Warehouse Shelf Allocation Optimization

Ting-Hung Tsai

1 Define the Fitness Function

The total walking distance for picking products is calculated by adding up the shortest distances needed to gather all the products of each order. To determine the shortest distance for each order, the function generates all possible permutations of shelf orders that a picker has to visit and selects the most efficient route.

For example, let's consider the second order in the 'OrderList' dataset which requires a picker to visit shelves 73, 62, and 57. In this case, the function assesses all six permutations and selects the optimal route.

In situations where a product is stored on two different shelves, the function evaluates all possible combinations to find the best route. For instance, if the first product of the order in previous example is stored not only on shelf 73 but also shelf 1, the function chooses the optimal routes from two sets of permutations: (73, 62, 57) and (1, 62, 57), namely 12 permutations. By doing so, it ensures the selection of the most efficient path for each order.

Regarding efficiency, the function takes 0.07 seconds to find the walking distance of the 'OrderList' dataset, which is 286428 meters.

2 A Greedy Approach

The heuristic is premised on the idea that by placing product groups with higher order frequencies closer to the entrance, we can reduce the walking distance for the majority of orders. This approach aims to enhance the efficiency of order picking operations within the warehouse.

The algorithm includes the following steps.

1. First, it begins by calculating the frequency of each product group within the set of orders. This frequency indicates how often each product group appears in orders.
2. Then, it sorts the product groups in descending order of their frequencies. This ensures that the most frequently ordered product groups are considered first for shelf allocation, aiming to place them in the most accessible locations.
3. Finally, it iterates through the sorted list of product groups for allocation of each product group.
   - Identify the closest available shelf to the entrance that has not yet been allocated. The closeness is determined by the distance from the shelf to the entrance.
   - Allocate this shelf to the current product group.
   - Mark the allocated shelf as no longer available for future allocations to ensure that each shelf is assigned to only one product group.

Using the same function from the first question, we calculated the walking distance to be 201402 meters, and the process took 0.30 seconds. We observed both shorter processing times and reduced distances.

3 Local Improvement

3.1 Neighbourhood

To improve the allocation by heuristic, we have designed a function that generates a new neighbourhood state. This function randomly selects two shelves from the current allocation and applies specific rules to modify them. If both shelves are empty, the function re-selects. If one shelf is empty, the product from the other shelf fills it. If the product on the one of selected shelves has only one shelf but the other has two, they either exchange the selected shelves with a 50% chance or the single-shelf product takes over the selected shelf of the double-shelf product's shelves. Finally, if none of the above conditions hold, two shelves simply switch their products.

3.2 Simulated Annealing

To minimize the total walking distance for a picker to pick up all orders, it is necessary to optimize product allocation. We have used a heuristic algorithm based on simulated annealing for this optimization.

The *simulated_annealing* function represents the computational process of simulated annealing. It uses the *get_neighbour* function to obtain a neighbor solution from the current solution, meaning it generates a new allocation scheme based on the current assignment of products to shelves. The *walking_distance* function is used to calculate the objective function value for the current solution, which is the total distance required to pick all products for all orders under the current allocation scheme.

Parameters are set in advance to determine the termination conditions of the simulated annealing algorithm. During the iterative process of updating solutions, better solutions are retained, and for worse solutions, the acceptance probability is adjusted according to the temperature. As the temperature is gradually reduced, the probability of accepting worse solutions decreases, and the final solution tends to converge, obtaining a better allocation scheme solution.

3.3 Result

We set the initial temperature of the simulated annealing process to 100. The cooling method adopts linear cooling, with a cooling rate of 0.999. The function terminates while the the maximal number of iterations has been reached or the temperature is lower than a threshold of 10^{-4}. We optimized both the current allocation scheme and the greedy allocation scheme obtained in question two, the results are as follows.

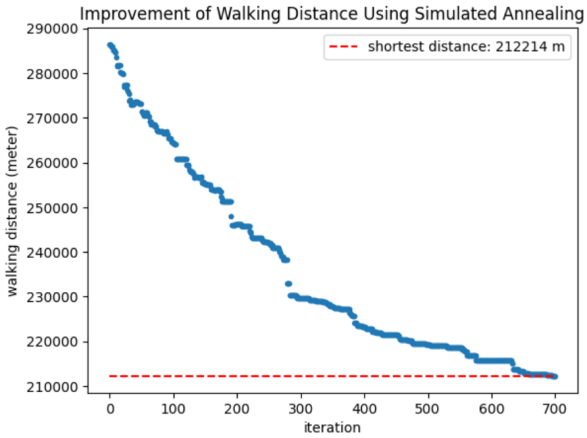

Figure 1: Current Allocation Simulated Annealing Process

From the Figure 1, after taking 55.25 second, the function terminated because it reaches an low enough temperature. The walking distance has reduced by 74214 meters and is 212214 meters. However, if we further modify the cooling rate, the walking distance would decrease to the similar result with a greedy approach.

