https://pubsonline.informs.org/journal/ijoc

INFORMS JOURNAL ON COMPUTING

Vol. 00, No. 0, Xxxxx 0000, pp. 000–000

Submitted to *INFORMS Journal on Computing*

ISSN 0899-1499, EISSN 1526-5528

Supplemental Material: Cluster Branching for Vehicle Routing Problems

João Marcos Pereira Silva
Departamento de Engenharia de Produção, Universidade Federal Fluminense, joaomarcossilva@id.uff.br

Eduardo Uchoa
Departamento de Engenharia de Produção, Universidade Federal Fluminense / INRIA International Chair (2022–2026), eduardo.uchoa@gmail.com

Anand Subramanian
Departamento de Sistemas de Computação, Universidade Federal da Paraíba, anand@ci.ufpb.br

Appendix A: Branch-Cut-and-Price algorithms

This section presents the VRPSolver models for the Capacitated Vehicle Routing Problem (CVRP),VRP with Time Windows (VRPTW), and Distance-Constrained CVRP (DCCVRP).

VRPSolver (Pessoa et al. 2019, 2020) is a framework designed to facilitate the creation Branch-Cut-and-Price (BCP) algorithms for different VRPs variants. VRPSolver model is a MIP that contains variables associated with resource-constrained paths over directed graphs defined by the user. Given that the number of paths can be exponentially large, these variables are generated dynamically, solving as pricing problems Resource Constrained Shortest Path (RCSP).

It is the user's task to properly define the RCSP graphs and the *mappings* $\mathcal{M}(x_j)$ of the variables x_j in the user-defined objective function and constraints of the Master Problem (MP). These mappings connect the variables in the MP to a subset of arcs induced by the RCSPs. In addition, the set $\mathcal{R}$ of resources, and the lower (L) and upper (U) bounds on the number of paths from the RCSP in a solution must be defined. To define the feasibility of the paths, each RCSP graph has special vertices v_{source} and v_{sink} representing the start and the end of the paths. Each arc a must have a consumption q_a^r of resource $r \in \mathcal{R}$, and accumulated resource consumption intervals $[l_i^r, u_i^r]$ of r must be defined on each vertex i. The consumption intervals can also be defined over arcs a, denoted as $[l_a^r, u_a^r]$. In all cases, it is mandatory for the accumulated resource consumption to satisfy the UB of the consumption interval. When a resource is defined as *disposable*, resources may be dropped if necessary to meet the lower bound of the consumption interval. However, for non-disposable resources, the consumption interval must be strictly satisfied.

To strengthen the formulation, it is possible to define the so-called *elementarity sets* ($\mathcal{E}$) and *packing sets* ($\mathcal{S}$) on vertices or arcs. Within each $S \in \mathcal{S}$, the elements (arcs or vertices) appear at most once across all paths within any optimal solution. In contrast, within each $S \in \mathcal{E}$, the elements appear at most once in each path that comprises any optimal solution.

It is also possible to define Rounded Capacity Cut (RCC) (Laporte and Nobert 1983) separators, route enumeration (Baldacci et al. 2008, Contardo and Martinelli 2014), Rank-1 cuts with limited memory (Jepsen et al. 2008, Pecin et al. 2017a,b), and branching strategies (Ryan and Foster 1981, Gélinas et al. 1995). For a comprehensive understanding of VRPSolver models, please refer to Pessoa et al. (2020).

A.1. VRPSolver model for the CVRP proposed by Pessoa et al. (2020)

Data: Undirected graph $G = (V = \{0\} \cup V_+, E)$, where 0 is the depot and $V_+ = \{1, \ldots, n\}$ are the customers; positive costs $c_e, e \in E$ and positive demands $d_i, i \in V_+$; vehicle capacity Q.

Goal: Find a minimum cost set of routes that start and end at the depot, visit all customers, and ensure that the sum of the demands of the customers on each route does not exceed the vehicle capacity.

