Tristan Ford

Optimizing Schedule Reliability for Public Transit

Interim Thesis Report

2024-04-30

Abstract

This work investigates the Single Depot Vehicle Scheduling Problem with propagated delays. Congestion, construction, and other road disruptions can introduce delays to a public transit network, resulting in late arrivals to the final destinations of select of trips. These delays may then propagate to subsequently scheduled trips if there is not sufficient buffer time for the delays to be absorbed. This phenomenon results in significant increases in operational costs for transit agencies as non-productive time must be inserted in the schedule to account for potential delays. We propose a mixed integer linear program which minimizes these increased costs as well as the delays themselves. Lagrangian relaxation and Bender's decomposition are applied, while only the latter is able to solve the problem to optimality. We find that the commercial solver, Gurobi, outperforms both decomposition methods attempted in terms of solution speed.

Contents

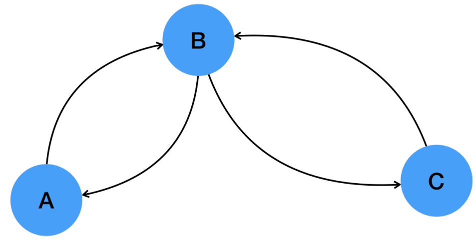
1 Introduction

The Vehicle Scheduling Problem (VSP) is an operational optimization problem with applications to many transportation network contexts. The VSP is concerned with assigning vehicles to a set of fixed origin-destination trips with prescribed start and end times whilst minimizing operational and fixed costs (Daduna & Pinto Paixão, 1995). Operational costs in public transportation encompass factors like fuel, productive and non-productive time, while fixed costs relate to vehicle investments and maintenance. Because fixed costs often dominate operational costs, the problem typically emphasizes finding vehicle schedules requiring the fewest vehicles (Bunte & Kliewer, 2009). In its simplest form, assuming a *homogeneous fleet* and a *single depot*, the VSP can be solved with polynomial-time algorithms by reducing to a minimum cost network flow problem. However, real-world challenges faced by transit agencies can make the problem more challenging to solve. For example, the VSP with *multiple depots* and/or *heterogeneous fleets* (e.g., diesel versus electric buses) have been shown to be NP-hard (Bertossi et al., 1987; Lenstra & Kan, 1981).

In this research, I will consider the Single Depot Vehicle Scheduling Problem with propagated delays for a bus-operated public transit network. This variation of the standard VSP considers the potential for trips in the network to be delayed and aims to minimize the impact on the system as well as fixed and operational costs. These delays may be characterized as *primary* or *secondary* delays. A primary delay is one that affects a trip while it is in service, that is while it is transporting passengers. These delays may occur due to traffic congestion, construction, or other road disruptions and are typically addressed through service design and infrastructure. For example, implementing a dedicated bus lane may improve the travel time reliability of trips operating along its alignment.

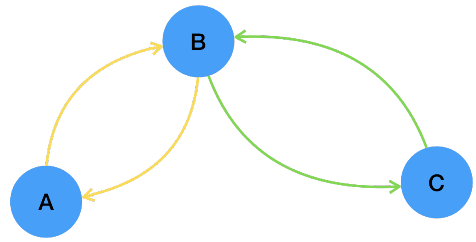
Secondary delays are measured as the time difference between the actual departure time of a given trip and its planned departure time. To illustrate the concept, consider the example shown in figure 1 where we have a network with four trips. The trip times, origins, and destinations allow for two vehicles to operate all trips. Figure 1b demonstrates the successful operation of all trips on time when the network suffers no primary delays. However, figure 1c shows how this schedule is susceptible to secondary delays should trip 1 have a primary delay greater than 5 minutes. Figure 1d then offers an alternate assignment of vehicles to trips (still utilizing two vehicles) that eliminates the secondary delay. This example demonstrates that while secondary delays are only incurred due to primary delays, we are able to moderate secondary delays through vehicle scheduling.

Trip	Start time	End time	Origin	Destination
1	9:00	10:00	A	B
2	9:15	9:55	C	B
3	10:05	11:05	B	A
4	10:10	10:50	B	C



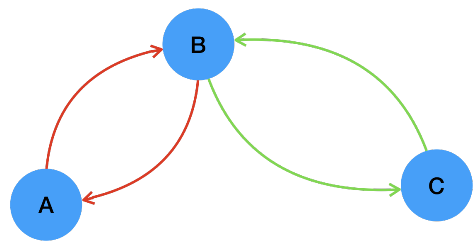
(a) the base network structure

Trip	Start time	End time	Origin	Destination
1	9:00	10:00	A	B
2	9:15	9:55	C	B
3	10:05	11:05	B	A
4	10:10	10:50	B	C



