Málaga, 22 de noviembre de 2008

Informe Ejecutivo

Título:	MULTIOBJ-1.0: Técnicas de Optimización Multi-objetivo y Problemas Escalables en el Número de Variables
Resumen:	Este documento tiene como objetivo estudiar la escalabilidad de un conjunto técnicas de optimización multi-objetivo cuando aumenta el número de variables de decisión del problema a resolver. Se han estudiado problemas con distinta topología, y con un número máximo de hasta 2048 variables. Las técnicas estudiadas son tres algoritmos genéticos (NSGA-II, SPEA2 y PESA-II), una estrategia evolutiva (PAES), un algoritmo genético celular (MOCell), un algorítmo basado en cúmulo de partículas (OMOPSO), una evolución diferencial (GDE3) y un algoritmo basado en búsqueda dispersa (AbYSS).

Objetivos:

1. Estudiar la escalabilidad de un conjunto de técnicas de optimización multi-objetivo cuando se resuelven problemas desde 8 hasta 2048 variables.
2. Estudiar la velocidad, en término del número de evaluaciones necesarias, para converger al frente solución del problema.
3. Establecer un criterio para el estudio de la escalabilidad y convergencia de este tipo de técnicas.

Conclusiones:

1. OMOPSO y GDE3 son las técnicas más prometedoras:
   (a) Presentan un alto grado de esalabilidad
   (b) Han demostrado estar entre las más rapidas convergiendo hacia el frente solución del problema
2. Las técnicas de optimización más modernas, MOCell y AbYSS, son las que garantizan el mejor rendimiento tras OMOPSO y GDE3.
3. Técnicas clásicas como SPEA2, PESA-II y PAES son los algoritmos que peor escalan y necesitan un mayor número de evaluaciones para converger

Relación con entregables:

Málaga, November 22nd, 2008

Executive Summary

TITLE:	MULTIOBJ-1.0: Multi-Objective Optimization Techniques and Parameter Scalable Problems

ABSTRACT:

This report studies the behavior of a number of state-of-the-art multi-objective optimization metaheuristics when solving parameter scalable problems. Five scalable problems having different features have been considered, with numbers of variables ranging from 8 to 2048. The studied techniques are three genetic algorithms (NSGA-II, SPEA2, and PESA-II), an evolution strategy (PAES), a cellular genetic algorithm (MOCell), a particle swarm optimization metaheuristic (OMOPSO), a differential evolution (GDE3), and a scatter search (AbYSS).

GOALS:

1. Study the behaviour of a number of multi-objective optimizers when solving problems having 8, 16, 32, 64, 128, 256, 512, 1024, and 2048 variables.
2. Study the speed of the problems, measured in number of evaluations, to converge to the true Pareto front of the problems.
3. Define criteria to study scalability and convergence.

CONCLUSIONS:

1. GDE3 and OMOPSO are the most promising techniques:
   (a) They scale better than the rest of algorithms
   (b) Their convergence speeds are among the fastest.
2. MOCell, AbYSS and NSGA-II provide the best performance after GDE3 and OMOPSO.
3. Classical algorithms (SPEA2, PESA-II, and PAES) scale the worst, and they need higher number of evaluations to converge than the rest of solvers.

RELATION WITH DELIVERABLES:

PRE: AFP-1.2-2007 (mandatory reading)

CO: MO-1.5-2008 (advisable reading)

MULTIOBJ-1.0: Multi-Objective Optimization Techniques and Parameter Scalable Problems

DIRICOM

November 2008

1 Introduction

Many real world problems are multi-objective in nature (i.e, they have to optimize more than one conflicting objective at the same time), and they also usually tend to be nonlinear and with objective functions that are very expensive to evaluate. This situation has lead to the use of *metaheuristics* [1, 6] to deal with them. Metaheuristics are a family of optimization techniques comprising *Evolutionary Algorithms* (EA), *Particle Swarm Optimization* (PSO), *Ant Colony Optimization* (ACO), *Tabu Search*, *Differential Evolution* (DE), *Scatter Search* (SS) and many others.

