New Benchmark Instances for the Capacitated Vehicle Routing Problem

Eduardo Uchoa*[1], Diego Pecin[2], Artur Pessoa[1], Marcus Poggi[2], Anand Subramanian[3], Thibaut Vidal[2]

[1] Universidade Federal Fluminense, Departamento de Engenharia de Produção, Brazil

[2] Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, Brazil

[3] Universidade Federal da Paraíba, Departamento de Engenharia de Produção, Brazil

Abstract. The recent research on the CVRP is being slowed down by the lack of a good set of benchmark instances. The existing sets suffer from at least one of the following drawbacks: (i) became too easy for current algorithms; (ii) are too artificial; (iii) are too homogeneous, not covering the wide range of characteristics found in real applications. We propose a new set of instances ranging from 100 to 1000 customers, designed in order to provide a more comprehensive and balanced experimental setting. We report results with state-of-the-art exact and heuristic methods.

Keywords. Vehicle Routing Problem, Benchmark Instances, Experimental Analysis of Algorithms

* Corresponding author

1 Introduction

The Vehicle Routing Problem (VRP) is among the most widely studied problems in the fields of operations research and combinatorial optimization. Its relevance stems from its direct application in the real world systems that distribute goods and provide services, vital to the modern economies. Reflecting the large variety of conditions present in those systems, the VRP literature is spread into dozens of variants. For example, there are variants that consider time windows, multiple depots, mixed vehicle fleet, split delivery, pickups and deliveries, precedences, complex loading constraints, etc.

In the vast landscape of VRP variants, the Capacitated VRP (CVRP) occupies a central position. It was defined by Dantzig and Ramser [12] as follows. The input consists of a set of $n + 1$ points, a *depot* and n *customers*; an $(n + 1) \times (n + 1)$ matrix $d = [d_{ij}]$ with the *distances* between every pair of points i and j; an n-dimensional *demand* vector $q = [q_i]$ giving the amount to be delivered to customer i; and a vehicle *capacity* Q. A solution is a set of routes, starting and ending at the depot, that visit every customer exactly once in such a way that the sum of the demands of the customers of each route does not exceed the vehicle capacity. The objective is to find a solution with minimum total route distance. Some authors also assume that the *number of routes* is fixed to an additional input number K (almost always defined as the minimum possible number of routes, K_{min}).

The CVRP plays a particular role on VRP algorithmic research, for both exact and on heuristics methods. Being the most basic variant, it is a natural testbed for trying new ideas. Its relative simplicity allows cleaner descriptions and implementations, without the additional conceptual burden necessary to handle more complex variants. Successful ideas for the CVRP are often later extended to more complex variants. For example, the classical CVRP heuristic by Clarke and Wright [9] was adapted for the VRP with Time Windows [29] and for many other variants, as surveyed in Rand [25].

The present work is motivated by the claim that the recent research on the CVRP is being hampered by the lack of a well-established set of benchmark instances able to push the limits of the state-of-the-art algorithms. Let us examine this claim:

- Since Augerat et al. [3], nearly all new exact methods for the CVRP have been tested over instances from six different classes. Classes A, B and P were proposed in [3], while classes E, F and M were proposed by Christofides and Eilon [7], by Fisher [13] and by Christofides et al. [8], respectively. Nearly all those instances are Euclidean. In the literature of exact methods it became usual to follow the TSPLIB convention of rounding distances to the nearest integer [26] and also to fix the number of routes.

The ABEFMP instances provided a good benchmark until the mid 2000's decade. Although 90 out of its 95 instances have no more than 100 customers, several such instances were still very challenging for the branch-and-cut algorithms (like [1, 2, 3, 6, 20, 24, 34]) which prevailed at that time. Then, the branch-cut-and-price of Fukasawa et al. [14] solved all its instances with up to 100 customers, as well as instances M-n121-k7 and F-n135-k7 (the later was already solved since [3]). Only instances M-n151-k12, M-n200-k16 and M-n200-k17 remained unsolved. From that moment, the benchmark was not satisfactory anymore. While the majority of its instances became quite easy, the now more interesting range of 101 to 200 customers was very thinly populated. In particular, the column and cut generation algorithms of Baldacci et al. [4, 5] significantly reduced the CPU time required to solve many ABEFMP instances, but could not solve the three larger M instances. Later, the algorithms of Contardo and Martinelli [10] and Røpke [28] were capable of solving M-n151-k12. Very recently, the algorithm of Pecin et al. [22] solved the last two open instances in the benchmark.

- Since the 1980's, almost all articles proposing new heuristic and metaheuristic methods for the CVRP reported results on a subset of the instances by Christofides and Eilon [7] and Christofides et al. [8]. In this literature, it is usual to follow the convention of not rounding the Euclidean distances and not to fix the number of routes. This classical benchmark is also exhausted, since most recent heuristics find systematically the best known solutions on nearly all instances. In fact, Rochat and Taillard [27] already published in 1995 what we now know to be the optimal solutions, except on a single instance with 199 customers, where the reported solution was only 0.012% off-optimal.

The more recent set of instances by Golden et al. [17] is now the second most frequently used benchmark. Having larger instances, ranging from 240 to 483 customers, it still has a good discriminating power. In fact, the heuristics of Nagata and Bräysy [21] and Vidal et al. [32] are considered the best available for the CVRP basically due to their superior performance on Golden's instances. Nevertheless, we believe that those instances are not sufficient for a good benchmark. A first drawback is their artificiality. In all instances the customers are positioned in concentric geometric figures, either circles, squares or six-pointed stars. The demands also follow very symmetric patterns. As a result, the solution space is partitioned into groups of equivalent solutions obtainable by rotations and flippings around several axis of symmetry. The second drawback is their relative homogeneity. For example, there are no instances with clusters of customers.

This work proposes a new set of 100 instances, ranging from 100 to 1000 customers, intended to be used by both exact and heuristic methods in the next years. They were designed in order to provide a more comprehensive and balanced experimental setting.

The remainder of the paper is organized as follows. Section 2 summarizes the current CVRP benchmark instances. Section 3 explains the procedure developed for generating new CVRP instances. Section 4 contains the results found by state-of-the-art heuristic and exact methods for the new set of instances. Section 5 briefly describes the features of the new CVRPLIB web site where all instances mentioned in this paper are available. Finally, Section 6 presents the concluding remarks of this work.

2 Current Benchmark Instances

In this section we provide detailed information about the existing benchmark instances. In particular, we tried to track back the generation process of all these instances.

Table 1 presents data regarding the E series. Although they are usually attributed to Christofides and Eilon [7], some instances actually come from Dantzig and Ramser [12] and from Gaskell [15], and some are modifications later suggested by Gillett and Miller [16]. As usual in the literature on exact methods, the naming of the instances reflects the number of points (including the depot) and the fixed number of routes. For example, E-n101-k8 is an instance with 100 customers and a requirement of 8 routes. Columns in Table 1 include the value of Q, the tightness (the ratio between the sum of all demands and KQ, the total capacity available in the fixed number of routes) and the optimal solution value.

Table 2 presents information about the M series and explains how each instance was generated. Instances M-n200-k16 and M-n200-k17 only differ by the required number of routes. In fact, M-n200-k17 is the only ABEFMP instance where the fixed number of routes does not match the minimum possible. This additional instance was created because M-n200-k16 has a tightness so close to 1 (0.995625) that finding good feasible solutions for it was very difficult. Surprisingly, it was recently discovered that the optimal solution of M-n200-k16 costs less than the optimal solution of M-n200-k17 [22].

Table 3 presents the three real-world instances that compose the F series. Tables 4 and 5 correspond to the A and B series, respectively. While in the A series the customers and the depot are randomly positioned; they are clustered in the B series. The instances from series P were generated by taking some instances from the A, B and E series and changing their capacities. Consequently, the required number of routes also changes. For example, P-n101-k4 was obtained from E-n101-k8 by doubling the capacity.

