NTNU
Kunnskap for en bedre verden

Department of Computer Science

IT3708 - Bio-Inspired Artificial Intelligence

# Optimizing Home Care

*Authors:*
Magnus Grini & Yasin Marouga

01.03.2026

# 1 Introduction

## 1.1 Background

In the home care industry, nurses spend a significant amount of time traveling between patients. In order to minimize travel time, meaningful time and resources are used in logistic planning. Artificial intelligence in the form of genetic algorithms can be well suited to taking over this routing task, as they can efficiently navigate these complex search spaces to reach near-optimal solutions efficiently.

## 1.2 Objective

The objective of our genetic algorithm is to create feasible nurse routes that minimize total travel time, while still staying below the nurse load threshold, staying within patient visitation hours, and nurse work hours. Performance is evaluated by comparing the total travel time to the provided benchmark value, with the goal of achieving solutions within 5% of the benchmark.

# 2 Methodology

## 2.1 Chromosome Representation

Each chromosome consists of two integer-encoded components that define the complete schedule:

- `sequence` vector: A permutation representing the global order of patient visits.
- `lengths` vector: A collection of integers representing the number of consecutive patients from the `sequence` assigned to each nurse.

This dual-vector representation maintains a one-dimensional structure for the patient data, significantly reducing computational overhead and memory fragmentation compared to a nested "list-of-lists" structure.

## 2.2 Initial Population

The initial population is generated by creating a random permutation of patients for the `sequence` vector, and by creating a random distribution of route lengths that sums up to the number of patients for the `lengths` vector. This produces a diverse, unbiased initial population.

## 2.3 Fitness Evaluation

Each chromosome is evaluated using a penalty-based fitness function that combines total travel time with constraint violations. For a set of nurse routes, the algorithm simulates each nurse's tour starting at the depot, accumulating travel time, waiting if arriving before a patient's time window, adding care time, and finally traveling back to the depot. Constraint violations are handled indirectly through penalties during this simulation, time-window violations and depot return-time violations receive a base penalty plus linear and quadratic terms proportional to the lateness, while capacity violations are penalized linearly by the amount of overload. Additional penalties are applied for invalid chromosomes like incorrect vector sizes, duplicated/missing patients, or out-of-range indices. If any infeasibility is detected, an additional flat infeasibility gap is added. The final fitness is mapped to the interval

$$f = \frac{1}{1 + T + P + G} \tag{1}$$

where $T$ is total travel time (including depot return time), $P$ is the accumulated penalty, and $G$ is the infeasibility gap (zero for feasible solutions).

## 2.4 Genetic Algorithm

### 2.4.1 Parent Selection and Replacement

Parent selection is performed uniformly at random from the current population. The replacement mechanism is a form of adaptive generalized crowding with elitism [1]. First, an elite subset of the population is transferred directly to the next generation. Elites are ranked by feasibility, then by lowest travel time, and finally by highest fitness. Infeasible solutions are ranked solely by fitness.

The remaining population slots are filled as follows. Two parents are selected at random and generate two offspring through crossover and mutation. Each child then competes with one parent, where the pairing between children and parents is chosen to minimize the total genetic distance across both child–parent pairs. For each pair, the survivor is selected probabilistically. The probability that the child survives is given by:

$$p(c) = \begin{cases} \dfrac{f(c)}{f(c) + f(p)\,\varphi}, & \text{if } f(c) > f(p) \\ \\ \dfrac{f(c)\,\varphi}{f(c)\,\varphi + f(p)}, & \text{if } f(c) \leq f(p) \end{cases}, \tag{2}$$

where $f(c)$ and $f(p)$ denote the fitness of the child and parent, respectively, and $\varphi$ is a scaling factor controlling selection pressure [1, p. 54].

The replacement strategy is generational, as a new population is constructed each generation and replaces the previous population in its entirety.

### 2.4.2 Crossover

The crossover method used in the genetic algorithm is edge recombination [2, p. 21]. This method was chosen because it preserves some edge information while keeping the route as a permutation.

An adjacency table is built from both parents by adding each nodes neighbors to its list while removing duplicates. The child's `sequence` vector is constructed sequentially by picking a start node from either parent, appending the current node, marking it visited, and removing it from all adjacency lists. The next node is chosen from the current node's unvisited neighbors, if multiple candidates exist, the neighbor with the smallest remaining adjacency degree is selected, with ties broken at random. If the current node has no unvisited neighbors, a random unvisited node is chosen. This continues until all patients are inserted. Finally the lengths vector is inherited independently by copying it from either parent with 50/50 probability.

This method differs from the typical edge crossover method by not prioritizing common edges, as from testing, classic edge crossover often failed to discover feasible solutions.

### 2.4.3 Mutation

Mutation is used to increase exploration by discovering new solutions. As the search space for vehicle routing problems is highly complex, a high degree of exploration is beneficial. To achieve this, the genetic algorithm implements one of four mutation operations, chosen at random for each mutation event:

- **Swap mutation**: Swaps the order of two randomly chosen patients in the `sequence` vector, irrespective of existing nurse routes.
- **Relocation mutation**: Relocates a patient from one nurse's route to another nurse's route.
- **Inversion mutation**: Inverts a random section of a nurse's route.
- **Length mutation**: Decreases the length of a nurse's route, and increases the length of another nurse's route

#### 2.4.4 Adaptive Variables

In order to maximize early exploration and avoid premature convergence, mutation rate and scaling factor were made to dynamically adjust based on entropy. Mutation rate starts low and increases as entropy drops. This allows the genetic algorithm to re-diversify itself, so that it can continue to find new solutions, rather than staying converged. The adaptive mutation rate is inspired by the scaling factor formula [1, p. 63] and is given by

$$\mu_c = \mu_i \cdot \frac{1 - H_c}{H_i} \tag{3}$$

where $\mu_c$ is current mutation rate, $\mu_i$ is initial mutation rate, $H_c$ is current entropy, and $H_i$ is initial entropy. Scaling factor is a variable that adjusts the selection pressure of parent-child competitions by modifying the fitness of the lower fitness individual. A higher scaling factor decreases selection pressure. In the genetic algorithm, scaling factor starts high and drops as entropy decreases. Scaling factor is given by

$$\varphi_c = \varphi_i \cdot \frac{H_c}{H_i} \tag{4}$$

where $\varphi_c$ is current scaling factor, $\varphi_i$ is initial scaling factor, $H_c$ is current entropy, and $H_i$ is initial entropy [1, p. 63].

#### 2.4.5 Intermittent Local Search

Upon every fixed number of generations, a light local search is run to increase the speed of convergence. This local search uses three methods to iteratively improve the fitnesses of the chromosomes in the population until either no more improvement is found, or until `max_steps` is reached. The methods used in local search are:

- **Route Reversal**: Goes through every segment of a nurse's route and attempts to reverse its order, continuing until a fitness improvement is found, or until all segments of every nurse's route are tested.
- **Patient Relocation**: Attempts to move a patient from its spot in one nurses route, to a spot in another nurses route. This is repeated for every spot in every other nurses route, or until an improvement in fitness is found.
- **Patient Swapping**: Swaps a patient's position with that of another patient in another nurse's route. This is repeated for every patient pair or until an improvement in fitness is found.

#### 2.4.6 Termination Criteria

The genetic algorithm terminates if no improvement in the best-so-far fitness is observed for a fixed number of consecutive generations. This stall-based method prevents unnecessary runtime while allowing the search to continue as long as progress is made.

