Software Framework for Vehicle Routing Problem with Hybrid Metaheuristic Algorithms

S.MASROM¹, A.M. NASIR²
Malaysia Institute of Transport (MITRANS)
Faculty of Computer and Mathematical Science
Universiti Teknologi MARA, PERAK, MALAYSIA
¹suray078@perak.uitm.edu.my
²arfah485@perak.uitm.edu.my

SITI.Z.Z.ABIDIN
Faculty of Computer and Mathematical Science
Universiti Teknologi MARA, MALAYSIA
sitizaleha533@salam.uitm.edu.my

A.S.A. RAHMAN
Computer and Information Science Department
Universiti Teknologi PETRONAS, MALAYSIA
sanirahman@petronas.com.my

Abstract: - The objective of vehicle routing problem (VRP) is to design a set of vehicle routes in which a fixed fleet of delivery vehicles from one or several depots to a number of customers have to be set with some constraints. To this date in the literature, many instances of VRP model have been introduced and applied for various types of scheduling problems. However, when implemented in a real life application, the VRP models proved to be very complex and time consuming, especially in the development phase. It is due to the fact that there are technical hurdles to overcome such as the steep learning curve, the diversity and complexity of the algorithms. This paper presents a generalize software framework for an effective development of VRP models. The software framework presented here is hybridized algorithm of two metaheuristics namely as Particle Swarm Optimization (PSO) and Genetic Algorithm (GA). The hybrid algorithm is used to optimize the best route for the vehicles that also incorporates a mechanism to trigger swarm condition for PSO algorithm. In order to test the functionality of the software framework, the applications of Pickup and Delivery Problem with Time Windows (PDPTW) and Vehicle Routing Problem with Time Windows (VRPTW) are developed based on the software framework. Experiments have been carried out by running the hybrid PSO with the VRPTW and PDPTW benchmark data set. The results indicate that the algorithm is able to produce significant improvement mostly to the PDPTW.

Key-Words: - Software framework, Vehicle Routing Problem, Particle Swarm Optimization, Genetic Algorithm, Mutation, Hybridization.

1 Introduction

The vehicle routing problem (VRP) [1] is a popular and well-studied combinatorial optimization problem. It is an intensive research area because of its usefulness to the logistics and transportation industry. Real world instances of the VRP are more complicated which may involve a thousand of customer locations and many complicated constraints. The necessity for fast and good optimal solution has directed many researchers to look seriously into metaheuristic based technique. By using metaheuristic, remarkable results can be achieved [2-3]. Some examples of metaheuristic algorithms are Genetic Algorithm (GA)[4], Ant Colony Optimization (ACO)[5], Particle Swarm Optimization (PSO)[6], Tabu search[7] and Simulated Annealing[8].

In the development of metaheuristic algorithms, there are some difficulties involved in the design and implementations. In order to shorten the development life cycle for metaheuristics based application, researchers may rely on open source software packages for metaheuristic algorithms. However, based on the literature, it is found that the available software packages are not originally designed for VRP based models. Modification and amendments are required to make the software packages relevant to the specific VRP models. Another difficult task is to design and implement...
appropriate solution representation for VRP with regard to the specific metaheuristics data structure. These processes also require in-depth understanding on the programming language, inner and working structure of the software packages.

In addition, many of the available software packages were originally designed for only a single metaheuristic algorithm. In many cases of complex problem such as VRP, hybridization of metaheuristic is sometime necessary. Metaheuristic hybridization is an improvement algorithm process to the original metaheuristic algorithm. It is proven by various empirical experiments that metaheuristic hybridization techniques are very promising in providing fast and high optimal solutions [9-11].

With the limitation issues in software packages, this research is initiated with the focus on various aspects of the design and implementation for VRP models and its metaheuristics hybridization. The selected metaheuristic algorithms are PSO and GA. These two algorithms have been widely applied for various kinds of VRP problems [12-14].

