ABSTRACT

KOSHUTE, PHILLIP. A Model and Optimal Schedule Design for a Fixed Bus Route. (Under the direction of Xiuli Chao.)

Transit passengers' average total trip time can be minimized by employing the schedule that achieves the optimal balance between initial delay (tardiness) and layover time (earliness). To compare different proposed schedules, we carry out a deterministic simulation which uses link times taken from historical data in the order that they originally occurred. The simulation’s arrival and departure times, in general, are not the same as those in the original data because a different schedule is in place. We measure total trip time by tracking deviations between the scheduled and simulation departure times and computing an appropriately weighted sum (reflecting the number of passengers affected by a given deviation).

We consider the number of vehicles serving the route, the number of time-points $N$, and the scheduled cycle length to be fixed. Schedules are represented by 1-by-$N$ solution vectors whose elements correspond to intervals between scheduled departure times of consecutive time-points. A Modified Steepest Descent (MSD) algorithm is used to search for the optimal solution.

Throughout the project, North Carolina State University's Werewolf A bus route is used as a test case. Twenty-four days of historical data are utilized. Empirical results suggest that the space of feasible solutions may be convex. An optimal solution is identified and verified by a brute force search of all feasible solutions.

# A MODEL AND OPTIMAL SCHEDULE DESIGN FOR A FIXED BUS ROUTE

by

**PHILLIP KOSHUTE**

A thesis submitted to the Graduate Faculty of

North Carolina State University

In partial fulfillment of the

Requirements for the degree of

Master of Science

**OPERATIONS RESEARCH**

Raleigh, NC

May 12, 2007

Approved by:

| | |
|---|---|
| _____ | _____ |
| Salah Elmaghraby | Billy Williams |

_____

Xiuli Chao
Chair of Advisory Committee

# Dedication

To the people of St. Joseph's Catholic Church in Raleigh, whose generosity, if repeated universally, would put public transit out of business. Without their friendship, encouragement, and prayerful support, my time as a master's student would have been far more difficult and far less meaningful.

# Biography

Phillip Koshute was born in Beaver, Pennsylvania. Beginning in August 2001, he studied at Case Western Reserve University in Cleveland, Ohio, where he rode many buses. In May 2005, he received his Bachelor of Science degree in Mathematics. In May 2007, he will receive his Master of Science degree in Operations Research from North Carolina State University. He hopes to use his skills and experiences to serve the less fortunate.

# Acknowledgements

I would like to express thanks to my professors, especially Dr. Xiuli Chao, Dr. Salah Elmaghraby, and Dr. Billy Williams, for their assistance and patience. Dr. John Baugh also shared numerous helpful ideas, including suggestions which directly led to the use of deterministic simulation and the Modified Steepest Descent search algorithm. Dr. Ralph Smith's encouragement was crucial to the development of my ability to type this report in LaTeX. Likewise, without our department secretary, Barbara Walls, I may never have kept all of the administrative details straight.

Slade McCalip, Tom Kendig, and Starr Wimberly from the North Carolina State University Department of Transportation offered a generous amount of time and correspondence helping me to understand the intricacies of the Wolfline system. They were also instrumental in my acquisition of data from TransLoc Incorporated's Dominique Bischof, who also deserves thanks for his eager replies to all of my questions.

Most significantly, I would like to thank my family: Throughout the sometimes-frustrating thesis process, Mom, Dad, Maria, Gina, Nick, and Missy gave constant reassurance and timely humor that always helped me keep an appropriate perspective. Likewise, the positive effects of their unconditional love and committed prayers cannot be overestimated.

Still, Blessed Pier Giorgio Frassati (1901-1925), a young Italian who used his engineering talents to serve the poor and share his love for the Gospel should not be forgotten. His joyful example of combining technical know-how with a servant's heart continually brought me inspiration.

Praise God for His many gifts. Only by His blessings, often bestowed through the generosity of others, have I been able to complete this project.

# Contents

# List of Tables

# List of Figures

# List of Symbols

For symbols that represent time, lowercase symbols generally correspond to interval lengths, while uppercase symbols generally correspond to clock instances.

$A_{k,m,y}$ : simulation arrival time at $s_k$ during cycle $m$ on day $y$.

$D_{k,m,y}$ : simulation departure time from $s_k$ during cycle $m$ on day $y$.

$d_k$ : dwell time at $s_k$.

$e_k$ : earliness weight for $s_k$.

$f(\cdot)$ : objective function value for explicit calculation measurement method.

$f_{wd}(\cdot)$ : objective function value for weighted deviations measurement method.

$h_k$ : layover time at $s_k$.

$i$ : time-point index.

$j$ : time-point index.

$l_k$ : link time from $s_{k-1}$ to $s_k$.

$M$ : total number of cycles.

$m$ : cycle index.

$N$ : total number of time-points on a route.

$p_{ij}$ : proportion of passengers with origin $s_i$ and destination $s_j$; typically, $i \neq j$.

$S_{k,m,y}$ : scheduled departure time from $s_k$ during cycle $m$ on day $y$.

$s_k$ : a time-point.

$t_{ij}$ : total trip time from origin $s_i$ to destination $s_j$.

$t_k$ : tardiness weight for $s_k$.

$w_k$ : initial delay for a passenger waiting to board at $s_k$.

$\mathbf{x}$ : solution vector of decision variables, adjusted during the minimization of average total trip time.

$\mathbf{x_{current}}$ : the solution vector corresponding to the current route schedule.

$x_i$ : an individual element of x; the interval between scheduled departure times of time-points $s_{i-1}$ and $s_i$.

$\mathbf{x_{init}}$ : an initial solution, used at the start of the search algorithm.

$Y$ : total number of days.

$y$ : day index.

# Glossary

**Ahead-of-schedule policy** : If a bus arrives at a time-point ahead-of-schedule, it should not depart from this time-point until its scheduled departure time.

**Alight** : To depart from the bus.

**Cost** : The objective function value, typically expressed in minutes.

**Cycle** : Progression from an initial time-point, to all time-points on a given route, and back to the initial time-point.

**Destination** : The time-point at which a passenger ends his trip and alights.

**Dwell time** : The time necessary for passengers to board and alight at a particular time-point ($D_{j,m,y} - A_{j,m,y}$). Also referred to as "non-layover dwell time."

**Headway** : The time interval between consecutive departures from a given time-point; that is, scheduled cycle length divided by the number of buses serving the route.

**Initial delay** : The time that a passenger waits at a stop before boarding the bus. For punctual passengers, initial delay is actual departure time minus scheduled departure time.

**Layover time** : The time that an ahead-of-schedule bus idles at a stop, in conformity with the ahead-of-schedule policy.

**Link time** : The travel time between consecutive time-points ($A_{j,m,y} - D_{j-1,m,y}$).

**Natural departure time** : The time when a bus would depart from a time-point if the ahead-of-schedule policy were not in place; that is, the sum of arrival and dwell times at that time-point.

**Origin** : The time-point at which a passenger begins his trip and boards the bus.

**Scheduled cycle length** : The time interval between scheduled departure times for the same time-point on consecutive cycles; that is, the frequency with which the schedule repeats.

**Time-point** : A special type of of bus stop distinguished by its available physical space, within which a bus arriving ahead-of-schedule can and must wait until the scheduled departure time.

**Total trip time** : The time between the passenger's arrival at the trip origin and the arrival of the bus at the passenger's trip destination.

# 1 Introduction

A challenge faced by any transit system is maintaining an efficient schedule. Planners seek to minimize both the time that boarding passengers must wait before a bus arrives and, once the passengers have boarded, the amount of time required to reach their respective destinations. This challenge is made more complex by the conflict into which these two objectives come.

