The Split Delivery Vehicle Routing Problem: Applications, Algorithms, Test Problems, and Computational Results

Si Chen and Bruce Golden
R. H. Smith School of Business, University of Maryland, College Park, Maryland 20742

Edward Wasil
Kogod School of Business, American University, Washington, DC 20016

In the split delivery vehicle routing problem (SDVRP), a customer's demand can be split among several vehicles. In this article, we review applications of the SDVRP including the routing of helicopters in the North Sea and solution methods such as integer programming and tabu search. We develop a new heuristic that combines a mixed integer program and a record-to-record travel algorithm. Our heuristic produces high-quality solutions to six benchmark problems that have 50–199 customers and generally performs much better than tabu search. On five other problems for which lower bounds exist, our heuristic obtains solutions within 5.85%, on average. Finally, we generate 21 new test problems that have 8–288 customers. A near-optimal solution can be visually estimated for each problem. We apply our heuristic to these new problems and report our computational results. © 2007 Wiley Periodicals, Inc. NETWORKS, Vol. 49(4), 318–329 2007

Keywords: vehicle routing problem; integer programming; heuristics

1. INTRODUCTION

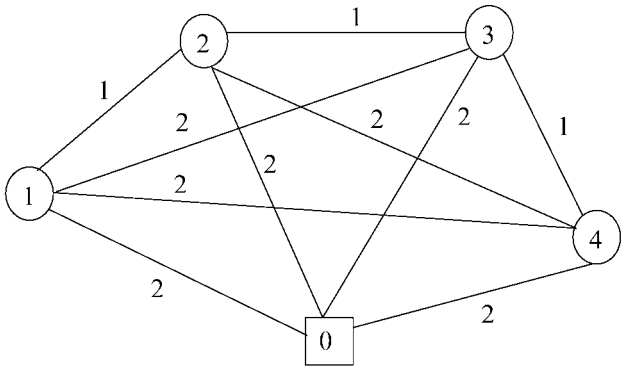
In the standard version of the vehicle routing problem (VRP), a homogeneous fleet of vehicles is based at a single depot. Each vehicle has a fixed capacity and must leave from and return to the depot. Each customer has a known demand and is serviced by exactly one visit of a single vehicle. A sequence of deliveries (known as a *route*) must be generated for each vehicle so that all customers are serviced and the total distance traveled by the fleet is minimized. Recent algorithmic developments and computational results for heuristics that solve the VRP are covered by Cordeau et al. [5].

Received January 2006; accepted September 2006
Correspondence to: B. Golden; e-mail: bgolden@rhsmith.umd.edu
DOI 10.1002/net.20181
Published online in Wiley InterScience (www.interscience.wiley.com).

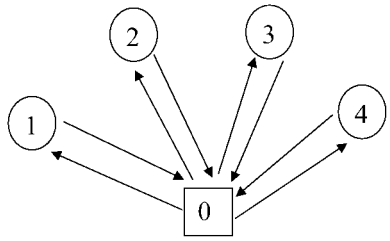
In the split delivery vehicle routing problem (SDVRP), a customer can be serviced by more than one vehicle, that is, a customer's demand can be split among several vehicles. By allowing split deliveries, the potential exists to use fewer vehicles and to reduce the total distance traveled by the fleet. We point out that, in general, for both the VRP and SDVRP, using fewer vehicles does not necessarily reduce the total distance (even if the triangle inequality holds). The SDVRP is NP-hard [8] and is difficult to solve optimally. In Figure 1, we show an example from [1] which illustrates the savings in distance and number of vehicles that can be achieved by using split deliveries. In this article, we consider an undirected graph (our algorithm also works for a directed graph).

Over the last decade or so, researchers have modeled practical problems as SDVRPs. Mullaseril et al. [14] studied the distribution of feed to cattle at a large livestock ranch in Arizona. About 100,000 head of cattle are kept in large pens that are connected by a road network. Six trucks deliver feed to the pens within a specified time window each day. Because of feed weighing and loading inaccuracies, the last pen on a route may not receive its full load and would need to have the rest of its load delivered by a second truck on a different route. The authors modeled this feed distribution problem as a capacitated rural postman problem with split deliveries and time windows. They developed a solution algorithm that uses k-split interchange and route addition (these are discussed in the next section). The results of computational experiments with five types of feed and time windows show that allowing split deliveries significantly reduces the total distance traveled by the fleet in four of five cases.

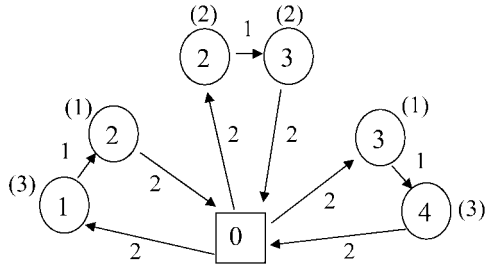
In the North Sea, off the coast of the Netherlands, there are 51 platforms for the production of natural gas. Each platform has a regular crew of 20–60 employees and each employee works every other week. Crew members are exchanged each week by helicopters based at an airport near Amsterdam. One person leaving the platform is exchanged for one person arriving for work at the platform. Each helicopter has a fixed capacity of 27 seats (23 seats are available for crew



(a) Node 0 is the depot. Customers 1, 2, 3, and 4 have a demand of 3 units. Vehicle capacity is 4 units. Distance between i and j is adjacent to the edge.



(b) VRP optimal solution with four vehicles and a total distance of 16.



(c) SDVRP optimal solution with three vehicles and a total distance of 15.

FIG. 1. Example from [1] showing split deliveries.

exchanges and four seats are left free for emergencies and cargo) and there are enough helicopters to make all of the required exchanges in a week. In addition, each helicopter has a specified fuel capacity that limits the range it can fly.

Sierksma and Tijssen [16] modeled crew exchanges as a discrete split delivery routing problem. They formulated the problem as an integer program and solved a relaxed linear program using column generation and a rounding procedure to produce integer solutions (denoted by CGRP). In addition, they developed a cluster-and-route procedure (denoted by CR).

Sierksma and Tijssen conducted a computational experiment with 11 different simulated data sets (51 platforms and 2–50 crew exchanges per platform). CGRP, CR, a sweep heuristic, and the Clarke–Wright algorithm are applied to each problem and the resulting solution is then improved by several post-processors including two-opt. Over all 11 instances, CGRP has the smallest average deviation from a lower bound for each problem, closely followed by CR. However, the running time of CR is very fast (a few seconds for the entire procedure), while the running time of CGRP is very slow (about 50 s for one solution).

Song et al. [17] considered the distribution of newspapers from printing plants to agents. They carried out a case study for Chosun Ilbo, a major newspaper in South Korea. Chosun Ilbo has three printing plants and 400 agents around Seoul, and it prints three local editions. Printing starts around midnight and finishes by 3:00 a.m. Newspapers are then dispatched to agents at 2:30 a.m., 3:00 a.m., and 3:30 a.m. Agents insert supplements into the newspapers, which are then delivered to residential (home) customers.

Some agents who are close to the printing plants are considered for split delivery of newspapers with a first dispatch at 2:30 a.m. and a second dispatch at 3:30 a.m. (at most two deliveries are allowed). The split delivery spreads out the work of inserting supplements and allows home delivery to start earlier. Agents far from the printing plants and agents with small demands are not considered for split deliveries.

Song et al. used a two-phase solution procedure. In Phase I, they allocated agents to plants by solving a 0–1 integer programming problem. In Phase II, the authors determined the split deliveries, generated the vehicle routes using a modified savings rule and a weighted savings rule, and scheduled the vehicles for dispatch. In the case study for Chosun Ilbo, the authors used a geographic information system to obtain road distances. Compared with the manual method used by Chosun Ilbo, the allocation, routing, and scheduling procedure reduced the delivery cost by an average of 15%.