RCSP graph: A single graph $\mathcal{G} = (\mathcal{V} = V, \mathcal{A}), \mathcal{A} = \{(i,j),(j,i) : \{i,j\} \in E\}$. $v_{\text{source}} = v_{\text{sink}} = 0$. $\mathcal{R} = \{1\}$, that is a single monotone disposable resource defined over the demands of the customers (define $d_0 = 0$); arc consumptions $q^1_a = (d_i + d_j)/2, a = (i,j) \in \mathcal{A}$; and interval consumptions $[l^1_i, u^1_i] = [0, Q], i \in V$.

MP: Define the integer variables $x_e, e \in E$. The formulation is:

$$\min \sum_{e \in E} c_e x_e \tag{1a}$$

$$\text{s.t.} \sum_{e \in \delta(i)} x_e = 2 \qquad i \in V_+ \tag{1b}$$

$$x_e \leq 1 \qquad e \in E \setminus \delta(0); \tag{1c}$$

$\mathcal{M}(x_e) = \{(i,j),(j,i)\}, e = \{i,j\} \in E$; $L = \lceil \sum_{i \in N} d_i / Q \rceil$ and $U = n$.

BCP elements: $\mathcal{S}^V = \mathcal{E}^V = \cup_{i \in V_+} \{\{i\}\}$. RCC separator given by $(\cup_{i \in V_+} \{(\{i\}, d_i)\}, Q)$. Branching is over variables x. The enumeration procedure is on.

Comments: Constraints (1b) are the degree constraints over the customers. Constraints (1c) are dynamically separated as user cuts. Packing and elementarity sets are defined on vertices. The RCC separator is defined by setting the capacity as the vehicle capacity Q and a demand function on the $d_i, i \in V_+$.

A.2. VRPSolver model for VRPTW proposed by Pessoa et al. (2020)

Data: The data of the CVRP with the inclusion of a time window $[l_i, u_i]$ and positive service time s_i for each customer $i \in V_+$. There are also positive travel times $t_e, e \in E$.

Goal: The same goal as for CVRP, with the additional constraints that the service at customer i must start within their time window, and the service at each customer has a duration of s_i.

RCSP graph: The same graph as CVRP, with the inclusion of a second main monotone disposable resource defining time, $\mathcal{R} = \{1,2\}$. For each $t_{e=\{i,j\}}$, let $t_{a=(i,j)} = t_e + s_i$ and $t_{a=(j,i)} = t_e + s_j$. For each arc $a = (i,j) \in A$, the arc consumptions $q^1_a = d_j$ and $q^2_a = t_a$; and interval consumptions $[l^1_i, u^1_i] = [0, Q]$ and $[l^2_i, u^2_i] = [l_i, u_i]$ for each $i \in V$.

MP and *BCP elements*: The same as CVRP model

Comments: Note that it is also possible to define only time as a graph resource, while capacity is enforced by RCCs.

A.3. VRPSolver model for DCCVRP

Data: The data of the CVRP; maximum travel distance (or time duration) T; a constant service time $s_i = s, i \in V_+$.

Goal: The same goal as for CVRP, with the additional constraint that each vehicle must not exceed a maximum travel distance T in its route, and the service at each customer has a duration of s_i.

RCSP graph: The same graph as CVRP, with the inclusion of a second main monotone disposable resource defining the travel distance, $\mathcal{R} = \{1,2\}$. For each arc $a = (i,j) \in A$, the arc consumptions $q^1_a = d_j$ and $q^2_a = c_a + (s_i + s_j)/2$ (define $s_0 = 0$); and interval consumptions $[l^1_i, u^1_i] = [0, Q]$ and $[l^2_i, u^2_i] = [0, T]$ for each $i \in V$.

MP and *BCP elements*: The same as CVRP model

Comments: Note that it is also possible to define only travel distance as a graph resource, while capacity is enforced by RCCs.

