DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch139

A hybrid genetic/immune strategy to tackle the multiobjective quadratic assignment problem

Arnaud ZINFLOU[1], Caroline GAGNÉ[2] and Marc GRAVEL[2]

[1] Measurement and information systems division
Institut de recherche d'Hydro-Québec – IREQ
1800, Lionel-Boulet, Varennes, Canada, J3X 1S1
zinflou.arnaud@ireq.ca
[2] Université du Québec à Chicoutimi - UQAC
555 boulevard de l'Université, Chicoutimi, Canada G7H 2B1
{caroline_gagne,mgravel}@uqac.ca

Abstract

The Genetic Immune Strategy for Multiple Objective Optimization (GISMOO) is a hybrid algorithm for solving multiobjective problems. The performance of this approach has been assessed using a classical combinatorial multiobjective optimization benchmark: the multiobjective 0/1 knapsack problem (MOKP) [1] and two-dimensional unconstrained multiobjective problems (ZDT) [2]. This paper shows that the GISMOO algorithm can also efficiently solve the multiobjective quadratic assignment problem (mQAP). A performance comparison carried out using well-known published algorithms and shows GISMOO to advantage.

Introduction

Even today, most of the work on optimization treats problems with a single objective to optimize [3, 4]. However, practical contexts have a multiobjective nature inherent in various performance measures. For this type of problems, there is generally no ideal solution which gives optimality for all the objectives. This is why the optimal solution concept becomes less relevant and is replaced by Pareto-optimality, where we obtain a set of solutions giving us a compromise among the different objectives. The solutions cannot be prioritized except by the decision maker's preferences. Multiobjective problems also often have a very large feasible solution set and are characterized by repetitive decisions. Consequently, we can define two goals in multiobjective optimization: (i) to discover solutions as close to the Pareto-optimal solutions as possible; and (ii) to find solutions as diverse as possible in the solution set thereby obtained. Satisfying these two goals is a challenge for any multiobjective algorithm [5-10].

Many authors have concluded that a promising research model for multiobjective problem solving is that of metaheuristics adaptation [11, 12]. In fact, these algorithms are among the greatest achievements of modern operations research, especially in solving large and complex real problems [13-15]. According to Whitacre [13], the growing interest in metaheuristics in the last few years is due to the successful adaptation of these algorithms to specific problems and hybrid approaches.

Many of the algorithms proposed for multi-objective problems are Evolutionary Algorithms (EA) [6, 8, 16-19]. This is so, doubtlessly because EA's can traverse a large search space to generate an approximation of the Pareto-optimal front in a single optimization step [20]. One of these algorithms, proposed by [21], is a hybrid between a Genetic Algorithm (GA) and an Artificial Immune System (AIS). This approach, called GISMOO, has a small number of parameters to calibrate, is easy to implement, and has been shown to be efficient in solving classical benchmarks in both discrete and continuous optimization.

The goal of this paper is to deepen the understanding of the Quadratic Assignment Problem (QAP) from the Pareto viewpoint, by adapting the GISMOO algorithm to solve this problem. On the one hand, we mean to show that this is an interesting approach in the solution of the multiobjective Quadratic Assignment Problem (mQAP). On the other hand, because only a few workers have treated this problem from a Pareto viewpoint, we wish to compare the performance of GISMOO with that of other known algorithms.

The remainder of this paper is organized as follows. Section II briefly describes the mQAP. In Section III we present the GISMOO algorithm in order to solve the problem in a Pareto sense. Section IV details the numerical experiments carried out in this paper. Section V compares the experimental results of GISMOO with those of two other evolutionary algorithms well known in the literature: NSGAII and PMS^{MO}. The last section offers some concluding remarks.

The multiobjective Quadratic Assignment Problem (mQAP)

The Quadratic Assignment Problem (*QAP*), defined by Koopmans and Beckmann [22] in 1957, is one of the most well known and widely studied problems of combinatorial optimization. It consists of assigning n interconnected facilities (factories, warehouses, etc.) to n locations in such a way as to minimize the sum of the product flows over the distances. The problem can be formulated as follows:

$$Minimise\, C(\pi) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{\pi_i \pi_j} \quad (1)$$

where n is the number of locations/facilities, a_{ij} is the distance between location i and location j, b_{ij} is the flow between facilities i and j, and π_i is the location of facility i in the permutation $\pi \in \Omega$ where Ω is the problem's solution space. The QAP belongs to the class of NP-hard problems [23, 24].

