Heuristic Scheme for Heterogeneous Vehicle Routing Problem on Trees Based on Generalized Assignment and Bin-Packing Upper Bounds

Roshan Kumar, Avinash Unnikrishnan, and S. Travis Waller

The vehicle routing problem (VRP) is a classical problem in logistics that aims to design minimum-cost delivery routes from a centralized depot. A special case of the VRP arises in situations in which the network has a tree structure (TVRP). Such tree networks arise when the cost of road construction and maintenance is much more than the routing cost or when the transportation network consists of a main highway (e.g., Interstate system) and the customer locations are located off the highway. A heuristic for a constrained case of TVRP in which the vehicle fleet is capacitated and heterogeneous is proposed. The heuristic first determines the customers that will be served by each vehicle by use of bin-packing and Lagrangian-based generalized assignment algorithms. The individual vehicle routes are then determined by use of a depth-first search method. A procedure for further refinement of the heuristic solution quality is also described. The heuristic algorithm was implemented on two real-world networks and on randomly generated networks that varied in size from 20 to 120 nodes. The heuristic solution was found to be between 2% and 10% for almost all of the 200 instances tested and took a fraction of the time taken to find the optimal solution.

The vehicle routing problem (VRP) is a classical problem in logistics that aims to design minimum-cost delivery routes from a centralized depot or depots to customers at different locations, subject to some side constraints. The VRP is known to be NP-hard and is especially difficult to solve because of the presence of subtour elimination constraints, which are exponential in number.

A special case of the VRP arises in situations in which the network has a tree structure (TVRP). The depot is located at the root of the tree, and the geographically dispersed customers form the nodes of the tree. A constrained case of the TVRP in which the vehicle fleet is capacitated and heterogeneous (Cap-HTVRP) is studied in this paper. A heuristic algorithm that explicitly considers the tree structure is presented. To the best of the authors' knowledge, no heuristic techniques for solution of heterogeneous fixed-fleet TVRPs exist.

R. Kumar, Center for Transportation Research, University of Texas at Austin, Austin, TX 78701. A. Unnikrishnan, West Virginia University, Room 627 ESB, P.O. Box 6103, Morgantown, WV 26506-6103. S. T. Waller, School of Civil and Environmental Engineering, University of New South Wales, CE110, Sydney, New South Wales 2052, Australia. Corresponding author: R. Kumar, kumarr1@pbworld.com.

Transportation Research Record: Journal of the Transportation Research Board, No. 2283, Transportation Research Board of the National Academies, Washington, D.C., 2012, pp. 1–11.
DOI: 10.3141/2283-01

Tree networks are encountered in river networks, railway networks, rural areas, areas where the transportation network primarily consists of a main highway, and multiproduct assembly line balancing problems. In some cases, groups of nodes can be merged to form a resulting tree network. A complete description of the occurrence of such tree networks, along with some real-world examples, is provided elsewhere (*1–5*).

The capacitated TVRP and all of its variants were shown to be NP-hard by Labbé et al. (*1*) and Hamaguchi and Katoh (*6*) by transformation of the bin-packing problem [shown to be NP-complete by Garey and Johnson (*7*)] to a special case of the TVRP. The main complicating factor in TVRPs of all variations is that a vehicle can visit a node without actually serving it. For every node in a tree, only one path from the root node to that node exists; therefore, a vehicle must visit some nodes, even though it does not serve them.

This is not the case with VRPs, in which case, whenever a node is visited by a vehicle, it is also served. All TVRPs can be solved as VRPs by detection of the all-to-all shortest paths between the nodes. Because of their special network structure, TVRPs have been of interest to researchers because it is possible to develop customized algorithms and faster solution techniques. For example, the integer programming (IP) formulations for TVRPs do not contain any subtour elimination constraints, which thereby makes the number of constraints in the number of nodes polynomial.

The problem considered in this paper can be briefly defined as follows. Given a tree network, nonnegative arc costs, demand at each node, and a heterogeneous fixed fleet of vehicles located at a depot, find a collection vehicle routes, such that

1. The total distance traveled by (total operating cost of) all vehicles used is minimized,
2. The demand at each node is satisfied by exactly one vehicle,
3. The total demand serviced by a vehicle does not exceed its capacity, and
4. All vehicle routes begin and end at the depot.

This problem is Cap-HTVRP. It is assumed that the arc costs remain constant over all vehicle types and that the total fleet mix is given and finite. Heuristics for both cases with and cases without fixed vehicle costs are presented.

Cap-HTVRP has embedded within it the generalized assignment problem (GAP), the bin-packing problem, and the tree traveling salesman problem (TTSP). Although the bin-packing problem is NP-hard, several efficient heuristics that produce near-optimal solutions have been developed (*8–11*). The four most popular

approximation algorithms are first fit, best fit, first fit decreasing, and best fit decreasing (*12*). The heuristic for the Cap-HTVRP proposed here iteratively finds seed nodes by the use of upper-bound heuristics for the bin-packing problem, assigns a vehicle to each seed node, and then uses a Lagrangian-based GAP algorithm to assign nodes to vehicles located at the seed node. The TTSP is solved to find the optimal route for every vehicle used. The TTSP is trivially solvable in polynomial time (*13*). A reforming operation based on the savings heuristic (*14*) is finally employed to further reduce the solution cost.

The rest of this paper is organized as follows. The next section describes the relevant past work carried out in the area. Some exact solution methods and their relation to GAP and the bin-packing problem are then discussed and the heuristic algorithm is presented. Finally, the computational results obtained by implementation of the heuristic and exact methods are presented.

LITERATURE REVIEW

The TVRP variants that have been studied previously include (*a*) capacitated TVRPs (TCVRPs), (*b*) distance-constrained TVRPs, (*c*) TVRPs that combine TCVRPs and distance-constrained TVRPs, (*d*) TVRPs with split delivery, (*e*) TVRPs with pickup and delivery, and (*f*) TCVRPs with backhauls.

TCVRPs were first introduced by Labbé et al. (*1*). They provided a proof for NP-hardness; conditions for lower bounds; a bin-packing-based two-approximation algorithm, which was further modified by Rennie (*15*); and an enumerative branch-and-bound scheme. A column-generation approach to solve TCVRPs on the basis of the depth first search (DFS) properties of TTSPs (*12*) was developed by Mbaraga et al. (*16*). Chandran and Raghavan (*4*) and Busch (*17*) formulated TCVRP as an integer program that uses the DFS properties of TTSPs. Chandran and Raghavan further proposed various valid inequalities to decrease the search space (*4*), and they also refined the approximation algorithm proposed by Labbé et al. (*1*) and Rennie (*15*). Basnet et al. developed two heuristics for the TCVRP (*3*). The first heuristic is based on the savings heuristic developed by Clarke and Wright (*14*). The second heuristic first assigns all nodes to a single vehicle and then subdivides these nodes into different vehicles such that capacity constraints are satisfied.

Apart from the heuristic proposed by Basnet et al. (*3*), none of the heuristics or approximation algorithms explicitly minimizes the total distance traveled by the total operating cost of the vehicles; rather, they seek to find a feasible solution by packing of the node demands into vehicles. Furthermore, none of these heuristics is capable of dealing with heterogeneous fleets. The heuristic presented in this paper addresses these two issues. Mbaraga et al. developed a heuristic and two exact algorithms for the capacitated and uncapacitated versions of the time-constrained VRP on trees (*16*). Their heuristic is an extension of the work of Labbé et al. (*1*) and converges in linear time. One of their exact algorithms is a branch-and-bound technique, whereas the other is a column-generation technique. Hamaguchi and Katoh developed a 1.5-approximation algorithm for the TVRPs with split delivery (*6*), whereas Asano et al. further refined it to a 1.35078-approximation algorithm (*18*). The first step of their algorithms involves performance of a set of seven reforming operations to reshape the tree. These reformations are done such that the lower bound of the problem remains the same. Some reforming operations include contraction of subtrees into a single node, removal of demand from all internal nodes, and assurance that all non-grandparent nodes have only one child (leaf) node.

