Metaheuristics and cooperative approaches for the Bi-objective Ring Star Problem

Arnaud Liefooghe, Laetitia Jourdan, El-Ghazali Talbi

To cite this version:

Arnaud Liefooghe, Laetitia Jourdan, El-Ghazali Talbi. Metaheuristics and cooperative approaches for the Bi-objective Ring Star Problem. Computers and Operations Research, Elsevier, 2010, 37 (6), pp.1033 - 1044. 10.1016/j.cor.2009.09.004 . hal-00522622

HAL Id: hal-00522622

https://hal.archives-ouvertes.fr/hal-00522622

Submitted on 4 Oct 2010

Metaheuristics and cooperative approaches for the bi-objective ring star problem

A. Liefooghe*, L. Jourdan, E.-G. Talbi

Laboratoire d'Informatique Fondamentale de Lille, UMR CNRS 8022,
INRIA Lille-Nord Europe, Université Lille 1
Parc Scientifique de la Haute Borne, 40 avenue Halley, 59650 Villeneuve d'Ascq, France

Abstract

This paper presents and investigates different approaches to solve a new bi-objective routing problem called the ring star problem. It consists of locating a simple cycle through a subset of nodes of a graph while optimizing two kinds of cost. The first objective is the minimization of a ring cost that is related to the length of the cycle. The second one is the minimization of an assignment cost from non-visited nodes to visited ones. In spite of its obvious bi-objective formulation, this problem has always been investigated in a single-objective way. To tackle the bi-objective ring star problem, we first investigate different stand-alone search methods. Then, we propose two cooperative strategies that combine two multi-objective metaheuristics: an elitist evolutionary algorithm and a population-based local search. We apply these new hybrid approaches to well-known benchmark test instances and demonstrate their effectiveness in comparison to non-hybrid algorithms and to state-of-the-art methods.

Key words: Ring star problem, Multi-objective optimization, Metaheuristic, Cooperative approach

1. Introduction

The Bi-objective Ring Star Problem (B-RSP) aims at locating a simple cycle through a subset of nodes of a graph. The first objective consists in

*Corresponding author

Email addresses: Arnaud.Liefooghe@lifl.fr (A. Liefooghe), Laetitia.Jourdan@lifl.fr (L. Jourdan), talbi@lifl.fr (E.-G. Talbi)

minimizing a *ring cost* related to the cycle length. The second objective is to minimize an *assignment cost* from non-visited nodes to visited ones. Two conflicting costs are then considered here, which naturally leads to a bi-objective formulation. But this problem has always been addressed in a single-objective way; either where both costs are combined [1], or where the assignment cost is treated as a constraint [2]. However, from a practitioner point of view, it is often necessary to take multiple objectives into account simultaneously, and the problem investigated in this paper is a good illustration of what can typically be found in the industry. Indeed, as pointed out in [3], a large number of routing problems can be formulated as multi-objective optimization problems, and according to the same paper, the problem at hand is a generalization of the single-objective problems introduced in [1, 2]. Besides, the B-RSP is highly combinatorial as, once is decided which nodes are to be visited, a standard traveling salesman problem is still to be solved. Therefore, in practice, large-scale problem instances generally cannot be solved exactly. This is the reason why heuristic search methods are considered in this paper. Note that the current work extends preliminary results presented during the EvoCOP 2008 conference [4]. Nevertheless, a deeper analysis, extra stand-alone methods and novel hybrid approaches are additionally provided in the actual paper.

The main contribution of this work is twofold and can be summarized as follows. First, a set of multi-objective search methods are proposed for a new bi-objective problem, the B-RSP. Second, this paper introduces new general-purpose cooperative schemes for multi-objective combinatorial optimization, and their features and efficiency are investigated and rather discussed on solving the problem at hand. Thus, as an initial step, we investigate four metaheuristics to approximate the set of efficient solutions for the problem under consideration. Hence, IBMOLS, a population-based local search recently proposed in [5], is fitted for the B-RSP resolution. Next, we present slight variations of two well-known multi-objective search methods, namely IBEA [6] and NSGA-II [7]. At last, we propose a general-purpose multi-objective metaheuristic called SEEA and presented in this paper for the first time. We compare all these methods to each other on state-of-the-art benchmark test instances and discuss their respective behaviors. As a second step, we propose two cooperative approaches combining the local search method and SEEA. These methods are based either on a periodic or on an adaptive strategy, each one trying to benefit of the advantages of each algorithm it is compound of. Experiments validate the contribution of these hybrid schemes over non-hybrid approaches by achieving a strict improvement of the generated set of non-dominated solutions.

The remainder of the paper is organized as follows. Section 2 is devoted to the B-RSP. Then, four metaheuristics for multi-objective optimization are introduced in Section 3. A general presentation of these methods and their application to the B-RSP is followed by a comparative study. In Section 4, we propose new cooperation schemes to solve multi-objective combinatorial optimization problems. We experiment the resulting search methods on the B-RSP and we compare the obtained computational results to the previous ones. At last, conclusions and perspectives are drawn in the last section.

2. The Bi-objective Ring Star Problem

In this section, we first present some basic concepts, notation and definitions related to multi-objective optimization. Next, we provide a formulation of the ring star problem as a bi-objective problem. Finally, we briefly survey the literature related to the problem at hand and discuss its industrial concerns.

2.1. Multi-objective Optimization

A *Multi-objective Optimization Problem* (MOP) aims of optimizing a set of $n \geq 2$ objective functions $f_1, f_2, \ldots, f_n$ simultaneously. Each objective function can be either minimized or maximized. Let X denote the set of feasible solutions in the *decision space*, and Z the set of feasible points in the *objective space*. Without loss of generality, we here assume that $Z \subseteq \Re^n$ and that all n objective functions are to be minimized. To each decision vector $x \in X$ is assigned exactly one objective vector $z \in Z$ on the basis of a vector function $f : X \rightarrow Z$ with $z = f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$. We will assume, throughout the paper, that objective values are normalized[1]. Therefore, a MOP can be formulated as follows:

$$(MOP) = \begin{cases} \text{`min'} & f(x) = (f_1(x), f_2(x), \ldots, f_n(x)) \\ \text{subject to} & x \in X \end{cases} \tag{1}$$

[1] To achieve this, the minimum and the maximum value of each objective function are used in order to adaptively replace each objective function by its corresponding normalized function, so that its values lie in the interval $[0, 1]$.

Definition 1. *An objective vector $z \in Z$* weakly dominates *another objective vector $z' \in Z$ if and only if $\forall i \in \{1, 2, \ldots, n\}$, $z_i \leq z'_i$.*

Definition 2. *An objective vector $z \in Z$* dominates[2] *another objective vector $z' \in Z$ if and only if $\forall i \in \{1, 2, \ldots, n\}$, $z_i \leq z'_i$ and $\exists j \in \{1, 2, \ldots, n\}$ such as $z_j < z'_j$.*

Definition 3. *An objective vector $z \in Z$ is* non-dominated *if and only if there does not exist another objective vector $z' \in Z$ such that z' dominates z.*

A solution $x \in X$ is said to be *efficient* (or *Pareto optimal*, *non-dominated*) if its mapping in the objective space results in a non-dominated point. The set of all efficient solutions is the *efficient* (or *Pareto optimal*) *set*, denoted by X_E. The set of all non-dominated vectors is the *non-dominated front* (or the *trade-off surface*), denoted by Z_N. A possible approach in MOP solving is to find the minimal set of efficient solutions, *i.e.* one solution $x \in X_E$ for each non-dominated vector $z \in Z_N$ such as $f(x) = z$ (in case multiple solutions map to the same non-dominated point). But, generating the entire set of Pareto optimal solutions is usually infeasible due to the complexity of the underlying problem or to the large number of optima. Therefore, the overall goal is often to identify a good approximation of it. Population-based metaheuristics are commonly used to this end as they naturally find multiple and well-spread non-dominated solutions in a single simulation run. The reader could refer to [8, 9] for more details about evolutionary multi-objective optimization and to [10] for more details about multi-objective combinatorial optimization.

2.2. Problem Definition

The *Ring Star Problem* (RSP) can be described as follows. Let $G = (V, E, A)$ be a complete mixed graph where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices, $E = \{[v_i, v_j] | v_i, v_j \in V, i < j\}$ is a set of edges, and $A = \{(v_i, v_j) | v_i, v_j \in V\}$ is a set of arcs. Vertex v_1 is the depot. To each edge $[v_i, v_j] \in E$ we assign a non-negative *ring cost* c_{ij}, and to each arc $(v_i, v_j) \in A$ we assign a non-negative *assignment cost* d_{ij}. The B-RSP consists of locating a simple cycle through a subset of nodes $V' \subset V$ (with $v_1 \in V'$) while (i) minimizing

[2] We will also say that a decision vector $x \in X$ *dominates* a decision vector $x' \in X$ if $f(x)$ dominates $f(x')$.

the sum of the ring costs related to all edges that belong to the cycle, and (*ii*) minimizing the sum of the assignment costs of arcs directed from every non-visited node to a visited one so that the associated cost is minimum. An example of solution is given in Figure 1, where solid lines represent edges that belong to the ring and dashed lines represent arcs of the assignments.

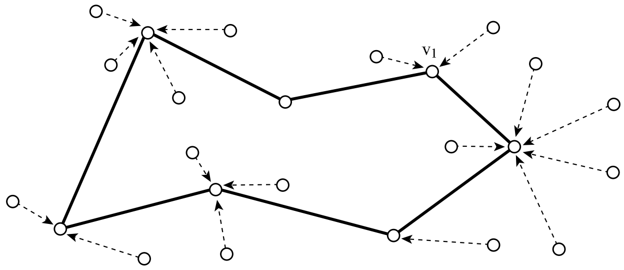

Figure 1: An example of solution for the ring star problem.