The multiobjective Quadratic Assignment Problem (mQAP) was formalized in 2002 [25]. This particular extension of the QAP considers several flow matrices between any two facilities. The mQAP can therefore model the situation of installations where the management of several types of products is a concern. For example, the flow of doctors, patients, visitors, pharmaceutical products, equipment, etc., between different facilities can be considered in the implantation of a (new) hospital. The mQAP can be formalized by the following equations:

$$Minimise\, \{C(\pi)\} = \{C^1(\pi),...,C^m(\pi)\}, \pi \in \Omega \quad (2)$$

with

$$C^k(\pi) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b^k_{\pi_i \pi_j}, k = 1,...,m \quad (3)$$

where m is the number of objectives (i.e. flow types), $C(\pi)$ is the objective function vector to be simultaneously optimized, $b^k_{\pi_i \pi_j}$ is the k-th flow between the facilities π_i et π_j, and *minimizing* means finding all non-dominated points [23]. Some research focuses on the resolution of mQAP [26-31].

Genetic Immune Strategy for MultiObjective Optimization (GISMOO)

GISMOO is a hybrid Pareto GA/AIS algorithm which offers an original iterative process in two phases: a Genetic phase and an Immune phase. New solutions (also called descendants) are then obtained from the offspring creation using classical genetic operators and from the creation of clones according to the AIS cloning selection principle.

As for most multiple objective evolutionary algorithms, one of the main difficulties in solving multiple objective problems is the performance assignment. Indeed, the quality of a solution in multiobjective optimization depends on the evaluation of multiple incommensurable and often competing objective functions. Then, instead of dealing with the real objective functions the performance assignment of a Pareto EA can schematically be viewed as being composed of two factors: a *dominance factor* and an *isolation factor* [9]. The first factor evaluates the dominance level of a given solution in Pareto sense and the second factor measures the density of solutions surrounding a given solution. Even if the performance assignment of most Pareto EAs is done using these two factors, each algorithm calculates the two factors in different ways.

Regarding GISMOO, the dominance factor is calculated in two steps as proposed in [9]. The first step consists of assigning to each individual x of the Parent population (POP) combined with the Descendant population (Q), a strength $S(x)$ corresponding to the number of solutions dominated by x. A solution x is said to *dominate* a solution y if and only if $\exists i \in Z \mid f_i(x) < f_i(y)$, and $\forall j \in Z, j \neq i\ f_j(x) \leq f_j(y)$ where $f_i(\cdot)$ indicates the value of the solution for the objective i and Z indicates the number of objectives to minimize. Using the value of $S(x)$, the dominance factor of an individual x, called $R^+(x)$, is determined in GISMOO using Equation (4) below, where $\succ$ indicates a dominance relationship of y with respect to x.

$$R^+(x) = \begin{cases} \dfrac{S(x)}{1+2*S(x)} & if \sum_{y \in POP \cup Q,\, y \succ x} S(y) = 0 \\ \sum_{y \in POP \cup Q,\, y \succ x} S(y) & \end{cases} \quad (4)$$

In this way, for the non-dominated individuals x for which $\sum_{y \in POP_t \cup Q_t,\, y \succ x} S(y) = 0$, the dominance factor value is not equal to 0, but rather between 0 and 0.5 according to the number of solutions it dominates. For the non-dominated solutions, this computation of the dominance factor allows us to better take into account the distribution of dominated solutions of POP and Q within the search space. Thus, the computation of the dominance factor favours non-dominated solutions situated in the less well explored regions.

The isolation factor, for its part, is based on the space metric sp introduced by Schott [32], which measures the distance $Dist(x)$ between a given individual x and its closest neighbour y (with $x \neq y$) as indicated in Equation (5).

$$Dist(x) = \min_{y \in POP \cup Q} \left[\sqrt{(f_1(x) - f_1(y))^2 + ... + (f_Z(x) - f_Z(y))^2} \right] \quad (5)$$

It should be noted that, in GISMOO, the isolation factor value is not directly added to the dominance factor value. In fact, the fitness of an individual in GISMOO is obtained using the dominance factor. and in case of a tie between individuals, the tie is broken using the isolation factor.