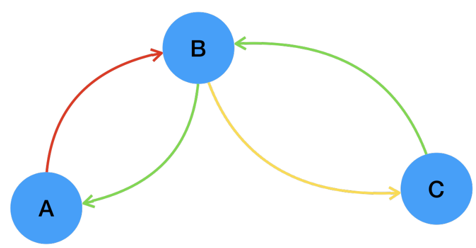
(b) the yellow vehicle completes trips 1 and 3, while the green vehicle completes trips 2 and 4

Trip	Start time	End time	Origin	Destination
1	9:00	10:00 + :10	A	B
2	9:15	9:55	C	B
3	10:05 + :05	11:05 + :05	B	A
4	10:10	10:50	B	C



(c) trip 1 suffers a primary delay of 10 minutes and causes a secondary delay of 5 minutes on trip 3

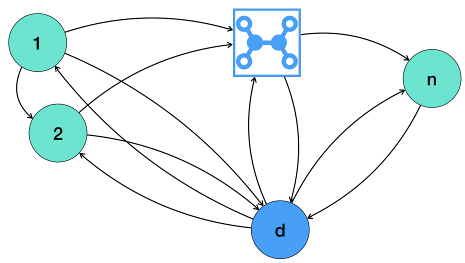
Trip	Start time	End time	Origin	Destination
1	9:00	10:00 + :10	A	B
2	9:15	9:55	C	B
3	10:05	11:05	B	A
4	10:10	10:50	B	C



(d) with alternate vehicle schedules, trip 1 still suffers a primary delay of 10 minutes but there are no secondary delays in the network

Figure 1: A simple network with four trips which may be operated with two vehicles. We observe that the alternate vehicle schedules are more robust to primary delays for trip 1.

For this problem, we introduce the following standard fully-connected network structure and notation. The set of nodes V includes all trips $T = \{1, \dots, n\}$ and the depot d. The set of arcs A includes deadheading arcs, A^{dh}, which are of the form (i, j). Deadheading arcs represent vehicle movements between any compatible trips i and j, meaning their departure and arrival times allow for them to be served by one vehicle. A also includes pull-in (A^{pi}) and pull-out (A^{po}) arcs which represent vehicle movements from the depot and to the depot, respectively.



$$
\begin{aligned}
G &= (V, A), \quad V = T \cup \{d\}, \quad V = A^{dh} \cup A^{pi} \cup A^{po} \\
x_{ij} &= \text{decision variable for arc } (i, j) \\
b_{ij} &= \text{buffer time between trips } i \text{ and } j \\
l_i &= \text{expected (primary) delay for trip } i \\
s_i &= \text{propagated delay at trip } i \\
&= \max\left\{0, \sum_{j \in T} x_{ji}(s_j + l_j - b_{ji})\right\}
\end{aligned}
\tag{1}
$$

The problem can then be formulated as a bi-objective MIP as shown below. Where (2) is the objective to minimize costs and propagated delays. Constraints (3) and (4) ensure our flows are integral and delays are nonnegative. Constraint (5) ensures the propagated delay at trip i is at least the sum of the propagated delay and primary delay of its preceding trip minus the buffer time between itself and its preceding trip. We assume the depot absorbs all delays and that no delay in incurred when leaving the depot, thus constraint (5) does not consider the depot, d. Constraints (6) and (7) are to ensure flow conservation and that each trip is operated once.

$$
\begin{aligned}
\min \sum_{i \in T} s_i + \sum_{i,j \in V} c_{ij} x_{ij} && && (2) \\
\text{subject to: } x_{ij} \in \mathbb{Z}_+ && \forall i, j \in V && (3) \\
s_i \geq 0 && \forall i \in T && (4) \\
s_i \geq \sum_{j \in T} x_{ji}(s_j + l_j - b_{ji}) && \forall i \in T && (5) \\
\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} && \forall i \in V && (6) \\
\sum_{j \in V} x_{ij} = 1 && \forall i \in T && (7)
\end{aligned}
$$

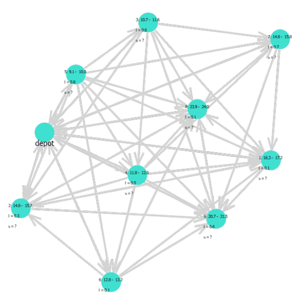
We linearize the non-linear constraint (5) by setting $\phi_{ij} = x_{ij} s_i$ and observing

$$
\begin{aligned}
&0 \leq x_{ij} \leq 1 && && 0 \leq s_i \leq \sum_{j \in T} l_j =: M \\
&0 \leq x_{ij} s_i && \implies && \phi_{ij} \geq 0 \\
&0 \leq x_{ij}(M - s_i) && \implies && \phi_{ij} \leq M x_{ij} \\
&0 \leq (1 - x_{ij}) s_i && \implies && \phi_{ij} \leq s_i \\
&0 \leq (1 - x_{ij})(M - s_i) && \implies && \phi_{ij} \geq s_i - M(1 - x_{ij}).
\end{aligned}
$$