### 2.5 Iterated Local Search

#### 2.5.1 Initialization from GA

Once the genetic algorithm has stagnated for a fixed number of generations, the fittest feasible chromosomes are selected to undergo an iterated local search (ILS) to attempt to reach a near optimal solution. This process is parallelized, as each candidate undergoes the final local search individually.

#### 2.5.2 Search Procedure

The iterated local search procedure has two steps that are continuously repeated; mutation and local search. The mutation uses the same mutation algorithm described in Section 2.4.3 but with a guaranteed mutation chance and a higher mutation strength. The mutation strength defines how many times mutation is applied, and is adaptively adjusted throughout the search procedure, increasing based on iterations since fitness improvement, given by

$$M = 2 + \frac{S_i}{40} \tag{5}$$

where $M$ is mutation strength and $S_i$ is iterations since improvement.

The local search is exactly the same as the intermittent local search described in Section 2.4.5 but with significantly higher `max_steps`. If the new chromosome is better than the current chromosome, the new chromosome replaces the current one. This cycle of mutation and local search ensures that the best possible variant of the found niche is created.

### 2.5.3 Termination Criteria

The search procedure for each candidate chromosome is terminated if its improvement has stagnated for a fixed number of iterations. Once all threads have terminated, the final lowest travel time solution is returned.

# 3 Experimental Setup

## 3.1 Dataset

Experiments were carried out on instances provided in JSON format. Each instance specifies the depot location, nurse capacity, number of nurses, patient coordinates, patient demand, and various time constraints, together with a complete travel time matrix. The objective is to minimize total nurse travel time while satisfying all constraints. The provided benchmark value represents the best known travel time for the instance. Separate training and testing datasets were provided.

## 3.2 Parameter Tuning

The memetic algorithm depends on a set of hyperparameters. A random search was performed on a subset of these parameters, which were considered the most influential. The tuned hyperparameters are listed below:

| **Parameter** | **Meaning** | **Range** |
|---|---|---|
| `refinement-candidates` | Individuals selected for refinement | $4, 6, 8, 10, 12$ |
| `ils-iterations` | Iterations of iterated local search (ILS) | $150, 250, 300, 400, 500$ |
| `ils-local-search-steps` | Local search steps per ILS iteration | $[40, 120]$ |
| `elite-size` | Number of elite individuals preserved | $2, 4, 6, 8, 10$ |
| `initial-mutation-rate` | Initial mutation rate ($\mu_i$) | $[0.6, 0.99]$ |
| `initial-scaling-factor` | Initial scaling factor ($\varphi_i$) | $[0.3, 3.0]$ |
| `ls-every` | Generations between local search runs | $100, 250, 500, 750, 1000$ |
| `ls-steps` | Steps in each local search | $[5, 40]$ |
| `stagnation-limit` | Generations without improvement before restart | $[300, 2500]$ |

Table 1: Hyperparameters and search ranges used during random search.

Hyperparameter tuning was performed in two stages: an initial shallow search across many configurations, followed by a reevaluation of the best-performing configurations. To rank each trial, the following scoring function was used:

$$S = 1000 \cdot \text{infeasible\_rate} + \text{mean\_gap\_pct} + 0.2 \cdot \text{std\_gap\_pct},$$

where `infeasible_rate` is the proportion of infeasible solutions generated by the algorithm, `mean_gap_pct` is the mean benchmark gap across the datasets in the trial, and `std_gap_pct` is the standard deviation of the benchmark gap.

The best configuration identified during the second stage of hyperparameter tuning is shown in Table 2. This configuration achieved a mean benchmark gap of 2.48% in the training datasets, with a standard deviation of 1.19% and no infeasible solutions.

| **Parameter** | **Value** |
|---|---|
| Population size | 200 |
| Refinement candidates | 5 |
| ILS iterations | 250 |
| ILS local search steps | 70 |
| Elite size | 10 |
| Initial mutation rate ($\mu_i$) | 0.8526 |
| Initial scaling factor ($\varphi_i$) | 2.0921 |
| Local search interval (`ls-every`) | 100 |
| Local search steps (`ls-steps`) | 24 |
| Stagnation limit | 2156 |

Table 2: Best hyperparameter configuration obtained from the second stage of the random search.

### 3.3 Execution Protocol

Each experiment consisted of a genetic algorithm (GA) phase followed by a parallelized iterated local search (ILS) phase. The initial population was generated randomly. After the GA terminated, the best feasible individuals from the final population, together with the best individual found during the run, were refined using ILS. The best solution obtained during the run was reported as the result of the run. To account for stochasticity, each test instance was executed 10 times using different random seeds.

### 3.4 Evaluation Metrics

Performance was evaluated using the total travel time of feasible solutions and the percentage gap to the benchmark. During optimization, best fitness, population entropy, and lowest feasible travel time per generation were recorded to analyze convergence. Convergence behavior was visualized using plots of lowest feasible travel time across generations and refinement iterations, and the final solution was visualized as a nurse route network.

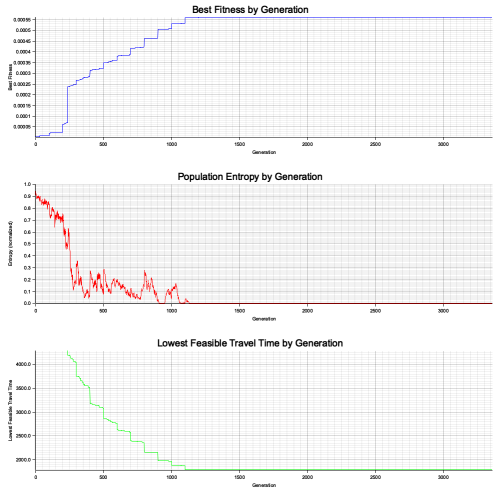
## 4 Results

Table 3 shows the final results of the memetic algorithm and compares the performance of the GA and the GA+ILS algorithm across the test instances. The results demonstrate that the ILS phase substantially improves solution quality, reducing the average travel time by approximately 13–19%. Across all instances, the algorithm consistently produces solutions within roughly 2–3% of the benchmark values. The results were generated using the hyperparameters listed in Table 1. Through additional manual tuning, the benchmark travel times for datasets `test_1` and `test_3` were matched or surpassed.

