A Route-Splitting Approach to the Vehicle Routing Problem

Sungmin Kang
The Department of Business, The Catholic University of Korea
L. Joseph Thomas
Johnson Graduate School of Management, Cornell University

Abstract

The column generation process for the set-partitioning model of the vehicle routing problem requires repeated solutions of column generation subproblems which has a combinatorial structure similar to that of the traveling salesman problem. This limits the size of the problem that can be addressed. We introduce a new modeling approach, termed route-splitting, which splits each vehicle route into segments, and results in more tractable subproblems. A lower bounding scheme that yields an updated bound at each iteration of the column generation process is developed. Implementation issues, including a technique of controlling columns in the master problem, are explored. Lower bounds are computed on standard benchmark problems with up to 199 customers.

1. Introduction

Given sets of geographically scattered customers with known demands and vehicles with known capacity, the vehicle routing problem (VRP) is to determine a set of feasible vehicle routes, one for each vehicle, such that each customer is visited exactly once and the total distance traveled by the vehicles is minimized. A feasible route is defined as a simple circuit including the depot such that the total demand of the customers in the route does not exceed the given vehicle capacity.

Being at the interface of theory and application, the VRP has been intensively studied. While there have been significant advances in exact solution methodology, VRP is not a well solved problem. We find most approaches still relying on the branch and bound scheme. These approaches employ various methodologies to compute a lower bound on the cost of the optimal solution. We cite four distinct approaches in this category: dynamic programming, constraint relaxation, minimum K-tree, and continuous set-partitioning relaxation.

The state-space relaxation of Christofides, Mingozzi and Toth (1981) is an example of the dynamic programming approach. The main weakness of dynamic programming approaches is that the search tree becomes very large easily. The state-space relaxation method is a relaxation on the domain of the states using q-paths (chains of nodes whose total weight is equal to q) to reduce the number of states in the tree search. The largest problem solved was a 25-customer, 8-vehicle VRP.

Laporte, Nobert and Desrochers (1985) adopted a relaxation of sub-tour elimination constraints. Integrality of the solution is sought by the addition of Gomory cuts and afterwards by a branch and bound algorithm, where violated subtour elimination constraints are added as required. Problems with up to 50 customers were solved to optimality.

Fisher (1994) generalized the 1-tree relaxation of Held and Karp (1970) and introduced the K-tree relaxation for the standard VRP, where K is the number of vehicles. He dualized the constraints enforcing vehicle capacity and node degree to obtain a Lagrangean lower bound. Based upon this relaxation, he developed a branch and bound algorithm which optimally solved up to a 100-customer, 10-vehicle problem.

A set partitioning formulation of the VRP selects the optimal set of routes from a menu of feasible routes. The main obstacle is that all feasible routes must be considered. Since these are exponential in number, a column generation scheme must be devised. However, the cost coefficient for each column is the cost of the TSP tour on the customers in the route represented by the column. As a result, the column generation subproblems can be solved efficiently only when the number of customers is small, or when the number of the feasible routes is substantially reduced due to side constraints which enables an efficient dynamic programming solution.

For an instance of the former, see Agarwal, Mathur and Salkin (1989) for a scheme that optimally solves problems with up to 25 customers. Desrochers, Desrosiers and Solomon (1992), in an instance of the latter, report a very successful application of this approach towards exact solutions of 100-customer VRP with tight time windows. In the absence of side constraints, the set-partitioning approach is computationally limited. However, its strength lies in its ability to accommodate side constraints and other real world variations in problem characteristics, and often with gain in computational efficacy.

In this paper, we seek to improve the lower bounding from the continuous set partitioning/column generation approach to the VRP. We introduce a new modeling approach, called *route-splitting*, that reduces the computational burden of generating columns by splitting a vehicle route into a number of segments. This allows an efficient solution of the column generation subproblems and offers a chance to compute lower bounds for VRPs beyond the current computational range.

2. The Route-Splitting Approach

Let $\mathbb{N}$ be the set of N customers ($i = 1$ to N) with demand q_i, and let $\mathbb{K}$ be the set of K vehicles ($k = 1$ to K) located at the central depot~0 with capacity Q_k. Define $G = (\mathbb{N} \cup \{0\}, \mathbb{A})$ to be a network, where arc $(i,j) \in \mathbb{A}$ represents travel from node~i to node~j of G at distance d_{ij}. VRP can be formulated as a set-partitioning problem, in which we say a row (constraint) is "covered" by a column if the column has a value of 1 in the row. The objective is to choose a minimum-cost set of columns such that the rows are partitioned into subsets each covered by exactly one column.

