# Optimal Seat Assignment for a Harmonious Wedding Reception

Loan Tricot

## Contents

# 1 Introduction

Weddings are joyous events that bring together friends and family from various aspects of a couple's life. One of the most crucial aspects of planning a wedding reception is ensuring that the seating arrangement contributes to an enjoyable atmosphere for everyone involved. Achieving this requires careful consideration of various factors, such as the guests' relationships with one another, their preferences for table locations, and the specific characteristics of each table. In this study, I explore the development of an integer optimization model to determine the optimal seating assignment for guests at a wedding reception.

# 2 The wedding model

I model the problem as a generalized assignment problem with a quadratic objective. There are $n$ guests and $m$ tables. The binary decision variable $t_{ik}$ is 1 if and only if guest $i$ is assigned to table $k$. The parameter $a_{ij}$ is the affinity of guest $i$ for guest $j$, while $v_{ik}$ describes how much guest $i$ values table $k$. $c_k$ is the capacity of table $k$. With these definitions, the problem of assigning guests to tables to maximize the affinities of guests seated together is expressed as follows:

$$\begin{aligned} \max \ & \sum_{i,j,k} t_{ik} t_{jk} a_{ij} + \sum_{i,k} t_{ik} v_{ik} \\ s.t. \ & \sum_{k} t_{ik} = 1 \\ & \sum_{i} t_{ik} \leq c_k \\ & t_{ik} \in \{0, 1\} \end{aligned} \tag{1}$$

The complexity of the problem lies mostly in the quadratic term of the objective. Separating the sums in the quadratic term yields:

$$\sum_{k} \sum_{i,j} t_{ik} t_{jk} a_{ij}$$

The inner sum represents the contribution table $k$. This is the sum of the affinities of guests seated at table $k$ for each other. Interchanging the sums gives:

$$\sum_{i} \sum_{j,k} t_{ik} t_{jk} a_{ij}$$

And the inner sum now represents the contribution of guest $i$ to the objective. This the sum of the affinities of guest $i$ for all other guests seated at their table.

It is sometimes easier to work with the problem with matrix notation. Denoting $T = (t_{ik})_{ik}$, $A = (a_{ij})_{ij}$ and $V = (v_{ik})_{ij}$, I write:

$$\begin{aligned} \max\ & \langle TT', A\rangle + \langle T, V\rangle \\ s.t.\ & T1 = 1 \\ & T'1 \preceq c \\ & T \in \{0,1\}^{n\times m} \end{aligned} \tag{2}$$

Throughout the rest of these notes, I will alternate between both notations.

# 3 Lagrangian Dual Decomposition

My main idea is to deal with the quadratic term in the objective by performing Lagrangian Dual Decomposition with a twist. I first introduce another decision variable $Q$ and constrain it to be the transpose of $T$ by setting $Q = T'$. I modify the objective to transform the quadratic term into a bilinear term:

$$\begin{aligned} \max\ & \langle TQ, A\rangle + \langle T, V\rangle \\ s.t.\ & T1 = 1 \\ & Q1 \preceq c \\ & T' = Q \\ & T, Q' \in \{0,1\}^{n\times m} \in \{0,1\}^{n\times m} \end{aligned}$$

I then dualize this constraint and introduce the dual variable $\Lambda \in \mathbb{R}^{m\times n}$:

$$\begin{aligned} \max\ & \langle TQ, A\rangle + \langle T, V\rangle + \langle \Lambda, T' - Q\rangle \\ s.t.\ & T1 = 1 \\ & Q1 \preceq c \\ & T, Q' \in \{0,1\}^{n\times m} \in \{0,1\}^{n\times m} \end{aligned} \tag{3}$$