The first objective is called the *ring cost* and is defined as:

$$\sum_{[v_i,v_j]\in E} c_{ij}b_{ij} \tag{2}$$

where b_{ij} is a binary variable equal to 1 if and only if the edge $[v_i, v_j]$ belongs to the cycle. The second objective, the *assignment cost*, can be computed as follows:

$$\sum_{v_i\in V\setminus V'} \min_{v_j\in V'} d_{ij} \tag{3}$$

Let us remark that these objective functions are comparable only if we assume that the ring cost and the assignment cost are proportional one to another. But this is rarely the case in practice. Moreover, the fact of privileging a cost compared to the other is closely related to the decision-maker preferences, whereas we deal here with *a posteriori* optimization. However, the B-RSP is a NP-hard combinatorial problem because the particular case of visiting the whole set of nodes is equivalent to a traditional traveling salesman problem.

2.3. Related Works

The RSP belongs to the class of location-allocation problems aiming at locating structures in a graph (see [11] for a review). It has initially been formulated by Labbé et al. [12] in two different ways. In the first formulation (denoted MCP1 in [12] and more often called 'ring star problem'), a weighted sum of both objective functions is to be optimized. In the second formulation (MCP2 [12], usually called 'median cycle problem'), the ring cost is to be minimized while the assignment cost is bounded by a prefixed value. Even if it was not explicitly noticed by the original authors, these two formulations are commonly employed to convert a MOP into a single-objective problem by using scalar approaches. They are respectively denoted by *aggregation* and *epsilon-constraint* methods in the multi-objective optimization literature [13]. The first formulation of the problem has been more widely studied in [1]. The authors used a branch-and-bound method and successfully solved TSPLIB and randomly generated instances involving up to 200 nodes in less than two hours. In [2], the same authors solved the second formulation of the problem by a similar method. Finally, one or both versions of the problem have been heuristically tackled by a variable neighborhood tabu search [14], an evolutionary algorithm [15], a multi-start greedy add heuristic [15], and a variable neighborhood tabu search hybridized with a greedy randomized adaptive search procedure [16].

As shown in the survey of Jozefowiez et al. [3], an increasing number of multi-objective routing problems appeared in the literature in recent years. However, in spite of its numerous industrial applications (see Subsection 2.4), the RSP has never been explicitly investigated in a multi-objective fashion. Nevertheless, Current and Schilling [17] defined two multi-objective variants of a very similar problem: the *median tour problem* and the *maximal covering tour problem*. In both versions, one objective is the minimization of the total length of the tour, while another one is the maximization of the access to the tour for non-visited nodes. To tackle these problems, the authors used a kind of lexicographic method, where a hierarchy is defined between objective functions. Additionally, Dorner et al. [18] recently formulated a three-objective optimization problem of tour planning for mobile health care facilities, closely related to the *Single Vehicle Routing-Allocation Problem* introduced in [19]. A mobile facility has to visit a subset of nodes. Non-visited nodes are then assigned to their closest tour stop or are regarded as unable to reach a tour stop (within a predefined maximum distance). The considered objectives are (*i*) the minimization of the ratio between medical working time and total working time, (*ii*) the minimization of the average distance to the nearest tour stops and (*iii*) the maximization of a coverage criterion. To do so, a Pareto ant colony optimization algorithm as well as two genetic algorithms (namely VEGA [20] and MOGA [21]) were designed to solve real-world instances.

2.4. Real-world Applications and Industrial Concerns

The RSP has a wide range of industrial interests, especially in telecommunications and vehicle routing. Of course, practical real-world applications may contain additional constraints. For instance, as noticed by Labbé et al. [1, 2], a ring-based network is designed to interconnect a set of hubs in the case of digital data services [22]. Some concentrators are installed on a subset of locations and are interconnected on a ring network (the Internet) while the remaining locations are assigned to this concentrators (the Intranet). Closely related problems arise in rapid transit systems planning [1] or while designing optical networks [23]. In addition, other kinds of applications appear in the postal collection or delivery routes design, where the distance between a customer and a collection point has to be reasonable. For instance, post-box location while taking both the collection cost and the user inconvenience into account has been studied in [24]. Besides, other applications closely related to the RSP are the location of circular shaped transportation infrastructure (such as metro lines or motorways), the location of recyclable garbage collection bins, and school bus routing. Finally, the routing of essential health care services, already investigated in [18, 25] among other authors, consists of a mobile clinic servicing an area without being able to visit every population nodes. Then, unvisited ones have to reach the nearest tour stop by their own to be medically treated.

3. Metaheuristics for the Bi-objective Ring Star Problem

In this section, four population-based metaheuristics are proposed to solve the bi-objective RSP. First is an iterative local search method called IBMOLS [5]. Next, three evolutionary algorithms are designed. They consist of slight variations of IBEA [6] and NSGA-II [7] and of a general-purpose search method called SEEA and introduced in this paper for the first time. IBMOLS and IBEA are both recent indicator-based metaheuristics, whereas NSGA-II can be considered as a state-of-the-art multi-objective resolution approach. A presentation of these metaheuristics is given below, and is followed by a detailed description of problem-specific components.

3.1. A Multi-objective Local Search

Since they are easily adaptable to the multi-objective context, many search methods proposed to tackle MOPs are evolutionary algorithms. However, local search algorithms are known to be effective metaheuristics for solving real-world applications [26, 27]. Several multi-objective neighborhood search methods have been proposed in the literature, see [10, 28] for a survey. Most of them are based on a set of aggregations of the objective functions. Dominance-based local search algorithms are more rare. However, an Indicator-Based Multi-Objective Local Search (IBMOLS for short) has recently been proposed in [5]. IBMOLS can be seen as a local search variant of IBEA [6], as the same strategy is used for fitness assignment. IBMOLS is a generic population-based multi-objective local search dealing with a fixed population size. This allows to obtain a set of efficient solutions in a single simulation run without specifying any mechanism to control the number of solutions during the search process. Moreover, IBMOLS presents an alternative to aggregation- and dominance-based multi-objective local search algorithms. Indeed, as proposed in [6], it is assumed that the optimization goal is given in terms of a binary quality indicator I [29] that can be regarded as an extension of the Pareto dominance relation. A value $I(A, B)$ quantifies the difference in quality between two approximated efficient sets A and B. So, if R denotes a reference set (that can be the Pareto-optimal set X_E or any other set), the overall optimization goal can be formulated as:

$$\arg\ \min_{A \in \Omega} I(A, R) \ , \tag{4}$$

where Ω denotes the space of all efficient set approximations. As noted in [6], R does not have to be known in advance, it is just required in the formalization of the optimization goal. Since R is fixed, I actually represents a unary function that assigns a real number reflecting the quality of any approximation set according to the optimization goal. If I is dominance preserving [6], $I(A, R)$ is minimum for $A = R$. One of the main advantages of indicator-based optimization is that no additional diversity preservation mechanism is generally required, according to the indicator being used. Indeed, diversity information is usually included into the indicator, since it may quantify the quality of a set both in terms of convergence and diversity.

Principle. The IBMOLS algorithm maintains a population P of size N. Then, it generates the neighborhood of a solution contained in P until an improving solution is found (*i.e.* one that is better than at least one solution of P in terms of the indicator being used), or until all its neighbors have been explored. If an improving solution is found, it replaces the worst solution of P. This corresponds to a *'first improving'* strategy. By iterating this simple principle to every solution of P, we obtain a local search step. The whole local search procedure stops when the archive of potentially efficient solutions has not received any new item during a complete local search step. Hence, the IBMOLS method provides a natural stopping criterion.

Quality Indicator. Several quality indicators can be used in the frame of IBMOLS. Some examples can be found in [29]. One of them is the binary additive ϵ-indicator ($I_{\epsilon+}$) [6, 29], inspired by the concept of ϵ-dominance [30]. This indicator appears to be particularly well-suited to indicator-based search. And its efficiency has been experimentally proven to approximate the efficient set of different kinds of problems, see for instance [5, 6]. $I_{\epsilon+}$ computes the minimum value by which a solution $x \in X$ has to or can be translated in the objective space to weakly dominate another solution $x' \in X$. For a minimization problem, it is defined as follows:

$$I_{\epsilon+}(x, x') = \max_{i \in \{1,\ldots,n\}} (f_i(x) - f_i(x')) \ , \tag{5}$$

where n stands for the number of objective functions. Furthermore, to evaluate the quality of a solution $x \in X$ according to a whole population P, and then to compute the fitness value of x, different approaches exist. As proposed in [6], we will here consider an additive technique that amplifies the influence of dominating solutions over dominated ones. It can be outlined as follows:

$$fitness(x) = \sum_{x^\star \in P \setminus \{x\}} -e^{-I(x^\star, x)/\kappa} \ , \tag{6}$$

where $\kappa > 0$ is a fitness scaling factor, see [6] for more details. However, the initial experiments were not satisfactory because the algorithm was not able to find the extreme points of the trade-off surface. As pointed out in [31], this is known to be one of the drawbacks of the ϵ-dominance relation, apparently due to the high convexity of the front. Indeed, the authors illustrate that a limitation of ϵ-dominance is that extreme points of the Pareto front are usually lost. To tackle this problem, we add a simple condition preventing the deletion of solutions corresponding to the extreme non-dominated vectors during the replacement step of IBMOLS.

