A Metaheuristic Algorithm for a Multi-period Orienteering Problem Arising in a Car Patrolling Application

Full paper

Giorgio Zucchi
School of Doctorate E4E, University of Modena and Reggio Emilia
R&D department, Coopservice s.c.p.a
Reggio Emilia, Italy
giorgio.zucchi@unimore.it

Victor Hugo Vidigal Corrêa
Department of Informatics, Federal University of Viçosa
Viçosa, Brazil
victor.vidigal@ufv.br

Manuel Iori
Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia
Reggio Emilia, Italy
manuel.iori@unimore.it

André Gustavo dos Santos
Department of Informatics, Federal University of Viçosa
Viçosa, Brazil
andre@dpi.ufv.br

Mutsunori Yagiura
Graduate School of Informatics, Nagoya University
Nagoya, Japan
yagiura@nagoya-u.jp

ABSTRACT

This paper addresses a real-life multi-period orienteering problem arising in a large Italian company that needs to patrol a vast area in order to provide security services. The area is divided into clusters, and each cluster is assigned to a patrol. A cluster comprises a set of customers, each requiring different services on a weekly basis. Some services are mandatory, while others are optional. It might be impossible to perform all optional services, and each of them is assigned a score when performed. The challenge is to determine a set of routes, one for each patrol and each day, that maximizes the total collected score, while meeting a number of operational constraints, including minimum quality of service, hard time windows, maximum riding time, and minimum time between two consecutive visits for the same service at the same customer. To solve the problem, we propose an iterated local search that invokes at each iteration an inner variable neighborhood descent procedure. Computational tests performed on a number of real-life instances prove that the developed algorithm is very efficient and finds in a short time solutions that are consistently better than those in use at the company.

1 INTRODUCTION

Every day, private security guards need to inspect structures, parks, buildings, and many other facilities to check for any anomalies, in order to counter potential criminal actions or simply restore normal safety conditions following forgetfulness or breakdowns. In this paper, we study a real-life security problem in which patrols are required to perform a set of services at customers located in a vast area. Some services are mandatory, while others are optional. The optional services, when performed, induce a score, and the aim is to maximize the collected score, while meeting different operational constraints.

The problem originates from the everyday activity of Coopservice, a large service provider company located in Italy (https://www.coopservice.it/). Counting on more than 25 000 employees, Coopservice operates a number of different services, including logistics, transportation, cleaning, maintenance, and security. The company also operates car patrolling services in a number of provinces all around Italy. An example of the patrolling activity performed in the province of Reggio Emilia is provided in Figure 1.

The customers are geographically dispersed in the area and are consequently divided into clusters. Each cluster is assigned to a patrol, who performs every day a route to visit customers and executes the required services. The cluster division does not vary from a day to the other, but the routes performed inside the clusters may change. Indeed, customers may require different services according to the day of the week, following the contract stipulated

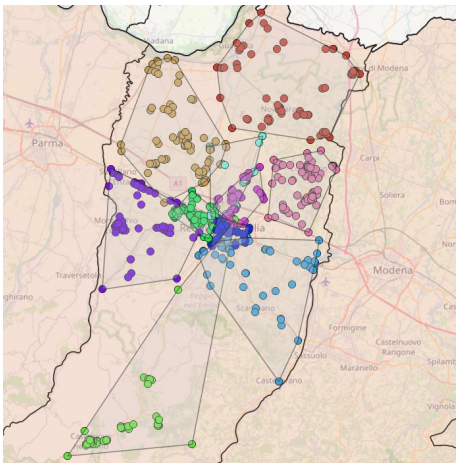

Figure 1: Customers and clusters for the patrolling of the Reggio Emilia province

with the company. More in detail, each customer may require multiple services, and, for each such service, multiple visits during the same period. Some services, such as the closing or opening of a commercial activity, are mandatory, whereas others, such as the inspection of an area or a building, are optional. The optional services create a score when performed, and the company is interested in maximizing the collected score.