Denoting $\Lambda = (\lambda_{ik})_{ik}$, the dual term in the objective is:

$$\lambda_{ki}\left(t_{ik} - q_{ki}\right)$$

Notice the quadratic term $TT'$ now shows up as a bilinear term $TQ$ in the objective. The problem in this form can further be separated into two subproblems, respectively in $T$ and in $Q$. Because the objective is bilinear, $Q$ appears in the objective of the subproblem in $T$ and vice versa. The problem in $T$ is a trivial assignment problem:

$$\max\{\langle T, AQ' + V + \Lambda'\rangle : T1 = 1, T \in \{0,1\}^{n\times m}\} \tag{4}$$

Effectively, solving for $T$ means assigning each guest to their favorite table, accounting for the guests currently seated. The value of each table for each guest is adjusted by the dual variable $\Lambda$. The problem in Q is a trivial knapsack problem:

$$\max\{\langle Q, T'A - \Lambda\rangle : Q1 \preceq c, Q \in \{0,1\}^{m\times n}\} \tag{5}$$

Solving for $Q$ means filling each table to maximize its contribution to the objective, assuming the value of tables for each guest is depends on the guests currently seated at the table. While my model weighs all guests equally (no guest can take two seats

at a table), the generalized assignment problem allows weights to vary. In the general case, the knapsack problem isn't trivial.

It is now straightforward to obtain good feasible solutions using a subgradient scheme over $\Lambda$. Because the objective is bilinear, there are no convergence guarantees.

## 3.1 Accelerating the subgradient scheme

One trick is to observe that, because every guest sits with themselves anyways, artificially inflating the diagonal of $A$ has no effect on the solution of Problem 2. It does however have an effect on the subgradient scheme: indeed, because $T$ and $Q$ are separate variables, guests don't always sit with themselves! Recall the coefficients for $T$ and $Q$ respectively for the two subproblems:

$$M = AQ' + V + \Lambda'$$
$$P = T'A - \Lambda$$

Let $\sigma \in \mathbb{R}^n$ and $\Sigma$ denote the matrix whose diagonal terms are the elements of $\sigma$. Inflating the diagonal of $A$ by the vector $\sigma \in \mathbb{R}^n$ amounts to modifying $M$ and $P$ the following way:

$$M_0 = (\Sigma + A)Q' + V + \Lambda' = M + \Sigma Q'$$
$$P_0 = T'(\Sigma + A) - \Lambda = P + \Sigma T'$$

Observe the objective now encourages $T$ to resemble $Q'$, and $Q$ to resemble $T'$. This idea can be formalized by reconsidering the constraint

$$T' = Q \tag{6}$$

in the Lagrangian Dual Decomposition. Indeed, the structure of $T$ and $Q$ allows us to considerably simplify this constraint, which currently enforces $n \times m$ equalities. Suppose firstly that the table's capacity constraints are binding. In Problem 2, this amounts to transforming the second constraint into $T'1 = c$. I find the following equivalence:

$$T' = Q \Leftrightarrow \langle T', Q \rangle = n \tag{7}$$

Thus reducing the constraint to a single equality, which can be enforced via a single dual variable. Denoting the terms of $T$ by $t_{ik}$ and those of $Q$ by $q_{ki}$, another constraint to look at is:

$$T' = Q \Leftrightarrow \sum_k t_{ik} q_{ki} = 1 \ \forall i \tag{8}$$

This constraint can be enforced by $n$ dual variables. The dual term in the objective is:

$$\sigma_i \left( \sum_k t_{ik} q_{ki} - 1 \right)$$

The 3 constraints described in equations 6, 7, and 8 correspond to subgradient schemes which exercise more or less granular control over the constraint $T' = Q$. The latter two constraints differ from the first in that they are bilinear in $T$ and $Q$. Suprisingly, this is what makes them so effective. I expand the Lagrangian Dual Decomposition for constraint 8, denoting $\sigma$ the vector of $n$ dual variables, and $\Sigma$ the square matrix whose diagonal terms are the elements of $\sigma$:

$$\begin{aligned} \max\ & \langle TQ, A + \Sigma \rangle + \langle T, V \rangle - \sigma' 1 \\ s.t.\ & T1 = 1 \\ & Q1 \precsim c \\ & T, Q' \in \{0,1\}^{n \times m} \in \{0,1\}^{n \times m} \end{aligned} \tag{9}$$

Notice that while minimizing over $\sigma$, the subgradient with respect to $\sigma$ cannot be positive: $\sigma$ increases monotonically throughout the optimization so that convergence is guaranteed.

