Minimum Makespan Vehicle Routing Problem with Compatibility Constraints

Miao Yu, Viswanath Nagarajan, and Siqian Shen

University of Michigan, Ann Arbor, Michigan, United States
{miaoyu, viswa, siqian}@umich.edu
orcid.org/0000-0002-7625-6315, 0000-0002-9514-5581, 0000-0002-2854-163X

Abstract. We study a multiple vehicle routing problem, in which a fleet of vehicles is available to serve different types of services demanded at locations. The goal is to minimize the makespan, i.e. the maximum length of any vehicle route. We formulate it as a mixed-integer linear program and propose a branch-cut-and-price algorithm. We also develop an efficient $O(\log n)$-approximation algorithm for this problem. We conduct numerical studies on Solomon's instances with various demand distributions, network topologies, and fleet sizes. Results show that the approximation algorithm solves all the instances very efficiently and produces solutions with good practical bounds.

Keywords: vehicle routing, compatibility constraints, branch-cut-and-price, approximation algorithm

1 Introduction

Vehicle routing problems (VRPs), with a goal of finding the optimal routing assignment for a fleet of vehicles to serve demands at various locations, are classical and well studied combinatorial optimization problems. Starting from Dantzig and Ramser [7], many variants of VRPs have been considered, including VRP with time windows, Capacitated VRP, VRP with heterogeneous fleet, VRP with multiple depots, as well as hybrid versions of these variants, all of which are discussed in detail by Golden et al. [15] and Toth and Vigo [24].

In this paper, we consider a minimum makespan VRP with compatibility constraints (VRPCC). We assume that multiple types of services are demanded at various locations of a given network and each type of service can only be served by certain vehicles. The goal of the problem is to minimize the maximum traveling time of all the routes designed for fulfilling the demand, i.e., the makespan for finishing all services. The motivation of studying the minimization of makespan comes from applications that focus on balancing routing assignment for all vehicles and minimizing the time to finish serving all customers. We consider a salient application as deploying shared vehicles for serving patient medical home care service demand, distributed in a geographical network, in which we aim to balance workload of different medical staff teams dispatched

together with the vehicles. The solution methods investigated in this paper can help scheduling and routing for medical home care delivery.

We review the main VRP literature focusing on different solution methods. To exactly optimize VRPs, branch-and-cut was the dominant approach before 2000s, and the related research [see, e.g., 1, 3, 19, 20, 22] develops effective valid inequalities to improve solution efficiency. Following Fukasawa et al. [13], branch-cut-and-price (BCP) became the best performing exact solution method for (capacitated) VRP: this combines branch-and-cut with column generation. There have also been many approximation algorithms for VRPs with a minimum makespan objective [see, e.g., 4, 8, 12] that provide polynomial time algorithms with a provable performance guarantee. To the best of our knowledge, no prior work has provided either exact solution methods or efficient approximation algorithms for VRPCC.

In this paper, we develop the following algorithms for VRPCC: (i) an exact algorithm based on the BCP approach and (ii) an $O(\log n)$-approximation algorithm based on a budgeted covering approach. We provide preliminary numerical experiments for both algorithms. Our results show that the approximation algorithm solves the problem efficiently and yields a practical approximation ratio of at most two (on the test instances).

The remainder of the paper is organized as follow. In Section 2, we formally define the problem and present a mixed-integer linear programming formulation. In Sections 3 and 4, we propose a BCP algorithm and an approximation algorithm for VRPCC, respectively. In Section 5, we test our algorithms on Solomon's instances with diverse graph sizes, vehicle fleet sizes, and demand distributions. We present the numerical results for our proposed solution methods. In Section 6, we conclude our findings and state future research.

2 Problem Statement and Formulation

Consider an undirected graph $G = (V, E)$, where $V = \{0, 1, \ldots, n\}$ is the set of $n+1$ nodes. Node 0 represents the depot and set $V^+ = \{1, 2, \ldots, n\}$ corresponds to customer locations. Each edge $(i, j) \in E,\ \forall i,\ j \in V$ has an associated distance d_{ij} satisfying the triangle inequality such that $d_{ij} \leq d_{ik} + d_{kj},\ \forall i,\ k,\ j \in V$. A fleet K of vehicles, with $K = \{1, 2, \ldots, m\}$, initially located at the depot can serve demand from customers. Each vehicle $k \in K$ can only visit a subset $V_k \subset V^+$, based on matches of vehicles and service types. In our problem, a routing decision assigns each vehicle a route such that: (a) the nodes visited by vehicle $k \in K$ are in the set V_k; (b) each node must be visited exactly once; and (c) the maximum distance of all assigned routes is minimized.