The resulting optimization problem is very difficult, as it involves a number of operational constraints. First of all, the services should be performed within hard time windows, and the routes should not exceed a maximum riding time. In addition, a customer might require multiple visits for the same service in the same period. In such a case, two consecutive visits should be separated by at least a threshold time (i.e., 90 minutes or so). This constraint is indeed very challenging, as it requires us to schedule endogenous time windows, induced by the consecutive visits, inside the exogenous time window received in input. Moreover, the company is interested in maintaining a minimum quality of service (QoS) level. The QoS level is computed as the ratio between the number of optional services that have been performed and the number of optional services that have been required. Ideally, the QoS level should be balanced among all customers at the end of the week.

Our aim is to determine a set of routes, one for each period and each cluster, by keeping unchanged the cluster configuration received in input and maximizing the total collected score. The resulting problem is thus a multi-period orienteering problem, which is a generalization of the well-known orienteering problem (OP) [4]. The OP is known to be strongly NP-hard and difficult to solve in practice, and the problem we are facing is a challenging generalization of the OP that includes different additional constraints. For this reason, we decided to solve the problem by means of a metaheuristic algorithm.

We developed an iterated local search (ILS), a metaheuristic that obtained in recent years relevant results on a huge number of optimization problems [7]. The ILS receives in input the set of customers, the set of services to be performed, the cluster configuration, and all details of the instance. It builds an initial solution by means of a constructive heuristic. Then, while the time limit is not reached, it introduces perturbations on the current solution and then improves it by means of a variable neighborhood descent (VND) procedure [6] based on five neighborhood operations.

Computational tests on a set of real-life instances provided by the company prove that the developed ILS works very well in practice. The solutions obtained by the ILS consistently improve the ones in use at the company in terms of the total score. In addition, the average QoS level is also improved.

The remainder of the paper is organized as follows. Section 2 contains a brief literature review of orienteering problems and car patrolling applications. Section 3 formally describes the problem we solve. Section 4 presents the ILS algorithm proposed in this paper. Section 5 shows the results obtained and, finally, Section 6 gives some concluding remarks and hints for future research directions.

2 LITERATURE REVIEW

Car patrolling is a security measure widely used to protect large areas from criminal activity. It consists of guards (patrols) using vehicles to move between points of interest in a region and taking actions that may prevent or respond to crimes. Samanta et al. [9] surveys the police patrolling problem considering every component involved in this complex operation and divides it into three categories: (i) resource allocation, (ii) district design, and (iii) route design. Among these categories, the route design is the one that most resembles our problem, since it is concerned with how the routes are selected and how they affect the patrols' efficiency. Many techniques are used to solve this type of problem and many study cases arise in this context. Some examples are the maximization of the vehicle patrolling coverage in Israel [1], and the minimization of patrols' idle time and unpredictability in London and Chicago [2].

In general, the problems evaluated in [9] differ from ours, due to how Coopservice needs to provide its patrolling service. In fact, in our routing design, not necessarily all clients need to be visited, so our problem is more similar to an orienteering problem (OP). The literature on OPs is very rich and we found plenty of applications. The first study on the OP dates back to 1984 [10], where the OP was presented as a generalization of the traveling salesman problem. Since it is an NP-hard problem, most studies use heuristic methods to solve the OP and its variants. Gendreau et al. [3] discuss why it is so difficult to design high-quality heuristics for this type of problem. The score of a location and the distance to reach it are independent and often in contrast to one another, so it is difficult to select the locations that are part of an optimal solution. Therefore, simple construction heuristics may direct the algorithm towards undesirable directions and are not sufficient to explore large parts of the solution space.