## 3.2 The Augmented Lagrangian Dual Decomposition

The preceding section introduces non-linear constraints for the dual decomposition. This section focuses on the augmented lagrangian dual with respect to the constraint $Q' = T$. The idea is to introduce a non-linear penalty for violating the decomposition constraint in the objective. I focus on the following penalty, parameterized by the dual variables $\sigma_{ik}$:

$$\sum_{ik} \sigma_{ik} \left(t_{ik} - q_{ki}\right)^2$$

Because of the binary nature of the variables, $\left(t_{ik} - q_{ki}\right)^2$ can be expanded into $t_{ik} + q_{ki} - 2t_{ik}q_{ki}$, and we find a familiar bilinear term enforcing the decomposition constraint. Denoting $\lambda_{ij}$ the dual variables for the standard penalty on $t_{ik} = q_{ki}$, I give the full objective of the augmented problem:

$$\sum_{ijk} a_{ij} t_{ik} q_{kj} + \sum_{ik} t_{ik} v_{ik} + \sum_{ik} \lambda_{ik} \left(t_{ik} - q_{ki}\right) - \sum_{ik} \sigma_{ik} \left(t_{ik} + q_{ki} - 2t_{ik}q_{ki}\right)$$

The coefficient of $t_{ik}$, which serves in the objective of the first subproblem, is:

$$\lambda_{ik} + \sigma_{ik} \left(2q_{ki} - 1\right) + v_{ik} + \sum_{j} a_{ij} q_{kj}$$

Notice $2q_{ik} - 1$ is 1 when $q_{ik} = 1$ and $-1$ otherwise. The role of $\sigma_{ik}$ is obvious. An immediate and valuable characteristic of the augmented problem is that there exists finite $\sigma_{ik}$ for which optimal solutions satisfy $t_{ik} = q_{ki}$. While this is a general property of augmented lagrangian duals, it is also obvious for this problem.

## 3.3 Regularizing dual variables

The last two preceding sections propose subgradient schemes which benefit from the monotonic increase of the dual variables. While this is beneficial for convergence, it

also has a drawback: inappropriate learning rates lead to overshooting. Large learning rates lead to fast convergence but overshoot. Small learning rates reduce the risk of overshoot at the cost of slower convergence.

To allow for large learning rates without suffering from overshoot, I have explored variable decay. This is equivalent to regularizing the dual variables with a quadratic penalty. The danger of introducing decay is that it enforces an upper bound on the dual variables, which does away with the convergence garantees of our last subgradient scheme. Additionally, decay necessarily makes convergence slower. Nevertheless, the feasible solutions found by the subgradient schemes are often slightly better with variable decay than without.

# 4 Heuristics and feasible solutions

The goal of heuristics is to obtain good feasible solutions from scratch or from good near feasible solutions. I begin by observing a property of partial solutions which is relevant for the design of heuristics.

## 4.1 Partial solutions and a disjunction

Consider the disjunction $t_{i1} = 1 \vee \cdots \vee t_{im} = 1$. The k$^{th}$ term represents assigning guest $i$ to table $k$. The conjunction is valid, as it represents a row of Problem 2, part of the constraint $T1 = 1$. This disjunction is interesting because each of the $m$ nodes it generates are problems with precisely the same structure as Problem 2. Indeed, setting $t_{ik} = 1$ and simplifying the formulation yields the following problem:

$$\begin{aligned} \max\ & \langle TT', A_{ii} \rangle + \langle T, V_{ik} \rangle \\ s.t.\ & T1 = 1 \\ & T'1 \lesseqgtr c_i \\ & T \in \{0,1\}^{(n-1)\times m} \end{aligned} \tag{10}$$

Where all of $A_{ii}$, $V_{ik}$ and $c_i$ are modified versions of $A$, $V$ and $c$. Effectively $i$ is removed as a guest, while the value of table $k$ increases for all guests proportionally to their affinity for guest $i$, and of course the capacity of table $k$ is decreased by 1. If the table's new capacity is 0, the problem can be further simplified.

