A HEURISTIC BASED ON MODIFIED LAGRANGIAN RELAXATION FOR THE VEHICLE ROUTING PROBLEM

Kuancheng HUANG  Tai-Yi WU
Assistant Professor  Research Associate
Dept. of Transportation Technology  Dept. of Transportation Technology
and Management  and Management
National Chiao Tung University  National Chiao Tung University
No. 1001, Ta Hsueh Road, Hsinchu City  No. 1001, Ta Hsueh Road, Hsinchu City
300 Taiwan  300 Taiwan
Fax: 886-3-5720844  Fax: 886-3-5720844
E-mail: kchuang@cc.nctu.edu.tw  E-mail: reticent.tem93g@nctu.edu.tw

Abstract: Given the high level of complexity and wide application of the vehicle routing problem (VRP), numerous solution algorithms have been developed for the past several decades, including many recent meta-heuristic algorithms with great success and popularity. In order to balance computational load and solution quality and to address the issue of flexibility and simplicity, this study developed a heuristic algorithm based on several classical mathematical programming techniques. The VRP is first formulated in the form of the set covering problem (SCP), and the Lagrangian relaxation is used as the backbone in designing the iterative algorithm. In addition, a concept similar to column generation is used to maintain a partial set of potential routes to reduce computational load. Based on the numerical experiment, the solution quality of the heuristic algorithm is stable. The result suggests that the solution algorithm should be able to deal with the operational problems arising from a highly dynamic environment.

Key Words: vehicle routing problem, set covering problem, Lagrangian relaxation

1. INTRODUCTION

Transportation cost is the largest component of the overall logistics costs in many industry sectors. Especially, in today’s business environment, customers tend to favor small and frequent shipments, which greatly complicate transportation operation and increase operating cost. Among the operational decisions faced by the suppliers or distributors, the Vehicle Routing Problem (VRP) is one of the most important problems.

VRP, a combinatorial optimization problem, can be viewed as a merge of two classical problems: the Traveling Salesman and the Bin Packing. As in the literature, it has drawn significant attention from the researchers. In particular, the development of the meta-heuristics has been a success in solving various types of VRPs. However, owing to the inherited complexity, it is still hard to solve large-scale problems within a short period of time. In addition, it remains difficult to take into consideration the operational requirements from all kinds of practical situations.

The goal of this research is to develop a suitable algorithm as the core module of the decision support system for the trucking companies or the operators of a large fleet. The VRP is first transformed to a well-known set covering problem (SCP) by treating a feasible route as a set. Lagrange Relaxation, a successful approach for SCP reported in prior researches, is used as the backbone to develop a recursive heuristic algorithm. However, the space of the potential routes (sets) in the SCP is huge as the number of feasible routes is enormous. Thus, based on a concept similar to column generation, only a partial set of the feasible routes is initially generated and carefully adjusted through the iterative solution procedure.
This paper is organized as follows: in the next chapter, in addition to problem formulation, some VRP related researches are reviewed, and a brief discussion about the features of various types of algorithms is provided. The solution algorithm developed in the study is presented in the third chapter. The numerical experiment based on the well-accepted test problems is described in the fourth chapter. Finally, the findings of this study are concluded in the fifth chapter.

2. SOLUTION ALGORITHMS FOR VRP AND PRIOR RESEARCHES

The formulation of VRP is presented as an integer programming (IP) model in the first section. As numerous algorithms have been developed for VRP, the general classification of VRP algorithms is provided in the second section. In addition, the key ideas and the solution techniques from the prior representative researches are provided. Finally, one specific approach, set-covering based algorithms, is discussed in the third section, as it is the foundation of the heuristic algorithm of this study.

2.1 Formulation of the Vehicle Routing Problem

The basic version of VRP, usually referred to as CVRP (capacitated vehicle routing problem), can be viewed as a merge of two classical problems: the Traveling Salesman and the Bin Packing. All customers must be assigned to one of the routes and served by a vehicle. The demands are deterministic and may not be split. The vehicles are identical and based at a single depot, and only the vehicle capacity constraint is imposed. The objective is to minimize the total cost, i.e., the sum of all route costs. The following is one of the various formulations for VRP (Bodin et al., 1983):

\[
\begin{align*}
\text{Min} & \quad \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{v=1}^{M} C_{ij} x_{ij}^v \\
\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij}^k & = 1 \quad j = 1, \ldots, N \\
\sum_{j=1}^{N} \sum_{k=1}^{M} x_{ij}^k & = 1 \quad i = 1, \ldots, N \\
\sum_{i=0}^{N} x_{ih}^k - \sum_{j=0}^{N} x_{ij}^k & = 0 \quad h = 0, \ldots, N; \; k = 1, \ldots, M \\
\sum_{i=0}^{N} d_{i}(\sum_{j=0}^{N} x_{ij}^k) & \leq K \quad k = 1, \ldots, M \\
\sum_{j=0}^{N} x_{ij}^k & \leq 1 \quad k = 1, \ldots, M \\
\sum_{i=0}^{N} x_{0j}^k & \leq 1 \quad k = 1, \ldots, M \\
\sum_{i=0}^{N} x_{ij}^k & \leq 1 \quad k = 1, \ldots, M \\
\sum_{i=0}^{N} \sum_{j=0}^{N} x_{ij}^k & \leq |S| - 1 \quad \forall \text{ subset } S \text{ with } |S| \geq 2, \forall k = 1, \ldots, M \\
x_{ij}^k & = \text{binary} \quad i = 1, \ldots, N \quad j = 1, \ldots, N \quad k = 1, \ldots, M
\end{align*}
\]