In commercial sanitation collection, businesses and organizations often place their trash in large containers or bins. An office building may have several of these bins. Each bin is lifted and its contents are emptied into the trash truck. Since the bins are large, several trucks may be required to pick up all of the trash at a particular office building. However, a bin's load cannot be split. This problem can be modeled in two ways. Either each bin is considered as a separate demand or the office building is handled as a single demand. In the latter case, a discrete number of splitting options is allowed (L. Levy, RouteSmart Technologies, private communication, 2006).

In Section 2, we review algorithms that have been used recently to solve the SDVRP. In Section 3, we formulate a mixed integer program for the SDVRP. In Section 4, we develop a new heuristic for solving the SDVRP that combines our mixed integer program and a record-to-record travel algorithm. We report computational results for our new heuristic on benchmark problems that have 50–199 customers and on five other problems for which lower bounds exist. In Section 5, we generate 21 new test problems that have 8–288 customers. A near-optimal solution can be visually estimated for each problem. We apply our new heuristic to these 21 problems and report our results. In Section 6, we summarize our contributions.

2. SOLVING THE SDVRP

Over the last 15 years or so, numerous procedures including tabu search have been used to solve the SDVRP. In this

section, we review the procedures and their computational results.

Dror and Trudeau [7,8] developed an algorithm for solving the SDVRP that uses a k-split interchange and route addition. A k-split interchange splits the demand of customer i among k routes as long as the sum of the spare (or residual) capacities of the k routes is not less than the demand of customer i and there is a positive savings in distance when customer i is removed from its current route and added to the k routes. Route addition serves as an inverse operation to a k-split interchange. A route is added to eliminate the split delivery of a customer if the addition of the route reduces the total distance.

The authors use a two-stage algorithm to solve the SDVRP. In Stage 1, an initial solution to the VRP is constructed and then customer interchanges and route improvements are considered. In Stage 2, k-split interchanges, route additions, customer interchanges, and route improvements are used to produce a solution to the SDVRP.

Dror and Trudeau conducted computational experiments on three problems with 75, 115, and 150 customers and a vehicle capacity of 160. The 75-point problem is taken from [9]. The 150-point problem and 115-point problem are obtained from the 75-point problem. Customer demand is generated according to six scenarios: [0.01–0.1], [0.1–0.3], [0.1–0.5], [0.1–0.9], [0.3–0.7], and [0.7–0.9]. For example, in the first scenario (very small customer demand), the demand for customer i is randomly selected from a uniform distribution on the interval $[160(0.01), 160(0.1)] = [1.6, 16]$. For each problem (75, 115, 150 customers), 30 instances are generated for each of the six scenarios. The two-stage algorithm is then applied to the 180 instances for each size problem. Dror and Trudeau solved each problem twice—as an SDVRP and a VRP. They observe that, when customer demand is small relative to vehicle capacity, there are almost no split deliveries. When customer demand is very large (e.g., [0.7–0.9]), split deliveries occur thereby reducing the total distance traveled (e.g., for the 75-customer problem, an average distance savings of 11.24% over the VRP solution) and reducing the number of vehicles used (e.g., for the 75-customer problem, an average reduction of 14.07 vehicles over the VRP solution).

Frizzell and Giffin [10] considered the SDVRP on a grid network and introduced upper and lower bounds on the size of a split delivery to a customer. They developed a construction heuristic and a splitting cost heuristic that assigned specific splitting costs for each customer. Frizzell and Giffin tested their heuristics on a set of 1,050 problems and found that the solutions generated by their construction heuristic reduced the total distance and number of vehicles, when compared with the solutions of a greedy construction heuristic.

Dror et al. [6] formulated the SDVRP as an integer linear program with several new classes of valid constraints. The authors develop a constraint relaxation algorithm using branch and bound for solving the SDVRP exactly, which uses the solution from the algorithm of Dror and Trudeau [7] as a first upper bound on the optimal solution value of the SDVRP. Computational experiments are run on three problems with 10, 15, and 20 customers and varying customer demands taken from the 75-point problem of Eilon et al. [9]. The computational results show that various constraints could successfully reduce the gap between the lower and upper bounds of the optimal solution value of SDVRP. In addition, the results establish the high quality of the solutions generated by the Dror and Trudeau algorithm.

Frizzell and Giffin [11] studied the SDVRP with time windows where customers are located on a grid network. They developed a construction heuristic for solving the problem. The solution that emerges from the construction heuristic is improved by either removing one customer from a route and inserting it on another route or exchanging two customers from two routes. The authors generate 6,480 problems and use five measures including drive time and number of routes to evaluate the performance of their heuristic. They also report results on six benchmark VRPs that have time windows.

Belenguer et al. [2] proposed a lower bound for the SDVRP based on a polyhedral study of the problem. They introduced a new family of valid inequalities and proposed a cutting-plane algorithm for generating a lower bound to the SDVRP. The algorithm is tested on 11 instances from TSPLIB and 14 random instances with 50, 75, and 100 customers (these instances are similar to the problems in [7]). The algorithm performs reasonably well with the average gap between the upper bound and the lower bound for the TSPLIB problems about 3% and about 8% for random instances.

Archetti et al. [1] developed a tabu search algorithm (called SPLITABU) for solving the SDVRP. Their algorithm has three phases: (1) Initial solution phase: Create as many direct trips to customers as possible and then solve a giant TSP tour using the GENIUS algorithm to produce a feasible solution. (2) Tabu search phase: Remove a customer from a current set of routes and insert it on a new route or an existing route with available capacity in the cheapest way. Consider inserting a customer on a route without removing it from its current route. (3) Final improvement phase: Improve the solution from the second phase (e.g., apply the GENIUS algorithm to individual routes). In a variant called SPLITABU-DT, the authors improve solutions using the customer (node) interchanges of Dror and Trudeau [7,8] and two-opt.

There are only two parameters that need to be set in SPLITABU: the length of the tabu list and the maximum number of iterations.

Archetti et al. tested three variants of SPLITABU on seven problems with 50–199 customers (the problems are obtained from seven classical capacitated VRP instances: instances 1, 2, and 3 from [3] and instances 4, 5, 11, and 12 from [4], by considering the original demands of the customers and demands defined according to the rules proposed by Dror and Trudeau [7,8]), running each variant five times on each problem, and comparing the results produced by the variants with the results produced by Dror and Trudeau's algorithm (denoted by DT). Overall, SPLITABU-DT is the clear winner with a smaller total distance traveled than that of DT for every problem. On average, the best solutions produced by

SPLITABU-DT are 5.38% lower than the solutions generated by DT.

We point out that the recent dissertations by Liu [13] and Nowak [15] modeled variants of the SDVRP, developed solution procedures, and reported computational results.

3. AN ENDPOINT MIXED INTEGER PROGRAM FOR THE SDVRP

We start with an initial solution to the SDVRP, say a solution from the Clarke–Wright algorithm to the corresponding VRP instance. For each route in the initial solution, we consider its one or two *endpoints* and the c closest neighbors to each endpoint (c is a parameter whose value we need to set and the neighbors are endpoints). Each endpoint is allowed to reallocate its demand among its neighbors. After reallocation, there are three possibilities for each endpoint (we assume symmetric distances here): (1) No change is made. (2) The endpoint i is removed from its current route(s) and *all* of its demand is moved to another route or routes. (3) The endpoint i is partially removed from its current route(s) and *part* of its demand is moved to another route or routes. The reallocation process and the savings that result are illustrated in Figure 2.