GISMOO's outline is shown in Algorithm 1. The algorithm starts building an initial population (POP_0) of size N, randomly or using greedy heuristics.

The main loop of GISMOO (lines 3-21) begins with the Genetic phase (lines 4-8) and generates $N/2$ offspring. This phase consists of the classical operations of a GA: selection, crossover and mutation. Notice that the selection procedure used in GISMOO is a binary tournament selection. In addition, even if two offspring are created during the recombination, only the best of the two is added to the Descendant population Q. It is important to mention that no crossover probability is needed in the Genetic phase of GISMOO, because the number of offspring to generate is related to the Parent population size. However, a mutation probability (p_m) is used to determine whether the generated offspring will be mutated or not (line 6). The crossover operator used in the Genetic phase to solve the mQAP problem is a cycle crossover operator [33] as proposed by [27]

for the same problem. For the mutation operator we choose to use a classical inversion operator.

Algorithm 1 : Outline of GISMOO procedure

1: Create an initial Parent population POP_0 of size N
2: Initialize t to 1
3: **While** no stopping rules is invoked
4: **While** $|Q_t| < N/2$
5: Select and recombine P_1 and $P_2 \in POP_t$ to obtain E_1 and E_2
6: Mutate E_1 and E_2 according to mutation probability p_m
7: Evaluate E_1 and E_2 and add the best offspring found in Q_t
8: **End While** `Genetic Phase`
9: Rank non-dominated solutions of POP_t
10: **For** each non-dominated solution $x \in POP_t$ **do**
11: Calculate nb_clones_x to produce for x according to Eq. (6)
12: $cpt = 0$
13: **While** $cpt < nb\_clones_x$
14: Create 2 clones C_1^{clo} et C_2^{clo} using x
15: Create $C_1^{hyp\alpha}$ (and $C_2^{hyp\beta}$) by hyper-mutation $\alpha(\beta)$ on $C_1(C_2)$
16: Evaluate $C_1^{hyp\alpha}$ and $C_2^{hyp\beta}$ and add the best of the two in Q_t
17: $cpt = cpt + 1$
18: **End While**
19: **End For** `Immune Phase`
20: Copy the N first solutions of $POP_t \cup Q_t$ into POP_{t+1}
21: $t = t + 1$
22: **End While**

Thereafter, the Immune phase (lines 9-19) adds $N/2$ solutions to the Descendant population Q. These solutions are generated using the cloning selection principle introduced by De Castro and Timmis [34]. This principle is used to model the fact that only the best "antibodies" will proliferate in the population. In GISMOO, antibodies correspond to the non-dominated solutions of the current population POP. The number of clones to create for each non-dominated solution x (nb_clones_x) is related to the isolation factor as shown in Equation (6) :

$$nb\_clones_x = round\left[\left(Dist(x) * \frac{N}{2}\right) \Big/ \sum_{x=1}^{nbSolND} Dist(x)\right] \quad (6)$$

where $nbSolND$ indicates the total number of non-dominated solutions in the current population and $Dist(x)$ corresponds to the isolation factor of an individual x as shown previously. The *round* function returns a number rounded to the nearest integer.

We note that a non-dominated solution in a less crowded region of the search space will generate a greater number of clones. This calculation is then dynamically adjusted during the iterative process of the algorithm and does not require the setting of additional parameters.

Once the number of clones to generate for an individual x is calculated, we generate two copies (c_1^{clo}, c_2^{clo}) of x. Thereafter, the two clones are hyper-mutated using two different mutation operators α and β, depending on the problem to solve, in order to obtain two mutated clones ($c_1^{hyp\alpha}$, $c_2^{hyp\beta}$). The hyper-mutation rate to apply to all clones generated for an individual x is set according to their rank. Indeed, for the first ten individuals with the highest rank the hyper-mutation rate is set at Z, the number of objectives to minimize. For each ten individuals the hyper-mutation rate is increased by one until we reach all non-dominated solutions. The two mutation operators used to solve the mQAP problem are respectively a classical inversion and a simple swap. We then compare the two obtained solutions, only adding the best in the Descendant population Q. This process is repeated until the total number of clones to be generated is reached.

