logistics

MDPI

Article

A Compact Model for the Clustered Orienteering Problem

Roberto Montemanni [1,*] and Derek H. Smith [2]

[1] Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Via Amendola 2, 42122 Reggio Emilia, Italy
[2] Computing and Mathematics, University of South Wales, Pontypridd CF37 1DL, UK; derek.smith@southwales.ac.uk
* Correspondence: roberto.montemanni@unimore.it

Abstract: *Background:* The Clustered Orienteering Problem is an optimization problem faced in last-mile logistics. The aim is, given an available time window, to visit vertices and to collect as much profit as possible in the given time. The vertices to visit have to be selected among a set of service requests. In particular, the vertices belong to clusters, the profits are associated with clusters, and the price relative to a cluster is collected only if all the vertices of a cluster are visited. Any solving methods providing better solutions also imply a new step towards sustainable logistics since companies can rely on more efficient delivery patterns, which, in turn, are associated with an improved urban environment with benefits both to the population and the administration thanks to an optimized and controlled last-mile delivery flow. *Methods:* In this paper, we propose a constraint programming model for the problem, and we empirically evaluate the potential of the new model by solving it with out-of-the-box software. *Results:* The results indicate that, when compared to the exact methods currently available in the literature, the new approach proposed stands out. Moreover, when comparing the quality of the heuristic solutions retrieved by the new model with those found by tailored methods, a good performance can be observed. In more detail, many new best-known upper bounds for the cost of the optimal solutions are reported, and several instances are solved to optimality for the first time. *Conclusions:* The paper provides a new practical and easy-to-implement tool to effectively deal with an optimization problem commonly faced in last-mile logistics.

Keywords: clustered orienteering problem; constraint programming; exact solutions; heuristic solutions

Citation: Montemanni, R.; Smith, D.H. A Compact Model for the Clustered Orienteering Problem. *Logistics* **2024**, *8*, 48. https://doi.org/10.3390/logistics8020048

Academic Editor: Robert Handfield

Received: 25 March 2024
Revised: 27 April 2024
Accepted: 2 May 2024
Published: 6 May 2024

1. Introduction

The Orienteering Problem (OP) was introduced in [1,2] in the 1980s. In an interpretation of the problem from the viewpoint of logistics, there is a single vehicle, leaving from and returning to a depot, which serves a set of customers, each one associated with a spacial location and profit, with such a profit collected upon visiting the location. The travel times among the locations are known, deterministic, and given. However, not all the customers can typically be serviced, since the vehicle mission cannot be longer than a given maximum time. The aim of the problem is to maximize the total profit collected by the vehicle in the available time. The problem has attracted a lot of attention due to its practical implications, and many variations of the original problem have been introduced over the years. We refer the interested reader to [3–6] for exhaustive reviews of the scientific literature on these problems. Optimization approaches for last-mile logistics, in general, have become more and more popular in recent years due to the dramatic increase in e-commerce [7], which automatically generates a great need for effective operational solutions [8–12].

A generalization of the OP, called the Clustered Orienteering Problem (COP), was originally proposed in [13], and is the topic of the present study. There is a set of customers that are grouped into clusters. Associated with each cluster there is a profit, which is collected once all customers of the cluster are visited. A classic application of the COP is in last-mile distribution in the retail sector: sometimes contracts allow collection of the profit only if all the warehouses (or shops) of a chain (cluster) are visited. Another typical application is again in last-mile logistics, which is about the collection of goods from the customers in order to aggregate them for further shipping. In this case, the profit is collected only if all the goods planned to be aggregated together (typically to form a container) are collected, otherwise the shipment cannot happen and there is a delay. The growth of e-commerce associated with globalization gives a measure of the impact on the pollution of the activities associated with these logistics problems. It is safe to state that any improvement on their solution is a clear step towards a more sustainable society.

The COP should not be confused with the Set Orienteering Problem [14,15], which shares the same input information, but with a different objective, namely, the profit is collected once a single vertex of a cluster is visited. This problem was introduced to model a different family of last-mile logistics applications.

The first algorithmic approaches to solve the COP were proposed in [13], where exact solving approaches, based on a mixed-integer linear program (MILP) and on a branch-and-cut method were proposed. In the same paper, three different variations of a tabu search heuristic were also proposed to deal with those instances that could not be effectively be treated by the exact solvers. More recently, other heuristic approaches were discussed, respectively, in [16], where a hybrid heuristic (HH) method was proposed, and in [17], where the authors propose a hybrid evolutionary algorithm (HEA). A cutting plane method was introduced as an exact method in [18] for the Clustered Team Orienteering Problem, a version of the COP with multiple vehicles. In the same paper, a generalization of the heuristic algorithm HH is also discussed. A summary of the contributions of these works can be found in Table 1.

Table 1. Summary of the contributions of the publications dealing with the solving methods for the Clustered Orienteering Problem and its generalizations.

Paper	Exact Methods	Heuristic Methods	Multiple Trucks
Angelelli et al. [13]	Yes	Yes	No
Yahiaoui et al. [16]	No	Yes	No
Yahiaoui et al. [18]	Yes	Yes	Yes
Wu et al. [17]	No	Yes	No
This paper	Yes	No	No

In the present work—which is along the line of recent successful applications of constraint programming (CP) to other last-mile logistic problems [19,20]—we introduce a novel exact solving method for the COP, which is able to take advantage of the technologies available in modern solvers and modern hardware. The results achieved indicate that the new approach is able to provide state-of-the-art results, notwithstanding an implementation complexity that is substantially lower than that of the exact methods previously available in the literature, with improved upper bounds for the optimal cost of several instances, and many instances closed here for the first time.

The real-world implications of our results in terms of last-mile logistics derive from the availability of the new tool we make available to practitioners, which allows them to have a better understanding and higher-quality solutions while carrying out operational planning. This, coupled with a better strategic organization, can lead to a more optimized and harmonized logistics, leading to direct social benefits for citizens (higher quality service), providers (less costs), and the cities themselves (less traffic in the urban environment).

2. Problem Description

Let $G = (V, A)$ be a directed graph, where $V = \{0\} \cup C$ is the set of vertices of the graph and A is the set of arcs. The depot (starting and ending point of the route) is vertex 0, while C is the set of locations of the customers. A set of $m+1$ clusters $C_0, C_1, \ldots, C_m$ is given, such that $C_i \subseteq C$ $\forall i \in \{0, 1, \ldots, m\}$ and they cover C: $(\bigcup_{i=0}^{m} C_i = C)$. Notice that clusters can overlap. Each vertex $j \in C$ is part of at least one cluster, but it can be part of several. A profit p_i is associated with each cluster C_i, and such a profit is collected only if all the vertices $j \in C_i$ are visited. Cluster C_0 contains only the depot 0 and has a null profit. A travel time c_{jk} is associated with each arc $(j, k) \in A$, representing travel times, and a maximum time T_{max} is given. The Clustered Orienteering Problem (COP) consists of finding a route on vertices V with a total travel time not longer than T_{max} that maximizes the total profit collected. In the remainder of the paper, we assume—consistently with the previous literature—that the travel times satisfy triangle inequalities. An example with a COP instance and a relative solution is provided in Figure 1.