Appendix B: Long runs

In this section, we present the results of additional long runs involving some "promising" CVRP and VRPTW open instances, i.e., those where the 24-hour ETS% forecasts that the instance may be solved by spending extra days of CPU time. Two of these long runs extended for several months (an exceptional effort to close the last Homberger and Gehring instances with 200 customers). Such very long runs were only possible because we could run the subtrees rooted at some open nodes of the same instance in parallel, using different processors and cores of the server. For each open instance solved, a customized BCP parameterization was employed.

Before presenting the results of the long runs, we list the instances and their results from the 24-hour experiments across various CB methods. The CVRP instances selected for the extended run are displayed in Table 1, while the VRPTW instances are given in Table 2. These tables show the ∇_{avg} and ETS% results from the 24-hour CB experiments for each MST-based clustering method, as well as for the standard EB and CSB.

Among the CVRP instances, three (`X-n256-k16`, `X-n351-k40`, and `X-n367-k17`) have clustered customer positioning, while `X-n344-k43` is randomly clustered, and `X-n670-k126` has randomly positioned customers. Only the last instance was not improved by combining EB or CSB with CB. Also, the performance of the pure version of EB was slightly better than the pure CSB test. We believe this happens because it is an example of a large random instance with no "hidden" cluster structures, and consequently, CB and CSB do not have the desired effect.

Table 1 Results for some open X instances. Runs with 24 hour TL.

	ETS%					∇_{avg}				
Branch type	X-n256-k16	X-n344-k43	X-n351-k40	X-n367-k17	X-n670-k126	X-n256-k16	X-n344-k43	X-n351-k40	X-n367-k17	X-n670-k126
EB	0.89	6.70	9.03	9.04	**39.26**	6.09	9.07	9.04	11.17	**15.04**
EB + MST0.5	7.61	**12.72**	16.19	13.89	14.45	8.16	**9.28**	**9.57**	12.05	12.40
EB + MST1.0	**8.86**	7.96	**20.57**	22.60	22.91	**8.56**	8.55	9.35	**12.68**	14.22
EB + MST1.5	5.60	11.76	14.24	**24.49**	17.62	7.77	9.20	8.98	11.47	14.08
CSB	5.79	7.04	13.82	14.12	**38.76**	8.34	8.58	**9.99**	15.29	**17.28**
CSB + MST0.5	**11.38**	12.90	**21.97**	17.33	21.81	7.71	9.78	8.94	13.36	13.75
CSB + MST1.0	10.86	**21.70**	19.13	18.22	25.36	8.35	**9.79**	9.27	12.80	11.05
CSB + MST1.5	11.00	13.76	16.02	**19.46**	26.32	**8.70**	9.16	8.66	**16.44**	9.11

A similar behavior is observed for the random instances `R1_4_5` and `R1_4_6` in Table 2. In these cases, the CSB version was not improved by CB, while in the EB, only `R1_4_6` was slightly improved by combining it with CB

Table 2 Results for some open VRPTW Homberger instances. Runs with 24-hour TL.