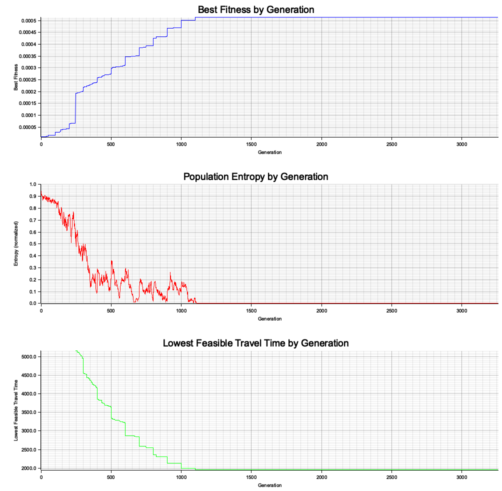
Convergence behavior is shown in Figure 1, refinement performance is shown in Figure 2, and the spatial structure of the final solution is shown in Figure 3.

| Dataset | Benchmark | GA avg | GA gap (%) | GA+ILS avg | Best | Best gap (%) | Avg gap (%) | Std. dev. |
|---|---|---|---|---|---|---|---|---|
| test_1 | 1512.76 | 1755.28 | 16.04 | 1552.28 | 1521.57 | 0.58 | 2.61 | 18.33 |
| test_2 | 1592.80 | 1854.18 | 16.41 | 1626.58 | 1601.55 | 0.54 | 2.12 | 11.49 |
| test_3 | 2301.75 | 2807.04 | 21.95 | 2363.63 | 2326.11 | 1.06 | 2.69 | 33.81 |

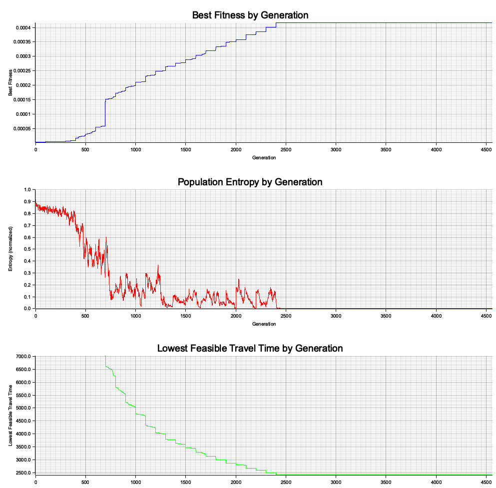
Table 3: Comparison of GA and GA+ILS performance across the test instances. Results are averaged over 10 runs.




(a) Test instance 1

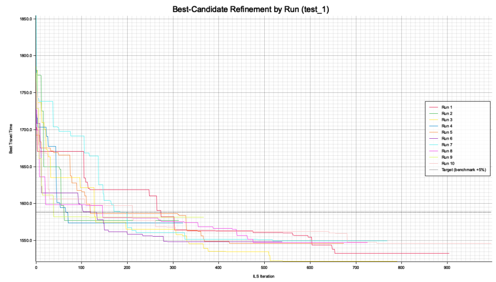



(b) Test instance 2

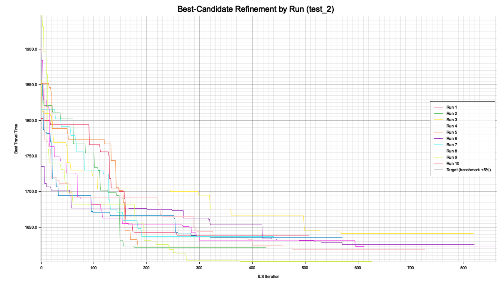



(c) Test instance 3

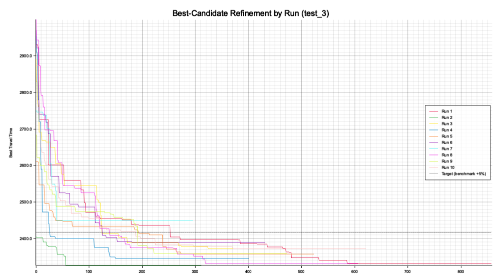
Figure 1: Evolution of best fitness, population entropy, and lowest feasible travel time across generations for the three test instances.



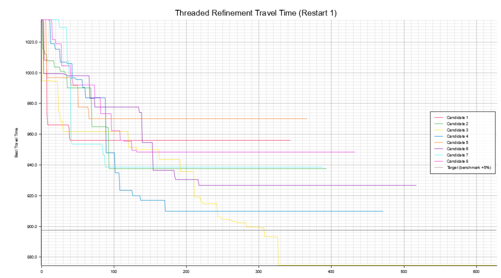

(a) Test instance 1




(b) Test instance 2

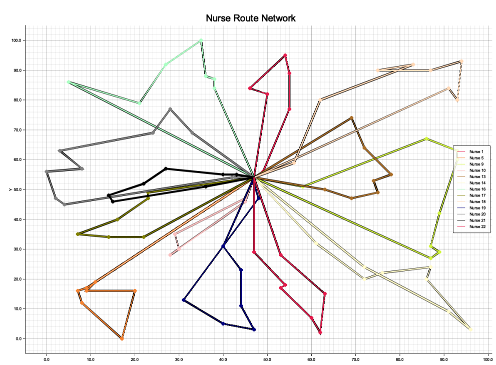



(c) Test instance 3

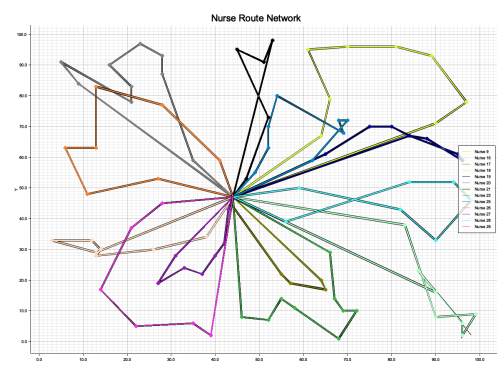



(d) Typical intra-run refinement

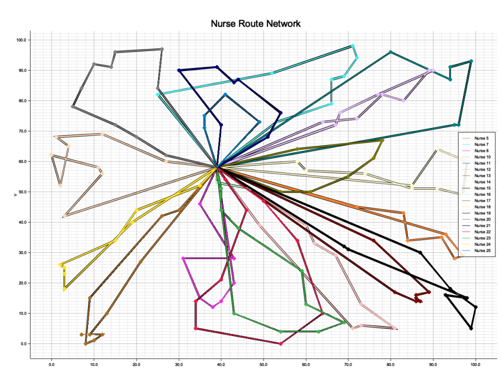
Figure 2: Best candidate travel time across refinement iterations for each run during the final local search for the three test instances, along with an example of typical intra-run refinement.




(a) Test instance 1




(b) Test instance 2




(c) Test instance 3

Figure 3: Nurse route network for the final best feasible solution for each test instance.

# 5 Findings

The convergence plots in Figure 1 show that best fitness initially increases gradually as solutions approach feasibility. Once the first feasible solution is discovered, fitness increases sharply. When this feasible solution is found, the entropy drops, indicating that the feasible solution is out-competing other parts of the population. After the initial entropy drop, mutation rate increases from the low entropy, causing another spike in entropy which in turn allows for the exploration of more solutions. This oscillation continues until entropy stabilizes and fitness converges. The continued improvement in fitness while entropy is zero occurs because entropy measures only route order and not nurse partitioning. The sharp increases in fitness at each 100 generation interval can be explained by the local search, which is applied to every chromosome in the population every 100 generations (as decided by the hyperparameter tuning).

Figure 2a–c shows that the final local search is consistently able to significantly reduce total travel time after the convergence of the genetic algorithm. The figure also shows that the best refinement candidate in most runs reaches within the 5% benchmark threshold in all three test instances.

Figure 2d shows the typical intra-run refinement process for each selected candidate and behaves as follows: in the early refinement iterations, every candidate improves, however different candidates plateau at different iterations and only some reach beyond the threshold. Once the final local search is completed, every candidate has plateaued, indicating convergence to local minima.

The final nurse route networks in Figure 3 show coherent nurse routes, where each nurse primarily serves patients in a single geographic direction, and only a limited number of nurses are utilized. This indicates that the returned solution maintains geographically logical patient assignments and avoids unnecessary travel time.