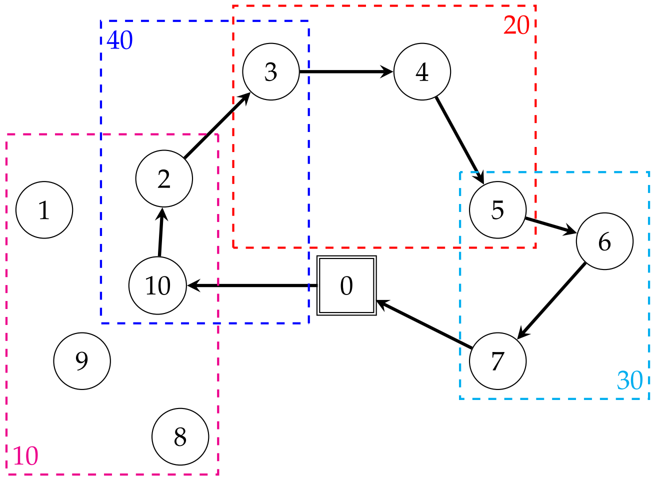

Figure 1. Example of a (simplified) COP instance. The squared node 0 is the depot, while the other vertices are customers. Clusters are represented as coloured rectangles, with the associated profit depicted in a corner. Travel times are omitted for the sake of simplicity, together with the threshold T_{max}. A tour with a total profit of 90 is drawn in black.

3. A Model Based on Constraint Programming

The COP can be described through the following constraint programming model, designed according to the syntax of the Google OR-Tools [21] CP-SAT solver [22]. The idea behind the model is to define for each cluster a representative vertex, which will be used to easily identify the visited clusters and to define constraints effectively. Let r_i be the representative of cluster C_i. Such a vertex is either selected as one of the vertices belonging solely to cluster C_i or, if such a vertex does not exist, by adding the artificial vertex a_i to the cluster C_i (and to the vertex set V), with distances defined as follows. Let $b_i \neq a_i$ an arbitrary vertex from C_i, then we define $c_{a_i b_i} = 0$ and $c_{j a_i} = c_{j b_i}, c_{a_i j} = +\infty \;\; \forall j \in V \backslash \{a_i, b_i\}$. In such a way, node a_i can be freely be visited without increasing the length of the tour as soon as node b_i is visited. Building on Figure 1, an example of the creation of an artificial representative vertex is provided in Figure 2, together with the sketch of a relative solution.

In the model, a binary variable x_{ij}, with $i, j \in V$, takes value 1 if vertex i is visited right before vertex j in the solution tour, and value 0 otherwise. In case a vertex $i \in C$ is not visited, then x_{ii} is set to 1, and 0 otherwise.

$$\max \sum_{i \in C} p_i \neg x_{r_i r_i} \tag{1}$$

$$s.t. \;\; \text{AddCircuit}(x_{ij}; i, j \in V; i \neq 0 \vee j \neq 0) \tag{2}$$

$$\sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij} x_{ij} \leq T_{max} \tag{3}$$

$$x_{jj} \implies x_{r_i r_i} \qquad i \in C, j \in C_i \backslash \{r_i\} \tag{4}$$

$$x_{ij} \in \{0; 1\} \qquad i, j \in V \tag{5}$$

The objective function (1) maximizes the profit collected in the tour. Constraint (2) imposes that the tour associated with the active x variables forms a feasible circuit. This is imposed by the CP-SAT statement *AddCircuit* that also ensures that $x_{ii} = 1$ for each variable $i \in C$ not touched by the circuit itself. Constraint (3) is a budget constraint requiring that the length of the tour described by the active x variables has a length of, at most, T_{max}. Constraints (4) use the *AddImplication* statement (here represented as $\Longrightarrow$) and impose that if a vertex of a cluster is not visited, then also the representative of the cluster cannot be visited. As a consequence, a representative vertex can be visited only if all the nodes of its cluster are visited. This is necessary in order not to overestimate the profit collected in the objective function. Constraints (5) finally define the domain of the variables.

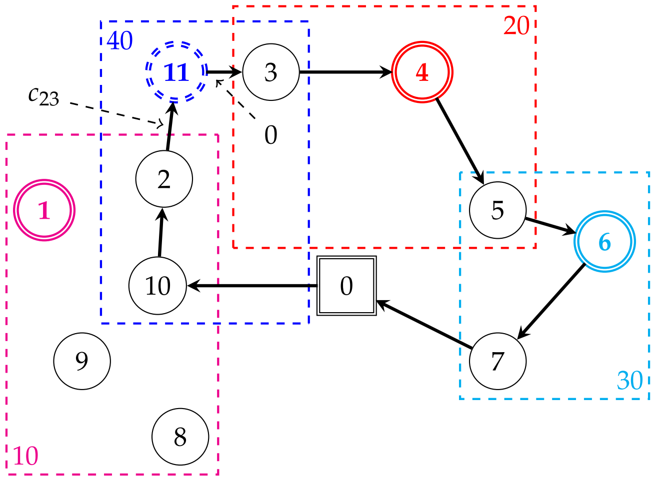

Figure 2. The example of Figure 1 with representative nodes (double framed) and the artificial node 11 added. The solution now goes through node 11. The new cost of the solution arcs incident to 11 is also depicted.

4. Computational Experiments

After having introduced the benchmark set adopted in Section 4.1 and described the experimental settings in Section 4.2, in Section 4.3 we position the new model within the exact methods that have previously appeared in the literature. In Section 4.4, the new best-known results obtained by the model CP are detailed.

4.1. Benchmark Instances

The benchmark instances commonly adopted in the previous COP literature for the exact algorithms are those proposed in [13] and are available at [23]. They are derived from the classic TSPLIB95 instances available at [24]. The instances have a number of vertices ranging between 42 and 318 (as indicated in the names), and the following elements have been added and reflected into the names to make them suitable for the COP (on top of slightly modifying some of the distance matrices):

- Clusters: the number of clusters (field s in the instance name) takes the value of 10, 15, 20, or 25;
- Profits: the profit of a given vertex is generated in two alternative ways ($g1$ or $g2$ in the instance name) based either on the number of vertices in a cluster, or on a pseudo-random value, as explained in details in [13]
- Threshold: two different values are considered for T_{max} ($q2$ or $q3$ in the instance name), which are obtained by considering a fraction of the optimal value of the original TSPLIB95 problem from which the instance is derived, as described in [13].

4.2. Experimental Settings

In this section, we present the results obtained by solving the model we proposed in Section 3 via the CP-SAT solver [22] version 9.8 using standard settings on a computer running an Intel Core i7 12700F processor and equipped with 32 GB of RAM. A maximum computation time of 3600 is allowed for each run, and the solver is launched from scratch without any clue about bounds previously calculated. Notice that we decided to leave out all the preprocessing rules described in [18] when solving the model described in Section 3. The reason is that the remaining rules were either not helping the solver or extremely time-consuming to calculate, and anyway, relying on previously calculated upper and lower bounds, which are normally not available in real applications, when a new instance is faced.

The results are compared with those obtained by all the other methods available in the literature that we are aware of.

The other exact methods we consider are as follows:

- The mixed-integer linear program *BASIC* presented in [13], with experiments run on a computer with an Intel Xeon W3680 CPU and 12 GB of RAM, using CPLEX 12.2 [25] as a MILP solver. A maximum computation time of 3600 is allowed for each run;
- The branch-and-cut algorithm *COP-CLU* presented in [13], with experiments run on a computer with an Intel Xeon W3680 CPU and 12 GB of RAM, using CPLEX 12.2 [25] as a MILP solver. A maximum computation time of 3600 is allowed for each run;
- The branch-and-cut algorithm *CUT-PLA* presented in [18], with experiments run on a computer with an Intel Xeon(R) E2-2670 and 128 GB of RAM, using CPLEX 12.6 [25] as a MILP solver. A maximum computation time of 3600 is allowed for each run. Unfortunately, for this method, only aggregated results are available.

