

**An ALNS-based decision support system for scheduling and routing in home healthcare with lunch break constraints****Gökberk Özsakallı<sup>a</sup>, Ömer Öztürkoğlu<sup>a</sup> and Syed Shah Sultan Mohiuddin Qadri<sup>b\*</sup>**<sup>a</sup>*Department of Business Administration, Yasar University, İzmir, Türkiye*<sup>b</sup>*Department of Industrial Engineering, Çankaya University, Ankara, Türkiye***CHRONICLE***Article history:*

Received September 2 2025

Received in Revised Format

December 25 2025

Accepted December 28 2025

Available online December 28 2025

**Keywords:***Home healthcare**Vehicle routing**Personnel scheduling**Lunch break**Decision support system***ABSTRACT**

This study addresses the daily scheduling and routing problem for home healthcare workers while incorporating lunch break requirements. The Home Healthcare Scheduling and Routing Problem is analysed alongside its common constraints, including patient and caregiver time windows, caregiver qualifications, and mandated breaks. To address this, four different variants of an effective Adaptive Large Neighbourhood Search (ALNS) algorithm were developed to provide high-quality solutions. The algorithms demonstrate significant efficiency, solving 30-patient instances optimally within an average of 12 seconds. For scenarios involving 100 patients, they maintained robust performance with a slight increase in computational time of about 54 seconds. Results indicate operational efficiency improvements of up to 36% through optimized travel routes and patient visitation schedules. To translate these findings into practice, a decision support system, the Home Healthcare Decision Support System (HHDSS), was designed to assist administrators by automating the complex task of scheduling and routing of caregivers. Tested using realistic patient data generated from Turkey, the system effectively allocates healthcare resources and improves responsiveness. Overall, the proposed framework shows strong potential as a valuable practical tool for improving the responsiveness and efficiency of home healthcare logistics.

**1. Introduction**

Home healthcare services (HHS) have gained global attention due to the rising elderly population, increasing hospital congestion, and medical costs in hospitals Cissé et al. (2017). According to the U.S. Labor's projections for 2014-2024, home healthcare services are expected to grow by 60% with an additional 800 thousand new jobs (U.S. Labor Department, 2015). It is an important alternative to the conventional healthcare system. An alternative that provides better and customized services at a low cost, not only for the elderly or disabled people but also for patients recovering from illness or surgery in their own homes. This growing relevance has motivated researchers and policy makers to design efficient operational strategies to manage the complexities of daily home healthcare delivery.

One of the most challenging tasks in this domain is the daily assignment and routing of caregivers, commonly known as the Home Healthcare Scheduling and Routing Problem (HHSRP). This problem involves two intertwined decisions: (i) matching patients to caregivers based on qualification and service requirements, and (ii) determining optimal schedules and travel routes for caregivers. These decisions must consider constraints such as time windows, work-hour regulations, skill compatibility, and increasingly, caregiver well-being. Therefore, in the HHSRP, decision makers aim to minimize travel and service costs by finding an efficient route and schedule for the caregivers while simultaneously assigning patients to them. The caregivers begin their route at a healthcare center (HHC), provide the required services to all the assigned patients, and return to HHC. In practice, the HHSRP is known for its operational and computational complexity. Mismanagement can lead to unserved patients, excessive caregiver workloads, and high service and transportation costs. According to Holm and Angelsen (2014), home health caregivers in Norway spend 22% of their workday travelling, and in the US, they travel nearly 8 billion miles each year (NAHC, 2015). Addressing these inefficiencies requires intelligent scheduling models and decision support tools, especially since healthcare administrators often lack training in operations research or systems optimization. To this end, this

\* Corresponding author Tel.: +903122331371

E-mail [syedshahsultan@cankaya.edu.tr](mailto:syedshahsultan@cankaya.edu.tr) (S. S. M. Qadri)

ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)

2026 Growing Science Ltd.

doi: 10.5267/j.ijiec.2025.12.007

study proposes a robust, efficient, and scalable solution that can be integrated into a real-world decision support system (DSS) for daily caregiver scheduling.

The origins of research in home healthcare logistics trace back to early studies such as Fernandez et al. (1974), with more detailed studies emerging in the late 1990s (Begur et al., 1997; Cheng and Rich, 1998). Since then, researchers have developed numerous variants of HHSRP addressed using mathematical formulation and heuristic approaches. Two comprehensive literature reviews (Cissé et al., 2017; Fikar and Hirsch 2017) summarize key characteristics such as patient and caregiver time windows, skill requirements, working time, temporal dependencies (e.g., synchronization of multiple services, service precedence requirements), continuity of care, workload balancing, single vs. multiple HHCs, and various transportation modes. More recent reviews (di Mascolo et al., 2021; Goodarzian et al., 2023) highlight solution approaches ranging from exact optimization to metaheuristics.

Among these challenges, the incorporation of lunch or rest breaks has emerged as a vital yet underexplored aspect of HHSRP models. Breaks are essential not only due to labor regulations but also to mitigate caregiver fatigue and burnout, factors linked to decreased service quality and increased error rates (Totterdell and Holman, 2003; Meissner et al., 2007). Therefore, the Home Healthcare Scheduling and Routing Problem with Lunch Breaks (HHSRP-LB) has become a recent focus of interest. The most relevant studies on the HHSRP-LB are briefly summarized in Table 1 based on the key features of the HHSRP. In compliance with real-life practices, most previous studies have focused on a single-day planning horizon and a single HHC in the HHSRP. Similarly, the following three features/constraints are commonly considered in the HHSRP studies:

1. **Patient time windows:** Each patient has a time window within which they must receive care, defining the earliest and latest possible start times for service.
2. **Caregiver qualifications:** A caregiver's skills must align with the patient's needs. These qualifications may include medical expertise, language proficiency, social skills, or gender-specific requirements.
3. **Working time regulations:** A caregiver's maximum allowable working hours per shift are typically enforced using a time window constraint.

In addition to these widely recognized features, some studies also consider the following constraints:

**Temporal dependency constraints:** These include synchronization, disjunctive services, and precedence constraints. Synchronization requires multiple caregivers to provide a service simultaneously. Disjunctive constraints prevent two services from being performed at the same time. Precedence constraints ensure that one service must be completed before another can begin.

**Break constraints:** These allow caregivers to take rest or lunch breaks within a specified time interval. Among these, break requirements are among the most common real-life practices due to labor regulations and personnel needs. Several studies have approached break scheduling from different perspectives. Some allow caregivers to schedule breaks after a specified number of working hours near patient's location without returning to a fixed base (Cheng and Rich, 1998; Villegas et al., 2018), and Yadav and Tanksale, (2022), while others allocate a fixed break period requiring them to return to their respective HHCs (Wang et al., 2024). Since explicitly enforcing a lunch break may lead to overtime and time window violations, overtime costs are incorporated into the objective function. Guo and Bard (2023) do not enforce strict break requirements to avoid these issues. Instead, they implement a post-processing step that integrates breaks into schedules only when it does not increase overtime hours or cause time window violations. However, this approach may force caregivers to sacrifice their lunch break in order to minimize overtime, especially when they need it the most. In contrast, studies, like Trautsamwieser and Hirsch (2011), Wirmitzer et al. (2016), and de Aguiar et al. (2023), allocate breaks based on total working hours, typically a 30-minute break after six hours of work, involving a designated break node before traveling to the next patient. To improve flexibility in break scheduling based on actual service timelines, Méndez-Fernández et al. (2023) introduce a dynamic break approach for both paid and unpaid breaks. In this approach, if the longest break of the day exceeds a certain duration, it is not counted as paid working time. Nabavizadeh et al. (2024) combine both predefined time intervals and working time-based break requirements, allowing caregivers to take a lunch break within a specific time window but only after working a designated number of hours. Additionally, a break node is incorporated into the model, providing a dedicated location for caregivers to take breaks. Considering a balance among teams' workloads, Varas et al. (2024) design the problem in a shift manner (morning and afternoon) with a mandatory lunch break at the hospital.

Regarding solution methodologies, some studies develop exact methods to determine optimal routes and schedules for caregivers (Bachouch et al., 2011; Liu et al., 2017; Kandakoglu et al., 2020). However, the most researchers prefer metaheuristic approaches to derive effective solutions due to the complexity of the HHSRP (Eveborn et al., 2006; Bertels and Fahle, 2006; Rest and Hirsch, 2016; Bazirha, 2023; Méndez-Fernández et al., 2023; Méndez-Fernández et al., 2024).

This study addresses the Home Healthcare Scheduling and Routing Problem (HHSRP) with a lunch break requirement due to its practical relevance. A mathematical model and an efficient solution algorithm are proposed to solve the problem. The problem formulated by Liu et al. (2017) is considered for its alignment with real-world applications. Liu et al. (2017) used an

exact method to derive optimal solutions for their publicly available, randomly generated problem instances. Therefore, we compare the efficiency of our algorithm against these optimal or best-known solutions. Furthermore, this paper introduces the Home Healthcare Decision Support System (HHDSS), designed to assist HHCs in daily scheduling and routing. The HHDSS employs the proposed Adaptive Large Neighborhood Search (ALNS) algorithm to optimize caregivers' routes. The system is tested using instances generated from COVID-19 patient data from Turkey's largest cities.

The remainder of this paper is structured as follows. Section 2 deals with the explanation of Liu et al. (2017)'s HHSRP and its mathematical model. The proposed ALNS algorithm is presented in section 3. Section 4 encompasses the efficiency of the developed ALNS algorithm by comparing its solutions with Liu et al. (2017), the architecture of HHDSS is explained in section 5. Experiments and empirical results on the data set of COVID-19 patients have been discussed in section 6 whereas the last section concludes the manuscript with our final remarks.

## 2. Problem Definition

As mentioned in Section 1, this study considers the Home Healthcare Scheduling and Routing Problem with a lunch break requirement, referred to as HHSRP-LB hereafter. Specifically, we examine the HHSRP-LB as defined and formulated by Liu et al. (2017). The key features of this problem are as follows:

- i. Caregivers start and end their travel at a single HHC in their daily plan.
- ii. Caregivers have a maximum working time per day and must return to the HHC before the end of their shift.
- iii. At the beginning of the day, all patients' requirements are known, and caregivers are ready to provide service at the HHC.
- iv. Each caregiver has a personal car for travelling to patients.
- v. Patients have a hard time windows constraint, meaning caregivers must visit patients within their specified earliest and latest service start time interval, if possible.
- vi. Patients should be visited by an eligible caregiver.
- vii. Patient service times are deterministic and predefined based on service requirements.
- viii. Travel times are deterministic.
- ix. Caregivers must take a lunch break within a predetermined time interval and for a specified duration.
- x. A penalty cost is incurred if a patient cannot be visited due to time window constraints or caregiver working time limitations.

**Table 1**  
The features of the HHSRP-LB in the literature

Reference	Multi HHC	Time Windows	Qualifications	Temporal Dependency	Break	Preferences	Multi Period	Working Time	Overtime	Instance
Cheng and Rich (1998)		✓	✓		✓			✓	✓	Random (NA) & Real-life (NA)
Bertels and Fahle (2006)		✓	✓		✓			✓		Random (NA)
Eveborn et al. (2006)		✓	✓	✓	✓	✓		✓	✓	Real-life (NA)
Bachouch et al. (2011)		✓	✓	✓	✓		✓	✓		Random (NA)
Trautswieser and Hirsch (2011)		✓	✓		✓			✓	✓	Random (NA) & Real-life (NA)
Trautswieser et al. (2011)		✓	✓		✓	✓		✓	✓	Random (NA) & Real-life (NA)
Shao et al. (2012)	✓	✓	✓		✓		✓	✓		Random (NA) & Real-life (NA)
Bard et al. (2013)	✓				✓		✓	✓	✓	Real-life (NA)
Trautswieser and Hirsch (2014)		✓	✓		✓		✓	✓		Random (NA) & Real-life (NA)
Bard et al. (2014a)	✓	✓	✓		✓		✓	✓	✓	Random (NA) & Real-life (NA)
Bard et al. (2014b)	✓				✓		✓	✓	✓	Random based on Real-life (NA)
Fikar and Hirsch (2015)		✓	✓		✓			✓		Real-life (NA)
Rest and Hirsch (2016)		✓	✓		✓	✓		✓	✓	Real-life (NA)
Wimitzer et al. (2016)		✓	✓		✓	✓	✓	✓	✓	Random based on Real-life (NA)
Liu et al. (2017)		✓	✓		✓			✓		Random (A) & Real-life (NA)
Villegas et al. (2018)			✓		✓					Real-life (NA)
Xiao et al. (2018)		✓		✓	✓	✓		✓		Real-life (NA)
Kandakoglu et al. (2020)		✓	✓		✓			✓	✓	Real-life (NA)
Liu et al. (2021)	✓	✓	✓	✓	✓			✓		Random (A)
Yadav and Tanksale (2022)	✓	✓	✓	✓	✓	✓		✓	✓	Random (A)
de Aguiar et al. (2023)		✓		✓	✓			✓		Random (NA)
Méndez-Fernández et al. (2023)		✓	✓		✓		✓	✓	✓	Real-life (NA)
Bazirha (2023)		✓			✓			✓		Random (NA)
Guo and Bard (2023)		✓	✓		✓		✓	✓	✓	Random (A)
Bazirha et al. (2024)		✓	✓		✓			✓		Random (NA)
Varas et al. (2024)	✓	✓	✓	✓	✓		✓	✓	✓	Real-life (NA)
Méndez-Fernández et al. (2024)		✓	✓		✓			✓	✓	Random (NA)
Nabavizadeh et al. (2024)	✓	✓	✓	✓	✓			✓		Random (NA)
Wang et al. (2024)	✓	✓	✓	✓	✓			✓	✓	Random (A)
This study		✓	✓		✓			✓		Random (A) & Real-life (A)

In this problem, the lunch break policy requires caregivers to decide whether to take a break before or after serving a patient while considering time window constraints. This policy adds complexity, making the problem even more challenging than a typical HHSRP. Additionally, patient assignments for each caregiver are determined to ensure qualification requirements, continuity of care, and adherence to special preferences. Finally, the objective of the model is to minimize the total travel cost of caregivers and the total penalty cost for unvisited patients.

Liu et al. (2017)'s formulation of the problem is presented below, with its notations defined as follows.