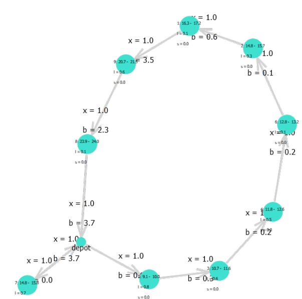
It is worth noting that the big-M value may be tightened. The value we have used simply takes the sum of all primary delays over all trips as the upper bound on an achievable propagated delay. We may improve this by determining the longest path in G, whose length is m, and setting M to be the sum of the m largest primary delays over all trips. The longest path problem is NP-hard and thus this may be challenging, however if we consider the depot as two distinct *source* and *sink* nodes our graph is a directed acyclic graph. The longest path problem can then be solved as a shortest path problem with all edge weights equal to -1.

The linearized MIP is shown in (8) and some example plots of the types of solutions we observe are shown in figure 2. Additionally, a plot of the Pareto frontier is shown in figure 3.

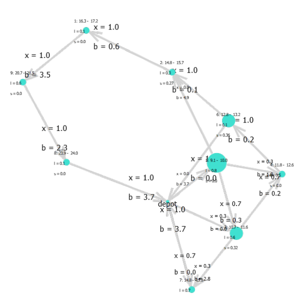
$$
\begin{aligned}
\min \sum_{i\in T} s_i + \sum_{i,j\in V} c_{ij}x_{ij} & & \\
\text{s.t.:}\quad x_{ij} \geq 0 & & \forall i,j \in V \\
s_i \geq 0 & & \forall i \in T \\
s_i \geq \sum_{j\in T} \phi_{ji} + x_{ji}(l_j - b_{ji}) & & \forall i \in T \\
\phi_{ij} \geq 0 & & \forall i,j \in T \\
\phi_{ij} \leq M x_{ij} & & \forall i,j \in T \\
\phi_{ij} \leq s_i & & \forall i,j \in T \\
\phi_{ij} \geq s_i - M(1 - x_{ij}) & & \forall i,j \in T \\
\sum_{j\in V} x_{ij} = \sum_{j\in V} x_{ji} & & \forall i \in V \\
\sum_{j\in V} x_{ij} = 1 & & \forall i \in T
\end{aligned} \tag{8}
$$



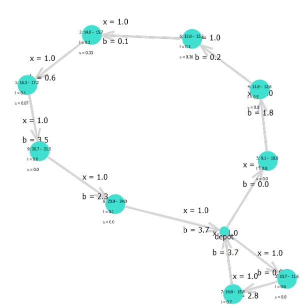
(a) instance



(b) min-cost flow solution (ignoring delays)



(c) linear solution with delays



(d) integer solution with delays

Figure 2: Network solutions from an instance with 9 trips.

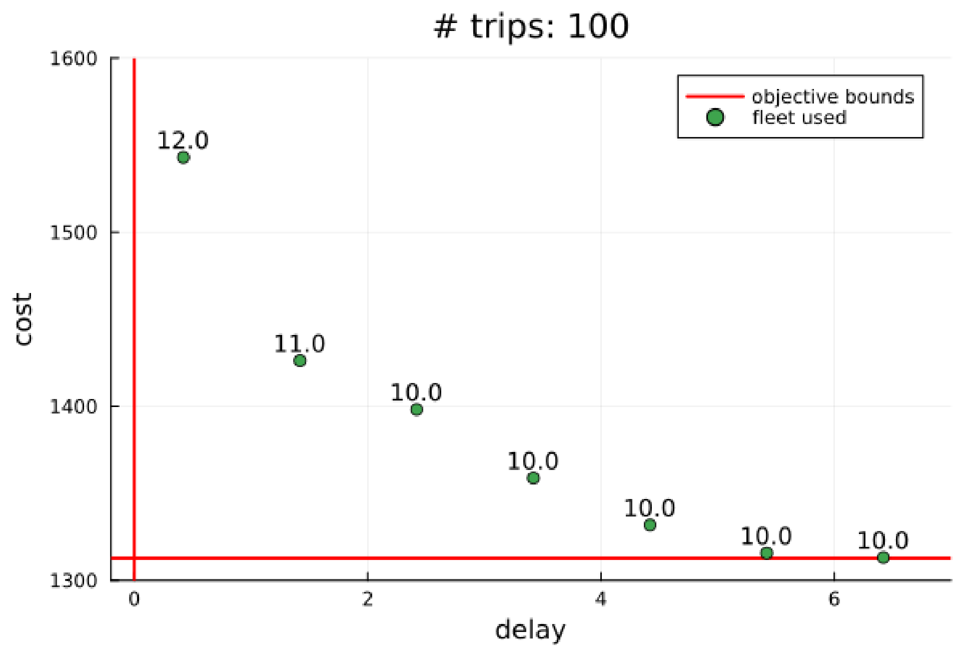


Figure 3: Pareto frontier generated for an instance with 100 trips. The number denotes the fleet required for each solution. We notice that reductions in delay require increases in operational costs as our cost objective increases with the same number of vehicles. Reductions in delay also require increases in fixed costs as some solutions use additional vehicles.

