Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrie en Operationele Research

Robust evacuation planning with multiple objectives

Boris Lucas Nijhoff (594930)

Supervisor:	Mette Wagenvoort
Second assessor:	Dr. Twan Dollevoet
Date final version:	27th June 2024

Abstract

This thesis considers a transit-based evacuation problem in a network, deciding on pickup locations and allocating buses. The model is formulated in a robust way by using an uncertainty set with multiple possible scenarios, which an evacuation plan must be feasible for. The formulation of this problem is adopted from Kulshrestha, Lou and Yin (2014), and the results are tested on their adaptation of the standard Sioux Falls network.

The replicability of their results is assessed in this thesis. Moreover, another way of reducing the model size while still solving to optimality is introduced. An extension of the model from Kulshrestha et al. (2014) is also introduced, with the three objectives being total evacuation time, maximum evacuation time and number of buses used.

The results found by Kulshrestha et al. (2014) are not entirely replicable, but similar results are obtained in this thesis. From a runtime analysis, a motivation is found for using other models than the full model: their method and the method introduced in this thesis reduce the model size and runtime. Their method is also found to be robust to adding more possible evacuation scenarios, as long as the model stays feasible.

On the model with three objectives, a simulation of the evacuation is carried out, where the order in which the objectives are optimised is varied. The latest arrival time objective minimises the distribution of average arrival times from this simulation the most, while the total evacuation time objective minimises the distribution of the latest arrival times.

1 Introduction

With growing climate change, there is an increased risk for more extreme weather situations requiring evacuation. This also means more and more places in the world risk some sort of emergency due to extreme weather. Therefore, planning for mass evacuations should be on the mind for an increasing number of governments, local and national. This is one reason why the number of studies in the field of transit-based evacuation has seen an increasing trend over the last decade (Khalili, Mojtahedi, Steinmetz-Weiss & Sanderson, 2024). This literature considers different kind of problems related to emergency situations (see Section 1.1), as some emergencies or regions require specific planning.

The robust transit pickup location and bus allocation model (RTPL) from Kulshrestha et al. (2014) is used in this thesis. This problem consists of choosing some locations as pickup points and allocating sufficient buses to transport people from pickup points to shelters, all while minimising the total evacuation time. The robustness of the model comes from the fact that an evacuation planning must be sufficient for different scenarios. Section 1.2 gives a more detailed problem description.

It is assessed whether the results of Kulshrestha et al. (2014) are replicable. Moreover, motivation is given for why solving the RTPL model directly and exactly is not a possibility. Lastly, the model is extended to have multiple objective functions to optimise. This model is used to simulate an evacuation and obtain an empirical distribution of arrival times at shelters. These times are considered of particular interest because the average arrival time is an indication for the average amount of stress endured during the evacuation process, and the latest arrival time for the maximum amount of stress endured of all evacuated people. Decision makers planning for evacuation are interested in keeping the total impact of an emergency as low as possible on the population, as well as the maximum impact on any single person of the population. After keeping the total and maximum evacuation time as low as possible, decision makers might also be interested in minimising costs by minimising the number of buses used. These three objectives are ordered in all possible ways to see the effect the ordering has on the distribution of arrival times. Minimising the latest possible arrival time first is found to lower the average arrival times most, while minimising the total evacuation time first is found to lower the actual latest arrival time.

The study by Kulshrestha et al. (2014) is concluded to not be entirely replicable. This is because the data used is not fully known. The cutting plane scheme from their study was found to decrease the runtime, thereby motivating its use. An equivalent model of the RTPL is also introduced and is also found to decrease the runtime.

The rest of this thesis is structured as follows. The rest of Section 1 give a review of related literature in Section 1.1 and a detailed description of the problem in Section 1.2. Section 2 provides all the models and solution methods for the problem. Descriptions of how these results are analysed are then given in Section 3. Section 4 contains this analysis, and Section 5 concludes the thesis.

1.1 Literature Review

The field of transit-based evacuation has seen a rise in publications over the last few years (Khalili et al., 2024). In these publications, various sorts of evacuation and emergency types are considered, as well as different types of assumptions and priorities. For example, evacuations due to wildfires (Shahparvari, Abbasi, Chhetri & Abareshi, 2019) or due to floods (Nadeem, Uduman & Dar, 2019).

The literature has multiple ways of handling uncertain demand. A robust method like Kulshrestha et al. (2014), Fereiduni and Shahanaghi (2017), Ahmadi, Tavakkoli-Moghaddam, Baboli and Najafi (2022) is one such solution. More complicated methods can also be used for this, like Zhang and len Chang (2014) assuming non-instantaneous stochastic demand.

Different problems are also considered in the literature. This thesis considers bus allocation and pickup location decisions, but shelter location decisions can be added to that, e.g. Nadeem et al. (2019), Goerigk, Grün and Heßler (2014). Routes can also be made more complicated by not allocating a bus to a single pickup point, e.g. Gao, Nayeem and Hezam (2019), Adhikari, Pyakurel and Dhamala (2020), Adhikari and Dhamala (2020). These route construction problems embedded in a location decision problem is generally harder and are thus mostly solved using heuristics, e.g. Nadeem et al. (2019), Gao et al. (2019), Shahparvari et al. (2019), Zheng (2014), Kyriakakis, Marinaki, Matsatsinis and Marinakis (2022).

This thesis considers total evacuation time as an objective first, and later on the maximum evacuation time and the number of buses used as well. Multiple studies also use multiple objectives, e.g. Gao et al. (2019), Nabavi, Vahdani, Nadjafi and Adibi (2022). Total or maximum evacuation time is frequently used as an objective, e.g. Ahmadi et al. (2022), Alam, Habib and Husk (2022), Kyriakakis et al. (2022), Nabavi et al. (2022), Goerigk et al. (2014), Adhikari and Dhamala (2020), Adhikari et al. (2020), Gao et al. (2019). Total evacuation time is chosen as an objective frequently, because it is a proxy for average evacuation time. The average, however, is not possible to capture in a linear objective because of demand uncertainty, such that it is unknown how much to divide the total time by. The number of buses used is also sometimes used as a secondary objective (Gao et al., 2019). For specific problems, objectives related to this are also used, e.g. total risk exposure during wildfire evacuation (Shahparvari et al., 2019) or a fair distribution of help over the affected area (Ahmadi et al., 2022).

The way this paper uses hierarchical objectives is very similar to Gao et al. (2019), which uses the three objectives of walking time, total transit-based evacuation time and the number of buses used. Walking time is of less interest for this problem because walking time is limited to such an extent that all people are at their pickup location before the first bus arrives. Therefore, maximum evacuation time was chosen instead of this, because maximum and total evacuation time are the two most used objectives. This thesis therefore looks which of the two provides the better results, by looking in which order they are minimised. This look at the objectives is what this thesis provides to the literature.

1.2 Problem Description

The problem the paper is based is the same one as the one investigated by Kulshrestha et al. (2014). A diagram of this is shown in Figure 1.

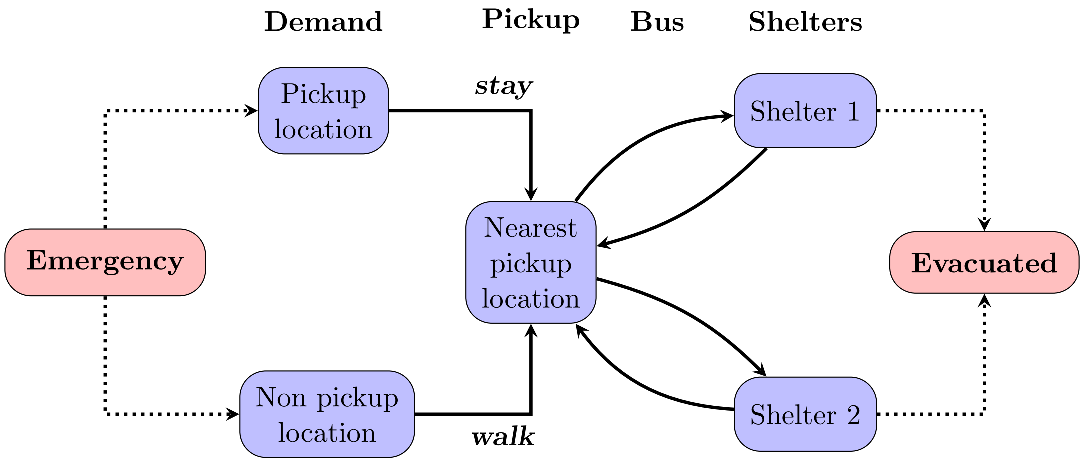

Figure 1. Flowchart of the problem

Figure 1 shows in particular the flow of people being evacuated. Once an emergency is called (on the left) people are assumed to be at some location from a given set of demand locations. The amount of people at each location is denoted as demand and is not exactly known. These people then go to the nearest of the other locations which is chosen (in the model) to be a pickup point, which means some people do not walk at all. There is some pre-determined maximum time that people can spend walking from their location to the nearest pickup location, which means every location where people are evacuated from has to have a pickup location within that maximum walking time. From this pickup location, buses transport people to shelters. A bus is assigned to a specific pickup location and can do multiple trips, to different shelters, while keeping the total time a bus is on the road under some maximum time and keeping the number of people at each shelter below its respective capacity. Once people arrive at a shelter they are considered safe and thus finished with the evacuation process. An evacuation is successful once all people have arrived at a shelter.

2 Solution Methodology

The first part of this methodology consists of the replicated methods from Kulshrestha et al. (2014). The notation and the robust transit pickup location and bus allocation model (RTPL) that follows is mostly taken from this.

The problem considers a network with a set of demand locations I and a set of shelters J. Arcs between demand locations and from demand locations to shelters are given, so that the network can be written as a graph $G = (I \cup J, I \times (I \cup J))$. Different travel times are considered in G: T_{ij} is the round-trip travel time for a bus from demand location $i \in I$ to shelter $j \in J$, and C_{ip} is the one-way walking time from demand location $i \in I$ to demand location $p \in I$. The maximum time any bus can travel is T_{max} and ω is the maximum walking time to a pickup location from a demand location. The set B denotes the set of buses, which have a capacity of β_b for $b \in B$. The shelter capacity is denoted as K_j for each shelter $j \in J$. The demand at each demand location in I is uncertain and thus each location $i \in I$ is considered to have S^i demand scenarios: $d_1^i, \ldots, d_{S^i}^i$, where d_1^i is the nominal demand.