Concerning heuristic approaches, we consider the following:

- The three different tabu search methods *COP-TS-*$*$ discussed in [13] with experiments run on a computer with an Intel Xeon W3680 CPU and 12 GB of RAM;
- The hybrid heuristic *HH* approach discussed in [16] (see also [18]) with experiments run on a computer with an Intel Xeon X7542 CPU;
- The hybrid evolutionary algorithm *HEA* presented in [17] with experiments run on a computer with an Intel Core i5-8400 CPU and 16 GB RAM.

4.3. Comparison with Exact Methods

A first comparison of the results obtained by the CP model compared to all the other exact solvers available in the literature, in terms of success and optimality gap, is presented in Table 2. For each of the exact methods considered, the results after one hour of computation are reported, covering for each class of instances the number of instances solved to optimality over the 16 ones in each group (# *Opt*); the average optimality gap (*Gap* %), calculated as $100(UB(m) - LB(m))/UB(m)$ where $UB(m)$ and $LB(m)$ are the upper and lower bounds retrieved by the generic method m at the end of the 3600 s; the average computation time (*Sec*).

Before commenting the results, some considerations have to be made. First, the different experiments are taken from different papers and made on different computers and using different solvers. These settings inevitably affect the comparison, but we believe the main conclusions remain valid independently of these (inevitable) perturbations. Moreover, no preprocessing rule is used by the methods *BASIC*, *COP-CLU* and *CP*, while an extremely time-consuming preprocessing phase is carried out before *CUT-PLA*. It involves the solution of multiple maximum clique problems [26], multiple smaller COPs, and relies on pre-calculated tight bounds for the original problem. Unfortunately, the time spent on these very time-consuming activities is not accounted for within the computation times reported. In our opinion, this gives the method *CUT-PLA* a clear advantage, turning the comparison against it biased. We report the results anyway for the sake of completeness.

Table 2. Comparison against published exact methods—aggregated results.

Classes of Instances	BASIC (Angelelli et al. [13])			COP-CLU (Angelelli et al. [13])			CUT-PLA (Yahiaoui et al. [18])			CP		
	# Opt	Gap %	Sec	# Opt	Gap %	Sec	# Opt	Gap %	Sec	# Opt	Gap %	Sec
dantzig42	16	0.0	8.1	16	0.0	0.6	16	0.0	2.9	16	0.0	4.6
swiss42	16	0.0	6.6	16	0.0	1.0	16	0.0	4.1	16	0.0	5.1
att48	16	0.0	13.2	16	0.0	8.5	16	0.0	34.8	16	0.0	25.9
gr48	16	0.0	11.9	16	0.0	4.9	16	0.0	12.5	16	0.0	18.5
hk48	16	0.0	10.9	16	0.0	5.5	16	0.0	12.4	16	0.0	19.3
eil51	16	0.0	16.7	16	0.0	6.8	16	0.0	16.6	16	0.0	30.2
berlin52	16	0.0	12.0	16	0.0	2.3	16	0.0	10.0	16	0.0	13.9
brazil58	16	0.0	16.4	16	0.0	5.5	16	0.0	26.7	16	0.0	24.0
st70	16	0.0	52.7	16	0.0	57.7	16	0.0	95.7	16	0.0	400.2
eil76	16	0.0	20.2	16	0.0	18.2	16	0.0	25.8	16	0.0	62.6
pr76	16	0.0	94.4	15	0.5	501.7	16	0.0	196.4	16	0.0	276.9
gr96	16	0.0	34.8	16	0.0	23.3	16	0.0	37.2	16	0.0	78.7
rat99	16	0.0	232.4	16	0.0	93.1	16	0.0	159.3	16	0.0	458.6
kroA100	14	3.7	1112.5	16	0.0	279.0	16	0.0	241.0	16	0.0	520.4
kroB100	11	7.0	1386.0	16	0.0	468.6	16	0.0	272.5	16	0.0	762.3
kroC100	11	7.2	1968.0	16	0.0	602.8	15	0.5	746.3	16	0.0	696.5
kroD100	14	2.1	1613.8	16	0.0	647.2	16	0.0	581.6	16	0.0	1022.8
kroE100	13	2.8	1107.8	15	2.1	562.5	16	0.0	179.1	16	0.0	717.3
rd100	10	3.8	1702.8	16	0.0	537.0	16	0.0	225.1	16	0.0	1387.9
eil101	16	0.0	94.0	16	0.0	50.9	16	0.0	116.0	16	0.0	236.5
lin105	16	0.0	104.4	16	0.0	59.4	16	0.0	72.7	16	0.0	184.1
pr107	16	0.0	171.3	16	0.0	61.6	15	1.9	332.5	16	0.0	112.2
gr120	10	4.8	1854.9	15	1.8	948.2	15	0.6	688.8	16	0.0	1013.0
pr124	16	0.0	139.7	16	0.0	78.5	16	0.0	100.5	16	0.0	293.6
bier127	16	0.0	283.8	16	0.0	97.4	16	0.0	92.2	16	0.0	211.0
ch130	5	11.7	2717.6	12	2.6	1844.0	15	0.4	767.3	12	3.1	2480.0
pr136	11	4.6	2024.1	16	0.0	765.1	16	0.0	326.3	16	0.0	1106.1
gr137	16	0.0	144.9	16	0.0	62.9	16	0.0	149.9	16	0.0	204.6
pr144	16	0.0	229.0	16	0.0	119.4	16	0.0	135.7	16	0.0	407.1
ch150	1	37.2	3438.7	0	44.4	3600.7	9	5.8	2098.0	5	13.3	2859.9
kroA150	2	32.5	3488.5	3	32.8	3232.8	10	5.8	1713.4	7	13.9	2641.9
kroB150	1	33.3	3591.2	1	34.6	3571.2	11	5.6	1913.8	6	12.2	2782.9
pr152	16	0.0	266.4	16	0.0	130.0	16	0.0	559.2	16	0.0	545.8
u159	13	1.3	923.5	16	0.0	442.1	14	1.1	1185.8	15	0.1	1844.1
si175	16	0.0	844.0	16	0.0	424.5	16	0.0	315.5	16	0.0	950.3
brg180	16	0.0	597.5	16	0.0	141.5	16	0.0	155.6	16	0.0	422.0
rat195	5	7.5	3020.1	7	8.1	3004.0	10	4.5	1757.7	8	4.9	2821.1
d198	16	0.0	179.5	14	1.4	1035.5	12	3.3	1431.6	16	0.0	1334.7
kroA200	0	62.5	3600.8	0	74.6	3601.0	6	14.7	2562.2	3	24.1	3281.1
kroB200	2	45.6	3471.5	0	64.2	3600.4	6	17.7	2550.0	3	24.6	3142.6
gr202	13	1.3	814.4	13	1.4	1255.6	16	0.0	557.1	15	0.6	1799.6
ts225	0	36.9	3600.5	0	71.2	3600.4	0	27.3	3600.7	0	23.5	3600.0
tsp225	1	26.2	3504.6	3	14.8	3335.7	10	6.8	2461.4	4	8.7	3428.1
pr226	10	11.0	1782.0	14	1.9	1485.8	5	14.4	3066.9	11	10.9	2342.3
gr229	16	0.0	299.6	13	2.1	1645.3	16	0.0	1063.6	14	0.5	2023.9
gil262	0	61.4	3601.5	0	91.8	3601.6	4	19.9	3084.4	1	28.9	3436.3
pr264	8	6.6	2499.8	10	17.5	2105.1	2	35.0	3155.6	8	8.6	3191.0
a280	1	17.6	3415.7	1	23.5	3464.0	2	17.7	3357.5	1	13.4	3424.3
pr299	1	17.6	3569.6	2	54.8	3554.8	1	28.4	3449.9	2	10.9	3421.6
lin318	1	15.3	3543.9	0	53.1	3600.6	1	15.5	3430.7	1	15.4	3563.4
Average	**11.2**	**9.2**	**1344.8**	**12.0**	**12.0**	**1166.9**	**12.9**	**4.5**	**982.6**	**12.6**	**4.3**	**1312.6**