From the plots in Figure 1, we observe that the fitness appears to converge after approximately 1000–2000 generations on all instances. Setting `stagnation-limit` = 2156 can therefore be considered inefficient, as substantial computational effort is spent after the GA stage has effectively converged. This suggests that the hyperparameter tuning may have been somewhat shallow. Furthermore, since the tuning procedure relied on random search rather than an exhaustive exploration of the parameter space, it is possible that better-performing configurations were not sampled. A more extensive tuning process could therefore yield improved parameter values. Given more time, an additional stage of hyperparameter tuning (see Section 3.2) would likely be conducted, iterating on the findings of the current search while also exploring alternative tuning strategies, such as *Bayesian optimization*, which can guide the search more efficiently than purely random sampling [1, p. 37].

During the development of the algorithm, several parent selection operators were evaluated, including tournament selection. Initially, tournament selection introduced excessive selection pressure during the GA phase. Since the purpose of the GA stage is primarily exploration/discovery of niches (see Table 3), while most exploitation is intended to occur during the LS phase, uniform parent selection was ultimately adopted to maintain greater population diversity.

We also experimented with *greedy initialization*; however, it provided little improvement in the convergence rate of the GA phase while reducing the diversity of the initial population. Consequently, it was omitted from the final algorithm.

# 6 Conclusion

This report presented an adaptive genetic algorithm with intermittent and final local search for nurse routing. The method produced feasible routes and achieved results well within the 5% benchmark threshold for each test instance. The results indicate that generalized crowding with adaptive mutation and intermittent local search, followed by final local refinement, can generate geographically coherent and near-optimal nurse routes.

## References

[1] Jon Å. Bergquist and Ole J. Mengshoel. It3708 bio-inspired artificial intelligence. Lecture slides, Norwegian University of Science and Technology (NTNU), 2025–2026. Weeks 7–8: Evolutionary Computing and Generalized Crowding.

[2] Jon Å. Bergquist and Ole J. Mengshoel. It3708 bio-inspired artificial intelligence. Lecture slides, Norwegian University of Science and Technology (NTNU), 2025–2026. Weeks 6: Permuation Representation.

# A Test Instance 1 Output

```
Average travel time across 10 successful runs: 1552.28

=== Best Run ===
Generation 1: Best Fitness = 0.000003331146476405426, Entropy = 0.9250428898059373, Lowest Feasible Travel Time = None
Generation 500: Best Fitness = 0.0003241263180683749, Entropy = 0.08674778734710263, Lowest Feasible Travel Time = 3084.21691777293
Generation 1000: Best Fitness = 0.0005070650537424761, Entropy = 0.04004029812356743, Lowest Feasible Travel Time = 1971.1335410897227
Generation 1500: Best Fitness = 0.0005599719065360578, Entropy = 0, Lowest Feasible Travel Time = 1784.8038739584658
Generation 2000: Best Fitness = 0.0005599719065360578, Entropy = 0, Lowest Feasible Travel Time = 1784.8038739584658
Generation 2500: Best Fitness = 0.0005599719065360578, Entropy = 0, Lowest Feasible Travel Time = 1784.8038739584658
Generation 3000: Best Fitness = 0.0005599719065360578, Entropy = 0, Lowest Feasible Travel Time = 1784.8038739584658
  Refining candidate 1/5 (threaded)...
  Refining candidate 2/5 (threaded)...
  Refining candidate 3/5 (threaded)...
  Refining candidate 4/5 (threaded)...
  Refining candidate 5/5 (threaded)...
    Iteration 50, best travel time 1635.0831
    Iteration 50, best travel time 1664.3992
    Iteration 50, best travel time 1596.3116
    Iteration 50, best travel time 1583.9950
    Iteration 50, best travel time 1616.4375
    Iteration 100, best travel time 1656.9883
    Iteration 100, best travel time 1559.7120
    Iteration 100, best travel time 1621.5095
    Iteration 100, best travel time 1605.5811
    Iteration 100, best travel time 1580.9916
    Iteration 150, best travel time 1558.2340
    Iteration 150, best travel time 1574.7199
    Iteration 150, best travel time 1624.9466
    Iteration 150, best travel time 1580.9916
    Iteration 150, best travel time 1605.5811
    Iteration 200, best travel time 1549.6384
    Iteration 200, best travel time 1565.0309
    Iteration 200, best travel time 1613.7624
    Iteration 200, best travel time 1605.5811
    Iteration 200, best travel time 1580.9916
    Iteration 250, best travel time 1605.5811
    Iteration 250, best travel time 1549.6384
    Iteration 300, best travel time 1605.5811
    Iteration 250, best travel time 1565.0309
    Iteration 250, best travel time 1580.9916
    Iteration 250, best travel time 1597.1227
    Iteration 300, best travel time 1580.9916
    Iteration 300, best travel time 1549.6384
    Iteration 300, best travel time 1557.8147
    Iteration 350, best travel time 1549.6384
    Iteration 300, best travel time 1597.1227
    Iteration 400, best travel time 1549.6384
    Iteration 350, best travel time 1545.2518
    Iteration 350, best travel time 1597.1227