The uncertainty set of demand possibilities is determined by a parameter $\Gamma \in \{0, \dots, |I|\}$, the degree of pessimism. The degree of pessimism is the maximum number of demand locations that can have a non-nominal demand. The demand uncertainty set is then defined as

$$D = \{d | d^i \in \{d_1^i, \dots, d_{S^i}^i\} \ \forall i \in I, \sum_{i \in I} \mathbf{1}_{\{d^i \neq d_1^i\}} \leq \Gamma\}, \tag{1}$$

where $\mathbf{1}_{\{d^i \neq d_1^i\}} = 1$ if $d^i \neq d_1^i$, 0 if $d^i = d_1^i$, denotes the indicator function. The following decision variables are then considered for the formulation of the problem. The binary variable y_i is 1 if and only if demand location $i \in I$ is a pickup location. The binary variable μ_{ib} is 1 if and only if bus $b \in B$ is assigned to pickup location $i \in I$. The binary variable δ_{ip} is 1 if and only if demand location $i \in I$ walks to pickup location $p \in I$, which must always be the closest pickup location. X_{bij} is a non-negative integer variable for the number of trips bus $b \in B$ takes from pickup location $i \in I$ to shelter $j \in J$. W_i is the walking time to a pickup location for demand location $i \in I$. Finally, define the sets $I^+(i) = \{p \in I | C_{ip} \leq \omega\}$ as the reachable pickup locations from demand location $i \in I$. It is assumed that C_{ip} is symmetric, such that $C_{ip} = C_{pi}$ for all $i, p \in I$. Thus, $I^+(p)$ also denotes the set of demand locations that can reach pickup point $p \in I$.

With the necessary parts introduced, the RTPL can be stated as:

$$\begin{aligned}
\min \xi = & \sum_{b \in B} \sum_{i \in I} \sum_{j \in J} T_{ij} X_{bij} && && \text{(2a)} \\
\text{s.t.} \ & \sum_{b \in B} \sum_{j \in J} \beta_b X_{bpj} \geq \sum_{i \in I^+(p)} \delta_{ip} d^i && \forall p \in I, d \in D && \text{(2b)} \\
& \sum_{b \in B} \sum_{i \in I} \beta_b X_{bij} \leq K_j && \forall j \in J && \text{(2c)} \\
& \sum_{i \in I} \mu_{ib} = 1 && \forall b \in B && \text{(2d)} \\
& \mu_{ib} \leq y_i && \forall i \in I, b \in B && \text{(2e)} \\
& \sum_{j \in J} T_{ij} X_{bij} \leq T_{max} \mu_{ib} && \forall i \in I, b \in B && \text{(2f)} \\
& W_i = \sum_{p \in I^+(i)} C_{ip} \delta_{ip} && \forall i \in I && \text{(2g)} \\
& W_i \leq C_{ip} y_p + \omega(1 - y_p) && \forall i \in I, p \in I^+(i) && \text{(2h)} \\
& \sum_{p \in I^+(i)} \delta_{ip} = 1 && \forall i \in I && \text{(2i)} \\
& \delta_{ip} \leq y_p && \forall i \in I, p \in I^+(i) && \text{(2j)} \\
& y_i, \delta_{ip}, \mu_{ib} \in \{0, 1\} && \forall i \in I, p \in I^+(i), b \in B && \text{(2k)} \\
& X_{bij} \in \mathbb{Z}_{\geq 0} && \forall b \in B, i \in I, j \in J && \text{(2l)}
\end{aligned}$$

The objective (2a) minimises total evacuation time. Constraints (2b) ensure every pickup point has more total bus capacity leaving from it than demand arriving there, for every possible demand vector. Constraints (2c) then ensure the total bus capacity arriving at each shelter does not exceed its capacity. Constraints (2d) ensure each bus is assigned to one location, while constraints (2e) ensure that this location is a pickup location. The constraints (2f) ensure that if a bus is assigned to a location, the total travel time the bus has from there to all shelters it visits does not exceed the maximum driving time, and if a bus is not assigned to that location there cannot be any trips from that location with that bus. Constraints (2g)–(2j) ensure that all demand from each demand location goes to its nearest pickup location. Constraints (2g) ensure that the walking time is equal to the chosen pickup location. The constraints (2h) ensure that for every pickup location, the walking time does not exceed the walking time to that pickup location, and for every non-pickup location, it becomes inactive because ω is the maximum walking time. Constraints (2i) ensure that each demand locations goes to exactly one location, and the constraints (2j) ensure this location is a pickup location. Constraints (2k) and (2l) are the domains of the variables, where $\mathbb{Z}_{\geq 0}$ denotes the set of non-negative integers.

2.1 Restricted RTPL

The RTPL can have a high number of constraints, by how D is constructed in Equation (1), for which the RTPL has a set of constraints for each element of that set (constraint (2b)). Therefore, additional care needs to be taken for this set of constraints. One solution to this, stated by Kulshrestha et al. (2014), is a cutting plane scheme for determining the elements in D that actually determine the outcome of the model (see Section 2.3). This scheme needs a restricted RTPL for a set of demand vector $\bar{D}$ which is a subset of D. This restricted RTPL (R–RTPL) is stated as follows:

$$\min \xi = \sum_{b\in B}\sum_{i\in I}\sum_{j\in J} T_{ij}X_{bij} \tag{3a}$$

$$\text{s.t.} \sum_{b\in B}\sum_{j\in J}\beta_b X_{bpj} \geq \sum_{i\in I^+(p)} \delta_{ip}d^i \qquad \forall p \in I, d \in \bar{D} \tag{3b}$$

$$(2c)\text{–}(2l) \tag{3c}$$

The R–RTPL is stated as R–RTPL($\bar{D}$) to show the dependence on $\bar{D}$. The R–RTPL is the RTPL where constraint (2b) is switched for (3b), so that the bus capacities only need to be high enough for all vectors in $\bar{D}$.

2.2 Worst-case Demand Model

The cutting plane scheme uses the determining demand vectors of D to solve the RTPL with a lower number of constraints, as said in Section 2.1. For this, Kulshrestha et al. (2014) use the worst-case demand model (WCD), that finds the demand vector with the highest number of people not arriving at a shelter. This model needs an optimal solution to the R–RTPL($\bar{D}$), denoted by $(\hat{y}, \hat{\delta}, \hat{X})$. For ease of notation, the set of chosen pickup points from the R–RTPL solution is defined as $I' = \{p \in I : \hat{y}_p = 1\}$.

The following decision variables are used in the WCD. The non-negative variable e_p is the excessive demand at pickup location $p \in I'$. The binary variable η_p is 1 if and only if there is excessive demand at pickup location $p \in I'$. The variable Δ_p denotes the difference between the demand and bus capacity at pickup location $p \in I'$. The variable d^i denotes the demand value for demand location $i \in I$, which is chosen by the binary decision variables z_s^i. z_s^i is 1 if and only if scenario $1 \leq s \leq S^i$ is chosen for $i \in I$. The WCD can then be stated as:

$$\max E = \sum_{p \in I'} e_p \tag{4a}$$

$$\text{s.t. } \Delta_p = \sum_{i \in I^+(p)} \hat{\delta}_{ip} d^i - \sum_{b \in B} \sum_{j \in J} \beta_b \hat{X}_{bpj} \qquad \forall p \in I' \tag{4b}$$

$$e_p \leq \eta_p \sum_{i \in I^+(p)} \hat{\delta}_{ip} d^i_{max} \qquad \forall p \in I' \tag{4c}$$

$$e_p - \Delta_p \leq (1 - \eta_p) \sum_{b \in B} \sum_{j \in J} \beta_b \hat{X}_{bpj} \qquad \forall p \in I' \tag{4d}$$

$$d^i = \sum_{s=1}^{S^i} z_s^i d_s^i \qquad \forall i \in I \tag{4e}$$

$$\sum_{s=1}^{S^i} z_s^i = 1 \qquad \forall i \in I \tag{4f}$$

$$\sum_{i \in I} \sum_{s=2}^{S^i} z_s^i \leq \Gamma \tag{4g}$$

$$e_p \geq 0 \qquad \forall p \in I' \tag{4h}$$

$$\eta_p \in \{0, 1\} \qquad \forall p \in I' \tag{4i}$$

$$z_s^i \in \{0, 1\} \qquad \forall s = 1, \ldots, S^i, i \in I \tag{4j}$$

The WCD is stated as WCD$(\hat{y}, \hat{\delta}, \hat{X})$ to show the dependence on the R–RTPL$(\bar{D})$ solution. The objective (4a) maximises total excessive demand. The constraints (4b) define the difference between the demand at a pickup point and the total capacity of buses leaving from that pickup point. The sets of constraints (4c) and (4d) make sure e_p is well-defined as the excessive demand at pickup point p, if there is too much demand. If η_p is zero, (4c) says that e_p is also zero, by its domain. If η_p is one, e_p is bounded to Δ_p with (4d). In optimality, the excessive demand will equal this difference Δ_p as the objective is maximised. The constants on the right-hand sides of (4c) and (4d) are big enough constants to make sure the constraints are inactive if η_p is one or zero, respectively, where d^i_{max} is the maximum demand value as defined in Section 2.4. Appendix A.1 proves why these constants are big enough. Constraints (4e)–(4g) ensure the demand vector d from combining $d^i, i \in I$, is contained in D. Finally, (4h)–(4j) are the domains of the decision variables.

If the WCD objective value E equals 0 in an optimal solution, all differences Δ_p are at most zero. As the objective is maximised, this means that all Δ_p are at most zero for every demand vector $d \in D$. The RTPL constraint (2b) is now satisfied for the R–RTPL solution $(\hat{y}, \hat{\delta}, \hat{X})$, because $\Delta_p \leq 0$ is exactly equivalent to this constraint, and for $p \in I \setminus I'$ the constraint is trivially satisfied. Therefore, because the feasible region of the RTPL is contained in that of the R–RTPL (as $\bar{D} \subseteq D$), the R–RTPL solution is also an optimal solution to the RPTL.

2.3 Cutting Plane Scheme

The cutting plane scheme is a way of finding an optimal solution to the RTPL without solving it directly, as the original set D increases combinatorially when the number of demand scenarios S^i or the size of I increases. The procedure has the following steps. First, an initial demand vector d is chosen to solve the R–RTPL on, i.e., $\bar{D} = \{d\}$. This initial demand vector is always the nominal demand vector ($d^i = d_1^i$ for all $i \in I$) as this is the fastest way to obtain a demand vector. The solution to the R–RTPL is then used to solve the WCD. If this yields a positive objective value E, the demand vector from the WCD is added to $\bar{D}$, and the procedure starting with the R–RTPL is done again until E is zero. When this is the case, the R–RTPL solution used for the WCD is also a solution to the RTPL (as seen in Section 2.2). The cutting plane scheme is graphically represented in Figure 2. Note that the check for the procedure is $E \leq \varepsilon$ instead of $E = 0$ to account for inaccuracy in numerical solvers.