In some sense, the reduced problem is obtained by shifting some of the objective from the quadratic term to the linear term. In the following I use $t_{ij}$, $a_{ij}$, and $v_{ik}$ respectively to refer to the terms of $T$, $A$ and $V$. Before assigning guest $i_0$ to table $k_0$, the contribution of any guest $j$ to the objective is

$$\sum_{i,k} t_{ik} t_{jk} a_{ij} + \sum_{k} t_{jk} v_{jk}$$

After assigning guest $i_0$ to table $k_0$, the new contribution is

$$\sum_{i \neq i_0, k} t_{ik} t_{jk} a_{ij} + \sum_{k} t_{jk} v_{jk} + t_{jk_0} a_{j,i_0}$$

Moreover, by assigning high value guests first, the quadratic term's contribution decreases faster and the problem becomes easier to solve.

## 4.2 Greedy heuristics

Motivated by the property of partial solutions, I begin by exploring heuristics which find good feasible solutions from scratch. There are two obvious greedy heuristics. The first iterates on guests and assigns each to a table greedily. The second iterates on tables and fills each with guests greedily. This latter heuristic assumes the capacity constraints are all binding, but nothing is lost with this assumption, as guests of no value can be added to the problem to make it so.

The heuristics described below resemble subproblems 4 and 5. The difference between these heuristics and their corresponding subproblem is that the heuristics respect all constraints of the original problem, and act on one guest at a time rather than all of them at once.

The greedy guest assignment heuristic performs the worst. This is because, so long as the $a_{ij}$ are positive, every iteration assigns the guest to the table with the most guests. As this is true for every iteration, this amounts to assigning consecutive guests to the same table. The greedy table filling heuristic performs on par, if not better, than the solutions obtained from randomly fixing partial solutions obtained from the first subgradient scheme. Combining the greedy table filling heuristic with the first subgradient scheme is a significant improvement.

### 4.2.1 Greedy guest assignment

The idea is to assign guests to tables based strictly on table capacity and the value guests assign to each table, reflected in $V$. When guest $i$ is assigned to table $k$, part of the quadratic term of the objective is shifted to the linear term, by adding the affinity of other guests for guest $i$ to the k$^{th}$ column of $V$.

1. For each guest $i$
    1. Assign guest $i$ to the best table $k_0$ with remaining capacity, that is $k_0 = \arg\max_k \{v_{ik} : c_k > 0\}$
    2. Adjust $V$ to reflect the new table values for the remaining guests by setting $v_{jk_0} \coloneqq v_{jk_0} + a_{ji}$
    3. Adjust the capacity of table $k_0$ by setting $c_{k_0} \coloneqq c_{k_0} - 1$

### 4.2.2 Greedy table filling

Greedy table filling is similar to greedy guest assignment, for each table. Indeed, a table selects a guest $i$ according to the values in its corresponding column of $V$. When guest $i$ is assigned, the column of $V$ is adjusted to reflect other guests' affinities for guest $i$. This process is iterated until the table is full. The greedy procedure then turns its attention to another table.