    Iteration 400, best travel time 1597.1227
    Iteration 450, best travel time 1597.1227
    Iteration 400, best travel time 1534.9683
    Iteration 450, best travel time 1534.9683
    Iteration 500, best travel time 1522.3489
    Iteration 550, best travel time 1521.5689
    Iteration 600, best travel time 1521.5689
    Iteration 650, best travel time 1521.5689
    Iteration 700, best travel time 1521.5689
Run X best travel time = 1521.5688 (gap 0.58%)

=== Final Solution ===
Instance: test_instance_1
Nurse capacity: 205
Depot return time: 1204

----------------------
Nurse  1:  Route duration:   89.0  Covered demand: 124  Patient sequence: 12 (236.0 - 309.0) [236 - 354] -> 06 (321.0 - 407.0) [321 - 795] -> 86 (420.6 - 519.6) [408 - 930] ->
↪  73 (777.0 - 870.0) [777 - 1158] -> 21 (882.0 - 972.0) [250 - 1117] -> 14 (993.8 - 1082.8) [668 - 1359]
Nurse  2:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  3:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  4:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  5:  Route duration:  155.5  Covered demand: 201  Patient sequence: 42 (53.0 - 169.0) [46 - 522] -> 50 (345.0 - 451.0) [345 - 1305] -> 08 (648.0 - 735.0) [648 - 1807] ->
↪  24 (766.0 - 839.0) [766 - 1588] -> 69 (855.3 - 931.3) [628 - 1030] -> 88 (942.3 - 1042.3) [765 - 1769] -> 90 (1045.9 - 1133.9) [515 - 1491]
Nurse  6:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  7:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  8:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse  9:  Route duration:  138.2  Covered demand: 205  Patient sequence: 16 (271.0 - 345.0) [271 - 522] -> 59 (377.2 - 490.2) [365 - 601] -> 35 (498.3 - 581.3) [223 - 1015] ->
↪  55 (584.3 - 657.3) [279 - 663] -> 72 (675.8 - 774.8) [509 - 1342] -> 32 (785.9 - 874.9) [740 - 1841] -> 58 (877.8 - 973.8) [744 - 1035] -> 67 (976.6 - 1090.6) [725 - 1127]
Nurse 10:  Route duration:   68.2  Covered demand: 144  Patient sequence: 40 (7.3 - 105.3) [4 - 527] -> 30 (148.0 - 251.0) [148 - 546] -> 75 (432.0 - 498.0) [432 - 743] -> 63
↪  (644.0 - 707.0) [644 - 1621] -> 37 (796.0 - 876.0) [796 - 901]
Nurse 11:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse 12:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse 13:  Route duration:   89.1  Covered demand: 180  Patient sequence: 85 (372.0 - 464.0) [372 - 669] -> 39 (474.4 - 553.4) [272 - 1124] -> 54 (639.0 - 754.0) [639 - 1319]
↪  -> 77 (758.5 - 836.5) [695 - 1762] -> 65 (840.6 - 918.6) [735 - 1583] -> 52 (924.9 - 1043.9) [620 - 1512] -> 22 (1050.6 - 1155.6) [704 - 1251]
Nurse 14:  Route duration:  152.9  Covered demand: 192  Patient sequence: 51 (68.0 - 160.0) [68 - 379] -> 61 (203.0 - 278.0) [203 - 977] -> 78 (285.8 - 349.8) [181 - 352] -> 62
↪  (369.5 - 461.5) [182 - 834] -> 56 (465.7 - 572.7) [410 - 1477] -> 33 (583.8 - 675.8) [188 - 1015] -> 07 (680.3 - 773.3) [633 - 786] -> 66 (789.6 - 876.6) [464 - 1504]
Nurse 15:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
Nurse 16:  Route duration:  144.5  Covered demand: 202  Patient sequence: 29 (152.0 - 259.0) [152 - 324] -> 44 (262.0 - 355.0) [221 - 1379] -> 34 (370.0 - 435.0) [370 - 678] ->
↪  45 (447.0 - 542.0) [306 - 1259] -> 19 (614.0 - 729.0) [614 - 764] -> 79 (743.3 - 859.3) [567 - 904] -> 81 (876.8 - 981.8) [643 - 1166]
Nurse 17:  Route duration:   93.8  Covered demand: 158  Patient sequence: 60 (272.0 - 391.0) [272 - 573] -> 91 (393.0 - 468.0) [134 - 626] -> 28 (478.0 - 540.0) [478 - 712] ->
↪  09 (660.0 - 776.0) [660 - 1542] -> 01 (783.1 - 865.1) [353 - 1119] -> 31 (873.1 - 986.1) [762 - 1792]
Nurse 18:  Route duration:  154.9  Covered demand: 140  Patient sequence: 83 (53.3 - 159.3) [39 - 168] -> 49 (163.7 - 272.7) [39 - 353] -> 36 (285.8 - 356.8) [76 - 568] -> 82
↪  (676.0 - 772.0) [676 - 1412] -> 80 (784.0 - 860.0) [372 - 946] -> 87 (868.2 - 949.2) [791 - 1523] -> 41 (973.4 - 1065.4) [488 - 1471] -> 11 (1087.3 - 1166.3) [647 - 1637]
Nurse 19:  Route duration:  118.0  Covered demand: 202  Patient sequence: 23 (182.0 - 248.0) [182 - 538] -> 46 (258.0 - 359.0) [258 - 574] -> 84 (397.0 - 473.0) [71 - 572] ->
↪  57 (512.0 - 611.0) [512 - 638] -> 68 (618.3 - 694.3) [530 - 982] -> 02 (702.8 - 794.8) [521 - 1264] -> 93 (806.8 - 904.8) [648 - 1742] -> 26 (913.8 - 982.8) [662 - 1127]
Nurse 20:  Route duration:  132.7  Covered demand: 204  Patient sequence: 48 (61.0 - 140.0) [61 - 354] -> 03 (149.4 - 215.4) [60 - 1215] -> 15 (367.0 - 444.0) [367 - 720] -> 92
↪  (465.8 - 576.8) [418 - 1187] -> 43 (621.0 - 700.0) [621 - 909] -> 25 (708.1 - 797.1) [178 - 863] -> 38 (806.3 - 903.3) [42 - 1194] -> 18 (906.1 - 1024.1) [572 - 1733] -> 71
↪  (1063.2 - 1169.2) [757 - 1171]
Nurse 21:  Route duration:   71.5  Covered demand: 191  Patient sequence: 20 (274.0 - 351.0) [274 - 374] -> 70 (354.0 - 452.0) [285 - 992] -> 74 (465.2 - 560.2) [340 - 713] ->
↪  47 (567.2 - 660.2) [440 - 683] -> 13 (669.2 - 758.2) [153 - 1015] -> 76 (760.4 - 850.4) [460 - 912] -> 10 (872.0 - 980.0) [486 - 1406]
Nurse 22:  Route duration:  113.2  Covered demand: 180  Patient sequence: 89 (26.7 - 90.7) [25 - 146] -> 64 (132.0 - 211.0) [132 - 440] -> 17 (529.0 - 593.0) [529 - 1194] -> 27
↪  (598.4 - 699.4) [198 - 1298] -> 53 (776.0 - 891.0) [776 - 1326] -> 05 (892.4 - 979.4) [487 - 1161] -> 04 (992.5 - 1104.5) [682 - 1391]
----------------------

Total travel time: 1521.5688817806474
```

