Chapter 3

Time window compatibility

The purpose of this chapter is to introduce a new concept, *Time Window Compatibility* (TWC). A matrix, referred to as the *Time Window Compatibility Matrix* (TWCM), is used as the mechanism to calculate the compatibility between all nodes in the network. A brief motivation for the concept is given. Various scenarios of the concept exist, and are illustrated individually. An attempt is made to define the concept in a generalized form.

An example is given to illustrate the use of the concept. Other than reducing the computational burden of existing heuristics, the example also illustrates the opportunity to improve the quality of the initial solution. A modified copy of the concepts discussed in this chapter have been communicated the the *European Journal of Operational Research.*

3.1 Motivation for a new approach

A shortcoming of the Sequential Insertion Heuristic (SIH) of Solomon [46] is that it considers all unrouted nodes when calculating the insertion and selection criteria for each iteration. The fact that *all* unrouted nodes are considered makes it computationally expensive. The VRP variant considered in this dissertation has multiple additional constraints. The occurrence of obvious infeasible nodes in a partially constructed route therefore becomes significant. The introduction of the time window compatibility concept assists in identifying, and eliminating, the obvious infeasible nodes. This results in a more effective and robust route construction heuristic.

3.2 Time window compatibility defined

The purpose of TWC is to determine the time overlap of all edges, or node combinations, (i, j), where $i, j \in \{0, 1, 2, \ldots, N\}$, and N the total number of nodes in the network. During the route construction phase, time win-

dow compatibility can be checked, and obvious infeasible nodes can be eliminated from the set of considered nodes. The TWCM is a non-symmetrical matrix as the sequence of two consecutive nodes, i and j, is critical.

Let:

N	be the total number of nodes
e_i	be the earliest allowed arrival time at customer i, where $i = \{0, 1, \ldots, N\}$
l_i	be the latest allowed arrival time at customer i, where $i = \{0, 1, \ldots, N\}$
s_i	be the service time at node i, where $i = \{0, 1, \ldots, N\}$
t_{ij}	be the travel time from node i to node j, where $i, j = \{0, 1, \ldots, N\}$
$a_j^{e_i}$	be the actual arrival time at node j, given that node j is visited directly after node i, and that the actual arrival time at node i was e_i, where $i, j = \{0, 1, \ldots, N\}$
$a_j^{l_i}$	be the actual arrival time at node j, given that node j is visited directly after node i, and that the actual arrival time at node i was l_i, where $i, j = \{0, 1, \ldots, N\}$
TWC_{ij}	be the time window compatibility when node i is directly followed by node j

TWC_{ij} indicates the entry in row i, column j of the TWCM. Consider the following five scenarios illustrating the calculation of the time window compatibility. Each scenario assume customer j to be serviced directly after customer i, a service time of one hour, and a travel time of two hours from node i to node j.

Scenario 1: if $a_j^{e_i} > e_j$ and $a_j^{l_i} < l_j$, illustrated in figure 3.1. Customer i

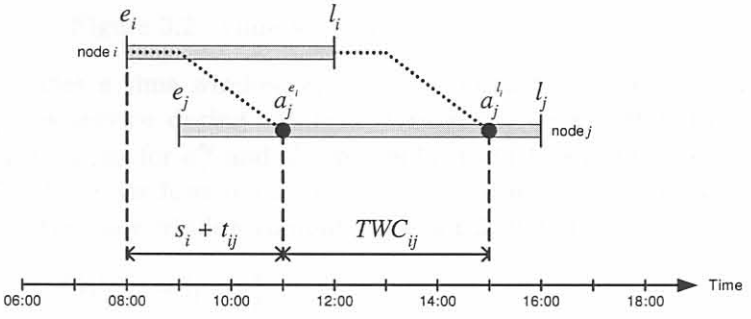

Figure 3.1: Time window compatibility scenario 1

specifies a time window $[e_i, l_i]$ = [08:00,12:00], while customer j requires service during the time window $[e_j, l_j]$ = [09:00,16:00]. If service

at customer i starts at the earliest allowed time, e_i, then the actual arrival time at customer j would be calculated as

$$a_j^{e_i} = e_i + s_i + t_{ij} \tag{3.1}$$

In this scenario $a_j^{e_i} = 11{:}00$. Similarly, $a_j^{l_i}$ would be the actual arrival time at customer j, given that the actual arrival time at customer i was l_i, and is calculated as

$$a_j^{l_i} = l_i + s_i + t_{ij} \tag{3.2}$$

The difference between $a_j^{e_i}$ and $a_j^{l_i}$ indicates the time window overlap between the two nodes. The time window compatibility is calculated as

$$TWC_{ij} = a_j^{l_i} - a_j^{e_i} \tag{3.3}$$

For this example, the time window compatibility is four hours (04:00).

