

## Article

# Vehicle Routing Problem with Time Windows to Minimize Total Completion Time in Home Healthcare Systems

Payakorn Saksuriya <sup>1</sup>  and Chulin Likasiri <sup>2,\*</sup> <sup>1</sup> PhD Degree Program in Mathematics, Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand; payakorn\_sak@cmu.ac.th<sup>2</sup> Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand

\* Correspondence: chulin.l@cmu.ac.th

**Abstract:** We propose a vehicle routing problem with time windows (VRPTW) with compatibility-matching constraints and total completion time as the objective function, with applications in home healthcare routing and scheduling. Mixed integer linear programming is provided with total completion time minimization as the objective function. The solution approach has two objectives, total completion time (primary objective) and total distance (secondary objective). A heuristic is proposed comprising three phases: initializing to find an initial feasible routing (inserting the procedure with a modified K-means algorithm), swapping and moving the procedure to find a local optimal routing, and shooting the procedure to move away from the local optimum. Proof of feasibility for the inserting procedure is provided to prevent unnecessary insertions. Phases 2 and 3 will be repeated as needed to ensure solution quality. Solving our model with the proposed heuristic algorithm increases the total distance by 90.00% but reduces the total completion time by 25.86%. To test our model and heuristic, we examined a system with 400 home-healthcare cases in Chiang Mai. The heuristic quickly solved the problem. When total completion time is minimized, some caretakers serve up to twice as many patients as their coworkers; when total distance is minimized, workload discrepancies can increase up to seven-fold.

**Keywords:** vehicle routing problem; time windows; compatibility; total completion time; home healthcare system

**MSC:** 90-08; 90-10

**Citation:** Saksuriya, P.; Likasiri, C. Vehicle Routing Problem with Time Windows to Minimize Total Completion Time in Home Healthcare Systems. *Mathematics* **2023**, *11*, 4846. <https://doi.org/10.3390/math11234846>

Academic Editor: Shih-Wei Lin

Received: 3 November 2023

Revised: 26 November 2023

Accepted: 29 November 2023

Published: 1 December 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

According to projections, the percentage of Thai nationals aged 65 years or older will surpass 28% in less than 30 years. This threshold is significant as it marks a country as a ‘super-aged country’ [1]. Projections by the Office of the National Economic and Social Development Board under the Office of the Prime Minister in Bangkok anticipate a rise in the population of elderly individuals living independently and experiencing health issues [2]. Home healthcare (HHC) is a crucial component of a proactive healthcare system, particularly for population segments who suffer from chronic illnesses or are bedridden and need long-term care [3]. Providing care for these individuals may involve treatments by psychiatrists, delivery of injections or wound care by nurses, and delivery of rehabilitation programs by physical therapists, as well as arrangement of health-related guidance. From the perspective of sustainability and quality-of-life enhancement, visiting the elderly at home to deliver care represents a more effective utilization of human resources compared to having them visit the hospital individually.

Chiang Mai is Thailand’s second largest city after Bangkok, in both population density and economic significance. Additionally, it is the primary medical hub of the country’s northern region. According to a recent study [4], Chiang Mai’s elderly population is currently estimated to be 20% of its total population. Approximately 38%,

or 675,440 individuals out of a total population of 1,779,254, reside within the city center, a similar percentage to contemporary trends in other major urban areas around the world. The urban area of the city of Chiang Mai has 310,771 households, and approximately 20% of the population have medical requirements and have duly enrolled themselves at one of the 15 healthcare facilities dispersed across the city. Assuming that 10% of the households registered for healthcare services require regular visits from healthcare personnel, it follows that each health center will be required to schedule appointments for approximately 400 clients, and this number is increasing over time. Trying to reduce the cost while managing the scheduling and routing of this care service requires specific strategies and techniques. According to a report by Glinskaya et al. [5], most elderly individuals in Thailand experience the aging process within their own households and receive care from their families. However, the conventional familial care structure faces growing challenges from the rise in the elderly population in relation to the working-age population. According to projections, the number of Thais aged 80 and above will increase more than six-fold, to approximately 2.5 million, over the next two decades. This group will require assistance.

The growing need for aged care services offers potential for business investment and employment opportunities. It is essential to have the ability to address problems in the provision of affordable and adequate care for elderly individuals with long-term care requirements, covering care delivered at home, within communities, and in institutional settings. From a financial perspective, the total cost must be minimized, but home healthcare is a people-centered business and focusing on cost reduction alone can result in unsatisfied staff and retention problems. One approach to solving this dilemma is to consider multiple objectives simultaneously. The contributions of this paper are to better capture the nature of the home visits, improve caretakers' workflow efficiency, and suggest a better solution to routing and scheduling. We consider models that cover all four major aspects of these undertakings: (1) compatibility matching of patients and healthcare staff (compatibility), (2) routing for healthcare staff to visit their clients (routing), (3) meeting applicable service conditions, such as the time availability of patients and staff (time windows), and (4) the total costs of the whole system (costs). We focus on routing and scheduling and incorporate all four major aspects of home healthcare. The objective function is set to only minimize the total completion time exclusively rather than minimizing total distance to the exclusion of all else. Moreover, we provide single objective model formulation based on vehicle routing problems with time windows and a solution approach that focuses primarily on total completion time followed by total distance. Our model still minimizes the single objective function, but the proposed solution method primarily minimizes total completion time, while also trying to minimize total distance if possible. The reason is to prevent the possibility that there exists a solution that has the same total completion time but a different total distance. The solution approach contains routing operators modified to suit the objective function as well as a proof to ensure feasibility, which will reduce the overall computational time.

A VRPTW problem in any of its variations falls in the NP-hard category, requiring the use of customized heuristics in its resolution. A home healthcare system presents a highly complicated problem with numerous factors to be considered. As much of the existing literature on the vehicle routing problem with time windows (VRPTW) is focused on minimizing the distance travelled, we decided to develop a model that prioritizes the minimization of total completion time. This alternative would enhance the model's suitability for real-world applications, as waiting-time minimization and a balanced allocation of jobs per vehicle are significant in the context of VRPTW. The process of assigning a healthcare worker to a client requires that such factors as arrival time, finishing time, and processing time be taken into consideration. The objective function can include various components such as travel cost, penalty cost, fixed cost, and staff overtime cost. Among the various objective functions considered, Wirnitzer et al. [6] employed Gurobi optimization version 5.6 to minimize the number of nurses assigned to each tour and per patient, as well as the number of nurse switches per patient. Nasir and Kuo [7] employed a genetic algorithm

to minimize the combined costs of travel and staff assignment. In contrast, Kandakoglu et al. [8] aimed to minimize a weighted sum of various factors, including the total distance travelled by nurses, overall travel expense, and overtime expense, as well as the maximum workload and number of working staff, in order to achieve a balanced distribution of workload. The problem is solved using the Gurobi optimization software version 8.0. In their research, Braekers et al. [9] studied the minimization of two objectives; the first objective applies to minimizing the total travel costs accrued by nurses and the associated overtime costs, while the second involves minimizing the combined penalty cost resulting from assigning an undesirable nurse to a patient and the inconvenience scores associated with nurses' job assignments. The problem is solved using a large neighborhood search algorithm. The authors Cissé et al. [10] and Mascolo et al. [11] provided comprehensive analyses on state-of-the-art HHC routing and scheduling problems by studying the literature on home visits in HHC systems. Goodarzian et al. [12] provided state-of-the-art operations research including models and applications in home healthcare systems.

The VRPTW has garnered significant attention in the research community due to its practical relevance and applicability to real-world scenarios. Numerous heuristic approaches have been extensively investigated, and researchers remain captivated by them. Of the recently suggested techniques for addressing the VRPTW, which are above the complexity of the NP-hard VRP, the majority fall under the category of metaheuristics. The benchmark instances provided in reference [13], commonly referred to as Solomon's instances, can be utilized for evaluating constructed heuristics. The dataset consists of a total of 100 customers and one depot node. Additionally, a distance matrix is provided, representing the distances between each pair of nodes. Every individual customer node is assigned a specific time window, demand, and service time. The primary goal is to minimize the total distance travelled. There are three categories of Solomon's instances. After plotting these instances, the customers in Group C instances are dispersed in groups or clusters. These instances are similar to real-world problems where customers live in geographically scattered villages. Group R instances have relatively random placement, corresponding to customers living in a large city. The customers in Group RC are randomly clustered, i.e., they appear in groups but their placement within these groups is more or less random.

The aforementioned properties can be seen in a conventional system in the real world. The application and utility of VRPTW in addressing practical problems have led to significant research efforts in developing heuristics for its solution, and a substantial number of researchers remain engaged in investigating and advancing the field of VRPTW heuristics. A model was proposed by Ahn and Shin [14] for the purpose of minimizing the total distance in the vehicle routing problem with time windows (VRPTW). This model is based on the utilization of insertion and saving heuristics. Rochat and Taillard [15] introduced an algorithm that relied on a probabilistic tabu search and used probabilistic techniques to enhance the effectiveness of the tabu search. Additionally, they presented the solutions obtained with their algorithm. Li and Lim [16] implemented a metaheuristic approach using the simulated annealing algorithm and incorporated K-restart strategies as a means of enhancing the local search process. The performance of their algorithm, evaluated in the Solomon benchmark instances, shows that it has the potential to enhance existing solutions reported in the literature. Following this, Pisinger and Popke [17] introduced the concept of an adaptive large neighborhood search (ALNS) and applied it to evaluate Solomon instances, as well as Gehring and Homberger benchmark instances [18], specifically focusing on scenarios involving over 100 customers. Low et al. [19] employed a genetic algorithm and conducted tests on the modified Solomon instances. The problem was adapted to align with the specific context of production and scheduling. After that, Khoo and Mohammad [20] proposed the parallelization of ruin and recreated strategies and the use of a genetic algorithm as a means to address the vehicle routing problem with time windows (VRPTW). Addressing a variant of multiple depots vehicle routing problem,

Bi et al. [21] applied the ant lion optimizer to solve two types of logistic modes in meal delivery routing with two objectives, customer satisfaction and cost.

Table 1 presents the minimum values of the objective function obtained from the most recent literature. Note that only works that consider hard time windows, i.e., the service providers need to wait for customers to become available, are presented. The problem type is denoted by the letters C, R, or RC in the first column, representing instances with nodes in clusters, randomly placed nodes, and clusters with randomly placed nodes inside each cluster, respectively. The objective function values obtained from the most recent literature are shown in Table 1. The first value in the table represents the total distance, followed by the required number of vehicles (in brackets). Note that some corresponding solutions are unavailable or inaccessible. We have underlined the works that provide solutions based on objective function values. The optimal results, according to the authors, are marked with “\*” next to the objective value. For each instance, the minimum objective value (total distance) across all publications, regardless of the associated solutions, is shown in boldface. It is notable that all works with corresponding solutions have objective function values that are relatively higher. Note also that some of the work may truncate the distance between a node to one digit, resulting in a lower objective value than others.