Christofides et al. [8] defined the benchmark set shown in Table 7. Instances CMT1, CMT2, CMT3, CMT, CMT5, CMT11 , and CMT12 correspond to instances E-n51-k5, E-n76-k10, E-n101-k8, M-n151-k12, M-n200-k16, M-n121-k7, and M-n100-k10, respectively. The only difference are the conventions: the euclidean distances are represented with full computer precision (without rounding), and the number of routes is not fixed. This set also contains instances for the duration constrained CVRP, obtained from the previous CVRP instances by adding maximum route duration (MD) and service time (ST) values. These values are also reported in Table 7. Column K_{BKS} gives the number of routes in the optimal/best known solution of each instance. Optimal solutions are marked with a *.

Table 8 corresponds to the benchmark proposed in Golden et al. [17]. There are 12 CVRP instances and 8 instances for Duration-constrained CVRP, the maximum durations (as there are no service times, this is equivalent to a bound on the maximum total distance traveled in a route) are given in column (MD). Table 8 also gives the geometric patterns used for positioning the customers in each instance.

Table 9 corresponds to a benchmark proposed in Rochat and Taillard [27]. Twelve instances from 75 to 150 customers were generated using a scheme where the depot is always in the center, the customers are clustered and demands are taken from an exponential distribution. An additional instance with 385 customers, obtained from real-world data, already appeared in [31]. Note that a best value of 2341.84 for instance tai150c was mentioned on some benchmark instance repositories and then relayed in some papers. Yet, the exact algorithm of [22] determined that a solution of 2358.66 is optimal and most recent heuristics found the same value. We thus assume that this previous solution was erroneous.

Although there are no pure CVRP instances in this set, for the sake of completeness, Table 10 presents a benchmark proposed in Li et al. [19], composed by 12 larger scale instances of the Duration-constrained CVRP. Typical CVRP heuristics can be easily adapted to handle duration/distance constraints, and some articles on the CVRP also reported results in the Li benchmark. However, most recent state-of-the-art methods for the CVRP [21, 32] did not. Therefore, in order to present solutions that correspond to the current point of algorithmic evolution, we performed runs with the algorithm of [32]. Table 10 has been updated to include the newly found best known solutions. The detailed computational results are reported in the Appendix. We also highlight an issue related to the published solution of instance pr32, generated by a manual process in [19] with a distance value of 36919.24. This solution seems to have 10 routes from the figure in the paper, and as such cannot comply with the distance limit of 3600 units. For this reason, it was not included in the table.

3 Newly Proposed Instances

This section describes how the instances in the proposed CVRP benchmark were generated. As happens in almost all the existing instances, the distances are two-dimensional Euclidean. Depot and costumers have integer coordinates corresponding to points in a $[0, 1000] \times [0, 1000]$ grid. Each instance is characterized by the following attributes: number of customers, depot positioning, customer positioning, demand distribution, and average route size. The possible values of each attribute and their effect in the generation are described in the next subsection.

3.1 Instance Attributes

3.1.1 Depot Positioning

Three different positions for the depot are considered:

Central (C) – depot in the center of the grid, point (500,500).

Eccentric (E) – depot in the corner of the grid, point (0,0).

Random (R) – depot in a random point of the grid.

The TC, TE and TR instances [18], used as benchmark on rooted network design problems, present similar alternatives for root positioning.

3.1.2 Customer Positioning

Three alternatives for customer positioning are considered, following the R, C and RC instance classes of the Solomon set for the VRPTW [29].

Random (R) – All customers are positioned in random points of the grid.

Clustered (C) – At first, a number S of customers that will act as cluster seeds is picked from an uniform discrete distribution UD[3,8]. Next, the S seeds are randomly positioned in the grid. The seeds will then attract, with an exponential decay, the

Table 1: Instances of the set E

Instance	Q	Tightness	Opt	Original Source
E-n13-k4[1]	6000	0.76	247	Dantzig & Ramser (1959)
E-n22-k4[1]	6000	0.94	375	Gaskell (1967)
E-n23-k3[1]	4500	0.75	569	Gaskell (1967)
E-n30-k3[1]	4500	0.94	534	Gaskell (1967)
E-n31-k7[2]	140	0.92	379	Clarke & Wright (1964)
E-n33-k4[1]	8000	0.92	835	Gaskell (1967)
E-n51-k5[3]	160	0.97	521	Christofides & Eilon (1969)
E-n76-k7[4]	220	0.89	682	Gillett & Miller (1974)
E-n76-k8[4]	180	0.95	735	Gillett & Miller (1974)
E-n76-k10[3]	140	0.97	830	Christofides & Eilon (1969)
E-n76-k14[4]	100	0.97	1021	Gillett & Miller (1974)
E-n101-k8[3]	200	0.91	815	Christofides & Eilon (1969)
E-n101-k14[5]	112	0.93	1067	Gillett & Miller (1974)

[1] No description about the generation

[2] Example involving UK cities with customers located far from from the depot

[3] Locations generated at random from an uniform distribution

[4] Instance E-n76-k10 with modified capacity

[5] Instance E-n101-k8 with modified capacity

Table 2: Instances of the set M, original source: Christofides et al. (1979)

Instance	Q	Tightness	Opt
M-n101-k10[1]	200	0.91	820
M-n121-k7[1]	200	0.98	1034
M-n151-k12[2]	200	0.93	1015
M-n200-k16[3]	200	1.00	1274
M-n200-k17[3]	200	0.94	1275

[1] Customers were grouped into clusters as an attempt to represent pratical cases

[2] Generated by adding customers from E-n51-k5 and E-n101-k8 and using the depot and capacity from E-n101-k8

[3] Generated by adding customers from M-n151-k12 and the first 49 customers from E-n76-k10 and using the depot and capacity from M-n151-k12

Table 3: Instances of the set F, original source: Fisher (1994)

Instance	Q	Tightness	Opt
F-n45-k4[1]	2010	0.90	724
F-n72-k4[2]	30000	0.96	237
F-n135-k7[1]	2210	0.95	1162

[1] From a day of grocery deliveries from the Peterboro (Ontario terminal) of National Grocers Limited

[2] Data obtained from Exxon associated to the delivery of tires, batteries and accessories to gasoline service stations

Table 4: Instances of the set A, original source: Augerat (1995)[1]

Instance	Q	**Tightness**	**Opt**
A-n32-k5	100	0.82	784
A-n33-k5	100	0.89	661
A-n33-k6	100	0.90	742
A-n34-k5	100	0.92	778
A-n36-k5	100	0.88	799
A-n37-k5	100	0.81	669
A-n37-k6	100	0.95	949
A-n38-k5	100	0.96	730
A-n39-k5	100	0.95	822
A-n39-k6	100	0.88	831
A-n44-k6	100	0.95	937
A-n45-k6	100	0.99	944
A-n45-k7	100	0.91	1146
A-n46-k7	100	0.86	914
A-n48-k7	100	0.89	1073
A-n53-k7	100	0.95	1010
A-n54-k7	100	0.96	1167
A-n55-k9	100	0.93	1073
A-n60-k9	100	0.92	1354
A-n61-k9	100	0.98	1034
A-n62-k8	100	0.92	1288
A-n63-k9	100	0.97	1616
A-n63-k10	100	0.93	1314
A-n64-k9	100	0.94	1401
A-n65-k9	100	0.97	1174
A-n69-k9	100	0.94	1159
A-n80-k10	100	0.94	1763

[1] Coordinates are random points in a $[0,100] \times [0,100]$ grid.
Demands are picked from an uniform distribution U(1,30), however $n/10$ of those demands are multiplied by 3.