# B Test Instance 2 Output

```
Average travel time across 10 successful runs: 1626.58

=== Best Run ===
Generation 1: Best Fitness = 0.00000826331744597765, Entropy = 0.929239045642717, Lowest Feasible Travel Time = None
Generation 500: Best Fitness = 0.0002748298671873017, Entropy = 0.2079196417066913, Lowest Feasible Travel Time = 3637.6147191143914
Generation 1000: Best Fitness = 0.0004707816317728358, Entropy = 0.11304504418714005, Lowest Feasible Travel Time = 2123.1270527787406
Generation 1500: Best Fitness = 0.0005132943972479286, Entropy = 0, Lowest Feasible Travel Time = 1947.1997180596256
Generation 2000: Best Fitness = 0.0005132943972479286, Entropy = 0, Lowest Feasible Travel Time = 1947.1997180596256
Generation 2500: Best Fitness = 0.0005132943972479286, Entropy = 0, Lowest Feasible Travel Time = 1947.1997180596256
Generation 3000: Best Fitness = 0.0005132943972479286, Entropy = 0, Lowest Feasible Travel Time = 1947.1997180596256
  Refining candidate 1/5 (threaded)...
  Refining candidate 2/5 (threaded)...
  Refining candidate 3/5 (threaded)...
  Refining candidate 4/5 (threaded)...
  Refining candidate 5/5 (threaded)...
    Iteration 50, best travel time 1739.9969
    Iteration 50, best travel time 1694.6018
    Iteration 50, best travel time 1803.8258
    Iteration 50, best travel time 1719.3440
    Iteration 50, best travel time 1681.2703
    Iteration 100, best travel time 1694.6018
    Iteration 100, best travel time 1702.8485
    Iteration 100, best travel time 1713.7263
    Iteration 100, best travel time 1681.2703
    Iteration 100, best travel time 1711.7822
    Iteration 150, best travel time 1690.7968
    Iteration 150, best travel time 1713.7263
    Iteration 150, best travel time 1681.2703
    Iteration 150, best travel time 1697.2534
    Iteration 150, best travel time 1688.8815
    Iteration 200, best travel time 1713.7263
    Iteration 200, best travel time 1690.7968
    Iteration 200, best travel time 1693.3052
    Iteration 250, best travel time 1713.7263
    Iteration 200, best travel time 1630.3729
    Iteration 250, best travel time 1690.7968
    Iteration 200, best travel time 1688.8815
    Iteration 300, best travel time 1713.7263
    Iteration 300, best travel time 1690.7968
    Iteration 250, best travel time 1693.3052
    Iteration 350, best travel time 1690.7968
    Iteration 250, best travel time 1613.5643
    Iteration 250, best travel time 1684.9299
    Iteration 300, best travel time 1653.2367
    Iteration 300, best travel time 1603.7106
    Iteration 300, best travel time 1661.8961
    Iteration 350, best travel time 1640.0836
    Iteration 350, best travel time 1603.7106
    Iteration 350, best travel time 1661.8961
    Iteration 400, best travel time 1628.9197
    Iteration 400, best travel time 1601.5477
    Iteration 400, best travel time 1661.8961
    Iteration 450, best travel time 1609.4360
    Iteration 450, best travel time 1661.8961
    Iteration 450, best travel time 1601.5477
    Iteration 500, best travel time 1609.4360
    Iteration 500, best travel time 1661.8961
    Iteration 500, best travel time 1601.5477
    Iteration 550, best travel time 1601.5477
    Iteration 550, best travel time 1609.4360
    Iteration 600, best travel time 1601.5477
    Iteration 600, best travel time 1609.4360
    Iteration 650, best travel time 1609.4360
Run 9 best travel time = 1601.5477 (gap 0.54%)
=== Final Solution ===
Instance: test_instance_2
Nurse capacity: 204
Depot return time: 1316

----------------------
 Nurse  1:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  2:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  3:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  4:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  5:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  6:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  7:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  8:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  9:  Route duration:  164.6  Covered demand: 204  Patient sequence: 02 (51.9 - 130.9) [38 - 191] -> 68 (140.8 - 245.8) [128 - 1323] -> 10 (262.8 - 363.8) [2 - 673] -> 88
↪  (537.0 - 598.0) [537 - 1408] -> 17 (609.0 - 677.0) [569 - 1658] -> 04 (725.0 - 839.0) [725 - 1131] -> 42 (855.8 - 967.8) [675 - 1365] -> 03 (979.9 - 1064.9) [436 - 1247]
 Nurse 10:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 11:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 12:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 13:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 14:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 15:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 16:  Route duration:  174.8  Covered demand: 193  Patient sequence: 50 (183.0 - 260.0) [183 - 415] -> 22 (288.0 - 350.0) [288 - 479] -> 27 (424.0 - 536.0) [424 - 1508] ->
↪  11 (540.0 - 634.0) [26 - 737] -> 34 (663.0 - 734.0) [663 - 1322] -> 13 (757.0 - 823.0) [757 - 1753] -> 26 (823.0 - 918.0) [691 - 1398] -> 64 (948.8 - 1044.8) [678 - 1403]
 Nurse 17:  Route duration:   76.2  Covered demand: 152  Patient sequence: 62 (332.0 - 396.0) [332 - 822] -> 31 (462.0 - 552.0) [462 - 562] -> 87 (700.0 - 810.0) [700 - 961] ->
↪  61 (818.2 - 881.2) [737 - 1917] -> 83 (884.4 - 961.4) [743 - 1271]
 Nurse 18:  Route duration:  101.6  Covered demand: 183  Patient sequence: 56 (164.0 - 236.0) [164 - 324] -> 36 (246.8 - 307.8) [158 - 432] -> 08 (316.8 - 412.8) [249 - 1275] ->
↪  58 (416.4 - 498.4) [119 - 550] -> 32 (500.6 - 609.6) [225 - 1333] -> 66 (622.8 - 717.8) [506 - 765] -> 52 (730.4 - 793.4) [617 - 845]
 Nurse 19:  Route duration:  122.9  Covered demand: 198  Patient sequence: 85 (122.0 - 216.0) [122 - 644] -> 79 (258.0 - 358.0) [258 - 1234] -> 41 (555.0 - 672.0) [555 - 778] ->
↪  18 (759.0 - 849.0) [759 - 1014] -> 30 (853.1 - 941.1) [563 - 1704] -> 70 (946.1 - 1045.1) [573 - 1736] -> 86 (1050.1 - 1166.1) [782 - 1827] -> 49 (1179.6 - 1269.6) [646 -
↪  1319]
 Nurse 20:  Route duration:  159.4  Covered demand: 203  Patient sequence: 12 (50.9 - 147.9) [41 - 469] -> 90 (201.0 - 275.0) [201 - 1315] -> 53 (301.0 - 415.0) [301 - 597] ->
↪  07 (420.0 - 534.0) [144 - 650] -> 75 (542.6 - 647.6) [301 - 1427] -> 69 (657.5 - 766.5) [299 - 1033] -> 92 (772.9 - 833.9) [552 - 1000] -> 47 (839.9 - 951.9) [662 - 1226]
↪  -> 93 (980.8 - 1098.8) [627 - 1798]
 Nurse 21:  Route duration:  116.7  Covered demand: 121  Patient sequence: 43 (52.0 - 137.0) [52 - 317] -> 77 (579.0 - 696.0) [579 - 883] -> 63 (787.0 - 850.0) [787 - 1689] ->
↪  28 (873.1 - 950.1) [440 - 1476] -> 38 (970.7 - 1056.7) [382 - 1306]
 Nurse 22:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 23:  Route duration:  131.9  Covered demand: 187  Patient sequence: 15 (211.0 - 329.0) [211 - 508] -> 81 (344.0 - 407.0) [206 - 642] -> 21 (411.5 - 500.5) [281 - 743] ->
↪  37 (503.5 - 602.5) [76 - 727] -> 09 (657.0 - 754.0) [657 - 1719] -> 05 (768.1 - 853.1) [196 - 1294] -> 44 (857.4 - 960.4) [674 - 1010] -> 55 (968.0 - 1060.0) [50 - 1211] ->
↪  67 (1066.1 - 1155.1) [756 - 1557]
 Nurse 24:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
```

```
Nurse 25:  Route duration:   97.4  Covered demand: 197  Patient sequence: 23 (49.0 - 121.0) [49 - 470] -> 25 (136.3 - 234.3) [109 - 434] -> 59 (294.0 - 378.0) [294 - 540] -> 76
↪  (382.1 - 495.1) [367 - 699] -> 20 (514.3 - 631.3) [494 - 817] -> 40 (641.5 - 729.5) [182 - 1190] -> 65 (736.5 - 846.5) [412 - 884] -> 57 (855.1 - 931.1) [651 - 1159]
Nurse 26:  Route duration:  128.3  Covered demand: 186  Patient sequence: 72 (221.0 - 323.0) [221 - 1137] -> 74 (339.8 - 444.8) [227 - 1416] -> 29 (460.6 - 558.6) [255 - 669]
↪  -> 80 (565.6 - 637.6) [133 - 979] -> 91 (657.6 - 743.6) [583 - 772] -> 94 (759.7 - 870.7) [726 - 1089] -> 16 (892.9 - 1002.9) [599 - 1171]
Nurse 27:  Route duration:   71.5  Covered demand: 204  Patient sequence: 73 (54.0 - 166.0) [54 - 1226] -> 89 (175.8 - 259.8) [114 - 675] -> 24 (267.7 - 382.7) [191 - 635] ->
↪  78 (702.0 - 797.0) [702 - 1877] -> 39 (803.7 - 921.7) [729 - 930] -> 60 (926.2 - 995.2) [719 - 996] -> 14 (1006.2 - 1124.2) [241 - 1360]
Nurse 28:  Route duration:  129.9  Covered demand: 198  Patient sequence: 33 (19.0 - 139.0) [19 - 890] -> 46 (269.0 - 361.0) [269 - 571] -> 51 (373.8 - 487.8) [217 - 989] -> 82
↪  (505.0 - 572.0) [505 - 1020] -> 19 (579.2 - 658.2) [145 - 1078] -> 45 (668.2 - 771.2) [643 - 1618] -> 48 (802.1 - 875.1) [581 - 964]
Nurse 29:  Route duration:  126.3  Covered demand: 186  Patient sequence: 54 (422.0 - 541.0) [422 - 710] -> 06 (551.6 - 629.6) [304 - 679] -> 01 (650.8 - 732.8) [524 - 1236] ->
↪  84 (747.2 - 819.2) [32 - 1171] -> 71 (832.3 - 903.3) [319 - 1182] -> 35 (908.9 - 984.9) [266 - 1010]
----------------------

Total travel time: 1601.547748684944
```