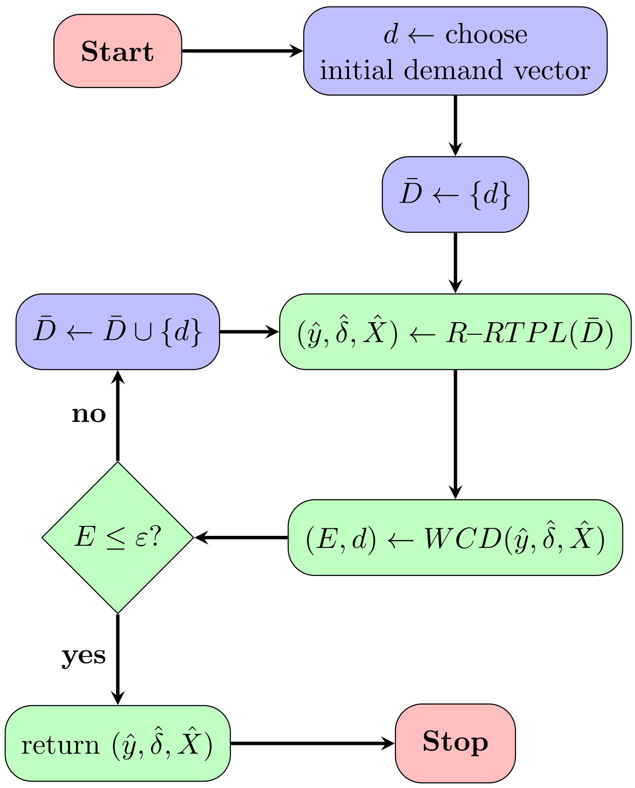

Figure 2. Flowchart of the cutting plane scheme

2.4 Equivalent RTPL

As stated in Section 2.1 the set of constraints Equation (2b) preferably have a smaller size than in the RTPL. One solution to this is the cutting plane scheme from Section 2.3. Another one is by constructing an equivalent uncertainty set D', which is done here. The set is called equivalent because the demand vectors that determine the solution are in D'. This is proven later on. For the construction of this set, define $d_{max}^i = \max_{1 \leq s \leq S^i} d_s^i$ as the maximum demand possible from a demand location, for all $i \in I$. It is assumed that $d_{max}^i \neq d_1^i$ for all $i \in I$, to make the notation and proof easier to read. Then define the equivalent uncertainty set as:

$$D' = \{d | d^i \in \{d_1^i, d_{max}^i\} \ \forall i \in I, \sum_{i \in I} \mathbf{1}_{\{d^i = d_{max}^i\}} = \Gamma\}. \tag{5}$$

The equivalent uncertainty set D' is a subset of D because $\{d_1^i, d_{max}^i\} \subseteq \{d_1^i, \ldots, d_{S^i}^i\}$ for all $i \in I$ and $\mathbf{1}_{\{d^i = d_{max}^i\}} = \mathbf{1}_{\{d^i \neq d_1^i\}}$ when $d^i \in \{d_1^i, d_{max}^i\}$ for all $i \in I$. Note that in contrast with D, D' considers only the most positive deviations from the nominal demand, and only the most deviations possible for a given degree of pessimism Γ. Defining the RTPL with D' instead of D does not change the model, because for every demand vector in D there exists a demand vector in D' having at least the same demand value for every demand location. This is formalised in Proposition 1.

Proposition 1 (Equivalent RTPL)**.** *The R–RTPL(D') model in* (3) *is equivalent to the RTPL model in* (2)*, where D' is defined in Equation (5).*

Proof. See Appendix A.2. □

The special case of the R–RTPL from Proposition 1 is further denoted as the equivalent RTPL (ERTPL). The ERTPL significantly reduces the potential size of the uncertainty set. Trivial calculations show that $|D| = \sum_{A \subseteq I : |A| \leq \Gamma} \left[\prod_{i \in A}(S^i - 1)\right]$, which can be simplified to $|D| = \sum_{k=0}^{\Gamma} \binom{|I|}{k}(S-1)^k$ if $S^i = S$ for all $i \in I$, whereas the size of the equivalent set is $|D'| = \binom{|I|}{\Gamma}$.

2.5 Tri-objective Model

To extend on the models presented thus far, a model with three objectives in hierarchical fashion is presented as the tri-objective RTPL (TRTPL). First the three objective functions are presented. The first objective function sums the total evacuation time, like ξ from the RTPL, and is thus defined as

$$\xi_1 = \sum_{b \in B} \sum_{i \in I} \sum_{j \in J} T_{ij} X_{bij}.$$

The second objective is the latest possible arrival time. For this, the decision variable T is defined to denote exactly this. The second objective is thus defined as

$$\xi_2 = T.$$

The third objective is the number of buses used. This can be obtained by summing the binary decision variables μ_{ib}, to determine if a bus is assigned to any demand location. The third objective is thus defined as

$$\xi_3 = \sum_{b \in B} \sum_{i \in I} \mu_{ib}.$$

As the objectives are hierarchical, they have a certain order in which they are optimised. This is shown in the notation of the TRTPL. The TRTPL$_{a\text{-}b\text{-}c}$ optimises objective a (ξ_a), then objective b (ξ_b) and finally objective c (ξ_c), where a, b and c are distinct numbers from $\{1, 2, 3\}$. Now, the TRTPL$_{a\text{-}b\text{-}c}$ is given by

$$\min \xi_a \tag{6a}$$

$$\min \xi_b \tag{6b}$$

$$\min \xi_c \tag{6c}$$

$$\text{s.t.} \sum_{i \in I} \mu_{ib} \leq 1 \qquad \forall b \in B \tag{6d}$$

$$T \geq \sum_{j \in J} T_{ij} X_{bij} \qquad \forall b \in B, i \in I \tag{6e}$$

$$(2b), (2c), (2e)–(2l) \tag{6f}$$

The objectives are optimised from top to bottom, as the notation was defined above. This means a solution is first obtained that minimises objective a. Then a solution is obtained that minimises objective b, while keeping the value of objective a at the earlier found minimum. This is also done for the last objective c, keeping objectives a and b at their minimum. The set of constraints in (6d) is adjusted into an inequality from the RTPL constraint (2d) to allow for a bus not being assigned to any demand location. The set of constraints in (6e) makes sure T is well-defined as the latest possible arrival time to a shelter, together with the objective ξ_2 being minimised. Finally, the other constraints (2b), (2c), (2e)–(2l) are all constraints from the RTPL, where (2d) is removed because of (6d).

As was done with the RTPL, the TRTPL can be represented in restricted form to allow for the cutting plane scheme. The restricted TRTPL (R–TRTPL) is defined for a subset $\bar{D}$ of D and is the model in (6) with constraint (2b) switched for constraint (3b). The cutting plane scheme with the R-TRTPL is then the algorithm from Section 2.3 where the R–RTPL is switched for the R–TRTPL.

3 Results Methodology

The methods described in Section 2 find solutions to the problem at hand. Some things from these solutions are analysed in Section 4. This section describes these analyses theoretically.

3.1 Replicated Results

To assess the replicability of the results from Kulshrestha et al. (2014) the same analysis of the solutions will be done. For this, a simulation is done by generating $10,000$ random demand vectors, where each demand scenario has equal probability for each demand location. For each generated demand vector it is checked if all demand can be picked up from all pickup points for the given solution. This can be mathematically written as d being the random demand vector and checking if for each $p \in I$ such that $y_p = 1$, $\Delta_p \leq 0$, where this is defined as in constraint (4b) of the WCD. The probability of meeting demand is then estimated by the percentage of times that all demand can be picked up.

3.2 Runtime Analysis

To motivate the use of the cutting plane scheme and the ERTPL, a runtime analysis is done. The runtime analysis is split into two parts. First, the RTPL, the cutting plane scheme, the

ERTPL, and the cutting plane scheme with the R–TRTPL$_{1-2-3}$ have their runtimes recorded for each value of the degree of pessimism Γ.

The second part of the runtime analysis consists of adding additional demand scenarios to the model and determining the average runtime of the cutting plane scheme with these additional scenarios. The additional scenarios are constructed to be non-determining, where the nominal and maximum demand values are the only determining demand scenarios by the ERTPL (see Section 2.4). The cutting plane scheme should thus be robust to these additional scenarios. Define d^i_{max} and d^i_{min} as the maximum and minimum demand values for each $i \in I$. Let n^{max} be the maximum number of additional demand scenarios. For each of the n^{max} additional demand scenarios, a demand scenario is randomly drawn from the continuous uniform distribution $U(d^i_{min}, d^i_{max})$ for each $i \in I$. This increases each S^i by one, with $d^i_{S^i}$ being that new demand scenario. The runtime of the cutting plane scheme with this additional demand scenarios is recorded multiple times for each degree of pessimism Γ.

3.3 Simulation

In this part an actual instance of the problem is simulated, where on top of the simulation from Section 3.1, the movement of people and buses is considered. For each replication a demand vectors is generated, each demand scenario having equal chance. The replication then consists of simulating the movement of people and buses.

To start, all people start at their corresponding demand location. They then walk to the nearest pickup location. Once they start walking, buses depart from the nearest shelter to their respective pickup locations. Once a bus reaches a pickup location it picks up as much demand as possible, constrained by the bus capacity and the number of people at the pickup location at that time. This is because the bus evacuation time must still stay below the maximum time T_{max}, which is not the case if a bus waits for enough people. When multiple buses arrive at a pickup location at the same time, the buses are filled up in order of the shelter they are going to, with the nearest being first. When multiple buses also go to the same shelter at the same time, they are filled arbitrarily as this does not affect the outcome. A bus is assumed to visit shelters in increasing order of travel time, if it visits multiple shelters.

The following is done to ensure the solution being used for the above has a low fail percentage, where fail percentage is the probability of *not* meeting demand with the simulation from Section 3.1. Let α be the maximum fail percentage of a solution. A solution is found by starting with degree of pessimism $\Gamma = 0$ and increasing Γ by 1, until the fail percentage of a solution is below α. This is done for each possible ordering of TRTPL$_{a\text{-}b\text{-}c}$, solved using the cutting plane scheme with the R–TRTPL$_{a\text{-}b\text{-}c}$. For each such solution of the TRPTL, the replication is performed multiple times to obtain empirical distributions of the average and latest arrival times.

4 Results

This section provides the analysis of the results, as described in Section 3, on the data which will be described in Section 4.1. The implementation was done on a 3.2 GHz Apple computer with 8 cores and 16 GB of RAM using Java and the Gurobi Optimizer for solving all linear models (see Appendix C).

4.1 Data

The data used in the rest of this section to perform the analysis of the solutions as described in Section 3 is the same as used by Kulshrestha et al. (2014). They use the Sioux Falls network, with some points as demand locations, some as shelters, and some as neither. Each demand location has a nominal, low and high demand and the shelters have different capacities. There are 10 buses all with capacity 30, for which the maximum driving time T_{max} equals 180 minutes. The maximum walking time ω is five minutes. The walking and bus driving times C_{ip} and T_{ij} are not explicitly mentioned by Kulshrestha et al. (2014). The free flow travel times from the Sioux Falls network were found to be values comparable to those used by Kulshrestha et al. (2014) after some scaling (see Appendix B for more detail), where the shortest paths are chosen for each node pair.

4.2 Replicated Results

This section provides the results as described in Section 3.1. The bus plans are contained in Tables 1 to 3. These tables are sorted and contain some empty rows to allow for a clear comparison. The three plans from this thesis compared to the ones from Kulshrestha et al. (2014) are mostly similar. Some trips are distributed in different ways across the buses, but this difference is arbitrary. For each degree of pessimism, there is also some actual difference in terms of pickup points and/or number of trips from a given pickup point. For the nominal plan ($\Gamma = 0$), two trips from node 3 to node 13 in the plan from Kulshrestha et al. (2014) are taken from node 12 in this thesis' plan. With the robust plan ($\Gamma = 3$) from this thesis, node 17 is not a pickup point, saving one bus trip from the total of nodes 17 and 18. Moreover, the shelters the different pickup points go to, changes from their plan to the plan from this thesis. This could be due to reaching the limit on shelter capacities from the first difference in the two plans. The worst-case plan ($\Gamma = 15$) from this thesis saves one bus trip by letting node 12 go to node 3, instead of node 12 being a pickup point itself.