Table 5: Instances of the set B, original source: Augerat (1995)[1]

Instance	Q	**Tightness**	**Opt**
B-n31-k5	100	0.82	672
B-n34-k5	100	0.91	788
B-n35-k5	100	0.87	955
B-n38-k6	100	0.85	805
B-n39-k5	100	0.88	549
B-n41-k6	100	0.95	829
B-n43-k6	100	0.87	742
B-n44-k7	100	0.92	909
B-n45-k5	100	0.97	751
B-n45-k6	100	0.99	678
B-n50-k7	100	0.87	741
B-n50-k8	100	0.92	1312
B-n51-k7	100	0.98	1032
B-n52-k7	100	0.87	747
B-n56-k7	100	0.88	707
B-n57-k7	100	1.00	1153
B-n57-k9	100	0.89	1598
B-n63-k10	100	0.92	1496
B-n64-k9	100	0.98	861
B-n66-k9	100	0.96	1316
B-n67-k10	100	0.91	1032
B-n68-k9	100	0.93	1272
B-n78-k10	100	0.94	1221

[1] Coordinates are points in a $[0, 100] \times [0, 100]$ grid, chosen in order to create NC clusters.
In all instances, $K \leq NC - 1$.
Demands are picked from an uniform distribution U(1,30), however $n/10$ of those demands are multiplied by 3.

Table 6: Instances of the set P, original source: Augerat (1995)[1]

Instance	Q	**Tightness**	**Opt**
P-n16-k8	35	0.88	450
P-n19-k2	160	0.97	212
P-n20-k2	160	0.97	216
P-n21-k2	160	0.93	211
P-n22-k2	160	0.96	216
P-n22-k8	3000	0.94	603
P-n23-k8	40	0.98	529
P-n40-k5	140	0.88	458
P-n45-k5	150	0.92	510
P-n50-k7	150	0.91	554
P-n50-k8	120	0.99	631
P-n50-k10	100	0.95	696
P-n51-k10	80	0.97	741
P-n55-k7	170	0.88	568
P-n55-k8	160	0.81	588
P-n55-k10	115	0.91	694
P-n55-k15	70	0.99	989
P-n60-k10	120	0.95	744
P-n60-k15	80	0.95	968
P-n65-k10	130	0.94	792
P-n70-k10	135	0.97	827
P-n76-k4	350	0.97	593
P-n76-k5	280	0.97	627
P-n101-k4	400	0.91	681

[1] Modifications in the capacity of some instances from A, B and E series. Required number of routes are adjusted accordingly.

Table 7: Instances of Christofides et al. [8]

Instance	n	Q	**MD**	**ST**	**BKS**	K_{BKS}
CMT1[1]	50	160	∞	0	524.61*	5
CMT2[1]	75	140	∞	0	835.26*	10
CMT3[1]	100	200	∞	0	826.14*	8
CMT4[1]	150	200	∞	0	1028.42*	12
CMT5[1]	199	200	∞	0	1291.29*	16
CMT11[1]	120	200	∞	0	1042.11*	7
CMT12[1]	100	200	∞	0	819.56*	10
CMT6[2]	50	160	200	10	555.43	6
CMT7[2]	75	140	160	10	909.68	11
CMT8[2]	100	200	230	10	865.94	9
CMT9[2]	150	200	200	10	1162.55	14
CMT10[2]	199	200	200	10	1395.85	18
CMT13[2]	120	200	720	50	1541.14	11
CMT14[2]	100	200	1040	90	866.37	11

[1] The data of instances CMT1, CMT2, CMT3, CMT12, CMT11, CMT4 and CMT5 are the same of E-n51-k5, E-n76-k10, E-n101-k8, M-n101-k10, M-n121-k7, M-n151-k12 and M-n200-k16(k17), respectively. The only difference is the convention of not rounding the costs and not fixing the number of routes.

[2] Instances CMT6, CMT7, CMT8, CMT14, CMT13, CMT9 and CMT10 were generated by adding maximum route duration and service time to CMT1, CMT2, CMT3, CMT12, CMT11, CMT4 and CMT5, respectively. Vehicles are assumed to travel at unitary speed.

other $n - S$ customers : the probability for a point p in the grid to receive a customer is proportional to

$$\sum_{s=1}^{S} \exp(-d(p,s)/40),$$

where $d(p,s)$ is the distance between p and seed s. The divisor 40 in the above formula was chosen after a number of experiments. Smaller values lead to excessively dense and isolated clusters. On the other hand, larger divisors result in clusters that are too sparse and mingled. When two or more seeds happen to be close, their combined attraction is likely to form a single larger cluster around them. This means that the size of the clusters may differ significantly. When two seeds are a little more apart, their clusters will not coalesce but a “bridge” of customers between them may appear. A seed that is sufficiently apart from other seeds will form an isolated cluster. The overall clustering scheme was devised to mimic the densities found in some large urban agglomerations that have grown from more or less isolated original nucleus (the seeds).

Table 8: Instances of Golden et al. [17]

Instance	n	Q	**MD**	**BKS**	K_{BKS}
G1[4]	240	550	650	5623.47	9
G2[4]	320	700	900	8404.61	10
G3[4]	400	900	1200	11036.22	10
G4[4]	480	1000	1600	13590.00	–
G5[5]	200	900	1800	6460.98	5
G6[5]	280	900	1500	8400.33	–
G7[5]	360	900	1300	10102.70	8
G8[5]	440	900	1200	11635.30	10
G9[3]	255	1000	∞	579.71	14
G10[3]	323	1000	∞	735.66	–
G11[3]	399	1000	∞	912.03	–
G12[3]	483	1000	∞	1101.50	–
G13[2]	252	1000	∞	857.19	26
G14[2]	320	1000	∞	1080.55*	30
G15[2]	396	1000	∞	1337.87	–
G16[2]	480	1000	∞	1611.56	–
G17[1]	240	200	∞	707.76*	22
G18[1]	300	200	∞	995.13*	27
G19[1]	360	200	∞	1365.60*	33
G20[1]	420	200	∞	1817.89	–

[1] Concentric six-pointed pointed stars
[2] Concentric squares, depot in the center
[3] Concentric squares, depot in a corner
[4] Concentric circles
[5] Concentric rays

Table 9: Instances of Rochat and Taillard [27]

Instance	n	Q	**BKS**	K_{BKS}
tai75a[1]	75	1445	1618.36*	10
tai75b[1]	75	1679	1344.64*	9
tai75c[1]	75	1122	1291.01*	9
tai75d[1]	75	1699	1365.42*	9
tai100a[1]	100	1409	2041.34*	11
tai100b[1]	100	1842	1939.90*	11
tai100c[1]	100	2043	1406.20*	11
tai100d[1]	100	1297	1580.46*	11
tai150a[1]	150	1544	3055.23*	15
tai150b[1]	150	1918	2727.03*	14
tai150c[1]	150	2021	2358.66*	15
tai150d[1]	150	1874	2645.40*	14
tai385[2]	385	65	24366.41	47

[1] Customers are non-uniformally spread in several clusters.
The number of clusters and their compactness are variable.
Demands are generated following a exponential distribution.
[2] The data for the customers were generated based on real information from the canton of Vaud in Switzerland. Already appeared in [31].

Table 10: Instances of Li et al. [19][1]

Instance	n	Q	**MD**	**BKS**	K_{BKS}
21	560	1200	1800	16212.74	–
22	600	900	1000	14499.04	15
23	640	1400	2200	18801.12	10
24	720	1500	2400	21389.33	–
25	760	900	900	16668.51	19
26	800	1700	2500	23971.74	–
27	840	900	900	17343.38	20
28	880	1800	2800	26565.92	–
29	960	2000	3000	29154.33	–
30	1040	2100	3200	31742.51	–
31	1120	2300	3500	34330.84	–
32	1200	2500	3600	37159.41	11

Random-Clustered (RC) – Half of the customers are clustered by the above described scheme, the remaining customers are randomly positioned.

It should be noted that superpositions are not allowed, all customers and the depot are located in distinct points on the grid.

3.1.3 Demand Distribution

Seven options of demand distributions have been selected in these instances.

Unitary (U) – All demands have value 1.

Small Values, Large Variance (1-10) – demands from UD[1,10].

Small Values, Small Variance (5-10) – demands from UD[5,10].

Large Values, Large Variance (1-100) – demands from UD[1,100].

Large Values, Small Variance (50-100) – demands from UD[50,100].

