A New GA Approach for the Vehicle Routing Problem *

Ang Juay Chin, Ho Wee Kit, Andrew Lim
School of Computing
National University of Singapore
Building S-16 Level 5 Room 05/08
3 Science Drive 2, Singapore 117543
{*angjc, howeekit, alim*}*@comp.nus.edu.sg*

Abstract

This paper focuses on the study of a hybrid of two search heuristics, Tabu Search (TS) and Genetic Algorithm (GA) on Vehicle Routing Problem with Time-Windows (VRPTW). TS is a local search technique that has been successfully applied to many NP-complete problems. On the other hand, GA which is capable of searching multiple search areas in a seach space is good in diversification. We shall investigate whether a hybrid of the two heuristic outperforms the individual heuristics alone.

1 Introduction

Vehicle routing problems are well known combinatorial optimization problems. There are many variants to vehicle routing problems [6]. One such variant is the Vehicle Routing Problem with Time-Windows (VRPTW).

A VRPTW consists of a set of customers that are to be serviced by a set of capacitated vehicles while minimizing the cost of service to the customers. The cost of service is measured by the number of vehicles required to service all customers and the total distance travelled by the vehicles. Each customer has a time window, comprising of an earliest release time, after which the customer will accept service and a latest time, after which service to the customer is consider tardy. A service time is also associated with each customer for the servicing vehicle.

2 Related Works

For the past decade, much research work has been done on vehicle routing problems. For instance, adaptations of the Tabu Search meta-heuristic for real-life problems with many side constraints including time windows have been successfully implemented by Rochat and Semet [10]. [5] [7] have investigated the VRPTW using Genetic Algorithms(GAs). A GA called GIDEON based on cluster-first route-second was developed. An adaptive clustering method using geometric shape for solving VRPTW was described in [4] and [8]. A GA heuristic was reported in [3] which was based on a hybrid combination of GA and a greedy constructive heuristic.

*This research was supported in part by the NUS Research Grant RP3972679.

Besides working on the problem-solving approaches, some heuristics operators were also devised in the course of research. The traditional crossover operator for order-based GAs such as Edge Recombination have been well studied and are described in [2]. [1] examined genetic operators for solutions coded as sequences in an attempt to improve the traditional crossover operators. The Intersection and Union crossover operator are two examples. [3] also presented two crossover operator, Merge Cross #1 and Merge Cross #2. The Merge Cross operators are based upon the notion of a global precedence among genes independent of any chromosome.

3 Tabu Search (TS)

Tabu Search is based on neighbourhood search with local-optimal avoidance, but in a deterministic way which tries to model human memory processes. Tabu Search algorithm is very similar to a neighbourhood search algorithm. The main difference is the maintenance of the Tabu list, and the verification on whether a move is taboo-ed. In every iteration, the algorithm polls a list of Tabu Operators and chooses the one that gives the best move to update the solution. Thereafter, the chosen move is taboo-ed but will be freed after a predetermined number of iterations. This process terminates when it reaches a predetermined number of non-improving iterations.

4 Genetic Algorithm (GA)

GA is an adaptive heuristic search method based on population genetics. GA consists of a population of chromosomes. Each chromosome has a fitness value associated with it. These chromsomes will evolve over a number of generations and are subjected to genetic operations at each generation. This process iterates for many generations until the termination criteria are met. In this paper, two newly devised genetic operators,namely Alteration and Extended Intersection operators are presented.

4.1 Alteration Operator

In genetic theory, the Principle of Dominance states that one of the parent genes is dominant over the other in most cases. As a result, only the dominant gene will be expressed physically when combined with a recessive gene. Using this same principle, a genetic operator is devised to make use of these dominant and recessive characteristics. However, this theory is applied on chromosomes. Moreover, instead of remain docile, the recessive chromosome will attempt to alter the dominant chromosome to its own traits, especially the good ones. Hopefully, after performing such an operation, the children will display the good traits of both parents.

The operator starts by making the child solution a duplicate of the dominant parent. Next we traverse the recessive parent. For each traversal, we have a current customer (CURR) and a next customer (NEXT). Then we shall find the corresponding position of the CURR and NEXT in the child solution. After finding the two positions, say CHILD_CURR and CHILD_NEXT, we will attempt to insert CHILD_NEXT after CHILD_CURR. If the insertion is not feasible, the CHILD_CURR will traverse to its next customer and try to insert CHILD_NEXT again. This search continues until an insertion is feasible and inexpensive or the CHILD_CURR has reached the depot. After the first insertion, the CURR and NEXT of the recessive parent are traversed to their next nodes. This continues until all the routes of the recessive parent have been traversed.