**Sets:**

$K$ : Set of Available Workers,  $K = \{1, 2, 3, \dots, m\}$

$N$ : Set of Clients,  $N = \{1, 2, 3, \dots, n\}$

$V$ : Set of all nodes,  $V = N \cup \{0, n + 1\} = \{0, 1, 2, 3, \dots, n + 1\}$

$N_k$ : Set indicating the clients whom worker  $k \in K$  can serve

**Parameters:**

$d_i$ : Service duration at client  $i \in N$

$[a_i, b_i]$ : Time Window at client  $i \in N$

$a_i$ : Earliest service time of client  $i \in N$

$b_i$ : Latest service time of client  $i \in N$

$L$ : Maximum working time of the worker

$[0, L]$ : Time Window of the worker

$t_{ij}$ : Travel time between node  $i$  and node  $j, i, j \in V$

$ct_{ij}$ : Travel cost between node  $i$  and node  $j, i, j \in V$

$P$ : Lunch break

$B$ : Lunch duration

$[a_p, b_p]$ : Time Window of lunch break

$Q_{ik}$ : 1, if worker  $k \in K$  is allowed to serve client  $i \in N$ . 0, otherwise

$cp_i$ : Penalty cost if client  $i \in N$  is not served

**Decision Variables:**

$z_i$ : 1, if the client  $i \in N$  is not visited. 0, otherwise

$x_{ijk}$ : 1, if worker  $k \in K$  directly visits client  $j \in N$  after  $i \in N$ . 0, otherwise

$y_{ik}$ : 1, if worker  $k \in K$  takes a break at client  $i \in N$  before service. 0, otherwise

$y_{ik}'$ : 1, if worker  $k \in K$  takes a break at the client  $i \in N$  after service. 0, otherwise

$ts_{ik}$ : Service start time of worker  $k \in K$  at a client  $i \in N$

$ts_{pk}$ : Start time of the lunch break of a worker  $k \in K$

**The mathematical model of Liu et al. (2017)'s HHSRP-LB**

**Objective:**

$$\min \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} ct_{ij} x_{ijk} + \sum_{i \in N} cp_i z_i \quad (1)$$

**subject to:**

$$\sum_{k \in K} \sum_{j \in V} x_{ijk} + z_i = 1, \quad \forall i \in N \quad (2)$$

$$\sum_{j \in V} x_{0jk} = 1, \quad \forall k \in K \quad (3)$$

$$\sum_{j \in V} x_{j(n+1)k} = 1, \quad \forall k \in K \quad (4)$$

$$\sum_{i \in V} x_{ijk} - \sum_{i \in V} x_{jik} = 0, \quad \forall k \in K, j \in N \quad (5)$$

$$\sum_{i \in V} y_{ik} + \sum_{i \in V} y_{ik}' = 1, \quad \forall k \in K \quad (6)$$

$$y_{ik} + y_{ik}' \leq \sum_{j \in V} x_{ijk}, \quad \forall k \in K, i \in V \quad (7)$$

$$ts_{ik} + (t_{ij} + d_i)x_{ijk} \leq ts_{jk} + (1 - x_{ijk})b_i, \quad \forall k \in K, i, j \in V \quad (8)$$

$$ts_{pk} + B \cdot y_{jk} \leq ts_{jk} + (1 - y_{jk})b_p, \quad \forall k \in K, j \in V \quad (9)$$

$$ts_{ik} + (t_{ij} + d_i)(x_{ijk} + y_{jk} - 1) \leq ts_{pk} + (2 - x_{ijk} - y_{jk})b_i, \quad \forall k \in K, i, j \in V \quad (10)$$

$$ts_{pk} + (B + t_{ij})(x_{ijk} + y_{ik}' - 1) \leq ts_{jk} + (2 - x_{ijk} - y_{jk}')b_p, \quad \forall k \in K, i, j \in V \quad (11)$$

$$ts_{ik} + d_i y_{ik}' \leq ts_{pk} + (1 - y_{ik}')b_i, \quad \forall k \in K, i \in V \quad (12)$$

$$a_i \sum_{j \in V} x_{ijk} \leq ts_{ik} \leq b_i \sum_{j \in V} x_{ijk}, \quad \forall k \in K, i \in N \quad (13)$$

$$0 \leq ts_{ik} \leq L, \quad k \in K, i \in \{0, n+1\} \quad (14)$$

$$a_p \leq ts_{pk} \leq b_p, \quad \forall k \in K \quad (15)$$

$$x_{ijk} \leq \min\{Q_{ik}, Q_{jk}\}, \quad \forall k \in K, i, j \in V \quad (16)$$

$$y_{ik}, y_{ik}' \leq Q_{ik}, \quad \forall k \in K, i \in V \quad (17)$$

$$x_{ijk} \in \{0, 1\}, \quad \forall k \in K, i, j \in V \quad (18)$$

$$y_{ik}, y_{ik}' \in \{0, 1\}, \quad \forall k \in K, i \in V \quad (19)$$

$$z_i \in \{0, 1\}, \quad \forall i \in N \quad (20)$$

Objective function (1) minimizes total travel cost and unvisited patient cost. Constraint set (2) ensures that each patient is visited by a caregiver or left unvisited. Constraint sets (3)-(5) impose network flow constraints, ensuring caregivers start from and return to a single HHC. Constraint sets (6) and (7) enforce lunch break constraints. Constraint set (8) ensures the feasibility of the route based on service start times. Constraint sets (9)-(12) determine caregivers' lunch break times. Constraint sets (13)-(15) enforce time window constraints. Finally, Constraint sets (18)-(20) define the feasible values of the decision variables. We refer readers to Liu et al. (2017) for a detailed explanation of the model.

### 3. Solution Algorithm

HHSRP-LB shares many features with a well-known variant of the Vehicle Routing Problem with Time Windows (VRPTW), which is proven to be NP-hard (Lenstra and Kan, 1981). Both problems determine the optimal sequence of patients/customers to be visited by caregivers/vehicles while minimizing the total travel distance/cost. Based on these similarities, we can conclude that HHSRP-LB is at least as complex as VRPTW.

Liu et al. (2017) developed a branch and price algorithm (BP) to optimally solve the problem. The authors developed labeling and Tabu Search algorithms to solve sub-pricing problems. Using randomly generated 168 problem instances, the authors showed that the BP algorithm is more effective than CPLEX commercial solver although it obtains optimal solutions for only 70% of the problem instances (48 unsolved instances). The authors highlighted that the bottleneck of the BP algorithm is

solving the pricing problem which becomes increasingly time-consuming as the problem size increases. To efficiently solve large instances of HHSRP-LB to near optimality, we propose an Adaptive Large Neighborhood Search (ALNS) algorithm that delivers optimal or near-optimal solutions within a short computation time.

ALNS is one of the most widely used metaheuristic algorithms for solving vehicle routing problems (VRPs) in recent years. Originally proposed by Ropke and Pisinger (2006a, 2006b), the algorithm iteratively improves a solution by "ruining" (partially destroying) and then "repairing" it. To achieve this, ALNS employs a set of removal and insertion heuristics designed for solution modification. After generating an initial solution, the algorithm selects a pair of removal and insertion heuristics and applies them to improve the current solution in each iteration. This process continues until the stopping criteria are met. Candidate solutions at each iteration are either accepted or rejected based on a probabilistic simulated annealing criterion. The pseudocode of the proposed ALNS algorithm is presented in Table 2. The following sections provide a detailed explanation of each algorithmic step, along with the selected set of removal and insertion heuristics.

**Table 2**

Pseudocode of the proposed ALNS algorithm A0.

---

**Inputs:**  $\theta$ , total number of iterations.  $\bar{\theta}$ , additional iterations.  $\omega$  update solution iteration.  $\tau_{Or}$ , Or-opt local search iteration.  $\tau_{Break}$ , break local search iteration.  $t$ , current iteration.

---

```

generate an initial solution  $s_{init}$ 
set  $s_{best} := s_{init}$  and  $s_{curr} := s_{init}$ 
while  $t \leq \theta$  and  $s_{best}$  has not improved last  $\bar{\theta}$  iterations
  if  $s_{best}$  has not improved last  $\omega$  iterations then
    apply Random Removal and Regret-3 Insertion to  $s_{curr}$  to generate  $s_{new}$ 
  else
    select a Removal Heuristic randomly and apply to  $s_{curr}$  to generate  $s_{new}$ 
    select an Insertion Heuristic randomly and apply to  $s_{new}$ 
  end if
  if  $(t \% \tau_{Or}) = 0$  then
    apply Or-opt heuristic to  $s_{curr}$ 
  end if
  if  $(t \% \tau_{Break}) = 0$  then
    apply Break local search heuristic to  $s_{curr}$ 
  end if
  if  $\text{cost } f(s_{new})$  meets the Acceptance Criteria, then
     $s_{curr} := s_{new}$ 
    if  $f(s_{new}) \leq f(s_{best})$  then
       $s_{best} := s_{new}$ 
    end if
  end if
   $t = t + 1$ 
end while

```

---

**Output:** Best solution,  $s_{best}$

---

The route of a caregiver is represented as a vector that includes the depot, lunch break and patient nodes. For example, the route of caregiver  $k$  is represented as  $\pi_k = \{v_0, v_1, \dots, v_b, v_i, \dots, v_n, v_{n+1}\}$  where  $v_0$  and  $v_{n+1}$  are depots,  $v_b$  is a lunch break node, and  $v_i$  represent  $n$  patient nodes. At the beginning of the algorithm, each route of caregivers contains depot nodes and a lunch break node. These three nodes cannot be removed from the solution during the search process. Then, a solution  $s$  is represented as the collection of routes of  $m$  caregivers such as  $s = \{\pi_1, \dots, \pi_k, \dots, \pi_m\}$ .

### 3.1. Initial Solution

The initial solution is found by applying regret-3 heuristic with noise, as described in Section 3.3. The regret-3 heuristic avoids the greedy approach that may assign later nodes to less favorable positions, thus increasing the objective value (Perttunen, 1994; Van Breedam, 2001; Sousa et al., 2016). In addition, incorporating a noise function helps assess the algorithm's robustness by starting each replication with a different initial solution. If the best solutions are consistent across replications, the algorithm is considered robust. At the beginning of the algorithm, all patients are placed into the request bank which is a set of patients that have not yet been assigned to any caregiver's route. Regret-3 with noise heuristics applied to all routes in parallel. Any unassigned patients remain in the request bank. Once a feasible solution is found, it becomes the current and best solution.

### 3.2. Removal Heuristics

At each iteration, a randomly selected removal heuristic removes a predetermined number of patients ( $q$ ) from the current solution  $s_{curr}$  and places them to the request bank. We use six common removal heuristics in the ALNS algorithm: random, worst, Shaw, proximity, time, and route removals (Ropke & Pisinger, 2006b; Pisinger & Ropke, 2007; Hemmelmayr et al., 2012; Grangier et al., 2016; Koç et al., 2019; Liu et al., 2023; Hunter, 2007).

- **Random Removal:** Selects  $q$  patients randomly, removes them from the solution, and adds them to the request bank.
- **Worst Removal:** Selects  $q$  patients with the highest travel cost# removes the selected patients from the current solution and adds them to the request bank.
- **Shaw Removal:** The Shaw removal heuristic (Shaw, 1997, 1998) aims to remove patients with similar locations and time windows, increasing the likelihood of assigning nearby patients to the same caregiver, thus reducing route length.

The heuristic starts by randomly removing a patient from the current solution and adding it to the request bank. Then, a patient is randomly selected from the request bank, and its similarity to each patient in the current solution is calculated using a relatedness measure, which combines the differences in distances and time windows between patients. The most similar patient is then removed and added to the request bank. This process repeats until  $q$  patients are removed.

- **Proximity Removal:** Similar to the Shaw removal but only considers the distances between patients when calculating the relatedness measure.
- **Time Removal:** Similar to the Shaw removal but only considers the time windows between patients when calculating the relatedness measure.
- **Route Removal:** Randomly selects a route, removes all patients, and adds them to the request bank, aiming to redesign the route for minimal travel time.

### 3.3 Insertion Heuristics

Insertion heuristics are typically classified as sequential or parallel (Solomon, 1987). Sequential insertion constructs routes one at a time, assigning patients to a selected route without considering other routes. In contrast, parallel insertion heuristics evaluate all routes simultaneously. While sequential insertion is faster, parallel insertion usually leads to better solutions (Liu and Shen, 1999). Thus, we use parallel insertion heuristics in this study, specifically Greedy and Regret- $k$  heuristics, along with their noise versions, which have been shown to improve solution quality (Pisinger & Ropke, 2007; Hemmelmayr et al., 2012; Riberio & Laporte, 2012; Kovacs et al., 2012).

- **Greedy Insertion:** This heuristic assigns patients from the request bank to the best possible position in the caregivers' routes based on insertion cost, considering only feasible assignments. The patient with the lowest insertion cost is placed in the selected route. This continues until all patients are assigned or no further insertion is possible.

Since only one route of caregivers is changed at each iteration, the insertion cost for other routes does not need recalculating after each iteration to speed up the process.

- **Greedy Insertion with Noise:** This variant adds randomness by modifying the insertion cost with a noise term:  $t_{max} * \mu * \varepsilon$ , where  $t_{max}$  is the maximum distance between patients,  $\mu$  is the noise parameter (set to 0.1), and  $\varepsilon$  is a randomly generated number between  $[-1, 1]$ .
- **Regret- $k$  Insertion:** Proposed by Potvin and Rousseau (1993), this heuristic extends greedy insertion by considering not only the best position but the  $k$  best positions for each patient. Patients are assigned to positions that maximize the regret cost, defined as the difference between the costs of the  $k$  best positions. In this respect, the greedy heuristic can be seen as a regret-1 heuristic. In this study, we use regret-2 and regret-3 versions.
- **Regret- $k$  Insertion with Noise:** This variant of regret- $k$  insertion modifies the insertion cost in the same way as greedy insertion with noise.

### 3.4 Lunch Break Position

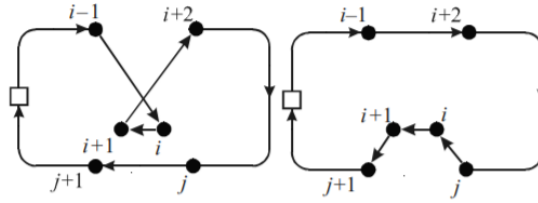
The removal and insertion heuristics described above focus on assigning patients to caregivers and determining the visit sequence. To incorporate lunch breaks, the algorithm is enhanced as follows.

When applying the removal and insertion heuristics, the lunch break node is automatically positioned in the route. A caregiver takes their lunch break when the break node appears before a patient node. Since the break can occur either before or after servicing a patient, the algorithm evaluates both options during each insertion phase and accepts the one that minimizes the caregiver's arrival time at the depot.

