Table of contents

1 INTRODUCTION ... 2

2 ALGORITHMS AND ASSUMPTIONS ... 2

- **2.1 Optimisation Model** ... 2
- **2.2 GA Heuristic** ... 2
- **2.3 FI Heuristic** ... 3
- **2.4 Underlying Assumptions** ... 3

3 CUSTOMER ALLOCATION WITH GA HEURISTIC ... 3

- **3.1 Data Preparation** ... 3
- **3.2 GA Heuristic in Python** ... 4
- **3.3 Result Analysis** ... 4

4 CUSTOMER REALLOCATION WITH OM AND FI HEURISTIC ... 4

- **4.1 OM in Python** ... 4
- **4.2 FI Heuristic in Python** ... 4
- **4.3 Result Analysis** ... 5
- **4.4 Sensitivity analysis** ... 5

5 SUMMARY ... 5

APPENDIX A: ALGORITHMS EXPLANATION ... 6

- **A.1 Flowchart of Greedy Adaptive Heuristic** ... 6
- **A.2 Flowchart of First Improvement Heuristic** ... 6
- **A.3 Construction Heuristics Comparison** ... 6
- **A.4 Improvement Heuristics Comparison** ... 6

APPENDIX B: RESULTS IN PYTHON ... 7

- **B.1 Cost of GA Heuristic solution with different shortlist size** ... 7
- **B.2 Improvement effect of OM and FI Heuristic on GA and GS Heuristic solution** ... 8
- **B.3 Improvement effect of FI heuristic with different random list size** ... 8
- **B.4 Improvement effect of OM and FI Heuristic without constraints 5 and 6** ... 9
- **B.5 Improvement effect of OM and FI heuristic with different capacity relaxation** ... 9

REFERENCES ... 10

1 Introduction

Prodnav Ltd., a new service provider in the UK, has 13 facilities in different regions and serves 380 customers national wide. In this report, we are to investigate the customer allocation problem, which aims to save travelling time and minimize total costs. The following parts are arranged as follows: first, three different algorithms and underlying assumptions are introduced; then, a customer allocation solution is proposed using Greedy Adaptive Heuristic; finally, OM and First Improvement Heuristic are applied to improve the proposed solution and several sensitivity analyses are conducted accordingly.

2 Algorithms and Assumptions

2.1 Optimisation Model

The objective function in the optimisation model is to minimise the total travelling cost, which can be formulated as (1), where h_i is the time of visits for customer i, d_{ij} is the distance from customer i to facility j. X_{ij} the binary decision variable, which indicates whether customer i is allocated to facility j.

$$Min \sum_{i \in I} \sum_{j \in J} h_i \cdot d_{ij} \cdot X_{ij} \tag{1}$$

The model is subjected to five constraints as shown from (2) to (6), where c_j is the maximum customer visits each facility can hold, a_{ij} is a binary variable indicating whether customer i is currently allocated to facility j, R_i is a binary variable indicating customer i is reallocated or not, and T is the reallocation cost per customer. Basically, (2) and (3) require that each customer can only be allocated to one facility and the total customer visits to one facility cannot exceed its capacity. (4) is included to guarantee the binarity of decision variables.

$$\sum_{j \in J} X_{ij} = 1, \qquad \forall i \in I \tag{2}$$

$$\sum_{i \in I} h_i \cdot X_{ij} \le c_j, \qquad \forall j \in j \tag{3}$$

$$X_{ij} \in \{0,1\}, \qquad \forall i \in I, \forall j \in j \tag{4}$$

(5) require that the reallocation should be beneficial not only to the company, but also to every single customer. If customer i is not reallocated, then $a_{ij} = X_{ij}$ and $R_i = 0$, the left and right part of (5) both equal to 0. If customer i is reallocated, then the left part of (5) represents the cost difference between current allocation a_{ij} and optimal allocation X_{ij}, and the right part equals to the reallocation cost T. Since h_i is the yearly visits demand, so (5) can be interpreted as for each customer, the cost reduction in one year should be no less than reallocation cost. If we set $T = 0$, then it means customer cannot be reallocated to a facility further than the current one.

