpopulations that VEGA performs involves averaging the fitness components associated with each of the objectives. The major drawback of objective switching in this case is that the population tends to converge to solutions which are superior in one objective, but poor in others. Regarding the NSGA-II and SMPSO analyzing the Pareto fronts we conclude that the differences are relatively small. The difference between algorithms comes in convergence speed, where the SMPSO performed the best since it has a great advantage over NSGA-II during first generations. Our claims are based both on our previous research work in microprocessor's DSE [7] and on the currently obtained results (hyper-volume metric – see figure 4b).

VI. CONCLUSIONS AND FURTHER WORK

Testing various scenarios for multi-objective optimization problems would require a lot of time and hardware resources in order to obtain satisfying results. Thus, through parallel evaluation of fitness we increased the speed of execution. Besides, our approach represents a formative necessity since teaching optimization methods is mainly approached in a descriptive manner. Through the GAFramework, students have the opportunity to be creative and innovative in Evolutionary Computing or in other research and didactical domains of computer science. Our developed software tool facilitates understanding the theoretical aspects, allowing students to feel more trustful when solving optimization problems. The ETTOP tool can be used by students and researchers with little knowledge in the field of Evolutionary Computing, because the user interaction with the system is supported by a graphical interface easily configured and extended.

For further work we are mainly concerned to solve the following issues: introducing other optimization methods – non-Pareto (Lexicographic ordering), Pareto (MOGA, SPEA2, etc) and mechanical design optimization problems. Also, we intend to attach to the GAFramework a database with test assignments, providing the possibility of testing the acquired knowledge.

Acknowledgements

We like to thank to our students George Dănilă and Sorin Gyorfi from Advanced Computing Systems specialization for providing useful help in the technical development of some algorithms.

References

[1] Deb, K., Pratap, A., Agarwal S., Meyarivan, T., A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation,* vol. 6, no. 2, pp. 182-197, 2002.

[2] Dorigo, M., Gambardella, L.M., Ant colony system: A cooperative learning approach to the Traveling Salesman Problem, *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997.

[3] Durillo, J.J., Nebro, A.J., Alba, E., The jMetal Framework for Multi-Objective Optimization: Design and Architecture, *Congress of Evolutionary Computing*, pp. 1-8, 2010.

[4] Eberhart, R.C., Kennedy, J., A new optimizer using particle swarm theory, In *Proceedings of the sixth international symposium on micro machine and human science*, vol. 43, New York, USA: IEEE, 1995.

[5] Gass, S., Saaty, T., Parametric Objective Function Part II. *Operations Research*, vol. 3, no. 4, pp. 316–319, 1955.

[6] Geiger, M.J., Teaching Modern Heuristics in Combinatorial Optimization, *Education for the 21st Century*, vol. 210 of IFIP, pp. 65-74, Springer, 2006.

[7] Gellert, A., Calborean, H., Vintan, L., Florea, A., Multi-objective optimisations for a superscalar architecture with selective value prediction, *IET Computers & Digital Techniques*, vol. 6, no. 4, pp. 205-213, 2012.

[8] Ivanovic, M., Jain, L.C., E-Learning Paradigms and Applications, Agent-based Approach, Eds. Springer, 2014.

[9] Klüver, C., Klüver, J., Soft Computing Tools for the Analysis of Complex Problems, *Proceedings of the First International Simulation Tools Conference & Expo*, Brussels, Belgium, 2013.

[10] Nebro, A., Durillo, J.J., Coello, C.A., Luna, F., Alba, E., SMPSO: A new PSO-based metaheuristic for multi-objective optimization, *IEEE Symposium Series on Computational Intelligence*, pp. 66–73, USA, 2009.

[11] Pareto, V., Cours D'Economie Politique, vol. I and II, F. Rouge, Lausanne, 1896.

[12] Peñalvo, F.J., Palacios, R.C., Lytras, M., Outcomes of International Research Projects on Technology Applied to Education, *Journal of Universal Computer Science*, vol. 18, no. 1, pp. 1-4, Austria, 2012.

[13] Prensky, M., Digital Natives, Digital Immigrants, Part II: Do They Really Think Differently?, *On the Horizon,* NCB University Press, vol. 9, no. 6, pp. 1-6, 2001.
[14] Rajan, S.D., Chen, S-Y., An Interactive Tool for Teaching Multiobjective Design Optimization, *Proceedings of 13th Conference on Analysis and Computation, Structural Engineering World Congress*, San Francisco, July, 1998.
[15] Schaffer, J.D., Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, PhD thesis, Vanderbilt University, Nashville, Tennessee, 1984.
[16] Smullen, W., Taha, T., PSATSim: An Interactive Graphical Superscalar Architecture Simulator for Power and Performance Analysis, *Proceedings of the Workshop on Computer Architecture Education*, 2006.
[17] Zitzler, E., Deb, K., Thiele, L., Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, vol. 8, no. 2, pp. 173-195, 2000.