### 3.5 Local Search Heuristics

The proposed algorithm contains two local search heuristics to avoid getting trapped in local optimal solutions. The first one is the well-known Or-opt heuristic, introduced by Or (1976). The idea of Or-opt is to relocate a selected sequence of consecutive patients to a different position within the route without changing the router's direction. An example of the Or-opt operation is illustrated in Fig. 1. In this example, nodes  $i$  and  $i+1$  are selected and their edges  $(i-1,i)$  and  $(i+1,i+2)$  are deleted

(as shown in the left figure). Then the selected nodes are relocated between other selected nodes  $j$  and  $j+1$  as seen in the right figure.



**Fig. 1.** Representation of Or-opt heuristic (sourced from Bräysy and Gendreau (2005)).

The second local search heuristic is a novel heuristic that is proposed for this problem, called “Break Local Search Heuristic”. This heuristic focuses on finding the optimal lunch break position for each caregiver's route. The goal is to determine the break position that minimizes the caregiver's total routing time. The pseudocode for the Break Local Search Heuristic is provided in Table 3. Or-opt and break local search heuristics are applied to the new solution at each  $\tau_{Or}$  and  $\tau_{Break}$  iterations, respectively.

### 3.6 Update Solution Criteria and Feasibility Check

To avoid getting trapped in local optima, a solution update mechanism is used (Öztürkoğlu and Hoser, 2014; Kocaman et al., 2019; Öztürkoğlu et al., 2019). The solution update function is called if the best solution has not been improved in the last  $\omega$  iterations. When this function is called, the best solution is set to a current solution. Random removal is selected as the removal heuristic, and Regret-3 insertion is selected as the insertion heuristic. A new solution is generated by applying these heuristics to the current solution, helping to avoid getting stuck in a worse local solution. A patient can only be assigned to a caregiver's route if the assignment is feasible. In the algorithm, the feasibility check is performed only for time window constraints.

**Table 3**

The pseudocode of the break local search heuristic algorithm.

<b>Inputs:</b> $\pi_k$ , route of each caregiver $k$ . $a_{i,k}$ , arrival time to node $i$ of route $k$ .
<b>for each</b> $k \in K$
$\bar{\pi}_k := \pi_k \setminus \{v_b\}$
<b>for each</b> $i \in \pi_k$
insert lunch break node $v_b$ into before node $i$
calculate the new route time; $cost_{i,k} = a_{(n+1),k}$
<b>end for</b>
$i^* = \underset{i}{\operatorname{argmin}} cost_{i,k}$
assign node $v_b$ before node $i^*$ in route $\bar{\pi}_k$
<b>end for</b>
<b>Output:</b> $s_{new} = \{\bar{\pi}_1, \dots, \bar{\pi}_m\}$

### 3.7 Acceptance and Stopping Criteria

When a new solution is accepted, it becomes the current solution for the next iteration. For this, two policies are used; (1) the new solution is accepted if its cost is lower than the current solution's cost, and (2) if the new solution has a higher cost, it may still be accepted with a probability. This probabilistic condition, based on the Simulated Annealing (SA) algorithm (Kirkpatrick, 1983), increases the algorithm's global search capability. The acceptance probability is given by:  $e^{(f(x_{new}) - f(x_{curr}))/T}$ , where,  $f(x_{new})$  and  $f(x_{curr})$  are the costs of the new and the current solutions, respectively;  $T$  is the temperature,  $c$  is the cooling rate that can take a value between 0 and 1. The temperature is decreased by multiplying it with the cooling rate at each iteration (i.e.,  $T = T * c$ ). The initial temperature  $T_{start}$  is set to  $\gamma\%$  higher than the initial solution's cost, where  $T_{start} = -\gamma f(s_{init}) / \ln 0.5$ , ensuring that the initial solution is accepted with a probability of 0.5 (Ropke and Pisinger, 2006a, 2006b).

To determine when to terminate the search process, we first run it for a predetermined number of iterations  $\theta$ . When the algorithm reaches  $\theta$ th iteration, it allows to search  $\theta$  more iterations. If the best solution is improved in the last  $\theta$  iterations, another  $\theta$  iteration is added to the search process. If the best solution cannot be improved in the last  $\theta$  number of iterations, the algorithm terminates and reports the best solution.

### 3.8 Alternative algorithms developed by the selective local search heuristics

To analyze the effectiveness of the proposed local search heuristics, four different variants of the algorithm are considered. The pseudocode of the most comprehensive variant of the proposed ALNS algorithm which includes all local search heuristics was previously presented in Table 2. This configuration is referred to as Algorithm A0. The second configuration includes

only the Or-opt local search heuristic. Therefore, the parameter that controls break local search heuristic,  $\tau_{Break}$  is set to a very large number (i. e.,  $\tau_{Break} \cong \infty$ ). This configuration is referred to as Algorithm A1. The third configuration includes the break local search heuristics and called A2 ( $\tau_{Or} \cong \infty$ ). The last configuration does not contain any local search heuristics ( $\tau_{Break}, \tau_{Or} \cong \infty$ ) and is referred to as Algorithm A3. This serves as the base algorithm, providing a comparison between the different configurations.

#### 4. Computational Study

Liu et al. (2017) tested their branch-and-price algorithm using both real and randomly generated data. While the randomly generated instances are publicly available, real-life data remain confidential. They modified Solomon (1987)'s VRP instances to create HHSRP-LB problem instances. The instances are categorized by patient and caregiver numbers: 30 patients with 4 caregivers, 50 patients with 5 caregivers, and 100 patients with 12 caregivers. Additionally, six classes of Solomon (1987) instances were used, differing in geographical distribution and time window tightness. Each instance is labeled as follows: the first character represents patient distribution in a region; the second indicates time window tightness; the third and fourth numbers form the instance number; and the numbers after the underscore show the number of patients. For instance, in C101\_30, "C" signifies clustered patients, "1" represents narrower time windows, "01" is the instance number, and "30" indicates 30 patients. In total, 168 instances were generated. Further details are available in Liu et al. (2017).

The objective is to minimize the total traveling cost and unvisited patient cost. The proposed ALNS algorithm was implemented in C# and all experiments are carried out on an Intel Core i7 with 2.20 GHz CPU and 16 GB RAM. Liu et al. (2017) run their BP algorithm on an Intel Xeon 2.60 CPU and 128 GB of RAM.

##### 4.1. Parameter Tuning

The proposed ALNS algorithm has nine parameters requiring configuration. Since we adopted the general ALNS framework from Ropke and Pisinger (2006a, 2006b), the following parameters were set according to Ropke and Pisinger (2006a, 2006b)'s implementation: The stopping criterion has two parameters: (1) the total number of iterations ( $\theta$ ) is 25000; (2) the additional number of iterations ( $\hat{\theta}$ ) set to 1500. The cooling rate ( $c$ ), the start temperature control parameter ( $\gamma$ ) and the noise parameters ( $r$ ) were defined as 0.99975, 0.05, and 0.1, respectively. Finally, the Shaw parameters were set as follows:  $(\alpha, \beta) = (0.3, 0.1)$ .

To determine the best values of the remaining parameters, update solution iteration ( $\omega$ ), Or-opt heuristic iteration ( $\tau_{Or}$ ) and break heuristic iteration ( $\tau_{Break}$ ), we conducted a full factorial design of experiment using algorithm A3. We determined 5 levels for  $\omega$  as  $\omega \in \{250, 500, 750, 1000, 1250\}$ , and 9 levels for  $\tau_{Or}$  ranging from 50 to 250 with a step size of 25. Finally, 9 levels for  $\tau_{Break}$  ranging from 75 to 275 with a step size of 25 were specified. A total of 12 instances with 50 patients were randomly selected from Liu et al. (2017) for parameter tuning. Each setting-instance pair was run five times with different random number seeds leading to 24,300 experiments in total ( $9 \times 5 \times 5 \times 5 \times 12$ ). To compare objective values across instances, the relative cost percentage (RCP) is calculated for each instance-setting pair  $i$ :

$$RCP_i = \left( \frac{f_i - f_{min}}{f_{min}} \right) \times 100$$

where,  $f_i$  is the solution found by the algorithm in instance-setting  $i$ , and  $f_{min}$  is the best solution among the experiments for that instance.

We performed ANOVA on Minitab 19 Statistical Software to investigate the effects of parameters on the algorithm's performance with a 95% confidence level. The ANOVA results are given in Table 4. ANOVA results indicate that update solution iteration ( $\omega$ ), break heuristic iteration ( $\tau_{Break}$ ) parameters are statistically significant (their p-values < 0.05), whereas the Or-opt heuristic iteration ( $\tau_{Or}$ ) is not significant. In addition,  $\omega$  has the greatest effect on RCP value since it has the greatest F-value. Moreover,  $\omega \times \tau_{Break}$  is the only significant interaction.

**Table 4**

ANOVA results for parameter tuning.

Source	df	Adj. SS	Adj. MS	F-value	P-value
$\omega$	4	864.38	216.09	13.08	0.00
$\tau_{Or}$	8	10.44	1.31	0.08	1.00
$\tau_{Break}$	8	276.18	34.52	2.09	0.03
$\omega \times \tau_{Or}$	32	75.46	2.36	0.14	1.00
$\omega \times \tau_{Break}$	32	3458.17	108.07	6.54	0.00
$\tau_{Or} \times \tau_{Break}$	64	166.25	2.60	0.16	1.00
$\omega \times \tau_{Or} \times \tau_{Break}$	256	802.53	3.13	0.19	1.00
<b>Error</b>	24151	395623	16.381		
<b>Total</b>	24299	400473			

While the Algorithms A0, A1, and A2 incorporate different local search heuristics, A3 includes none but retains  $\omega$ . Therefore, parameters were tuned separately for each algorithm based on their components.

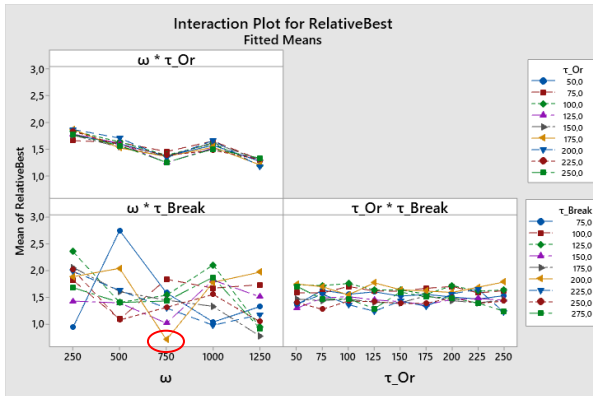


Fig. 2. Interaction effects of parameters.

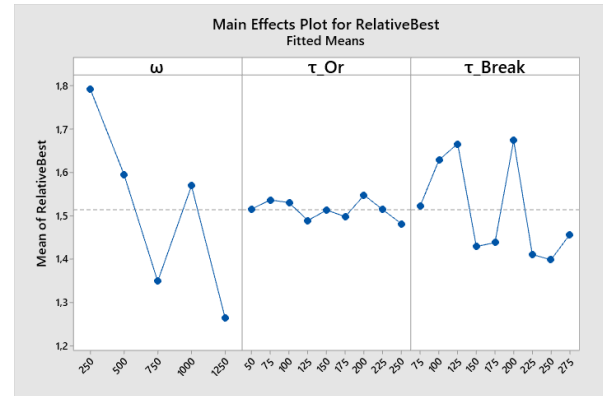


Fig. 3 Main effects plot of parameters.

For A0, which includes both heuristics, we optimized all parameters despite the three-way interaction being statistically insignificant. Using Response Optimizer module of Minitab 19, the optimal values were determined as:  $(\omega, \tau_{Or}, \tau_{Break}) = (750, 150, 200)$ .

In A1, which implements only the lunch break local search heuristic, we optimized  $\omega$  and  $\tau_{Break}$ . ANOVA analysis indicated a statistically significant interaction between them. Based on the interaction plot (Fig. 2), the lowest cost was achieved with  $(\omega, \tau_{Break}) = (750, 200)$ . For A2, which includes only the Or-opt heuristic, we optimized  $\omega$  and  $\tau_{Or}$ . Although their interaction was not statistically significant, the interaction plot (Fig. 2) showed that  $(\omega, \tau_{Or}) = (1250, 200)$  resulted in the lowest cost. Since A3 only includes  $\omega$ , its optimal value was determined using the main effect plot (Fig. 3). The best result was obtained when  $\omega = 1250$ . Finally, Table 5 summarizes the optimized parameter values for all algorithms.

Table 5

Parameters used in the proposed heuristic.

Description	Value	Description	Value
Total number of iterations ( $\theta$ )	25,000	Startup temperature control ( $\gamma$ )	0.05
Additional iterations ( $\hat{\theta}$ )	1,500	Cooling rate ( $c$ )	0.99975
Update solution iterations ( $\omega$ )	750 (in A1 and A0); 1250 (in A2 and A3)	Shaw parameters ( $\alpha, \beta$ )	(0.3, 0.1)
Total number of removed patients	Random (0.1n, 0.3n)	Noise ( $r$ )	0.1
Or-opt heuristic iteration ( $\tau_{Or}$ )	150 (in A0); 200 (in A2)	Break heuristic iteration ( $\tau_{Break}$ )	200 (in A0 and A1)

## 4.2. Computational Results

To assess the robustness of the proposed algorithms, each problem instance was executed five times. We first compare their effectiveness based on the best-found solutions to evaluate the impact of the local search heuristics. Next, we analyze computational times to determine whether the heuristics introduce significant overhead. Finally, we compare the performance of the proposed algorithms with the best-found solutions from Liu et al. (2017).

*4.2.1 Comparisons of the proposed algorithms in terms of their best-found solutions:* Tables A1, A2, and A3 in the appendix present the best solutions (referred to as "Best") obtained by Algorithms A0, A1, A2, and A3 across five replications for 30-, 50-, and 100-patient instances. These tables also include the number of unvisited patients in the Best solutions and the average computational time for five runs. For clarity, the average of the best solutions is reported only for A0.

To test whether the best objective values differ significantly between algorithms, we conducted a one-way ANOVA using the Best solutions. The null hypothesis  $H_0$  states that the mean best objective values are the same across algorithms. Using Minitab 19 with a 95% confidence interval, the results in Table 6 indicate no significant difference between algorithms, regardless of the number of patients ( $H_0$  is rejected because the p-value is 1.0). Tukey's pairwise comparison further confirms that all proposed algorithms belong to the same performance group.

Although the algorithm variants are not statistically different, A0 slightly outperforms the others, especially for larger problem instances, providing the lowest mean cost (10087). This suggests that the insertion and removal heuristics in the ALNS algorithm, along with the selected update parameters, work efficiently for larger instances. Additionally, considering the lunch break position during the removal and insertion processes (as discussed in Section 3.4) appears to reduce the need for an extra local search heuristic for scheduling lunch breaks.