Iterative Version. Firstly, let us remind that IBMOLS handles a natural stopping condition. However, it is commonly known that iteratively repeating a local search algorithm can lead to very high performances and can significantly improve the results. In the frame of single-objective optimization, two main iterative strategies are generally distinguished, that is *random restart* and *perturbation techniques* [32]. Random restart is the simplest possibility, and a well-designed perturbation technique generally achieves better results. Furthermore, within IBMOLS, we now have to deal with a population of solutions that needs to be re-initialized between each local search phase. As proposed in [5], we decide to perturb a set of solutions from the archive to obtain the restarting population. The choice of such a strategy has been motivated by preliminary experiments as well as conclusive results obtained in [5] for another problem. Then, the population re-initialization scheme is based on random noise, such as in a basic simulated annealing [26]. This noise consists of multiple mutations applied to N different randomly chosen solutions contained in the elite set. If the size of the archive is less than N, the population is filled with random solutions, just like in random restart.

More details about IBMOLS and I-IBMOLS as well as algorithm pseudocodes can be found in [5].

3.2. Multi-objective Evolutionary Algorithms

The first two multi-objective evolutionary algorithms designed to tackle the bi-objective RSP are small variations of two state-of-the-art search methods, namely IBEA [6] and NSGA-II [7]. Some minor modifications have been carried out in order to save the whole set of non-dominated solutions found during the search process. Finally, a general-purpose approach for MOP solving, called SEEA, is proposed in this paper for the first time and is presented in details.

3.2.1. IBEA

Introduced by Zitzler and Künzli [6], the *Indicator-Based Evolutionary Algorithm* (IBEA) is, like IBMOLS, an indicator-based metaheuristic. The fitness assignment scheme of this evolutionary algorithm is based on a pairwise comparison of solutions contained in a population by using a binary quality indicator. The selection scheme for reproduction is a binary tournament between randomly chosen individuals. The replacement is based on an iterative elitist strategy that consists in deleting, one-by-one, the worst individuals, and in updating the fitness values of the remaining solutions each time there is a deletion; this is continued until the required population size is reached. Moreover, an archive stores solutions mapping to potentially non-dominated points in order to prevent their loss during the stochastic search process. However, in our case, and in contrast to the IBEA defined in [6], this archive is updated at each generation since the beginning of the evolutionary algorithm, so that the output size is not necessarily less than or equal to the population size. Just like for the IBMOLS algorithm, the indicator used within IBEA is the additive ϵ-indicator; and the same mechanism has been used to prevent the loss of the extreme points on the trade-off surface. More information about this algorithm can be found in [6].

3.2.2. NSGA-II

At each generation of NSGA-II (the *Non-dominated Sorting Genetic Algorithm II* introduced by Deb et al. in [7]), the solutions contained in the population are ranked into several classes. Individuals mapping to vectors from the first front all belong to the best efficient set; individuals mapping to vectors from the second front all belong to the second best efficient set; and so on. Two values are computed for every solution of the population. The first one corresponds to the *rank* the corresponding solution belongs to, and represents the quality of the solution in terms of convergence. The second one, the *crowding distance*, consists in estimating the density of solutions surrounding a particular point of the objective space, and represents the quality of the solution in terms of diversity. A solution is said to be better than another solution if it has a best rank value, or in case of equality, if it has the best crowding distance. The selection strategy is a deterministic tournament between two random solutions. At the replacement step, only the best individuals survive, with respect to the population size. Similarly, an external population is added to NSGA-II in order to store every potentially efficient solution found during the search. The reader is referred to [7] for more details about NSGA-II.

3.2.3. SEEA

If evaluating a solution in the objective space is not too much time consuming (which is the case for our problem), computing fitness and diversity information is generally the most computationally expensive step of a multi-objective evolutionary algorithm. Based on this observation, we here propose a simple search method for which none of these phases is required. The resulting evolutionary algorithm, called *Simple Elitist Evolutionary Algorithm* (SEEA for short), is detailed in Algorithm 1. An archive of potentially efficient solutions is updated at each generation, and the individuals contained in the main population are generated by applying variation operators to randomly chosen archive members. The replacement step is a

Algorithm 1 Simple Elitist Evolutionary Algorithm (SEEA)

```
Input:   $P$   Initial population
Output:  $A$   Efficient set approximation

Step 1:  Initialization. $A \leftarrow$ non-dominated solutions of $P$; $N \leftarrow |P|$; $P' \leftarrow \emptyset$.
Step 2:  Selection. Repeat until $|P'| = N$: randomly select an individual from $A$ and
         add it to the offspring population $P'$.
Step 3:  Variation. Apply crossover and mutation operators to individuals of the off-
         spring population $P'$.
Step 4:  Replacement. $P \leftarrow P'$; $P' \leftarrow \emptyset$.
Step 5:  Elitism. $A \leftarrow$ non-dominated solutions of $A \cup P$.
Step 6:  Termination. If a stopping criteria is satisfied return $A$, else go to Step 2.
```

generational one, *i.e.* the parent population is replaced by the offspring one. Note that the initial population can, for instance, be filled with random solutions. Thus, as proposed in [33] among other authors, the archive is not only used as an external storage, but it is integrated into the optimization process during the selection phase of the evolutionary algorithm; that is called *elitism.* Elitism is an important issue in the field of evolutionary multi-objective optimization [34] and SEEA is in somehow related to other elitist evolutionary algorithms such as SPEA [33], PESA [35] or SEAMO [36]. But, contrary to other approaches, no strategy to preserve diversity or to manage the size of the archive is involved here, as solutions are selected randomly and the archive is unbounded. Note that to be used for solving optimization problems where an exponential number of efficient solutions are involved, an additional mechanism should be designed to bound the archive size, see [37]. The biggest advantage of this evolutionary algorithm is that the population (or the population size if solutions are randomly initialized) is the only problem-independent parameter. If non-dominated solutions are relatively close to each other in the decision space and if the archive is not too small compared to the main population, we believe that SEEA may convergence to a good approximation of the efficient set in a very short runtime.

3.3. Application to the Bi-objective Ring Star Problem

This section presents the problem-specific components that are necessary to instantiate the metaheuristics introduced above for the resolution of our bi-objective RSP. Hence, the encoding mechanism, the population initialization as well as the neighborhood, the mutation and the recombination operators, are described in details below.

3.3.1. Solution Encoding

The representation of a RSP solution is based on the random keys mechanism proposed by Bean [38]. Such an encoding mechanism has already been successfully applied for solving a single-objective version of the RSP in [15]. A *random key* $k_i \in [0, 1[$ is assigned to every node v_i that belongs to the ring, with $k_1 = 0$. A special value is assigned to unvisited nodes. Thus, the ring route associated to a solution corresponds to the nodes read according to their random keys in the increasing order; *i.e.* if $k_i < k_j$, then v_j comes after v_i. A possible representation for the cycle (v_1,v_7,v_4,v_9,v_2,v_6) is given in Figure 2. Nodes v_3,v_5,v_8 and v_{10} are assigned to a visited node in such a way that the associated assignment cost is minimum.

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
Random key	0	0.7	-	0.3	-	0.8	0.2	-	0.5	-

Figure 2: A RSP solution represented by random keys.

3.3.2. Population Initialization

For each search method, the initial population has been generated randomly. Each node has a probability $p = 0.5$ to be visited or not, and to each visited vertex v_i we associate a random key k_i uniformly generated in $[0, 1[$.

3.3.3. Neighborhood and Mutation Operators

As the RSP is both a routing and an assignment problem, different move and mutation operators have to be designed. For this kind of problems, usual operators consists in removing or adding a node to the cycle. Here we also consider an operator specifically dedicated to the ring improvement by means of a 2-opt move. The resulting three operators are the following:

- *remove operator*: selects a visited node $v_i \in V'$ at random and removes it from the ring
- *insert operator*: selects an unvisited node $v_i \in V \setminus V'$ at random and adds it to the cycle, the position to insert v_i is chosen so that the increment on the ring cost is minimum

- *2-opt exchange operator*: selects two visited nodes $v_i \in V'$ and $v_j \in V'$ at random and applies a 2-opt move between v_i and v_j, so that the sequence of visited nodes located between v_i and v_j is reversed.

Note that it is not necessary to re-evaluate a solution each time a neighborhood or a mutation operator is applied. Thus, after a remove operator, we just have to re-assign the unvisited nodes previously assigned to the one that has been removed. After an insert operator, we just have to re-assign unvisited nodes in order to minimize the assignment cost. Finally, after a 2-opt exchange operator, we just have to recompute the ring cost, the assignment cost being unchanged. For the local search method, the neighbors of a given solution are randomly explored, without considering any order between these three operators. And each neighbor is at most visited once during a search step.

3.3.4. Recombination Operator

The recombination operator is a one-point crossover closely related to the one proposed in [15]. Two randomly selected solutions x_1 and x_2 are first divided according to a random position. Then, the first part of x_1 is combined with the second part of x_2 to build a first offspring, and the first part of x_2 is combined with the second part of x_1 to build a second offspring. Every node retains its random key so that it enables an easy reconstruction of the new individuals. Due to the random keys encoding mechanism, solutions having different ring sizes can easily be recombined, even if the initial ring structures are generally broken in the offspring solutions. Figure 3 illustrates a recombination between two solutions $(v_1,v_7,v_4,v_9,v_2,v_6)$ and (v_1,v_8,v_4,v_3,v_5) after vertex v_6, which gives rise to a couple of new solutions (v_1,v_8,v_4,v_2,v_6) and $(v_1,v_7,v_9,v_4,v_3,v_5)$.

Another recombination operator preserving a bigger part of the initial ring structures, has been experimented. But it was tending to reduce the number of nodes belonging to the cycle and then was causing a premature convergence with solutions having a small number of visited nodes.

