Swarm Intell (2009) 3: 179–201
DOI 10.1007/s11721-009-0029-5

# Metaheuristics for the bi-objective orienteering problem

**Michael Schilde · Karl F. Doerner · Richard F. Hartl · Guenter Kiechle**

Received: 3 November 2008 / Accepted: 23 April 2009 / Published online: 15 May 2009

**Abstract** In this paper, heuristic solution techniques for the multi-objective orienteering problem are developed. The motivation stems from the problem of planning individual tourist routes in a city. Each point of interest in a city provides different benefits for different categories (e.g., culture, shopping). Each tourist has different preferences for the different categories when selecting and visiting the points of interests (e.g., museums, churches). Hence, a multi-objective decision situation arises. To determine all the Pareto optimal solutions, two metaheuristic search techniques are developed and applied. We use the Pareto ant colony optimization algorithm and extend the design of the variable neighborhood search method to the multi-objective case. Both methods are hybridized with path relinking procedures. The performances of the two algorithms are tested on several benchmark instances as well as on real world instances from different Austrian regions and the cities of Vienna and Padua. The computational results show that both implemented methods are well performing algorithms to solve the multi-objective orienteering problem.

**Keywords** Ant colony optimization · Variable neighborhood search · Multiobjective optimization · Orienteering problem

M. Schilde · K.F. Doerner (✉) · R.F. Hartl
Department of Business Administration, University of Vienna, Bruenner Strasse 72, 1210 Vienna, Austria
e-mail: Karl.Doerner@univie.ac.at

M. Schilde
e-mail: Michael.Schilde@univie.ac.at

R.F. Hartl
e-mail: Richard.Hartl@univie.ac.at

K.F. Doerner · G. Kiechle
Salzburg Research Forschungsgesellschaft m.b.H., Mobile and Web-based Information Systems, Jakob-Haringer-Strasse 5/3, 5020 Salzburg, Austria

G. Kiechle
e-mail: guenter.kiechle@salzburgresearch.at

## 1 Introduction

Decision support systems are in use across the different departments in many companies, for example, in the finance department to assist the manager in making investment decisions, in the marketing department for data mining, in the production department for planning the operational production decisions, and in the logistics department for planning the trips of the drivers for distributing goods.

Surprisingly, despite the great success of decision support systems in our daily business, they are not in use in assisting to plan our leisure time or our holidays. Millions of people every year visit large cities like Vienna during their holidays. They spend a couple of days or even one or two weeks in the city and during that period they visit different locations, the so-called Points Of Interest (POI). These POIs can be churches, museums, castles, or typical restaurants. In large cities or in cities with a rich cultural heritage, several hundred POIs are of potential interest for the tourist. In addition to that, a large variety of restaurants and cafes are around. Each tourist has different preferences for the different categories of POIs (e.g., culture, leisure). Hence, a multi-objective decision situation arises.

When there is a large variety of POIs, it is obvious that the tourist cannot visit all the POIs during her stay. She has to select those POIs that are most important for her and then she has to determine the sequence of their visit. For this complex decision situation, a decision support system in order to assist the tourist in planning an efficient and interesting route for each day of her visit can improve the quality of her vacation considerably.

In most of the guide books, preplanned routes with different lengths are described and suggested, depending on how much time the tourist can use for her visit. This represents a very strong simplification of the real world situation, as each tourist has different preferences. Therefore, an individual decision support system might be useful. However, psychologically it is important that the final decision is made by the tourist herself. The decision support system only makes suggestions, and the tourist can interact with the decision support system and select the route of her choice.

In this paper, solution techniques for a decision support system are developed with the aim to determine (almost) all efficient tourist routes based on certain constraints so that it is not necessary to integrate a priori preference information for the routes. The efficient solutions are presented in a structured way to the tourist so that she can navigate through the set of potential solutions to select the routes of her choice.

The problem of designing tourist routes can be considered as an Orienteering Problem (OP) (Tsiligirides 1984). In particular, we are interested in the Multi-Objective Orienteering Problem (MOOP), where different categories for each POI exist (e.g., category culture, category leisure) and each POI provides different benefits for each category. Multi-objective decision-making for combinatorial problems has attracted a significant amount of activity during the past few decades (Steuer et al. 1996). Within multi-objective combinatorial optimization, the treatment of multi-objective orienteering problems (MOOP) is a branch of high practical relevance. However, to the best of our knowledge, no research has been conducted in this area to date.

Basically, two ways of dealing with multiple objective functions exist. The first approach involves building a function that aggregates the different objectives. A major drawback of this approach is caused by the fact that it requires a priori preference information (e.g., weights or thresholds). Then the result is just a number of solutions in the objective space around the influence area of the weights. A different approach consists in including several objectives separated from each other in the model and in determining the efficient (i.e., non-dominated or Pareto-optimal) solution candidates. After this initial phase, the decision-maker is given the opportunity to explore the set of efficient solutions, guided by an interactive procedure. This exploration continues until a satisfactory solution is found. In this paper, we take the second approach.

Modifications of well-known metaheuristics for the multi-objective case have been developed and applied successfully to several problem classes (see, e.g., Ehrgott and Gandibleux 2000, 2004). In particular, several multi-objective versions of genetic algorithms (GA) (Coello Coello et al. 2007; Deb et al. 2002), evolutionary algorithms (EA) (e.g., Jozefowiez et al. 2008) and ant colony optimization (ACO) (Doerner et al. 2004) have been developed. For a recent survey on multi-objective ACO algorithms, see Garcia-Martinez et al. (2007).

For the MOOP in this paper, we develop two metaheuristic solution techniques. The first is an adaptation of the Pareto Ant Colony Optimization (P-ACO) metaheuristic developed by Doerner et al. (2007, 2004, 2006a, 2006b). The second is a multi-objective extension of Variable Neighborhood Search (VNS) (see Mladenović 1995; Mladenović and Hansen 1997). Both algorithms were combined with a path relinking procedure initially proposed by Glover and Laguna (1997) and Glover et al. (2003).

The remainder of the paper is organized as follows. In Sect. 2, the formal model of the MOOP is defined. In Sect. 3, P-ACO and Pareto VNS (P-VNS) are described. In Sect. 4, the efficiency of these two solution procedures is evaluated by solving the standard benchmark instances generated and published in Chao et al. (1996a, 1996b) and Tsiligirides (1984) and by comparing the results with those published in the literature. Moreover, the results of the different multi-objective solution procedures for the real world data are reported and the strengths and the weaknesses of the different methods are discussed. Finally, in Sect. 5, some conclusions are drawn.

## 2 Problem description

The MOOP is the multi-objective extension of the OP. The OP, introduced by Tsiligirides (1984), is a generalization of the well known Traveling Salesman Problem (TSP). The OP is also known as the selective traveling salesman problem. Each vertex of the problem provides a certain benefit. The aim of a traditional OP is to select a subset of vertices within a given complete graph such that the achieved sum of revenues collected by visiting these vertices is maximized subject to a given tour length restriction.

Feillet et al. (2005) classified these problem types as TSP with profits. The used objective functions may be the maximization of the collected total profit (OP), the minimization of the total traveling costs (Prize-Collecting TSP), or the optimization of a combination of both (Profitable Tour Problem). Archetti et al. (2007) name the extension of the TSP with profits to multiple tours as Vehicle Routing Problems (VRP) with profits. The extension of the OP to multiple tours, which is a special case of VRP with profits, was introduced under the name Team Orienteering Problem by Chao et al. (1996b). Other researchers model a similar problem that was also motivated by tourists as a single objective team orienteering problem. They propose a guided local search metaheuristic (Vansteenwegen et al. 2009) and a GRASP method that uses path relinking as a long term memory component (Souffriau et al. 2008) to solve this problem.

As the standard OP, the MOOP can be defined on the directed graph $G = (V, A)$. This graph consists of a set of vertices, $V = \{v_0, v_1, v_2, \dots, v_{n+1}\}$, and a set of arcs, $A = \{(v_i, v_j) : v_i, v_j \in V \wedge v_i \neq v_j \wedge v_i \neq v_{n+1} \wedge v_j \neq v_0\}$. A certain number of $K$ benefit values $s_{ik}$ ($k =$

$1, \ldots, K$) is assigned to each vertex $v_i \in V \setminus \{v_0, v_{n+1}\}$. The starting vertex $v_0$ and the ending vertex $v_{n+1}$ are assigned benefit values of zero for each objective. Additionally, every arc $(v_i, v_j) \in A$ is assigned a cost value $c_{ij}$ which can be interpreted as time, money spent, or distance traveled when going from vertex $v_i$ to vertex $v_j$.