**Table 6**

The comparison of the proposed algorithms in terms of their objective values

Instance Groups	One-way ANOVA results						Grouping information using Tukey Method			
	Source	DF	SS	MS	F	p	Rank	N	Mean	Grouping
30-patient instances	Factor	3	16	5	0.00	1.0	A1	56	1855	A
	Error	220	1325624854	6025568			A0	56	1854	A
	Total	223	1325624870				A3	56	1854	A
	S = 2455 R-Sq = 0.00% R-Sq(adj) = 0.00%							A2	56	1854
50-patient instances	Factor	3	94523	31508	0.00	1.0	A2	56	5210	A
	Error	220	8104196492	36837257			A3	56	5209	A
	Total	223	8104291015				A0	56	5178	A
	S = 6069 R-Sq = 0.00% R-Sq(adj) = 0.00%							A1	56	5162
100-patient instances	Factor	3	142549	47516	0.00	1.0	A3	56	10153	A
	Error	220	25590578913	116320813			A1	56	10141	A
	Total	223	25590721462				A2	56	10136	A
	S = 10785 R-Sq = 0.00% R-Sq(adj) = 0.00%							A0	56	10087

4.2.2 *Comparisons of the proposed algorithms in terms of their average solution times:* To assess the impact of local search heuristics on computational time, we performed a one-way ANOVA. Table 7 presents the results, along with Tukey's pairwise test for grouping information. For 30- and 50-patient instances, the null hypothesis, that all algorithms have the same mean computational time, is rejected ( $p = 0.00$ , 95% confidence). However, Tukey's test shows that A0, A1, and A2 require statistically similar run times, while A3 runs significantly faster. This indicates that local search heuristics add extra computation time, though not significantly different across A0, A1, and A2.

**Table 7**

The comparison of the proposed algorithms in terms of their computational time.

Instance	One-way ANOVA results						Grouping information using Tukey Method			
	Source	DF	SS	MS	F	p	Rank	N	Mean	Grouping
30-patient instances	Factor	3	655.05	218.35	26.90	0.00	A1	56	13.27	A
	Error	220	1785.93	8.12			A2	56	12.45	A
	Total	223	2440.98				A0	56	12.41	A
	S = 2.849 R-Sq = 26.84% R-Sq(adj) = 25.84%							A3	56	8.84
50-patient instances	Factor	3	1449.33	483.11	50.03	0.00	A1	56	23.29	A
	Error	220	2124.63	9.66			A0	56	23.29	A
	Total	223	3573.96				A2	56	22.32	A
	S = 3.108 R-Sq = 40.55% R-Sq(adj) = 39.74%							A3	56	17.16
100-patient instances	Factor	3	140.1	46.7	1.13	0.34	A3	56	54.88	A
	Error	220	9091.4	41.3			A0	56	54.05	A
	Total	223	9231.5				A2	56	53.16	A
	S = 6.428 R-Sq = 1.52% R-Sq(adj) = 0.17%							A1	56	52.86

100-patient instances, ANOVA results indicate no significant difference in run time among the algorithms (see Table 7). This suggests that A3 requires more iterations to reach its best solutions since it lacks local search heuristics for improving the current solution.

#### 4.2.3 Comparisons of the proposed algorithms with Liu et al. (2017)

The best and the average solutions from the five replications are presented in Tables A1, A2, and A3 in the appendix. Table 8 provides a summary comparison between the best solutions found by our algorithms (*Alg\_Best*) and those reported by Liu et al. (2017), called *Liu\_Best*. Since Liu et al. (2017) did not always provide feasible or optimal solutions, we denote the number of optimal solutions found by our algorithm as *#Opt* and those reported by Liu et al. (2017) as *#LiuOpt*. Similarly, *#Imp* and *#Worse* indicate the number of instances where *Alg\_Best* improved or worsened compared to *Liu\_Best*. The performance gap between the two algorithms (*AvgGap*) is calculated as  $100 \times (Liu\_Best - Alg\_Best) / Liu\_Best$ , where the positive value indicates an improvement over *Liu\_Best*. Similarly, *StdGap* indicates the standard deviation of the *AvgGap*. Finally, *AvgCpu* and *AvgCpu\_Liu* indicate the average computational time in seconds.

As shown in Table 8, Algorithm A0 slightly outperforms the others in terms of *#Opt*, *#Imp*, and *AvgGap*. The algorithm improved 36 solutions and found 88% of the optimal solutions. Its worse solutions were only 0.03% and 0.02% above *Liu\_Best* in 30- and 50-patient instances, respectively, while achieving a 0.7% improvement in 100-patient instances. The other algorithms also performed well, finding 83% of the optimal solutions on average and improving 36 instances compared to *Liu\_Best*.

Since A0 provides slightly better solutions than the other algorithms, we selected it for a detailed comparison with *Liu\_Best*. Table A4 in the appendix presents the results of this comparison in detail, while Table 9 summarizes key findings. The following paragraphs explain the notations used in Table 9, with additional clarifications provided in the appendix.

**Table 8**

The brief results of the comparisons of the proposed algorithms with Liu et al. (2017)' best solutions.

Proposed algorithm	Instances with #patients	#LiuOpt	#Opt	#Imp	#Worse	AvgGap (%)	StdGap	AvgCpu (s)	AvgCpu Liu (s)
A0	30	54	53	1	2	-0.03	0.17	12.41	1106.3
	50	44	38	11	7	-0.02	1.13	23.29	1075.5
	100	22	14	24	18	0.66	3.23	54.05	2509.7
A1	30	54	52	1	3	-0.01	0.07	13.27	1106.3
	50	44	33	11	12	-0.28	1.76	23.29	1075.5
	100	22	12	25	19	0.15	3.57	52.86	2509.7
A2	30	54	52	1	3	-0.04	0.21	12.45	1106.3
	50	44	37	11	8	0.09	0.74	22.32	1075.5
	100	22	12	22	22	-0.10	3.97	53.16	2509.7
A3	30	54	52	1	3	-0.01	0.07	8.84	1106.3
	50	44	35	11	10	-0.26	1.75	17.16	1075.5
	100	22	12	24	20	-0.23	4.98	54.88	2509.7

In Table A4, *OptValAvg* and *OptValStd* represent the mean and standard deviation of *Liu\_Best* across problem instances, while *Avg.Unvisited* indicates the average number of unvisited patients in *Liu\_Best*. For comparison, *BestAvg* and *BestStd* denote the mean and standard deviation of the best solutions from Algorithm A0, and *Avg.BestUnvisited* refers to the average number of unvisited patients in those solutions.

**Table 9**

The comparisons of *Best* by algorithm A0 and *Liu\_Best*.

Instance Classes	Dif BestAvg (%)	CV Best	CV Liu	Dif AvgUnvisited	#Opt	#Imp	#Worse
C1_30	0.00	0.79	0.79	0.00	9	0	0
C2_30	0.00	0.02	0.02	0.00	8	0	0
R1_30	0.00	0.83	0.83	0.00	12	0	0
R2_30	0.01	0.06	0.06	0.00	11	0	0
RC1_30	4.00	0.31	0.29	0.2	7	0	1
RC2_30	1.92	0.11	0.10	0.00	6	1	1
C1_50	8.77	0.92	0.85	0.2	8	1	0
C2_50	1.41	0.05	0.04	0.00	6	1	1
R1_50	18.41	0.49	0.36	2.3	6	4	2
R2_50	5.64	0.10	0.08	0.00	6	3	2
RC1_50	-0.80	0.24	0.23	-0.1	7	0	1
RC2_50	2.55	0.10	0.10	0.00	5	2	1
C1_100	16.38	0.63	0.53	1.7	1	4	4
C2_100	1.68	0.07	0.05	0.00	5	0	3
R1_100	35.15	0.49	0.28	10.4	1	5	6
R2_100	16.65	0.08	0.00	0.00	1	10	0
RC1_100	19.86	0.24	0.12	6.4	3	0	5
RC2_100	7.71	0.09	0.06	0.00	3	5	0

In Table 9, *Dif\_BestAvg* is the percentage of the difference between *OptValAvg* and *BestAvg* and calculated as  $100(\text{OptValAvg} - \text{BestAvg})/\text{OptValAvg}$ . *CV\_Best* and *CV\_Liu* refer to the coefficient of variance of the *Best* and *Liu\_Best*. *Dif\_AvgUnvisited* is equal to  $(\text{Avg.Unvisited} - \text{Avg.BestUnvisited})$ . The following observations could be made from the results in Table 9 and Table A4.

1. Liu et al. (2017) presented 120 optimal solutions out of 168 instances, while Algorithm A0 found 105 optimal solutions and improved 36 additional instances. Particularly for 100-patient instances, Algorithm A0 improved solutions in 24 instances, whereas Liu et al. (2017) had an 8.3% average gap due to the complexity of these instances. Consequently, Liu et al. (2017) reported average gaps of 5% and 8.3% for 50- and 100-patient instances, respectively. The gap was especially high for R1-class instances, where patients are randomly distributed with narrow time windows, reaching 14% for 50-patient instances and 22% for 100-patient instances. Algorithm A0 improved solution quality by 6% and 16% for 50- and 100-patient instances, respectively. Specifically, for R1 instances, Algorithm A0 achieved 18.4% and 35% lower costs than *LiuBest* in 50- and 100-patient cases, respectively.
2. The model penalizes unvisited patients due to working-hour constraints, ensuring maximum patient coverage and improved home care service efficiency. As seen in column *Dif\_AvgUnvisited* in Table 9, Algorithm A0 consistently visited more patients than Liu et al. (2017), particularly in larger instances with greater gaps. Specifically, for 100-patient cases,

Algorithm A0 covered 10 more patients in R1 instances and 6 more in RC1 instances. Additionally, for 50-patient R1 instances, Algorithm A0 covered 2 more patients than *LiuBest*.

- Similar to Liu et al. (2017), Algorithm A0 demonstrates robust performance, with an average coefficient of variance of 0.3, comparable to *LiuBest* has 0.27.
- Algorithm A0, enhanced with two additional local heuristics, requires significantly less computation time than Liu et al. (2017). On average, while *LiuBest* required 1100, 1075, and 2510 seconds for solving 30-, 50-, and 100-patient instances, respectively, Algorithm A0 solved them in 12, 23, and 54 seconds. Additionally, Liu et al. (2017) ran their model on a more powerful computer, yet they failed to obtain solutions within three hours for some instances. Moreover, the computational burden of Algorithm A0 increases almost linearly, making it highly scalable.

In conclusion, the proposed algorithm is highly effective for daily HHSRP planning due to its superior solution quality, robustness, and low computational cost. Moreover, because the proposed ALNS algorithm is less complex than Liu et al. (2017)'s BP algorithm, it can be easily integrated into a DSS for practical use.

## 5. Home Healthcare Decision Support System (HHDSS)

A DSS is a desktop-based information system that enhances decision-making by integrating data, information, software packages, and mathematical models, and provides the best possible solution available to the users. In this context, a prototype of a home healthcare decision support system (HHDSS) has also been developed to assist the HHC experts and decision-makers in HHSRP. This desktop application makes use of different Python modules/libraries such as matplotlib, NumPy, scikit-learn, and Tkinter, the standard Graphical User Interface library for Python. This framework provides an object-oriented environment for efficiently creating a DSS.

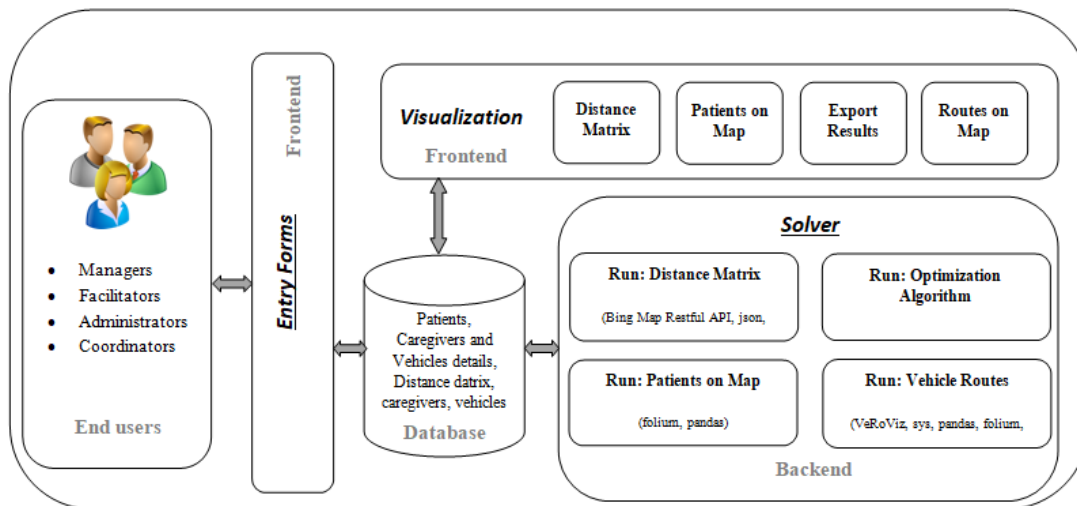


Fig. 4 Architecture of HHDSS

### 5.1. Architecture of HHDSS

The designed HHDSS is a model-driven single installation system that has all the essential programs and databases stored locally. The HHDSS architect is divided into three main parts. "Entry Form", "Solver", and "Visualization", as shown in Fig. 4. This part allows the users to add the details of new patients, caregivers, and vehicles into the database, which is saved in .xlsx format. Additionally, the information about the existing patients, caregivers, and vehicles can also be seen at the bottom of their respective menus to avoid overwriting.

**5.1.1 Entry Forms:** This part allows the users to add the details of new patients, caregivers, and vehicles into the database, which is saved in .xlsx format. Additionally, the information about the existing patients, caregivers, and vehicles can also be seen at the bottom of their respective menus to avoid overwriting.

**5.1.2 Solver:** This component is further divided into four sub-modules: that are Distance Matrix Generator, Patients on Map, Optimization Algorithms, and Vehicles Routes. All these sub-modules can be seen in the system menu (Fig. 5) and can be activated by corresponding "Run:" buttons. For instance, the "Run: Distance Matrix" button, calculates road distances using BingTM Maps Restful API, for which a Python code is developed using JSON and urllib libraries.

For optimizing the vehicle routes, the ALNS code (Algorithm A0) is used which has been explained in Section 3. The ALNS takes three inputs: distance matrix, patient data, and the caregivers' information. By clicking the "Run: Optimization

Algorithm", the .exe file will be executed from the Python environment to generate the result. The result of vehicle route optimization can be seen through the "Export Results" button.

