**Travelling Salesman Problem with Sliding Windows for Concert Venue Delivery – Group 3 Project**

**MGSC 695: Advanced Topics in Management Science – Decision Analytics for Operations**

**David Escobar-Vargas**

**Tyler Nagano 261099339**

**Ruhi Mahendra 261079954**

**Vibhu Bhardwaj 261113187**

**Amr Maraqa 261101609**

**Rohana Habib 261110469**

**1. Introduction**

Our given case study is a Travelling Salesman Problem with Time Windows (TSPTW). TSPTW is an extension of the familiar Travelling Salesman Problem (TSP) where a list of cities and respective distances between each pair of cities is given and the goal is to minimize the total distance travelled. In a TSPTW, time windows in which the salesman can visit each city are added. The objective is to find the route which minimizes total distance travelled while ensuring each city is visited within its time window. In the case study provided, the band needs to visit a set of venues each in different cities within a certain time window to meet staff time availabilities while minimizing the total distance travelled. Additionally, each venue location must be visited once, and the plane capacity and staff time windows must be respected.

There is a known problem structure in the scientific literature that fits the case study provided. The paper *Evaluation of the size of time windows for the travelling salesman problem in delivery operations* looks at a TSPTW problem as an integer linear programming model. The model is applied to a set of benchmark instances to find the optimal routes that minimize the tour duration.

**2. Problem Formulation**

There are two main components in building an optimization model. The first is the procurement and definition of the data. Depending on the way the data is defined, the variables, objective function and the constraints are constructed mathematically. This section discusses the definition of the factors in the data and the subsequent process of problem formulation.

**2.1. Data Definition & Preprocessing**

The data used for this optimization problem is a table containing information pertaining to 152 venues in which the band may perform. Table 2.1 below is a dictionary of the variables that represent each venue.

|  |  |
| --- | --- |
| *X-coordinate* | *The position along the x-axis of the venue* |
| *Y-coordinate* | *The position along the y-axis of the venue* |
| *Demand* | *The weight, in kg, of the equipment required for each venue* |
| *Ready Time* | *The time a venue would be ready to host a concert* |
| *Due Date* | *The time after which a venue would not be able to host a concert* |
| *Service Time* | *The length of the concert at the venue* |

*Table 2.1. Data Dictionary*

There is an important assumption to be aware of while formulating the model. The demand and service time are assumed to be the same across all venues. This makes the problem much easier to formulate, as it eliminates the need to create variables for these aspects of a venue.

Once the data and the assumptions were defined, the distance between each venue and all the other venues was calculated and, later, stored in a distance matrix containing the distances between all the venues. This matrix was used in the objective function, as described later in the paper.

**2.2. Model Construction**

The process of building a model entails the definition of several elements, namely variables, objective function, and constraints that help describe the problem at hand. The following subsections present these elements and their descriptions.

2.2.1. Sets:

The only set that is necessary to specify for the TSP problem the band is facing is that of the venues. There are 152 venues, in total, as depicted below.

*V: Venues {1,2,…,151}*

2.2.2. Decision variables:

The variables in the band’s itinerary address the aspects of the trip pertaining to distance and time. Since the distance between the venues is fixed, the distance of the trip is determined by which venues are visited and the order in which they’re visited. Hence, a binary variable, eij, is created to indicate whether a route between venues i and j is taken or not. Since each venue can host concerts within a different time window, variables for arrival and departure time are created to make sure venues are visited within their designated time windows. Table 2.2.2. summarizes the model variables.

|  |  |
| --- | --- |
| *eij* | *Edge between i and j, binary variable, i ∈ V, j ∈ V, eij = 1 if band travels from i to j* |
| *ti* | *Band’s Arrival Time to i, i ∈ V* |
| *tj* | *Band’s Departure Time from j, j ∈ V* |