A hybrid heuristic composed of a greedy randomized adaptive search procedure (GRASP) and a variable neighborhood search (VNS) has been proposed by [8] to solve a generalization of the OP. This variant has constraints related to mandatory visits and incompatibilities among nodes. Those constraints mean that there is a set of nodes that must be visited in order for the solution to be considered feasible, while visiting some other nodes is optional. In addition, incompatible nodes cannot share the same route. The hybrid heuristic takes advantage of the multi-start feature of the GRASP to generate initial solutions that are then optimized with the VNS. The authors report that the heuristic was able to find 128 optimal solutions on a set of 131 instances and required, on average, only 0.8% of the time required by an integer linear programming model solved with commercial software.

Recent applications of the OP were studied also for tourism trip planning. For example, Vansteenwegen et al. [11] developed an iterated local search (ILS) metaheuristic to a problem that requires determining a set of touristic locations to visit, while satisfying time limits and budget criteria. The local search step is done through an insertion procedure in which feasible points are added into routes until a locally optimal solution is reached. In the shake step, some visited points are removed from routes according to some predefined parameters. The authors report that, in their experiments, the average gap between the optimal solution value and the ILS one is 2.1%. In [5], the goal is to optimize touristic routes considering constraints such as visit redundancy avoidance and time windows, where some attractions might have a shorter visiting time than others. An integer linear programming model and an ILS metaheuristic

are proposed. The authors report that the ILS could almost match the results from the solver Gurobi, used to execute the model, for the smaller instances, while for the larger instances it could provide better solutions in most cases.

Outside the scope of metaheuristic algorithms, in [12] an approximation algorithm for a variant of the team orienteering problem (TOP) was proposed. In addition to the basic TOP constraints and the basic objective of maximizing the collected score, their problem includes a set of new features to better model Internet of things applications. The new features of the problem are the following: a limited budget is imposed on the vehicles to perform the routes, the node costs are included in the path cost function in addition to the edge costs, and the nodes can be served by multiple vehicles. Computational experiments proved that the developed algorithm provided up to a 17.5% increase in the collected score than the state-of-the-art algorithms for this same TOP variant.

3 PROBLEM DESCRIPTION

The area to be patrolled is divided into clusters, but the cluster configuration is not part of the optimization process. For this reason, we formally describe our problem for a unique cluster served by a unique car.

The problem we face can be viewed as a multi-period orienteering problem with time windows (MPOPTW). In the MPOPTW, we are given a graph $G = (V_0, A)$. The set of vertices is defined as $V_0 = \{0, 1, \ldots, n\}$, where 0 is the depot at which the single vehicle starts and ends each route, and $V = \{1, \ldots, n\}$ is the set of customer locations. The graph is complete, and with each arc $(i, j) \in A$, we associate a traveling time t_{ij}. The time matrix is asymmetric.

Let T be the set of services provided by the company. A standard service time is given for each service $t \in T$. This is denoted by q_t and gives the time duration necessary for a patrol to stay at a customer location to execute the service. The set of services is partitioned as $T = M \cup U$, where M is the set of mandatory services and U is the set of optional services. A score w_t is associated with each optional service $t \in U$, which represents the level of importance of the service.

The activities should be executed on a given set $D \subseteq \{1, \ldots, 7\}$ of periods. Each period corresponds to the working hours from 22:00 of a day to 06:00 of the next day in our instances, as all activities are performed at night time. Each customer $v \in V$ requires services on a subset $D_v \subseteq D$ of periods. Formally, we denote by $T_{vd} \subseteq T$ the set of services to be performed at customer v on period d. This set is partitioned as $T_{vd} = M_{vd} \cup U_{vd}$, where $M_{vd} \subseteq M$ comprises mandatory services and $U_{vd} \subseteq U$ optional ones.

Let n_{vdt} be the number of times service t is required by customer v in period d, and let $\widetilde{n}_{vdt}$ be the number of services that have been actually performed. We define the QoS level as $QoS = \sum_{v \in V,\, d \in D_v,\, t \in T_{vd}} \widetilde{n}_{vdt} / n_{vdt}$. This index represents the ratio of services that have been performed in the entire set of periods, and it should be at least a required value $QoS_{\min}$ (which is set to 75% in our instances).

