Table of Contents

Part A: Concert Ticket Model Optimization Model

1. Model with Equal Price to all Groups

We are informed that we should assume that the decision variables in this problem are continuous. This means that we can sell fractional tickets. The initial model has the following sets, parameters, decision variables, objective function and constraints:

Sets:

$I: Set\ of\ Segments$

Parameters:

Cap: Total amount of tickets available given venue capacity

D_i: Maximum demand for segment i (intercept in demand function)

S_i: Slope of demand function for segment i

Decision variables:

Q_i: Demand from segment i

P: Price (no subscript since it is equal for both segments)

Objective function:

$$Max\ Revenue\ = \sum_{i \in I} Q_i P$$

Constraints:

Total number of of tickets sold cannot be greater than the available quantity:

(1) $\sum_{i \in I} Q_i \leq Cap$

Each segment must make up at least 20% of the venue capacity:

(2) $Q_i \geq 0.2Cap \quad \forall i \in I$

Tickets sold to segment i cannot be higher than the demand from the segment at the set price:

(3) $Q_i \leq D_i + S_i P \quad \forall i \in I$

The model is solved in AMPL using the solver minos. The mod, run and dat-files can be found in the attached folder A, named "Proj3A-1.{mod, run, dat}". The results are stored in the same folder in the txt-file named "results3A-1". The optimal revenue for this model is 39,600,000 NOK[1] with a price of 720 NOK for both groups. 44,000 tickets are sold to the general public while 11,000 is sold to students. Plugging the price into the demand functions yields the following results:

Students: $20{,}000 - 1{,}250(7.2) = 11{,}000$

General: $120{,}000 - 3{,}000(7.2) = 98{,}400$

[1] We have rounded the revenue to the nearest integer for all models

The quantity demanded for the general public at a price of 720 NOK is significantly higher than the capacity of the stadium, which means that the organizers could have increased the price for this group. Because of the requirement of equal prices between segments, they cannot do this. They therefore choose the maximum possible price for the student segment, where the demand from students are exactly 11,000.

2. Model with Price Discrimination

This model requires some adjustments of the previous model. The changes are shown below:

1. A subscript i is added to the price variable, because the price can be different in the two segments:
   P_i: Price for segment i
2. Objective function now includes the new price variable:
   $$Max\ Revenue\ = \sum_{i \in I} Q_i P_i$$
3. Constraint (3) now includes the new price variable:
   (3) $Q_i \leq D_i + S_i P_i \quad \forall i \in I$

The model is solved in AMPL using the solver minos. The mod, run and dat-files can be found in the attached folder A, named "Proj3A-2.{mod, run, dat}". The results are stored in the same folder in the txt-file named "results3A-2". The optimal revenue for this model is 119,387,000 NOK with a price of 2,533.33 NOK for the general public and 720 NOK for students. 44,000 tickets are sold to the general public while 11,000 tickets are sold to students. The increase in revenue from the previous model is due to the price increase for the general public. At this price level there is a demand of exactly 44,000 tickets in this segment (calculated in the same manner as in A1).

3. Models with Senior Segment

The following models have included an additional Senior segment.

3-a. Model with Partial Price Discrimination Between Segments

The changes from the model in A2 are:

1. Seniors are added to the set I.
2. Constraint (2) is divided into two constraints:
   (2) $Q_{General} \geq 0.2Cap$
   (3) $Q_{\text{Students}} + Q_{Seniors} \geq 0.2Cap$
3. A new constraint is added to ensure that the price for Students and Seniors are equal:
   (5) $P_{Students} = P_{Seniors}$

The model is solved in AMPL using the solver minos. The mod, run and dat-files can be found in the attached folder A, named "Proj3A-3-a.{mod, run, dat}". The results are stored in the same folder in the txt-file named "results3A-3a". The optimal revenue for this model is 121,429,000 NOK with a price of 2,533.33 NOK for the general public, 905.66 NOK for students and 905.66 NOK for seniors. 44,000 tickets are sold to the general public, 2,320.75 tickets are sold to students and 8,679.25 tickets are sold to seniors.

3-b. Model with Full Price Discrimination Between Segments

The changes from the model in A3-a are:

1. Constraints (2) and (3) are modified to ensure that at least 10% of the venue capacity is sold to Students, at least 10% is sold to Seniors, and at least 20% is sold to the general public:

   (2) $Q_i \geq 0.1Cap \quad \forall i \in I: i \neq General$

   (3) $Q_{General} \geq 0.2Cap$
2. Constraint (5) is removed, allowing for full price discrimination between the segments.