Unfortunately, the solve time utilizing the Gurobi solver grows quickly with instance size as shown in figure 4. Therefore, to solve practically sized instances we attempt two decomposition techniques, namely Lagrangian relaxation and Bender's decomposition, in the following sections.

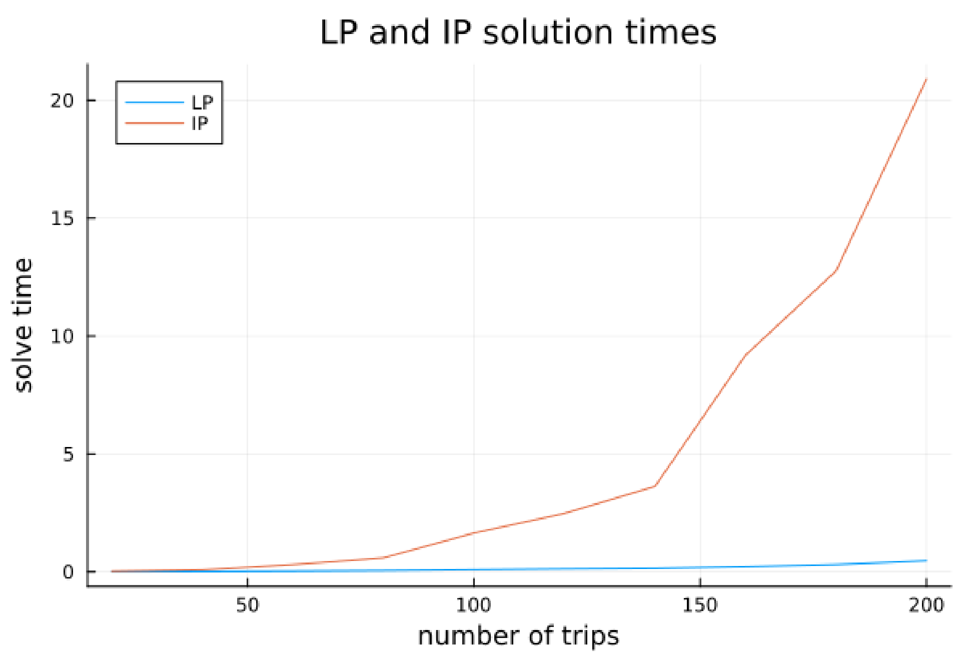


Figure 4: Solution times for solving (8) and its associated integer program using the Gurobi solver for a variety of instance sizes.

2 Lagrangian Relaxation

This problem lends itself to Lagrangian relaxation due to its underlying network-flow structure. If we are able to relax constraint (5), and potentially (4), the remaining Lagrangian dual problem may be easier to solve. The Lagrangian problem is formulated in (9).

$$\begin{aligned}
\min \sum_{i \in T} s_i + \sum_{i \in V}\sum_{j \in V} c_{ij}x_{ij} + \sum_{i \in T} \lambda_i \left[\sum_{j \in T} x_{ji}(s_j + l_j - b_{ji}) - s_i\right] & \\
\text{subject to: } x_{ij} \geq 0 \qquad & \forall i, j \in V \\
s_i \geq 0 \qquad & \forall i \in T \\
\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} \qquad & \forall i \in V \\
\sum_{j \in V} x_{ij} = 1 \qquad & \forall i \in T
\end{aligned} \tag{9}$$

We observe that the objective can be re-written as (for convenience let $\lambda_d = s_d = l_d = b_{di} = b_{id} = 0$, though these do not exist in our problem)

$$Z_{LD}(\boldsymbol{\lambda}) = \sum_{i \in T} s_i(1 - \lambda_i) + \sum_{i \in V}\sum_{j \in V} x_{ij} \underbrace{[c_{ij} + \lambda_j(s_i + l_i - b_{ij})]}_{c'_{ij}},$$

which, for fixed $\boldsymbol{s}$, allows for an interpretation of the relaxation as a minimum cost flow problem with arc costs c'_{ij}. Thus, we propose a subgradient method to solve this problem as follows. Note that the structure of this algorithm has been adapted from Kwon (2019) and Fisher (2004).