We formulate an endpoint mixed integer program (EMIP), whose objective is to find a set of reallocation operations that maximizes the total savings. Let i and j be the endpoints of current routes, l_{ij} is the distance between i and j (we use Euclidean distances in our computational experiments; other types of distances could be used), R_i is the residual capacity on the route with i as an endpoint, D_i is the demand of endpoint i carried on its route, $p(i)$ is the predecessor of endpoint i, $s(i)$ is the successor of endpoint i, R is the set of routes, N is the set of endpoints, and $NC(i)$ is the set of c closest neighbors of endpoint i.

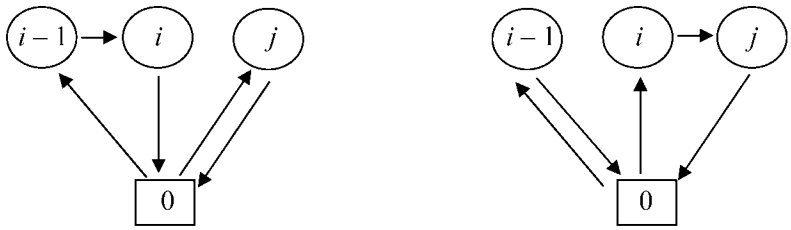
The decision variables are defined next. Let d_{ij} be the amount of i's demand moved before endpoint j, m_{ij} equals 1 if endpoint i is inserted before endpoint j and equals 0 otherwise, and b_i equals 1 if endpoint i's entire demand is removed from the route on which it was an endpoint and equals 0 otherwise.

The objective function and constraints of the EMIP are given by the following.

$$\text{maximize} \quad \sum_{i\in N} b_i(l_{is(i)} + l_{p(i)i} - l_{p(i)s(i)}) - \sum_{i\in N}\sum_{j\in NC(i)} m_{ij}(l_{ij} + l_{p(j)i} - l_{p(j)j}) \quad (1)$$

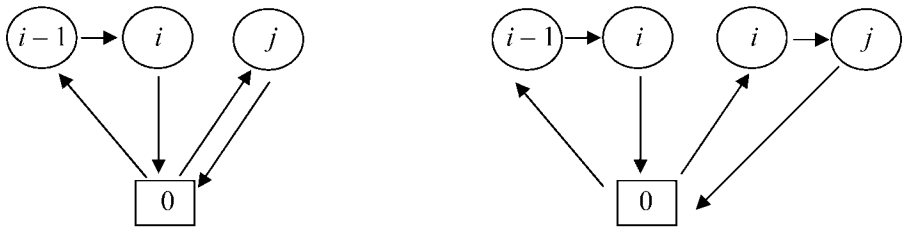
subject to

$$\sum_{i:j\in NC(i)} d_{ij} + \sum_{q:k\in NC(q)} d_{qk} - \sum_{l\in NC(k)} d_{kl} - \sum_{t\in NC(j)} d_{jt} \le R_r \quad \forall r \in R;\ k, j \text{ are two endpoints of route } r \quad (2)$$



savings = $(l_{i\text{-}1,i} + l_{i,0} - l_{i\text{-}1,0}) - (l_{0,i} + l_{i,j} - l_{0,j})$

(a) Endpoint i removed from its current route and all of its demand is moved to another route.



savings = $-(l_{0,i} + l_{i,j} - l_{0,j})$

(b) Part of the demand of endpoint i is moved to another route.

FIG. 2. Reallocating the demand of endpoint i to other routes.

$$\sum_{j \in NC(i)} d_{ij} \le D_i \quad \forall i \in N \tag{3}$$

$$\sum_{j \in NC(i)} d_{ij} \ge D_i \times b_i \quad \forall i \in N \tag{4}$$

$$D_i m_{ij} \ge d_{ij} \quad \forall i \in N, j \in NC(i) \tag{5}$$

$$1 - b_i \ge \sum_{j : i \in NC(j)} m_{ji} \quad \forall i \in N \tag{6}$$

$$1 - b_{p(i)} \ge \sum_{j : i \in NC(j)} m_{ji} \quad \forall i \in N \tag{7}$$

$$b_k + b_j \le 1 \quad \forall r \in R;\ k \text{ and } j \text{ are two endpoints of route } r \tag{8}$$

$$d_{ij} \ge 0 \quad \forall i \in N, j \in NC(i) \tag{9}$$

$$m_{ij} = 0, 1 \quad \forall i \in N, j \in NC(i) \tag{10}$$

$$b_i = 0, 1 \quad \forall i \in N \tag{11}$$

The objective function (1) maximizes the total savings from the reallocation process. In (2), the amount added to a route minus the amount taken away from a route must be less than or equal to the residual capacity. In (3), the amount diverted from an endpoint on a route must be less than or equal to the demand of that endpoint on the route. In (4), if an endpoint is removed from a route, then all of its demand must be diverted to other routes. In (5), if $d_{ij} > 0$ (that is, we move some of node i's demand before node j), then $m_{ij} = 1$ (that is, node i is inserted before node j). In (6), if node i is removed from a route, then no node can be inserted before node i. In (7), if the predecessor of node i is removed from a route, then no node can be inserted before node i. In addition, from (6) and (7), if endpoints i and $p(i)$ are not removed from their route, then at most one endpoint can be inserted before endpoint i. In (8), if a route has only two endpoints, then we cannot remove both endpoints at the same time.

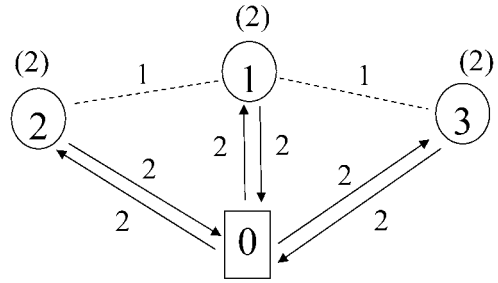
We now present the EMIP formulation of the small example given in Figure 3a.

$$\begin{aligned}\text{maximize} \quad & 2b_1 l_{01} + 2b_2 l_{02} + 2b_3 l_{03} - m_{12}(l_{01} + l_{12} - l_{02}) \\ & - m_{13}(l_{01} + l_{13} - l_{03}) - m_{21}(l_{02} + l_{21} - l_{01}) \\ & - m_{23}(l_{02} + l_{23} - l_{03}) - m_{31}(l_{03} + l_{31} - l_{01}) \\ & - m_{32}(l_{03} + l_{32} - l_{02})\end{aligned}$$

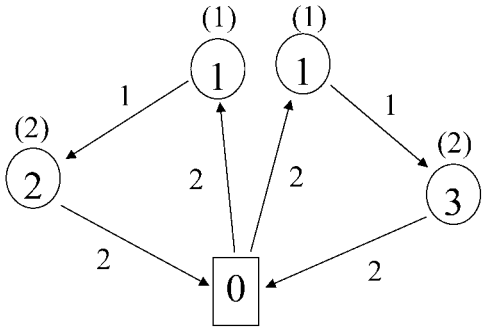
subject to

$$\begin{aligned} d_{21} + d_{31} - d_{12} - d_{13} &\le R_1 \\ d_{12} + d_{32} - d_{21} - d_{23} &\le R_2 \\ d_{13} + d_{23} - d_{31} - d_{32} &\le R_3 \end{aligned}$$

$$\begin{aligned} d_{12} + d_{13} &\le D_1 \\ d_{21} + d_{23} &\le D_2 \\ d_{31} + d_{32} &\le D_3 \end{aligned}$$



(a) Node 0 is the depot. Customers 1, 2, and 3 have a demand of two units each. Vehicle capacity is three units. Distance between nodes is adjacent to the edge. The Clarke-Wright solution is given by the solid edges and has a total distance of 12.