We define the vector $x = (x_{ij}^k,\ (i, j) \in E,\ k \in K)^T$ where x_{ij}^k takes value 1 if edge $(i, j) \in E$ is used in the route for vehicle k, and 0 otherwise. Consider the binary parameter $u = (u_i^k,\ k \in K,\ i \in V^+)^T$ where u_i^k takes value 1 if $i \in V_k$ for vehicle $k \in K$, and 0 otherwise. Let τ represent the maximum distance of all the routes. We formulate a mixed-integer program for the VRPCC as follows.

$$\textbf{(MIP)} \qquad \underset{x,\tau}{\text{minimize}}\ \tau \tag{1}$$

$$\text{subject to} \sum_{(i,j)\in E} d_{ij}x_{ij}^k \le \tau \qquad \forall k \in K, \tag{2}$$

$$\sum_{j:(v,j)\in E} x_{vj}^k - \sum_{i:(i,v)\in E} x_{iv}^k = 0 \qquad \forall v \in V,\ \forall k \in K, \tag{3}$$

$$\sum_{k\in K}\sum_{i:(i,j)\in E} x_{ij}^k = 1 \qquad \forall j \in V^+, \tag{4}$$

$$\sum_{i:(i,j)\in E} x_{ij}^k \le u_j^k \qquad \forall j \in V^+,\ \forall k \in K, \tag{5}$$

$$\sum_{(i,j)\in E, i,j\in S} x_{ij}^k \le |S| - 1 \qquad \forall S \subset V^+,\ \forall k \in K, \tag{6}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall (i,j) \in E,\ \forall k \in K, \tag{7}$$

where constraints (2) ensure that τ equals to the maximum distance of all the routes, which is minimized in the objective function (1); constraints (3) enforce flow balance at each node for routing each vehicle; constraints (4) ensure that each node is visited exactly once; constraints (5) ensure that vehicle k can only visit nodes in V_k; constraints (6) are sub-tour elimination constraints.

3 Branch-Cut-and-Price Algorithm

We describe a BCP approach for VRPCC based on a set partition formulation where each decision variable represents a feasible route [18]. Let P^k be the set containing all feasible routes for vehicle $k \in K$. We define the binary decision variable $\lambda = (\lambda_p,\ p \in P^k,\ k \in K)^T$ where λ_p takes value 1 if route $p \in P^k$ is used by vehicle k and 0 otherwise. Denote the binary parameter $a = (a_{ip},\ i \in V,\ p \in P^k,\ k \in K)^T$ where a_{ip} takes value 1 if $p \in P^k$ visits node $i \in V$, and 0 otherwise. We consider c_p as the cost of route $p \in P^k,\ k \in K$. The set partition formulation is given by

$$\textbf{(SP)} \quad \underset{\lambda,\tau}{\text{minimize}} \quad \tau \tag{8}$$

$$\text{subject to} \sum_{k\in K}\sum_{p\in P^k} a_{jp}\lambda_p = 1 \qquad \forall j \in V^+, \tag{9}$$

$$\sum_{p\in P^k} c_p\lambda_p \le \tau \qquad \forall k \in K, \tag{10}$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in P^k,\ \forall k \in K, \tag{11}$$

where constraints (9) enforce that each node is visited exactly once and constraints (10) enforce that τ is the maximum cost of all routes. Due to the exponential size of $P^k,\ k \in K$, we exploit the BCP method to solve SP.

3.1 Column Generation

The idea of column generation is to maintain a subset $\bar{P}^k \subset P^k$ for each $k \in K$. We solve SP with P^k replaced by $\bar{P}^k$ and detect whether any $p \in P^k \backslash \bar{P}^K$ can

improve the solution: if yes, we add those favorable routes p to $\bar{P}^k$ and repeat the process; otherwise we claim optimality. We define a restricted master problem (RMP) as linear relaxation of SP with P^k replaced by $\bar{P}^k$ for all $k \in K$.

