Food Regulated Pareto Multi-Species: a new ACO Approach for the Multi-objective Shortest Path Problem

Leonardo C. T. Bezerra *Elizabeth F. G. Goldbarg*
Marco C. Goldbarg *Luciana S. Buriol*

Technical Report - UFRN-DIMAp-2011-104-RT - Relatório Técnico
April - 2011 - Abril

Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte
www.dimap.ufrn.br

Food Regulated Pareto Multi-Species: a new ACO Approach for the Multi-objective Shortest Path Problem

Leonardo C. T. Bezerra [*]
leo.tbezerra@gmail.com

Elizabeth F. G. Goldbarg [†]
beth@dimap.ufrn.br

Marco C. Goldbarg [‡]
gold@dimap.ufrn.br

Luciana S. Buriol [§]
buriol@inf.ufrgs.br

27 de abril de 2011

Abstract. *The use of metaheuristics in Multi-objective Combinatorial Optimization, particularly Ant Colony Optimization (ACO), has grown recently. This paper proposes an approach where multi-species ants compete for food resources. Each species has its own search strategy and do not access pheromone information of other species. As in nature, successful ant populations are allowed to grow, whereas the others shrink. This approach is applied to the Multi-objective Shortest Path Problem and shows to inherit the behavior of succesful strategies from different types of problems. It is also compared to an existing ACO and to NSGA-II. Results show that the proposed approach is able to produce significantly better approximation sets than other methods.*

Keywords: Multi-objective Shortest Path, Ant Colony Optimization, Performance Assessment, Multi-species, Food Regulation

Resumo. *O uso de metaheurísticas na Otimização Combinatória Multiobjetivo, particularmente a Otimização por Colônias de Formigas (ACO), têm crescido recentemente. Neste trabalho, propõe-se uma abordagem onde múltiplas espécies de formigas competem por fontes de comida. Cada espécie usa uma busca estratégia de busca própria e não tem acesso ao feromônio das demais. Como na natureza, populações bem sucedidas crescem, enquanto as demais encolhem. Esta abordagem é aplicada ao Caminho mais Curto Multiobjetivo e mostra-se capaz de herdar o comportamento de estratégias bem sucedidas em diferentes problemas. Compara-se também tal abordagem com um ACO da literatura e com o NSGA-II, e o algoritmo proposto apresenta conjuntos de aproximação significativamente melhores que os demais.*

[*]Aluno do Programa de Pós-Graduação em Sistemas e Computação, UFRN
[†]Departamento de Informática e Matemática Aplicada, UFRN
[‡]Departamento de Informática e Matemática Aplicada, UFRN
[§]Departamento de Informática, UFRGS

Palavras-Chave: Caminho mais Curto Multiobjetivo, Otimização por Colônias de Formigas, Experimentação Multiobjetivo, Múltiplas espécies, Regulação por disponibilidade de comida

1 Introduction

Multi-objective Combinatorial Optimization Problems have drawn the attention of the scientific community due to their ability to represent real-world situations more appropriately than the original single objective models. For these problems a single solution will rarely be able to minimize (or maximize) all criteria, but rather there will be a set of compromise solutions. These are called *efficient non-dominated solutions* and are also referred to as *Pareto optimal set*. To each solution corresponds an objective vector composed of the objective values of each criterion. A set of objective vectors is called an *approximation front*, and the set of objective vectors corresponding to the solutions of the Pareto optimal set is called *Pareto front*.

Since many of the MOCO problems are NP-Hard [42], exact algorithms are expected to perform poorly as instances grow larger or the amount of objectives increases. This fact together with the broad applicability of the multi-objective models led to the development of metaheuristic approaches specially designed to deal with multi-criteria decision problems. These approaches can generate suboptimal sets of solutions that approximate the Pareto optimal set, partially retrieve the actual Pareto set, or both. When a metaheuristic is applied to a multi-objective problem, several design questions arise which may lead to many different approaches.

In this paper, one such metaheuristic approach is investigated, the Ant Colony Optimization (ACO). Recently, some papers have been published on the comparison of Multi-objective ACOs [30, 15, 31, 32] where some results show that the best approaches depend on the type of instance being tested. This paper introduces a new strategy for Multi-objective ACOs called food regulated Pareto multi-species. This metaphor consists of having multiple species of ants in a same algorithm, each using a different strategy which is expected to perform well on a particular type of problem. No interactions are allowed between the different species. However, since finding good solutions means finding food, the colonies either grow larger or shrink depending on their performances.

The proposed approach is applied to the well-known Multi-objective Shortest Path Problem (MSP) which has been proven to be NP-hard [42]. Exact and heuristic algorithms have been proposed for the MSP [14, 22, 39, 28, 35, 19, 16, 3] but most of them focus on instances with two objectives. In this work, a multi-objective ACO, called GRACE, which was recently proposed for the tri-criteria MSP [3] is extended. A first round of experiments is performed where different approaches for multi-objective ACOs found in literature are implemented on GRACE leading to different algorithmic versions. The existing approaches concern mainly different strategies for generation of scalarization vectors and pheromone update. To evaluate these approaches, two sets of 9 instances with different characteristics are used [37]. Pareto compliant indicators and statistical tests [27, 50] are used to determine which of the tested approaches is able to produce the best approximation sets. Results show that GRACE produces the best approximation sets for one set of instances and one of the implemented variations produces the best approximation sets for the other. Then, computational experiments are conducted using two species, and show that, while simply combining both approaches without food regulation jeopardizes the overall algorithm efficiency, the consideration of food regulation, allowing successful colonies to grow within the algorithm, produces approximation sets as good as those produced by the best single-species strategies. Finally, the new approach is compared to an ACO proposed previously for the MSP [19] and to an effective multi-objective evolutionary

approach named NSGA-II [6].

This paper is organized as follows. In Section 2, the ACO metaheuristic is revised and the different approaches found over literature are detailed. Section 3 describes the MSP and reviews its state-of-the-art. Section 4 presents the methodology utilized in the computational experiments. In Section 5, GRACE and its single-species variants are described and compared. The food regulated Pareto multi-species metaphor is described and tested in Section 6. Finally, conclusions and future work possibilities are pointed out in Section 7.

2 Ant Colony Optimization

Ant Colony Optimization (ACO) is a bio-inspired metaheuristic that uses the concept of *swarm intelligence*, i.e., the ability of groups to communicate even in the absence of a central coordination, through a stimulus that is at the same time physical and local. This phenomenon is called *stigmergy* [17], and in the case of ants, this stimulus is called *pheromone* [12]. When looking for food, ants are able to identify and perform pheromone deposits over the paths they tread in order to guide other ants to follow their trails. Once pheromone evaporates with time, shorter paths are more likely to be reinforced than longer ones. Hence, ants tend to converge to shorter paths. In an attempt to mimic this natural behavior, the *Ant System* (AS) metaheuristic was proposed [13], later improved into the *Ant Colony System* (ACS) [11].

In ACO algorithms, solutions are iteratively built by agents called *ants*. Each ant constructs a solution and is able to evaluate its current state, i.e., to calculate the objective value of its current solution (or partial solution). While building its solution, an ant makes decisions at each state based on information available for the possible choices. That information comes from the experience of other ants, by means of the amount of pheromone deposited on a given path, and from its own experience, by means of a heuristic. The probability that an ant makes the transition from state i to state j, $p(e_{ij})$, is given in Equation (1) where $\mathcal{N}_i$ denotes the set of states that are reachable from i, π_{ij} denotes the amount of pheromone between states i and j, η_{ij} denotes the heuristic value associated to the transition to state j, and α and β are parameters that weight the importance of the information that comes from the pheromone and the heuristic, respectively. Each time a solution is built, a pheromone *deposit* may be performed, which naturally *evaporates* over time.

$$p(e_{ij}) = \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{h \in \mathcal{N}_i} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Two major changes were proposed to improve AS into ACS. First, when evaluating transitions to choose the next state, ants may choose the best transition available instead of using stochasticity. The second change refers to pheromone update. In AS, all ants perform pheromone update altogether when they finish building their solutions. For the ACS, Dorigo and Gambardella [11] proposed a *local pheromone update*, i.e., each ant is allowed to deposit pheromone as soon as it finishes its own constructive procedure. The aim of this adaptation is to allow ants in a same iteration to use the "knowledge" achieved by the others, instead of searching based on the pheromone information of the previous iteration.

Proposals of ACO for multi-objective scenarios, called MOACO, were presented in [23, 18, 10, 8, 1]. Regarding the objectives, approaches sometimes might consider objectives separately or simultaneously. The focus of this paper is on simultaneous consideration of the objectives.

2.1 Multi-objective ACOs

Iredi *et al.* [23] proposed a multi-colony ant system for a scheduling problem. They utilize multiple colonies to search in different regions of the Pareto front. Each ant is associated to a scalarization vector which is used for weighting both heuristic and pheromone information. Once they deal with a bi-objective problem, two pheromone matrices are used, one for each objective. There are two possible strategies for selecting which ants should be allowed to update the pheromone information: (i) ants that find non-dominated solutions regarding local archives (one per colony), or; (ii) ants that find non-dominated solutions regarding a global archive. *et al.* Iredi *et al.* [23] affirm the first option leads to the same results as a multi-start approach, which is not the authors intention. Hence, the second strategy was adopted with two update methods: *update by origin* and *update by region*. In the update by origin method, ants are only allowed to update the pheromone matrix of their own colony. When the update by region strategy is used, non-dominated solutions found during a given iteration are sorted lexicographically and divided into r regions (r is the number of colonies). Each ant is, then, allowed to update the pheromone matrix of colony i, $1 \leq i \leq r$, where i stands for the region where the ant found its solution. The authors also propose the overlapping zones which consist of scalarization vectors common to more than one colony, promoting cooperation in the algorithm. The experimentation performed on six bi-criteria instances show the 10 colony version combined with the update by region strategy produces better results than the single colony algorithm. The authors also show that, depending on the amount of colonies used, allowing all ants to search the entire objective space performs worse than versions where the space of objectives is divided in disjoint intervals. Furthermore, using overlapping zones improves overall results. For two colonies, this difference is clearly perceived. When five colonies are used, it becomes less significant, and practically disappears for ten colonies.

Guntsch and Middendorf [18] proposed a population based ACO for the same scheduling problem used by Iredi *et al.* [23]. In their work, a population P is obtained from a super-population Q, by choosing the k (a parameter of the algorithm) closest solutions to a random solution π. The pheromone information is extracted from P: everytime a solution enters this set, a pheromone update is performed. When a solution leaves this set its influence is removed. Two pheromone matrices are used (one per objective). The authors report tests for population sizes 1, 3 and 5, concluding that the best approximation sets are generated with size 1 for a given bi-criteria instance. For a second bi-objective instance, population size 1 performs worse, whereas sizes 3 and 5 are roughly equal.

Doerner, Hartl and Reimann [10] proposed a MOACO based on the competition idea, and tested it for a bi-objective transportation problem. The COMPETants, as they are called, consist of a two-colony algorithm, each focusing on a specific criterion, and the concept of spies, i.e., ants that can analyze the pheromone information of the other colony, and decide whether to use it. At each iteration, ants choose which colony they will belong to. The lower the average quality solution of each colony, the greater its chance for receiving ants. In their work, the objective function is calculated as a weighted function of both criteria given by routing costs and fleet size. The authors claim that using spies and adaptive size colonies lead to better results than when these strategies are not used.

Doerner *et al.* [8] proposed the P-ACO, a population based ACO for the multi-objective portfolio selection that uses scalarizations for the constructive procedure of the ants. These weights are randomly generated from the [0,1) domain. Their algorithm uses the pseudo-random proportional rule, and the constructed non-dominated solutions are stored in a global archive. An ant is allowed to perform pheromone update only in case it finds either the best or the second best solution per objective of a given iteration. A local negative pheromone update

is also used and the algorithm has one pheromone vector for each criterion. A comparison was made with implementations of Pareto Simulated Annealing (PSA) [5] and NSGA [44] on 18 instances ranging from 5 to 10 objectives, with 20 and 30 projects. All tests used the number of solutions and the number of efficient solutions as metrics (the authors retrieved the Pareto set through an exact algorithm *a priori*). For instances with 20 projects, PSA is better than the other algorithms for small run times, whereas P-ACO outperforms the other two algorithms as this parameter increases. When the number of projects is increased to 30, P-ACO always produces better results than the rest of the optimizers. These results also hold true for a six objective real world instance with 30 projects. After 40 minutes of run time, PSA generates bigger approximation sets than P-ACO, but with less efficient solutions. Doerner *et al.* [9] improved this algorithm by adding a preprocessing Integer Linear Programming (ILP) procedure that allows the ACO to find supported efficient solutions, and to use these solutions to warm up the pheromone vectors. The authors affirm that using the whole set of solutions found in that stage for the warming of the vectors is better than using only the best and the second best per objective. They expanded their comparison to include an implementation of the Multi-objective Tabu Search [20] showing that P-ACO is able to outperform the former on the same real-world instance used in the previous experiment.