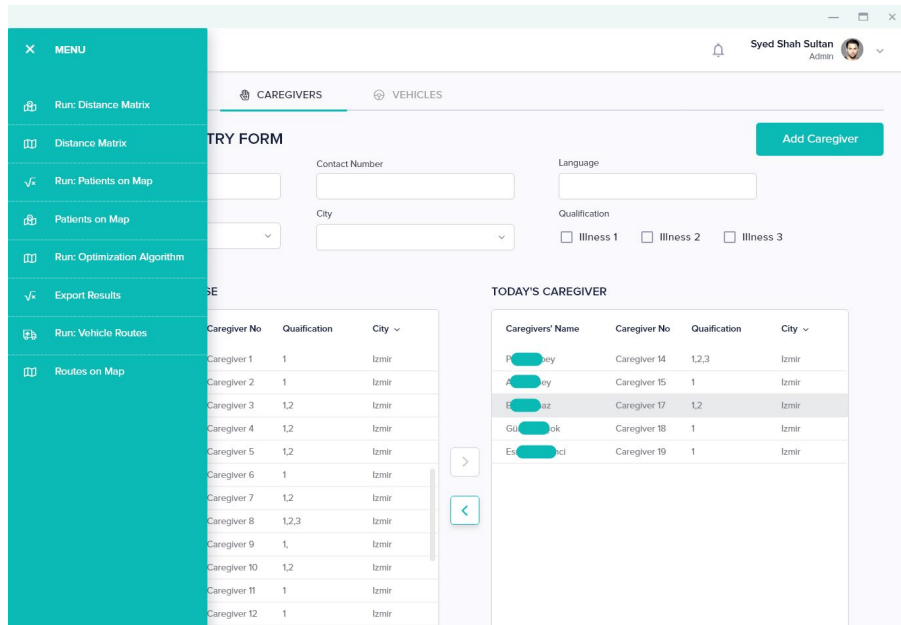
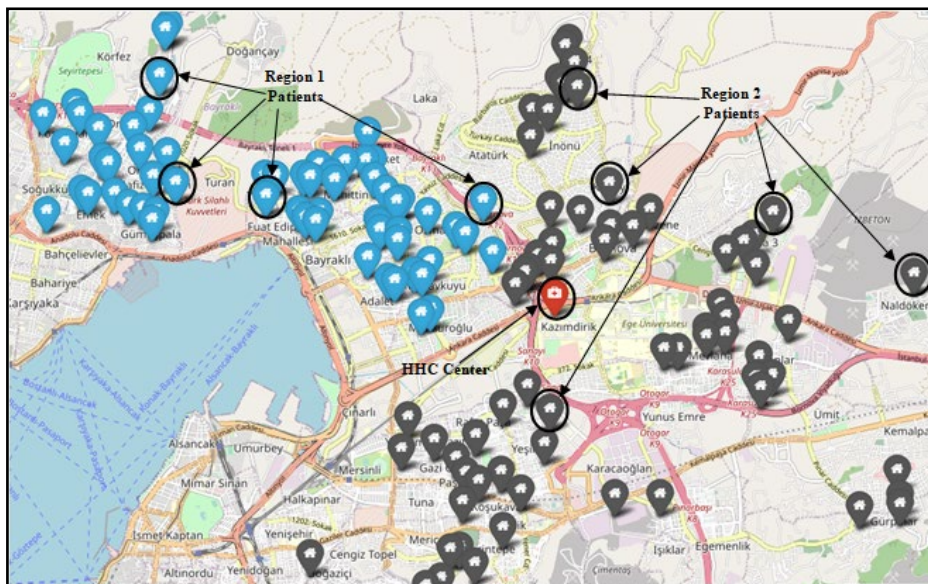


Fig. 5. Outlook of HHDSS menu

- The code for the shortest vehicles' routes according to the actual road network is written in Python 3 and can be run by clicking "Run: Vehicle Routes". VeRoViz (Vehicle Routing Visualization) (Peng and Murray, 2019), an open-source package for generating and visualizing the nodes and vehicle routes on the road networks, and an API from Open Route Service (ORS), a directional service is used while coding. The result of this coding can be seen by clicking the last remaining button, i.e., "Road Routes on Map."



\*Region 1: Bayrakli (Izmir); Region 2: Bornova (Izmir)

Fig. 6. Example representation of patients' locations on the map in Izmir, Turkey.

5.1.3 *Visualization*: The folium library is used to visualize the locations of the patients on an interactive map, executed by the "Run: Patients on Map" button. The results of this operation are viewable as an HTML file ("Patients Map"). Fig. 6 shows an example visualization of the locations of patients on a map that needs to be visited with a distinct color variation depending on the area in Izmir, Turkey. This map is obtained by running the backend code of the "Run Code: Map on Patients" button.

Additionally, the “Visualization” section of HHDSS includes several other interface buttons, such as “Distance Matrix”, Export Results”, and “Routes on Map”. These buttons are used to display results generated from earlier operations. Fig. 7 provides a clearer overview of the entire HHDSS process.

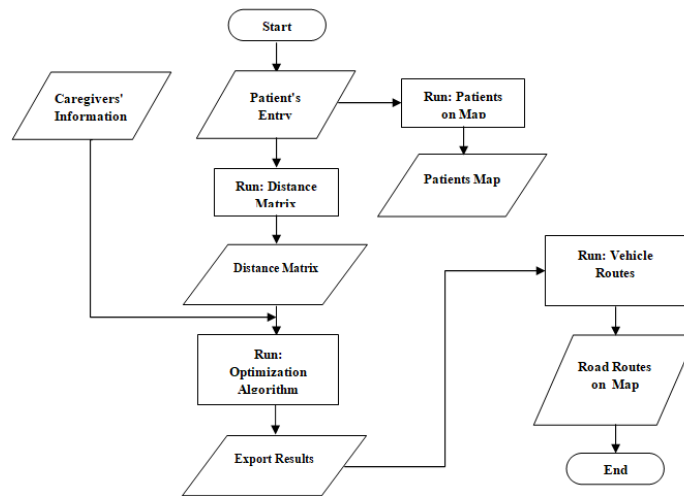


Fig. 7 Flowchart of HHDSS

## 6. Experiments and Empirical Results on the Data Set of COVID-19 Patients as Case Study

In this section, the proposed algorithm is analyzed using COVID-19 patient data. The approximate locations of patients in two neighboring districts of Turkey’s three largest cities, Ankara, Istanbul, and Izmir, were extracted. These cities were identified as high-risk areas for COVID-19 transmission (COVID-19 Istanbul, Ankara, and Izmir Density and Risk Map, September 07, 2020). In Izmir alone, 200-250 positive cases were reported daily (Izmir Coronavirus Table, 2020). Neighboring districts were selected based on their diplomatic, tourism, and financial significance. The locations were obtained using the heatmap feature of the mobile application “Life Fits into Home” (TR Ministry of Health, 2020), developed and updated by Turkey’s Digital Transformation Office to help citizens avoid high-risk areas. Fig. 8 presents heat maps of COVID-19 cases from various provinces in Turkey.

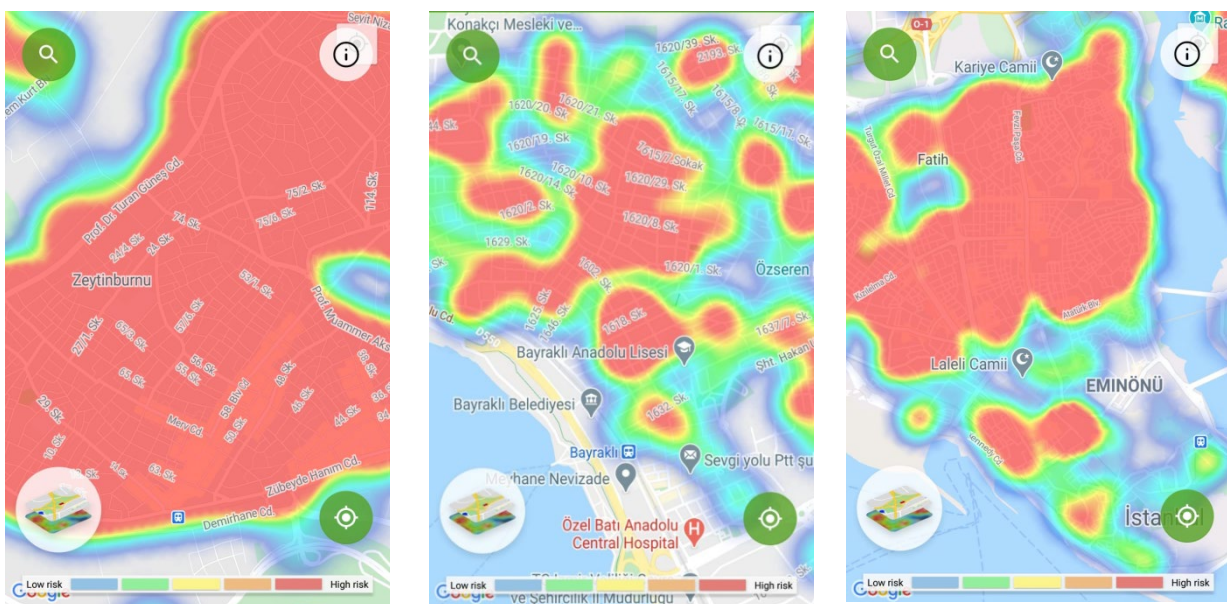


Fig. 8. Heat map images from "Life Fits into Home" application

### 6.1 Test Instances

Small to large instances were generated using patient data from selected districts in Istanbul, Ankara, and Izmir. The largest government hospital in each district was designated as the HHC to provide adequate caregivers. Table 10 details the selected districts, the number of patients, and their distances from the HHC. The minimum, average, and maximum distance of patients from the HHC center is given on the right-hand side of the same table.

**Table 10**

Details of the selected neighborhood with respect to their cities, and the number of caregivers

City	Neighboring Districts	# Patients	Land Areas (km <sup>2</sup> )	Total # Patients	Distance to HHC (km)		
					Min.	Avg.	Max.
Ankara (AN)	Çankaya (C)	195	268	344	0.68	12.6	24
	Altındağ (A)	149	174.5				
Istanbul (IS)	Fatih (F)	101	13.08	184	0.78	4.78	11.61
	Zeytinburnu (Z)	83	12.08				
Izmir (IZ)	Bornova (O)	68	224	131	1.44	6.56	12.71
	Bayraklı (B)	63	30				

Six instance classes were generated for each city, as shown in Table 11. These instances follow the format 50SO1, 100IS2, and ANAC344, representing: 50 randomly selected patients from Bornova in the first instance, 100 randomly selected patients from all of Istanbul (from a total of 184) in the second instance, and all 344 patients from Ankara (A+C) in the third instance.

**Table 11**

Details of the different classes of instances within cities

Classes of Instances (Ankara)	Classes of Instances (Istanbul)	Classes of Instances (Izmir)
SA: Sample of Altındağ	SF: Sample of Fatih	SO: sample of Bornova
SC: Sample of Cankaya	SZ: Sample of Zeytinburnu	SB: sample of Bayraklı
AN: Sample of Ankara	IS: Sample of Istanbul	IZ: Sample of Izmir (B+O)
ANA: Whole Altındağ	ISF: Whole Fatih	IZB: Whole Bayraklı
ANC: Whole Cankaya	ISZ: Whole Zeytinburnu	IZO: Whole Bornova
ANAC: Whole Ankara (A+C)	ISFZ: Whole Istanbul (F+Z)	IZBO: Whole Izmir (B+O)

Three different types of services for caregivers have been considered based on our experience and interviews with practitioners. These types include:

- Type-I service: In this service, a caregiver visits the patient to take samples in order to diagnose COVID-19.
- Type-II service: For administering simple medication and caring of positive COVID-19 patients without chronic diseases.
- Type-III service: This service includes the medication of positive COVID-19 patients with chronic diseases but not in a high-risk group.

**Table 12**

Patients' and caregivers' details concerning the type of services

# patients	# available caregivers	# patients suffering according to the type of illnesses			# available caregivers with respect to the type of illnesses		
		Type-I	Type-II	Type-III	Type-I	Type-II	Type-III
30	2	18	9	3	2	1	0
50	3	30	15	5	3	2	1
100	5	60	30	10	5	3	1
63 (Bayraklı)	4	38	19	6	4	2	1
68 (Bornova)	4	41	20	7	4	2	1
131 (Izmir)	7	79	39	13	7	4	1
40	2	24	12	4	2	1	0
60	3	36	18	6	3	2	1
100	5	60	30	10	5	3	1
101 (Fatih)	6	61	30	10	6	3	1
83 (Zeytinburnu)	5	50	25	8	5	3	1
184 (Istanbul)	10	110	55	19	10	5	2
75	4	45	23	7	4	2	1
100	5	54	27	9	5	3	1
149 (Altındağ)	8	89	45	15	8	4	2
195 (Cankaya)	10	117	59	19	10	5	2
344 (Ankara)	18	206	104	34	18	9	4

High-risk patients requiring hospital or isolation center treatment were excluded as they were required to be treated in hospitals or in some specially built isolation centers. Therefore, the proposed service types were a standard procedure with low variability. However, the service times of different types of services differ from one another. The mean and coefficient of

variation values are estimated for each type of service. The mean service time values of Type-I, Type-II, and Type-III services are set at 10, 15, and 20 minutes, respectively. The coefficient of variation for all services was 0.25, reflecting low variability. Following consultations with practitioners, it was projected that 60%, 30%, and 10% of patients seek Type-I, Type-II, and Type-III services, respectively. On the other hand, hierarchical qualification levels were established for caregivers, which means all of the caregivers are capable of providing Type-I services; 50% of caregivers can handle Type-II, and only 20% of caregivers are qualified for Type-III services. Caregivers are scheduled to work between 9:00 and 18:00, with a mandatory 60 minute lunch break between 11:00 and 14:00. Table 12 provides details on patient and caregiver data for each service type.

## 6.2 Experiment and Results

This section discusses the results of various generated instances of COVID-19 patients. Tables A5-A7 in the Appendix detail the test results, while Table 13 summarizes key findings by city and patient size. These results were obtained using the developed DSS. The average of the total travel time spent by caregivers for all corresponding instances is identified by “Avg. Total Travel Time” in minutes, whereas “Avg. Total Working Time” shows how much time all caregivers spent servicing and traveling in a single day on average. For instance, in Bayraklı, two caregivers spent on average 44.46 minutes traveling and 437.46 minutes (simplified 393 minutes for care) servicing and traveling all 30 patients. Hence, average total service time can be calculated by (Avg. Total Working Time- Avg. Total Travel Time.” “Avg. Cpu” is the average computational time to obtain a good solution, and “Std. Dev.” is the standard deviation of caregivers’ travel time.