1. For each table $k$, until $c_k = 0$
    1. Assign to table $k$ the best guest $i_0$, that is $i_0 = \arg\max_i \{v_{ik} : i \text{ unassigned}\}$
    2. Adjust $V$ to reflect the new table values for the remaining guests by setting $v_{jk} \coloneqq v_{jk} + a_{ji_0}$
    3. Adjust the capacity of table $k$ by setting $c_k \coloneqq c_k - 1$

### 4.3 Fixing a near feasible solution naively

The most obvious heuristic is the most naive. Starting from an optimal solution $T$ to Problem 4, for each table with too many guests, reassign an excess guest to a table with excess capacity. This heuristic can benefit from simple ideas:

- The guest to be reassigned can be the least valuable guest at the table
- The table to which the guest is reassigned can be such that the objective is improved the most, or deteriorated the least

This heuristic works by making local modifications. It's possible to look at the problem more globally without significantly complicating the reasoning. For each table with excess guests, select all the guests which will be reassigned, and fix the rest of the solution. The reassignment problem shares the same structure as the original problem, but there are less decision variables and the linear term is relatively more important than the quadratic term, so that heuristics will perform well.

### 4.4 Locally improving solutions

Any two guests can change seats. It is easy to check if doing so is beneficial. This is a local improvement rule which does not garantee convergence, but which somewhat improves the value of feasible solutions output by heuristics in practice.

Any local search is defined on a set of moves. I discussed swapping the seats of two guests. It is possible to consider more complex moves. The most general version of the guest swap rule makes use of the property of partial solutions. Fixing the seats of all but $w$ guests, I recover an assignment problem of the same structure, which can be solved exactly when $w$ is small and approximately when $w$ is larger. How to choose which $w$ guests to act on remains unspecified.

## 5 Branch and bound

### 5.1 Upper bounds

The lagrangian dual decomposition is not amenable to generating upper bounds efficiently. Indeed, maximizing a non-convex bilinear objective with respect to binary variables to optimality is not easy.

An easy way to generate upper bounds is to bound the maximal contribution to the objective of each guest. The best any guest can hope for at a given table is to be seated with their favorite guests. The score for guest $i$ and table $k$ is then $v_{ik}$ added

to the sum of the best $c_k - 1$ affinities $a_{ij}$ above 0. Taking the max across $k$ and the sum across $i$ yields an upper bound for the full objective. Note this is similar to the computation performed to solve Problem 5.

### 5.2 Exploring the tree

The disjunction explored in the section about heuristics generates a tree which can be explored by a branch and bound algorithm. The branching strategy remains to be specified. It is fruitful to query information from the subproblems obtained through the lagrangian dual decomposition:

- The first subproblem ranks guests by their contribution to the objective
- The second subproblem ranks tables by their contribution to the objective

### 5.3 The Ryan disjunction

For any $i, j$, Ryan considers the following disjunction, which expresses that guests either sit together or at different tables:

$$\forall k \; t_{ik} = t_{jk} \vee \exists k \; t_{ik} \neq t_{jk}$$

Under the assignment constraint, there is a simpler way to express this nonlinearly:

$$\sum_k t_{ik} t_{jk} = 1 \vee \sum_k t_{ik} t_{jk} = 0$$

Which is reminiscent of the bilinear constraints explored in the section about the lagrangian dual decomposition.

## 6 Linearizations

### 6.1 A basic linearization

Because the decision variables are binary, it is possible to linearize the quadratic term in the objective by introducing additional decision variables. This is more easily understood by working with the index formulation of Problem 1.

I introduce the decision variables $z_{ijk}$, where $i$ and $j$ range over guests while $k$ ranges over tables. It's possible to ensure $z_{ijk} = t_{ik} t_{jk}$ by constraining $z_{ijk}$ linearly as follows:

$$z_{ijk} \leq t_{ik}$$
$$z_{ijk} \leq t_{jk}$$
$$z_{ijk} \geq t_{ik} + t_{jk} - 1$$