$$\sum_{j \in J} h_i \cdot d_{ij} \cdot (a_{ij} - X_{ij}) \ge R_i \cdot T, \qquad \forall i \in I \tag{5}$$

(6) formulates the relationship between R_i, a_{ij} and X_{ij}. If customer i is not reallocated, then $\sum_j a_{ij} X_{ij} = (j-1) * 0 * 0 + 1 * 1 = 1$ and R_i equals to 0. If customer i is reallocated, then $\sum_j a_{ij} X_{ij} = (j-2) * 0 * 0 + 1 * 0 + 0 * 1 = 1$ and R_i equals to 1.

$$R_i = 1 - \sum_j a_{ij} X_{ij} \; \forall i, \qquad \forall i \in I \tag{6}$$

2.2 GA Heuristic

Though OM always provides the optimal result satisfying all constraints, most managers lack the necessary skills to understand and run complex mathematical models, thus heuristics are preferrable to solve this business problem. According to Haket (2020), Greedy Adaptive Heuristic is a construction heuristic, which creates a shortlist of the customers with the highest demands, randomly select one customer from the shortlist and then assign it to the closest available facility (**Appendix A.1**), the length of the shortlist is also a random number (Haket, C. 2020). Comparison between GA Heuristic and other construction heuristics is summarised in **Appendix A.3**, which shows that GA Heuristic performs quite well within reasonable running time. We are motivated to choose GA Heuristic because Greedy Sequential and Savings Regret easily stick to one local optimum while Greedy Random is time-consuming.

Different from Haket's method, we advanced the ranking criteria to ensure every decision made in each iteration is the best for objective function. Instead of ranking by demand, customers are ranked by the multiple of demand and distance, which would be further explained in Part 3. Furthermore, the length of shortlist is no longer a random number, but a pre-determined parameter.

2.3 FI Heuristic

Apart from construction heuristics, improvement heuristics are also widely applied in solving optimisation problems, no single best heuristics has been identified. As shown in **Appendix A.2**, First Improvement Heuristic randomize the customer list at the beginning of each iteration to avoid being mired in one local optimum. After that, it goes through the list and reallocates customer to the first facility capable of realizing an improvement. The process keeps going until no reallocation happens in the last iteration. **Appendix A.4** shows that compared to Best Improvement and One-opt improvement, FI Heuristic has the best performance and the second shortest running time.

In Part 3, we advanced FI Heuristic by adding two parameters to control the running time: (1) random list size, each iteration will only go through part of the customer list; (2) maximum iteration times, the algorithm will stop anyway after this number of iterations happens.

2.4 Underlying Assumptions

All algorithms and models are abstractions of real-world problems, and therefore it is imperative to disclose the underlying assumptions made in their formulation. Necessary assumptions are clarified as below:

(1) There are a fixed number of facilities, merging or closures of facilities is not considered
(2) The location of clients and facilities are known and stable
(3) Customer can only be served by one facility, though in reality multiple servers are possible
(4) Mechanics visit only one customer per day, so routing is not taken into consideration.
(5) Additional capacity over the current capacity is feasible for all facilities.
(6) Reallocation cost is tangible and independent of the size of the customer.
(7) Fluctuating travelling time is averaged to make data input deterministic.

3 Customer Allocation with GA Heuristic

3.1 Data Preparation

To solve the customer allocation problem with GA Heuristic in Python, we need the data of capacities (already given), demands and distance (**Appendix B.1**). As per the assignment brief, the local authorities (LAs) represented the customers, whose demands were calculated as 0.1% of the population given by ONS website. However, an issue came up for 14 LAs who did not have individual population estimates as the population of these LAs were summed together. Thus, their population were researched from the LA websites and cross-checked with the sum in the given datasets.