The model is solved in AMPL using the solver minos. The mod, run and dat-files can be found in the attached folder A, named "Proj3A-3-b.{mod, run, dat}". The results are stored in the same folder in the txt-file named "results3A-3b". The optimal revenue for this model is 121,579,000 NOK with a price of 2,533.33 NOK for the general public, 678.57 NOK for students and 1,160 NOK for seniors. 44,000 tickets are sold to the general public, 5,500 tickets are sold to students and 5,500 tickets are sold to seniors.

3-c. Model with Full Price Discrimination Between Segments and Price Constraint

The changes from the model in A3-b are:

1. 2. A new parameter P^{MIN}, equal to 6, is added to indicate the minimum allowed price for a ticket.
2. Constraint (2) and (3) are replaced by a new constraint, that changes the minimum allocation for each segment:

   (2) $Q_i \geq 0.05Cap \quad \forall i \in I$
3. A new constraint is added to ensure that the price is at least equal to the minimum allowed price, P^{MIN}, for all segments:

   (4) $P_i \geq P^{MIN} \quad \forall i \in I$
4. A new constraint is added to ensure that the price for one segment cannot be more than twice the price for another segment:

   (5) $P_i \leq 2P_m \quad \forall\, i \in I, m \in I: i \neq m$

The model is solved in AMPL using the solver minos. The mod, run and dat-files can be found in the attached folder A, named "Proj3A-3-c.{mod, run, dat}". The results are stored in the same folder in the txt-file named "results3A-3c". The optimal revenue for this model is 92,826,200 NOK with a price of 1,750 NOK for the general public, 1,380 NOK for students and 875 NOK for seniors. 49,500 tickets are sold to the general public, 2,750 tickets are sold to students and 2,750 tickets are sold to seniors.

A comparison of all the pricing models are shown in Table 1.

Model Comparison

	Price ('00NOK)			Tickets					
Model	General	Students	Seniors	General	Students	Seniors	Total	Revenue ('00NOK)	Rank
1	7.20	7.20	-	44,000	11,000.00	-	55,000	396,000	5
2	25.33	7.20	-	44,000	11,000.00	-	55,000	1,193,870	3
3a	25.33	9.06	9.06	44,000	2,320.75	8679.25	55,000	1,214,290	2
3b	25.33	11.60	6.79	44,000	5,500.00	5500	55,000	1,215,790	1
3c	17.50	13.80	8.75	49,500	2,750.00	2750	55,000	928,262	4

Table 1: Comparison of Models in Part A

As the organizers' goal is to maximize revenue, we would suggest alternative 3b as this provides the highest revenue.

Part B: Menu Engineering Model Modifications

1. Model with new Requirement for Nutrient k_1

The decision variable $y_{p1,i,j,t}$ indicates whether Romeo and Juliet (p_1) picks item i in meal j on day t. The parameter $\omega_{k1,i}$ is the amount of nutrient k_1 in item i. By multiplying these for all values of i and j, we find the units of nutrient k_1 that Romeo and Juliet consumes in a day t. If we sum this for a period of three consecutive days (t, t+1 and t+2) we find the units of nutrient k_1 consumed in the three day period. This sum must be at least 250 for all three day periods. Mathematically, this can be written as:

$$\sum_{i\in I}\sum_{j\in M}\sum_{t}^{t+2} y_{p1,i,j,t}\omega_{k1,i} \geq 250 \quad \forall t \in T: t < |T| - 2$$

2. Model with Happy Saturday Requirement

$$HS_{j,t} = \begin{cases} 1\ if\ Happy\ Saturday\ occurs\ on\ day\ t\ for\ meal\ j \\ 0\ otherwise \end{cases}$$

$y_{p1,"PB",j,t}$ indicates whether Romeo and Juliet eats Pasta Bolognese in meal j on day t. $y_{p1,"SIC",j,t}$ and $y_{p1,"VIC",j,t}$ indicates whether strawberry ice cream or vanilla ice cream are served them in meal j on day t. If the sum of these are equal to 2 on any Saturday for any meal, a Happy Saturday occurs. We create a new subset $T_{Saturdays} \subset T$ with all Saturdays in the planning period. The following constraints are then added:

If two of the items Pasta Bolognese, vanilla ice cream or strawberry ice cream are served in meal j at day t, $HS_{j,t}$ must be 1:[2]

(1) $y_{p1,"PB",j,t} + y_{p1,"SIC",j,t} + y_{p1,"VIC",j,t} \leq 1 + HS_{j,t} \quad \forall j \in M, t \in T_{Saturdays}$

Constraint (2) and (3) ensures that $HS_{j,t}$ is zero if a meal does not include Pasta Bolognese and one of the ice cream varieties:

(2) $HS_{j,t} \leq y_{p1,"PB",j,t} \quad \forall j \in M, t \in T_{Saturdays}$

(3) $HS_{j,t} \leq y_{p1,"SIC",j,t} + y_{p1,"VIC",j,t} \quad \forall j \in M, t \in T_{Saturdays}$

[2] It is assumed that only one of the ice creams can be served at one meal, in other words $ns_{j,\text{"Dessert"}} \leq 1$ for all meals. This assumption is consistent with the authors sample application model parameters in Table 3 (Kulturel-Konak et al., 2022, p. 11).

A Happy Saturday can at most occur one time in one Saturday:

(4) $\sum_{j \in M} HS_{j,t} \leq 1 \quad \forall\, t \in T_{Saturdays}$

Ensures that there are at least two Happy Saturdays in the planning horizon (T):[3]

(5) $\sum_{j \in M} \sum_{t \in T} HS_{j,t} \geq 2$

[3] Note that there is a constraint stating that a main dish can be served a maximum of two times in the planning period (Kulturel-Konak et al., 2022, p. 18). Therefore constraint (5) implies that a Happy Saturday must occur exactly two times in the planning period.

Part C: Nord Pool Day-Ahead Market

1. Comparison Between Linearized and Step Function Curves for Day-Ahead Market

We start by plotting the step function curves and the linearized curves for supply and demand in the fourth period. These are shown in Figure 1:

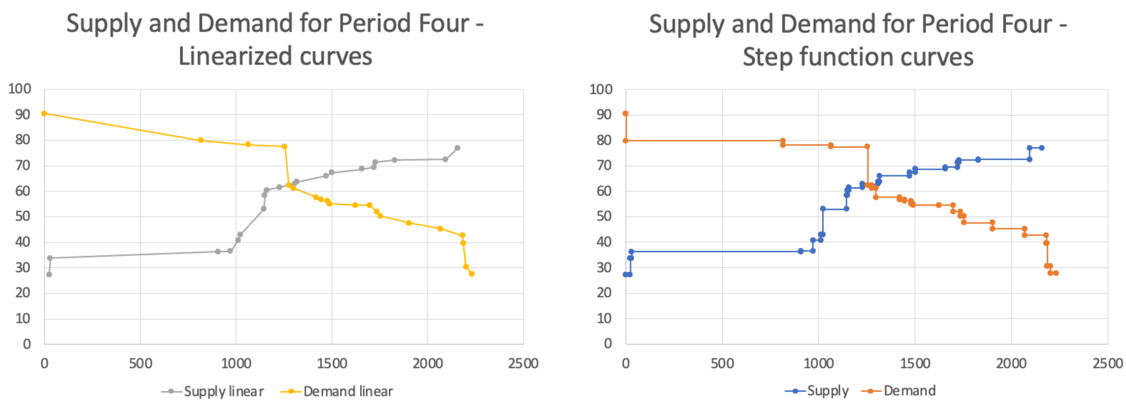

Figure 1: Supply and Demand for Period Four – Step Function and Linearized Curves

Based on the linearized curves, we estimate that the price will be €61 per mWh with a total volume of around 1250 mWh. There is however a quite large drop in willingness to pay, from around €78 to around €61, at this volume level. By studying the stepwise function closely we observe that there is no intersection between buyer and seller at this price point. By using the stepwise function to propose the system price, we know that it will be somewhere between €61 and €78 dollars. Buyer bids at around €78 should therefore be accepted and seller bids above €61 should be accepted.

Step function curves provide an accurate representation of the price and quantity at which buyers and sellers are willing to buy and sell. This is useful for analyzing buyer and seller bids, which we used to determine the best bidding strategy in task C2. However, determining an equilibrium price can be difficult because the curves are unlikely to intersect.