Clearly, any feasible primal solution to RMP is feasible to the linear relaxation of SP, but this is not necessarily true for their dual solutions. For each vehicle $k \in K$, we want to find if there exists some favorable route $p \in P^k \backslash \bar{P}^k$ that can improve the objective value of RMP. We define α, β as the dual variables corresponding to constraints (9) and (10), respectively, and define the decision variable $y = (y_{ijp},\ (i,j) \in E,\ p \in P^k,\ k \in K)^T$ where y_{ijp} takes value 1 if $(i,j) \in p$ and 0 otherwise. For each vehicle $k \in K$, with $\hat{\alpha}, \hat{\beta}$ being current optimal dual solution, the problem of pricing out a new route p (if exists) is given by

$$\textbf{(PP)} \quad z^k_{\mathrm{PP}}(\hat{\alpha}, \hat{\beta}) = \min_y \left\{ \hat{\beta}_k \sum_{(i,j)\in E} d_{ij} y^k_{ijp} - \sum_{(i,j)\in E, j\neq 0} \hat{\alpha}_j y^k_{ijp} \mid p \in P^k \right\} \tag{12}$$

If the optimal objective value of PP is less than 0, then route p can potentially improve the current best solution to RMP. For each vehicle $k \in K$, PP is equivalent to the problem finding the shortest path in $G(V, E)$, where the cost of the edge is replaced by its reduced cost $\bar{c}_{ij} = \hat{\beta}_k d_{ij} - \hat{\alpha}_j$ for each edge $(i,j) \in E,\ j \in V^+$, and $\bar{c}_{i0} = \hat{\beta}_k d_{i,0}$ for each edge $(i,0) \in E$. Since edge costs can be negative in PP, this problem is NP-hard [2]. Solution approaches for PP have been studied in [5, 10]. In this paper, we identify and generate the so-called "ng-route" by implementing the following procedure. Suppose that we assign a neighbor set to each node. An *ng-route* is a route where a node $i \in V^+$ can be revisited after a vehicle visits a node whose neighbor set does not contain node i. A label-setting algorithm, as shown in [5], can solve *ng-route* relaxation to PP that would improve the solution time of RMP.

3.2 Cutting Planes

Valid inequalities help improving the quality of the solution produced by RMP. We add subset-row inequalities from [17] as valid cuts when solving RMP. For any set $S = \{i_1, i_2, i_3\}$ containing three vertices $i_1, i_2, i_3 \in V^+$, the corresponding subset-row inequalities ensure that the sum of corresponding variables of all selected routes that visit at least two vertices in S is at most 1. With set I_S containing all the routes in $\bigcup_{k\in K} \bar{P}^k$ that visit at least two nodes in S, subset-row inequalities are in the form of:

$$\sum_{k\in K} \sum_{p\in I_S} \lambda_p \leq 1 \qquad \forall S \subseteq V,\ |S| = 3. \tag{13}$$

We use the algorithm from [21] to solve PP with additional dual variables associated with the added (13).

3.3 Branching Rule

We adopt the following branching rules from [9]: we calculate the sum of flows for each edge as $f_{ij}^k = \sum_{p \in P^k} a_{ijp} \lambda_p$ for all $(i,j) \in E$ and $k \in K$. If f_{ij}^k is fractional, we generate two branches: $f_{ij}^k = 1$, where vehicle k has to use edge (i,j), and $f_{ij}^k = 0$, where we exclude edge (i,j) from any route traveled by vehicle $k \in K$.

4 Approximation Algorithm

Our later computational results show that a straightforward implementation of BCP is time consuming for VRPCC. Here we propose an $O(\log n)$-approximation algorithm, where we recall that n is the number of customer locations. The algorithm is based on the solution approach for the following problem defined on a network $G = (V, E)$.

Problem 1. Maximum Budgeted Cover Problem
Input: A node subset $X \subset V$, a fleet K of vehicles and a budget $B \geq 0$.
Output: A set H of routes, one for each vehicle $k \in K$, where each route has cost less than B.
Objective: Maximize $|H \cap X|$.

A greedy 2-approximation algorithm for this problem follows from [6]. This algorithm was based on the idea of iteratively picking a route that covers the maximum number of remaining nodes for the current node subset X. We propose a variant of this algorithm which is faster and still achieves a 2-approximation.