Although we define the MOOP using a starting vertex $v_0$ and an ending vertex $v_{n+1}$, these two vertices typically represent the same physical location. This is true for all of the test instances used in this paper. Therefore, we denominate a solution for the MOOP as "tour" instead of using the word "path". The aim of the MOOP is to find all Pareto efficient tours that start at vertex $v_0$, end at vertex $v_{n+1}$, and do not violate the maximum tour length restriction $T_{\max}$.

Because of the multi-objective nature of our problem formulation, it is not possible to provide the decision maker with a single (near) optimal solution. However, by filtering out so-called dominated solutions, the choice can be restricted to a small number of promising solution candidates. The following definitions make this consideration precise:

A solution $x$ *dominates* a solution $x'$ if $x$ is at least equally as good as $x'$ with respect to all objective functions, and better than $x'$ with respect to at least one objective function. In formal terms: for $(f_1, \ldots, f_K)$ to be maximized, $x$ dominates $x'$, if $f_k(x) \geq f_k(x')$ for all $k = 1, \ldots, K$, and $f_k(x) > f_k(x')$ for at least one $k$. In this case, we write $x \succ x'$.

A solution $x^*$ is called *Pareto-efficient* (or: non-dominated) if there is no feasible solution that dominates $x^*$. In addition, if $x^*$ is Pareto-efficient, then $z^* = f(x^*) = (f_1(x^*), \ldots, f_K(x^*))$ is called a *non-dominated vector*. The set of all non-dominated vectors is referred to as the *Pareto front* (or: non-dominated frontier). We extend the relation $\succ$ from the solution space to the objective space by defining, for two vectors $z = (z_1, \ldots, z_K)$ and $z' = (z'_1, \ldots, z'_K)$, that $z \succ z'$ holds iff $z_k \geq z'_k$ for all $k = 1, \ldots, K$ and $z_k > z'_k$ for at least one $k$.

Formally the problem can be described as follows:

$$(f_k) \rightarrow \max \quad (k = 1, \ldots, K) \tag{1}$$

$$f_k = \sum_{v_i \in V \setminus \{v_0, v_{n+1}\}} (s_{ik} \cdot y_i) \quad (k = 1, \ldots, K) \tag{2}$$

such that

$$\sum_{v_j \in V \setminus \{v_i\}} x_{ij} = y_i \quad \big(v_i \in V \setminus \{v_{n+1}\}\big), \tag{3}$$

$$\sum_{v_i \in V \setminus \{v_j\}} x_{ij} = y_j \quad \big(v_j \in V \setminus \{v_0\}\big), \tag{4}$$

$$\sum_{\{v_i, v_j\} \in S} x_{ij} \leq |S| - 1 \quad (S \subseteq V \wedge S \neq \emptyset), \tag{5}$$

$$y_0 = y_{n+1} = 1, \tag{6}$$

$$\sum_{(v_i, v_j) \in A} c_{ij} \cdot x_{ij} \leq T_{\max}, \tag{7}$$

$$x_{ij} \in \{0, 1\} \quad \big((v_i, v_j) \in A\big), \tag{8}$$

$$y_i \in \{0, 1\} \quad (v_i \in V). \tag{9}$$

The first type of binary variables, $x_{ij}$, is assigned a value of $x_{ij} = 1$ if the arc $(v_i, v_j) \in A$ is used in the tour, and $x_{ij} = 0$ otherwise. The second type of binary variables, $y_i$, is assigned a value of $y_i = 1$ if the vertex $v_i$ is visited by the current tour, and $y_i = 0$ otherwise. Equation (2) determines the $K$ objective functions as the sum of the corresponding benefit values over all visited vertices of the tour for each objective. Constraints (3) and (4) ensure that for each visited POI exactly one ingoing and one outgoing arc is used. Constraint (5) eliminates subtours, whereas Constraint (6) ensures that the start and end point are included in every tour. Constraint (7) ensures that the cost limit $T_{\max}$ is not exceeded by the tour.

## 3 Solution procedures

As the OP is a generalization of the TSP which is known to be NP-hard, no efficient solution techniques for the problem at hand exist, especially when considering multiple objectives. Hence, we apply two different metaheuristic solution techniques to the MOOP. In the next two sections, the implementation details and the design decisions for P-ACO and P-VNS are described. P-ACO was already introduced by Doerner et al. (2004, 2006a, 2006b, 2007) and is here adapted to the specific requirements of the MOOP. Differently, the P-VNS solution approach for the MOOP is new and is based on the ideas of the single-objective VNS developed by Mladenović (1995) and Mladenović and Hansen (1997). Both techniques are combined with a Path Relinking (PR) procedure (Glover and Laguna 1997; Glover et al. 2003; Pasia et al. 2006; 2007).

### 3.1 Pareto Ant Colony Optimization

P-ACO was originally developed and applied to a multi-objective portfolio selection problem by Doerner et al. (2004) as a Pareto extension to the general ACO framework (Dorigo and Di Caro 1999; Dorigo and Stuetzle 2004). The P-ACO algorithm was then applied to the multi-objective activity crashing problem (Doerner et al. 2006a) and to the multi-criteria tour planning for mobile healthcare facilities by Doerner et al. (2007). Furthermore, this approach was applied to the bi-objective permutation flow-shop problem in combination with path relinking procedures by Pasia et al. (2006, 2007).

The steps of P-ACO are shown in Fig. 1. The initialization phase of the implemented P-ACO algorithm starts with creating a population of artificial ants which initially are positioned at the starting vertex $v_0$. Each ant is assigned an initial tour $x = \{v_0\}$ containing only the starting vertex and a deterministically generated set of objective weights

*Repeat* the following steps until the stopping condition is met:

1. *Create an ant population* positioned at the starting vertex $v_0$.
2. *Assign* a weight vector $p$ and an initial tour $x = \{v_0\}$ to each ant.
3. *Repeat* the following steps until each ant has built a feasible tour:
   (a) *Select next vertex* $j$ to visit for each ant's (partial) tour $x$.
   (b) *Perform local pheromone update* on used arcs $(v_i, v_j) \in A$.
4. *Apply iterative improvement* to each ant's tour $x$.
5. *Perform global pheromone update.*

**Fig. 1** Steps of the P-ACO

$p = \{p_1, \dots, p_K\}$. In the current paper, we present algorithms which are applied to problems with $K = 2$ objective values. Therefore, the weights are generated in a way such that for all ants within a population the weights are equally spread between the two extreme weight values ($p_1 = 0$, $p_2 = 1$ and $p_1 = 1$, $p_2 = 0$). Additionally, each arc $(v_i, v_j) \in A$ is assigned an initial pheromone value $\tau_{ijk} = \tau_0$ for each objective value $k$.

During the construction phase of the algorithm each ant constructs a feasible tour from the starting vertex to the ending vertex following the ant colony system scheme proposed by Dorigo and Gambardella (1997). For the calculation of the probability to visit the next customer, two ingredients are used, namely the heuristic information $\eta_{ijk}$ and the pheromone information $\tau_{ijk}$. After each construction step (i.e., the inclusion of an additional POI in the current partial solution), a local pheromone update is performed on the arc $(v_i, v_j)$ chosen for each of the objectives $k$. At the end of the construction phase, after each ant has built a complete tour, an iterative improvement is applied. For all the generated solutions of a population, the efficiency is checked and all the efficient solutions are stored in an external memory. Finally, a global pheromone update procedure is performed, taking into account the best and the second-best solution found for each objective value $k$ during the current iteration. After each iteration, a new colony of ants is used.

### *Heuristic information*

The heuristic information is a priority measure that serves as an indicator for the attractiveness of visiting a candidate vertex $v_j$ from the ant's current position vertex $v_i$. A heuristic priority number is calculated for each vertex that is not already included in the partial tour and that does not violate the maximum tour length constraint (including the way to the ending point). This measure is calculated for each objective value $k$ using

$$\eta_{ijk} = \frac{s_{jk}}{c_{ij}}, \tag{10}$$

and represents the ratio between the benefit $s_{jk}$ of adding vertex $v_j$ and the corresponding cost $c_{ij}$.

### *Pheromone information*

Artificial ants use artificial pheromone trails to store information about the quality of solutions they built in the past. The pheromone $\tau_{ijk}$ is stored in a matrix $\tau$ for each objective value $k$ and for each arc $(v_i, v_j) \in A$ of the problem's graph.