The performance of these tecniques has been typically assessed by using benchmark problems, such as the Zitzler-Deb-Thiele (ZDT) test problems [15], the Deb-Thiele-Laumanns-Zitzler (DTLZ) benchmark [3], and the Walking-Fish-Group (WFG) test suite [7]).The methodology commonly adopted in the specialized literature is to compare several algorithms using a fixed (pre-defined) number of objective function evaluations and to compare the values of different quality indicators (e.g., *generational distance* [14] or *hypervolume* [17], among others).

The motivation driving us is that many real-world problems have hundred and thousand variables, and the aforementioned benchmarks have been normally adopted using a maximum of up to 30 variables. Thus, the studies currently available use not to consider the capability of current multi-objective metaheuristic algorithms to properly scale when dealing with a very large number of decision variables.

Another interesting issue that has been scarcely covered in the specialized literature is the analysis of the behavior of a multi-objective metaheuristic until reaching the true Pareto front of a problem. Typically, a fixed number of evaluations (and, in consequence, of iterations) is defined by the user, and the performance of the different algorithms studied is compared. However, this sort of comparison only measure the front aspect and does not provide any indication regarding the computational effort that a certain algorithm requires to reach the true Pareto front of a problem, i.e., the speed of the algorithm. We believe that this is an important issue because if we take into account that evaluating the objective functions of a MOP can be time-consuming, it becomes of interest to know how expensive is for a certain algorithm to reach the true Pareto front.

Our motivation in this work is to analyze the behavior of eight state-of-the-art multi-objective metaheuristics when solving a set of scalable parameter-wise problems, those comprising the ZDT benchmark, considering their formulation ranging from 8 up to 2048 variables. The considered algorithms are three genetic algorithms (NSGA-II [2], SPEA2 [16], and PESA-II [18]), an evolution strategy (PAES [8]), a particle swarm optimization technique (OMOPSO [13]), a cellular genetic algorithm (MOCell [12]), an differential evolution (GDE3 [9]), and a scatter search (AbYSS [10]).

2 Optimization Problems Addressed

To carry out our study, it would be helpful to use problems which are scalable in terms of the number of variables while keeping an invariable Pareto front. The ZDT test function family [15] fulfills this requirement. It offers, furthermore, a group of problems with different properties: convex, non-convex, disconnected, multi-frontal, and non-uniformly spaced. These problems have been widely used in many studies in the field since they were formulated.

Since we were interested in studying the behavior of the algorithms when solving scalable parameter-wise problems, we have evaluated each ZDT problem with 8, 16, 32, 64, 128, 256, 512, 1024, and 2048 variables. This way, we can study not only what techniques behaves better when solving problems having many variables, but also if their search capabilities remain constant or not when the number of problem variables arguments.

3 Algorithms

In this section we describe briefly the eight metaheuristics that we have considered in this study. We have used the implementation of these algorithms provided by jMetal [5], a Java-based framework aimed at multi-objective optimization[1].

The NSGA-II algorithm was proposed by Deb *et al.* [2]. It is based on obtaining a new population from the original one by applying the typical genetic operators (selection, crossover, and mutation); then, the individuals in the two populations are sorted according to their rank, and the best solutions are chosen to create a new population. In the case of having to select some individuals with the same rank, a density estimation based on measuring the crowding distance to the surrounding individuals belonging to the same rank is used to get the most promising solutions.

SPEA2 was proposed by Zitler *et al.* in [16]. In this algorithm, each individual has assigned a fitness value that is the sum of its strength raw fitness and a density estimation. The algorithm applies the selection, crossover, and mutation operators to fill an archive of individuals; then, the non-dominated individuals of both the original population and the archive are copied into a new population. If the number of non-dominated individuals is greater than the population size, a truncation operator based on calculating the distances to the k-th nearest neighbor is used. This way, the individuals having the minimum distance to any other individual are chosen.

