



# Clustered Orienteering Problem with Function-based Rewards

Vinícius L. C. Faria , and Douglas G. Macharet 

**Abstract**—The Orienteering Problem is a well-studied problem that aims to determine an optimal route that maximizes the cumulative rewards while attaining a predefined travel budget. This classical formulation has been generalized in many forms, including grouping customers (locations) into *clusters*. However, such cluster-based formulation typically demands visiting all customers to collect the reward, thereby restricting its applicability. In this paper, we introduce a novel variant in which rewards associated with a cluster can be partially collected following a function that correlates with the number of customers attended within the cluster. This new formulation is a suitable alternative to model scenarios where the reward of the clusters can exhibit distinct and unique scaling patterns. We propose using an evolutionary algorithm to solve this problem and evaluate its performance considering different scenarios and aspects. The results demonstrate that our method effectively selects an appropriate number of clients to attend to based on the assigned function for each cluster.

**Link to graphical and video abstracts, and to code:**  
<https://latamtx.ieeer9.org/index.php/transactions/article/view/9483>

**Index Terms**—Autonomous Agents, Orienteering Problem, Vehicle Routing, Evolutionary Algorithms

## I. INTRODUCTION

THE wide scope of modern routing problems is presented with a broad selection of real-world variants, restrictions, and particularities. Specifically, the class of routing problems with rewards associated, for example, the Orienteering Problem (OP) [1], are observable in just about any context that involves maximizing the collected reward given certain constraints, such as a time limit. Furthermore, in many scenarios, the customers to be served are not all fully independent and can be related to each other by forming groups, which originates the Clustered Orienteering Problem (COP) [2]. Expanding this idea even further, the reward associated with each group can also scale differently and uniquely as more and more customers of the group are served: a certain cluster might need a high amount of customers served to generate significant returns, while another produces diminishing rewards after a certain amount of customers visited.

The associate editor coordinating the review of this manuscript and approving it for publication was Alejandro Dzul (*Corresponding author: Douglas Guimarães Macharet*).

This work was supported by CAPES/Brazil - Finance Code 001, CNPq/Brazil, and FAPEMIG/Brazil.

V. L. C. Faria, and Douglas Guimarães Macharet are with the Computer Vision and Robotics Laboratory (VeRLab), Department of Computer Science, Universidade Federal de Minas Gerais, Brazil (e-mails: [viniciuscensi@dcc.ufmg.br](mailto:viniciuscensi@dcc.ufmg.br), and [doug@dcc.ufmg.br](mailto:doug@dcc.ufmg.br)).

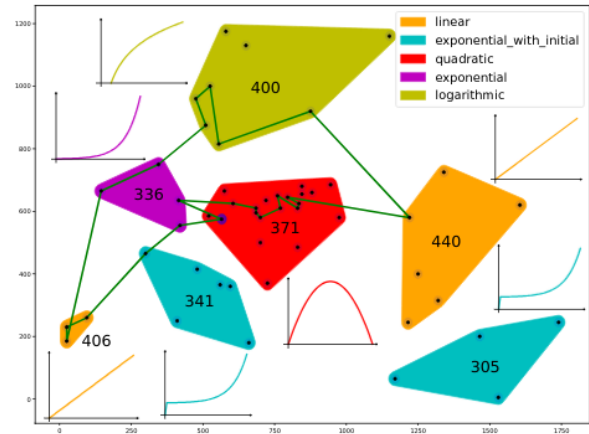


Fig. 1. Example result of a COPF instance adapted from the classic 11berlin52 instance for the Set OP.

The described general scenario has the potential to encapsulate many different applications. This versatility stems from the presence of diverse individual clusters, each accommodating multiple customers and characterized by distinct rewards-generation functions contingent upon the number of individuals served within their respective clusters. For example, when designing 3D aerial scanning routes, it is possible to configure clusters wherein each customer signifies a distinct scanning section dedicated to a particular object while adhering to the temporal constraint defined by the drone's battery life. Since each target object might have different complexities and sizes, the quality of the scan, represented as the rewards, scales differently and uniquely. In this context, it should be noted that most traditional cluster-based OP variants cannot accommodate the scalability factor inherent in various problems. To overcome this limitation, we introduce a novel problem formulation.

This new variant, called the Clustered Orienteering Problem with Function-based Rewards (COPF), is a routing problem where potential customers are divided into groups denominated *clusters*. A reward (profit or utility) is associated with each cluster, and its amount collected is defined by a function related to the number of customers served in that cluster. The objective is to determine a single route traversing a predefined set of clusters/customers that is constrained by a certain travel budget (e.g., time or energy) and such that the total rewards collected is maximized (Fig. 1).

To the best of our knowledge, this is the first work to address this novel OP variant considering a *partial reward* scheme. We propose using an evolutionary algorithm, a well-known approach for solving routing problems, and investigate

different models and strategies that explore diverse avenues to achieve the solutions. Our primary goal is to obtain comprehensive insights into this specific category of problems rather than solely achieving the most optimized algorithm. In our experimental analysis, we examined various scenarios and utilized instances of the COP [2], in which specific functions were assigned to the clusters.