The aggregated results presented in Table 2 indicate that the new model CP is competitive with the state-of-the-art exact solvers discussed in the literature. In particular, the statistics indicate that the CP model is able to provide the smallest optimality gap of all the methods, notwithstanding it does not take advantage of the aggressive preprocessing run before the solver *CUT-PLA*. The latter has slightly better results in terms of the average number of instances solved to optimality, and shorter computation times, although the time of the heavy preprocessing run for *CUT-PLA* is not accounted for in these figures. The methods *BASIC* and *COP-CLU* appear to deliver worse performance, although the fact that for some groups of instances they are the best, indicate that a clear dominance among the approaches is not present.

In Figures 3 and 4, a graphical representation of the results of Table 2 is proposed, in terms of average optimality gaps and computation times. The instances are presented in a non-decreasing order of size, in order to allow considerations on the scalability for the different approaches. For this purpose, quadratic trend lines are also inserted in the charts.

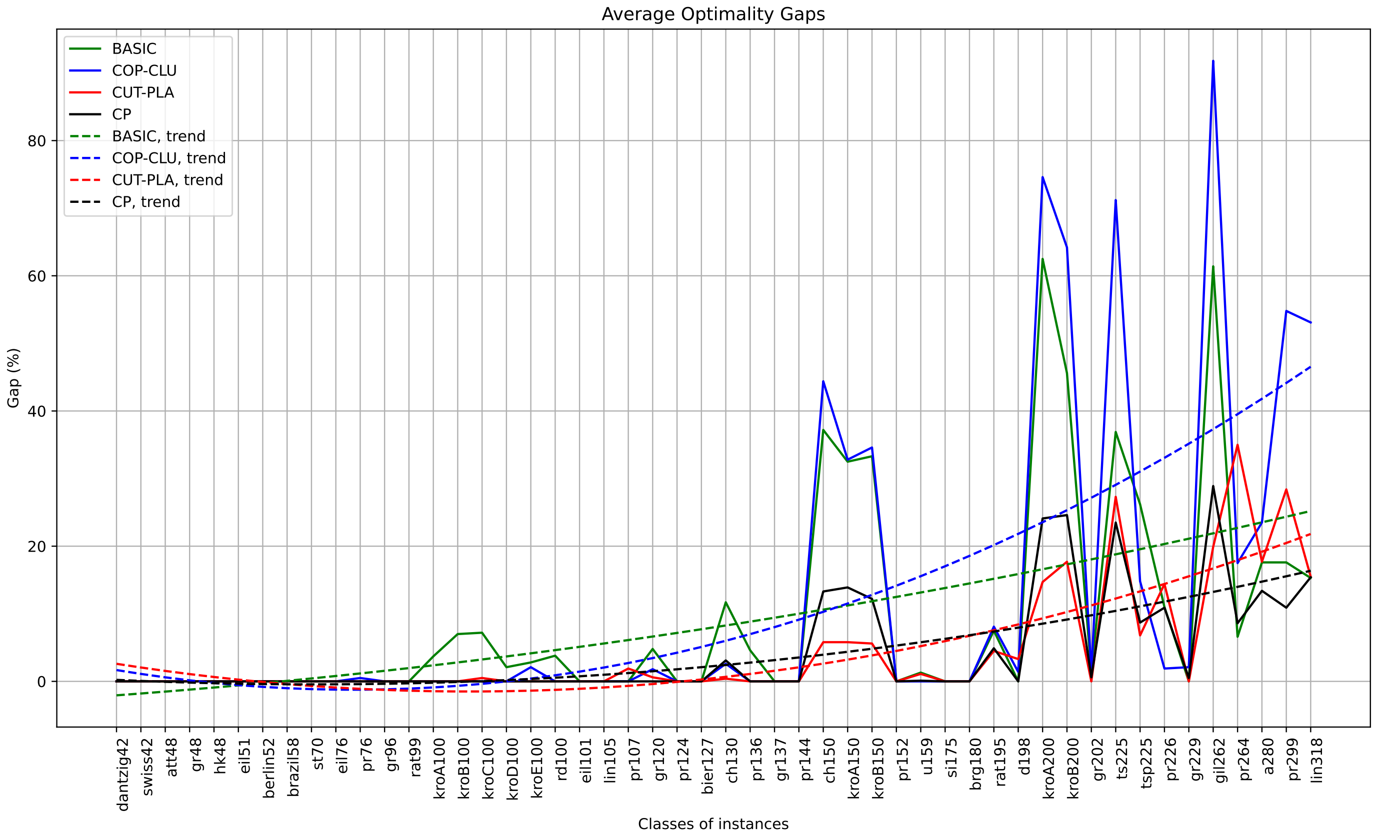

Figure 3. Average optimality gaps for the different exact methods for instances of increasing size.

Figure 3 suggests that our proposal CP is the one scaling better among the methods, having a fairly flat curvature of the trend line. In detail, CP emerges as the most effective method on the larger instances, while CUT-PLA is superior on the small and medium ones—always keeping in mind the preprocessing advantage of the latter method mentioned before. The method COP-CLU presents the worst figures in terms of scaling. It is finally interesting to observe the presence of a cluster of hard instances, difficult to solve for all the methods considered, with a size around 150.

Figure 4 highlights that the approach CP we propose appears to have a similar trend to BASIC and COP-CLU in terms of average computation times, although with different spikes, sometimes worse, sometimes better, with a trend line worse than that of BASIC but better than that of COP-CLU. Although it is not possible to draw accurate conclusions about the method CUT-PLA due to the unknown time spent by the method in preprocessing, it appears to be the fastest of all, but with a degradation in performance associated with the larger instances that poses doubts on scalability, as highlighted by the least promising trend line of all.

The detailed results obtained by solving the model CP are available in Appendix A for future reference.

