Cogent Business & Management

ISSN: (Print) 2331-1975 (Online) Journal homepage: www.tandfonline.com/journals/oabm20

Enhancing heuristic bubble algorithm with simulated annealing

Mehmet Fatih Yuce, Erhan Musaoglu & Ali Gunes |

To cite this article: Mehmet Fatih Yuce, Erhan Musaoglu & Ali Gunes | (2016) Enhancing heuristic bubble algorithm with simulated annealing, Cogent Business & Management, 3:1, 1220662, DOI: 10.1080/23311975.2016.1220662

To link to this article: https://doi.org/10.1080/23311975.2016.1220662

Yuce et al., *Cogent Business & Management* (2016), 3: 1220662
http://dx.doi.org/10.1080/23311975.2016.1220662

cogent business & management

OPERATIONS, INFORMATION & TECHNOLOGY | RESEARCH ARTICLE

Enhancing heuristic bubble algorithm with simulated annealing

Mehmet Fatih Yuce[1], Erhan Musaoglu[1] and Ali Gunes[2*]

Received: 18 May 2016
Accepted: 01 August 2016
Published: 12 August 2016

*Corresponding author: Ali Gunes, Department of Computer Engineering, Istanbul Aydin University, Kucukcekmece, Istanbul, Turkey
E-mail: aligunes@aydin.edu.tr

Reviewing editor:
Shaofeng Liu, University of Plymouth, UK

Additional information is available at the end of the article

Abstract: In this study, a new way to improve the Heuristic Bubble Algorithm (HBA) is presented. HBA is a nature-inspired algorithm, which is a new approach to and initially implemented for, vehicle routing problems of pickup and delivery (VRPPD). Later, it was reinforced to solve other routing problems, such as vehicle routing problem with time windows (VRPTW), and vehicle routing problem with stochastic demands (VRPSD). HBA is a greedy algorithm. It will mostly find local optimal solutions. The proposed method is an improvement over HBA enabling it to reach the global minimum. It uses specialized simulated annealing methods in its operators. A well-known data-set is used to benchmark the proposed method. Better results over HBA and some best results in literature are recorded.

Subjects: Operational Research/Management Science; Operations Management; Road Transport Industries; Shipping Industries; Supply Chain Management

Keywords: logistics; optimization; VRPPD; VRPTW; simulated annealing; supply chain

1. Introduction

Vehicle routing problem (VRP), or "Truck Dispatching Problem", is a deterministic polynomial-time hard (NP-hard) problem and can be formulated as a generalization of the "Travelling Salesman Problem" (TSP). TSP, although its beginning is unclear (Hoffman et al., 1986), was first publicized by Flood (1955).

Ali Gunes

ABOUT THE AUTHOR

Ali Gunes is a professor of Computer Engineering Department of Istanbul Aydin University. He is also the Head of Distance Education Research and Application Centre at Istanbul Aydin University. He currently teaches, consults and conducts research on computer software models and software development, application of information technologies on different sectors and new learning technologies in computer education. His minor is distance education, e-learning and transportation of educational media on different platforms. He has been studying computer software modelling and development, integrated education and technological developments on education for more than 30 years in the academic field. He has written more than 40 articles and more than 10 books and other academic works, on computer software, integrated software development and e-learning published in several journals and publishing companies.

PUBLIC INTEREST STATEMENT

The importance of transportation services for businesses is very great. Nowadays, the issue of transportation of goods and services contains many problems that enterprises need to resolve. In this regard two most important problems of transportation that can be done timely and at optimal cost. One of the most important issues in determining the cost of transportation is to allow goods to be made via the shortest possible route. Many studies have been done and several models and algorithms have been proposed to solve this problem. In this study, a new way to improve the Heuristic Bubble Algorithm (HBA) is presented. HBA is a new approach to and initially implemented for, vehicle routing problems of pickup and delivery. The proposed method is an improvement over HBA. A well-known data-set is used to benchmark the proposed method. Better results over HBA and some best results in literature are recorded.

The problem is formulated as such:

- G: a Hamiltonian path or a complete graph,
- V: set of vertices,
- E: set of edges,
- c_{ij}: cost linked with each element inside E,

  is given. c_{ij} is the cost to traverse from $k \in V$ to $m \in V$ (Zambito, 2006).

  VRP, on the other hand, is first presented by Dantzig and Ramser (1959) and is formulated as:
- R: a set of routes,
- V: that is served by a fleet of vehicles,
- P: that is visiting a set of customers,
- C: with some constraints.

(Kumar & Panneerselvam, 2012). Thus, each of the routes inside a VRP solution becomes a TSP instance.

VRPs could be classified according to its taxonomy or its framework model (Granada, 2016). In this regard, following is an incomplete list of VRP variants;

- Two-Echelon VRP (2E-VRP) (Crainic, Perboli, Mancini, & Tadei, 2010),
- Asymmetric capacitated VRP (ACVRP) (Vigo, 1996),
- Arc Routing Problem (ARP) (Eiselt, Gendreau, & Laporte, 1995),
- The Capacitated VRP (CVRP) (Ralphs, Kopman, Pulleyblank, & Trotter, 2003),
- Dial-a-ride Problem (DARP) (Cordeau & Laporte, 2003),
- The Emissions VRP (EVRP) (Jemai, Zekri, & Mellouli, 2012),
- Generalized VRP (GVRP) (Baldacci, Bartolini, & Laporte, 2010),
- Location Routing Problem (LRP) (Nagy & Salhi, 2007),
- Distance-Constrained VRP (DCVRP) (Laporte, Desrochers, & Nobert, 1984),
- VRP with Backhauls (VRPB) (Crispim & Brandão, 2005),
- VRP with Time Windows (VRPTW) (Bräysy & Gendreau, 2005),
- VRP with Pickup and Delivery (VRPPD) (Çatay, 2010).

The Heuristic Bubble Algorithm (HBA) was presented previously in Sakalli, Yesil, Musaoglu, Ozturk, and Dodurka (2013), while the Enhanced HBA (E-HBA) was introduced by Savran, Yuce, and Yesil (2014) and Savran, Musaoglu, Yildiz, Yuce, and Yesil (2015).

Simulated Annealing (SA) has been proved to solve VRPPD problems (Czech & Czarnas, 2002). In this study we extended E-HBA with SA and applied it to the entire schedule resulting from a single run of the HBA algorithm.

Section 2 of this paper discusses SA as it is applied to other problems. Section 3 describes the E-HBA. Section 4 presents the way SA is applied to E-HBA, as well as the results from Gehring and Homberger's (1999) data-sets. Section 5 is a computational study and Sections 6 and 7 end with sample results, conclusions and future directions.

2. Simulated annealing

SA was first formulated by Černý (1985) and Kirkpatrick, Gelatt, and Vecchi (1983), as a re-engineering of a Monte-Carlo Method (Metropolis–Hastings algorithm) (Metropolis, Rosenbluth, Rosenbluth, & Teller, 1953). In Kirkpatrick et al. (1983), interestingly, it was first implemented to solve a TSP problem using a probability provided by the Boltzmann–Gibbs distribution.

SA is a local search method capable of escaping the local minimum by taking a worse solution according to a probability (Lin, Yu, & Chou, 2009). It has been applied successfully to complicated combinatorial optimization problems (Abramson, 1991; Jayaraman & Ross, 2003; McKendall, Shang, & Kuppusamy, 2006).

The terminology is taken from metallurgical engineering. Annealing is the process of slow-cooling metals to produce better aligned crystallization. SA uses a similar method for getting close to the global minima, by slowly decreasing the heat. In each run, SA accepts a new best solution from the neighbouring solutions. If the solution is better, it is accepted as the new best solution. If it is worse, then according to a probability it may be accepted to continue with. By doing so, SA tries to escape from the local minima. This new solution becomes the starting point of the next run.

The probability calculations come from Boltzmann–Gibbs distribution (Černý, 1985; Kirkpatrick et al., 1983) which is expressed as

$$e^{-\frac{E}{k_b T}} \tag{1}$$

where E is the energy of the state, k_b is the Boltzmann constant and T is the heat. The ratio of two states in the distribution is called Boltzmann Factor and is given as

$$e^{-\frac{E_1 - E_2}{k_b T}} \tag{2}$$

which in turn simply can be expressed by

$$e^{-\frac{\Delta}{k_b T}} \tag{3}$$

where Δ is the change in energy.

3. Enhanced heuristic bubble algorithm

HBA is an iterative heuristic inspired by the division and union of water bubbles (Sakalli et al., 2013). Bubbles are the goods that are carried between distribution centres. During transportation, goods are combined and separated from each other to better serve scheduling needs. In each run, HBA tries to find the bubble that, according to the objective function, is the best route in the entire route set which can be constructed using the remaining orders. This structure is best suited for pickup and delivery problems.

HBA was enhanced with Split, Result Elimination (RE) and Swap Operators (Savran et al., 2014), hence the new name "Enhanced HBA" (E-HBA) Each addition provides a better enhancement over initial bubbles.

Next we will briefly describe the Merge Operator, as it is fundamental in understanding E-HBA. Because we have integrated SA as a RE operator, we will give a short explanation on this operator.

3.1. Merge Operator

The Merge Operator is the core of the HBA. Together with the Join Operator, it is the initial implementation of the algorithm. It searches the entire solution space for a single route and, according to the criteria presented to it, finds the best one. The length of the route is the main input provided for the operator. Owing to the combinatorial nature of the problem, small lengths are usually preferred. When a new route is found, unscheduled orders are processed to find the next route. During this

process, the first route found is the overall "best route". Later routes are called "worse routes"; because they are constructed after better routes preceding them, and they have not been proved to be the best in the solution space. If the first route were to be another route, later routes could be constructed differently. In this, the merge is blind; it just tries to find the best route from remaining orders, without thinking about later routes.

3.2. Result elimination

As we have indicated, the routes that are generated later in the merge process are called the "worse routes". The last route is called the "worst route". What RE does is to eliminate worse routes by combining them with better routes; if possible, combining them with the best route. When doing this, parts of the routes can be replaced. RE tries all the orders in the worse route one by one, and adds them to the better route. If any of the orders added to the better route makes the schedule better, it accepts the change and removes the order from the worse route and adds it to the better route. Thus, the worse route either diminishes completely or shrinks. During these operations, a couple of options manage how the RE is executed. For example, index selection is the process of finding the index on the best route to insert the order from the worse route. FIRST_FOUND method inserts the order into the first index found. This is a search process that tries to find indexes that do not violate constraints and makes the schedule better. FIRST_FOUND is very fast but cannot guarantee the best solution. On the other hand, the BEST_FOUND method considers all the indexes and inserts the order into the best index that provides maximum gain.

4. Applying simulated annealing to E-HBA

SA is applied to E-HBA as a RE Operator. As indicated, RE always tries to eliminate the worst routes, which are the later routes on the schedule. This is because when the E-HBA tries to construct the routes, it tries to find the best available from all the iterations, and so the remaining routes become worse according to the objective function, which is a simple distance function in our case.

Previous RE Operators were blind even if a worse objective would provide a better objective later in the iteration. In this situation, the new RE Operator, which is designed according to SA principles, can accept a worse objective according to the SA probability. This method, after applying it to the resulting schedule, can provide a better solution to the entire schedule, which we have shown below using Gehring and Homberger's (1999) data-set (referred to as GH in the coming sections). Appendix A gives the entire schedule for two of the best results we found (C1_10_2 and C1_10_6, respectively). However, we did not put all of the result schedules for the sake of brevity.

SA has a random change principle which takes a new solution from the result neighbourhood. Here our next neighbour is taken from the output of RE. In previous operators, a worse route could not be accepted, but this brand new operator can accept a worse route that could lead to a better solution when the heat goes to the minimum.

Basic algorithm steps are as follows (see Appendix B for the terminology);

(1) Initially run a simple clustering algorithm (K-Means) to generate stop clusters.
(2) Set Merge Join distance (MJD) to a minimum value such that only neighbours in the same clusters are selected.
(3) Construct initial routes using EHBA
    (a) Merge
    (b) Finalize
    (c) RE
    (d) Swap
    (e) Sequence

(4) Change (MJD) so that routes can grow taking stops nearby.

(5) If enough or no changes continue with 6, else go to 2.

(6) Now we have a reasonable schedule at hand.

(7) Set KM-Based_SA_RE_Operator as the RE operator.

(8) Run RE.

  (a) Select two routes according to the RE selection rules,

  (b) Select some stops from the first route according to RE line selection rules, remove and try to insert them to the other route,

  (c) If constraints are not violated, continue else go to 8-a.

  (d) If new distance is better then accept and go to 8-a.

  (e) If the new distance is not better, calculate SA principles and decrease the heat if probability allows the new state.

  (f) If the heat reaches a minimum, go to 9.

  (g) Else go to 8-a.

(9) If enough or no changes in the routes, change RE settings, continue with 8, if no more settings could be changed exit.

5. Computational study

After initial experimentations, we concluded that when enabling each of the features of simulated annealing, it was best just trying with the minimum setting. Thus, our SA engine only displays two settings to be arranged.

The two controls both arrange the same setting for the number of runs needed. If the "estimated number of runs" control is changed, then "Initial Temperature" also changes. In simulated annealing, it is crucial to set the Initial Temperature and the minimum temperature appropriately. They must be arranged according to the problem at hand. "Initial Temperature" is the heat to be set for the system. Other settings for SA are taken as shown below:

$$\alpha = 0.9999 \tag{4}$$

$$\varepsilon = 1 \tag{5}$$

$$\text{Default Heat} = 1.5 \tag{6}$$

α is the heat changer in the SA system. If a new result is found, then the heat is multiplied by this value and the system continues to cool. ε is the minimum temperature we allow. "Default Heat" is the initial heat if no settings are given. These values are constant in our system. The only thing we change is the Initial Temperature and with it, the number of runs. The number of runs and the heat are calculated as follows in pseudo code, respectively;

```
getAnnealingNumRuns(heat)
begin
    curNumRun = 0
    while heat > ε
        heat * = α
        curNumRun + = 1
    return curNumRun
end.
```

```
double GetAnnealingTemperature(int p)
begin
        double reverseAlpha = 2−α
        double epsilon = ε
        var curNumRun = 0
        while curNumRun < p
                epsilon * = reverseAlpha
                curNumRun + = 1
        return epsilon
end.
```

Since we have previous results, we can deduce appropriate values for SA to continue. What we did in each step of the algorithms is as follows:

- If the sixth step is executed for the first time, initial temperature is set to 6. In turn sets the number of run to around 18,000. With these values we are trying to give SA enough room and length so that large gains can be obtained. With the first run, we know that the system is still hot and SA must know this.
- After each subsequent run, this value is changed until we arrive at a good result or no other changes take place.

If we give initial heat larger values than 5, then SA accepts too many poor results so that the overall distance increases and little or no gain is observed.

Our algorithm is coded in C# programming language on an Intel Core i7 2.4-GHz computer. We performed around 30 runs for each instance.

6. Sample results

Here we will give short descriptions on how we arranged our parameters to obtain the results. Also, the proposed method has many options that can be tweaked to make the compromises among time, distance or objective function at hand.

We take the routes as Hamiltonian paths; that is, the vehicles return to their bases. When returning, the distance and the time taken are added to the resulting value. Distances and the time they take are set to the same value, as is done in other papers. For example, if a single navigation takes ten units of length, then the time it takes that distance is again taken as ten units of time.

It can be seen from the previous best results that most of them are not directly comparable. Because the parameters they have taken are so different, one needs to fix the options to compare the two results chosen. To make the comparison between the proposed methodology and the others, we tried to find better results using the values they had as parameters to their algorithms.

6.1. Distance

The GH data-set contains many individual instances. Each configuration contains end points (or customers/orders) ranging from 200 to 1,000. Each instance may have a different configuration in terms of distance, location and time window. The routes constructed must have Hamiltonian path features. A Hamiltonian path is a graph where each vertex is visited only once and the route must construct a cycle. That is to say, the route must return to the first point or stop. There is only one central depot and many individual end points where orders must be delivered within the time window of each end point. Each customer may have orders in different volumes. The data-set contains the location in discrete X and Y coordinate values. The distance is calculated using Euclidian 2D

Table 1. Sample results

Data-set	BKS	References	BKSP	BKS NV	EHBA + SA	HBA NV
C1_10_1	42,516.63	Gehring and Homberger (2001)	42,478.95	100	42,516.23	100
C1_10_2	42,278.45	Nalepa and Blocho (2014)		90	42,378.69445	90
C1_10_3	40,187.99	Quintiq (2014)		90	40,231.5127	90
C1_10_4	39,468.6	Quintiq (2014)		90	39,490.52076	90
C1_10_5	42,469.18	Ropke and Pisinger (2007)		100	42,501.58	100
C1_10_6	43,830.21	Quintiq (2014)		99	42,495.4	100
C1_10_7	43,453.92	Quintiq (2014)		97	42,495.76	100
C1_10_8	41,853.36	Blocho and Czech (2013)		93	41,854.906	93
C1_10_9	40,570.6	Vidal, Crainic, Gendreau, and Prins (2013)		90	40,622.08445	90
C1_10_10	39,933.06	Vidal et al. (2013)		90	39,995.34413	90
C2_10_1	16,879.24	Li and Lim (2001)		30	16,919.32	30
C2_10_2	17,126.39	Blocho and Czech (2013)		29	17,180.14135	29
C2_10_3	17,126.39	Blocho and Czech (2013)		29	17,191.18	29
C2_10_4	15,656.75	Vidal et al. (2013)		28	15,699.55928	28
C2_10_5	16,561.29	Vidal et al. (2013)		30	16,612.21	30
C2_10_6	16,920.33	Blocho and Czech (2013)		29	16,948.08903	29
C2_10_7	17,882.42	Blocho and Czech (2013)		29	17,947.73	29
C2_10_8	16,577.32	Vidal et al. (2013)		28	16,654.43	28
C2_10_9	16,370.44	Nalepa and Blocho (2014)		29	16,456.48	29
C2_10_10	15,944.72	Vidal et al. (2013)		28	16,013.45777	28
Total	583,607.29			1,228	582,204.6299	1,232

Notes: BKS: Best known heuristic solution; BKSP: Best known heuristic solution's published distance; NV: Number of vehicles used.

arithmetic. The duration of a path is equal to the distance value. Each instance has a maximum number of vehicles with a maximum capacity. Only one vehicle is allowed to carry an order. An order cannot be carried with different vehicles in the same schedule (that is to say, no split may occur).

C1_10 instances are clustered instances that allow around 90 routes. They can be called lightly clustered (as opposed to C2_10 instances) instances. C2_10 instances are massively clustered instances and only allow around 30 routes. There are other instance types, such as R1_10, R2_10, RC1_10 and RC2_10, but in this study, we concentrated on C1_10 and C2_10 instances.

6.2. Results

Table 1 shows some benchmark outputs that we obtained from the data-set. With these results in overall distance, our algorithm is 1.03% better. In overall number of routes our algorithm is 0.33% worse.

7. Conclusions and future work

HBA and E-HBA show their strength and performance in VRPPD (Sakalli et al., 2013) and VRPTW (Savran et al., 2015). However, they lacked some features proposed in the methodology provided, such as 'random worse operation acceptance' in its operators. E-HBA can now find better results in data-sets in Solomon (2014), which we did not mention here, and best results in Gehring and Homberger (1999), which is an extended version. Since the GH data-set is relatively large in terms of

customer numbers, it tests the performance of an algorithm in near real-life accuracy. The E-HBA solving in terms of affordable times and with good results, will allow larger data-sets to be solved more quickly and well.

Here, we have solved GH clustered instances (C1 and C2). Others, namely "random" (R) and "random clustered" (RC), are still to be solved and enhanced. Each instance type has unique features which need to be taken into consideration when optimizing. Schedule timing is another area that can be improved, with smart algorithms making compromises between distance and time.

Funding
This project was supported by TÜBİTAK TEYDEB Industrial Research Funding Program [grant number 7131341], awarded to LA LOJİSTİK Danışmanlık Hiz. Tic. Ltd. Şti.

Author details
Mehmet Fatih Yuce[1]
E-mail: mehmetyuce@stu.aydin.edu.tr
Erhan Musaoglu[1]
E-mail: erhanm@la.com.tr
Ali Gunes[2]
E-mail: aligunes@aydin.edu.tr
[1] R&D, LA Software Group, Cumhuriyet Cad Ozkan Sk 2/7, Kavacik, Istanbul, Turkey.
[2] Department of Computer Engineering, Istanbul Aydin University, Kucukcekmece, Istanbul, Turkey.

Citation information
Cite this article as: Enhancing heuristic bubble algorithm with simulated annealing, Mehmet Fatih Yuce, Erhan Musaoglu & Ali Gunes, *Cogent Business & Management* (2016), 3: 1220662.

References

Abramson, D. (1991). Constructing school timetables using simulated annealing: Sequential and parallel algorithms. *Management Science, 37*, 98–113. http://dx.doi.org/10.1287/mnsc.37.1.98

Baldacci, R., & Bartolini, E., & Laporte, G. (2010). Some applications of the generalized vehicle routing problem. *Journal of the Operational Research Society, 61*, 1072–1077. http://dx.doi.org/10.1057/jors.2009.51

Blocho, M., & Czech, Z. J. (2013). A parallel memetic algorithm for the vehicle routing problem with time windows. *Proc. 8th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)* (pp. 144–151). Compiegne.

Bräysy, O., & Gendreau, M. (2005). Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation Science, 39*, 104–118. http://dx.doi.org/10.1287/trsc.1030.0056

Çatay, B. (2010). A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications, 37*, 6809–6817.

Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. *Journal of Optimization Theory and Applications, 45*, 41–51.

Cordeau, J. F., & Laporte, G. (2003). The dial-a-ride problem (DARP): Variants, modeling issues and algorithms. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies, 1*, 89–101.

Crainic, T. G., Perboli, G., Mancini, S., & Tadei, R. (2010). Two-echelon vehicle routing problem: A satellite location analysis. *Procedia-Social and Behavioral Sciences, 2*, 5944–5955. http://dx.doi.org/10.1016/j.sbspro.2010.04.009

Crispim, J., & Brandão, J. (2005). Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. *Journal of the Operational Research Society, 56*, 1296–1302. http://dx.doi.org/10.1057/palgrave.jors.2601935

Czech, Z. J., Czarnas, P. (2002). A parallel simulated annealing for the vehicle routing problem with time windows. *Proc. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing* (pp. 376–383). Canary Islands, Spain.

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science, 6*, 80–91. http://dx.doi.org/10.1287/mnsc.6.1.80

Eiselt, H. A., Gendreau, M., & Laporte, G. (1995). Arc routing problems, Part II: The rural postman problem. *Operations Research, 43*, 399–414. http://dx.doi.org/10.1287/opre.43.3.399

Flood, M. M. (1955). The traveling-salesman problem. *Journal of the Operations Research Society of America, 4*, 61–75.

Gehring, H., & Homberger, J. (1999, January). A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. *In Proceedings of EUROGEN99* (Vol. 2, pp. 57–64). Berlin: Springer.

Gehring, H., & Homberger, J. (2001). A parallel two-phase metaheuristic for routing problems with time windows. *Asia-Pacific Journal of Operational Research, 18*, 35–47.

Granada, M. (2016). Literature review on the vehicle routing problem in the green transportation context. *revista. luna. azúl, 42*, 362–387.

Hoffman, A. J., Wolfe, J., Garfinkel, R. S., Johnson, D. S., Papadimitriou, C. H., Gilmore, P. C., ... Golden, B. L. (1986). *The traveling salesman problem: a guided tour of combinatorial optimization*. New York, NY: J. Wiley & Sons.

Jayaraman, V., & Ross, A. (2003). A simulated annealing methodology to distribution network design and management. *European Journal of Operational Research, 144*, 629–645. http://dx.doi.org/10.1016/S0377-2217(02)00153-4

Jemai, J., Zekri, M., & Mellouli, K. (2012, April). An NSGA-II algorithm for the green vehicle routing problem. In J.-K. Hao & M. Middendorf (Eds.). *European Conference on Evolutionary Computation in Combinatorial Optimization* (pp. 37–48). Berlin Heidelberg: Springer.

Kirkpatrick, S., Gelatt, Jr., C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680. http://dx.doi.org/10.1126/science.220.4598.671

Kumar, S. N., & Panneerselvam, R. (2012). A survey on the vehicle routing problem and its variants. *Intelligent Information Management, 4*, 66–74. http://dx.doi.org/10.4236/iim.2012.43010

Laporte, G., Desrochers, M., & Nobert, Y. (1984). Two exact algorithms for the distance-constrained vehicle routing problem. *Networks, 14*, 161–172. http://dx.doi.org/10.1002/(ISSN)1097-0037

Li, H., & Lim, A. (2001). *Large scale time-constrained vehicle routing problems: A general metaheuristic framework with extensive experimental results* (Working paper). Singapore: National University of Singapore.

Lin, S. W., Yu, V. F., & Chou, S. Y. (2009). Solving the Truck and Trailer routing problem based on a simulated annealing heuristic. *Computers and Operations Research, 36*, 1683–1692. http://dx.doi.org/10.1016/j.cor.2008.04.005

McKendall, Jr., A. R., Shang, J., & Kuppusamy, S. (2006). Simulated annealing heuristics for the dynamic facility layout problem. *Computers and Operations Research, 33*, 2431–2444. http://dx.doi.org/10.1016/j.cor.2005.02.021

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., & Teller, A. H. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics, 21*, 1087–1092. http://dx.doi.org/10.1063/1.1699114

Nagy, G., & Salhi, S. (2007). Location-routing: Issues, models and methods. *European Journal of Operational Research, 177*, 649–672. http://dx.doi.org/10.1016/j.ejor.2006.04.004

Nalepa, J., & Blocho, M. (2014). Co-operation in the parallel memetic algorithm. *International Journal of Parallel Programming, 43*, 812–839. doi:10.1007/s10766-014-0343-4

Quintiq. (2014, December 2). Retrieved from http://www.quintiq.com/optimization-world-records.aspx

Ralphs, T. K., Kopman, L., Pulleyblank, W. R., & Trotter, L. E. (2003). On the capacitated vehicle routing problem. *Mathematical Programming, 94*, 343–359. http://dx.doi.org/10.1007/s10107-002-0323-0

Ropke, S., & Pisinger, D. (2007). A general heuristic for vehicle routing problems. *Computers and Operations Research, 34*, 2403–2435.

Sakalli, A., Yesil, E., Musaoglu, E., Ozturk, C., & Dodurka, M. F. (2013). Heuristic bubble algorithm for a Linehaul routing problem: An extension of a vehicle routing problem with pickup and delivery. (pp. 435–439). *Proceedings of 14th IEEE International Symposium on Computational Intelligence and Informatics*, Budapest.

Savran, A. I., Musaoglu, E., Yildiz, C., Yuce, M. F., & Yesil, E. (2015). Extended heuristic bubble algorithm for the pickup and delivery problem with time windows. In *Applied Machine Intelligence and Informatics (SAMI), 2015 IEEE 13th International Symposium* (pp. 145–150). Herl'any, Slovakia: IEEE. http://dx.doi.org/10.1109/SAMI.2015.7061864

Savran, A. I., Yuce, M. F., & Yesil, E. (2014). *RouteArt: A new framework for vehicle routing problem with pickup and delivery using heuristic bubble algorithm*. Presentation 15th IEEE International Symposium on Computational Intelligence and Informatics (CINTI2014), Budapest.

Solomon, M. M. (2014, December 3). *Best Heuristic Solutions for Solomon Data Set*. Retrieved from http://w.cba.neu.edu/~msolomon/heuristi.htm

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows. *Computers & Operations Research, 40*, 475–489.

Vigo, D. (1996). A heuristic algorithm for the asymmetric capacitated vehicle routing problem. *European Journal of Operational Research, 89*, 108–126. http://dx.doi.org/10.1016/S0377-2217(96)90060-0

Zambito, L. (2006). The traveling salesman problem: a comprehensive survey. *Project for CSE,* 4080.

Appendix A

Routing samples

Table 1. C1_10_6

Route number	Route
1	0, 231, 1, 70, 496, 166, 515, 307, 617, 676
2	0, 660, 3, 402, 456, 565, 193, 670, 646, 263, 207
3	0, 856, 758, 774, 478, 410, 5, 477, 398
4	0, 451, 605, 578, 346, 443, 7, 206, 427, 67
5	0, 380, 44, 4, 17, 9, 89, 18, 105, 97, 919, 666
6	0, 599, 755, 704, 99, 10, 649, 556, 598, 273, 717
7	0, 130, 725, 612, 904, 855, 13, 78, 794, 971, 628, 376, 817
8	0, 184, 139, 955, 14, 645, 754, 700, 889
9	0, 363, 797, 45, 68, 575, 295, 180, 302, 187, 19
10	0, 577, 533, 329, 249, 945, 20, 46, 689, 138, 905, 225, 946
11	0, 183, 36, 663, 714, 561, 498, 438, 22, 311, 893, 266
12	0, 931, 173, 151, 137, 998, 163, 26, 665, 340
13	0, 985, 884, 305, 30, 136, 900, 644, 25, 1000
14	0, 963, 726, 801, 987, 466, 279, 31, 276, 41
15	0, 532, 33, 209, 709, 991, 372, 894, 672, 641
16	0, 922, 736, 34, 912, 61, 744, 911, 680
17	0, 394, 441, 40, 282, 298, 119, 610, 23, 524
18	0, 482, 625, 990, 584, 60, 809, 993, 864, 790, 280, 43
19	0, 122, 705, 316, 678, 461, 47, 773, 389
20	0, 190, 842, 50, 675, 822, 459, 830, 872, 903
21	0, 581, 933, 52, 335, 368, 654, 918, 530, 444, 699, 59, 140
22	0, 655, 976, 458, 798, 51, 591, 141, 239, 220, 55, 883, 637, 848, 647
23	0, 960, 810, 620, 959, 841, 580, 304, 480, 684, 833, 56
24	0, 957, 787, 806, 710, 318, 831, 961, 587, 58, 290
25	0, 38, 633, 262, 63, 618, 632, 366, 192, 12, 929
26	0, 836, 69, 694, 796, 951, 495, 420, 566, 21, 583
27	0, 76, 631, 467, 333, 639, 229, 730, 88, 659, 110, 42
28	0, 79, 664, 49, 489, 490, 16, 669, 731, 375, 701, 436, 387
29	0, 764, 879, 501, 683, 344, 426, 799, 594, 503, 399, 98, 82, 156
30	0, 194, 84, 867, 779, 908, 595, 468, 847, 916
31	0, 982, 707, 111, 551, 94, 843, 942, 885, 609, 877, 753
32	0, 571, 681, 948, 91, 96, 378, 275, 865, 75, 713, 185
33	0, 937, 181, 407, 102, 934, 720, 126, 2, 265, 53, 62
34	0, 785, 267, 837, 104, 132, 925, 526, 223, 24, 217, 568
35	0, 536, 257, 121, 966, 635, 247, 113, 452, 448, 367, 347, 83, 500
36	0, 616, 313, 481, 74, 876, 795, 114, 338, 895, 416, 284, 805, 878, 234
37	0, 997, 87, 585, 365, 120, 521, 996
38	0, 115, 962, 486, 743, 440, 512, 397, 939, 123
39	0, 106, 158, 81, 385, 537, 124, 816, 412, 160
40	0, 351, 127, 682, 475, 235, 596, 749, 429
41	0, 66, 545, 778, 129, 8, 491, 821, 150, 383, 695, 297

Route number	Route
42	0, 175, 144, 373, 692, 133, 913, 539, 891, 892, 507
43	0, 535, 303, 226, 671, 978, 851, 134, 358
44	0, 54, 483, 531, 607, 177, 195, 769, 135, 724, 379, 525, 791
45	0, 652, 823, 258, 317, 958, 505, 142, 516
46	0, 968, 745, 218, 371, 643, 143, 844, 956, 661
47	0, 386, 792, 901, 909, 259, 319, 145, 803, 165, 128, 691, 668, 874
48	0, 208, 425, 149, 332, 345, 936, 409, 739, 391
49	0, 35, 825, 932, 868, 154, 370, 827, 832, 270, 171, 146
50	0, 924, 155, 224, 590, 312, 702, 199, 737, 873, 576
51	0, 562, 148, 291, 57, 131, 593, 953, 560, 418, 159, 513, 374
52	0, 819, 999, 246, 943, 723, 920, 227, 850, 828, 433, 567, 161, 71, 862
53	0, 623, 734, 228, 107, 776, 162, 992
54	0, 364, 492, 219, 170, 240, 216, 349, 178, 237, 176
55	0, 116, 813, 840, 463, 586, 906, 541, 886, 174, 522
56	0, 473, 109, 77, 417, 424, 90, 395, 519, 728, 65, 186
57	0, 544, 278, 511, 308, 191, 712, 898, 979, 915, 732
58	0, 648, 949, 197, 767, 214, 198, 64, 589
59	0, 328, 92, 200, 846, 634, 624, 201, 445, 517, 454
60	0, 108, 182, 470, 93, 203, 716, 706, 875, 693, 762, 601, 232
61	0, 582, 479, 437, 719, 204, 37, 292, 696, 205, 746, 514
62	0, 488, 800, 211, 359, 442, 804, 715, 888, 923, 355
63	0, 212, 859, 324, 421, 369, 261, 838, 697, 814
64	0, 95, 834, 557, 381, 271, 215, 630, 80, 236, 264, 750, 642, 310
65	0, 336, 508, 254, 152, 112, 157, 29, 757, 101, 392, 453, 860, 283, 221
66	0, 306, 294, 504, 766, 343, 930, 756, 241, 230, 169, 540
67	0, 164, 323, 242, 419, 179, 622, 450
68	0, 941, 32, 520, 357, 686, 243, 327, 348
69	0, 984, 896, 244, 497, 48, 935, 484, 653, 975, 434, 765
70	0, 28, 775, 973, 188, 527, 245, 793, 636, 147, 747
71	0, 309, 890, 11, 721, 253, 651
72	0, 487, 255, 899, 289, 572, 950, 423, 85, 554, 233, 881, 650
73	0, 981, 125, 954, 404, 39, 782, 256, 857, 510
74	0, 238, 597, 287, 772, 506, 626, 471, 260
75	0, 570, 350, 882, 400, 752, 269, 341, 431, 606
76	0, 615, 667, 808, 926, 964, 988, 277, 286, 824
77	0, 815, 658, 656, 771, 761, 168, 592, 464, 281
78	0, 285, 356, 472, 393, 967, 474, 688, 315, 518
79	0, 384, 853, 432, 727, 288, 835, 858, 361, 839, 608, 994, 447
80	0, 167, 293, 542, 928, 952, 619, 446, 396, 733, 15, 73, 741, 469
81	0, 299, 449, 602, 546, 759, 377, 296, 995
82	0, 301, 777, 564, 320, 360, 763
83	0, 611, 940, 534, 738, 807, 314, 770, 687
84	0, 388, 760, 272, 321, 353, 983, 499, 405, 548
85	0, 415, 322, 627, 462, 698, 559, 172, 569, 603
86	0, 854, 783, 640, 861, 339, 457, 748, 325, 673, 735

Route number	Route
87	0, 722, 718, 326, 334, 100, 414, 430, 802, 550
88	0, 977, 390, 352, 600, 870, 786, 274, 869
89	0, 789, 553, 103, 331, 972, 974, 422, 944, 401, 579, 538
90	0, 711, 403, 970, 729, 413, 866, 887, 408, 690, 812, 614, 845
91	0, 406, 820, 502, 222, 927, 509, 465, 196, 674, 871, 662
92	0, 382, 784, 460, 708, 852, 428, 907
93	0, 849, 476, 788, 528, 494, 354, 555, 342, 558, 685, 549, 552
94	0, 914, 362, 621, 969, 213, 251, 455, 252, 86, 485
95	0, 780, 529, 300, 965, 740, 677, 921, 989, 917
96	0, 6, 268, 980, 210, 574, 118, 897, 202, 547
97	0, 411, 563, 986, 938, 818, 910, 27, 189, 153, 811
98	0, 902, 543, 679, 604, 742, 781, 880, 703, 493
99	0, 638, 72, 573, 337, 613, 947, 588, 439, 751
100	0, 435, 250, 629, 829, 523, 117, 768, 657, 863, 330, 248, 826

Table 2. C1_10_7

Route number	Route
1	0, 231, 1, 70, 496, 166, 515, 307, 687, 576, 676
2	0, 660, 3, 402, 456, 565, 193, 670, 646, 263, 207
3	0, 856, 758, 774, 478, 410, 5, 477, 398
4	0, 184, 139, 955, 14, 645, 754, 700, 889, 724
5	0, 167, 293, 542, 928, 952, 619, 446, 396, 733, 15, 73, 741, 469
6	0, 79, 664, 49, 489, 490, 16, 669, 731, 375, 701, 436, 387
7	0, 380, 44, 4, 17, 9, 89, 18, 105, 97, 919, 666
8	0, 577, 533, 329, 249, 945, 20, 46, 689, 138, 905, 946
9	0, 836, 69, 694, 796, 951, 495, 420, 566, 21, 583
10	0, 394, 441, 40, 282, 298, 119, 610, 23, 524
11	0, 28, 775, 973, 188, 527, 245, 793, 636, 147, 747
12	0, 985, 884, 305, 30, 136, 900, 644, 25, 1000
13	0, 963, 726, 801, 987, 466, 279, 31, 276, 41
14	0, 922, 736, 34, 912, 61, 744, 911, 680, 225
15	0, 183, 36, 663, 714, 561, 498, 438, 22, 311, 893, 266
16	0, 582, 479, 437, 719, 204, 37, 292, 696, 205, 746, 514
17	0, 981, 125, 954, 404, 39, 782, 256, 857, 510
18	0, 363, 797, 45, 68, 575, 295, 180, 302, 187, 19
19	0, 122, 705, 316, 678, 461, 47, 773, 389
20	0, 984, 896, 244, 497, 48, 935, 484, 653, 975, 434, 765
21	0, 190, 842, 50, 675, 822, 459, 830, 872, 903
22	0, 655, 976, 458, 798, 51, 591, 141, 239, 220, 55, 883, 637, 848, 647
23	0, 581, 933, 52, 335, 368, 654, 918, 530, 444, 699, 59, 140
24	0, 937, 181, 407, 102, 934, 720, 126, 265, 2, 53, 62
25	0, 562, 148, 291, 57, 131, 593, 953, 560, 418, 159, 513, 374
26	0, 482, 625, 990, 584, 60, 809, 993, 864, 790, 280, 43
27	0, 38, 633, 262, 63, 618, 632, 366, 192, 12, 929

Route number	Route
28	0, 638, 72, 573, 337, 613, 947, 588, 439, 751
29	0, 106, 158, 81, 385, 537, 124, 816, 412, 160
30	0, 194, 84, 867, 779, 908, 595, 468, 847, 916
31	0, 487, 255, 899, 289, 572, 950, 423, 85, 554, 233, 881, 650
32	0, 997, 87, 585, 365, 120, 521, 996
33	0, 571, 681, 948, 91, 96, 378, 865, 75, 713, 185
34	0, 328, 92, 200, 846, 634, 624, 201, 445, 517, 454
35	0, 722, 718, 326, 334, 100, 414, 430, 802, 550
36	0, 336, 508, 254, 152, 112, 157, 29, 757, 101, 392, 453, 860, 283, 221
37	0, 785, 267, 837, 104, 132, 925, 526, 223, 24, 217, 568
38	0, 623, 734, 228, 107, 776, 162, 992
39	0, 473, 109, 77, 417, 424, 90, 395, 519, 728, 65, 186
40	0, 115, 962, 486, 743, 440, 512, 397, 939, 123, 549
41	0, 435, 250, 629, 829, 523, 117, 768, 657, 863, 330, 248, 826
42	0, 6, 268, 980, 210, 574, 118, 897, 202, 547
43	0, 66, 545, 778, 129, 8, 491, 821, 150, 383, 695, 297
44	0, 130, 725, 612, 904, 855, 13, 78, 794, 971, 628, 376, 817
45	0, 175, 144, 373, 692, 133, 913, 539, 891, 892
46	0, 652, 823, 258, 317, 958, 505, 142, 516
47	0, 968, 745, 218, 371, 643, 143, 844, 956, 661
48	0, 208, 425, 149, 332, 345, 936, 739, 281, 391
49	0, 411, 563, 986, 938, 818, 910, 27, 189, 153, 811
50	0, 924, 155, 224, 590, 312, 702, 199, 737, 873
51	0, 764, 879, 501, 683, 344, 426, 799, 594, 503, 399, 98, 82, 156
52	0, 819, 999, 246, 943, 723, 920, 227, 850, 828, 433, 567, 161, 71, 862
53	0, 815, 658, 656, 771, 761, 168, 592, 464
54	0, 364, 492, 219, 170, 240, 216, 349, 178, 237, 176
55	0, 35, 825, 932, 868, 154, 370, 827, 832, 270, 171, 146
56	0, 116, 813, 840, 463, 586, 906, 541, 886, 174, 522
57	0, 54, 483, 531, 607, 177, 195, 769, 135, 379, 525, 791
58	0, 108, 182, 470, 93, 203, 716, 706, 875, 693, 762, 601, 232
59	0, 406, 820, 502, 222, 927, 509, 465, 196, 674, 871, 662
60	0, 648, 949, 197, 767, 214, 198, 64, 589
61	0, 532, 33, 209, 709, 991, 372, 894, 672, 641
62	0, 488, 800, 211, 359, 442, 804, 715, 888, 923, 355
63	0, 212, 859, 324, 421, 369, 261, 838, 697, 814
64	0, 914, 362, 621, 969, 213, 251, 455, 252, 86, 485
65	0, 95, 834, 557, 381, 271, 215, 630, 80, 236, 264, 750, 642, 310
66	0, 535, 303, 226, 671, 978, 851, 134, 409, 358
67	0, 76, 631, 467, 333, 639, 229, 730, 88, 659, 110, 42
68	0, 306, 294, 504, 766, 343, 930, 756, 241, 230, 169
69	0, 616, 313, 481, 74, 876, 795, 114, 338, 895, 416, 284, 805, 878, 234
70	0, 351, 127, 682, 475, 235, 596, 749, 429
71	0, 941, 32, 520, 357, 686, 243, 327, 348
72	0, 309, 890, 11, 721, 253, 651
73	0, 386, 792, 901, 909, 259, 319, 145, 803, 165, 128, 691, 668, 874

Route number	Route
74	0, 238, 597, 287, 772, 506, 626, 471, 260
75	0, 570, 350, 882, 400, 752, 269, 275, 341, 431, 606
76	0, 599, 755, 704, 99, 10, 649, 556, 598, 273, 717
77	0, 977, 390, 352, 600, 870, 786, 274, 869
78	0, 615, 667, 808, 926, 964, 988, 277, 286, 824
79	0, 544, 278, 511, 308, 191, 712, 898, 979, 915, 732
80	0, 384, 853, 432, 727, 288, 835, 858, 361, 839, 608, 994, 447
81	0, 982, 707, 111, 551, 94, 843, 942, 885, 609, 507, 753
82	0, 299, 449, 602, 546, 759, 377, 296, 995
83	0, 301, 777, 564, 320, 360, 763, 569
84	0, 960, 810, 620, 959, 841, 580, 304, 480, 684, 833, 56
85	0, 611, 940, 534, 738, 807, 314, 770, 617
86	0, 285, 356, 472, 393, 967, 474, 688, 315, 518
87	0, 957, 787, 806, 710, 318, 831, 961, 587, 58, 877, 290
88	0, 388, 760, 272, 321, 353, 983, 499, 405, 548
89	0, 415, 322, 627, 462, 698, 559, 172, 603
90	0, 789, 553, 103, 331, 972, 974, 422, 944, 401, 579, 538
91	0, 931, 173, 151, 137, 998, 163, 26, 665, 340
92	0, 382, 784, 708, 460, 852, 428, 907
93	0, 711, 403, 970, 729, 413, 866, 887, 408, 690, 812, 614, 845
94	0, 164, 323, 242, 419, 179, 622, 450
95	0, 849, 476, 788, 528, 494, 354, 555, 342, 558, 685, 552
96	0, 902, 543, 679, 604, 742, 781, 880, 703, 493
97	0, 536, 257, 121, 966, 635, 247, 113, 452, 448, 367, 347, 83, 500
98	0, 780, 529, 300, 965, 740, 677, 921, 989, 917, 540
99	0, 451, 605, 578, 346, 443, 7, 206, 427, 67
100	0, 854, 783, 640, 861, 339, 457, 748, 325, 673, 735

Appendix B

E-HBA in Depth

i. Merge Operator

Basically the time it takes for Merge Operator to generate one route is:

$$T_1 = \prod_{x=S_1-R}^{S_1} (xN) \tag{1}$$

where T_1 is the time it takes to generate the first route, S_1 is the number of stops available, R is the route length, and N is the time to control (constraint checks, time window allocation etc.) a single route generated. So the total time it takes for Merge Operator to finish a schedule is:

$$T_t = \sum_{y=1}^{C} \prod_{x=S_y-R}^{S_y} (xN) \tag{2}$$

where T_t is the time it takes to generate the entire schedule. C is the number of routes in the schedule. S_y is the number of stops available when the yth route is being generated, R is the route length,

N is the time to control a single route generated. As can be seen, increasing the route length is expensive for the Merge Operator. This is why the Join Operator is created.

ii. Join Operator

The Merge Operator cannot construct longer routes in an acceptable computational time. The longer the route length, the longer it will take to generate a single route, because it tries every combination of stops that could be constructed. Usually, when the data-set is good enough, merge will quickly generate routes with four stops, in around one hundred milliseconds. Getting longer routes generally makes the processing time longer and more expensive. To overcome that, a local search method called "join" is created as an operator. This operator searches the remaining solution space for a better candidate with a longer route length.

There are two options that can be set for the Join Operator: the first is the "delta distance". Delta distance is computed between three stops. When the Join Operator tries to insert a new stop into the route, it selects a "navigation" edge. A "navigation" edge can be thought of as a directed graph edge, going from one stop (vertex) to another in the same route. The candidate stops that can be inserted between these two stops (creating two edges), is taken from the delta distance set. A "delta distance set" for an edge is the set of all the stops that, when inserted in between, will not increase the route more than the "delta distance".

$$\Delta DS = \left\{ \left(D\left[v_f, v_s\right] + D\left[v_s, v_t\right] \right) - D\left[v_f, v_t\right] < \Delta, \forall s \in S \right\} \tag{3}$$

In Equation (3), ΔDS is the delta distance set available for the selected edge. D is the two-dimensional distance matrix, v_f and v_t are from and to vertices (stops), respectively, v_s is the newly generated vertex to be created, Δ is the delta distance setting provided to the HBA and S is the set of all stops available.

The other option provided is the "Alpha Distance" setting, which is used for performance reasons. The S set above indicates all the stops available to be joined to the selected edge of the route. Alpha distance filters those stops, yielding a smaller set called "Alpha distance set". Then when calculating delta distance, the stops provided will be taken from this set.

$$\alpha DS = \left\{ D\left[v_f, s\right] < \alpha, \forall s \in S \right\} \tag{4}$$

In Equation (4), αDS is the alpha distance set available for v_f; the "from stop". From stop or from vertex is the left-hand side of the edge, when the route is thought out as a directed graph from left to right. D is the two-dimensional distance matrix, s is a stop that is near to v_f. α is the alpha distance setting provided to the HBA, and again, S is the set of all stops available.

The time it takes to make a single join to a resulting route is considerably less than generating the same route using merge, as the maximum number of operations that will be executed is only proportional to the number of stops available, not to any combinatorial combination of them.

iii. Result elimination settings

1. Cluster distance

Cluster distance is the same as **delta distance** in the Join Operator.

2. Route selection

Decides which routes should be selected for the worse route's order to be distributed upon. Centre of Gravity, Centre of Mass, ALL etc. are the basic options.

3. Order selection

Decides on the orders of the worse route to be removed and added to the better route. Multiple All, Multiple Random, Single etc. are basic options.

iv. Split Operator

Split Operator is another operator that executes after Merge and Join Operators, together with RE and Swap. This operator also tries to make the schedule better by utilizing cross sections of routes. If two routes visit the same stop, and if the time window and other constraints are preserved, then one route can leave some of its orders on the stop and the other route can pick those orders and take them to their customer locations, but only if the resulting routes creates a better schedule.

v. Swap Operator

Swap Operator is similar to RE Operator. The only difference is that it can exchange the order between routes. That is to say, if an order is selected in one route, then another order can be selected on the other routes. These two orders can be exchanged if the resulting schedule will become better. This order selection is stochastic. Swap Operator may not select two orders but one from a single route and insert it into the other route. If this is the case, it is technically what the Result Elimination does.

vi. Merge Join Distance

Merge Join Distance (MJD) is the maximum distance a stop can increase the route length. If a stop should be added to a route, it cannot increase the route distance by more than the MJD.

For example, Let us assume that

- a route has two stops, a and b,
- and the route has a length of 10.
- we assume that previously we have set the MJD as 4.

Later we needed to add a new stop, c, to the end after b. The distance between b and c, d [b, c], is 5. Because the MJD is 4, c cannot be added to the route.

Also, if we needed to add a new stop, d, to the end after b and the distance between b and d, d [b, c], is 3. Because the MJD is 4, d can be added to the route.

vii. Finalize Operator

It is always not possible to route all the orders in a set. Usually some orders will remain as unscheduled. It is because if they are routed, they will violate some constraint. Finalize Operator will schedule those orders even if they break the validity of the schedule. This is done partly because usually after running other operators, such as result elimination, these orders can be inserted into a better valid route to remove the violation.

Cogent Business & Management **(ISSN: 2331-1975) is published by Cogent OA, part of Taylor & Francis Group.**