2.1 Subgradient Algorithm

1. Set $\boldsymbol{\lambda}^{(0)} = \boldsymbol{s}^{(0)} = \mathbf{0}$, and solve the underlying network flow problem for $\boldsymbol{x}^{(0)}$. The solution of which provides a lower bound for the original problem as $\sum_i s_i + \sum_{i,j} c_{ij}x_{ij} \geq \sum_{i,j} c_{ij}x_{ij} = Z_{LD}(\boldsymbol{\lambda}^{(0)})$.

$$\begin{aligned}
\min \sum_{i \in V}\sum_{j \in V} c_{ij}x_{ij} & \\
\text{subject to: } x_{ij} \geq 0 \qquad & \forall i, j \in V \\
\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} \qquad & \forall i \in V \\
\sum_{j \in V} x_{ij} = 1 \qquad & \forall i \in T
\end{aligned}$$

Generate a feasible $\hat{\boldsymbol{s}}^{(0)}$ (obeying (4) and (5)) and obtain the upper bound $\sum_i \hat{s}_i^{(0)} + \sum_{i,j} c_{ij}x_{ij}^{(0)} = Z_U^{(0)}$. Then for $k = 0, 1, \ldots$

2. Let $\overline{Z}_U = \min_r\{Z_U^{(r)}\}$ and check if

$$\frac{\overline{Z}_U - Z_{LD}(\boldsymbol{\lambda}^{(k)})}{\overline{Z}_U} \leq \epsilon \rightarrow \text{STOP}.$$

3. Update the lagrange multipliers as

$$\lambda_i^{(k+1)} = \lambda_i^{(k)} + \alpha^{(k)} \left[\sum_j x_{ji}^{(k)} (s_j^{(k)} + l_j - b_{ji}) - s_i^{(k)} \right], \quad \alpha^{(k)} = \frac{\theta_k(\overline{Z}_U - Z_{LD}(\boldsymbol{\lambda}^{(k)}))}{\sum_i \left[\sum_j x_{ji}^{(k)} (s_j^{(k)} + l_j - b_{ji}) - s_i^{(k)} \right]^2},$$

   and set any negative λ_is to 0 to ensure to we are not penalizing satisfied (5) constraints. The θ_k value in the step-size is recommended to take values between $(0, 2]$ as per the cited resources and is constant throughout iterations. We selected $\theta = 1$.

4. Due to the nonlinearity of the objective and to exploit its network flow-like structure, we will update $\boldsymbol{s}^{(k)}$ to a new candidate solution, $\boldsymbol{s}^{(k+1)}$ prior to solving for $\boldsymbol{x}^{(k+1)}$. Define

$$g(\boldsymbol{s}^{(k)}) := \sum_i s_i^{(k)} + \sum_i \lambda_i^{(k+1)} \left[\sum_j x_{ji}^{(k)} (s_j^{(k)} + l_j - b_{ji}) - s_i^{(k)} \right],$$

   which represents the effect of $\boldsymbol{s}$ on the objective of (9). The gradient of this function is

$$\nabla g(\boldsymbol{s}^{(k)}) = \mathbf{1} - \boldsymbol{\lambda}^{(k+1)} + \sum_i \lambda_i^{(k+1)} \boldsymbol{x}_{\cdot i}^{(k)} = \mathbf{1} + (\boldsymbol{x}^{(k)} - \mathbb{I}) \boldsymbol{\lambda}^{(k+1)}.$$

   As we would like to minimize the objective, we can apply a step of gradient descent to obtain a candidate $\boldsymbol{s}^{(k+1)}$ as

$$\boldsymbol{s}^{(k+1)} = \boldsymbol{s}^{(k)} - \mu \nabla g(\boldsymbol{s}^{(k)}), \quad \mu = \frac{1}{n}.$$

5. With fixed $\boldsymbol{s}^{(k+1)}$ can then solve the network flow problem with objective

$$\min \sum_i \sum_j x_{ij} \underbrace{\left[c_{ij} + \lambda_j^{(k+1)} (s_i^{(k+1)} + l_i - b_{ij}) \right]}_{c'_{ij}}.$$

   Finally, generate a feasible $\hat{\boldsymbol{s}}^{(k+1)}$ and obtain the upper and lower bounds $Z_U^{(k+1)}$ and $Z_{LD}(\boldsymbol{\lambda}^{(k+1)})$.

6. Return to step 2.

2.2 Results and Discussion

We must acknowledge that this method is an incorrect implementation of Lagrangian relaxation. The fault lies within our assumption that the prescribed construction of $Z_{LD}(\boldsymbol{\lambda}^{(k)})$ is a lower bound on the optimal objective value Z^*. This is not true as updating $\boldsymbol{s}^{(k+1)} = \boldsymbol{s}^{(k)} - \mu \nabla g(\boldsymbol{s}^{(k)})$ via gradient descent and solving for $\boldsymbol{x}^{(k+1)}$ does not ensure that $(\boldsymbol{x}^{(k+1)}, \boldsymbol{s}^{(k+1)})$ is optimal for (9) as

$$Z^* = \sum_i s_i^* + \sum_{i,j} c_{ij} x_{ij}^* \geq \sum_{i,j} c_{ij} x_{ij}^* \not\geq \sum_{i,j} c_{ij} x_{ij}, \quad \forall \boldsymbol{x}.$$

Therefore, the solutions produced from this method do not have a metric to measure their optimality gap and are not useful. A potential avenue for improving this method could look to relax the constraints on $\boldsymbol{s}$ and $\boldsymbol{\phi}$ from (8) which would allow for direct optimization of the Lagrangian dual problem. This would result in many Lagrange multipliers and a column generation approach may be suitable.