*Table 2.2.2. Model variables*

**2.2.3. Parameters**

Model parameters are constants that are used frequently throughout the different components of the model. The parameters defined in this problem are the minimum arrival time and the maximum departure time of each venue, the time of travel from venue i to venue j, and the distance between i and j (the elements of the distance matrix previously discussed).

|  |  |
| --- | --- |
| *ai* | *The earliest time band can arrive to venue i, i ∈ V* |
| *bi* | *The latest time band can arrive to venue i, i ∈ V* |
| *tij* | *Travel time from i to j, i ∈ V, j ∈ V* |
| *dij* | *Euclidean distance from i to j i ∈ V, j ∈ V* |

*Table 2.2.3. Model parameters*

**2.2.4. Objective Function**

The goal of the model is to optimize the band’s concert tour itinerary by minimizing the total traveled distance. The following equation displays the mathematical formula of the objective function.

**2.2.5. Constraints**

Some constraints were specified to bind the final solution to reflect the reality of the band’s journey. The constraints appended are listed in the table below.

|  |  |  |
| --- | --- | --- |
|  |  | *Initiation of arrival to 1st node: the arrival to venue i occurs after traveling from initial point to i.* |
|  |  | *Ensures arrival occurs after the min arrival time of the venue* |
|  |  | *Sub-tour elimination constraint* |
|  |  | *Visit each venue once & only once* |
|  |  | *Depart from each venue once & only once* |
|  |  | *Number of venues constraint: min number of venues to visit is 80 and max is 120.* |
|  |  | *Ensures departure occurs before the max departure time of the venue* |
|  |  | *Tour duration constraint* |
|  |  | *Max Capacity constraint: the plane can carry max 500 kg of supplies* |
|  |  | *Venue Demand Range: each venue needs 1 to 4 kg of supplies* |

*Table 2.2.4. Model constraints*

The sub-tour elimination constraint ensures that no smaller travel circuits are recommended by the model. This is done by ensuring that the time of arrival in j is greater than that in i plus the time to travel from i to j. This makes sure that the algorithm cannot choose the path from j to I, as that would contradict the constraint just mentioned. Moreover, the constraint aligns with the minimum arrival and maximum departure constraints by incorporating the parameters.

**3. Python Code Implementation**

The Python code implementation can be found in the Group-3-Project.ipynb notebook file. This file includes the data preprocessing, model formulation and parameter tuning results. All results of the project can be found on [Github](https://github.com/tnagano99/Concert_TSPTW_Optimization).

Note that In the Python Code Implementation, we did not consider the plane capacity as even in the worst case (120 venues, 4kg per venue) is 480kg needed which is less than the 500kg capacity of the plane.

**4. Scenario Testing**

We defined 3 different scenarios to test our model formulation using the minimum and maximum venues given. The small scenario has 80 venues, the medium scenario has 100 venues and the large scenario has 120 venues.

To create the scenarios, we used data from the provided dataset in TSPTW\_150.txt file. Each dataset includes the first entry which is the starting location for the problem. To add the rest of the venues to the formulation we sampled from the population without replacement.

After generating the scenarios, the distance matrix for each formulation was calculated using the Euclidean distance for each pair of locations. In our formulation, we decided to use the Euclidean distance in place of time between two locations.

**5. Gurobi Parameter Exploration**

**5.1. Results of Parameter Exploration**

We had 3 scenarios (Small, Medium and Large) and decided to test the effect of MIP cuts parameters individually on different scenarios. The parameters tested along with ranges are mentioned below can be found in Appendix: MIP Cuts Parameters.

Appendix: Parameter Optimization Results contain subsets of the best and worst performing tuning parameters. A complete output of all combinations is given in Cut\_Params.csv and Passes\_Params.csv.

**5.2 Parameter Tuning with .tune() Model method**

For the small scenario, 18 different parameter sets were tested automatically to find a more optimal solution run time. The small scenario improved the run time from an average of 0.06s to an average of 0.03s. The most optimal parameter set changed the Heuristics parameter set to 0.5 instead of the default value.

