Operations Research, Spring 2024 (112-2)
Final Project Proposal
Team J

B10705052 葉又銘　B10705014 許群佑　B10705034 許文鑫　B10705010 蔡可亮

Department of Information Management, National Taiwan University

May 28, 2024

1 Introduction

In this project, we aim to optimize the seating arrangement for customers in restaurants. A restaurant typically has various table sizes and serves two types of customers: customers without a reservation and customers who have reserved a table in advance for a specific time period. Customers without reservation arrive at the restaurant and request a table from the server. If tables are available, they are immediately seated. However, if all tables are occupied, they must wait until a table becomes available. We wish to minimize the total waiting time for customers without reservation.

Since customers with reservation have reserved a table in advance for a specific time, the restaurant must ensure that a table is available for them at their reserved time. If a table cannot be provided at the reserved time, it would lead to customer dissatisfaction and potential loss of business. Currently, the seating arrangement is managed manually by the servers in the restaurant based on their experience and judgment. However, we are interested in exploring an operations research model to optimize the seating arrangement more efficiently and effectively.

2 Problem description

The problem involves optimizing the seating arrangement in a restaurant over a day to minimize the total waiting time for all customers and ensure that reservation customers are seated at their reserved times. The restaurant operates during specific opening hours, and customers arrive and depart dynamically throughout the day. There are three main aspects to consider:

1. **Dining customers:** The restaurant should ensure that customers currently enjoying their meal are not interrupted or asked to relocate, as such disruptions could lead to dissatisfaction. At times, the restaurant might need to extend the expected meal duration for these customers, which can be efficiently managed through recomputing in the model.

2. **Customers without reservation:** Upon the arrival of new customers without reservations, the restaurant faces the decision of immediately seating them or providing an estimated waiting time when tables are unavailable. This process must be managed to optimize customer flow and satisfaction.

3. **Customers with reservation:** For customers holding reservations, the restaurant must verify the availability of tables for the requested time periods before finalizing the reservation. Once a reservation is confirmed, the restaurant is committed to ensuring a table is available for these customers at their specified time.

Additionally, the optimization will tackle situations where a single table's capacity is inadequate for a group by enabling table combinations, as permitted by the restaurant's ability to merge tables. The goal is to accommodate larger groups while minimizing the need for table combining.

The optimization problem requires the model to dynamically update the seating arrangement in response to new customers without reservation, new reservations, or variations in dining times faster or slower than expected. Each time these changes occur, the model reruns to ensure alignment with the real-time situation, aiming to minimize total customer waiting time while guaranteeing reservations.

The optimization model should also take into account the restaurant's layout, table sizes, and customer preferences (such as number of customers in the group or specific table requests). Additionally, the model should consider any time dependencies or constraints, such as the expected duration of meals or the possibility of combining or splitting tables to accommodate larger or smaller group.

By implementing an operations research model, the restaurant aims to improve customer satisfaction, maximize operational efficiency, and potentially increase revenue by reducing waiting time and better managing reservations.

3 Mathematical model

In this section, we formulate a mixed linear integer program to precisely describe our problem presented in Section 2.

We first define the indices. Let $T = \{1, \ldots, T^*\}$ be the set of time periods, $G = \{1, \ldots, G^*\}$ be the set of groups of customers, and $D = \{1, \ldots, D^*\}$ be the set of tables. In the sequel, we use t as the index of time period, g as the index of group of customers, and d as the index of tables.

Next, we define the parameters and decision variables. Here is the notation:

Parameters

N_g: Number of customer in group g.

M_d: Number of seats of table d.

C_{ij}: Binary variable, 1 if tables i and j can be combined for larger groups, 0 otherwise.

P_g: Meal duration for group g, measured in time periods.

U_g: Maximum waiting time allowed for group g before seating, measured in time periods.

S_g: Number of time periods group g has already waited.

R_g: Binary variable indicates if group g has a reservation.

H_g: Maximum number of tables group g is willing to be assigned to.

α: Reward weight to minimize the number of tables is willing to groups of customers.

Within this model, the parameters N_g, M_d, C_{ij}, U_g, R_g, and H_g result from negotiations between the restaurant and its customers, or are determined by the restaurant, in alignment with its seating policies and agreements. In contrast, P_g, S_g, and α are established in advance, based on pre-computation by the programs or predetermined by operational strategies.

Decision Variables:

a_{gtd}: Binary variable; 1 if group g is seated at table d during time period t, otherwise 0

b_{gd}: Binary variable; 1 if group g is seated at table d, otherwise 0

x_{gt}: Binary variable; 1 if group g is being seated during time period t, otherwise, 0.