**Table 13**

The results of the COVID-19 instances

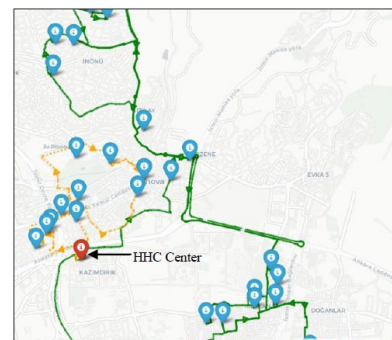
IZMIR					ISTANBUL				
Instance Clusters	Avg. Total Travel Time (min)	Std. Dev.	Avg. Total Working Time	Avg. Cpu (sec)	Instance Clusters	Avg. Travel Time (min)	Std. Dev.	Total Working Time	Avg. Cpu (sec)
30SB	44.46	1.84	437.46	96.12	40SF	37.30	2.04	564.49	172.16
30SO	53.18	3.19	449.93	70.76	40SZ	50.54	3.40	557.22	147.44
30IZ	67.04	5.39	459.45	77.00	40IS	50.59	2.32	606.25	162.64
50SB	60.16	1.88	712.92	179.32	60SF	49.86	2.45	836.84	287.88
50SO	72.22	1.83	720.15	162.32	60SZ	66.96	1.55	831.39	270.64
50IZ	88.11	5.49	739.42	157.20	60IS	66.00	1.49	863.92	287.84
100IZ	142.40	4.51	1278.76	695.56	100IS	98.67	3.85	1420.66	760.56
68IZB	87.96	1.56	1408.96	299.80	101ISF	72.70	0.71	1403.25	813.40
63IZO	74.51	3.40	1396.51	317.40	83ISZ	91.95	1.37	1193.86	559.80
131IZBO	157.31	4.07	1877.43	825.80	184ISFZ	152.10	5.74	2590.32	1238.40

ANKARA									
Instance Clusters	Avg. Travel Time (min)	Std. Dev.	Total Working Time	Avg. Cpu (sec)	Instance Clusters	Avg. Travel Time (min)	Std. Dev.	Total Working Time	Avg. Cpu (sec)
75SA	121.07	2.09	1096.61	423.12	100AN	262.91	12.26	1604.52	652.32
75SC	207.08	14.16	1177.70	385.20	149ANA	203.23	2.38	2167.06	993.00
75AN	206.17	8.57	1187.38	380.04	195ANC	326.99	10.03	2886.16	1158.00
100SA	273.94	9.51	1613.66	631.56	344ANAC	599.56	17.11	5104.66	1678.80
100SC	159.07	3.69	1489.73	732.28					



(a) Zeytinburnu neighborhood (Istanbul) with 2 caregivers covering 40 patients



(b) Bornova neighborhood (Izmir) with 2 caregivers covering 30 patients

**Fig. 9.** The representation of the best-found road routes of vehicles (caregivers) obtained by HHDSS.

In Izmir, travel time increased when combining patients from both neighboring districts, unlike in Istanbul and Ankara. This was due to Istanbul's smaller, denser districts due to high urbanization, and Ankara's large but dispersed patient distribution. This can also be observed by comparing the average travel time spent by the caregivers of 131IZ, 184IS, and 344AN, which are 157.31, 152.10, and 599.96 minutes, respectively.

It is noteworthy that caregivers spend over 82% of their total working time on patient care, with travel time never exceeding 18% across instances. Service time was directly proportional to the number of patients. Fig. 9 demonstrates the vehicle routes for two specific instances, 40SZ and 30SO, with the patient visiting sequence determined by HHDSS.

### 6.3 Benefits of HHDSS:

Although initially tested with COVID-19 patient data, HHDSS has broad applications beyond crisis management. The system is expected to significantly relieve the management of the HHC from the tedious task of scheduling and routing and reduce the administrative burden of manually coordinating caregivers. This allows management to focus on enhancing patient communication, addressing service-related concerns, and improving overall healthcare quality.

Additionally, HHDSS enhances operational efficiency by enabling real-time tracking of caregivers, patients, and vehicle routes. This functionality not only ensures optimized service delivery but also improves caregiver safety, reduces response times, and enhances accountability. The system's data-driven approach helps in better decision-making, allowing healthcare providers to allocate resources more effectively based on demand and urgency.

Beyond pandemic response, HHDSS is highly adaptable for routine home healthcare operations. It can be seamlessly integrated into HHCs managing chronic disease care, elderly patient visits, post-surgical follow-ups, and palliative care. By automating scheduling and minimizing unnecessary travel, it improves service accessibility, reduces caregiver fatigue, and ultimately enhances patient satisfaction.

Moreover, the system prioritizes care quality over cost reduction, ensuring that every patient receives timely and effective treatment. With minor modifications, HHDSS can be tailored to meet the specific needs of different healthcare settings, making it a versatile tool for improving healthcare delivery at scale. Its ability to enhance operational efficiency, optimize resource utilization, and support high-quality patient care highlights its potential as a game-changer in the field of home healthcare services.

## 7. Conclusion

In this study, we address the Home Healthcare Scheduling and Routing Problem with Lunch Break requirements (HHSRP-LB). Specifically, we focus on the problem studied by Liu et al. (2017), which aligns with many real-world applications. Liu et al. (2017) formulated HHSRP-LB using a three-index mathematical model, incorporating key features commonly found in HHSRP literature: single HHC, single-period planning, patient time windows, caregiver working hours, skill requirements, and mandatory lunch breaks for caregivers. The objective is to minimize total travel distance and penalty costs for unvisited patients. Due to the complexity of the problem, Liu et al. (2017) struggled to obtain optimal solutions for large instances within three hours using the branch-and-price algorithm they developed. In practice, managers prioritize obtaining high-quality solutions quickly for daily planning rather than exact optimal solutions. Therefore, we developed an Adaptive Large Variable Neighborhood Search (ALNS) algorithm to efficiently generate high-quality solutions in a short amount of time for Liu et al. (2017)'s problem instances.

We introduced four ALNS variants using different local search heuristics: A0 combines break local search and Or-opt, A1 includes only break, A2 includes only Or-opt, and A3 excludes local search entirely. A design of experiments was conducted to determine the best parameter values for each variant. ANOVA results showed no statistically significant differences among the algorithms in terms of their best solutions, although A0 produced slightly lower costs, particularly for large instances. While A0 required slightly more computational time than A3 due to additional local search, there was no statistical difference for 100-patient instances. These findings indicate that the adapted removal and insertion heuristics, combined with lunch break positioning and Or-opt local searches, effectively enhance the search process without significantly increasing computation time.

Compared to Liu et al. (2017)'s best results, the proposed algorithms achieved approximately 85% of the optimal solutions and improved 36 problem instances. Although we did not obtain all optimal solutions, the average gap was only 0.03%. Additionally, A0 reduced average route costs by up to 35% and visited up to 10 more patients in certain instances, such as R1\_100, RC1\_100, and R1\_50. The algorithm also demonstrated robustness, with a low coefficient of variance (0.3 on average) across five replications. Furthermore, the computational efficiency of A0 was evident, requiring only 54 seconds on average to solve 100-patient instances.

To address daily home healthcare challenges, we developed the Home Healthcare Decision Support System (HHDSS). Using real-world data, we extracted approximate locations of COVID-19 patients from two neighboring districts in Turkey's three

largest cities—Ankara, Istanbul, and Izmir—via the "Life Fits into Home" mobile application, developed by the Turkish Ministry of Health. The ALNS algorithm A0 was implemented to determine the optimal patient visit sequence while considering real road distances and integrating visualization modules.

In conclusion, our proposed algorithms, improved solutions, and the developed HHDSS make significant contributions to home healthcare scheduling and routing literature. Additionally, the ALNS algorithm can be easily adapted or enhanced with additional local search heuristics for various HHSRP variants. Future research could incorporate constraint-handling techniques or develop specialized heuristics for HHSRP problems with synchronization or precedence constraints. The problem could also be extended to consider multiple HHCs or multi-period planning. Thus, the proposed ALNS algorithm serves as a strong foundation for further advancements in home healthcare optimization.

## References

- Bachouch, R. B., Guinet, A., & Hajri-Gabouj, S. (2011). A decision-making tool for home health care nurses' planning. In *Supply Chain Forum: an International Journal* (Vol. 12, No. 1, pp. 14-20). Taylor & Francis.
- Bard, J. F., Shao, Y., & Wang, H. (2013). Weekly scheduling models for traveling therapists. *Socio-Economic Planning Sciences*, 47(3), 191-204.
- Bard, J. F., Shao, Y., & Jarrah, A. I. (2014a). A sequential GRASP for the therapist routing and scheduling problem. *Journal of Scheduling*, 17(2), 109-133.
- Bard, J. F., Shao, Y., Qi, X., & Jarrah, A. I. (2014b). The traveling therapist scheduling problem. *IIE Transactions*, 46(7), 683-706.
- Bazirha, M. (2023). A novel MILP formulation and an efficient heuristic for the vehicle routing problem with lunch break. *Annals of Operations Research*, 1-26.
- Bazirha, M., Kadrani, A., & Benmansour, R. (2024). Pareto and decomposition based approaches for the multi-objective home health care routing and scheduling problem with lunch breaks. *Engineering Applications of Artificial Intelligence*, 128, 107502.
- Begur, S. V., Miller, D. M., & Weaver, J. R. (1997). An integrated spatial DSS for scheduling and routing home-health-care nurses. *Interfaces*, 27(4), 35-48.
- Bertels, S., & Fahle, T. (2006). A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem. *Computers & Operations Research*, 33(10), 2866-2890.
- Bräysy, O., & Gendreau, M. (2005). Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation Science*, 39(1), 104-118.
- Cheng, E., & Rich, J. L. (1998). A home health care routing and scheduling problem. Technical Report TR98-04, Rice University, ABD.
- Cissé, M., Yalçındağ, S., Kergosien, Y., Şahin, E., Lenté, C., & Matta, A. (2017). OR problems related to Home Health Care: A review of relevant routing and scheduling problems. *Operations Research for Health Care*, 13, 1-22.
- Covidien 19 in Istanbul, Ankara, and Izmir density and risk mapping. (2020, May 10). Hurriyet. Retrieved September 07, 2020, from <https://www.hurriyet.com.tr/galeri-coronavirus-koronavirus-10-mayis-turkiye-tablosu-son-durum-covid-19-istanbul-ankara-ve-izmir-yogunluk-ve-risk-haritasi-41512987>
- de Aguiar, A. R. P., Ramos, T. R. P., & Gomes, M. I. (2023). Home care routing and scheduling problem with teams' synchronization. *Socio-Economic Planning Sciences*, 86, 101503.
- Di Mascolo, M., Martinez, C., & Espinouse, M. L. (2021). Routing and scheduling in home health care: A literature survey and bibliometric analysis. *Computers & Industrial Engineering*, 158, 107255.
- Eveborn, P., Flisberg, P., & Rönnqvist, M. (2006). Laps Care—an operational system for staff planning of home care. *European Journal of Operational Research*, 171(3), 962-976.
- Fernandez, A., Gregory, G., Hindle, A., & Lee, A. C. (1974). A model for community nursing in a rural county. *Journal of the Operational Research Society*, 25(2), 231-239.
- Fikar, C., & Hirsch, P. (2015). A matheuristic for routing real-world home service transport systems facilitating walking. *Journal of Cleaner Production*, 105, 300-310.
- Fikar, C., & Hirsch, P. (2017). Home health care routing and scheduling: A review. *Computers & Operations Research*, 77, 86-95.
- Goodarzian, F., Garjan, H. S., & Ghasemi, P. (2023). A state-of-the-art review of operation research models and applications in home healthcare. *Healthcare Analytics*, 4, 100228.
- Grangier, P., Gendreau, M., Lehuédé, F., & Rousseau, L. M. (2016). An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254(1), 80-91.
- Guo, J., & Bard, J. F. (2023). A three-step optimization-based algorithm for home healthcare delivery. *Socio-Economic Planning Sciences*, 87, 101517.
- Hemmelmayr, V. C., Cordeau, J. F., & Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research*, 39(12), 3215-3228.
- Holm, S. G., & Angelsen, R. O. (2014). A descriptive retrospective study of time consumption in home care services: how do employees use their working time?. *BMC health services research*, 14(1), 439.