Thus, the plans in this thesis have some small differences compared to Kulshrestha et al. (2014). The reason for these differences is clear. The travel times used in this thesis cannot be verified to be the same as the ones used by Kulshrestha et al. (2014), see Section 4.1. The fact that the plans are mostly similar shows that the data used is similar. Still, in each plan some trips are moved from one node to another, showing that these travel times are lower (or others higher) than the ones used by Kulshrestha et al. (2014).

The found optimal objective value for each degree of pessimism in minutes is depicted in Figure 3. The respective objective values from Kulshrestha et al. (2014) are subtracted from these in Figure 4. From Figure 3 one can see that in general the higher the degree of pessimism, the higher the objective value. This is because more possible demand vectors need to be considered. It can also be seen that for this particular dataset a degree of pessimism of five already has the highest objective value. This means higher degrees of pessimism do not make an impact on the optimal solution.

Table 1. Nominal plans ($\Gamma = 0$) compared.

This thesis				Kulshrestha et al. (2014)			
Bus	Trips	Pickup	Shelter	Bus	Trips	Pickup	Shelter
3	5	Node 3	Node 13	2	7	Node 3	Node 13
4	4	Node 6	Node 20	5	4	Node 6	Node 20
1	1	Node 6	Node 20	4	2	Node 6	Node 20
6	1	Node 6	Node 20				
2	5	Node 10	Node 22	3	5	Node 10	Node 22
9	2	Node 12	Node 13				
7	5	Node 18	Node 20	1	5	Node 18	Node 20

Table 2. Robust plans ($\Gamma = 3$) compared.

This thesis				Kulshrestha et al. (2014)			
Bus	Trips	Pickup	Shelter	Bus	Trips	Pickup	Shelter
2	5	Node 3	Node 13	4	7	Node 3	Node 13
1	3	Node 3	Node 13	9	1	Node 3	Node 13
1	2	Node 3	Node 21	9	2	Node 3	Node 21
6	4	Node 6	Node 20	5	4	Node 6	Node 20
8	4	Node 6	Node 20	1	2	Node 6	Node 20
5	1	Node 6	Node 22	8	3	Node 6	Node 21
3	5	Node 10	Node 22	2	1	Node 10	Node 21
9	4	Node 10	Node 22	2	4	Node 10	Node 22
				3	4	Node 10	Node 22
4	3	Node 18	Node 20	6	1	Node 17	Node 20
7	4	Node 18	Node 21	10	1	Node 17	Node 21
				6	2	Node 17	Node 22
				7	4	Node 18	Node 20

Table 3. Worst-case plans ($\Gamma = 15$) compared.

This thesis				Kulshrestha et al. (2014)			
Bus	Trips	Pickup	Shelter	Bus	Trips	Pickup	Shelter
10	5	Node 3	Node 13	7	7	Node 3	Node 13
3	3	Node 3	Node 13	5	1	Node 3	Node 13
3	2	Node 3	Node 21	5	1	Node 3	Node 21
10	1	Node 3	Node 21				
8	4	Node 6	Node 20	6	4	Node 6	Node 20
9	3	Node 6	Node 20	4	3	Node 6	Node 20
5	3	Node 6	Node 21	2	3	Node 6	Node 21
4	3	Node 10	Node 21	10	3	Node 10	Node 21
1	4	Node 10	Node 22	1	5	Node 10	Node 22
4	2	Node 10	Node 22	10	1	Node 10	Node 22
				9	3	Node 12	Node 21
2	4	Node 17	Node 22	8	4	Node 17	Node 22
7	4	Node 18	Node 20	3	4	Node 18	Node 20

Figure 4 shows that the differences in the objective values compared to Kulshrestha et al. (2014) are relatively small, as they are at most 40 in absolute value compared to the objective values of at least 580 minutes. The travel times used for this thesis were based on the three bus plans provided by Kulshrestha et al. (2014) (see Appendix B). For these three degrees of pessimism, the objective value found in this thesis is lower. That is why different bus plans were found for these degrees of pessimism in Tables 1 to 3. For the other degrees of pessimism it cannot be said with certainty why different objective values were found. These others do again show that the travel time values used by this thesis and Kulshrestha et al. (2014) are similar but different.

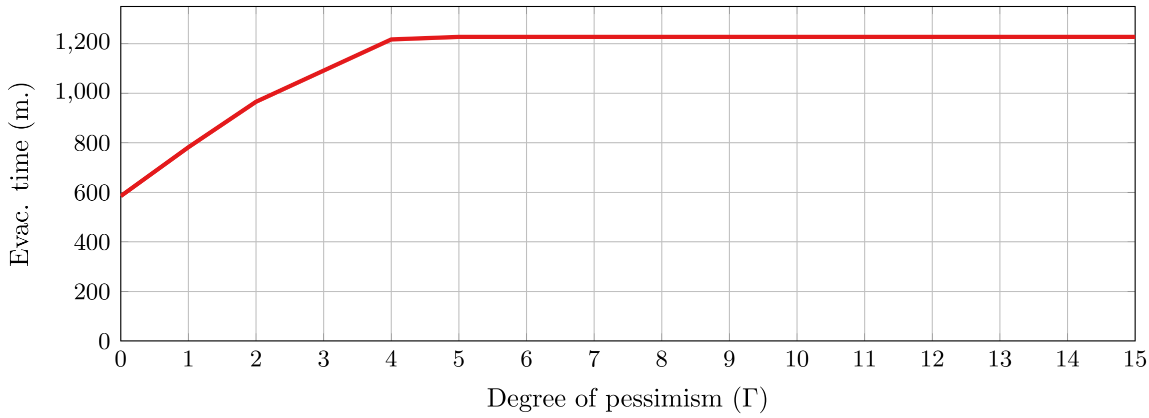

Figure 3. Optimal objective value for each degree of pessimism in minutes.

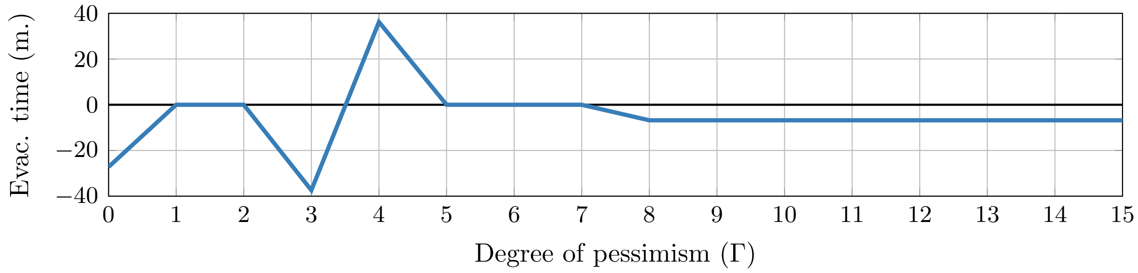

Figure 4. The difference in optimal objective value in minutes for each degree of pessimism compared to Kulshrestha et al. (2014): The objective from this thesis minus the objective reported by Kulshrestha et al. (2014).

The probability of meeting demand, from 10,000 iterations, for each degree of pessimism in % is depicted in Figure 5. The respective probabilities from Kulshrestha et al. (2014) are subtracted from these in Figure 6. From Figure 5 one can see that, just like in Figure 3, a solution does not change starting at a degree of pessimism of five. Before this, the probability increases in a concave manner. From Figure 3 and Figure 5 together, one can also see the cost of increasing the probability of meeting demand, because it also increases the total evacuation time. From Figure 6 it can be concluded that the solutions found are very similar to those from Kulshrestha et al. (2014), as the difference is at most 1% in absolute value. This means that even though the solutions are somewhat different compared to Kulshrestha et al. (2014), the robustness of them remains.

The runtime of the cutting plane scheme for each degree of pessimism is shown in Table 4,

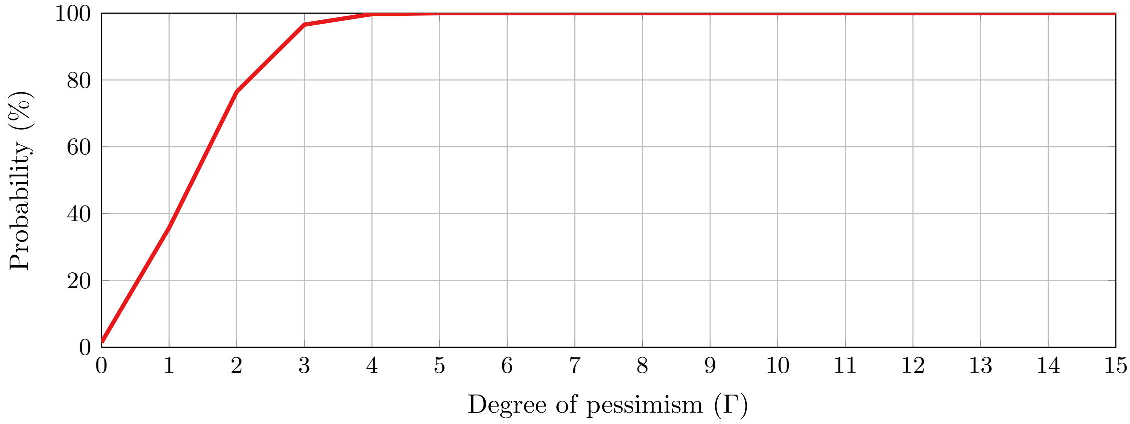

Figure 5. Probability of meeting demand from $10,000$ iterations for the optimal solution for each degree of pessimism in %.

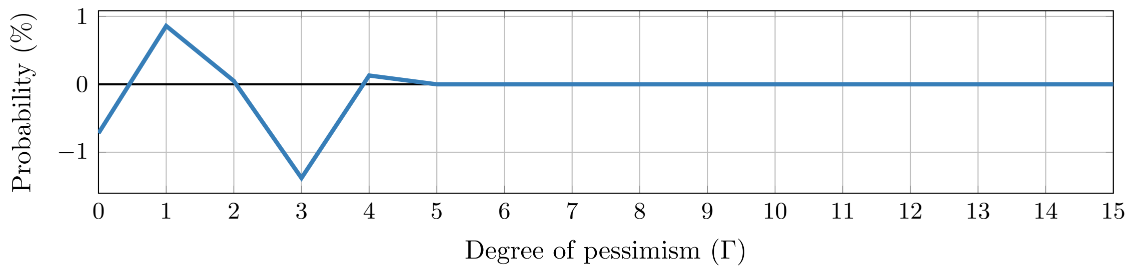

Figure 6. The difference in probability of meeting demand in % for each degree of pessimism compared to Kulshrestha et al. (2014): The probability from this thesis minus the probability reported by Kulshrestha et al. (2014).

where degrees of pessimism until seven are on the left and the others on the right. The columns This and Kul show the runtime in milliseconds of the cutting plane scheme for this thesis and (where available) Kulshrestha et al. (2014), respectively. The fourth and eighth columns contain the number of iterations for the cutting plane scheme, from this thesis.