Every service t required by a customer v in a day d has a time window $[e_{vdt}, l_{vdt}]$. This defines the earliest and latest possible times to start the execution of each of the n_{vdt} visits for that customer in that period. The time window defines a hard constraint, so late arrivals are forbidden and waiting on site is imposed in case of early arrivals. A time window is also imposed on the depot and corresponds to the total working time (from 22:00 of a day to 06:00 of the next day in our instances).

For some services, such as closing or opening a commercial activity, the time window is strict (e.g., 10 minutes) and just one visit per night is required. This typically happens for mandatory services. For other services, such as checking a private house, the time window is usually loose (e.g., several hours during the night) but multiple visits are required in a period. This typically happens for optional services. In the latter case, if two or more visits are performed for the same service at the same customer in the same period, the start times of any two of such visits should be separated by at least a given threshold δ (which is equal to 90 minutes in our instances). This is imposed to enforce a balanced patrol of the customer during the execution of a route.

For each period, a patrol starts its route at the depot, performs visits to customers to execute the services, and then returns to the depot. The total riding time, which comprises traveling, service, and waiting times, is the difference between the start and end times of the route, and it should not exceed a given upper bound $Q_{\max}$.

To summarize, the aim of the MPOPTW is to define a set of routes, one per period, in such a way that (i) constraints on hard time windows, the QoS level, the time distance between consecutive visits, and the total riding time are satisfied; (ii) all mandatory services are performed; and (iii) the score of the optional services that have been actually performed is maximized.

The problem is of interest not only because of its real-life application, but also because it is very general and models a large number of other possible applications arising in the context of car patrolling and attended home services. In the next section, its solution is pursued by means of a metaheuristic algorithm.

4 PROPOSED METHODOLOGY

To solve the problem, we developed an ILS metaheuristic [7]. The ILS receives as input the set of customers, the set of services to be performed, the cluster configuration, and other details of the real-life application. It builds an initial solution using a constructive heuristic and then applies perturbations and local search iteratively in the current solution until a time limit is reached. An acceptance function decides at each iteration whether to keep the current solution or to move to a newly-generated one. In our algorithm, it always chooses the best solution among the two, and if their fitness values are the same, our algorithm keeps the current one. The overall ILS procedure is presented in Algorithm 1. Each step is described in detail in the remaining part of this section. The local search is performed by means of a VND, an algorithm that sequentially invokes a set of neighborhoods [6].

4.1 Solution evaluation

The objective of the problem is to maximize the total score achieved with the optional services performed by the patrol. However, in order to guide our algorithm, we use a fitness function that, together with the score, takes into account also the riding time of the routes. Let $\mathcal{S}(\sigma)$ be the total score of a given route σ and $\mathcal{T}(\sigma)$ be its riding

Algorithm 1 ILS algorithm

```
1: procedure ILS(T_max)
2:     s* ← ConstructiveHeuristic
3:     s* ← VND(s*)
4:     while elapsedTime ≤ T_max do
5:         s' ← Perturbation(s*)
6:         s'' ← VND(s')
7:         s* ← Accept(s*, s'') ▷ the best of s* and s'' with respect to F
8:     end while
9:     return the best feasible solution found during the search
10: end procedure
```

time. The fitness function used by the proposed ILS algorithm to evaluate a solution s is a weighted average of these values, namely

$$\mathcal{F}(s) = \alpha \sum_{\sigma \in s} \mathcal{S}(\sigma) - \beta \sum_{\sigma \in s} \mathcal{T}(\sigma), \tag{1}$$

where α and β are weights to be calibrated. By doing this, we expect that the algorithm favors shorter routes during the optimization in order to add more services later on, thus improving the score.

