Vehicle Routing for Food Rescue Programs: A comparison of different approaches

Canan Gunes, Willem-Jan van Hoeve, and Sridhar Tayur

Tepper School of Business, Carnegie Mellon University

1 Introduction

The 1-Commodity Pickup and Delivery Vehicle Routing Problem (1-PDVRP) asks to deliver a single commodity from a set of supply nodes to a set of demand nodes, which are unpaired. That is, a demand node can be served by any supply node. In this paper, we further assume that the supply and demand is unsplittable, which implies that we can visit each node only once. The 1-PDVRP arises in several practical contexts, ranging from bike-sharing programs in which bikes at each station need to be redistributed at various points in time, to food rescue programs in which excess food is collected from, e.g., restaurants and schools, and redistributed through agencies to people in need. The latter application is the main motivation of our study.

Pickup and delivery vehicle routing problems have been studied extensively; see, e.g., Berbeglia et al. [1] for a recent survey. However, the 1-commodity pickup and delivery vehicle routing problem (1-PDVRP) has received limited attention. When only one vehicle is considered, the problem can be regarded as a traveling salesman problem, or 1-PDTSP. For the 1-PDTSP, different solution methods have been proposed, including [3, 4]. On the other hand, the only paper that addresses the 1-PDVRP is by Dror, Fortin, and Roucairol [2], to the best of our knowledge. Dror et al. [2] present different approaches, including MIP, CP and Local Search, which are applied to instances involving up to nine locations.

The main goal of this work is to compare off-the-shelf solution methods for the 1-PDVRP, using state-of-the-art solvers. In particular, how many vehicles, and how many locations, can still be handled (optimally) by these methods? The secondary goal of this work is to evaluate the potential (cost) savings in the context of food rescue programs. We note that the approaches we consider (MIP, CP, CBLS) are similar in spirit to those of Dror et al. [2]. Our MIP model is quite different, however. Further, although the CP and CBLS models are based on the same modeling concepts, the underlying solver technology has been greatly improved over the years.

2 Different Approaches to the 1-PDVRP

2.1 Input Data and Parameters

Let the set V denote the set of locations, and let $O \in V$ denote the origin (or depot) from which the vehicles depart and return. With each location i in V we associate a number $q_i \in \mathbb{R}$ representing the quantity to be picked up ($q_i > 0$) or delivered ($q_i < 0$) at i. The distance between two locations i and j in V will be denoted by d_{ij}. Distance can be represented by length or time units.

Let T denote the set of vehicles (or trucks). For simplicity, we assume that all vehicles have an equal 'volume' capacity Q of the same unit as the quantities q to be picked up (e.g., pounds). In addition, all vehicles are assumed to have an equal 'horizon' capacity H of the same unit as the distances d.

2.2 Mixed Integer Programming

Our MIP model is based on column generation. The master problem of our column generation procedure consists of a set of 'columns' S representing feasible routes. The routes are encoded as binary vectors on the index set V of locations; that is, the actual order of the route is implictly encoded. The columns are assumed to be grouped together in a matrix A of size V by S. The length of the routes is represented by a 'cost' vector $c \in \mathbb{R}^{|S|}$. We let $z \in \{0,1\}^{|S|}$ be a vector of binary variables representing the selected routes. The master problem can then be encoded as the following set covering model:

$$\begin{aligned} &\min c^{\mathsf{T}} z \\ &\text{ s.t. } Az = \mathbf{1} \end{aligned} \tag{1}$$

For our column generation procedure, we will actually solve the continous relaxation of (1), which allows us to use the shadow prices corresponding to the constraints. We let λ_j denote the shadow price of constraint j in (1), where $j \in V$.

The subproblem for generating new feasible routes uses a model that employs a flow-based representation on a layered graph, where each layer consists of nodes representing all locations. The new route comprises M steps, where each step represents the next location to be visited. We can safely assume that M is the minimum of $|V| + 1$ and (an estimate on) the maximum number of locations that 'fit' in the horizon H for each vehicle.

We let x_{ijk} be a binary variable that represents whether we travel from location i to location j in step k. We further let y_j be a binary variable representing whether we visit location j at any time step. The vector of variables y will represent the column to be generated. Further, variable I_k represents the inventory of the vehicle, while variable D_k represents the total distance traveled up to step k, where $k = 0, \ldots, M$. We let $D_0 = 0$, while $0 \leq I_0 \leq Q$. The problem of finding an improving route can then be modeled as presented in Figure 1.