Katoh and Yano presented a 2-approximation algorithm for the TCVRP with pickup and delivery (*19*). They assumed splittable demand and developed an algorithm that finds a set of feasible vehicle tours such that at any time during the tour, the sum of the goods to be delivered and picked up does not exceed the vehicle capacity. That is, they do not consider all delivery to take place before pickup. Their algorithm consists of two main steps. In the first step, they perform a set of seven reforming operations similar to those performed by Asano et al. (*18*). In the second step, an appropriate subgraph and a serving strategy are chosen. Four subgraph types are defined, and for each case a different serving strategy is developed. An exact and heuristic solution method for the TVRPs with backhauls was proposed by Kumar et al. (*5*). They proposed a new IP formulation that uses some properties and observations that are true of TVRPs with backhauls at optimality and a 2-approximation algorithm that uses a heuristic two-dimensional bin-packing procedure.

The heterogeneous VRP can be solved to optimality by trivial modification of the network flow-based integer formulation for the capacitated VRP on general networks first presented by Laporte et al. (*20*) and Fisher and Jaikumar (*21*). Fisher and Jaikumar also provided a cluster-first route-second heuristic based on the GAP and traveling salesman problem to solve capacitated VRPs on general networks (*21*). With some modifications, their algorithm can be applied to the heterogeneous VRP. An exact solution method based on solution of the linear programming relaxation by a column-generation method was proposed by Choi and Tcha (*22*). Heuristics for heterogeneous VRPs based on existing VRP techniques have been studied by Golden et al. (*23*), Salhi and Rand (*24*), Taillard (*25*), Desrochers and Verhoog (*26*), and Renaud and Boctor (*27*).

PRELIMINARIES

Consider a tree network rooted at the depot. N_D is the set of nodes including the depot. Denote the set of nodes excluding the depot node as N, that is, $N = N_D \backslash \{\text{depot}\}$. The demand at a node $i \in N$ is denoted by $d(i)$. The set of edges in the network is represented by E; all edges facilitate movement in both directions. The cost of traversing edge $\{i, j\}$ is given by $c_{\{i,j\}}$. Replace the set of undirected edges by the set of directed arcs A, such that every edge is replaced with two arcs, one in each direction, and $c_{ij} = c_{ji} = c_{\{i,j\}}$. A directed arc between i and j is represented as arc (i, j). Let $|N|$ equal n, and let $|E|$ equal m; m is then equal to $n - 1$ and $|A|$ is equal to $2m$. Let K be the set of vehicles, let $\text{cap}_k : k \in K$ denote the capacity of each vehicle, and let $f_k : k \in K$ denote the fixed operating cost of each vehicle. Assume that the vehicles are indexed in decreasing order of their capacities, that is, $\text{cap}_{k_1} \geq \text{cap}_{k_2} \geq \ldots \geq \text{cap}_{k_{|K|-1}} \geq \text{cap}_{k_{|K|}}$. In this paper, the number of vehicles in the fleet and the fleet mix are given. Although existing VRP techniques can be used to solve this problem (quite inefficiently), no heuristics or approximation algorithms for TVRPs that deal with either heterogeneous fleets or fixed fleets that explicitly take advantage of the tree structure exist in the open literature. The robustness of the heuristic in dealing with nonfixed and homogeneous fleets is demonstrated in the next section.

In compliance with other literature on trees, a leaf node of tree T is defined to be a node with degree 1. For notational convenience and without a loss of generality, assume that the tree grows downward from the root node. That is, all nodes are topologically below the root node. Therefore, the ancestors of a node are above it, and its descendants are below it. The immediate ancestor of a node is called its parent. For a given node i, its parent is denoted by P_i. Every node has a single unique parent node. The immediate descendant of a

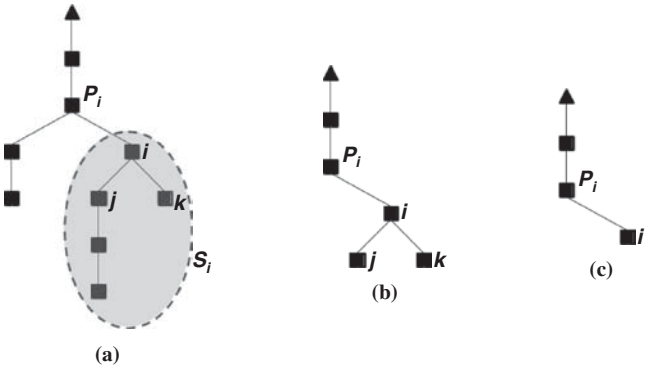

FIGURE 1 Example of notation: (*a*) $c(i) = \{j, k\}$, S_i is subtree rooted at i, and P_i is parent of i; (*b*) minimal-covering subtree with $R = \{j, k\}$; and (*c*) set of nodes in PfD_i with $R = \{i\}$.

node is called its child. The set of children of node i is given by $c(i)$. A node can have many children. For further notational convenience, assume that the nodes of the tree are numbered in the DFS order. This implies that if $\{i, j\} \in E$ and if $i < j$, then i is the parent of node j. Similarly, if $\{i, j\} \in E$ and if $i > j$, then i is the child of node j. A subtree of tree T (S_i) is defined as a tree rooted at a node i, such that the nodes i and all of its descendants are part of the subtree. It is a connected subgraph of the tree, and all of its nodes and edges are also part of tree T. The example shown in Figure 1 shows all the notation explained above. The nodes j and k are the children of node i, and the subgraph containing the nodes within the shaded region is subtree S_i.

The minimal-covering subtree of a tree is defined as follows. Given a subset of nodes $R \subset N$, the minimal-covering subtree CS_R is the set of all of the nodes in the unique paths from each node in R to the depot. The minimal-covering subtree will always include the depot and will be rooted at the depot. For example, the minimal-covering subtree CS_R for the tree in Figure 1*a* with R equal to $\{j, k\}$ is shown in Figure 1*b*.

Furthermore, the set of nodes in the unique path from the depot to a node i is denoted by PfD_i, that is, $PfD_i = CS_R : R = \{i\}$. The cumulative demand of the nodes in PfD_i (dem_i) is given by $\sum_{j \in PfD_i} d_j$, and the cost of the path from the depot to i (L_i) is given by $L_i = \sum_{j \in PfD_i} \sum_{q \in PfD_i} c_{j,q} : (j, q) \in A$.