Further complexity arises from the stochastic character of link times (travel times between consecutive time-points[1]). In practice, these link times depend on a variety of factors, including traffic congestion, number of stops made, driver behavior, signalized intersections, and weather. Consequently, to accurately reflect the system, link times must be represented with a distribution, rather than a fixed time amount. In practice, this distribution can be obtained directly from historical data by considering a discrete number of exact values of past link times, or indirectly by fitting a more conventional continuous distribution (e.g., skewed normal).

By gathering and applying appropriate information from the distribution, arrival times can be made arbitrarily reliable; that is, the probability that a bus will arrive late can be made arbitrarily small. The intervals between scheduled departure times of consecutive time-points simply must be set sufficiently large. For instance, if the link time between two time-points, $s_{n-1}$ and $s_n$, is a random variable with cumulative distribution function $G_l(t)$, minimum $l_{min}$, and maximum $l_{max}$, and the dwell time (departure time minus arrival time) at time-point $s_n$ is a random variable with cumulative distribution function $G_d(t)$, minimum $d_{min}$, and maximum $d_{max}$, the probability that a bus will be able to depart on-time from time-point $s_n$ will be high if the interval between the scheduled departure times of these two time-points is set to be $l_{max} + d_{max}$.

However, standard transit policy mandates that if a bus arrives at a time-point ahead-of-schedule, it should not depart from this time-point until its scheduled departure time. Herein, allotting large intervals between scheduled departures times has a major draw-

[1] A time-point is a special type of bus stop distinguished by its available physical space, within which a bus arriving ahead-of-schedule can and must wait until the scheduled departure time. Other terms are similarly defined in the Glossary on page xi.

back. The layover time during which buses that arrive ahead-of-schedule must idle at a time-point $s_n$ can significantly increase total trip times for passengers that have boarded before $s_n$ and will alight at $s_{n+1}$ or later. Therefore, this practice of allotting large intervals between scheduled departure times must be moderated.

On the other hand, the probability of a bus arriving soon enough at a time-point that it must idle until the scheduled departure time can be made arbitrarily small by making the interval between scheduled departure times of consecutive time-points sufficiently short. For instance, for the same example cited above, if the interval between scheduled departure times of the two time-points is set at $l_{min}+ d_{min}$, a bus will avoid incurring a layover at time-point $s_n$ on nearly every cycle.

However, when a bus arrives late at time-point $s_n$, the initial delays (and therefore the total trip times) for passengers that have arrived punctually to board at time-point $s_n$ will increase. Moreover, a late arrival at time-point $s_n$ increases the probability of a late arrival at time-point $s_{n+1}$, and along a route, delay at a single time-point can lead to delays at multiple subsequent time-points. For passengers boarding at each of these time-points, the later that a bus arrives, the more that their total trip times will increase. This effect creates a need for the practice of allotting short intervals between scheduled departure times to also be moderated.

It is reasonable to expect that the optimal schedule, that is, the schedule which minimizes passengers' average total trip time[2] will lie somewhere in between these two extremes. However, neither the schedule that achieves this optimal balance nor the means by which such a schedule can be determined is immediately clear.

Provided that no significant changes have been made to the surrounding transportation network, it is reasonable to assume that historical data is a relatively accurate predictor of future transportation system behavior. Herein, historical data can be a key component to the search for an optimal schedule.

[2]Total trip time is the time between the passenger's arrival at his or her trip origin and the arrival of the bus at the passenger's trip destination.

# 2 Project description

## 2.1 Objective

The objective of this project is to develop a process by which the schedule that minimizes passengers' average total trip time (and therein achieves the optimal balance between the two extremes described above) can be identified. Historical data will be used to evaluate potential schedules. Improvement to schedule efficiency and minimization of passengers' average total trip time will be sought by varying the intervals between consecutive time-points' scheduled departure times.

The test case for this project will be the Werewolf A bus route which provides evening service to North Carolina State University (NCSU) as part of the Wolfline bus system. This route, served by a single bus, runs thirteen full forty-minute cycles each evening.

## 2.2 Assumptions

Our assumptions for this project can be summarized by the following:

- All boarding passengers arrive promptly at exactly the scheduled departure time from the time-point at which they are boarding the bus. Therefore, the arrival of all passengers boarding at a given time-point during a given cycle on a given day is simultaneous. Though, in practice, passengers' arrival times may vary, this assumption allows initial delay to be measured from the scheduled departure time.

- Moreover, passengers board and alight only at the designated time-points. Though this is unrealistic in practice, passengers boarding at a stop between time-points $s_n$ and $s_{n+1}$ still must arrive at their stop by the scheduled departure time of the preceding time-point $s_n$ to ensure that they do not miss their bus.

- Origin-destination trip pairs are uniformly distributed; that is, a passenger that boards the bus at a given time-point is equally likely to alight at any other time-point. The distribution of origin-destination pairs can be expressed as a matrix $P$,

where $p_{ij}$ gives the probability that a passenger will board at $s_i$ and alight at $s_j$. Variations of this assumption are discussed in Sections 6.1.5 and 12.

- Non-layover dwell times are constant, which reflects the limited variability exhibited by non-layover dwell times in the historical data and is consistent with our assumption that origin-destination trip pairs are uniformly distributed.
- The number of vehicles serving the route (one) is fixed.
- The number of time-points on the route (six) is fixed.
- According to the preferences of the NCSU Department of Transportation, the scheduled cycle length[3] (forty minutes) is fixed. This assumption is relaxed in Section 8.3.
- The transportation network including and surrounding the Werewolf A route will not undergo significant changes in the near future and therefore this route's historical data is a reasonably accurate predictor of its link times in the future.

## 2.3 Overview

The remainder of this report is organized as follows. Section 3 contains a more detailed description of the test case. Section 4 explains the acquisition and organization of the route's historical data. Section 5 reviews past developments in the modeling of and optimal schedule design for bus routes. Section 6 considers several methods to measure total trip time and gives an algorithm that uses one of these methods to search for the optimal schedule. Results are summarized in Section 7. The report concludes with sensitivity analysis (Section 8), comparison with alternative methods that provide verification (Section 9), recommendations (Section 10), conclusions (Section 11), and avenues for future research (Section 12).

[3]Scheduled cycle length is the interval between scheduled departure times for the same time-point on consecutive cycles.

# 3 Test case description

In all, NCSU and its surrounding areas are served by thirteen Wolfline bus routes, some of which are simultaneously served by multiple vehicles. Two evening routes, Werewolf A and Werewolf B, provide supplementary service to areas that are served by combinations of the Wolfline's eleven other routes during the day. The Werewolf A route operates seven days a week and its current first and last scheduled departure times are 5:55 PM (from D.H. Hill Library) and 2:27 AM (from Fraternity Court, returning to D.H. Hill Library a final time).

During each of its thirteen forty-minute cycles, the Werewolf A route has numerous designated points at which passengers may board or alight the bus. Six of the eight stops currently listed on the route schedule exist outside the flow of traffic and have layover capacity (physical space within which a bus arriving ahead-of-schedule can wait until the scheduled departure time). At these time-points, the buses must abide by the ahead-of-schedule policy described in Section 1. Our analysis focuses on the scheduled departure times at these six time-points.

Though the current Werewolf A schedule includes two additional stops, they are located in the middle of the flow of traffic and therefore do not have the layover capacity necessary for buses to adhere to the ahead-of-schedule policy described in Section 1. Herein, scheduling the departure times for these non-time-point stops is simple: each scheduled departure time is set as the sum of the preceding stop's scheduled departure time and the minimum conceivable link time for the preceding link. In contrast, setting the scheduled departure time as any later time creates the risk that passengers that arrive on-time to board at this stop will miss their bus.[4] Because of the different process by which their scheduled departure times are determined, we exclude these two non-time-point stops from our analysis.

[4]For routes with small headways, it is possible for passengers that have missed their intended bus to board a subsequent bus without incurring large delay. In such cases, planners may knowingly select a schedule that can give rise to buses departing ahead-of-schedule because it nevertheless minimizes average total trip time. The forty-minute headway of the Werewolf A route, however, seems to be significantly beyond the threshold below which such practice is realistic.