Alaya et al. [1] present four variants of a generic MOACO parameterized by the number of ant colonies and the number of pheromone trails. Given a problem with k objectives, the MOACO variant has k+1 colonies. k of these colonies focus on a unique objective. Each has one pheromone matrix and one source of heuristic information. There is also an extra colony which leads with all objectives simultaneously. This extra colony selects which pheromone information to use, aggregates the heuristic information from all other colonies, and updates all pheromone matrices. For the colonies focused on a single objective, only the best solution of the corresponding objective is allowed to update the corresponding pheromone matrix. The second MOACO version is similar to the first one, except by the extra colony that uses an aggregation of the pheromone information from all other colonies instead of choosing one at random. The third variant uses only one colony and one pheromone matrix, and the heuristic information is the sum of the heuristic information of all objectives. In this version only non-dominated solutions are allowed to perform pheromone update. Finally, the fourth variant uses one colony with one pheromone matrix per objective. The pheromone matrix to be used is randomly chosen at each step, and the heuristic information is calculated as in the third variant. The best solution per objective is allowed to update the corresponding pheromone matrix. The comparison held by the authors use the Multi-objective Knapsack Problem and several MOEAs. Based on the C metric [51], the authors conclude that the single colony strategies are generally better than the multi-colony ones. The performance of the fourth version is much similar to the third's, but the former generates better sets for some instances. The same happens between the first and second versions: they are initially alike, but as instances grow larger, the first version performs better. The fourth variant is also better than most compared MOEAs, but no significant difference is found when it is compared with SPEA [51].

2.2 Comparative Studies

López-Ibáñez, Paquete and Stützle [30] present a discussion on several strategies to deal with ACOs in the multi-objective context. They conduct an experimental investigation of different MOACO algorithms on the Bi-objective Quadratic Assignment Problem (bQAP) which are based on the $\mathcal{MIN}$-$\mathcal{MAX}$ Ant System ($\mathcal{MM}$AS) [47]. No heuristic information is used. Only the best solution per objective is allowed to perform pheromone update using the update by

region strategy (when multiple colonies are used). They also tested hybridizing the algorithms with local search procedures. Using instances generated by [25] and the binary and unary ϵ-indicators [52], the authors reported that results strongly depended on the correlation of the objectives, but the usage of local search procedures improved the overall algorithm performance. They also compared the ACO algorithms with the Robust Tabu Search algorithm [49], and showed that the latter produced better approximation sets for unstructured instances, whereas the opposite happened on structured ones.

López-Ibáñez and Stützle [31] present an experimental analysis of MOACOs for the Bi-objective Traveling Salesman Problem (bTSP). The authors include information from two heuristics (one per objective). If one pheromone matrix is used, each heuristic is weighted equally and summed. Otherwise, informations are weighted according to some ponderation, and summed. Experiments are carried on 3 Euclidean instances with 500 cities. They conclude that the correlation of the objectives does not interfere as strongly as in the bQAP and that the use of local search is, again, determinant for the optimizer's efficiency. They also observe that the use of the same scalarization vectors by all ants produces better results than when several vectors are considered in the same iteration. In the absence of local search, versions with a single pheromone matrix perform better in the central region of the objective space than versions with multiple matrices, whereas the latter ones obtain the best results at extreme regions. The use of multiple colonies improves results where the single versions did not perform well: extreme regions for one pheromone matrix and center region for multiple matrices.

García-Martínez, Cordón and Herrera [15] review several MOACO proposals classifying them according to the number of solutions returned, the number of heuristic information and the number of pheromone trails the algorithm maintains. The authors also implemented adaptations of many of these algorithms for the bTSP, and performed a comparison with other approaches which included the NSGA-II [6] and the SPEA2 [53]. They report that these MOEAs are faster than the MOACOs, but perform poorly in regard to dominance of solutions. Among the MOACOs, some generate well distributed approximation sets, sometimes dominated by sets generated by other optimizers that are concentrated on the center region of the objective space. The authors also report that the proposed taxonomy is not a determinant factor to the performance of the algorithm, since most MOACOs from the same families behaved differently. Angus and Woodward [2] present a theoretical study that aims to revise the taxonomy presented in [15]. The authors classify the existing algorithms according to number of pheromone matrices, how to use weights for the construction process, the ways of evaluating solutions, the methods used to update pheromone and how solutions are archived. No experimentation was performed.

Finally, López-Ibáñez and Stützle [32] also compared several existing MOACOs using the bTSP and proposed a general framework for designing this type of algorithm [29]. They argue that the comparison conducted in [15] did not target the specific differences between each algorithm, but rather which algorithm performed better, whereas [2] was only a theoretical work. The authors applied all the characteristics of the MOACOs on both ACS [11] and $\mathcal{MM}$AS [47]. For each algorithm, the authors analyze the use of different approaches. The proposed general framework for MOACOs [29] can vary regarding: (i) the number of colonies (N^{col}); (ii) pheromone update strategy (region or origin), maximum number of solutions allowed to perform update (N^{upd}), and selection strategy (non-dominated, best-of-objective or best-of-objective-per-weight); (iii) the number of weights per colony ($|\Lambda|$), and if all weights will be used at each iteration or if only one will be shared by all ants; (iv) number of pheromone ($[\pi]$) and heuristic ($[\eta]$) matrices, and; (v) how weights are used to aggregate different matrices (weighted sum, weighted product or random). The authors show that their automatically tuned MOACO outperforms the BicriterionAnt [23] for 3 bi-criteria Euclidean instances [29].

3 The Multi-objective Shortest Path Problem

The Multi-objective Shortest Path problem studied in this paper is a generalization of the classical point-to-point shortest path problem, and is presented by Raith and Ehrgott using a network flow formulation for two objectives [41]. In this work, we expand this formulation to deal with any number of objectives:

$$\min \quad z(x) = \begin{cases} z_1(x) = \sum\limits_{(i,j)\in A} c_{ij}^1 x_{ij} \\ \qquad\qquad ... \\ z_k(x) = \sum\limits_{(i,j)\in A} c_{ij}^k x_{ij} \end{cases} \tag{2}$$

$$\text{s.t.} \quad \sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = \begin{cases} 1 & \text{if } i = s, \\ 0 & \text{if } i \neq s, t \\ \text{-1} & \text{if } i = t, \end{cases} \tag{3}$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in A\,. \tag{4}$$

where s and t are, respectively, source and terminal nodes, c is a k-dimensional cost matrix for each edge (i, j), and z is the objective vector composed by k objective functions.

Shortest paths are a classical problem from graph theory and combinatorial optimization areas. Among several different real-world applications, routing scenarios are of particular interest. On the Internet, for example, determining the best route for sending a package following path-based protocols is an important step for ensuring routing efficiency. On navigation systems, which have become very common on popular vehicles, shortest paths help on both planning and optimization of the resources. On emergency situations, retrieving minimal routes is critical to the rescue and survival of citizens. Considering multiple (and sometimes conflicting) objectives has become a common approach on combinatorial optimization; many real-world situations present this characteristic. When traveling from a deposit to a deliver spot, for example, a truck has not only to consider time, but also traffic, conservation of the tracks and even the landscape, for it directly affects the amount of gas used. Analogously, a router determining the path packets should travel in order to reach their destination has to consider the bandwidth of the links besides of the number of intermediary nodes. Optimizing multiple objectives simultaneously, however, has an important drawback: the amount of solutions is expected to grow exponentially as more objectives are considered or larger instances are used. Raith and Ehrgott [41] state the BSP (Bi-objective Shortest Path problem) is considered to be intractable [21].

The MSP belongs to a class of problems that have polynomial algorithms for their original single versions. This allows optimizers to use a two-phase strategy that was shown to perform efficiently on the bi-objective context [40, 41]. This approach takes advantage of the fact that some of the solutions contained in the Pareto set can be retrieved with the help of *scalarizations*, i.e., by attributing weights to each objective. This means that part of the Pareto set can be retrieved polynomially. Having these solutions (or part of them) *a priori*, an optimizer can narrow the search space when looking for non-supported solutions. The literature on this problem includes exact and heuristic algorithms, which include evolutionary and ant colony optimizers. Raith and Ehrgott [41] recently compared different exact proposals [33, 43, 36] for the Bi-objective Shortest Path (BSP) problem, and showed that the two-phase approach is able to

solve instances up to 21000 nodes in less than 30 seconds, and up to 300000 nodes in less than 40 seconds. This algorithm, however, has not been expanded to handle problems with more than two objectives (this generalization is not trivial).

Among the Multi-objective Evolutionary Algorithms (MOEA), Mooney and Winstansley [35] use two populations (elitist approach), random walking (initial population), path encoding (representation by vertices), one-point crossover, binary tournament and a path mutation operator that substitutes the gene of a random locus by another random node (operators only succeed in case the path feasibility is maintained). Their experimentation showed the random walking strategy enabled the algorithm to cover a set of nodes and edges considered to be satisfactory by the authors. For three and four objectives, with instances ranging from 100 to 3500 nodes, their algorithm was able to approximate a reference set created by three algorithms: Dijkstra's [7], a k-th shortest path and many executions of their own algorithm. Their EA is also able to find extreme supported solutions faster than Dijkstra's algorithm [7] on real-world networks, but no information is given on how many objectives are used.

He *et al.* [22] also used two populations (elitist approach), depth-first search (initial individuals), dominance ranking with niche count, and path encoding (using vertices). The one-point crossover with binary tournament chooses a random locus from one parent, and in case the same node is also present on the other parent, the chromosomes are paired and the crossover is performed (a repair function eliminates possible loops). The mutation operator reconstructs the chromosome from a random locus through depth-first search mechanism. An illustrative example showed the efficiency of the proposed MOEA on a 50 nodes 3-objective instance.

Pangilinan and Janseens [39] tested SPEA2 [53] for the MSP using the implementation available on PISA framework [4]. They also used path encoding, but generated the initial population randomly. The one-point crossover and the mutation operator are identical to [22], but mutation reconstructs individuals randomly. Using three objectives instances, the approximation set presented good diversity on two of the objectives, but the authors were unable to determine the optimality of the solutions, since the actual Pareto sets were unavailable. Computational results showed that their MOEA is slower than [33].

Finally, Lin and Gen [28] proposed an MOEA for the BSP. Their adaptive weight algorithm uses priority-based encoding (fixed chromosome sizes), roulette selection, weight-mapping crossover, a random mutation operator and also an immigration operator [34]. No information is given on how the initial population is generated. The authors use two fuzzy logic controllers in order to auto-tune the parameters of their algorithm. They compare their MOEA against implementations of NSGA-II [6], SPEA [51] and rwGA [24] that use the same encoding and operators, analyzing diversity, cardinality, ratio of non-dominated solutions and computational time. Regarding these indicators, results show that priority-based encoding and auto-tuning are good strategies, and that their algorithm outperforms the others.

As far as the authors are aware of, only three Ant Colony Optimization algorithms have been proposed for the MSP [19, 16, 3]. Häckel *et al.* [19] proposed a version of the algorithm presented in [23]. Their algorithm, however, does not contain overlapping zones. The heuristic information comes from a Dynamic Programming algorithm called *Look-Ahead Heuristic* (LAH), proposed by the authors. As for the pheromone matrices, equations are presented as if multiple matrices were used, but it is stated that only one matrix is used. The experiments conducted in their work show that the usage of LAH improves overall results, and that their ACO obtains solutions well distributed along the objective space in comparison to a dynamic multi-objective algorithm not referenced by the authors on three objective instances.

An ACO for the BSP is proposed by Ghoseiri and Nadjari [16] and is compared to an exact label correcting algorithm, also not referenced. A single colony and two pheromone ma-

trices, one for each objective, were used. As for the heuristic information, two sources are used: normalized edge weights and number of nodes to the destiny node. Scalarizations are used to compute both pheromone and heuristic information. This algorithm resembles ACS, except for the fact that local pheromone update is performed at each step of the construction procedure (after each transition is chosen). The experimentation conducted by the authors used two objective NetMaker [43] instances. Results showed their algorithm was able to generate an approximation set well distributed over the objective space, but the Pareto fronts graphically presented contained no efficient solutions. Their ACO was computationally faster than the label correcting method.

Bezerra *et al.* [3] proposed a two-phase ant colony optimization algorithm, called GRACE. In the first phase, a search strategy is used to retrieve supported solutions. In the second phase each ant, during a set of iterations called generational cycle, receives a random scalarization vector for its search. The heuristic information comes from the application of Dijkstra's algorithm [7] to the inverted graph. A single pheromone matrix is used. An ant is allowed to update the pheromone information only when it finds a new non-dominated solution regarding an external archive of non-dominated solutions found during the search. No evaporation is used. GRACE was compared with an NSGA-II algorithm and to an ACO previously published for the MSP [19]. Pareto compliant quality indicator are used to assess the performance of the algorithms on 18 instances divided into two classes with 9 instances each. The results show that GRACE is able to produce better approximation sets than the NSGA-II for both classes of tested instances. The former algorithm also outperformed the ACO from Häckel *et al.* [19] on 12 out of 18 instances used.