- \(i, j, h\): index for customers (0 represents the depot, and \(N\) is the number of customers.)
- \(k\): index for vehicles (\(M\) is the number of available vehicles.)
• $C_{ij}$: cost from customer $i$ to customer $j$
• $x_{ij}^k$: binary decision variable representing vehicle $k$ travel from customer $i$ to customer $j$
• $d_i$: demand of customer $i$
• $K$: vehicle capacity
• $S$: subset of customers (|$S$| is the number of customers in $S$.)

Given the definition of the binary decision variables, the objective function (1) is to minimize the total cost by combining the costs of the selected routes. Constraint (2) and Constraint (3) ensures that each customer is served by one of the vehicles. Constraint (4) is the flow conservation for each specific vehicle at each node (customer). Constraint (5) enforces the capacity limitation of the vehicles. Constraint (6) and Constraint (7) regulate that the number of vehicles used does not exceed the number of the vehicles available. Constraint (8) represents one of the ways to eliminate sub-tours. Finally, in Constraint (9), the decision variables $x_{ij}^k$ are required to be binary.

2.2 Classification of the VRP Algorithms

The solution algorithms for VRP can generally be classified as two broad categories: exact-solution algorithms and heuristics (Toth and Vigo, 2002). For the former, branch-and-bound and branch-and-cut are two important approaches. In addition, there have been several researches based on set covering formulation, which can be thought as third approach for exact VRP algorithms. Exact algorithms usually involve a high computational load and are applicable to small-scale problems, which generally has the number of customers less than 50 customers (Maffioli, 2003).

On the other hand, heuristics greatly alleviate the computational load and are more suitable for the problems with practical size, though the solution generated is not guaranteed to be optimal. There are two major stages for the development of VRP heuristics: the classical heuristics and the meta-heuristics. The former includes the constructive methods, the two-phase methods, and the related improvement techniques. Although the classical heuristics is fast and easy for implementation, the solution quality is generally much worse when compared to the meta-heuristics.

Several main types of meta-heuristics have been applied to VRP, such as Tabu Search, Simulated Annealing, Genetic Algorithms, and Ant Systems. In general, a meta-heuristic explores the solution space based on the techniques that can lead to a solution beyond local optimality, and it is common that some standard route construction and improvement methods are incorporated. The solution generated by meta-heuristics can be very close to optimal, but the complexity for implementation and the long computation time are the main concerns for meta-heuristics. For a summary of VRP meta-heuristics, Gendreau et al. (2002) serves as an excellent reference. As for the overall review of VRP related algorithms, the book edited by Toth and Vigo (2002) as well as some survey papers (such as Fisher, 1995 and Laporte et al., 2000) are great sources for more details.

Cordeau et al. (2002) highlights that there are four important issues while designing or comparing heuristics: accuracy, speed, simplicity and flexibility. Classical heuristics and meta-heuristics have their own advantages and disadvantages, though the latter appears to be much more promising given its recent dominating development. Nonetheless, in addition to these two broad categories, a heuristics based on the modification of the exact-solution algorithms can be an alternative to achieve a better balance among the different objectives. In
particular, if the flexibility needed for today’s dynamic and uncertain environment is the key concern, set covering approach can be a right direction to deal with this issue. Therefore, in order to design the core algorithm of the decision support system for the trucking companies or the operators of a large fleet, this study chose to develop a heuristic VRP algorithm based on set covering approach, which is discussed in the next section.

2.3 VRP Solution Algorithms Based on Set Covering Formulation

First suggested by Balinski and Quandt (1964), solving the VRP by set covering formulation is a classical method. The key concept is to enumerate all feasible routes, which start and end at the depot and have the overall load of the served customers less than the vehicle capacity. By treating a feasible route as a set, the mathematical programming model for the VRP by this approach is as follows:

\[
\text{Min} \sum_{r \in R} c_r x_r \quad (10)
\]
\[
\sum_{r \in R} a_{ir} x_r \geq 1 \quad \forall i \in I \quad (11)
\]

\[x_r : \text{Binary} \quad (12)\]

- \(i\): index for customers (\(I\) is set of all customers.)
- \(r\): index for routes (\(R\) is set of all routes.)
- \(c_r\): cost of route \(r\)
- \(a_{ir}\): binary constant; \(a_{ir}=1\) if customer \(i\) is in route \(r\), and \(a_{ir}=0\) if otherwise.
- \(x_r\): binary decision variable representing that route \(r\) is selected.