The Werewolf A route traverses urban arterial, residential, and campus roads, including numerous signalized and unsignalized intersections. Though passenger demand may vary throughout the evening (with higher volumes typically occurring earlier), the purposes of the majority of trips can be presumed to be relatively similar. Werewolf A passengers likely consist largely of students or faculty leaving campus to return to their apartments or homes or traveling to a parking lot to retrieve their cars.

The Werewolf A route includes several significant sources of variability for link times. Buses arrive at the signalized intersections on Western Boulevard, Avent Ferry Road, and Gorman Road at different points during their signal cycles, some of which exceed 90 seconds. Earlier in the evening, many students and faculty are still departing from campus, and therefore passenger demand is higher during that time and boarding and alighting occurs at a greater proportion of stops. The route's initial cycles may also experience the effects of rush-hour congestion.[5]

In turn, the Werewolf A route experiences many deviations from the scheduled departure times. As illustration, for the days described by the historical data used in this project, over a quarter of all buses arrived more than three minutes before the scheduled departure time (2.3 %) or departed more than five minutes after the scheduled departure time (23.8 %).

# 4 Data description

TransLoc Incorporated, developer of the real-time bus locating system used by the NCSU Department of Transportation, provided historical data for the Werewolf A route from August 23, 2006, to October 24, 2006. Information, recorded through TransLoc's Transit Visualization System, included bus number, stop identification, date, arrival time, and departure time for every visit to a designated stop along the route. From this, dwell times at a stop and link times between consecutive stops are easily calculated.

To reduce the effect of missing information, the first and last two of each day's thirteen cycles are truncated. Still, some cycles or days have incomplete information, and

[5]Additional schedule information, including a route map, can be found in Appendix 1.

not all of the data can be utilized. In some cases, measurements have been missed as a result of malfunctions of the detection system. In other cases, malfunctioning vehicles or dispatcher errors have caused the bus to miss a stop or multiple stops and no measurements were taken. In theory, interpolation can be used to estimate this missing information, but because of its additional requisite assumptions, we reject this option. In the end, twenty-four days with complete ten-cycle data sets have been identified for use. A sample of the data is available in Appendix 2.

# 5 Literature review

Ceder (2002) classifies setting a departure schedule as one of the four basic components of transit operational planning, preceded chronologically by network route design and followed by vehicle scheduling and driver assignment. Additionally, Ceder asserts that, although the components should be considered simultaneously to increase overall efficiency, setting a departure schedule deserves particular attention because of its role as "one of the predominant bridges between the operator (and/or the community) and the passengers" [3].

Numerous objectives have been considered for the design of departure schedules, with varying assumptions about number of routes and number of transfer points, and passenger arrival rates and preferences. Chakroborty et al. (1998) consider origin-destination pairs spanning multiple routes and attempt to minimize the sum of initial delay and transfer time. Focusing on the departure times for a single stop on a single route, Palma and Lindsey (2001) assume that riders have different times at which they prefer to travel and seek to minimize the corresponding "penalties" incurred when the bus departs at another time (which by their assumption is exactly the scheduled departure time). Likewise, Wirasinghe (2003) includes a weight for initial delay and an additional penalty corresponding to the inconvenience created by the delay. Hickman (2001) investigates routes with small headways, assumes that passenger arrival rates are uniform, and seeks to determine the holding policy (whereby departures are intentionally delayed) that minimizes initial delay for passengers boarding at subsequent stops. Similarly, Hall et al.

(2001) study the optimal holding policy at transfer points when connecting buses have yet to arrive. In the results of both of the latter cases, total trip time will be increased for some passengers in return for the reduction of the total trip times of other passengers.

Furthermore, various techniques to search for optimal schedules have been employed. Genetic algorithms are among the most common. For instance, Chakroborty et al. (1998) adjust headway and holding times using a genetic algorithm that allows both cross-over and mutation. Alternatively, Palma and Lindsey (2001) extend models from location theory and think of passengers' preferred departure times and the scheduled departure times as physical "points," an approach that allows the optimal schedule (that which minimizes total "distance") to be determined analytically.

Studying transit systems requires assumptions about vehicle behavior. Many authors, including Chakroborty et al. (1998), assume link times to be constant. On the other hand, in their model, Anderson and Scalia-Tomba (1981) consciously include the variability inherent to link times, reflecting trends based on day of the week, time of day, and driver behavior, as well as incorporating an allowance for additional random variation.

Historical data has been utilized to more accurately represent the stochastic character of a transit system by Anderson and Scalia-Tomba (1981), Strathman et al. (2002), and Bertini and El-Geneidy (2004), among others. These studies, though, use the historical data to estimate input parameters for continuous distribution models, rather than applying the discrete values of the historical data directly. To our knowledge, no comparable attempt has been undertaken to compare proposed schedules, and thereby search for an optimal schedule, by using historical data in the order that it originally occurred.

# 6 Optimizaton

This section describes the development of the two steps of the optimization process: (*i*) a method to measure a passenger's total trip time, which corresponds to a mathematical formulation of the process; and (*ii*) a search algorithm which uses the measurement method to seek the optimal solution. For the measurement method and corresponding objective function, we first present an intuitive approach, then an alternative but equiv-

alent formulation that reduces computational complexity and gives rise to additional observations.

## 6.1 Model formulation

### 6.1.1 Notation for decision variables

For a route with $N$ time-points in a cycle, a schedule can be represented as a 1-by-$N$ solution vector, where each element gives the interval between the scheduled departure times of consecutive stops. For instance, for solution vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]$, $x_i$ is the interval between the scheduled departure times of time-points $s_{i-1}$ and $s_i$.[6]

By selecting these intervals, rather than departure times, as decision variables, we are able to use a single solution vector, coupled with an initial scheduled departure time $S_0$, to uniquely determine an entire schedule. For example, the current Werewolf A schedule can represented with $\mathbf{x_{current}} = [5, 11, 4, 7, 5, 8]$ and $S_0 =$ 5:55 PM. Now, with $C$ as the total scheduled cycle length and $M$ as the number of cycles per day, we can express the resulting schedule as an $M$-by-$N$ matrix $X$ of scheduled departure times,[7]

$$X = \begin{pmatrix} S_0 & S_0 + x_1 & \ldots & S_0 + \sum_{i=1}^{N-1} x_i \\ C + S_0 & C + S_0 + x_1 & \ldots & C + S_0 + \sum_{i=1}^{N-1} x_i \\ \vdots & \vdots & \ddots & \vdots \\ C(M-1) + S_0 & C(M-1) + S_0 + x_1 & \ldots & C(M-1) + S_0 + \sum_{i=1}^{N-1} x_i \end{pmatrix}.$$

Additionally, the schedule departure time $S_{j,m,y}$ for time-point $s_j$ during cycle $m$ on day $y$ is given by

$$S_{j,m,y} = C(m-1) + S_0 + \sum_{i=1}^{j} x_i. \tag{1}$$

For this project, we restrict all times to integers, reflecting the industry standard of publishing transit schedule times exclusively with integer numbers of minutes.

[6] It should be noted that the time-points are indexed from $s_0$ to $s_{N-1}$ and that $x_N$ gives the interval between scheduled departure times for $S_{N-1}$ and $S_0$.

[7] This example can be compared with the current Werewolf A schedule reproduced in Appendix 1.

### 6.1.2 A deterministic simulation

The basis for our measurement of total trip time is a deterministic simulation that combines historical data and the schedule constructed from the current iteration's solution vector. For this simulation, link times are taken directly from the historical data in the order that they originally occurred. In general, though, arrival and departure times in the simulation are not the same as those in the original data because a different schedule is in place, causing buses to arrive ahead-of-schedule (and therefore to idle until the scheduled departure time) at different time-points. In turn, for each new solution vector, the deterministic simulation yields a new data set of arrival and departure times.