### *Decision rule*

Let $v_i$ be the last vertex visited by a (partial) tour $x$. Given the heuristic information, the pheromone value and the set of candidate vertices $\Omega(x) = \{v_j : v_j \notin x, (c_{ij} + c_{jn+1}) \le T_{\max}\}$, the following pseudo-random decision rule (11) is used to select the vertex $v_j$ to add to $x$:

$$v_j = \begin{cases} \operatorname{argmax}_{v_l \in \Omega(x)} \left\{ \sum_{k=1}^{K} \left( \tau_{ilk}^{\alpha} \cdot \eta_{ilk}^{\beta} \cdot p_k \right) \right\} & \text{if } q \le q_0, \\ \hat{v_j} & \text{otherwise.} \end{cases} \tag{11}$$

The variable $q$ represents a random number uniformly distributed in [0, 1], whereas $q_0$ is a parameter specified by the user ($0 \le q_0 \le 1$). This parameter specifies the probability that the vertex with the highest aggregated rating is chosen. Vertex $\hat{v}_j$ is chosen by using a roulette wheel selection according to the probabilistic distribution described in (12). The two user defined parameters $\alpha$ and $\beta$ determine the relative influence of the pheromone values and the heuristic information, respectively. The factor $p_k$ represents the objective weight as described before.

$$\mathcal{P}(v_j) = \begin{cases} \dfrac{\sum_{k=1}^{K}(\tau_{ijk}^{\alpha} \cdot \eta_{ijk}^{\beta} \cdot p_k)}{\sum_{v_h \in \Omega(x)}[\sum_{k=1}^{K}(\tau_{ijk}^{\alpha} \cdot \eta_{ijk}^{\beta} \cdot p_k)]} & \text{if } v_j \in \Omega(x), \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

*Pheromone update*

Each time an ant adds a vertex $v_j$ to the (partial) tour $x = \{v_0, \dots, v_i\}$, a local pheromone update, which decreases the amount of pheromone on the visited arc, is performed on the arc $(v_i, v_j) \in A$. This local update mechanism leads to a stronger diversification of the solutions within one ant population. Hence, unexplored vertex combinations with a reasonably high pheromone value become more attractive. The following equation (13) shows how the current amount of pheromone is reduced for each objective value $k$. The parameter $\rho$ defines the evaporation rate (with $0 \le \rho \le 1$):

$$\tau_{ijk} \leftarrow (1-\rho) \cdot \tau_{ijk} + \rho \cdot \tau_0. \tag{13}$$

The global pheromone update is performed after each of the ants has completed its tour: The best and the second best solution for each objective value within the current iteration are used to update the pheromone matrices. The elements $\tau_{ijk}$, which correspond to arcs traversed in the iteration best and second best solution, are updated according to (14):

$$\tau_{ijk} \leftarrow \tau_{ijk} + \Delta\tau_{ijk}. \tag{14}$$

This update mechanism is similar to the update mechanism in Doerner et al. (2004, 2006b). After performing tests with different values for the factor $\Delta\tau_{ijk}$, it was fixed to a value of $\Delta\tau_{ijk} = \tau_0$ for the best solution and $\Delta\tau_{ijk} = \frac{\tau_0}{2}$ for the second best solution, whereby $\tau_0$ is the initial amount of pheromone assigned to each arc. Note that the local pheromone update is performed before the iterative improvement and the global pheromone update after the iterative improvement, explained in the next section.

*Iterative improvement*

Iterative improvement consists of three basic operators: shortening, vertex insert and vertex exchange. The shortening operator tries to rearrange the vertices within a tour in order to minimize the cost of the tour. As shortening operator a 2-opt is applied (Croes 1958). For runtime reasons, the maximum sequence length of a route segment considered for inversion is set to 30. For an overview of the different local search operators for routing problems, see Kindervater and Savelsbergh (1997). The vertex insert operator checks the possibility to insert additional POIs after the shortening operator has been applied. Every possible insertion position for each non-included POI which lies within the maximum route length is checked. The first improvement strategy is applied: After an improvement of the solution quality with respect to the weighted objective function of the current ant, the 2-opt procedure is reapplied. The vertex exchange operator tries to remove every existing POI and to exchange it with a non-visited POI. Again, when the aggregated objective function value is improved, the iterative improvement procedure starts applying 2-opt.

```
Initialization:    Construct an initial solution $x$;
Repeat the following steps until the stopping condition is met:

1. Set $\kappa \leftarrow 1$.
2. Repeat the following steps until $\kappa = \kappa_{\max}$:
   (a) Generate Random Weights $p_k$.
   (b) Shaking. Generate a point $x'$ at random from $\kappa$th neighborhood of $x$ ($x' \in N_\kappa(x)$).
   (c) Iterative Improvement. Apply iterative improvement with $x'$ as initial solution; the
       obtained local optimum is denoted with $x''$; the evaluation is done by using the
       current weight vector.
   (d) Store $x''$ in an external memory if it is efficient.
   (e) Move or not. If this local optimum $x''$ is better than the incumbent by using the
       current weight vector, or if some acceptance criterion is met, move ($x \leftarrow x''$), and
       continue the search with $N_1(x)$ ($\kappa \leftarrow 1$); otherwise, set $\kappa \leftarrow \kappa + 1$.
```

**Fig. 2** Steps of the P-VNS

## 3.2 Pareto Variable Neighborhood Search

In this section, the P-VNS method is described. P-VNS is an extension of the well known multiple-neighborhood local search VNS developed by Mladenović (1995) and Mladenović and Hansen (1997) to tackle multi-objective problems. The basic idea behind the traditional VNS solution method is to make use of a simple iterative improvement method to explore the local neighborhood of an initial solution in combination with a set of different neighborhood operators to avoid getting stuck in local optima. The main important modifications of the standard VNS are the use of randomly generated weight vectors and a storage for the efficient solutions.

The steps of the P-VNS are shown in Fig. 2. $N_\kappa$ ($\kappa = 1, \dots, \kappa_{\max}$) is the set of neighborhoods. The general stopping condition can be a limit on CPU time, a limit on the number of iterations, or a limit on the number of iterations between two improvements.

The different components of the P-VNS are described in detail in the following.

### *Initialization and initial solution*

The initial solution for P-VNS is generated by means of a greedy algorithm that takes into account all vertices $v_i$ that are located within the cost limit $T_{\max}$. These points are sorted in descending order regarding the sum of their objective values. Afterwards, the algorithm starts with a tour only including the starting and ending point and successively inserts the points from this list at the first position in which they can feasibly be inserted.

### *Random weights*

In the P-VNS algorithm, a random weight vector $p = \{p_1, \dots, p_k\}$ is used for evaluating the solutions within the shaking phase and the iterative improvement phase. The weight vector initially is set to one extreme point. Afterwards, the weight vector is changed after each shaking step. The values of the weight vector are adjusted by adding or subtracting a value $\Delta p_k$ that is randomly generated in the interval [0, 0.01]. After the modification, the weights are normalized. On the one hand, this procedure guarantees that the search space is systematically explored. On the other hand, smooth changes in the weight vector usually do not change the solution completely. Therefore, the good solution found with a similar previous weight vector is a good starting point for finding a new solution for a slightly modified weight vector. Moreover, it is avoided to use a fixed discretization.

*Shaking*

The selection of neighborhood structures $N_\kappa$ used for shaking is an important design decision for P-VNS. Each neighborhood operator should provide a proper balance between perturbing the incumbent solution and retaining the good parts of the incumbent solution.

The used neighborhood operator is based on a two point exchange move. A move in the $\kappa$th neighborhood consists of removing $\kappa$ consecutive vertices from the tour, starting at a randomly selected position. Afterwards, a sorted list of all vertices not yet included in the current tour is built. The vertices are sorted in descending order with respect to the objective value increase using the current weights. Out of the first three entries with the highest ranking in this list, one randomly selected vertex is reinserted into the current tour at the same position as the removed vertices. This way, $l$ new vertices are inserted into the tour. The largest neighborhood is a complete exchange of all vertices on the tour.

The shaking procedure does not guarantee that the new tour does not exceed the cost limit $T_{\max}$. Therefore, in a repair step, a sorted list of all vertices in the tour is created. The vertices are sorted in descending order with respect to costs saved when removing the vertex from the tour. Vertices are removed as long as the cost limit is violated.

The same iterative improvement operators as for P-ACO were used (see Sect. 3.1).