The objective function (10) is to minimize the total expense by combining the costs of the selected routes (sets). Constraint (11) ensures that each customer is incorporated in at least one of the selected routes. The greater-or-equal-to sign is used in (11), instead of the equal sign. The advantage of choosing the set covering problem rather than the set partitioning problem (SPP) is that the SCP is generally easier to solve than the SPP. Besides, if the distance matrix satisfies the triangle inequality, any feasible SPP solution is feasible to SCP. However, as it is possible that one customer can be included in multiple selected routes, extra steps must be taken to correct this situation. Finally, in Constraint (12), the binary variable \(x_r\) represents the selection of the routes.

Given the great flexibility imbedded in the SCP approach, there are at least two outstanding advantages for solving the VRP by above formulation (Toth and Vigo, 2002, p.11). First, it allows extremely general route costs. For example, the route cost may depend on the whole sequence of customers, instead of the sum of individual link costs. Second, any additional side constraint (e.g., time window and precedence requirement for pickup and delivery) can be taken into account implicitly while considering the feasibility of a single route. This level of flexibility can be valuable for various practical considerations and requirements. However, there is a serious limitation for this approach. That is, the number of feasible routes turns out to be huge even for a small-scale problem with a few customers. This implies that the set \(R\) in (10) actually may not be available, as it is impossible to enumerate all of the potential routes for most practical problems. Thus, several techniques have been developed to overcome this shortcoming.

Set covering approach has been used in several VRP algorithms. Especially for CVRP, Agarwal et al. (1989) developed an exact-solution algorithm. The general solution scheme for this type of approach can be summarized as the followings steps (Bramel and Simchi-Levi,
2002). First, the linear relaxation of the SCP is solved by column generation, which deals with all potential routes (columns) without explicitly enumerating them. Second, from the set of generated routes, which is usually a small fraction of all potential routes, a integer solution is derived by, for example, a cutting plane or branch-and-cut approach. The whole process is quite computationally intensive. Especially, the step to identify the column with the least reduced cost and generate the new route is the most time consuming.

In general, if an SCP is well defined (i.e., all of the sets and the associated set costs are given), it can be solved by all solution techniques for integer programming (IP), such as branch-and-bound used in Balas and Carrera (1996). In particular, Lagrangian relaxation, which maintains the integer property of the relaxed solution, has been proven to be very effective for SCP (for example, Beasley, 1990 and Caprara et al., 1999). Thus, this study has taken a hybrid approach to design the heuristic algorithm. The VRP is first modeled by the set covering formulation. Then, Lagrangian relaxation is used as the backbone in designing the iterative heuristics. A set of a limited number of feasible routes is adjusted through the iterative procedure based on a basic concept similar to column generation. In particular, the information derived from the Lagrangian procedure is re-used in adjusting the partial set of routes. The development of the solution algorithm is presented in the next chapter.

3. DEVELOPMENT OF SOLUTION ALGORITHM

To illustrate the development of the solution algorithm, several key design concepts are expanded upon, including the Lagrangian-relaxed solution, the realization of the feasible solution, and the adjustment of the solution space. At the end, the stopping criterion of the recursive algorithm is presented.

3.1 Determination of the Lagrangian Relaxed Solution

The above SCP (10) to (12) can be re-formulated, similar to the approach in Caprara et al. (1999), by relaxing Constraint (11) to derive the following Lagrangian relaxed problem:

\[ L(u) = \text{Min} \sum_{r \in R} c_r(u)x_r + \sum_{i \in I} u_i \]

\[ x_r : \text{Binary} \]

where \( c_r(u) = c_r - \sum_{i \in I_r} u_i \quad \forall r \in R \) \label{eq:lagrangian}

- \( u_i \): Lagrangian multiplier for customer \( i \), where \( u_i \geq 0 \quad \forall \ i; \ u \) is the column vector consisting of all Lagrangian multipliers.
- \( I_r \): set of customers covered by route \( r \), i.e., \( I_r = \{ i \in I : a_{ir} = 1 \} \).

In particular, the number of feasible routes in the set \( R \) becomes huge when the problem size is increased. One key idea of this study, similar to the concept of column generation, is to include only a limited number of routes. This partial set of potential routes, later referred to as the solution space, is adjusted in the recursive solution algorithm based on the information derived in each iteration.

This Lagrangian relaxed problem can be solved easily; that is, all the routes with negative \( c_r(u) \) are selected, i.e., \( x_r = 1 \) for \( c_r(u) \leq 0 \), and a lower bound \( L(u) \) is generated. To provide the information needed for the rest of the solution algorithm, this decision is implemented by sorting the values of \( c_r(u) \)'s for all routes in \( R \). The underlying reason is that, conceptually, the value of \( c_r(u) \) can be viewed as how well route \( r \) is. Based on (15), the value of \( c_r(u) \) is...
computed by subtracting the sum of the associated Lagrangian multipliers from the cost of route \( r \). It is worth noting that the Lagrangian multipliers are related to the dual variables. If they are interpreted as the price offered by an outside common carrier for serving a customer, the difference in (15) can be thought of as the cost saving for choosing a specific route. Therefore, the more negative the value is, the more desirable the route is. For the rest of the paper, \( c_r(u) \) may simply be referred to as the Lagrangian cost of route \( r \).