Because buses that arrive ahead-of-schedule incur artificially increased dwell times caused by layovers, historical data does not afford an ordered list of dwell times comparable to that of link times. Because we have assumed that boarding passengers arrive exactly at the scheduled departure time, the number of passengers waiting to board an arriving bus is independent of the arrival time of the bus. Therefore, as a simple compromise, we assume that dwell times are constant. Since dwell time is directly related to the number of boarding and alighting passengers [1], this is consistent with our assumption that trips are uniformly distributed over all origin-destination pairs (which implies that the number of boarding and alighting passengers is constant over all time-points).[8] The 85th-percentile of a distribution of dwell times recorded at a Werewolf A stop without layover capacity ($d_{j,m,y} \equiv 0.8$ minutes) is employed.

Table 1 gives an example of the deterministic solution for a single cycle. Here, link times $\hat{l}_i$ have been taken from historical data from August 27, 2006, beginning at 7:15 PM. The arrival times ("AT") and departure times ("DT") resulting from two different schedules, corresponding to $\mathbf{x_1} = [6, 7, 7, 7, 7, 6]$ and $\mathbf{x_2} = [5, 12, 3, 10, 6, 4]$, have been calculated. In Table 1, "Simulation AT" gives the arrival time which would occur if the given schedule were in place and the link times were the same. "Natural DT" gives the sum of the simulation arrival time and the dwell time needed for passengers to board and alight. According to the ahead-of-schedule policy described in Section 1,

[8] If information about passenger behavior is available, in addition to updating the origin-destination pairs matrix $P$, it can be used to modifye the assumed values for dwell times accordingly.

"Simulation DT" gives the maximum of the scheduled and natural departure times. In the "Deviation" column, early arrivals are indicated by negative numbers, and tardy departures are indicated by positive numbers. Dwell times are assumed to be constant at 0.8 minutes (0:00:48).

Table 1: An example of the deterministic simulation.

| | time-point $i$ | 1 | 2 | 3 | 4 | 5 | 0 |
|---|---|---|---|---|---|---|---|
| | link time $\hat{l}_i$ | 0:06:10 | 0:09:13 | 0:04:27 | 0:07:10 | 0:02:49 | 0:05:57 |
| $\mathbf{x_1}$ | Scheduled DT | 7:21:00 | 7:28:00 | 7:35:00 | 7:42:00 | 7:49:00 | 7:55:00 |
| | Simulation AT | 7:21:10 | 7:31:11 | 7:36:26 | 7:44:24 | 7:48:01 | 7:54:57 |
| | Natural DT | 7:21:58 | 7:31:59 | 7:37:14 | 7:45:12 | 7:48:49 | 7:55:45 |
| | Simulation DT | 7:21:58 | 7:31:59 | 7:37:14 | 7:45:12 | 7:49:00 | 7:55:45 |
| | Deviation | +0:00:58 | +0:03:59 | +0:02:14 | +0:03:12 | -0:00:11 | +0:00:45 |
| $\mathbf{x_2}$ | Scheduled DT | 7:20:00 | 7:32:00 | 7:35:00 | 7:45:00 | 7:51:00 | 7:55:00 |
| | Simulation AT | 7:21:10 | 7:31:11 | 7:36:27 | 7:44:25 | 7:48:02 | 7:56:57 |
| | Natural DT | 7:21:58 | 7:31:59 | 7:37:15 | 7:45:13 | 7:48:50 | 7:57:45 |
| | Simulation DT | 7:21:58 | 7:32:00 | 7:37:15 | 7:45:13 | 7:51:00 | 7:57:45 |
| | Deviation | +0:01:58 | -0:00:01 | +0:02:15 | +0:00:13 | -0:02:10 | +0:02:45 |

It is important to note that the total trip time $t_{ijmy}$ is a function of the current solution $\mathbf{x} = [x_1, x_2, \ldots, x_N]$ by the following: The current solution gives rise to a schedule. Because the link times $l_{j,m,y}$ and dwell times $d_{j,m,y}$ are pre-set, the schedule determines the time-points at which layovers occur (indicated in Table 1 by negative deviations), as well as the time-points at which the bus departs behind-schedule (indicated in Table 1 by positive deviations). The length of these layovers and the extent of the late departures significantly affect a passenger's total trip time. The nonlinear relationship between arrival times $A_{j,m,y}$ and solution vector $\mathbf{x}$ is described in (2) and (3).

$$A_{j,m,y} = \max\left(A_{j-1,m,y} + d_{j-1,m,y}, S_{j-1,m,y}\right) + l_{j,m,y} \tag{2}$$

where

$$S_{j,m,y} = C(m-1) + S_0 + \sum_{i=1}^{j} x_i. \tag{3}$$

### 6.1.3 Explicit calculation

Following an intuitive approach, the total trip time $t_{ijmy}$ during cycle $m$ on day $y$ is then computed by subtracting the most recent scheduled departure time from origin time-point $s_i$ from the simulation arrival time at destination time-point $s_j$. We can express this computation as

$$t_{ijmy} = \begin{cases} A_{j,m,y} - S_{i,m,y}, & \text{if } i < j \\ A_{j,m+1,y} - S_{i,m,y}, & \text{if } i > j, \end{cases} \tag{4}$$

where $A_{i,m,y}$ is the simulation arrival time at $s_i$ and $S_{j,m,y}$ is the scheduled departure time from $s_j$, both during cycle $m$ on day $y$.

Using this approach, we formulate a mathematical program to minimize the average total trip time for passengers over all possible origin-destination pairs. This formulation, presented in Figure 1, involves the explicit calculation of total trip time given in (4). Here, $Y$ is the number of days being considered, $M$ is the number of cycles considered from a single day, $t_{ijmy}$ is the total trip time from $s_i$ to $s_j$ beginning in cycle $m$ on day $y$, and $p_{ij}$ is the probability that a given passenger travels from $s_i$ to $s_j$.

The relationship between the solution vector and the objective function is recapped in (8) through (10). A fixed cycle length, which may in some cases be desirable in order to preserve a schedule that yields times that allow the schedule's minute component to repeat, is specified in (11).[9] The decision variables are limited to feasible solutions in (12) and (13). It is significant to note that the objective function value for this formulation corresponds to the average total trip time (in minutes) over all origin-destination trip pairs.

For this test case, we assume that all origin-destination trip pairs are uniformly distributed, and therefore

[9]Relaxation of this constraint is discussed in the Section 8.3.

$$p_{ij} \equiv \frac{1}{N(N-1)} \ \forall \ i, j. \tag{5}$$

Also, our historical data has $Y = 24$ and $M = 10$. We retain $C = 40$ minutes, the current cycle length for the Werewolf A route.[10]

---

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{6}$$

where

$$f(\mathbf{x}) = \frac{1}{YMN} \sum_{y=1}^{Y} \sum_{m=1}^{M} \sum_{i=0}^{N-1} \sum_{j \neq i}^{N-1} t_{ijmy} \times p_{ij}, \tag{7}$$

$$t_{ijmy} = \begin{cases} A_{j,m,y} - S_{i,m,y}, & \text{if } i < j \\ A_{j,m+1,y} - S_{i,m,y}, & \text{if } i > j, \end{cases} \tag{8}$$

$$A_{j,m,y} = \max\left(A_{j-1,m,y} + d_{j-1,m,y}, S_{j-1,m,y}\right) + l_{j,m,y}, \tag{9}$$

and

$$S_{j,m,y} = C(m-1) + S_0 + \sum_{i=1}^{j} x_i. \tag{10}$$

subject to

$$\sum_{i=1}^{N} x_i = C, \tag{11}$$

$$x_i \geq 0, \ \text{for } i = 1, ..., N, \tag{12}$$

and

$$x_i \text{ integer}, \ \text{ for } i = 1, ..., N. \tag{13}$$

---

Figure 1: Formulation of a mathematical program that minimizes average total trip time using explicit calculation.

