Distribution Network of Stores and Warehouses

Author: Rodolfo Croes

As a major supermarket, saving costs in any aspect of the business is a priority. One example of this being delivery services from a warehouse to stores on a map so that the supply of stores does not run out. This is an essential task that is done daily because of the demand needed. But how would we go about minimizing this cost? By minimizing the distance travelled by each vehicle we spend less on fuel. Given that we have trucks available that can only visit a maximum of 4 stores before returning to a warehouse, and lorry's who can visit a maximum of 16 stores before returning to a warehouse, but at double the cost of distance compared to a truck, which one would be better? Or should we consider a combination of the two? Which would result in the minimum distance? These questions are what I aim to answer by the end of this project.

We start by visually analyzing the map of our area where our stores and warehouses are located. From the map we can see that we have two warehouses. The first warehouse is located on the south-east side of the map and the second warehouse is located on the north-west side of the map. Given this we can say that the area for each warehouse can be split into two so that each warehouse can send vehicles out and let them deliver to their assigned stores. How we assign stores to each area varies but I have decided to test two approaches: One is split the area in half and whichever stores falls in each half of the map is assigned to a warehouse. This can lead to one warehouse having more stores to cover than the other warehouse, which leads to either more vehicles or one vehicle running for longer. The other assignment method is to ensure that each warehouse covers the same number of stores. However, seeing that this map has an uneven number of stores, one warehouse will have to handle one extra store. For that reason, we must test both methods to see which one produces the least distance in one continuous path.

To achieve this continuous path, the only viable option is to consider only using lorries for the deliveries because they can cover the most distance in one path. We will start by getting 2 paths based on if they are more east than north on their coordinates. This in turn splits the stores into lists that are considered south-east and north-west. The result of this gives us the list of stores that are assigned to each warehouse. These being [2, 6, 12, 14, 15, 16, 17, 18, 23] assigned to warehouse 1 and [1, 3, 4, 5, 7, 8, 9, 10, 11, 13, 19, 20, 21, 22] assigned to warehouse 2. We see that warehouse 2 is assigned more stores than warehouse 1 but both warehouses do not have 17 or more stores to deliver to thus the lorries can be used to traverse these paths.

For the second assignment method, we will assign each store to roughly the same number of stores. The way that this is achieved is by taking turns finding the store with the shortest distance from a warehouse and assigning them to the warehouse that they are closest to. Once a store is assigned it is removed from the selection pool of stores. This process continues until all stores are assigned to a warehouse. The

resulting store assignments are [21, 2, 23, 6, 17, 12, 14, 18, 16, 15, 20] assigned to warehouse 1 and [7, 4, 3, 9, 13, 22, 11, 5, 19, 10, 8, 1] are assigned to warehouse 2. From this assignment we see that the lorries would also work for the possible paths.

Both assignments give us paths, but we do not know for certain that these are the paths with the smallest distance from start to finish. This is where we apply algorithms to attempt to find the smallest distance from starting warehouse, visit each store and return to a warehouse. The algorithm I have chosen to implement is a hill climb inspired algorithm. This algorithm takes in the list of stores, along with their warehouse, the distance matrix of the map, and the number of steps to take for this hill climb and returns the best path found with its distance. The first step for the algorithm is to create an initial solution for the problem by taking the given stores and warehouse and shuffling the order of the stores. Then we append the warehouse to the start and depending on which store is last visited, it determines which warehouse is closer and goes there to end the path. We test the fitness of this path by summing the distances between stores and warehouses. These values are read from the distance matrix.

After initialization, we consider each step in the hill climb algorithm as a swap method where we change the order of the path by swapping two stores with each other. The stores are selected randomly from the initial path and then the last store is reassessed to determine where this new path will terminate. After this we calculate the total distance in the same manner as the initial solution. Finally, we compare the newly generated distance to the best distance and if it is less, the new path replaces the current best path. This algorithm runs many times to create many best paths so that the results avoid falling into a local maximum. We collect the results from each run and sort them and display the best 5 performing paths. If these are equal, then we can say that the best path has been found.

The process described above is applied for both warehouse zones with the first assignment strategy and one lorry starting at each warehouse. The best paths determined by the algorithm are ['W1', 15, 16, 14, 18, 12, 17, 2, 6, 23, 'W1'] and ['W2', 9, 8, 21, 20, 19, 5, 11, 13, 4, 3, 22, 10, 7, 1, 'W1'] respectively. The path starting at warehouse 1 travels 202 miles which costs £202 * 2 = £404. The path starting at warehouse 2 takes 290 miles to complete and costs the company £290 * 2 = £580. So, the total daily cost of this will be £404 + £580 = £984.