3.4. Experiments

All the metaheuristics presented in this paper have been implemented using the ParadisEO-MOEO software framework[3] [39]. ParadisEO-MOEO

[3] ParadisEO is available at `http://paradiseo.gforge.inria.fr`.

Parent 1

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
Random key	0	0.7	-	0.3	-	0.8	0.2	-	0.5	-

Parent 2

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
Random key	0	-	0.8	0.7	0.9	-	-	0.2	-	-

Offspring 1

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
Random key	0	0.7	-	0.3	-	0.8	-	0.2	-	-

Offspring 2

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
Random key	0	-	0.8	0.7	0.9	-	0.2	-	0.5	-

Figure 3: Recombination operator.

is a C++ white-box object-oriented framework dedicated to the reusable design of metaheuristics for multi-objective optimization. All the algorithms share the same base components for a fair comparison between them. Computational runs were performed on an Intel Core 2 Duo 6600 (2 × 2.40 GHz) machine, with 2 GB RAM.

3.4.1. Experimental Protocol

Benchmark Test Instances. Experiments were conducted on a set of eight benchmark instances taken from the TSPLIB[4] [40]. These instances contain between 51 and 299 nodes. The number at the end of an instance name represents the number of nodes involved for the instance under consideration. Let l_{ij} denote the distance between two nodes v_i and v_j of a TSPLIB file. The ring cost c_{ij} and the assignment cost d_{ij} have both been set to l_{ij} for every pair of nodes v_i and v_j.

Stopping Conditions. From our point of view, there does not exist a standard approach to define an unquestionable condition for stopping a multi-objective metaheuristic. Common strategies are quite basic and generally consist of an arbitrary user-given number of iterations or evaluations. However, in the frame of this paper, an evolutionary algorithm iteration has nothing to do with a local search iteration. Moreover, computing the number of evaluations performed from the beginning of the search process does not make any sense here, as mutated or neighbor solutions are not fully, but incrementally

[4] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

evaluated. Therefore, in this study, the search process simply stops after a fixed amount of runtime. Indeed, all methods share the same base components and their respective implementations can then be compared in term of computational time. But, as small instances are supposedly easier to solve than large ones, this stopping criteria has been arbitrary set according to the size of the instance under consideration, as shown in Table 1. Note that the maximum runtime available stands for a single simulation run *per* instance and *per* algorithm.

Table 1: Stopping criteria: runtime *per* simulation run.

Instance	Runtime	Instance	Runtime
eil51	20"	*kroA150*	10'
st70	1'	*kroA200*	20'
kroA100	2'	*pr264*	30'
bier127	5'	*pr299*	50'

Performance Assessment. In the frame of multi-objective optimization, the performance assessment of a number of algorithms in solving the same problem is a key issue. In this study, a set of 20 runs *per* instance, with different initial populations, has been performed for each search method. In order to evaluate the quality of the non-dominated front approximations for every instance we experimented, we follow the protocol given by Knowles et al. in [41]. For a given instance, let Z^{all} denote the union of the outputs we obtained during all our experiments. Note that this set probably contains both dominated and non-dominated points, as a given approximation may contain vectors dominating the ones of another approximation, and vice versa. We first compute a reference set $Z_N^\star$ containing all the non-dominated points of Z^{all}. Second, we define $z^{max} = (z_1^{max}, z_2^{max})$, where z_1^{max} (respectively z_2^{max}) denotes the upper bound of the first (respectively second) objective for all the points contained in Z^{all}.

Now, to measure the quality of an output set A in comparison to $Z_N^\star$, we compute the difference between these two sets by using the unary hypervolume metric [33], z^{max} being the reference point. As illustrated in Figure 4, the hypervolume difference indicator (I_H^-) computes the portion of the objective space that is weakly dominated by $Z_N^\star$ and not by A, cf. Definition 1; the closer this measure to 0, the better the approximation A. Furthermore, we also consider the additive ϵ-indicator proposed in [29]. Contrary to the one proposed in Equation 5, this indicator is used to compare non-dominated

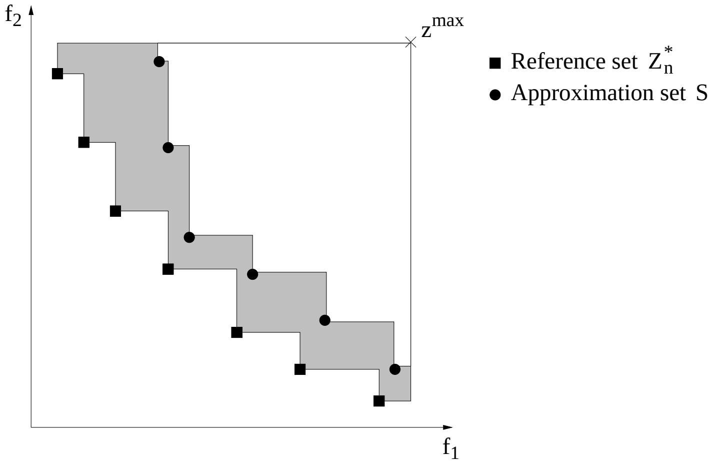

Figure 4: Illustration of the hypervolume difference (I_H^-) between a reference set $Z_N^\star$ and a non-dominated front approximation S (shaded area).

set approximations, and not solutions. The unary additive ϵ-indicator ($\mathrm{I}_{\epsilon+}^1$) gives the minimum factor by which an approximation A has to be translated in the objective space to weakly dominate the reference set $Z_N^\star$. $\mathrm{I}_{\epsilon+}^1$ can then be defined as follows:

$$I_{\epsilon+}^1(A) = \min_{\epsilon}\{\forall z \in Z_N^\star, \exists z' \in A : z_i' - \epsilon \leq z_i, \forall 1 \leq i \leq n\} \ . \tag{7}$$

As a consequence, for each test instance, we obtain 20 I_H^- measures and 20 $\mathrm{I}_{\epsilon+}$ measures, corresponding to the 20 runs, *per* algorithm. As suggested by Knowles et al. [41], once all these values are computed, we perform a statistical analysis for a pairwise comparison of methods. To this end, we use the Wilcoxon signed rank test. Such a non-parametric statistical test is motivated by the fact that the samples collected here can be considered as matched samples. Indeed, for a given run, both the initial population and the random seed are identical for all algorithms, so that the final indicator values can be taken as pairs. Details for this statistical testing procedure are given in [41]. Hence, for a given test instance, and according to the p-value and to the metric under consideration, this statistical test reveals if the sample of approximation sets obtained by a given search method is significantly better than the ones of another search method, or if there is no significant difference between both. Note that all the performance assessment procedures have been achieved using the performance assessment tool suite provided in PISA[5] [42].

3.4.2. Parameter Setting

A preliminary experimental phase has been performed to determine the following parameters, see [43] for more details. Hence, as no general trend has been identified for some algorithms, the population size were fixed according to the instance under consideration, as shown in Table 2. The noise rate for I-IBMOLS has been set to a fixed percentage of the number of nodes involved in the problem at hand. Next, following [6], the scaling factor κ of Equation 6 has been set to 0.05 for indicator-based metaheuristics, *i.e.* IBEA and I-IBMOLS. The remainder parameters are shared by all evolutionary algorithms and consist of a crossover probability of 0.25, and of a mutation probability of 1.00, with rates of 0.25, 0.25 and 0.50 for the *remove*, the *insert* and the *2-opt exchange* operator, respectively.

Table 2: Parameter setting.

	I-IBMOLS		IBEA	NSGA-II	SEEA
Instance	pop. size	noise rate	pop. size	pop. size	pop.size
eil51	20	10%	100	100	100
st70	20	10%	50	100	100
kroA100	30	10%	100	200	100
bier127	30	10%	100	200	100
kroA150	30	10%	200	200	100
kroA200	30	10%	200	200	100
pr264	30	20%	50	200	100
pr299	50	10%	50	200	100

3.4.3. Computational Results

Table 3 and Table 4 provide a comparison of SEEA, NSGA-II, IBEA and I-IBMOLS with respect to the I_H^- metric and to the $\mathrm{I}_{\epsilon+}^1$ metric, respectively. According to the experimental protocol used in the paper, I-IBMOLS is never statistically outperformed by any other algorithm with respect to both metrics. The only exceptions are for the *st70* and the *pr264* instances, where SEEA obtained better results with respect to the $\mathrm{I}_{\epsilon+}^1$ metric. About NSGA-II, it is outperformed by all the other algorithms on every test instances, except for *pr264* where there is no significant difference between its results

[5] The package is available at `http://www.tik.ee.ethz.ch/pisa/assessment.html`.

Table 3: Algorithms comparison with respect to the I_H^- metric. The number in brackets denotes the average metric value for the algorithm and the instance under consideration (multiplied by 10^{-3}). For each instance, either the algorithm located at a specific row significantly dominates the algorithm located at a specific column ($\succ$ or $\succeq$ for a p-value less or equal to 0.01 or to 0.05, respectively), either it is significantly dominated ($\prec$ or $\preceq$ for a p-value less or equal to 0.01 or to 0.05, respectively), or there is no significant difference between both ($\equiv$).