EXACT Cap-HTVRP SOLUTION METHODS

Exact Cap-HTVRP solution methods were developed by Chandran and Raghavan (*4*) and Mbaraga et al. (*16*). Chandran and Raghavan developed two integer programs for solving TCVRPs (*4*). The first one builds off the fact that, once the nodes that are part of a vehicle route are determined, nodes will be served in the order of their DFS index. A similar IP was also developed by Busch (*17*). The second formulation uses the fact that only one path exists between a node and any other node in the depot and that every node has a unique parent node. The second formulation, Cap-HTVRP(1), which will be further used in the heuristic proposed here, is as follows:

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \text{ traverses arc } (i, j) \\ 0 & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1 & \text{if vehicle } k \text{ serves node } i \\ 0 & \text{otherwise} \end{cases}$$

minimize

$$2 \times \sum_i \sum_j \sum_k c_{ij} x_{ijk} \tag{1}$$

subject to

$$x_{P_i ik} \geq x_{ijk} \qquad \forall k \in K, i \in N, j \in c(i) \tag{2}$$

$$x_{P_i ik} \geq y_{ik} \qquad \forall k \in K, i \in N \tag{3}$$

$$\sum_{i \in N} y_{ik} d_i \leq \text{cap}_k \qquad \forall k \in K \tag{4}$$

$$\sum_{k \in K} y_{ik} = 1 \qquad \forall i \in N \tag{5}$$

$$y_{ik} \in \{0, 1\} \qquad \forall i \in N, \forall k \in K \tag{6}$$

$$x_{ijk} \in \{0, 1\} \qquad \forall (i, j) \in A, \forall k \in K \tag{7}$$

The objective function minimizes the total distance traveled. It is multiplied by 2 because each vehicle must return to the depot after service. Constraint 2 states that if a vehicle k traverses arc (i, j), then it should also have traversed the arc (P_i, i). This ensures that if a vehicle traverses an arc, then it must first traverse the parent arc to reach that arc. Constraint 3 ensures that if a vehicle decides to serve node i, then it should travel along the arc leading to that node i, that is, arc (P_i, i). Constraint 4 ensures that every node is serviced by only one vehicle, whereas Constraint 5 ensures that the vehicle capacity is not exceeded. To further expedite the convergence of the IP, some valid inequalities were also proposed by Chandran and Raghavan (*4*). The formulation can easily accommodate fixed costs by modification of the objective function as follows:

$$2 \times \sum_i \sum_j \sum_k c_{ij} x_{ijk} + \sum_k f_k x_{12k} \tag{8}$$

Let the depot be denoted Node 1. Because the depot has a degree of 1, all vehicles used will always traverse arc (1, 2). Thus, fixed costs can be incorporated into the formulation without the need for definition of new variables or constraints. When the arcs have asymmetric costs associated with them, a flow conservation constraint for every vehicle and for every node must be added to the formulation.

$$\sum_{j \in \{P_i \cup c(i)\}} x_{ijk} - \sum_{j \in \{P_i \cup c(i)\}} x_{jik} = 0 \qquad \forall i \in \{N_D\}, \forall k \in K \tag{9}$$

Mbaraga et al. define a set covering-based formulation for Cap-HTVRP that is solved by the use of column generation (*16*). The master problem of the column-generation scheme solves Cap-HTVRP for a subset of variables. These variables are then parsed to the subproblem. The subproblem is a capacitated shortest-path problem whose arc costs are defined such that the shortest path will generate a column with the most negative reduced cost, which will in turn be parsed to the master problem. The tree arcs and costs are modified to form an acyclic graph for every vehicle as vehicles serve nodes in DFS order. The constrained shortest path is then solved on these acyclic

graphs by use of the algorithm proposed by Desrosiers et al. (*28*). The shortest-path algorithm has a pseudopolynomial running time.

HEURISTIC FOR Cap-HTVRP

Let $L_{i,j}$ be the cost of the path between nodes i and j. It is also known that, given a vehicle and the nodes that it is going to serve, the least-cost vehicle route will visit the nodes in increasing order of the DFS index (*13*). Given this, the formulation Cap-HTVRP(1) can be rewritten such that the objective function enforces DFS movement, whereas the constraints enforce the capacity and service requirements. When no fixed costs are given, the new formulation, Cap-HTVRP(2), with the depot as Node 1, is given as follows:

minimize

$$\sum_{k\in K}\Delta(k) \tag{10}$$

subject to Constraints 4 to 6, where $\Delta(k)$ for every vehicle $k \in K$ is defined as

$$\Delta(k) = L_{1,p} + \sum_{i:y_{ik}=1} L_{i,q} + L_{r,1} \tag{11}$$

$$p = \min_{j:1<j}\{j \mid y_{jk} = 1\} \tag{12}$$

$$q = \min_{j:i<j}\{j \mid y_{jk} = 1\} \tag{13}$$

$$r = \max_{j:j>1}\{j \mid y_{jk} = 1\} \tag{14}$$

Objective Function 11 is the sum of the costs required to serve the lowest DFS-indexed node from the depot, serve the nondepot nodes in DFS order, and traverse from the highest indexed node back to the depot for each vehicle. The constraints of the formulation Cap-HTVRP(2) are the same as those of a GAP. The objective function, however, is more complex. The objective is dependent not only on the cost accrued by a vehicle to serve a node but also on the order in which the nodes are served. Therefore, it is difficult to assign these costs a priori, because costs for every precedence order of nodes must be defined. Therefore, Cap-HTVRP(2) cannot be solved as a GAP. When fixed costs $f_k\ \forall k \in K$ are given, the formulation can be modified to the formulation Cap-HTVRP(3), as follows:

$$z_k = \begin{cases} 1 & \text{if vehicle } k \text{ traverses arc } (i, j) \\ 0 & \text{otherwise} \end{cases}$$

minimize

$$\sum_{k\in K}\Delta(k) + \sum_{k} f_k z_k \tag{15}$$

subject to

$$\sum_{i\in N} y_{ik} d_i \le \text{cap}_k z_k \qquad \forall k \in K \tag{16}$$

$$\sum_{k\in K} y_{ik} = 1 \qquad \forall i \in N \tag{17}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in N, \forall k \in K \tag{18}$$

$$z_k \in \{0,1\} \qquad \forall k \in K \tag{19}$$

$\Delta(k)$ is as defined in Equations 11 to 14. The Constraint Set 16 to 19 is that of the capacitated facility location problem (CFLP). However, the cost function $\Delta(k)$ in the objective is not a simple linear cost, because of which Cap-HTVRP(3) cannot be solved as a CFLP.

However, current GAP and CFLP methods can be used to find a heuristic solution to Cap-HTVRP if $\Delta(k)$ can be approximated as a linear function of y_{ik}. This calculation is done by a method similar to the one described by Fisher and Jaikumar (*21*). First, the concept of a seed node for every vehicle is defined. A seed node of a vehicle is a node that is assigned to a vehicle a priori. That is, fix nodes $i_{k_1}, i_{k_2}, \ldots, i_{k_{|K|-1}}, i_{k_{|K|}}$ that will be served by vehicle $k_1, k_2, \ldots, k_{|K|-1}, k_{|K|}$, respectively. Next, compute a penalty δ_{ik} associated with insertion of node i into the route of vehicle k. Thus, computation of a heuristic solution to Cap-HTVRP by the GAP or CFLP solution method consists of detection of seed nodes or customers and computation of $\delta_{ik}\ \forall i \in N, k \in K$.

Once this is done, solution of GAP or CFLP will result in a set of nodes that every vehicle k will serve. Detection of the optimal route for these set of nodes is trivial, as it involves detection of the TTSP for every vehicle k. Finally, to further reduce the value of the Cap-HTVRP heuristic solution, a refining operation is also performed, as described below.