**Table 1.** The best objective function values from the literature on Solomon’s instances.

| Instance | [15]         | [22]          | [23]          | [24]          | [25]         | [26]         | [27]         | [20]         | [28]        |
|----------|--------------|---------------|---------------|---------------|--------------|--------------|--------------|--------------|-------------|
| C101     | s            | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C102     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C103     | 828.06 (10)  | 826.3 * (10)  | -             | 826.3 * (10)  | 828.06 (10)  | 828.06 (10)  | 828.06 (10)  | 828.07 (10)  | 826.3 (10)  |
| C104     | 824.78 (10)  | 822.9 * (10)  | -             | 822.9 * (10)  | 824.78 (10)  | 825.65 (10)  | 824.78 (10)  | 824.78 (10)  | 822.9 (10)  |
| C105     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C106     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C107     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C108     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C109     | 828.94 (10)  | 827.3 * (10)  | -             | 827.3 * (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 828.94 (10)  | 827.3 (10)  |
| C201     | 591.56 (3)   | -             | 589.1 * (-)   | 589.1 * (3)   | 591.56 (3)   | 591.56 (3)   | 591.56 (3)   | 591.56 (3)   | 589.1 (3)   |
| C202     | 591.56 (3)   | -             | 589.1 * (-)   | 589.1 * (3)   | 591.56 (3)   | 591.56 (3)   | 591.56 (3)   | 591.56 (3)   | 589.1 (3)   |
| C203     | 591.17 (3)   | -             | -             | 588.7 * (3)   | 591.17 (3)   | 591.17 (3)   | 591.17 (3)   | 591.17 (3)   | 588.7 (3)   |
| C204     | 590.60 (3)   | -             | -             | -             | 590.60 (3)   | 596.55 (3)   | 590.60 (3)   | 590.60 (3)   | 588.1 (3)   |
| C205     | 588.88 (3)   | -             | 586.4 * (-)   | -             | 588.88 (3)   | 588.88 (3)   | 588.88 (3)   | 588.88 (3)   | 586.4 (3)   |
| C206     | 588.49 (3)   | -             | 586.0 * (-)   | 586.4 * (3)   | 588.49 (3)   | 588.49 (3)   | 588.49 (3)   | 588.49 (3)   | 586.0 (3)   |
| C207     | 588.29 (3)   | -             | 585.8 * (-)   | 586.0 * (3)   | 588.29 (3)   | 588.29 (3)   | 588.29 (3)   | 588.29 (3)   | 585.8 (3)   |
| C208     | 588.32 (3)   | -             | -             | 585.8 * (3)   | 588.32 (3)   | 588.32 (3)   | 588.32 (3)   | 588.32 (3)   | 585.8 (3)   |
| R101     | 1650.80 (19) | 1637.7 * (20) | 1637.7 * (20) | 1637.7 * (20) | 1642.88 (20) | 1664.13 (20) | 1642.87 (20) | 1642.88 (20) | 1637.7 (20) |
| R102     | 1486.12 (17) | 1466.6 * (18) | 1466.6 * (18) | 1466.6 * (18) | 1472.81 (18) | 1487.07 (18) | 1472.62 (18) | 1472.82 (18) | 1467.7 (18) |
| R103     | 1213.62 (14) | 1208.7 * (14) | 1208.7 * (14) | 1208.7 * (14) | 1213.62 (14) | 1237.05 (14) | 1213.62 (14) | 1213.62 (14) | 1220.3 (15) |
| R104     | 982.01 (10)  | -             | -             | -             | 976.61 (11)  | 1010.24 (11) | 986.10 (11)  | 976.61 (11)  | 984.5 (10)  |
| R105     | 1377.11 (14) | 1355.3 * (15) | 1355.3 * (15) | 1355.3 * (15) | 1360.78 (15) | 1390.12 (15) | 1360.78 (15) | 1360.78 (15) | 1373.1 (16) |
| R106     | 1252.03 (12) | 1234.6 * (13) | 1234.6 * (13) | 1234.6 * (13) | 1240.47 (13) | 1254.22 (13) | 1241.52 (13) | 1239.37 (13) | 1259.3 (14) |
| R107     | 1159.86 (10) | 1064.6 * (11) | 1064.6 * (11) | 1064.6 * (11) | 1073.34 (11) | 1100.52 (11) | 1076.13 (11) | 1072.12 (11) | 1084.6 (12) |
| R108     | 980.95 (9)   | -             | -             | -             | 947.55 (10)  | 975.34 (10)  | 948.57 (10)  | 938.20 (10)  | 952.30 (11) |
| R109     | 1235.68 (11) | -             | 1146.9 * (13) | 1146.9 * (13) | 1151.84 (13) | 1166.09 (13) | 1151.84 (13) | 1151.84 (13) | 1165.9 (13) |
| R110     | 1080.36 (11) | -             | 1068.0 * (12) | 1068.0 * (12) | 1072.41 (12) | 1112.21 (11) | 1092.35 (12) | 1072.41 (12) | 1091.2 (12) |
| R111     | 1129.88 (10) | -             | 1048.7 * (12) | 1048.7 * (12) | 1053.50 (12) | 1079.82 (12) | 1053.50 (12) | 1053.50 (12) | 1065.3 (12) |
| R112     | 953.63 (10)  | -             | -             | -             | 953.63 (10)  | 976.99 (10)  | 960.68 (10)  | 958.03 (10)  | 971.80 (11) |
| R201     | 1281.58 (4)  | -             | -             | 1143.2 * (8)  | 1149.68 (9)  | 1173.75 (7)  | 1148.49 (9)  | 1147.80 (8)  | 1146.6 (8)  |
| R202     | 1088.07 (4)  | -             | -             | -             | 1034.35 (8)  | 1046.16 (5)  | 1049.74 (7)  | 1034.97 (6)  | 1035.8 (7)  |
| R203     | 948.74 (3)   | -             | -             | -             | 874.87 (6)   | 890.50 (5)   | 900.08 (5)   | 874.87 (6)   | 877.00 (6)  |
| R204     | 869.29 (2)   | -             | -             | -             | 736.52 (4)   | 760.82 (3)   | 772.33 (4)   | 735.80 (5)   | 742.40 (5)  |
| R205     | 1063.24 (3)  | -             | -             | -             | 955.82 (5)   | 954.16 (5)   | 970.89 (6)   | 954.16 (5)   | 957.20 (5)  |
| R206     | 912.97 (3)   | -             | -             | -             | 879.86 (4)   | 889.39 (4)   | 898.91 (5)   | 884.85 (5)   | 894.40 (6)  |
| R207     | 814.78 (3)   | -             | -             | -             | 799.86 (4)   | 822.90 (4)   | 834.93 (4)   | 797.99 (4)   | 808.60 (5)  |
| R208     | 738.60 (2)   | -             | -             | -             | 705.45 (4)   | 719.17 (3)   | 723.61 (3)   | 705.33 (4)   | 718.70 (4)  |
| R209     | 944.64 (3)   | -             | -             | -             | 859.39 (5)   | 874.95 (5)   | 879.51 (6)   | 860.11 (5)   | 870.30 (5)  |
| R210     | 967.50 (3)   | -             | -             | -             | 910.70 (5)   | 930.42 (5)   | 932.89 (7)   | 905.21 (6)   | 916.00 (6)  |
| R211     | 949.49 (2)   | -             | -             | -             | 755.96 (4)   | 761.10 (4)   | 787.51 (5)   | 753.15 (4)   | 758.60 (5)  |

Table 1. Cont.

| Instance | [15]         | [22] | [23]          | [24]                | [25]         | [26]         | [27]         | [20]         | [28]        |
|----------|--------------|------|---------------|---------------------|--------------|--------------|--------------|--------------|-------------|
| RC101    | 1623.58 (15) | -    | 1619.8 * (14) | 1619.8 * (14)       | 1643.41 (16) | 1636.92 (15) | 1639.97 (16) | 1623.58 (15) | 1647.5 (16) |
| RC102    | 1477.54 (13) | -    | 1457.4 * (14) | 1457.5 * (14)       | 1461.23 (14) | 1488.36 (14) | 1466.84 (14) | 1461.23 (14) | 1473.5 (14) |
| RC103    | 1262.02 (11) | -    | 1258.0 * (11) | 1258.2 * (11)       | 1277.54 (12) | 1306.42 (14) | 1264.71 (11) | 1261.67 (11) | 1282.5 (12) |
| RC104    | 1135.83 (10) | -    | -             | -                   | 1136.81 (10) | 1140.70 (10) | 1135.52 (10) | 1135.52 (10) | 1159.2 (11) |
| RC105    | 1733.56 (13) | -    | 1513.7 * (15) | 1513.7 * (15)       | 1518.58 (16) | 1590.25 (16) | 1518.60 (16) | 1518.58 (16) | 1554.9 (15) |
| RC106    | 1384.92 (12) | -    | -             | -                   | 1381.23 (13) | 1408.70 (13) | 1377.35 (16) | 1376.99 (13) | 1398.2 (14) |
| RC107    | 1230.95 (11) | -    | -             | -                   | 1212.83 (12) | 1254.26 (12) | 1212.83 (12) | 1212.83 (12) | 1251.0 (12) |
| RC108    | 1170.70 (10) | -    | -             | -                   | 1117.53 (11) | 1254.26 (12) | 1117.53 (11) | 1118.07 (11) | 1132.5 (11) |
| RC201    | 1438.89 (4)  | -    | -             | <u>1261.8 * (9)</u> | 1265.56 (9)  | 1306.34 (7)  | 1274.54 (9)  | 1265.56 (9)  | 1268.8 (9)  |
| RC202    | 1165.57 (4)  | -    | -             | -                   | 1095.64 (8)  | 1118.05 (8)  | 1113.53 (8)  | 1095.64 (8)  | 1096.3 (8)  |
| RC203    | 1079.57 (3)  | -    | -             | -                   | 928.51 (5)   | 951.08 (5)   | 945.96 (5)   | 926.82 (5)   | 934.40 (5)  |
| RC204    | 806.75 (3)   | -    | -             | -                   | 786.38 (4)   | 796.14 (4)   | 799.67 (4)   | 788.66 (4)   | 793.60 (4)  |
| RC205    | 1333.71 (4)  | -    | -             | -                   | 1157.55 (7)  | 1181.86 (7)  | 1161.81 (7)  | 1157.55 (7)  | 1162.7 (8)  |
| RC206    | 1212.64 (3)  | -    | -             | -                   | 1054.61 (7)  | 1080.50 (7)  | 1059.89 (7)  | 1054.61 (7)  | 1070.2 (7)  |
| RC207    | 1085.61 (3)  | -    | -             | -                   | 966.08 (6)   | 982.58 (5)   | 976.40 (7)   | 969.80 (6)   | 967.70 (6)  |
| RC208    | 833.97 (3)   | -    | -             | -                   | 779.31 (4)   | 785.93 (4)   | 795.40 (5)   | 778.93 (4)   | 778.90 (4)  |

\* is optimal results according to the authors. Underline is the works that provide the solutions based on the objective function value.

The remaining sections of this paper are structured as follows: Section 2 presents model construction, with the model being modified from the vehicle routing problem with time windows. Section 3 describes the solution approach. The results and simulation analysis are presented in Section 4. The case study is shown Section 5. Section 6 presents the conclusions.

## 2. Model Construction

The routing and scheduling in a home healthcare system (HHC) is based on the vehicle routing problem, where the vehicles visit all customer nodes. Each node can be visited by only one vehicle within the time frame. This particular problem involves a collection of vehicles (caretakers), denoted as  $\{1, 2, \dots, m\}$ , and a set of time intervals associated with each caretaker. The main goal is to determine the optimal routes for each caretaker, starting and ending at the depot location, in order to minimize the total completion time required to complete all tasks. Each patient is set to receive a single visit within a designated time frame, and each caretaker is obligated to start and finish their duties within the specified window.

The concept in a HHC system can be represented mathematically as a complete graph denoted as  $G = (N, E)$ . In this graph,  $N$  represents the set of clients or patients, ranging from 0 to  $n$ , written as  $N = \{0, 1, \dots, n\}$ . The value 0 corresponds to the starting location of the route (depot node), which is the health center.  $E$  represents the set of edges in the graph, indicating the connections between different clients or patients. The provided data include the distances (duration of travel between each pair of points), as well as the amount of time required to deliver service at the respective point. The compatibility for visiting between the nodes and vehicles is given, which takes the value 1 if the caretaker is allowed to visit the patient, and otherwise, 0. The mathematical programming problem is formulated as a vehicle routing problem with time windows (VRPTW) including strict limitations on time, inspired by situations where patients have time-sensitive medical requirements and caretakers who arrive before the designated time windows will be required to wait.

The parameters used in the HHC system are as follows:

Indices:

- 0 The starting point of the route, which can be the health center
- $i, j$  The patients  $i, j \in \{1, 2, \dots, n\}$
- $k$  The caretaker  $k \in \{1, 2, \dots, m\}$

## Parameters:

|                                        |                                                                                                                                                        |
|----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| $t_{ij}$                               | The travel time from patient $i$ to patient $j$                                                                                                        |
| $p_j$                                  | The service time for patient $j$                                                                                                                       |
| $e_j$                                  | The start time (or available time) of patient $j$                                                                                                      |
| $l_j$                                  | The finish time of patient $j$                                                                                                                         |
| $q_j$                                  | The requirement of patient $j$                                                                                                                         |
| $Q$                                    | The capacity of all vehicles                                                                                                                           |
| $M$                                    | A sufficiently large number                                                                                                                            |
| $\begin{bmatrix} r_{jk} \end{bmatrix}$ | A compatibility matching matrix ( $n \times m$ ) whose element $r_{jk} = 1$ means patient $j$ can be served by caretaker $k$ ; otherwise, $r_{jk} = 0$ |

## Variables:

|            |                                                                              |
|------------|------------------------------------------------------------------------------|
| $x_{0j}^k$ | 1 if caretaker $k$ travels from the origin to patient $j$ ; otherwise, 0     |
| $x_{ij}^k$ | 1 if caretaker $k$ travels from node $i$ to patient $j$ ; otherwise, 0       |
| $x_{j0}^k$ | 1 if patient $j$ is the last patient visited by caretaker $k$ ; otherwise, 0 |
| $t_j$      | Starting (available) time of patient $j$                                     |
| $c_j$      | Completion time of patient $j$                                               |
| $u_j$      | Variables for subtour elimination                                            |

The problem, which can be formulated as a mixed integer linear program, is given below. The problem aims to minimize total completion time, with each caretaker following restrictions, including routing and time constraints.