PESA-II [18] uses an internal population from which parents are selected to create new solutions, and a external population in where non-dominated solutions found are stored. This last population uses the same hyper-grid division of phenotype (i.e., objective funcion) space adopted by PAES to maintain diversity in which region-based selection is adopted. In region-based selection, the unit of selection is an hyperbox rather than an individual. The procedure consists of selecting (using any of the traditional selection techniques) a hyperbox and then randomly select an individual within such hyperbox.

PAES is a metaheuristic proposed by Knowles and Corne [8]. The algorithm is based on a simple (1+1) evolution strategy. To find diverse solutions in the Pareto optimal set, PAES uses an external archive of nondominated solutions, which is also used to decide about the new candidate solutions. An adaptive grid is used as density estimator in the archive. We have used a real coded version of PAES, applying a polynomial mutation operator.

OMOPSO (Optimized MOPSO) is a particle swarm optimization algorithm for solving MOPs [13]. Its main features include the use of the crowding distance of NSGA-II to filter out leader solutions and the use of mutation operators to accelerate the convergence of the swarm. The original OMOPSO algorithm makes use of the concept of ϵ-dominance to limit the number of solutions produced by the algorithm. We consider here a variant discarding the use ϵ-dominance, being the leader population obtained after the algorithm has finished the result of the execution of the technique.

GDE3 [9] starts with a population of random solutions, which becomes the current population. In each generation, an offspring population is created using the differential evolution operators; then, the current population for the next generation is updated using the solutions of both, the offspring and the current populations. Before continuing to the next generation, the size of the population is reduced using non-dominated sorting and a pruning technique based aimed at diversity preservation, in a similar way as NSGA-II, althouth the pruning used in GDE3 modifies the crowding distance of NSGA-II in order to solve some drawbacks when dealing with problems having more than two objectives.

MOCell [12] is a cellular genetic algorithm (cGA). As other multi-objective metaheuristics, it includes an external archive to store the non-dominated solutions found so far. This archive is bounded and uses the crowding distance of NSGA-II to keep diversity in the Pareto Front. We have used here an asynchronous version of MOCell, called aMOCell4 in [11], in which the cells are explored sequentially (asynchronously). The selection is based on taking an individual from the neighborhood of the current cell and another one chosen randomly from the archive. After applying the genetic crossover and mutation operators, the new offspring is compared with the current one, replacing it if better; in the case of both solution be non-dominated, the worst individual in the neighborhood is replaced by the current one. In this two cases, the new individual is inserted into the archive.

AbYSS is an adaptation of the *scatter search* metaheuristic to the multi-objective domain [10]. It uses an external archive similar to the one employed by MOCell. The algorithm incorporates operators of the evolutionary algorithms domain, including polynomial mutation and simulated binary crossover in the improvement and solution combination methods, respectively.

4 Experiments

We are interested in two goals: the behavior of the algorithms when solving the scalable ZDT benchmark and to know which algorithms are faster in reaching to the Pareto front. Given that the true Pareto fronts of the ZDT problem are known, a strategy could be to run the algorithms until they found them, but then it is possible that some of them never achieve the optimal front. Our approach is to establish a stopping condition based on the the *high quality* of the found Pareto front, and we have used the hypervolume [17] quality indicator for that purpose.

The hypervolume calculates the volume (in the objective space) covered by members of a non-dominated set of solutions Q for problems where all objectives are to be minimized. Mathematically, for each solution $i \in Q$, a hypercube v_i is constructed with a reference point W and the solution i as the diagonal corners of the hypercube.

[1]jMetal is freely available to download at the following Web address: `http://neo.lcc.uma.es/metal/`.