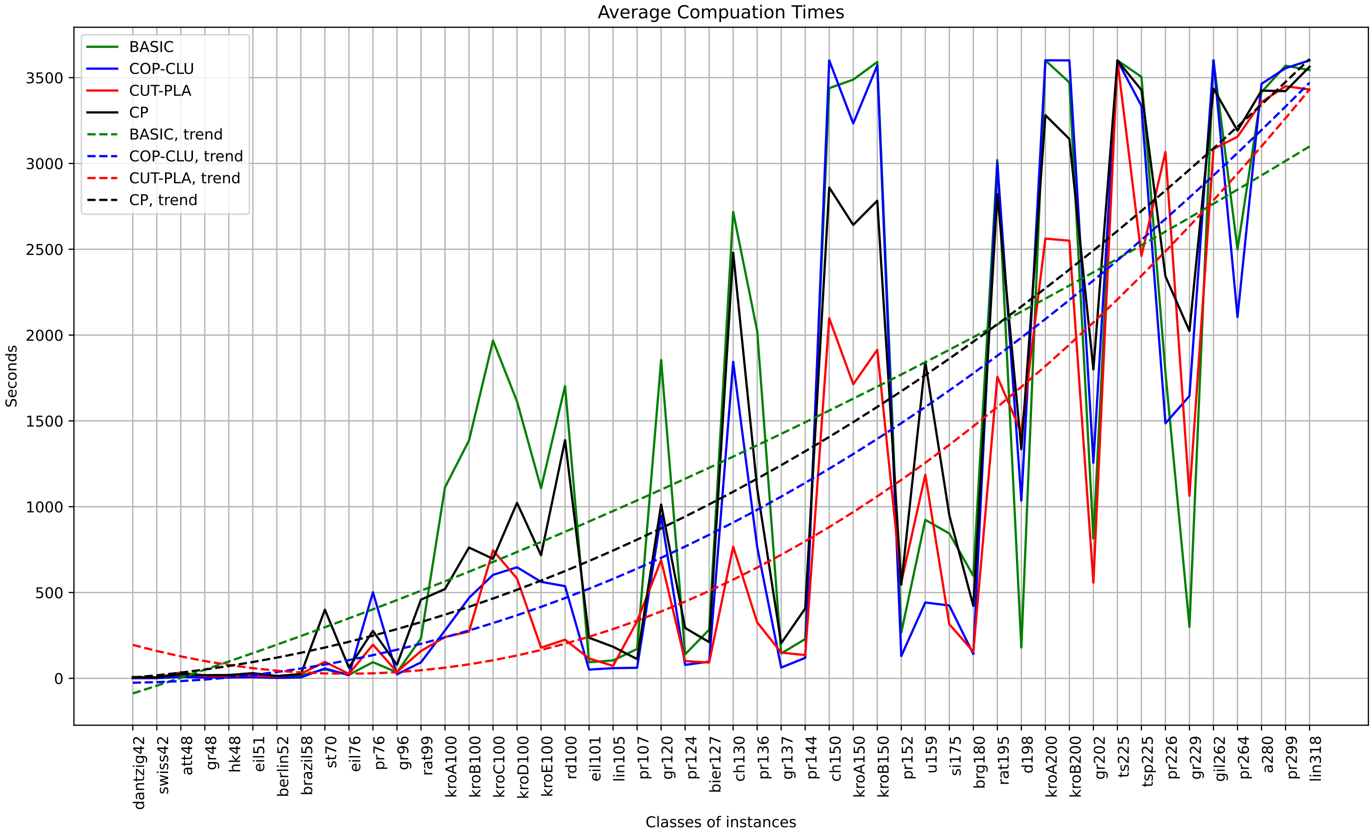

Figure 4. Average computation times for the different exact methods for instances of increasing size.

4.4. Improved Best-Known Results

In this section, we compare the results retrieved by CP with the best-known lower and upper bounds for the optimal costs available in the literature, and obtained by all the exact and heuristic methods listed in Section 4.2. The best lower bounds are mainly obtained by the tailored heuristic methods, while the upper bounds are exclusively made available by exact methods. In general, the new CP model was able to retrieve many improved upper bounds with respect to those reported in the previous literature, leading also to the first optimality proof for the solutions of several instances. In the following Table 3, we detail these new state-of-the-art bounds. In the table we indicate, for each instance affected by an improvement, the previous best-known lower and upper bounds, and the upper bounds obtained by the new model CP. In the column *Opt*, we finally mark (with $\star$) those instances for which an exact solution is documented for the first time in this study. According to the table, a total of 118 new improved lower bounds are presented, with optimality proven for the first time for 37 of the instances. For the sake of completeness, we report that for 27 of these 37 instances the model CP was able to close the instance, while for the remaining 10 optimality was inferred by comparing the new upper bounds with the lower bounds previously known.

Table 3. New best bounds and new optimality proofs.

Instance	*s*	*g*	*q*	Best Known LB	Best Known UB	CP UB	CP Opt	Instance	*s*	*g*	*q*	Best Known LB	Best Known UB	CP UB	CP Opt
gr120	15	1	2	49	56.1	49	$\star$	kroB200	15	1	3	123	156.2	139	
ch130	25	1	2	52	59.7	59		kroB200	15	2	3	6204	7828.3	7565	
ch150	10	1	3	85	105.3	85	$\star$	kroB200	20	1	2	48	106.2	84	
ch150	10	2	2	1773	2580.9	1773	$\star$	kroB200	20	1	3	132	166.9	144	
ch150	10	2	3	4380	5119.7	4380	$\star$	kroB200	20	2	3	6814	8354.2	8239	

Table 3. *Cont.*

Instance	s	g	q	Best Known LB	Best Known UB	CP UB	CP Opt	Instance	s	g	q	Best Known LB	Best Known UB	CP UB	CP Opt
ch150	15	2	2	1882	3639.6	1882	⋆	kroB200	25	1	2	60	120.0	90	
ch150	15	2	3	4961	5736.1	5515		kroB200	25	1	3	140	176.8	160	
ch150	20	1	2	56	79.3	65		kroB200	25	2	3	7390	8931.2	8671	
ch150	20	1	3	114	127.0	117		gr202	15	1	2	124	133.2	124	⋆
ch150	20	2	2	2781	3913.2	3541		ts225	10	1	2	81	131.4	107	
ch150	20	2	3	5665	6524.3	6308		ts225	10	1	3	160	197.3	186	
ch150	25	1	2	48	94.1	72		ts225	10	2	2	4058	6613.6	6127	
ch150	25	1	3	120	130.7	128		ts225	10	2	3	8036	9680.9	9386	
ch150	25	2	2	2597	4593.7	3801		ts225	15	1	2	102	129.2	102	⋆
kroA150	10	1	3	85	102.2	85	⋆	ts225	15	1	3	170	198.6	187	
kroA150	10	2	3	4379	4990.1	4379	⋆	ts225	20	1	2	107	136.8	121	
kroA150	15	1	2	36	60.0	36	⋆	ts225	20	1	3	186	206.3	199	
kroA150	15	2	2	1882	3302.7	2925		ts225	25	1	2	121	144.0	132	
kroA150	20	1	2	40	74.1	59		ts225	25	1	3	187	215.0	209	
kroA150	20	1	3	108	127.9	120		ts225	25	2	2	6072	7269.2	7147	
kroA150	20	2	2	2109	3634.2	3354		tsp225	10	1	2	107	110.9	108	
kroA150	20	2	3	5532	6338.2	6196		tsp225	10	2	2	5274	5591.4	5362	
kroA150	25	1	2	48	78.1	64		tsp225	15	1	2	102	112.9	102	⋆
kroA150	25	1	3	120	128.3	127		tsp225	15	1	3	170	180.9	170	⋆
kroA150	25	2	2	2424	4003.0	3890		tsp225	20	1	2	107	117.3	108	
kroB150	10	1	2	34	38.9	34	⋆	tsp225	20	1	3	186	187.2	186	⋆
kroB150	10	1	3	85	105.9	85	⋆	gil262	10	1	2	61	110.4	61	⋆
kroB150	10	2	3	4390	5431.1	4390	⋆	gil262	10	1	3	151	203.9	181	
kroB150	15	1	2	36	64.7	48		gil262	10	2	2	3125	5337.1	4640	
kroB150	15	1	3	107	116.0	107	⋆	gil262	10	2	3	7750	10,191.2	9245	
kroB150	15	2	2	1882	3654.9	2496		gil262	15	1	2	78	123.6	99	
kroB150	15	2	3	5421	5762.8	5421	⋆	gil262	15	1	3	175	205.9	195	
kroB150	20	1	2	45	68.7	59		gil262	15	2	3	8961	10,313.3	10,012	
kroB150	20	1	3	112	122.7	117		gil262	20	1	2	76	139.7	121	
kroB150	20	2	2	2212	3782.5	3336		gil262	20	1	3	181	214.1	196	
kroB150	20	2	3	5733	6222.1	6194		gil262	20	2	3	9184	11,051.5	10,780	
kroB150	25	1	2	48	96.3	64		gil262	25	1	2	87	135.0	133	
kroB150	25	1	3	119	130.8	128		gil262	25	1	3	188	221.7	215	
kroB150	25	2	2	2488	4270.7	3724		gil262	25	2	3	9649	11,393.6	11,283	
rat195	10	1	2	87	90.8	87	⋆	pr264	10	2	2	4682	7196.4	5416	
rat195	20	1	3	167	167.3	167	⋆	pr264	15	2	3	8985	10,164.7	9967	
rat195	25	2	2	5695	5948.4	5695	⋆	a280	10	1	2	128	140.2	128	⋆
kroA200	10	1	2	44	76.7	44	⋆	a280	10	1	3	224	230.1	224	⋆
kroA200	10	1	3	110	140.8	110	⋆	a280	20	1	2	144	154.0	144	⋆
kroA200	10	2	2	2258	3698.8	2258	⋆	a280	20	1	3	224	233.5	224	⋆
kroA200	10	2	3	5655	7390.1	6660		a280	25	1	2	147	158.4	157	
kroA200	15	1	2	48	105.1	78		pr299	10	1	2	136	146.7	136	⋆
kroA200	15	1	3	124	157.7	139		pr299	10	1	3	236	242.7	238	
kroA200	15	2	2	2470	4643.7	4598		pr299	10	2	2	6792	7720.7	7263	
kroA200	15	2	3	6200	8056.5	7613		pr299	10	2	3	11,986	12,117.3	12,057	
kroA200	20	1	2	48	98.8	84		pr299	15	1	2	132	153.2	153	
kroA200	20	1	3	132	178.0	144		pr299	20	1	3	238	253.9	238	⋆
kroA200	20	2	3	6654	8429.2	8302		pr299	25	1	3	238	257.2	252	
kroA200	25	1	2	60	116.7	80		lin318	10	1	2	152	173.5	152	⋆
kroA200	25	1	3	140	174.0	159		lin318	10	1	3	265	265.2	265	⋆
kroA200	25	2	3	7170	9118.2	8901		lin318	15	1	2	152	174.3	152	⋆
kroB200	10	1	3	110	143.1	110	⋆	lin318	15	1	3	252	271.0	252	⋆
kroB200	10	2	3	5675	7276.5	6680		lin318	20	1	2	162	176.9	162	⋆
kroB200	15	1	2	47	89.2	78		lin318	25	1	2	175	184.4	180	