4.2 Constructive heuristic

An initial solution is constructed by a greedy algorithm. The services of each period are sorted in non-decreasing order of the start time of their time windows. A route is constructed for each period in two phases: first, the mandatory services are inserted sequentially, in the order in which they were sorted (this is always feasible for the mandatory services in the instances provided by the company); later, while possible, optional services are appended one by one in the solution (i.e., an optional service is appended if the solution remains feasible, otherwise it is skipped). Algorithm 2 summarizes the steps of this heuristic. In the ILS, the solution obtained by Algorithm 2 is optimized by invoking a VND procedure to the solution built by the constructive heuristic.

Algorithm 2 The greedy constructive heuristic

```
1: procedure Constructive Heuristic(M, U)
2:     for each period d ∈ D do
3:         M_d, U_d ← services in M and U for period d
4:         Sort M_d and U_d in non-decreasing order of e_vdt
5:         σ_d ← AppendAllMandatory(M_d)
6:         while CanAppendOptionals(σ_d, U_d) do
7:             σ_d ← AppendOptionals(σ_d, U_d)
8:         end while
9:     end for
10:    return s = {σ_1, ..., σ_D}
11: end procedure
```

4.3 VND heuristic

A VND procedure is used to find a locally optimal solution using a sequence of different neighborhoods N_k ($k = 1, \ldots, k_{\max}$). Algorithm 3 shows the main steps of the VND heuristic. Starting with the first neighborhood ($k = 1$), the heuristic explores the solution space searching through the sequence of neighborhoods in a deterministic way, controlled by the neighborhood change procedure presented in Algorithm 4. The steps of these algorithms are detailed in the following.

At each step of the VND, a neighbor s' of the current solution s is selected from the neighborhood $N_k(s)$. A neighbor is always selected by a first improvement move in the fitness function $\mathcal{F}$ defined in (1). If there is no better neighbor in the kth neighborhood (i.e., the current solution is locally optimal with respect to this neighborhood), the algorithm changes to the next neighborhood, N_{k+1}. If, instead, a better neighbor is found, the algorithm returns to the first neighborhood. The process continues while there is a neighborhood to be explored, that is, it stops when the current solution is locally optimal with respect to all neighborhoods.

For the VND, we implemented five classical neighborhood operators from the traveling salesman and vehicle routing literature. **Swap**: this operator swaps the positions of two visits inside a route. **2-opt**: this operator reverses the visiting order between two visits in a route. **Relocate**: this operator moves a visit to another position in the route. **Swap unrouted**: this operator swaps the status of two services, an unrouted service takes place of a performed service. **Insertion unrouted**: this last operator inserts a visit in a route to perform an unrouted service, increasing the number of services performed by the patrol. Note that the first three operators, Swap, 2-opt and Relocate, may improve the fitness function by reducing the riding time of a route, as they do not change the collected score. The operator Swap unrouted may instead improve the fitness function by increasing the score or by reducing the riding time. The last operator, Insertion unrouted, may improve the score at the expense of an increase in the riding time.

Note that all described neighborhoods consist of intra-period movements, in which they change routes of each period independently. A solution may be further improved by performing inter-period movements, exchanging services from different periods. For example, a customer requiring services in more than one period that is currently served in only a subset of those periods may have the visits changed without changing the total score, perhaps decreasing the riding time. Inter-period movements are costly to be evaluated because of the large number of neighbors. Hence, they are not fully explored in a deterministic way, but are considered in the perturbation step, described next.

Algorithm 3 Variable neighborhood descent heuristic

```
1: procedure VND(s)
2:     k ← 0
3:     s* ← s
4:     while k ≤ k_max do
5:         Let s be an improved sol. in N_k(s*) if any; otherwise s ← s*
6:         s*, k ← NeighborhoodChange(s*, s, k)
7:     end while
8:     return s*
9: end procedure
```

4.4 Perturbation procedure

The perturbation procedure is introduced to escape from the locally optimal solution obtained by the VND used in the local search step. Two inter-period operators are used, which both attempt to modify the current solution in the set of periods.