## II. RELATED WORK

Vehicle routing problems constitute a well-explored domain within the literature, encompassing a wide range of classes and variants that find applications in many real-world scenarios [3]. One of the most prominent problems in this domain is the Travelling Salesman Problem (TSP) [4], which aims to find the shortest path that visits a given set of cities precisely once before returning to the starting city. The TSP has a vast number of variants, including the Generalized Travelling Salesman Problem (GTSP) [5], also known as the Set TSP, which separates the cities into disjoint subsets and a solution requires visiting exactly one city of each subset, eliminating the need to visit all cities unless all subsets are singletons.

The category of routing problems with distinct rewards is also very prominent, distinguishing itself from the TSP by aiming to maximize a cumulative reward considering the clients served (*i.e.*, locations visited). A prime example of this domain is the Orienteering Problem [1], which in addition to maximizing the total reward, strives to determine the most efficient route, adhering to constraints such as limited time, energy, or distance traveled. The OP has been extensively studied and involves a diverse range of variants. For instance, the Team Orienteering Problem (TOP) extends the scope to include the utilization of multiple vehicles [6], [7]. Additionally, the Orienteering Problem with Time Windows (OPTW) [8] introduces specific time intervals within which each customer must be served. Other variants domains encompass fault-tolerance [9], environmental properties such as ocean currents [10] and threatening zones [11], correlated rewards [12], among others. Due to the inherent complexity of a NP-hard problem, which means it is at least as hard as any problem that can be verified in polynomial time, its different formulations are often tackled using heuristic methods, with genetic algorithms proving to be an effective alternative [13], [14].

More closely related to our problem, we have *cluster-based* variants, which introduce an abstraction to organize customers (locations) into groups. In this particular setup, a reward is associated with each cluster, unlike customers individually, as in the OP. Here, we are particularly interested in two specific variants: the Clustered Orienteering Problem and the Set Orienteering Problem. The SOP [15], [16] considers scenarios where a cluster reward is collected if *at least one customer* is visited. In contrast, in the COP [2], [17], the reward is obtained when *all customers* within a cluster are visited. Additionally, the Orienteering Problem with Functional Profits (OPFP) [18], although not cluster-based, presents a similar structure to our problem. The reward associated with each customer becomes variable, subject to change based on attributes such as their order within the route.

Our proposed formulation distinguishes itself from the aforementioned problems in a key aspect: it does not entail a binary, *all-or-nothing*, reward acquisition. Rather, the COPF adopts an intermediary approach, bridging the gap between the COP and the SOP, employing cluster-assigned functions to facilitate the acquisition of partial rewards. Furthermore, in contrast to the OPFP, our proposed variant emphasizes the relationship between customers within a cluster instead of individually assigned variable rewards for each customer.

In this paper, we propose the use of a multi-objective evolutionary algorithm for solving the Clustered Orienteering Problem with Function-based Rewards. We employ operators designed to accommodate the clustered nature of the problem, allowing for a more specialized selection of multiple clients at the same time and improving the collected reward.

## III. PROBLEM FORMULATION

Given an undirected graph  $G = (V, E)$ , where  $V = \{v_1, \dots, v_m\}$  is the set of vertices and  $E = \{e_{ij} \mid \forall (i, j) \in \{1, \dots, m\} \text{ and } i \neq j\}$  the set of edges. Vertices are clustered into arbitrary non-disjoint sets such that  $C = \{c_1, \dots, c_n\}$ , where each vertex  $v_i$  belongs to *at least one* cluster, and each cluster  $c_k$  has an associated maximum reward  $r_k$ . The amount of reward collected when a cluster is visited is defined by a function  $f_k(|c_k^v|) \rightarrow \mathbb{R}^+ \leq r_k$ , where  $c_k^v \subseteq c_k$  is the subset of vertices visited within a cluster.

The starting and ending clusters are labeled  $c_1$  and  $c_n$ , respectively, and are not associated with any reward, *i.e.*,  $f_1(\cdot) = f_n(\cdot) = 0$ . We can define  $c_1 = c_n$  in order to obtain a circular route. We assume that each edge  $e_{ij} \in E$  has a known traversal cost  $\tau_{ij}$ , based on the distance or the time required to traverse it. Moreover, if a customer  $v_j$  belongs to more than one cluster, visiting such a customer increases  $|c_k^v|$  by one,  $\forall c_k \in C$  such that  $v_j \in c_k$ , affecting the collected reward.

A COPF solution can be described as a permutation of the vertex indexes  $\Pi = (\pi_1, \dots, \pi_p)$ ,  $1 \leq \pi_p \leq m$ . The permutation represents the visiting order of each vertex, forming a tour which starts in  $v_{\pi_1}$  and ends in  $v_{\pi_p}$ .

**Problem 1** (Clustered Orienteering Problem with Function-based Rewards). *Given a collection of clusters  $C$ , each cluster  $c_k$  with an associated maximum reward  $r_k$  and acquisition function  $f_k(\cdot)$ . The objective is to determine a tour, starting in a vertex  $v_s \in c_1$  and ending in a vertex  $v_e \in c_n$ , that maximizes the total collected reward  $R$ , while not exceeding a predefined travel budget  $T_{max}$ . Formally:*

$$\max_{\Pi} R = \sum_{k=1}^n f_k(|c_k^v|) \quad (1)$$

s.t.

$$v_{\pi_1} \in c_1, v_{\pi_p} \in c_n \quad (2a)$$

$$L = \sum_{i=2}^p \tau_{\pi_{i-1}, \pi_i} \leq T_{max} \quad (2b)$$

$$c_k^v = \{\pi_j \in \Pi \mid v_{\pi_j} \in c_k\} \quad (2c)$$

#### IV. METHODOLOGY

A Genetic Algorithm (GA) is a metaheuristic optimization technique that mimics natural selection and genetic mechanisms to iteratively evolve potential solutions [19]. The proposed method for addressing the COPF follows the general scheme of traditional GAs applied to routing problems as shown in Algorithm 1.