# C Test Instance 3 Output

```
Average travel time across 10 successful runs: 2363.63

=== Best Run ===
Generation 1: Best Fitness = 0.00000192039629374583, Entropy = 0.9011114051607073, Lowest Feasible Travel Time = None
Generation 500: Best Fitness = 0.00002720875886614256, Entropy = 0.4760021779916595, Lowest Feasible Travel Time = None
Generation 1000: Best Fitness = 0.00019788461471077792, Entropy = 0.07925174366264187, Lowest Feasible Travel Time = 5052.449968617163
Generation 1500: Best Fitness = 0.00027827784489751616, Entropy = 0.030039156155128678, Lowest Feasible Travel Time = 3592.5307762940265
Generation 2000: Best Fitness = 0.00035012350623591385, Entropy = 0.0030408447078746443, Lowest Feasible Travel Time = 2855.1349986201676
Generation 2500: Best Fitness = 0.0004156984331106427, Entropy = 0, Lowest Feasible Travel Time = 2404.5900151392657
Generation 3000: Best Fitness = 0.0004156984331106427, Entropy = 0, Lowest Feasible Travel Time = 2404.5900151392657
Generation 3500: Best Fitness = 0.0004156984331106427, Entropy = 0, Lowest Feasible Travel Time = 2404.5900151392657
Generation 4000: Best Fitness = 0.0004156984331106427, Entropy = 0, Lowest Feasible Travel Time = 2404.5900151392657
Generation 4500: Best Fitness = 0.0004156984331106427, Entropy = 0, Lowest Feasible Travel Time = 2404.5900151392657
  Refining candidate 1/5 (threaded)...
  Refining candidate 2/5 (threaded)...
  Refining candidate 3/5 (threaded)...
  Refining candidate 4/5 (threaded)...
  Refining candidate 5/5 (threaded)...
    Iteration 50, best travel time 2404.5900
    Iteration 50, best travel time 2326.2155
    Iteration 50, best travel time 2371.5950
    Iteration 50, best travel time 2390.0696
    Iteration 50, best travel time 2359.6742
    Iteration 100, best travel time 2404.5900
    Iteration 100, best travel time 2326.1068
    Iteration 100, best travel time 2371.5950
    Iteration 150, best travel time 2404.5900
    Iteration 100, best travel time 2359.6742
    Iteration 200, best travel time 2404.5900
    Iteration 250, best travel time 2404.5900
    Iteration 150, best travel time 2371.5950
    Iteration 100, best travel time 2379.7551
    Iteration 150, best travel time 2326.1068
    Iteration 200, best travel time 2371.5950
    Iteration 250, best travel time 2371.5950
    Iteration 150, best travel time 2359.6742
    Iteration 200, best travel time 2326.1068
    Iteration 250, best travel time 2326.1068
    Iteration 300, best travel time 2326.1068
    Iteration 200, best travel time 2359.6742
    Iteration 250, best travel time 2359.6742
    Iteration 150, best travel time 2379.7551
    Iteration 200, best travel time 2379.7551
    Iteration 250, best travel time 2379.7551
    Iteration 300, best travel time 2379.7551
Run 2 best travel time = 2326.1068 (gap 1.06%)

=== Final Solution ===
Instance: test_instance_3
Nurse capacity: 209
Depot return time: 1475
----------------------
 Nurse  1:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  2:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  3:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  4:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  5:  Route duration:  140.0  Covered demand: 208  Patient sequence: 99 (193.0 - 270.0) [193 - 709] -> 92 (392.0 - 510.0) [392 - 526] -> 96 (609.0 - 702.0) [609 - 1001] ->
↪  33 (731.0 - 794.0) [731 - 1591] -> 49 (802.1 - 884.1) [274 - 1259] -> 76 (893.1 - 1003.1) [530 - 1119] -> 116 (1014.3 - 1091.3) [181 - 1360]
 Nurse  6:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse  7:  Route duration:  133.5  Covered demand: 198  Patient sequence: 123 (132.0 - 196.0) [132 - 707] -> 97 (293.0 - 360.0) [293 - 1368] -> 65 (381.0 - 475.0) [51 - 1196]
↪  -> 75 (479.1 - 577.1) [466 - 583] -> 54 (583.9 - 683.9) [471 - 911] -> 100 (687.0 - 775.0) [634 - 830] -> 18 (783.0 - 887.0) [257 - 1359] -> 104 (901.3 - 976.3) [632 -
↪  1599]
 Nurse  8:  Route duration:  109.9  Covered demand: 195  Patient sequence: 130 (111.0 - 210.0) [111 - 384] -> 32 (216.7 - 324.7) [93 - 1044] -> 88 (373.0 - 491.0) [373 - 496] ->
↪  122 (495.2 - 602.2) [273 - 1226] -> 48 (615.8 - 706.8) [296 - 713] -> 07 (746.0 - 808.0) [746 - 1636] -> 135 (827.7 - 892.7) [121 - 1175]
 Nurse  9:  Route duration:    0.0  Covered demand:   0  Patient sequence: None
 Nurse 10:  Route duration:  141.7  Covered demand: 209  Patient sequence: 34 (137.0 - 242.0) [137 - 492] -> 138 (282.5 - 366.5) [134 - 368] -> 29 (462.0 - 546.0) [462 - 1057]
↪  -> 109 (740.0 - 846.0) [740 - 861] -> 117 (857.3 - 939.3) [136 - 1161] -> 44 (956.4 - 1048.4) [701 - 1060] -> 47 (1054.8 - 1151.8) [332 - 1460] -> 133 (1158.1 - 1261.1)
↪  [206 - 1340]
 Nurse 11:  Route duration:  161.9  Covered demand: 191  Patient sequence: 01 (105.0 - 204.0) [105 - 359] -> 13 (424.0 - 487.0) [424 - 622] -> 119 (679.0 - 778.0) [679 - 1384]
↪  -> 108 (783.4 - 853.4) [293 - 912] -> 98 (874.6 - 951.6) [597 - 1441] -> 72 (952.6 - 1060.6) [529 - 1661]