4.2 Extended Intersection Operator

This operator was motivated by the principle that characteristics that are common to two good parent solutions should be passed on to their children. In addition, characteristics that appear in one of the parents and not the other should be placed at a higher priority when compared to a characteristic that is absent in both parents.

In VRPTW context, there are basically four such characteristics. In order to extract such characteristics, the Intersection Operator described in [1] is used. The Intersection Operator suggested, however, can only be used to extract the common precedence characteristics. This method is extended in order to represent all the four characteristics in a single matrix.

The first step is to form a precedence matrix for each parent solution. After obtaining two precedence matrices (described in [1]) from the parents, the two matrices are joined together in the following manner. Given two parent precedenc e matrices, PM1 and PM2, the value of every entry of the joint matrix JM are determined as follows:

```
A // Both customers share the same vehicle and
  // have same precedence in both solutions
  If (PM1[X,Y] = 1 and PM2[X,Y] = 1)
  then      JM[X,Y] := A1
  else if (PM1[Y,X] = 1 and PM2[Y,X] = 1)
  then      JM[X,Y] := A2

B // Both customers share the same vehicle but
  // have different precedence
  else if (PM1[X,Y] = 1 and PM2[Y,X] = 1) or
  (PM1[Y,X] = 1 and PM2[X,Y] = 1)
  then      JM[X,Y] := B

C // In one solution, both customers share same vehicle
  else if (PM1[X,Y] = 1) or (PM1[Y,X] = 1) or
  (PM2[X,Y] = 1) or (PM2[Y,X] = 1)
  then      JM[X,Y] := C

D // Both customers have no vehicle in common
  else JM[X,Y] := D
```

The value “A1” denotes that customer Y precedes customer X in both parents. Similarly “A2” denotes that customer X precedes customer Y. This information is useful when we need to create the child solutions.

After obtaining the joint matrix, the next step is to use the information in the matrix to generate children solutions. The PFIH is modified to use the joint matrix to generate children solutions. Instead of selecting the first customer with furthest distance from depot, we select the customer that has the most entries set either a “A1”, “A2”, “B” or “C” in the joint matrix. The rational is that in doing so, we could use this customer to attract his related customers to the current route. In order to attract such related customers, we reduce the insertion cost by a given weight. For each case, a weight is assigned with the highest weight placed on “A” and the lowest weight on “D”. In addition, when trying to attract related customers, we also attract the related customer who is least in common with other customers so that it is less likely to affect the routing of other routes. So, the insertion cost is increased by an amount proportionate to the number of entries set at “A1”, “A2”,“B” or “C”.

5 Hybrid Approach

Both TS and GA possess different capabilities of search. TS is good in performing local search while on the other hand, GA is capable of diversifying its search to different regions in the search space. Nevertheless, in many situations, a simple GA or TS alone does not perform particularly well. Here, a method of hybridization of the above two heuristics is proposed.

In this hybrid approach, we recognize that GA performs better if it is allowed to diversify its search ”freely”. In consideration of this, no TS is performed in the early generations. The population will gradually become fitter and more refined after many generations. Eventually, the population will show signs of converging. We will then apply TS on the individual solutions in the later generations. In addition, the TS will gradually becomes more intense as the population converges further, as illustrated in Figure 1. This allows the converging process to be more gradual.

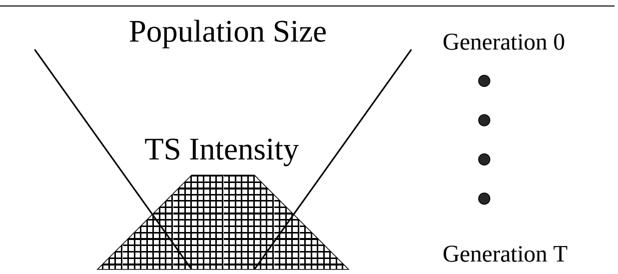

Figure 1: TS Increasing Intensification in Hybrid Approach

The search intensity of TS is varied based on the number of non-improving moves before TS termination. This intensity value may be determined by the following equation:

$$NIM = \frac{1}{M_c} \times \lambda. \quad (1)$$

where NIM is the number of non-improving moves, M_c is the current population size and λ is the maximum number of non-improving moves for a given TS operation.

When the population size of the current generation is large, a small or no TS is performed. As the population converges, a larger TS is performed for every individual in the population.

6 Computational Results

In our experiments, the Soloman's test set is used for testing the performance on various heuristics. There are altogether 56 test instances. Each test instance has 100 customers. These problems vary in fleet size, vehicle capacity, travel time of vehicles, spatial distribution of customers and width of time window.

A simple weightless scheme is used so that given two solutions, it is possible to distinguish which solution is better without knowing the weights used. The simple measure used is as follows:

$$C(S) = TD(S) \times TV(S) \quad (2)$$

where $C(S), TD(S)$ and $TV(S)$ is the cost incurred, total distance travelled and number of vehicle used by solution S respectively.