---

**Algorithm 1** Genetic Algorithm
 

---

- 1: Generate initial population
  - 2: Evaluate fitness of current individuals
  - 3: **while** stopping condition is not satisfied **do**
  - 4:     Iteratively select pairs of parent individuals
  - 5:     Generate new individuals by crossover/mutation
  - 6:     Evaluate fitness of new individuals
  - 7:     Replace current individuals with new ones
  - 8: **end while**
  - 9: **return** Return individual with best fitness
- 

In summary, the approach involves a procedure for generating random initial solutions, a designated selection mechanism, and an offspring generation process based on a point-based crossover and a randomized mutation operator. These new individuals contribute to the formation of the succeeding generation, fostering the perpetuation of favorable traits within the population across subsequent iterations.

##### A. Individual and Cluster Representation

An individual (solution) is represented through a vector  $\Sigma = \{a_1, a_2, \dots, a_m\}$ , with each value  $a_i$  representing a  $v_j$  customer of the instance. Customers belonging to the same cluster are positioned sequentially in the vector; therefore, if  $v_j, v_h, v_l$  belong to the same cluster, they are represented by indexes  $a_i, a_{i+1}, a_{i+2}$ . For the permutation aspect, each  $a_i$  is then assigned a value in the domain  $[-1, 0) \cup (0, 1]$ , which identifies if a customer is visited in the final decoded solution through the number sign, and the visiting order through its magnitude. Negative values indicate that the corresponding customer is not visited in the individual's route, and customers can be included or excluded from the route by flipping the number sign, and therefore 0 is omitted from the domain.

The individual can then be decoded into the visiting order by discarding negative values and indirectly sorting the list in ascending order, hence the number magnitude, finally obtaining  $\Pi$  by adding the starting/ending customer at the beginning and ending of the route, respectively. This representation assumes that  $|c_1| = |c_n| = 1$ , meaning there is only one starting/ending customer, which is the case in the vast majority of testing instances, but can be expanded into the general case by enforcing values to customers in  $|c_1|$  and  $|c_n|$ .

For the cluster representation, only a list of the corresponding customer indexes and the assigned function of the cluster are required since they are only used for calculating the reward of a route.

##### B. Initialization

To create an initial population of  $p$  individuals, we divide the initialization operators into first defining the omission probability for each customer when building random solutions and then building the solutions using a truncated normal distribution. Using both steps guarantees a more well-behaved initialization process while still maintaining randomness.

1) *Defining Customer Omission Probability*: Firstly, we establish an omission probability  $\epsilon$ , representing the likelihood of a customer  $v_i$  not being included in an initial route. The determination of this variable is analogous to the methodology presented in [14], wherein  $maxloop$  individuals are generated, each with a random route, composed of a random length and permutation of customers. The number of solutions that respect the  $T_{max}$  constraint,  $count$ , is then calculated to define our omission probability using  $\epsilon = 1 - \frac{count}{maxloop}$ .

2) *Initializing Random Individuals*: Customers within the same cluster tend to be spatially proximate, forming the foundation for our initial path optimization strategy. To define such a probabilistic method, we adopt a truncated normal distribution to assign a  $(0, 1]$  value to each  $a_i$ . Consequently, during the generation of each individual, every cluster is endowed with the aforementioned distribution with a random mean in  $(0, 1]$  and a fixed standard deviation  $\sigma$ . Subsequently, each customer associated with the cluster is assigned a value based on the established truncated normal distribution, establishing our partially established random solution  $\Sigma'$ .

Since all customers are considered visited in  $\Sigma'$ , we finish our initialization by applying our previously calculated omission probability,  $\forall a_i \in \Sigma$ : if  $\xi \leq \epsilon$  then  $a_i \leftarrow -a_i$ , with  $\xi$  being a random number in the range  $[0, 1]$ , effectively removing certain customers from the individual's corresponding route according to  $\epsilon$ , forming our final initial random individual  $\Sigma$ . Using this strategy, the randomly generated individuals are more inclined to conform to the constraint  $T_{max}$  and take advantage of the cluster nature of our problem.

##### C. Fitness Evaluation

Multiple factors and implementations were carefully considered to comprehensively evaluate individual fitness within the proposed GA. Initially, it is appropriate to underscore that the COPF represents an optimization problem focused solely in maximizing the total collected reward (Eq. 1). However, the inclusion of the total distance traveled in the fitness assessment is also interesting, as the total reward is intricately linked to the specific path taken accordingly to the budget constraint, being fundamental for high-quality solutions.

In light of these considerations, individual fitness will be determined based upon two pivotal parameters, previously established in the problem evaluation:  $R$ , the total collected reward, and  $L$ , the traveled distance. Prior to the evaluation of an individual, the algorithm ascertains its validity, ensuring that the total distance adheres to  $T_{max}$ . In cases where this condition is not met, the algorithm systematically removes a random customer from the individual's path until compliance with the constraint is achieved.