Algorithm 4 Neighborhood change procedure

```
procedure NeighborhoodChange(s*, s, k)
    if F(s) > F(s*) then
        s* ← s
        k ← 1
    else
        k ← k + 1
    end if
    return s*, k
end procedure
```

Two periods are randomly chosen and an arbitrary number of shaking movements are applied, limited by a maximum number of tries and a maximum number of successful movements. The **Swap inter-period** operator randomly chooses a service in the route of a chosen period and tries to swap it with every other service in a successive period. The **Move inter-period** chooses one service routed in a period and tries to insert it in every position of the route of a different period. In either case, if a try succeeds in finding an improved solution, then the move is applied. Only feasible moves are considered.

5 COMPUTATIONAL EVALUATION

In this section, we present the computational results that we obtained. First, we briefly describe the instances we address. Then, we report the calibration of the main ILS parameters. Finally, we present the results obtained by the ILS. The algorithm was coded in Python 3.7.3 and executed on an Intel Xeon CPU E5-2640 v3 2.60 GHz with 64 GB of memory, running under Windows 10 Pro 20H2 64-bits.

5.1 Instances

The company provides security services in a number of provinces of Italy. We were provided with the data of four of such provinces. Each of them is different in the number of customers and frequency of tasks. Table 1 reports for each province, in order, the number of instances, the number of periods (column $|D|$), the total number of customers (column $|V|$), and the total number of services (column $|T|$). The traveling times have been obtained by using the OSRM application.

Table 1: Details of the real-life instances

Province	Instances	$\|D\|$	$\|V\|$	$\|T\|$
Parma	5	7	175	1382
Pescara	5	7	160	987
Sassari	9	7	122	1044
Roma	8	7	121	1234

5.2 Parameter calibration

As the size of the instances changes by province, we defined a different time limit $T_{\max}$ (the maximum time allowed for the ILS execution) for each of them, based on the number of customers. We set $T_{\max}$, in minutes, to 120, 120, 60, and 60 for Parma, Pescara, Sassari, and Roma instances, respectively.

As the fitness function $\mathcal{F}$ defined by (1) used to guide the algorithm has two parameters, α and β, we ran preliminary tests for all combinations of $\alpha = [1, 2, 3, 4, 5]$ and $\beta = [0.1, 0.3, 0.5, 0.7, 0.9]$. For any combination, we evaluated the reported solutions considering (i) the total score, (ii) the mean of the distances traveled by the patrols, (iii) the mean of the patrol times, (iv) the mean of the waiting times, and (v) the QoS level obtained. For each instance, a ranking of the parameter combination results was created. To merge all rankings into a unique one, we assigned scores for each ranking as follows: from position 1 to 5 of a ranking the score is 5, from 6 to 10 the score is 4, from 11 to 15 the score is 3, from 16 to 20 the score is 2, and from 21 to 25 the score is 1. Then, we summed the scores for each combination. Note that the higher is the ranking, the larger is the sum. The winning combination was determined as $\alpha = 5$ and $\beta = 0.9$ and was used in all successive experiments.

5.3 Computational experiments

To compare the solutions obtained by the ILS with the ones in use at the company, several criteria have been considered: $\overline{km}$, the average distance traveled by the patrols; $\overline{dv}$, the average distance between two consecutive visits; $\overline{\mathcal{T}}$, the average riding time; $\overline{uc}$, the average number of unvisited customers; $\overline{QoS}$, the average QoS; $\overline{\delta}$, the average time between two visits at the same customer; and $\mathcal{S}$, the total collected score.

Table 2 reports the above parameters computed for the solutions in use at the company. It is important to highlight that the solutions by the company do not always satisfy the constraint on the desired interval of $\delta = 90$ minutes between two visits to the same customer.