The geolocation data of all customers and facilities was downloaded from ONS website, and VLOOKUP was used to match it up with data of demands and capacities. For those facilities not matchable, we filled the blanks by googling their latitudes and longitudes as geolocation data does not change. All data resources are included in references.
Finally, distances between customers and facilities were simply calculated as the square root of the sum of the squared differences between the latitudes and longitudes, approximating the earth as a plane. We assume that the allocation cost is proportional to distance, thus minimizing the total allocation cost equals to minimizing the total distance.

3.2 GA Heuristic in Python

At the beginning of the algorithm, we create an empty array to store the allocation decision after each iteration. For each customer i that not allocated, we find the closest available facility j and calculate the possible cost increase of i by multiplying Distance [i, j] and Demand [i]. Next step is to rank the customer from the lowest to highest by possible cost increase and select the top (or lowest) ones to create the shortlist. We randomly choose a customer i from the shortlist and allocate it to the closest available facility j. Finally, the customer is removed for the list of remaining customers and the current capacity of j is updated. The algorithm is "adaptive" because the shortlist changes in each iteration based on information updated from the last iteration.

3.3 Result Analysis

To validate the result of the adaptive greedy adding algorithm, we can compare it to Reality Simulation, a non-optimal solution, and Greedy Sequential (GS) heuristic. With the seed equals to 1, the total cost in Reality Simulation is 87307.93 and in GS heuristic is 56051.49, while in GA Heuristic is 52279.68, the lowest cost among the three algorithms. It is because GA Heuristic is looking for the best step in each iteration to minimize our objective function, while involves some randomisation to avoid sticking in one local optimum.
As shown in **Appendix B.2**, GA Heuristic performance varies little by the shortlist size. Considering both the running time and performance, we decide to choose 12 as the optimal shortlist size in GA Heuristic and use it for further analysis, though it's the situation only when seed equals to 1.

4 Customer Reallocation with OM and FI Heuristic

4.1 OM in Python

Pulp package in Python was used to solve OM introduced in Part 2. First, a list of binary decision variables is created for all pairs [customer i, facility j], where 1 indicates allocating i to j. In addition, a list of binary reallocation variables was declared for all customers, where 1 indicates i is reallocated. Then, objective function and constraints were formulated, setting the array of GA Heuristic solution as a_{ij} and reallocation cost equals to zero. We decide to focus on the long-term benefits of reallocation, so reallocation cost can be split over many years and approximates 0.

4.2 FI Heuristic in Python

For every customer i in the random subset, if we find another facility j' closer to i than current facility j, we will check if j' has enough capacity for simple reallocation. If simple reallocation is not applicable, then we will try swapping i and i' (a customer currently allocated to j') as long as j is closer to i' than j', and both two facilities have enough capacity for swapping. During the process, we use variables (c, m, t) to track reallocation times, the number of customers examined for reallocation, and iteration times.

4.3 Result Analysis

The result in **Appendix B.3** shows that neither OM nor FI Heuristic can improve GA Heuristic solution, all three achieving the same cost at 52279.68. We also applied OM and FI Heuristic to improve GS Heuristic solution as a robust check, which turned out to be not improvable either. Both GA and GS Heuristic provide the local optimum solution, where every customer is allocated to the closest available facility in each iteration. In this case, reallocation is impossible if the cost of any one customer cannot be increased.

4.4 Sensitivity analysis

We use Reality Simulation as the initial solution to test how the two parameters in FI Heuristic would affect the improvements. **Appendix B.4** shows that when random list size is smaller than 7, the iteration stops easily and improvement is limited by iteration times. As the list size expands, FI Heuristic solution improved gradually but little after list size reached 50, so we choose 45 as the best list size. We set the maximum iteration times as 50, but it's not a tight constraint for any situation we tested in **Appendix B.4**.

Next, we removed constraints 5 and 6, which could possibly capture the benefit from trade-offs among customers and allow for optimisation across the entire system. It is allowed to increase cost for one customer and reduce for another, as long as the total cost is minimised. In this case, both OM and FI Heuristic can improve the GA Heuristic solution, reducing total cost to 48652.39 and 51897.95 separately (**Appendix B.5**).