### 3.2 Realization of the Feasible Solution

As Constraint (11) is relaxed, the Lagrangian relaxed solution derived from the previous section is generally infeasible as some customers may not be covered by the selected routes. In addition, the desired solution is an SPP solution, instead of the SCP-type solution. Customers covered in multiple routes should be kept only in the most suitable route, and it is a challenge to identify such a route. In general, the feasible solution in a Lagrangian relaxation based method can be derived from the relaxed solution. In finding such a feasible solution, the degradation in the objective function value is supposed to be minimized. However, the procedure to obtain the feasible solution should not be too complicated so as not to result in a heavy computational load.

To find out the coverage of the customers in the relaxed solution, \( s_i(u) \) is computed as (16), which is also used as the direction to update the Lagrangian multipliers in (17). If \( s_i(u) \) happens to be 0, customer \( i \) is served exactly once. On the other hand, if the value of 1 implies that the customer is not covered by the selected routes. Similarly, the negative values indicate that the customer is covered more than once and must be adjusted so as to derive a SPP solution.

\[
s_i(u) = 1 - \sum_{r \in R_i} x_r(u) \quad \forall i \in I
\]  

- \( R_i \): set of routes covering customer \( i \)
- \( x_r(u) \): the Lagrangian relaxed solution of (13) to (15)

To generate the feasible SPP solution, the ranking list of routes by sorting the Lagrangian costs \( c_r(u) \)'s serves an excellent basis as the values is a good indicator regarding how good a route is made relatively. The procedure is summarized as the following steps:

1. Suppose the set \( U \) represents the routes contained in the final SPP solution. It is initially set as \( \phi \). Select one route from the top of the ranked list, i.e., the route with the most negative value of \( c_r(u) \).

2. Suppose the set of the customers in the currently considered route \( r \) is denoted by \( V \). If \( U \cap V = \phi \), let \( U = U \cup V \) and include the route \( r \) in the SPP solution. On the other hand, if \( U \cap V \neq \phi \), some customers already included in the solution also exist in the currently considered route \( r \). To avoid redundant coverage, the overlapped customers are removed from the route \( r \) to generate a revised route \( r' \). As it is not guaranteed that \( r' \) would be a nicely-arranged route, it is not included in the solution. Instead, the value of \( c_r(u) \) is calculated based on (15), and the revised route \( r' \) is re-inserted into the ranking list accordingly.

3. Check whether \( U \) is equal to \( N \), the set of all customers. If yes, terminate the process, and the feasible SPP solution is found. Otherwise, consider the next route in the ranked list and go to Step 2.
Once the SPP solution is derived, the value of the objective function can be computed based on (10). If it is better than the current upper bound (denoted by $B$), a better solution is derived and the upper bound is updated. Otherwise, $B$ remains unchanged.

### 3.3 Updating the Lagrangian Multipliers

The initial value of the Lagrangian multiplier is chosen as the distance between the depot and the customer. Through iterations, the value is generally become smaller and eventually stabilized. For each iteration, the objective function values of the currently best solution and the relaxed solution of this iteration, denoted by $B$ and $L(u)$, are used in updating the Lagrangian multipliers according to the following equation (Held and Karp, 1970).

$$
u^t_i = \max \left\{ u^t_i + \lambda \frac{B - L(u^t)}{\|s(u^t)\|^2} s_i(u^t), 0 \right\} \quad \forall i \in I$$

- $t$: index for iteration.
- $s(u)$: direction vector, where $s_i(u)$ is the $i$th element as computed in (16).
- $\lambda$: step size parameter

The direction for updating the Lagrangian multipliers is computed as in (16). Given this direction, the step size is determined as in (17) by taking into account the gap between the upper and the lower bounds as well as the magnitude of the direction vector and a pre-determined constant $\lambda$, which can also be adjusted dynamically based on the progress of the iterative procedure.

### 3.4 Adjustment of Solution Space

As mentioned earlier, the number of possible routes is huge even for a problem with a mild size. Therefore, instead of generating all the possible routes, only the routes consisting of one or two customers are generated initially, so no effort is required to solve the associated traveling salesman problems and the initial routes are guaranteed to be optimal for TSP.

This partial set of potential routes is adjusted in the recursive solution algorithm based on the information derived in each iteration. The adjustment procedure is designed to maintain a list of promising routes, which serves as a basis to produce a solution with good quality. The steps for removing the poor routes from the solution are described in the first sub-section. The surviving routes are used to generate new routes that are potentially well-arranged by two different approaches, 1) remove existing customers from and 2) insert new customers into the surviving routes. The procedures for these two approaches are described respectively in the second and third sub-sections. Finally, some remarks related to solution space adjustment are given.