In Table 4 it can be seen that the runtimes from this thesis and Kulshrestha et al. (2014) differ by a factor of 10 to 50. This can be due to the fact that different hardware and programs are used. The number of iterations seems to correlate loosely with the runtime, where the degree of pessimism of three is an outlier. For the others every iteration takes around ~50 milliseconds or less. Note that the degree of pessimism of fifteen has two iterations while only one demand vector needs to be considered. This is because the cutting plane scheme uses the nominal demand vector as the initial demand vector, so this most pessimistic demand vector is only used after one iteration. The runtimes however show that always choosing the nominal demand vector may be faster, as degree of pessimism zero and fifteen have similar runtimes, even with one more iteration for fifteen.

From this section it can be concluded that the study by Kulshrestha et al. (2014) is not fully replicable. Similar bus plans were found in Tables 1 to 3, as well as a similar development of the objective value in Figure 4 and of the probability of meeting demand in Figure 6. In all these, there were also some small differences which can be attributed to different data (in particular travel times) being used. Still, similar conclusions can be drawn based on these different results.

Table 4. Running times in milliseconds for the cutting plane scheme from this thesis (This) and Kulshrestha et al. (2014) where values are available (Kul.). Iterations are for the cutting plane scheme of this thesis.

Degree of pessimism	Runtime (ms.)		Iterations	Degree of pessimism	Runtime (ms.)		Iterations
	This	Kul.			This	Kul.	
0	54	1,560	1	8	154		4
1	177		5	9	277		6
2	272		6	10	236		5
3	661	8,760	9	11	254		5
4	499		9	12	221		4
5	502		9	13	232		5
6	402		8	14	99		3
7	244		6	15	61	2,400	2

Namely, some locations are centrally located to be the best for pickup locations, and the objective and probability of meeting demand rise somewhat constantly until a degree of pessimism of four.

4.3 Runtime Analysis

This section provides the results described by Section 3.2. First, Figure 7 depicts the runtimes in milliseconds for each degree of pessimism for the RTPL, the ERTPL, the cutting plane scheme (CPS) and the cutting plane scheme with the R–TRTPL (CPST). The graph does not show the runtime for each degree of pessimism for the RTPL, because the RTPL runtime starting at a degree of pessimism of five is not viable to run. To cut off long runtimes quickly, the RTPL was stopped when it ran for more than 60 seconds. The graph is also limited to 6,000 milliseconds, because it makes the runtimes of the other models easier to see.

Figure 7 shows that D grows at such a rate that solving the RTPL it for all degrees of pessimism takes too long. The ERTPL solves this as D' is a lot smaller compared to D, as seen in Section 2.4. This results in the ERTPL runtimes being at most two seconds. The cutting plane scheme is another way of reducing the runtime, and can be seen to be quite a lot lower than those of the ERTPL in most cases, as it only considers the actual determining demand vectors. For bigger datasets the cutting plane scheme may therefore be the best method to obtain a low runtime. Lastly, the tri-objective cutting plane scheme shows a similar but scaled version of the runtimes of the regular cutting plane scheme. This can be explained by the fact that every iteration takes more time to solve as three objectives are minimised.

Table 5 shows some selected results of the second part of the runtime analysis, as described in Section 3.2. This consists of adding additional demand scenarios and running the cutting plane scheme with these. The table shows the minimum (Min.), average (Avg.) and maximum (Max.) runtimes for 20 runs when adding n additional demand scenarios up to $n^{max} = 10$ scenarios for the degrees of pessimism three, nine and fifteen. These were chosen, because three and nine showed relative peaks of runtimes in Figure 7, and fifteen can use all deviations from nominal so also all additional demand scenarios. Because of how these additional demand scenarios are generated, see Section 3.2, the feasible region is not affected. Therefore, this part is mostly to see if the runtime increases when the number of decision variables increases, or if it matters how

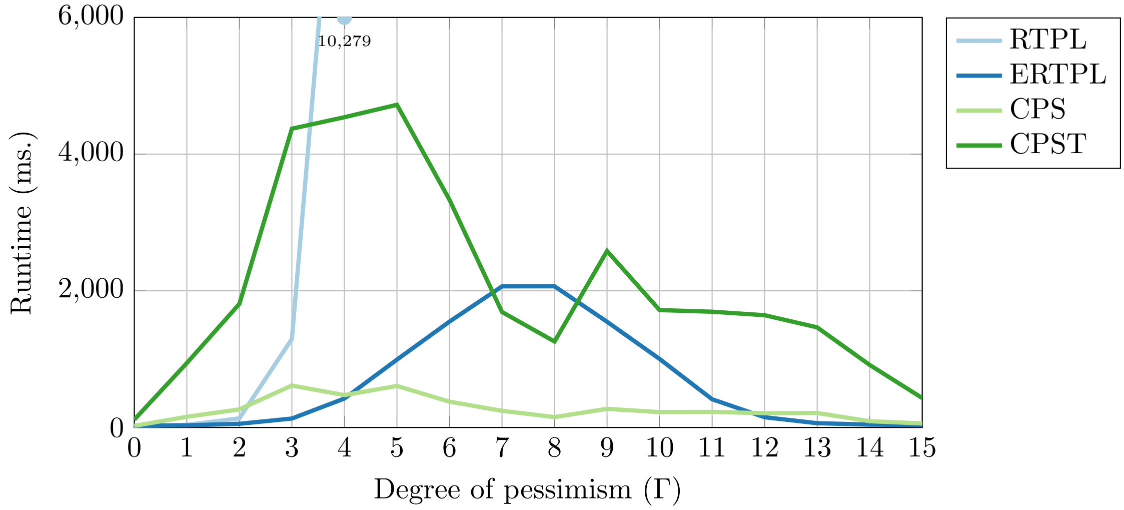

Figure 7. Runtimes of all solution methods in milliseconds. CPS is the cutting plane scheme from Section 2.3, and CPST is the cutting plane scheme with the R-TRTPL. The RTPL runtime was limited to 60 seconds.

many demand scenarios are given as input. If this effect is not much, a problem with a very high number of demand scenarios might also be considered in the future. This would for example be to approximate a continuous distribution of demand values.

From Table 5 it appears that the runtime of the cutting plane scheme is not very much affected by adding additional demand scenarios. For all three degrees of pessimism, the average and minimum runtimes are largely unaffected when the number of additional demand scenarios increases. The same can be said for the maximum of the degree of pessimism of fifteen. The other two maximums, three and nine, fluctuate some more. However, this is hard to predict and seems to be without much reason. For example, for $\Gamma = 3$, the maximum for $n = 0$ is close to the maximum for $n = 10$, but not to the maximum for $n = 8$. Still, the average runtime of the cutting plane scheme is not affected. This was to be expected as the cutting plane scheme considers only the maximum and nominal demand values through the WCD model. This shows that the cutting plane scheme is robust to additional demand scenarios, as long as the feasible region is not changed.

From this section it can be concluded that the solving the RTPL directly can be computationally heavy. The cutting plane scheme, and ERTPL provide noticeable reductions in runtime, with the cutting plane scheme being faster in most cases. The cutting plane scheme has iterations that are relatively fast, so that even when multiple demand vectors are determining, the runtime stays low. The runtime of the cutting plane scheme with the tri-objective model is as expected with mostly a scaled version of the regular cutting plane scheme. Lastly, adding additional demand scenarios that do not determine the solution have little impact on the runtime of the cutting plane scheme showing that it is robust to these kinds of modifications to the model.

4.4 Simulation

The results of the simulation procedure described in Section 3.3 are presented below. This procedure was done for 10,000 iterations and for all orderings of the TRTPL, using the cutting

Table 5. Minimum (Min.), average (Avg.) and maximum (Max.) runtimes in milliseconds from 20 runs, when adding n additional demand scenarios, up to $n^{max} = 10$ for degree of pessimism $\Gamma \in \{3, 9, 15\}$.

	$\Gamma = 3$			$\Gamma = 9$			$\Gamma = 15$		
n	Min.	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.
0	575.00	595.50	634.00	260.00	268.55	281.00	58.00	60.25	63.00
1	576.00	595.40	629.00	261.00	279.15	371.00	58.00	61.45	69.00
2	572.00	599.70	686.00	261.00	270.40	286.00	59.00	60.40	65.00
3	577.00	594.45	611.00	261.00	268.65	278.00	58.00	60.65	63.00
4	578.00	593.70	612.00	262.00	270.55	300.00	59.00	61.30	75.00
6	577.00	603.15	718.00	261.00	268.85	277.00	58.00	61.00	67.00
5	580.00	600.70	653.00	262.00	270.60	302.00	59.00	61.85	74.00
7	579.00	600.10	627.00	262.00	271.25	301.00	59.00	62.15	79.00
8	577.00	612.35	924.00	261.00	274.65	348.00	59.00	62.20	73.00
9	580.00	597.70	638.00	263.00	271.55	290.00	59.00	61.15	67.00
10	577.00	599.10	662.00	262.00	268.70	277.00	59.00	60.80	66.00

plane scheme to solve it. The acceptable fail percentage $\alpha = 1\%$ is used. Each ordering reached a fail percentage of at most α for a degree of pessimism of four. Table 6 shows for each ordering of the objectives, the fail percentage (Fail) in %, and the values of the three objectives. Recall the three objectives: 1) total evacuation time, 2) the latest arrival time, 3) the number of buses used. The fail percentages are fairly similar and because they are below α the number of people not arriving at a shelter is not a big effect in the following results. Notice that orderings 2-1-3 and 2-3-1 have the same three objective values, as well as 1-3-2, 3-1-2 and 3-2-1. The feasible regions for these different orderings must therefore also be the same. So, the following results will be similar for these sets of orderings, although the same solution being used is not guaranteed.

Table 6. Solutions used for each ordering of the TRTPL. The ordering of TRTPL$_{a\text{-}b\text{-}c}$ is denoted by a-b-c. Each solution used a degree of pessimism of four. Fail denotes the fail percentage of the solution in %. Objectives 1, 2, 3 denote the values of ξ_1, ξ_2, ξ_3 from the TRTPL, respectively.

Ordering	Fail (%)	Objectives		
		1	2	3
1-2-3	0.29	1,217.20	153.02	10.00
1-3-2	0.32	1,217.20	166.60	8.00
2-1-3	0.38	1,234.20	149.62	10.00
2-3-1	0.42	1,234.20	149.62	10.00
3-1-2	0.37	1,217.20	166.60	8.00
3-2-1	0.34	1,217.20	166.62	8.00

Figure 8 displays the empirical distribution of average arrival times (over the people arriving at a shelter) from these 10,000 iterations for each ordering of the TRTPL objectives. The bars in each plot show the relative frequency of each interval of average arrival times. Note that each plot is shown on the same interval to allow for easier comparison.