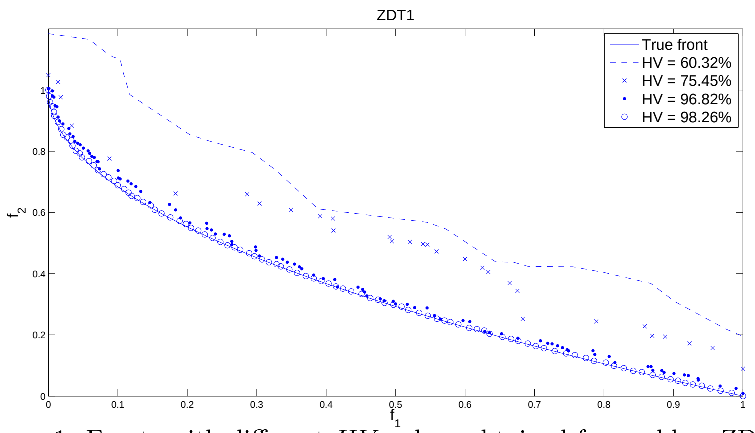

Figure 1: Fronts with different HV values obtained for problem ZDT1

The reference point can simply be found by constructing a vector of worst objective function values. Thereafter, a union of all hypercubes is found and its hypervolume (HV) is calculated:

$$HV = volume \left(\bigcup_{i=1}^{|Q|} v_i \right). \tag{1}$$

Higher values of the hypervolume metrics are desirable. A property of this quality indicator is that it measures both convergece to the true Pareto front and diversity of the obtained fronts.

Once the quality indicator we are going to use have been described, we need to establish a stopping condition to be used in the executions of the algorithms. The idea is that the metaheuristics stop when they reach a percentage of the HV of the true Pareto front which assures that the obtained front represents an accurate approximation to it. To decide about that percentage, we show different approximations of the Pareto front for the problem ZDT1 with different percentages of HV in Fig. 1. We can observe that a front with a hypervolume of 98.26% represents a reasonable approximation to the true Pareto fronts in terms of convergence and diversity of solutions. So, we have taken 98% of the hypervolume of the true Pareto front as a criterion to consider that a MOP has been successfully solved. Furthermore, those algorithms requiring fewer function evaluations to achieve this termination condition can be consider to be *faster*. In those situations in which an algorithm is unable to get a front fulfilling this condition after one maximum number of function evaluations, we consider that it has failed in solving the problem; this way, we can obtain a hit rate for the algorithms, i.e., their percentage of successful executions. We set the maximum of evaluations to ten million (500,000 were considered in [4]).

In our experiments, we check the stopping condition every 100 evaluations (that is, each iteration in the population based metaheuristics), where we measure the hypervolume of the non-dominated solutions found so far. Therefore, in NSGA-II, SPEA2, and GDE3 we have considered the non-dominated solutions in each generation, in PESA-II, PAES, AbYSS, and MOCell the external population and, in MOPSO, the leader archive.

The obtained results are summarized in Tables 1 and 2, which include a rank of the algorithms according to their scalability and speed, respectively.

Table 1: Ranking of the algorithms: scalability

ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	Global rank
1. GDE3	1. GDE3	1. GDE3	1. **MOCell**	1. OMOPSO	1. GDE3
2. MOCell	2. OMOPSO	2. AbYSS	2. **PESA-II**	2. GDE3	1. MOCell
3. SPEA2	3. MOCell	3. MOCell	3. **SPEA2**	3. AbYSS	3. AbYSS
4. AbYSS	4. AbYSS	4. NSGA-II	4. **NSGA-II**	4. MOCell	4. OMOPSO
5. OMOPSO	5. SPEA2	5. SPEA2	5. **PAES**	5. NSGA-II	4. SPEA2
6. NSGA-II	6. NSGA-II	6. PESA-II	6. **AbYSS**	6. SPEA2	6. NSGA-II
7. PESA-II	7. **PAES**	7. OMOPSO	7. **GDE3**	7. PESA-II	7. PESA-II
8. **PAES**	8. **PESA-II**	8. **PAES**	8. **OMOPSO**	8. **PAES**	8. PAES