For the medium scenario, 17 different parameter sets were tested automatically to find a more optimal solution run time. The medium scenario improved the run time from an average of 0.08s to an average of 0.04s. The most optimal parameter set changed the MIPFocus parameter set to 1 and Cuts to 0.

For the small scenario, 14 different parameter sets were tested automatically to find a more optimal solution run time. The small scenario improved the run time from an average of 0.11s to an average of 0.07s. The most optimal parameter set changed the MIPFocus parameter set to 1 and Heuristics to 0.

A screenshot of each output can be seen in Appendix: Model Tune Results.

**5.3 Gurobi Parameter Optimization Recommendations**

**5.3.1. Small Scenario**

We can use the parameters and their respective values to achieve better results than from using the .tune() method. In all cases, the objective function value did not change only the solution runtime improved.

**Table 1. Optimal Parameters and Values for Small Scenario**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Parameter Value | Obj Value | Runtime |
| GomoryPasses | 27 | 545.3112 | 0.024826 |
| GomoryPasses | 24 | 545.3112 | 0.028723 |
| GomoryPasses | 6 | 545.3112 | 0.03075 |
| CutAggPasses | 97 | 545.3112 | 0.031216 |

**5.3.2. Medium Scenario**

We can use the parameters and their respective values to achieve better results than from using the .tune() method. In all cases, the objective function value did not change only the solution runtime improved.

**Table 2. Optimal Parameters and Values for Medium Scenario**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Parameter Value | Obj Value | Runtime |
| GomoryPasses | 67 | 599.7767 | 0.043198 |
| GomoryPasses | 61 | 599.7767 | 0.045654 |
| GomoryPasses | 3 | 599.7767 | 0.046886 |
| CutPasses | 29 | 599.7767 | 0.046963 |
| CutPasses | 34 | 599.7767 | 0.047274 |

**5.3.3. Large Scenario**

We can use the parameters and their respective values to achieve better results than from using the .tune() method. In all cases, the objective function value did not change only the solution runtime improved.

**Table 3. Optimal Parameters and Values for Large Scenario**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Parameter Value | Obj Value | Runtime |
| GomoryPasses | 99 | 644.0674 | 0.062202 |
| CutAggPasses | 22 | 644.0674 | 0.062334 |
| GomoryPasses | 62 | 644.0674 | 0.06424 |
| GomoryPasses | 93 | 644.0674 | 0.065002 |
| LiftProjectCuts | -1 | 644.0674 | 0.072546 |

**6. Paper Summary and Analysis**

**6.1. Problem Definition (Everyone)**

The paper investigates the impact of time windows on the travelling salesman problem (TSP) in delivery operations. The authors developed a mathematical model to find the optimal delivery routes that minimize the tour duration while satisfying the time windows. They use an integer linear programming model to solve the problem and evaluate the results based on solution quality, computational time, and convergence rate. The study uses a simulated delivery environment with a fixed number of customers, delivery locations, and time windows. The delivery environment is subject to various factors that affect delivery time, including traffic congestion and delivery time variability. The simulation is run with different time window sizes, ranging from 30 minutes to 4 hours. The results show that extremely small- or large-time windows can render the delivery operations infeasible or result in poor customer satisfaction and high delivery costs respectively. There is a trade-off between customer satisfaction and delivery cost, and a multi-objective optimization approach is needed to balance the two objectives. Problems such as TSPTW are relevant as it has applications in various fields, including logistics, transportation, and scheduling.

**6.2. Model (the reason/advantage of using this formulation) (Vibhu)**

The formulation the authors used is different than other formulations of the TSP problem more generally. Some other formulations implement the problem without a sub-tour elimination constraint, but instead use lazy constraints in which they iteratively add new constraints after each time the model solves. This is done to eliminate sub-tours found in the current solution for the next iteration. The formulation in the paper does implement a sub-tour elimination constraint which requires N2 constraints to be added to ensure a feasible solution without sub-tours. It is beneficial to use lazy constraints in the model as this can decrease the memory required to find a solution, but this also requires that the model be run many times. The implementation in the paper is better if computational power is not an issue and the process of subtour elimination is much clearer to understand.