From Figure 8 it can be seen that the orderings of the objectives matter for the average arrival times, and that each empirical distribution is a bell curve like plot. It can also be seen that the average arrival times are well below the maximum time a bus can travel of 180 minutes. The orderings 1-2-3, 2-1-3 and 2-3-1 provide the lowest (leftmost) average arrival time distributions. So, if the latest arrival time is first minimised, the average is not necessarily lower. The fact that ordering 1-3-2 has a higher distribution than the ones where 2 is first minimised suggests that objective 2, the latest arrival time, is better at lowering the average arrival time distribution. This may be due to the fact that if the latest arrival time is minimised, the arrival times of all buses are minimised with it while in case of the total only the sum is minimised, so that one or more buses can have higher arrival times if the other ones are lower (which in turn increases the average arrival time). Minimising the number of buses first (3-1-2 and 3-2-1) results in the highest overall average arrival times, as fewer buses can be used. This can be seen from the objective 3 values in Table 6.

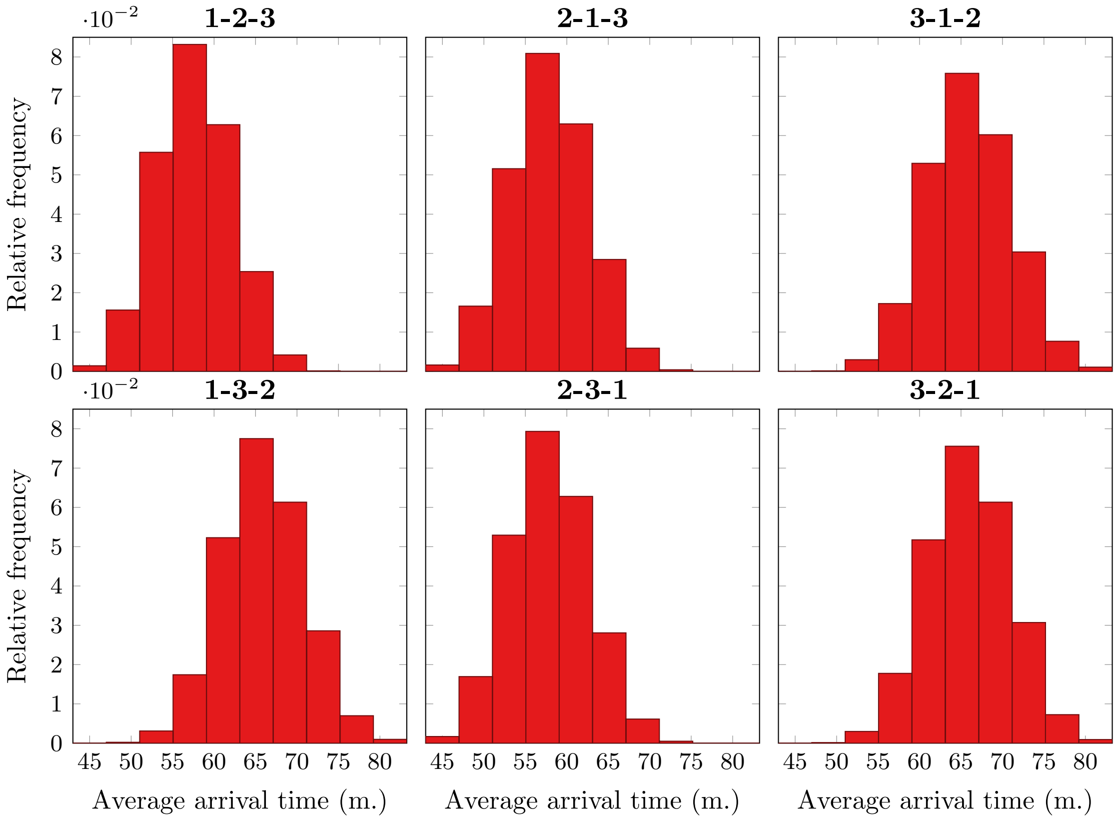

Figure 8. Empirical distributions of average arrival times for the solutions of the TRTPL for all orderings of the objectives (denoted above). The result of the TRTPL$_{a\text{-}b\text{-}c}$ is denoted by a-b-c. Each empirical distribution is obtained with 10,000 iterations.

Figure 9 has the same structure as Figure 8, but displays the empirical distribution of the latest arrival times (over the people arriving at a shelter). The empirical distributions in Figure 9 do not show bell curve like plots as seen in Figure 8. Here, two clear peeks in relative frequency occur for each ordering, except for 1-2-3. For the orderings 2-1-3 and 2-3-1 the second peek occurs one interval earlier than the other orderings. This is the expected result, because the latest arrival time was first minimised in this case. As in Figure 8, the orderings 3-1-2 and 3-2-1 have the highest arrival times, with the later peek having a higher frequency. The ordering 1-3-2 also again has the highest arrival times of the 1- or 2- orderings, with the same peeks as orderings 3-1-2 and 3-2-1.

The plot of 1-2-3 has more concentration towards lower arrival times compared to 2-1-3 and 2-3-1, so that, paradoxically, minimising total evacuation time first produces the lowest latest arrival time distribution. This may be due to the fact that objective 2 is the latest *possible* arrival time, over all demand vectors in D. Thus, objective 2 is the maximum of these plots, which can be seen when looking at the objective 2 values in Table 6. Objective 1 is the sum of the arrival times, meaning that earlier arrival times are also taken into account. This explains why the ordering 1-2-3 seems to have the lowest distribution of the latest arrival time.

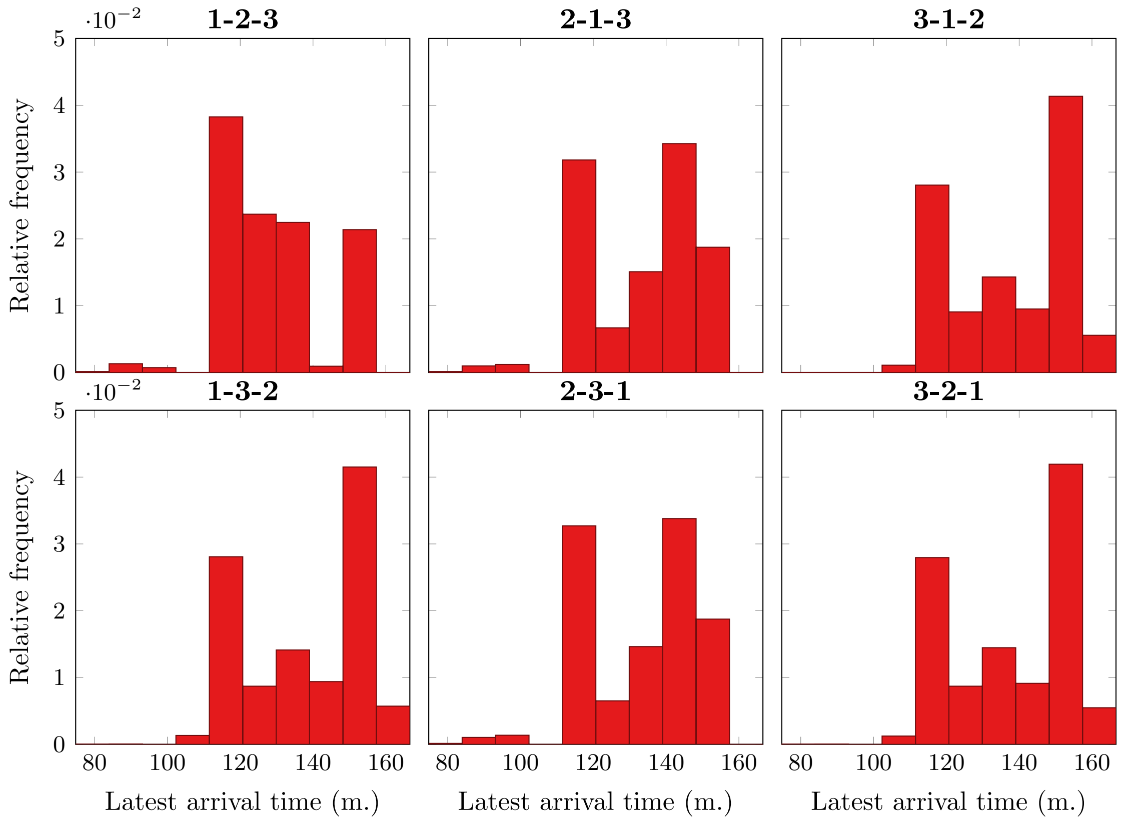

Figure 9. Empirical distributions of the latest arrival times for the solutions of the TRTPL for all orderings of the objectives (denoted above). The result of the TRTPL$_{a\text{-}b\text{-}c}$ is denoted by a-b-c. Each empirical distribution is obtained with 10,000 iterations.

From this section it can be concluded that the average arrival times display a normal like empirical distribution, which shifts based on the order in which the objectives are optimised. As one would expect, minimising an objective related to arrival times creates the lowest average arrival times. When objective 3 comes before 2, the number of buses is 8 instead of 10 from Table 6 which is why the average arrival time distribution is shifted for all these solutions in Figure 8. The latest arrival times in Figure 9 show irregular distributions. Objective 2 is for minimising the maximum of this plot, and objective 1 shifts more weight of the graph to the left (lower latest arrival times).

5 Conclusion

This paper considers an evacuation problem in a network, where some points in the network are shelters and the others have people that need to be evacuated to these shelters with buses. The number of people at each point is unknown but assumed to be from a known set of scenarios. Buses do not go to each point but go to pickup points which are chosen, so that all people can walk to a pickup point. The results are obtained using a modification of the Sioux Falls network from Kulshrestha et al. (2014).

Their methods are also replicated. That paper formulates a robust transit pickup location and bus allocation model (RTPL) and a cutting plane scheme for reducing the number of constraints in that model, while still getting an exact solution. This thesis introduces an equivalent version of the RTPL as another way to reduce the number of constraints, and an RTPL with three objectives. The three objectives being total evacuation time, the latest arrival time and the number of buses used. These objectives are optimised in some order.

The results found by Kulshrestha et al. (2014) are not entirely replicable, but similar results are obtained from the same methods. The cutting plane scheme and the equivalent version of the RTPL are also found to be actual reductions in runtime, so that using them makes sense. The cutting plane scheme is also found to be robust in runtime to adding additional data to the model.

On the model with three objectives, a simulation of the evacuation is carried out, where the order in which the objectives are optimised is varied. The latest arrival time objective minimises the distribution of average arrival times from this simulation the most. This is because, when the latest arrival time is minimised all buses must be used. With all buses, more people are being evacuated at the same time, bringing the average arrival time down. The total evacuation time objective minimises the distribution of the latest arrival times, this objective is minimised, all arrival times of all buses are minimised through a sum.

This study acknowledges the shortcomings of the Sioux Falls dataset that is used to obtain the results. It is a fairly small dataset, and in the modified network by Kulshrestha et al. (2014) only fifteen demand locations and four shelters were considered. Therefore, getting results for bigger (more complex) datasets is a logical next research step. The runtime for the RTPL already increased quite a lot for this dataset, but this might also happen to the cutting plane scheme or equivalent RTPL when a bigger dataset is used.