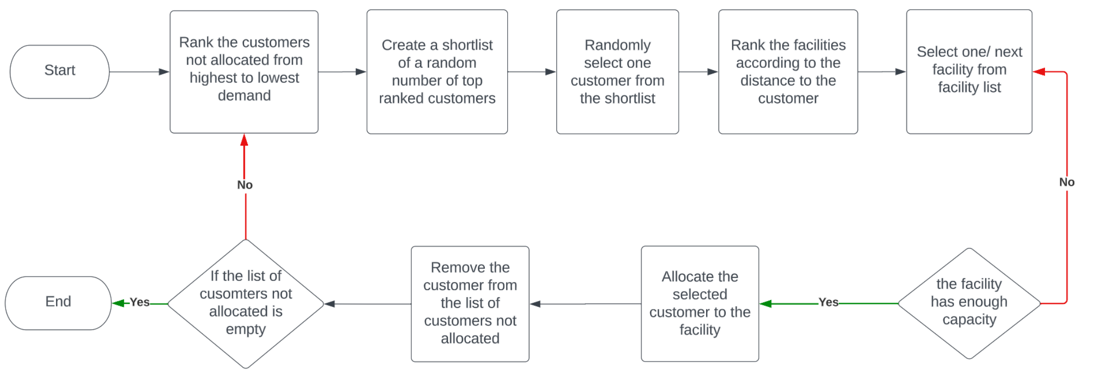
Finally, we test the effect of capacity relaxation on improving GA solution. As shown in **Appendix B.6**, the more capacity being relaxed, the more cost can be reduced by OM and FI Heuristic, and OM provides better results than FI Heuristic under all relaxation levels. 10% capacity relaxation has the equal effect as removing constraint 5 and 6. With unlimited capacity, OM and FI Heuristic can reduce total cost to 38572.31 and 43665.93 respectively.

5 Summary

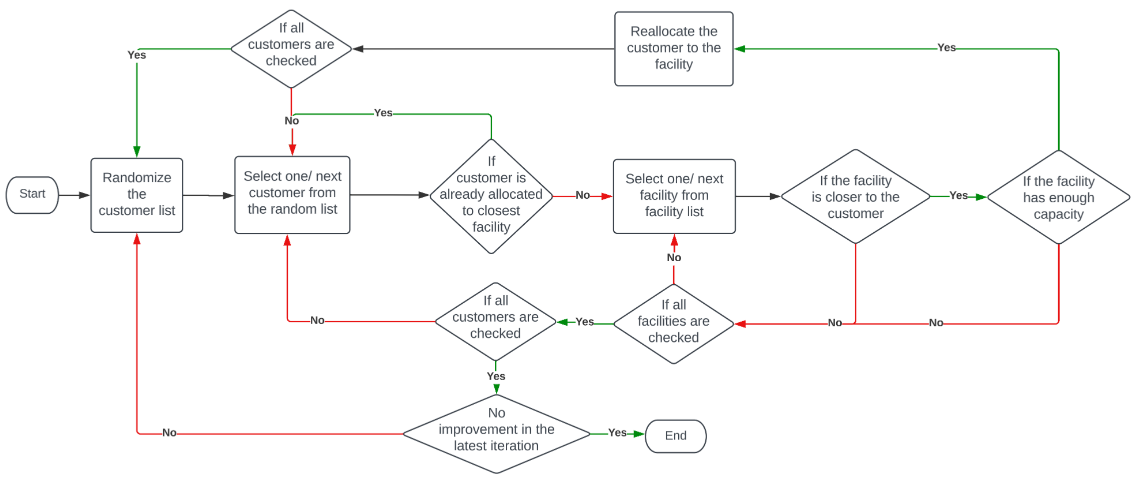
In conclusion, GA Heuristic provides a local optimum that can be improved by OM or FI Heuristic only with relaxation of constraints or capacity, where the algorithm displays higher sensitivity to capacity limit. Future research could be conducted in the following areas: 1) comparing the average performance of the algorithm with various seeds to determine the optimal parameter; and 2) exploring the combined effect of different sensitivity scenarios, such as simultaneous relaxation of capacity and constraint removal.