- Izmir Coronavirus table is getting heavier: An average of 200 cases per day.* (2020, August 18). Gözlem. Retrieved September 07, 2020, from <https://www.gozlemgazetesi.com/HaberDetay/212/1127998/izmir-koronavirus-tablosu-agirlasiyor-gunluk-ortalama-200-vaka.html>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9\*(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Kandakoglu, A., Sauré, A., Michalowski, W., Aquino, M., Graham, J., & McCormick, B. (2020). A decision support system for home dialysis visit scheduling and nurse routing. *Decision Support Systems*, 130, 113224.
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671-680.
- Kocaman, Y., Öztürkoğlu, Ö., & Gümüšoğlu, Ş. (2019). Aisle designs in unit-load warehouses with different flow policies of multiple pickup and deposit points. *Central European Journal of Operations Research*, 1-33, <https://doi.org/10.1007/s10100-019-00646-9>.
- Koç, Ç., Jabali, O., Mendoza, J. E., & Laporte, G. (2019). The electric vehicle routing problem with shared charging stations. *International Transactions in Operational Research*, 26(4), 1211-1243.
- Kovacs, A. A., Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2012). Adaptive large neighborhood search for service technician routing and scheduling problems. *Journal of Scheduling*, 15(5), 579-600.
- Lenstra, J. K., & Kan, A. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2), 221-227.
- Liu, F. H. F., & Shen, S. Y. (1999). A route-neighborhood-based metaheuristic for vehicle routing problem with time windows. *European Journal of Operational Research*, 118(3), 485-504.
- Liu, R., Yuan, B., & Jiang, Z. (2017). Mathematical model and exact algorithm for the home care worker scheduling and routing problem with lunch break requirements. *International Journal of Production Research*, 55(2), 558-575.
- Liu, W., Dridi, M., Fei, H., & El Hassani, A. H. (2020). Mid-Term Home Health Care Planning Problem with Flexible Departing Way for Caregivers. *Nature Inspired Computing for Data Science*, 871, 29-56.
- Liu, Z., Chen, Y., & Qin, J. (2023). The pollution-routing problem with one general period of congestion. *Journal of Modelling in Management*, 18(5), 1529-1560.
- Liu, W., Dridi, M., Fei, H., & El Hassani, A. H. (2021). Hybrid metaheuristics for solving a home health care routing and scheduling problem with time windows, synchronized visits and lunch breaks. *Expert Systems with Applications*, 183, 115307.
- Meissner, A., Hasselhorn, H. M., Estry-Behar, M., Nezet, O., Pokorski, J., & Gould, D. (2007). Nurses' perception of shift handovers in Europe—results from the European Nurses' Early Exit Study. *Journal of advanced nursing*, 57(5), 535-542.
- Méndez-Fernández, I., Lorenzo-Freire, S., & González-Rueda, Á. M. (2023). An Adaptive Large Neighbourhood Search algorithm for a real-world Home Care Scheduling Problem with time windows and dynamic breaks. *Computers & Operations Research*, 159, 106351.
- Méndez-Fernández, I., Lorenzo-Freire, S., & González-Rueda, Á. M. (2024). A biobjective Home Care Scheduling Problem with dynamic breaks. *arXiv preprint arXiv:2406.03217*.
- NAHC (National Association for Home Care & Hospice). Foundation for Hospice and Homecare and NAHC Hold Press Conference on Miles Traveled Each Year By Home Care Nurses, (December 2015) <https://report.nahc.org/foundation-for-hospice-and-homecare-and-nahc-hold-press-conference-on-miles-traveled-each-year-by-home-care-nurses/> accessed 01 July 2019.
- Nabavizadeh, N., Kayvanfar, V., & Rafiee, M. (2024). A mixed integer linear programming model for quarantine-based home healthcare scheduling under uncertainty. *Healthcare Analytics*, 6, 100356.
- Or, I. (1976), *Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking*, Ph.D. thesis, Northwestern University, Evanston, Illinois.
- Öztürkoğlu, Ö., Gue, K. R., & Meller, R. D. (2014). A constructive aisle design model for unit-load warehouses with multiple pickup and deposit points. *European Journal of Operational Research*, 236(1), 382-394.
- Öztürkoğlu, Ö., & Hoser, D. (2019). A discrete cross aisle design model for order-picking warehouses. *European Journal of Operational Research*, 275(2), 411-430.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12, pp.2825-2830.
- Peng, L., & Murray, C. (2019). VeRoViz: A vehicle routing visualization. Accessed January 05, 2020, from <https://veroviz.org>
- Perttunen, J. (1994). On the significance of the initial solution in travelling salesman heuristics. *Journal of the Operational Research Society*, 45(10), 1131-1140.
- Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8), 2403-2435.
- Potvin, J. Y., & Rousseau, J. M. (1993). A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3), 331-340.
- Rest, K. D., & Hirsch, P. (2016). Daily scheduling of home health care services using time-dependent public transport. *Flexible Services and Manufacturing Journal*, 28(3), 495-525.
- Ribeiro, G. M., & Laporte, G. (2012). An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 39(3), 728-735.
- Ropke, S., & Pisinger, D. (2006a). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4), 455-472.

- Ropke, S., & Pisinger, D. (2006b). A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, 171(3), 750-775.
- Shao, Y., Bard, J. F., & Jarrah, A. I. (2012). The therapist routing and scheduling problem. *IIE Transactions*, 44(10), 868-893.
- Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*.
- Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. *International conference on principles and practice of constraint programming* (pp. 417-431). Springer, Berlin, Heidelberg.
- Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2), 254-265.
- Sousa, T., Morais, H., Castro, R., & Vale, Z. (2016). Evaluation of different initial solution algorithms to be used in the heuristics optimization to solve the energy resource scheduling in smart grids. *Applied Soft Computing*, 48, 491-506.
- Totterdell, P., & Holman, D. (2003). Emotion regulation in customer service roles: testing a model of emotional labor. *Journal of occupational health psychology*, 8(1), 55.
- Trautsamwieser, A., Gronalt, M., & Hirsch, P. (2011). Securing home health care in times of natural disasters. *OR spectrum*, 33(3), 787-813.
- Trautsamwieser, A., & Hirsch, P. (2011). Optimization of daily scheduling for home health care services. *Journal of Applied Operational Research*, 3(3), 124-136.
- Trautsamwieser, A., & Hirsch, P. (2014). A Branch-Price-and-Cut approach for solving the medium-term home health care planning problem. *Networks*, 64(3), 143-159.
- TR Ministry of Health. (2020). *Life Fits into Home* (Version 2.1.1) [Mobile app]. Retrieved from <https://itunes.apple.com/> or <https://play.google.com/>
- US Department of. Labor, Projections of industry employment, 2014–24, (December 2015) <https://www.bls.gov/careeroutlook/2015/article/projections-industry.htm>, accessed 01 July 2019.
- Van Breedam, A. (2001). Comparing descent heuristics and metaheuristics for the vehicle routing problem. *Computers & Operations Research*, 28(4), 289-315.
- Varas, M., Baesler, F., Basso, F., Contreras, J. P., Pezoa, R., Rojas-Goldsack, M. F., & Ronco, R. (2024). A home hospitalization assignment and routing problem with multiple time windows, mandatory returns and perishable biological samples: A Chilean case study. *Computers & Industrial Engineering*, 189, 109951.
- Villegas, J.G., Gueret, C., Mendoza, J.E., Montoya, A. (2018). The technician routing and scheduling problem with conventional and electric vehicles. Working paper. Available at <https://hal.archives-ouvertes.fr/hal-01813887/document> (accessed February 5, 2020)
- Wirnitzer, J., Heckmann, I., Meyer, A., & Nickel, S. (2016). Patient-based nurse rostering in home care. *Operations Research for Health Care*, 8, 91-102.
- Xiao, L., Dridi, M., & El Hassani, A. H. (2018). Mathematical model for the home health care scheduling and routing problem with flexible lunch break requirements. *IFAC-PapersOnLine*, 51(11), 334-339.
- Yadav, N., & Tanksale, A. (2022). An integrated routing and scheduling problem for home healthcare delivery with limited person-to-person contact. *European Journal of Operational Research*, 303(3), 1100-1125.

## Appendix

The notations used in Tables A1, A2 and A3 were partly explained in the manuscript in section 4.2.3. Additionally, in the column *Liu\_Best*, *NA* indicates the unsolved problem instance by Liu et al. (2017) in these tables. Therefore, the best objective value is not known for these problems. In the column *Alg\_Best*, the italic and underlined bold values specify the worse and the improved solutions over *Liu\_Best*, respectively. The other values are the same to *Liu\_Best*.

Table A1

The best-found solutions by the proposed algorithms and Liu et al. (2017) for the problem instances with 30 patients.

Instances	Algorithm A0				Algorithm A1			Algorithm A2			Algorithm A3		
	<i>Liu_Best</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>
C101 30	1383	1383	1	9	1383	1	12	1383	1	9	1383	1	7
C102 30	344	344	0	9	344	0	11	344	0	10	344	0	7
C103 30	303	303	0	12	303	0	12	303	0	12	303	0	7
C104 30	271	271	0	14	271	0	14	271	0	14	271	0	11
C105 30	378	378	0	7	378	0	9	378	0	8	378	0	7
C106 30	1361	1361	1	9	1361	1	12	1361	1	9	1361	1	7
C107 30	344	344	0	9	344	0	12	344	0	9	344	0	7
C108 30	307	307	0	9	307	0	12	307	0	10	307	0	7
C109 30	293	293	0	11	293	0	12	293	0	11	293	0	7
C201 30	370	370	0	11	370	0	12	370	0	11	370	0	7
C202 30	352	352	0	14	352	0	18	352	0	14	352	0	12
C203 30	349	349	0	18	349	0	20	349	0	17	349	0	13
C204 30	339	339	0	21	339	0	24	339	0	22	339	0	16
C205 30	358	358	0	12	358	0	12	358	0	12	358	0	8
C206 30	353	353	0	12	353	0	12	353	0	12	353	0	9
C207 30	354	354	0	12	354	0	14	354	0	12	354	0	9
C208 30	353	353	0	12	353	0	13	353	0	12	353	0	9
R101 30	11379	11379	11	10	11379	11	12	11379	11	11	11379	11	8
R102 30	6408	6408	6	12	6408	6	12	6408	6	12	6408	6	9
R103 30	3438	3438	3	12	3438	3	12	3438	3	12	3438	3	8
R104 30	1479	1479	1	11	1479	1	12	1479	1	11	1479	1	7
R105 30	7448	7448	7	12	7448	7	12	7448	7	12	7448	7	9
R106 30	4443	4443	4	12	4443	4	12	4443	4	12	4443	4	9
R107 30	1488	1488	1	12	1488	1	12	1488	1	12	1488	1	7
R108 30	508	508	0	11	508	0	12	508	0	11	508	0	7
R109 30	4462	4462	4	12	4462	4	12	4462	4	12	4462	4	9
R110 30	2507	2507	2	10	2507	2	10	2508	2	10	2508	2	7
R111 30	1502	1502	1	11	1502	1	9	1502	1	12	1502	1	7
R112 30	501	501	0	9	501	0	9	501	0	9	501	0	7
R201 30	522	522	0	11	522	0	11	522	0	11	522	0	7
R202 30	472	472	0	12	472	0	12	472	0	12	472	0	7
R203 30	457	457	0	16	457	0	14	457	0	16	457	0	12
R204 30	431	431	0	18	431	0	17	431	0	17	431	0	14
R205 30	479	479	0	12	479	0	12	479	0	12	479	0	7
R206 30	467	467	0	12	467	0	12	467	0	12	467	0	11
R207 30	444	444	0	16	444	0	16	444	0	16	444	0	12
R208 30	416	416	0	21	416	0	21	416	0	21	416	0	16
R209 30	475	475	0	12	475	0	12	475	0	12	475	0	11
R210 30	471	471	0	14	471	0	14	471	0	14	471	0	11
R211 30	440	440	0	16	440	0	17	440	0	17	440	0	12
RC101 30	7467	7467	7	10	7467	7	12	7467	7	10	7467	7	7
RC102 30	5544	5544	5	12	5544	5	12	5544	5	12	5544	5	7
RC103 30	3568	3568	3	12	3568	3	12	3568	3	12	3568	3	8
RC104 30	2583	2583	2	11	2583	2	11	2583	2	12	2583	2	7
RC105 30	6515	6515	6	12	6515	6	13	6515	6	12	6515	6	7
RC106 30	5549	5549	5	10	5549	5	12	5549	5	10	5549	5	7
RC107 30	5485	5485	5	12	5485	5	12	5485	5	12	5485	5	7
RC108 30	3527	3564	3	12	3572	3	12	3542	3	12	3542	3	7
RC201 30	775	775	0	10	777	0	12	777	0	9	777	0	7
RC202 30	648	648	0	12	648	0	12	648	0	12	648	0	7
RC203 30	602	602	0	14	602	0	16	602	0	14	602	0	11
RC204 30	NA	<b>549</b>	0	16	<b>549</b>	0	19	<b>549</b>	0	16	<b>549</b>	0	12
RC205 30	702	702	0	12	702	0	12	702	0	12	702	0	7
RC206 30	679	679	0	11	679	0	12	679	0	11	679	0	7
RC207 30	615	615	0	15	615	0	18	615	0	12	615	0	11
RC208 30	552	556	0	19	557	0	21	552	0	19	552	0	12

**Table A2**

The best-found solutions by the proposed algorithms and Liu et al. (2017) for the problem instances with 50 patients.

Instances	Liu_Bes <i>t</i>	Algorithm A0			Algorithm A1			Algorithm A2			Algorithm A3		
		Alg_Best	Avg_Best Unvisited	AvgCpu (s)	Alg_Best	Avg_Best Unvisited	AvgCpu (s)	Alg_Best	Avg_Best Unvisited	AvgCpu (s)	Alg_Best	Avg_Best Unvisited	AvgCpu (s)
C101 50	7778	7778	7	22	7778	7	25	7778	7	22	7778	7	17
C102 50	1904	1904	1	19	1904	1	19	1904	1	19	1904	1	12
C103 50	768	768	0	19	768	0	19	768	0	19	768	0	14
C104 50	637	<b>621</b>	0	24	<b>621</b>	0	29	<b>621</b>	0	24	<b>621</b>	0	19
C105 50	4000	4000	3	20	4000	3	24	4000	3	21	4000	3	17
C106 50	5834	5834	5	21	5834	5	24	5834	5	21	5834	5	17
C107 50	1784	1784	1	19	1784	1	23	1784	1	19	1784	1	14
C108 50	824	824	0	19	824	0	22	824	0	19	824	0	14
C109 50	673	673	0	19	673	0	24	673	0	19	673	0	16
C201 50	813	813	0	17	813	0	21	813	0	17	813	0	13
C202 50	739	739	0	30	739	0	24	739	0	21	739	0	17
C203 50	724	724	0	31	725	0	31	725	0	27	724	0	21
C204 50	713	<b>683</b>	0	35	<b>683</b>	0	35	<b>683</b>	0	30	<b>683</b>	0	24
C205 50	800	800	0	22	800	0	21	800	0	19	800	0	15
C206 50	784	784	0	21	784	0	23	784	0	19	784	0	16
C207 50	779	779	0	24	779	0	24	779	0	21	779	0	16
C208 50	781	785	0	24	785	0	24	785	0	19	785	0	16
R101 50	22648	22648	22	26	22649	22	27	22649	22	22	22649	22	17
R102 50	14717	14717	14	24	14717	14	24	14717	14	22	14717	14	17
R103 50	9794	9821	9	22	9821	9	22	9805	9	22	9805	9	17
R104 50	4743	<b>4739</b>	4	22	<b>4739</b>	4	21	<b>4739</b>	4	22	<b>4739</b>	4	17
R105 50	16713	16713	16	22	16713	16	22	16713	16	22	16713	16	17
R106 50	10797	10797	10	22	10797	10	22	10797	10	22	10797	10	17
R107 50	8757	<b>8749</b>	8	25	<b>8749</b>	8	22	<b>8749</b>	8	22	<b>8748</b>	8	17
R108 50	3773	<b>3769</b>	3	22	<b>3769</b>	3	21	<b>3763</b>	3	21	<b>3763</b>	3	17
R109 50	11780	11780	11	22	11780	11	22	11780	11	22	11780	11	17
R110 50	8808	8829	8	24	8829	8	22	8821	8	22	8821	8	17
R111 50	7810	7810	7	23	7810	7	22	7812	7	22	7812	7	17
R112 50	5782	<b>5765</b>	5	24	<b>5765</b>	5	22	<b>5773</b>	5	22	<b>5773</b>	5	17
R201 50	1090	1090	0	19	1090	0	19	1090	0	19	1090	0	14
R202 50	999	999	0	24	999	0	23	999	0	24	999	0	17
R203 50	876	876	0	26	876	0	26	876	0	26	876	0	19
R204 50	<i>N/A</i>	<b>791</b>	0	28	<b>791</b>	0	28	<b>791</b>	0	28	<b>791</b>	0	22
R205 50	984	984	0	21	984	0	21	984	0	21	984	0	16
R206 50	893	893	0	24	893	0	24	893	0	24	893	0	19
R207 50	833	<b>828</b>	0	29	<b>828</b>	0	28	<b>828</b>	0	28	<b>828</b>	0	21
R208 50	<i>N/A</i>	<b>773</b>	0	32	<b>773</b>	0	31	<b>773</b>	0	32	<b>773</b>	0	24
R209 50	896	896	0	24	896	0	24	896	0	24	896	0	19
R210 50	900	<i>904</i>	0	24	<i>904</i>	0	23	900	0	24	900	0	18
R211 50	807	<i>809</i>	0	28	807	0	28	807	0	28	807	0	21
RC101 50	19686	19686	19	22	19686	19	22	19686	19	22	19686	19	17
RC102 50	14837	<i>15748</i>	15	22	14837	14	22	<i>15748</i>	15	22	<i>15748</i>	15	17
RC103 50	12772	12772	12	23	12772	12	22	12772	12	22	12772	12	17
RC104 50	8869	8869	8	22	8869	8	22	8869	8	22	8869	8	17
RC105 50	17703	17703	17	22	17703	17	22	<i>17704</i>	17	22	17703	17	17
RC106 50	15859	15859	15	22	15859	15	22	<i>16758</i>	16	22	<i>16758</i>	15	17
RC107 50	12835	12835	12	23	<i>12879</i>	12	22	<i>12876</i>	12	22	<i>12836</i>	12	17
RC108 50	10917	10917	10	24	10917	10	22	<i>11806</i>	11	22	<i>11806</i>	10	17
RC201 50	1347	1347	0	19	1347	0	17	1347	0	18	1347	0	12
RC202 50	1243	1243	0	22	1243	0	19	1243	0	21	1243	0	16
RC203 50	1122	<b>1117</b>	0	24	<b>1117</b>	0	23	<b>1117</b>	0	24	<b>1117</b>	0	18
RC204 50	<i>N/A</i>	<b>1004</b>	0	28	<b>1004</b>	0	27	<b>1004</b>	0	28	<b>1004</b>	0	20
RC205 50	1254	1254	0	20	1254	0	19	1254	0	19	1254	0	15
RC206 50	1228	1228	0	20	1228	0	19	1228	0	19	1228	0	16
RC207 50	1073	1073	0	24	1073	0	24	1073	0	24	1073	0	17
RC208 50	998	<i>1015</i>	0	24	<i>1016</i>	0	24	<i>1016</i>	0	24	<i>1014</i>	0	19