5. Conclusions

In this work, we have considered the Clustered Orienteering Problem, an optimization problem faced in last-mile logistics. The aim is, given an available time window, to select the vertices to visit among a set of service requests in order to maximize the total profit collected. In particular, vertices belong to clusters, and the profits are associated to clusters, and the price relative to a cluster is collected only if all the vertices of such a cluster are visited.

We propose a constraint programming model for the problem and we empirically evaluate the potential of the new model by solving it with out-of-the-box software. The results indicate that, when compared to the exact methods currently available in the literature, the new approach proposed is dominant. Several improved upper bounds are provided in the study, and different instances are closed here for the first time. Overall, we provided an easy-to-implement tool providing well-optimized solutions, which, in turn, translates into a more sustainable logistics organization.

Future work should be in the direction of integrating the new model we propose with the high-performance heuristic methods available in order to enhance the quality of the lower bounds. Moreover, the approach should be extended to deal with the Team Orienteering version of the problem—where several vehicles operate together—and compared with the existing literature on these settings.

Author Contributions: Conceptualization, R.M. and D.H.S.; methodology, R.M.; software, R.M.; validation, R.M. and D.H.S.; formal analysis, R.M. and D.H.S.; investigation, R.M. and D.H.S.; resources, R.M.; data curation, R.M.; writing—original draft preparation, R.M. and D.H.S.; writing—review and editing, R.M. and D.H.S.; visualization, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available from [23].

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

In this appendix, we report extensively the results obtained by the CP model for reference.

The results were obtained by attacking the model with the CP-SAT solver [22] version 9.8 using standard settings on a computer equipped with an Intel Core i7 12700F processor and 32 GB of RAM, with a maximum computation time of 1 h.

Table A1. Detailed results retrieved by the new model.

Instance	s	$(g,q)=(1,2)$ LB	UB	$(g,q)=(1,3)$ LB	UB	$(g,q)=(2,2)$ LB	UB	$(g,q)=(2,3)$ LB	UB
dantzig42	10	31	31	43	43	1519	1519	2133	2133
	15	36	36	51	51	1789	1789	2499	2499
	20	41	41	61	61	2074	2074	3004	3004
	25	48	48	71	71	2451	2451	3499	3499
swiss42	10	25	25	37	37	1231	1231	1885	1885
	15	30	30	49	49	1486	1486	2443	2443
	20	37	37	57	57	1829	1829	2807	2807
	25	40	40	60	60	1973	1973	3042	3042
att48	10	14	14	35	35	733	733	1820	1820
	15	20	20	46	46	1005	1005	2280	2280
	20	23	23	49	49	1145	1145	2581	2581
	25	24	24	56	56	1236	1236	2856	2856
gr48	10	18	18	42	42	832	832	2109	2109
	15	21	21	52	52	1050	1050	2519	2519
	20	28	28	57	57	1339	1339	2792	2792
	25	33	33	66	66	1633	1633	3208	3208

Table A1. *Cont.*

Instance	s	$(g,q)=(1,2)$ LB	UB	$(g,q)=(1,3)$ LB	UB	$(g,q)=(2,2)$ LB	UB	$(g,q)=(2,3)$ LB	UB
hk48	10	14	14	35	35	733	733	1820	1820
	15	25	25	45	45	1250	1250	2220	2220
	20	24	24	53	53	1188	1188	2590	2590
	25	32	32	63	63	1616	1616	3025	3025
eil51	10	21	21	42	42	1087	1087	2149	2149
	15	26	26	50	50	1193	1193	2325	2325
	20	29	29	59	59	1320	1320	2793	2793
	25	32	32	64	64	1610	1610	3188	3188
berlin52	10	21	21	42	42	1158	1158	2251	2251
	15	32	32	53	53	1694	1694	2707	2707
	20	36	36	62	62	1870	1870	3111	3111
	25	44	44	68	68	2322	2322	3486	3486
brazil58	10	22	22	39	39	1045	1045	1945	1945
	15	34	34	57	57	1775	1775	2822	2822
	20	37	37	63	63	1865	1865	3235	3235
	25	44	44	73	73	2106	2106	3438	3438
st70	10	18	18	45	45	999	999	2321	2321
	15	27	27	58	58	1202	1202	2811	2811
	20	29	29	68	68	1424	1424	3359	3359
	25	34	34	73	73	1665	1665	3660	3660
eil76	10	38	38	59	59	1726	1726	2902	2902
	15	42	42	70	70	1994	1994	3440	3440
	20	47	47	81	81	2217	2217	3868	3868
	25	50	50	85	85	2440	2440	4275	4275
pr76	10	48	48	77	77	2362	2362	3755	3755
	15	49	49	84	84	2523	2523	4178	4178
	20	54	54	94	94	2827	2827	4721	4721
	25	65	65	100	100	3295	3295	5130	5130
gr96	10	59	59	82	82	2895	2895	4062	4062
	15	66	66	93	93	3305	3305	4719	4719
	20	70	70	105	105	3515	3515	5268	5268
	25	78	78	118	118	3987	3987	6067	6067
rat99	10	48	48	72	72	2376	2376	3664	3664
	15	54	54	80	80	2693	2693	3996	3996
	20	56	56	91	91	2812	2812	4587	4587
	25	60	60	102	102	3094	3094	5055	5055
kroA100	10	12	12	60	60	654	654	3069	3069
	15	27	27	70	70	1505	1505	3564	3564
	20	35	35	77	77	1890	1890	3999	3999
	25	30	30	84	84	1691	1691	4346	4346
kroB100	10	23	23	71	71	1154	1154	3469	3469
	15	26	26	77	77	1439	1439	4040	4040
	20	34	34	83	83	1756	1756	4349	4349
	25	41	41	95	95	2159	2159	4917	4917
kroC100	10	12	12	60	60	634	634	2990	2990
	15	26	26	71	71	1341	1341	3494	3494
	20	28	28	70	70	1366	1366	3640	3640
	25	30	30	83	83	1575	1575	4087	4087
kroD100	10	12	12	59	59	654	654	2936	2936
	15	26	26	67	67	1405	1405	3467	3467
	20	28	28	77	77	1636	1636	4143	4143
	25	36	36	83	83	1889	1889	4319	4319
kroE100	10	24	24	71	71	1288	1288	3590	3590
	15	26	26	76	76	1456	1456	3930	3930
	20	28	28	84	84	1646	1646	4387	4387
	25	36	36	90	90	1950	1950	4737	4737