Let R be the set of all feasible routes, and let a_r be a binary column representing route $r \in R$. Define x_r to be a decision variable having a value 1 if route r is taken, and 0 otherwise. Let d_r be the traveling distance of route r, the sum of the distance of the arcs in the route. The set-partitioning model of VRP is then

(SPP) minimize $\sum_{r \in R} d_r x_r$

subject to

$$\sum_{r \in R} a_r x_r = 1$$

$$\sum_{r \in R} x_r = K$$

where a_r satisfies the feasibility condition $q \cdot a_r \leq Q$. Here q is an vector of customer demands, and Q is the vehicle capacity.

Define the *cardinality* of a route to be the number of customers in the route. Due to the combinatorial nature, the column generating subproblem can be solved efficiently when the route cardinality of each route is small. This observation suggests a new VRP model in which routes are split into smaller segments called *pieces*. The route-splitting approach has the effect of transferring the computational burden from the subproblem to the master problem. It is worth the effort since the master problem is a linear programming problem, whereas the subproblem is a combinatorial problem. The master problem, being an LP, is computationally less sensitive to size.

The route-splitting approach allows an asymmetric distance matrix and non-identical vehicles. The objective of the route-splitting VRP is to find a set of pieces which will be connected to make a set of routes that is the optimal solution to the VRP.

In the route-splitting model (denoted RS), a column (variable) represents a piece. The number of pieces in a route is fixed *a priori*. Let ns be the fixed number of pieces in a route. Now x_r has a value of 1 if the piece represented by the column $r \in R$ is chosen as a segment of a route in the solution to the VRP, and 0 otherwise, where R is the set of all columns representing valid pieces.

Below, we will use terms variable r and piece r interchangeably. Let $b(r)$ be the first node in piece r, and $e(r)$, the last node. For 1-piece r, $b(r)$=0, and $e(r) = 0$ for ns-piece r. For 0-piece r, $b(r) = e(r)$ =0. Let $N(r)$ be the set of customers in piece r. The traveling distance of piece r, d_r, is the sum of the distances of the arcs in piece~r. The load of piece~r, q_r, is the sum of the demands of the customers in piece~r excluding that of $e(r)$: that is, $q_r = \sum_{i \in N(r) \setminus e(r)} q_i$. Finally, let $R(k,s)$ be the set of all valid s-pieces generated for vehicle k, and define $R(k)$ be the union of all $R(k,s)$. We now formulate RS as:

(RS) minimize $\sum_{r \in R} d_r x_r$

subject to

$$\sum_{r \mid i \in N(r) \setminus e(r)} x_r = 1 \quad \forall i$$

$$\sum_{r \in R(k)} q_r x_r \leqq Q_k \quad \forall k$$

$$\sum_{r \in R(k,0)} x_r + \sum_{r \in R(k,s)} x_r = 1 \quad \forall k, s$$

$$\sum_{r \in R(k,s) \mid b(r)=i} x_r - \sum_{r \in R(k,s-1) \mid e(r)=i} x_r = 0 \quad \forall i, k, s \geqq 2$$

$$x_r \in \{0,1\} \quad \forall r \in R$$

A set of valid pieces should conform to the following conditions to be a feasible solution to the VRP:

1. They comprise K groups, each of which consists of either a single 0-piece or ns pieces such that each piece is connected to the next piece in sequence.
2. Total demand of each group does not exceed the corresponding vehicle capacity.
3. Each customer is visited exactly once.

3. Solving the LP-Relaxation of RS

We consider a LP relaxation of RS (LRS) to be solved by a column generation process. We then introduce a method to obtain lower bounds that are to be updated during the column generation process. The column generation process starts each iteration with a subset of feasible columns that contains a feasible basis. These columns define an LP which we call a (restricted) master problem; the dual vector obtained by solving it is used to form a subproblem to determine the minimum reduced cost column among all feasible columns to the master problem. If the subproblem fails to find a column with a negative reduced cost, the algorithm terminates and the current solution of the restricted master problem is optimal to the unrestricted LP containing all feasible columns. Otherwise, a column with a negative reduced cost is found, added to the master problem, and the algorithm solves the master problem again.