(b) Optimal solution. The demand for customer 1 is split between two routes. The total distance is 10.

FIG. 3. Small example illustrating the EMIP formulation and an optimal solution.

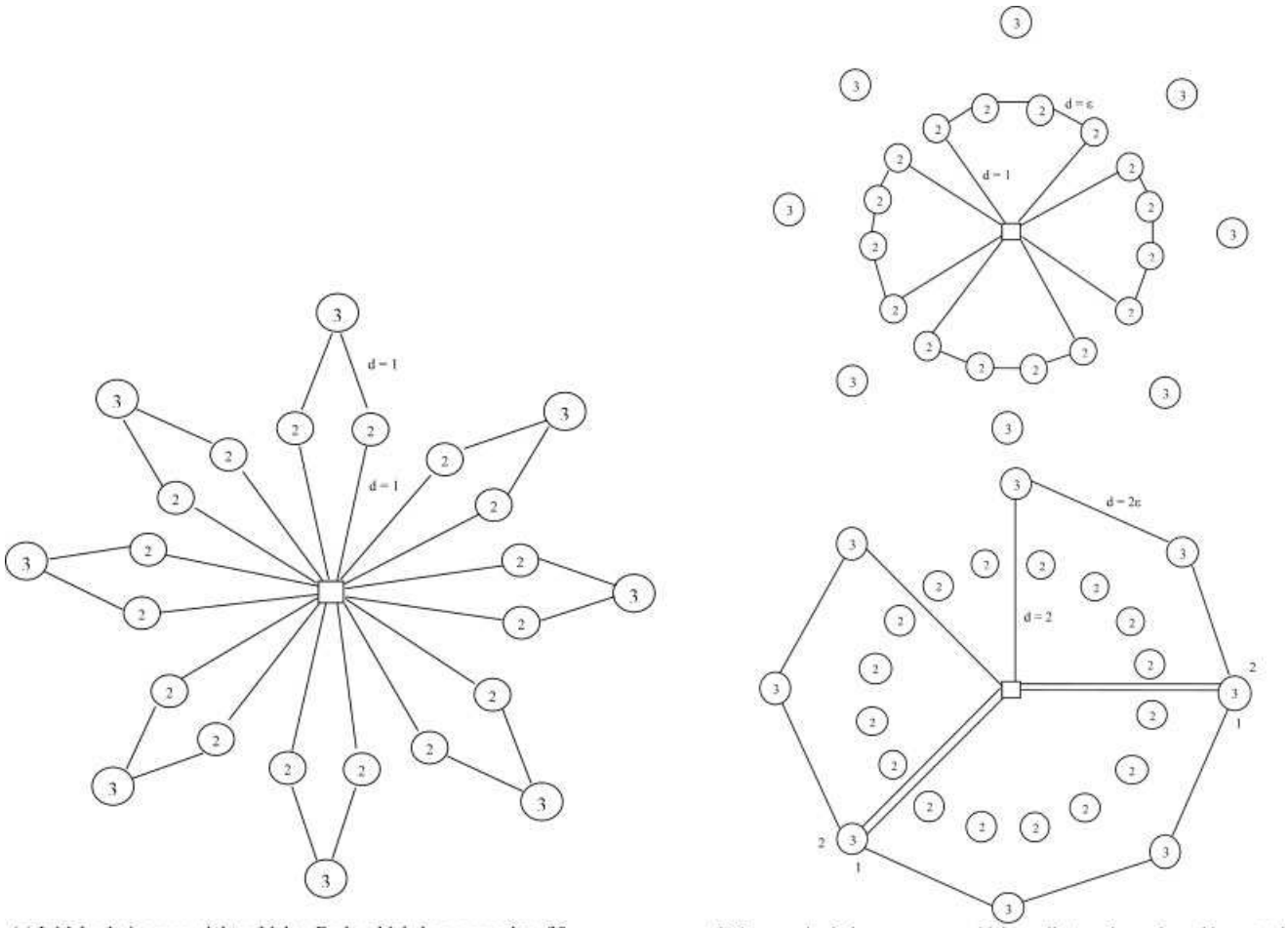
$$\begin{aligned} d_{12} + d_{13} &\ge D_1 b_1 \\ d_{21} + d_{23} &\ge D_2 b_2 \\ d_{31} + d_{32} &\ge D_3 b_3 \end{aligned}$$

$$\begin{aligned} D_1 m_{12} &\ge d_{12} \\ D_1 m_{13} &\ge d_{13} \\ D_2 m_{21} &\ge d_{21} \\ D_2 m_{23} &\ge d_{23} \\ D_3 m_{31} &\ge d_{31} \\ D_3 m_{32} &\ge d_{32} \end{aligned}$$

$$\begin{aligned} 1 - b_1 &\ge m_{21} + m_{31} \\ 1 - b_2 &\ge m_{32} + m_{12} \\ 1 - b_3 &\ge m_{23} + m_{13} \end{aligned}$$

$$\begin{aligned} d_{ij} &\ge 0 \quad \text{for} \quad i, j = 1, 2, 3 \\ b_i &= 0, 1 \quad \text{for} \quad i = 1, 2, 3 \\ m_{ij} &= 0, 1 \quad \text{for} \quad i, j = 1, 2, 3 \end{aligned}$$

For this example, we have $R_i = 1$ and $D_i = 2$ for $i = 1$, 2, 3. The distances are given by $l_{01} = 2$, $l_{02} = 2$, $l_{03} = 2$,



(a) Initial solution uses eight vehicles. Each vehicle has a capacity of 8. The demand of each customer (written inside the circles representing nodes) is either two units or three units. With a distance of 1 for each edge (d =1), the total distance of this solution is 32.

(b) Improved solution uses seven vehicles, splits two demands, and has a total distance of 20 + 26ε. When ε < 6/13, the total distance is less than 32.

FIG. 4. Example illustrating the limitation of EMIP.

$l_{10} = 2$, $l_{12} = 1$, $l_{13} = 1$, $l_{20} = 2$, $l_{21} = 1$, $l_{23} = 2$, $l_{30} = 2$, $l_{31} = 1$, and $l_{32} = 2$ and the objective function is

$$\text{maximize} \quad 4b_1 + 4b_2 + 4b_3 - m_{12} - m_{13} - m_{21} - m_{23} - m_{31} - m_{32}.$$

Since all of the routes in the initial Clarke–Wright solution have only one endpoint, the constraints in (7) and (8) are not needed in this example. An optimal solution is given in Figure 3b and it has a total distance of 10 (there are multiple optimal solutions for this problem).

We point out that, for some problems, not all feasible solutions can be reached by solving the endpoint mixed integer program. We illustrate this limitation of the EMIP with a non-Euclidean problem in Figure 4. In Figure 4a, an initial solution uses eight vehicles and has a total distance of 32. In Figure 4b, an improved solution uses seven vehicles, splits two demands, and has a total distance of $20 + 26\varepsilon$ which is less than 32 when $\varepsilon < 6/13$. This improved solution cannot be generated by the EMIP because the customers with a demand of three units are not endpoints and, therefore, cannot be split.