Table A1. *Cont.*

Instance	s	$(g,q)=(1,2)$ LB	UB	$(g,q)=(1,3)$ LB	UB	$(g,q)=(2,2)$ LB	UB	$(g,q)=(2,3)$ LB	UB
rd100	10	24	24	60	60	1288	1288	3090	3090
	15	34	34	71	71	1879	1879	3641	3641
	20	35	35	84	84	1995	1995	4303	4303
	25	42	42	90	90	2169	2169	4573	4573
eil101	10	48	48	84	84	2456	2456	4298	4298
	15	51	51	86	86	2613	2613	4516	4516
	20	56	56	98	98	3002	3002	5161	5161
	25	66	66	108	108	3393	3393	5458	5458
lin105	10	51	51	88	88	2500	2500	4368	4368
	15	54	54	99	99	2789	2789	4940	4940
	20	60	60	109	109	2920	2920	5234	5234
	25	69	69	112	112	3531	3531	5748	5748
pr107	10	39	39	65	65	1958	1958	3210	3210
	15	54	54	73	73	2709	2709	3766	3766
	20	61	61	80	80	2958	2958	4016	4016
	25	65	65	89	89	3255	3255	4535	4535
gr120	10	42	42	84	84	2181	2181	4350	4350
	15	49	49	99	99	2491	2491	4955	4955
	20	56	56	104	104	2800	2800	5339	5339
	25	56	56	116	116	3102	3102	5941	5941
pr124	10	44	44	87	87	2280	2280	4455	4455
	15	62	62	103	103	3102	3102	5139	5139
	20	66	66	115	115	3221	3221	5631	5631
	25	77	77	124	124	3779	3779	6246	6246
bier127	10	75	75	118	118	3700	3700	5882	5882
	15	86	86	136	136	4315	4315	6874	6874
	20	94	94	142	142	4751	4751	7125	7125
	25	99	99	148	148	5069	5069	7530	7530
ch130	10	30	30	89	89	1546	1546	4421	4421
	15	44	44	97	97	2266	2266	4483	4844
	20	53	53	104	104	2665	2665	5246	5585
	25	52	59	115	115	2605	3414	5785	5785
pr136	10	48	48	95	95	2500	2500	4781	4781
	15	66	66	110	110	3415	3415	5635	5635
	20	72	72	117	117	3624	3624	5899	5899
	25	79	79	128	128	4019	4019	6592	6592
gr137	10	64	64	111	111	3256	3256	5635	5635
	15	77	77	122	122	3892	3892	6115	6115
	20	81	81	132	132	4113	4113	6618	6618
	25	87	87	137	137	4470	4470	7060	7060
pr144	10	66	66	98	98	3265	3265	5001	5001
	15	72	72	117	117	3584	3584	5942	5942
	20	74	74	128	128	3743	3743	6464	6464
	25	80	80	136	136	4040	4040	6784	6784
ch150	10	34	34	85	85	1773	1773	4380	4380
	15	36	60	96	107	1882	1882	4961	5515
	20	56	65	107	117	2781	3541	5340	6308
	25	48	72	119	128	2185	3801	6022	6707
kroA150	10	34	34	85	85	1708	1708	4379	4379
	15	36	36	108	108	1882	2925	5475	5475
	20	40	59	107	120	2072	3354	5512	6196
	25	40	64	120	127	2300	3890	6000	6669
kroB150	10	34	34	85	85	1753	1753	4390	4390
	15	36	48	107	107	1882	2496	5421	5421
	20	45	59	112	117	2212	3336	5733	6194
	25	48	64	119	128	2488	3724	5821	6545

Table A1. *Cont.*

Instance	s	$(g,q)=(1,2)$ LB	UB	$(g,q)=(1,3)$ LB	UB	$(g,q)=(2,2)$ LB	UB	$(g,q)=(2,3)$ LB	UB
pr152	10	68	68	102	102	3414	3414	5141	5141
	15	72	72	120	120	3656	3656	6060	6060
	20	80	80	127	127	4100	4100	6541	6541
	25	88	88	136	136	4436	4436	6960	6960
u159	10	72	72	125	125	3696	3696	6227	6227
	15	85	85	138	138	4072	4072	6845	6845
	20	88	90	148	148	4374	4374	7364	7364
	25	99	99	157	157	4955	4955	7716	7716
si175	10	80	80	137	137	4075	4075	6861	6861
	15	98	98	149	149	5033	5033	7456	7456
	20	99	99	153	153	5171	5171	7831	7831
	25	108	108	162	162	5478	5478	8249	8249
brg180	10	80	80	140	140	4100	4100	6990	6990
	15	97	97	140	140	4830	4830	7186	7186
	20	99	99	154	154	5107	5107	7779	7779
	25	103	103	166	166	5249	5249	8312	8312
rat195	10	87	87	129	150	4481	4481	6555	7699
	15	105	105	150	164	5265	5265	7475	8126
	20	108	108	156	167	5486	5486	7915	8734
	25	110	110	158	179	5695	5695	8685	9095
d198	10	110	110	153	153	5595	5595	7738	7738
	15	121	121	166	166	6228	6228	8513	8513
	20	120	120	192	192	6100	6100	9624	9624
	25	130	130	200	200	6625	6625	10,100	10,100
kroA200	10	44	44	110	110	2258	2258	5540	6660
	15	48	78	108	139	2415	4598	5957	7613
	20	48	84	132	144	2496	5047	6654	8302
	25	60	80	140	159	2970	5612	6006	8901
kroB200	10	44	44	110	110	2298	2298	5541	6680
	15	47	78	122	139	2438	4697	5458	7565
	20	48	84	132	144	2516	5178	6119	8239
	25	60	90	140	160	2685	5499	6631	8671
gr202	10	121	121	193	193	6280	6280	9869	9869
	15	124	124	200	200	6306	6946	10,080	10,080
	20	144	144	205	205	7332	7332	10,342	10,342
	25	150	150	221	221	7525	7525	11,104	11,104
ts225	10	79	107	159	186	3983	6127	7987	9386
	15	68	102	135	187	4355	6607	6912	10,107
	20	92	121	173	199	5491	7009	9063	10,843
	25	99	132	164	209	5497	7147	9407	11,014
tsp225	10	81	108	158	183	5267	5362	7867	9283
	15	85	102	170	170	5089	6118	8551	9379
	20	107	108	186	186	5562	6537	9431	9956
	25	121	121	197	197	5505	6545	9887	10,242
pr226	10	81	107	186	186	4137	6762	9633	9633
	15	102	102	204	204	4390	7641	10,388	10,388
	20	117	117	212	212	5642	8142	11,197	11,197
	25	110	110	220	220	5100	8210	11,195	11,195
gr229	10	162	162	215	215	8139	8139	10,809	10,809
	15	155	155	206	206	7858	7858	10,439	10,439
	20	187	187	228	228	9179	9179	10,768	11,348
	25	188	188	233	233	9466	9466	11,718	12,041
gil262	10	61	61	151	181	3125	4640	7690	9245
	15	78	99	136	195	3101	6710	7951	10,012
	20	75	121	181	196	3669	7060	8350	10,780
	25	83	133	174	215	3227	7394	6709	11,283