6.1 Result

Various heuristics such as the Push-Forward Insertion Heuristic with Tabu Search post-optimization (PFIH+TS), the Genetic Algorithm (GA) and the Hybrid Method (Hybrid2) were put through the Soloman's test. The TS parameters values for the number of non-improving moves and tabu list size are set as 1000 and 20 respectively. For the GA, the initial population size is set as 1000. Besides the mentioned new operators, we have also used a mutation operator at a rate of 10%. The GA will terminate when either the population size converges to a size of 2 or less, or the number of generations reaches 1000. The parameters used in Hybrid2 are similar to that of TS and GA except that the number of non-improving moves are calculated dynamically with $\lambda = 50$.

We first compare the performance of PFIH+TS, GA and Hybrid2 with respect to GIDEON system [5] which is also based on Genetic Algorithm. GIDEON consists mainly of two steps. It first uses a global customers clustering method to assign vehicles to customers. Then the best solution obtained from the clustering method is improved by a local post-optimization method.

To compare our heuristics with that of GIDEON, we determine the percentage savings in cost by a given heuristic with respect to GIDEON. The percentage savings in cost for test instance i using heuristic H is calculated as follows:

$$ExtraCost_{H,i}(\%) = \frac{Cost(GIDEON)_i - Cost(H)_i}{Cost(GIDEON)_i} \times 100 \quad (3)$$

where $Cost(H)_i$ is the cost calculated for test instance i using heuristic H

Figure 2 compares different heuristics namely, PFIH+TS, GA(1000) and Hybrid2(1000) against GIDEON. We observed that PFIH+TS is moderately comparable to that of GIDEON. We also observed that our GA has obtained better results than GIDEON in 47 out of the 56 test instances. The average percentage of the cost savings is about 8%. In addition, Hybrid2 performs even better as it pushes the cost savings further to about 11%. Hybrid2 has obtained 51 results better than GIDEON. From these observations, we notice that our GA approach is comparatively better than the GA used in GIDEON and the hybrid method appears to perform relatively better than GA alone.

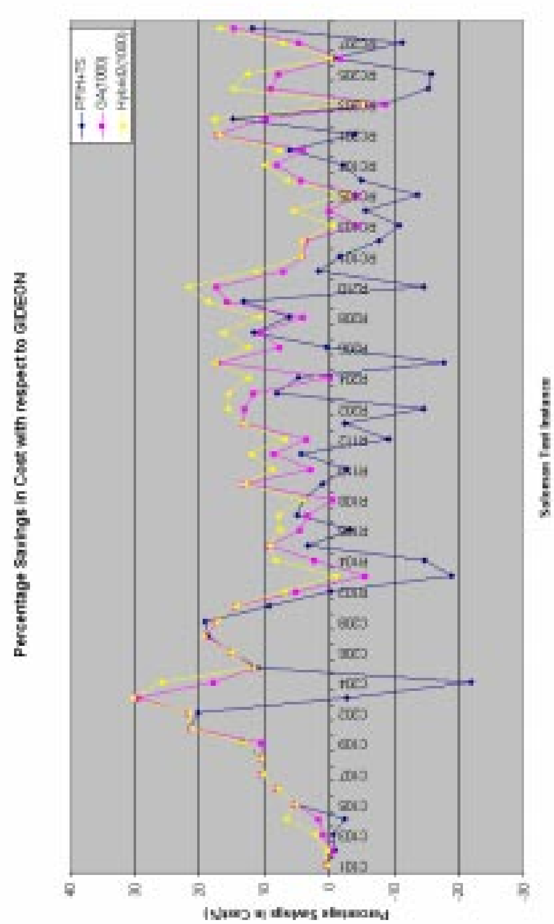

Figure 2: Comparison of Various Heuristics with respect to GIDEON

In the next comparison, we would like to determine the effectiveness of our hybrid approach. Since we claimed that our hybrid approach performs both diversification and intensification, one would ask why not randomly generate a set of initial solutions and performs Tabu Search for each of the solutions. In this way, we too diversify as well as intensify our search. Again, we take GIDEON as our baseline and determine the percentage of cost savings for TS(50) and Hybrid2(1000) against GIDEON. Based on the relative difference of the cost savings, the heuristic with a higher cost savings is deemed to be better than the other. This comparison is summarised in Figure 3.