Depending on Quadrant (Q) – demands taken from UD[1,50] if customer is in an even quadrant (with respect to point (500,500)), and from UD[51,100] otherwise. This kind of demand distribution leads to solutions containing some routes that are significantly longer (in terms of number of served customers) than others.

Many Small Values, Few Large Values (SL) – Most demands (70% to 95% of the customers) are taken from UD[1,10], the remaining demands are taken from UD[50,100].

3.1.4 Average Route Size

The previous experience of the authors with CVRP algorithms indicated that the value n/K_{min}, the average route size (assuming solutions with the minimum possible number of routes) has a large impact on the performance of current exact methods. The impact of this attribute on heuristic methods is not so pronounced, but is still quite significant. We can not make a general statement that instances with shorter routes are easier than instances with longer routes or the opposite. What happens is that some methods are more suited for short routes and other methods for longer routes. Therefore, a comprehensive benchmark set should contain instances where this attribute vary over a wide range of values. Yet, generating an instance with exactly a given value of n/K_{min} would be difficult, since even computing K_{min} requires the solution of a bin-packing problem. The generator actually uses as attribute a value r representing the *desired* value of n/K_{min}. This causes the instance capacity to be defined as:

$$Q = \lceil \frac{r \sum_{i=1}^{n} q_i}{n} \rceil,$$

where the q vector represents the demands, already obtained according to the specified demand distribution. As $\sum_{i=1}^{n} q_i/Q$ is usually a good lower bound on K_{min}, instances have values of n/K_{min} that are sufficiently close to r.

3.2 Instance Generation

A benchmark set with instances corresponding to the cartesian product of so many parameter values (even restricting the "continuous" parameters n and r to a reasonably small number of values) would be huge. Thus, we generated a sample of 100 instances, presented in Tables 11 to 13. We believe that this number of instances is large enough to obtain the desired level of diversification, but still small enough to allow that future users of the benchmark can report detailed results for each instance. We now explain how the attribute values of those 100 instances were obtained.

- The instances are ordered by the number of customers n. For the first 50 instances, ranging from 100 to 330 customers, n is increased in linear steps. For the last 50 instances, n is increased in exponential steps from 335 to 1000. For the time being, instances with around 200 customers can already be hard for exact methods, and larger instances are sufficiently challenging to highlight significant quality differences between competing heuristics.

- The set of values of r were taken from a continuous triangular distribution T(3,6,25) (minimum 3, mode 6 and maximum 25). The 100 values of r were partitioned in quintiles, corresponding to very small routes, small routes, medium routes, long routes and very long routes. The k-th instance receives an r from the $((k-1) \bmod 5)+1$ quintile. In this way, it is guaranteed that every set containing $5t$ instances with consecutive values of n will have exactly t instances from each quintile. It can be observed in the end of Table 13 that the resulting set of n/K_{min} values have minimum 3.0, maximum 24.4, median 9.8 and average 11.1.

- A random permutation of the 3 possible values for the depot positioning attribute (C, E and R) provides the values for the first 3 instances. Another random permutation gives the values for the next 3 instances and so on. A similar scheme is used for obtaining the values for the customer positioning and demand distribution attributes. This guarantees that every subset of the instances with consecutive values of n will have a near-balanced number of instances having the same value of an attribute.

The name of an instance follows the ABEFMP standard, and has a format X-nA-kB, where A represents $n+1$, the number of points in the instance including the depot, and B is the minimum possible number of routes K_{min}, calculated by solving a bin-packing problem. Since there are no two instances with the same number of points, in contexts where the K_{min} information is not considered much important, we propose the format XA as an alternative shorter name. For example, the shorter name for X-n284-k15 would be X284.

3.3 Two Decisions on Conventions

3.3.1 Rounding the Distances or Not?

Following the TSPLIB convention [26], the euclidean distances are rounded to the nearest integer in the literature on exact methods. On the other hand, the distances are seldom rounded in the literature on heuristics.

Advantages of Rounding – Most mathematical programming based algorithms (including standard MIP solvers) have a limited optimality precision. For example, CPLEX 12.5 default precision is only 10^{-4} (0.01%). It is possible to increase the precision up to a point by adjusting parameters (say, 10^{-6} or 10^{-7}). Going further requires special software, using more bits in the floating-point numbers or even exact rational arithmetic [11], that is not easily available or implementable. The practice of distance rounding is convenient for avoiding those pitfalls and usually makes the optimal values found by exact methods based on standard mathematical programming software reliable. Remark that the practice of not rounding, but only reporting two decimal places does not solve the problem. For example, an algorithm with a precision of 10^{-6} may declare a solution having value 853.2351 (published as 853.24) as optimal. Later, someone finds a solution with value 853.2349 and publishes 853.23.

Disavantages of Rounding – A benchmark of rounded instances has less power for comparing competing algorithms. Especially for heuristics, the search space is formed by a relatively small number of plateaus, i.e., sets of solutions with the same value. Guiding the search on such plateaus is usually done by trial and error since there is no indication that the distance is –even slightly– reduced. There may also be in practice several distinct optimal solutions, leading to more frequent ties between competitors. This effect is quite significant on ABEFMP instances, where the optimal solution values have magnitudes around 10^3. In addition, some people claim (but never publish, this is part of the community folklore) that rounding can artificially enhance the performance of exact methods. For example, if an upper bound of 1000 is known, a branch-and-bound node with lower bound $999 + \epsilon$ can be fathomed. This effect is significant on ABEFMP instances, enough to make some algorithms to run at least twice as fast.

We took the decision that the newly proposed benchmark set will follow the TSPLIB convention of rounding distances. Nevertheless, the instances were devised in order to minimize the above mentioned disadvantages. The use of a $[0, 1000] \times [0, 1000]$ grid (instead of the $[0, 100] \times [0, 100]$ grids of most ABEFMP instances) makes the optimal solutions to have magnitudes between 10^4-10^5. This is still quite safe for exact methods having a precision of 10^{-6}. However, the plateau effect and the artificial enhancement of exact methods are much reduced.

3.3.2 Fix the Number of Routes or Not?

The literature on exact methods usually follows the convention of fixing the number of routes to a value K. Except for instance M-n200-k17, K is always set to K_{min}. The standard explanation for that fixing is that K represents the number of vehicles available at the depot. A more sophisticated explanation is that fixing the number of routes to the minimum is an indirect way of minimizing the fixed costs for using a vehicle. We do not think that this is necessarily true : since the CVRP definition allows solutions containing routes that are much shorter than others, why not assigning the two shortest routes to the same vehicle? In fact, the CVRP may be used to model real-world situations where a single vehicle (a truly homogeneous fleet!) will perform all routes in sequence. In other words, CVRP routes do not need to correspond to vehicles. Based on that reasoning, *we decided that the newly proposed benchmark set will follow the convention of not fixing the number of routes.* Therefore, the number K_{min} indicated in each instance should be taken only as a lower bound on the number of routes in a solution. A second reason for taking that decision is that the number of routes was not fixed in the original CVRP definition [12].

4 Experiments with State-of-the-Art Methods

In order to provide an initial set of results to the new benchmark, experiments have been conducted with recent state-of-the-art heuristic and exact methods. The experiments also provide an assessment of the level of difficulty of the proposed instances and even some hints on how the attributes used in their generation impact on that difficulty. Those results are reported jointly in Tables 11 to 13, which describe the characteristics of each instance, the average solution quality, best solutions quality and average CPU time of two state-of-the-art metaheuristics, the lower bound, root CPU time, number of search nodes and overall time of a recent state-of-the-art exact method, and finally the cost and number of vehicles of the best solution (BKS) ever found since the instances were created, including preliminary runs of those heuristics with alternative parameterizations. Table 13 also reports additional statistics on instances characteristics and results, such as the minimum, maximum, average and median results on the instance set for n, Q, r, n/K_{min}, as well as for the Gaps(%) and CPU time of ILS-SP, UHGS and BCP (root relaxation).