Computation of δ_{ik}

δ_{ik} can be interpreted to be the penalty accrued by vehicle k when it serves node i. This vehicle k already serves seed node i_k, so δ_{ik} computes the additional cost required to serve i, given that k already serves i_k. Recall that the set of nodes in the unique path from the depot to the node i_k is denoted by PfD_{i_k}. Now, if $i \in PfD_{i_k}$, then this node i will already be visited by vehicle k on its way to serving the seed node, i_k. Therefore, the penalty associated with serving this node i or inserting this node i into the vehicle route will be 0. For any node $i \notin PfD_{i_k}$, the penalty associated with inserting that node i into the vehicle route k is equal to twice the distance of that node from the last common node in PfD_{i_k} and PfD_i. This is explained in detail as follows. Consider Figure 2. Let node i be the node to be inserted, and let node i_k be the seed node. Now, before node i is added, the cost of the vehicle route k is twice the cost of traversing from the depot to i_k, $2 \times L_{i_k}$. However, if node i is added to the route, the cost of the route of vehicle k will be the sum of the costs of traversal from the depot to i_k (L_{i_k}) from i_k to i ($L_{i_k,i}$) and from i back to the depot (L_i). The penalty cost δ_{ik} can now be computed. In more formal terms,

$$\delta_{ik} = \begin{cases} 0 & \text{if } i \in PfD_{i_k} \\ L_{i_k} + L_{i_k,i} + L_i - 2 \times L_{i_k} & \text{if } i \in N \setminus PfD_{i_k} \end{cases} \tag{20}$$

When no fixed costs are given, Objective Function 10 can be replaced with

$$\sum_{i\in N}\sum_{k\in K} \delta_{ik} y_{ik} \tag{21}$$

and when fixed costs are given, Objective Function 15 can be replaced with

$$\sum_{i\in N}\sum_{k\in K} \delta_{ik} y_{ik} + \sum_{k} f_k z_k \tag{22}$$

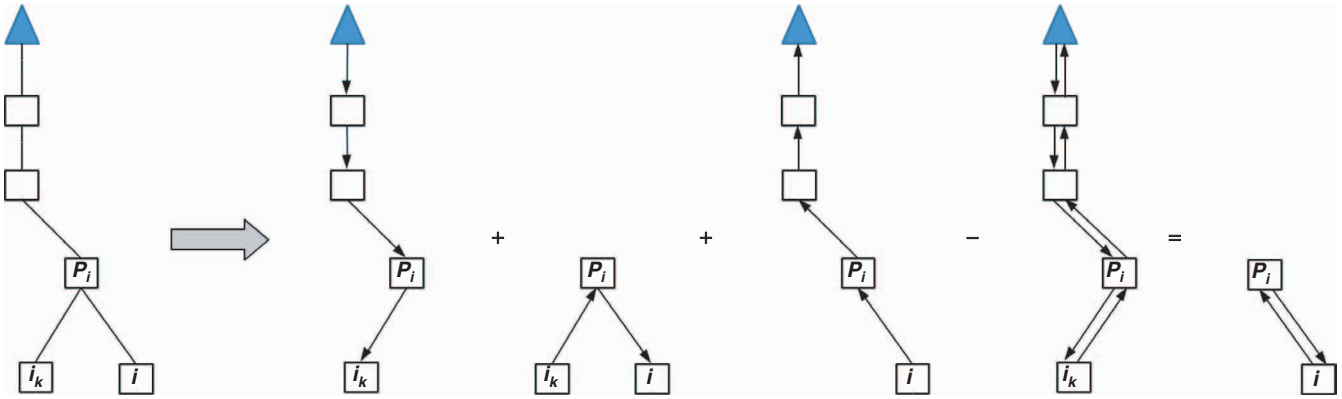

FIGURE 2 Computation of δ_{ik}, $\forall i \in N \backslash PfD_{ik}$, $\forall k \in K$.

Detection of Seed Customers

Seed nodes must be chosen such that the resulting GAP or CFLP solution is as close to the optimal solution as possible. First, seed nodes should be selected such that in the resulting GAP solution the same vehicle does not serve two seed nodes. Second, seed nodes should be located such that the total insertion cost, δ_{ik}, is minimized. Because the vehicle fleet is heterogeneous, the procedure assigns vehicles to nodes in decreasing order of capacity. The seed node selection procedure defined here has three steps:

Step 1. Assign vehicle k to seed node i_k.

Step 2. Determine potential nodes that vehicle k will serve; delete these nodes.

Step 3. Select the next seed node i_{k+1} from the remaining set of nodes.

It can be seen that the number of iterations is equal to the number of vehicles. First, start with the entire tree, and then at every iteration of the procedure delete a subtree, so that the pool of seed node candidates keeps decreasing. The node deletion at every stage ensures that no two seed nodes are served by the same vehicle. Second, delete nodes in such a manner that the resulting GAP solution is minimized. This is done with an attempt to guarantee that a vehicle k will choose to serve a node i such that δ_{ik} is equal to 0.

Assume that the node in the tree at iteration k is N_k and that $|N_k|$ is equal to n_k. This assumption implies that an $n - n_k$ number of nodes has been deleted by iteration $k-1$. Next, it is easy to see that the tree at iteration k actually covers a subtree with $|R|$ equal to n_k; therefore, the tree at an iteration k is denoted by CS_{R_k}. The tree at the first iteration is the original tree network, T, with n_1 being equal to $n+1$ (the additional node is the depot). Lastly, at iteration k, $|K| - (k+1)$ vehicles are yet to be assigned.

The function used to assign a vehicle to a seed node at iteration k is denoted by the parameter VehicleAssign (k). Contract$_k$ is the function that deletes nodes from the tree CS_{R_k} such that the next seed node, i_{k+1}, that is chosen is not assigned to vehicle k by the GAP. The first method, VehicleAssign$_{H1}(k)$, is as follows. At every iteration k, the n_k nodes are sorted in decreasing order of their cost, L_i. The function quicksort $(L_i : i \in N_k)$ performs this function and stores the nodes in the array SORT$_k$. The pth element of SORT$_k$ is denoted SORT$_k(p)$. Now, from the first $|K| - (k+1)$ elements of SORT$_k$, the node with the highest cumulative demand, Dem$_i$, is selected to be seed node i_k and is assigned to vehicle k. The details are presented in Algorithm 1 in Equation Box 1.

The second method of assignment of seed nodes, VehicleAssign$_{H2}(k)$, is a slight modification of the first one. From the first $|K| - (k+1)$ elements of SORT$_k$, instead of selection of the node with the highest cumulative demand Dem$_i$ to be the seed node i_k, the node that is the farthest away from the seed node i_{k-1} is selected.

Proposition 1: At every iteration k, seed node i_k is a leaf node of the covering subtree CS_{R_k}.

Proof: Assume that the chosen seed node i_k is not a leaf node of the tree CS_{R_k}. However, any node that is a direct descendant of i_k will also be in the first $|K| - (k+1)$ elements of SORT$_k$. Therefore, Step 7 of Algorithm 1 will always select a leaf node. ■

The next step in iteration k of the seed node selection procedure deletes nodes from the tree CS_{R_k} that will potentially be served by the current vehicle k, which has been assigned seed node i_k. Contract$_k$ contains the set of nodes whose deletion from the tree CS_{R_k} results in the tree for the next iteration $k+1$, $CS_{R_{k+1}}$. Let bin$_{UB}(V_i)$ be the bin-packing upper bound with bin capacity Cap$_k$ on the number of vehicles required to serve subtree S_i. Start at the leaf node i_k and

EQUATION BOX 1 Algorithm 1 Computation of VehicleAssign$_{H1}(k)$ at Iteration k

Input: N_k, $L_i \forall i \in N_k$, $\text{Dem}_i \forall i \in N_k$, f_k

Output: i_k

1: if $f_k \neq 0$ then

2: for $i_k = 1 \rightarrow n_k$ do

3: $L_i \leftarrow L_i + f_k$

4: end for

5: end if

6: $\text{SORT}_k \leftarrow \text{quicksort}\left(L_i : i \in N_k\right)$ 3: $L_i \leftarrow L_i + f_k$

7: $i_k = \arg\max_i \left\{\text{Dem}(i) : i = \text{SORT}_k(p) \middle| 1 \le p \le |K| - k + 1\right\}$