Our greedy algorithm works with an oracle for the orienteering problem, where $\mathcal{O}(X, B, i)$ outputs a route, with cost less than B, for vehicle $i \in K$, which covers the maximum number of nodes in X. The greedy algorithm, $Greedy(X, B)$, for the maximum budgeted cover problem is described in Algorithm 1.

Algorithm 1: A greedy algorithm for maximum budgeted cover

input : A fleet K of vehicles, a subset $X \subset V$ and a budget of route B
output: A set H of routes with cost less than B, one route for each vehicle

$H \leftarrow \emptyset$, $X' \leftarrow X$
for i *in* K **do**
  $A_i = \mathcal{O}(X', B, i)$
  $H \leftarrow H \cup \{A_i\}$, $X' \leftarrow X' \backslash A_i$
end
return H

We propose the following lemmas to analyze the above algorithm. The proofs are deferred to a full version.

Lemma 1. *Algorithm 1 is a 2-approximation algorithm for the maximum budgeted cover problem.*

Lemma 2. *Greedy Algorithm needs to be executed at most* $\log|X|+1$ *times to cover all nodes in* X *with sufficient large budget* B*.*

Next, we propose an approximation algorithm for VRPCC based on Algorithm 1. We set $X = V^+$, where recall that V^+ is the set of customer locations. If budget B is sufficiently large that an optimal solution of maximum budgeted cover problem covers every node in X, then by Lemma 2, we can use Algorithm 1 to produce a feasible solution to VRPCC, where each vehicle carries routes with total cost at most $(\log n + 1)B$. We apply binary search to find the smallest budget B. We detail the algorithmic steps in Algorithm 2.

Algorithm 2: An approximation algorithm for VRPCC

input : A network $G = (V, E)$, a fleet set of vehicles K, an budget B
output: Routing assignment for each vehicle in $k \in K$

1 Initialize $S_k = \emptyset$ for all $k \in K$, $X \leftarrow V^+$, $Solve \leftarrow$ **true**
2 **while** $X \neq \emptyset$ **do**
3 $\quad H \leftarrow$ output of Algorithm 1 with input K, X, B
4 $\quad n = |X|$, $X \leftarrow X \backslash H$
5 $\quad$ **if** $|X| > \frac{n}{2}$ **then** $Solve \leftarrow$ **false**, go to Step 8
6 $\quad$ Update S_k with $A_k \in H$ for all $k \in K$
7 **end**
8 Apply binary search to find optimal B based on the result of $Solve$, go to Step 1 if B is not optimal
9 Shortcut the routes in S_k, $k \in K$ to produce solution for VRPCC

The approximation factor of Algorithm 2 depends on the oracle for orienteering problem in Algorithm 1. If we use a "bicriteria" approximation for orienteering that covers the optimal number of nodes with cost at most βB then Algorithm 2 yields an approximation factor of $\beta(\log n + 1)$ following the result from Lemma 2. In particular, we use a procedure from [14] which implies $\beta = 3$.

5 Numerical Results

We conduct numerical experiments to evaluate the performance of directly computing the MIP, the BCP algorithm, and the approximation algorithm. We conduct experiments on Solomon's instances [23] adapted to VRPCC settings. The Solomon's instances are classified into three classes according to different distributions of customer locations: random distribution (R), clustered distribution (C), and a mix of both (RC). The customer locations are distributed on a $[0, 100]^2$ square. In the Solomon's instances, there are 100 locations for each instance. We pick the first n locations to test our algorithms, where $n \in \{10, 15, 20, 25, 30\}$. We also test the performance of our algorithms on different fleet sizes $m \in \{3, 5\}$. For each location $v \in V^+$, we randomly pick two vehilces that are capable of serving the corresponding customer. In practice, the average productivity of a medical home crew ranges from 4–6 visits per day [11]. Therefore, the sizes of our test instances are close to real-world medical home care instances.

In our numerical experiments, for each test instance, we first solve the problem with model MIP and use the result as a benchmark. We use Gurobi 6.5 [16] as the optimization solver. For the BCP algorithm, we use Gurobi as a linear programming solver of RMP at each branched node, and managing the branching process by following the branching rule in Section 3.3. For the approximation algorithm, we apply a procedure by Garg [14] that solves orienteering problem for optimal number of nodes but with a budget $5B$ in Algorithm 1. We set the running time limit for all programs as 1,200 seconds. We use Java and perform numerical test on a Dell desktop with an Intel i7-3770 processor and 16 GB memory.