Linearized curves can be used to find the intersection of the supply and demand curves. The equilibrium price is found by finding the intersection of the linearized curve. The disadvantages of linearized curves are that we always overestimate the willingness to pay on the demand side and always underestimate the price suppliers are willing to sell for. However, the system's rules are clear, and the players are aware that this occurs. They will therefore account for this when bidding.

2. Bidding Strategy to Maximize Profit

2-a. Bidding Strategy with same Volume every Period

Our strategy and the resulting profit can be seen in Table 2.

Starting Point - All Available Capacity to the Highest Possible Price

Period	Quantity Bid (MW)	Quantity Accepted (MW)	Ask Price (€)	Equilibrium Price (€)	cost (€)	Revenue (€)	Profit (€)
1	1,200	1,200	31	32.05	11	38,460	25,260
2	1,200	1,200	46	47.28	11	56,736	43,536
3	1,200	1,200	54	54.55	11	65,460	52,260
4	1,200	1,200	40	40.24	11	48,288	35,088
5	1,200	1,200	49	52.92	11	63,504	50,304
Total						272,448	**206,448**

Table 2: Initial Strategy

The strategy consists of using the highest possible ask price that allows us to sell all available capacity for each period as a starting point. In contrast to this, one could argue that it is more profitable to place a bid with a lower quantity in order to increase the linearized equilibrium price. The increased equilibrium price could then increase profits more than the reduction in quantity sold reduces it. While this might be the case for a given period, we believe that it is unlikely to be simultaneously true for all periods for the same quantity reduction.

To further analyze this assumption, we reduce quantity by a given amount and employ a method of tuning the ask price. Tuning the ask price, in the case where the whole bid is accepted, consists of increasing the ask price as much as possible in order to push up the linearized equilibrium price. Eventually the marginal increase in the ask price will result in a reduction of quantity accepted. At this point we will consider whether the effect on profit from the quantity reduction is greater than that of the potential increase in the equilibrium price. If this is the case the price increase is stopped, otherwise we continue until it is.

In the case where the entire bid is not accepted, tuning consists of decreasing the ask price until we achieve an acceptance of the whole bid. This is only done granted that the effect on profits from a potential decrease in the equilibrium price is not greater than the effect of the increased quantity. This way we can progressively search for a better price for a given quantity bid. An example can be seen in Table 3, where we decrease quantity with 100 and perform the ask price tuning. Tuned prices and the total profit are marked in bold.

Alternative Starting Point - Example of Reduced Quantity with Ask Price Tuning							
Period	Quantity Bid (MW)	Quantity Accepted (MW)	Ask Price (€)	Equilibrium Price (€)	cost (€)	Revenue (€)	Profit (€)
1	1,100	1,100	31	33.49	11	36,839	24,739
2	1,100	1,100	**48**	48.36	11	53,196	41,096
3	1,100	1,100	54	54.63	11	60,093	47,993
4	1,100	1,100	**36**	43.84	11	48,224	36,124
5	1,100	1,100	**43**	55.48	11	61,028	48,928
Total						259,380	**198,880**

Table 3: Strategy Iteration with Reduced Quantity and Ask Price Tuning

We try a few other quantity reductions with ask price tuning. Our findings remain the same: While some quantity reductions with price tuning yield a higher profit for a certain period, the total profit over all periods remain lower than that of the original starting point.

Finally, it might be of interest to use ask price tuning also on our original starting point. The idea is to see whether this will increase the linearized equilibrium price enough to more than make up for the likely decrease in quantity for any of the periods. From Table 4, we see that this clearly leads to a less profitable result. The best bids we can come up with is therefore our original starting point which is shown in Table 2.

Original Starting Point with Marginal Ask Price Increase								
Period	Quantity Bid (MW)	Quantity Accepted (MW)	Ask Price (€)	Equilibrium Price (€)	cost (€)	Revenue (€)	Profit (€)	Original Profit (€)
1	1,200	1,089	32	32.17	11	35,033	23,054	25,260
2	1,200	1,117	47	47.56	11	53,125	40,838	43,536
3	1,200	657	55	55.00	11	36,135	28,908	52,260
4	1,200	0	41	40.99	11	0	0	35,088
5	1,200	1,195	50	52.94	11	63,263	50,118	50,304

Table 4: Initial Strategy with Marginal Price Increase

2-b. Bidding Strategy with Flexible Volume every Period

In C2-a we argued that reducing quantity by the same amount across all periods would not be favorable compared to the starting point. Now that the quantity of the bids can vary across periods, we can establish a search pattern to see which (if any) periods would benefit from a quantity reduction. More precisely the search pattern will progressively help us determining how much the reduction in quantity should be for the different periods. We will also employ ask price tuning in this process. As quantity is reduced the increase in the linearized equilibrium price needed to achieve the same original profit increases. Large price jumps early in the iteration are therefore more likely to prove profitable. We therefore stop the iteration at a quantity reduction of 400.