 Nurse 12:  Route duration:  126.8  Covered demand: 199  Patient sequence: 114 (270.0 - 340.0) [270 - 421] -> 16 (344.1 - 420.1) [91 - 823] -> 09 (430.9 - 518.9) [163 - 627] ->
↪  132 (575.0 - 678.0) [575 - 800] -> 19 (679.0 - 785.0) [626 - 884] -> 59 (796.7 - 898.7) [514 - 1589] -> 143 (904.0 - 992.0) [640 - 1088] -> 74 (1002.0 - 1087.0) [577 -
↪  1375] -> 126 (1095.1 - 1190.1) [445 - 1529]
 Nurse 13:  Route duration:  142.8  Covered demand: 208  Patient sequence: 137 (35.8 - 135.8) [30 - 254] -> 46 (243.0 - 314.0) [243 - 869] -> 125 (322.1 - 441.1) [42 - 623] ->
↪  91 (484.0 - 567.0) [484 - 589] -> 73 (572.0 - 643.0) [183 - 842] -> 21 (645.8 - 706.8) [369 - 1228] -> 103 (709.1 - 799.1) [680 - 1781] -> 15 (814.1 - 878.1) [539 - 1298]
↪  -> 02 (910.0 - 1027.0) [607 - 1148] -> 69 (1031.5 - 1132.5) [356 - 1482]
 Nurse 14:  Route duration:  140.1  Covered demand: 199  Patient sequence: 110 (267.0 - 360.0) [267 - 869] -> 60 (373.4 - 445.4) [330 - 668] -> 37 (453.5 - 532.5) [251 - 1331]
↪  -> 28 (543.5 - 644.5) [36 - 1153] -> 141 (756.0 - 844.0) [756 - 969] -> 80 (850.0 - 959.0) [769 - 1736] -> 64 (971.1 - 1040.1) [534 - 1490] -> 81 (1054.1 - 1126.1) [478 -
↪  1450] -> 50 (1129.7 - 1230.7) [539 - 1489]
 Nurse 15:  Route duration:  136.6  Covered demand: 206  Patient sequence: 70 (124.0 - 184.0) [124 - 683] -> 127 (227.0 - 346.0) [227 - 586] -> 06 (459.0 - 564.0) [459 - 782] ->
↪  120 (615.0 - 685.0) [615 - 1607] -> 57 (686.4 - 781.4) [538 - 1199] -> 61 (782.4 - 863.4) [479 - 1468] -> 105 (869.2 - 977.2) [633 - 1578]
 Nurse 16:  Route duration:   23.9  Covered demand:  71  Patient sequence: 136 (737.0 - 857.0) [737 - 873] -> 23 (865.2 - 941.2) [718 - 1316]
 Nurse 17:  Route duration:   89.3  Covered demand: 182  Patient sequence: 42 (103.0 - 209.0) [103 - 445] -> 94 (216.0 - 331.0) [186 - 771] -> 115 (394.0 - 514.0) [394 - 1228]
↪  -> 140 (782.0 - 894.0) [782 - 943] -> 95 (900.3 - 989.3) [605 - 1375] -> 43 (1009.5 - 1112.5) [668 - 1641]
 Nurse 18:  Route duration:  138.3  Covered demand: 209  Patient sequence: 63 (25.0 - 87.0) [25 - 1152] -> 56 (104.5 - 174.5) [12 - 310] -> 142 (211.0 - 290.0) [211 - 1312] ->
↪  79 (294.2 - 402.2) [289 - 451] -> 62 (415.4 - 487.4) [183 - 633] -> 121 (497.6 - 599.6) [465 - 1456] -> 128 (635.0 - 717.0) [635 - 737] -> 84 (719.2 - 805.2) [668 - 1521]
↪  -> 112 (821.9 - 897.9) [735 - 1293]
 Nurse 19:  Route duration:   90.5  Covered demand: 195  Patient sequence: 87 (277.0 - 362.0) [277 - 779] -> 11 (370.5 - 473.5) [282 - 1259] -> 118 (488.4 - 595.4) [302 - 1086]
↪  -> 113 (596.8 - 663.8) [382 - 699] -> 68 (670.2 - 765.2) [494 - 1332] -> 139 (774.3 - 880.3) [725 - 1005] -> 124 (900.9 - 977.9) [660 - 1773]
 Nurse 20:  Route duration:  117.6  Covered demand: 207  Patient sequence: 111 (178.0 - 276.0) [178 - 631] -> 30 (285.2 - 358.2) [145 - 1332] -> 90 (364.6 - 430.6) [62 - 1046]
↪  -> 89 (442.3 - 520.3) [28 - 1182] -> 40 (526.6 - 635.6) [473 - 877] -> 04 (764.0 - 826.0) [764 - 1094] -> 24 (830.1 - 892.1) [615 - 1293] -> 03 (897.2 - 964.2) [76 - 1150]
↪  -> 10 (975.3 - 1094.3) [222 - 1122] -> 71 (1107.3 - 1180.3) [93 - 1269]
 Nurse 21:  Route duration:  176.0  Covered demand: 184  Patient sequence: 27 (126.0 - 216.0) [126 - 280] -> 22 (217.4 - 319.4) [116 - 416] -> 38 (351.7 - 428.7) [47 - 646] ->
↪  106 (736.0 - 849.0) [736 - 882] -> 101 (861.5 - 933.5) [732 - 1545] -> 55 (940.6 - 1041.6) [354 - 1340] -> 45 (1050.1 - 1131.1) [645 - 1518] -> 36 (1145.0 - 1259.0) [533 -
↪  1450]
 Nurse 22:  Route duration:  148.0  Covered demand: 200  Patient sequence: 17 (421.0 - 492.0) [421 - 832] -> 66 (503.2 - 614.2) [272 - 1155] -> 51 (638.0 - 754.0) [603 - 1538]
↪  -> 26 (795.0 - 910.0) [795 - 1047] -> 77 (919.0 - 1015.0) [584 - 1379] -> 86 (1035.6 - 1126.6) [735 - 1351] -> 83 (1140.8 - 1213.8) [367 - 1459] -> 58 (1238.5 - 1301.5)
↪  [770 - 1697] -> 52 (1316.8 - 1425.8) [708 - 1627]
 Nurse 23:  Route duration:  140.1  Covered demand: 208  Patient sequence: 85 (48.2 - 118.2) [8 - 232] -> 134 (130.7 - 242.7) [56 - 935] -> 107 (251.7 - 336.7) [216 - 1183] ->
↪  102 (670.0 - 773.0) [670 - 1184] -> 12 (783.8 - 895.8) [698 - 951] -> 131 (906.9 - 1018.9) [406 - 1153] -> 31 (1039.4 - 1143.4) [611 - 1451] -> 08 (1150.6 - 1249.6) [648 -
↪  1424] -> 82 (1256.6 - 1330.6) [471 - 1441]
 Nurse 24:  Route duration:  113.2  Covered demand: 209  Patient sequence: 14 (275.0 - 361.0) [275 - 527] -> 39 (388.2 - 475.2) [293 - 540] -> 35 (480.2 - 555.2) [247 - 1067] ->
↪  78 (557.2 - 636.2) [304 - 1111] -> 93 (637.6 - 710.6) [278 - 1040] -> 67 (725.9 - 804.9) [225 - 1051] -> 25 (816.1 - 903.1) [733 - 1427] -> 20 (920.1 - 995.1) [178 - 1117]
 Nurse 25:  Route duration:   56.0  Covered demand: 126  Patient sequence: 53 (53.0 - 146.0) [53 - 576] -> 129 (641.0 - 727.0) [641 - 782] -> 41 (735.6 - 826.6) [707 - 1189] ->
↪  05 (838.6 - 929.6) [532 - 1641]
----------------------

 Total travel time: 2326.1067759956645
```