3 Bender's Decomposition

Bender's decomposition is a technique that is well-suited to problems with two decision stages, where the first produces integer variables and the second continuous. Furthermore, this technique is able to handle some nonlinearity in the second-stage subproblem. Note that our problem fits this description as our arc decisions, $\boldsymbol{x}$, are integral and our delay variables, $\boldsymbol{s}$, are continuous.

We return to our original problem formulation, shown below, and highlight our second-stage problem in red.

$$\begin{aligned}
\min \; & \sum_{i \in T} s_i + \sum_{i,j \in V} c_{ij} x_{ij} & \\
\text{s.t. } & x_{ij} \in \mathbb{Z}_+ & \forall i, j \in V \\
& s_i \geq 0 & \forall i \in T \\
& s_i \geq \sum_{j \in T} x_{ji}(s_j + l_j - b_{ji}) & \forall i \in T \\
& \sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} & \forall i \in V \\
& \sum_{j \in V} x_{ij} = 1 & \forall i \in T
\end{aligned}$$

We define the problem in red as $Q(\boldsymbol{x})$, and we may re-write as

$$\begin{aligned}
\min \; & Q(\boldsymbol{x}) + \sum_{i,j \in V} c_{ij} x_{ij} & \\
\text{s.t. } & x_{ij} \in \mathbb{Z}_+ & \forall i, j \in V \\
& \sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} & \forall i \in V \\
& \sum_{j \in V} x_{ij} = 1 & \forall i \in T.
\end{aligned}
\qquad
\begin{aligned}
Q(\boldsymbol{x}) := \min \; & \sum_{i \in T} s_i & \\
\text{s.t. } & s_i \geq 0 & \forall i \in T \\
& s_i - \sum_{j \in T} x_{ji} s_j \geq \sum_{j \in T} x_{ji}(l_j - b_{ji}) & \forall i \in T
\end{aligned}$$

Letting $q \geq Q(\boldsymbol{x})$, and p_i be the dual variable associated with constraints in our second-stage subproblem, we have

$$
\begin{aligned}
&\min q + \sum_{i,j \in V} c_{ij} x_{ij} && && Q(\boldsymbol{x}) = \max \sum_{i \in T} p_i \sum_{j \in T} x_{ji}(l_j - b_{ji}) && \\
&\text{s.t. } x_{ij} \in \mathbb{Z}_+ && \forall i, j \in V && \text{s.t. } p_i \geq 0 && \forall i \in T \\
&\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} && \forall i \in V && p_i - \sum_{j \in T} x_{ij} p_j \leq 1 && \forall i \in T \\
&\sum_{j \in V} x_{ij} = 1 && \forall i \in T && && \\
&\sum_{i,j \in T} x_{ji}(l_j - b_{ji}) r_i \leq 0 && \boldsymbol{r} \in \mathcal{R} && && \\
&\sum_{i,j \in T} x_{ji}(l_j - b_{ji}) v_i \leq q && \boldsymbol{v} \in \mathcal{V}, && &&
\end{aligned}
$$

where, $\mathcal{R}$ is the set of extreme rays and $\mathcal{V}$ the set of vertices for $\mathcal{P} = \{\boldsymbol{p} \in \mathbb{R}^n : (\mathbb{I} - \boldsymbol{x})\boldsymbol{p} \leq 1, \boldsymbol{p} \geq \boldsymbol{0}\}$. We note that $\mathcal{R} = \emptyset$ as the constraints require $p_i \geq 0$ and either $p_i \leq 1$ or $p_i - p_j \leq 1$ where (i, j) is an arc chosen in the candidate solution. This ensures that $0 \leq p_i \leq n$ which could only occur if all trips were operated by one vehicle.

To solve this problem, we obtain a candidate solution to the first-stage problem $\overline{\boldsymbol{x}}$, and then solve $Q(\overline{\boldsymbol{x}})$. There are two outcomes.

1. $Q(\overline{\boldsymbol{x}}) > q$ and is finite. Then we have found a violated constraint in our first-stage problem. We add the new optimality cut and re-solve the first-stage problem.

2. $Q(\overline{\boldsymbol{x}}) \leq q$, then we have found an optimal solution and can terminate.

We may then determine the appropriate propagated delay variables, $\boldsymbol{s}$, and we are done.