4.1 Heuristic solutions

We selected a pair of recent successful metaheuristics to illustrate the two main current types of approaches in the literature. We consider an efficient neighborhood-based method, the iterated local search based matheuristic algorithm (ILS-SP) of Subramanian et al. [30], and a recent population-based method, the unified hybrid genetic search (UHGS) of Vidal et al. [32, 33]. These two methods rely extensively on local search to improve new solutions generated either by shaking or recombinations of parents. They also include specific strategies to explore new choices of customer-to-route assignments: ILS is coupled with a integer programming solver over a set partitioning (SP) formulation, which seeks to create new solutions based on known routes from past local optimums, while UHGS implements

Table 11: New set of benchmark instances : characteristics and results of current state-of-the-art algorithms (Part I)

		Instance Characteristics							ILS-SP			UHGS			BCP				BKS	
#	Name	n	Dep	Cust	Dem	Q	r	n/K_{min}	Avg	Best	T(min)	Avg	Best	T(min)	RLB	RT(min)	Nds	T(min)	Value	NV
1	X-n101-k25	100	R	RC (7)	1-100	206	4.0	4.0	27591.0	27591	0.13	27591.0	27591	1.43	27591	0.1	1	0.1	27591	26
2	X-n106-k14	105	E	C (3)	50-100	600	8.0	7.5	26375.9	26362	2.01	26381.8	26378	4.04	26362	3.5	1	3.5	26362	14
3	X-n110-k13	109	C	R	5-10	66	8.8	8.4	14971.0	14971	0.20	14971.0	14971	1.58	14971	0.3	1	0.3	14971	13
4	X-n115-k10	114	C	R	SL	169	12.5	11.4	12747.0	12747	0.18	12747.0	12747	1.81	12747	2.1	1	2.1	12747	10
5	X-n120-k6	119	E	RC (8)	U	21	21.8	19.8	13337.6	13332	1.69	13332.0	13332	2.31	13234	2.2	63	88.1	13332	6
6	X-n125-k30	124	R	C (5)	Q	188	4.2	4.1	55673.8	55539	1.43	55542.1	55539	2.66	55539	2.5	1	2.5	55539	30
7	X-n129-k18	128	E	RC (8)	1-10	39	7.4	7.1	28998.0	28948	1.92	28948.5	28940	2.71	28897	1.3	3	2.5	28940	18
8	X-n134-k13	133	R	C (4)	Q	643	10.4	10.2	10947.4	10916	2.07	10934.9	10916	3.32	10840	8.3	2955	399.1	10916	13
9	X-n139-k10	138	C	R	5-10	106	14.0	13.8	13603.1	13590	1.60	13590.0	13590	2.28	13590	17.0	1	17.0	13590	10
10	X-n143-k7	142	E	R	1-100	1190	22.6	20.3	15745.2	15726	1.64	15700.2	15700	3.10	15634	20.9	1825	1553	15700	7
11	X-n148-k46	147	R	RC (7)	1-10	18	3.2	3.2	43452.1	43448	0.84	43448.0	43448	3.18	43448	0.3	1	0.3	43448	47
12	X-n153-k22	152	C	C (3)	SL	144	7.1	6.9	21400.0	21340	0.49	21226.3	21220	5.47	21140	4.7	143	37.7	21220	23
13	X-n157-k13	156	R	C (3)	U	12	12.0	12.0	16876.0	16876	0.76	16876.0	16876	3.19	16876	1.0	1	1.0	16876	13
14	X-n162-k11	161	C	RC (8)	50-100	1174	15.5	14.6	14160.1	14138	0.54	14141.3	14138	3.32	14053	35.6	101	187.0	14138	11
15	X-n167-k10	166	E	R	5-10	133	17.8	16.6	20608.7	20562	0.86	20563.2	20557	3.73	20476	8.5	85	1024	20557	10
16	X-n172-k51	171	C	RC (5)	Q	161	3.4	3.4	45616.1	45607	0.64	45607.0	45607	3.83	45549	1.5	3	3.8	45607	53
17	X-n176-k26	175	E	R	SL	142	6.8	6.7	48249.8	48140	1.11	47957.2	47812	7.56	47721	2.4	43	9.2	47812	26
18	X-n181-k23	180	R	C (6)	U	8	8.4	7.8	25571.5	25569	1.59	25591.1	25569	6.28	25511	1.9	73	18.2	25569	23
19	X-n186-k15	185	R	R	50-100	974	13.0	12.3	24186.0	24145	1.72	24147.2	24145	5.92	23980	26.8	1211	7305	24145	15
20	X-n190-k8	189	E	C (3)	1-10	138	25.0	23.6	17143.1	17085	2.10	16987.9	16980	12.08	16939	22.8			16980	8
21	X-n195-k51	194	C	RC (5)	1-100	181	3.8	3.8	44234.3	44225	0.87	44244.1	44225	6.10	44225	2.4	1	2.4	44225	53
22	X-n200-k36	199	R	C (8)	Q	402	5.6	5.5	58697.2	58626	7.48	58626.4	58578	7.97	58455	1.7	1469	901.7	58578	36
23	X-n204-k19	203	C	RC (6)	50-100	836	11.2	10.7	19625.2	19570	1.08	19571.5	19565	5.35	19484	75.4	85	501.6	19565	19
24	X-n209-k16	208	E	R	5-10	101	13.5	13.0	30765.4	30667	3.80	30680.4	30656	8.62	30480	3.2	603	1303	30656	16
25	X-n214-k11	213	C	C (4)	1-100	944	19.4	19.4	11126.9	10985	2.26	10877.4	10856	10.22	10809	264.9			10856	11
26	X-n219-k73	218	E	R	U	3	3.6	3.0	117595.0	117595	0.85	117604.9	117595	7.73	117595	0.5	1	0.5	117595	73
27	X-n223-k34	222	R	RC (5)	1-10	37	6.5	6.5	40533.5	40471	8.48	40499.0	40437	8.26	40311	13.2	343	303.5	40437	34
28	X-n228-k23	227	R	C (8)	SL	154	10.0	9.9	25795.8	25743	2.40	25779.3	25742	9.80	25657	15.0	163	252.3	25742	23
29	X-n233-k16	232	C	RC (7)	Q	631	14.5	14.5	19336.7	19266	3.01	19288.4	19230	6.84	19070	133.0			19230	17
30	X-n237-k14	236	E	R	U	18	18.6	16.9	27078.8	27042	3.46	27067.3	27042	8.90	26930	2.0	2695	1398	27042	14
31	X-n242-k48	241	E	R	1-10	28	5.0	5.0	82874.2	82774	17.83	82948.7	82804	12.42	82589	2.0	2661	819.0	82751	48
32	X-n247-k50	246	C	C (4)	SL	134	5.3	4.9	37507.2	37289	2.06	37284.4	37274	20.41	37256	3.7	7	9.5	37274	51
33	X-n251-k28	250	R	RC (3)	5-10	69	9.2	8.9	38840.0	38727	10.77	38796.4	38699	11.69	38473	2.1	2531	4767	38684	28
34	X-n256-k16	255	C	C (8)	50-100	1225	16.0	15.9	18883.9	18880	2.02	18880.0	18880	6.52	18826	201.5	25	1255	18880	17
35	X-n261-k13	260	E	R	1-100	1081	21.0	20.0	26869.0	26706	6.67	26629.6	26558	12.67	26407	373.7			26558	13

Table 12: New set of benchmark instances : characteristics and results of current state-of-the-art algorithms (Part II)