$$\text{minimize } \sum_{j=0}^n c_j \quad (1)$$

The routing constraints:

$$\sum_{j=1}^n x_{0j}^k = 1, \forall k \in K \quad (2)$$

$$\sum_{j=1}^n x_{j0}^k = 1, \forall k \in K \quad (3)$$

$$\sum_{k=1}^m \sum_{i=0, i \neq j}^n x_{ij}^k = 1, \forall j \in N \setminus \{0\} \quad (4)$$

$$\sum_{k=1}^m \sum_{j=0, j \neq i}^n x_{ij}^k = 1, \forall i \in N \setminus \{0\} \quad (5)$$

$$\sum_{i=0, i \neq j}^n x_{ij}^k - \sum_{l=0, l \neq j}^n x_{jl}^k = 0, \forall j \in N \setminus \{0\}, \forall k \in K \quad (6)$$

The subtour elimination constraint combined with capacity constraints:

$$u_i - u_j + q_j \leq Q(1 - x_{ij}^k), \forall i, j \in N \setminus \{0\}, i \neq j, \forall k \in K \quad (7)$$

$$q_j \leq u_j \leq Q, \forall j \in N \setminus \{0\} \quad (8)$$

The compatibility constraints:

$$\sum_{i=0, i \neq j}^n x_{ij}^k \leq r_{jk}, \forall j \in N, \forall k \in K \quad (9)$$

The starting time and time windows constraints:

$$t_i + p_i + d_{ij} - M(1 - x_{ij}^k) \leq t_j, \forall i \in N, i \neq j, \forall j \in N, \forall k \in K \quad (10)$$

$$e_j \leq t_j \leq l_j, \forall j \in N \quad (11)$$

The calculation of completion time:

$$t_i + p_i \leq c_i, \forall i \in N, \forall k \in K \quad (12)$$

The variable constraints:

$$x_{ij}^k \in \{0, 1\}, \forall i, j \in N, i \neq j, \forall k \in K \quad (13)$$

A detailed description of the constraints is as follows: routing constraints (2)–(8) relate to the organization of caretaker routes. Constraints (2) and (3) guarantee that each caretaker starts and ends their route at the designated starting point. Constraints (4) and (5) ensure that each patient is visited and departed by only one caretaker. Constraint (6) ensures that a caretaker departs from a patient after completing their service and proceeds to the next patient. Constraints (7) and (8) are designed to eliminate subtours. Constraints (9) are compatibility-matching constraints limiting the caretaker's ability to travel only to particular patients. The activation of constraints (10) occurs when caretaker  $k$  switches from going to patient  $i$  to going to patient  $j$  ( $x_{ij}^k = 1$ ). In other words, the sum of the duration of patient  $i$ 's completion time (or the expected start time and service duration at patient  $i$ ) and the travel time from patient  $i$  to patient  $j$  must be shorter than the starting time window for patient  $j$ . Constraints (11) make sure that the start time of patient  $j$  falls within his/her specific time window. Constraints (12) apply to the calculation of the completion times for all patients. Lastly, constraints (13) guarantee the requirements of integrality and positivity.

### 3. Solution Approach

As mentioned earlier, VRPTW is an extension of the vehicle routing problem which lies in the NP-hard problem category; as such, VRPTW is also an NP-hard problem. Given the NP-hard nature of the proposed problem, the exact approach cannot solve or even provide a solution when the number of nodes increases.

Our proposed algorithm begins with Phase 1, where the jobs are partitioned into clusters using a K-means algorithm. The number of clusters is predefined. Subsequently, within each cluster, the insertion procedure is employed to organize jobs in a manner that ensures there are no cases of tardiness or lateness among the assigned jobs. Positions that are unable to take on any job candidates will be considered tardy. Starting with two clusters and increasing when the inserting procedure cannot find a feasible solution, Phase 1 is iterated until all remaining tasks have been allocated; otherwise, the problem is infeasible. Phase 2 is subsequently implemented to enhance the solution by reassigning tasks (moving and swapping) either within or between clusters to achieve a more optimal outcome. When the process of swapping and moving does not lead to an improvement in the solution, it is referred to as a local optimal state for the current solution. Phase 3 involves implementing a shooting procedure, which aims to enhance the solution by randomly repositioning jobs before the next use of Phase 2. The implementation of Phase 3 is conditional upon obtaining the designated number of iterations. The procedures are summarized in Section 3 and described in detail in the following section.

As presented in Section 2, our model is a single objective function, which aims to exclusively minimize the total completion time. Since total completion time is the combination of the starting time, service time, and distance between nodes, minimizing the total completion time can result in a reduction in the total distance. In contrast, minimizing the total distance does not take the service time into account. Minimizing the total completion time will therefore result in simultaneous reduction of both operation time and distance. Moreover, minimizing only the total completion time obviously increases the total distance (minimizing one objective leads to an increase in another objective), which in turn will increase the overall cost of the system. For this reason, we modify the algorithm to consider more than one aspect rather than turn it into a problem with multiple objective functions.

Note that this approach can be reversed, i.e., using total distance as the primary objective function and total completion time as the secondary objective function.

The remainder of this section is organized as follows: the solution representation is given in Section 3.1. The modified K-means clustering is described in Section 3.2. Section 3.3 explains the inserting procedure. The local search procedures (swapping and moving procedures) are given in Sections 3.4 and 3.5. Finally, the algorithm, including Phases 1, 2, and 3, is summarized.

### 3.1. Solution Representation

For convenience's sake, the solution in this problem is defined as a single vector which starts and ends at node "0". Next to node 0 is the subsequent node that the vehicle travels to in the sequence. The representation contains all routes where each vehicle is separated by "0". For example, suppose the solution has the first vehicle start from the depot node and travel to node 1, 5, and 4, respectively, before returning to the depot node; the second route starts from the depot and travels to node 6 and 2; and the third route starts from the depot, goes to node 3, and then returns to the depot node. The solution which has two vehicles can be written as  $[0, 1, 5, 4, 0, 6, 2, 0, 6, 0]$ , as shown in Figure 1.

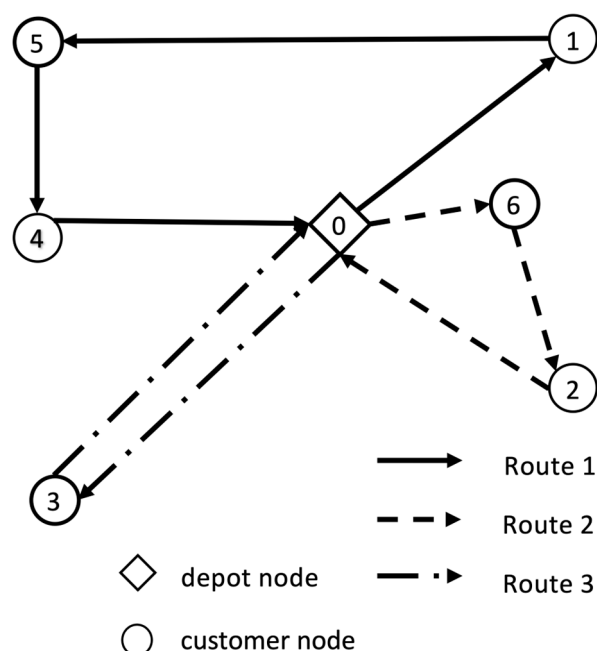


Figure 1. Example of the routing.

**Definition 1.** The routing sequence with  $k$  vehicles can be written as  $R = [R_1, R_2, \dots, R_k]$ , where each route  $i$  is represented by  $R_i = [0, \sigma_1^i, \sigma_2^i, \dots, \sigma_{|R_i|}^i, 0]$ , and  $|R_i|$  represents the number of visiting nodes, excluding depot, in route  $i$ .

**Definition 2.** The starting time of the job,  $t(\sigma_j^i)$ , is the minimum between the combination of the starting time of the immediately processed job with the service time and the travel distance, i.e.,  $t(\sigma_j^i) = t(\sigma_{j-1}^i) + p_{\sigma_{j-1}^i} + d_{\sigma_{j-1}^i, \sigma_j^i}$ , and the lower time window of the jobs.

**Definition 3.** The completion time of a job  $\sigma_j^i$ , written as  $c(\sigma_j^i)$ , is the combination of its starting time and its service time, i.e.,  $c(\sigma_j^i) = t(\sigma_j^i) + p_{\sigma_j^i}$ .

**Example 1.** The routing sequence is as shown in Figure 1, with 3 routes and 6 nodes. The first vehicle starts from the depot and travels to nodes 1, 5, and 4, and then returns to depot. Vehicle 2 starts from the depot and travels to node 6 and node 2. Vehicle 3 travels from the depot node, goes to node 3, and then returns to the depot. The routes can be written as follows:  $R_1 = [0, 1, 5, 4, 0]$ ,  $R_2 = [0, 6, 2, 0]$ ,  $R_3 = [0, 3, 0]$ . The routing sequence can be written as  $R = [R_1, R_2, R_3] = [0, 1, 5, 4, 0, 6, 2, 0, 3, 0]$ . Note that the duplicate zero in the routing sequence is deleted. From the definition, the three routes can be described as:

Route  $R_1$ ,  $\sigma_1^1 = 1, \sigma_2^1 = 5, \sigma_3^1 = 4$ , and  $|R_1| = 3$ .  
 Route  $R_2$ ,  $\sigma_1^2 = 6, \sigma_2^2 = 2$ , and  $|R_2| = 2$ .  
 Route  $R_3$ ,  $\sigma_1^3 = 3$  and  $|R_3| = 1$ .

### 3.2. Modified K-Means Clustering

K-means clustering is a classification method for classifying data by identifying the features. Classification methods can be divided into hierarchical clustering and partitioned clustering. In hierarchical clustering, data are clustered using a dendrogram of an agglomerative method and a divisive method. The partitioned clustering method clusters data based on the heuristic approach using a single criteria function (feature function) to all data objects [29]. In this study, the K-means algorithm, a partitioned clustering method, is selected for clustering the nodes based on the distance function. The K-means algorithm is selected because it is one of the most common methods for clustering, and because of its simple calculation, ease of implementation, and efficiency.

The procedure of K-means proceeds as follows: the number of clusters, called  $k$ , must be predefined before the start of algorithm. The  $k$  centroids are then generated, each one representing a cluster. After that, each remaining node is added to the nearest compatible cluster. The process will iterate by changing the centroids and recalculating the distance again until there is no change in the centroids. Due to the objective function to minimize total completion time and the time window constraints, we present a modified K-means clustering which changes the feature function from distance to the average upper time windows, indicated as (1) and (2), respectively, in the pseudocode. The first version is the original version of K-means, while the second version is the modified version. The pseudocode is given below (Algorithm 1).

---

#### Algorithm 1: Modified K-means clustering

---

1. **Input:** The set of all nodes  $N_0 = \{1, 2, \dots, n\}$ , the number of clusters ( $k$ ).
  2. Randomly generate  $k$  nodes within the space, set as centroids  $K_c = \{c_1, c_2, \dots, c_k\}$ .
  3. **While** the set centroid is changed **do**
  4.     Define the set of clusters as  $K = \{K_1, K_2, \dots, K_k\}$  where  $K_i = \emptyset$  for  $i \in \{1, 2, \dots, k\}$ .
  5.     **For**  $i$  in the set of all nodes ( $N$ ) **do**
  6.         (1) Add node  $i$  to the compatible cluster  $K_j$  ( $r_{ij} = 1$ ) that has the shortest distance between centroid  $c_j$  and node  $i$ .
  7.         (2) Add node  $i$  to the compatible cluster  $K_j$  ( $r_{ij} = 1$ ) that has the lowest of the average of upper time windows between all nodes in that cluster and node  $i$ .
  8.     **End For**
  9.     (1) Find new set of centroid nodes by the average of each position of node in the cluster.
  10.    (2) Find new set of centroid nodes by the average of position of three nodes in the cluster: the previous center, the node that has maximum upper bound, and the node that has minimum upper bound.
  11. **End While**
  12. **Return:** Set of clustered nodes.
-

### 3.3. Inserting Procedure

The inserting procedure aims to create an initial solution by inserting the incoming jobs in the best position until there are no jobs left. The algorithm aims to create a route from the solution obtained from the K-means clustering. Due to the restriction of the hard time windows, both lower and upper, some of the inserts may not be feasible to apply. We provide some conditions that ensure a node will be inserted into the routing position.

For each cluster, the first step is to sort and reorder all jobs' upper time windows in ascending order in the cluster. Each job (customer node) will then be inserted one by one until there are no jobs left. In each iteration, there is more than one possible position to insert. As the aim is to minimize total completion time, the best insertion must therefore be in the position that increases it the least. One way to obtain the best solution is to insert jobs into all possible positions and then check for feasibility and objective value; after that, the best inserting position will be obtained. Since the possible inserting position increases at each iteration, in factorial, considering all jobs will cause the execution time to increase. It is better to check inserting conditions before fully checking feasibility and calculate objective value after insertion. In this study, two conditions are provided, each requiring only small calculation steps.

