The Vehicle Routing Problem with Skill Sets

Ramesh Krishnamurti, Ehsan Iranmanesh, Wen Sun

January 31, 2012
Abstract

In this report we discuss the Vehicle Routing Problem with Skill Sets. We first provide an Integer Programming formulation for the problem, which we solve optimally using CPLEX for small instances. We then present a local search algorithm to solve larger instances.
Contents

1 Introduction 2
2 The Model 3
3 Methods 5
  3.1 Optimal Solution ......................... 5
  3.2 Local Search ........................... 5
4 Data 7
5 Experimental Results 8
  Bibliography 10
Chapter 1

Introduction

The Vehicle Routing Problem (VRP) is a classic optimization problem in operations research. Given a set of vehicles, the problem involves finding a route for each vehicle such that each route begins and ends at a depot, and the set of routes obtained to service a set of customers. VRP is an important problem in the fields of transportation, distribution, and logistics [1]. The objective is usually to minimize the total distance traveled by the set of vehicles, but other criteria, such as minimizing the number of vehicles used to service the set of customers, could also be considered. One important generalization of the VRP, the Vehicle Routing Problem with Time Windows (VRPTW), has received much attention in the literature. In VRPTW, each customer has a time interval, and the service for each customer has to start within this interval. In this report, we consider another generalization of VRP, the Vehicle Routing Problem with Skill Sets (VPRSS). In this problem, each vehicle (or worker) has a set of skills to service a customer and each customer has a given skill requirement. A vehicle can service a customer if the customer's skill requirement belongs to the skill set of that vehicle. To the best of our knowledge, this is the first time this problem has received attention in the literature.
Chapter 2

The Model

The VRPSS is defined by a set of vehicles $V$, a set of customers $C$, and a directed graph $G$. The graph consists of $|C|+1$ vertices, where the customers are denoted $1, 2, ..., n$ and the depot is represented by the Vertex 0. The set of all vertices $0, 1, ..., n$ is denoted $N$. The set of arcs $A$ represents links between the depot and the customers and between the customers. We associate cost $c_{ij}$ with each arc $(i, j), i \neq j$. Each vehicle $i$ has a set of skills $S_i$ and each customer $j$ has a skill requirement $s_j$. Vehicle $i$ can service customer $j$ if $s_j \in S_i$.

It is assumed that the $c_{ij}$'s are non-negative integers and the triangle inequality is satisfied (for a set of three customers $i, j, k$, $c_{ij} + c_{jk} \geq c_{ik}$). The model contains a set of decision variables $x$ and a set of continuous variables $u$. For each arc $(i, j), i \neq j$, and each vehicle $k$:

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \text{ drives directly from vertex } i \text{ to vertex } j \\ 0 & \text{otherwise} \end{cases}$$

The goal is to design a set of routes that minimizes total cost, such that

- each customer is serviced exactly once
- every route originates and ends at Vertex 0 and
- the skill requirement of each customer is in the skill set of each vehicle (worker) that services the customer.
\[
\min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \quad (2.1)
\]

subject to
\[
\sum_{k \in V} \sum_{j \in N, s_j \in S_k} x_{0jk} = |V| \quad (2.2)
\]
\[
\sum_{k \in V} \sum_{i \in N, s_i \in S_k} x_{ijk} = 1 \quad \forall j \in N \quad (2.3)
\]
\[
\sum_{i \in N, s_i \in S_k} x_{ijk} - \sum_{i \in N, s_i \in S_k} x_{jik} = 0 \quad \forall k \in V, \forall j \in N : s_j \in S_k \quad (2.4)
\]
\[
u_i - u_j + n \sum_{k \in V} x_{ijk} \leq n - 1 \quad \forall i, j \in N : s_i, s_j \in S_k \quad (2.5)
\]
\[
x_{ijk} \in \{0, 1\} \quad \forall k \in V, \forall i, j \in N \quad (2.6)
\]
\[u_i \geq 0 \quad \forall i \in N \quad (2.7)
\]

The objective function 2.1 minimizes the total travel cost. Constraint 2.2 ensures that exactly \(|V|\) vehicles depart from and return back to the depot. Constraints 2.3 and 2.6 are the usual assignment constraints and ensure that each customer is visited exactly once. Constraint 2.4 ensures that all the customers of a route must be serviced by the same vehicle. Constraint 2.5 is the subtour elimination constraint, eliminating tours that do not begin and end at the depot as well as tours that visit more than \(n\) cities [4]. Constraint 2.7 specifies non-negativity for the continuous variables \(u\).
Chapter 3

Methods

3.1 Optimal Solution

CPLEX, a standard optimization tool for solving linear and integer programming problems, is used to solve the VRPSS formulation optimally. Since this problem is computationally intractable, we can only obtain the optimal solution for problem instances that comprise at most 150 customers and 15 vehicles. For larger instances, even finding a feasible integer solution is extremely time consuming. We outline below a local search algorithm for VRPSS, that may be used to solve large instances of VRPSS.

3.2 Local Search

Let $s = (s_1, s_2, \ldots, s_v)$ be a feasible solution. The neighborhood $N(s)$ of feasible solution $s$ is a set of solutions obtained from $s$ by applying specific operations. We define the two operations \textit{intra-route} and \textit{2-opt$^*$} to obtain a neighborhood.