EQUATION BOX 2 Algorithm 2 Computation of Set Contract$_k$ at Iteration k

Input: CS_{R_k} $d_i : i \in N_k$ Cap_k

Output: $Contract_k$ $CS_{R_{k+1}}$

1: $i \leftarrow i_{k+1}$

2: CurrentNode = i_k

3: while $bin_{UB}(V_i) < 2$ do

4: CurrentNode = i

5: $i \leftarrow p_i$

6: end while

7: $Contract_k = \{i : i \in s_{CurrentNode} \cap N_k\}$

8: $N_{k+1} = N_k \backslash Contract_k$

9: $A_{k+1} = A_k \backslash \{(i, j) \in A_k : i \in Contract_k\}$
$\backslash \{(i, j) \in A_k : j \in Contract_k\}$
$\backslash \{(i, j) \in A_k : i \in Contract_k, j \in Contract_k\}$

10: $CS_{R_{k+1}} = (N_{k+1}, A_{k+1})$

move upwards in the tree. A subtree S_i is then deleted from the tree CS_{R_k} if $bin_{UB}(V_i)$ is less than 2. The reason for this is straightforward, as this procedure deletes nodes closest to the seed node that can be served by vehicle k. Here, the first fit-decreasing procedure is used to calculate $bin_{UB}(V_i)$. The steps of this procedure are described in Algorithm 2 in Equation Box 2.

Solution of GAP

As stated above, after the seed nodes have been identified, the next step involves solution of GAP (CFLP, if fixed costs are given). δ_{ik} is calculated for every seed node and every vehicle. The formulation Cap-HTVRP(2) is then used to solve GAP. Because GAP is a well-known problem in logistics and operations research, many polynomial techniques that provide close to optimal solutions exist in the literature. In this paper, the Lagrangian-based lower-bound regret and greedy regret developed by Jeet and Kutanoglu (*29*) are used. It is easy to see that after the GAP is solved, a feasible solution to the Cap-HTVRP is always obtained.

Refining Operation

A feasible solution to Cap-HTVRP is obtained by solution of GAP (or CFLP). This solution can be refined further by performance of a myopic savings operation. Let the current best solution value obtained by solution of GAP be $\overline{z}_w$, and let the current best solution be $\overline{X}_w$, $\overline{Y}_w$, where w is the number of vehicles used by the solution and $1 \le w \le |K|$. The refining operation examines one node at a time. Thus, the refining operation has n iterations. First, for every node the total savings in cost obtained by removal of that node from the current route and placement of it in one of the available $(w - 1)$ routes or a new $(w + 1)$ route is calculated, provided that the insertion is possible. Next, let the solution cost of removal of node i from the current route and placement of it in route k be given by $z_{w_{k_i}}$ for each node i. That is, for every node i, the savings (Sav_i) is

$$Sav_i = \max_k \left\{ \overline{z}_w - \overline{z}_{w_{k_i}} \right\} \tag{23}$$

where $k = \{1, 2, \ldots, w, w + 1\}$.

LIST is the set of nodes in nonincreasing order of the Sav_i value. LIST(r) denotes the rth element of LIST. Now, if the maximum savings, LIST(1), is positive, then the node i that yields this savings is inserted into its new location and $\overline{z}_{w_{k_i}}$ is the new incumbent best solution. If the maximum savings is 0, then the node i that yields this zero savings is discarded and the next iteration is initiated. Let $\overline{z}$ be the best heuristic solution value, and let $\overline{X}$, $\overline{Y}$ be the best heuristic solution obtained at the end of the refining operation. The details of the algorithm are presented in Algorithm 3 in Equation Box 3.

Heuristic Cap-HTVRP

The complete heuristic proceeds in the following manner:

Step 1. Find seed customers by use of the functions Vehicle Assign(k) and $Contract_k$ until either $|K|$ nodes have been assigned or all nodes N have been covered.

Step 2. Calculate δ_{ik} for the seed customers i_k.

EQUATION BOX 3 Algorithm 3 Refining Operation to Improve Heuristic Solution

Input: $\overline{X}_w$, $\overline{Y}_w$, $\overline{z}_w$

Output: $\overline{X}$, $\overline{Y}$, $\overline{z}$

1: Calculate $Sav_i = \max_k \left\{ \overline{z}_w - \overline{z}_{w_{ki}} \right\}$
where $k = \{1, 2, \ldots, w, w+1\}$ $\forall i \in N$

2: $LIST = \{ p : Sav_{p-1} \ge Sav_p \ge Sav_{p+1} \}$ $|LIST| = n$

3: $\overline{z} = \overline{z}_w$ $\overline{X} = \overline{X}_w$ $\overline{Y} = \overline{Y}_w 4$:

4: while LIST $\neq 0$ do

5: if LIST(1) > 0 then

6: $\overline{z} \leftarrow \overline{z} + Sav_{LIST(1)}$

7: Calculate w

8: $\overline{z}_w \leftarrow \overline{z}$

9: Update $\overline{X}$ and $\overline{Y}$

10: LIST = LIST\LIST(1)

11: else

12: LIST = LIST\LIST(1)

13: end if

14: Calculate Sav_i $\forall i \in$ LIST

15: $LIST = \{ p : Sav_{p-1} \ge Sav_p \ge Sav_{p+1} \}$

16: end while

EQUATION BOX 4 Algorithm 4 Heuristic Algorithm for Cap-HTVRP

Input: $T = (N_D, A)$ $d_i, \forall i \in N$ $c_{ij}, \forall (i,j) \in A$ $\text{Cap}_k, \forall k \in K$

Output: $\bar{z}$ $\bar{X}$ $\bar{Y}$

1: $N_1 = N$

2: for $k = 1 \rightarrow |K|$ do

3: if $N_k \neq \varnothing$ then

4: Compute i_k using VehicleAssign(k)

5: Compute $\delta_{ik}, \forall i \in N$

6: Compute set Contract$_k$ w

7: end if

8: $N_{k+1} = N_k \backslash \text{Contract}_k$

10: end for

11: Solve GAP using Cap-HTVRP(2) with Objective Function 21

12: Calculate $\bar{z}_w$, $\bar{X}_w$, and $\bar{Y}_w$

13: $\bar{z}$, $\bar{X}$, and $\bar{Y}$ using Algorithm 3

Step 3. Solve GAP by Cap-HTVRP(2) with Objective Function 21, if fixed costs are given, and then solve Cap-HTVRP(3) with Objective Function 22.

Step 4. Calculate the current best heuristic solution $\bar{X}_w$ and $\bar{Y}_w$ and the best heuristic solution value $\bar{z}_w$.

Step 5. Perform the refining operation described in Algorithm 4.4 to obtain $\bar{X}$, $\bar{Y}$, and $\bar{z}$. The pseudocode for the heuristic is given in Algorithm 4 in Equation Box 4.

The heuristic is explained by means of the example shown in Figure 3. The original network is shown on the left. The arc costs are shown on the edges, and the demand at every node is shown in a box next to the node. It is assumed that three vehicles with capacities of 900, 600, and 300 are given. The first step, Step I, involves assignment of every vehicle to a seed node in T. Vehicle k_1 is assigned to Node 8, and vehicle k_2 is assigned to Node 4. Note that Vehicle 3 remains unused. After the seed nodes have been assigned, δ_{ik} is calculated for the assigned vehicles (Step II). The GAP is then solved (Step III) by use of δ_{ik}, d_i, and Cap_k as parameters. In the GAP solution, k_1 serves Nodes 5, 6, 7, 8, and 9 and k_2 serves Nodes 1, 2, 3, and 4. The solution cost is 272. The solution cannot be improved any further and so Step IV is empty. The optimal solution to this problem is also 272 and contains the same assignment of nodes.