Suppose that the current cluster has  $n$  number of jobs (nodes) to be scheduled. In the middle of the procedure, let the current partial routing sequence at cluster  $i$  be  $R_i = [0, \sigma_1^i, \sigma_2^i, \dots, \sigma_{n-k}^i, 0]$  and the remaining  $k$  jobs waiting to be inserted be  $K_i = \{j_{k+1}, \dots, j_n\}$ . The next candidate to be inserted is the job  $j_{k+1}$ . To check the feasibility, each constraint in the proposed model must be satisfied. However, while the procedure has not completely inserted all remaining jobs, some of the constraints are not satisfied. The steps in the inserting procedure will ensure the feasibility of the routing constraints, and the feasibility of compatibility constraints is ensured by the clustering. The feasibility of the solution can therefore be obtained by checking only time windows constraints, i.e., Constraints (10) and (11).

**Lemma 1.** Consider a routing sequence  $R_k = [0, \sigma_1^k, \sigma_2^k, \dots, \sigma_{i-1}^k, \sigma_i^k, \dots, \sigma_{|R_k|}^k, 0]$ . A job  $j$  can feasibly insert in a position between job  $\sigma_{i-1}^k$  and  $\sigma_i^k$ , i.e., the inserted routing sequence  $R_k^* = [0, \sigma_1^k, \sigma_2^k, \dots, \sigma_{i-1}^k, j, \sigma_i^k, \dots, \sigma_{|R_k|}^k, 0]$ , if two of the following conditions are satisfied: the lower time window of job  $j$  does not exceed the completion time of job  $\sigma_{i-1}^k$ , i.e.,  $e_j \leq c(\sigma_{i-1}^k)$  and the upper time window of job  $j$  is after the lower time window of job  $\sigma_i^k$ , i.e.,  $l_j \geq e_{\sigma_i^k}$ , and the completion time of job  $\sigma_{i-1}^k$  together with the distance from job  $\sigma_{i-1}^k$  to job  $j$ ,  $j$  to  $\sigma_i^k$ , and the service time of job  $j$  does not exceed the lower time window of  $\sigma_i^k$ . That is,  $c(\sigma_{i-1}^k) + d_{\sigma_{i-1}^k, j} + p_j + d_{j, \sigma_i^k} \leq e_{\sigma_i^k}$ .

**Proof.** Suppose job  $j$  is inserted in the position between job  $\sigma_{i-1}^k$  and  $\sigma_i^k$ .

Since  $e_j \leq c(\sigma_{i-1}^k)$ , job  $j$  can start immediately after arriving at node  $\sigma_i^k$ .

Moreover, the starting time of job  $j$  is equal to the combination between the completion time of job  $\sigma_{i-1}^k$ , i.e.,

$$c(\sigma_{i-1}^k) + d_{\sigma_{i-1}^k, j} = s(j). \quad (14)$$

The possible starting time of job  $\sigma_i^k$ , defined as  $s^*(\sigma_i^k)$ , is thus the completion time of job  $j$  together with the service time and the travel distance, i.e.,

$$s(j) + p_j + d_{j, \sigma_i^k} = s^*(\sigma_i^k). \quad (15)$$

From Equations (14) and (15), and Condition 2,

$$s^*(\sigma_i^k) = s(j) + p_j + d_{j, \sigma_i^k} = c(\sigma_{i-1}^k) + d_{\sigma_{i-1}^k, j} + p_j + d_{j, \sigma_i^k} \leq e_{\sigma_i^k};$$

thus,  $s^*(\sigma_i^k) \leq e_{\sigma_i^k}$ .

From the definition of the problem, if the vehicle arrives before the lower time window, it must wait and cannot provide service until the time is reached. That is, the vehicle must start at time  $e_{\sigma_i^k}$  or,

$$s(\sigma_i^k) = e_{\sigma_i^k}. \quad (16)$$

Next, we ascertain the feasibility of the inserted routing sequence by checking only Constraints (10) and (11), i.e.,  $e_\sigma \leq s(\sigma) \leq l_\sigma$  for all  $\sigma \in R_k^*$ . Since the starting time of each of the jobs processed before job  $j$ , i.e.,  $\sigma_a$  for  $a = 1, 2, \dots, i-1$ , has not been changed, Constraints (10) and (11) are satisfied for job  $a$  for  $a = 1, 2, \dots, i-1$ . For job  $j$ , from Equation (14),  $s(j) = c(\sigma_{i-1}^k) + d_{\sigma_{i-1}^k j}$ , and Condition 1,  $e_j \leq c(\sigma_{i-1}^k)$ , we have  $e_j \leq s(j)$ . From  $l_j \geq e_{\sigma_i^k}$  and Equation (16), we have  $s(j) \leq e_{\sigma_i^k} \leq l_j$ . Constraints (10) and (11) are therefore satisfied for job  $j$ .

From (16), the new starting time of job  $\sigma_i^k$  is the lower time window, which must be less than the starting time before inserting job  $j$ . The starting time, after inserting job  $j$ , of each job processed after job  $\sigma_i^k$  therefore remains the same. Thus, Constraints (10) and (11) are satisfied for all job  $\sigma \in \{\sigma_i^k, \sigma_{i+1}^k, \dots, 0\}$ .

Constraints (10) and (11) are satisfied for route  $R_k^*$ , and therefore the routing sequence is feasible.  $\square$

**Lemma 2.** A job  $j$  cannot be inserted into any position in a routing sequence  $R_k = [0, \sigma_1^k, \sigma_2^k, \dots, \sigma_{i-1}^k, \sigma_i^k, \dots, \sigma_{|R_k|}^k, 0]$  if the starting time (starting time together with service time) of each of the jobs in the routing sequence and its service time combined with the double minimum distance from the job to each node exceeds its upper time windows. That is, for each  $i \in \{1, 2, \dots, |R_k|\}$ ,  $s(\sigma_{i-1}^k) + p_j + 2\min_{i \in R_i}(d_{ij}) > l_{\sigma_i^k}$ .

**Proof.** Similar to Lemma 1, the feasibility can be checked only on Constraints (10) and (11).

Suppose the inserting of job  $j$  is feasible.

The routing, when inserting job  $j$  into some position, will always have at least one job processed after job  $j$ . Let  $\sigma_{i-1}$  and  $\sigma_j$  be jobs in routing sequence  $R_k$  that are processed before and after job  $j$ , respectively.

The starting time of job  $\sigma_i$  is:

$$s(\sigma_i) = s(j) + p_j + d_{j\sigma_i}, \quad (17)$$

and the starting time of job  $j$  is:

$$s(j) = s(\sigma_{j-1}) + p_{\sigma_{j-1}} + d_{j\sigma_{j-1}}. \quad (18)$$

From (17) and (18),

$$s(\sigma_{j-1}) + p_{\sigma_{j-1}} + d_{j\sigma_{j-1}} + p_j + d_{j\sigma_i} \leq s(j) + p_j + d_{j\sigma_i} = s(\sigma_i). \quad (19)$$

Then, from the assumption,

$$l_{\sigma_i} < s(\sigma_{j-1}) + p_j + 2\min_{i \in R_i}(d_{ij}) \leq s(\sigma_{j-1}) + p_{\sigma_{j-1}} + d_{j\sigma_{j-1}} + p_j + d_{j\sigma_i} = s(\sigma_i),$$

which contradicts the feasibility of the problem; therefore, the insertion is infeasible.  $\square$

The pseudocode for the inserting procedure is given below (Algorithm 2).

**Algorithm 2:** Inserting Procedure

---

```

1. Input: The set of nodes  $N = \{0, 1, \dots, n\}$ .
2. Set the number of clusters  $k = 2$ 
3. Apply Algorithm 1 (K – means) to cluster all jobs, set as  $K = \{K_1, K_2, \dots, K_k\}$ .
4. Check the feasibility condition from Lemma 2. If the cluster is infeasible, regenerate clusters again using the second criteria in Algorithm 1. If the cluster is still infeasible, increase the number of clusters by  $k = k + 1$  and repeat the process in line 3.
5. For cluster  $K_i \in K$  do
6.   Define empty route  $R_i = [0, 0]$ .
7.   For job  $j$  in the cluster  $K_i$  do
8.     Insert job  $j$  in the position that satisfies conditions in Lemma 1.
           If there is more than one candidate, prioritize the one having lowest total completion time first, followed by the one with minimum distance.
           Elseif there is no position that satisfies Lemma 1, insert job  $j$  into the best feasible position in the current route giving the lowest total completion time.
           Else generate a new cluster and increase the number of clusters by 1.
           End If
9.     Delete job  $j$  from cluster  $K_i$ .
10.   End For
11. End For
12. Return: Routing sequence  $R = [R_1, R_2, \dots, R_k]$ .

```

---

**3.4. Local Search Algorithms**

After applying the inserting procedure, we now obtain an initial solution, and this is the end of Phase 1. After this, the local search, or neighborhood search, seeks to find an improved solution by searching around their neighborhoods. In this study, moving and swapping procedures are selected as the local search algorithm.

**3.4.1. Moving Procedure**

The moving procedure will be applied to find the best new solution by moving some jobs to other positions in a routing sequence. The move can occur on the same route or to another route. That is, the move is equivalent to deleting followed by inserting. The conditions from Lemmas 1 and 2 can therefore also be applied to check the feasibility. In each iteration, the job with the lowest upper time window that has never been moved is selected to move. The selected job will be checked for feasibility by Lemma 1 and 2 before calculating the objective value. The pseudocode is given below (Algorithm 3).

**Algorithm 3:** Moving Procedure

---

```

1. Input: The routing sequence  $R = \{R_1, R_2, \dots, R_k\}$ .
2. Generate candidate set  $CD$  that is the sorted upper time window of all nodes.
3. For job  $j \in CD$  do
4.   Remove job  $j$  from its original position in  $R$ .
5.   For routing  $R_i$  in the routing sequence  $R$  do
6.     Insert job  $j$  into the position that satisfies conditions in Lemma 1.
           If there is more than one candidate, prioritize the one having lowest total completion time first, followed by the one with minimum distance.
           Elseif there is no position that satisfies Lemma 1, insert job  $j$  into the best feasible position in the current route giving the lowest total completion time.
           Else generate a new cluster and increase the number of clusters by 1.
           End If
7.     Save the increment of total completion time and distance for the routing sequence to  $(\Delta c_i, \Delta d_i)$ .
8.   End For
9.   Insert job  $j$  into the routing with the lowest  $\Delta c$ ; if there is more than one candidate, choose the one with the lowest  $\Delta d$ .
10. End For
11. Return: Routing sequence  $R = [R_1, R_2, \dots, R_k]$ .

```

---

**3.4.2. Swapping Procedure**

The swapping procedure aims to switch the positions of two jobs when switching results in a feasible routing sequence with total completion time reduction. If the upper time windows are close to each other, the swapping processes will likely return a feasible

routing sequence. The candidate list is modified to be the list of two jobs sorted by the differences of their upper time windows, i.e.,

$$CD = ((j_1, j_2) | j_1, j_2 \in R \times R),$$

where:

$$|l_{j_1} - l_{j_2}| \leq |l_{j_3} - l_{j_4}| \leq \dots \leq |l_{j_{n-1}} - l_{j_n}|.$$

The pseudocode is given below (Algorithm 4).

---

**Algorithm 4:** Swapping Procedure

---

1. **Input:** The routing sequence  $R = \{R_1, R_2, \dots, R_k\}$ .
  2. Generate candidate set  $CD$  that is the sorted upper time window of all nodes.
  3. **For** job  $(j_1, j_2) \in CD$  **do**
  4.     Remove jobs  $j_1, j_2$  from their original positions in  $R$ .  
      Switch the positions of job  $j_1$  and job  $j_2$ .
  5.     **If** the solution after switching the jobs is not feasible or the new total completion time is increased **do**
  6.         Switch the positions of job  $j_1$  and job  $j_2$  back to their original positions.
  - End If**
  7. **End For**
  8. **Return:** Routing sequence  $R = [R_1, R_2, \dots, R_k]$ .
- 

### 3.5. Shooting Procedure

The procedure contains two main search mechanisms, indicated as (1) and (2), which are the random search and the tabu search. The tabu search, a metaheuristic search introduced by Glover [30], explores solutions that are in its neighborhoods, that have not been visited before. The search can therefore avoid the cycling of the visits and also reduce the search space, which means it can avoid sticking in the local optimum.

If the local search (Phase 2) cannot improve the solution further, we consider the solution to be stuck in a local optimum. To get away from the local optimum, we constructed the shooting procedure to find a new local optimum. The idea of the procedure is to randomly swap or move some jobs to another position in the routing sequence without considering the objective value or feasibility of routing. The infeasible jobs will then be removed from the routing and inserted into a feasible position using the inserting procedure and then local search procedure will be applied. To obtain the best performance, the tabu search will be applied once the random search can no longer find an improved solution, rather than at the start of the procedure. Additionally, if the procedure fails to find an improved solution after several attempts, it will deploy the tabu search strategy (2), which will then reject the swapping and moving that have been tried before to widen the exploration. The tabu list is updated each iteration by adding nodes that immediately process before and after the moving and swapping nodes. These processes will continue until either the maximum iteration number is reached, or the total completion time does not change for  $m$  iterations. The pseudocode is given below (Algorithm 5).