Scenario 2: if $a_j^{e_i} > e_j$ **and** $a_j^{l_i} > l_j$, illustrated in figure 3.2. Customer i

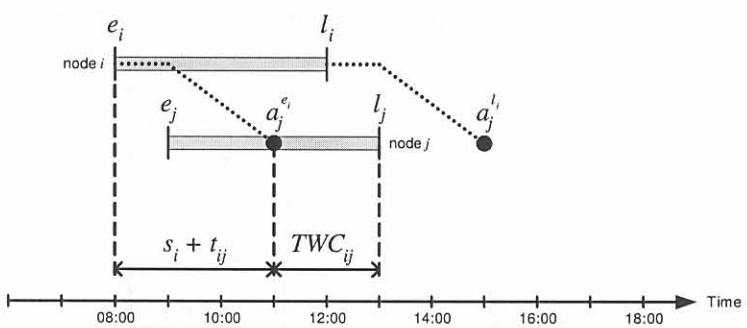

Figure 3.2: Time window compatibility scenario 2

specifies a time window $[e_i, l_i] = [08{:}00,12{:}00]$, while customer j requires service during the time window $[e_j, l_j] = [09{:}00,13{:}00]$. The calculations for $a_j^{e_i}$ and $a_j^{l_i}$ are similar to (3.1) and (3.2), respectively. The time windows of customer i and customer j only partly overlap, and the time window compatibility is calculated as

$$TWC_{ij} = l_j - a_j^{e_i} \tag{3.4}$$

For this example, the time window compatibility is two hours (02:00).

Scenario 3: if $a_j^{e_i} < e_j$ **and** $a_j^{l_i} < l_j$, illustrated in figure 3.3. Customer i specifies a time window $[e_i, l_i] = [08{:}00,12{:}00]$, while customer j requires service during the time window $[e_j, l_j] = [12{:}00,16{:}00]$. The

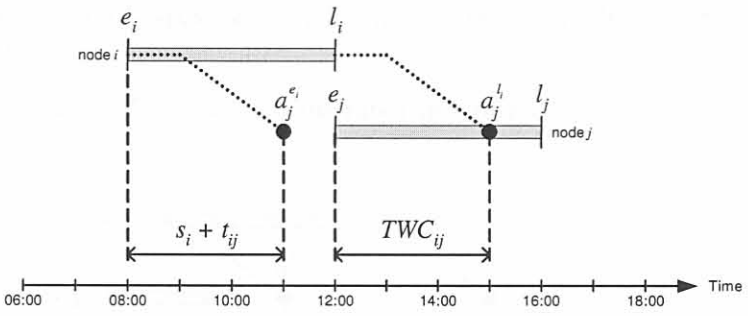

Figure 3.3: Time window compatibility scenario 3

calculations for $a_j^{e_i}$ and $a_j^{l_i}$ are similar to (3.1) and (3.2), respectively. The time windows of customer i and customer j only partly overlap, and the time window compatibility is calculated as

$$TWC_{ij} = a_j^{l_i} - e_j \tag{3.5}$$

For this example, the time window compatibility is three hours (03:00).

Scenario 4: if $a_j^{e_i}, a_j^{l_i} < e_j$, illustrated in figure 3.4. Customer i specifies

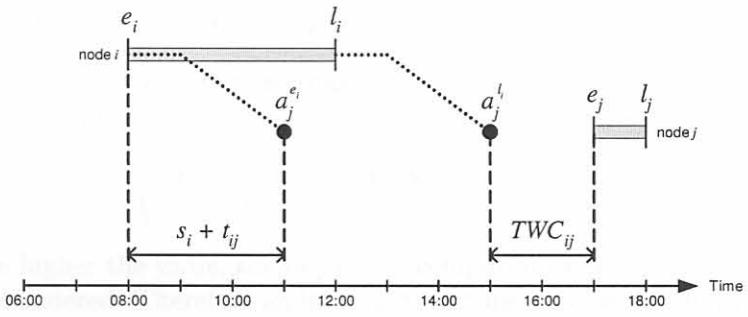

Figure 3.4: Time window compatibility scenario 4

a time window $[e_i, l_i]$ = [08:00,12:00], while customer j requires service during the time window $[e_j, l_j]$ = [17:00,18:00]. The calculations for $a_j^{e_i}$ and $a_j^{l_i}$ are similar to (3.1) and (3.2), respectively. The time windows of customer i and customer j do not overlap. Even if customer i is serviced as late as possible, l_i, a waiting time is incurred at customer j. The time window compatibility is calculated as

$$TWC_{ij} = a_j^{l_i} - e_j \tag{3.6}$$

For this example, the time window compatibility is negative two hours (-02:00). The significance of the negative time is that it is possible, in

this case, to service customer j after customer i, although the waiting time is penalized.