The remaining content of this paper is organized as follows; Section 2 presents the related works on metaheuristics and available software packages for VRP. Section 3 describes the proposed software framework including its architecture and hybridization technique. Then, the computational results for two instances of VRP models are given in Section 4. Lastly, Section 5 concludes the paper with a short summary.

2 Related works
To this date of literature, a lot of research works have been undertaken on solving VRP models by using metaheuristic algorithms. The implementation complexity in metaheuristic algorithms and its hybridization has directed researchers’ attention to use available software packages. However, the number of software framework for VRP application with metaheuristic algorithm is very limited.

2.1 Metaheuristics for VRP
In recent years, the numbers of works that implementing VRP based application with metaheuristic algorithm is increasing. A variant of VRP problems have been successfully solved by using GA [11, 15-16] , ACO [9, 17-18] , Tabu search [15, 19-20] and also with PSO[12, 21].

Based on the literature, it can be seen that most of the metaheuristics techniques on VRP were concentrated on metaheuristic hybridization rather than the single algorithm implementation [9-11]. The metaheuristic hybridizations have shown significant improvements to the VRP optimal solutions. As for example, a combination of PSO with multiple phase neighborhood search and greedy randomized adaptive search (MPNS-GRASP) has been effectively solved VRP problem[14]. The purpose of this hybridization is for expanding the neighborhood strategy in PSO. In another work, PSO has also been hybridized with ACO for performance improvement [13]. This technique is proven to be effective when it is used in the implementation of VRP for grain logistic. A method that combines PSO with local search and simulated annealing is another technique for PSO hybridization. The computational experiments have showed that the proposed algorithm is feasible and effective for capacitated vehicle routing problem, especially for large scale problems [10].

2.2 Software framework for metaheuristics and VRP
The development of programming code for optimization problem like VRP and its metaheuristic algorithm from scratch has been considered as complex [22-23]. The process will have to go through huge works especially to those with no fundamental knowledge on computer programming concept and languages such as C, C++, C# and JAVA.

Recently, several software packages have been developed with concern on reducing the development complexity and phase cycle. Researchers may use the freely available and widely used Concorde package[24]. It is an exact solver for the travelling salesman problem (TSP), the most related model to VRP. Another type for TSP is Lin-Kernighan-Helsgaun(LKH) package developed by Hesgaun [25]. However, this software package is not supported by metaheuristic algorithms.

Some of the available software packages with metaheuristic algorithms are METASIS [26], EasyLocal++ [27], ParadisEO [28] and HeuristicLab [29]. All of these software packages are not originally designed for VRP models. Moreover, many of them were developed for single metaheuristic implementation.

3 The Software Framework
The software framework has been developed by using JAVA programming language. Generally, the architecture is composed of front and back end components. The front end provides user interface for application configurations while all the software engines are located at the back end. Fig. 1 illustrates the architecture of the software framework.
3.1 Front end components
The front end provides interfaces for experiment configurations including problem, algorithm and database setting. In the problem configuration, users can choose the predefined VRP models such as Pickup and Delivery VRP with Time Windows (PDPTW), VRP with time windows (VRPTW), VRP with split delivery and multiple depots VRP.

The hybridization technique can be defined through algorithm configuration components. An important configuration is to select operators from the GA to be incorporated into the PSO algorithm. Other configurations consist of basic PSO parameters such as coefficient acceleration (c1 and c2), population size, particle dimension, iteration time, position range and velocity range.

In order to be connected to a specific database of a particular VRP application, database interface is provided for files browsing purposes. The interface is also designed to facilitate data entry and manipulation of process on vehicles and customers information.

3.2 Back end components
At the back end, three major components have been developed for the software framework. These components are problem formulation, hybrid algorithm and database engines.

3.2.1 Problem engine
All the predefined classes for VRP models and its variant are located in the problem formulation engine. In this component, the general PSO solution representation for all VRP has been defined according to the method introduced by Ai & Kachitvichyanukul [30]. Fig. 2 shows the solution representation.