A straightforward and intuitive assessment method involves utilizing  $R$  alone, a practice employed in previous literature for addressing the classical OP [14]. The algorithm optimizes the path *indirectly* as achieving higher scores requires more efficient routes for visiting multiple customers. Nevertheless, for a more informed evolution process, it is paramount for the algorithm to possess a more *direct* focus on optimizing the distance, thereby enhancing its overall awareness during the evolutionary progression.

In this sense, an evaluation method presented in the literature that includes the path length is  $R^3/L$  [13]. Although it is still possible to continue to increase the reward  $R$  by a larger exponent, and therefore increasing its priority in the evaluation, the formula only indicates the overall *efficiency* (reward per distance) of the path: a high value does not necessarily indicate a longer or a shorter path. Instead, it only indicates a good proportion between them, which does not directly translate into the highest score, therefore contradicting our problem.

As an alternative to solely evaluating a ratio between  $R$  and  $L$ , we modify the evaluation methodology for the genetic algorithm, transforming it into a multi-objective problem throughout the evolutionary process. This adjustment aims to concurrently maximize both  $R$  and  $R/L$ , which also discards the need to assign an exponent to  $R$  similar to the aforementioned  $R^3/L$ , since we are already also analyzing  $R$  individually. Consequently, the genetic algorithm considers the raw total score while also scrutinizing the efficiency of the path. This strategic shift results in a more discerning evolutionary trajectory, advancing when attaining higher total scores and when establishing a more optimized path for an existing score. Upon the conclusion of the evolutionary process, the selection of the individual with the highest  $R$  within the population persists.

#### D. Crossover and Mutation

Concerning the procedures of crossover and mutation, specific measures are unnecessary, given that our individual representation maintains the order based on indexes and includes a mechanism to track whether a customer has been visited.

In the case of the crossover operator, a two-point crossover has been designated. Herein, when presented with two parents represented by arrays of size  $m$ , two integers  $i, j \in [0, m)$ ,  $i < j$  are selected. Subsequently, two new offspring are generated by duplicating both parents and exchanging the values between indices  $i$  and  $j$ . Since customers in the same cluster have sequential indexes in  $\Sigma$ , they also tend to be grouped.

Regarding mutation, the operator starts by selecting a random cluster  $c_r$  from the pool, iterating through every value  $a_i$  whose corresponding customer  $v_j \in c_r$  and applying two probabilistic procedures. Initially, the customer toggle operator, which assigns  $a_i \leftarrow -a_i$ , is applied with probability  $\mu$ . Subsequently, if  $a_i$  was not affected by the previous operator,  $\nu$  is used to define the chance of choosing another random value  $a_{i'}$  in the cluster and swapping both values. These two procedures jointly form a mutation operator that is particularly well-suited to our clustered context, as they take advantage

of the underlying cluster structure to balance exploration and exploitation, enhancing the efficiency and relevance of the generated solutions.

#### E. Selection and Evolution

Finally, for the selection operator, due to the adoption of a multi-objective evaluation process, we utilize the Non-Dominated Sorting Genetic Algorithm III (NSGA-III) [20], which selects individuals based on Pareto curves.

The general procedure and evolutionary process then follows Algorithm 1 with the aforementioned operators, applying crossover and mutation based on CXPB and MUTPB probabilities. Both the prior population and offspring are subjected to the selection operator when composing the population for the ensuing generation. The total score of the best individual is also checked every  $\lambda$  generations, ending the algorithm prematurely if the score is unchanged.

### V. EXPERIMENTS

We implemented the simulation framework with Python 3.7 and uses the DEAP [21] library. Table I summarizes the experimental parameters.

TABLE I  
ALGORITHM PARAMETERS

Instance	Value
Population size	400
Number of generations	1000
Selection method	NSGA-III
CXPB	0.7
Crossover operator	Two-point crossover
MUTPB (individual)	0.4
$\sigma$	0.05
$\mu$	0.15
$\nu$	0.15
$\lambda$	100

The benchmarked instances were based on a modified version of the SOP dataset [16], commonly used for comparing SOP solvers, which was originally adapted from the Generalized Traveling Salesman Problem dataset [22], in which the disjoint subsets are interpreted as clusters. Initially, the first customer from each instance is removed from its cluster and added into new starting/ending clusters  $c_1 = c_n$ . Two types of cluster profit  $p_g = \{g_1, g_2\}$  were taken into account, where  $g_1$  defines cluster profit as the size of the cluster  $|c_k|$ , and  $g_2$  uses the pseudo-random profit attribution of each customer  $v_j$  as  $1 + (7141j) \bmod 100$ , thus defining each cluster profit as the sum of each of its customers' profits. The budget  $T_{max}$  is generated according to the  $\omega$  ratio of the GTSP, the best known costs of GTSP solutions taken from [22]: a ratio of  $\omega = 0.8$ , for example, assigns  $T_{max}$  as 80% of the total cost of the best known GTSP solution.

Considering the new parameters required by the COPF compared to SOP/COP, further modifications were made to the dataset before testing by assigning functions to each cluster, which were determined in accordance with Table II, given a cluster size  $|c|$ , its total reward  $r$  and the number of customers  $n$  visited within the cluster. All functions were defined such



that  $f(|c|) = r$ , except for the quadratic function, which follows  $f(\frac{|c|}{2}) = r$ .