**6.3. Solution Method (Tyler)**

The mathematical solution includes customers (C) and nodes in the network (N), such as the starting node (N0) and nodes that the vehicle can visit (N+). The decision variables (xij) indicate whether the vehicle travels from node i to node j, with ti and tj representing the time at which the vehicle arrives at nodes i and j, respectively. The parameters include the earliest and latest delivery times for each customer (ai and bi) and the travel time between nodes (tij). The mathematical solution of the TSPTW described in the paper uses an objective function which attempts to minimize the time it takes to visit all customers, including starting and ending nodes. This objective function implementation is effective when the speed of transportation varies between different locations in the network. An alternative formulation, which we implemented in this project, uses an objective function which attempts to minimize the total distance travelled to reach every location while respecting time windows. This formulation is more effective when the speed of transportation is consistent between all locations in the network. Therefore, the formulation implemented in the paper and other formulations are useful to solve the same problem using different metrics to optimize for.

**6.4. Strong Points (Ruhi)**

Strong points of the paper include that it addresses an important and practical problem in the context of home delivery operations, which is relevant to many industries and customers. The paper's findings are likely to be generalizable to other applications of the TSP beyond delivery operations, as the TSP is a widely studied problem with many real-world applications. The authors develop a mathematical model and conduct experiments to provide insights and proposes a new approach for generating realistic time window instances, which could be useful for other researchers studying the TSPTW. The paper is also well-structured and clearly presented, with detailed explanations of the mathematical model and experiments. The paper also uses a simulation-based approach to test the performance of the TSP under different time window sizes. This approach allows the authors to test the impact of time windows in a controlled and repeatable manner, providing strong evidence for the results.

**6.5. Weak Points (Amr)**

Weak points of the paper include that it only uses five benchmark problems to examine the impact of different approaches to adjust the size of time windows. While these problems are prepared by experts and widely used in the literature, they may not fully represent the diversity and complexity of real-world delivery operations. The paper also does not consider other important factors affecting delivery operations, such as weather conditions, traffic congestion, and labour availability. Another potential weakness is that the paper does not provide a detailed analysis of the computational performance of the mathematical model and experiments, such as the solution time and convergence rate. The simulation-based approach used in the study makes several simplifying assumptions about the delivery environment, such as assuming that customers are evenly distributed across the delivery area and that delivery times are normally distributed. These assumptions may not hold in real-world delivery environments and could limit the generalizability of the findings.

**6.6. Should we use this paper as the basis to address the given case study? (Rohana)**

This paper is good as a basis to address the given case study as it provides insight into the impact of time windows on the TSPTW, which is relevant to the band's objective of respecting the staff time window at each venue. However, there are other popular papers that study other instances of the travelling salesman problem (TSP) such as Vehicle Routing Problem with Time Windows (VRPTW) and the Capacitated Vehicle Routing Problem (CVRP). The VRPTW is a well-known extension of the TSP where a set of vehicles are used to serve a set of customers with time windows, and the goal is to minimize the total distance traveled by the vehicles. The CVRP is another variant of the VRP that introduces capacity constraints, where each vehicle has a maximum capacity, and each customer has a demand that must be satisfied. These may be better suited for this case as opposed to TSPTW as it considers the capacity constraints of the single plane that the band must transport their equipment and material to each venue.

**7. Extra: Stochastic Optimization Consideration**

From a stochastic optimization perspective, the most meaningful stochastic consideration of this case study would be the uncertainty associated with the travel time between locations. In reality, travel time would be affected by various factors such as the weather or traffic which would result in potential delays in the tour schedule.

The added value of considering the problem as stochastic compared to deterministic is that the stochastic approach takes a more realistic stance providing insight to the inherent uncertainty of real-life operations. Furthermore, a stochastic approach aids in identifying potential risks and uncertainties with the tour schedule, which allows for better decision making.