[10]The Matlab code for this measurement process can be found in Appendix 3.

A drawback of this formulation is its requisite computational complexity. Upon each arrival, the scheduled departure times of all other time-points must be considered. Therefore, a single evaluation of the objective function has complexity $O(YMN^2)$.[11] Though this process is still manageable for small routes, such as our test case ($N = 6$), for longer routes, computational demands increase significantly, particularly when combined with a search algorithm. When evaluating networks consisting of multiple routes, where transfers must be considered, the process could become computationally infeasible.

### 6.1.4 Calculation with weighted deviations

To reformulate this problem in a way that reduces computational demands, we separate total trip time into individual components: initial delay; link times; non-layover dwell times; and layover times. This separation is shown in Figure 2.

$$t_{ijmy} = w_i + \sum_{k=i+1}^{j} l_k + \sum_{k=i+1}^{j-1} (d_k + h_k) \text{ for } m = 1, \ldots, M \text{ and } y = 1, \ldots, Y, \qquad (14)$$

where

$t_{ijmy}$ is total trip time from origin $s_i$ to destination $s_j$ during cycle $m$ on day $y$,
$w_i$ is initial delay at $s_i$,
$l_k$ is the link time from $s_{k-1}$ to $s_k$, and
$d_k$ and $h_k$ are the non-layover dwell time and layover time at $s_k$.

Figure 2: Calculation of total trip time from separate components.

Two of these quantities, link times and non-layover dwell times, are independent of the schedule: assuming that short-term traffic changes are negligible, the necessary time for a bus to travel from time-point $s_{k-1}$ to time-point $s_k$ and for passengers to board and alight at $s_k$ is independent of the interval between scheduled departure times of these consecutive time-points. For a given cycle and day, changing $x_k$ will not affect the link time $l_k$ or non-layover dwell time $h_k$. Therefore, we simply need to track initial delays

[11] Recall that $N$ = number of time-points, $Y$ = number of days, and $M$ = number of cycles.

and layover times, each of which involves deviation from the scheduled departure time.[12]

Simple minimization of the deviations from the scheduled departure times, though, fails to reflect the difference between the two quantities. A passenger will experience at most one initial delay. However, depending on the length of his or her trip, a passenger may experience multiple layover times.

Nevertheless, we can utilize the deviations from the scheduled departure times to measure total trip time by applying appropriate weights that reflect the number of passengers affected by the deviation at a particular time-point. The weights for a particular time-point come from the distribution of origin-destination pairs $p_{kj}$: the tardiness weight $t_k$ is simply the proportion of all passengers whose trip originates at stop $s_k$; the earliness weight $e_k$ is the proportion of passengers that are continuing on-board at $s_k$ but neither board nor alight at $s_k$. This new measurement method gives rise to the formulation in Figure 3.

The constraints in (23) through (25) are the same as in the previous formulation. The quantities, $S_{k,m,y}$, $A_{k,m,y}$, and $D_{k,m,y}$, are the scheduled departure, simulation arrival, and simulation departure times at stop $s_k$ in cycle $m$ on day $y$. The quantity $d_{k,m,y}$ is a reasonable estimate of what the non-layover dwell time would have been had the layover not occurred (for our analysis, $d_{k,m,y} \equiv 0.8$), while $A_{k,m,y} + d_{k,m,y}$ is the natural departure time. The deviations from the scheduled departure times in (17) are described by the following notation

$$z^+ = \begin{cases} z, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0. \end{cases} \tag{15}$$

It should be noted that the new objective function value corresponds to the sum of the average initial delay and average layover time delay per passenger trip.

Computationally, this weighted deviations measurement method only needs to consider the current time-point and its comparison to the corresponding scheduled departure time, reducing the requisite complexity of evaluating the objective function from

[12]Recall that initial delay is the difference between actual and scheduled departure times. Layover time is the difference between natural and scheduled departure times.

$$\min_{\mathbf{x}} f_{wd}(\mathbf{x}) \tag{16}$$

where

$$f_{wd}(\mathbf{x}) = \frac{1}{YM}\sum_{y=1}^{Y}\sum_{m=1}^{M}\sum_{k=0}^{N-1} e_k \times (S_{k,m,y} - A_{k,m,y} - d_{k,m,y})^+ + t_k \times (D_{k,m,y} - S_{k,m,y})^+, \tag{17}$$

$$t_{ijmy} = \begin{cases} A_{j,m,y} - S_{i,m,y}, & \text{if } i < j \\ A_{j,m+1,y} - S_{i,m,y}, & \text{if } i > j, \end{cases} \tag{18}$$

$$A_{j,m,y} = \max\left(A_{j-1,m,y} + d_{j-1,m,y}, S_{j-1,m,y}\right) + l_{j,m,y}, \tag{19}$$

$$S_{j,m,y} = C(m-1) + S_0 + \sum_{i=1}^{j} x_i. \tag{20}$$

$$e_k = \sum_{i=2}^{N-1}\sum_{j=1}^{i-1} p_{(k+i) \bmod N,(k+j) \bmod N}, \tag{21}$$

and

$$t_k = \sum_{j=1}^{N} p_{kj}, \tag{22}$$

subject to

$$\sum_{i=1}^{N} x_i = C, \tag{23}$$

$$x_i \geq 0, \text{ for } i = 1, ..., N, \tag{24}$$

and

$$x_i \text{ integer}, \text{ for } i = 1, ..., N. \tag{25}$$

Figure 3: Formulation of a mathematical program that minimizes average total trip time using weighted deviations.

$O(YMN^2)$ to $O(YMN)$.

### 6.1.5 Discussion on using weighted deviations

Both $S_{k,m,y} - A_{k,m,y} - d_{k,m,y}$ (earliness) and $D_{k,m,y} - S_{k,m,y}$ (tardiness) correspond to deviations from the scheduled departure times. The multiplication of these deviations from the scheduled departure times by the weights, $e_k$ and $t_k$, points to the different effects of earliness and tardiness. In general, being one minute early and being one minute late are not equivalent, because a different number of passengers are affected by earliness and tardiness respectively. If a deviation from the scheduled departure time is to occur, it is preferable for the deviation to affect the smallest number of passengers as possible. For instance, if $e_k > t_k$ (that is, at time-point $s_k$, the number of passengers continuing on-board exceeds the number of passengers boarding), more passengers will be affected by earliness, and therefore tardiness is preferable to earliness.

When origin-destination trip pairs are assumed to be uniformly distributed, the earliness and tardiness weights, $e_k$ and $t_k$, both depend on the route's number of time-points $N$. In such a case, the definition of $t_k$ in (22) illustrates that

$$t_k = \frac{1}{N}, \text{ for } k = 0, ..., N-1. \tag{26}$$

It can also be shown that

$$e_k = \frac{N-2}{2N}, \text{ for } k = 0, ..., N-1. \tag{27}$$

Further explanation of (27) is included in Appendix 4.

For our test case, since $N = 6$, (5), (26), and (27) give $p_{ij} \equiv \frac{1}{30}$, $t_k = \frac{1}{6}$, and $e_k = \frac{1}{3}$. Since $e_k > t_k$, if a deviation from the scheduled departure time is to occur, we prefer tardiness to earliness. At first glance, it seems as though we may now be able to conclude that the scheduled cycle length corresponding to the optimal schedule will be less than the sum of the means of the link and dwell times. However, the scheduled cycle length constraint in (23), as well as this rationale's failure to consider the detrimental effects of tardiness at the beginning of a cycle, prevent us from making definitive conclusions

about such a relationship.

When origin-destination trip pairs are not assumed to be uniformly distributed, calculation of $p_{ij}$, $t_k$, and $e_k$ at individual time-points can suggest which deviation is preferred at those time-points. As in the previous case, for an individual time-point, if $e_k > t_k$, tardiness is preferred to earliness. Likewise, if $e_k < t_k$, earliness is preferred to tardiness.