4 Methodology, Platform and Instances

Performance assessment for multi-objective problems is a complex subject and a number of proposals exist in the literature. The assessment of the results produced in the computational experiments reported in this paper is done with basis on the methodology presented by Knowles, Thiele and Zitzler [26]. At first, the approximation sets delivered by the tested algorithms are compared by means of dominance ranking. Given a list of q optimizers, r_i $(i = 1, ..., q)$ independent executions for each of them and a collection $\mathbf{C}$ containing all approximation sets C_j^i $(i = 1, ..., q,\ j = 1, ..., r_i)$ generated at each run of the corresponding optimizer, to each set C_j^i is given a rank, $rank(C_j^i)$, equal to one plus the number of sets that are better than it. To state the concept of *better* between two approximation sets, some definitions are necessary. Given two objective vectors z_1 and z_2, z_1 is said to weakly dominate z_2 if z_1 is not worse than z_2 in all objectives [52]. This concept is extended to approximation sets by the relation sets *better*. Let A_1 and A_2 be two approximation sets. A_1 is said to be better than A_2 when every objective vector $z_2 \in A_2$ is weakly dominated by at least one $z_1 \in A_1$ and the approximation sets differ from each other, at least, in one objective vector. Hence, every optimizer is characterized by a sample $(rank(C_0^i), ..., rank(C_{r_i}^i))$, and these samples can be compared using statistical tests. Knowles, Thiele and Zitzler [26] claim that, if an optimizer Q^1 has dominance rankings statistically lower than another optimizer Q^2, then it can be said that Q^1 generates better approximation sets than Q^2, and no further testing is required.

In case no statistical difference is pointed out with the data produced with the dominance ranking method, quality indicators are used. Quality indicators are functions that attribute qualitative values to approximation sets. A quality indicator I is defined as a mapping from the set of all approximation sets to the set of real numbers. Some quality indicators use a reference set to compare the approximation sets generated by an optimizer. This reference set can be generated

by merging the approximation sets of all optimizers being tested [26]. Each indicator measures a specific characteristic of the set and, therefore, combining multiple indicators is suitable. In this work, two of these indicators are used:

1. the hypervolume indicator (I_H) [51], which calculates the hypervolume of the objective space that is weakly dominated by the approximation set being tested (limited by a reference point);

2. the additive unary-ϵ indicator ($I^1_{\epsilon+}$) [52], which calculates the minimum ϵ which must be added to each solution in a set A_2 for it to become weakly dominated by another set A_1.

To compare more than two optimizers, we initially use the Kruskal-Wallis test [27], and if a statistically significant difference is found at a significance level 0.05, new samples/values are generated for a pairwise comparison, and two more tests are applied: the Wilcoxon two-tailed test and the Wilcoxon one-tailed test [50]. If only two optimizers are being compared at a time, there is no need to use Kruskal-Wallis test, and the Wilcoxon tests are directly applied. The results returned by the Wilcoxon test are, finally, compared using Taillard's test [48] for comparison of proportions with significance level of 0.05. In this work, $p\text{-}values$ from Taillard's test results are rounded to two decimal places.

All tests were executed on an Intel Xeon QuadCore W3520 2.8 GHz, with 8G of RAM running Scientific Linux 5.5 64bits distribution. Experiments were executed with 18 instances generated by Santos [37], from two distinct classes: *grid*, which represents a square grid, and *complete*, which contains complete graphs. Fixed processing times for each instance are adopted as the stopping criterion. The limits for processing times are set according to each instance size and class. The instances are presented in Table 1 where column # shows the instance identification, column *Type* shows identification <*type*>N-<*size*> where <*type*> stands for instance type (grid or complete) and <*size*> stands for the instance size (small, medium or large). Moreover, $|N|$, $|A|$ and k are respectively the number of vertices, edges and objectives of each instance, *t(s)* is the maximum processing time seconds and $Seed$ is the seed used for generating the instance. Throughout this work the basic parameters presented in the original GRACE will remain unchanged, that is, $n_{ants} = 300$, $\alpha = 0.6$, $\beta = 0.6$, $\tau_0 = 1$, $\tau_{deposit} = 10$ and R=5.

5 Extending GRACE

As seen on the literature review on MOACO algorithms, there are many possibilities on designing an ACO for the multi-objective context. Some of those approaches are not trivially extended to problems with more than two objectives. In this section, MOACO algorithms are presented to the MSP where some critical design questions are discussed for the three-objective context. These algorithmic versions extend the GRACE algorithm [3] which is described in this section firstly. Then, the proposals are presented and tested, and results are discussed.

5.1 GRACE

Proposed as a two-phase algorithm, GRACE initially uses a search procedure, named *Logos*, to retrieve supported efficient solutions. This search strategy was proposed for bi and tri-criteria scenarios (2 and 3-Logos, respectively). 2-Logos iteratively divides each search region into 2 sub-regions, resembling a logarithmic function. The general framework of 2-Logos is presented in Algorithm 1. In the first execution of 2-Logos two solutions s_i and s_f are obtained

Table 1: List of instances. Only the first three objectives were used on *large* instances.

#	$Type$	$\|N\|$	$\|A\|$	k	$t(s)$	$Seed$
1	CompleteN-small	25	600	3	5	45
2	CompleteN-small	50	2450	3	10	12
3	CompleteN-small	100	9900	3	12	13
4	CompleteN-medium	40	780	3	5	18
5	CompleteN-medium	120	14280	3	10	14
6	CompleteN-medium	200	39800	3	15	21
7	CompleteN-large	100	9900	6	8	1
8	CompleteN-large	150	22350	6	40	10
9	CompleteN-large	200	39800	6	40	1
10	GridN-small	64	224	3	7	14
11	GridN-small	144	528	3	36	1
12	GridN-small	256	960	3	81	26
13	GridN-medium	484	1848	3	100	1
14	GridN-medium	961	3720	3	100	40
15	GridN-medium	1225	4760	3	100	2
16	GridN-large	121	440	6	60	41
17	GridN-large	484	1848	6	100	42
18	GridN-large	900	3480	6	100	43

by solving the single-objective minimum shortest path for each objective separately. Each solution corresponds to solving an scalarized version of the MSP problem with scalarization vectors (1,0) and (0,1). More generally, solutions s_i and s_f correspond respectively to points (x_i, y_i) and (x_f, y_f) in the objective space. A scalarization vector corresponding to $(x_{midpoint}, y_{midpoint})$ is generated, and the single-objective minimum shortest path using this ponderation is retrieved, returning a solution s_{mid} (line 1). If this solution is non-dominated regarding the global archive, the procedure is recursively called for (s_i, s_{mid}) and (s_{mid}, s_f) (lines 2-4). Otherwise, the recursion stops. This strategy is similar to [46] and [41].

Algorithm 1 Procedure 2-Logos

Require: archive S, solution s_i, solution s_f
1: s_{mid} = Dijkstra(midpoint(s_i.weights, s_f.weights));
2: **if** newSolution(S, s_{mid}) **then**
3: 2-Logos(S, s_i, s_{mid});
4: 2-Logos(S, s_{mid}, s_f);
5: **end if**

Analogously, the 3-Logos initially finds extreme supported efficient solutions using Dijkstra's algorithm [7]. These three solutions are considered vertices of a triangle where each edge is formed by the line that connects each pair of input solutions. Each edge is scanned by the 2-Logos procedure (line 1). After completion of 2-Logos executions on the three edges (line 2), the centroid of the triangle is calculated and, in case the obtained solution is new in the

set S of non dominated solutions, 3-Logos is recursively called for these three new subtriangles (lines 4-6). Both 2-Logos and 3-Logos are illustrated on Fig. 1.

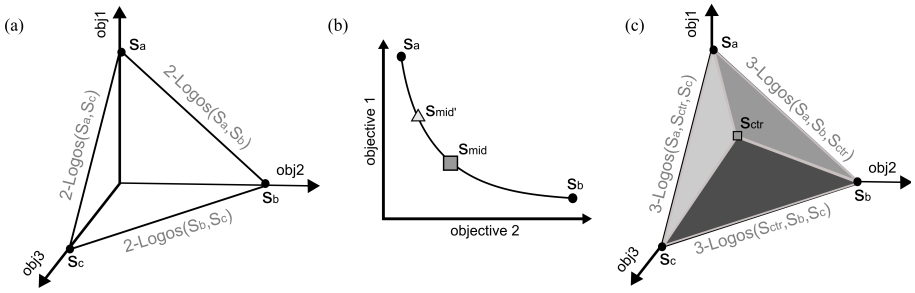

Figure 1: Example of one iteration of 3-Logos.

Algorithm 2 Procedure 3-Logos

Require: Pareto set S, solution s_a, solution s_b, solution s_c
1: S = addSolutions(S, 2-Logos(s_a, s_b), 2-Logos(s_a, s_c), 2-Logos(s_b, s_c));
2: s_{ctr} = Dijkstra(centroid(s_a.weights, s_b.weights, s_c.weights));
3: **if** newSolution(S, s_{ctr}) **then**
4: 3-Logos(S, s_a, s_b, s_{ctr});
5: 3-Logos(S, s_a, s_{ctr}, s_c);
6: 3-Logos(S, s_{ctr}, s_b, s_c);
7: **end if**

The second phase of GRACE can be seen on Algorithm 3. A single colony approach is adopted. The solutions found in the first phase are used to warm the single pheromone matrix up (line 4): every time an edge is found on a supported efficient solution, a deposit is made. Dijkstra's algorithm [7] provides heuristic information to the ants: a scalarization vector is used to obtain a single-objective problem, and Dijkstra's algorithm [7] is applied on an inverted graph, which is obtained from the original graph with the inversion of the direction of all edges (line 7). In other words, the heuristic information available in each node v is the distance from v to the terminal node t, under a given scalarization. These scalarization vectors are randomly generated at the beginning of each iteration (line 6), and all ants use the same vectors per iteration. New non-dominated solutions found by the ants are added to an external archive (line 11), which is unlimited. The ants are only allowed to perform pheromone updates if they find new non-dominated solutions (lines 10-13). Each of the ants fallowed to update the pheromone matrix deposits an equal amount of pheromone, regardless of the objective values of the solution found. The main loop (line 5) comprises a group of R iterations each called *cycles*. If during a cycle a new non-dominated solution arises, then a new cycle of iterations is allowed and the algorithm receives more processing time. This verification is made in lines 15-17, where the boolean procedure newSolution(cycle) returns true if a new non-dominated solution was generated in the corresponding cycle.

5.2 Proposals and Experimentation

Extensions of this algorithm are presented in the following sections, with single and multiple-colony approaches. When the concept behind the version do not allow the single colony approach, it is explicitly stated. In the multi-colony approaches, different colonies do not interact directly: each of them has its own pheromone matrix and heuristic information. A single type of heuristic information and pheromone is used for all colonies. However, when

Algorithm 3 GRACE

```
Require: graph G, vertex s, vertex t
extreme_solutions = Dijkstra(canonicalVectors());
supported = addSolutions(supported, extreme_solutions);
3-Logos(supported, extreme_solutions);
pheromoneM = pheromoneWarmup(supported);
for i = 0; i < R; i++ do
    l = randomScalarizationVector();
    heuristicM = heuristicInfo(Dijkstra(reverse(G), t, l));
    for j = 0; j < n_ants; ++j do
        s = buildSolution(pheromoneM, heuristicM, s, t, l);
        if newSolution(unsupported, s) then
            unsupported = addSolution(unsupported, s);
            pheromoneUpdate(s);
        end if
    end for
    if ((i == R) and (newSolution(cycle)) then
        i = 0;
    end if
end for
```

selecting which ants are allowed to perform pheromone update, all colonies test their solutions according to dominance in regard to the external global archive that stores the solutions found by all colonies during the whole execution of the algorithm. An important question arises, which is easily solved for bi-objective problems, regarding the update by region strategy: how to order solutions to determine the regions when this update strategy is utilized on three objectives problems. Whereas a simple lexicographic ordering divided the objective space *"equally"* among objectives in the bi-objective case, there is no trivial solution for the three objective context. The strategies utilized to overcome this problem are described in the corresponding sections.

Four versions of the basic algorithm were developed with the aim of evaluating methods for the creation of the scalarization vectors used to provide heuristic information to the ants. These versions are named: *Global-Random*, *Fixed-Unique*, *Fixed-Multi*, *Pseudo-Global-Random* and *Mixed*. In the *Global-Random* (*GRan*) version, the scalarization vectors are generated randomly. There is no division of the Pareto front region, whatever the amount of colonies used. Therefore, it is only possible to use the *update by origin* strategy for the pheromone matrices update. There is no direct relation between the number of scalarization vectors generated and the number of colonies of the algorithm.