4. NEW HEURISTIC AND COMPUTATIONAL RESULTS

4.1. Combining EMIP and a Record-to-Record Travel Algorithm

Our new heuristic uses a Clarke–Wright (CW) algorithm to generate a starting solution. Using the CW solution, an EMIP is formulated and solved. In this first EMIP, the size of the neighbor list depends on the number of endpoints in the starting solution (e.g., when there are 24–120 endpoints, the size of the neighbor list is 10). Furthermore, the endpoint mixed integer programs can be considerable in size and difficult to solve. Based on our computational work, a 200-node problem, with a vehicle capacity of 200 units and a demand between 140 and 180 units for each customer, has an average of 200 endpoints, 2,200 integer variables, 2,000 continuous variables, and 2,800 constraints in the EMIP. We set a limit of T seconds for solving the first EMIP and save

the best feasible solution found during a run. We denote this solution by E1 and point out that it may not be the optimal solution.

Using E1 as the initial solution, a second EMIP is formulated and solved (we denote the solution by E2) with a larger size for the neighbor list and a smaller limit for the running time than those for the first EMIP. The final solution is obtained by post-processing E2 with a variable length record-to-record travel algorithm (VRTR) developed by Li et al. [12]. For each customer i, all edges (for customers in the neighbor list) with length greater than $0.8 \times L$ are removed, where L is the maximum length among all edges in i's neighbor list. VRTR considers one-point, two-point, and two-opt moves within and between routes. The details of our new heuristic (denoted by EMIP + VRTR) are given in Table 1.

4.2. Computational Results

4.2.1. Six Benchmark Problems We select six problems (1, 2, 4, 5, 11, 12) from [3] and [4] with 50, 75, 100, 120, 150, and 199 customers. For each problem, we generate a customer's demand according to the six scenarios ([0.01–0.1], [0.1–0.3], [0.1–0.5], [0.1–0.9], [0.3–0.7], [0.7–0.9]) given by [7]. The demand for customer i in scenario $[\alpha–\beta]$ with a vehicle capacity of k units is randomly selected from a uniform distribution on the interval $[\alpha k, \beta k]$. For each size problem, we solve 30 instances for each of the six scenarios with EMIP + VRTR. We use ILOG CPLEX 9.0 with Visual C++ (version 6.0) to solve the EMIPs. Our experiments were carried out on a PC with a 1.7-GHz Pentium 4 processor and 512 MB of RAM.

The results of our experiments are given in Table 2. For each scenario, the EMIP + VRTR result is the median value of the solutions from 30 instances. Recall that, in the paper by Archetti et al. [1], SPLITABU-DT was run five times on one instance of each scenario and it clearly outperformed Dror and Trudeau's procedure (DT). Archetti (private communication, 2005) provided us with the results for SPLITABU-DT given in Table 2 (the actual problem instances were not available). These results were generated on a PC with a 2.4-GHz Pentium 4 processor and 256 MB of RAM.

It is not a straightforward task to compare the results given in Table 2. For each problem size and each scenario,

TABLE 1. SDVRP heuristic: Endpoint mixed integer program followed by variable-length neighbor list record-to-record travel algorithm.

Step 0	Initialization Parameters are I,K, NL (size of the neighbor list), and T (time limit in s). Set $I = 30$ and $K = 5$.
Step 1	Starting solution. Generate a feasible solution using the Clarke–Wright algorithm. N is the number of endpoints in the solution. If $N < 24$, $NL = N - 1$. If $24 \le N < 120$, $NL = 10$. If $120 \le N < 300$, $NL = 8$. If $N < 24$, $T = \infty$. If $24 \le N < 120$, $T = 400$. If $120 \le N < 300$, $T = 5{,}000$. Using the Clarke–Wright solution, formulate and solve an EMIP using NL. If no optimal solution is found after running the EMIP for T s, then terminate the EMIP. Record the best feasible solution found and denote it by E1. Using E1, formulate and solve an EMIP using $1.5 \times NL$. If no optimal solution is found after running the EMIP for $0.6 \times T$ s, then terminate the EMIP. Denote the current solution by E2. Set Record = objective function value of the current solution. Set Deviation = $0.01\times$ Record.
Step 2	Improve the current solution. For $i = 1$ to I(I loop) Do one-point move with record-to-record travel, two-point move with record-to-record travel between routes, and two-opt move with record-to-record travel. Feasibility must be maintained. If no feasible record-to-record travel move is made, go to Step 3. If a new record is produced, update Record and Deviation. End I loop
Step 3	For the current solution, apply one-point move (within and between routes), two-point move (between routes), two-opt move (between routes), and two-opt move (within and between routes). Only downhill moves are allowed. If a new record is produced, update Record and Deviation.
Step 4	Repeat Steps 2 and 3 for K consecutive iterations. If no new record is produced, go to Step 5. Otherwise, go to Step 2.
Step 5	Perturb the solution. Compare the solution generated after perturbation to the best solution generated so far and keep the better solution. Stop.

TABLE 2. Computational results for six benchmark problems.

Scenario	EMIP + VRTR	SPLITABU-DT 1	2	3	4	5
50 customers with vehicle capacity 160						
[0.01–0.1]	457.21	464.64	464.64	466.19	460.79	462.54
[0.1–0.3]	723.57	751.60	767.46	752.84	760.57	774.56
[0.1–0.5]	943.86	1,013.00	1,015.15	997.22	1,007.13	1,010.86
[0.1–0.9]	1,408.34	1,461.01	1,473.29	1,470.11	1,443.84	1,501.39
[0.3–0.7]	1,408.68	1,507.60	1,491.92	1,490.73	1,487.02	1,507.25
[0.7–0.9]	2,056.01	2,166.34	2,174.81	2,166.11	2,170.43	2,148.38
75 customers with vehicle capacity 140						
[0.01–0.1]	598.25	606.52	601.62	607.07	608.61	602.39
[0.1–0.3]	1,081.10	1,092.69	1,087.94	1,094.34	1,104.34	1,097.33
[0.1–0.5]	1,393.53	1,449.11	1,449.54	1,432.17	1,439.00	1,448.34
[0.1–0.9]	2,056.54	2,132.02	2,136.56	2,136.66	2,109.70	2,107.22
[0.3–0.7]	2,112.61	2,149.74	2,156.29	2,167.22	2,159.12	2,170.18
[0.7–0.9]	3,067.19	3,181.50	3,138.18	3,183.61	3,216.11	3,183.83
100 customers with vehicle capacity 200						
[0.01–0.1]	651.44	635.89	665.87	643.41	641.42	657.11
[0.1–0.3]	1,414.33	1,495.76	1,437.96	1,438.32	1,482.18	1,455.84
[0.1–0.5]	1,973.34	2,043.70	1,981.55	2,013.69	2,077.36	2,033.69
[0.1–0.9]	3,162.22	3,172.64	3,061.38	3,010.51	3,069.86	3,193.27
[0.3–0.7]	3,134.56	2,936.92	3,090.85	3,154.24	3,126.00	2,882.13
[0.7–0.9]	4,779.13	4,883.37	4,851.42	4,909.15	4,773.60	4,921.42
120 customers with vehicle capacity 200						
[0.01–0.1]	985.17	1,084.08	1,085.96	1,076.09	1,084.34	1,093.03
[0.1–0.3]	2,568.90	2,914.58	2,916.92	2,927.75	2,915.13	2,919.19
[0.1–0.5]	3,687.06	4,176.24	4,242.15	4,131.13	4,160.77	4,320.33
[0.1–0.9]	6,079.14	6,636.46	6,684.42	6,565.91	6,259.68	6,773.43
[0.3–0.7]	6,123.96	6,433.02	6,746.46	6,585.96	6,598.34	6,834.01
[0.7–0.9]	8,941.79	10,086.02	10,494.49	10,399.11	10,072.61	10,468.19
150 customers with vehicle capacity 200						
[0.01–0.1]	875.16	899.10	890.67	887.55	895.46	882.00
[0.1–0.3]	1,844.96	1,922.49	1,915.15	1,926.86	1,918.83	1,907.93
[0.1–0.5]	2,532.93	2,639.28	2,644.44	2,632.87	2,608.92	2,638.08
[0.1–0.9]	3,945.38	3,907.38	3,851.04	3,849.74	4,056.01	3,884.49
[0.3–0.7]	4,011.74	4,001.80	4,098.43	4,071.44	4,059.75	3,967.11
[0.7–0.9]	5,950.35	6,099.87	6,239.28	6,215.92	6,215.49	6,211.25
199 customers with vehicle capacity 200						
[0.01–0.1]	1,040.20	1,051.61	1,058.60	1,060.41	1,047.88	1,062.87
[0.1–0.3]	2,258.66	2,383.90	2,378.06	2,386.29	2,389.44	2,383.11
[0.1–0.5]	3,202.57	3,298.49	3,265.60	3,247.32	3,333.66	3,277.32
[0.1–0.9]	5,094.61	4,737.47	4,902.00	4,893.66	4,835.13	4,900.89
[0.3–0.7]	5,088.08	5,184.25	5,103.60	5,001.46	5,066.96	5,157.95
[0.7–0.9]	7,207.04	8,065.69	7,676.12	8,007.30	8,022.49	7,951.60