Table 2: Evaluation of the solutions in use at the company

Province	$\overline{km}$	$\overline{dv}$	$\overline{\mathcal{T}}$	$\overline{uc}$	$\overline{QoS}$	$\overline{\delta}$	$\mathcal{S}$
Parma	156.85	2.51	431.63	1.40	92.56	85.39	4117.57
Pescara	53.42	1.14	339.69	21.12	48.63	30.74	693.43
Sassari	40.81	2.39	217.21	1.12	92.72	45.38	718.86
Roma	92.56	5.65	379.76	0.50	83.12	87.73	1393.28

Since the ILS algorithm contains a random factor in the perturbation step, we have run the algorithm multiple times for each instance. Table 3 reports the average results obtained for five runs of the ILS, and Table 4 shows the comparison with the real-life solutions currently adopted by the company. Table 5 reports the difference reported in Table 4, but in terms of percentages in order to better highlight the results (we have decided to exclude column $\overline{uc}$ from Table 4 because all values were -100%).

The results obtained by the ILS show several improvements with respect to the real ones. The only negative result is noticed for the distance $\overline{km}$ traveled by the vehicles. This happens because, in order to maximize the score, more optional services have been performed, and this requires more km to be traveled. The two measures used in our guide fitness function, $\mathcal{T}$ and $\mathcal{S}$, were highly improved. The average travel time was reduced by 10 to 42% for the tested instances, while the total score was increased by 11 to 96%. Moreover, although not directly used in the fitness function, other criteria were also improved, as they were used to choose the weights

α and β in this function. Note that in our solutions, no customer is left unvisited. Moreover, almost all tasks are performed, as the $\overline{QoS}$ is above 97%. The distance $\overline{dv}$ between two visits is similar to the one obtained by the company, which indicates that the neighborhood operators were able to find routes as good as those designed by the experienced workers of the company. Furthermore, the δ parameter, the interval between two consecutive visits to the same customer, is always respected in the solutions of our ILS algorithm, which we recall is not the case in the solutions by the company.

Table 3: Evaluation of the solutions obtained by the ILS

Province	$\overline{km}$	$\overline{dv}$	$\overline{\mathcal{T}}$	$\overline{uc}$	$\overline{QoS}$	$\overline{\delta}$	$\mathcal{S}$
Parma	179.69	2.17	335.46	0	98.22	153.51	4572.60
Pescara	127.78	3.32	197.35	0	98.29	93.25	1365.20
Sassari	69.98	4.29	170.64	0	98.52	122.85	1052.60
Roma	115.61	5.40	338.04	0	97.10	131.08	1750.60

Table 4: Absolute differences between ILS and company

Province	$\overline{km}$	$\overline{dv}$	$\overline{\mathcal{T}}$	$\overline{uc}$	$\overline{QoS}$	$\overline{\delta}$	$\mathcal{S}$
Parma	22.84	−0.34	−96.17	−1.40	5.66	68.12	455.03
Pescara	74.36	2.18	−142.34	−21.12	49.65	62.50	671.77
Sassari	29.17	1.90	−46.57	−1.12	5.80	77.46	333.74
Roma	23.05	−0.25	−41.71	−0.50	13.98	43.34	357.32

Table 5: Percentage differences between ILS and company

Province	$\overline{km}$	$\overline{dv}$	$\overline{\mathcal{T}}$	$\overline{QoS}$	$\overline{\delta}$	$\mathcal{S}$
Parma	14%	−13%	−22%	6%	79%	11%
Pescara	139%	190%	−42%	102%	203%	96%
Sassari	71%	79%	−21%	6%	171%	46%
Roma	24%	−4%	−10%	17%	49%	26%
Average	**62%**	**63%**	**−23%**	**58%**	**125%**	**44%**

6 CONCLUSIONS AND FUTURE RESEARCH

This paper presented a study on a car patrolling application that arose from a large Italian company, which has to plan routes to perform mandatory and optional services at customers. The resulting optimization problem is a challenging variant of a multi-period orienteering problem. Because of its difficulty, we decided to solve it with an iterated local search (ILS) metaheuristic, which was enriched with a variable neighborhood descent.