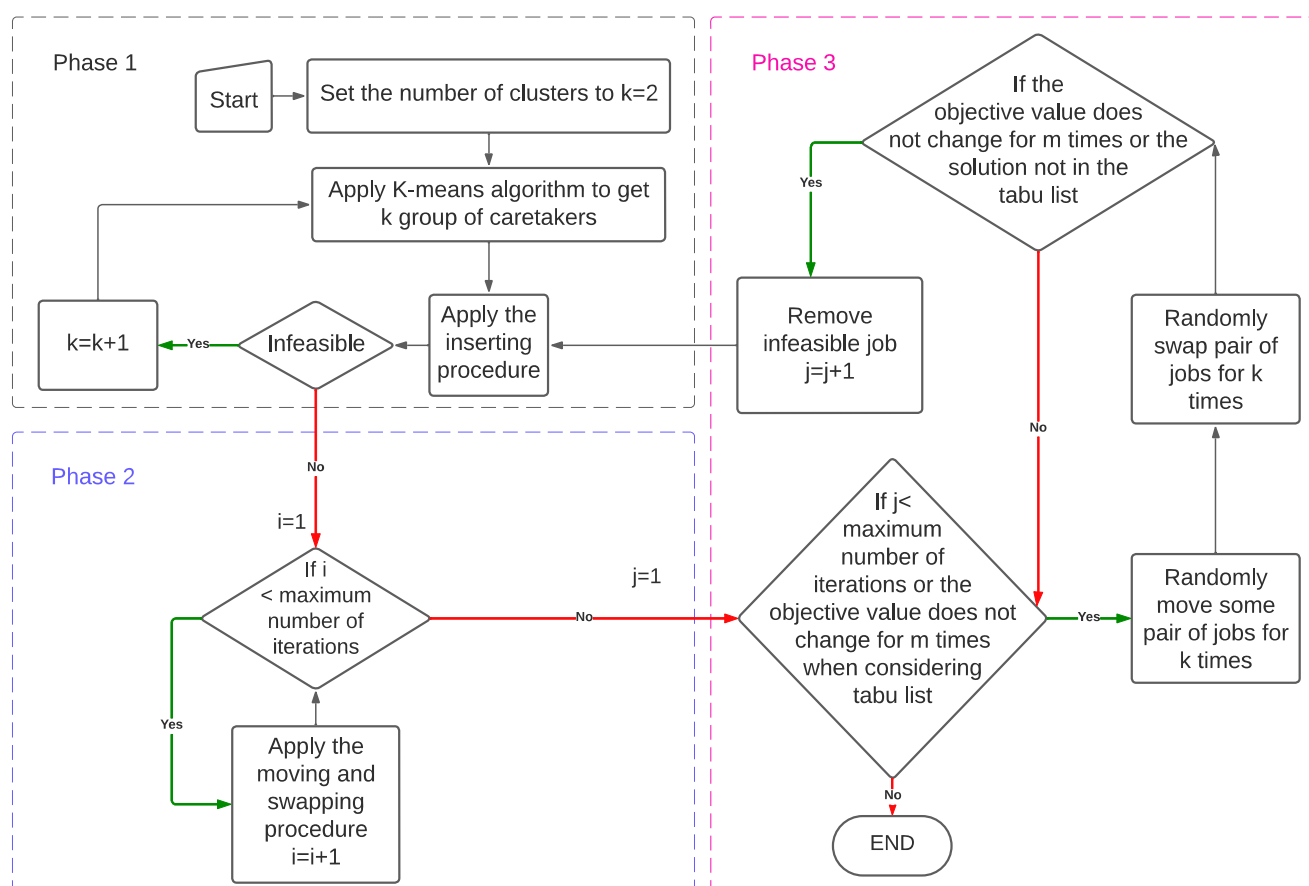
---

**Algorithm 5:** Shooting Procedure

---

1. **Input:** The routing sequence  $R = \{R_1, R_2, \dots, R_k\}$ .
  2. (1) Randomly swap (move) any pair of jobs for  $k$  times.  
   (2) Randomly swap (move) any pair of jobs for  $k$  times if it is not in the TabuList.
  3. Add the nodes that immediately visit before and after the moving nodes to the TabuList.
  4. Remove the infeasible jobs from the routing  $R$ , add to the remaining list,  $RE$ .
  5. **For** job  $j \in RE$  **do**
  6.     Apply inserting procedure to job  $j$  to the compatible routing.  
      If the inserting procedure cannot find a feasible position for job  $j$ , stop the procedure and return to the original solution.
  - End For**
  7. Apply swapping procedure and moving procedure
  8. **Return:** Routing sequence  $R = [R_1, R_2, \dots, R_k]$ .
-

To summarize, the heuristic is divided into three phases: the initializing phase, the improvement phase, and the randomization phase. The summary flowchart is given in Figure 2. Phase 1 includes K-means clustering followed by the inserting procedure. K-means clustering will apply a strategy to select the centroids based on distance. If the inserting procedure cannot find a feasible solution, the K-means clustering will apply a strategy to select centroids based on the average upper bound. Phase 2 improves the solution obtained from Phase 1 using the swapping and moving procedures until there is no change in the total completion time. Phase 3 seeks to improve the solution by randomly modifying the solution, or randomly modifying only if the solution is not in the tabu list, without considering the feasibility or total completion time, then applying the inserting procedure to fix the feasibility and reapplying Phase 2 again. Phase 3 will be attempted until the maximum number of iterations is reached, or the objective value does not change for a certain number of iterations.



**Figure 2.** Flowchart of the proposed algorithm.

#### 4. Experimental Results

To demonstrate its performance, the heuristic is implemented using the Julia programming language [31,32] and executed on AMD Ryzen 9 12-Core processors with a CPU clock speed of 3.8 GHz and RAM of 64 GB. The problem is solved several times to obtain the parameter settings. The settings are as follows: the number of vehicles will start as two and continue to increase by one when the algorithm cannot find any feasible solution; the maximum number of iterations is set to 500; the number of iterations to attempt when the objective value (total completion time) does not change is set to five (used in Phase 3); and the number of random swaps and moves is set to 10 (used for Phase 3).

Solomon instances are used to measure the algorithm efficiency. The benchmark instances are classified into three groups: clustered (C-type), random (R-type), and random clustered (RC-type). The numbers that follow the distribution of nodes are “1” and “2”,

indicating narrow time windows and wide time windows, respectively. Each instance includes 100 customers, time windows for each customer, the demand for each customer, and the distance between each pair of customer nodes. Note that in the benchmark, one unit of distance is equal to one unit of traveled time.

The simulation results are divided into two settings: the results using the total distance as the primary objective function and the results using the total completion time as the primary objective function. The first simulation setting seeks to test the efficiency of using our proposed objective function (total completion time) instead of comparing total distance. Since there is no benchmark instance using the total completion time as the objective function to demonstrate our algorithm's efficiency, the second aims to test the efficiency of the proposed algorithm by using the total distance as the objective function to solve the problem. Therefore, the results can be directly compared to Solomon's instances.

Table 2 presents the results for total distance minimization as the objective function. Each problem is solved using the proposed algorithm, while the objective function is changed from the total completion time to the total distance. Columns 3 to 5 show the calculated total completion times from the results associated with the distances shown in column 6 to 8, respectively. The results show that the algorithm can find the solution that is, on average, within 9.04% of the best known total distance from the literature and can, moreover, find the optimal solution in all clustered cases. Note that in most cases, reducing the total distance results in reduced total completion time as well. Moreover, Phase 2 can reduce the total distance from Phase 1 by 29.1% on average, while Phase 3 achieves a reduction of only 3% from Phase 2.

**Table 2.** Comparisons of total completion times and total distances for each phase using the total distance as the objective function.

| Instance | Number of Vehicles | Objective Function to Minimize Total Distance          |           |           |                |         |         | Best Known |
|----------|--------------------|--------------------------------------------------------|-----------|-----------|----------------|---------|---------|------------|
|          |                    | Total Completion Time (Associated with Total Distance) |           |           | Total Distance |         |         |            |
|          |                    | Phase 1                                                | Phase 2   | Phase 3   | Phase 1        | Phase 2 | Phase 3 |            |
| C101     | 10                 | 46,708.5                                               | 46,708.5  | 46,708.5  | 827.3          | 827.3   | 827.3   | 827.3      |
| C102     | 10                 | 66,170.9                                               | 64,529.1  | 46,708.5  | 1166.2         | 953.9   | 827.3   | 827.3      |
| C103     | 10                 | 66,321.3                                               | 65,275.3  | 47,718.7  | 1123.2         | 883.9   | 826.3   | 826.3      |
| C104     | 10                 | 70,654.4                                               | 67,154.7  | 47,366.7  | 1086.3         | 889.7   | 822.9   | 822.9      |
| C105     | 10                 | 65,382.7                                               | 63,469.5  | 46,708.5  | 1279.3         | 1033.1  | 827.3   | 827.3      |
| C106     | 10                 | 64,345.5                                               | 46,708.5  | 46,708.5  | 941.0          | 827.3   | 827.3   | 827.3      |
| C107     | 10                 | 62,414.6                                               | 62,660.1  | 46,708.5  | 1196.3         | 1064.6  | 827.3   | 827.3      |
| C108     | 10                 | 65,734.0                                               | 63,820.7  | 46,708.5  | 999.6          | 901.5   | 827.3   | 827.3      |
| C109     | 10                 | 63,203.7                                               | 46,708.5  | 46,708.5  | 938.9          | 827.3   | 827.3   | 827.3      |
| C201     | 3                  | 175,896.9                                              | 175,511.5 | 155,221.0 | 825.2          | 678.3   | 589.1   | 589.1      |
| C202     | 3                  | 175,413.0                                              | 174,082.3 | 155,221.0 | 1072.1         | 681.5   | 589.1   | 589.1      |
| C203     | 3                  | 172,474.4                                              | 161,382.4 | 155,266.1 | 1295.3         | 758.6   | 588.7   | 588.7      |
| C204     | 3                  | 192,759.2                                              | 154,606.0 | 155,216.2 | 1341.6         | 791.7   | 590.6   | 590.6      |
| C205     | 3                  | 178,335.1                                              | 172,008.5 | 155,180.2 | 1101.1         | 673.1   | 586.4   | 586.4      |
| C206     | 3                  | 173,859.6                                              | 171,608.2 | 155,183.2 | 1396.6         | 714.0   | 586.0   | 586.0      |
| C207     | 3                  | 171,321.9                                              | 168,964.1 | 156,185.4 | 1109.8         | 689.7   | 585.8   | 585.8      |
| C208     | 3                  | 173,535.7                                              | 166,018.2 | 157,201.4 | 1209.4         | 679.2   | 585.8   | 585.8      |
| R101     | 20                 | 17,075.5                                               | 15,169.5  | 15,169.5  | 2586.7         | 1827.9  | 1827.9  | 1637.7     |
| R102     | 18                 | 15,231.7                                               | 15,590.9  | 15,590.9  | 2240.8         | 1673.1  | 1673.1  | 1466.6     |
| R103     | 15                 | 13,879.7                                               | 14,546.0  | 14,546.0  | 1989.9         | 1422.0  | 1422.0  | 1208.7     |
| R104     | 10                 | 13,192.5                                               | 14,279.9  | 14,279.9  | 1527.5         | 1204.1  | 1204.1  | 976.61     |

Table 2. Cont.