Table 2: Ranking of the algorithms: speed

ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	Global rank
1. OMOPSO	1. OMOPSO	1. GDE3	1. MOCell	1. OMOPSO	1. GDE3
1. GDE3	2. GDE3	2. MOCell	2. PESA-II	2. GDE3	1. MOCell
3. MOCell	3. MOCell	3. NSGA-II	3. SPEA2	3. AbYSS	3. OMOPSO
4. PAES	4. AbYSS	4. OMOPSO	4. NSGA-II	4. MOCell	4. AbYSS
5. SPEA2	5. SPEA2	5. AbYSS	5. AbYSS	5. NSGA-II	5. NSGA-II
6. NSGA-II	6. NSGA-II	6. SPEA2	6. PAES	6. PESA-II	6. SPEA2
6. AbYSS	7. PESA-II	7. PESA-II	7. GDE3	7. SPEA2	7. PESA-II
8. PESA-II	8. PAES	8. PAES	8. OMOPSO	8. PAES	8. PAES

The scalabity ranking considers first those algorithms solving the problems with the highest number of variables. The ties are broken considering the number of evaluations in the most difficult instances. To make the discusion clearer, we have marked in boldface those optimizers having a hit rate lower than 1.0 in at least one experiment, which indicates that the algorithm does not scale well. According to this ranking, GDE3 is the most salient metaheuristic: it achieves two best and two second best ranks. However, given the difficulties of this algorithm when solving ZDT4, MOCell is the technique that appears as more reliable, in the sense that it is able to solve all the instances considered in this study but the largest one on ZDT4, and it occupies the first rank in ZDT4. OMOPSO scales the best in two problems (ZDT2 and ZDT6), but it is unable to solve ZDT4 and it tends to require more evaluations than other algorithms when solving the larger instances of ZDT1 and ZDT3. SPEA2, AbYSS, and NSGA-II are in the middle of the ranking: they never obtain the best result nor they are beyond the six position in the ranking. PESA-II is in the last positions mainly because it does not scale well in ZDT2 and ZDT4. Finally, PAES is the last algorithm in the ranking because of its low hit-rate in many experiments.

The ordering in Table 2 relies on the algorithms requiring globally the lower number of evaluations to find the target Pareto front, i.e., we sort them according their speed. To make this ranking, we consider all the instances, not only the largest ones. Thus, for each problem we have sorted the evaluations of the algorithms when solving each of the instances, and the sum of the obtained positions determine the order of the techniques. If we do not consider the ZDT4 problem, OMOPSO is globally the fastest algorithm: it requires the less number of evaluations in problems ZDT1, ZDT2, ZDT6, and it is the fourth one in the ranking of ZDT3. GDE3 is the second algorithm in the ranking, because it is first one in a problem, ZDT3, and the second one in ZDT1, ZDT2, and ZDT6. The next algorithms are MOCell (first rank in ZDT4, a second position, and two third ones), AbYSS, SPEA2 (the first GA in the speed ranking), and NSGA-II. Among the slowest metaheuristics we find again PESA-II and PAES. An interesting fact is that, if we observe the two tables, the rankings are the same in problems ZDT2, ZDT4, and ZDT6. This suggests that when an algorithm scales well with a problem, it may require a low number of function evaluations to solve it.

5 Conclusions

Our study have revealed differential evolution and particle swarm optimization are the most promising approaches to deal with the scalable problems used in this work. GDE3 and OMOPSO do not only scale well but they are among the fastest algorithms. Furthemore, we have shown that their search capabilities can be improved to solve ZDT4 (multi-frontal), the problem which has appeared as the most difficult one to solve.