In this model, the first four sets of constraints ensure that we leave from and finish at the origin. The fifth set of constraints enforce that we can enter the origin at any time, but not leave it again. The sixth set of constraints model the flow conservation at each node, while the seventh set of constraints (the first set in the right column) prevent the route from visiting a location more than once. The following four sets of constraints represent the capacity constraints of the vehicle in terms of quantities picked up and delivered, and in terms of

$$
\begin{array}{lllll}
\min & \sum_{i\in V}\sum_{j\in V}\sum_{k=1}^{M} d_{ij}x_{ijk} - \sum_{j\in V}\lambda_j y_j & & & \\
\text{s.t.} & \sum_{j\in V} x_{O,j,1} = 1 & & \sum_{j\in V\setminus\{O\}}\sum_{k=1}^{M} x_{ijk} \le 1 & \forall j \in V\setminus\{O\} \\
& \sum_{j\in V} x_{i,j,1} = 0 & \forall i \in V\setminus\{O\} & I_k = I_{k-1} + \sum_{i\in V}\sum_{j\in V} q_i x_{ijk} & \forall k \in [1..M] \\
& \sum_{i\in V} x_{i,O,M} = 1 & & 0 \le I_k \le Q & \forall k \in [0..M] \\
& \sum_{i\in V} x_{i,j,M} = 0 & \forall j \in V\setminus\{O\} & D_k = D_{k-1} + \sum_{i\in V}\sum_{j\in V} d_{ij}x_{ijk} & \forall k \in [1..M] \\
& x_{O,j,k} = 0 & \forall j \in V\setminus\{O\}, \forall k \in [1..M] & 0 \le D_k \le H & \forall k \in [0..M] \\
& \sum_{i\in V} x_{ijk} = \sum_{l\in V} x_{j,l,k+1} & \forall j \in V, \forall k \in [1..M-1] & \sum_{i\in V}\sum_{k=1}^{M} x_{ijk} = y_j & \forall j \in V
\end{array}
$$

Fig. 1. MIP model for finding an improving route.

distance. The last set of constraints link together the 'flow' variables x with the new column represented by the variables y.

As noted above, throughout the iterative process, we apply a continuous relaxation of the master problem (1). When this process terminates (it reaches a fixed point, or it meets a stopping criterion), we run the master problem as an integer program. Therefore, our procedure may not provably find the optimal solution, but it does provide a guaranteed optimality gap.

As a final remark, when only one vehicle is involved, the MIP model amounts to solving only the subproblem, to which the constraints are added that we must visit all locations.

2.3 Constraint Programming

Our CP model is based on a well-known interpretation of the VRP as a multi-machine job scheduling problem with sequence-dependent setup times. In the CP literature, this is usually modeled using alternative resources (the machines) and activities (the jobs). That is, each visit to a location corresponds to an activity, and each vehicle corresponds to two (linked) resources: one 'unary resource' modeling the distance constraint, and one 'reservoir' modeling the inventory of the vehicle. With each activity we associate variables representing its start time and end time, as well as a fixed duration (this can be 0 if we assume that the (un-)loading time is negligible). Further, each activity either depletes or replenishes the inventory reservoir of a vehicle. The distance between two locations is modeled as the 'transition time' between the corresponding activities. We minimize the sum of the completion times of all vehicles.

All these concepts are readily available in most industrial CP solvers. We have implemented the model in ILOG Solver 6.6 (which includes ILOG Scheduler). A snapshot of the ILOG model for a single vehicle is provided in Figure 2. It shows that the concepts presented above can almost literally be encoded as a CP model.

```
IloReservoir truckReservoir(ReservoirCapacity, 0);
truckReservoir.setLevelMax(0, TimeHorizon, ReservoirCapacity);

IloUnaryResource truckTime();
IloTransitionTime T(truckTime, Distances);

vector<IloActivity> visit;
visit = vector<IloActivity>(N);

for (int i=0; i<N; i++) {
  visit[i].requires(truckTime);
  if (supply[i] > 0)
    visit[i].produces(truckReservoir, supply[i]);
  else
    visit[i].consumes(truckReservoir, -1*supply[i]);
}
```

```
class RoutingModel {
   ...
   IloDimension2 _time;
   IloDimension2 _distance;
   IloDimension1 _weight;
   ...
}

IloNode node( <read coordinates from file> );

IloVisit visit(node);
visit.getTransitVar(_weight) == Supply);
minTime <= visit.getCumulVar(_time) <= maxTime;
visit.getCumulVar(_weight) >= 0);

IloVehicle vehicle(firstNode, lastNode);
vehicle.setCapacity(_weight, Capacity);
vehicle.setCost(_distance);
```

Fig. 2. Snapshots of the ILOG Scheduler model (left) and ILOG Dispatcher model (right), for a single vehicle.