	ETS%										∇_{avg}									
Branch type	RC1_2_9	RC1_2_10	C2_2_4	RC2_2_10	RC1_4_1	RC1_4_2	R1_4_2	R1_4_5	R1_4_6	C1_6_10	RC1_2_9	RC1_2_10	C2_2_4	RC2_2_10	RC1_4_1	RC1_4_2	R1_4_2	R1_4_5	R1_4_6	C1_6_10
EB	20.63	40.11	2.43	**3.14**	10.34	11.46	26.75	**38.84**	58.85	6.26	10.75	13.8	8.08	**9.18**	3.79	10.38	**7.90**	8.31	9.87	9.71
EB+MST0.5	25.42	61.71	3.22	2.24	12.42	9.94	25.30	30.77	59.51	12.80	11.03	**15.00**	7.88	8.89	**5.65**	11.25	7.46	**8.43**	**10.34**	10.91
EB+MST1.0	24.08	61.22	1.31	2.23	**21.39**	**15.28**	**30.36**	31.94	**59.76**	**18.66**	10.68	13.9	7.70	8.52	4.62	**11.97**	7.69	8.22	9.64	11.00
EB+MST1.5	**32.82**	**62.79**	**3.78**	2.03	18.84	13.17	24.98	24.04	51.71	17.88	**12.17**	13.61	**8.89**	8.76	4.00	11.32	7.51	8.18	9.91	**11.28**
CSB	23.58	50.76	23.18	1.10	10.05	9.40	13.96	**27.90**	**52.05**	6.68	9.41	13.08	11.44	7.97	2.73	10.81	6.80	**7.90**	**9.61**	8.98
CSB+MST0.5	26.32	57.25	5.94	3.31	11.68	12.11	**17.02**	17.33	48.16	9.59	10.51	**13.47**	9.09	8.96	**6.44**	**11.91**	6.96	7.67	9.44	8.81
CSB+MST1.0	**30.96**	**62.29**	9.72	**3.65**	**13.51**	**21.86**	16.91	15.86	31.07	18.00	**10.55**	11.59	9.21	8.91	3.38	10.66	**6.99**	7.29	8.33	**10.67**
CSB+MST1.5	28.20	61.56	**28.60**	3.28	6.04	17.17	12.07	18.91	50.71	**18.68**	10.34	11.63	**12.39**	**9.23**	4.93	11.47	6.80	7.55	9.38	9.69

MST1.0. Note that, in both instances, the pure EB outperformed CSB and any combination of it with CB MST-based clustering.

Tables 3 and 4 present the results of the long runs for the CVRP and VRPTW instances, respectively. For these extended runs, only one strategy — either EB+CB, CSB+CB, pure EB, or pure CSB — was considered. Although Tables 1 and 2 identify the best configurations in terms of branching strategy, these optimal configurations were not always applied in the long runs. The choice of branching strategy during the long runs followed the progression of the research, during which some of the previously shown strategies or evaluation metrics were not yet considered.

Tables 3 and 4 show the optimal cost (column **Opt. cost**), the size of the B&B tree (column **#bb**), and the time required to solve the instance (column **t (days)**). These columns provide statistics related to the BCP algorithm. In addition, the last set of columns indicates the branching strategy used, and for those utilizing CB, the number of clusters m obtained by the clustering method is also presented.

By employing the EB+MST1.0, the instance `X-n256-k16` was eventually solved after exploring 2897 nodes, requiring a total CPU time of 35.2 days. Despite the extensive duration, it is worth noting that the ETS% of the standard edge branching, as shown in Table 1, exceeds that of the CB utilizing the MST1.0 by almost a factor of 10. Even if the edge branching were to operate within the same time limit, it would still fall significantly short of solving it.

Regarding the other CVRP instances, the time consumed is acceptable given that these are large and challenging instances for exact methods. In addition, it is surprising to note that `X-n670-k126` was solved in a very reasonable time by applying only the standard edge branching as a branching strategy, which was expected based on the results in Table 1.

Table 3 Results obtained on the CVRP X instances with extended running time.

	BCP			Branching	
Instance	**Opt. cost**	**#bb**	**t (days)**	**strategy**	m
`X-n256-k16`	**18839**	2897	35.2	EB+MST1.0	43
`X-n344-k43`	**42050**	897	4.4	EB+MST1.5	30
`X-n351-k40`	**25896**	2569	8.4	EB+MST1.0	37
`X-n367-k17`	**22814**	265	2.8	EB+MST1.0	42
`X-n670-k126`	**146332**	1691	8.3	EB	–

From Table 4, the experiments conducted on problem instances `RC1_2_9`, `RC1_2_10`, `RC1_4_1`, `RC1_4_2`, `R1_4_2`, `R1_4_5`, `R1_4_6`, and `C1_6_10` ratified the positive results anticipated from the 24-hour experiments. However, for `RC2_2_10` and `C2_2_4`, the same level of success was not observed, as both instances required a significantly long time to solve. This outcome was expected, as anticipated in Table 2. Given that both instances involve solutions with very long routes, they naturally pose significant challenges for BCP algorithms. It is worth noting that instances `RC1_2_9`, `RC1_2_10`, `C2_2_4`, and `RC2_2_10` were the last four open instances with 200 customers in the Homberger and Gehring benchmark.