*Move or not*

The acceptance decision to move to another incumbent solution is based on the dominance criterion. If the candidate tour is not dominated by any of the previously found tours, it is stored and accepted as new incumbent tour, otherwise it is rejected. Only every $\phi$ iterations a deterioration in solution quality can be accepted, if none of the objective values of the candidate tour is below a given percentage $\psi$ of the current incumbent's corresponding objective value.

## 3.3 Path Relinking procedure

PR has first been introduced by Glover and Laguna (1997) as an approach for integrating intensification and diversification strategies in the context of tabu search. The main idea behind PR is to search for additional solutions by exploring trajectories that connect solutions that are known to be of high quality. Therefore, starting at one solution, the so-called initial solution, a path is created through its neighborhood space, that leads to the other solution, the so called guiding solution. Especially in multi-objective optimization, this is a promising concept to find additional efficient solutions by relinking two proposed efficient solutions (e.g., Pasia et al. 2006; 2007). In our approach, PR is applied as a post-processing stage after the solution procedures in order to improve the results produced by P-ACO or P-VNS, respectively.

PR is applied to all pairs of solutions. The algorithm starts from the current solution. Vertices not visited in the guiding solution are iteratively removed from the current solution and replaced by vertices included in the guiding solution. The algorithm stops when the current solution equals the guiding solution. To increase performance, the procedure does not always enumerate all possible combinations of removal and insertion steps between the initial and the guiding solution. If there are more than 16 possible removal/insertion combinations, one of them is selected and performed on a random basis and the procedure restarts. Otherwise, all possible combinations are tried. All intermediate efficient solutions are stored.

## 4 Computational results

Both solution approaches are implemented in C++ using the GNU compiler g++ in its version 4.1.0 on a 64-bit Linux platform. All test runs are performed on an Intel Pentium 4 D CPU with 3.2 GHz and 4 GB of RAM.

### 4.1 Test instances

In a first analysis, the algorithms are applied to the standard benchmark instances generated for single objective optimization. Tsiligirides (1984) generated 49 test instances and Chao et al. (1996a) published 40 problem instances.

For the performance evaluation of the developed bi-objective optimization algorithms these test instances are augmented with additional benefit values. Moreover 127 new bi-objective problem instances based on real world data of different Austrian regions, the city of Vienna and the town of Padua were generated.

The problem sets published by Tsiligirides consist of three problem classes, referred to as 1_p21, 1_p32, and 1_p33. They contain 21, 32, and 33 vertices with 11, 18, and 20 instances, respectively. The problem sets published by Chao, Golden, and Wasil (CGW) consist of two problem classes, referred to as 1_p64 and 1_p66, containing 64 and 66 vertices with 14 and 26 instances, respectively. The instances within the problem classes differ only in the value for the cost limit $T_{\max}$ (interpreted as distance limit in our case). The bi-objective problem sets denoted as 2_p21, 2_p32 and 2_p33 are derived from the single objective problem set 1_p21, 1_p32 and 1_p33. In this case, a second benefit value for each vertex is randomly generated. The instances generated by CGW were also augmented with a second benefit value and denoted as 2_p64 and 2_p66. All the instances can be downloaded following the link http://prolog.univie.ac.at/research/OP/.

The real world problem sets consist of POIs in the cities Padua in Italy and Vienna in Austria, and POIs in the Austrian regions Styria, Carinthia and Lower Austria. The two small problem sets 2_p97 and 2_p273 contain 97 POIs in Padua and 273 POIs in Vienna, respectively. The two medium sized problem sets 2_p559 and 2_p562 contain 559 and 562 tourist attractions in Styria and Carinthia, respectively. The large problem set 2_p2143 contains 2143 POIs within Lower Austria. The benefit values have been randomly generated. Different $T_{\max}$ values are used for the creation of different real-world based instances. The two small problem sets 2_p97 and 2_p273 have been split into 20 test instances with maximum distance restrictions between 1 and 20 kilometers. For each of the medium problems (2_p559 and 2_p562) and the large problem (2_p2143) 29 test instances with maximum distance restrictions between 10 and 150 kilometers have been created. The distance matrix is based on the real road network provided by Teleatlas, except for the instances of Padua. Here Euclidean distances were used. In the real world problem sets service time is added for each vertex. This service time can be interpreted as time spent for visiting a POI. The service time is 0.5 km for all problem sets.

## 4.2 Evaluation metrics

All metrics used in this paper have been calculated using normalized objective function values. For the unary metrics a reference set is required. As the set of all Pareto optimal solutions cannot be computed, we have to generate an approximation set as a reference set. All Pareto efficient solutions found by any of the two solution methods in any of the ten runs have been stored in the reference set $R$. For the comparison of the solution quality different attainment ratios were used. We report the metric values when considering the 20%, 50%, and 80% attainment surface of the ten runs (see Grunert da Fonseca et al. 2001; Knowles et al. 2006).

*Hypervolume indicator*

The hypervolume indicator as described by Zitzler and Thiele (1999) measures the hypervolume of the objective space of a set $A$ that is weakly dominated by an approximation set or the set containing the non-dominated frontier of an approximation method. The calculation of the indicator value is done using:

$$I_H(A) = s_1(z^1) \cdot s_2(z^1) + \sum_{z^i, z^j \in A: i+1=j, i \geq 1} \{[s_2(z^j) - s_2(z^i)] \cdot s_1(z^j)\}, \tag{15}$$

where $s_k(z^i)$ represents the objective function value for objective $k$ achieved by solution $z^i$. All objective values are normalized to the range $[1.0, 2.0]$, with 2.0 being the best result and 1.0 the worst. The point $(0, 0)$ was used as a nadir point. Therefore, the values of the hypervolume indicator reside in the interval $[1, 4]$, whereby values closer to 4 represent better solutions. A larger value for the hypervolume indicator represents a larger area covered by the nadir point and the set of found solutions.

*Unary epsilon indicator*

The unary epsilon indicator was described by Zitzler et al. (2003). Here, we use the multiplicative unary version. This indicator calculates the minimum factor $\varepsilon$: when every point in the reference set $R$ is multiplied by $\varepsilon$, then the resulting approximation set is weakly dominated by the approximation set $A$. The $\varepsilon$ indicator is calculated using:

$$I_\varepsilon(A, R) = \inf_{\varepsilon \in \mathbb{R}} \{\forall z^2 \in R \exists z^1 \succeq_\varepsilon z^2\}. \tag{16}$$

For a maximization problem, a solution $z^1$ $\varepsilon$-dominates a solution $z^2$, written as $z^1 \succeq_\varepsilon z^2$, if and only if $z_i^1 \cdot \varepsilon \geq z_i^2$ is fulfilled for all objective values $i$. All objective values are normalized to the range $[1.0, 2.0]$ with 1.0 being the best result and 2.0 the worst. When the unary epsilon indicator has the value 1.0 then all the solutions of the reference set were found.

*R3 indicator*

The R3 indicator was described by Hansen and Jaszkiewicz (1998) and is used to assess and compare the approximation sets found by our two solution methods, based on a set of utility functions. Therefore, a set $\Lambda$ of 500 uniformly dispersed weight vectors $\vec{\lambda} = \{\lambda_1, \ldots, \lambda_n\} \in \Lambda$ is generated and used for calculating the utilities for the corresponding approximation set using an augmented Tchebycheff function (17). All objective values are again normalized to the range [1.0, 2.0], with 1.0 being the best result and 2.0 the worst.

Knowing that

$$u_\lambda(z) = -\left(\max_{j \in 1 \dots n} \lambda_j \cdot \left|z_j^* - z_j\right| + \rho \cdot \sum_{j \in 1 \dots n} \left|z_j^* - z_j\right|\right) \tag{17}$$

is the utility reached by a solution $z$ and

$$u^*\left(\vec{\lambda}, A\right) = \max_{z \in A} u_\lambda(z) \tag{18}$$

is the maximum utility reached with the weight vector $\vec{\lambda}$ on an approximation set A, the R3 indicator can be calculated as

$$I_{R3}(A, R) = \frac{\sum_{\vec{\lambda} \in \Lambda} \frac{u^*(\vec{\lambda}, R) - u^*(\vec{\lambda}, A)}{u^*(\vec{\lambda}, R)}}{|\Lambda|}. \tag{19}$$

The factor $\rho$ is fixed to a value of 0.01, and the value $z^*$ is fixed to a value of 1.0 (a point that weakly dominates all points in the approximation set). If the indicator value is close to zero, the approximation set provides utility values for the generated weight vectors that are close to the utility values provided by the reference set.