TABLE II  
DEFINED FUNCTIONS FOR EXPERIMENTATION

Linear	Exponential	Quadratic	Logarithmic
$\frac{n \cdot r}{ c }$	$\frac{n}{r_i  c }$	$\frac{-4r}{ c ^2} n^2 + \frac{4r}{ c } n$	$r \log_{ c } n$

All experiments were carried out on a PC equipped with an Intel Core i7-7770 clocked at 3.6GHz and 16GB of RAM.

#### A. Illustrative Example

Initially, to enhance visualization and comprehension of the results, we begin by providing an illustrative example. We consider two traditional benchmark instances and assign random functions to the clusters. Fig. 2 shows cluster/point placement, attributed functions and the best path determined by the algorithm out of 15 runs on the selected instances.

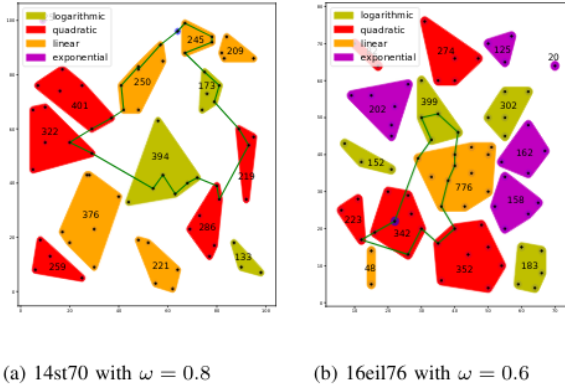


Fig. 2. Example results on benchmark instances from the “All Random” set. Other sets, containing the same customers but with all clusters having the same reward function, were also established. Each subplot denotes the final resulting path obtained by the algorithm in the given instance.

The results provide a clear illustration of the strategic navigation adopted in both scenarios. Specifically, they highlight the principle of diminishing returns in fully visiting logarithmic clusters, as well as the deliberate approach of visiting no more than half of the customers within each quadratic cluster. Additionally, no exponential clusters were visited, a phenomenon that will be discussed in subsequent sections.

#### B. Comparison with SOP and COP Formulations

For benchmarking the algorithm according to the established instances, results will be compared with values from the VNS-SOP paper [16] and the COP formulation with solution obtained using the COPS Tabu search algorithm [23], both of which, as mentioned previously, our problem lies between by taking an intermediary approach to the *all or nothing* reward acquisition. The adapted instances included assigning every cluster to a single function type (e.g., assigning all clusters a linear function), or by randomly assigning each cluster to a

function type. For consistency, clusters from random instances with the same base name (e.g., 11berlin52) have the same function types, regardless of  $p_g$  and  $\omega$ .

Due to the probabilistic nature of genetic algorithms, for our analysis, each instance was run a total of 15 times, with  $P$  representing the rounded average score obtained by the algorithm, along with the average runtime  $T$  in seconds.

Results in Table III show that scores obtained by our COPF methodology, despite being always lower than the SOP as expected, are not always higher than the COP, especially in larger instances, despite the significantly lower runtimes. This is due to the fact that fully visiting a certain amount of clusters can be just as optimized as partially visiting other clusters with their respective functions. Furthermore, certain cluster functions, such as the exponential, can be hard to incorporate into solutions due to the incompatibility of the greedy aspect of genetic algorithms and its slow rate of change. As a result, the algorithm usually prioritizes fast-growing functions, such as the quadratic.

Overall, our approach to the COPF has demonstrated the ability to achieve commendable total scores within reasonable computational times, particularly when compared to the COP approach on medium to large instances.

#### C. Performance Analysis

Regarding the methodology presented for solving the COPF, a multi-objective and non-sequential approach was taken as the basis solution, i.e., a path with no precedence restrictions. However, we could also define a sequential route which the clusters are visited in order, meaning that all visited customers belonging to a single cluster are to be visited before proceeding to the next cluster. In this context, the solution can be modeled as single-objective or sequential while still using equal or slightly altered representations, crossover, and mutation methods. For single-objective, the selection operator was replaced with tournament selection between two individuals. For sequential visiting, the  $[-1, 0) \cup (0, 1]$  representation is also used for ordering cluster visits, and the aforementioned crossover/mutation operators are used.

Non-sequential single-objective and sequential single-objective solution methods were tested along our previously defined non-sequential multi-objective method by running a small, medium and large  $g_2$  type of instance on 11berlin52, 115rat575, and 201pr1002, respectively, while varying the  $\omega$  value, thus gradually increasing  $T_{max}$ . Each instance and  $\omega$  combination were run 100 times, and the average scores were plotted. Randomized instances, which have the same customers as the original instance, but with each customer assigned to a random cluster, which tends to generate incoherent spatially distant clusters, were also evaluated.