Scenario 5: if $a_j^{e_i}, a_j^{l_i} > l_j$, illustrated in figure 3.5. Customer i specifies

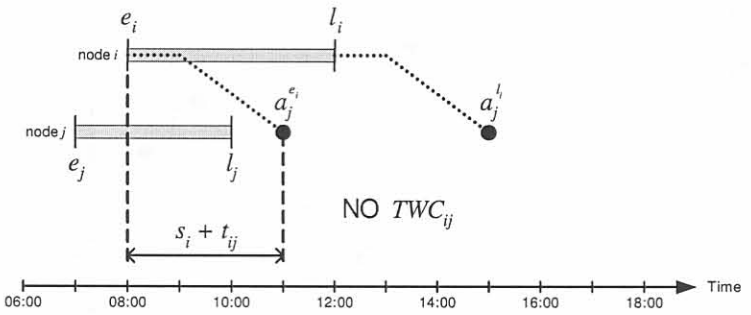

Figure 3.5: Time window compatibility scenario 5

a time window $[e_i, l_i]$ = [08:00,12:00], while customer j requires service during the time window $[e_j, l_j]$ = [07:00,11:00]. The calculations for $a_j^{e_i}$ and $a_j^{l_i}$ are similar to (3.1) and (3.2), respectively. Although the time windows of customer i and customer j partly overlap, it is impossible to service customer j, even if customer i is serviced as early as possible, e_i. Therefor, no time window compatibility exist.

A generalized equation is proposed that will address all five scenarios illustrated, and is given as

$$TWC_{ij} = \begin{cases} \min\{a_j^{l_i}, l_j\} - \max\{a_j^{e_i}, e_j\}, \text{ if } l_j - a_j^{e_i} > 0 \\ -\infty \text{ otherwise} \end{cases} \tag{3.7}$$

The higher the value, the better the compatibility of the two time windows considered. Therefore an incompatible time window is defined to have a compatibility of negative infinity.

Example. Consider the following example with five nodes geographical distributed around a depot in figure 3.6. In the example, node c has indicated two possible time windows. As discussed in section 2.3.1, the customer is artificially split and treated as two separate nodes, c^1 and c^2, respectively. The time windows for each customer, including the depot, as well as the service time at each node, are given in table 3.1. The distance matrix, $\overline{D}$, is calculated using the rectangular distance

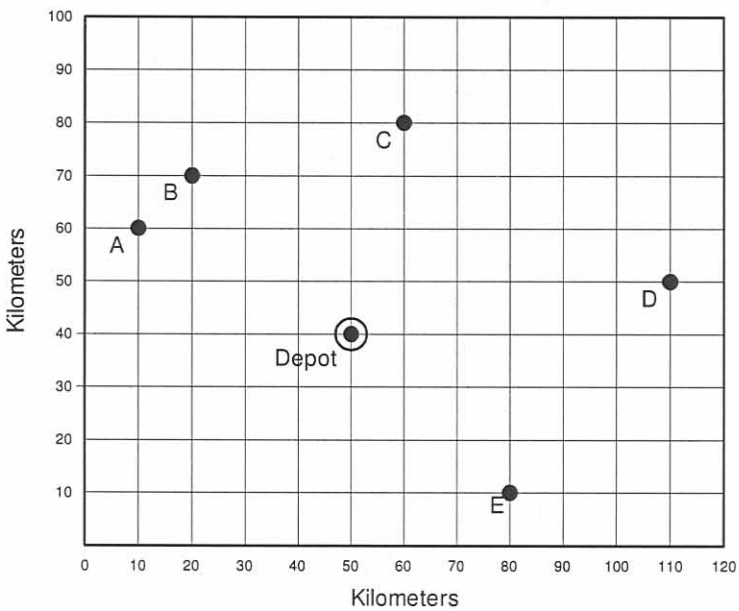

Figure 3.6: Geographical distribution of nodes around a depot

Table 3.1: Time windows and service times

Node (i)	**Time window** $(e_i; l_i)$	**Service time (in hours)** s_i
Depot	07:00 – 18:00	0.00
a	08:00 – 12:00	0.50
b	11:00 – 13:00	0.25
c^1	08:00 – 09:00	0.25
c^2	15:00 – 17:00	0.25
d	08:00 – 12:00	0.50
e	10:00 – 15:00	0.25

between nodes, as taken from figure 3.6.

$$\overline{\mathrm{D}} = \begin{bmatrix} 0 & 60 & 60 & 50 & 50 & 70 & 60 \\ 60 & 0 & 20 & 70 & 70 & 110 & 120 \\ 60 & 20 & 0 & 50 & 50 & 110 & 120 \\ 50 & 70 & 50 & 0 & 0 & 80 & 90 \\ 50 & 70 & 50 & 0 & 0 & 80 & 90 \\ 70 & 110 & 110 & 80 & 80 & 0 & 70 \\ 60 & 120 & 120 & 90 & 90 & 70 & 0 \end{bmatrix}$$