			I-IBMOLS	IBEA	NSGA-II	SEEA
eil51	I-IBMOLS	(4.456)	-	$\succ$	$\succ$	$\equiv$
	IBEA	(6.710)	$\prec$	-	$\succ$	$\prec$
	NSGA-II	(12.573)	$\prec$	$\prec$	-	$\prec$
	SEEA	(4.957)	$\equiv$	$\succ$	$\succ$	-
st70	I-IBMOLS	(3.143)	-	$\succ$	$\succ$	$\succ$
	IBEA	(4.037)	$\prec$	-	$\succ$	$\equiv$
	NSGA-II	(8.920)	$\prec$	$\prec$	-	$\prec$
	SEEA	(3.718)	$\prec$	$\equiv$	$\succ$	-
kroA100	I-IBMOLS	(4.251)	-	$\succeq$	$\succ$	$\succ$
	IBEA	(5.273)	$\preceq$	-	$\succ$	$\equiv$
	NSGA-II	(12.370)	$\prec$	$\prec$	-	$\prec$
	SEEA	(5.015)	$\prec$	$\equiv$	$\succ$	-
bier127	I-IBMOLS	(3.219)	-	$\succ$	$\succ$	$\succ$
	IBEA	(4.236)	$\prec$	-	$\succ$	$\succ$
	NSGA-II	(9.606)	$\prec$	$\prec$	-	$\prec$
	SEEA	(6.751)	$\prec$	$\prec$	$\succ$	-
kroA150	I-IBMOLS	(3.959)	-	$\succ$	$\succ$	$\succ$
	IBEA	(4.562)	$\prec$	-	$\succ$	$\equiv$
	NSGA-II	(8.973)	$\prec$	$\prec$	-	$\prec$
	SEEA	(4.747)	$\prec$	$\equiv$	$\succ$	-
kroA200	I-IBMOLS	(2.875)	-	$\equiv$	$\succ$	$\succ$
	IBEA	(2.980)	$\equiv$	-	$\succ$	$\succ$
	NSGA-II	(8.515)	$\prec$	$\prec$	-	$\prec$
	SEEA	(3.822)	$\prec$	$\prec$	$\succ$	-
pr264	I-IBMOLS	(1.535)	-	$\succ$	$\succ$	$\equiv$
	IBEA	(1.912)	$\prec$	-	$\succ$	$\prec$
	NSGA-II	(3.663)	$\prec$	$\prec$	-	$\prec$
	SEEA	(1.520)	$\equiv$	$\succ$	$\succ$	-
pr299	I-IBMOLS	(1.303)	-	$\succ$	$\succ$	$\succ$
	IBEA	(1.964)	$\prec$	-	$\succ$	$\equiv$
	NSGA-II	(4.185)	$\prec$	$\prec$	-	$\prec$
	SEEA	(2.179)	$\prec$	$\equiv$	$\succ$	-

Table 4: Algorithms comparison with respect to the $\mathrm{I}^1_{\epsilon+}$ metric. The number in brackets denotes the average metric value for the algorithm and the instance under consideration (multiplied by 10^{-3}). For each instance, either the algorithm located at a specific row significantly dominates the algorithm located at a specific column ($\succ$ or $\succeq$ for a p-value less or equal to 0.01 or to 0.05, respectively), either it is significantly dominated ($\prec$ or $\preceq$ for a p-value less or equal to 0.01 or to 0.05, respectively), or there is no significant difference between both ($\equiv$).

			I-IBMOLS	IBEA	NSGA-II	SEEA
eil51	I-IBMOLS	(9.307)	-	$\succ$	$\succ$	$\equiv$
	IBEA	(12.094)	$\prec$	-	$\succ$	$\prec$
	NSGA-II	(19.165)	$\prec$	$\prec$	-	$\prec$
	SEEA	(9.613)	$\equiv$	$\succ$	$\succ$	-
st70	I-IBMOLS	(7.321)	-	$\succ$	$\succ$	$\preceq$
	IBEA	(10.334)	$\prec$	-	$\succ$	$\prec$
	NSGA-II	(13.639)	$\prec$	$\prec$	-	$\prec$
	SEEA	(6.298)	$\succeq$	$\succ$	$\succ$	-
kroA100	I-IBMOLS	(9.833)	-	$\succeq$	$\succ$	$\equiv$
	IBEA	(11.771)	$\preceq$	-	$\succ$	$\preceq$
	NSGA-II	(17.718)	$\prec$	$\prec$	-	$\prec$
	SEEA	(9.606)	$\equiv$	$\succeq$	$\succ$	-
bier127	I-IBMOLS	(8.421)	-	$\succ$	$\succ$	$\succ$
	IBEA	(11.993)	$\prec$	-	$\succ$	$\succ$
	NSGA-II	(21.522)	$\prec$	$\prec$	-	$\preceq$
	SEEA	(19.377)	$\prec$	$\prec$	$\succeq$	-
kroA150	I-IBMOLS	(7.853)	-	$\succ$	$\succ$	$\succ$
	IBEA	(10.708)	$\prec$	-	$\succ$	$\prec$
	NSGA-II	(13.383)	$\prec$	$\prec$	-	$\prec$
	SEEA	(9.056)	$\prec$	$\succ$	$\succ$	-
kroA200	I-IBMOLS	(7.829)	-	$\equiv$	$\succ$	$\equiv$
	IBEA	(7.288)	$\equiv$	-	$\succ$	$\succ$
	NSGA-II	(14.473)	$\prec$	$\prec$	-	$\prec$
	SEEA	(8.204)	$\equiv$	$\prec$	$\succ$	-
pr264	I-IBMOLS	(5.259)	-	$\succ$	$\succ$	$\prec$
	IBEA	(9.055)	$\prec$	-	$\equiv$	$\prec$
	NSGA-II	(8.403)	$\prec$	$\equiv$	-	$\prec$
	SEEA	(4.343)	$\succ$	$\succ$	$\succ$	-
pr299	I-IBMOLS	(4.023)	-	$\succ$	$\succ$	$\succ$
	IBEA	(8.993)	$\prec$	-	$\succeq$	$\prec$
	NSGA-II	(10.403)	$\prec$	$\preceq$	-	$\prec$
	SEEA	(5.768)	$\prec$	$\succ$	$\succ$	-

and the ones of IBEA according to the $\mathrm{I}^1_{\epsilon+}$ metric. At last, with respect to the I^-_H metric, the results of IBEA and SEEA are quite heterogeneous, so that no general trend can be identified. However, according to the $\mathrm{I}^1_{\epsilon+}$ metric, it seems that SEEA is more efficient that IBEA on most test instances.

3.5. Discussion

We can conclude that the iterative version of IBMOLS is globally significantly better than all the evolutionary algorithms we investigated. Nevertheless, the behavior of such a simple search method as SEEA in comparison to state-of-art algorithms like IBEA and NSGA-II is very encouraging with regard to combinatorial problem solving. One of the main characteristics of the problem under consideration seems to be the high number of points located in the trade-off surface. Then, after a couple of iterations, a large part of the population involved in I-IBMOLS, IBEA and NSGA-II might map to non-dominated points. That could explain the low efficiency of NSGA-II. Indeed, since the same fitness value is assigned to the major part of the population, only the crowding distance is used to compare solutions. The indicator-based fitness assignment scheme of I-IBMOLS and IBEA is obviously much more suited to discriminate potentially efficient solutions than the single crowding distance. Note that this is not the case within SEEA because all non-dominated solutions contained in the archive can potentially take part in the evolution engine. However, the good performance of I-IBMOLS might also depend on how close are solutions mapping to non-dominated points in the decision space. If these solutions are close to each other according to the neighborhood operators, a local search method is known to be particularly well-suited to find additional interesting solutions. As a next step, it could then be interesting to design a cooperation scheme between two different search methods in order to benefit of their individual features.

4. Cooperative Approaches for the Bi-objective Ring Star Problem

This section presents a general-purpose cooperative approach combining SEEA and the non-iterative version of IBMOLS for multi-objective combinatorial optimization. Two variants are proposed: a periodic one that operates a systematic cooperation and an adaptive one that decides on-line when the cooperation must occur. These hybrid models are next experimented on the bi-objective RSP investigated in this paper.

4.1. Motivations

Designing metaheuristics for solving combinatorial optimization problems is generally a matter of intensification and diversification. This is even more pronounced for MOPs where the goal is to find a well-converged and well-diversified efficient set approximation. However, local search methods are known to be particularly efficient as intensifying methods whereas evolutionary algorithms are clearly powerful to explore the decision space thanks to their variation operators. Instead of trying to improve one method in term of diversification or the other in term of intensification, a common approach is to hybridize both in order to make them cooperate and then to benefit of their respective behaviors. Thus, hybrid metaheuristics have shown their efficiency to solve different optimization problems [44], including MOPs [45].

In the previous section, we saw that SEEA and I-IBMOLS were the overall more efficient method to approximate the efficient set for the problem under consideration. Moreover, these two methods are quite different to each other and do not explore the search space in the same way. Indeed, SEEA has been conceived in order to find a rough approximation of the Pareto set in a very short amount of time whereas the non-iterative version of IBMOLS is able to improve an approximated set in a very efficient way. It could then be interesting to design a cooperation scheme between these two algorithms. The resulting hybrid metaheuristic could be particularly efficient for solving large size problems. Furthermore, both methods maintain a secondary population (the archive) in parallel of the main population to store non-dominated solutions. This archive is not only used as an external storage, but also takes part in the evolution engine as it serves to build new solutions to explore. Thus, each method can manage its own population and therefore use the archive as a single shared memory.

4.2. Cooperative Schemes

The general idea of our hybridization scheme is to run SEEA and to launch IBMOLS regularly by using a subset of archive items as an initial population. Since the non-iterative version of IBMOLS naturally stops when its own archive does not receive any new efficient solution anymore, we can restart the SEEA process until the next step of the hybrid algorithm. A step can, for instance, be defined by a certain amount of time or by a certain number of generations. Besides, as SEEA uses the non-dominated solutions found by IBMOLS to create new ones and vice versa, the global archive is the only memory shared by the two search agents to exchange information.