| Instance | Number of Vehicles | Objective Function to Minimize Total Distance          |          |          |                |         |         | Best Known |
|----------|--------------------|--------------------------------------------------------|----------|----------|----------------|---------|---------|------------|
|          |                    | Total Completion Time (Associated with Total Distance) |          |          | Total Distance |         |         |            |
|          |                    | Phase 1                                                | Phase 2  | Phase 3  | Phase 1        | Phase 2 | Phase 3 |            |
| R105     | 16                 | 14,219.4                                               | 14,014.3 | 14,014.3 | 1819.5         | 1576.6  | 1576.6  | 1355.3     |
| R106     | 14                 | 13,800.8                                               | 14,462.4 | 14,462.4 | 1805.4         | 1367.6  | 1367.6  | 1234.6     |
| R107     | 12                 | 13,270.4                                               | 14,023.1 | 14,023.1 | 1622.8         | 1176.1  | 1176.1  | 1064.6     |
| R108     | 11                 | 11,958.2                                               | 13,392.7 | 13,392.7 | 1532.1         | 1016.3  | 1016.3  | 938.2      |
| R109     | 13                 | 13,228.0                                               | 12,865.0 | 12,865.0 | 1550.6         | 1296.9  | 1296.9  | 1146.9     |
| R110     | 12                 | 13,078.7                                               | 12,868.2 | 12,868.2 | 1621.2         | 1237.5  | 1237.5  | 1068.0     |
| R111     | 12                 | 13,654.8                                               | 14,186.8 | 14,186.8 | 1658.9         | 1211.1  | 1211.1  | 1048.7     |
| R112     | 11                 | 12,816.3                                               | 12,776.7 | 12,776.7 | 1421.9         | 990.5   | 990.5   | 953.63     |
| R201     | 8                  | 48,872.8                                               | 48,912.5 | 48,912.5 | 1686.3         | 1297.8  | 1297.8  | 1143.2     |
| R202     | 7                  | 53,436.5                                               | 50,022.4 | 50,022.4 | 1592.6         | 1096.6  | 1096.6  | 1034.4     |
| R203     | 6                  | 49,275.0                                               | 48,349.2 | 48,349.2 | 1558.9         | 949.7   | 949.7   | 874.87     |
| R204     | 5                  | 49,075.2                                               | 41,745.6 | 41,745.6 | 1700.2         | 863.3   | 863.3   | 735.8      |
| R205     | 5                  | 45,477.3                                               | 46,508.8 | 46,508.8 | 1717.8         | 1039.4  | 1039.4  | 954.16     |
| R206     | 6                  | 46,932.5                                               | 46,668.1 | 46,668.1 | 1696.1         | 998.6   | 998.6   | 879.86     |
| R207     | 5                  | 45,948.3                                               | 43,573.2 | 43,573.2 | 1657.4         | 914.1   | 914.1   | 797.99     |
| R208     | 4                  | 41,766.1                                               | 38,064.3 | 38,064.3 | 1555.2         | 792.7   | 792.7   | 705.33     |
| R209     | 5                  | 46,829.5                                               | 44,337.1 | 44,337.1 | 1740.9         | 917.5   | 917.5   | 859.39     |
| R210     | 6                  | 47,285.4                                               | 46,121.9 | 46,121.9 | 1628.0         | 1012.7  | 1012.7  | 905.21     |
| R211     | 5                  | 46,686.0                                               | 46,065.0 | 46,065.0 | 1577.0         | 858.2   | 858.2   | 753.15     |
| RC101    | 16                 | 14,745.0                                               | 14,644.5 | 14,644.5 | 2105.7         | 1843.1  | 1843.1  | 1619.8     |
| RC102    | 14                 | 15,027.7                                               | 14,728.6 | 14,728.6 | 2037.9         | 1613.5  | 1613.5  | 1457.4     |
| RC103    | 12                 | 13,433.8                                               | 14,831.6 | 14,831.6 | 1862.0         | 1433.1  | 1433.1  | 1258.0     |
| RC104    | 11                 | 12,685.2                                               | 12,985.7 | 12,985.7 | 1489.6         | 1226.9  | 1226.9  | 1135.5     |
| RC105    | 15                 | 15,050.8                                               | 14,847.3 | 14,847.3 | 2145.7         | 1717.2  | 1717.2  | 1513.7     |
| RC106    | 14                 | 14,298.1                                               | 13,730.5 | 13,730.5 | 1884.7         | 1523.3  | 1523.3  | 1378.0     |
| RC107    | 12                 | 13,540.3                                               | 13,970.5 | 13,970.5 | 1760.2         | 1361.9  | 1361.9  | 1212.8     |
| RC108    | 11                 | 13,903.3                                               | 14,340.4 | 14,340.4 | 1773.2         | 1421.6  | 1421.6  | 1117.5     |
| RC201    | 9                  | 46,255.2                                               | 46,783.3 | 46,783.3 | 1957.5         | 1409.6  | 1409.6  | 1261.8     |
| RC202    | 8                  | 51,022.2                                               | 47,113.7 | 47,113.7 | 1884.0         | 1315.5  | 1315.5  | 1095.6     |
| RC203    | 5                  | 50,050.6                                               | 51,088.5 | 51,088.5 | 1738.5         | 1064.1  | 1064.1  | 926.82     |
| RC204    | 4                  | 41,403.8                                               | 40,090.2 | 40,090.2 | 1612.2         | 819.5   | 819.5   | 786.38     |
| RC205    | 8                  | 45,620.5                                               | 45,992.3 | 45,992.3 | 1921.4         | 1383.9  | 1383.9  | 1157.6     |
| RC206    | 7                  | 43,675.6                                               | 44,489.7 | 44,489.7 | 1868.3         | 1161.4  | 1161.4  | 1054.6     |
| RC207    | 6                  | 45,548.0                                               | 44,417.7 | 44,417.7 | 1759.8         | 1053.7  | 1053.7  | 966.08     |
| RC208    | 4                  | 42,070.2                                               | 47,748.1 | 47,748.1 | 1704.2         | 924.9   | 924.9   | 778.93     |

To demonstrate the efficacy of the methodologies employed and the importance of using the total completion time as the objective function (the primary objective, with total distance as a second objective), a comparative analysis is conducted on our proposed solution and the best solution from Solomon's benchmark instances, as shown in Table 1. Note that some of the best known solutions are not available, preventing us from calculating the total completion time associated with the best distance solutions. To obtain that information, we apply our algorithm to optimize the total distance and obtain the corresponding travel time. Table 3 presents the most widely recognized solutions for total distance, along with the corresponding total completion time of the caretaker. Note that the number of vehicles is the same as that shown in Table 2. Column 7 shows the solution of our algorithm by minimizing the total distance, and the corresponding calculated total completion time is shown in column 8. Columns 2–4 show the results from our algorithm for the total completion time minimization objective function, while columns 5–7 show the total distance calculated from the corresponding solution from columns 2–4. This value is consistently smaller in all cases when compared to the solutions provided by Solomon's

instances. However, it is important to note that the total distance traveled is expected to increase. Moreover, when the algorithm minimizes the total completion time for instance C101, Phase 1 (the initial solution) obtains the best solution for total distance, and then the algorithm obtains the best new solution in which the total distance increases by 14.13% and total completion time decreases by 0.05%. In this case, we found that some caretakers travel more distance to cover a node that is currently available.

**Table 3.** Comparisons of the total completion time and total distance for each phase using the total completion time as the objective function.

| Instance | Objective Function to Minimize Total Completion Time |           |           |                                                        |         |         | Objective Function to Minimize Total Distance |                       |
|----------|------------------------------------------------------|-----------|-----------|--------------------------------------------------------|---------|---------|-----------------------------------------------|-----------------------|
|          | Total Completion Time                                |           |           | Total Distance (Associated with Total Completion Time) |         |         | Total Distance                                | Total Completion Time |
|          | Phase 1                                              | Phase 2   | Phase 3   | Phase 1                                                | Phase 2 | Phase 3 |                                               |                       |
| C101     | 46,708.5                                             | 46,469.9  | 45,365.7  | 827.3                                                  | 944.2   | 942.4   | 827.3                                         | 46,708.5              |
| C102     | 45,086.2                                             | 37,576.4  | 36,038.8  | 1161.6                                                 | 1719.7  | 1716.8  | 827.3                                         | 46,708.5              |
| C103     | 47,241.2                                             | 43,291.3  | 42,065.3  | 1206.4                                                 | 1233.2  | 1230.4  | 826.3                                         | 47,718.7              |
| C104     | 51,939.6                                             | 42,616.3  | 41,333.9  | 1089.4                                                 | 1154.6  | 1151.9  | 822.9                                         | 47,366.7              |
| C105     | 45,310.2                                             | 42,130.3  | 40,521.1  | 1242.5                                                 | 1948.8  | 1945.7  | 827.3                                         | 46,708.5              |
| C106     | 45,163.7                                             | 43,102.3  | 41,813.9  | 1024.0                                                 | 1520.3  | 1517.6  | 827.3                                         | 46,708.5              |
| C107     | 44,553.8                                             | 40,448.5  | 39,052.9  | 1149.4                                                 | 1616.8  | 1614.3  | 827.3                                         | 46,708.5              |
| C108     | 45,796.7                                             | 40,560.0  | 39,269.7  | 973.2                                                  | 1399.9  | 1397.4  | 827.3                                         | 46,708.5              |
| C109     | 45,237.6                                             | 42,291.5  | 41,026.8  | 932.1                                                  | 1021.8  | 1019.6  | 827.3                                         | 46,708.5              |
| C201     | 151,405.1                                            | 147,642.9 | 147,113.0 | 884.3                                                  | 1686.9  | 1683.3  | 589.1                                         | 155,221.0             |
| C202     | 151,495.0                                            | 126,593.8 | 125,942.9 | 1239.0                                                 | 2181.2  | 2167.9  | 589.1                                         | 155,221.0             |
| C203     | 158,923.8                                            | 97,109.1  | 96,336.1  | 1293.3                                                 | 2370.1  | 2366.2  | 588.7                                         | 155,266.1             |
| C204     | 156,737.6                                            | 91,417.2  | 89,479.4  | 1659.2                                                 | 1637.6  | 1463.2  | 590.6                                         | 155,216.2             |
| C205     | 152,232.4                                            | 140,234.9 | 139,484.9 | 1055.2                                                 | 3149.9  | 3146.0  | 586.4                                         | 155,180.2             |
| C206     | 148,228.9                                            | 136,870.1 | 136,421.6 | 1102.8                                                 | 1559.8  | 1556.1  | 586.0                                         | 155,183.2             |
| C207     | 148,788.3                                            | 127,461.6 | 126,912.0 | 1165.7                                                 | 1980.2  | 1976.7  | 585.8                                         | 156,185.4             |
| C208     | 152,350.9                                            | 131,323.8 | 130,248.6 | 1185.6                                                 | 1439.3  | 1267.3  | 585.8                                         | 157,201.4             |
| R101     | 11,111.4                                             | 10,413.8  | 9657.7    | 2519.0                                                 | 2464.9  | 2459.5  | 1637.7                                        | 10,408.2              |
| R102     | 10,647.2                                             | 8823.4    | 8039.3    | 2221.2                                                 | 2342.8  | 2336.9  | 1466.6                                        | 10,777.3              |
| R103     | 10,361.0                                             | 7542.0    | 6849.6    | 1813.5                                                 | 2108.6  | 2103.4  | 1208.7                                        | 11,386.2              |
| R104     | 8337.5                                               | 7196.2    | 6645.1    | 1622.0                                                 | 1641.5  | 1636.3  | 976.61                                        | 11,307.6              |
| R105     | 10,341.6                                             | 9474.7    | 8863.5    | 1904.5                                                 | 1955.7  | 1950.6  | 1355.3                                        | 10,296.0              |
| R106     | 9889.3                                               | 8204.7    | 7655.4    | 1742.9                                                 | 1901.9  | 1896.9  | 1234.6                                        | 10,463.8              |
| R107     | 9780.1                                               | 7853.7    | 7350.8    | 1632.4                                                 | 1621.2  | 1615.9  | 1064.6                                        | 11,876.1              |
| R108     | 8788.9                                               | 7080.6    | 6561.3    | 1483.6                                                 | 1440.1  | 1435.6  | 938.20                                        | 10,603.0              |
| R109     | 10,025.7                                             | 8788.7    | 8314.2    | 1723.2                                                 | 1598.4  | 1593.4  | 1146.9                                        | 10,377.7              |
| R110     | 9995.5                                               | 7624.5    | 7049.5    | 1683.6                                                 | 1588.5  | 1583.8  | 1068.0                                        | 10,739.7              |
| R111     | 10,002.5                                             | 7836.5    | 7289.0    | 1585.3                                                 | 1635.1  | 1630.2  | 1048.7                                        | 10,619.6              |
| R112     | 8984.2                                               | 8256.1    | 7805.4    | 1405.2                                                 | 1261.6  | 1256.6  | 953.63                                        | 10,752.4              |
| R201     | 42,983.2                                             | 39,654.6  | 39,498.5  | 1671.7                                                 | 2235.8  | 2231.1  | 1143.2                                        | 44,065.3              |
| R202     | 47,676.9                                             | 32,516.2  | 32,353.0  | 1597.1                                                 | 2020.5  | 2016.0  | 1034.4                                        | 43,462.4              |
| R203     | 44,592.0                                             | 26,909.5  | 26,759.1  | 1579.4                                                 | 1696.2  | 1691.7  | 874.87                                        | 45,651.9              |
| R204     | 44,797.4                                             | 24,347.7  | 23,800.8  | 1704.1                                                 | 1113.0  | 1067.1  | 735.80                                        | 51,001.0              |
| R205     | 41,474.1                                             | 35,453.9  | 35,344.7  | 1643.4                                                 | 1741.3  | 1737.0  | 954.16                                        | 43,690.2              |
| R206     | 41,127.1                                             | 28,848.0  | 28,708.9  | 1691.9                                                 | 1687.7  | 1683.2  | 879.86                                        | 41,257.9              |
| R207     | 42,052.1                                             | 27,440.0  | 27,296.9  | 1661.4                                                 | 1462.3  | 1458.0  | 797.99                                        | 53,451.7              |
| R208     | 51,761.3                                             | 23,102.9  | 22,748.4  | 1627.9                                                 | 1001.1  | 1024.7  | 705.33                                        | 46,639.7              |
| R209     | 42,820.9                                             | 29,626.0  | 29,474.1  | 1781.1                                                 | 1766.0  | 1761.3  | 859.39                                        | 45,218.1              |
| R210     | 44,966.5                                             | 29,680.5  | 29,528.4  | 1688.1                                                 | 1735.2  | 1730.4  | 905.21                                        | 45,920.8              |