Analyzing Figs. 3 and 4, all algorithms presented consistent results, given the confidence interval used. However, the original non-sequential multi-objective seems to perform above other implementations, losing only on 11berlin52, the smallest instance presented, which tends to benefit a sequential order due to the presence of small, distanced clusters. However, the sequential approach struggles with larger and randomized

TABLE III  
COMPARISON WITH EXISTING METHODS ON SELECTED GTSP DATASET INSTANCES

Instance	$p_g$	$\omega$	$T_{max}$	VNS-SOP [16]		COP-Tabu [23]		All Random		All Linear		All Quadratic		All Exponential		All Logarithmic	
				P	T	P	T	P	T	P	T	P	T	P	T	P	T
11berlin52	$g_1$	0.4	1616	37	0.11	20	17.0	10	22.7	17	22.6	32	23.3	10	22.5	21	31.2
11berlin52	$g_2$	0.4	1616	1829	0.11	382	10.5	395	19.8	804	24	1589	22.9	298	21.5	1018	30.3
11berlin52	$g_1$	0.6	2424	43	0.16	24	28.7	14	26.8	21	21.2	37	21.6	14	27.8	25	36.1
11berlin52	$g_2$	0.6	2424	2190	0.15	615	17.2	583	22.7	1066	25.2	1830	23.3	508	24.9	1245	36.1
11berlin52	$g_1$	0.8	3232	47	0.19	30	41.9	17	26.3	25	24.2	40	24.6	17	29.4	29	37.9
11berlin52	$g_2$	0.8	3232	2384	0.19	1342	72.0	740	27.8	1261	23.5	2001	22.9	691	30.1	1458	39.4
14st70	$g_1$	0.4	126	33	0.14	13	23.9	14	32.6	13	27.5	23	26.4	13	30.1	15	32.6
14st70	$g_2$	0.4	126	1672	0.15	627	25.0	710	29.6	630	28.2	1111	26.0	552	25.5	669	31.2
14st70	$g_1$	0.8	252	65	0.31	27	136.3	32	41.8	22	32.7	46	45.3	22	41.5	26	54.6
14st70	$g_2$	0.8	252	3355	0.33	1278	100.5	1707	49.0	1110	33.9	2325	44.6	873	32.9	1181	54.3
16eil76	$g_1$	0.4	83	40	0.19	9	10.8	18	21.2	10	24.2	27	34.5	10	27.8	14	36.1
16eil76	$g_2$	0.4	83	2223	0.20	565	9.6	946	26.2	603	26.6	1536	32.6	406	27.2	831	29.5
16eil76	$g_1$	0.6	125	59	0.31	14	46.7	23	39.0	16	33.8	36	32.6	14	36.0	20	43.6
16eil76	$g_2$	0.6	125	3119	0.32	999	52.1	1328	36.9	1027	31.4	2135	35.3	659	29.8	1284	43.4
20kroA100	$g_2$	1.0	9711	4868	0.8	2143	545.4	2199	72.9	1821	52.4	3003	51.8	1330	48.6	1897	88.5
21eil101	$g_2$	1.0	249	4993	1.1	1786	211.5	2193	68.5	2050	55.9	3734	65.6	1114	51.1	2468	92.5
40krob200	$g_2$	1.0	13111	9990	5.8	3545	3437.4	3420	139.6	3112	142.8	5204	145.2	1943	88.7	3291	180.5
46pr226	$g_2$	1.0	64007	11,368	4.6	8902	33,250.4	3611	121.5	4369	154.8	7002	162.1	2297	120.1	4841	161.8
89pcb442	$g_2$	1.0	21657	22,300	36.2	7254	20,655.2	6486	243.5	5888	184.5	9865	246.4	3248	224.6	5911	314.5
115rat575	$g_2$	1.0	2388	28,361	75.6	7134	22,392.5	7370	290.2	5608	288.1	10,523	245.0	3592	255.9	5967	391.9
201pr1002	$g_2$	1.0	11,4311	50,172	534.7	15,184	128,900.9	11,796	435.0	9563	445.9	16,850	400.1	4335	415.4	10,110	564.0
212u1060	$g_2$	1.0	10,6007	53,437	641.8	16,940	339,953.2	11,475	426.8	9540	398.0	16,789	483.6	4819	388.5	10,585	563.0

instances due to the nature of both types of instances. Furthermore, the total score increases almost linearly according to  $T_{max}$  in base instances, indicating the general consistency of the algorithm:  $g_2$  clusters of an instance tend to have similar maximum scores, indicating an also linear increase of clusters visited, or sections of clusters visited.

## VI. CONCLUSION AND FUTURE WORK

In various routing scenarios, the acquisition of valuable information from points of interest within an environment can be contingent upon distinct characteristics. Within this context, the Orienteering Problem stands as a foundational model for diverse formulations. It conceptualizes information gain as a form of reward to be maximized, while simultaneously accommodating vehicle's travel constraints.

In this work, we introduce a novel variant called the Clustered Orienteering Problem with Function-based Rewards. In this formulation, customers are organized into clusters, and a variable reward is obtained based on a designated function that considers the number of customers served. This innovative partial reward scheme provides a more refined representation of a broad spectrum of scenarios, including applications in environmental monitoring.

We propose the use of a multi-objective Genetic Algorithm that aims to directly improve not only the score, but also the total distance travelled by analyzing route efficiency, alongside genetic operators which account for the clustered nature of the problem. Experiments show that consistent results can be achieved in reasonable runtimes when compared to results from other problems, such as the Set Orienteering Problem and the Clustered Orienteering Problem. However, the greedy aspect of the genetic algorithm can make evolution difficult for functions with multiple local maxima and minima, or low rates of change.