The total waiting time is $\sum_g (1 - R_g) N_g \left(\sum_t t x_{gt} + S_g\right)$. If $(1 - R_g) = 0$, it means that group g has made a reservation, we do not need to consider their waiting time. For the rest, the waiting time of group g would be indicated by the decision variable tx_{gt}, where $x_{gt} = 1$ means that group g is assigned to a table at time period t, and by timing the time period gives us the waiting time for group g. Also, we have to add this number by the time periods group g has already waited, which is by S_g. Lastly, we would time the sum by the number of customers in the group, which is N_g, giving us the total waiting time of each person in group g.

Also, we would also want to reward the objective function if we assign each group to as few tables as possible, which is wrote as $\alpha \sum_g \left(H_g - \sum_d b_{gd}\right)$, where α is the reward weight. Additionally, H_g is the maximum number of table group g is willing to be assigned to, and $\sum_d b_{gd}$ is the total number of table group g is assigned to.

This gives us the objective function (1):

$$\min \sum_g (1 - R_g) N_g \left(\sum_t t x_{gt} + S_g\right) - \alpha \sum_g \left(H_g - \sum_d b_{gd}\right) \tag{1}$$

Lastly, we define the constraints.

First, we list constraints for the tables. Constraint (2) makes sure the number of seats should satisfy the number of customers in each group, meaning that the total number of the seats of the tables assigned to each group are no less than the number of customers in the group. In addition, constraint (3) ensures that if a group is assigned to two tables or more, the tables must be able to be combined. Next, constraint (4)ensures that the tables of each group in a period matches their seated tables. Constraint (6) shows the time interval of when group g is assigned to table d. Then, we would ask the maximum number of tables each group is willing to be assigned to, and constraint (7) disallows us to split them into tables more than that number. Moreover, we get the starting time of group g at table d with constraint (8).

$$N_g \leq \sum_d M_d b_{gd} \qquad \forall g \in G \tag{2}$$

$$b_{gi} + b_{gj} \leq C_{ij} + 1 \qquad \forall i \in D, j \in D \tag{3}$$

$$a_{gdt} \leq b_{gd} \qquad \forall g \in G, d \in D, t \in T \tag{4}$$

$$\sum_{t \in T} a_{gdt} \geq P_g b_{gd} \qquad \forall g \in G, d \in D \tag{5}$$

$$\sum_{t=t'}^{t+P_g-1} a_{gdt'} \geq P_g x_{gt} \qquad \forall g \in G, d \in D, t \in \{1, \cdots, T^* - P_g + 1\} \tag{6}$$

$$\sum_d b_{gd} \leq H_g \qquad \forall g \in G \tag{7}$$

$$2x_{gt} \leq a_{gdt} - a_{gd,t-1} + 1 \qquad \forall g \in G, d \in D, t \in T \tag{8}$$

$$\tag{9}$$

Secondly, we list constraints for arranging seats and meal duration. Constraint (10) implies the starting time of a group would not be later than $U_g - S_g$, which is unacceptable. Constraint (11) makes sure that a table at a certain time can only be assigned to a group. Next, Constraint (12) means that the starting time of a group can only and must appear once. Lastly, constraint (13) indicates that a_{gtd}, b_{gd}, x_{gt} are binary variables.

$$x_{gt} \leq 0 \qquad \forall t \in \{U_g - S_g + 1, \cdots, T^*\}, g \in G \tag{10}$$

$$\sum_g a_{gtd} \leq 1 \qquad \forall t \in T, d \in D \tag{11}$$

$$\sum_t x_{gt} = 1 \qquad \forall g \in G \tag{12}$$

$$a_{gtd}, b_{gd}, x_{gt} \in \{0, 1\} \qquad \forall g \in G, t \in T, d \in D \tag{13}$$

4 Expected results

To approach our problem, we will randomly generate data according to the estimated distribution of the real-world situation faced by the restaurant. Moreover, we will write a computer program to calculate the values of parameters required for our mathematical program from the raw data.

For small-scale instances, we plan to implement the mixed linear integer programming model by writing a Python program that invokes Gurobi Optimizer to obtain an optimal rearrangement and upgrade plan. For large-scale instances, finding an optimal solution may be too time-consuming and thus not practical; therefore, we seek to develop a heuristic algorithm that can generate near-optimal solutions efficiently. To demonstrate the effectiveness of our algorithm, we will conduct case studies to compare our solution approach with the current approach adopted by the restaurant, some very naive heuristics, and some upper bound of an optimal solution. Furthermore, extended experiments will be performed to evaluate the robustness of our algorithm through various scenarios. Finally, we will provide some closing remarks to conclude our study.