*Pareto frontier approximation indicator*

This indicator value serves as a measure of how well an approximation set $A$ approximates the Pareto frontier. The indicator has been proposed by Czyzak and Jaszkiewicz (1996). The indicator value is calculated using:

$$I_A(A, R) = \frac{\sum_{r \in R} [\min_{z \in A} D(z, r)]}{|R|}, \tag{20}$$

where $D(z, r)$ is a distance measure in the objective space for which Euclidean distance was used. For this indicator, the smaller the value, the closer the solutions of the approximation set are to the reference set.

*Spacing indicator*

This indicator measures how uniformly the points in an approximation set $A$ are distributed in objective space. It was proposed by Schott (1995) and is calculated using:

$$I_S(A) = \sqrt{\frac{1}{|A| - 1} \cdot \sum_{z \in A} \left[\bar{D} - D(z)\right]}. \tag{21}$$

The value $D(z)$ can be calculated using

$$D(z) = \min_{z' \in A} \sum_{k=1}^{K} \left|z_k - z'_k\right|, \tag{22}$$

and $\bar{D}$ is the mean of all values $D(z)$. For this indicator, smaller values correspond to better solutions. The smaller the value, the better the distribution of the solution of the Pareto frontier. When there is a large value, it means that clusters of solutions exist and in several regions almost no solutions were found.

*Range covering indicator*

This simple indicator measures how well the whole possible range of the Pareto frontier is covered by the points in an approximation set $A$. It is calculated using

$$I_R(A) = \frac{1}{K} \cdot \sum_{k=1}^{K} \left[R_k^{\max}(A) - R_k^{\min}(A)\right]. \tag{23}$$

$R_k^{\max}(A)$ and $R_k^{\min}(A)$ are calculated using

$$R_k^{\max}(A) = \max\left\{s_k(z) | z \in A\right\} \tag{24}$$

and

$$R_k^{\min}(A) = \min\left\{s_k(z) | z \in A\right\}, \tag{25}$$

respectively, and $s_k(z)$ represents the score for objective value $k$ achieved by solution $z$. For this indicator, larger values mean that the set of found solution reaches points in the solution space that are close to the two extreme solutions (in the bi-objective case).

### 4.3 Parameter setting

After performing preliminary tests, the parameters for the P-ACO algorithm have been fixed to a colony size of 100 ants, 100 iterations, an initial pheromone value of $\tau_0 = 10$, an evaporation rate of $\rho = 0.01$, a weight for the pheromone information of $\alpha = 1$, and a weight for the heuristic information of $\beta = 2$. The P-VNS algorithm was stopped after using the same runtime as the P-ACO algorithm.

The parameters used for P-VNS are the threshold parameters for the acceptance of a deteriorating solution. Every $\phi = 1000$ shaking steps a deteriorating solution is accepted. The maximum deterioration is $\psi = 5\%$ of the current incumbent solution.

### 4.4 Numerical tests

*Single objective results*

The presented results are always average values over 10 independent runs of the solution method with one parameter setting. Computational results show that both P-ACO and P-VNS provide competitive results for the single objective problem sets presented in the literature. For the three test instances presented by Tsiligirides (1984) (1_p21, 1_ p32, and 1_p33), both algorithms find results that are equal to the optimal results. For the newer test instances 1_p64 and 1_p66, published by Chao et al. (1996a), the two methods find new best results for 7 out of 40 instances (see Tables 1 and 2) with an average improvement in the objective value of 0.14% for 1_p64 and 0.26% for 1_p66. We compare our results to the ones published by Chao et al. (1996a) (CGW being their proposed heuristic and TA being their reimplementation of Tsiligirides' heuristic) and more recent results published by Vansteenwegen et al. (2009) (named GLS in the tables).

**Table 1** Results for problem set 1_p64 ($T_{\max}$: allowed tour length, CGW: results reported by CGW, $Z_{\max}$: best solution found, $Z_{\text{avg}}$: average solution found, $Z_{\min}$: worst solution found, $S_{\text{inc}}$(ACO): run time when the best solution of the current ACO run was found, $S_{\text{inc}}$(VNS): run time when the best solution of the current VNS run was found, gap(ACO) and gap(VNS): gap to the best solution found by CGW)

| $T_{\max}$ | CGW | GLS | P-ACO | | | P-VNS | | | $S_{\text{inc}}$(ACO) | $S_{\text{inc}}$(VNS) | gap(ACO) | gap(VNS) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | | | | |
| 15 | 96 | 96 | 96 | 96.0 | 96 | 96 | 96.0 | 96 | 0.007 | 0.000 | 0.00% | 0.00% |
| 20 | 294 | 294 | 294 | 294.0 | 294 | 294 | 294.0 | 294 | 0.017 | 0.002 | 0.00% | 0.00% |
| 25 | 390 | 390 | 390 | 390.0 | 390 | 390 | 390.0 | 390 | 0.025 | 0.022 | 0.00% | 0.00% |
| 30 | 474 | 474 | 474 | 474.0 | 474 | 474 | 472.8 | 468 | 0.034 | 1.217 | 0.00% | 0.00% |
| 35 | 570 | 552 | 576 | 575.4 | 570 | 576 | 576.0 | 576 | 0.508 | 0.148 | +1.05% | +1.05% |
| 40 | 714 | 702 | 714 | 714.0 | 714 | 714 | 714.0 | 714 | 0.409 | 2.453 | 0.00% | 0.00% |
| 45 | 816 | 780 | 816 | 810.0 | 804 | 816 | 816.0 | 816 | 7.013 | 0.587 | 0.00% | 0.00% |
| 50 | 900 | 888 | 900 | 899.4 | 894 | 900 | 899.4 | 894 | 4.492 | 3.872 | 0.00% | 0.00% |
| 55 | 984 | 972 | 984 | 978.6 | 978 | 984 | 974.4 | 966 | 8.323 | 3.011 | 0.00% | 0.00% |
| 60 | 1044 | 1062 | 1062 | 1060.2 | 1056 | 1062 | 1054.2 | 1050 | 0.991 | 1.606 | 0.00% | 0.00% |
| 65 | 1116 | 1110 | 1116 | 1116.0 | 1116 | 1116 | 1116.0 | 1116 | 0.711 | 8.268 | 0.00% | 0.00% |
| 70 | 1176 | 1188 | 1188 | 1188.0 | 1188 | 1188 | 1186.8 | 1182 | 2.281 | 0.975 | 0.00% | 0.00% |
| 75 | 1224 | 1236 | 1236 | 1236.0 | 1236 | 1236 | 1235.4 | 1230 | 0.721 | 1.158 | 0.00% | 0.00% |
| 80 | 1272 | 1260 | 1284 | 1281.0 | 1278 | 1284 | 1281.6 | 1278 | 2.109 | 12.593 | +0.94% | +0.94% |
| Avg. | – | – | – | – | – | – | – | – | – | – | +0.14% | +0.14% |

**Table 2** Results for problem set 1_p66 ($T_{\max}$: allowed tour length, CGW: results reported by CGW, $Z_{\max}$: best solution found, $Z_{\text{avg}}$: average solution found, $Z_{\min}$: worst solution found, $S_{\text{inc}}$(ACO): run time when the best solution of the current ACO run was found, $S_{\text{inc}}$(VNS): run time when the best solution of the current VNS run was found, gap(ACO) and gap(VNS): gap to the best solution found by CGW)