Table 5 captures the iterative process of this strategy.

Price Jump Search Through Quantity Reduction with Ask Price Tuning

Iteration	Period	Quantity Bid (MW)	Quantity Accepted (MW)	Ask Price (€)	Equilibrium Price (€)	cost (€)	Revenue (€)	Profit (€)
Original	1	1,200	1,200	31	32.05	11	38,460	25,260
it1.1	1	1,100	1,100	31	33.40	11	36,740	24,640
it1.2	1	1,000	1,000	31	39.73	11	39,730	28,730
it1.3	1	900	900	31	41.22	11	37,098	27,198
it1.4	1	800	800	31	42.49	11	33,992	25,192
Original	2	1,200	1,200	46	47.28	11	56,736	43,536
it2.1	2	1,100	1,100	46	48.15	11	52,965	40,865
it2.2	2	1,000	1,000	46	48.74	11	48,740	37,740
it2.3	2	900	900	46	49.16	11	44,244	34,344
it.2.4	2	800	800	46	50.80	11	40,640	31,840
Original	3	1,200	1,200	54	54.55	11	65,460	52,260
it3.1	3	1,100	1,100	54	54.63	11	60,093	47,993
it3.3	3	1,000	1,000	54	54.72	11	54,720	43,720
it3.3	3	900	900	54	54.80	11	49,320	39,420
it3.4	3	800	800	54	54.89	11	43,912	35,112
Original	4	1,200	1,200	40	40.24	11	48,288	35,088
it4.1	4	1,100	1,097	40	43.84	11	48,092	36,025
it4.1.1	4	1,100	1,100	36	43.84	11	48,224	36,124
it4.2	4	1,000	932	40	45.51	11	42,415	32,163
it4.3	4	900	900	40	46.71	11	42,039	32,139
it4.4	4	800	781	40	47.98	11	37,472	28,881
Original	5	1,200	1,200	49	52.92	11	63,504	50,304
it5.1	5	1,100	1,079	49	55.48	11	59,863	47,994
it5.2	5	1,150	1,150	49	54.79	11	63,009	50,359
it5.2.1	5	1,150	1,150	50	54.82	11	63,043	50,393
it5.3	5	1,000	1,000	49	56.71	11	56,710	45,710
it5.4	5	900	900	49	59.02	11	53,118	43,218
it5.5	5	800	800	49	59.76	11	47,808	39,008

Table 5: Price Jump Search Through Quantity Reduction with Ask Price Tuning

For period 1 we see that once we reduce the quantity by 200, there is a significant change in the linearized equilibrium price. As we can see the price jump is sufficiently great to make up for the reduction in quantity. Price tuning in this case where the whole bid is accepted involves increasing the ask price. This results in a reduction of accepted quantity without an accompanying effect on the equilibrium price. We therefore keep the ask price unchanged and decrease quantity by 200 for period 1.

For period two and three we find that decreasing the quantity is not beneficial. We therefore keep the original bids for these periods. For period 4 we find that a decrease of 100 with price tuning is better than the original starting point and we therefore place this bid for period four. Note that a relatively large price jump can be spotted early and still not prove to be profitable. In this case it might be of interest to see if we can increase quantity while still benefitting from the relatively large price jump. This is the case in period 5. We therefore increase the quantity by 50 to see if we can find a better solution. From Table 5 we see that we do in fact find a better solution. This solution is further improved by tuning the ask price.

We conclude that while it can be beneficial to decrease quantity for some periods, it is not necessarily true for all periods for the same reduction. More precisely, the quantity reduction needed to make one period more profitable might not match that of another period. This is evident in Table 5. However, when we can vary the quantities of the bids across time periods, we can select different favorable quantity reductions. A summary of the outcome can be found in Table 6.