Table 4 **Experiment results on the VRPTW Homberger instances with extended running time.**

	BCP			Branching	
Instance	**Opt. cost**	**#bb**	**t (days)**	**strategy**	m
RC1_2_9	**3073.3**	1434	5.2	EB+MST1.5	23
RC1_2_10	**2990.5**	562	1.5	EB+MST1.5	23
C2_2_4	**1695.0**	20738	188.6	EB+MST0.5	45
RC2_2_10	**1989.2**	8765	93.8	EB+MST1.0	32
RC1_4_1	**8522.9**	3289	6.8	EB+MST1.0	70
RC1_4_2	**7878.2**	839	7.8	EB+MST1.0	70
R1_4_2	**8873.2**	2355	3.2	EB+MST1.0	61
R1_4_5	**9184.6**	7043	5.8	EB+MST1.0	62
R1_4_6	**8340.4**	1073	2.4	EB+MST1.0	62
C1_6_10	**13617.5**	1940	12.3	EB+MST1.5	43

Appendix C: Additional results on the CMT13 instance

The six well-defined customer clusters in Figure 2 (original paper) are numbered, with the depot designated as C_0. Cluster 1 (C_1) comprises the vertices surrounding the depot, while the remaining clusters are numbered clockwise. The Branch-and-Bound (B&B) tree in Figure 1 illustrates the outcome of applying CSB and CB strategies to solve the CMT13 instance. Cyan nodes denote instances where strong branching (SB) favored the Cluster Branching over ω_C variables, while gray nodes represent CB branching on ψ_{C_1,C_2} variables. Notably, SB preferred CB branching in a total of 26 branching, with 15 on ω_C and 11 on ψ_{C_1,C_2}. The yellow node denotes the sole node where SB opted for a branch on cutsets ω_S. Moreover, the green node marks the point where the solution with a cost of 1541.14 was proven optimal.

References

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* 115:351–385.

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.

Gélinas S, Desrochers M, Desrosiers J, Solomon MM (1995) A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research* 61:91–109.