Through a computational analysis, we observed that the ILS consistently improved the company solutions. The improvement obtained was remarkable for both the total collected score and for the overall quality of service (a measure of the percentage of optional services that were performed). In the province of Pescara, just to give an example, the quality of service was remarkably increased by 102%. Another important result obtained by the ILS comes from the fact that all solutions were feasible with respect to the constraint that imposes a required elapsed time between two consecutive visits to the same customer. This constraint is indeed not always satisfied in the solutions by the company.

We present four suggestions for future studies. First, the modification of the cluster: the current clusters were provided by the company, but we foresee that changing their configuration might help improve the solutions even further. Second, the insertion of dynamic and stochastic features in the problem: in the current version of the problem, dynamic occurrences such as alarm triggering or unexpected urgent services are not considered; by embedding them into a new problem that considers dynamic and stochastic aspects, we may obtain a more sophisticated model, to be used on-the-fly during the execution of the activities. Third, a more elaborated evaluation of the quality of service: in our study, the quality of service is evaluated using an overall measure of the number of services performed, which may introduce unfairness as some customers may have been poorly served while others fully served; introducing a quality of service measured per single customer might improve the fairness of the resulting solutions. Finally, the development of an integer linear programming model represents another interesting topic for future research. This model might be used to provide proven optimal solutions, or at least to assess the quality of the solutions found by the metaheuristic.

ACKNOWLEDGMENTS

We acknowledge financial support by Coopservice and by JSPS KAKENHI Grant No. 20H02388. A special thank to Hang Dong, Nicolas Porto Campana and Matteo Magnavacchi for their support to the research.

REFERENCES

[1] Ramon Auad and Rajan Batta. 2017. Location-coverage models for preventing attacks on interurban transportation networks. Annals of Operations Research 258, 2 (2017), 679–717.
[2] Huanfa Chen, Tao Cheng, and Sarah Wise. 2017. Developing an online cooperative police patrol routing strategy. Computers, Environment and Urban Systems 62 (2017), 19–29.
[3] Michel Gendreau, Gilbert Laporte, and Frédéric Semet. 1998. A tabu search heuristic for the undirected selective travelling salesman problem. European Journal of Operational Research 106, 2-3 (1998), 539–545.
[4] Bruce L. Golden, Larry Levy, and Rakesh Vohra. 1987. The orienteering problem. Naval Research Logistics 34, 3 (1987), 307–318.
[5] Felix Gündling and Tim Witzel. 2020. Time-dependent tourist tour planning with adjustable profits. In 20th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2020), Dennis Huisman and Christos D. Zaroliagis (Eds.), Vol. 85. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 14:1–14:14.
[6] Pierre Hansen, Nenad Mladenović, Jack Brimberg, and José A. Moreno Pérez. 2019. Variable neighborhood search. In Handbook of Metaheuristics. Springer, 57–97.
[7] Helena Ramalhinho Lourenço, Olivier C. Martin, and Thomas Stützle. 2019. Iterated local search: Framework and applications. In Handbook of Metaheuristics. Springer, 129–168.
[8] Pamela J. Palomo-Martínez, M. Angélica Salazar-Aguilar, Gilbert Laporte, and André Langevin. 2017. A hybrid variable neighborhood search for the orienteering problem with mandatory visits and exclusionary constraints. Computers & Operations Research 78 (2017), 408–419.
[9] Sukanya Samanta, Goutam Sen, and Soumya Kanti Ghosh. 2021. A literature review on police patrolling problems. Annals of Operations Research (2021).
[10] Theodore Tsiligirides. 1984. Heuristic methods applied to orienteering. Journal of the Operational Research Society 35, 9 (1984), 797–809.
[11] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk van Oudheusden. 2009. Metaheuristics for tourist trip planning. In Metaheuristics in the Service Industry. Springer, 15–31.
[12] Wenzheng Xu, Weifa Liang, Zichuan Xu, Jian Peng, Dezhong Peng, Tang Liu, Xiaohua Jia, and Sajal K. Das. 2021. Approximation algorithms for the generalized team orienteering problem and its applications. IEEE/ACM Transactions on Networking 29, 1 (2021), 176–189.