#### 3.4.1 Removal of Non-promising Routes

As explained earlier, conceptually the Lagrangian cost $c_r(u)$ can be viewed as an indicator or a score of how desirable a route $r$ is. Through iterations, the score may change as the Lagrangian multipliers are updated and can be used as the yardstick to determine whether a route should be kept in the solution space. Only a limited number of routes (e.g., a pre-determined parameter $m$) with top-ranking scores survive for the next iteration, and the rest of the routes are removed from the solutions space to keep the computation tractable.
Lastly, one technical minor action is performed with respect to the surviving routes that have been modified during the process of finding the feasible SPP solution, explained in section 3.2. As the modified routes tend to be short and generally unlikely to be part of the optimal solution, they are replaced by the original routes before the overlapped customers are removed in Step 2 in the process.

### 3.4.2 Generation of Promising Routes - Removing Existing Customers

As the Lagrangian multiplier $u_i$ can be interpreted as the price offered by an outside common carriers or simply as a fair price to serve the customer $i$, it serves as an reasonable reference to judge how appropriate a customer is included in a route. The steps to identify the relatively unsuitable customers and to generate new routes are summarized as follows:

1. Suppose a number of $a$ (a pre-determined parameter, which can be set as, e.g., $m/5$) routes are selected randomly from the surviving routes as the targets for Step 2 and Step 3. In addition, let the set $W$ represents the customers to be considered in the insertion process described in the next sub-section. $W$ initially contains only the customers not covered in the relaxed solution, i.e., the customer $i$ with $s_i(u)=1$.

2. For each target route, estimate the extra cost for serving each customer $i$ by calculating $e_i = (C_{i-1,i}+C_{i,i+1}-C_{i-1,i+1})$, which is the distance from $i-1$ to $i+1$ by way of $i$ minus the direct distance from $i-1$ to $i+1$. If $e_i$ is greater than $u_i$, it is reasonable to believe that customer $i$ probably should not be included in the route. Among the customers of a route, remove the one with the largest $(e_i - u_i)$ from the route to make a new route and calculate its Lagrangian cost based on (15). If it is negative, incorporate the new route into the solution space and update $W$ to include this customer. Otherwise, discard the new route just generated.

3. For each target route, perform a check procedure similar to Step 2, but consider two consecutive customers of a route at the same time. Also, update $W$ to include the customers qualified to be removed.

### 3.4.3 Generation of Promising Routes - Inserting New Customers

To generate new routes by insertion, the removed customers in $W$ are considered. Besides, once again, the route list ranked according to the Lagrangian costs is used. It is believed it is more effective to insert the customers to the relatively well-arranged routes, which are on the top of the list. The steps are summarized as follow:

1. Suppose a number of $b$ (a pre-determined parameter, which can be set as, e.g., $m/5$) routes are selected from the ranked list as the targets for Step 2.

2. For each target route, randomly select $c$ (a pre-determined parameter, which can be set as, e.g., $m/10$) customers from $W$. For each selected customer, insert it into the route at the position with the least cost increase and calculate its Lagrangian cost based on (15). If it is negative, incorporate the new route into the solution space. Otherwise, discard the new route just generated.

### 3.4.4 Remarks about Solution Space Adjustment

Agarwal et al. (1989) proved that, for an optimal TSP solution, if a new customer is inserted at the position with the least cost increase, the new route remains to be optimal. Based on the procedure described in the previous two sub-sections, the generated new routes, either by removal or insertion, are optimal TSP routes, as the solution space initially contains only optimal TSP routes.
Of course, there exist many other possible ways to adjust the solution space, which plays an important role in the whole solution algorithm. Based on the numerical experiment presented in the next chapter, the procedure described in the previous sub-sections appears to be effective, and the computational effort is acceptable. Nonetheless, the setting of the parameters \( m, a, b, \) and \( c \) is an issue that may need further attention.

### 3.5 Stopping Criteria and Summary of the Solution Algorithm

The whole solution algorithm is summarized as the flowchart in Figure 1. The stopping criterion particularly needs to be further addressed. Generally speaking, the solution algorithm of a Lagrangian relaxation problem terminates if the upper bound from the feasible solution and the lower bound from the relaxed problem are identical. When this does not happen, the algorithm usually stops if it is found that the gap between the upper bound and the lower bound is within an acceptable value, which is usually relatively small.