While difficult to solve by a standard IP approach, the subproblem is constrained enough to be solved by a *tree-search-with-bounds* algorithm. The algorithm traverses a search tree for each piece type. The root of each tree consists of a singleton: a customer, except for a 0-piece or a 1-piece in which it is the depot. Traversing down the tree, a customer is sequentially added to the piece. Note that the cardinality restriction constrains the tree's height and provides a lower bound on the reduced cost of the column, allowing faster pruning. Algorithmic descriptions of the column generation subproblem are given in Kang (1994).

4. Computational Issues

For the standard VRP, a good feasible solution can be found using one of the several fast heuristics available. In our experience, such initialization did not lead to a faster convergence. Instead a basis initialization using artificial variables was superior. Some previous research using column generation, Lavoie, Minoux and Odier (1988), reports a similar experience.

Multiple pricing is to append to the master problem not only the minimal reduced cost column, but also several other negative reduced cost columns in each iteration. It often leads to a reduction in the total number of iterations, but at the cost of increased solution time for each iteration. This multiple pricing scheme was effective in accelerating the convergence of the column generation process (Desrochers and Soumis, 1989). Our subproblem solution algorithm allows multiple pricing without additional computation.

Computationally, there are two important concerns in implementing a column generation algorithm. First, the management of a typically large number of columns. We note that the columns generated in the route-splitting approach are likely to be fewer than in the set-partitioning approach. The reason is that pieces can be used to form many routes in combination. Once generated, a piece is more "versatile" than a route. However, the problem of a large number of columns remains. Second, slow convergence, or stalling, due to heavy degeneracy is not uncommon with set-partitioning based formulations (Desrochers, Desrosiers and Solomon, 1992), because each positive basic variable can cover many rows exactly. Other variables with a positive coefficient in already covered rows are forced to zero values. This means that we have many zero-valued basic variables which can be chosen freely from a number of forced-to-zero variables. This results in many bases associated with any feasible schedule.

We now introduce an implementation scheme to deal with the above computational concerns. In the column generation process, once a variable leaves the basis, it is not likely to enter the basis again, because dual variables in an earlier stage are generally far different from the optimal set of dual variables. We applied the following scheme, called *column-set reduction*, in our implementation: Delete all non-positive variables, including basic variables, when (1) the number of columns is larger than a predetermined limit, and (2) the improvement in the objective function value since the last column-set reduction exceeds a predetermined threshold value. Even though some of the deleted columns may have to be re-generated during the process, the total computational time will be reduced significantly since master problems stay relatively small so as to be solved fast, and the convergence problem would be relieved as the number of zero-valued columns, the cause of degeneracy, are controlled to a relatively small number. We tried less radical alternatives such as not deleting some number of most recently generated columns, pricing out the set of deleted columns, and solving the subproblem only when no negative reduced cost columns are found among the deleted ones. Our computational experiments supported the scheme outlined above.

5. Computational Experiences

The algorithm was tested on five test problems. Each problem has identical vehicles and a symmetric distance matrix. Problems v50, v75, and v100 are taken from Eilon, Watson-Gandy and Christofides (1971). Problem v150 was obtained by aggregating the customers of problems v50 and v100 with the depot and vehicle capacities as in v100. Problem v199 was obtained by aggregating the customers of problem v150 with the first 49 customers of problem v75. Euclidean distance between the points is computed as a real value.

	UB	KT LB	KT CPU	maxBr	RS CPU	RS LB
v50	525	507	96	4	1	502
				5	1	503
				6	5	504
v75	835	756	184	4	3	766
				5	3	770
				6	51	772
v100	826	786	308	4	13	768
				5	17	772
v150	1028	933	682	4	73	914
				5	103	924
v199	1335	1097	1186	4	234	1148
				5	288	1156

1. CPU times are in minutes
2. UB is the OFV of the known best solutions.
3. KT LB is the lower bounds by Fisher's K-tree approach (1994)
4. RS LB is the lower bounds by Fisher's K-tree approach (1994)
5. KT CPU is the CPU time on an Apollo Domain 3000, and RS CPU, on Pentium II 800MHz..
6. maxBr is the maximum cardnality of a piece.

Results of B&B Search on Cardinalities of the Vehicles for Lower Bounds

	maxBr	depth	CPU	LB
v50	5	1	2	507
		2	3	508
		3	4	509
		4	8	509
		5	8	509
v75	5	2	12	783
v100	5	2	90	786
v150	5	2	418	939
v199	5	2	980	1170

0. Times are in minutes.
1. "depth" is the number of vehicles whose cardinalities are limited in range.