After the Genetic and Immune phases, an elitist replacement of the population in order to keep the N best solutions of the combined Parent and Descendant population (line 20) is made. The replacement strategy used is a (λ+μ) type of deterministic replacement where λ indicates the size of the Parent population and μ the size of the Descendant population. In the proposed approach, we have λ=μ=N. Finally, notice that GISMOO also used an archive in order to store non-dominated solutions found during the search. For a detailed description of GISMOO, the reader can consult Zinflou *et al*. [21].

Numerical experiments

The focus of the experiments was on comparing the performance of GISMOO to those of two state-of-the-art algorithms: NSGAII [6] and PMS^{MO} [9]. For a detailed description of these two evolutionary algorithms, the reader can respectively consult [6] [9].

Test problems

We used a set of 22 benchmark mQAP instances to test the performance of the three evolutionary multiple objective algorithms. These test instances were generated by Knowles and Corne [25] and are available at http://dbkgroup.org/knowles/mQAP/. The number of locations, number of objectives and the category of each instance are indicated in Table 1.

Experimental conditions

All the algorithms used in this work were implemented in C++ and compiled with Visual Studio .Net 2010, using the same main data structures in our implementation. All the algorithms use the same population size set at 100 individuals, and the same computational time for the same test instance. The computational times used for the instances with 10, 20, and 30 locations are set to 10, 20, and 30 seconds, respectively [27].

The parameter settings for NSGAII, PMS^{MO} and GISMOO are determined empirically according to our previous works [1, 21] and the work of [27]. For the three algorithms, the mutation probability (p_m) was set at 0.06. It is important to mention here that no crossover probability is required in the GISMOO algorithm. For the two other algorithms, the crossover probability was set at 1. The population size N is the same for all algorithms and was set at 100. To perform a recombination, all the algorithms in this paper use a cycle crossover operator [33]. As a mutation operator, each algorithm uses a classical inversion.

Problem	*Instance category*	*# of locations*	*# of objectives*
KC10-2fl-1uni*	Uniform	10	2
KC10-2fl-2uni*	Uniform	10	2
KC10-2fl-3uni*	Uniform	10	2
KC20-2fl-1uni	Uniform	20	2
KC20-2fl-2uni	Uniform	20	2
KC20-2fl-3uni	Uniform	20	2
KC30-3fl-1uni	Uniform	30	3
KC30-3fl-2uni	Uniform	30	3
KC30-3fl-3uni	Uniform	30	3
KC10-2fl-1rl *	Real-like	10	2
KC10-2fl-2rl *	Real-like	10	2
KC10-2fl-3rl *	Real-like	10	2
KC10-2fl-4rl *	Real-like	10	2
KC10-2fl-5rl*	Real-like	10	2
KC20-2fl-1rl	Real-like	20	2
KC20-2fl-2rl	Real-like	20	2
KC20-2fl-3rl	Real-like	20	2
KC20-2fl-4rl	Real-like	20	2
KC20-2fl-5rl	Real-like	20	2
KC30-3fl-1rl	Real-like	30	3
KC30-3fl-2rl	Real-like	30	3
KC30-3fl-3rl	Real-like	30	3

Table 1: Characteristics of the test suites used

The computational experiments were run on a HP Z600 workstation with 2.13 GHz quad core Intel Xeon processor and 4 Gb of RAM, with Windows XP. Each instance was solved 30 times for each algorithm with random seeds.

Performance assessment

The test procedure for performance assessment uses the generational distance (GD) metric [27]. This metric evaluates the average distance between the obtained non-dominated set (E) and the reference set P^* of Pareto-optimal solutions or a very good approximation. The GD metric is computed using the following equation:

$$GD(E,P^*)=\frac{1}{|E|}\sum_{u\in E}\min\{dist(u,v)\mid v\in P^*\} \qquad (7)$$

where $dist(u,v)$ is the Euclidean distance (in objective spaces) between the solution $u \in E$ and the nearest member v in P^* and $/E/$ represents the cardinality of the set E. The smaller the value of this metric, the better the convergence toward the Pareto-optimal front. When all obtained solutions lie exactly on P^* chosen solutions, this metric has a value of zero.

The GD allows us to compare the preliminary results obtained by GISMOO to those of well known algorithms. However, it is obvious that additional performance assessments are needed in multiobjective optimization in order to prove conclusively the superiority of an algorithm. In this paper, the reference sets for the 10 location instances are the true Pareto front available at http://dbkgroup.org/knowles/mQAP/. From the approximation sets found by the three algorithms for each 20 and 30 location instances, the set containing only non-dominated solutions was computed and used as the reference set.