COMPUTATIONAL PERFORMANCE

The solution quality and computational efficiency of the proposed algorithm were tested on randomly generated test networks along with two real-world networks.

Test Instances

The first real-world network is a tree network in Wyoming (Figure 4). Denver, Colorado, was considered the depot, and on the maps, all of the markers with letters denote the nodes where supply or demand originates. The paths traced in Figure 4 represent the links. A tree network arising in the area of Amarillo, Texas, was considered the second test network. The test networks were generated by a procedure similar to the one described by Labbé et al. (*1*), Mbaraga et al. (*16*), and Chandran and Raghavan (*4*). Every node in the tree had between one and five children. Without a loss of generality, the depot was forced

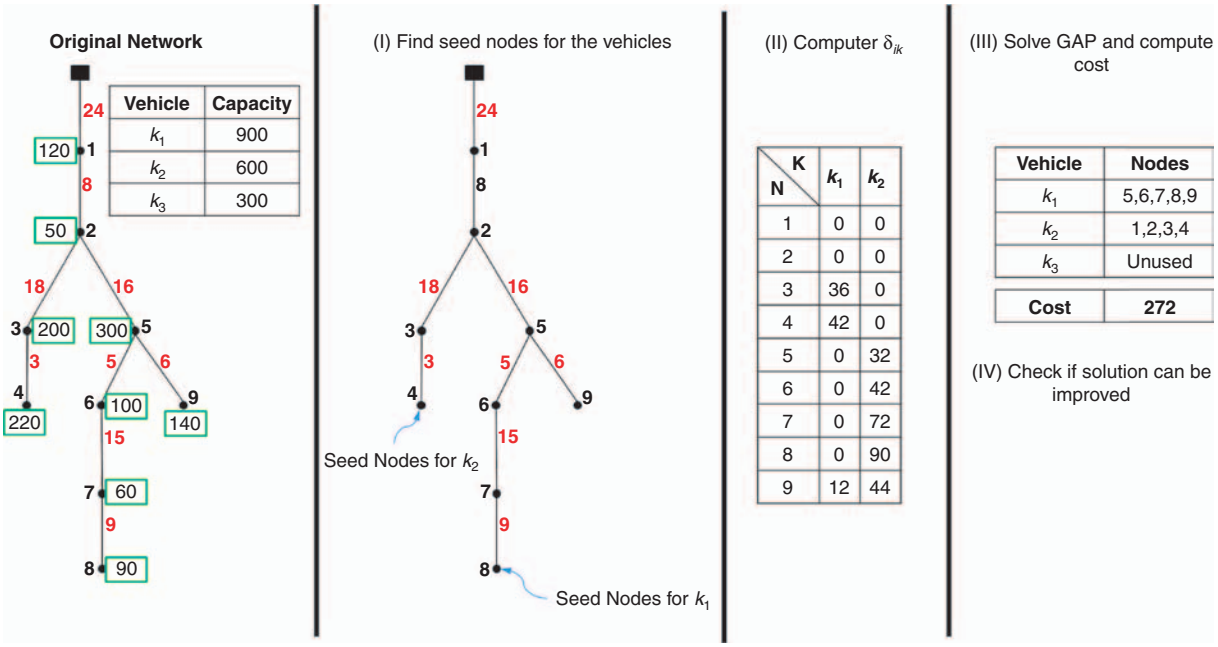

FIGURE 3 Illustration of steps in heuristic.

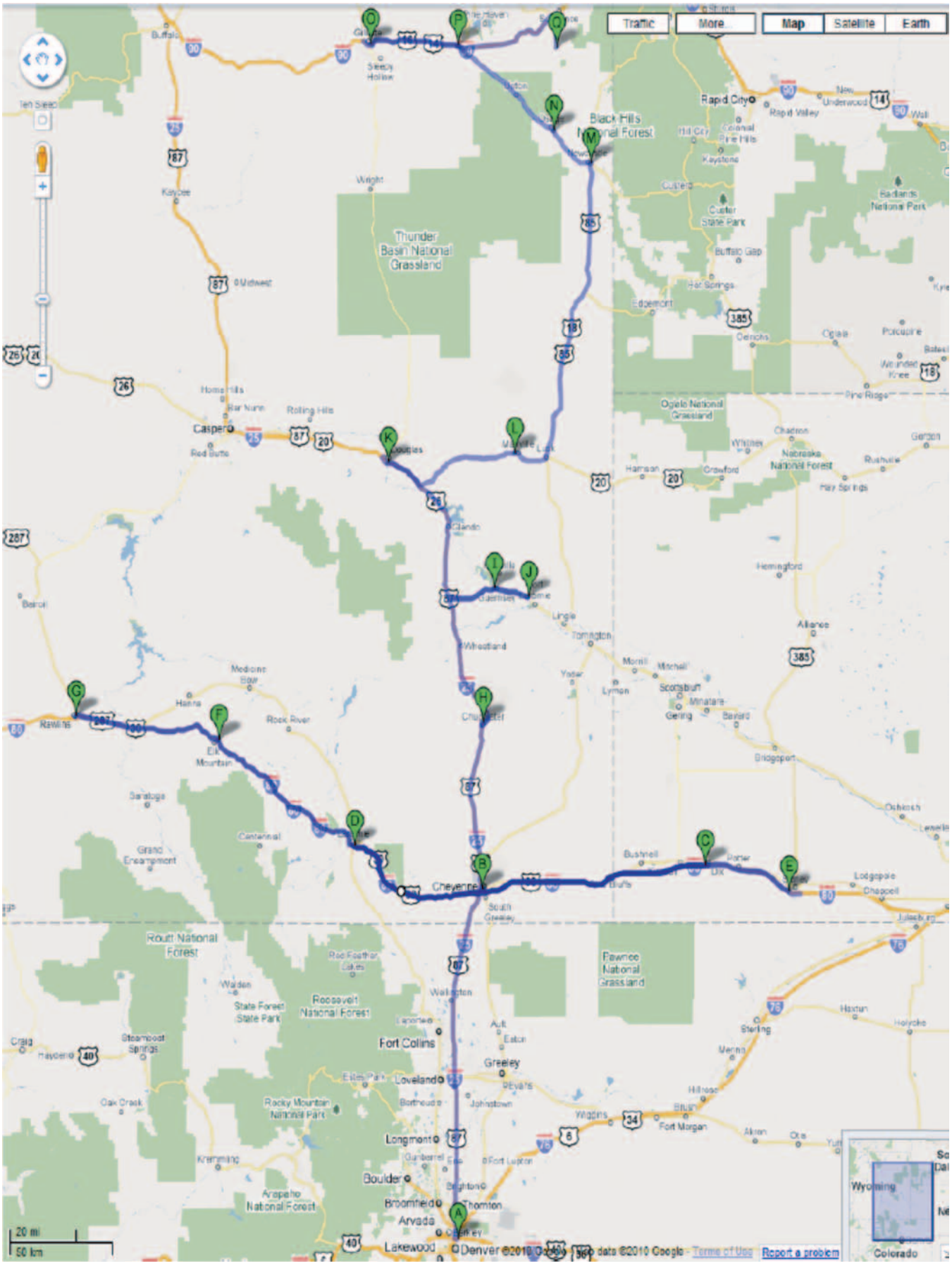

FIGURE 4 Tree network arising in Wyoming with Point A as depot. (Source: Google Maps.)

to have a degree of 1. The arc costs were uniformly distributed in [1 100]. The algorithm was tested on 20-, 40-, 70-, and 120-node networks.