	SPLITABU-DT 1	2	3	4	5
Number of times SPLITABU-DT solution is better than the median solution of EMIP + VRTR	5	4	5	6	4

For each scenario, the EMIP + VRTR result is the median value of the solutions from 30 instances. SPLITABU-DT solves one instance of each scenario five times.

EMIP + VRTR solves 30 instances while SPLITABU-DT solves one instance five times. To compare the results, we propose the following statistical test. If EMIP + VRTR and SPLITABU-DT are equally good with respect to solution quality, then SPLITABU-DT would beat the median EMIP + VRTR result about half the time. Using a binomial distribution with $n = 36$ (this corresponds to a column of SPLITABU-DT results in Table 2, i.e., one run over 36 cases) and $p = 1/2$, we can test the null hypothesis that the results of the two methods are equally good (H_0: $p = 0.50$) against the alternative hypothesis that SPLITABU-DT performs worse than the median value of EMIP + VRTR (H_a: $p < 0.50$).

TABLE 3. Average running time in seconds for EMIP + VRTR and SPLITABU-DT on six benchmark problems.

Scenario	EMIP + VRTR	SPLITABU-DT
50 customers with vehicle capacity 160		
[0.01–0.1]	1.9	4.8
[0.1–0.3]	3.4	21.8
[0.1–0.5]	14.7	28.2
[0.1–0.9]	55.4	60.8
[0.3–0.7]	47.9	48.6
[0.7–0.9]	135.4	106.0
75 customers with vehicle capacity 140		
[0.01–0.1]	25.8	13.0
[0.1–0.3]	57.0	45.4
[0.1–0.5]	214.0	123.0
[0.1–0.9]	401.1	193.0
[0.3–0.7]	509.6	129.0
[0.7–0.9]	811.0	869.0
100 customers with vehicle capacity 200		
[0.01–0.1]	53.9	57.8
[0.1–0.3]	126.5	146.0
[0.1–0.5]	287.6	292.8
[0.1–0.9]	251.2	259.6
[0.3–0.7]	716.5	777.8
[0.7–0.9]	1,024.3	1,004.4
120 customers with vehicle capacity 200		
[0.01–0.1]	36.4	42.4
[0.1–0.3]	136.4	142.6
[0.1–0.5]	220.7	268.0
[0.1–0.9]	722.8	877.8
[0.3–0.7]	605.4	658.6
[0.7–0.9]	725.4	1,825.6
150 customers with vehicle capacity 200		
[0.01–0.1]	107.8	172.8
[0.1–0.3]	308.0	393.2
[0.1–0.5]	630.5	739.2
[0.1–0.9]	2,220.0	2,278.0
[0.3–0.7]	3,028.3	3,008.0
[0.7–0.9]	10,038.8	10,223.0
199 customers with vehicle capacity 200		
[0.01–0.1]	413.4	525.8
[0.1–0.3]	618.5	754.8
[0.1–0.5]	1,775.7	2,668.0
[0.1–0.9]	3,038.1	3,297.2
[0.3–0.7]	3,035.7	3,565.6
[0.7–0.9]	12,542.3	21,849.0

Using a significance level of 0.01, we would reject the null hypothesis when

$$(\hat{p} - 0.5)/\sqrt{(0.5)(0.5)/36} \leq -2.33$$

or $\hat{p} \leq 0.3058$. In words, if SPLITABU-DT performs better than the median value of EMIP + VRTR in $(0.3058)(36) = 11$ instances or less for a single run over 36 cases, then we would reject the null hypothesis.

In Table 2, we point out that, for scenarios with small customer demands, most of the savings are attributable to VRTR. In the first two or three scenarios, the greater emphasis is on routing the vehicles. For scenarios with large customer demands, most of the savings are attributable to EMIP. The last three scenarios emphasize packing the vehicles. For example (using the median solutions throughout), in scenario 1 (small demand) of the 50-node problems, the EMIP solution (E2) averages a 0.61% savings over the Clarke–Wright solution, while the final solution after VRTR averages an 8.11% savings over E2. In scenario 6 (large demand), the EMIP solution (E2) averages a 13.89% savings over the Clarke–Wright solution, while the final solution after VRTR averages a 0.60% savings over E2.

For each of the five runs of SPLITABU-DT over the 36 cases, we count the number of times the SPLITABU-DT solution is better (i.e., less) than the median solution of EMIP + VRTR and provide these counts in the bottom row of Table 2. For each run, the count for SPLITABU-DT is much less than 11 (the counts are 4, 5, or 6) and, therefore, we reject the null hypothesis and conclude that SPLITABU-DT performs worse than EMIP + VRTR.

In Table 3, we give the average running times for EMIP + VRTR and SPLITABU-DT (Archetti [private communication, 2005] provided the running times for each of the five runs) on the six benchmark problems. Recall that, in EMIP + VRTR, we need to specify the values of two parameters: the size of the neighbor list (*NL*) and the time limit in seconds (T). We selected values for *NL* and T that approximately equalized the running times of EMIP + VRTR and SPLITABU-DT (these values are provided in Table 1). Clearly, for a fixed problem size, the running times of both methods increase considerably as we move from the first scenario with small customer demand ([0.01–0.1]) and almost no splits in the final solution to the sixth scenario with very large customer demand ([0.7–0.9]) and several splits in the final solution.

Finally, at the suggestion of a referee, we applied EMIP + VRTR to the six problems with the original demands of the customers (in other words, we solved the standard, capacitated versions of these problems). In Table 4, we present the solution values and running times generated by EMIP + VRTR and SPLITABU-DT (Archetti [private communication, 2005] provided these results each of which is the average of five runs). EMIP + VRTR was very fast and produced solutions that were better than the average solutions produced by SPLITABU-DT.

4.2.2. Five Benchmark Problems with Lower Bounds Next, we tested our heuristic on problems taken from the set of 14 randomly generated problems given in [2]. We used

TABLE 4. Computational results for six capacitated VRPs.