When applying the same process to the case of the warehouses being assigned roughly the same number of stores, and each warehouse having one lorry available, we end up with the best paths ['W1', 21, 20, 18, 12, 17, 2, 14, 16, 15, 6, 23, 'W1'] and ['W2', 9, 8, 19, 5, 11, 13, 4, 3, 22, 10, 7, 1, 'W1']. The route starting from warehouse 1 travels 277 miles and costs £277 * 2 = £554. The path that starts from warehouse 2 takes 227 miles to complete and costs £227 * 2 = £454. The total daily cost for taking this solution is £554 + £454 = £1008.

The second solution is more costly than the first solution so we can say that the first assignment strategy seems to be better than the second strategy. With this in mind,

we will keep the first assignments and use this to test the case of all trucks. Trucks can start at any warehouse, but they can only visit a maximum of 4 stores. Because of this, we must modify the hill climb algorithm to take this into account. We start this modification by determining how many trucks are needed if we start at one warehouse. This is equal to the number of stores divided by 4 and rounded up. Once we have this, we need to repeat the initialization and hill climb section of the algorithm for each truck. We will also need a global list to keep track of what stores have already been assigned to prevent trucks revisiting stores. The initialization of one truck path is done by randomly selecting 4 stores from the list of available stores to visit. If there are less than 4 stores left, then it takes the remaining stores instead. If it is only one store remaining, then the path of starting warehouse to one store to closest warehouse is saved to the results list and the algorithm is stopped from running. The distance of a path is determined the same way as the previous hill climb algorithm. The hill climb section of the algorithm now swaps one store from the path with a store from the remaining stores list. Then the fitness is assessed of the new path and is kept if the distance is less than the current best. The process is repeated for the number of steps given and at the end the best path and its distance is appended to a list of results. After this the stores that were saved in a best path are removed from the remaining stores list. The algorithm will finish once all trucks have a path with unique stores, and the best path and distance are returned.

The algorithm above is ran many times to collect many best paths for trucks traversing the zones that are assigned by area. From this we get the paths for 7 trucks where 3 of them start at warehouse 1 and 4 of them start at warehouse 2. The resulting best paths are ['W1', 15, 6, 2, 23, 'W1'], ['W1', 14, 18, 17, 12, 'W2'], ['W1', 16, 'W1'], ['W2', 21, 20, 19, 5, 'W1'], ['W2', 9, 4, 3, 22, 'W1'], ['W2', 1, 13, 11, 8, 'W2'], ['W2', 7, 10, 'W2']. The paths that start at warehouse 1 travel 251 miles and cost £251. The paths that start at warehouse 2 travel 376 miles and cost £376. So, the total cost for running all trucks is £251 + £376 = £627.

Using these results, we can also get the costs for combining one lorry starting at one warehouse and all trucks starting from the other warehouse. Consider the path of one lorry starting at warehouse 1 and 4 trucks starting at warehouse 2. The best solution for this would give us the paths ['W1', 15, 16, 14, 18, 12, 17, 2, 6, 23, 'W1'], ['W2', 21, 20, 19, 5, 'W1'], ['W2', 9, 4, 3, 22, 'W1'], ['W2', 1, 13, 11, 8, 'W2'], ['W2', 7, 10, 'W2']. The total cost for this solution is £404 + £376 = £780. On the other hand, the best path with 3 trucks starting at warehouse 1 and 1 lorry starting at warehouse 2 is ['W2', 9, 8, 21, 20, 19, 5, 11, 13, 4, 3, 22, 10, 7, 1, 'W1'], ['W1', 15, 6, 2, 23, 'W1'], ['W1', 14, 18, 17, 12, 'W2'], ['W1', 16, 'W1']. These paths have a total cost of £580 + £251 = £831.

In conclusion, from all the results produced, we can say that warehouse 1 should supply stores [2, 6, 12, 14, 15, 16, 17, 18, 23] and warehouse 2 should supply stores [1, 3, 4, 5, 7, 8, 9, 10, 11, 13, 19, 20, 21, 22]. The number of lorries and trucks each warehouse should start with to minimize total daily costs is warehouse 1 has 3 trucks and 0 lorries and warehouse 2 has 4 trucks and 0 lorries. The routes each vehicle

should take are ['W1', 15, 6, 2, 23, 'W1'], ['W1', 14, 18, 17, 12, 'W2'], ['W1', 16, 'W1'], ['W2', 21, 20, 19, 5, 'W1'], ['W2', 9, 4, 3, 22, 'W1'], ['W2', 1, 13, 11, 8, 'W2'], ['W2', 7, 10, 'W2']. The total cost for this daily operation is £627.

If I had more time available to continue exploring the search space, I would have liked to incorporate the genetic algorithm that we were taught in lab. I would have also liked to try combining truck paths so that one truck can cover the work of two trucks. One more thing is that this solution did not consider all possible ways of assigning stores to warehouses. I am content with the cost that I have found but this problem should be tested further with everything mentioned above.