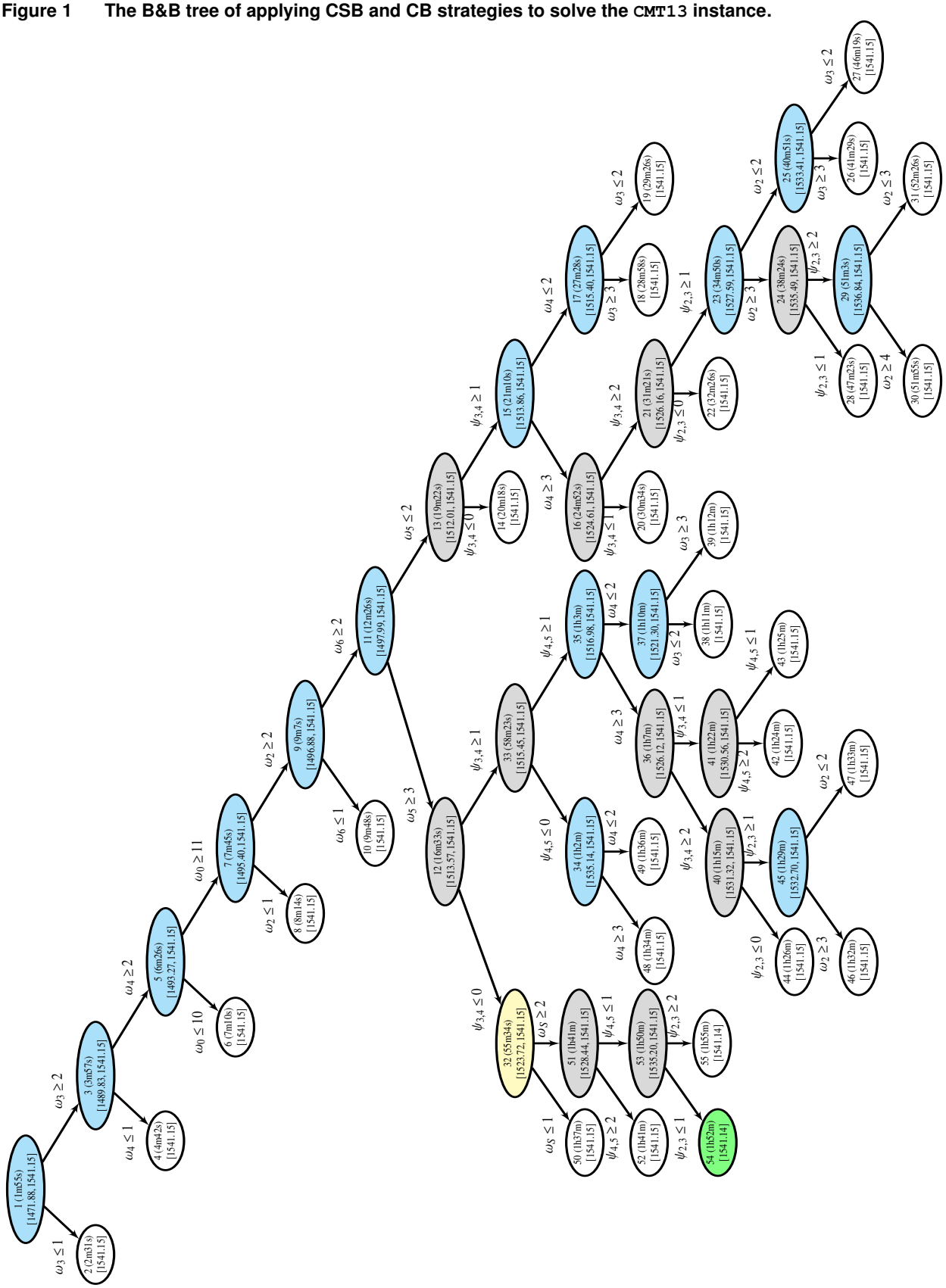
Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research* 56(2):497–511.

Laporte G, Nobert Y (1983) A branch and bound algorithm for the capacitated vehicle routing problem. *Operations-Research-Spektrum* 5:77–85.

Pecin D, Pessoa A, Poggi M, Uchoa E (2017a) Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation* 9(1):61–100.

Pecin D, Pessoa A, Poggi M, Uchoa E, Santos H (2017b) Limited memory rank-1 cuts for vehicle routing problems. *Operations Research Letters* 45(3):206 – 209.

Figure 1 The B&B tree of applying CSB and CB strategies to solve the `CMT13` instance.



Pessoa A, Sadykov R, Uchoa E, Vanderbeck F (2019) A generic exact solver for vehicle routing and related problems. Lodi A, Nagarajan V, eds., *Integer Programming and Combinatorial Optimization*, 354–369 (Cham: Springer International Publishing).

Pessoa A, Sadykov R, Uchoa E, Vanderbeck F (2020) A generic exact solver for vehicle routing and related problems. *Mathematical Programming B* 183:483–523.

Ryan DM, Foster BA (1981) An integer programming approach to scheduling. Wren A, ed., *Computer scheduling of public transport: Urban passenger vehicle and crew scheduling*, 269–280 (North-Holland, Amsterdam).