If the average speed is known, the time matrix, $\overline{T}$, can be calculated. In this example, $\overline{T}$ is given. Values are given in hours.

$$\overline{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0.5 & 1 & 1 & 2 & 2 \\ 1 & 0.5 & 0 & 1 & 1 & 2 & 2 \\ 1 & 1 & 1 & 0 & 0 & 1.5 & 1.5 \\ 1 & 1 & 1 & 0 & 0 & 1.5 & 1.5 \\ 1 & 2 & 2 & 1.5 & 1.5 & 0 & 1 \\ 1 & 2 & 2 & 1.5 & 1.5 & 1 & 0 \end{bmatrix}$$

With the information at hand, the time window compatibility matrix can be calculated. For the given example,

$$\overline{\mathrm{TWCM}} = \begin{bmatrix} 11 & 4 & 2 & 1 & 2 & 4 & 5 \\ 4 & 3.5 & 2 & -\infty & -1.5 & 1.5 & 4 \\ 2 & 0.25 & 1.75 & -\infty & -0.75 & -\infty & 1.75 \\ 1 & 1 & -0.75 & 0.75 & -5.75 & 1 & 0.75 \\ 1.75 & -\infty & -\infty & -\infty & 1.75 & -\infty & -\infty \\ 4 & 1.5 & 2 & -\infty & -1 & 3.5 & 3.5 \\ 5 & -\infty & 0.75 & -\infty & 1.75 & 0.75 & 4.75 \end{bmatrix}$$

3.3 Using the time window compatibility matrix

The time window compatibility matrix, TWCM, is calculated before the route building heuristic is evoked.

3.3.1 Reducing computational complexity

Solomon [46] calculates the insertion and selection criteria by means of (2.17) and (2.25), respectively. In each iteration, these criteria are calculated for each edge on the partially constructed route, irrespective of the compatibility of the time window of the node considered for insertion with the time windows of the two nodes forming the edge.

This chapter presents an improved case. Consider the example where node u is considered for insertion between nodes i and j. As the TWCM is already calculated, it is possible to check the compatibility of node u with the routed nodes i and j. If either TWC_{iu} or TWC_{uj} is negative infinity $(-\infty)$, indicating an incompatible time window, the insertion heuristic moves on and considers the next edge, without wasting computational effort on calculating the insertion and selection criteria.

In the earlier example, eleven instances of infeasible time windows occur. If these instances are identified and eliminated, a computational saving in excess of 22% is achieved.

3.3.2 Identifying the seed customer

When looking at the TWCM for the example, it is clear that the incompatibility is distinct for specific nodes. It is therefor possible to identify *incompatible* nodes. As opposed to the two most common initialization criteria, namely *customer with earliest deadline*, and *furthest customer*, as suggested by Dullaert [19], this dissertation proposes the use of the TWCM to identify seed nodes based on their time window compatibility.

Table 3.2 indicates the number of instances where a node has an infeasible time window with another node, either as origin, or as destination. Both

Table 3.2: Number of infeasible time window instances

Node	Number of infeasible time windows, as origin	Number of infeasible time windows, as destination	Total
Depot	0	0	0
a	1	2	3
b	2	1	3
c^1	0	5	5
c^2	5	0	5
d	1	2	3
e	2	1	3

nodes c^1 and c^2 have five infeasible instances. The two artificial nodes are representing the same customer c. It can be concluded that customer c is the most incompatible node, and is therefore identified as the seed customer. Ties are broken arbitrarily. Should two nodes have the same number of infeasible time window instances, any of the two customers could be selected as seed customer.

It may be possible to not have any infeasible time window instances. In such a scenario, a *total compatibility* value can be determined for each node a, and is calculated as

$$\sum_{i=1,i\neq a}^{M} TWC_{ia} + \sum_{j=1,j\neq a}^{M} TWC_{aj} + TWC_{aa} \tag{3.8}$$

or

$$\sum_{i=1}^{M} TWC_{ia} + \sum_{j=1}^{M} TWC_{aj} - TWC_{aa} \tag{3.9}$$

where M refers to all the unrouted nodes, including all instances of those nodes that are split artificially. The customer with the lowest total compatibility is selected as seed customer.

3.4 Conclusion

A new concept, *Time Window Compatibility* (TWC) is introduced in this chapter. A matrix, referred to as the *Time Window Compatibility Matrix* (TWCM), is used as the mechanism to calculate the compatibility between all nodes in the network. A numerical example illustrates the use of the concept to reduce the computational burden of existing heuristics, and proposes an improved criteria for seed customer selection.

The time window compatibility is used in the next chapter when the formal model and initial solution algorithm is defined.