Resulting from this, we can imagine two versions of the hybrid algorithm: (i) a *periodic version*, in which IBMOLS is launched at each step, and (ii) an *adaptive version*, in which IBMOLS is launched at a specific step only if a condition is verified. These two approaches will be denoted by PCS (for *Periodic Cooperative Search*) and ACS (for *Adaptive Cooperative Search*) in the remainder of the paper, and are respectively illustrated in Figure 5 and Figure 6.

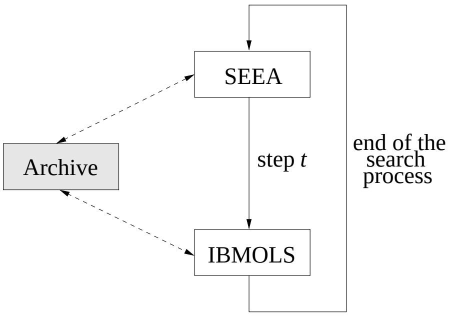

Figure 5: Illustration of the Periodic Cooperative Search (PCS).

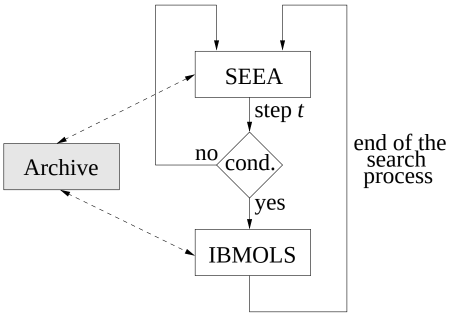

Figure 6: Illustration of the Adaptive Cooperative Search (ACS).

The ACS method decides by itself, and on-line, if it is interesting to launch IBMOLS at a given step of the search process. The condition outlined here is that the archive of potentially efficient solutions does not improve enough with regards to the optimization scenario. A possibility is to measure the quality of the current archive A^t in comparison to the one of the previous step A^{t-1}. Different metrics exist to evaluate the convergence properties of an approximated efficient set in comparison to another. For instance, the hypervolume metric could have been envisaged, but its computation has the drawback of being time consuming. Thus, let us introduce the *contribution* metric C proposed by Meunier et al. [46]. This metric gives an idea of the quality of an approximated efficient set in comparison to another one in term of convergence, and can be computed in a reasonable runtime. In our case, at each step t of the ACS, we compute the contribution of the current archive A^t on the archive of the previous step A^{t-1}. Thus, as non-dominated solutions are not lost between two steps, A^t is at least as good as A^{t+1} , so that we know that $C(A^t, A^{t-1}) \in [0.5, 1]$ due to the contribution metric properties; see [46]. Assuming that the archive does not improve enough if the contribution of A^t on A^{t-1} is less than a user-given threshold $\delta \in [0.5, 1]$, we choose to launch

IBMOLS only if $C(A^t, A^{t-1}) \leq \delta$. Let us remark that an ACS with a $\delta = 0.5$ is equivalent to SEEA, and that an ACS with a $\delta = 1.0$ is equivalent to PCS, the time spent in calculating the different contribution values in less.

Thus, thanks to the definition of a step t and to the δ parameter, it is possible to intentionally bias the balance between evolutionary search and local search within both PCS and ACS. This issue was pointed out in [47] for hybrid multi-objective metaheuristics. For instance, the higher the δ value, the more often the local search will be launched in the frame of the ACS method. Hence, the proposed hybrid algorithm can directly handle different specifications of the evolutionary-local search balance, *i.e.* from almost pure SEEA ($\delta = 0.5$) to almost pure IBMOLS ($\delta = 1.0$).

4.3. Related Works

Different schemes exist on how two search methods can be combined. According to the taxonomy proposed in [44], the hybrid metaheuristic investigated in this paper can be classified on the *High-level Teamwork Hybrid* (HTH) class, and can then be denoted by HTH(SEEA+IBMOLS). In their survey on hybrid metaheuristics to solve combinatorial MOPs [45], Ehrgott and Gandibleux identify three categories of methods hybridizing an evolutionary algorithm with a neighborhood search algorithm: (*i*) an hybridization to make a method more aggressive, (*ii*) an hybridization to drive a method and (*iii*) an hybridization for exploiting complementary strengths. The last one consists of alternating between both search methods, which is the case within our hybridization. But, most existing approaches occur in a pipeline way: first the evolutionary algorithm, then the local search. The few teamwork hybridization techniques from the literature are most often compound of the same metaheuristic, like in the island model. Moreover, this class of hybrid methods often uses a neighborhood search method combining a set of scalarizing functions. The originality of the approach proposed in this paper is that search agents are based on different types of metaheuristics, are hybridized in a teamwork mode, and do not use any scalar approach to convert the multi-objective vector function into a single-objective one. Furthermore, one of the variants we propose, the adaptive one, automatically detects when to start the local search according to the optimization scenario.

4.4. Experiments

In order to experiment the efficiency of our two cooperative approaches, we compare them to SEEA and to the iterative version of IBMOLS by using the same experimental protocol as the one defined in Section 3.4.1. The parameter setting for each hybrid method is followed by some computational results and by a discussion on the contribution of the hybridization for the resolution of the RSP as a bi-objective problem. Finally, we give an idea of the behavior of our approaches in comparison to the state-of-art single-objective approach proposed in [1] for the problem under study.

4.4.1. Parameter Setting

For both cooperative search methods, the population size managed by SEEA is set to 100, and the population size managed by IBMOLS is set according to the instance under consideration. These sizes have been set on the same way that for the stand-alone iterative version of IBMOLS on the previous section, see Table 2. But, for large-size instances, initial experiments were not satisfying since the hybrid algorithms were generally not able to launch IBMOLS more than once during the search process, as it was too much time consuming. For this reason, we bounded the IBMOLS population size to 30. Then, a IBMOLS population of 20 individuals has been set for instances with less than 100 nodes, and a IBMOLS population of 30 individuals has been set for instances with 100 nodes and more. Moreover, note that in addition to the benchmark test instances we investigated previously, we experimented two larger TSPLIB problems, namely *pr439* and *pr1002*. Following the general trend identified in the previous section, we set the instance-specific parameters for I-IBMOLS as follows: a population of 70 and 100 individuals respectively, and a noise rate of 10%. The other parameters were set in the same way than for other instances. The step t of the hybrid algorithms has been set to 0.5% of the maximum runtime available for the instance under consideration. In [43], we investigated different δ-values for the ACS method: 0.6, 0.7, 0.8 and 0.9. As pointed out above, an ACS with a δ-value of 0.5 and 1.0 is similar to PCS and SEEA, respectively. This has lead to an effective δ-value of 0.8 for the stopping criterion we adopted.

Table 5: Stopping criteria for additional test instances. For other instances, see Table 1.

Instance	Runtime
pr439	50'
pr1002	50'

4.4.2. Computational Results

Table 6 and Table 7 give a comparison between the results obtained by SEEA, I-IBMOLS, PCS and ACS according to the I_H^- and $\mathrm{I}_{\epsilon+}^1$ metric, respectively. Firstly, except for the *eil51* instance, PCS and ACS both always statistically outperform SEEA with respect to, at least, one metric. In comparison to I-IBMOLS, PCS and ACS often obtain better results, in particular for instances with 150 nodes and more. However, in any case, both hybrid metaheuristics are never statistically outperformed by a stand-alone search method. Thus, the benefit of the cooperation scheme appears relevant in most large-size instances, but the addition of an adaptive mechanism does not seem to have a big influence on the results. Indeed, for almost every test instance, the outcome measures did not return a statistically significant difference between PCS and ACS. The few exceptions to notice are for the *bier127* and the *pr439* instances where PCS obtain better results with regard to one or both metrics, and for the *pr1002* instance where ACS outperforms PCS with respect to I_H^-. However, during our experiments, we observed that the difference between the average number of times that IBMOLS is launched during the search process of PCS and ACS is relatively thin. This can be explained by the fact that (*i*) IBMOLS takes more time to find non-dominated solutions by starting with a population of poorer quality, that is the case in PCS comparing to ACS (at least for the first launch), and (*ii*) a part of the runtime allocated to the algorithm is used to compute a contribution value at every step of ACS, whereas PCS devotes all of its runtime to the search process. These two aspects lead to the fact that the number of times that IBMOLS is launched is in the end, more or less balanced between both cooperative methods.

Comparison with Exact Single-objective Results. As a last step, we provide a comparison between the results found for the bi-objective RSP investigated in this paper and the ones of the single-objective RSP investigated in [1], where both costs are summed up[6]. In order to provide optimal solutions visiting approximately 25, 50, 75 and 100% of the total number of nodes, the authors set the ring cost c_{ij} and the assignment cost d_{ij} between two nodes v_i and v_j in the following way: $c_{ij} = \lceil \alpha l_{ij} \rceil$ and $d_{ij} = \lceil (10 - \alpha) l_{ij} \rceil$ with

[6] Note that it was not possible to compare our results to the ones of the other formulation of a single-objective RSP investigated in [2, 14, 15], where the assignment cost is subject to a constraint, due to the way the bound has been fixed.

Table 6: Algorithms comparison with respect to the I_H^- metric. The number in brackets denotes the average metric value for the algorithm and the instance under consideration (multiplied by 10^{-3}). For each instance, either the algorithm located at a specific row significantly dominates the algorithm located at a specific column ($\succ$ or $\succeq$ for a p-value less or equal to 0.01 or to 0.05, respectively), either it is significantly dominated ($\prec$ or $\preceq$ for a p-value less or equal to 0.01 or to 0.05, respectively), or there is no significant difference between both ($\equiv$).