In addition to GD, performance assessment was also undertaken using PISA and the guidelines from [35] and [36]. Consider for example the comparison between GISMOO and PMS^{MO} on a problem. First, the bounds of approximation sets of both algorithms were calculated so that the approximation sets could be normalized to the interval [1, 2]. After that, a dominance rank was calculated for each of the 60 approximation sets by simply counting the number of approximation sets that are better than the observed one. The Mann-Whitney rank sum test was used to discover if there are significant differences between the dominance ranks of the two algorithms.

Results and discussion

Table 2 presents the average GD values obtained by each algorithm for the 22 mQAP instances. The first column of each table indicates the name of the problem. In this column a "*" right after the name of the problem denotes the problems on which the GD was calculated using true Pareto optimal solutions, while "†" indicates that we used an approximation set to calculate the GD values. The following columns respectively show the average results over 30 runs for each algorithm. With regard to the average best results obtained, they are indicated by a shaded area.

Table 2 shows that GISMOO globally outperforms the other algorithms. Indeed, GISMOO's GD metric is better than that of the other two algorithms in every one of the 22 tested instances.

Problem	Algorithm		
	PMS^{MO}	NSGAII	GISMOO
KC10-2fl-1uni*	1703.65	2248.78	0
KC10-2fl-2uni*	4671.3	10032.7	0
KC10-2fl-3uni*	240.129	321.019	56.4565
KC20-2fl-1uni†	9064.52	15357.5	2329.48
KC20-2fl-2uni†	19008.3	30414.5	8999.31
KC20-2fl-3uni†	1751.35	3095.25	551.016
KC30-3fl-1uni†	6212.65	12861.4	2405.68
KC30-3fl-2uni†	23244.3	38251.6	5880.51
KC30-3fl-3uni†	3400.69	5398.18	890.908
KC10-2fl-1rl *	36076.6	83310.2	4534.06
KC10-2fl-2rl *	85563.9	104218	0
KC10-2fl-3rl *	18781.2	84171.2	2014.34
KC10-2fl-4rl *	5578.69	16619.8	0
KC10-2fl-5rl*	58147.3	99401.8	2797.36
KC20-2fl-1rl †	415416	694862	64192.5
KC20-2fl-2rl †	206671	345675	29807.8
KC20-2fl-3rl	162788	245988	16266.3
KC20-2fl-4rl†	371847	706505	84916.1
KC20-2fl-5rl†	1373590	2529950	222566
KC30-3fl-1rl†	172467	314176	51039.8
KC30-3fl-2rl†	242664	522926	56675.9
KC30-3fl-3rl†	195027	250777	23985.2

Table 2: Average GD values of non-dominated solutions found by PMS^{MO}, NSGAII and GISMOO in 30 runs

In Table 3 and 4, we summarize the outcome of the Mann-Whitney rank sum tests for a significance level α of 0.0125. In these tables, the "↑ p-value" (↓ p-value) denotes the problems, on which GISMOO is significantly better (worse) than the other algorithm in comparison, while "↔" indicates there are no significant differences between the two algorithms.

Problem	
KC10-2fl-1uni*	↑
KC10-2fl-2uni*	↑
KC10-2fl-3uni*	↑
KC20-2fl-1uni†	↑
KC20-2fl-2uni†	↑
KC20-2fl-3uni†	↑
KC30-3fl-1uni†	↑
KC30-3fl-2uni†	↑
KC30-3fl-3uni†	↑
KC10-2fl-1rl *	↑
KC10-2fl-2rl *	↑
KC10-2fl-3rl *	↑
KC10-2fl-4rl *	↑
KC10-2fl-5rl*	↑
KC20-2fl-1rl †	↑
KC20-2fl-2rl †	↑
KC20-2fl-3rl	↑
KC20-2fl-4rl†	↑
KC20-2fl-5rl†	↑
KC30-3fl-1rl†	↑
KC30-3fl-2rl†	↑
KC30-3fl-3rl†	↑

Table 3: Outcomes of the Mann-Whitney rank sum test on dominance ranking for GISMOO and NSGAII