Parameter Generation

Each test network was tested for five demand profiles uniformly distributed between [1 10], [1 30], [1 100], [20 20], and [20 80]. The minimum vehicle capacity for each instance was randomly generated to be one to two times the maximum vehicle demand. The number of vehicles $|K|$ was the upper bound on the number of vehicles required to serve the network when all vehicles have minimum capacity. This upper bound can either be a bin-packing upper bound or a TCVRP upper bound suggested by Chandran and Raghavan (*4*). Of the vehicles in $|K|$, 0% to 10% were assumed to be four times the minimum capacity, 0% to 20% were assumed to be three times the vehicle capacity, and 0% to 20% were assumed to be twice the vehicle capacity. The rest of the vehicles were assumed to be of minimum capacity. The fixed costs were assumed to be 0. The paper studies the effectiveness of the seed node selection and the contraction and refining operations, which are not dependent on vehicle fixed costs.

Computational Results

For every demand profile and for every node size, 10 instances were solved. Thus, in total, the heuristic was tested for its solution quality and computational efficiency on 200 networks. The exact formulation, along with the valid inequalities, was coded in GAMS (version 22.9) and solved with the CPLEX solver on a PC with a 32-bit architecture, 4 GB of RAM, and a 2.93-GHz processor. In GAMS, the iteration limit for branch and bound was set to 100,000, the time limit was set to 1,000 s, and the optimality criterion for the mixed IP (relative gap between the best found integer solution value and best found linear programming value) was set to 0.0001. The heuristic was coded in MATLAB. A summary of the computational results is presented in Table 1. The summary contains the following information:

- Solved: the number of instances (of 10) that were solved to optimality within 1,000 s.
- TimeOPT: the time taken by CPLEX to solve the mixed IP formulation to optimality (reported only for solved instances).
- Time$H1$ and Time$H2$: the time taken to solve the heuristic by use of VehicleAssign$_{H1}$ and VehicleAssign$_{H2}$, respectively. It is the sum of times required to solve VehicleAssign, Contract, GAP, and refinement algorithms.
- Gap$H1$ and Gap$H2$: the gaps between the heuristic algorithm $H1$ and $H2$ and the optimal solution (z^*), respectively. This gap is given by $(\overline{z} - z^*)/z^*$. It conveys how far off the algorithm solution value was from the optimal solution value and is reported only for solved instances of mixed IP.
- First Gap$H1$ and First Gap$H2$: the gaps between the heuristic algorithm $H1$ and $H2$ solution value and the first best integer solution found by the CPLEX solver (z_F), respectively. It is computed as $(\overline{z} - z_F)/\overline{z}$ and is reported only for unsolved instances.
- Last Gap$H1$ and Last Gap$H2$: the gaps between the heuristic algorithm solution value and the last best integer solution found by the CPLEX solver (z_L), respectively. It is computed as $(\overline{z} - z_L)/\overline{z}$ and is reported only for unsolved instances.

Solution Quality

Table 1 shows that for solved instances, the gap varied from 2% to 8% for heuristic $H1$ and from 3% to 11% for heuristic $H2$. On average, $H1$ seems to perform better than $H2$. Also, as the problem size increased, the gap did not increase substantially for either $H1$ or $H2$. This is because the seed node selection depended on the network geometry and not on the network size. Moreover, δ_{ik} was computed on the basis of tree geometry and not on the basis of network size. The demand distribution [1 10] was the easiest to solve for the heuristic and for CPLEX, whereas the networks with demand [20 20] were the hardest to solve. The [20 20] demand is such that the linear programming relaxation produces many fractional solutions, as a result of which the number of branch-and-bound nodes increases substantially. Only a few 70-node instances could not be solved to optimality, whereas all 120-node instances remained unsolved after 1,000 s. This is because the method used to generate $|K|$ results in high $|K|$ values for larger problems, and the result thereby increases the number of variables that need to be branched. The first gaps reported for the unsolved instances were at least 53% lower than the best initial integer solution found by CPLEX. Therefore, the heuristic can be used to initialize the CPLEX solver effectively. Finally, the last gap results suggest that the heuristic solutions were close to the best last integer solution found by CPLEX at termination. In fact, for 120-node networks with [20 20] demand, the heuristic solution was better than the last CPLEX solution. This reaffirms that for tougher problems, the heuristic can be used to find good solutions.

Computational Performance

As expected, the CPLEX solution time increases with problem size. For 120-node networks, CPLEX was unable to find a solution within 1,000 s. The heuristic took much less time to find good solutions that were well within 10% of the optimal solution for most cases. $H1$ performed marginally better than $H2$, because $H2$ required one extra step. The maximum reported time required to solve the heuristic without application of the refining operation was only about 5 s. The refinement operation performs $|N|^2|K|$ iterations for every network. Therefore, because of their high $|K|$ values, the time taken for instances with [20 20] and [20 80] demands was higher than the time taken for other demand profiles. Thus, if one wanted to initialize CPLEX with a solution, it can be done so extremely quickly.

CONCLUSIONS

In this paper, a special case of the VRP in which the network is a tree and the vehicle fleet is fixed and heterogeneous was studied. Because of this heterogeneity, existing heuristic solution methods cannot be used to solve the problem.

The relation of Cap-HTVRP to GAP and CFLP was discussed. It was shown that when the nodes were ordered in DFS order, the existing IP formulation can be modified to have only GAP constraints. A linear approximation to this modified formulation was presented. A heuristic was then developed to use this GAP formulation to solve the Cap-HTVRP. The heuristic iteratively finds seed nodes by use of upper-bound heuristics for the bin-packing problem, assigns a vehicle to each seed node, and then uses a Lagrangian-based GAP algorithm to assign nodes to vehicles located at the seed node. Two methods for detection of seed nodes were also presented.

TABLE 1 Summary of Numerical Results

Network	Demand	Solved	Nodes	Gap$H1$	Gap$H2$	TimeOPT (s)	Time$H1$ (s)	Time$H2$ (s)	First Gap$H1$	First Gap$H2$	Last Gap$H1$	Last Gap$H2$
20-node	[1 10]	10	398	0.027	0.029	1.35	0.52	0.51	—	—	—	—
	[1 30]	10	202	0.037	0.053	0.95	0.31	0.33	—	—	—	—
Includes the	[1 100]	10	339	0.05	0.051	1.67	0.57	0.61	—	—	—	—
two real-world	[20 20]	10	3,790	0.027	0.064	2.93	1.23	1.23	—	—	—	—
networks	[20 80]	10	5,625	0.026	0.059	5.25	0.89	0.93	—	—	—	—
	[1 10]	10	4,054	0.033	0.063	5.88	2.13	2.24	—	—	—	—
	[1 30]	10	1,259	0.056	0.066	4.98	2.46	2.53	—	—	—	—
40-node	[1 100]	10	1,047	0.07	0.056	44.49	2.35	2.97	—	—	—	—
	[20 20]	10	3,764	0.026	0.099	32.72	7.82	8.1	—	—	—	—
	[20 80]	10	3,256	0.073	0.078	15.74	4.22	4.93	—	—	—	—
	[1 10]	10	9,244	0.079	0.103	195.68	11.86	12.54	—	—	—	—
	[1 30]	8	30,005	0.07	0.082	358.5	12.42	13.24	–0.65	–0.58	0.047	0.044
70-node	[1 100]	10	9,722	0.083	0.113	133.9	7.45	7.79	—	—	—	—
	[20 20]	0	19,028	—	—	—	48.65	53.57	–0.718	–0.564	0.056	0.133
	[20 80]	5	33,719	0.085	0.087	585.8	39.65	40.67	–0.525	–0.488	0.133	0.156
	[1 10]	0	10,618	—	—	—	28.45	29.35	–0.574	–0.566	0.104	0.102
	[1 30]	0	30,005	—	—	—	32.46	32.36	–0.589	–0.597	0.123	0.118
120-node	[1 100]	0	12,318	—	—	—	30.23	30.3	–0.749	–0.694	0.081	0.112
	[20 20]	0	2,429	—	—	—	95.67	98.54	–743	–0.615	–0.045	0.029
	[20 80]	0	9,327	—	—	—	79.94	80.59	–0.614	–0.613	0.091	0.091

NOTE: — = not applicable.