---

1. Compute the initial Lagrangian multipliers
2. Generate the initial solution space

---

Solve Lagrangian relaxed problem (Sec. 3.1)
1. Sort Lagrangian costs, \( c_r(u) \)
2. Choose the routes with negative, \( c_r(u) \)
3. Compute the lower bound, \( L(u) \)

---

Determine SPP feasible solution and compute the upper bound (Sec. 3.2)

---

Update Lagrangian multipliers (Sec. 3.3)

---

Meet stopping criterion?

| Yes | No |
---|---|
| Stop | Select surviving routes and remove poor routes (Sub-Sec. 3.4.1) |

---

Generate promising routes
1. Remove customers from existing route (Sub-Sec. 3.4.2)
2. Insert customers to surviving routes (Sub-Sec. 3.4.3)

---

Select surviving routes and remove poor routes (Sub-Sec. 3.4.1)

---

Insert customers to surviving routes (Sub-Sec. 3.4.3)

---

Remove customers from existing route (Sub-Sec. 3.4.2)

---

Remove customers from existing route (Sub-Sec. 3.4.2)

---

Figure 1 Flowchart of Solution Algorithm
Based on the key ideas of the solution algorithm described in the previous sections, the lower bound is only valid for the SCP problem with respect to the solution space in each iteration, and it is not valid for the original vehicle routing problem. The reason for this is that the solution space only contains a partial set of possible routes. Thus, the stopping criterion must be adjusted even though the upper bound remains valid. This study chooses to stop the iterative solution algorithm based on the number of iterations performed, similar to the criterion used in some Tabu Search algorithms, such as Taillard (1993). Generally speaking, the number of iterations required is increased slightly for a larger problem.

4. NUMERICAL EXPERIMENT

The numerical experiment was performed based on the well-known test problems of Solomon (2006). In particular, among the CVRP instances available on the website, those from Christofides and Eilon as well as from Set A and Set B of Augerat et al. are chosen. They are solved by the heuristic algorithm described in the previous chapter, and the resulted objective function value is compared with the optimal solution or the best known solution provided by Solomon (2006). The machine used to perform the experiment is with CPU AMD Athlon XP 2000+ 1.67GHz and 1.5GB RAM.

For the instances from Christofides and Eilon, the problem size ranges from 20 to 100 customers, and the locations and demands of the customers were randomly generated when the instances were designed. Besides, the capacities are different among the vehicles. The optimal solution has been found for some instances. Eight problems were tested in the experiment. The average gap between the solution derived by the heuristics algorithm and that of the best known or optimal solution is about 1.52%. In terms of computation time, the average is about 240 seconds, and it increases roughly in a linear way when the problem size increases. The details of the results are summarized in Table 1.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Best known solution, Solomon</th>
<th>Result from this study</th>
<th>No. of trucks</th>
<th>Gap (%)</th>
<th>CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>n22-k4</td>
<td>375*</td>
<td>375*</td>
<td>4</td>
<td>0</td>
<td>74</td>
</tr>
<tr>
<td>n23-k3</td>
<td>569*</td>
<td>569*</td>
<td>3</td>
<td>0</td>
<td>104</td>
</tr>
<tr>
<td>n33-k4</td>
<td>835*</td>
<td>837</td>
<td>4</td>
<td>0.24</td>
<td>155</td>
</tr>
<tr>
<td>n51-k5</td>
<td>521*</td>
<td>524</td>
<td>5</td>
<td>0.58</td>
<td>185</td>
</tr>
<tr>
<td>n76-k7</td>
<td>682*</td>
<td>700</td>
<td>7</td>
<td>2.64</td>
<td>291</td>
</tr>
<tr>
<td>n76-k8</td>
<td>735</td>
<td>742</td>
<td>8</td>
<td>0.95</td>
<td>285</td>
</tr>
<tr>
<td>n101-k8</td>
<td>815</td>
<td>848</td>
<td>8</td>
<td>4.05</td>
<td>540</td>
</tr>
<tr>
<td>n101-k14</td>
<td>1071</td>
<td>1111</td>
<td>14</td>
<td>3.73</td>
<td>403</td>
</tr>
</tbody>
</table>

(* represents optimal solution)