2.4 Constraint-Based Local Search

Our final approach uses Constraint-Based Local Search (CBLS). With CBLS we can express the problem similar to a CP model, which will then be used to automatically derive the neighborhoods and penalty function needed to define a local search procedure. Our CBLS is based on the semantics offered by ILOG Dispatcher (included in ILOG Solver 6.6). These semantics are specifically designed to model routing problems.

ILOG Dispatcher uses the concepts *nodes*, *vehicles*, and *visits*. The nodes are defined by the coordinates of the locations, and contain as an attribute the amount to be picked up or delivered. The vehicles contain several attributes, including time, distance, and weight (load). Vehicles also contain, by default, a 'unary resource' constraint with respect to time, and a 'capacity' constraint with respect to the load, similar to the resources in ILOG Scheduler. The attributes of visits include the location, the quantity to be picked up (positive) or delivered (negative), a time window, and possibly other problem-specific constraints.

In a first phase, we create a feasible solution. ILOG Dispatcher uses various heuristics for this, including a nearest-neighbour heuristic that we applied in our experiments. Where applicable, we started from the current schedule that we extracted from the data.

The second phase improves upon the starting solution using various local search methods. We applied successively the methods IloTwoOpt, IloOrOpt, IloRelocate, IloCross and IloExchange. Within each method, we take the first legal cost-decreasing move encountered.

3 Evaluation

Our experimental results are performed on data provided by the Pittsburgh Food Bank. Their food rescue program visits 130 locations per week. The provided data allowed us to extract a fairly accurate estimate on the expected pickup amount for the donor locations. The precise delivery amounts were unknown, and we therefore approximate the demand based on the population served by each location (which is known accurately), scaled by the total supply. We allow the total demand to be slightly smaller than the total supply, to avoid pathological behavior of the algorithm. We note however, that although this additional 'slack' influences the results, the qualitative behavior of the different techniques remains the same. The MIP model is solved using ILOG CPLEX 11.2, while the CP and CBLS model are solved using ILOG Solver 6.6, all on a 2.33GHz Intel Xeon machine.

The first set of instances are for individual vehicles, on routes serving 13 to 18 locations (corresponding to a daily schedule). The second set of instances group together schedules over multiple days, ranging from 30 to 130 locations. The results are presented in Figure 3. We report for each instance the cost savings (in terms of total distance traveled) with respect to the current operational schedule. Here, $|V|$ and $|T|$ denote the number of locations and vehicles, respectively. The optimal solutions found with MIP and CP took several (2–3) minutes to compute, while the solutions found with CBLS took several seconds or less. The time limit was set to 30 minutes.

Our experimental results indicate that on this problem domain, our MIP model is outperformed by our CP model to find an optimal solution (we note that a specialized 1-PDTSP MIP approach such as [4] might perform better than our 'generic' MIP model on the single-vehicle instances). Further, the CP model is able to find optimal solutions for up to 18 locations and one vehicle; for a higher number of locations or vehicles, the CP model is unable to find even a single solution. Lastly, the CBLS approach is able to handle large-scale instances, up to 130 locations and 9 vehicles. The expected savings are substantial, being at least 10% on the largest instance.

$\|V\|$	$\|T\|$	MIP	CP	CBLS
13	1	12%	12%	12%
14	1	15%	15%	14%
15	1	-	7%	6%
16	1	-	5%	3%
18	1	-	16%	15%
30	2	-	-	4%
60	4	-	-	8%
130	9	-	-	10%

Fig. 3. Savings obtained with different approaches.

Bibliography

[1] G. Berbeglia, J.F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: A classification scheme and survey. *TOP*, 15(1):1–31, 2007.

[2] M. Dror, D. Fortin, and C. Roucairol. Redistribution of self-service electric cars: A case of pickup and delivery. Technical Report W.P. 3543, INRIA-Rocquencourt, 1998.

[3] H. Hernández-Pérez and J.J. Salazar-González. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145:126–139, 2004.

[4] H. Hernández-Pérez and J.J. Salazar-González. The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks*, 50:258–272, 2007.