In this method, every particle’s position in PSO is divided into \((n+2m)\) dimension and each dimension is encoded as a real number. The first \(n\) dimensions represent the number of customers and the last \(2m\) dimensions are related to the number of vehicle with its reference point in Cartesian map. A number of two dimensions are dedicated for each vehicle as a reference point.

Developing a set of customer routes for vehicles is an important procedure in problem formulation engine. The procedure is different for every VRP model. The sum of total cost for all routes with a set of constraints will be passed over to the algorithm engine.

3.2.2 Hybrid engine
Hybrid engine is the component for optimization process. It uses PSO as the master algorithm and GA as the subordinate algorithm. The GA provides several operators to be used by PSO including selection, crossover and mutation operators. Selection of operators can be done at the front end component.

In every population generation of PSO, each velocity of particle has to move towards the best fitness based on the information of previous experience which consists of previous velocity, personal best position of each particle, and global best position of the swarm. Afterward, the particle position will be adjusted according to the latest velocity.

Before advancing to the next evaluation process, the condition of particle will be examined and determined. The idea behind this process is adapted from the PSO Triggered Mutation algorithm introduced by You & Ying [31]. A particle is considered healthy if it has
found a new personal best position at the current iteration. The number of particles with personal best improvement will be identified from the swarm population. When a certain percentage of particles in the swarm population are not healthy, the population condition can be considered as unhealthy and not strong. In this condition, the particles have fallen into local optima and stagnated.

Every time of iteration, the occurrence of unhealthy condition is counted. If the counted number is bigger than a certain number of iterations, the swarm is ready for mutation process. The purpose of mutation operation is to change slightly the value of particles’ positions so that it will increase the possibility of many potential solutions to be explored. A variety type of mutation operations have been proposed for PSO algorithm[32]. However, the operations are not really suitable and not specific for VRPTW solutions. In VRPTW solutions, the customer priority position is hardly dependent on the initialization position. So, mutating at random particle position will change the customer priority arrangement. It will cause to result inferiority to solutions. So, in the hybrid PSO (HPSO), the mutation is done on the vehicle location or reference point. The formula for mutation operation is

\[ X_{v, id}^{i} = X_{v, id}^{i} + \text{gaussian}(p) \]  

(1)

The \( \text{gaussian}(p) \) function returns a random number drawn from a Gaussian distribution with a standard deviation of \( p \). The value of \( p \) is 0.1 times the length of dynamic range of the particle dimension [33]. This is to ensure that the vehicle reference point for each vehicle is within the search vector.

3.2.3 Database engine
The database engine is a database management system (DBMS) used to create, read, update and delete records. It is included with application programming interface (API) that allows user to interact with its underlying engine. In the software framework, MySQL has been used as database engine and the database is composed of vehicle, depot and customer records.

4 Experiments and Results
Computational experiments have been conducted to evaluate the performance of the Hybrid PSO (HPSO) for solving two different kinds of VRPTW [34] and PDPTW [19] that have been generated from the software framework. The experiments run on a personal computer with Intel Pentium 1.20 GHz-3GBRAM and processing under Windows 7. A total of 20 runs for each case are conducted and an average result is obtained. The algorithm configuration is listed in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of particle</td>
<td>100</td>
</tr>
<tr>
<td>Number of iteration</td>
<td>1000</td>
</tr>
<tr>
<td>Inertia weight, ( w )</td>
<td>Linear decreasing from 0.9 to 0.4</td>
</tr>
<tr>
<td>( X_{\text{min}} )</td>
<td>0</td>
</tr>
<tr>
<td>( X_{\text{max}}, V_{\text{max}} )</td>
<td>100</td>
</tr>
<tr>
<td>Personal and global acceleration</td>
<td>2</td>
</tr>
<tr>
<td>Healthy percentage</td>
<td>60%</td>
</tr>
<tr>
<td>Unhealthy occurrence</td>
<td>3</td>
</tr>
<tr>
<td>mutation rate</td>
<td>0.2</td>
</tr>
</tbody>
</table>