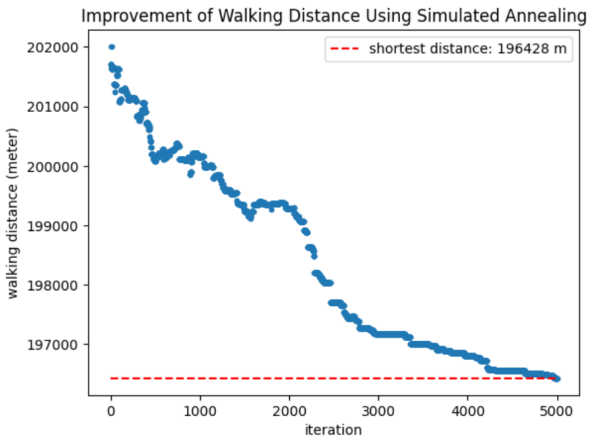

Figure 2: Greedy Allocation Simulated Annealing Process

From the Figure 2, we can see after 5000 iterations the optimization process terminated upon reaching the maximum number of iterations, taking a total of 406.40 seconds. The walking distance has reduced by 5220 meters and is 196428 meters.

In summary, for this problem, the simulated annealing algorithm can obtain an improved solution efficiently, and the selected parameters for the simulated annealing algorithm are reasonable.

4 Layout Modification

The current warehouse shelf layout presents inefficiencies on the utilization of space. Through a review of literature, we have identified several key strategies to optimize our warehouse layout.

- **Shortened Corridors:** By reducing the length of the vertical corridors and adding another one, we minimize the longest distance staff must travel to access products.
- **Surrounded Zone:** Additional columns of shelves are placed at the front and back of the zone, optimizing the use of horizontal space and improving accessibility for items stored in these areas.
- **Center Aisle:** A new corridor is introduced in the mid-way through the shelving units, enabling pickers to navigate in the front of the packing area swiftly, thus minimize the travel distance.

Our new floor plan is shown in Figure 3. It strategically locates shelves to maximize space utilization and reduce travel time. Products on shelves numbered 1 to 69 are accessible from the vertical corridor while shelves numbered 70 to 96 are accessed from the horizontal corridor.

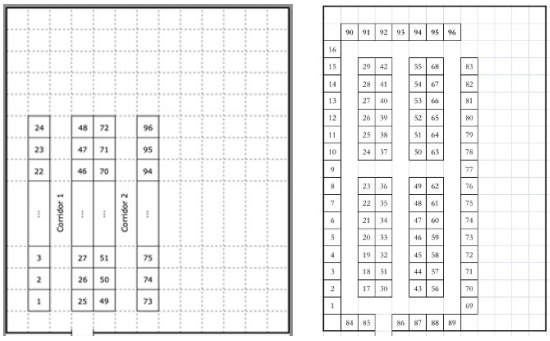

Figure 3: Left: Original Floor Plan; Right: New Floor Plan

The *bfs_maze_shortest_path* function calculates the shortest distance between two points in a maze-like structure using the Breadth-First Search algorithm. After computing distances for all pairs of shelves,

we obtain the new distance matrix.

Upon running the construction heuristic in Q2, we obtained a new shelf allocation plan with the total distance of 148356 meters.

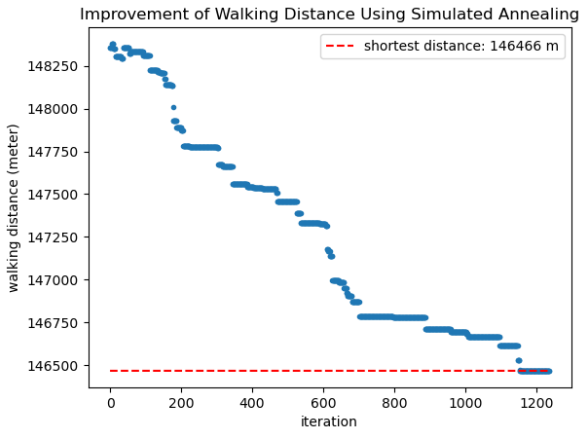

Figure 4: Greedy Allocation for New Floor Plan Simulated Annealing Process

Then we apply improvement heuristic on the greedy allocation. From the Figure 4, we can see after 1200 iterations the optimization process terminated upon reaching the maximum number of iterations, taking a total of 450.09 seconds. The walking distance has reduced by 1890 meters and is 146466 meters.