The heuristic was tested on 200 test networks of various sizes. It was found that the heuristic performs consistently well, irrespective of problem size, with solutions ranging from 2% to 10% of the optimal solution value, whereas it takes much less time than the CPLEX solver. It was also shown that significant improvements in optimal solution time can be achieved if the heuristic solution was used to initialize the CPLEX solver.

Future research will include exploration of other variants of the problem, such as the TVRP with time windows and stochastic and online versions of TCVRPs. Exploration of other heuristic solution techniques that explicitly take advantage of the tree structure must also be explored.

REFERENCES

1. Labbé, M., G. Laporte, and H. Mercure. Capacitated Vehicle Routing on Trees. *Operations Research,* Vol. 39, No. 4, 1991, pp. 616–622.
2. Berger, I., J.-M. Bourjolly, and G. Laporte. Branch-and-Bound Algorithms for the Multi-Product Assembly Line Balancing Problem. *European Journal of Operational Research,* Vol. 58, No. 2, 1992, pp. 215–222.
3. Basnet, C., L. Foulds, and J. Wilson. Heuristics for Vehicle Routing on Tree-Like Networks. *Journal of the Operational Research Society,* Vol. 50, 1999, pp. 627–635.
4. Chandran, B., and S. Raghavan. Modeling and Solving the Capacitated Vehicle Routing Problem on Trees. In *The Vehicle Routing Problem: Latest Advances and New Challenges,* (B. Golden, S. Raghavan, and E. Wasil, eds.), Springer, New York, 2008, pp. 239–261.
5. Kumar, R., A. Unnikrishnan, and S. T. Waller. Capacitated-Vehicle Routing Problem with Backhauls on Trees: Model, Properties, Formulation, and Algorithm. In *Transportation Research Record: Journal of the Transportation Research Board, No. 2224,* Transportation Research Board of the National Academies, Washington, D.C., 2011, pp. 92–102.
6. Hamaguchi, S., and N. Katoh. A Capacitated Vehicle Routing Problem on a Tree. *Algorithms and Computation,* Vol. 1533, 1998, pp. 399–407.
7. Garey, M., and D. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness.* A Series of Books in the Mathematical Sciences. W. H. Freeman and Company, San Francisco, Calif., 1979.
8. Martello, S., and P. Toth. Lower Bounds and Reduction Procedures for the Bin Packing Problem. *Discrete Applied Mathematics,* Vol. 28, No. 1, 1990, pp. 59–70.
9. Scholl, A., R. Klein, and C. Jürgens. Bison: A Fast Hybrid Procedure for Exactly Solving the One-Dimensional Bin Packing Problem. *Computers and Operations Research,* Vol. 24, No. 7, 1997, pp. 627–645.
10. Vanderbeck, F. Computational Study of a Column Generation Algorithm for Bin Packing and Cutting Stock Problems. *Mathematical Programming,* Vol. 86, No. 3, 1999, pp. 565–594.
11. Fleszar, K., and K. Hindi. New Heuristics for One-Dimensional Bin-Packing. *Computers and Operations Research,* Vol. 29, No. 7, 2002, pp. 821–839.
12. Lewis, R. A General-Purpose Hill-Climbing Method for Order Independent Minimum Grouping Problems: A Case Study in Graph Colouring and Bin Packing. *Computers and Operations Research,* Vol. 36, No. 7, 2009, pp. 2295–2310.
13. Tsitsiklis, J. Special Cases of Traveling Salesman and Repairman Problems with Time Windows. *Networks,* Vol. 22, No. 3, 1992, pp. 263–282.
14. Clarke, C., and J. Q. Wright. Scheduling of Vehicle from a Central Depot to a Number of Delivery Points. *Operations Research,* Vol. 12, No. 4, 1964, pp. 568–581.
15. Rennie, S. Optimal Dispatching and Routing of Milk Tanker for Northland Dairy Board. *Proc., 30th Annual Conference of the Operational Research Society of New Zealand.* Operational Research Society of New Zealand, Wellington, 1995, pp. 95–102.
16. Mbaraga, P., A. Langevin, and G. Laporte. Two Exact Algorithms for the Vehicle Routing Problem on Trees. *Naval Research Logistics,* Vol. 46, No. 1, 1999, pp. 75–89.
17. Busch, I. *Vehicle Routing on Acyclic Networks.* PhD dissertation. Department of Applied Mathematics, Johns Hopkins University, Baltimore, Md., 1990.
18. Asano, T., N. Katoh, and K. Kawashima. A New Approximation Algorithm for the Capacitated Vehicle Routing Problem on a Tree. *Journal of Combinatorial Optimization,* Vol. 5, No. 2, 2001, pp. 213–231.
19. Katoh, N., and T. Yano. An Approximation Algorithm for the Pickup and Delivery Vehicle Routing Problem on Trees. *Discrete Applied Mathematics,* Vol. 154, No. 16, 2006, pp. 2335–2349.
20. Laporte, G., Y. Nobert, and M. Desrochers. Optimal Routing Under Capacity and Distance Restrictions. *Operations Research,* Vol. 33, No. 5, 1985, pp. 1050–1073.
21. Fisher, M., and R. Jaikumar. A Generalized Assignment Heuristic for Vehicle Routing. *Networks,* Vol. 11, No. 2, 1981, pp. 109–124.
22. Choi, E., and D. Tcha. A Column Generation Approach to the Heterogeneous Fleet Vehicle Routing Problem. *Computers and Operations Research,* Vol. 34, No. 7, 2007, pp. 2080–2095.
23. Golden, B., A. Assad, L. Levy, and F. Gheysens. The Fleet Size and Mix Vehicle Routing Problem. *Computers and Operations Research,* Vol. 11, No. 1, 1984, pp. 49–66.
24. Salhi, S., and G. Rand. Incorporating Vehicle Routing into the Vehicle Fleet Composition Problem. *European Journal of Operational Research,* Vol. 66, No. 3, 1993, pp. 313–330.
25. Taillard, É. A Heuristic Column Generation Method for the Heterogeneous Fleet VRP. *RAIRO Operations Research,* Vol. 33, No. 1, 1999, pp. 1–14.
26. Desrochers, M., and T. Verhoog. A New Heuristic for the Fleet Size and Mix Vehicle Routing Problem. *Computers and Operations Research,* Vol. 18, No. 3, 1991, pp. 263–274.
27. Renaud, J., and F. Boctor. A Sweep-Based Algorithm for the Fleet Size and Mix Vehicle Routing Problem. *European Journal of Operational Research,* Vol. 140, No. 3, 2002, pp. 618–628.
28. Desrosiers, J., Y. Dumas, M. Solomon, and F. Soumis. Time Constrained Routing and Scheduling. *Handbooks in Operations Research and Management Science,* Vol. 8, 1995, pp. 35–139.
29. Jeet, V., and E. Kutanoglu. Lagrangian Relaxation Guided Problem Space Search Heuristics for Generalized Assignment Problems. *European Journal of Operational Research,* Vol. 182, No. 3, 2007, pp. 1039–1056.

The Transportation Network Modeling Committee peer-reviewed this paper.