While a deterministic approach would assume perfect knowledge of the travel time, a stochastic approach would generate multiple scenarios based on different travel time distributions and probabilistic models of disruptions. The model would provide an optimal tour schedule that is robust to the uncertainty of travel time which would minimize the risk of potential delays.

**Appendix: MIP Cuts Parameters**

**Table 3.** List of MIP Cuts Parameters and Values Tested

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter Name | Min Value Tested | Max Value Tested | Use Cases |
| BQPCuts | -1 | 2 | Small,Medium,Large |
| CliqueCuts | -1 | 2 | Small,Medium,Large |
| CoverCuts | -1 | 2 | Small,Medium,Large |
| FlowCoverCuts | -1 | 2 | Small,Medium,Large |
| FlowPathCuts | -1 | 2 | Small,Medium,Large |
| GUBCoverCuts | -1 | 2 | Small,Medium,Large |
| ImpliedCuts | -1 | 2 | Small,Medium,Large |
| InfProofCuts | -1 | 2 | Small,Medium,Large |
| LiftProjectCuts | -1 | 2 | Small,Medium,Large |
| MIPSepCuts | -1 | 2 | Small,Medium,Large |
| MIRCuts | -1 | 2 | Small,Medium,Large |
| ModKCuts | -1 | 2 | Small,Medium,Large |
| NetworkCuts | -1 | 2 | Small,Medium,Large |
| ProjImpliedCuts | -1 | 2 | Small,Medium,Large |
| PSDCuts | -1 | 2 | Small,Medium,Large |
| RelaxLiftCuts | -1 | 2 | Small,Medium,Large |
| RLTCuts | -1 | 2 | Small,Medium,Large |
| StrongCGCuts | -1 | 2 | Small,Medium,Large |
| SubMIPCuts | -1 | 2 | Small,Medium,Large |
| ZeroHalfCuts | -1 | 2 | Small,Medium,Large |
| CutPasses | 0 | 99 | Small,Medium,Large |
| CutAggPasses | 0 | 99 | Small,Medium,Large |
| GomoryPasses | 0 | 99 | Small,Medium,Large |

**Appendix: Parameter Optimization Results**

**Table 4.** Best 10 tuning (Lowest Run time in seconds) for Small Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Small | GomoryPasses | 27 | 545.3112 | 0.024826 |
| Small | GomoryPasses | 24 | 545.3112 | 0.028723 |
| Small | GomoryPasses | 6 | 545.3112 | 0.03075 |
| Small | CutAggPasses | 97 | 545.3112 | 0.031216 |
| Small | CutAggPasses | 50 | 545.3112 | 0.031239 |
| Small | RLTCuts | -1 | 545.3112 | 0.031256 |
| Small | GomoryPasses | 5 | 545.3112 | 0.031546 |
| Small | CutAggPasses | 20 | 545.3112 | 0.031639 |
| Small | RelaxLiftCuts | 0 | 545.3112 | 0.03167 |
| Small | CutAggPasses | 96 | 545.3112 | 0.031672 |

**Table 5.** Worst 10 tuning (Highest Run time in seconds) for Small Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Small | CutAggPasses | 2 | 545.3112 | 0.233616 |
| Small | CutAggPasses | 3 | 545.3112 | 0.228769 |
| Small | CutAggPasses | 0 | 545.3112 | 0.171688 |
| Small | ImpliedCuts | 1 | 545.3112 | 0.148838 |
| Small | ModKCuts | 2 | 545.3112 | 0.136362 |
| Small | CutAggPasses | 58 | 545.3112 | 0.13418 |
| Small | CutPasses | 65 | 545.3112 | 0.130344 |
| Small | PSDCuts | 2 | 545.3112 | 0.125961 |
| Small | LiftProjectCuts | 0 | 545.3112 | 0.108524 |
| Small | RLTCuts | 2 | 545.3112 | 0.104519 |