Best Found Bids with Varying Quantity and Price Across Periods

Period	Quantity Bid (MW)	Quantity Accepted (MW)	Ask Price (€)	Equilibrium Price (€)	cost (€)	Revenue (€)	Profit (€)	Original Profit (€)
1	1,000	1,000	31	39.73	11	39,730	28,730	25,260
2	1,200	1,200	46	47.28	11	56,736	43,536	43,536
3	1,200	1,200	54	54.55	11	65,460	52,260	52,260
4	1,100	1,100	36	43.84	11	48,224	36,124	35,088
5	1,150	1,150	50	54.82	11	63,043	50,393	50,304
Total							**211,043**	206,448

Table 6: Best Found Bids with Varying Quantity and Price Across Periods

Because we have relaxed a restriction, we would expect that the profit is greater or equal to that of the original starting point. Unsurprisingly this is the case, and the profit has increased by €4,595 to €211,043.

The iterations could of course have been done with quantity reductions of a lesser magnitude (e.g. 10) and continued beyond a total reduction of 400. This would likely yield a better solution. However, as the objective in this task is simply to demonstrate how the method works, we find it expedient to use a larger quantity reduction of 100 and stop the iteration at a total quantity reduction of 400.

3. Bidding Strategy to Maximize Profit with Blockbids

In this task we have to choose between the four consecutive time periods 1-4 or 2-5. As period 1 has a significantly lower price and volume than periods 2-5, we choose periods 2-5 as the four consecutive time periods in our blockbid. We then test what our profit will be if we produce at maximum capacity, 1200 MW. This is done by setting the volume to 1200 and tune the price until we find the maximum price we can charge and get out blockbid accepted. As the average price in periods 2-5 is €53.73 with blockbids, we bid 53.7. This bid gets accepted, and leads to a total profit of €135,828. There are then four blockbids accepted, including ours, as shown in Table 7

Supply Block Bids Accepted when Bidding 1200 MW at €53.7

Ask Price (€)	Volume (MW)	Id	Starting Period	Ending Period	Acceptance Order
42.0	141	3	3	5	1
48.6	18	4	2	4	3
44.0	20	7	3	5	2
53.7	1,200	11	2	5	4

Table 7: Accepted Supply Block Bids when Bidding 1200 MW at €53.7

The average price is now €37.66 periods 2-4 and €42.29 in periods 3-5. The reason why the blockbids with id's 4 and 7 are accepted is because they are placed before our blockbid. We therefore explore what effect undercutting these players have. When reducing our price to €48.5, the blockbid with id 4 is no longer accepted and the prices increase. This leads to an increase in profit to €136,992. When undercutting the blockbid with id 7, by bidding €43.9, our profit increases to €137,508. We are not able to move the average price in periods 3-5 to below €42, thus the blockbid with id 3 is accepted regardless of what price we bid. It is also worth mentioning that we can reduce our bid to €1, without affecting the price nor the profit.

To test whether we can increase our profits by reducing the volume we sell, we reduce the volume by 10 MW and calculate the profit for each iteration. We use the price €43.9, as reducing the price more than this does not affect the equilibrium price or, as mentioned above. We find that the highest profit is achieved with a volume of 1160 MW per period, which gives a profit of €138,608. As we see the trend going downwards from this volume level, we do not iterate in levels of 10 MW below 1100 MW. Our strategy is therefore to place a blockbid in the periods 2-5 with a price of €43.9 per mWh and a volume of 1160 MW per period. The iterations we performed to arrive at this strategy are shown in Table A1 in the Appendix.

Part D: Bike Sharing Optimization Model

We start by defining some key terms that are used in the model. A *link* is defined as the distance between inhabitant n and location i. *Eligible* refers to whether a link is at most *t* km and a bike rack has been installed the location related to that link.

The model have the following sets, parameters, decision variables, objective function and constraints:

Sets:

I: Set of locations to install bike racks

N: Set of inhabitants in the city

Parameters:

c: Cost of acquiring conventional bikes

e: Cost of acquiring electric bikes

v_i: Variable cost per bike initially allocated to rack i

f_i: Fixed cost incurred by installing rack i

$d_{i,n}$: Distance to location rack i for inhabitant n (also referred to as a *link*)

t: Maximum distance from a rack's location to an inhabitant that still qualifies them as a potential user

ε: Small value used to determine $y_{i,n}$ for instances where $d_{i,n} = t$

M_1: Big M used to determine x_i (e.g. the maximum number of bikes allowed on a bike rack)

M_2: Big M used to determine $u_{i,n}$ (e.g. the maximum distance from an inhabitant to a bike rack in the dataset)

Decision variables:

B_i^{CO}: Number of conventional bikes bought and installed at rack i

B_i^{EL}: Number of electric bikes bought and installed at rack i

$x_i : \begin{cases} 1 \text{ if a bike rack is installed at location i} \\ 0 \text{ otherwise} \end{cases}$

$y_{i,n} : \begin{cases} 1 \text{ if inhabitant n is located at most t km from location i} \\ 0 \text{ otherwise} \end{cases}$

$b_{i,n} : \begin{cases} 1 \text{ if the link between inhabitant n and location i is eligible} \\ 0 \text{ otherwise} \end{cases}$

$u_{i,n} : \begin{cases} 1 \text{ if the link is eligible and represents the shortest distance among the eligible links} \\ 0 \text{ otherwise} \end{cases}$ [4]

$k_{i,n} : \begin{cases} 1 \text{ if the link is eligible and represents the shortest distance among the eligible links} \\ 0 \text{ otherwise} \end{cases}$

[4] In the case where no links between inhabitant n and bike rack i are eligible, $u_{i,n}$ will take the value 1 for an arbitrary i. However, we introduce the variable $k_{i,n}$, which is equal to 0 in all these cases and 1 in the case where the link is eligible and represents the shortest distance among the eligible links.

Objective function:

$$\text{Min Total Budget} = \sum_{i \in I} B_i^{CO}(c + v_i) + B_i^{EL}(e + v_i) + x_i f_i$$

Constraints:

There must be exactly two electric bikes at every rack installed:

(1) $B_i^{EL} = 2x_i \qquad \forall i \in I$

Ensures that x_i takes the value 1 if a bike rack is installed at location i and 0 otherwise:

(2) $B_i^{EL} + B_i^{CO} \leq M_1 x_i \qquad \forall i \in I$

Constraints (3) – (4) ensures that $y_{i,n}$ takes the value 1 if $d_{i,n} \leq t$ and 0 otherwise:

(3) $d_{i,n} - (t + \varepsilon) \geq \left(d_{i,n} - (t + \varepsilon)\right) y_{i,n} \qquad \forall i \in I, n \in N$

(4) $\left(d_{i,n} - (t + \varepsilon)\right) y_{i,n} \leq 0 \qquad \forall i \in I, n \in N$

Constraints (5) – (7) ensures that $b_{i,n}$ takes the value 1 if a link is eligible and 0 otherwise:

(5) $y_{i,n} + x_i \leq b_{i,n} + 1 \qquad \forall i \in I, n \in N$

(6) $b_{i,n} \leq x_i \qquad \forall i \in I, n \in N$

(7) $b_{i,n} \leq y_{i,n} \qquad \forall i \in I, n \in N$

Constraints (8) – (9) ensures that $u_{i,n}$ takes the value 1 if a link is eligible and represents the shortest distance among the eligible links:[5]

(8) $b_{i,n} d_{j,n} - b_{j,n} d_{i,n} \geq M_2\left(u_{i,n} - 1\right) \qquad \forall i \in I, n \in N, j \in I : i \neq j$

(9) $\sum_{i \in I} u_{i,n} = 1 \qquad \forall n \in N$

Constraints (10) – (12) ensures that $k_{i,n}$ takes the value 1 if a link is eligible and represents the shortest distance among the eligible links, and 0 otherwise:

(10) $b_{i,n} + u_{i,n} \leq k_{i,n} + 1 \qquad \forall i \in I, n \in N$

(11) $k_{i,n} \leq u_{i,n} \qquad \forall i \in I, n \in N$

(12) $k_{i,n} \leq b_{i,n} \qquad \forall i \in I, n \in N$

At least 50 percent of the inhabitants must become potential users:

(13) $\sum_{n \in N} \sum_{i \in I} k_{i,n} \geq 0.5|N|$

Number of bikes allocated to each installed rack must be at least 5% of the potential users linked to the rack:

(14) $B_i^{CO} + B_i^{EL} \geq 0.05 \sum_{n \in N} k_{i,n} \qquad \forall i \in I$

[5] To clarify the functionality of constraints (8)-(9), we have included an explicit example in Figure A1 in the Appendix.

Although not required, we have validated our model by implementing it in AMPL and running it on a sample dataset. The files can be found in the folder “D”.

Bibliography

Kulturel-Konak, S., Konak, A., Jakielaszek, L., & Gavirneni, N. (2022). Menu Engineering for Continuing Care Senior Living Facilities with Captive Dining Patrons. *INFORMS Journal on Applied Analytics*, 1-22.