			I-IBMOLS	SEEA	PCS	ACS
eil51	PCS	(4.751)	$\equiv$	$\equiv$	-	$\equiv$
	ACS	(4.482)	$\equiv$	$\equiv$	$\equiv$	-
st70	PCS	(2.691)	$\succeq$	$\succ$	-	$\equiv$
	ACS	(2.865)	$\equiv$	$\succ$	$\equiv$	-
kroA100	PCS	(3.738)	$\equiv$	$\succ$	-	$\equiv$
	ACS	(3.326)	$\succeq$	$\succ$	$\equiv$	-
bier127	PCS	(3.071)	$\equiv$	$\succ$	-	$\succ$
	ACS	(3.693)	$\equiv$	$\succ$	$\prec$	-
kroA150	PCS	(2.792)	$\succ$	$\succ$	-	$\equiv$
	ACS	(2.624)	$\succ$	$\succ$	$\equiv$	-
kroA200	PCS	(2.247)	$\succ$	$\succ$	-	$\equiv$
	ACS	(2.260)	$\succ$	$\succ$	$\equiv$	-
pr264	PCS	(1.342)	$\succ$	$\succ$	-	$\equiv$
	ACS	(1.404)	$\succeq$	$\succ$	$\equiv$	-
pr299	PCS	(1.277)	$\equiv$	$\succ$	-	$\equiv$
	ACS	(1.293)	$\equiv$	$\succ$	$\equiv$	-
pr439	PCS	(0.348)	$\succ$	$\succ$	-	$\succ$
	ACS	(0.733)	$\succ$	$\equiv$	$\prec$	-
pr1002	PCS	(2.449)	$\succ$	$\succ$	-	$\prec$
	ACS	(0.707)	$\succ$	$\succ$	$\succ$	-

Table 7: Algorithms comparison with respect to the $I^1_{\epsilon+}$ metric. The number in brackets denotes the average metric value for the algorithm and the instance under consideration (multiplied by 10^{-3}). For each instance, either the algorithm located at a specific row significantly dominates the algorithm located at a specific column ($\succ$ or $\succeq$ for a p-value less or equal to 0.01 or to 0.05, respectively), either it is significantly dominated ($\prec$ or $\preceq$ for a p-value less or equal to 0.01 or to 0.05, respectively), or there is no significant difference between both ($\equiv$).

			I-IBMOLS	SEEA	PCS	ACS
eil51	PCS	(9.561)	$\equiv$	$\equiv$	-	$\equiv$
	ACS	(9.363)	$\equiv$	$\equiv$	$\equiv$	-
st70	PCS	(6.328)	$\succeq$	$\equiv$	-	$\equiv$
	ACS	(7.064)	$\equiv$	$\equiv$	$\equiv$	-
kroA100	PCS	(8.963)	$\equiv$	$\equiv$	-	$\equiv$
	ACS	(7.533)	$\succ$	$\succ$	$\equiv$	-
bier127	PCS	(8.114)	$\equiv$	$\succ$	-	$\succ$
	ACS	(9.818)	$\equiv$	$\succ$	$\prec$	-
kroA150	PCS	(5.450)	$\succ$	$\succ$	-	$\equiv$
	ACS	(5.587)	$\succ$	$\succ$	$\equiv$	-
kroA200	PCS	(5.057)	$\succ$	$\succ$	-	$\equiv$
	ACS	(5.702)	$\succ$	$\succ$	$\equiv$	-
pr264	PCS	(4.242)	$\succ$	$\equiv$	-	$\equiv$
	ACS	(4.317)	$\succ$	$\equiv$	$\equiv$	-
pr299	PCS	(4.501)	$\equiv$	$\succ$	-	$\equiv$
	ACS	(4.048)	$\equiv$	$\succ$	$\equiv$	-
pr439	PCS	(2.760)	$\succ$	$\succ$	-	$\succ$
	ACS	(5.553)	$\succeq$	$\succ$	$\prec$	-
pr1002	PCS	(5.391)	$\succ$	$\succ$	-	$\equiv$
	ACS	(4.309)	$\succ$	$\succ$	$\equiv$	-

$\alpha \in \{3, 5, 7, 9\}$, where l_{ij} denotes the distance between v_i and v_j given in the TSPLIB files. In order to give a rough idea of the results we obtained, we compare the best found scalar value as detailed above to the optimum[7] found in [1]. Additionally, we compare the best solution visiting every node we have found with the optimal TSP solution available on the TSPLIB website[8]. Table 8 gives the error ratio between the best known value and the best one we have found for every identified (single) objective and every benchmark test instance. In comparison to [1], this ratio is always under 1.5%, and is mostly below 1% for every instance. The optimum is even found for the *kroA100* and the *kroA150* instances with an $\alpha = 9$, and a better solution is found for the *kroA200* instances with an $\alpha = 3$ and an $\alpha = 9$. As regards to optimal TSP solutions, our results are quite close for instances with 200 nodes or less. For larger instances, they are not quite as good, especially for the *pr1002* instance, where the error ratio is close to 15%. To summarize, in comparison to single-objective optimal or near optimal results, the search methods we proposed in this paper to solve the B-RSP are quite promising with regard to the relatively small computational time available and to the size of the problem instances to be solved. Nevertheless, let us remind that the comparison is here done with the best results we obtained during the whole set of experiments we performed, and may not be as good for a single simulation run.

4.5. Discussion

Two general-purpose cooperative schemes combining SEEA and the non-iterative version of IBMOLS have been designed in this section for multi-objective combinatorial optimization: a periodic one (PCS), and an adaptive one (ACS). Indeed, ACS evolves adaptively according to the search scenario and decides by itself, and on-line, when the cooperation must occur. Another benefit of ACS is the possibility of deliberately orienting the balance between evolutionary search and local search. In comparison to stand-alone metaheuristics, these two hybrid search methods statistically improve the results on a large number of RSP instances, and particularly on large-size ones. However, the efficiency difference between PCS and ACS is almost

[7] In fact, the authors imposed a time limit for their experiments. They report the best solution found so far for the instances exceeding this time limit, which is the case for the *kroA200* instance with an $\alpha = 3$, 5 and 9 in Table 8.

[8] `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`.

Table 8: Error ratio between the cost value of the best known TSP solution and the best single-objective ring cost value found in [1] in comparison to the best value found during our experiments.

Instance	Optimal TSP solution	Optimal RSP solution [1]			
		$\alpha = 3$	$\alpha = 5$	$\alpha = 7$	$\alpha = 9$
eil51	0.67%	0.67%	0.75%	0.37%	0.69%
st70	0.31%	0.31%	0.49%	0.56%	0.42%
kroA100	0.08%	0.08%	0.05%	0.12%	**0.00%**
bier127	0.64%	0.64%	0.34%	0.52%	0.01%
kroA150	1.38%	1.01%	1.04%	0.17%	**0.00%**
kroA200	1.22%	**-4.82%**	1.05%	0.69%	**-1.55%**
pr264	2.26%	-	-	-	-
pr299	1.72%	-	-	-	-
pr439	4.63%	-	-	-	-
pr1002	14.51%	-	-	-	-

negligible. This can be explained by the fact that ACS spends, at each step, a significant time to compute whether the cooperation should occur or not. This is not the case in the frame of PCS, so that the latter devotes all of its computational time to the search process. Moreover, in comparison to single-objective approaches from the literature, the algorithms proposed in this paper for the bi-objective RSP seem to provide good-quality solutions.

5. Further Comments

A new multi-objective routing problem, the bi-objective ring star problem, has been investigated in this paper for the first time. It aims at locating a cycle through a subset of nodes of a graph while minimizing a ring cost, related to the length of the cycle, and an assignment cost, from non-visited nodes to visited ones. In spite of its clear bi-objective nature, this problem has always been addressed in a single-objective way, either where both costs are combined [1], or where one cost is regarded as a constraint [2, 15]. As a first step, we proposed a set of four population-based metaheuristics to approximate the efficient set for the problem under consideration, namely I-IBMOLS [5], IBEA [6], NSGA-II [7], and SEEA, proposed here for the first time. We concluded that I-IBMOLS was the overall most competitive method, and that SEEA was a solid competitor. As a second step, we designed two general-purpose cooperative schemes between SEEA and the non-iterative version of IBMOLS for multi-objective combinatorial optimization problems. And we illustrated the contribution of these hybridization mechanisms to improve the performance of stand-alone methods.

Although the approaches proposed in this paper are already promising, a few research directions are still open. First is the possibility to improve the population initialization strategy used within every search method. Second, we pointed out that the recombination operator designed for the problem under study has a tendency to break the parent ring structures in the offspring individuals. Then, as proposed in [15], we could employ a traveling salesman problem heuristic to improve the ring cost of newly generated solutions. Third, given the number of mutation operators designed within the evolutionary algorithms, it would be interesting to determine the appropriate rate for each one of them in an adaptive way, such as in [48]. Note that the last two points could largely improve the efficiency of evolutionary algorithms, and then would be beneficial for the hybrid methods as well. Finally, we found out that SEEA was a good alternative to state-of-the-art multi-objective evolutionary algorithms when a given, relatively small, amount of computational time is available. In the future, we plan to tackle other kinds of combinatorial problems within SEEA to verify if our observations are still valid, especially for problems where more than two objectives are involved. The same remark can also be done with regard to the cooperative approaches proposed in this paper.

Acknowledgment

A subpart of this work has already been published in [4]. The authors would like to acknowledge Matthieu Basseur and Edmund K. Burke for their helpful contribution. Furthermore, the authors are grateful for valuable discussions held with Nicolas Jozefowiez and Clarisse Dhaenens.