3.1 Solving the Subproblem

The second-stage subproblem may be solved itself as an LP, and we observe that the constraint matrix is totally unimodular. This can be seen as each row of the constraint matrix sums to either ± 1 or 0. We can partition the rows to satisfy the theorem of Ghouila-Houri thus proving it is TU, and therefore all solutions will be integral. Moreover, the optimal solution can be constructed algorithmically to improve solution times. The procedure for generating a solution is as follows.

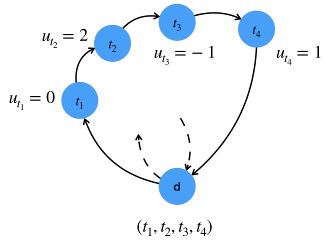
1. Calculate $u_i = \sum_j \overline{x}_{ji}(l_j - b_{ji})$ which represents the 'profit' associated with each p_i. Initialize all $p_i = 0$.

2. Generate the vehicle schedules associated with the candidate solution $\overline{\boldsymbol{x}}$. These will be paths in G of the form $(t_1, \ldots, t_k)$ (which would be cycles if we included the depot).

3. For each vehicle schedule, create a vector of profits $(u_{t_1}, \ldots, u_{t_k})$. Determine the contiguous sub-vector(s) which maximizes $(r+1, \ldots, 1)^T (u_{t_i}, \ldots, u_{t_{i+r}}) + \cdots + (q+1, \ldots, 1)^T (u_{t_j}, \ldots, u_{t_{j+q}})$ and is greater than 0.

4. Set $p_{t_i} = r + 1$, $p_{t_{i+1}} = r$, $\ldots$, $p_{t_{i+r}} = 1$ if such sub-vector(s) exists.

In doing so, we ensure that all $p_i \geq 0$ and that $p_i - p_j \leq 1$ where arc (i, j) is in the candidate solution. In figure 5, we can see how this process works for a single vehicle schedule.



(a) a path (excluding d) in G

u_{t_1}	u_{t_2}	u_{t_3}	u_{t_4}	
0	**2**	**-1**	**1**	**Total**
1				0
	1			2
	2	1		3
	3	2	1	5
		2	1	-1
			1	1
	2	1	1	4

$p_{t_1} = 0\ p_{t_2} = 3\ p_{t_3} = 2\ p_{t_4} = 1$

(b) select sub-vector calculations with the optimal highlighted in red

Figure 5: Algorithmic example of solving the second-stage subproblem for one vehicle schedule.

This subproblem and its solution reveals insights into the structure of the problem and the instance. The value of p_i indicates the number of times trip is delay is propagated forward in the current schedule before it is absorbed by buffer times. The sequential p_i values indicate paths in G that produce propagated delay.

3.2 Stochastic Delays

Another benefit of utilizing Bender's decomposition is that we are able to solve the problem under uncertain delays. While the previous discourse has been focused on the case where we have some deterministic value for delay l_i of each trip, we are able to expand on this analysis.

Consider the set $\mathcal{S}$ of possible delay scenarios which includes every combination of conceivable delays across all trips in the network. We define $h_{ij} = l_i - b_{ij}$ which represents the *slack time* between trips i and j. In practice, we may set a range for each trip such that $h_{ij} \in [h_{ij}^L, h_{ij}^U]$, where $h_{ij}^L, h_{ij}^U \in \mathbb{R}$ are lower and upper bounds on the possible *slack time* between trips i and j. We can then sample these values at a specified interval to generate delay scenarios.

We can separate the second-stage problem by scenario k and solve each in parallel as follows (note we omit the feasibility cuts as we established there are no extreme rays),

$$
\begin{aligned}
\min \sum_{k\in\mathcal{S}} q_k + &\sum_{i,j\in V} c_{ij}x_{ij} && && q_k \geq \max \sum_{i\in T} p_i \sum_{j\in T} x_{ji}h^k_{ji} \\
\text{s.t. } x_{ij} &\in \mathbb{Z}_+ && \forall i,j \in V && \text{s.t. } p_i \geq 0 && \forall i \in T \\
\sum_{j\in V} x_{ij} &= \sum_{j\in V} x_{ji} && \forall i \in V && p_i - \sum_{j\in T} x_{ij}p_j \leq 1 && \forall i \in T \\
\sum_{j\in V} x_{ij} &= 1 && \forall i \in T \\
\sum_{i,j\in T} x_{ji}(l_j - b_{ji})v_i &\leq q_k && \boldsymbol{v} \in \mathcal{V}^k \ \forall k \in \mathcal{S}.
\end{aligned}
$$