Appendix

Overview of Different Strategies and the Resulting Profits

Volume (MW)	Ask Price (€)	Period 2 Price (€)	Period 3 Price (€)	Period 4 Price (€)	Period 5 Price (€)	Profit (€)	Strategy
1,200	53.7	30.33	46.31	36.34	44.21	135,828.0	Max Quantity and Ask Price
1,200	48.5	31.21	46.33	36.41	44.21	136,992.0	Max Q, undercut blockbid 4
1,200	43.9	31.21	46.35	36.48	44.55	137,508.0	Max Q, undercut blockbid 4 and 7
1,200	41.9	31.21	46.35	36.48	44.55	137,508.0	Max Q, undercut blockbid 4, 7 and 3
1,200	1.0	31.21	46.35	36.48	44.55	137,508.0	Max Q and Minimum Ask Price
1,190	43.9	31.74	46.35	36.52	44.79	137,326.0	Reduce Q by 10
1,180	43.9	32.27	46.36	36.56	45.03	137,139.6	Reduce Q by 20
1,170	43.9	33.23	46.37	36.60	45.33	137,510.1	Reduce Q by 30
1,160	43.9	34.63	46.38	36.71	45.77	138,608.4	Reduce Q by 40
1,150	43.9	34.84	46.39	36.82	46.21	138,299.0	Reduce Q by 50
1,140	43.9	35.02	46.40	36.94	46.65	137,951.4	Reduce Q by 60
1,130	43.9	35.20	46.41	37.06	46.88	137,351.5	Reduce Q by 70
1,120	43.9	35.38	46.41	37.18	47.00	136,606.4	Reduce Q by 80
1,110	43.9	35.56	46.42	37.29	47.12	135,852.9	Reduce Q by 90
1,100	43.9	35.74	46.43	37.41	47.25	135,113.0	Reduce Q by 100
1,000	43.9	37.41	46.50	38.35	48.23	126,490.0	Reduce Q by 200
900	43.9	38.61	46.58	39.51	50.12	117,738.0	Reduce Q by 300
800	43.9	39.88	46.67	40.43	51.42	107,520.0	Reduce Q by 400
700	43.9	41.39	46.75	40.85	52.62	96,327.0	Reduce Q by 500

Table A1: Overview of Different Strategies and the Resulting Profits in Part C

Index	d_i,"Jay"	b_i,"Jay"
1	6	1
2	5	1
3	7	1
4	2	0
5	25	0

i	j	$b_{i,"Jay"}$	$d_{j,"Jay"}$	$b_{j,"Jay"}$	$d_{i,"Jay"}$	$b_{i,"Jay"} * d_{j,"Jay"} - b_{j,"Jay"} * d_{i,"Jay"}$	$u_{i,"Jay"}$
1	2	1	5	1	6	-1	Must be 0
1	3	1	7	1	6	1	Can take both 1 and 0
1	4	1	2	0	6	2	Can take both 1 and 0
1	5	1	25	0	6	25	Can take both 1 and 0
Resulting u_1,"Jay"							**0**
2	1	1	6	1	5	1	Can take both 1 and 0
2	3	1	7	1	5	2	Can take both 1 and 0
2	4	1	2	0	5	2	Can take both 1 and 0
2	5	1	25	0	5	25	Can take both 1 and 0
Resulting u_2,"Jay"							**1**
3	1	1	6	1	7	-1	Must be 0
3	2	1	5	1	7	-2	Must be 0
3	4	1	2	0	7	2	Can take both 1 and 0
3	5	1	25	0	7	25	Can take both 1 and 0
Resulting u_3,"Jay"							**0**
4	1	0	6	1	2	-2	Must be 0
4	2	0	5	1	2	-2	Must be 0
4	3	0	7	1	2	-2	Must be 0
4	5	0	25	0	2	0	Can take both 1 and 0
Resulting u_4,"Jay"							**0**
5	1	0	6	1	25	-25	Must be 0
5	2	0	5	1	25	-25	Must be 0
5	3	0	7	1	25	-25	Must be 0
5	4	0	2	0	25	0	Can take both 1 and 0
Resulting u_5,"Jay"							**0**

$$\sum_{i \in I} u_{i,"Jay"} = 1$$

Figure A1: Explicit Example of Constraints (8)-(9) in Part D