Table A1. *Cont.*

Instance	s	$(g,q) = (1,2)$ LB	UB	$(g,q) = (1,3)$ LB	UB	$(g,q) = (2,2)$ LB	UB	$(g,q) = (2,3)$ LB	UB
pr264	10	92	92	182	182	4682	5416	7802	10,423
	15	120	120	178	197	6080	6080	8971	9967
	20	122	122	197	197	6155	6155	8594	12,019
	25	130	186	215	229	6565	6565	10,331	12,024
a280	10	128	128	191	224	6296	7610	9554	11,304
	15	105	147	186	226	6137	7667	10,405	11,400
	20	128	144	223	224	6318	7797	11,165	11,809
	25	131	157	210	238	6808	8306	11,411	12,533
pr299	10	136	136	204	238	5269	7263	10,274	12,057
	15	132	153	219	242	6752	8308	11,099	12,649
	20	136	153	238	238	7704	8598	12,024	12,923
	25	154	167	236	252	7795	8878	12,102	13,337
lin318	10	76	152	226	265	5854	9471	9682	13,502
	15	151	152	227	252	7660	9162	11,466	14,012
	20	162	162	251	270	8215	9294	12,780	13,941
	25	161	180	264	279	7341	9337	13,190	13,894

References

1. Tsiligirides, T. Heuristic methods applied to orienteering. *J. Oper. Res. Soc.* **1984**, *35*, 797–809. [CrossRef]
2. Golden, B.L.; Levy, L.; Vohra, R. The orienteering problem. *Nav. Res. Logist.* **1987**, *3*, 307–318. [CrossRef]
3. Vansteenwegen, P.; Souffriau, W.; van Oudheusden, D. The orienteering problem: A survey. *Eur. J. Oper. Res.* **2011**, *209*, 1–10. [CrossRef]
4. Gunawan, A.; Lau, H.C.; Vansteenwegen, P. Orienteering problem: A survey of recent variants, solution approaches and applications. *Eur. J. Oper. Res.* **2016**, *2*, 315–332. [CrossRef]
5. Fischetti, M.; Salazar-Gonzalez, J.; Toth, P. The generalized traveling salesman and orienteering problems. In *The Traveling Salesman Problem and Its Variations*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 609–662.
6. Archetti, C.; Speranza, M.G.; Vigo, D. Vehicle routing problems with profits. In *Vehicle Routing: Problems, Methods, and Applications*; Toth, P., Vigo, D., Eds.; MOS-SIAM Series on Optimization; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2014; pp. 273–298.
7. Statista. E-Commerce. 2024. Available online: https://www.statista.com/markets/413/e-commerce/ (accessed on 25 April 2024).
8. Alidaee, B.; Wang, H.; Sua, L. The last-mile delivery of heavy, bulky, oversized products: Literature review and research agenda. *Logistics* **2023**, *7*, 98. [CrossRef]
9. Rüther, C.; Rieck, J. A bayesian optimization approach for tuning a grouping genetic algorithm for folving practically oriented pickup and delivery problems. *Logistics* **2024**, *8*, 14. [CrossRef]
10. Dell'Amico, M.; Montemanni, R.; Novellani, S. Pickup and delivery with lockers. *Transp. Res. Part C Emerg. Technol.* **2023**, *148*, 104022. [CrossRef]
11. Li, F.; Kunze, O. A comparative review of air drones (UAVs) and delivery bots (SUGVs) for automated last mile home delivery. *Logistics* **2023**, *7*, 21. [CrossRef]
12. Saker, A.; Eltawil, A.; Ali, I. Adaptive large neighborhood search metaheuristic for the capacitated vehicle routing problem with parcel lockers. *Logistics* **2023**, *7*, 72. [CrossRef]
13. Angelelli, E.; Archetti, C.; Vindigni, M. The clustered orienteering problem. *Eur. J. Oper. Res.* **2014**, *238*, 404–414. [CrossRef]
14. Archetti, C.; Carrabs, F.; Cerulli, R. The set orienteering problem. *Eur. J. Oper. Res.* **2018**, *1*, 264–272. [CrossRef]
15. Archetti, C.; Carrabs, F.; Cerulli, R.; Laureana, F. A new formulation and a branch-and-cut algorithm for the set orienteering problem. *Eur. J. Oper. Res.* **2024**, 314, 446–465. [CrossRef]
16. Yahiaoui, A.E.; Moukrim, A.; Serairi, M. Hybrid Heuristic for the Clustered Orienteering Problem. In *Computational Logistics: 8th International Conference, ICCL 2017, Southampton, UK, 18–20 October 2017*; Springer International Publishing: Berlin, Germany, 2017; Volume 10572.
17. Wu, Q.; He, M.; Has, J.K.; Lu, Y. An effective hybrid evolutionary algorithm for the clustered orienteering problem. *Eur. J. Oper. Res.* **2024**, *313*, 418–434. [CrossRef]
18. Yahiaoui, A.E.; Moukrim, A.; Serairi, M. The clustered team orienteering problem. *Comput. Oper. Res.* **2019**, *111*, 386–399. [CrossRef]
19. Montemanni, R.; Dell'Amico, M. Solving the parallel drone scheduling traveling salesman problem via constraint programming. *Algorithms* **2023**, *16*, 40. [CrossRef]
20. Montemanni, R.; Smith, D.H. On solving the set orienteering problem. *Symmetry* **2024**, *26*, 340. [CrossRef]
21. Google. OR-Tools. Available online: https://developers.google.com/optimization/ (accessed on 14 March 2024).

22. Perron, L.; Didier, F. CP-SAT. Available online: https://developers.google.com/optimization/cp/cp_solver/ (accessed on 14 March 2024).
23. The Clustered Orienteering Problem. Available online: http://or-brescia.unibs.it/ (accessed on 14 March 2024).
24. TSPLIB95. Available online: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95 (accessed on 14 March 2024).
25. IBM. IBM CPLEX Optimizer. 2024. Available online: https://www.ibm.com/de-de/analytics/cplex-optimizer (accessed on 14 March 2024).
26. Pardalos, P. The maximum clique problem. *J. Glob. Optim.* **1994**, *4*, 301–328. [CrossRef]