In future work, we aim to delve deeper into additional factors, including vehicle's with specific motion constraints (e.g., fixed-wing UAVs), environmental conditions (e.g., the presence of currents), and the coordination of multi-robot

assemblies. We also intend to evaluate more dynamic scenarios by incorporating time-varying reward functions.

## REFERENCES

- [1] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics (NRL)*, vol. 34, no. 3, pp. 307–318, 1987. [https://doi.org/10.1002/1520-6750\(198706\)34:3<307::AID-NAV3220340302>3.0.CO;2-D](https://doi.org/10.1002/1520-6750(198706)34:3<307::AID-NAV3220340302>3.0.CO;2-D).
- [2] E. Angelelli, C. Archetti, and M. Vindigni, "The clustered orienteering problem," *European Journal of Operational Research*, vol. 238, no. 2, pp. 404–414, 2014. <https://doi.org/10.1016/j.ejor.2014.04.006>.
- [3] A. Mor and M. G. Speranza, "Vehicle routing problems over time: a survey," *4OR*, vol. 18, pp. 129–149, Jun 2020. <https://doi.org/10.1007/s10288-020-00433-2>.
- [4] K. L. Hoffman, M. Padberg, and G. Rinaldi, *Traveling Salesman Problem*, pp. 1573–1578. Boston, MA: Springer US, 2013. [https://doi.org/10.1007/978-1-4419-1153-7\\_1068](https://doi.org/10.1007/978-1-4419-1153-7_1068).
- [5] C. Archetti, F. Carrabs, and R. Cerulli, "The set orienteering problem," *European Journal of Operational Research*, vol. 267, no. 1, pp. 264–272, 2018. <https://doi.org/10.1016/j.ejor.2017.11.009>.
- [6] I.-M. Chao, B. L. Golden, and E. A. Wasil, "The team orienteering problem," *European Journal of Operational Research*, vol. 88, pp. 464–474, 2 1996. [https://doi.org/10.1016/0377-2217\(94\)00289-4](https://doi.org/10.1016/0377-2217(94)00289-4).
- [7] A. Mansfield, S. Manjanna, D. G. Macharet, and M. A. Hsieh, "Multi-robot scheduling for environmental monitoring as a team orienteering problem," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6398–6404, IEEE, 2021. <https://doi.org/10.1109/IROS51168.2021.9636854>.
- [8] M. G. Kantor and M. B. Rosenwein, "The Orienteering Problem with Time Windows," *Journal of the Operational Research Society*, vol. 43, no. 6, pp. 629–635, 1992. <https://doi.org/10.1057/jors.1992.88>.
- [9] R. F. dos Santos, E. R. Nascimento, and D. G. Macharet, "Anytime Fault-tolerant Adaptive Routing for Multi-Robot Teams," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7936–7942, 2021. <https://doi.org/10.1109/ICRA48506.2021.9561944>.
- [10] A. Mansfield, D. G. Macharet, and M. A. Hsieh, "Energy-efficient Orienteering Problem in the Presence of Ocean Currents," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10081–10087, 2022. <https://doi.org/10.1109/IROS47612.2022.9981818>.
- [11] D. G. Macharet, A. Alves Neto, and D. Shishika, "Minimal Exposure Dubins Orienteering Problem," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2280–2287, 2021. <https://doi.org/10.1109/LRA.2021.3061004>.
- [12] J. Yu, M. Schwager, and D. Rus, "Correlated Orienteering Problem and its Application to Persistent Monitoring Tasks," *IEEE Transactions on Robotics*, vol. 32, pp. 1106–1118, 10 2016. <https://doi.org/10.1109/TRO.2016.2593450>.

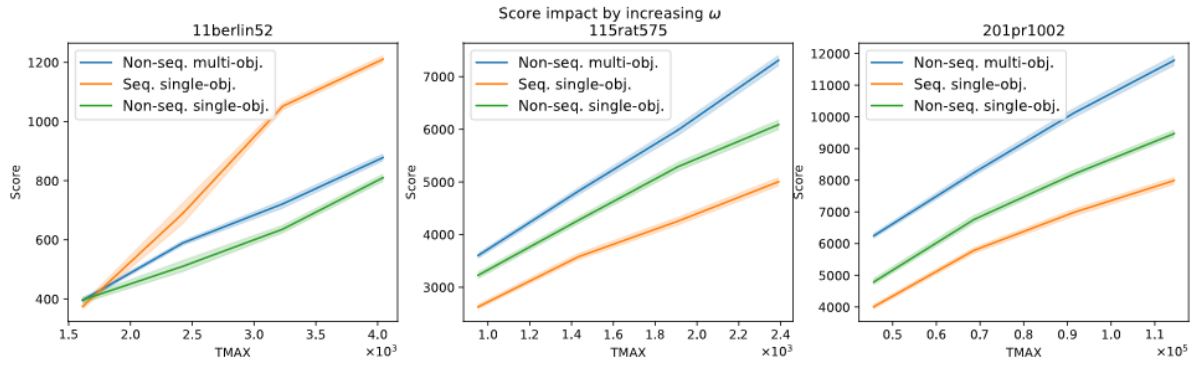


Fig. 3. Results on benchmark instances varying  $\omega$  and  $T_{max}$ , considering a 95% confidence interval. Each subplot describes the relationship between the TMAX and Score values on the 11berlin52, 115rat575 and 201pr1002 instances respectively, which are of increasing size.