**Table A3**

The best-found solutions by the proposed algorithms and Liu et al. (2017) for the problem instances with 100 patients.

Instances	Algorithm A0				Algorithm A1			Algorithm A2			Algorithm A3		
	<i>Liu_Best</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>BestAvg</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>	<i>Alg_Best</i>	<i>Avg. Best Unvisited</i>	<i>AvgCpu (s)</i>
C101 100	19273	19273	17	55	19273	17	57	19273	17	53	19273	17	54
C102 100	10396	10398	8	51	10396	8	49	10398	8	49	10396	8	48
C103 100	4281	4294	2	48	4325	2	46	4324	2	46	4320	2	46
C104 100	1943	1930	0	48	1926	0	49	1936	0	50	1952	0	50
C105 100	13448	13414	11	51	13576	11	52	13414	11	52	13414	11	53
C106 100	12676	12705	10	51	12686	10	52	12732	10	51	12665	10	53
C107 100	7697	8454	6	48	8537	6	48	8470	6	48	9357	7	50
C108 100	5749	5563	3	48	6369	4	47	6421	4	48	6396	4	48
C109 100	2392	2327	0	44	2327	0	44	2335	0	45	2318	0	45
C201 100	2222	2222	0	41	2222	0	41	2222	0	41	2222	0	41
C202 100	2000	2000	0	50	2000	0	49	2000	0	49	2000	0	51
C203 100	1845	1845	0	57	1845	0	56	1845	0	57	1848	0	57
C204 100	1759	1768	0	63	1760	0	63	1754	0	64	1754	0	65
C205 100	2111	2111	0	41	2111	0	42	2111	0	42	2111	0	43
C206 100	2087	2089	0	43	2089	0	43	2089	0	44	2089	0	45
C207 100	2057	2057	0	45	2057	0	45	2058	0	47	2058	0	47
C208 100	2015	2018	0	45	2018	0	46	2018	0	46	2018	0	47
R101 100	41225	41229	40	56	41229	40	58	41225	40	57	41225	40	61
R102 100	26496	26496	25	55	26496	25	57	26496	25	57	26496	25	58
R103 100	16582	16534	15	53	16534	15	54	16534	15	54	16534	15	56
R104 100	10575	10600	9	51	10600	9	52	10541	9	55	10543	9	55
R105 100	30412	30442	29	58	30442	29	58	30428	29	58	30428	29	60
R106 100	18686	19593	18	56	20550	19	55	19596	18	55	19596	18	57
R107 100	13560	12596	11	54	12596	11	53	12574	11	54	12574	11	55
R108 100	7622	7576	6	54	8581	7	52	7576	6	52	8586	7	54
R109 100	20587	21506	20	57	21506	20	57	21516	20	57	21510	20	58
R110 100	16572	16581	15	58	16581	15	55	16524	15	57	16570	15	57
R111 100	13584	13571	12	54	13571	12	53	13541	12	53	14550	13	55
R112 100	11530	10559	9	56	11538	10	55	11534	10	56	10548	9	57
R201 100	2011	2011	0	43	2011	0	41	2011	0	42	2012	0	44
R202 100	1794	1793	0	47	1793	0	47	1793	0	48	1793	0	48
R203 100	1716	1615	0	54	1621	0	54	1615	0	54	1621	0	54
R204 100	NA	1547	0	62	1543	0	64	1543	0	62	1547	0	66
R205 100	1890	1821	0	48	1811	0	48	1798	0	47	1811	0	53
R206 100	1765	1686	0	52	1686	0	52	1686	0	52	1681	0	59
R207 100	1718	1589	0	58	1593	0	58	1596	0	59	1593	0	63
R208 100	NA	1516	0	66	1516	0	66	1516	0	66	1516	0	72
R209 100	1770	1667	0	53	1675	0	53	1663	0	55	1675	0	57
R210 100	1719	1645	0	53	1645	0	52	1661	0	54	1645	0	54
R211 100	1609	1547	0	58	1558	0	58	1553	0	58	1548	0	58
RC101 100	37507	37507	36	57	37507	36	58	37507	36	58	37507	36	61
RC102 100	28703	28703	27	58	28704	27	57	28703	27	59	28703	27	60
RC103 100	20778	21707	20	56	21707	20	57	21738	20	59	21738	20	62
RC104 100	16711	16713	15	57	16713	15	57	16750	15	57	16736	15	62
RC105 100	30823	30823	29	65	30823	29	60	31753	30	59	30823	29	64
RC106 100	28517	28527	27	63	28527	27	57	28531	27	57	28551	27	64
RC107 100	21756	22672	21	63	22672	21	58	23637	22	59	23637	22	64
RC108 100	19744	20715	19	63	19739	18	60	19750	18	60	19750	18	65
RC201 100	2516	2516	0	49	2516	0	43	2516	0	43	2516	0	48
RC202 100	2353	2299	0	53	2299	0	48	2299	0	48	2299	0	48
RC203 100	2135	2078	0	60	2078	0	54	2078	0	55	2081	0	54
RC204 100	NA	1923	0	69	1923	0	65	1925	0	63	1917	0	64
RC205 100	2316	2316	0	54	2316	0	46	2316	0	46	2316	0	46
RC206 100	2199	2199	0	52	2221	0	47	2201	0	47	2201	0	47
RC207 100	2084	2044	0	57	2044	0	53	2056	0	53	2056	0	52
RC208 100	2006	1929	0	66	1929	0	59	1930	0	60	1930	0	58

**Table A4**

The comparison of the best solutions presented by Liu et al. (2017) and provided by algorithm A0 in details.

Instance Classes	Tot. No. Instances	Liu et al. (2017)							Algorithm A0			
		Unsolved	Tgap %	Solved No	OptVal Avg	OptVal Std	Avg. Unvisited	TCPU (s)	BestAvg	BestStd	Avg. Best Unvisited	Avg. Best CPU (s)
C1 30	9	0	0.0	9	553.8	438.4	0.2	49.3	553.78	438.4	0.2	9.89
C2 30	8	0	0.0	8	353.5	8.1	0.0	1517.7	353.50	8.1	0	14.00
R1 30	12	0	0.0	12	3796.9	3135.0	3.3	226.0	3796.92	3135.0	3.3	11.17
R2 30	11	0	0.0	11	461.3	27.2	0.0	1178.8	461.27	27.2	0	14.55
RC1 30	8	1	5.3	7	5244.4	1544.0	4.7	923.6	5034.38	1547.5	4.5	11.38
RC2 30	8	1	---	7	653.3	67.9	0.0	3416.8	640.75	71.6	0	13.63
C1 50	9	1	3.5	8	2945.6	2491.2	2.1	546.5	2687.33	2459.7	1.9	20.22
C2 50	8	1	7.2	7	774.3	29.5	0.0	1293.0	763.38	41.1	0	25.50
R1 50	12	4	14.0	8	12883.4	4621.1	12.1	1741.5	10511.4	5165.2	9.8	23.17
R2 50	11	4	0.7	7	948.3	73.0	0.0	1076.5	894.82	92.8	0	25.36
RC1 50	8	0	0.0	8	14184.8	3328.4	13.4	565.8	14298.6	3364.1	13.5	22.50
RC2 50	8	2	4.4	6	1190.5	118.0	0.0	1317.3	1160.13	117.3	0	22.63
C1 100	9	5	7.4	4	10411.8	5556.0	8.0	1970.2	8706.44	5464.0	6.3	49.33
C2 100	8	1	2.6	7	2048.1	107.2	0.0	1765.1	2013.75	136.7	0	48.13
R1 100	12	8	21.9	4	29204.8	8122.9	27.8	2547.0	18940.2	9321.0	17.4	55.17
R2 100	11	10	6.8	1	2011.0	0.0	0.0	4418.9	1676.09	140.9	0	54.00
RC1 10	8	5	6.0	3	32344.3	3751.8	30.7	2769.1	25920.8	6255.0	24.3	60.25
RC2 10	8	5	5.3	3	2343.7	130.9	0.0	4020.8	2163.00	194.1	0	57.50

In Table A4, *Unsolved* indicates the number of instances that were not solved optimally by Liu et al. (2017). *Tgap* presents the average gap between the lower and upper bounds of *Liu\_Best*. *OptVal Avg*, *OptVal Std* refer the mean and the standard deviations of the *Liu\_Best* in the given set of problem instances. Similarly, *Avg. Unvisited* indicate the average number of unvisited patients in *Liu\_Best* for the specified instance class. *TCPU* is the average computational time of *Liu\_Bests*. To make an accurate comparison, *BestAvg* and *BestStd* defines the mean and the standard deviations of the *Best* of the algorithm A0. *Avg. Best Unvisited* and *Avg. Best CPU* indicate the average number of unvisited patients and run time in *Best* provided by the algorithm A0.

**Table A5**

The detail of COVID-19 patients results of Izmir

Instances	Best Total Travel Time	Avg. Total Travel Time	Max Total Travel Time	GAP1 (%)	GAP2 (%)	Avg. CPU
30SB1	45.74	45.82	45.93	0.18	0.25	96.4
30SB2	44.34	44.51	44.61	0.39	0.22	86.8
30SB3	42.94	42.94	42.94	0.00	0.00	97.6
30SB4	42.26	42.26	42.27	0.01	0.02	100.2
30SB5	47.04	47.24	47.42	0.41	0.37	99.6
30SO1	54.64	54.64	54.64	0.00	0.00	65
30SO2	58.24	58.25	58.28	0.02	0.05	66.6
30SO3	49.25	49.28	49.33	0.07	0.10	56.6
30SO4	50.95	50.95	50.95	0.00	0.00	87.8
30SO5	52.82	52.83	52.86	0.01	0.06	77.8
30IZ1	68.20	68.41	68.98	0.30	0.80	74.6
30IZ2	62.60	62.84	63.17	0.39	0.51	86
30IZ3	70.03	70.50	71.11	0.68	0.85	67.4
30IZ4	74.57	74.67	74.82	0.14	0.20	69.2
30IZ5	59.83	59.86	59.87	0.05	0.02	87.8
50SB1	60.97	61.32	61.75	0.56	0.70	183.8
50SB2	61.47	61.68	61.93	0.34	0.40	178.2
50SB3	60.59	60.80	61.14	0.34	0.56	182.8
50SB4	56.82	57.30	57.80	0.84	0.85	184.8
50SB5	60.96	61.20	61.52	0.37	0.52	167
50SO1	72.07	72.52	73.03	0.62	0.68	162.8
50SO2	73.90	74.20	74.73	0.40	0.71	164.6
50SO3	71.99	72.23	72.43	0.32	0.27	160.4
50SO4	69.87	70.46	70.98	0.84	0.72	167.6
50SO5	73.29	73.69	74.36	0.54	0.88	156.2
50IZ1	87.10	87.87	88.75	0.88	0.99	158.6
50IZ2	93.47	94.41	95.59	0.98	1.20	155.2
50IZ3	87.58	88.19	88.99	0.70	0.89	148.8
50IZ4	92.83	93.36	94.32	0.56	1.01	159.4
50IZ5	79.59	79.87	80.16	0.35	0.36	164
100IZ1	144.00	146.84	150.44	1.93	2.39	668.8
100IZ2	142.99	145.70	149.09	1.86	2.27	719.4
100IZ3	141.33	144.62	147.60	2.27	2.01	709.8
100IZ4	144.40	147.19	149.54	1.89	1.57	704.6
100IZ5	139.29	141.54	143.52	1.59	1.37	675.2
68IZB	87.96	88.80	89.72	0.95	1.01	299.8
63IZO	74.51	75.17	76.45	0.89	1.68	317.4
131IZBO	157.31	161.35	165.12	2.51	2.27	825.8

The notations used in Tables A5, A6, and A7 are described as in the followings: “Best Total Travel Time” shows the best found solution that minimizes the total travel time spent by caregivers, “Avg. Total Travel Time” is the average of the best total travel times obtained in each replication, “Max Total Travel Time” is the worst solution, “Gap1” is the percentage gap between the best and the average solutions, and “Gap2” is the percentage gap between the best and the worst solutions. Last, “Avg. CPU” is the average of computational time spent for solving all corresponding instances and replications.



© 2026 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (<http://creativecommons.org/licenses/by/4.0/>).