List of Symbols

G	Network to be evacuated
I	Set of demand locations
J	Set of shelters
T_{ij}	Round-trip travel time for a bus from $i \in I$ to $j \in J$
C_{ip}	Walking time from $i \in I$ to $p \in I$
T_{max}	Maximum total travel time for a bus
ω	Maximum walking time
B	Set of buses
β_b	Capacity of bus $b \in B$
K_j	Capacity of shelter $j \in J$
S^i	Number of demand scenarios for $i \in I$
d_s^i	Demand scenario s for $i \in I$, where $s = 1$ is nominal
Γ	Degree of pessimism
D	Demand uncertainty set
d^i	Demand value of $i \in I$ in the vector $d \in D$, also the decision variable for the WCD with the same meaning
$\mathbf{1}$	Indicator function
y_i	Binary decision variable being 1 if $i \in I$ is a pickup location
μ_{ib}	Binary decision variable being 1 if $b \in B$ is assigned to $i \in I$
δ_{ip}	Binary decision variable being 1 if $i \in I$ walks to $p \in I$
X_{bij}	Integer decision variable for the number of trips $b \in B$ takes from $i \in I$ to $j \in J$
W_i	Walking time for $i \in I$
$I^+(i)$	Reachable locations in I from $i \in I$ within maximum walking time
ξ and ξ_1	Objective of total evacuation time in the RTPL, R–RTPL and TRTPL
$\bar{D}$	Some subset of D
d_{max}^i	Maximum demand value for $i \in I$
D'	Equivalent demand uncertainty set
I'	Locations in I that are pickup locations
e_p	Decision variable for the excessive demand at $p \in I'$
η_p	Binary decision variable being 1 if $p \in I'$ has excessive demand
Δ_p	Decision variable for the difference between demand and bus capacity at $p \in I'$
z_s^i	Binary decision variable being 1 if demand scenario s is chosen for $i \in I$
E	Objective of total excessive demand in the WCD
ε	Tolerance on the total excessive demand objective for the cutting plane scheme
T	Decision variable being the maximum evacuation time of a bus
ξ_2	Objective of maximum evacuation time of a bus
ξ_3	Objective of number of buses used
d_{min}^i	Minimum demand value for $i \in I$
$U(a, b)$	The continuous uniform distribution on the interval (a, b)

n^{max}	Maximum number of additional demand scenarios for the runtime analysis
α	Maximum acceptable fail percentage of a solution for the simulation

List of Abbreviations

RTPL	Robust transit pickup location and bus allocation model
R–RTPL	Restricted RTPL
ERTPL	Equivalent RTPL
WCD	Worst-case demand model
TRTPL	Tri-objective RTPL
R–TRTPL	Restricted TRTPL
CPS	cutting plane scheme
CPST	cutting plane scheme with the R–TRTPL

References

Adhikari, I. M. & Dhamala, T. N. (2020). On the transit-based evacuation strategies in an integrated network topology. *The Nepali Mathematical Sciences Report*, *37*(1-2), 1–13. doi: 10.3126/nmsr.v37i1-2.34063

Adhikari, I. M., Pyakurel, U. & Dhamala, T. N. (2020). An integrated solution approach for the time minimization evacuation planning problem. *International Journal of Operations Research*, *17*(1). doi: 10.6886/IJOR.202003_17(1).0002

Ahmadi, G., Tavakkoli-Moghaddam, R., Baboli, A. & Najafi, M. (2022). A decision support model for robust allocation and routing of search and rescue resources after earthquake: a case study. *Operational Research*, 1–43. doi: 10.1007/s12351-020-00591-5

Alam, M. J., Habib, M. A. & Husk, D. (2022). Evacuation planning for persons with mobility needs: A combined optimization and traffic microsimulation modelling approach. *International Journal of Disaster Risk Reduction*, *80*, 103164. doi: 10.1016/j.ijdrr.2022.103164

Fereiduni, M. & Shahanaghi, K. (2017). A robust optimization model for distribution and evacuation in the disaster response phase. *Journal of Industrial Engineering International*, *13*, 117–141. doi: 10.1007/s40092-016-0173-7

Gao, X., Nayeem, M. K. & Hezam, I. M. (2019). A robust two-stage transit-based evacuation model for large-scale disaster response. *Measurement*, *145*, 713-723. doi: 10.1016/j.measurement.2019.05.067

Goerigk, M., Grün, B. & Heßler, P. (2014). Combining bus evacuation with location decisions: A branch-and-price approach. *Transportation Research Procedia*, *2*, 783-791. (The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands) doi: 10.1016/j.trpro.2014.09.088

Khalili, S. M., Mojtahedi, M., Steinmetz-Weiss, C. & Sanderson, D. (2024). A systematic literature review on transit-based evacuation planning in emergency logistics management: Optimisation and modelling approaches. *Buildings*, *14*(1). doi: 10.3390/buildings14010176

Kulshrestha, A., Lou, Y. & Yin, Y. (2014). Pick-up locations and bus allocation for transit-based evacuation planning with demand uncertainty. *Journal of Advanced Transportation*, *48*(7), 721-733. doi: 10.1002/atr.1221

Kyriakakis, N. A., Marinaki, M., Matsatsinis, N. & Marinakis, Y. (2022). A cumulative unmanned aerial vehicle routing problem approach for humanitarian coverage path planning. *European Journal of Operational Research*, *300*(3), 992-1004. doi: 10.1016/j.ejor.2021.09.008

Nabavi, S., Vahdani, B., Nadjafi, B. A. & Adibi, M. (2022). Synchronizing victim evacuation and debris removal: A data-driven robust prediction approach. *European Journal of Operational Research*, *300*(2), 689-712. doi: 10.1016/j.ejor.2021.09.051

Nadeem, I., Uduman, P. S. & Dar, A. A. (2019). An integrated bus-based routing and dispatching approach for flood evacuation. *Yugoslav Journal of Operations Research*, *30*(4), 443–460. doi: 10.2298/YJOR190415028N

Shahparvari, S., Abbasi, B., Chhetri, P. & Abareshi, A. (2019). Fleet routing and scheduling in bushfire emergency evacuation: A regional case study of the black saturday bushfires in australia. *Transportation Research Part D: Transport and Environment*, *67*, 703-722. doi: 10.1016/j.trd.2016.11.015

Zhang, X. & len Chang, G. (2014). A transit-based evacuation model for metropolitan areas. *Journal of Public Transportation*, *17*(3), 129-148. doi: 10.5038/2375-0901.17.3.9

Zheng, H. (2014). Optimization of bus routing strategies for evacuation. *Journal of Advanced Transportation*, *48*(7), 734-749. doi: 10.1002/atr.1224

A Proofs

A.1 Proof for WCD constraints

The following proves that the constraints from the WCD ensure that in an optimal solution

$$e_p = \max\{\Delta_p, 0\} \tag{7}$$

for all $p \in I'$.

Suppose (Δ, e, η, d) are an optimal solution to the WCD and let $p \in I'$. Three cases for the value of Δ_p are considered:

Case I. Suppose $\Delta_p < 0$. If $\eta_p = 0$, constraints (4c) and (4h) imply $e_p = 0$, and if $\eta_p = 1$ constraint (4d) implies $e_p \leq \Delta_p$, which is a contradiction with constraint (4h). Thus, $\eta_p = 0$ and $e_p = 0$, so that Equation (7) holds. Note that in this case constraint (4d) becomes inactive because constraint (4b) implies it, as the first term in (4b) is non-negative.

Case II. Suppose $\Delta_p = 0$. As seen in Case I, if $\eta_p = 0$ it follows that $e_p = 0$ and if $\eta_p = 1$ it follows that $0 \leq e_p \leq \Delta_p$. Both imply $e_p = 0$, so that Equation (7) is satisfied.

Case III. Suppose $\Delta_p > 0$. As seen in Case I, if $\eta_p = 0$ it follows that $e_p = 0$ and if $\eta_p = 1$ it follows that $e_p \leq \Delta_p$. If $e_p < \Delta_p$, while all other constraints of (4) are satisfied, the objective (4a) can be set strictly higher by setting $e_p = \Delta_p$. This increase does not result in constraint (4c) being no longer satisfied, as $\Delta_p \leq \sum_{i\in I^+(p)} \hat{\delta}_{ip} d^i_{max}$, by definition of d^i_{max} and constraint (4b). Thus, in an optimal solution $e_p = \Delta_p$ which means Equation (7) is satisfied.

A.2 Proof of Proposition 1 (Equivalent RTPL)

It is enough to show that for $X_{bpj} \in \mathbb{Z}_{\geq 0}$, and $\delta_{ip} \in \{0,1\}$, for all $b \in B, i \in I, p \in I^+(i), j \in J$, it holds that:

$$\sum_{b\in B}\sum_{j\in J} \beta_b X_{bpj} \geq \sum_{i\in I^+(p)} \delta_{ip} d^i \quad \forall p \in I, d \in D \tag{8}$$

$$\Longleftrightarrow$$

$$\sum_{b\in B}\sum_{j\in J} \beta_b X_{bpj} \geq \sum_{i\in I^+(p)} \delta_{ip} d^i \quad \forall p \in I, d \in D'. \tag{9}$$

The proof is split in the implication and reverse implication:
"(8) $\Longrightarrow$ (9)": As $D' \subseteq D$, this is clear.
"(8) $\Longleftarrow$ (9)": For the sake of contradiction, suppose (8) does not hold. This implies the inequality is not satisfied for some pair $\left(\tilde{p}, \tilde{d}\right) \in I \times D$. We will construct a demand vector $\bar{d} \in D'$ from $\tilde{d}$. If $\tilde{d} \in D'$ we are done by setting $\bar{d} = \tilde{d}$, otherwise we do the following. To start, set $\bar{d}'$ equal to $\tilde{d}$. Then, for every $i \in I$ such that $\bar{d}^i \notin \{d^i_1, d^i_{max}\}$, set $\bar{d}^i = d^i_{max}$. This results in $n := \Gamma - \sum_{i\in I} \mathbf{1}_{\{\bar{d}^i = d^i_{max}\}} \geq 0$ degrees of freedom in the demand vector, as $\tilde{d} \in D$ and $\sum_{i\in I} \mathbf{1}_{\{d^i \neq d^i_1\}} \leq \Gamma$ for all $d \in D$, which is equivalent by the previous operation on $\bar{d}$. Lastly, take the first n demand locations i from I for which $\bar{d}^i = d^i_1$ and for all these set $\bar{d}^i = d^i_{max}$. These n demand locations always exist because $\Gamma \leq |I|$ and $d^i_{max} \neq d^i_1$ for all $i \in I$ by assumption. By construction $\bar{d}$ is now in D', and $\tilde{d}^i \leq \bar{d}^i$ for all $i \in I$. This results in:

$$\sum_{b\in B}\sum_{j\in J} \beta_b X_{b\tilde{p}j} < \sum_{i\in I^+(\tilde{p})} \delta_{i\tilde{p}} \tilde{d}^i \leq \sum_{i\in I^+(\tilde{p})} \delta_{i\tilde{p}} \bar{d}^i,$$

as (8) does not hold for $\left(\tilde{p}, \tilde{d}\right)$, and $\delta_{i\tilde{p}} \geq 0$ for all $i \in I^+(\tilde{p})$. Now, (9) does not hold because of the pair $\left(\tilde{p}, \bar{d}\right)$, showing the reverse implication by contradiction.