| $T_{\max}$ | CGW | TA | GLS | P-ACO | | | P-VNS | | | $S_{\text{inc}}$(ACO) | $S_{\text{inc}}$(VNS) | gap(ACO) | gap(VNS) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | | | | |
| 5 | 10 | 10 | 10 | 10 | 10.0 | 10 | 10 | 10.0 | 10 | 0.002 | 0.000 | 0.00% | 0.00% |
| 10 | 40 | 40 | 40 | 40 | 40.0 | 40 | 40 | 40.0 | 40 | 0.004 | 0.000 | 0.00% | 0.00% |
| 15 | 120 | 100 | 120 | 120 | 120.0 | 120 | 120 | 120.0 | 120 | 0.009 | 0.000 | 0.00% | 0.00% |
| 20 | 195 | 190 | 175 | 205 | 205.0 | 205 | 205 | 205.0 | 205 | 0.061 | 0.007 | +5.13% | +5.13% |
| 25 | 290 | 290 | 290 | 290 | 290.0 | 290 | 290 | 290.0 | 290 | 0.022 | 0.008 | 0.00% | 0.00% |
| 30 | 400 | 400 | 400 | 400 | 400.0 | 400 | 400 | 400.0 | 400 | 0.027 | 0.007 | 0.00% | 0.00% |
| 35 | 460 | 460 | 465 | 465 | 465.0 | 465 | 465 | 465.0 | 465 | 0.033 | 0.129 | 0.00% | 0.00% |
| 40 | 575 | 575 | 575 | 575 | 575.0 | 575 | 575 | 575.0 | 575 | 0.039 | 1.936 | 0.00% | 0.00% |
| 45 | 650 | 645 | 640 | 650 | 650.0 | 650 | 650 | 650.0 | 650 | 0.046 | 0.032 | 0.00% | 0.00% |
| 50 | 730 | 730 | 710 | 730 | 730.0 | 730 | 730 | 730.0 | 730 | 0.053 | 0.131 | 0.00% | 0.00% |
| 55 | 825 | 820 | 825 | 825 | 825.0 | 825 | 825 | 825.0 | 825 | 0.061 | 3.040 | 0.00% | 0.00% |
| 60 | 915 | 915 | 905 | 915 | 915.0 | 915 | 915 | 915.0 | 915 | 0.135 | 0.037 | 0.00% | 0.00% |
| 65 | 980 | 980 | 935 | 980 | 980.0 | 980 | 980 | 980.0 | 980 | 0.079 | 1.635 | 0.00% | 0.00% |
| 70 | 1070 | 1070 | 1070 | 1070 | 1070.0 | 1070 | 1070 | 1070.0 | 1070 | 0.084 | 0.405 | 0.00% | 0.00% |
| 75 | 1140 | 1140 | 1140 | 1140 | 1140.0 | 1140 | 1140 | 1139.5 | 1135 | 0.092 | 0.060 | 0.00% | 0.00% |
| 80 | 1215 | 1215 | 1195 | 1215 | 1215.0 | 1215 | 1215 | 1206.0 | 1195 | 1.273 | 2.612 | 0.00% | 0.00% |
| 85 | 1270 | 1265 | 1265 | 1270 | 1270.0 | 1270 | 1270 | 1269.5 | 1265 | 0.223 | 0.971 | 0.00% | 0.00% |
| 90 | 1340 | 1340 | 1300 | 1340 | 1340.0 | 1340 | 1340 | 1336.0 | 1330 | 0.481 | 0.609 | 0.00% | 0.00% |
| 95 | 1380 | 1390 | 1385 | 1395 | 1395.0 | 1395 | 1395 | 1394.0 | 1390 | 0.389 | 0.842 | +0.36% | +0.36% |
| 100 | 1435 | 1455 | 1445 | 1465 | 1465.0 | 1465 | 1465 | 1451.0 | 1445 | 1.408 | 11.169 | +0.69% | +0.69% |

**Table 2** (*Continued*)

| $T_{\max}$ | CGW | TA | GLS | P-ACO | | | P-VNS | | | $S_{\text{inc}}$(ACO) | $S_{\text{inc}}$(VNS) | gap(ACO) | gap(VNS) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | $Z_{\max}$ | $Z_{\text{avg}}$ | $Z_{\min}$ | | | | |
| 105 | 1510 | 1515 | 1505 | 1520 | 1520.0 | 1520 | 1520 | 1508.5 | 1495 | 0.148 | 0.328 | +0.33% | +0.33% |
| 110 | 1550 | 1550 | 1560 | 1560 | 1560.0 | 1560 | 1560 | 1555.0 | 1550 | 0.319 | 1.332 | 0.00% | 0.00% |
| 115 | 1595 | 1590 | 1580 | 1595 | 1595.0 | 1595 | 1595 | 1594.0 | 1585 | 0.163 | 8.458 | 0.00% | 0.00% |
| 120 | 1635 | 1635 | 1635 | 1635 | 1635.0 | 1635 | 1635 | 1633.0 | 1625 | 1.222 | 1.180 | 0.00% | 0.00% |
| 125 | 1655 | 1655 | 1665 | 1670 | 1670.0 | 1670 | 1670 | 1669.5 | 1665 | 0.186 | 1.066 | +0.30% | +0.30% |
| 130 | 1680 | 1670 | 1680 | 1680 | 1680.0 | 1680 | 1680 | 1680.0 | 1680 | 0.188 | 0.020 | 0.00% | 0.00% |
| Avg. | – | – | – | – | – | – | – | – | – | – | – | +0.26% | +0.26% |

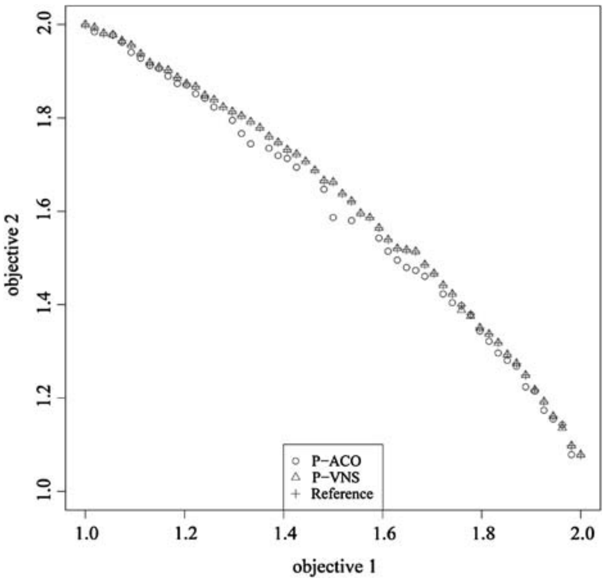

**Fig. 3** Pareto front approximation (20% attainment) for problem set 2_p64 with $T_{\max} = 70$ km

*Bi-objective and real world results*

In the following section, the results of the P-ACO and P-VNS algorithms (including PR) are reported and the performances of the algorithms are compared. Although the required runtime for the PR procedure is negligible, the PR procedure slightly improves the P-ACO as well as the P-VNS solution quality. All computational results are obtained with the PR procedure. Regarding the number of solutions found by our two algorithms, computational results show that the implemented P-VNS algorithm on average produces 451.84% more solutions within the same amount of time. Figure 3 shows a graphical representation of the normalized reference set and all the solutions in the 20% attainment surface for problem set 2_p64 with a total distance limit of $T_{\max} = 70$ km. In this example, it can clearly be seen that the P-VNS algorithm provides more solutions which are closer to the reference set.

Tables 3, 4 and 5 show the averages of the indicator values calculated for each problem set. The different instances in each problem set consist of the same geographical information of the POI, but represent different maximal tour lengths $T_{\max}$. Each column corresponds to one of the six indicators described in Sect. 4.2: $I_H$ corresponds to the hypervolume indicator, $I_\epsilon$ to the unary epsilon indicator, $I_{R3}$ to the R3 indicator, $I_A$ to the Pareto front approximation indicator, $I_S$ to the spacing indicator, and $I_R$ to the range covering indicator. We report the results for the 20%, 50% and 80% attainment surfaces on all the instances in order to discuss the different quality measures with respect to different attainment ratios. The numbers in bold indicate the superiority of the solution method with respect to the considered quality measure. For $I_H$ and $I_R$ larger values are desirable, whereas for $I_\epsilon$, $I_{R3}$, $I_A$, and $I_S$ smaller values are beneficial.

It can be seen that for most of the measures the P-ACO algorithm provides better results than P-VNS on average over all instances. This is consistent for all three attainment surfaces (20%, 50% and 80%).