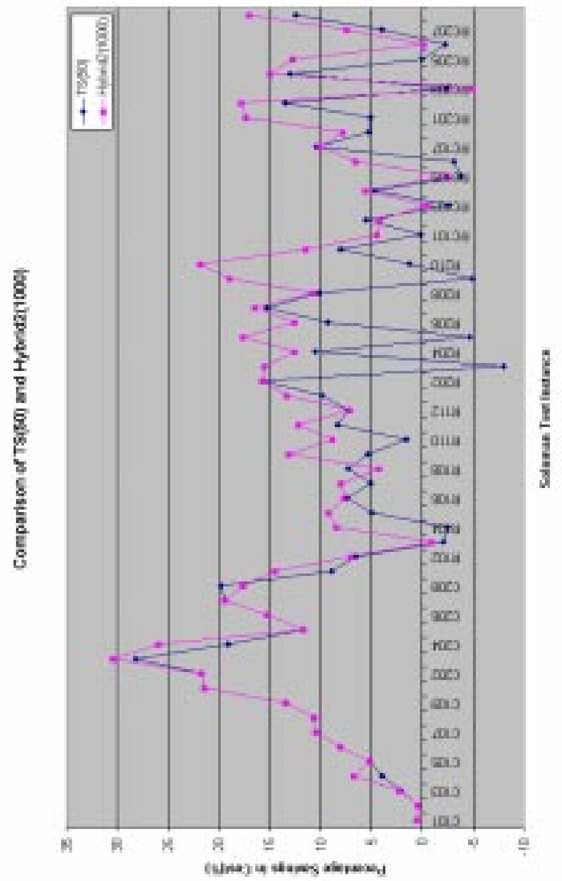

Figure 3: Comparison of TS(50) and Hybrid2(1000)

In this figure, we observed that TS(50) and Hybrid2 are comparable for test sets C1 and C2. However, their performance differs significantly in other test sets. In particular, the cost savings of TS(50) varies greatly on different test instances. However, the performance of Hybrid2 is relatively more stable than TS(50). From the graph, the Hybrid2's line generally lies above the TS(50)'s. The average percentage cost savings for TS(50) is approximately 8% as compared to the 11% for Hybrid2.

The above observations suggest that Hybrid2 is better than TS(50). There are a few possible reasons to this. Firstly, the performance of TS(50) is very much dependent on the initial set of solutions. If all the initial solutions are near the less desirable regions, it may then perform badly. Hybrid2 reduces such dependency with the use of GA. The GA is capable of diversifying its search even if all the initial solutions are bad. Moreover, the generation of a solution in Hybrid2 is not solely based on one solution. Instead, it attempts to find some common good properties from two parent solutions for generating new and possibly better child solutions. In this sense, Hybrid2 is deemed to be a better strategy.

7 Conclusion

We have devised a hybrid approach of integrating both TS and GA such that both their respective intensifying and diversifying process are exploited and integrated. In our comparisons, the results suggests that an hybrid approach can perform better than its individual heuristics alone.

References

[1] B R Fox and M B McMahon. Genetic Operators for Sequencing Problems. *Foundations of Genetic Algorithms*, 1:284–300, 1991.

[2] D Whitley and T StarkWeather. GENTOR II : a distributed genetic algorithm. *Journal of Experimental and Theoretical Artificial Intelligence*, 2:189–214, 1990.

[3] J L Blanton and R L Wainwright. Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1:452–459, 1993.

[4] S R Thangiah. An Adaptive Clustering using a Geometric Shape for Vehicle Routing Problems with Time Windows. *Proceedings of the Sixth International Conference on Genetic Algorithms*, 1:536–543, 1995.

[5] S R Thangiah. Vehicle Routing with Time Windows using Genetic Algorithms. *Application Handbook of Genetic Algorithms:New Frontier*, 2:253–277, 1995.

[6] S R Thangiah. Introduction to Genetic Heuristics and Vehicle Routing Problems with Complex Constraints. *Advances in Computational And Stochastic Optimization, Logic Programming, and Heuristic Search:Interfaces in Computer Science and Operations Research*, 1:253–286, 1998.

[7] S R Thangiah, I H Osman and T Sun. Metaheuristics for Vehicle Routing Problems with Time Windows. *Technical Report, CpSc-TR-94-27, Artificial Intelligence and Robotics Laboratory, Computer Science Department, Slippery Rock University, Slippery Rock, PA 16057*, 1, 1994.

[8] S R Thangiah, I H Osman, R Vinayagamoorthy and T Sun. Algorithms For The Vehicle Routing With Time Deadlines. *American Journal of Mathematical and Management Sciences, 13(3&4)*, 1:323–355, 1993.

[9] Y Rochat and E D Tailard. Probabilistic Diversification And Intensification In Local Search For Vehicle Routing. *Journal of Heuristics*, 1:147–167, 1995.

[10] Y Rochat and F Semet. A Tabu Search Approach for Delivering Pet Food and Flour in Switzerland. *Journal of the Operational Research Society*, 1:1233–1246, 1994.