The scalarization vectors are generated systematically in the *Fixed-Unique* (*FUn*) version. At first, a parameter d is chosen to establish the number of equal divisions of the interval [0,1]. This number determines the possible weighting values each component of the scalarization vectors can be set. For a tri-criteria problem, the number of combinations is given by $((d+1)*(d+2))/2$. Then, all possible scalarization vectors are generated, uniformly distributed in the region defined by the extreme points of the objective space. For instance, if k=3 that region is defined by the extreme points (1,0,0), (0,1,0) and (0,0,1). When $d = 2$ is used, there are three possible weighting values: 0, 0.5 and 1. Then, for k=3 and d=2, the scalarization vectors are [1,0,0], [0.5,0.5,0], [0.5,0,0.5], [0,0.5,0.5], [0,1,0] and [0,0,1]. In this algorithmic version, the number of colonies is equal to the number of ponderations, since each of these vectors will be assigned to a single colony.

The *Fixed-Multi* (*FMu*) version comprises a variation of FUn where the generated scalarization vectors are *equally* distributed among the m colonies. Therefore, for m colonies and

d divisions, $((d+1)*(d+2))/2m$ scalarization vectors are assigned to each colony (for the cases of non-exact division, the m-th colony may receive extra vectors corresponding to the remainder of the integer division). This variant is expected to have the same characteristics of *FUn*, but to be computationally more efficient.

The *Pseudo-Global-Random* (*PGRan*) joins the characteristics of the previous versions. For tri-criteria problems, the scalarization vectors are generated randomly within limits established by a triangulation procedure. Given a number of d divisions, the same procedure used in FUn to generate the weighting vectors is used to generate vectors in the region limited by points (1,0,0), (0,1,0) and (0,0,1) as shown in Fig. 2. These points define the vertices of triangular regions. The scalarization vectors of each colony are randomly drawn within each of these regions. Thus, in this algorithmic version there are d^2 colonies. Once the computation effort grows rapidly for increasing values of d, small values are preferred.

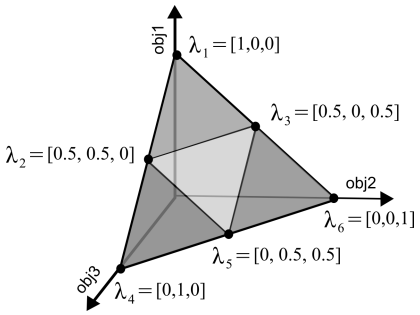

Figure 2: Example of ponderations and colonies for PGRan. With $d = 2$, six vectors are generated, resulting in four colonies.

For the assessment of these versions, the parameters of each variant was initially tuned. The GRan version is tested with 1, 3 and 5 colonies, where the single colony version corresponds to the original GRACE. Ants only update the pheromone matrix of its own colony. The p-values obtained with the Kruskal-Wallis test [27] are shown in Table 2, where column # shows the instance identification and columns DR, $I^1_{\epsilon+}$ and I_H show, respectively, the p-values corresponding to the dominance ranking, the unary epsilon indicator and the hypervolume indicator. Regarding significance level of 0.05 and the dominance ranking, the results presented in Table 2 show that significant differences exist for 3 of each instances class. For the instances where dominance ranking did not point out differences, the I_H shows significant results on 1 an 4 instances of classes complete and grid, respectively. On these four grid instances the $I^1_{\epsilon+}$ also shows significant results.

Pairwise comparisons with the Wilcoxon's test were carried out on instances where significant differences were found. Then, the number of successes of each optimizer was counted and submitted to the test for proportions comparison proposed by Taillard, Waelti and Zuber [48] and available at [45]. A success is defined when one approach outperforms another one concerning the dominance ranking and the quality indicators. Table 3 shows the $p\text{-}values$ returned when only the dominance rankings comparison is considered (DR), and when the indicators are included (DR + QI). Considering significance level 0.05, an entry inferior to this value in line i and column j states that the GRan version with the number of colonies (column Colonies) shown in line i is significantly better than the version with the number of colonies shown in column j, for the considered type of instance.

When only the dominance ranking methodology is considered, GRan using a single colony performs better than using three colonies for the complete set, and that using five colonies for the grid set. The overall results show that the use of multi-colonies (as defined

Table 2: Results from Kruskal-Wallis test on dominance rankings and quality indicators from GLOBAL using 1, 3 and 5 colonies. NaN means rankings were equal.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**0.00195**	-	-	10	**0.02245**	-	-
2	**0.00013**	-	-	11	**1.38e-06**	-	-
3	0.15987	0.21236	0.58046	12	NaN	**0.00014**	**4.66e-15**
4	0.35899	0.50174	0.16563	13	NaN	**3.51e-05**	**1.49e-13**
5	0.46921	0.30966	0.59756	14	NaN	0.54740	0.11592
6	0.16459	0.26382	0.90371	15	NaN	0.25459	0.50540
7	**0.04576**	-	-	16	**4.75e-13**	-	-
8	0.60301	0.00376	**0.00821**	17	NaN	**1.55e-11**	**9.07e-17**
9	0.84017	0.79358	0.24028	18	NaN	**0.03452**	**0.00796**

Table 3: p-values for proportions comparison on GRan.

		DR			DR + UQI		
	Colonies	1	3	5	1	3	5
Complete	1	-	**0.04**	0.09	-	**0.01**	**0.01**
	3	0.96	-	0.75	0.99	-	0.75
	5	0.91	0.25	-	0.99	0.25	-
Grid	1	-	0.09	**0.04**	-	**0**	**0**
	3	0.91	-	0.25	1	-	**0.01**
	5	0.96	0.75	-	1	0.99	-

for this paper) does not improve the performance of the original GRACE. The results also show that the version with 3 colonies is better than the one with 5 colonies for Grid instances. A possible explanation for these results is that, since this multi-colony approach is equivalent to a multi-start strategy, using multiple colonies only increases the computational effort.

For the FUn version, values 1, 2 and 3 were tested for parameter d which correspond to 3, 6 and 10 scalarization vectors and colonies. Again, only update by origin is used. Results for comparisons among these three versions and the GRan with one colony are presented on Table 4. Those results take in account the dominance ranking and the quality indicators. Considering dominance ranking, significant differences were found among the optimizers on four complete and three grid instances. The unary additive ϵ detected significant differences on another complete instance and on three grid instances. The hypervolume indicator shows significant differences on five grid instances.

The resultant p-values from the test for proportions comparison concerning the pairwise examinations are shown in Table 5. Those results show that the single-colony GRan performs better than all FUn versions for all instances. Among the different parameter settings tested for FUn, no statistically significant differences were found on the complete instances. On the other set of instances, however, the results show that the smaller the number of divisions, the better the performance of the algorithm version.

For the FMu version, tests were conducted using $m \in \{1, 3, 5\}$ and $d \in \{3, 6, 10\}$, which correspond to a total of 10, 28 and 66 scalarization vectors, respectively. At first, versions with m=1 were tested for the different values of parameter d. Significant different results were not found either with dominance ranking or the quality indicators among those versions. The same occurred for m=3 and 5. This means the m parameter could much likely be tested with any

Table 4: Kruskal-Wallis $p\text{-}values$: single-colony GRan against FUn.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**3.85e-05**	-	-	10	**7.48e-06**	-	-
2	**0.00251**	-	-	11	**4.92e-09**	-	-
3	**0.00541**	-	-	12	NaN	**2.30e-08**	**6.27e-21**
4	0.06157	0.59784	0.11722	13	NaN	**0.00063**	**7.85e-16**
5	0.07878	0.08633	0.01941	14	NaN	0.46923	0.06072
6	0.61764	0.76049	0.093	15	NaN	0.21608	**0.00357**
7	0.1909	0.28956	0.05886	16	**6.94e-19**	-	-
8	0.10916	**0.00476**	0.33655	17	NaN	**7.19e-14**	**4.14e-09**
9	**0.00974**	-	-	18	NaN	0.12736	**0.00020**

Table 5: p-values for proportions comparison on GRan against FUn.

		DR				DR + UQI			
	Versions	G	3	6	10	G	3	6	10
Complete	G	-	**0.04**	**0.04**	0.25	-	**0**	**0**	**0**
	3	0.96	-	0.5	0.5	1	-	0.25	0.25
	6	0.96	0.5	-	0.5	1	0.75	-	0.5
	10	0.75	0.5	0.5	-	0.91	0.75	0.5	-
Grid	G	-	**0.04**	**0.04**	**0.04**	-	**0**	**0**	**0**
	3	0.96	-	0.09	0.09	1	-	**0**	**0**
	6	0.96	0.91	-	0.25	1	1	-	**0.04**
	10	0.96	0.91	0.75	-	1	1	0.96	-

value for d, and results are expected to be similar. Therefore, fixing d=6, Table 6 shows the results of the Kruskal-Wallis test for the comparison of the different m values.

Table 6: Kruskal-Wallis analysis of parameter m in FMu: $d = 6$, $m \in \{1, 3, 5\}$. NaN means samples were equal.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	NaN	NaN	NaN	10	0.19424	**0.03246**	**0.03246**
2	NaN	NaN	NaN	11	**0.01064**	-	-
3	NaN	NaN	NaN	12	NaN	**0.00023**	**9.61e-16**
4	NaN	NaN	NaN	13	NaN	**0.00386**	**1.96e-15**
5	NaN	NaN	NaN	14	NaN	0.58568	**0.03837**
6	NaN	NaN	NaN	15	NaN	0.14769	0.50766
7	NaN	NaN	NaN	16	**2.27e-09**	-	-
8	NaN	NaN	NaN	17	NaN	**4.15e-14**	**3.39e-16**
9	NaN	NaN	NaN	18	NaN	0.28622	**0.00019**

Apparently, the same approximation sets have been generated for complete instances, regardless of the configuration used. However, there is statistically significant difference according to dominance ranking for three grid instances, and, regarding the quality indicators, for six other instances. Table 7 shows the Taillard test applied on the Wilcoxon's test [50] on the pairwise comparison. For any number of divisions, the smaller the amount of colonies used, the

better the overall performance of the algorithm on grid instances.

Table 7: Results from Taillard's test on the pairwise Wilcoxon's test result.

		DR			DR + UQI		
	Colonies	1	3	5	1	3	5
Grid	1	-	0.09	0.09	-	**0**	**0**
	3	0.91	-	0.25	1	-	**0.01**
	5	0.91	0.75	-	1	0.99	-

Finally, for PGRan, tests were conducted with $d \in \{1, 2, 3, 4, 5\}$, which correspond to 1, 4, 9, 16 and 25 colonies, respectively. Table 8 lists the results for the comparison among all versions with the Kruskal-Wallis test. The p-values presented in Table 8 show that there is significant difference for the majority of instances. The results of the pairwise comparisons, presented in Table 9, show that the single colony version, which corresponds to the single colony GRan, performs better than all other versions.

Table 8: Results for the comparison of PGRan with $d \in \{1, 2, 3, 4, 5\}$.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**6.30e-06**	-	-	10	**1.17e-12**	-	-
2	0.07911	**0.00568**	**0.00031**	11	**1.34e-08**	-	-
3	0.07548	**0.00406**	0.05932	12	NaN	**4.48e-14**	**6.24e-24**
4	**0.02135**	-	-	13	NaN	**4.00e-06**	**1.38e-20**
5	0.32284	0.41566	0.37870	14	NaN	0.88783	**0.00022**
6	0.20164	0.74161	0.58289	15	NaN	0.09336	**0.00021**
7	**0.02269**	-	-	16	**4.02e-21**	-	-
8	0.66612	**0.04934**	**0.00142**	17	NaN	**2.35e-15**	**2.89e-21**
9	0.09430	0.35075	**0.00032**	18	NaN	**0.03802**	**6.12e-09**

Table 9: p-values for proportions comparison on PGRan with $d \in \{1, 2, 3, 4, 5\}$.

	DR					DR + UQI				
Complete	1	2	3	4	5	1	2	3	4	5
1	-	0.09	0.09	0.09	**0.04**	-	**0**	**0**	**0.01**	**0**
2	0.91	-	0.09	0.25	0.25	1	-	0.25	0.25	**0.04**
3	0.91	0.91	-	0.5	0.25	1	0.75	-	0.75	0.09
4	0.91	0.75	0.5	-	0.25	0.99	0.75	0.25	-	**0**
5	0.96	0.75	0.75	0.75	-	1	0.96	0.91	1	-
Grid	1	2	3	4	5	1	2	3	4	5
1	-	0.09	**0.04**	**0.04**	**0.04**	-	**0**	**0**	**0**	**0**
2	0.91	-	0.09	0.09	0.09	1	-	**0**	**0**	**0**
3	0.96	0.91	-	0.09	0.09	1	1	-	**0**	**0**
4	0.96	0.91	0.91	-	0.5	1	1	1	-	**0**
5	0.96	0.91	0.91	0.5	-	1	1	1	1	-

Two overall comparisons were performed: one with the single colony and the other with the multiple colony approaches. For one colony configuration, the tested algorithmic versions

were: GRan, FMu with d=3 and PGRan with d=1. Furthermore, FUn is substituted by FMu with d=1, aliased Fix. Table 13 presents the results obtained with the Kruskal-Wallis test. Those results show significant differences for all Complete instances and two Grid instances regarding dominance ranking. Among the remaining Grid instances, the hypervolume indicates significant differences on six instances and the unary additive ϵ shows differences on three instances at significance level 0.05. The results of the pairwise comparisons exhibited in Table 10 show that for the different type of instances, different performances among the algorithms is observed. Regarding Complete instances, Table 10 shows the best overall performances of approaches GRan and PGRan. For Grid instances, the opposite happens. The systematic approaches FUn and FMu outperform the others for almost all instances.