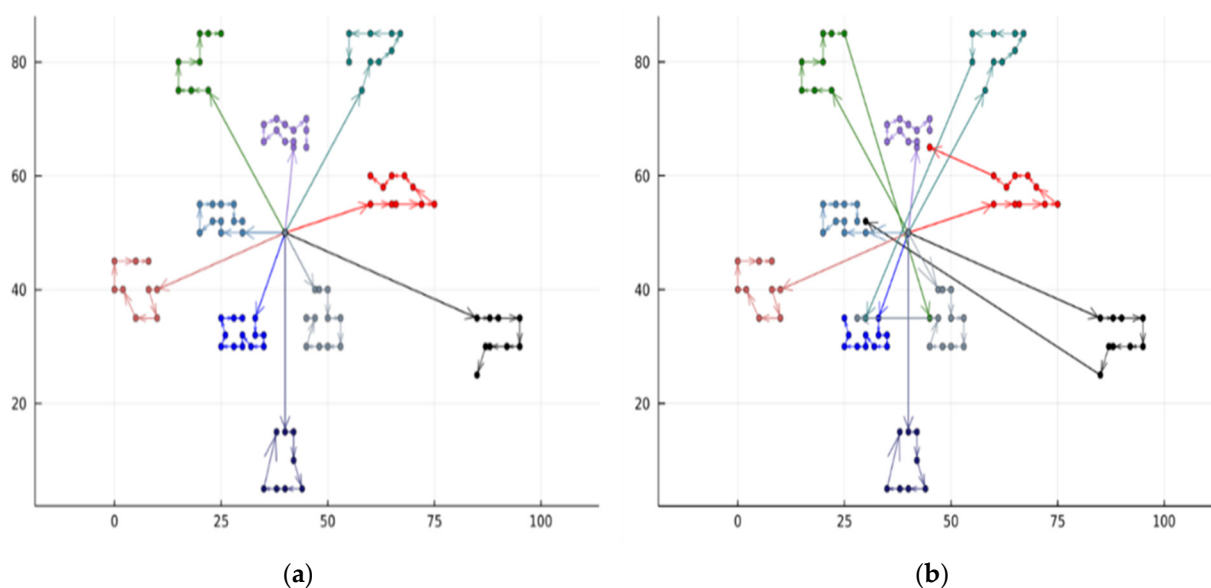
Table 3. Cont.

| Instance | Objective Function to Minimize Total Completion Time |          |          |                                                        |         |         | Objective Function to Minimize Total Distance |                       |
|----------|------------------------------------------------------|----------|----------|--------------------------------------------------------|---------|---------|-----------------------------------------------|-----------------------|
|          | Total Completion Time                                |          |          | Total Distance (Associated with Total Completion Time) |         |         | Total Distance                                | Total Completion Time |
|          | Phase 1                                              | Phase 2  | Phase 3  | Phase 1                                                | Phase 2 | Phase 3 |                                               |                       |
| R211     | 42,941.7                                             | 27,788.0 | 27,654.8 | 1581.2                                                 | 1306.2  | 1302.0  | 753.15                                        | 46,238.9              |
| RC101    | 10,822.6                                             | 10,263.9 | 9568.8   | 2129.6                                                 | 2172.4  | 2167.8  | 1619.8                                        | 10,979.1              |
| RC102    | 10,976.2                                             | 9109.4   | 8390.8   | 2166.4                                                 | 2145.1  | 2140.3  | 1457.4                                        | 11,507.8              |
| RC103    | 10,315.7                                             | 8420.8   | 7767.2   | 1814.0                                                 | 1995.4  | 1990.8  | 1258.0                                        | 11,693.0              |
| RC104    | 9673.7                                               | 8414.5   | 7788.4   | 1494.3                                                 | 1547.3  | 1543.9  | 1135.48                                       | 11,227.9              |
| RC105    | 11,096.6                                             | 9298.3   | 8512.6   | 2174.6                                                 | 2442.8  | 2438.2  | 1513.7                                        | 11,142.7              |
| RC106    | 10,815.1                                             | 9654.0   | 9041.8   | 1842.1                                                 | 1731.6  | 1727.3  | 1376.99                                       | 11,093.5              |
| RC107    | 10,078.3                                             | 8858.5   | 8209.1   | 1780.0                                                 | 1754.6  | 1750.8  | 1212.83                                       | 11,163.1              |
| RC108    | 10,625.3                                             | 8922.2   | 8388.1   | 1812.9                                                 | 1478.3  | 1474.4  | 1117.53                                       | 11,362.6              |
| RC201    | 40,735.5                                             | 37,809.8 | 37,586.4 | 1961.5                                                 | 2521.0  | 2516.5  | 1261.80                                       | 41,647.9              |
| RC202    | 45,519.5                                             | 32,159.1 | 31,934.7 | 1832.3                                                 | 2032.5  | 2105.1  | 1095.64                                       | 43,558.9              |
| RC203    | 45,462.2                                             | 27,123.5 | 26,943.8 | 1743.0                                                 | 1926.9  | 1922.9  | 926.82                                        | 40,691.6              |
| RC204    | 33,426.9                                             | 21,225.2 | 20,904.2 | 1497.4                                                 | 1470.5  | 1258.6  | 786.38                                        | 44,398.0              |
| RC205    | 40,434.0                                             | 34,394.1 | 34,223.2 | 1919.4                                                 | 2128.0  | 2123.8  | 1157.55                                       | 42,784.1              |
| RC206    | 39,235.4                                             | 33,022.9 | 32,827.2 | 1872.3                                                 | 2046.0  | 2042.3  | 1054.61                                       | 41,911.4              |
| RC207    | 41,213.1                                             | 29,106.3 | 28,531.0 | 1764.2                                                 | 1984.8  | 1973.0  | 966.08                                        | 42,569.6              |
| RC208    | 38,449.1                                             | 28,815.4 | 27,040.0 | 1707.1                                                 | 1352.3  | 1245.3  | 778.90                                        | 42,289.3              |

The execution time for Phase 1 is approximately 0.28 s for R-type instances, 0.06 s for C-type instances, and 0.17 s for RC-type instances. Phase 2 requires a comparable amount of time to Phase 1, whereas Phase 3 accounts for approximately 99.68% of the time spent. It should be noted that the duration of execution in Phase 3 is based on the maximum number of iterations, set at 500 for this particular instance. The average reduction in the objective function value was 19.40% from Phase 1 to Phase 2 and 3.55% from Phase 2 to Phase 3.

When determining the percentage difference between the total completion times solved using Solomon's instances and the completion times obtained from each phase of our algorithm, we observed that in all instances, the total completion times achieved with our algorithm were, on average, 25.86% lower. Additionally, the average total distance traveled increased by 90.00%. Note that there is a small improvement in the total completion time from Phase 2 to Phase 3, while the algorithm spends the most time in this phase. In most cases, the total completion time improvement from Phase 2 to Phase 3 may result in an increase in the total distance. In contrast, several instances in Table 3, C202, C203, R101, RC105, RC106, and RC204, show a reduction in the total completion time and total distance simultaneously. This demonstrates that the total distance may decrease simultaneously with the total completion time when the distribution of nodes is more random than clustered. The algorithm can find the solution that minimizes the total completion time and try to reduce the total distance at the same time.

Figure 3 illustrates a representative path within the solutions of the "C101" instance. Figure 3a displays the routings obtained from the optimal benchmark solution resulting from minimizing the total distance traveled. Figure 3b shows the solution obtained with our procedure, where the objective was to minimize the total completion time. Note that, to prevent confusion when several lines cross each other, the lines from the last node to the depot node are not shown. The final customer in the purple route of the optimal solution in Figure 3a transitions to being a customer in the red route illustrated in Figure 3b. The result leads to an increase in total distance covered, while simultaneously decreasing the total completion time of the entire system.



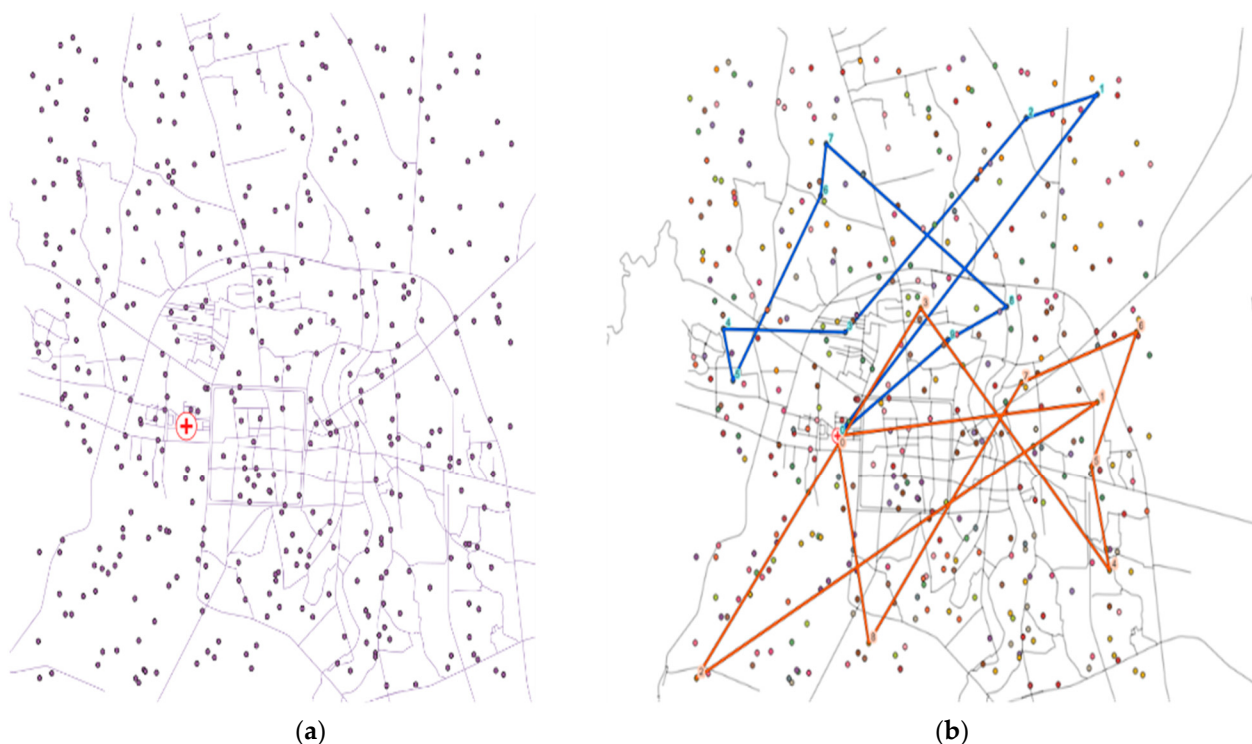
**Figure 3.** (a) Best solution of the “C101” benchmark instance when the total distance is minimized. (b) The solution of the “C101” instance when the total completion time is minimized.

### 5. Case Study

To examine the advantages of the total completion time objective function, our study focuses on minimizing the total completion time in a case study simulating the HHC system in Chiang Mai, Thailand. The proposed heuristic uses the same parameter settings as when solving Solomon’s instances.

We assume that 400 households currently registered with the largest health center in central Chiang Mai, an area with over a half a million people and over 300,000 households, require regular home visits. These prospective customers are dispersed at random across the map of the city center, where the location of the designated health center is denoted by a red cross (Figure 4a). It is expected that caretakers will serve all customers in a single workday. A caretaker will not accept a job that is beyond his or her capabilities, and there are enough caretakers to attend to all the customers. The times listed for travel between each pair of nodes come directly from Google Maps. The time windows for each client are randomly generated, anywhere from zero to eight hours (28,800 s). The amount of time spent serving each customer is randomly determined, ranging from 5 to 20 min.