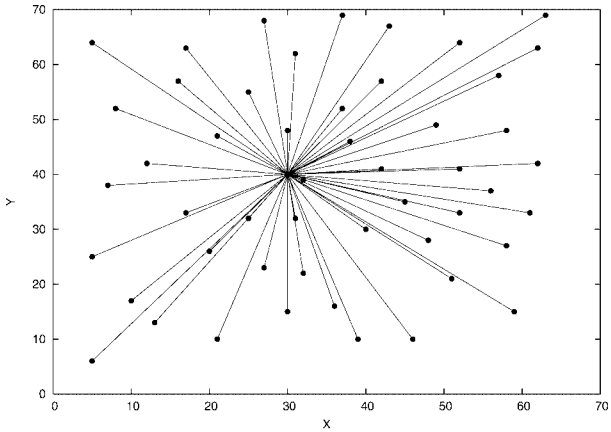
	EMIP + VRTR		SPLITABU-DT	
Customers	Solution	Time (s)	Solution	Time (s)
50	524.61	1.8	533.55	13.2
75	840.18	4.0	849.54	35.8
100	819.56	3.7	835.62	57.6
120	1,043.18	5.6	1,056.01	38.4
150	1,041.99	10.0	1,069.84	389.0
199	1,307.40	18.1	1,342.85	386.4

TABLE 5. Computational results for five benchmark problems with lower bounds.

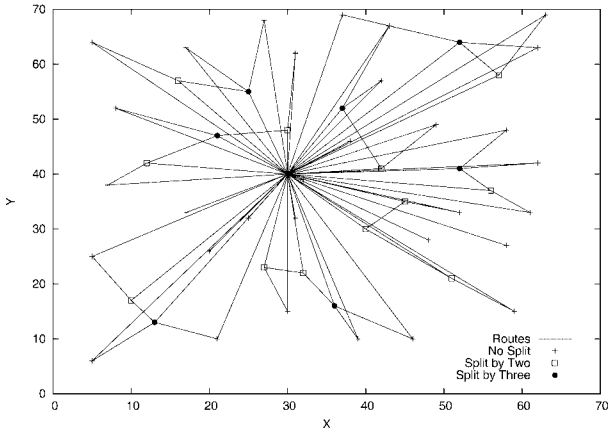
Problem	Belenguer et al. [2] lower bound	EMIP + VRTR	Time (s)	% above lower bound
S51D4	1,520.67	1,586.5	201.74	4.33
S51D5	1,272.86	1,355.5	201.62	6.49
S51D6	2,113.03	2,197.8	301.90	4.01
S76D4	2,011.64	2,136.4	601.92	6.20
S101D5	2,630.43[a]	2,846.2	645.99	8.20

[a]Belenguer et al. [2] report that their cutting-plane algorithm terminated early due to memory overflow. This may not be a tight lower bound.

the same settings of the parameters given in Table 1 and did not fine-tune EMIP + VRTR. We focused on five problems with large customer demands that were generated along the lines of scenarios 4, 5, and 6 ([0.1–0.9], [0.3–0.7], [0.7–0.9]) from [7]. These problems are denoted by S51D4, S51D5, S51D6, S76D4, and S101D5, where S51D4 is a problem with 51 nodes including the depot, and customer demands are randomly generated according to scenario 4. Each problem has a lower bound on the optimal solution. All problems are available online at www.uv.es/belengue/sdvrp.html.



(a) Clarke-Wright solution. Total distance is 2,402.34 with 50 vehicles.



(b) EMIP+VRTR solution. Total distance is 2,197.8 with 42 vehicles.

FIG. 5. Solutions produced by Clarke–Wright and EMIP + VRTR to problem S51D6 from [2].

Our results are presented in Table 5. EMIP + VRTR generates high-quality solutions that are 5.85% above the lower bound on average. EMIP + VRTR takes 390 s on average to solve these five problems. In Figure 5, we show the Clarke–Wright and EMIP + VRTR solutions to S51D6. The EMIP + VRTR solution has 42 routes with seven customers on three routes, 12 customers on two routes, and 31 customers on one route.

5. TWENTY-ONE NEW TEST PROBLEMS AND COMPUTATIONAL RESULTS

We created 21 new test problems that range in size from 8 customers to 288 customers. Vehicle capacity is 100 units. Customer demand is either 60 units or 90 units, so our problems are generated along the lines of scenario 6 with very large customer demand ([0.7–0.9]) from [7]. Each problem has a geometric symmetry (star shape) with customers located in concentric circles around the depot that allows us to visually estimate a near-optimal solution (details on how to generate the estimated solutions are available from the authors). The problem generator and specifications for the 21 problems are given in the Appendix. The 21 problems are available on-line at www.rhsmith.umd.edu/faculty/bgolden/index.html.

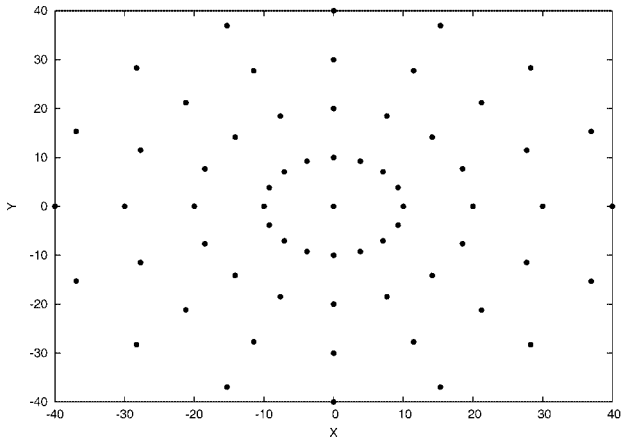
We apply EMIP + VRTR to these new problems with the same settings for the parameters given in Table 1. In Table 6, we present the total distance for each problem given by the visually estimated solution (denoted by ES) and the EMIP + VRTR solution. Computation times for EMIP + VRTR are also presented.

TABLE 6. Computational results for 21 new problems.

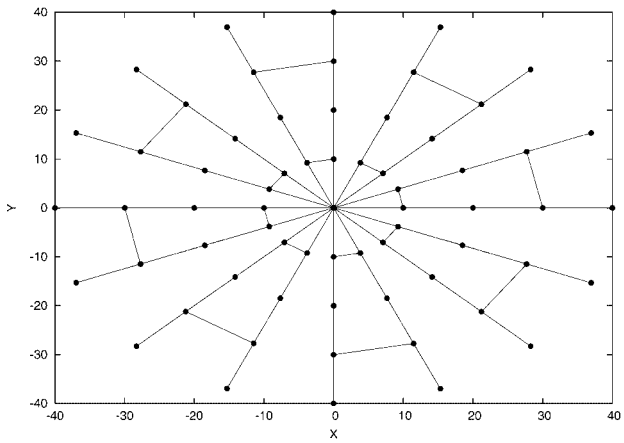
Problem	n	ES	EMIP + VRTR	Time (s)	% above ES
SD1	8	228.28	228.28	0.7	0.00
SD2	16	708.28	714.40	54.4	0.86
SD3	16	430.61	430.61	67.3	0.00
SD4	24	631.06	631.06	400.0	0.00
SD5	32	1,390.61	1,408.12	402.7	1.26
SD6	32	831.21	831.21	408.3	0.00
SD7	40	3,640.00	3,714.40	403.2	2.04
SD8	48	5,068.28	5,200.00	404.1	2.60
SD9	48	2,044.23	2,059.84	404.3	0.76
SD10	64	2,684.85	2,749.11	400.0	2.39
SD11	80	13,280.00	13,612.12	400.1	2.50
SD12	80	7,280.00	7,399.06	408.3	1.64
SD13	96	10,110.60	10,367.06	404.5	2.54
SD14	120	10,920.00	11,023.00	5,021.7	0.94
SD15	144	15,151.10	15,271.77	5,042.3	0.80
SD16	144	3,381.32	3,449.05	5,014.7	2.00
SD17	160	26,560.00	26,665.76	5,023.6	0.40
SD18	160	14,380.30	14,546.58	5,028.6	1.16
SD19	192	20,191.20	20,559.21	5,034.2	1.82
SD20	240	39,840.00	40,408.22	5,053.0	1.43
SD21	288	11,271.10	11,491.67	5,051.0	1.96

ES is the total distance of the visually estimated solution. EMIP + VRTR is run once on each problem.