References

[1] M. Labbé, G. Laporte, I. Rodríguez Martín, J. J. Salazar González, The ring star problem: Polyhedral analysis and exact algorithm, Networks 43 (2004) 177–189.

[2] M. Labbé, G. Laporte, I. Rodríguez Martín, J. J. Salazar González, Locating median cycles in networks, European Journal of Operational Research 160 (2) (2005) 457–470.

[3] N. Jozefowiez, F. Semet, E.-G. Talbi, Multi-objective vehicle routing problems, European Journal of Operational Research 189 (2) (2008) 293–309.

[4] A. Liefooghe, L. Jourdan, M. Basseur, E.-G. Talbi, E. K. Burke, Metaheuristics for the bi-objective ring star problem, in: Eighth European Conference on Evolutionary Computation in Combinatorial Optimisation (EvoCOP 2008), Vol. 4472 of Lecture Notes in Computer Science, Springer-Verlag, Napoli, Italy, 2008, pp. 206–217.

[5] M. Basseur, E. K. Burke, Indicator-based multi-objective local search, in: IEEE Congress on Evolutionary Computation (CEC 2007), IEEE Press, Singapore, 2007, pp. 3100–3107.

[6] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, in: Eighth International Conference on Parallel Problem Solving from Nature (PPSN VIII), Vol. 3242 of Lecture Notes in Computer Science, Springer-Verlag, Birmingham, UK, 2004, pp. 832–842.

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

[8] C. A. Coello Coello, D. A. Van Veldhuizen, G. B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, Boston, MA, USA, 2002.

[9] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, UK, 2001.

[10] M. Ehrgott, X. Gandibleux, Approximative solution methods for multiobjective combinatorial optimization, Top 12 (1) (2004) 1–89.

[11] M. Labbé, G. Laporte, I. Rodríguez Martín, Path, tree and cycle location, in: T. Crainic, G. Laporte (Eds.), Fleet Management and Logistics, Kluwer Academic Publishers, Boston, MA, USA, 1998, pp. 187–204.

[12] M. Labbé, G. Laporte, I. Rodríguez Martín, J. J. Salazar González, The median cycle problem, working paper, CRT-99-29, Université de Montréal (1999).

[13] K. Miettinen, Nonlinear Multiobjective Optimization, Vol. 12, Kluwer Academic Publishers, Boston, MA, USA, 1999.

[14] J. A. Moreno Pérez, J. M. Moreno-Vega, I. Rodríguez Martín, Variable neighborhood tabu search and its application to the median cycle problem, European Journal of Operations Research 151 (2) (2003) 365–378.

[15] J. Renaud, F. F. Boctor, G. Laporte, Efficient heuristics for median cycle problems, Journal of the Operational Research Society 55 (2) (2004) 179–186.

[16] T. C. S. Dias, G. F. de Sousa Filho, E. M. Macambira, L. A. F. Cabral, M. H. C. Fampa, An efficient heuristic for the ring star problem, in: Fifth International Workshop on Experimental Algorithms (WEA 2006), Vol. 4007 of Lecture Notes in Computer Science, Springer-Verlag, Menorca, Spain, 2006, pp. 24–35.

[17] J. R. Current, D. A. Schilling, The median tour and maximal covering tour problems: Formulations and heuristics, European Journal of Operational Research 73 (1994) 114–126.

[18] K. Doerner, A. Focke, W. J. Gutjahr, Multicriteria tour planning for mobile healthcare facilities in a developing country, European Journal of Operational Research 179 (3) (2007) 1078–1096.

[19] J. E. Beasley, E. M. Nascimento, The vehicle routing-allocation problem: A unifying framework, Top 4 (1) (1996) 65–86.

[20] J. D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, in: First International Conference on Genetic Algorithms (ICGA 1985), Lawrence Erlbaum Associates, Pittsburgh, PA, USA, 1985, pp. 93–100.

[21] C. M. Fonseca, P. J. Fleming, Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization, in: Fifth International Conference on Genetic Algorithms (ICGA 1993), Morgan Kaufmann, Urbana-Champaign, IL, USA, 1993, pp. 416–423.

[22] J. Xu, S. Y. Chiu, F. Glover, Optimizing a ring-based private line telecommunication network using tabu search, Management Science 45 (3) (1999) 330–345.

[23] R. Baldacci, M. Dell'Amico, J. J. Salazar Gonzàlez, The capacitated m-ring star problem, Operations Research 55 (6) (2007) 1147–1162.

[24] M. Labbé, G. Laporte, Maximizing user convenience and postal service effiency in post box location, Belgian Journal of Operations Research, Statistics and Computer Science 26 (1986) 21–35.

[25] U. Akinc, K. Srikanth, Optimal routing and process scheduling for a mobile service facility, Networks 22 (2) (1992) 163–183.

[26] E. K. Burke, G. Kendall (Eds.), Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, Springer-Verlag, Berlin, Germany, 2005.

[27] F. W. Glover, G. A. Kochenberger (Eds.), Handbook of Metaheuristics, Vol. 57 of International Series in Operations Research & Management Science, Kluwer Academic Publishers, Boston, MA, USA, 2003.

[28] L. Paquete, T. Stützle, Stochastic local search algorithms for multiobjective combinatorial optimization: A review, in: T. F. Gonzalez (Ed.), Handbook of Approximation Algorithms and Metaheuristics, Vol. 13 of Computer & Information Science Series, Chapman & Hall / CRC, 2007.

[29] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, V. Grunert da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.

[30] S. Helbig, D. Pateva, On several concepts for ϵ-efficiency, OR Spektrum 16 (3) (1994) 179–186.

[31] A. G. Hernández-Díaz, L. V. Santana-Quintero, C. A. C. Coello, J. Molina, Pareto-adaptive epsilon-dominance, Evolutionary Computation 15 (4) (2007) 493–517.

[32] H. R. Lourenco, O. Martin, T. Stützle, Iterated local search, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, Vol. 57 of International Series in Operations Research & Management Science, Kluwer Academic Publishers, Norwell, MA, USA, 2002, Ch. 11, pp. 321–353.

[33] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach, IEEE Transactions on Evolutionary Computation 3 (4) (1999) 257–271.

[34] M. Laumanns, E. Zitzler, L. Thiele, On the effects of archiving, elitism, and density based selection in evolutionary multi-objective optimization, in: First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001), Vol. 1993 of Lecture Notes in Computer Science, Springer-Verlag, Zurich, Switzerland, 2001, pp. 181–196.

[35] D. W. Corne, J. D. Knowles, M. J. Oates, The pareto envelope-based selection algorithm for multiobjective optimization, in: Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI), Vol. 1917 of Lecture Notes in Computer Science, Springer-Verlag, Paris, France, 2000, pp. 839–848.

[36] C. L. Valenzuela, A simple evolutionary algorithm for multi-objective optimization (SEAMO), in: IEEE Congress on Evolutionary Computation (CEC 2002), IEEE Press, Piscataway, NJ, USA, 2002, pp. 717–722.

[37] J. Knowles, D. Corne, Bounded Pareto archiving: Theory and practice, in: X. Gandibleux, M. Sevaux, K. Sörensen, V. T'Kindt (Eds.), Metaheuristics for Multiobjective Optimisation, Vol. 535 of Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, 2004, Ch. 2, pp. 39–64.

[38] J. Bean, Genetic algorithms and random keys for sequencing and optimization, ORSA Journal on Computing 6 (2) (1994) 154–160.

[39] A. Liefooghe, L. Jourdan, E.-G. Talbi, A unified model for evolutionary multiobjective optimization and its implementation in a general purpose software framework: ParadisEO-MOEO, Research Report RR-6906, Institut National de Recherche en Informatique et Automatique (INRIA) (2009).

[40] G. Reinelt, TSPLIB – A traveling salesman problem library, ORSA Journal on Computing 3 (4) (1991) 376–384.

[41] J. Knowles, L. Thiele, E. Zitzler, A tutorial on the performance assessment of stochastic multiobjective optimizers, Tech. rep., Computer

Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, (revised version) (2006).

[42] S. Bleuler, M. Laumanns, L. Thiele, E. Zitzler, PISA — a platform and programming language independent interface for search algorithms, in: Second International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Vol. 2632 of Lecture Notes in Computer Science, Springer-Verlag, Faro, Portugal, 2003, pp. 494–508.

[43] A. Liefooghe, L. Jourdan, E.-G. Talbi, Metaheuristics and their hybridization to solve the bi-objective ring star problem: a comparative study, Research Report RR-6515, Institut National de Recherche en Informatique et Automatique (INRIA) (2008).

[44] E.-G. Talbi, A taxonomy of hybrid metaheuristics, Journal of Heuristics 8 (2) (2002) 541–564.

[45] M. Ehrgott, X. Gandibleux, Hybrid metaheuristics for multi-objective combinatorial optimization, in: C. Blum, M. J. Blesa, A. Roli, M. Sampels (Eds.), Hybrid Metaheuristics - An emergent approach for optimization, Vol. 114 of Studies in Computational Intelligence, Springer-Verlag, Berlin, Germany, 2008, Ch. 9, pp. 221–259.

[46] H. Meunier, E.-G. Talbi, P. Reininger, A multiobjective genetic algorithm for radio network optimization, in: IEEE Congress on Evolutionary Computation (CEC 2000), IEEE Press, San Diego, USA, 2000, pp. 317–324.

[47] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 204–223.

[48] M. Basseur, F. Seynhaeve, E.-G. Talbi, Design of multi-objective evolutionary algorithms: Application to the flow shop scheduling problem, in: IEEE Congress on Evolutionary Computation (CEC 2002), IEEE Press, Piscataway, NJ, USA, 2002, pp. 1151–1156.