B Data Manipulation

For this discussion, let x and y be two arbitrary nodes in the Sioux Falls network and denote $\tau_{x,y}$ as the shortest path from x to y using the free flow travel times.[1] Comparing $\tau_{x,y}$ with the objectives, pickup points and bus plans provided by Kulshrestha et al. (2014), it follows that:

1. In an optimal solution it is logical that buses are used most effectively. This means as many demand locations go to some pickup point as possible. The limit on this is five minutes (ω). So, one would expect the pickup points be chosen such that most or some demand

[1] These can be found at: `https://raw.githubusercontent.com/bstabler/TransportationNetworks/master/SiouxFalls/SiouxFalls_net.tntp`

locations walk close to that limit. This point can be made stronger by the fact that walking time is not considered in the objective so that these may be as high as possible. For the nominal plan, with the least pickup points, from Kulshrestha et al. (2014) the following holds:

$$\tau_{x,y} \leq \omega$$

for all demand location $x \in I$, where y is the closest pickup location based on all $\tau_{x,y}$. For every demand location x, except node 8, this only holds for one such y (the closest). By the discussion above, these free flow travel times are thus a logical choice for the walking times, as the least possible number of pickup locations seem to be chosen for this optimal solution. Thus, $C_{ip} = \tau_{i,p}$ is used for all $i, p \in I$.

2. The three bus plans provided by Kulshrestha et al. (2014) also give some information. The following holds for each of the three plans individually. Let $\hat{X}$ denote the bus plans of that solution. Then

$$3.4 \cdot \sum_{b \in B} \sum_{i \in I} \sum_{j \in J} \tau_{i,j} \hat{X}_{bij} = \sum_{b \in B} \sum_{i \in I} \sum_{j \in J} T_{ij} \hat{X}_{bij},$$

where the left-hand side is the objective of the plan (also given). The free flow travel times are symmetric, so each $\tau_{i,j}$ can also be seen as $(\tau_{i,j} + \tau_{j,i})/2$ (As T_{ij} are round-trip and $\tau_{i,j}$ is one-way travel times). As this was found to hold for all three bus plans $T_{ij} = 3.4 \cdot \tau_{i,j}$ is used for all $i \in I$ and $j \in J$. Important to note is the limitation of this: only the sums of travel times are given (the objective) so that verifying the correctness (i.e. same used as by Kulshrestha et al. (2014)) of individual travel times is not possible. What is sure, is that for the exact same bus plan the objective will have the same value.

C Programming Code

The code is contained in package/Java project named `thesis`. This is divided into several 'sub-packages' or folders for clarity. The list that follows describes all Java files in the project. Whenever `package thesis.`<folder name> is listed, it means all the Java files that follow are contained in the folder <folder name>.

`package thesis` contains the following:

- `Main.java`: The main class of the project, that combines everything. When this class is run it obtains *all* the results presented in Section 4 at once. Classes were named so that every method call speaks for itself in this main class.

- `Solver.java`: The class that combines all models, as well as all parameters and data on the network. It first of all stores the `Dataset` and `Parameters` instances used in the whole solve (see later on for explanations on those). It also stores the Gurobi environment which is needed for every Gurobi model to be able to solve. Because this class stores it, the environment only needs to be initialised once. The most important methods `Solver` has, are one for each type of model/solution method used in the thesis: RTPL, R–RTPL, WCD, cutting plane scheme, TRTPL, R–TRTPL, cutting plane scheme tri-objective. These methods take as inputs the various things needed for that particular model, just like mentioned in Section 2. Note that the cutting plane scheme methods contain the cutting plane scheme algorithms as well because these become very short by the other methods, while the rest of the models have separate classes. Note that there is no separate class for the R–RTPL because the RTPL class was made to have a set D as input, which is either D, obtained from Γ, or $\bar{D}$ as input to the method, or D', also obtained from Γ.

- `package thesis.containers`: These classes are 'containers' as they only store data that can be retrieved or changed, with get and set methods.
  - `Dataset.java`: This class stores all information regarding the dataset. All inputs to the model that are specific to this dataset are in this: everything regarding the network (demand locations and shelters) and travel times, and bus capacities.
  - `Parameters.java`: This class stores all information that is used as input to the model, and thus can be changed to obtain possibly different results. This class stores everything that is an input to some model or simulation that is not specific to the dataset, i.e. every input to some model that is not in the `Dataset` class.
- `package thesis.models`: These classes are all models that are needed. As a lot of the same constraints are used between different models, not all models mentioned in Section 2 are present here (e.g. see explanation of `Solver.java`).
  - `Model.java`: A default class for all models. Defines the different parts all models should have (not implemented), bus does implement a `run()` method that calls all these methods in the correct order. This is done so that this code does not need to be written in every model class separately.
  - `RTPL.java`: The class that implements `Model.java` and thus all its unimplemented methods. These have methods that speak for itself. Furthermore, every constraint has its own method. A static method is used to call the model and solve it (to keep `Solver.java` clean), which takes the set D as an input so that the R–RTPL can also be solved without any change to this class. Also contains the class `Solution` in it which is just a container class for the necessary parts of the solution needed. Note that the RTPL computes all the three objectives from the TRTPL but only adds the first one to the model, so that the value of the other two can also be obtained from the `RTPL.Solution`.
  - `TRTPL.java`: The class that adds the three objectives. The objectives are already computed in the RTPL (see why above), so that the only thing different is to add the other objectives to the model as well. This is done in the order provided by `sigma`. This is a permutation of the indices 0,1,2 which stand for objectives 1,2,3 respectively (as arrays are 0-indexed).
  - `WCD.java`: The class that implements `Model.java` for the WCD model. It also has methods for all its constraints. And also a `Solution` class that stores all the necessary parts (E and d as seen in cutting plane scheme). This `Solution` class being named the same as the one for the `RTPL` does not create a problem and is deliberate. It

means <Model>`.Solution` can be called to obtain the solution for that model, which makes it instantly readable whenever it is used.

- `package thesis.results`: These classes are for all parts of the results in Section 4 (or implementing Section 3 where every subsection gets its own class with the same name).
  - `Replication.java`: The class that does the replication part of the thesis. The static `run()` method is the main one, called in `Main.java` which solves the cutting plane scheme for each degree of pessimism Γ. For each Γ it records the things that were described in Section 3.1: runtime, objective, probability of meeting demand (denoted as successPercentage), and number of iterations for the cutting plane scheme. An inner private class `Result` is contained in this class that stores these things for every Γ. This makes it easy to first write a file on a particular solution, containing all this information and the pickup points and bus plans, and later on making one file csv file for all results, with all the information mentioned above. The csv file also computes the difference in objectives and probabilities with the paper itself, so that it can be easily shown in a graph.
  - `RuntimeAnalysis.java`: The class that does the runtime analysis part of the thesis. The static `run()` method is again the main one, which has two parts just like described in Section 3.2. First, the runtime is done for each of the 4 models, with a time limit of 1 minute. When the time limit is once exceeded this model is then skipped for subsequent solves to save time. Otherwise, the runtime is recorded which is then put into one csv file. The second part adds demand scenarios. This is done by making a copy of the `Solver` class and for each iteration of the `Dataset` class. A randomly chosen demand vector is chosen n^{max} times and this is added to the demand scenarios and then the dataset copy. Then the runtime for each Γ is recorded for each added demand scenario using the `SummaryStatistics` class from `apache.commons.math3` so that the average, minimum and maximum runtime for each Γ and number of extra scenarios can be easily obtained. These three (average, min, max) are put into a matrix each and into separate csv files.
  - `Simulation.java`: The class that does the simulation part of the thesis. The static `run()` method is again the main one. For ease of use S_3 is stored here, as all orderings of the objectives 1,2 and 3. First, the solutions are found, using the procedure with α as described in Section 3.3, which gets a list of `SolutionInfo`, which is an inner private class that stores the solution along with some additional information that is needed for this simulation part (Γ, fail percentage, and σ if applicable). The list of solutions is written to a csv with their objectives and this additional information, where the list contains solutions to the RTPL and to each ordering of the TRTPL. Then for each solution in this list, the events of the solution are simulated with a randomly generated demand vector. As this simulation of events is quite some code it gets its own class (see `EventSimulator.java`).
- `package thesis.utils`: These classes are named 'utils' as they provide utilities for the other classes mentioned above, it makes them more readable and it means every class does

not contain too many lines of code.

- `DatasetReader.java`: This util reads a preprocessed dataset from a given directory (using the static `readFromDirectory()` method). It assumes some structure to the given directory containing certain named files, which will be present if the `Preprocessor` class was first used (see later).
- `EventSimulator.java`: This util simulates the events of a solution and writes certain things of interest in csv format into a `String` that is returned. The things mentioned in Section 3.3 are the things of interest. As the solution does not change, first a list of all events is obtained. The inner private `Event` class contains the needed information of the event, which can have multiple types. This inner class also contains some information that is only needed for some type of event. Some number of iterations each event is visited, in the order of the time of the event. Note that when events have the same time, it was mentioned in Section 3.3 that buses going to closer shelters are given priority which is done by sorting on the time of the arrival event if the event times are equal. The walking through time is done by going through the sorted list and then doing different things depending on the type of event: `demandAtPickup` is a simple addition of demand, `busDeparturePickup` picks up to most people possible and records this in its corresponding arrival event, and `busArrivalShelter` records the various variables of interest (only if it has some quantity).
- `OutputWriter.java`: This class is used to easily write files in all classes. It makes a directory based on the time and the number of folders in the output directory (to have a unique number). It has a method to write a simple info file, containing the information on the run. And it has methods for making a directory and a file, where the file is put into the last made directory (so that a folder for each part of the results is easily obtained).
- `ParameterReader.java`: Similarly to the `DatasetReader.java` it reads from a file all parameters. It assumes this file has some structure with the exact variable names. If the seed is set to 0 in the file, the seed of the `Parameters` class it returns is set to the current system time in milliseconds.
- `Preprocessor.java`: This preprocesses the dataset of a very particular structure. The most important part is that it uses the Dijkstra algorithm to compute the shortest paths between every node pair in the network, this is then converted to contain only those from i to p in I for C_{ip}. and from i in I to j in J for T_{ij}. Even though these i, p, j may not be increasing by 1, and starting at 0, the arrays are stored this way. The node numbers can be obtained then by the sets I and J.
- `Setmaker.java`: This util makes D and D' as described in Section 3, using recursion. They are very simple algorithms, where the current deviation is also kept, so that some part of the recursion tree can be instantly cut off when the deviation is too high or low for Γ. It returns a list of double arrays in both cases so that the sets can be easily iterated over.
- `SimulationHelper.java`: This util is mostly to provide some methods that are used

in the simulation of the replication part, and in the actual simulation part so that these methods are not doubly written. It would also not make sense to write these in any of the two classes from `thesis.results` as these are preferably completely separate classes.