Table 10: Kruskal-Wallis for 1-colony versions: GRan, Fix, FMu, PGRan.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**5.33e-23**	-	-	10	0.28296	0.28296	0.28296
2	**1.07e-23**	-	-	11	**0.00337**	-	-
3	**7.51e-25**	-	-	12	NaN	0.66977	**1.46e-11**
4	**4.17e-22**	-	-	13	NaN	0.27017	**7.01e-12**
5	**4.39e-22**	-	-	14	NaN	**0.04226**	**2.20e-07**
6	**1.62e-24**	-	-	15	NaN	0.71302	**0.00170**
7	**2.14e-22**	-	-	16	**2.33e-02**	-	-
8	**1.27e-25**	-	-	17	NaN	**6.77e-06**	**1.79e-18**
9	**3.23e-25**	-	-	18	NaN	**5.82e-05**	**6.00e-11**

Table 11: p-values for proportions comparison among single colony versions: GRan, Fix, FMu, PGRan.

	DR				DR + UQI			
Complete	GRan	Fix	FMu	PGRan	GRan	Fix	FMu	PGRan
GRan	-	**0**	**0**	0.5	-	**0**	**0**	0.5
Fix	1	-	0.5	1	1	-	0.5	1
FMu	1	0.5	-	1	1	0.5	-	1
PGRan	0.5	**0**	**0**	-	0.5	**0**	**0**	-
GRan	-	0.91	0.75	0.5	-	1	1	0.5
Fix	0.09	-	0.5	0.09	**0**	-	0.5	**0**
FMu	0.25	0.5	-	0.25	**0**	0.5	-	**0**
PGRan	0.5	0.91	0.75	-	0.5	1	1	-

Using m=3, GRan, FMu with d=3, PGRan with d=2 (the closest configuration to m=3) and Fix (FMu with d=1) are compared to assess the multiple colony versions. Table 12 shows that, as in the single colony overall comparison, dominance ranking indicates significant differences on the whole set of complete instances. For grid instances, the optimizers' performance can be considered different for all but one instance. Table 13 shows that all optimizers are able to generate better approximation sets than FMu for all complete instances, whereas the opposite happens for the grids.

Summarizing the results from this group of versions, the single colony optimizers present better behavior on the tested instances than the versions with multiple colonies. Among instances of the first group, GRan generates the best approximation sets for complete instances,

Table 12: Overall comparison of 3-colony versions using Kruskal-Wallis test: GRan, Fix, FMu and PGRan.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**1.66e-15**	-	-	10	0.08541	0.08154	0.08101
2	**1.39e-15**	-	-	11	0.10916	**1.09e-08**	**7.89e-13**
3	**7.94e-20**	-	-	12	NaN	**0.03913**	**7.03e-14**
4	**8.24e-15**	-	-	13	NaN	**1.55e-05**	**9.75e-12**
5	**2.33e-13**	-	-	14	NaN	0.09508	**5.87e-07**
6	**8.58e-19**	-	-	15	NaN	**0.02027**	**0.02720**
7	**3.42e-15**	-	-	16	**1.38e-08**	-	-
8	**1.92e-24**	-	-	17	NaN	**6.44e-05**	**1.82e-15**
9	**4.16e-22**	-	-	18	NaN	**0.00074**	**5.74e-09**

Table 13: Proportions comparison among GRan, Fix, FMu and PGRan (m=3).

	DR				DR + UQI			
Complete	GRan	Fix	FMu	PGRan	GRan	Fix	FMu	PGRan
GRan	-	0.25	**0**	0.5	-	**0.04**	**0**	0.5
Fix	0.75	-	**0**	0.5	0.96	-	**0**	0.91
FMu	1	1	-	1	1	1	-	1
PGRan	0.5	0.5	**0**	-	0.5	0.09	**0**	-
Grid	GRan	Fix	FMu	PGRan	GRan	Fix	FMu	PGRan
GRan	-	0.5	0.75	0.5	-	0.75	1	**0.04**
Fix	0.5	-	0.75	0.25	0.25	-	1	**0.01**
FMu	0.25	0.25	-	0.25	**0**	**0**	-	**0**
PGRan	0.5	0.75	0.75	-	0.96	0.99	1	-

but presents the worst performance for grids. The opposite situation happens for FMu. As for the multi-colony versions, GRan also generates better sets than FMu and FUn for complete instances. For grids, FMu shows the best overall performance.

A fifth version, named *Mixed* (*Mix*), was developed as another aggregation attempt to use stochasticity is a systematic way. This variant is equal to FMu, except for the inclusion of a random scalarization vector per generation in each colony. This allows colonies to search different regions of the objective space, which could either interfere positively or negatively on the overall efficiency of the algorithm. Furthermore, a second group of algorithmic versions was devised to allow different pheromone update strategies to be used among multi-colonies approach. Since there is no trivial way to order solutions for a division of the objective space, objective-driven colonies were used. A first version, called Obj, is proposed, where the scalarization vectors are generated as in FMu, but to each colony are assigned only ponderations which give priority to the specific criterion the colony is focused on. For example, a vector [0.5, 0.5, 0] would be rejected by the colony specialized in the first objective, for it does not reflect the colony's priority. The number of scalarization vectors per colony is smaller than $(d+1)(d+2)/2$, where d is the number of divisions (as in FMu). The amount of colonies is directly related to the number of objectives. Since there must be at least one colony per objective, a single colony version cannot be created. However, there is no limitation for how many colonies there might be per objective. In the experiments reported here, only one colony per objective is used. Each colony has its own pheromone matrix, that specializes for its primary objective.

For the pheromone update, three strategies were tested: *update by origin*, *update by re-*

gion and using both strategies at the same time. In the case of the update by region strategy it is necessary to determine the region to which a new generated solution belongs a solution. Given a new solution s_{new} and an external archive A, the policy adopted states that:

1. if a solution $s \in A$ is dominated by the new solution s_{new}, then the pheromone matrices of the colonies that focus on the objectives that have been optimized are updated;

2. if s_{new} does not dominate any solution of A, it is compared to the last solution found. For every objective $i \in \{1, ..., k\}$, if $f^i(s_{new}) < f^i(s_{last})$, a pheromone update will be performed on the matrix of the colony that focus that objective.

When update by origin and update by region are used simultaneously, the strategy applied to determine which regions a solution belongs to is done first, but the region which corresponds to the colony the ant belongs to is always updated. A second version, called ObjMixed (OMix), is a variant of Obj, created by the inclusion of a random scalarization vector per generation to each colony. It is an attempt to allow colonies to eventually search regions of the objective space they would not be able to. For the evaluation of Mix, Obj and OMix, parameter tuning was done first, and then an overall comparison that include the best approaches found in the first group of algorithmic versions was performed.

Regarding Mix, the results of the previous experiments are used to reduce the number of possible values to be tested for tuning parameters m and d. The values tested for m are 1 and 3, and for d are 3, 6 and 10. The Kruskal-Wallis test does not indicate significant differences among the versions with the different values for d when m is set to 1 or 3. The same effect was not observed when parameter m was tested. Significant differences were pointed out by the Kruskal-Wallis test among simple and multi-colonies versions depending on the value set to d on the set of Complete instances. For d=6, the single colony produces better approximation sets than the version with m=3, whereas for d=1 and d=3 this does not hold true. For Grid instances, all d values behave similarly. An example of these results is shown in Table 14 for d=6. The results of the statistical tests concerning pairwise comparisons confirm that the single-colony version prevails on both instance sets ($p\text{-}value < 0.01$).

Table 14: Wilcoxon two-tailed test results for Mix: m=1 and m=3 (d=6).

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	0.69334	0.53947	0.37614	10	0.08599	0.06904	0.06904
2	**0.03959**	-	-	11	**3.11e-05**	-	-
3	0.16491	0.09387	**0.04962**	12	NaN	**0.00020**	**7.40e-17**
4	0.36820	0.96358	0.81861	13	NaN	**0.00015**	**3.38e-17**
5	0.27834	0.23760	**0.02753**	14	NaN	0.71329	0.18226
6	0.20222	0.44755	0.93568	15	NaN	0.65548	0.70819
7	0.37640	0.33371	0.45529	16	**9.63e-11**	-	-
8	0.16074	**0.00279**	0.41467	17	NaN	0.00010	**1.69e-17**
9	0.33371	0.73703	0.063228	18	NaN	0.15839	**1.40e-05**

For the tuning of Obj, experiments were conducted initially to set parameter d with testing values 3, 4 and 5. Considering any of the pheromone update methods used in this work (update by origin, update by region and using both strategies at the same time), no significant differences were pointed out by the Kruskal-Wallis test at significance level 0.05. In the second part of the test, the three pheromone update strategies were directly compared to understand their individual contribution to the performance of the algorithm. Table 15 shows the

p-values for dominance ranking and the quality indicators when d=3. The Kruskal-Wallis test with significance level 0.05 does not indicate significant differences on the Complete instances. Significant results were produced on seven Grid instances. The results obtained with the pairwise comparison are presented in Table 16, and show that the update by origin strategy is the one that performs worst among the tested Grid instances. No significant differences were found between the performances of the algorithm that updates by region, and that the one that utilized both strategies simultaneously. This result indicates that the efficiency of the algorithm is directly related to the update by region method.

Table 15: Kruskal-Wallis test results for Obj: update by origin, by region or using both strategies at the same time.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	0.74524	NaN	NaN	10	**0.03702**	-	-
2	0.87877	NaN	NaN	11	**0.00277**	-	-
3	0.96529	NaN	NaN	12	NaN	0.61858	**6.84e-10**
4	0.60516	NaN	NaN	13	NaN	0.09566	**1.28e-10**
5	0.09474	NaN	NaN	14	NaN	0.95142	0.61685
6	0.14312	NaN	NaN	15	NaN	0.68661	0.45992
7	0.88791	NaN	NaN	16	**1.87e-13**	-	-
8	0.35550	NaN	NaN	17	NaN	**2.80e-05**	**2.51e-13**
9	0.14968	NaN	NaN	18	NaN	0.08899	**0.00690**

Table 16: Proportions comparison of Obj with different pheromone update methods.

	DR			DR + UQI		
Grid	Origin	Region	Both	Origin	Region	Both
Origin	-	0.96	0.91	-	1	1
Region	**0.04**	-	0.5	**0**	-	0.5
Both	0.09	0.5	-	**0**	0.5	-

The same values for parameter d were tested for OMix versions. No significant difference was found for the grid instances, regardless of the pheromone update strategy used. For the Complete instances set, however, two different scenarios were found. For all methods, d=3 produced approximation sets significantly better than the versions with d=5. No significant differences were detected between versions with d=3 and d=4.

The different update strategies were tested for d=3. Table 17 shows for the class of Complete instances that no significant differences were found, except for instances 6 and 8 where the quality indicators hypervolume and ϵ, respectively, detected significant differences with significance level 0.05. On the other hand, significant differences were pointed out for seven of the nine Grid instances. The test for proportions comparison on the results provided by the Wilcoxon test showed that there is no difference among the versions with the three different updating strategies on the complete instances. For the Grid instances, however, the same behavior found for Obj is repeated: the update by region strategy is critical to the efficiency of the algorithm concerning the quality of the approximation sets.

To conclude this group of experiments, three comparisons were conducted, concerning versions: GRan, FMu, Mix, Obj and OMix. The first considers parameters m and d set to 3, and Obj and OMix with update by origin. The second also uses m=d=3, but Obj and OMix using

Table 17: Kruskal-Wallis test results for OMix with d=3: update by origin, update by region and using both strategies simultaneously.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	0.74524	0.26767	0.51013	10	**0.00293**	-	-
2	0.47284	0.79410	0.21648	11	0.07043	**0.00017**	**4.22e-06**
3	0.96529	0.56443	0.44202	12	NaN	0.15397	**6.33e-12**
4	0.60516	0.48452	0.84530	13	NaN	**0.00848**	**1.00e-08**
5	0.09474	0.35550	0.09422	14	NaN	0.90868	0.97435
6	0.14312	0.06349	**0.03547**	15	NaN	0.85855	0.49478
7	0.88791	0.34751	0.07134	16	**7.40e-12**	-	-
8	0.35550	**0.04803**	0.66676	17	NaN	**7.87e-10**	**4.75e-16**
9	0.14968	0.75931	0.63876	18	NaN	**0.00592**	**0.00157**

Table 18: p-values for the proportions comparison of OMix with different pheromone update strategies.

	DR			DR + UQI		
Complete	Origin	Region	Both	Origin	Region	Both
Origin	-	0.5	0.5	-	0.25	0.5
Region	0.5	-	0.5	0.75	-	0.25
Both	0.5	0.5	-	0.5	0.75	-
Grid	Origin	Region	Both	Origin	Region	Both
Origin	-	0.91	0.91	-	1	1
Region	0.09	-	0.5	**0**	-	0.75
Both	0.09	0.5	-	**0**	0.25	-

update by origin and update by region simultaneously. The motivation behind this separation is to determine the impact of the update by region strategy on these two version when only multiple colony approaches are used. Finally, in the third comparison, parameter m is set to 1 and d to 3. Again, both update strategies are used simultaneously.

Tables 19 and 20 show the results when m and d are set to 3, and only update by region is considered for Obj and OMix. Table 19 shows that significant differences are identified on 12 of the 18 instances with the dominance ranking, at significance level 0.05. Significant differences are also found for the remaining instances by, at least, one quality indicator. The results from pairwise comparison exhibited in Table 20 show that Mix behaves exactly like GRan, meaning the former loses FMu's characteristics when random scalarization vectors are added to it. A similar performance is also observed for Obj and FMu on the two sets of instances. The same table also shows that versions GRan and Mix are the best and the worst ones for classes Complete and Grid, respectively. OMix performs better than FMu and Obj for the set Complete, but not as well as GRan and Mix.

When the update by region is allied to the update by origin strategy, again there is significant difference among the approximation sets for all instances. Table 21 shows that the performances of Obj and OMix are significantly improved on the Grid instances, and both outperform FMu. No changes are observed for the complete set regarding these two versions.

Finally, Table 22 shows the results for the third overall comparison. Significant differences are found on 16 of the 18 instances by the Kruskal-Wallis test with significance level 0.05. Table 23 exhibits the results for the pairwise comparisons, where the same results ob-

Table 19: Kruskal-Wallis test results (m=d=3): GRan, FMu, Mix, Obj, and OMix (considering these last two versions with update by origin).

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**5.73e-29**	-	-	10	**0.04356**	-	-
2	**9.76e-29**	-	-	11	**0.00261**	-	-
3	**1.50e-29**	-	-	12	NaN	**0.01786**	**7.37e-17**
4	**1.70e-27**	-	-	13	NaN	**3.67e-05**	**1.21e-16**
5	**1.53e-26**	-	-	14	NaN	0.94244	**6.50e-06**
6	**1.38e-27**	-	-	15	NaN	0.28765	**4.53e-05**
7	**1.14e-26**	-	-	16	**5.06e-07**	-	-
8	**4.29e-30**	-	-	17	NaN	0.11306	**4.77e-18**
9	**1.60e-29**	-	-	18	NaN	**0.00019**	**6.30e-11**

Table 20: Proportions comparison with m=d=3: GRan, FMu, Mix, Obj, and OMix (considering these last two versions with update by origin).

	DR					DR + UQI				
Complete	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	**0**	0.25	**0**	**0**	-	**0**	0.25	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
Mix	0.75	**0**	-	**0**	**0**	0.75	**0**	-	**0**	**0**
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	1	**0**	-	1	**0**	1	**0**	-
Grid	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	0.91	0.5	0.91	0.75	-	1	0.5	1	1
FMu	0.09	-	0.09	0.5	0.5	**0**	-	**0**	0.5	**0.04**
Mix	0.5	0.91	-	0.91	0.75	0.5	1	-	1	1
Obj	0.09	0.5	0.09	-	0.25	**0**	0.5	**0**	-	0.25
OMix	0.25	0.5	0.25	0.75	-	**0**	0.96	**0**	0.75	-

Table 21: Proportions comparison with m=d=3: GRan, FMu, Mix, Obj, and OMix (considering these last two versions using update by origin and by region simultaneously).

	DR					DR + UQI				
Complete	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	**0**	0.25	**0**	**0**	-	**0**	0.25	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
Mix	0.75	**0**	-	**0**	**0**	0.75	**0**	-	**0**	**0**
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	1	**0**	-	1	**0**	1	**0**	-
Grid	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	0.91	0.5	0.96	0.96	-	1	0.5	1	1
FMu	0.09	-	0.09	0.91	0.96	**0**	-	**0**	1	1
Mix	0.5	0.91	-	0.96	0.96	0.5	1	-	1	1
Obj	**0.04**	0.09	**0.04**	-	0.5	**0**	**0**	**0**	-	1
OMix	**0.04**	**0.04**	**0.04**	0.5	-	**0**	**0**	**0**	0.09	-

served previously for the set of Complete instances are seen. Obj and OMix, however, are not able to outperform FMu on the Grid instances. These results confirm the single colony versions as superior to the multi-colony approaches tested in this paper for this problem, regarding the instance sets used.

Table 22: p-values of the Kruskal-Wallis test that compared 1-colonies GRan, FMu, and Mix, and 3-colonies Obj and OMix with update by origin and by region simultaneously.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**3.66e-29**	-	-	10	0.07402	0.07402	0.07402
2	**3.63e-29**	-	-	11	0.07596	**1.21e-06**	**1.54e-10**
3	**4.36e-29**	-	-	12	NaN	0.57686	**8.97e-10**
4	**3.17e-28**	-	-	13	NaN	**0.02665**	**1.15e-10**
5	**1.19e-24**	-	-	14	NaN	0.19158	**3.81e-07**
6	**3.95e-27**	-	-	15	NaN	0.75345	**0.01110**
7	**4.57e-28**	-	-	16	0.23917	0.33997	0.11440
8	**1.81e-30**	-	-	17	NaN	**1.04e-06**	**2.41e-18**
9	**8.47e-30**	-	-	18	NaN	**1.81e-05**	**3.09e-09**

Table 23: p-values from pairwise comparisons concerning single-colonies GRan, FMu, and Mix, and 3-colonies Obj and OMix with update by origin and by region simultaneously.

	DR					DR + UQI				
Complete	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	**0**	0.09	**0**	**0**	-	**0**	0.09	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
Mix	0.91	**0**	-	**0**	**0**	0.91	**0**	-	**0**	**0**
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	1	**0**	-	1	**0**	1	**0**	-
Grid	GRan	FMu	Mix	Obj	OMix	GRan	FMu	Mix	Obj	OMix
GRan	-	0.5	0.5	0.5	0.5	-	1	0.5	1	1
FMu	0.5	-	0.5	0.5	0.5	**0**	-	**0**	**0.04**	**0**
Mix	0.5	0.5	-	0.5	0.5	0.6	1	-	1	1
Obj	0.5	0.5	0.5	-	0.5	**0**	0.96	**0**	-	0.09
OMix	0.5	0.5	0.5	0.5	-	**0**	1	**0**	0.91	-

6 Food Regulated Pareto Multi-Species

The results of the experiments reported in the previous section showed that different algorithm configurations produced the best approximation sets for each instance set. Some attempts were made to combine the stochastic and the systematic characteristics of those versions into a single algorithm, but no version prevailed over the others for the different instance classes. In order to obtain a framework that identifies potentialities among different algorithmic strategies, this section aims at devising an algorithm version that adapts itself depending on the characteristics of the instance. To achieve this goal the idea of food regulation among different ant species is introduced and compared to the other best performing versions previously presented.

Then, a final experiment is performed to compare this version with two existing algorithms: an ACO previously proposed for the MSP [19] and an adaptation of an available implementation of NSGA-II [6], a well-known efficient MOEA.

This section presents an Ant Colony Optimization algorithm which is based on the idea that ants from different species can walk on the same tracks looking for food, causing no interference in the pheromone updating of each other. Once they are from different species, they can only recognize their own pheromone. Each species is specialized on a search strategy. In this work, two species are used: one with the stochastic strategy from GRan and the other with the systematic aspect of FMu. Each species focuses on all criteria simultaneously, instead of focusing on a particular objective. There is no pre-determined number of species to be used, and each can follow a different search strategy. There is also no limitation for the number of colonies of each species and how they interact.

The first algorithmic version presented in this section is called Red&Black (R&B). In this version the ants are partitioned into two sets of equal sizes. Each set represents one species and uses its own search strategy. The spieces do not interact with each other. An external archive stores the non-dominated solutions found by all ants. A possible drawback in this strategy is that the usage of multiple species might jeopardize the overall algorithm efficiency, since more computation effort is needed. With each strategy receiving less processing time to work, this could result in worse approximation sets than those generated by the algorithmic versions presented previously. To overcome this drawback, a second concept is proposed: food regulation. The mimetism originally devised by Dorigo [13] came from the observation of ants looking for food. However, the ability of ants to find food sources is only used to create a reinforcement learning process where information emerges from pheromone deposit. In nature, if an ant colony is able to find food, it grows. Otherwise, it can shrink to the point of disappearing. In this work, the same phenomenon is proposed to occur within the species of artificial ants. The basic R&B algorithm is, then, modified to allow species regulation. If the search strategy being used by a particular species is being effective, it means that food is being found, which allows the colony to grow. Since the species are looking for food in the same environment (search space) and food resources are limited (the number of efficient non-dominated solutions of a MOCO problem is finite), the growth of a species causes the shrinking of the other.

The food regulated versions can also be implemented with single or multiple colonies of each species. In this work, the same number of colonies and ants in each colony are used for both species. A counter of new non-dominated solutions is associated to each colony. At each iteration, all ants from all colonies of all species build their solutions. Whenever an ant of a given colony finds a new non-dominated solution regarding the external archive, the counter corresponding to its colony is increased. In order to operationalize the process of food regulation, colonies are arranged in pairs. Each pair contains one colony from each species. The pairs remain the same over the execution of the algorithm. Every f_{gen} generations, the counters of the colonies of each pair are compared. The colonies that found more new non-dominated solutions are given the chance to grow, whereas the others shrink. This is controlled by a variation rate f_{rate} that is initially set to one. The amount of ants in each colony $f_{rate} \cdot n_{ij}$, where n_{ij} stands for the number of ants in the i-th colony of species j, i=1,...,m, j=1, 2. When the colony grows, variable f_{rate} is increased by a parameter Δf_{rate}. The opposite happens when the number of ants decreases in a colony. The growth of a colony can be limited or unlimited. Algorithmic versions allowing each of these features are proposed.

Four variants of food regulated R&B are investigated where two of them allow unlimited growth of the size of colonies. In the other two variants the size growth is limited: when the f_{rate} of one of the colonies from a pair reaches zero, the other is not allowed to grow any further.

Versions named R&B-FRd and R&B-FRld correspond to R&B with unlimited and limited size growth of the colonies, respectively. The initial numbers of ants of each species also leads to different variants, particularly for our processing time limited experiment: whether each species will receive an amount of ants equal to the total number of ants when a single species was used, or if there will be a proportional division. While R&B-FRd and R&B-FRld use this proportional division, in R&B-FR and R&B-FRl the amount of ants per species equals the amount of ants from the original R&B.

Since R&B inherits GRan and FMu features, it also inherits their parameters. Tests were conducted for $m \in \{1, 3\}$ and $d \in \{3, 6, 10\}$. The variation of values of d for fixed values of m produced similar behaviors. Table 24 shows the results obtained for m=1. Significant differences are pointed out for six Complete instances on the dominance ranking, whereas for Grid instances a significant difference is observed only on one instance which is detected by the hypervolume indicator. Pairwise comparisons concerning Wilcoxon's test with significance level 0.05 on the set of Complete instances shows that the best results are obtained for d=3. Significant differences are also found in the comparison among d=6 and d=10, with the former exhibiting the best results.

Table 24: Comparison from R&B experimentation of parameter d with m=1.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**6.47e-05**	-	-	10	0.36788	0.36788	0.36788
2	**8.32e-08**	-	-	11	0.72339	0.21347	0.44775
3	**0.00139**	-	-	12	NaN	0.18566	0.32502
4	0.08576	0.75264	0.17745	13	NaN	0.17040	0.44394
5	0.60309	0.36788	0.60302	14	NaN	0.46112	0.31178
6	0.74014	0.52739	0.77601	15	NaN	0.68690	0.12025
7	**0.00063**	-	-	16	0.12535	0.29756	0.30742
8	**2.17e-07**	-	-	17	NaN	0.10729	**0.00190**
9	**2.77e-07**	-	-	18	NaN	0.61626	0.25463

To tune parameter m, comparisons were performed using fixed values for d=3, 6 and 10. Results are very similar, whichever the value of d is. The p-values obtained with the Wilcoxon test are presented in Table 25. Significant results are found on eight of the nine Complete instances and three Grid instances concerning dominance ranking and on the other four Grid instances concerning, at least, one quality indicator. The test for proportions comparison shows that the single version outperforms the three colony version for Grid instances ($p\text{-}value < 0$) whereas the opposite occurs on the instances of set Complete ($p\text{-}value < 0$).

When comparing the three colonies versions R&B, GRan, FMu, Obj, and OMix (these last two using update by origin and by region simultaneously), Kruskal-Wallis test with significance level 0.05 shows that significant differences are found on all instances as shown in Table 26. Results listed on Table 27 show that for set Complete R&B is able to generate better approximation sets than FMu and Obj, but not than GRan. Significant differences are not pointed out for R&B and OMix. As for the grid set, the approximation sets obtained with R&B are only considered statistically better than the sets produced by GRan.

Comparing single-colony versions R&B, GRan and FMu with 3-colonies Obj and OMix (which use update by origin and by region simultaneously), the Kruskal-Wallis test continues pointing out significant differences for all instances. In this case, approximation sets produced for grid instances by R&B are better than those generated by Obj and OMix, and there is no difference regarding the ones produced by FMu. Table 28 summarizes these results. For the set

Table 25: p-values obtained with the two-tailed Wilcoxon test for R&B with $m \in \{1, 3\}$ and d=3.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**3.39e-05**	-	-	10	**0.00270**	-	-
2	**6.68e-05**	-	-	11	**4.29e-05**	-	-
3	**2.02e-06**	-	-	12	NaN	**0.00155**	**1.69e-17**
4	**4.47e-05**	-	-	13	NaN	**0.00677**	**1.64e-15**
5	0.16040	0.16080	0.16042	14	NaN	0.37928	0.06112
6	**0.00183**	-	-	15	NaN	0.80926	0.53250
7	**0.00012**	-	-	16	**2.12e-10**	-	-
8	**4.38e-05**	-	-	17	NaN	**4.32e-09**	**1.69e-17**
9	**0.00239**	-	-	18	NaN	0.55379	**0.00087**

Table 26: p-values obtained with Kruskal-Wallis test for three-colonies GRan, FMu, R&B, Obj, and OMix with d=3.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**1.52e-28**	-	-	10	**0.00061**	-	-
2	**5.65e-28**	-	-	11	**1.01e-15**	-	-
3	**5.61e-28**	-	-	12	NaN	**3.18e-06**	**6.66e-25**
4	**1.06e-26**	-	-	13	NaN	**1.29e-08**	**1.85e-22**
5	**3.28e-22**	-	-	14	NaN	0.75412	**0.00014**
6	**1.28e-21**	-	-	15	NaN	**0.02611**	0.05925
7	**9.00e-27**	-	-	16	**4.29e-15**	-	-
8	**2.10e-28**	-	-	17	NaN	**3.80e-16**	**7.04e-26**
9	**7.79e-28**	-	-	18	NaN	**0.00011**	**1.27e-08**

Table 27: Pairwise proportions comparison test results of the three-colony versions R&B, GRan, FMu, Obj, and OMix (with d=3).

	DR					DR + UQI				
Complete	GRan	FMu	R&B	Obj	OMix	GRan	FMu	R&B	Obj	OMix
GRan	-	**0**	**0**	**0**	**0**	-	**0**	**0**	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
R&B	1	**0**	-	**0**	1	1	**0**	-	**0**	0.25
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	1	**0**	-	1	**0**	0.75	**0**	-
Grid	GRan	FMu	R&B	Obj	OMix	GRan	FMu	R&B	Obj	OMix
GRan	-	0.91	0.75	0.96	0.96	-	1	1	1	1
FMu	0.09	-	0.5	0.5	0.09	**0**	-	**0.02**	0.98	0
R&B	0.25	0.5	-	0.96	0.91	**0**	0.98	-	1	1
Obj	**0.04**	0.5	**0.04**	-	0.75	**0**	**0.02**	**0**	-	0.75
OMix	**0.04**	0.91	0.09	0.25	-	**0**	**0**	**0**	0.25	-

Complete, however, the sets generated by R&B are better than the ones generated by FMu and Obj, but worse than those produced by GRan and OMix.

Results show that using multiple species allowed the algorithm to generate approximation sets as good as the ones produced by FMu for the Grid instances, but not as good as the

Table 28: p-values obtained with the (pairwise) proportions comparison test for three colonies version of R&B, GRan, FMu, Obj, and OMix.

	DR					DR + UQI				
Complete	GRan	FMu	R&B	Obj	OMix	GRan	FMu	R&B	Obj	OMix
GRan	-	**0**	**0**	**0**	**0**	-	**0**	**0**	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
R&B	1	**0**	-	**0**	1	1	**0**	-	**0**	1
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	**0**	**0**	-	1	**0**	**0**	**0**	-
Grid	GRan	FMu	R&B	Obj	OMix	GRan	FMu	R&B	Obj	OMix
GRan	-	0.75	0.91	0.5	1	-	1	1	1	1
FMu	0.25	-	0.5	0.5	0.5	**0**	-	0.09	**0.04**	**0**
R&B	0.09	0.5	-	0.5	0.5	**0**	0.91	-	**0.04**	**0.01**
Obj	0.5	0.5	0.5	-	0.5	**0**	0.96	0.96	-	0.09
OMix	0.25	0.5	0.5	0.5	-	**0**	1	0.99	0.91	-

ones generated by GRan for the set Complete. This can be due to what had been previously discussed: the division of the computational resources among the different species impairs the overall algorithm's efficiency. To overcome this problem, food regulated versions of R&B automatically check where the search effort is being compensated and send more resources to the corresponding colonies.

Besides m and d, parameters f_{gen} and Δf_{rate} need to be tuned for the food regulated versions. Values tested for these parameters are: $m \in \{1, 3\}$ per species, $d \in \{1, 3, 6\}$, $\Delta f_{rate} \in \{0.1, 0.3, 0.5\}$ and $f_{gen} \in \{1, 5\}$. Regarding parameter d, the experiments showed that the smaller the amount of scalarization vectors, the better the approximation sets obtained for Complete instances were. Then d=1 was adopted for testing with these instances. For Grid instances, the best performance was found for d=3. Therefore, this value was chosen for this set. The second testing aimed at tuning Δf_{rate} and f_{gen}. Since this parameters are highly correlated all six combinations were compared. Generally no configuration could be attested better than the others, but some were found to be worse than the others: Δf_{rate}=0.1 with f_{gen}=1 and Δf_{rate}=0.1 with f_{gen}=5. Values Δf_{rate}=0.3 with f_{gen}=1 were chosen. Finally, regarding the number of colonies in each species, pairwise comparison using the Wilcoxon test showed that no significant differences are found on Complete instances, but exist for the Grid set: the single colony version outperforms the 3-colonies version at significance level 0.01.

Overall comparisons were performed between R&B and its variants. At first three colonies versions of the algorithm were tested. Table 29 shows that significant differences are identified by dominance ranking on all Complete instances. Differences were also pointed out on five of the nine Grid instances by at least one indicator. Pairwise comparisons showed that all food regulated versions improve the results obtained with R&B ($p\text{-}value$=0). Those results did not point out significant differences among the four variants. For single colony versions results are different for Grid instances where none of the versions produce approximation sets significantly better than the others.

Once the four food regulated R&B variants presented similar results concerning the quality of the generated approximation sets, single colony version R&B-FRd was chosen to be compared to single colonies GRan and FMu, and 3-colonies Obj and OMix (which use update by origin and by region simultaneously). The proportions comparison statistical test on the results produced by the Wilcoxon tests among each pair of approaches (all tests at significance

Table 29: p-values obtained with the Kruskal-Wallis test on the comparison of food regulated versions with R&B.

#	DR	$I^1_{\epsilon+}$	I_H	#	DR	$I^1_{\epsilon+}$	I_H
1	**1.17e-13**	-	-	10	0.27478	0.23883	0.27224
2	**5.82e-15**	-	-	11	NaN	0.54268	**0.00088**
3	**1.30e-30**	-	-	12	NaN	0.75195	**0.00031**
4	**7.22e-13**	-	-	13	NaN	0.91722	0.23137
5	**0.00165**	-	-	14	NaN	**0.01836**	**0.04177**
6	**1.37e-11**	-	-	15	NaN	0.96750	0.41583
7	**8.69e-10**	-	-	16	**0.03203**	-	-
8	**6.36e-29**	-	-	17	NaN	0.16866	**7.18e-05**
9	**1.07e-15**	-	-	18	NaN	0.80297	0.51552

level 0.05), showed that R&B-FRd behaves as well as GRan on the instances of set Complete and as well as FMu on the Grid instances. Results are listed on Table 30. Once R&B-FRd was found to perform as well as the best approaches on each instance set, these results confirm that the food regulated multi-species approach is able to identify search strategies that are best for each instance type and direct search efforts. Therefore, this approach is able to perform well regardless of the characteristics of the instances.

Table 30: Proportions comparison test for versions single colonies R&B-FRd, GRan and FMu, and 3-colonies Obj and OMix (with update by origin and by region simultaneously).

	DR					DR + UQI				
Complete	GRan	FMu	FRd	Obj	OMix	GRan	FMu	FRd	Obj	OMix
GRan	-	**0**	0.09	**0**	**0**	-	**0**	0.09	**0**	**0**
FMu	1	-	1	0.5	1	1	-	1	0.5	1
FRd	0.91	**0**	-	**0**	**0**	0.91	**0**	-	**0**	**0**
Obj	1	0.5	1	-	1	1	0.5	1	-	1
OMix	1	**0**	1	**0**	-	1	**0**	1	**0**	-
Grid	GRan	FMu	FRd	Obj	OMix	GRan	FMu	FRd	Obj	OMix
GRan	-	0.75	0.75	0.5	0.75	-	1	1	1	1
FMu	0.25	-	0.5	0.5	0.5	**0**	-	0.5	0.09	**0**
FRd	0.25	0.5	-	0.5	0.5	**0**	0.5	-	**0.04**	**0.01**
Obj	0.5	0.5	0.5	-	0.5	**0**	0.91	0.96	-	0.09
OMix	0.25	0.5	0.5	0.5	-	**0**	1	0.99	0.91	-

To conclude the experimentation of the food regulated versions, single version R&B-FRd is compared to two existing multi-objective algorithms for the MSP, each representing a different metaheuristic. For the MOACOs, an ACO previously published for the MSP [19] is used. For the MOEAs, since no available implementation for the MSP of the published works exists, an adaptation an available implementation of the NSGA-II [38] was made. Parameters are set as in [3]. For the NSGA-II, population varies according to instance class and number. For the first seven Complete instances 2500 individuals are used, whereas for the last two this number is increased to 3000. For the Grid instances, the linear equation $p(x)$=$a \cdot t(x) + b$ determines the population p of instance x according to the processing time t it is given; a=25 and b=800 are used. Crossover and mutation rates are set to 0.9 and 0.005, respectively. For the ACO proposed by [19], which is referred to as HACO in this work, parameters are set according to

the original paper where it was published. The only parameter which had to be adjusted was the negative pheromone update ξ, which is set to 0.8. Final settings for HACO are: $\chi_{colonies} = 3$, $\chi_{ants} = 4$, $\alpha = 0.5$, $\beta = 3$, $\tau_0 = 0.05$, $\rho = 0.1$, $q_0 = 0.5$ and $\xi = 0.8$.

Regarding dominance ranking, results from the Kruskal-Wallis test show significant differences for all instances. The pairwise proportions comparison test results are shown on Table 31. For the Complete set, R&B-FRd shows the best performance among the three optimizers. Moreover, HACO is also considered statistically better than NSGA-II. For the Grid instances, NSGA-II outperforms HACO, but once again the approximation sets produced by R&B-FRd are considered to be significantly better than the ones produced by the other optimizers.

Table 31: p-values from proportions comparison: single-colony R&B-FRd, NSGA-II [38] and HACO [19].

	DR			DR + UQI		
Complete	R&B-FRd	NSGA-II	HACO	R&B-FRd	NSGA-II	HACO
R&B-FRd	-	**0**	0.25	-	**0**	**0**
NSGA-II	1	-	1	1	-	1
HACO	0.75	**0**	-	1	**0**	-
Grid	R&B-FRd	NSGA-II	HACO	R&B-FRd	NSGA-II	HACO
R&B-FRd	-	**0**	**0**	-	**0**	**0**
NSGA-II	1	-	**0**	1	-	**0**
HACO	1	1	-	1	1	-

7 Conclusions and Future Work

This paper introduced the *food regulated Pareto multi-species ACO*, a new strategy for Multi-objective Ant Colony Optimization (MOACO). The proposed approach was applied to the well-known Multi-objective Shortest Path (MSP). In order to show the potentiality of the proposed method, a initial investigation was performed concerning several extensions of an ACO algorithm recently shown to perform well on the MSP. Each of those extensions implemented a different strategy related to an important design feature of MOACO algorithms. The objective was to carry out an investigation of the performance of different MOACO approaches on the problem studied in this work. Two different sets of tri-criteria instances, each one with its own structural characteristics, were considered in the experiments. The results showed that the best strategy depended on the type of the instances being tackled. Two algorithmic versions were shown to produce significant better results than the others, each for one set of instances. To devise a single algorithmic version capable of performing well on different type instances, two concepts for ACOs were adopted: the use of multiple species, each with a different search strategy, combined with the use of food regulation to allow more computational effort to the species with best performance.

The algorithmic version that used the concept of multi-species without food regulation, combining in a single algorithm different search strategies, did not achieve the objective of performing as well as the best single-species versions. However, the experimentation results confirmed the hypothesis that multiple species could allow a single algorithm to inherit different virtues of good optimizers. When multiple species are combined with food regulation, ants are given enough computational time and this version performs as well as the best single-species

approaches. Furthermore, for what concerns the MSP, the developed version is able to improve the results previously found by GRACE on the Grid instance set. When comparing to HACO and NSGA-II, results are even better than when the original comparison was performed, likely due to the different platform used in this work. This shows that, when more time is given to all algorithms, GRACE is able to use it more effectively than the other two optimizers.

In short, a new line of investigation concerning MOACO algorithms is proposed in which a number of issues can be further explored. For instance, future works can address metrics to evaluate the relative performance among colonies or species and new ways of combining the different species.

8 Acknowledgements

This work was supported by Conselho Nacional de Pesquisa e Desenvolvimento under project no. 579226/2008-5.

Referências

[1] ALAYA, I; SOLNON, C; GHÉDIRA, K. **Ant colony optimization for multi-objective optimization problems**. In: IEEE INTERNATIONAL CONFERENCE ON TOOLS WITH ARTIFICIAL INTELLIGENCE, p. 450–457, 2007.

[2] ANGUS, D; WOODWARD, C. **Multiple objective ant colony optimisation**. Swarm Intell., 3:69–85, 2009.

[3] BEZERRA, L. C. T; GOLDBARG, E. F. G; GOLDBARG, M. C; BURIOL, L. S. **Grace: A generational randomized ACO for the multi-objective shortest path problem**. Evolutionary Multi-criterion Optimizarion. In print, 2011.

[4] BLEULER, S; LAUMANNS, M; THIELE, L; ZITZLER, E. **Pisa a platform and programming language independent interface for search algorithms**. In: Fonseca, C; Fleming, P; Zitzler, E; Thiele, L; Deb, K, editors, EMO 2003, volume 2632 de **Lecture Notes in Computer Science**, p. 1–1. Springer Berlin / Heidelberg, 2003.

[5] CZYZAK, P; JASZKIEWICZ, A. **Pareto simulated annealing: A metaheuristic technique for multiple-objective combinatorial optimization**. J. Multi-Criteria Decis. Anal., 7:34–47, 1998.

[6] DEB, K; PRATAP, A; AGARWAL, S; MEYARIVAN, T. **A fast and elitist multiobjective genetic algorithm: Nsga-ii**. IEEE Trans. Evol. Comput., 6(2):182–197, 2002.

[7] DIJKSTRA, E. W. **A note on two problems in connexion with graphs**. Nr. math., 1959.

[8] DOERNER, K; GUTJAHR, W; HARTL, R; STRAUSS, C; STUMMER, C. **Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection**. Ann. Oper. Res., 131:79–99, 2004.

[9] DOERNER, K; GUTJAHR, W; HARTL, R; STRAUSS, C; STUMMER, C. **Pareto ant colony optimization with ilp preprocessing in multiobjective project portfolio selection**. Eur. J. Oper. Res., 171:830–841, 2006.

[10] DOERNER, K; HARTL, R. F; REIMANN, M. **Are COMPETants more competent for problem solving? - the case of a multiple objective transportation problem**. In: PROCEEDINGS OF THE GECCO'01, p. 802, Berlin, Heidelberg, 2001. Morgan Kaufmann.

[11] DORIGO, M; GAMBARDELLA, L. M. **Ant colony system: a cooperative learning approach to the traveling salesman problem**. IEEE Trans. Evolut. Comput., 1(1):53–66, 1997.

[12] DORIGO, M; SOCHA, K. **An introduction to ant colony optimization**. Technical Report TR/IRIDIA/2006-010, Institut de Recherches Interdisciplinaires et de Développements em Intelligence Artificialle, Université Libre de Bruxelles, 2006.

[13] DORIGO, M. **Optimization, learning and natural algorithms**. PhD thesis, Dipartamento de Elettronica, Politecnico de Milano, 1992.

[14] EHRGOTT, M; GANDIBLEUX, X. **A survey and annotated bibliography of multiobjective combinatorial optimization**. OR Spektrum, 22(4):425–460, 2000.

[15] GARCÍA-MARTÍNEZ, C; CORDÓN, O; HERRERA, F. **A taxonomy and empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP**. Eur. J. Oper. Res., 180(1):116–148, 2007.

[16] GHOSEIRI, K; NADJARI, B. **An ant colony optimization algorithm for the bi-objective shortest path problem**. Appl. Soft Comput., 10(4):1237–1246, 2010.

[17] GRASSÉ, P.-P. **La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. La théorie de la stigmergie: essai dínterprétation du comportement des termites constructeurs**. Insectes Sociaux, 6:41, 1959.

[18] GUNTSCH, M; MIDDENDORF, M. **Solving multi-criteria optimization problems with population-based ACO**. In: EMO 2003, volume 2632 de **Lecture Notes in Computer Science**, p. 464–478. Springer, 2003.

[19] HÄCKEL, S; FISCHER, M; ZECHEL, D; TEICH, T. **A multi-objective ant colony approach for pareto-optimization using dynamic programming**. In: PROCEEDINGS OF THE 10TH ANNUAL CONFERENCE ON GENETIC AND EVOLUTIONARY COMPUTATION (GECCO), p. 33–40. ACM, 2008.

[20] HANSEN, M. **Tabu search for multiobjective combinatorial optimization**. Control. and Cybern., 29:799–818, 2000.

[21] HANSEN, P. **Bicriterion path problems**. In: Fandel, G; Gal, T, editors, MULTIPLE CRITERIA DECISION MAKING, THEORY AND APPLICATION. PROCEEDINGS OF THE 3RD INTERNATIONAL CONFERENCE, volume 177 de **Lecture Notes in Economics and Mathematical Systems**, p. 109–127. Springer Berlin, 1980.

[22] HE, F; QI, H; FAN, Q. **An evolutionary algorithm for the multi-objective shortest path problem**. In: PROCEEDINGS OF THE ADVANCES IN INTELLIGENT SYSTEMS RESEARCH (ISKE), 2007.

[23] IREDI, S; MERKLE, D; MIDDENDORF, M. **Bi-criterion optimization with multi-colony ant algorithms**. In: Zitzler, E; Thiele, L; Deb, K; Coello, C. A. C; Corne, D, editors, EMO 2001, volume 1993 de **Lecture Notes in Computer Science**, p. 359–372. Springer, 2001.

[24] ISHIBUCHI, H; MURATA, T. **A multi-objective genetic local search algorithm and its application to flowshop scheduling**. IEEE Trans. Syst., Man, Cybern., Part C: Appl. and Rev., 28(3):392–403, 1998.

[25] KNOWLES, J; CORNE, D. **Instance generators and test suites for the multiobjective quadratic assignment problem**. 2632:295–310, 2003.

[26] KNOWLES, J. D; THIELE, L; ZITZLER, E. **A tutorial on the performance assessment of stochastic multiobjective optimizers**. Technical Report TIK-Report No. 214, Computer Engineering and Networks Laboratory, ETH Zurich, 2006.

[27] KRUSKAL, W. H; WALLIS, W. A. **Use of ranks in one-criterion variance analysis**. J. Am. Stat. Assoc., 47(260):pp. 583–621, 1952.

[28] LIN, L; GEN, M. **A bicriteria shortest path routing problems by hybrid genetic algorithm in communication networks**. In: PROCEEDINGS OF THE IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (CEC), p. 4577–4582, 2007.

[29] LÓPEZ-IBÁÑEZ; STÜTZLE, T. **Automatic configuration of multi-objective aco algorithms**. Technical Report TR/IRIDIA/2010-011, Institut de Recherches Interdisciplinaires et de Développements em Intelligence Artificialle, Université Libre de Bruxelles, 2000.

[30] LÓPEZ-IBÁÑEZ, M; PAQUETE, L; STÜTZLE, T. **On the design of ACO for the biobjective quadratic assignment problem**. In: Dorigo, M; Birattari, M; Blum, C; Gambardella, L; Montada, F; StÃ$\frac{1}{4}$tzle, T, editors, ANTS 2004, volume 3172 de **Lecture Notes in Computer Science**. Springer-Verlag, 2004.

[31] LÓPEZ-IBÁÑEZ, M; STÜTZLE, T. **An analysis of algorithmic components for multiobjective ant colony optimization: A case study on the biobjective tsp**. In: Collet, P; MonmarchÃ©, N; Legrand, P; Schoenauer, M; Lutton, E, editors, ARTIFICAL EVOLUTION, volume 5975 de **Lecture Notes in Computer Science**, p. 134–145. Springer Berlin / Heidelberg, 2010.

[32] LÓPEZ-IBÁÑEZ, M; STÜTZLE, T. **The impact of design choices of multiobjective ant colony optimization algorithms on performance: an experimental study on the biobjective tsp**. In: PROCEEDINGS OF THE 12TH ANNUAL CONFERENCE ON GENETIC AND EVOLUTIONARY COMPUTATION, GECCO '10, p. 71–78, New York, NY, USA, 2010. ACM.

[33] MARTINS, E. Q. V. **On a multicritera shortest path problem**. Eur. J. Oper. Res., 16:236 – 245, 1984.

[34] MICHAEL, C. M; STEWART, C. V; KELLY, R. B. **Reducing the search time of a steady state genetic algorithm using the immigration operator**. In: PROCEEDINGS OF THE IEEE CONGRESS ON TOOLS FOR AI, p. 500–501, 1991.

[35] MOONEY, P; WINSTANSLEY, A. **An evolutionary algorithm for multicriteria path optimization problems**. Int. J. Geogr. Inf. Sci., 20(4):401 – 423, 2006.

[36] MOTE, J; MURPHY, I; OLSON, D. L. **A parametric approach to solving bicriterion shortest path problems**. Eur. J. Oper. Res., 53:81 – 92, 1991.

[37] MSPP, 2010. www.mat.uc.pt/∼zeluis/INVESTIG/MSPP/mspp.htm.

[38] MULTI-OBJECTIVE NSGA-II CODE IN C. REVISION 1.1 (10 JUNE 2005) (FOR LINUX ONLY), 2010. www.iitk.ac.in/kangal/codes.shtml.

[39] PANGILINAN, J. M. A; JANSEENS, G. K. **Evolutionary algorithms for the multiobjective shortest path problem**. Int. J. Comput. Inf. Sci. Eng., 1, 2007.

[40] PAQUETE, L; STÜTZLE, T. **A two-phase local search for the biobjective traveling salesman problem**. In: Fonseca, C; Fleming, P; Zitzler, E; Thiele, L; Deb, K, editors, EMO 2003, volume 2632 de **Lecture Notes in Computer Science**, p. 69–69. Springer Berlin / Heidelberg, 2003.

[41] RAITH, A; EHRGOTT, M. **A comparison of solution strategies for the biobjective shortest path problem**. Comput. & Oper. Res., 36(4):1299–1331, 2009.

[42] SERAFINI, P. **Some considerations about computational complexity for multi objective combinatorial problems**. In: Jahn, J; Krabs, W, editors, RECENT ADVANCES AND HISTORICAL DEVELOPMENT OF VECTOR OPTIMIZATION, volume 294 de **Lecture Notes in Economics and Mathematical Systems**. Springer Verlag, Berlin, 1986.

[43] SKRIVER, A. J. V; ANDERSEN, K. A. **A label correcting approach for solving bicriterion shortest-path problems**. Comput. & Oper. Res., 27(6):507 – 524, 2000.

[44] SRINIVAS, N; DEB, K. **Multiobjective optimization using nondominated sorting in genetic algorithms**. Evolut. Comput., 2(3):221–248, 1994.

[45] STAMP PROJECT, 2010. http://qualopt.eivd.ch/stats/.

[46] STEINER, S; RADZIK, T. **Solving the biobjective minimum spanning tree problem using a k-best algorithm**. Technical Report TR-03-06, Department of Computer Science, Kingś College London, 2003.

[47] STÜTZLE, T; HOOS, H. H. **Max-min ant system**. Future Gener. Comput. Syst., 16:889–914, 2000.

[48] TAILLARD, E; WAELTI, P; ZUBER, J. **Few statistical tests for proportions comparison**. Eur. J. Oper. Res., 185:1336–1350, 2009.

[49] TAILLARD, E. D. **Robust taboo search for the quadratic assignment problem**. Parallel Comput., 17:443–455, 1991.

[50] WILCOXON, F. **Individual comparisons by ranking methods**. Biom. bull., 1(6):80–83, 1945.

[51] ZITZLER, E; THIELE, L. **Multiobjective evolutionary algorithms: a comparative case study and the strength pareto evolutionary algorithm**. IEEE Trans. Evolut. Comput., 3(4):257–271, 1999.

[52] ZITZLER, E; THIELE, L; LAUMANNS, M; FONSECA, C. M; ; DA FONSECA, V. G. **Performance assessment of multiobjective optimizers: an analysis and review**. Technical Report TIK-Report No. 139, Institut für Technische Informatik und Kommunikationsnetze, ETH Zürich, 2002.

[53] ZITZLER, E; LAUMANNS, M; THIELE, L. **Spea2: Improving the strength pareto evolutionary algorithm**. Technical Report TIK-Report 103, Computer Engineering and Networks Laboratory (TIK), Departament of Electrical Engeneering, Swiss Federal Institute fo Technology (ETH) Zurich, 2001.