Appendix A: Algorithms explanation

A.1 Flowchart of Greedy Adaptive Heuristic



A.2 Flowchart of First Improvement Heuristic



A.3 Construction Heuristics Comparison

	Greedy Sequential	Greedy Random	Savings Regret	Greedy Adaptive
Runtime	< 90 secs	~320 secs	< 90 secs	~170 secs
Performance	4	1	3	2
Stuck in a local optimum	YES	NO	YES	NO

Note: Performance ranks the cost of each solution from the lowest to the highest

A.4 Improvement Heuristics Comparison

	First Improvement	Best Improvement	One-opt Improvement
Runtime	~15 mins	~7 mins	Slowest
Performance	1	2	3
Stuck in a local optimum	NO	YES	NO

Note: Performance ranks the cost of each solution from the lowest to the highest

Appendix B: Results in Python

B.1 Data input in Python

	Facility	Capacity
1	MOT	4000
2	NAY	3000
	…	…
	…	…
	…	…
13	SOU	13000

	Facility	latitude	longitude
1	MOT	55.7925	-3.99894
2	NAY	54.61842	-1.5719
	…	…	…
	…	…	…
	…	…	…
13	SOU	50.9105	1.4049

	Customer	Demand
1	E06000001	93.242
2	E06000002	140.545
	…	…
	…	…
	…	…
380	W06000024	60.183

	Customer	latitude	longitude
1	E06000001	-1.27023	54.6762
2	E06000002	-1.21099	54.5447
	…	…	…
	…	…	…
	…	…	…
380	W06000024	-3.36425	51.7486

B.2 Cost of GA Heuristic solution with different shortlist size

Shortlist size	GA Heuristic	GS Heuristic	Reality Simulation
1	52614.3	56051.49	87307.93
2	52674.76	56051.49	87307.93
3	52489.19	56051.49	87307.93
4	52675.03	56051.49	87307.93
5	52330.12	56051.49	87307.93
6	52674.76	56051.49	87307.93
7	52614.3	56051.49	87307.93
8	52347.48	56051.49	87307.93
9	52520.74	56051.49	87307.93
10	52516.89	56051.49	87307.93
11	52651.56	56051.49	87307.93
12	52279.68	56051.49	87307.93
13	52764.9	56051.49	87307.93
14	52632.33	56051.49	87307.93
15	52617.9	56051.49	87307.93
20	52767.7	56051.49	87307.93
30	52548.95	56051.49	87307.93
40	52621.46	56051.49	87307.93
50	52644.65	56051.49	87307.93

Note: GA Heuristic column is the cost of Greedy Adaptive Heuristic solution, GS Heuristic column is the cost of Greedy Sequential Heuristic solution, Reality Simulation column is the cost of Reality Simulation solution. In Reality Simulation, we rank facilities for each customer by distance and randomly allocate them to one of the top 2 closest available facilities, which is not a local optimum but closer to situation in reality.

B.3 Improvement effect of OM and FI Heuristic on GA and GS Heuristic solution

	GA as initial solution		GS as initial solution	
	Cost	Improvement	Cost	Improvement
GA / GS	52279.68		56051.49	
OM	52279.68	0.00%	56051.49	0.00%
FI heuristic	52279.68	0.00%	56051.49	0.00%

Note: Improvement column is the percentage of cost reduction compared to the cost of initial solution, GA / GS Heuristic solution.