The intra-route operation removes four paths of length one from a route and inserts them into other positions of the same route [2]. The 2-opt$^*$ operation is a variant of the 2-opt operation for the traveling salesman problem [3]. A 2-opt$^*$ operation removes two paths of length one from two different routes (one from each) and then exchanges these paths if it is possible (if the workers of these two routes can service the customers of the exchanged paths).

The neighborhoods $N^{\text{intra}}(s)$ and $N^{2-\text{opt}^*}(s)$ are the set of all feasible solutions obtainable by the intra-route and 2-opt$^*$ operations on feasible solution $s$. The LS starts from an initial solution $s$ and repeatedly replaces $s$ with a better solution $s'$ in its neighborhood (using either $N^{\text{intra}}(s)$ or
$N^{2-\text{opt}\ast}(s)$) until no better solution is found. The pseudocode for LS is given below:

**Algorithm 1 Local Search**

while there is an improvement do
    if there is a feasible solution $s' \in N^{\text{intra}}(s)$ such that $\text{cost}(s') < \text{cost}(s)$ then
        let $s = s'$
    else
        break
    end if
end while

while there is an improvement do
    if there is a feasible solution $s' \in N^{2-\text{opt}\ast}(s)$ such that $\text{cost}(s') < \text{cost}(s)$ then
        let $s = s'$
    else
        break
    end if
end while
Chapter 4

Data

In this section, we describe the dataset used to evaluate LS. Since no benchmark data exists for VRPSS, we generate a text file with the following information for each problem instance:

- The number of jobs and vehicles
- The skill requirement for each job
- The skill set for each vehicle
- The distance matrix

The first line contains the number of jobs, followed by the number of vehicles. The second line provides the skill requirement, a number randomly chosen between 1 and 10, for each job. This is followed by $|V|$ lines, one for each vehicle, which contains a randomly chosen subset of $\{1, 2, \ldots, 10\}$, denoting the skill set assigned to each vehicle. This is followed by the distance matrix, specifying the euclidian distance between pairs of jobs and the distance between each job and the depot. Thus the next line contains the first row of this matrix, specifying the distance between the depot and each job. The following line contains the second row, specifying the distance between Job 1 and the depot, followed by the distance between Job 1 and the other jobs. This continues until the last line contains the last row of the distance matrix.
Chapter 5

Experimental Results

We conduct several experiments to evaluate the efficiency of LS. We summarize the results in the following tables. Table 5.1 compares the solution obtained by LS with the optimal solution. Instances of size up to 100 jobs and 10 vehicles can be solved optimally using CPLEX. The table provides a comparison of LS and the optimal, both in the quality of their solutions, as well as the running times required, with increasing sizes of instances.

<table>
<thead>
<tr>
<th>#jobs(#vehicles)</th>
<th>LS solution</th>
<th>LS time</th>
<th>Optimal solution</th>
<th>Time to compute optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>30-3</td>
<td>264</td>
<td>1 sec</td>
<td>239</td>
<td>1 sec</td>
</tr>
<tr>
<td>40-4</td>
<td>410</td>
<td>1 sec</td>
<td>321</td>
<td>1 sec</td>
</tr>
<tr>
<td>50-5</td>
<td>353</td>
<td>1 sec</td>
<td>265</td>
<td>5 min</td>
</tr>
<tr>
<td>70-7</td>
<td>583</td>
<td>1 sec</td>
<td>411</td>
<td>5 min</td>
</tr>
<tr>
<td>90-9</td>
<td>570</td>
<td>1 sec</td>
<td>450</td>
<td>1 hour</td>
</tr>
<tr>
<td>100-10</td>
<td>682</td>
<td>2 sec</td>
<td>472</td>
<td>1.5 hour</td>
</tr>
</tbody>
</table>

Table 5.1: Comparing LS solution with the optimal solution

For larger instances, Table 5.2 provides a comparison of LS with a Linear Programming (LP) bound on the optimal solution value.
<table>
<thead>
<tr>
<th>#jobs(#vehicles)</th>
<th>LS solution</th>
<th>LS time</th>
<th>LP solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>300-30</td>
<td>1605</td>
<td>1 min</td>
<td>655</td>
</tr>
<tr>
<td>300-30</td>
<td>1561</td>
<td>1 min</td>
<td>676</td>
</tr>
<tr>
<td>300-30</td>
<td>1715</td>
<td>1 min</td>
<td>779</td>
</tr>
<tr>
<td>300-30</td>
<td>1500</td>
<td>1 min</td>
<td>624</td>
</tr>
<tr>
<td>300-30</td>
<td>1554</td>
<td>1 min</td>
<td>632</td>
</tr>
<tr>
<td>400-40</td>
<td>1825</td>
<td>1 min</td>
<td>770</td>
</tr>
<tr>
<td>400-40</td>
<td>1855</td>
<td>1 min</td>
<td>830</td>
</tr>
<tr>
<td>400-40</td>
<td>2000</td>
<td>1 min</td>
<td>808</td>
</tr>
<tr>
<td>400-40</td>
<td>1865</td>
<td>1 min</td>
<td>785</td>
</tr>
<tr>
<td>400-40</td>
<td>1987</td>
<td>1 min</td>
<td>870</td>
</tr>
</tbody>
</table>

Table 5.2: Comparing LS solution with the LP solution
Bibliography