## 6.2 Modified Steepest Descent search algorithm

Using the weighted deviations method to measure total trip time for a given solution and thereby to compare different solutions, we seek the optimal solution by applying a modification of the Steepest Descent algorithm developed by Curry (1944). Instead of finding the direction which yields the greatest local improvement (the gradient) as Curry originally proposed, we find the adjacent solution[13] that yields the greatest improvement. We continue until we reach a solution whose objective function value is lower than that of all adjacent solutions (a local, possibly global minimum) or until a predetermined maximum number of iterations are performed. Figure 4 summarizes this Modified Steepest Descent (MSD) algorithm.

In cases where historical data is available, it is likely that a schedule will already be in place. It is natural, when possible, to initialize the algorithm with the solution corresponding to this existing schedule.

Table 2 summarizes a single iteration of the Modified Steepest Descent (MSD) algorithm. In the "Solution" column, italics are added to the adjusted elements. In the "Cost" column, italics are added when a new best cost is achieved.[14] This iteration began with $\mathbf{x_{init}} = [5, 11, 4, 7, 5, 8]$. In Table 2, the iteration progresses from left to right, then down.

The combination of the weighted deviations measurement method and the MSD

[13] Adjacent solutions have exactly two elements that differ by exactly one minute with each solution having exactly one element that is greater than the same element in the other solution. An adjacent solution can be obtained by lowering $x_i$ by one minute and increasing $x_j$ by one minute, where $i \neq j$. For instance, $\mathbf{x_1} = [x_1, x_2, ..., x_i, ....x_j, ...x_n]$ and $\mathbf{x_2} = [x_1, x_2, ..., x_i + 1, ....x_j - 1, ...x_n]$ are adjacent.

[14] Costs, describing the average sum of initial delay and non-layover dwell times per passenger, are given in minutes.

(1) Select initial solution $\mathbf{x_{init}}$ and measure its objective function.

(2) Evaluate the objective function at all feasible adjacent solutions.

(3) Keep the solution $\mathbf{x_{new}}$ which has the lowest objective function.

(4) If $f_{wd}(\mathbf{x_{new}}) > f_{wd}(\mathbf{x_{previous}})$, stop. Select $\mathbf{x_{previous}}$ as the best solution.

(5) Otherwise, return to step 2.

Figure 4: The Modified Steepest Descent (MSD) algorithm.

Table 2: Summary of a single iteration of the Modified Steepest Descent algorithm.

| Solution | Cost | Best Previous Cost | Solution | Cost | Best Previous Cost |
|---|---|---|---|---|---|
| [*6*, *10*, 4, 7, 5, 8] | *2.450* | 2.450 | [*6*, 11, *3*, 7, 5, 8] | 2.830 | 2.450 |
| [*6*, 11, 4, *6*, 5, 8] | 2.664 | 2.450 | [*6*, 11, 4, 7, *4*, 8] | 2.519 | 2.450 |
| [*6*, 11, 4, 7, 5, *7*], | *2.438* | 2.438 | [*4*, *12*, 4, 7, 5, 8] | 2.708 | 2.438 |
| [5, *12*, *3*, 7, 5, 8] | 2.933 | 2.438 | [5, *12*, 4, *6*, 5, 8] | 2.767 | 2.438 |
| [5, *12*, 4, 7, *4*, 8] | 2.621 | 2.438 | [5, *12*, 4, 7, 5, *7*] | 2.541 | 2.438 |
| [*4*, 11, *5*, 7, 5, 8] | 2.513 | 2.438 | [5, *10*, *5*, 7, 5, 8] | *2.353* | 2.353 |
| [5, 11, *5*, *6*, 5, 8] | 2.396 | 2.353 | [5, 11, *5*, 7, *4*, 8] | *2.291* | 2.291 |
| [5, 11, *5*, 7, 5, *7*] | *2.260* | 2.260 | [*4*, 11, 4, *8*, 5, 8] | 2.669 | 2.260 |
| [5, *10*, 4, *8*, 5, 8] | 2.510 | 2.260 | [5, 11, *3*, *8*, 5, 8] | 2.715 | 2.260 |
| [5, 11, 4, *8*, *4*, 8] | 2.446 | 2.260 | [5, 11, 4, *8*, 5, *7*] | 2.415 | 2.260 |
| [*4*, 11, 4, 7, *6*, 8] | 2.775 | 2.260 | [5, *10*, 4, 7, *6*, 8] | 2.616 | 2.260 |
| [5, 11, *3*, 7, *6*, 8] | 2.861 | 2.260 | [5, 11, 4, *6*, *6*, 8] | 2.695 | 2.260 |
| [5, 11, 4, 7, *6*, *7*] | 2.520 | 2.260 | [*4*, 11, 4, 7, 6, *9*] | 2.824 | 2.260 |
| [5, *10*, 4, 7, 5, *9*] | 2.665 | 2.260 | [5, 11, *3*, 7, 5, *9*] | 2.945 | 2.260 |
| [5, 11, 4, *6*, 5, *9*] | 2.779 | 2.260 | [5, 11, 4, 7, *4*, *9*] | 2.633 | 2.260 |

algorithm has the advantage of easily adapting to non-uniform distribution of origin-destination pairs by making appropriate changes in the calculation of the earliness and tardiness weights, $e_k$ and $t_k$.

# 7 Results

Table 3 shows the progression of the MSD algorithm when the solution corresponding to the existing Werewolf A schedule is used as $\mathbf{x_{init}} = [5, 11, 4, 7, 5, 8]$. In the "Solution" column, italics are added to the elements that have been adjusted relative to the previous iteration.

Table 3: Searching for the optimal schedule with the Modified Steepest Descent algorithm.

| Iteration | Solution | Cost | Best Previous Cost |
|---|---|---|---|
| 0 | [5, 11, 4, 7, 5, 8] | 2.549 | N/A |
| 1 | [5, 11, *5*, 7, 5, *7*] | 2.260 | 2.549 |
| 2 | [*6*, *10*, 5, 7, 5, 7] | 2.161 | 2.260 |
| 3 | [6, 10, *6*, 7, *4*, 7] | 2.092 | 2.260 |
| 4 | [*7*, *9*, 6, 7, 4, 7] | 2.074 | 2.092 |
| 5 | [7, *10*, 6, 7, 4, *6*] | 2.085 | 2.074 |

In this instance, we identify a best solution, $\mathbf{x} = [7, 9, 6, 7, 4, 7]$, after just five iterations. Assessment of the optimality of this solution can be found in Section 9.1. As Figure 4 specifies, the algorithm terminates when the new cost exceeds the best previous cost.

# 8 Sensitivity analysis

It is desirable to have a robust algorithm whose results do not change significantly when the parameters are altered slightly. With this in mind, we investigate the separate effects of altering three different parameters.

## 8.1 Dwell time

To this point, we have assumed constant non-layover dwell times of $d_{j,m,y} \equiv 0.8$ minutes.[15] In practice, though, different estimates may be deemed more appropriate. Table 4 gives the solution identified as best and its corresponding objective function value for different values of constant dwell time, as well as for a scenario where dwell times are assumed to be constant for a given time-point but are different from time-point to time-point. The dwell times for the latter case are obtained from the 85th-percentiles of the historical distributions of dwell times at each of the respective time-points, excluding instances where a layover occurred due to the bus arriving ahead-of-schedule.