B.4 Improvement effect of FI heuristic with different random list size

List size	Initial Solution	OM	FI heuristic	Reallocation	Iteration
1	87307.93	52038.0	87307.93	0	1
2	87307.93	52038.0	85324.11	14	8
3	87307.93	52038.0	81228.12	47	19
4	87307.93	52038.0	80163.2	54	19
5	87307.93	52038.0	81228.12	47	12
6	87307.93	52038.0	81228.12	47	10
7	87307.93	52038.0	70192.41	161	34
8	87307.93	52038.0	65459.91	200	46
9	87307.93	52038.0	65459.91	200	41
10	87307.93	52038.0	64623.13	224	45
11	87307.93	52038.0	66517.45	193	31
12	87307.93	52038.0	64623.13	224	37
13	87307.93	52038.0	64892.49	219	32
14	87307.93	52038.0	64623.13	224	32
15	87307.93	52038.0	64623.13	224	30
16	87307.93	52038.0	65149.52	223	28
17	87307.93	52038.0	63922.3	219	30
18	87307.93	52038.0	61025.09	261	31
19	87307.93	52038.0	65029.37	227	25
20	87307.93	52038.0	61025.09	261	34
25	87307.93	52038.0	61025.09	261	21
30	87307.93	52038.0	60016.76	275	33
35	87307.93	52038.0	60678.56	265	21
40	87307.93	52038.0	59933.23	272	25
45	87307.93	52038.0	59933.23	272	22
50	87307.93	52038.0	59808.92	274	20
100	87307.93	52038.0	59726.96	276	12
150	87307.93	52038.0	59722.61	275	9
200	87307.93	52038.0	59840.24	279	9

250	87307.93	52038.0	59771.59	284	5
300	87307.93	52038.0	59719.7	281	5
380	87307.93	52038.0	59840.92	277	3

Note: Initial Solution column is the cost of Reality Simulation, OM column is the cost of Optimisation Model, FI heuristic column is the cost of First Improvement heuristic, Reallocation column is the total reallocation times, Iteration column is the total iteration times.

B.5 Improvement effect of OM and FI Heuristic without constraints 5 and 6

	With constraints 5 and 6		Without constraints 5 and 6	
	Cost	Improvement	Cost	Improvement
GA heuristic	52279.68		52279.68	
OM	52279.68	0.00%	48652.39	6.94%
FI heuristic	52279.68	0.00%	51879.95	0.76%

Note: Improvement column is the percentage of cost reduction compared to the cost of GA heuristic. For FI heuristic, the random list size is set as 45 and the maximum iteration time is set as 50.

B.6 Improvement effect of OM and FI heuristic with different capacity relaxation

Capacity relaxation	Initial Solution	OM	FI heuristic	Reallocation	Iteration
0	52279.68	52279.68	52279.68	0	1
10%	52279.68	49497.63	51500.13	7	3
20%	52279.68	47160.59	49879.49	14	3
30%	52279.68	45358.23	48436.06	25	7
40%	52279.68	44142.15	47834.16	28	6
50%	52279.68	43165.86	47345.22	37	7
60%	52279.68	42264.28	46604.03	43	9
70%	52279.68	41633.26	46384.42	48	9
80%	52279.68	41052.19	46046.49	50	9
90%	52279.68	40615.72	45879.82	55	9
100%	52279.68	40298.33	45385.82	60	9
Unlimited	52279.68	38572.31	43775.93	64	9

Note: Initial Solution column is the cost of GA heuristic, OM column is the cost of Optimisation Model, FI heuristic column is the cost of First Improvement heuristic, Reallocation column is the total reallocation times, Iteration column is the total iteration times. For FI heuristic, the random list size is set as 45 and the maximum iteration time is set as 50.

References

Haket, C., Van Der Rhee, B., & De Swart, J. J. B. (2020). Saving Time and Money and Reducing Carbon Dioxide Emissions by Efficiently Allocating Customers. *INFORMS Journal on Applied Analytics*, *50*(3), 153–165. https://doi.org/10.1287/inte.2020.1028

Home - Office for National Statistics. (n.d.-b). Office for National Statistics. https://www.ons.gov.uk/

Open Geography Portal. (n.d.). https://geoportal.statistics.gov.uk/

Roskams, M. (2022, November 2). *Population and household estimates, England and Wales - Office for National Statistics*. https://www.ons.gov.uk/peoplepopulationandcommunity/populationandmigration/populationestimates/bulletins/populationandhouseholdestimatesenglandandwales/census2021unroundeddata

Search for an area profile - Dorset Council. (n.d.). https://mapping.dorsetcouncil.gov.uk/statistics-and-insights/AreaProfiles/Search