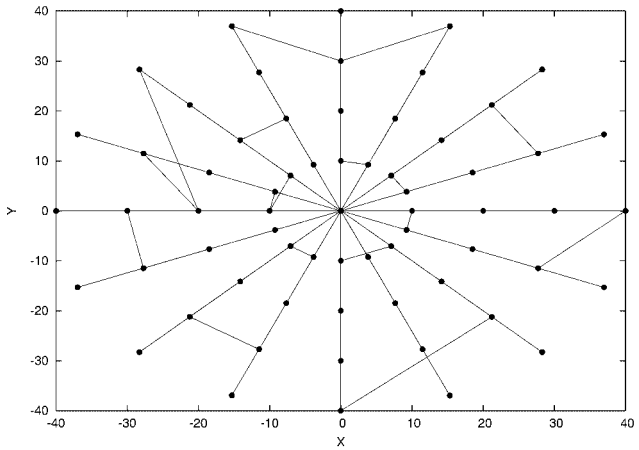
EMIP + VRTR performs very well when compared with ES. On average, EMIP + VRTR generates solutions that are 1.29% above the near-optimal solutions generated by ES. For problems with 24 or more customers, we see that our procedure uses the maximum amount of computing time to solve the endpoint mixed integer program (400 s for SD4 to SD13 and 5,000 s for SD14 to SD21). In Figure 6, we show the visually estimated and EMIP + VRTR solutions to SD10. The ES solution has 48 routes with 32 customers on two routes and 32 customers on one route. The EMIP + VRTR solution has 49 routes with two customers on three routes, 25 customers on two routes, and 37 customers on one route.



(a) Problem SD10 with 64 customers.



(b) Visually estimated solution. Total distance is 2,684.85 with 48 vehicles.



(c) EMIP+VRTR solution. Total distance is 2,749.11 with 49 vehicles.

FIG. 6. Problem SD10 with visually estimated solution and EMIP + VRTR solution.

6. CONCLUSIONS

Over the last decade or so, practical problems that involve routing helicopters in the North Sea, distributing feed to cattle, and delivering newspapers to residential customers have been modeled using split deliveries. A variety of procedures based on mathematical programs and tabu search have been used to solve SDVRPs with some success.

We combined a mixed integer program and a record-to-record travel algorithm to form a new solution procedure (EMIP + VRTR) with only two parameters. We applied EMIP + VRTR to six benchmark problems with 50–199 customers and found that it consistently outperformed tabu search. On five other problems for which lower bounds exist, EMIP + VRTR again performed well. Finally, we developed 21 new problems with 8–288 customers that have near-optimal estimated solutions and found that our new solution procedure generated very high-quality solutions to these new problems.

In the future, we hope to refine the process of setting the values of parameters in EMIP + VRTR and to determine if some of the visually estimated solutions are optimal.

Acknowledgment

We thank an anonymous referee for reading our paper very carefully and offering a number of constructive suggestions which have resulted in an improved paper.

APPENDIX

Generator for New SDVRPs

$(x(i), y(i))$ are the coordinates of customer i, where $i = 0$ is the depot.

$q(i)$ is the demand of customer i.

A and B are parameters that determine the number of customers n, where $n = A \times B$.

Vehicle capacity is 100 units. Customer demand is either 60 or 90 units.

All data are recorded to four decimal places.

begin
  $\omega = 0$
  $x(\omega) = 0, y(\omega) = 0, q(\omega) = 0$
  for $k := 1$ **to** B **do**
    begin
      $\gamma = 10k$
      for $i := 1$ **to** A **do**
        begin
          $\omega = \omega + 1$
          $x(\omega) = \gamma \cos[2(i-1)\pi/A]$

$y(\omega) = \gamma \sin[2(i-1)\pi/A]$
if $\mathrm{mod}(i,2) = 1$
then $q(\omega) = 60$
else $q(\omega) = 90$
end
end
end

Problem	A	B	n
SD1	4	2	8
SD2	4	4	16
SD3	8	2	16
SD4	12	2	24
SD5	8	4	32
SD6	16	2	32
SD7	4	10	40
SD8	4	12	48
SD9	12	4	48
SD10	16	4	64
SD11	4	20	80
SD12	8	10	80
SD13	8	12	96
SD14	12	10	120
SD15	12	12	144
SD16	72	2	144
SD17	8	20	160
SD18	16	10	160
SD19	16	12	192
SD20	12	20	240
SD21	72	4	288

REFERENCES

[1] C. Archetti, A. Hertz, and M. Speranza, A tabu search algorithm for the split delivery vehicle routing problem, Transport Sci 40 (2006), 64–73.

[2] J. Belenguer, M. Martinez, and E. Mota, A lower bound for the split delivery vehicle routing problem, Oper Res 48 (2000), 801–810.

[3] N. Christofides and S. Eilon, An algorithm for the vehicle-dispatching problem, Oper Res Quart 20 (1969), 309–318.

[4] N. Christofides, A. Mingozzi, and P. Toth, "The vehicle routing problem," Combinatorial optimization, N. Christofides, A. Mingozzi, P. Toth, and C. Sandi (Editors), John Wiley, Chichester, UK, 1979, pp. 315–338.

[5] J.-F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.-S. Sormany, "New heuristics for the vehicle routing problem," Logistics systems: Design and optimization, A. Langevin and D. Riopel (Editors), Springer, New York, 2005, pp. 270–297.

[6] M. Dror, G. Laporte, and P. Trudeau, Vehicle routing with split deliveries, Discrete Appl Math 50 (1994), 239–254.

[7] M. Dror and P. Trudeau, Savings by split delivery routing, Transport Sci 23 (1989), 141–145.

[8] M. Dror and P. Trudeau, Split delivery routing, Naval Res Logist 37 (1990), 383–402.

[9] S. Eilon, C. Watson-Gandy, and N. Christofides, Distribution management: Mathematical modeling and practical analysis, Griffin, London, 1971.

[10] P. Frizzell and J. Giffin, The bounded split delivery vehicle routing problem with grid network distances, Asia-Pacific J Oper Res 9 (1992), 101–116.

[11] P. Frizzell and J. Giffin, The split delivery vehicle scheduling problem with time windows and grid network distances, Comput Oper Res 22 (1995), 655–667.

[12] F. Li, B. Golden, and E. Wasil, Very large-scale vehicle routing: New test problems, algorithms, and results, Comput Oper Res 32 (2005), 1197–1212.

[13] K. Liu, A study on the split delivery vehicle routing problem, Ph.D. Dissertation, Department of Industrial Engineering, Mississippi State University, Mississippi State, Mississippi, 2005.

[14] P. Mullaseril, M. Dror, and J. Leung, Split-delivery routing heuristics in livestock feed distribution, J Oper Res Soc 48 (1997), 107–116.

[15] M.A. Nowak, The pickup and delivery problem with split loads, Ph.D. Thesis, Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia, 2005.

[16] G. Sierksma and G. Tijssen, Routing helicopters for crew exchanges on off-shore locations, Ann Oper Res 76 (1998), 261–286.

[17] S. Song, K. Lee, and G. Kim, A practical approach to solving a newspaper logistics problem using a digital map, Comput Ind Eng 43 (2002), 315–330.