4.1 Benchmark problem instance
The Solomon’s [34] and Li & Lim [19] benchmark data set are used respectively for the VRPTW and PDPTW applications. Both Solomon and Li & Lim data is comprised of variety problem situations such as customer location, vehicles information, customers’ demand and corresponding time windows. However, in Li & Lim dataset, it is included with pickup and delivery data series. Table 2 lists the criteria for all data sets. The C1, C2, R1 and RC are all Solomon data sets while Li & Lim is represented with LC1, LC2, LR1 and LRC1. The short horizon is to represent less than 10 routes and long horizon used to describe more than 30 routes. All instances are tested on 100 number of customers.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Distribution</th>
<th>Horizon</th>
<th>Vehicle capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1, LC1</td>
<td>Clustered</td>
<td>Short</td>
<td>200</td>
</tr>
<tr>
<td>C2, LC2</td>
<td>Clustered</td>
<td>Long</td>
<td>700</td>
</tr>
<tr>
<td>R1, LR1</td>
<td>Random</td>
<td>Short</td>
<td>200</td>
</tr>
<tr>
<td>RC1, LRC1</td>
<td>Random clustered</td>
<td>Short</td>
<td>200</td>
</tr>
</tbody>
</table>

4.2 Performance metrics
The results are measured primarily on three performance metrics namely as number of vehicles (NV), best distance and computational time. These values are compared with the best known solutions from Solomon [34] and Li & Lim[19].

4.3 Results
Based on the results in Table 3, the HPSO algorithm used in the software framework is capable to produce shorter distance than the Solomon best distance. The improvements can be seen in all cases of VRPTW, extremely to the random (R)
distribution (more than 60%). However, the HPSO algorithm could not reduce the number of vehicle used in all cases. The figures are similar with Solomon’s vehicle numbers.

Table 3: Computational results for VRPTW

<table>
<thead>
<tr>
<th>Prob.</th>
<th>NV</th>
<th>HPSO</th>
<th>Best</th>
<th>Best known</th>
<th>Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C101</td>
<td>10</td>
<td>820.1</td>
<td>10</td>
<td>827.3</td>
<td>0.15</td>
</tr>
<tr>
<td>C102</td>
<td>10</td>
<td>820.2</td>
<td>10</td>
<td>827.3</td>
<td>0.18</td>
</tr>
<tr>
<td>C201</td>
<td>3</td>
<td>553.6</td>
<td>3</td>
<td>589.1</td>
<td>0.2</td>
</tr>
<tr>
<td>C202</td>
<td>3</td>
<td>574.3</td>
<td>3</td>
<td>589.1</td>
<td>0.21</td>
</tr>
<tr>
<td>R101</td>
<td>20</td>
<td>1035.8</td>
<td>20</td>
<td>1637.7</td>
<td>0.25</td>
</tr>
<tr>
<td>R102</td>
<td>18</td>
<td>918.5</td>
<td>18</td>
<td>1466.6</td>
<td>0.28</td>
</tr>
<tr>
<td>RC 101</td>
<td>15</td>
<td>1518.5</td>
<td>15</td>
<td>1619.8</td>
<td>0.27</td>
</tr>
<tr>
<td>RC 102</td>
<td>14</td>
<td>1175.3</td>
<td>14</td>
<td>1457.4</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Table 4 has showed that all problem instances of PDPTW have gained some improvement in the optimum solutions (distance) by using HPSO. Similar to VRPTW, an apparent achievement can be seen from the random distribution (R1) customers. But, the achievement is slightly lower than VRPTW instances with only about 50% improvement. On the contrary, most of the random and random clustered instances in PDPTW have a smaller number of vehicles when using HPSO.