Problem	
KC10-2fl-1uni*	↑
KC10-2fl-2uni*	↑
KC10-2fl-3uni*	↑
KC20-2fl-1uni†	↑
KC20-2fl-2uni†	↑
KC20-2fl-3uni†	↑
KC30-3fl-1uni†	↑
KC30-3fl-2uni†	↑
KC30-3fl-3uni†	↑
KC10-2fl-1rl *	↑
KC10-2fl-2rl *	↑
KC10-2fl-3rl *	↑
KC10-2fl-4rl *	↑
KC10-2fl-5rl*	↑
KC20-2fl-1rl †	↑
KC20-2fl-2rl †	↑
KC20-2fl-3rl	↑
KC20-2fl-4rl†	↑
KC20-2fl-5rl†	↑
KC30-3fl-1rl†	↑
KC30-3fl-2rl†	↑
KC30-3fl-3rl†	↑

Table 4: Outcomes of the Mann-Whitney rank sum test on dominance ranking for GISMOO and PMS^{MO}

The Mann-Whitney rank sum test results also show that GISMOO globally outperforms the other two genetic algorithms. Indeed, GISMOO is never worse than NSGAII or PMS^{MO} in any of the 22 mQAP instances. In fact, GISMOO is significantly better than NSGAII and PMS^{MO} on all the instances. These results confirm the results obtained with the GD metric.

Beside the GD and Mann-Whitney rank sum test results, Fig. 1 and 2 graphically represent the solution sets found by GISMOO, NSGAII and PMS^{MO} on a typical run using respectively two ten location problems: KC10-2fl-1uni and KC10-2fl-1rl. The KC10-2fl-1uni is a uniform instance while KC10-2fl-1rl is a real-like instance. In these two particular examples, we notice that GISMOO globally outperforms NSGAII and PMS^{MO}. Indeed, in both cases the curves proposed by GISMOO dominated and are more extended than those of NSGAII and PMS^{MO}. We also notice that for these two problems the approximation sets found by GISMOO are very close to the true Pareto optimal solution. In fact, for the problem KC10-2fl-1uni all the solutions found by GISMOO are also in the Pareto. These graphs confirm the results of the GD metric and the ability of our approach to find diversified solutions and to efficiently explore the solution space.

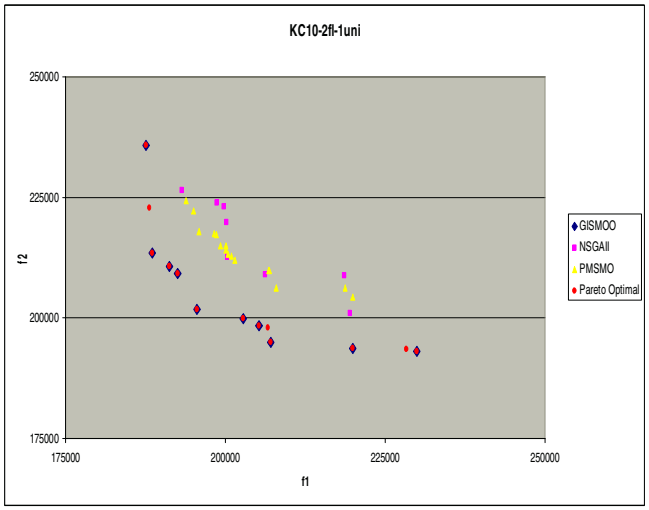

Figure 1: Non-dominated sets of a typical run of GISMOO, NSGAII and PMS^{MO}

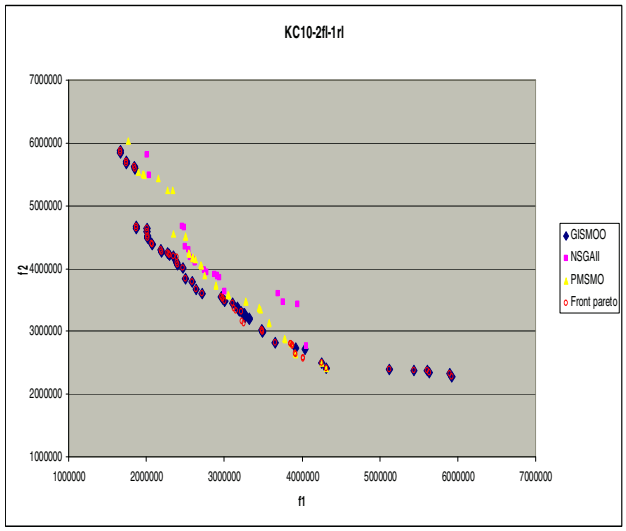

Figure 2: Non-dominated sets of a typical run of GISMOO, NSGAII and PMS^{MO} on the KC10-2fl-1rl problem

Conclusion

In this paper, we compared our GISMOO algorithm with two well known multiobjective algorithms (NSGAII, PMS^{MO}) on twenty-two state-of-the-art benchmark mQAP problems. The biggest difference between GISMOO and other two algorithms lies in the environmental selection and the way in which the immune metaphor is used in a Pareto GA to identify and emphasize the solutions located in less crowded regions found during the iteration process of the algorithm. The preliminary experimental results obtained on the benchmark problems considered here have shown that our approach is efficient: on every problem, GISMOO outperformed the other algorithms with regard to the quality indicators used here. These results confirm those obtained in [21] and [2] respectively for other combinatorial and continuous optimization problems.

On the basis of these results we suggest that GISMOO is an efficient tool to solve the mQAP problem. In future work, we will seek to extend the numerical experiments to other performance metrics. We will also seek to extend the application field of GISMOO to other multiobjective problems in industry and elsewhere.

References

[1] A. Zinflou, C. Gagné, and M. Gravel, "Solving multiple-objective optimization problems using GISSMO algorithm," in *World Congress on Nature and Biologically Inspired Computing (NABIC 2009)*, Coimbatore, India, 2009, pages 239-244.

[2] A. Zinflou, C. Gagné, and M. Gravel, "GISMOO on Continuous Multiple-Objective Problems: a Comparison Study," in *Proceedings of the 2010 International Conference on Genetic and Evolutionary Methods (GEM'10)*, Las vegas, 2010, pages 196-202.

[3] T. Loukil, J. Teghem, and D. Tuyttens, "Solving multi-objective production scheduling problems using metaheuristics," *European Journal of Operational Research,* vol. 161, pages 42-61, 2005.

[4] R. Ruiz and J. A. Vázquez-Rodríhuez, "The hybrid flow shop scheduling problem," *European Journal of Operational Research,* vol. 205, pages 1-18, 2010.

[5] A. C. Coello Coello and G. T. Pulido, "Multiobjective Optimization using a Micro-genetic Algorithm," in *In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), pages 274-282*, San Francisco, California, 2001.

[6] K. Deb, "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multiobjective Optimization : NSGA II," in *Parallel problem Solving form Nature – PPSN VI, M. Schoenauer et al. (Eds)*, Springer Lecture Notes in Computer Science, pages 849-858, 2000.

[7] J. D. Knowles and D. W. Corne, "The Pareto Archived Evolution Strategy : A New Baseline Algorithm for Multiobjective Optimisation," in *Congress on Evolutionary Computation*, Washington, 1999, pages 98-105.

[8] J. D. Knowles and D. W. Corne, "The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization," in *In Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*, pages 839-848, 2000.

[9] A. Zinflou, C. Gagné, M. Gravel, and W. L. Price, "Pareto memetic algorithm for multiple objective optimization with an industrial application," *Journal of Heuristics,* vol. 14, pages 313-333, 2008.

[10] E. Zitzler, "Evolutionary Algorithms for Multiobjective Optimization," in *EUROGEN 2001 - Evolutionary Methods for Design*, Optimisation and Control with Applications to Industrial Problems., 2001.

[11] J. Teghem and A. Jaszkiewicz, "Multiple objective metaheuristics for combinatorial optimization: a tutorial," in *the 5th Metaheuristics International Conference* Kyoto, Japon, 2003.

[12] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spektrum,* vol. 22, pages 425-460, 2000.

[13] J. M. Whitacre, "Survival of the flexible: explaining the recent popularity of nature-inspired optimization within a rapidly evolving world," *Computing,* vol. 93, pages 135-146, 2011.

[14] H. Zang, S. Zang, and K. Hapeshi, "A Review of Nature-Inspired Algorithms," *Journal of Bionic Engineering,* vol. 7, pages S232-S237, 2010.

[15] U. Derigs and S. Voß, "Meta-Heuristics - Theory, Applications and Software " *Annals of Operations Research,* vol. 131, pages 17-20, 2004.

[16] V. Cutello, G. Narzisi, and G. Nicosia, "A Class of Pareto Archived Evolution Strategy Algorithms using Immune inspired Operators for Ab-Initio Protein Structure Prediction," in *Third European Workshop on Evolutionary Computation and Bioinformatics, EvoWorkshops 2005, EvoBio 2005*, Lausanne, Switzerland, 2005, pages 3449.

[17] V. Cutello, G. Narzisi, and G. Nicosia, "A Multi-Objective Evolutionary Approach to the Protein Structure Prediction Problem," *Journal of the Royal Society Interface,* vol. 3, pages 139-151, 2006.

[18] J. D. Knowles and D. W. Corne, "M-PAES : A Memetic Algorithm for Multiobjective Optimization," in *In Proceedings of the 2000 Congress on Evolutionary Computation*, 2000, pages 325-332.

[19] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2 : Improving the Strength Pareto Evolutionary Algorithm," Technical Report 103, Computer Engineering and Communication Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland.2001.

[20] D. Francisci, "Algorithmes évolutionnaires et optimisation multi-objectifs en data mining," Laboratoire informatique, signaux et systèmes de Sophia Antipolis UMR 60702002.

[21] A. Zinflou, C. Gagné, and M. Gravel, "GISMOO: A New Hybrid Genetic/Immune Strategy for Multiple Objective Optimization," *Computers & Operations Research,* vol. 39, pages 1951-1968, 2012.

[22] T. C. Koopmans and M. J. Beckmann, "Assignment problems and the location of economic activities," *Econometrica,* vol. 25, pages 53-76, 1957.

[23] E. Çela, *The Quadratic Assignment Problem - Theory and Algorithms*. Boston, MA: Kluwer Academic Publishers, 1998.

[24] M. S. Garey and D. S. Johnson, *Computer and Intractability : A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman and Co., 1979.

[25] J. Knowles and D. Corne, "Instance generators and test suites for the multiobjective quadratic assignment problem," presented at the Evolutionary Multi-Criterion Optimization, Second International Conference, EMO 2003, Faro, Portugal, 2003.

[26] R. O. Day, M. P. Kleeman, and G. B. Lamont, "Solving the Multi-objective Quadratic Assignment Problem Using a fast messy Genetic Algorithm.," in *Congress on Evolutionary Computation (CEC'2003)*, Piscataway, New Jersey, 2003, pages 2277-2283.

[27] H. Li and D. Landa-Silva, "En Elitist GRASP Metaheuristic for the Multi-objective Quadratic Assignment Problem," in *5th International Conference on Evolutionary Multi-Criterion Optimization*, Heidelberg, 2009, pages 481-494.

[28] J. D. Knowles and D. W. Corne, "Towards landscape analyses to inform the design of hybrid local search for the multiobjective quadratic assignment problem," in *Soft Computing Systems - Design, Management and Applications (HIS 2002)*, 2002, pages 271–279

[29] M. P. Kleeman, R. O. Day, and G. B. Lamont, "Analysis of a Parallel MOEA Solving the Multi-objective Quadratic Assignment Problem," in *GECCO*, 2004, pages 402-403.

[30] R. O. Day and G. B. Lamont, "Multiobjective Quadratic Assignment Problem Solved by an Explicit Building Block Search Algorithm - MOMGA-IIa," in *Evocop 2005*, 2005, pages 91-100.

[31] L. Paquete and T. Stützle, "A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices," *European Journal of Operational Research,* vol. 169, pages 943–959 2006.

[32] Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," Master's thesis, Department of Aeronautics and Astronautics Massachusetts Institute of Technology, 1995.

[33] I. M. Oliver, D. J. Smith, and J. R. C. Holland, "A study of permutation crossover operators on the traveling salesman problem," in *the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, Hillsdale, NJ, USA, 1987, pages 224–230.

[34] L. N. De Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. London, 2002.

[35] J. D. Knowles, L. Thiele, and E. Zitzler, "A tutorial on the performance assessment of stochastic multiobjective optimizers.," TIK-Report No. 214, Computer Engineering and Networks Laboratory2006.

[36] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. D. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation,* vol. 7, 2003.