Table 4: Sensitivity analysis for alternative dwell time assumptions.

| Dwell Time (min) | Best Solution | Cost |
|---|---|---|
| 0.50 | [7, 9, 6, 7, 4, 7] | 2.400 |
| 0.60 | [7, 9, 6, 7, 4, 7] | 2.259 |
| 0.70 | [7, 9, 6, 7, 4, 7] | 2.139 |
| 0.80 | [7, 9, 6, 7, 4, 7] | 2.074 |
| 0.90 | [7, 9, 6, 7, 4, 7] | 2.067 |
| 1.00 | [7, 9, 6, 8, 4, 6] | 2.142 |
| 1.10 | [7, 9, 6, 8, 4, 6] | 2.356 |
| $[1.83, 1.30, 1.32, 0.75, 0.85, 0.80]$ | [8, 10, 6, 7, 4, 5] | 2.504 |

Though a different solution is identified as best when dwell time is assumed to be one minute or greater, only 7 % of dwell times from a non-time-point stop in the historical data reach this length. On the other hand, when lower and more common dwell times are assumed as the constant value, the solution identified as best remains unchanged.

Because dwell times at a given time-point are directly related to the number of boarding and alighting passengers, the distribution of origin-destination trip pairs can be used to estimate dwell times for different time-points. The results corresponding to the latter

[15] Please refer to Section 6.1.2 for the justification.

set of assumed dwell times in Table 4 illustrate that passenger information should be utilized when it is available.

## 8.2 Starting solutions

When there is a risk of converging to a non-global local optimum, the starting solution has a significant effect on the optimality of the solution that the algorithm identifies as best. Therefore, we repeat the MSD algorithm using a diverse set of starting solutions. The results are summarized in Table 5.

Table 5: Sensitivity analysis for different starting solutions.

| Starting Vector | Best Solution | Iterations |
|---|---|---|
| $[40, 0, 0, 0, 0, 0]$ | [7, 9, 6, 7, 4, 7] | 34 |
| $[0, 0, 0, 0, 0, 40]$ | [7, 9, 6, 7, 4, 7] | 34 |
| $[10, 10, 0, 0, 10, 10]$ | [7, 9, 6, 7, 4, 7] | 14 |
| $[6, 6, 7, 7, 7, 7]$ | [7, 9, 6, 7, 4, 7] | 5 |
| $[11, 5, 7, 4, 8, 5]$ | [7, 9, 6, 7, 4, 7] | 10 |
| $[0, 0, 20, 20, 0, 0]$ | [7, 9, 6, 7, 4, 7] | 28 |
| $[10, 3, 5, 5, 7, 10]$ | [7, 9, 6, 7, 4, 7] | 10 |

We note that for our Werewolf A test case, the MSD search algorithm identifies the same solution, $\mathbf{x} = [7, 9, 6, 7, 4, 7]$, as best for all attempted starting solutions. Though demonstrating the convergence properties of the MSD search algorithm is beyond the scope of this project, this sample of empirical results suggests that the space of feasible solutions may be convex.

## 8.3 Cycle length

Heretofore, we have set the cycle length to forty minutes, which matches the current Werewolf A route scheduled cycle length. This cycle length has the benefit of repeating the minute components of scheduled departure times every two hours (every three cycles),

desirable for a transit schedule serving a diverse clientele. Still, it is of interest to compare the results obtained when different cycle lengths are considered. If a significantly lower objective function value is achievable with a different cycle length, it may be worthwhile to consider relaxing our constraints. Computationally, we can evaluate this situation simply by modifying the constraint in (23). Table 6 summarizes the results.

It is significant to note that, for our test case, the objective function value corresponding to a forty-minute cycle length is the lowest even when other cycle lengths are permitted.

Table 6: Sensitivity analysis for different cycle lengths.

| Cycle Length | Best Solution | Cost |
|---|---|---|
| 35 | [7, 9, 5, 7, 4, 3] | 8.492 |
| 36 | [7, 9, 5, 7, 4, 4] | 5.494 |
| 37 | [7, 9, 5, 7, 4, 5] | 3.489 |
| 38 | [7, 9, 6, 7, 4, 5] | 2.492 |
| 39 | [7, 9, 6, 7, 4, 6] | 2.095 |
| *40* | *[7, 9, 6, 7, 4, 7]* | *2.074* |
| 41 | [7, 9, 6, 8, 4, 7] | 2.213 |
| 42 | [7, 10, 6, 8, 4, 7] | 2.439 |
| 43 | [8, 10, 6, 8, 4, 7] | 2.721 |
| 44 | [8, 10, 6, 8, 4, 8] | 3.030 |
| 45 | [8, 11, 6, 8, 4, 8] | 3.347 |

The MSD search algorithm could be adjusted to allow non-specified scheduled cycle lengths or any of a set of scheduled cycle lengths. "Adjacency" could be redefined to describe solutions differing by one minute at one element, rather than at two elements. In turn, each iteration would check only $2N$ solutions rather than $N(N-1)$ solutions, reducing the computational complexity of the search component of each iteration to $O(YMN)$.[16] However, by decreasing the set of alternative solutions tested at each it-

[16]This reduction in computational complexity is a natural reasonable consequence of relaxing the scheduled cycle length constraint in (23).

eration, we increase the chance that the algorithm will terminate at a non-global local optimum. For instance, under these modified constraints, with $\mathbf{x_{init}} = [5, 11, 4, 7, 5, 8]$, the algorithm terminates at $\mathbf{x} = [5, 11, 5, 7, 5, 7]$ with $f_{wd}(\mathbf{x}) = 2.261$, which exceeds $f_{wd}([7, 9, 6, 7, 4, 7]) = 2.074$.

# 9 Alternative methods, verification, and convergence

## 9.1 Exhaustive enumeration

These test case results can be verified by evaluating the objective function at all feasible solutions. A simple program[17] identifies the entire set of feasible solutions, as determined by the constraints listed in (23) through (25). For the Werewolf A route, the optimal solution identified by this exhaustive enumeration or "brute force" approach is the same as the best solution that MSD identifies. Therein, the optimality of the solution identified by MSD is established.

Table 7 gives the solutions with the ten smallest objective function values. Incidentally, the solution corresponding to the current schedule yields the 1809th smallest objective function out of 1,221,759 total feasible solutions.

Table 7: Ten lowest objective function values, with $C \equiv 40$, among all feasible solutions.

| Rank | Solution | Cost | Rank | Solution | Cost |
|---|---|---|---|---|---|
| 1 | [7, 9, 6, 7, 4, 7] | 2.074 | 8 | [6, 9, 6, 8, 4, 7] | 2.112 |
| 2 | [7, 10, 6, 7, 4, 6] | 2.084 | 9 | [7, 9, 6, 7, 5, 6] | 2.112 |
| 3 | [7, 9, 6, 8, 4, 6] | 2.086 | 10 | [6, 10, 5, 8, 4, 7] | 2.115 |
| 4 | [6, 10, 6, 7, 4, 7] | 2.092 | ⋮ | ⋮ | ⋮ |
| 5 | [8, 9, 6, 7, 4, 6] | 2.100 | 1809 | [5, 11, 4, 7, 5, 8] | 2.549 |
| 6 | [7, 10, 5, 8, 4, 6] | 2.100 | ⋮ | ⋮ | ⋮ |
| 7 | [7, 9, 5, 8, 4, 7] | 2.109 | 1221759 | [40, 0, 0, 0, 0, 0] | 18.329 |

[17]Please see Appendix 3 for the Matlab code used for exhaustive enumeration.

This exhaustive enumeration approach is guaranteed to identify the global optimum. However, for a route with $N$ time-points, it has computational complexity $\mathrm{O}(YMNC^N)$, which may be prohibitive for some routes or networks.

## 9.2 Random simulation

By considering the distributions of link times given by the historical data independent of the order in which they occurred, arbitrarily many days can be simulated. In the same manner as in previous total trip time measurement methods, a schedule can be constructed from the given solution vector, link times can be applied to the schedule, and layovers can be identified. Now, though, rather than utilizing the historically ordered list, link times are randomly selected. For instance, rather than considering all link times from a given cycle and day from the historical data in succession, the simulation may select the link time from cycle 3 on day 5 as $l_{1,1,1}$ and the link time from cycle 7 on day 12 as $l_{2,1,1}$.