Table 4: Computational results for PDPTW

<table>
<thead>
<tr>
<th>Prob.</th>
<th>NV</th>
<th>HPSO</th>
<th>Best</th>
<th>Best known</th>
<th>Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LC101</td>
<td>9</td>
<td>820.3</td>
<td>10</td>
<td>828.94</td>
<td>0.21</td>
</tr>
<tr>
<td>LC102</td>
<td>9</td>
<td>820.6</td>
<td>10</td>
<td>828.94</td>
<td>0.25</td>
</tr>
<tr>
<td>LC201</td>
<td>3</td>
<td>552.7</td>
<td>3</td>
<td>591.56</td>
<td>0.28</td>
</tr>
<tr>
<td>LC202</td>
<td>3</td>
<td>574.1</td>
<td>3</td>
<td>591.56</td>
<td>0.27</td>
</tr>
<tr>
<td>LR101</td>
<td>18</td>
<td>764.3</td>
<td>17</td>
<td>1487.57</td>
<td>0.28</td>
</tr>
<tr>
<td>LR102</td>
<td>16</td>
<td>1517.3</td>
<td>14</td>
<td>1708.80</td>
<td>0.27</td>
</tr>
<tr>
<td>LRC101</td>
<td>14</td>
<td>1172.8</td>
<td>13</td>
<td>1563.55</td>
<td>0.32</td>
</tr>
<tr>
<td>LRC102</td>
<td>12</td>
<td>1172.8</td>
<td>13</td>
<td>1563.55</td>
<td>0.32</td>
</tr>
</tbody>
</table>

As shown in Table 5, the computational time for implementing HPSO in both VRPTW and PDPTW instances can be slightly longer than the original PSO. However, the average time taken for PDPRW instances is a little bit shorter than the VRPTW instances with 0.75 different.

Table 5: Computational time (CT) in second

<table>
<thead>
<tr>
<th>VRPTW</th>
<th>PSO</th>
<th>HPSO</th>
<th>PDPTW</th>
<th>PSO</th>
<th>HPSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>C101</td>
<td>125</td>
<td>140</td>
<td>LC101</td>
<td>119</td>
<td>136</td>
</tr>
<tr>
<td>C102</td>
<td>125</td>
<td>140</td>
<td>LC102</td>
<td>120</td>
<td>138</td>
</tr>
<tr>
<td>C201</td>
<td>138</td>
<td>145</td>
<td>LC201</td>
<td>137</td>
<td>140</td>
</tr>
<tr>
<td>C202</td>
<td>137</td>
<td>143</td>
<td>LC202</td>
<td>138</td>
<td>140</td>
</tr>
<tr>
<td>R101</td>
<td>124</td>
<td>138</td>
<td>LR101</td>
<td>125</td>
<td>139</td>
</tr>
<tr>
<td>R101</td>
<td>122</td>
<td>138</td>
<td>LR102</td>
<td>126</td>
<td>140</td>
</tr>
<tr>
<td>RC 101</td>
<td>123</td>
<td>137</td>
<td>LRC101</td>
<td>119</td>
<td>140</td>
</tr>
<tr>
<td>RC 102</td>
<td>125</td>
<td>137</td>
<td>LRC102</td>
<td>152</td>
<td>139</td>
</tr>
</tbody>
</table>

5 Conclusions
This paper presents a software framework that offers effective implementation of VRP instances by using hybrid PSO. The developments of VRPTW and PDPTW have demonstrated the capability of the software framework. From the empirical results, it can be revealed that the hybrid method from the software framework is proved to be more effective than the best known solutions of benchmark problems. In some of the PDPTW instances, the numbers of vehicles are smaller than other methods. This translates into cost reduction in the vehicle maintenance as well as drivers involvement.

Acknowledgements
The authors would like to thank Kementerian Pengajian Tinggi Malaysia and Universiti Teknologi MARA for their financial support to this project.

References:


