Vehicle Routing Problem with Multiple Time Windows and Batch Splitting Based on Inferior First Bidirectional Search Algorithm

Y H Li\(^1\), H J Mao\(^2\) and Y M Qin\(^3\)

\(^1\) School of Transportation, Southeast University, Nan Jing, China
\(^2\) School of Mechanical and Transportation Engineering, Guangxi University of Science and Technology, Liu Zhou, China

Abstract. Considering continuous scheduling and full use of time in vehicle routing problem, we proposed a vehicle routing problem model with multi-time windows based on batch splitting, constructed an inferior-first bidirectional search algorithm (IFBSA) and established a quantitative measurement benchmark of “inferiority level” using Gaussian kernel function. We designed 30 examples based on some actual road network data and chose nearest neighbour search algorithm (NNSA) for comparing. Our results showed that, IFBSA reduced vehicles in 20 examples and got superior results in 23 examples for all examples compared with NNSA.

1. Introduction
In urban distribution businesses, limited by various environmental conditions such as business hours, regional restrictions, temporary traffic control, etc., nodes’ service times are not available 24 hours a day, even some nodes are blocked in some area temporarily or long-term for reasons. Usual distribution works need to schedule delivery time of vehicles during these limited time sections, sometimes requiring multiple round trips. Therefore, how to make full use of available service time for nodes and vehicle resources to find some cost-effective and efficient distribution schemes is a practical research topic. Considering time segmentation in distribution and multiple dispatching scheduling, we study how to complete distribution demand efficiently during feasible period of segmentation.

2. Articles review
Vehicle routing problem with multiple time windows (VRPMTW), as a branch of vehicle path optimization problem, has been studied by many scholars. These literature research mainly focused on the problem with one-time distribution. For design of examples, researchers mainly used theoretical node distance simulation based on satisfying triangle inequality, like Solomon\(^1\) for 56 benchmarks. For solving algorithms, Asvin \(^2\) built a long-distance truck driver scheduling problem model based on multi-time windows using large neighbourhood search algorithm, Pesant et al.\(^3\) used constraint programming method to solve multi-time windows TSP problem, Favaretto et al.\(^4\) tried an ant colony algorithm for the VRP with multiple time windows and multiple visits, Peng et al.\(^5\) constructed a hybrid ant colony algorithm and Ma et al.\(^6\) used simulated annealing algorithm for solving VRPMTW problem, Belhaiza et al.\(^7\) made a variable neighbourhood tabu search algorithm to solve problem, Doerner et al.\(^8\) found a heuristic constructive procedure mixed with branch-and-bound method to compute vehicle routing problem with multiple interdependent time windows, Tricoire et al.\(^9\) studied some heuristics for multi-period orienteering problem with multiple time windows, Christiansen et
al.\cite{10} used a set partitioning approach to solve ship scheduling with multiple time windows. For designing nodes’ service time, the main literature’s service time can be elastically inside or outside the chosen time window.

However, in some actual conditions, vehicles need to make service time in arriving time window and travel to and from distribution centre more than one time. So, we make a way of completing delivery in the chosen time window, which puts the service time in the time window to ensure nodes can send delivery more reliably and safely. And for the vehicle path setting, vehicle(s) will also be restricted to some extent by one-way or two-way traffic rules. The two-way distances of nodes are not strictly equal to the theoretical distances. Therefore, we propose a distribution optimization problem model based on batch splitting, different distance of two-way distribution, multiple distribution periods and continuous scheduling of vehicles. We call Vehicle Routing Problem with Multiple Time Windows and Batch Splitting, abbreviated as VRPMTWBS.

3. Problem definition and mathematical model

We define VRPMTWBS problem as follows:

(1) Time window(s) of each node are known, and only normal distribution is considered;
(2) All vehicles responsible for distribution have same capacity, and can be continuously dispatched for distribution services;
(3) Service time of distribution centre is fixed, and each time window is within time period that distribution centre can reach and return;
(4) Distribution demand and service time of each node are known;
(5) Each node needs once and only once delivery service, and only one of node’s time window(s) is selected for delivery service. Node’s demand is not divided;
(6) The first priority goal is to minimize number of vehicles. The second is to minimize total delivery time. The first target has absolute priority compared to the second one.

3.1 Variables definition

$G=<V,E>$ is the distribution network structure, $V=\{1,\ldots,n\}$ represents all nodes, the “1” node in $V$ represents the distribution centre, $N=V\setminus\{1\}$ is customer nodes, and $E$ represents distance arcs set between nodes. The available service time interval of distribution centre is $[x,y]$ and $K=\{1,\ldots,r\}$ is the number of available vehicles which are always sufficient for distribution, every vehicle $k$’s capacity is $Q$, $d_{e_i}$ indicates demand quantity of node $i$, $a$ in the $[e_i^a,l_i^a]$ represents $a$’s time window of node $i$, $e_i^a$ is start edge of time window $a$, and $l_i^a$ is end edge of time window $a$, $a\in\{1,2,\ldots,B_i\}$, where $B_i$ is time window total number of node $i$. $d_{i,j}$ indicates direct distance from node $i$ to node $j$, $t_{i,j}$ is direct travel time from node $i$ to node $j$. $a_{ij}$ represents time arrived at node $i$, $s_{t_i}$ indicates service time for node $i$. $w_{t_i}$ is waiting time at node $i$. The batch time split segment number is $P=\{p_1,\ldots,p_m\}$, and the split time position is $T=\{t_1,\ldots,t_m\}$, and $Q(P,k)$ indicates total load capacity of vehicle $k$ in the chosen path segment $x$. At each time split position, the loading time of the vehicle returning to the distribution centre is $T_X$, which is a fixed constant. The number of vehicles required to solve the model algorithm is $X$. $M$ is a large enough integer.

The decision variables are introduced:

$$x_{i,j,k}^l = \begin{cases} 
1, & \text{if vehicle } k \text{ visits node } j \text{ straight after node } i \\
0, & \text{otherwise}
\end{cases}$$
\[ C^a_i = \begin{cases} 1, & \text{if } a \text{ is the choosen service time window of node } i \\ 0, & \text{otherwise} \end{cases} \]

### 3.2 Mathematical model

**Object function:**

\[
\min(M \times X + \sum_{k=1}^{K} \sum_{i=2}^{N} \sum_{j=2}^{N} t_{i,j} \times x_{i,j}^k + \cdots + \sum_{k=1}^{K} \sum_{i=2}^{N} \sum_{j=2}^{N} t_{i,j} \times x_{i,j}^k)
\]

Subject to:

\[
\sum_{k=1}^{K} \sum_{i=2}^{N} x_{i,j}^k = 1, \forall j \in V \quad (2)
\]

\[
\sum_{k=1}^{K} \sum_{j=2}^{N} x_{i,j}^k = 1, \forall i \in V \quad (3)
\]

\[
\sum_{a=1}^{a} C_i^a = 1, \forall i \in N \quad (4)
\]

\[
Q(P_i, k) \leq Q, x \in \{1, \ldots, m\}, k \in K \quad (5)
\]

\[
e_i^a = a_t + w_t \leq l_i^a, C_i^a = 1, i \in N \quad (6)
\]

\[
e_i^a = a_t + w_t + t_i \leq l_i^a, C_i^a = 1, i \in N \quad (7)
\]

\[
C_i^a \times x_{i,j}^k \leq a_t, C_i^a = 1; x_{i,j}^k = 1; i, j \in N, i \neq j, \forall k \in K \quad (8)
\]

Equation (1) represents model’s objective function. \((M \times X)\) indicates the total number of vehicles required. Constraints (2) and (3) ensure that each non-distribution centre node has to be and can be served by only one vehicle; Constraints (4) indicates that each non-distribution node finally has to and can select only one time window to complete its delivery service; Constraints (5) indicates that the total load of each sub-path cannot exceed capacity of current vehicle; Constraints (6) indicates that start service time of each node must be within node’s selected time window; Constraints (7) indicates that the node’s cargo handover service must be completed within the node’s feasible time window; Constraints (8) indicates that the variables between nodes need to satisfy progressive relationship for time line.

### 3.3 Heuristic Algorithm

Our algorithm is designed in a hierarchical and step-by-step way. The algorithm is called Inferior First Bidirectional Searching Algorithm (IFBSA). All steps are as follows:

1. **First node selection**

   The "inferior" nodes are selected to enter path first. The Quantitative measure of nodes’ "inferiority" is \(\xi, \zeta\) has four indicators: \(\rho\) as relativeness space of node \(i\) with all other nodes, \(\beta\) as ratio of total available time window length of node \(i\) to total time window length of distribution centre, \(B_i\) as the number of time window(s) of node \(i\), \(\lambda\) as ratio of total time window length of node \(i\) to the average total time window length of all non-distribution nodes.

   For \(\rho\), We use continuous Gaussian kernel function adopted by Rodriguez et al.[11] to convert the relativity between nodes into quantitative values. \(d_{i,j}\) is path distance between node \(i\) and node \(j\). \(dc\) is the relative concentration boundary value, which is selected from the matrix \(Y\) generated by all distance values of nodes’ distance matrix made by ascend sort. The position of the selected distance value is determined by \(\theta\), \(\theta = [(n-1)^2 \times \alpha / 100]\). \(dc\)’s value is the distance value corresponding to \(\theta\)’s position. \(\theta\) is determined by \(\alpha\), which represents proportion of all distance value sets selected. The final value we chose for \(\alpha\) is 2.

   Five parameter scales are designed as follows:
1) \( S_1: \ \xi_i = \rho_i \); 2) \( S_2: \ \xi_i = \beta_i \); 3) \( S_3: \ \xi_i = \rho_i \times \beta_i \); 4) \( S_4: \ \xi_i = \rho_i \times \beta_i \times \lambda_i \); 5) \( S_5: \ \xi_i = \rho_i \times \beta_i \times \lambda_i \times B_i \);

In order to ensure that the time window finally selected is as long as possible, the time window having larger width is preferentially selected for insertion. The first sub-path position that satisfies the condition is inserted into the node, then the insertion time window of the selected node is recorded.

(2) Two-way simultaneous insertion

We use a two-way insertion method to allow each time window of each node to have more time elasticity and positional flexibility to enter current path. Starting from first insertion point, we simultaneously calculate forward and reverse phases, until all the nodes that have not entered the construction path are checked. If there are some node(s) worked, the node that increases total service time least is chosen for inserting. Record insertion node and it’s time window, then update the node matrix for nodes not inserted. Go to the next cycle and repeat step (2);

(3) Insertion failure settings

If all the nodes not inserted in the current sub-path fail to being inserted, enter next sub-path, return to step (1) for selecting first inserted node. If some insertions are successful, the bidirectional selection insertion position is also working in the subsequent sub-path, the algorithm repeat procedure until the current sub-path is constructed fully. If current path cannot join new node and there are some nodes not added to the path, start a new path, Route=Route+1;

(4) Repeat steps

Repeat (1), (2), and (3), until all nodes have joined the path, then our algorithm ends.

4. Data and examples

Nodes’ distance data comes from API port of BAIDU developer based on the GPS coordinates. There are 5 groups in all, each group has 6 examples, totally 30 examples, labeled from 1A to 5F. Each group has same time windowupper bound. The upper bound ranges from 1 to 5. A, B, C, D, E and F represent the elastic width range of time window(s), which increases gradually from A to F. All the examples’ time unit are minutes.

5. Results and analysis

Our algorithm runs under condition of 64-bit operating system of Windows 7 flagship edition, Intel 2.40 GHz processor and 8 GB running memory. The results are obtained by programming with MATLAB. Every group of data is computed ten times for better results. The comparing algorithm is Nearest Neighbourhood Searching Algorithms (NNSA). After comparing, we finally choose \( S_4 \) as the best \( \xi_i \) scale. The results of IFBSA and NNSA are shown in Table 1.

| Date | IFBSA | NNSA | Difference (%) |
|------|------|------|----------------|
|      | \( TV^a \) | \( TT^b \) | \( TV \) | \( TT \) | \( TV \) | \( TT \) |
| 1A   | 24   | 18240| 25 | 19000 | -4.00 | -4.00 |
| 1B   | 19   | 11033| 20 | 11219 | -5.00 | -1.66 |
| 1C   | 17   | 12848| 18 | 13328 | -5.56 | -3.60 |
| 1D   | 12   | 9459 | 14 | 10403 | -14.29 | -9.07 |
| 1E   | 12   | 7695 | 13 | 7693  | -7.69 | 0.03  |
| 1F   | 13   | 8074 | 13 | 7515  | 0.00  | 7.44  |
| 2A   | 13   | 10578| 13 | 9966  | 0.00  | 6.14  |
| 2B   | 12   | 10279| 13 | 10568 | -7.69 | -2.73 |
| 2C   | 14   | 9871 | 15 | 9530  | -6.67 | 3.58  |
| 2D   | 13   | 9211 | 14 | 8956  | -7.14 | 2.85  |
| 2E   | 12   | 7419 | 13 | 6858  | -7.69 | 8.18  |
| 2F   | 12   | 6926 | 13 | 6588  | -7.69 | 5.13  |
Table 1 showed that, IFBSA algorithm had fewer vehicles in 20 cases. In other nine examples obtained same number of vehicles, three of which have less total delivery time. Number 5A example was the only one which has vehicles more than NNSA. IFBSA algorithm obtained dominant results in 23 cases out of 30, accounting for about 76.7% in all results. In addition, under the same conditions, IFBSA algorithm and NNSA algorithm both show that the increase of upper number of time windows has a significant impact on outputs. Especially in the short-time window example of 1A-1C to 2A-2C with the upper number from 1 to 2, the number of reduced vehicles in IFBSA algorithm and NNSA algorithm are 11, 7, 3 and 12, 7, 3, respectively. When the upper number time windows in the same group is constant, the wider the single time window is, the better the output is. However, as the time window width increases, the influence of time window width is gradually reduced.

### 6. Conclusions

In this paper, actual road network is used as source of the matrix data of examples, service times are considered in nodes’ time windows, and feasible continuity of continuous scheduling too. Based on above three points, VRPMTWBS is made, and IFBSA algorithm is created to solve it. The validity of the algorithm is verified by constructing a numerical example. The results show that, compared with NNSA algorithm, IFBSA algorithm propose outperforms NNSA algorithm in more than 76% of examples, and its vehicle number advantage is obtained in 20 examples. We also see some significant impact on delivery target by time split location.

### References

[1] Marius M. Solomon 1987 Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints *Operations Research* 35(2):254-265

[2] Goel A 2009 Vehicle Scheduling and Routing with Drivers’ Working Hours *Transportation Science* 43(1):17-26

[3] Gilles P, Michel G, Jean-Yves P and Jean-Marc R 1999 On the flexibility of constraint programming models: From single to multiple time windows for the traveling salesman problem *European Journal of Operational Research* 117(2):253-263

[4] Daniela F, Elena M and Paola P 2007 Ant colony system for a VRP with multiple time windows and multiple visits *Journal of Interdisciplinary Mathematics* 10(2):263–284
[5] Bi-tao P and Yong-wu ZH 2010 Hybrid ant colony algorithm for vehicle routing problem with multiple time windows *Computer Engineering and Applications* **46**(31):28-31
[6] Hua-wei M, Chun-rong Z and Shan-lin Y 2009 Modeling and solving for vehicle routing problem with multiple time windows *Journal of Systems Engineering* **24**(5):607-613
[7] Slim B, Pierre H and Gilbert L 2014 A hybrid variable neighbourhood tabu search heuristic for the vehicle routing problem with multiple time windows *Computers & Operations Research* **52**:269–281
[8] Karl D, Manfred G, Richard H, Guenter K and Marc R 2008 Exact and heuristic algorithms for the vehicle routing problem with multiple interdependent time windows *Computers & Operations Research* (35):3034–3048
[9] Fabien T, Martin R, Karl D and Richard H 2010 Heuristics for the multi-period orienteering problem with multiple time windows *Computers & Operations Research* **37**:351-367
[10] Marielle C and Kjetil F 2002 Robust Ship Scheduling with Multiple Time Windows *Naval Research Logistics* **49**: 611-625
[11] Alex R and Alessandro L 2014 Clustering by fast search and find of density peaks. *Science* **344**(6191):1492–1496