For each instance, we report the following information: for MIP and BCP, we report their best upper bounds ("UB") and lower bounds ("LB") achieved, their gaps ("Gap") and computational time ("Time(s)"); for approximation algorithm, we report the objective values of its solutions ("Obj") and computational time ("Time(s)"). Tables 1 and 2 summarize the numerical performances of the three methods. We use "–" to indicate that time limit for the computation of the instance is reached.

Table 1. Numerical results with $m = 3$

		MIP				BCP				Approx.	
Type	n	LB	UB	Gap	Time(s)	LB	UB	Gap	Time(s)	Obj	Time(s)
R101	10	87.00	87.00	0.00%	0.15	87.00	87.00	0.00%	4.26	147.00	0.16
	15	100.00	100.00	0.00%	3.77	100.00	100.00	0.00%	288.20	155.00	0.19
	20	114.00	114.00	0.00%	26.94	104.00	143.00	27.27%	–	200.00	0.57
	25	140.00	140.00	0.00%	912.77	122.00	223.00	45.29%	–	230.00	1.10
	30	138.00	153.00	9.80%	–	132.58	252.00	47.39%	–	247.00	1.83
C101	10	40.00	40.00	0.00%	1.12	40.00	40.00	0.00%	6.58	48.00	0.18
	15	59.00	79.00	25.32%	–	40.84	80.00	48.95%	–	93.00	0.20
	20	42.00	87.00	51.72%	–	53.00	93.00	43.01%	–	102.00	0.58
	25	44.00	93.00	52.69%	–	57.71	109.00	47.06%	–	111.00	1.20
	30	44.00	94.00	53.19%	–	62.41	136.00	54.11%	–	140.00	3.48
RC101	10	87.00	87.00	0.00%	0.63	87.00	87.00	0.00%	5.90	131.00	0.14
	15	119.00	119.00	0.00%	1137.84	71.02	120.00	40.82%	–	171.00	0.27
	20	112.00	132.00	15.15%	–	86.48	132.00	34.48%	–	199.00	0.43
	25	92.00	157.00	41.40%	–	105.27	192.00	45.17%	–	261.00	1.40
	30	137.00	218.00	37.16%	–	126.90	332.00	61.78%	–	282.00	1.88

In Table 1, when $m = 3$, we see that MIP performs well for instances of type R, in which it can solve up to 25-node instances. For instances of types C and RC, we can no longer use MIP to solve instances with more than 15 nodes. This shows that clustered distribution of nodes is more challenging to handle. At the same time, BCP does not outperform MIP, and we can only use BCP to solve instances with 15 nodes or fewer. Although BCP algorithm cannot be solved very efficiently yet, it can produce better lower bound for instances that both MIP and BCP cannot solve to optimality, e.g., instances with 20, 25, and 30 nodes of type C. On the other hand, approximation algorithm solves all the instances very quickly.

Table 2. Numerical results with $m = 5$

		MIP				BCP				Approx.	
Type	n	LB	UB	Gap	Time(s)	LB	UB	Gap	Time(s)	Obj	Time(s)
R	10	69.00	69.00	0.00%	0.02	69.00	69.00	0.00%	0.74	119.00	0.04
	15	97.00	97.00	0.00%	0.23	97.00	97.00	0.00%	115.94	162.00	0.09
	20	106.00	106.00	0.00%	3.75	93.00	116.00	19.83%	–	210.00	0.25
	25	120.00	120.00	0.00%	29.45	98.00	142.00	30.99%	–	220.00	0.50
	30	123.00	123.00	0.00%	59.36	103.92	148.00	29.78%	–	193.00	0.78
C	10	40.00	40.00	0.00%	0.04	40.00	40.00	0.00%	0.99	42.00	0.03
	15	78.00	78.00	0.00%	4.71	78.00	78.00	0.00%	465.68	85.00	0.10
	20	82.00	82.00	0.00%	914.80	50.00	85.00	41.18%	–	96.00	0.22
	25	62.00	85.00	27.06%	–	52.38	123.00	57.41%	–	127.00	0.45
	30	61.00	87.00	29.89%	–	55.43	106.00	47.71%	–	110.00	1.00
RC	10	83.00	83.00	0.00%	0.09	83.00	83.00	0.00%	2.08	115.00	0.03
	15	105.00	105.00	0.00%	7.00	105.00	105.00	0.00%	344.88	131.00	0.12
	20	130.00	130.00	0.00%	25.76	94.33	169.00	44.18%	–	197.00	0.28
	25	137.00	139.00	1.44%	–	95.28	202.00	52.83%	–	246.00	0.50
	30	167.00	171.00	2.34%	–	109.27	328.00	66.69%	–	307.00	0.82