A linear equivalent to Problem 1 is expressed as follows:

$$
\begin{aligned}
\max \quad & \sum_{i,j,k} z_{ijk} a_{ij} + \sum_{i,k} t_{ik} v_{ik} \\
s.t. \quad & \sum_{k} t_{ik} = 1 \\
& \sum_{i} t_{ik} \leq c_k \\
& z_{ijk} \leq t_{ik} \\
& z_{ijk} \leq t_{jk} \\
& z_{ijk} \geq t_{ik} + t_{jk} - 1 \\
& t_{ik} \in \{0, 1\} \\
& z_{ijk} \in \mathbb{R}
\end{aligned}
\tag{11}
$$

Though this problem is linear, yet it is much larger in size than the original problem. Indeed, it has over $n^2m$ columns and rows, while the original problem has $nm$ columns and $n + m$ rows. The linearization is however redeemed by the fact that the $z_{ijk}$ need not be constrained to be binary: this is in fact implied by their relationship with the $t_{ik}$. Gurobi can solve this linearization a bit faster than the quadratic formulation.

## 6.2 RLT Cuts

Multipliying the constraint $\sum_i t_{ik} \leq c_k$ by $t_{jk}$ and substituting in the linearized variables yields a valid inequality for any $i$ and $j$:

$$
\sum_{i} z_{ijk} \leq c_k t_{jk}
$$

## 6.3 Compactification of the naive linearization

An idea from Leo Liberti.

The naive linearization would have us introduce $z_{ikjl} = t_{ik} t_{jl}$. With this linearization it is possible to leverage the assignment constraint to replace the linearization constraints, which number on the order of $n^2m^2$, and make the formulation more compact:

$$
\begin{aligned}
& \sum_{k} t_{ik} = 1 \ \forall i \\
\Leftrightarrow & \sum_{k} t_{ik} t_{jl} = t_{jl} \ \forall i, j, l \\
\Leftrightarrow & \sum_{k} z_{ikjl} = t_{jl} \ \forall i, j, l
\end{aligned}
$$

Unfortunately the compactification isn't enough to compensate for the extra decision variables. Gurobi cannot solve even the smallest instances of the problem in reasonable time with this formulation.

### 6.4 The linear formulation of the max clique problem

The formulations given in this subsection are taken from a survey on Unconstrained Binary Quadratic Programming.

The problem I have studied is related to the clique partitioning problem, which aims to partition the nodes of a graph into cliques of maximum size. The standard formulation of the clique partitioning problem is as follows:

$$\begin{aligned} \max \quad & \sum_{ij} w_{ij} x_{ij} \\ s.t. \quad & x_{ij} + x_{ir} - x_{jr} \leq 1 \\ & x_{ij} \in \{0, 1\} \end{aligned}$$

The decision variable $x_{ij}$ is equal to 1 if the edge from node $i$ to node $j$ is included in the partition. This formulation does not bound the number of cliques, nor the size of each clique. An alternative formulation of this problem bears striking resemblance to my formulation:

$$\begin{aligned} \max \quad & \sum_{ij} w_{ij} \sum_{k} x_{ik} x_{jk} \\ s.t. \quad & \sum_{k} x_{ik} = 1 \\ & x_{ik} \in \{0, 1\} \end{aligned}$$

Where $x_{ik}$ is 1 if node $i$ is assigned to clique $k$. The number of cliques can be bounded by fixing the range over which $k$ indexes. My problem is only different in that the size of each clique is fixed.

## 7 Unconstrained Binary Quadratic Programming

Many combinatorial problems expressed as linear binary programs can be reformulated as unconstrained binary quadratic programs. This is because linear constraints can be transformed into quadratic penalty terms. My problem is currently formulated as a binary quadratic program with linear constraints, but these linear constraints can be transformed into quadratic penalty terms.

An assignment constraint of the form $\sum_k x_{ik} = 1$ can be transformed into a penalty term of the form $M\left(\sum_k x_{ik} - 1\right)^2$, which can be simplified to

$$M \sum_{k} \sum_{l>k} 2 x_{ik} x_{il}$$

Thus the objective of the problem remains quadratic.