In Table 5, the obtained results for the metrics for 80% attainment are reported. For small size instances, the P-ACO algorithm provides better results for the $I_H$, $I_\varepsilon$, $I_S$, and $I_R$ measure on average. The values for measures $I_{R3}$ and $I_A$ are comparable. For medium

**Table 3** Metrics for 20% attainment

| Set | P-ACO | | | | | | P-VNS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ |
| 2_p21 | 3.480 | 1.045 | −0.116 | 0.009 | 0.987 | 0.503 | **3.525** | **1.000** | **0.000** | **0.000** | **0.975** | **0.545** |
| 2_p32 | **3.393** | **1.002** | **−0.002** | **0.000** | 0.556 | **0.936** | 3.288 | 1.106 | −0.278 | 0.063 | **0.543** | 0.805 |
| 2_p33 | 3.642 | **1.007** | −0.011 | 0.002 | **0.308** | 0.820 | **3.646** | 1.008 | **−0.003** | **0.001** | **0.308** | **0.824** |
| Average (small) | **3.505** | **1.018** | **−0.043** | **0.004** | 0.617 | **0.753** | 3.486 | 1.038 | −0.093 | 0.021 | **0.608** | 0.725 |
| 2_p64(dia) | 3.635 | 1.027 | −0.037 | 0.016 | 0.084 | 0.851 | **3.662** | **1.015** | **−0.007** | **0.002** | **0.081** | **0.867** |
| 2_p66(squ) | **3.636** | **1.022** | **−0.026** | 0.013 | 0.156 | **0.849** | 3.633 | 1.031 | −0.035 | **0.003** | **0.155** | 0.833 |
| 2_p97(pad) | 3.581 | 1.009 | −0.002 | 0.007 | **0.964** | **0.449** | **3.586** | **1.000** | **0.000** | **0.000** | 0.976 | **0.449** |
| Average (medium) | 3.617 | 1.019 | −0.022 | 0.012 | **0.401** | **0.716** | **3.627** | **1.015** | **−0.014** | **0.002** | 0.404 | **0.716** |
| 2_p292(wie) | 3.574 | 1.104 | −0.245 | 0.058 | 0.181 | 0.757 | **3.630** | **1.061** | **−0.126** | **0.029** | **0.123** | **0.820** |
| 2_p559(stm) | 3.587 | 1.068 | −0.334 | 0.036 | 0.412 | **0.623** | **3.642** | **1.044** | **−0.120** | **0.018** | **0.399** | 0.615 |
| 2_p562(ktn) | **3.636** | **1.054** | **−0.149** | **0.026** | 0.186 | **0.797** | 3.582 | 1.080 | −0.279 | 0.039 | **0.159** | 0.743 |
| 2_p2143(noe) | **3.765** | **1.084** | **−0.496** | **0.054** | 0.088 | **0.572** | 3.683 | 1.106 | −0.963 | 0.071 | **0.034** | 0.502 |
| Average (large) | **3.641** | 1.078 | **−0.306** | 0.044 | 0.217 | **0.687** | 3.635 | **1.073** | −0.372 | **0.039** | **0.179** | 0.670 |
| Average (all) | **3.593** | **1.042** | **−0.142** | **0.022** | 0.392 | **0.716** | 3.588 | 1.045 | −0.181 | 0.023 | **0.375** | 0.700 |

**Table 4** Metrics for 50% attainment

| Set | P-ACO | | | | | | P-VNS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ |
| 2_p21 | 3.472 | 1.058 | −0.122 | 0.013 | 1.013 | **0.503** | **3.480** | **1.045** | **−0.116** | **0.009** | **0.987** | **0.503** |
| 2_p32 | **3.379** | **1.016** | **−0.030** | **0.005** | 0.558 | **0.914** | 3.272 | 1.120 | −0.308 | 0.068 | **0.544** | 0.785 |
| 2_p33 | **3.624** | **1.027** | **−0.042** | 0.008 | **0.309** | **0.803** | 3.598 | 1.050 | −0.107 | **0.007** | 0.320 | 0.737 |
| Average (small) | **3.492** | **1.034** | **−0.065** | **0.009** | 0.626 | 0.740 | 3.450 | 1.072 | −0.177 | 0.028 | **0.617** | **0.675** |
| 2_p64(dia) | 3.597 | **1.039** | −0.081 | 0.030 | 0.081 | **0.846** | **3.600** | 1.043 | **−0.075** | **0.013** | **0.079** | 0.829 |
| 2_p66(squ) | **3.612** | **1.034** | **−0.056** | 0.024 | **0.156** | **0.855** | 3.578 | 1.062 | −0.124 | **0.014** | **0.156** | 0.786 |
| 2_p97(pad) | 3.581 | 1.009 | −0.002 | 0.007 | **0.964** | **0.449** | **3.586** | **1.000** | **0.000** | **0.000** | 0.976 | **0.449** |
| Average (medium) | **3.597** | **1.028** | **−0.046** | 0.020 | **0.400** | 0.717 | 3.588 | 1.035 | −0.066 | **0.009** | 0.403 | **0.688** |
| 2_p292(wie) | 3.437 | 1.162 | −0.503 | 0.105 | 0.178 | 0.673 | **3.461** | **1.104** | **−0.455** | **0.072** | **0.121** | **0.791** |
| 2_p559(stm) | 3.462 | **1.112** | −0.764 | **0.078** | **0.408** | **0.623** | **3.487** | 1.123 | **−0.568** | 0.080 | 0.437 | 0.541 |
| 2_p562(ktn) | **3.498** | **1.110** | **−0.455** | 0.081 | 0.192 | **0.725** | 3.427 | 1.146 | −0.721 | **0.072** | **0.158** | 0.688 |
| 2_p2143(noe) | **3.413** | **1.199** | **−1.986** | **0.154** | 0.037 | **0.537** | 3.205 | 1.247 | −3.368 | 0.212 | **0.036** | 0.437 |
| Average (large) | **3.453** | **1.146** | **−0.927** | **0.104** | 0.204 | 0.640 | 3.395 | 1.155 | −1.278 | 0.109 | **0.188** | **0.614** |
| Average (all) | **3.508** | **1.077** | **−0.404** | **0.051** | 0.389 | **0.693** | 3.469 | 1.094 | −0.584 | 0.055 | **0.381** | 0.655 |

**Table 5** Metrics for 80% attainment

| Set | P-ACO | | | | | | P-VNS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ | $I_H$ | $I_\varepsilon$ | $I_{R3}$ | $I_A$ | $I_S$ | $I_R$ |
| 2_p21 | 3.329 | 1.173 | −0.486 | 0.081 | **0.940** | 0.394 | **3.480** | **1.045** | **−0.116** | **0.009** | 0.987 | **0.503** |
| 2_p32 | **3.376** | **1.022** | **−0.033** | **0.007** | 0.558 | **0.914** | 3.236 | 1.160 | −0.377 | 0.077 | **0.547** | 0.760 |
| 2_p33 | **3.599** | **1.051** | **−0.111** | **0.010** | **0.315** | **0.781** | 3.583 | 1.063 | −0.133 | 0.011 | 0.323 | 0.710 |
| Average (small) | **3.435** | **1.082** | −0.210 | 0.033 | **0.604** | **0.696** | 3.433 | 1.089 | **−0.209** | **0.032** | 0.619 | 0.658 |
| 2_p64(dia) | **3.563** | **1.052** | **−0.131** | 0.037 | 0.083 | **0.832** | 3.543 | 1.071 | −0.168 | **0.021** | **0.079** | 0.798 |
| 2_p66(squ) | **3.593** | **1.039** | **−0.082** | 0.031 | 0.157 | **0.837** | 3.532 | 1.087 | −0.214 | **0.024** | **0.156** | 0.777 |
| 2_p97(pad) | 3.541 | 1.038 | −0.116 | 0.007 | **0.964** | 0.431 | **3.586** | **1.000** | **0.000** | **0.000** | 0.976 | **0.449** |
| Average (medium) | **3.565** | **1.043** | **−0.109** | 0.025 | **0.401** | **0.700** | 3.554 | 1.053 | −0.128 | **0.015** | 0.404 | 0.675 |
| 2_p292(wie) | **3.365** | 1.192 | **−0.668** | 0.133 | 0.179 | 0.618 | 3.321 | **1.151** | −0.804 | **0.108** | **0.136** | **0.707** |
| 2_p559(stm) | 3.376 | **1.144** | −1.118 | 0.115 | **0.417** | **0.615** | **3.390** | 1.160 | **−0.878** | **0.111** | 0.443 | 0.518 |
| 2_p562(ktn) | **3.413** | **1.134** | **−0.697** | **0.104** | **0.197** | **0.741** | 3.337 | 1.179 | −0.986 | 0.106 | 0.199 | 0.644 |
| 2_p2143(noe) | **3.263** | **1.248** | **−2.731** | **0.202** | 0.044 | **0.556** | 2.815 | 1.376 | −5.716 | 0.344 | **0.029** | 0.352 |
| Average (large) | **3.354** | **1.179** | **−1.304** | **0.138** | 0.209 | **0.632** | 3.216 | 1.217 | −2.096 | 0.167 | **0.202** | 0.555 |
| Average (all) | **3.442** | **1.109** | **−0.617** | **0.073** | **0.385** | **0.672** | 3.382 | 1.129 | −0.939 | 0.081 | 0.387 | 0.622 |

size instances, P-ACO outperforms P-VNS in all the measures except for $I_A$. The same picture can be observed for larger instances. P-ACO provides better results for almost all performance indicators on the average values except for indicator $I_S$. The average values over all sizes for 80% attainment show that the P-ACO algorithm always outperforms the P-VNS algorithm. Only for the spacing indicator $I_S$, it can be seen that for 20% attainment and for 50% attainment P-VNS provides a better distribution of the solutions on the Pareto frontier. For all the other indicators, the P-ACO algorithm is superior.