		Instance Characteristics							**ILS-SP**			**UHGS**			**BCP**				**BKS**	
#	Name	n	Dep	Cust	Dem	Q	r	n/K_{min}	Avg	Best	T(min)	Avg	Best	T(min)	RLB	RT(min)	Nds	T(min)	Value	NV
36	X-n266-k58	265	R	RC (6)	5-10	35	4.6	4.6	75563.3	75478	10.03	75759.3	75517	21.36	75350	9.5	185	150.1	75478	58
37	X-n270-k35	269	C	RC (5)	50-100	585	7.7	7.7	35363.4	35324	9.07	35367.2	35303	11.25	35156	14.5	389	3422	35291	36
38	X-n275-k28	274	R	C (3)	U	10	10.8	9.8	21256.0	21245	3.59	21280.6	21245	12.04	21245	3.7	1	3.7	21245	28
39	X-n280-k17	279	E	R	SL	192	16.5	16.4	33769.4	33624	9.62	33605.8	33505	19.09	33286	87.7			33503	17
40	X-n284-k15	283	R	C (8)	1-10	109	20.2	18.9	20448.5	20295	8.64	20286.4	20227	19.91	20139	49.4			20226	15
41	X-n289-k60	288	E	RC (7)	Q	267	4.8	4.8	95450.6	95315	16.11	95469.5	95244	21.28	94928	25.1			95185	61
42	X-n294-k50	293	C	R	1-100	285	5.9	5.9	47254.7	47190	12.42	47259.0	47171	14.70	46911	26.7			47167	51
43	X-n298-k31	297	R	R	1-10	55	9.6	9.6	34356.0	34239	6.92	34292.1	34231	10.93	34105	1.8	195	531.1	34231	31
44	X-n303-k21	302	C	C (8)	1-100	794	15.0	14.4	21895.8	21812	14.15	21850.9	21748	17.28	21546	481.0			21744	21
45	X-n308-k13	307	E	RC (6)	SL	246	24.2	23.6	26101.1	25901	9.53	25895.4	25859	15.31	25587	247.7			25859	13
46	X-n313-k71	312	R	RC (3)	Q	248	4.4	4.4	94297.3	94192	17.50	94265.2	94093	22.41	93851	11.6			94044	72
47	X-n317-k53	316	E	C (4)	U	6	6.2	6.0	78356.0	78355	8.56	78387.8	78355	22.37	78334	1.7	5	4.6	78355	53
48	X-n322-k28	321	C	R	50-100	868	11.6	11.5	29991.3	29877	14.68	29956.1	29870	15.16	29722	547.2			29866	28
49	X-n327-k20	326	R	RC (7)	5-10	128	17.0	16.3	27812.4	27599	19.13	27628.2	27564	18.19	27378	221.9			27556	20
50	X-n331-k15	330	E	R	U	23	23.4	22.0	31235.5	31105	15.70	31159.6	31103	24.43	31027	25.7			31103	15
51	X-n336-k84	335	E	R	Q	203	4.0	4.0	139461.0	139197	21.41	139534.9	139210	37.96	138706	6.6			139197	86
52	X-n344-k43	343	C	RC (7)	5-10	61	8.0	8.0	42284.0	42146	22.58	42208.8	42099	21.67	41881	20.4			42099	43
53	X-n351-k40	350	C	C (3)	1-100	436	8.8	8.8	26150.3	26021	25.21	26014.0	25946	33.73	25809	170.4			25946	41
54	X-n359-k29	358	E	RC (7)	1-10	68	12.5	12.3	52076.5	51706	48.86	51721.7	51509	34.85	51381	82.3			51509	29
55	X-n367-k17	366	R	C (4)	SL	218	21.8	21.5	23003.2	22902	13.13	22838.4	22814	22.02	22747	247.9			22814	17
56	X-n376-k94	375	E	R	U	4	4.2	4.0	147713.0	147713	7.10	147750.2	147717	28.26	147713	3.3	1	3.3	147713	94
57	X-n384-k52	383	R	R	50-100	564	7.4	7.4	66372.5	66116	34.47	66270.2	66081	40.20	65681	256.7			66081	53
58	X-n393-k38	392	C	RC (5)	5-10	78	10.4	10.3	38457.4	38298	20.82	38374.9	38269	28.65	38167	46.7			38269	38
59	X-n401-k29	400	E	C (6)	Q	745	14.0	13.8	66715.1	66453	60.36	66365.4	66243	49.52	65971	318.1			66243	29
60	X-n411-k19	410	R	C (5)	SL	216	22.6	21.6	19954.9	19792	23.76	19743.8	19718	34.71	19640	433.9			19718	19
61	X-n420-k130	419	C	RC (3)	1-10	18	3.2	3.2	107838.0	107798	22.19	107924.1	107798	53.19	107704	5.4	169	115.0	107798	130
62	X-n429-k61	428	R	R	50-100	536	7.1	7.0	65746.6	65563	38.22	65648.5	65501	41.45	64930	31.6			65501	62
63	X-n439-k37	438	C	RC (8)	U	12	12.0	11.8	36441.6	36395	39.63	36451.1	36395	34.55	36289	20.0			36395	37
64	X-n449-k29	448	E	R	1-100	777	15.5	15.4	56204.9	55761	59.94	55553.1	55378	64.92	54928	804.9			55358	29
65	X-n459-k26	458	C	C (4)	Q	1106	17.8	17.6	24462.4	24209	60.59	24272.6	24181	42.80	23931	327.1			24181	26
66	X-n469-k138	468	E	R	50-100	256	3.4	3.4	222182.0	221909	36.32	222617.1	222070	86.65	221429	10.9			221909	140
67	X-n480-k70	479	R	C (8)	5-10	52	6.8	6.8	89871.2	89694	50.40	89760.1	89535	66.96	89235	59.5			89535	70
68	X-n491-k59	490	R	RC (6)	1-100	428	8.4	8.3	67226.7	66965	52.23	66898.0	66633	71.94	66263	418.1			66633	60
69	X-n502-k39	501	E	C (3)	U	13	13.0	12.8	69346.8	69284	80.75	69328.8	69253	63.61	69120	16.5			69253	39
70	X-n513-k21	512	C	RC (4)	1-10	142	25.0	24.4	24434.0	24332	35.04	24296.6	24201	33.09	24053	262.1			24201	21

Table 13: New set of benchmark instances : characteristics and results of current state-of-the-art algorithms (Part III)