For the instances from Set A of Augerat et al., the problem size ranges from 30 to 80 customers, and the locations and demands of the customers were also randomly generated when the instances were designed. However, the capacity is set as 100 for all vehicles. The optimal solution has been found for all 27 tested problems. For the gaps between the heuristic solution and the optimal solution, the mean and the standard deviation are 2.16% and 0.012 respectively, and the worst one is 4.27%. The average computation time is about 220 seconds. The details of the results are summarized in Table 2.
Table 2 Test results of the CVRP instances from Set A of Augerat et al.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Best known solution, Solomon</th>
<th>Result from this study</th>
<th>No. of trucks</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>n32-k5</td>
<td>784*</td>
<td>787</td>
<td>5</td>
<td>0.38</td>
</tr>
<tr>
<td>n33-k5</td>
<td>661*</td>
<td>668</td>
<td>5</td>
<td>1.06</td>
</tr>
<tr>
<td>n33-k6</td>
<td>742*</td>
<td>744</td>
<td>6</td>
<td>0.27</td>
</tr>
<tr>
<td>n34-k5</td>
<td>778*</td>
<td>790</td>
<td>5</td>
<td>1.54</td>
</tr>
<tr>
<td>n36-k5</td>
<td>799*</td>
<td>809</td>
<td>5</td>
<td>1.25</td>
</tr>
<tr>
<td>n37-k5</td>
<td>669*</td>
<td>682</td>
<td>5</td>
<td>1.94</td>
</tr>
<tr>
<td>n37-k6</td>
<td>949*</td>
<td>950</td>
<td>6</td>
<td>0.11</td>
</tr>
<tr>
<td>n38-k5</td>
<td>730*</td>
<td>737</td>
<td>5</td>
<td>0.96</td>
</tr>
<tr>
<td>n39-k5</td>
<td>822*</td>
<td>830</td>
<td>5</td>
<td>0.97</td>
</tr>
<tr>
<td>n39-k6</td>
<td>831*</td>
<td>854</td>
<td>6</td>
<td>2.77</td>
</tr>
<tr>
<td>n44-k6</td>
<td>937*</td>
<td>949</td>
<td>6</td>
<td>1.28</td>
</tr>
<tr>
<td>n45-k6</td>
<td>944*</td>
<td>957</td>
<td>7</td>
<td>1.38</td>
</tr>
<tr>
<td>n45-k7</td>
<td>1146*</td>
<td>1181</td>
<td>7</td>
<td>3.05</td>
</tr>
<tr>
<td>n46-k7</td>
<td>914*</td>
<td>936</td>
<td>7</td>
<td>2.41</td>
</tr>
<tr>
<td>n48-k8</td>
<td>1073*</td>
<td>1096</td>
<td>7</td>
<td>2.14</td>
</tr>
<tr>
<td>n53-k7</td>
<td>1010*</td>
<td>1033</td>
<td>7</td>
<td>2.28</td>
</tr>
<tr>
<td>n54-k7</td>
<td>1167*</td>
<td>1209</td>
<td>7</td>
<td>3.60</td>
</tr>
<tr>
<td>n55-k9</td>
<td>1073*</td>
<td>1098</td>
<td>9</td>
<td>2.33</td>
</tr>
<tr>
<td>n60-k9</td>
<td>1354*</td>
<td>1375</td>
<td>9</td>
<td>1.55</td>
</tr>
<tr>
<td>n61-k9</td>
<td>1034*</td>
<td>1055</td>
<td>9</td>
<td>2.03</td>
</tr>
<tr>
<td>n62-k8</td>
<td>1288*</td>
<td>1328</td>
<td>8</td>
<td>3.11</td>
</tr>
<tr>
<td>n63-k9</td>
<td>1616*</td>
<td>1685</td>
<td>9</td>
<td>4.27</td>
</tr>
<tr>
<td>n63-k10</td>
<td>1314*</td>
<td>1360</td>
<td>10</td>
<td>3.50</td>
</tr>
<tr>
<td>n64-k9</td>
<td>1401*</td>
<td>1458</td>
<td>9</td>
<td>4.07</td>
</tr>
<tr>
<td>n65-k9</td>
<td>1174*</td>
<td>1218</td>
<td>9</td>
<td>3.75</td>
</tr>
<tr>
<td>n69-k9</td>
<td>1159*</td>
<td>1189</td>
<td>9</td>
<td>2.59</td>
</tr>
<tr>
<td>n80-k10</td>
<td>1763*</td>
<td>1828</td>
<td>10</td>
<td>3.69</td>
</tr>
</tbody>
</table>

(* represents optimal solution)

As for the instances from Set B of Augerat et al., the locations of the customers are clustered when the instances were designed. For the gaps between the heuristic solution and the optimal solution, the mean and the deviation are 2.28% and 0.011 respectively, and the worst one is 4.57%. The average computation time is about 200 seconds. The details of the results are summarized in Table 3.