Summarizing the results of Tables 3 to 5, we can observe that, while P-ACO seems to work slightly better, the average indicator values for both methods are rather similar. Hence we can conclude that:

- Both methods are appropriate approaches for approximating the Pareto frontier.
- When uniform dispersion of the solutions in the objective space is very important, P-VNS shows some advantages.
- For all other indicators, like, for example, the attainment of extreme points or the size of the region covered by the found solution set, P-ACO is the method of choice, since it provides slightly better indicators on average.
- There is no class of instances for which one method is always better on all instances. P-ACO is also better in the class of medium sized instances, but as the example of Fig. 3 shows, there are some cases for which P-VNS is clearly superior.
- For small, medium and large instances the conclusion is the same. This holds true also for random (2_p21 to 2_p33), geometric (2_p64 and 2_p66) and real world instances (2_p97 to 2_p2143).
- The comparison might be slightly biased in favor of P-ACO since the run times were chosen appropriate for P-ACO and then P-VNS was given the same run time. It might be possible that for smaller or larger run times, P-VNS can catch up with P-ACO.

## 5 Conclusions

We developed two metaheuristic solution techniques for the bi-objective orienteering problem. The motivation of the problem stems from planning individual tourist routes in cities and rural areas. The P-ACO metaheuristic was adapted to solve the MOOP. Moreover, a novel variable neighborhood search based Pareto extension (the P-VNS) was developed. The solution methods were applied to artificial and real world problem sets. The real world sets stem from the cities of Vienna and Padua and some rural regions in Austria. State-of-the-art performance indicators for multi-objective optimization were used to report the performance of the developed solution techniques. The computational results showed that both approaches are well suited to tackle the MOOP also for problems of real-world size.

Both implemented approaches were also competitive for single objective optimization problems. New best results were found on standard benchmark instances (Chao et al. 1996a).

For the bi-objective case, both solution approaches are valuable. It can be seen that the P-ACO procedure slightly outperforms the P-VNS on average over all test instances. In the future, these solution procedures will be integrated in an interactive decision support system to assist tourists in planning their trips and excursions.

**Acknowledgements** We would like to thank Martina Firtik and Magdalena Wipfler for collecting and processing the real world data. We are grateful to the editor in chief and two anonymous referees for their valuable comments that helped to improve the quality of the paper.

## References

Archetti, C., Hertz, A., & Speranza, M. G. (2007). Metaheuristics for the team orienteering problem. *Journal of Heuristics*, *13*(1), 49–76.

Chao, I.-M., Golden, B. L., & Wasil, E. A. (1996a). A fast and effective heuristic for the orienteering problem. *European Journal of Operations Research*, *88*(3), 475–489.

Chao, I.-M., Golden, B. L., & Wasil, E. A. (1996b). The team orienteering problem. *European Journal of Operational Research*, *88*(3), 464–474.

Coello Coello, C. A., Van Veldhuizen, D. A., & Lamont, G. B. (2007). *Evolutionary algorithms for solving multi-objective problems* (2nd ed.). *Genetic algorithms and evolutionary computation* (Vol. 5, pp. 88–113) New York: Springer. Chap. 2

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, *6*(6), 791–812.

Czyzak, P., & Jaszkiewicz, A. (1996). A multiobjective metaheuristic approach to the localization of a chain of petrol stations by the capital budgeting model. *Control and Cybernetics*, *25*(1), 177–187.

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

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research*, *131*(1), 79–99.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2006a). Nature-inspired metaheuristics for multiobjective activity crashing. *Omega*, *36*(6), 1019–1037.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2006b). Pareto ant colony optimization in multiobjective project portfolio selection with ilp preprocessing. *European Journal of Operations Research*, *171*(3), 830–841.

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

Dorigo, M., & Di Caro, G. (1999). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimization* (pp. 11–32). London: McGraw-Hill. Chap. 2.

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the travelling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66.

Dorigo, M., & Stuetzle, T. (2004). *Ant colony optimization*. Cambridge: MIT Press.

Ehrgott, M., & Gandibleux, X. (2000). A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, *22*(4), 425–460.

Ehrgott, M., & Gandibleux, X. (2004). Approximative solution methods for multiobjective combinatorial optimization. *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research*, *12*(1), 1–63.

Feillet, D., Dejax, P., & Gendreau, M. (2005). Travelling salesman problems with profits. *Transportation Science*, *39*(2), 188–205.

Garcia-Martinez, C., Cordon, O., & Herrera, F. (2007). A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for bi-criteria TSP. *European Journal of Operational Research*, *180*(1), 116–148.

Glover, F., & Laguna, M. (1997). *Tabu search*. Norwell: Kluwer Academic.

Glover, F., Laguna, M., & Martí, R. (2003). Scatter search and path relinking: Advances and applications. In F. Glover & G. A. Kochenberger (Eds.), *International series in operations research and management science: Vol. 57. Handbook of Metaheuristics* (pp. 1–35). New York: Springer.

Grunert da Fonseca, V., Fonseca, C. M., & Hall, A. O. (2001). Inferential performance assessment of stochastic optimisers and the attainment function. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello, & D. Corne (Eds.), *Lecture notes in computer science: Vol. 1993. First International Conference on Evolutionary Multi-Criterion Optimization* (pp. 213–225). New York: Springer.

Hansen, M. P., & Jaszkiewicz, A. (1998). *Evaluating the quality of approximations to the non-dominated set* (Technical Report IMM-REP-1998-7). Technical University of Denmark, Kgs. Lyngby, Denmark.

Jozefowiez, N., Glover, F., & Laguna, M. (2008). Multi-objective meta-heuristics for the traveling salesman problem with profits. *Journal of Mathematical Modelling and Algorithms*, *7*(2), 177–195.

Kindervater, G. A. P., & Savelsbergh, M. W. P. (1997). Vehicle routing: Handling edge exchanges. In E. Aarts & J. K. Lenstra (Eds.), *Local search in combinatorial optimization* (pp. 311–336). Chichester: Wiley.

Knowles, J. D., Thiele, L., & Zitzler, E. (2006). *A tutorial on the performance assessment of stochastic multiobjective optimizers* (Technical Report 214). Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, February 2006.

Mladenović, N. (1995). A variable neighborhood algorithm: A new metaheuristic for combinatorial optimization. In *Abstract of papers presented at Optimization Days*, p. 112, Montréal, Canada, 1995.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, *24*(11), 1097–1100.

Pasia, J. M., Hartl, R. F., & Doerner, K. F. (2006). Solving a bi-objective flowshop scheduling problem by Pareto-ant colony optimization. In M. Dorigo, L. M. Gambardella, M. Birattari, A. Martinoli, R. Poli, & T. Stützle (Eds.), *Lecture notes in computer science: Vol. 4150. Ant colony optimization and swarm intelligence* (pp. 294–305). Berlin: Springer.

Pasia, J. M., Gandibleux, X., Hartl, R. F., & Doerner, K. F. (2007). Local search guided by path relinking and heuristic bounds. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, & T. Murata (Eds.), *Lecture notes in computer science: Vol. 4403. Evolutionary multi-criterion optimization* (pp. 501–515). Berlin: Springer.

Schott, J. R. (1995). *Fault tolerant design using single and multicriteria genetic algorithm optimization*. Master's thesis, Department of Aeronautics and Astronautics, MIT, Cambridge, MA.

Souffriau, W., Vansteenwegen, P., Vanden Berghe, G., & Van Oudheusden, D. (2008). A greedy randomised adaptive search procedure for the team orienteering problem. In *EU/MEeting 2008*, Troyes, France, October 23–24, 2008.

Steuer, R. E., Gardiner, L. R., & Gray, J. (1996). A bibliographic survey of the activities and international nature of multiple criteria decision making. *Journal of Multi-Criteria Decision Analysis*, *5*(3), 195–217.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *The Journal of the Operational Research Society*, *35*(9), 797–809.

Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., & Van Oudheusden, D. (2009). A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, *196*(1), 118–127.

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

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