		Instance Characteristics							**ILS-SP**			**UHGS**			**BCP**				**BKS**	
#	Name	n	Dep	Cust	Dem	Q	r	n/K_{min}	Avg	Best	T(min)	Avg	Best	T(min)	RLB	RT(min)	Nds	T(min)	Value	NV
70	X-n513-k21	512	C	RC (4)	1-10	142	25.0	24.4	24434.0	24332	35.04	24296.6	24201	33.09	24053	262.1			24201	21
71	X-n524-k153	523	R	R	SL	125	3.8	3.4	155005.0	154709	27.27	154979.5	154774	80.70	154533	12.5	381	212.1	154594	155
72	X-n536-k96	535	C	C (7)	Q	371	5.6	5.6	95700.7	95524	62.07	95330.6	95122	107.53	94409	47.9			95122	97
73	X-n548-k50	547	E	R	U	11	11.2	10.9	86874.1	86710	63.95	86998.5	86822	84.24	86604	16.1			86710	50
74	X-n561-k42	560	C	RC (7)	1-10	74	13.5	13.3	43131.3	42952	68.86	42866.4	42756	60.60	42495	302.3			42756	42
75	X-n573-k30	572	E	C (3)	SL	210	19.4	19.1	51173.0	51092	112.03	50915.1	50780	188.15	50575	1306.9			50780	30
76	X-n586-k159	585	R	RC (4)	5-10	28	3.6	3.7	190919.0	190612	78.54	190838.0	190543	175.29	189950	7.1			190543	159
77	X-n599-k92	598	R	R	50-100	487	6.5	6.5	109384.0	109056	72.96	109064.2	108813	125.91	108000	264.3			108813	94
78	X-n613-k62	612	C	R	1-100	523	10.0	9.9	60444.2	60229	74.80	59960.0	59778	117.31	59323	1429.8			59778	62
79	X-n627-k43	626	E	C (5)	5-10	110	14.5	14.6	62905.6	62783	162.67	62524.1	62366	239.68	62018	600.2			62366	43
80	X-n641-k35	640	E	RC (8)	50-100	1381	18.6	18.3	64606.1	64462	140.42	64192.0	63839	158.81	63228	401.0			63839	35
81	X-n655-k131	654	C	C (4)	U	5	5.0	5.0	106782.0	106780	47.24	106899.1	106829	150.48	106766	8.5	21	41.5	106780	131
82	X-n670-k130	669	R	R	SL	129	5.3	5.1	147676.0	147045	61.24	147222.7	146705	264.10	146211	103.1			146705	134
83	X-n685-k75	684	C	RC (6)	Q	408	9.2	9.1	68988.2	68646	73.85	68654.1	68425	156.71	67925	1643.2			68425	75
84	X-n701-k44	700	E	RC (7)	1-10	87	16.0	15.9	83042.2	82888	210.08	82487.4	82293	253.17	81694	680.6			82292	44
85	X-n716-k35	715	R	C (3)	1-100	1007	21.0	20.4	44171.6	44021	225.79	43641.4	43525	264.28	43113	419.5			43525	35
86	X-n733-k159	732	C	R	1-10	25	4.6	4.6	137045.0	136832	111.56	136587.6	136366	244.53	135748	57.9			136366	160
87	X-n749-k98	748	R	C (8)	1-100	396	7.7	7.6	78275.9	77952	127.24	77864.9	77715	313.88	76924	259.2			77700	98
88	X-n766-k71	765	E	RC (7)	SL	166	10.8	10.8	115738.0	115443	242.11	115147.9	114683	382.99	114108	262.8			114683	71
89	X-n783-k48	782	R	R	Q	832	16.5	16.3	73722.9	73447	235.48	73009.6	72781	269.70	71728	332.8			72727	48
90	X-n801-k40	800	E	R	U	20	20.2	20.0	74005.7	73830	432.64	73731.0	73587	289.24	73124	258.0			73587	40
91	X-n819-k171	818	C	C (6)	50-100	358	4.8	4.8	159425.0	159164	148.91	158899.3	158611	374.28	157627	257.2			158611	173
92	X-n837-k142	836	R	RC (7)	5-10	44	5.9	5.9	195027.0	194804	173.17	194476.5	194266	463.36	193245	253.7			194266	142
93	X-n856-k95	855	C	RC (3)	U	9	9.6	9.0	89277.6	89060	153.65	89238.7	89118	288.43	88839	43.6			89060	95
94	X-n876-k59	875	E	C (5)	1-100	764	15.0	14.8	100417.0	100177	409.31	99884.1	99715	495.38	98880	433.2			99715	59
95	X-n895-k37	894	R	R	50-100	1816	24.2	24.2	54958.5	54713	410.17	54439.8	54172	321.89	53147	984.5			54172	38
96	X-n916-k207	915	E	RC (6)	5-10	33	4.4	4.4	330948.0	330639	226.08	330198.3	329836	560.81	328588	97.1			329836	208
97	X-n936-k151	935	C	R	SL	138	6.2	6.2	134530.0	133592	202.50	133512.9	133140	531.50	132496	395.8			133105	159
98	X-n957-k87	956	R	RC (4)	U	11	11.6	11.0	85936.6	85697	311.20	85822.6	85672	432.90	85328	257.1			85672	87
99	X-n979-k58	978	E	C (6)	Q	998	17.0	16.9	120253.0	119994	687.22	119502.1	119194	553.96	118399	937.5			119194	58
100	X-n1001-k43	1000	R	R –	1-10	131	23.4	23.3	73985.4	73776	792.75	72956.0	72742	549.03	71812	308.8			72742	43
	Min	100				3	3.2	3.0	0.00%	0.00%	0.13	0.00%	0.00%	1.43	0.00%	0.1				
	Max	1000				1816	25.0	24.4	2.50%	1.42%	792.75	0.55%	0.13%	560.81	1.89%	1643.2				
	Avg.	412.2				324.6	11.4	11.1	0.52%	0.25%	71.71	0.19%	0.01%	98.79	0.44%	189.4				
	Median	333				149	10.2	9.9	0.38%	0.10%	17.67	0.20%	0.00%	22.39	0.40%	33.6				

a continuous diversification procedure by modifying the objective during parents and survivors selection to promote not only good but also diverse solutions. Both methods are known to achieve very high quality results on the previous sets of CVRP instances as well as on various other vehicle routing variants.

The two methods have been run with the same parameter setting specified in the original papers. To produce solutions that can stand the test of time, and counting on the fact that more computing power will be surely available in the coming years, we selected a slightly larger termination criterion for UHGS by allowing up to 50.000 consecutive iterations without improvement. These tests have been conducted on a Xeon CPU with 3.07 GHz and 16 GB of RAM, running under Oracle Linux Server 6.4. The average and best results on 50 runs are reported in the Table, as well as the average computational time per instance.

From these tests, it appears that both ILS-SP and UHGS produce solutions of consistent quality, with an average gap of 0.52% and 0.19%, respectively, with respect to the best known solutions (BKS) ever found during all experiments. Considering the subset of instances that are solved to optimality (see Section 4.2), ILS-SP and UHGS achieve average gaps of 0.18% and 0.09%, respectively. A direct comparison of the best solutions found by each method provides the following score: UHGS solutions are better 58 times, ILS-SP solutions are better 9 times and there are 33 ties.

UHGS produces solutions of generally higher quality than ILS for a comparable amount of CPU time, except for some instances containing few customers per route. For this type of problems, the set partitioning solver can produce new high-quality solutions from existing routes in a very efficient manner, while generating new structurally different solutions with other choices of assignments is more challenging for local searches and even for randomized crossovers. On the other hand, for problems with a large number of customers per route, the hybrid ILS exhibits a slower convergence and generally leads to solutions of lower quality.

Overall, these experiments show that some state-of-the-art methods may perform well on different classes of instances. Therefore, a promising research path would involve the extension of these methods and/or further hybridizations to cover all problems in the best possible way.

4.2 Exact solutions

The Branch-Cut-and-Price (BCP) in Pecin et al. [22] is a complex algorithm that combines and improves several ideas proposed by other authors [4, 5, 10, 28], as also detailed in [23].

The algorithm was run in a single core of an Intel i7-3960X 3.30GHz processor with 64 GB RAM. The value of the best solution found by the previously mentioned heuristics was inputted as an initial upper bound. Actually, in order to increase our confidence in the correctness of the implementation, we always add 1 to that upper bound. In this way, the BCP always has to find by itself a feasible solution with value at least as good as the upper bound.

The BCP could solve 40 out of the 100 new instances to optimality in reasonable times (maximum of 5 days, for X-n186-k15). For those instances we report the root node lower bound, the time to obtain that bound, the number of nodes in the search tree and the total time. For the remaining 60 instances we only report a lower bound and the corresponding time.

- The BCP can solve most instances with up to 275 customers. In the 38 instances in that range, it only failed when the routes are very long (X-n190-k8, X-n214-k11, X-n233-k16, and X-n261-k13). On the other hand, only 6 larger instances could be solved (X-n298-k31, X-n317-k53, X-376-k94, X-n420-k130, X-n524-k153, and X-nX655-k131). Those more favorable instances have short routes and small values of Q.

- The BCP attested the general good quality of the solutions that can be found by current heuristics. Consider the best solution provided either by ILS-SP or UHGS. In 96 out of the 100 instances, this solution is less than 1% above the computed lower bounds.

5 The CVRPLIB Web Site

The typical instance repository of today is a web page that allows downloading the instance files and includes additional textual information, like file format description, instance source, best known/optimal solution values, etc. The CVRLIB web page, where the new instances (and all the previous CVRP instances described in Section 2) are available (`http://vrp.galgos.inf.puc-rio.br/index.php/en/`), is more sophisticated:

- Its core is a full-fledged database containing, for each instance: (i) its actual data, like n, Q, customer coordinates and demands, (ii) its best known/optimal solution, represented as set of routes, (iii) a miscellanea of additional information, like original source, the values of the attributes used in its generation (for the new instances), and even comments about remarkable characteristics. The visible pieces of information that appear in the web site are produced by queries. The objective of that design is having a larger degree of consistency and flexibility in the maintenance of the page.