Based on the result of the numerical experiment, the solution quality of the heuristic algorithm is quite stable, though it is a little degraded for larger problems. The solution quality can be raised, if the number of surviving routes in the removal procedure (described in the sub-section 3.4.1) is increased to allow more routes contained in the solution space. However, the computation time can be significantly increased. For practical operation, the solution quality and the computation time of this solution algorithm should be acceptable, though there is room for further improvement.
Table 3 Test results of the CVRP instances from Set B of Augerat et al.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Best known solution, Solomon</th>
<th>Result from this study</th>
<th>No. of trucks</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>n31-k5</td>
<td>672*</td>
<td>688</td>
<td>5</td>
<td>2.38</td>
</tr>
<tr>
<td>n34-k5</td>
<td>788*</td>
<td>791</td>
<td>5</td>
<td>0.38</td>
</tr>
<tr>
<td>n35-k5</td>
<td>955*</td>
<td>975</td>
<td>5</td>
<td>2.09</td>
</tr>
<tr>
<td>n38-k6</td>
<td>805*</td>
<td>809</td>
<td>6</td>
<td>0.50</td>
</tr>
<tr>
<td>n39-k5</td>
<td>549*</td>
<td>555</td>
<td>5</td>
<td>1.09</td>
</tr>
<tr>
<td>n41-k6</td>
<td>829*</td>
<td>855</td>
<td>6</td>
<td>3.14</td>
</tr>
<tr>
<td>n43-k6</td>
<td>742*</td>
<td>759</td>
<td>6</td>
<td>2.29</td>
</tr>
<tr>
<td>n44-k7</td>
<td>909*</td>
<td>945</td>
<td>7</td>
<td>3.96</td>
</tr>
<tr>
<td>n45-k5</td>
<td>751*</td>
<td>766</td>
<td>6</td>
<td>2.00</td>
</tr>
<tr>
<td>n45-k6</td>
<td>678*</td>
<td>709</td>
<td>6</td>
<td>4.57</td>
</tr>
<tr>
<td>n50-k7</td>
<td>741*</td>
<td>759</td>
<td>7</td>
<td>2.43</td>
</tr>
<tr>
<td>n50-k8</td>
<td>1312*</td>
<td>1336</td>
<td>8</td>
<td>1.83</td>
</tr>
<tr>
<td>n51-k7</td>
<td>1032*</td>
<td>1039</td>
<td>8</td>
<td>0.68</td>
</tr>
<tr>
<td>n52-k7</td>
<td>747*</td>
<td>760</td>
<td>7</td>
<td>1.74</td>
</tr>
<tr>
<td>n56-k7</td>
<td>707*</td>
<td>726</td>
<td>7</td>
<td>2.69</td>
</tr>
<tr>
<td>n57-k7</td>
<td>1153*</td>
<td>1175</td>
<td>8</td>
<td>1.91</td>
</tr>
<tr>
<td>n57-k9</td>
<td>1598*</td>
<td>1628</td>
<td>9</td>
<td>1.88</td>
</tr>
<tr>
<td>n63-k10</td>
<td>1496*</td>
<td>1542</td>
<td>10</td>
<td>3.07</td>
</tr>
<tr>
<td>n64-k9</td>
<td>861*</td>
<td>876</td>
<td>9</td>
<td>1.74</td>
</tr>
<tr>
<td>n66-k9</td>
<td>1316*</td>
<td>1356</td>
<td>9</td>
<td>3.04</td>
</tr>
<tr>
<td>n67-k10</td>
<td>1032*</td>
<td>1053</td>
<td>10</td>
<td>2.03</td>
</tr>
<tr>
<td>n68-k9</td>
<td>1272*</td>
<td>1320</td>
<td>9</td>
<td>3.77</td>
</tr>
<tr>
<td>n78-k10</td>
<td>1221*</td>
<td>1260</td>
<td>10</td>
<td>X.19</td>
</tr>
</tbody>
</table>

(* represents optimal solution)

5. CONCLUSIONS

The vehicle routing problem has drawn significant attention from the researchers. Especially, the meta-heuristics has achieved a success in solving various kinds of vehicle routing problems. However, owing to the inherited complexity, it is still hard to solve large-scale problems within a short period of time. In addition, it remains difficult to take into consideration the individual requirements from all kinds of special situations.

The goal of this research is to develop a suitable algorithm as the core module of the decision support system for the trucking companies or the operators of a large fleet. The VRP is first transformed to a well-known set covering problem by treating a feasible route as a set. Lagrange Relaxation, a successful approach for SCP reported in prior researches, is used as the backbone to develop an iterative heuristic algorithm. As the number of feasible routes is huge, the solution space consisting of a limited number of feasible routes is carefully adjusted through the iterative solution procedure. The study performs the numerical experiment based on the widely-distributed test problems, and the solution quality and the computation time is acceptable.

In general, the solution algorithm of this study has the advantages from the aspects of computational load and flexibility. Based on the numerical experiment, only a small number
of routes need to be maintained in the solution space, given the almost infinite number of feasible routes. Besides, as the problem size increases, the size of the solution space only needs to be expanded in a linear fashion. On the other hand, as the routing decision is modeled simply as a set, all kinds of constraints can easily be examined and conformed while creating the new sets in the procedure of solution space adjustment. Thus, the considerations and limitations arising from real operating environment can be incorporated into the solution approach without difficulty. Given these promising features, the solution algorithm developed in the study should have the potential to be used in developing a decision support system for practical purposes. The following issues are the directions for research extension:

- A lower bound should be developed to modify the termination criterion of the iterative algorithm and to ensure the strength of the solution algorithm for large-scale problems.
- The method to find the feasible SPP solution from the infeasible solution of the relaxed SCP problem can be re-considered as the associated computation can be too heavy.
- The procedure for solution space adjustment should be further improved as it is critical for reaching a good quality solution. In particular, the concepts from various kinds of meta-heuristics can be imported to generate the new routes in the solution space.
- The whole solution procedure may be refined with regards to parameter setting. For example, the parameters related to the solution space adjustment (m, a, b, and c) should be tuned based on more numerical experiments.
- The efficiency of the heuristic algorithm can be improved by advanced coding techniques.
- Given the flexibility of the SCP formulation, the solution algorithm can take into account all kinds of operational requirement and consideration. The solution approach can be modified for the application to other kinds of vehicle routing problems.

REFERENCES