**Table 6.** Best 10 tuning (Lowest Run time in seconds) for Medium Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Medium | GomoryPasses | 67 | 599.7767 | 0.043198 |
| Medium | GomoryPasses | 61 | 599.7767 | 0.045654 |
| Medium | GomoryPasses | 3 | 599.7767 | 0.046886 |
| Medium | CutPasses | 29 | 599.7767 | 0.046963 |
| Medium | CutPasses | 34 | 599.7767 | 0.047274 |
| Medium | CutAggPasses | 1 | 599.7767 | 0.047289 |
| Medium | CutAggPasses | 97 | 599.7767 | 0.047295 |
| Medium | CutPasses | 73 | 599.7767 | 0.0473 |
| Medium | CutAggPasses | 98 | 599.7767 | 0.04738 |
| Medium | CutAggPasses | 83 | 599.7767 | 0.047447 |

**Table 7.** Worst 10 tuning (Highest Run time in seconds) for Medium Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Medium | FlowCoverCuts | 0 | 599.7767 | 0.21133 |
| Medium | MIRCuts | 2 | 599.7767 | 0.140862 |
| Medium | CutPasses | 41 | 599.7767 | 0.135921 |
| Medium | GomoryPasses | 96 | 599.7767 | 0.130253 |
| Medium | ZeroHalfCuts | -1 | 599.7767 | 0.129698 |
| Medium | MIPSepCuts | -1 | 599.7767 | 0.126982 |
| Medium | CutPasses | 38 | 599.7767 | 0.126856 |
| Medium | GomoryPasses | 93 | 599.7767 | 0.125744 |
| Medium | GomoryPasses | 18 | 599.7767 | 0.123627 |
| Medium | StrongCGCuts | 2 | 599.7767 | 0.115671 |

**Table 8.** Best 10 tuning (Lowest Run time in seconds) for Large Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Large | GomoryPasses | 99 | 644.0674 | 0.062202 |
| Large | CutAggPasses | 22 | 644.0674 | 0.062334 |
| Large | GomoryPasses | 62 | 644.0674 | 0.06424 |
| Large | GomoryPasses | 93 | 644.0674 | 0.065002 |
| Large | LiftProjectCuts | -1 | 644.0674 | 0.072546 |
| Large | RelaxLiftCuts | 2 | 644.0674 | 0.07263 |
| Large | PSDCuts | -1 | 644.0674 | 0.073503 |
| Large | GomoryPasses | 65 | 644.0674 | 0.073606 |
| Large | CutAggPasses | 23 | 644.0674 | 0.073624 |
| Large | LiftProjectCuts | 0 | 644.0674 | 0.073673 |

**Table 9.** Worst 10 tuning (Highest Run time in seconds) for Large Scenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Usecase | Parameter | Parameter Value | Obj Value | Runtime |
| Large | CutAggPasses | 28 | 644.0674 | 0.215067 |
| Large | CutAggPasses | 32 | 644.0674 | 0.214622 |
| Large | GomoryPasses | 37 | 644.0674 | 0.214067 |
| Large | CutPasses | 21 | 644.0674 | 0.211384 |
| Large | BQPCuts | 2 | 644.0674 | 0.203266 |
| Large | CutAggPasses | 57 | 644.0674 | 0.193108 |
| Large | RLTCuts | 2 | 644.0674 | 0.188112 |
| Large | ZeroHalfCuts | 1 | 644.0674 | 0.187328 |
| Large | CutAggPasses | 14 | 644.0674 | 0.181456 |
| Large | CutPasses | 83 | 644.0674 | 0.180746 |

**Appendix: Model Tune Results**

Text

Description automatically generated  
**Figure 1.** Screenshot of Small Scenario Model Tune Results

Text

Description automatically generated

**Figure 2.** Screenshot of Medium Scenario Model Tune Results

Text

Description automatically generated

**Figure 3.** Screenshot of Large Scenario Model Tune Results