- The best known solutions are automatically verified, their feasibility is checked and their costs are calculated from the original instance data. This eliminates the possibility that false best known solutions appear due to typos or misunderstandings about the conventions.

- Instances with euclidean distances (the vast majority) can be depicted graphically, along with their best known/optimal solutions.

6 Conclusions

We hope that the proposed set of instances will contribute to spark new interest on the "classic" CVRP, helping to test new algorithmic developments in the coming years. In particular, by proposing a common set of benchmark instances for both heuristic and exact methods, this work contributes to end a bizarre situation where minor convention details were sufficient for isolating both communities. For example, most of the CVRP instances in Table 7 could have their best known solutions proven to be optimal by the algorithms in [4, 5, 10, 14, 28]. But none of those authors effectively changed a few lines of code to adapt for the different conventions. This instance-induced isolation was harmful and prevented a more effective cross-fertilization between exact and heuristic methods. In fact, this paper has shown that it is already possible to perform direct comparisons of heuristic results with good lower bounds or even proven optimal solutions on fairly large-sized instances.

A major concern in the design of the new benchmark set was problem diversity. By having instances covering a wider set of characteristics, the benchmark will favor the development of more flexible methods (perhaps hybrids), that will eventually be applied to real-life situations with lesser risks of failure due to an uncommon instance type.

Finally, we point out that the proposed CVRP instances can be extended to other classical VRP variants as needed, by providing additional data fields like duration constraints, time windows, or heterogeneous fleet specifications.

Acknowledgments

The authors would like to thank Teobaldo Bulhões Jr. for computing the values of K_{min} for the new set of proposed instances and Ivan Xavier de Lima for his help on creating the CVRPLIB web page.

References

[1] Achuthan, N., Caccetta, L., Hill, S., 2003. An improved branch-and-cut algorithm for the capacitated vehicle routing problem. Transportation Science 37, 153–169.

[2] Araque, J., Kudva, G., Morin, T., Pekny, J., 1994. A branch-and-cut algorithm for the vehicle routing problem. Annals of Operations Research 50, 37–59.

[3] Augerat, P., Belenguer, J., Benavent, E., Corberán, A., Naddef, D., Rinaldi, G., 1995. Computational results with a branch and cut code for the capacitated vehicle routing problem. Tech. Rep. 949-M, Université Joseph Fourier, Grenoble, France.

[4] Baldacci, R., Christofides, N., Mingozzi, A., 2008. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. Mathematical Programming 115 (2), 351–385.

[5] Baldacci, R., Mingozzi, A., Roberti, R., 2011. New route relaxation and pricing strategies for the vehicle routing problem. Operations Research 59 (5), 1269–1283.

[6] Blasum, U., Hochstättler, W., 2000. Application of the branch and cut method to the vehicle routing problem. Tech. Rep. ZPR2000-386, Zentrum fur Angewandte Informatik Köln.

[7] Christofides, N., Eilon, S., 1969. An algorithm for the vehicle-dispatching problem. Operational Research Quarterly 20, 309–318.

[8] Christofides, N., Mingozzi, A., Toth, P., 1979. The vehicle routing problem. In: Christofides, N., Mingozzi, A., Toth, P., Sandi, C. (Eds.), Combinatorial Optimization. Vol. 1. Wiley Interscience, pp. 315–338.

[9] Clarke, G., Wright, J., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations research 12 (4), 568–581.

[10] Contardo, C., Martinelli, R., 2014. A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. Discrete Optimization 12, 129–146.

[11] Cook, W., Koch, T., Steffy, D. E., Wolter, K., 2011. An exact rational mixed-integer programming solver. In: Integer Programming and Combinatoral Optimization. Springer, pp. 104–116.

[12] Dantzig, G. B., Ramser, J. H., 1959. The truck dispatching problem. Management science 6 (1), 80–91.

[13] Fisher, M., 1994. Optimal solution of vehicle routing problems using minimum K-trees. Operations research 42 (4), 626–642.

[14] Fukasawa, R., Longo, H., Lysgaard, J., Poggi de Aragão, M., Reis, M., Uchoa, E., Werneck, R., 2006. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. Mathematical programming 106 (3), 491–511.

[15] Gaskell, T., 1967. Bases for vehicle fleet scheduling. OR, 281–295.

[16] Gillett, B. E., Miller, L. R., 1974. A heuristic algorithm for the vehicle-dispatch problem. Operations research 22 (2), 340–349.

[17] Golden, B., Wasil, E., Kelly, J., Chao, I., 1998. The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: Fleet management and logistics. Springer, pp. 33–56.

[18] Gouveia, L., 1996. Multicommodity flow models for spanning trees with hop constraints. European Journal of Operational Research 95 (1), 178–190.

[19] Li, F., Golden, B., Wasil, E., 2005. Very large-scale vehicle routing: new test problems, algorithms, and results. Computers & Operations Research 32 (5), 1165–1179.

[20] Lysgaard, J., Letchford, A., Eglese, R., 2004. A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming 100, 423–445.

[21] Nagata, Y., Bräysy, O., 2009. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. Networks 54 (4), 205–215.

[22] Pecin, D., Pessoa, A., Poggi, M., Uchoa, E., 2014. Improved branch-cut-and-price for capacitated vehicle routing. In: Integer Programming and Combinatorial Optimization. Springer, pp. 393–403.

[23] Poggi, M., Uchoa, E., 2014. New exact approaches for the capacitated VRP. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications,. SIAM, Ch. 3, pp. 59–86.

[24] Ralphs, T., Kopman, L., Pulleyblank, W., Jr., L. T., 2003. On the capacitated vehicle routing problem. Mathematical Programming 94, 343–359.

[25] Rand, G. K., 2009. The life and times of the Savings Method for vehicle routing problems. ORiON: The Journal of ORSSA 25 (2), 125–145.

[26] Reinelt, G., 1991. Tsplib - a traveling salesman problem library. ORSA journal on computing 3 (4), 376–384.

[27] Rochat, Y., Taillard, É. D., 1995. Probabilistic diversification and intensification in local search for vehicle routing. Journal of heuristics 1 (1), 147–167.

[28] Røpke, S., 2012. Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems. Presentation in Column Generation 2012.

[29] Solomon, M. M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations research 35 (2), 254–265.

[30] Subramanian, A., Uchoa, E., Ochi, L. S., 2013. A hybrid algorithm for a class of vehicle routing problems. Computers & Operations Research 40 (10), 2519–2531.

[31] Taillard, É., 1993. Parallel iterative search methods for vehicle routing problems. Networks 23 (8), 661–673.

[32] Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Operations Research 60 (3), 611–624.

[33] Vidal, T., Crainic, T. G., Gendreau, M., Prins, C., 2014. A unified solution framework for multi-attribute vehicle routing problems. European Journal of Operational Research 234 (3), 658 – 673.

[34] Wenger, K., 2003. Generic cut generation methods for routing problems. Ph.D. thesis, Institute of Computer Science, University of Heidelberg.

A Results on the instances of Li et al. [19]

Table 14 reports the results achieved with 10 runs of the hybrid genetic algorithm of [32], using the same parameter configuration as in the original paper and the same computing environment as other tests of this paper. The columns provide, in turn, the instance number and size, the average and best solutions quality, and CPU time per run.

Table 14: Instances of Li et al. [19] – results of [32].

Instance	n	**Avg.**	**Best**	**T(min)**
21	560	16212.83	16212.83	8.08
22	600	14545.76	14528.19	35.37
23	640	18826.22	18801.13	10.96
24	720	21389.43	21389.43	13.74
25	760	16753.56	16705.11	54.38
26	800	23977.73	23977.73	19.19
27	840	17411.96	17383.18	59.66
28	880	26566.04	26566.04	20.87
29	960	29154.34	29154.34	25.26
30	1040	31742.64	31742.64	30.11
31	1120	34330.94	34330.94	35.01
32	1200	37159.41	37159.41	51.07