Two modern optimizers, MOCell and AbYSS, have shown a high degree of regularity in the tests. With the exception of MOCell in ZDT4 (where it is the best technique), they are not in the first position in the scalabily and the speed rankings, but they are around the third and fourth positions. Both metaheuristics are in the group of algorithms having solved a higher number of instances, only failing again in ZDT4.

From the group of NSGA-II, SPEA2, and PESA-II, the first one is clearly the best in our experiments. In fact, NSGA-II is very close to MOCell and AbYSS in the rankings. SPEA2 and PESA-II have difficulties in ZDT2 and they use to be among the algorithms requering higher numers of function evaluations to reach the stopping condition.

Finally, PAES, the simplest of the optimizers in the study, is the algorithm scaling the worst, which is due to the low hit rates it obtains in many instances.

References

[1] C. Blum and A. Roli. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[3] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multi-Objective Optimization. Technical Report 112, Zurich, Switzerland, 2001.

[4] J.J. Durillo, A.J. Nebro, C.A. Coello Coello, F. Luna, and E. Alba. A comparative study of the effect of parameter scalability in multi-objective metaheuristics. In *CEC 2008*, Hong Kong, 2008.

[5] J.J. Durillo, A.J. Nebro, F. Luna, B. Dorronsoro, and E. Alba. jMetal: a Java Framework for Developing Multi-objective Optimization Metaheuristics. Technical Report ITI-2006-10, Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, E.T.S.I. Informática, Campus de Teatinos, 2006.

[6] F. W. Glover and G. A. Kochenberger. *Handbook of Metaheuristics.* Kluwer, 2003.

[7] S. Huband, L. Barone, R.L. While, and P. Hingston. A scalable multi-objective test problem toolkit. In C.A. Coello, A. Hernández, and E. Zitler, editors, *Third International Conference on Evolutionary MultiCriterion Optimization, EMO 2005*, volume 3410 of *Lecture Notes in Computer Science*, pages 280–295. Springer, 2005.

[8] J. Knowles and D. Corne. The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 9–105, Piscataway, NJ, 1999. IEEE Press.

[9] S. Kukkonen and J. Lampinen. GDE3: The third Evolution Step of Generalized Differential Evolution. In *IEEE Congress on Evolutionary Computation (CEC'2005)*, pages 443 – 450, 2005.

[10] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham. AbYSS: Adapting scatter search to multiobjective optimization. IEEE Transactions on Evolutionary Computation (In press), 2008.

[11] A.J. Nebro, J.J. Durillo, F. Luna, B. Dorronsoro, and E. Alba. Design issues in a multiobjective cellular genetic algorithm. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *Evolutionary Multi-Criterion Optimization. 4th International Conference, EMO 2007*, volume 4403 of *Lecture Notes in Computer Science*, pages 126–140. Springer, 2007.

[12] A.J. Nebro, J.J. Durillo, F. Luna, Bernabé. Dorronsoro, and Enrique Alba. A cellular genetic algorithm for multiobjective optimization. In David A. Pelta and Natalio Krasnogor, editors, *Proceedings of the Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO 2006)*, pages 25–36, Granada, Spain, 2006.

[13] M. Reyes and C.A. Coello Coello. Improving pso-based multi-objective optimization using crowding, mutation and ϵ-dominance. In C.A. Coello, A. Hernández, and E. Zitler, editors, *Third International Conference on Evolutionary MultiCriterion Optimization, EMO 2005*, volume 3410 of *LNCS*, pages 509–519. Springer, 2005.

[14] D. A. Van Veldhuizen and G. B. Lamont. Multiobjective Evolutionary Algorithm Research: A History and Analysis. Technical Report TR-98-03, Dept. Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol., Wright-Patterson, AFB, OH, 1998.

[15] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *IEEE Transactions on Evolutionary Computation*, 8(2):173–195, 2000.

[16] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 2001.

[17] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[18] D.W. Corne, N.R. Jerram, J.D. Knowles, and Martin J. Oates. Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In *Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 283–290. Morgan Kaufmann, 2001.