The computation can be finished within 4 seconds for all instances, and there is no significant difference in computational time when solving instances of type C/RC than type R. Moreover, despite that the theoretical approximation bound of our algorithm could be large when n increases, the practical approximation ratio is within a factor of 2 when comparing the objective values of solutions produced by the approximation algorithm with the best lower bounds produced by MIP and BCP for all instances.

Table 2 summarizes the numerical results for instances with larger fleet size $m = 5$. Results show that the computational time has been significantly improved and more instances can be solved by MIP and BCP. Our approximation algorithm can solve all instances within one second, and the practical approximation ratio is still within a factor of 2 comparing its objective values of solutions with the best lower bounds of MIP and BCP.

To summarize, we recognize that different distributions of demand locations could affect the computational time of our proposed exact solution methods. Despite lacking efficient exact solution approaches for VRPCC, our proposed approximation algorithm can efficiently solve the problem and provides good practical solutions.

6 Conclusions and Future Research

In this paper, we formulated a minimum makespan routing problem with compatibility constraints. We proposed three solution approaches for the problem: MIP, BCP, and an approximation algorithm. Numerical results showed that MIP and BCP could not solve many of these instances to optimality (within our time limit), whereas the approximation algorithm obtained good quality solutions

within seconds. Moreover, our approximate solutions (for these instances) are within two times the best lower bounds from MIP or BCP. Future research includes further investigation on BCP to improve its efficiency and designing good implementations to improve the practical approximation factor of the approximation algorithm.

Bibliography

[1] N. R. Achuthan, L. Caccetta, and S. P. Hill. An improved branch-and-cut algorithm for the capacitated vehicle routing problem. *Transportation Science*, 37(2):153–169, 2003.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: Theory, algorithms, and applications*. Prentice hall, 1993.

[3] D. Applegate, W. Cook, S. Dash, and A. Rohe. Solution of a min-max vehicle routing problem. *INFORMS Journal on Computing*, 14(2):132–143, 2002.

[4] E. M. Arkin, R. Hassin, and A. Levin. Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms*, 59(1):1–18, 2006.

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

[6] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 72–83. Springer, 2004.

[7] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, 1959.

[8] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Min–max tree covers of graphs. *Operations Research Letters*, 32(4):309–315, 2004.

[9] D. Feillet. A tutorial on column generation and branch-and-price for vehicle routing problems. *A Quarterly Journal of Operations Research*, 8(4):407–424, 2010.

[10] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[11] N. A. for Home Care & Hospice. Basic statistics about home care. *Washington, DC: National Association for Home Care & Hospice*, pages 1–14, 2010.

[12] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. In *Foundations of Computer Science, 1976., 17th Annual Symposium on*, pages 216–227. IEEE, 1976.

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

[14] N. Garg. A 3-approximation for the minimum tree spanning k vertices. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, FOCS '96, pages 302–309, Washington, DC, USA, 1996. IEEE Computer Society.

[15] B. L. Golden, S. Raghavan, and E. A. Wasil. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer Science & Business Media, 2008.

[16] Gurobi Optimization, Inc. Gurobi optimizer reference manual, 2016. URL `http://www.gurobi.com`.

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

[18] B. Kallehauge, J. Larsen, O. B. Madsen, and M. M. Solomon. Vehicle routing problem with time windows. In *Column Generation*, pages 67–98. Springer, 2005.

[19] A. N. Letchford, R. W. Eglese, and J. Lysgaard. Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming*, 94(1):21–40, 2002.

[20] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[21] D. Pecin, A. Pessoa, M. Poggi, and E. Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 393–403. Springer, 2014.

[22] T. K. Ralphs, L. Kopman, W. R. Pulleyblank, and L. E. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94(2-3): 343–359, 2003.

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

[24] P. Toth and D. Vigo. *Vehicle Routing: Problems, Methods, and Applications*. SIAM, 2014.