3.3 Results and Discussion

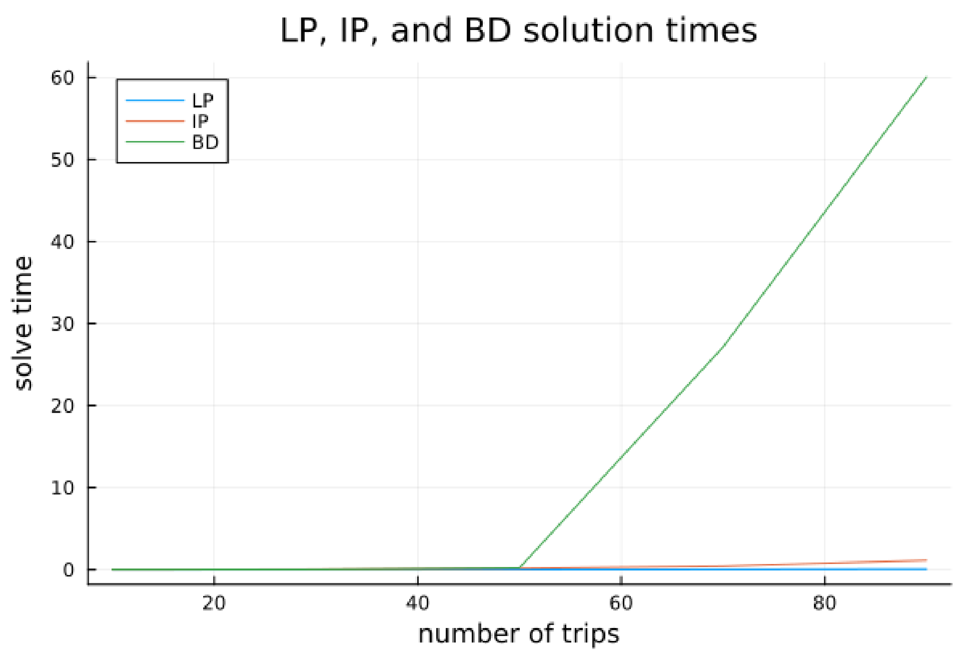


Figure 6: Solution times for solving (8) and its associated integer program using the Gurobi solver and the Bender's decomposition model for a variety of instance sizes.

While we did not perform an assessment of stochastic delays, we show how the implementation of Bender's decomposition compares against the default Gurobi optimization of (8). In figure 6, we observe that the solution time for Bender's decomposition grows quickly and reached our manual time limit of 60 seconds with instances of 90 trips or greater. Whereas, the generic Gurobi optimization of the IP maintains very low solution times.

In table 1, we observe that the Bender's decomposition implementation also lags behind the Gurobi implementation in terms of finding a good solution within a 10 second time limit.

No. trips	Gurobi	Benders
100	0%	16%
200	3%	47%
300	53%	77%
400	71%	85%

Table 1: Optimality gap obtained with a 10 second time limit on both the default Gurobi solution and the Benders decomposition solution.

4 Conclusion

In conclusion, we proposed a mixed integer linear program in (8) that successfully optimizes for minimum network cost and propagated delays in a single depot transit system. We demonstrated how this bi-objective problem may be tuned to produce solutions prioritizing either cost or delay depending on the transit system priorities. The big-M constraints used in this research may be tightened as discussed in the introduction to improve the problem formulation.

We demonstrate two solution approaches to decompose this problem in Lagrangian relaxation and Bender's decomposition. While the former did not produce useful results, further work may yield a correct implementation of LR. The latter correctly found the optimal solution to our problem. Moreover, we implemented an algorithmic solution method for the second-stage subproblem to improve solution times. Nevertheless, Bender's decomposition lagged behind the default Gurobi optimization of our problem in terms of solution speed for finding good and optimal solutions.

Further work may look to enhance the approaches discussed in this research. For example, a greedy solution may be constructed from the base problem instance to warm-start the Bender's model which may improve solution times. Heuristic methods may be explored to generate feasible solutions accompanied by destroy and reconstruct algorithms to find optimal or good solutions in shorter time frames. Additionally, rounding of the LP solution may be investigated to obtain good solutions in much shorter time frames as demonstrated in figure 4.

References

Bertossi, A. A., Carraresi, P., & Gallo, G. (1987). On some matching problems arising in vehicle scheduling models. *Networks*, *17*(3), 271–281.

Bunte, S., & Kliewer, N. (2009). An overview on vehicle scheduling models. *Public Transport*, *1*(4), 299–317.

Daduna, J. R., & Pinto Paixão, J. M. (1995). Vehicle scheduling for public mass transit—an overview. *Computer-Aided Transit Scheduling: Proceedings of the Sixth International Workshop on Computer-Aided Scheduling of Public Transport*, 76–90.

Fisher, M. L. (2004). The lagrangian relaxation method for solving integer programming problems. *Management science*, *50*(12_supplement), 1861–1871.

Kwon, C. (2019). *Julia programming for operations research*. Changhyun Kwon.

Lenstra, J. K., & Kan, A. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, *11*(2), 221–227.