Table 4 presents a summary of the results obtained from both objective functions. The second column, labelled “NC”, shows the number of caretakers needed. The columns labelled as “Min”, “Max”, and “Avg” correspond to the minimum, maximum, and average number of patients attended to by a caretaker, respectively. The lower and upper bounds for the number of patients in relation to the total distance objective function are three and forty-two, respectively. This implies that in the process of minimizing the total distance, certain caretakers are required to attend to significantly more patients, up to 14 times more, than their coworkers. This number drops by a factor of seven when the objective of minimizing the overall completion time is achieved. Additionally, in seeking to minimize the completion time, the required number of caretakers will increase to effectively distribute the workload among them. In the context of caretaker management, we propose that the total completion time is a more suitable objective function compared to the total distance, as it leads to a decrease in both the average number and maximum number of patients that a caretaker serves.



**Figure 4.** (a) Map of the city of Chiang Mai with 400 randomly placed potential clients for home visits. (b) Examples of routing from our heuristic when total completion time is minimized.

**Table 4.** Summary of results for Chiang Mai home visiting with 400 nodes.

| Objective Function    | Number of Jobs per Caretaker |     |     |      | Max Completion Time per Caretaker (Hours) |     |     | Travel Time per Caretaker (Hours) |     |     |
|-----------------------|------------------------------|-----|-----|------|-------------------------------------------|-----|-----|-----------------------------------|-----|-----|
|                       | NC                           | Min | Max | Avg  | Min                                       | Max | Avg | Min                               | Max | Avg |
| Total completion time | 34                           | 8   | 18  | 11.7 | 1.5                                       | 6.9 | 4.4 | 1.3                               | 2.2 | 1.5 |
| Total distance        | 34                           | 3   | 42  | 16.0 | 1.4                                       | 6.7 | 4.6 | 0.2                               | 3.3 | 1.3 |

The two routes depicted in Figure 4b illustrate the routing obtained by implementing our heuristic algorithm. The numerical values assigned to the routes correspond to the sequential order in which a caretaker performs his/her visits. Both routes show chaos as a result of the time constraints imposed by patients' availability (time windows). Upon completion of service to patient 2 along the blue route, the care staff makes the decision to attend to patient 3 due to their earlier time window, despite the fact that visiting patient 7 first would result in a shorter travel distance overall.

## 6. Conclusions and Discussions

In this paper, we study home healthcare routing and scheduling involving the need for staff members to visit designated patients within predefined time windows and with compatibility requirements. In this scenario, it is crucial for healthcare staff to start and finish their service to patients strictly within the designated timeframe, and they are only able to visit compatible patients. In the event that staff arrives at the patient node before the designated time, they are required to wait until the scheduled time to deliver care. The given problem can be reformulated as a vehicle routing problem with time windows (VRPTW) and compatibility constraints, a problem that originally seeks to minimize the total distance travelled. To ensure suitability for practical applications, we choose to set minimizing the completion time as our objective function rather than focus on minimizing the total distance alone.

The vehicle routing problem is an NP-hard problem. Adding constraints or changing the objective function makes the problem even more complex. Our problem on its own is an NP-hard problem; addressing situations where optimal solutions cannot be obtained in a reasonable time with an exact algorithm requires that heuristics be developed. The heuristic employed in our study comprises three distinct phases. The primary phase (Phase 1) aims to generate a solution from the ground up, with patients systematically clustered and inserted one by one into positions that minimize the total completion time. The second phase (Phase 2) focuses on enhancing the solution by implementing swap and move operators. Phase 3 seeks to prevent the problem of local optimum by randomly exchanging or moving a pair of patients that do not violate the tabu list within the solution. Moreover, the heuristic simultaneously seeks to minimize the total distance as a secondary objective.

To demonstrate the performance of the algorithm, comparisons are made between the results from our model and an existing benchmark instance which includes 100 customer nodes. When the total completion time is minimized, the total distance traveled increases. However, our total completion time tends to be lower generally. We modified our algorithm to simultaneously reduce the total distance as a second priority. After the initial phase, the algorithm can, in some instances, find a solution with a reduction in both the total completion time and total distance. Further, the algorithm is compared to the best known solutions with total distance minimization as the primary objective. The optimal solution is found by the algorithm in all clustered cases, and the overall percentage gap is within 10%. From these results, we can see that switching from minimizing the total distance to minimizing the total completion time will increase the total distance by around 90.00%, while the total completion time drops by 25.86%. In practice, both objective functions should be minimized separately, and the decision made by the administrator which of the solutions will minimize operation costs. Note that the reduction in the total distance may cause some staff to work longer, a scenario that can be more costly than increasing the total distance traveled.

Our case study pertains to the home healthcare system in Chiang Mai, Thailand's second most populous city. The system consists of 15 health centers serving over 300,000 households. Based on the results from our simulation, we recommend that minimizing the total completion time, rather than the total distance, should be prioritized in order to ensure staff satisfaction, which is vital in a people-centered business with ever-increasing demand for qualified staff. When total completion time minimization is prioritized, each caretaker is assigned a roughly equal number of patients, which is not the case when minimizing the total distance is the primary objective function.

**Author Contributions:** Conceptualization, P.S. and C.L.; Data curation, P.S.; Formal analysis, C.L.; Investigation, P.S.; Methodology, P.S. and C.L.; Software, P.S.; Supervision, C.L.; Validation, P.S. and C.L.; Visualization, P.S. and C.L.; Writing—original draft, P.S.; Writing—review and editing, C.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Royal Golden Jubilee (RGJ) Ph.D. Program [Grant No. PHD/0041/2560] and the Fundamental Fund 2023 [Grant No. FF66/024], Chiang Mai University.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author.

**Acknowledgments:** The authors would like to thank Chiang Mai University and the National Research Council of Thailand (NRCT). The authors also appreciate the proofreading help of Wiriya Sungkhaniyom.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. United Nations Department of Economic and Social Affairs. *World Population Prospects 2019, Volume II: Demographic Profiles*; United Nations: New York, NY, USA, 2019.
2. Office of the National Economic and Social Development Board Office of the Prime Minister. *Summary the Twelfth National Economic and Social Development Plan (2017–2021)*; Office of the National Economic and Social Development Board Office of the Prime Minister: Bangkok, Thailand, 2017.
3. Suriyanrattakorn, S.; Chang, C.-L. Long-Term Care (LTC) Policy in Thailand on the Homebound and Bedridden Elderly Happiness. *Health Policy Open* **2021**, *2*, 100026. [\[CrossRef\]](#) [\[PubMed\]](#)
4. National Statistical Office Ministry of Digital Economy and Society. *Statistical Yearbook Thailand 2020*; Statistical Forecasting Division: Bangkok, Thailand, 2020.
5. Glinskaya, E.E.; Walker, T.W. *Thisuri Jinadhi Labor Markets and Social Policy in a Rapidly Transforming: Caring for Thailand's Aging Population (English)*; World Bank Group: Washington, DC, USA, 2021.
6. Wirnitzer, J.; Heckmann, I.; Meyer, A.; Nickel, S. Patient-Based Nurse Rostering in Home Care. *Oper. Res. Health Care* **2016**, *8*, 91–102. [\[CrossRef\]](#)
7. Nasir, J.A.; Kuo, Y.-H. A Decision Support Framework for Home Health Care Transportation with Simultaneous Multi-Vehicle Routing and Staff Scheduling Synchronization. *Decis. Support. Syst.* **2020**, *138*, 113361. [\[CrossRef\]](#)
8. Kandakoglu, A.; Sauré, A.; Michalowski, W.; Aquino, M.; Graham, J.; McCormick, B. A Decision Support System for Home Dialysis Visit Scheduling and Nurse Routing. *Decis. Support. Syst.* **2020**, *130*, 113224. [\[CrossRef\]](#)
9. Braekers, K.; Hartl, R.F.; Parragh, S.N.; Tricoire, F. A Bi-Objective Home Care Scheduling Problem: Analyzing the Trade-off between Costs and Client Inconvenience. *Eur. J. Oper. Res.* **2016**, *248*, 428–443. [\[CrossRef\]](#)
10. Cissé, M.; Yalçındağ, S.; Kergosien, Y.; Şahin, E.; Lenté, C.; Matta, A. OR Problems Related to Home Health Care: A Review of Relevant Routing and Scheduling Problems. *Oper. Res. Health Care* **2017**, *13–14*, 1–22. [\[CrossRef\]](#)
11. Di Mascolo, M.; Martinez, C.; Espinouse, M.-L. Routing and Scheduling in Home Health Care: A Literature Survey and Bibliometric Analysis. *Comput. Ind. Eng.* **2021**, *158*, 107255. [\[CrossRef\]](#)
12. Goodarzian, F.; Garjan, H.S.; Ghasemi, P. A State-of-the-Art Review of Operation Research Models and Applications in Home Healthcare. *Healthc. Anal.* **2023**, *4*, 100228. [\[CrossRef\]](#)
13. Solomon, M.M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Oper. Res.* **1987**, *35*, 254–265. [\[CrossRef\]](#)
14. Ahn, B.-H.; Shin, J.-Y. Vehicle-Routing with Time Windows and Time-Varying Congestion. *J. Oper. Res. Soc.* **1991**, *42*, 393. [\[CrossRef\]](#)
15. Rochat, Y.; Taillard, É.D. Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. *J. Heuristics* **1995**, *1*, 147–167. [\[CrossRef\]](#)
16. Li, H.; Lim, A. Local Search with Annealing-like Restarts to Solve the VRPTW. *Eur. J. Oper. Res.* **2003**, *150*, 115–127. [\[CrossRef\]](#)
17. Pisinger, D.; Ropke, S. A General Heuristic for Vehicle Routing Problems. *Comput. Oper. Res.* **2007**, *34*, 2403–2435. [\[CrossRef\]](#)
18. Gehring, H.; Homberger, J. A Parallel Hybrid Evolutionary Metaheuristic for the Vehicle Routing Problem with Time Windows. In Proceedings of the EUROGEN99, Jyväskylä, Finland, 29 May–3 June 1999; pp. 57–64.
19. Low, C.; Chang, C.-M.; Li, R.-K.; Huang, C.-L. Coordination of Production Scheduling and Delivery Problems with Heterogeneous Fleet. *Int. J. Prod. Econ.* **2014**, *153*, 139–148. [\[CrossRef\]](#)
20. Khoo, T.S.; Mohammad, B.B. The Parallelization of a Two-Phase Distributed Hybrid Ruin-and-Recreate Genetic Algorithm for Solving Multi-Objective Vehicle Routing Problem with Time Windows. *Expert. Syst. Appl.* **2021**, *168*, 114408. [\[CrossRef\]](#)
21. Bi, H.; Zhu, X.; Lu, F.; Huang, M. The Meal Delivery Routing Problem in E-commerce Platforms under the Shared Logistics Mode. *J. Theor. Appl. Electron. Commer. Res.* **2023**, *18*, 1799–1819. [\[CrossRef\]](#)
22. Larsen, J. *Parallelization of the Vehicle Routing Problem with Time Windows*; Technical University of Denmark: Kongens Lyngby, Denmark, 1999.
23. Cook, W.; Rich, J.L. *A Parallel Cutting-Plane Algorithm for the Vehicle Routing Problem with Time Windows*; Computational and Applied Mathematics Rice University: Houston, TX, USA, 1999.
24. Kallehauge, B.; Larsen, J.; Madsen, O.B. Lagrangean Duality Applied on Vehicle Routing with Time Windows—Experimental Results. *Comput. Oper. Res.* **2001**, *33*, 1464–1487. [\[CrossRef\]](#)
25. Jung, S.; Moon, B.R. A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows. In Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation, New York, NY, USA, 9–13 July 2002; pp. 1309–1316.
26. Ombuki, B.; Ross, B.J.; Hanshar, F. Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows. *Appl. Intell.* **2006**, *24*, 17–30. [\[CrossRef\]](#)
27. Alvarenga, G.B.; Mateus, G.R.; de Tomi, G. A Genetic and Set Partitioning Two-Phase Approach for the Vehicle Routing Problem with Time Windows. *Comput. Oper. Res.* **2007**, *34*, 1561–1584. [\[CrossRef\]](#)
28. Saksuriya, P.; Likasiri, C. Hybrid Heuristic for Vehicle Routing Problem with Time Windows and Compatibility Constraints in Home Healthcare System. *Appl. Sci.* **2022**, *12*, 6486. [\[CrossRef\]](#)
29. Ikotun, A.M.; Ezugwu, A.E.; Abualigah, L.; Abuhaija, B.; Heming, J. K-Means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data. *Inf. Sci.* **2023**, *622*, 178–210. [\[CrossRef\]](#)
30. Glover, F. Tabu Search—Part I. *ORSA J. Comput.* **1989**, *1*, 190–206. [\[CrossRef\]](#)

31. Bezanson, J.; Karpinski, S.; Shah, V.B.; Edelman, A. Julia: A Fast Dynamic Language for Technical Computing. *arXiv* **2012**, arXiv:1209.5145. [[CrossRef](#)]
32. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V.B. Julia: A Fresh Approach to Numerical Computing. *SIAM Rev.* **2017**, *59*, 65–98. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.