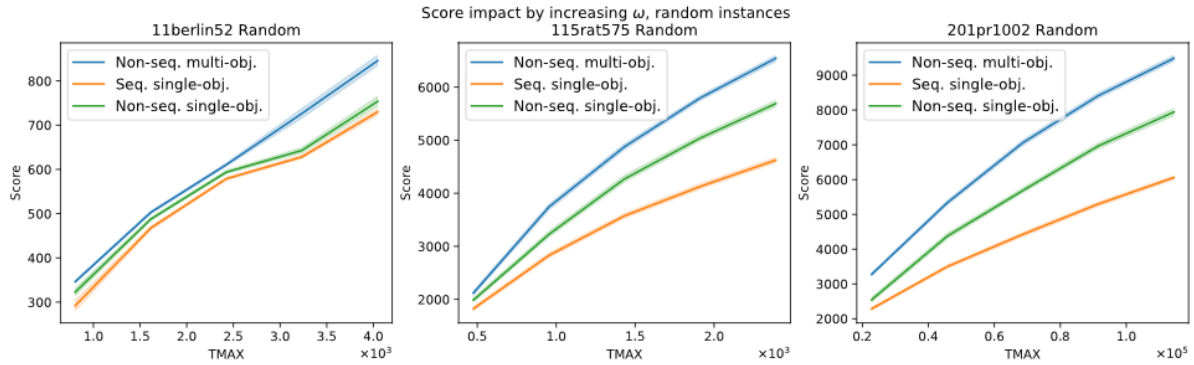
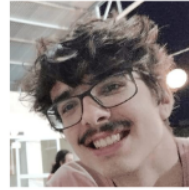


Fig. 4. Results on random instances varying  $\omega$  and  $T_{max}$ , considering a 95% confidence interval. Each subplot describes the relationship between the TMAX and Score values on the 11berlin52, 115rat575 and 201pr1002 instances with random customer to cluster association, which are of increasing size.

- [13] J. Karbowska-Chilinska and P. Zabielski, *Genetic Algorithm Solving the Orienteering Problem with Time Windows*, vol. 240, pp. 09 2013. [https://doi.org/10.1007/978-3-319-01857-7\\_59](https://doi.org/10.1007/978-3-319-01857-7_59).
- [14] M. Tasgetiren and A. Smith, "A genetic algorithm for the orienteering problem," in *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, vol. 2, pp. 910–915 vol.2, 2000. <https://doi.org/10.1109/CEC.2000.870739>.
- [15] C. Archetti, F. Carrabs, and R. Cerulli, "The set orienteering problem," *European Journal of Operational Research*, vol. 267, no. 1, pp. 264–272, 2018. <https://doi.org/10.1016/j.ejor.2017.11.009>.
- [16] R. Pěnička, J. Faigl, and M. Saska, "Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants," *European Journal of Operational Research*, vol. 276, no. 3, pp. 816–825, 2019. <https://doi.org/10.1016/j.ejor.2019.01.047>.
- [17] Q. Wu, M. He, J.-K. Hao, and Y. Lu, "An effective hybrid evolutionary algorithm for the clustered orienteering problem," *European Journal of Operational Research*, 2023. <https://doi.org/10.1016/j.ejor.2023.08.006>.
- [18] K. D. Mukhina, A. A. Visheratin, and D. Nasonov, "Orienteering problem with functional profits for multi-source dynamic path construction," *PLOS ONE*, vol. 14, pp. 1–15, 04 2019. <https://doi.org/10.1371/journal.pone.0213777>.
- [19] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998. <https://doi.org/10.7551/mitpress/3927.001.0001>.
- [20] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, pp. 577–601, 08 2014. <https://doi.org/10.1109/TEVC.2013.2281535>.
- [21] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizéau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012. <https://api.semanticscholar.org/CorpusID:15629107>.
- [22] M. Fischetti, J. J. Salazar González, and P. Toth, "A branch-and-cut algorithm for the symmetric generalized traveling salesman problem," *Operations Research*, vol. 45, pp. 378–394, 06 1997. <https://doi.org/10.1287/opre.45.3.378>.
- [23] L. E. Almeida and D. G. Macharet, "Clustered Orienteering Problem with Subgroups," *CoRR*, vol. abs/2312.16154, 2023. <https://doi.org/10.48550/arXiv.2312.16154>.



**Vinícius L. C. Faria** is a Computer Science Undergraduate Student at the Universidade Federal de Minas Gerais (UFMG), previously an undergraduate of Computer Engineering at the Universidade Federal de São Carlos (UFSCar). He is with the Computer Vision and Robotics Laboratory (VeRLab), currently studying various optimization routing problems utilizing heuristic methods.



**Douglas G. Macharet** is an Associate Professor in the Dept. of Computer Science at the Universidade Federal de Minas Gerais (UFMG). He holds an M.Sc. and Ph.D. in Computer Science from UFMG, obtained in 2009 and 2013, respectively. He is with the Computer Vision and Robotics Laboratory (VeRLab), specializing in mobile robotics, with a focus on motion planning, navigation, multi-robot systems, swarm robotics, and human-robot interaction.