A drawback of this process is that the correlation of link times near each other with respect to time of day is discarded, rendering the simulation less valid. Moreover, though this random simulation can evaluate the objective function in the same manner as the deterministic simulation, comparisons of objective function values are not necessarily repeatable. For instance, the random nature of the link times may yield $f_{wd}(\mathbf{x_1}) < f_{wd}(\mathbf{x_2})$ in one instance, but $f_{wd}(\mathbf{x_1}) > f_{wd}(\mathbf{x_2})$ in another. Consequently, even when identical starting solutions and historical data are used, the solution that a search algorithm identifies as best may not always be the same.

In theory, increasing the number of simulated days can decrease the probability of contradicting results like those mentioned in the previous paragraph by narrowing the range of possible objective function values for a given solution. The extent to which this range's width is decreased depends on the variance of the distribution of link times that is used. However, as Table 8 shows, even when 600 days are simulated at each iteration, the algorithm may identify different solutions as best on different runs.

It should be noted that this observation alone does not imply that random simulation causes the algorithm to not reach the global optimum. Applying different link times

on each run has the effect of creating new system behavior, different each time from the actual historical behavior. Therefore, using random simulation leaves the search algorithm equally likely to find the solution corresponding to the optimal schedule as when deterministic simulation is used. The unavoidable difficulty, however, is that the resulting schedule will be optimal for a system corresponding to behavior that has never actually occurred.

Nevertheless, comparisons with search algorithm results obtained when the random simulation is employed can provide verification to the search algorithm results obtained with the deterministic simulation. Table 8 summarizes results obtained by using the MSD algorithm and the random simulation for $Y = 600$ days. In each case, $\mathbf{x_{init}} = [5, 11, 4, 7, 5, 8]$ is used as the initial solution. The "Iteration Index" column gives the iteration during which the solution identified as best first appears.

Table 8: Ten different optimization runs using random simulation.

| Best Solution | Cost | Iteration Index |
|---|---|---|
| $[7, 9, 6, 7, 4, 7]$ | 1.878 | 5 |
| $[7, 9, 6, 7, 4, 7]$ | 1.872 | 5 |
| $[6, 10, 6, 7, 4, 7]$ | 1.876 | 6 |
| $[7, 9, 6, 7, 4, 7]$ | 1.872 | 5 |
| $[7, 9, 6, 7, 4, 7]$ | 1.871 | 5 |
| $[7, 9, 6, 7, 4, 7]$ | 1.879 | 5 |
| $[7, 9, 6, 7, 4, 7]$ | 1.871 | 5 |
| $[7, 9, 6, 7, 4, 7]$ | 1.877 | 5 |
| $[6, 10, 6, 7, 4, 7]$ | 1.876 | 4 |
| $[7, 9, 6, 7, 4, 7]$ | 1.885 | 5 |

The significant proportion (eight out of ten) of the runs using random simulation where the solution identified as best matches $\mathbf{x} = [7, 9, 6, 7, 4, 7]$ adds verification to the weighted deviation and MSD process.

The occurrence of cost functions that are lower than those measured by deterministic

simulation may arise from the loss of correlation of link times near each other with respect to time of day. With random simulation, longer link times that had previously all occurred during the earlier rush-hour cycles may now be distributed uniformly throughout the evening.

## 9.3 An alternative search algorithm: One-Up-One-Down

Computational complexity for each iteration can be decreased by checking less than every adjacent feasible solution (as the MSD algorithm does). For instance, we can limit our search to identifying the element with the greatest one-sided weighted deviation (early or tardy). This information can be implemented into an algorithm which maintains cycle length by adjusting that element (decreasing the element if the bus has been generally early or increasing it if the bus has been generally tardy) and a randomly-selected complementary element. The search component of each iteration of such a One-Up-One-Down (OUOD) algorithm has computational complexity $O(YMN)$. Additionally, because measurement is performed in series with the search component rather than as a part of a loop, each iteration's overall complexity is reduced from $O(YMN^3)$ to $O(YMN)$. The OUOD algorithm is described in further detail in Figure 5.

By randomly determining the complementary element, this algorithm has the potential to avert non-terminal periodicity around non-global optima. Moreover, because of its random component, the OUOD algorithm avoids the possibility of permanently converging to a non-global local optimum possessed by the MSD algorithm.

Table 9 summarizes a single iteration of this OUOD algorithm. Italics are added to the element corresponding to the greatest one-sided weighted deviation.

Table 10 shows the progression of the OUOD algorithm when the existing Werewolf A schedule, $\mathbf{x} = [5, 11, 4, 7, 5, 8]$, is used as the initial solution $\mathbf{x_{init}}$. In the "Solution" column, italics are added to the adjusted elements. In the "Cost" column, italics are added when a new best cost is achieved. We note that the OUOD algorithm yields the same best solution, $\mathbf{x} = [7, 9, 6, 7, 4, 7]$, as the MSD algorithm.

The OUOD algorithm's major drawback, however, is that because it does not check all adjacent solutions, it is possible that the optimal solution will never be reached even

1. Set *bestcost* to be a sufficiently large number (e.g., 1000).

2. Select initial solution $\mathbf{x_{init}}$.

3. Carry out deterministic simulation.

4. Identify the time-point $s_i$ with the greatest weighted deviation (earliness or tardiness).

5. If bus is tardy to time-point $s_i$ , $x_i = x_i + 1$. Randomly determine $1 - i < j < N - i$ but $j \neq 0$ for $x_{i+j} = x_{i+j} - 1$.
Else (if bus is early to time-point $s_i$), $x_i = x_i - 1$. Randomly determine $1 - i < j < N - i$ but $j \neq 0$ for $x_{i+j} = x_{i+j} + 1$.

6. Evaluate $f_{wd}(\mathbf{x_{new}})$. If $f_{wd}(\mathbf{x_{new}}) < bestcost$, set $bestcost = f_{wd}(\mathbf{x_{new}})$.

7. Return to step 1, using $\mathbf{x_{init}} = \mathbf{x_{new}}$.

8. Repeat for a fixed number of iterations, keeping track of the overall lowest objective function value and the corresponding solution.

Figure 5: The One-Up-One-Down (OUOD) algorithm.

Table 9: Summary of a single iteration of the One-Up-One-Down algorithm.

| | |
|---|---|
| Starting vector | [5, 11, 4, 7, 5, 8] |
| Corresponding cost | 2.549 |
| Weighted earliness vector | [0.015, *0.739*, 0.000, 0.044, 0.223, 0.609] |
| Weighted tardiness vector | [0.206, 0.067, 0.269, 0.220, 0.105, 0.052] |
| New solution vector | [5, *10*, *5*, 7, 5, 8] |
| Corresponding cost | 2.353 |

if a solution adjacent to the optimal solution is reached. In these cases, the elements that need to be changed in order to move to the optimal solution from an adjacent solution do not possess the greatest weighted deviation. Therefore, the algorithm never gets any closer than the adjacent solution.

Table 10: Searching for the optimal schedule with the One-Up-One-Down algorithm.

| Solution | Cost | Best Previous Cost | Solution | Cost | Best Previous Cost |
|---|---|---|---|---|---|
| $[5, 11, 4, 7, 5, 8]$ | 2.549 | N/A | $[5, \textit{10}, \textit{5}, 7, 5, 8]$ | 2.353 | 2.353 |
| $[5, 10, 5, \textit{8}, 5, \textit{7}]$ | 2.293 | 2.293 | $[5, \textit{9}, 5, 8, 5, \textit{8}]$ | *2.435* | 2.293 |
| $[5, \textit{10}, 5, 8, 5, \textit{7}]$ | 2.293 | 2.293 | $[5, \textit{9}, \textit{6}, 8, 5, 7]$ | 2.324 | 2.293 |
| $[5, 9, 6, 8, \textit{4}, \textit{8}]$ | 2.308 | 2.293 | $[5, 9, 6, \textit{9}, 4, \textit{7}]$ | 2.307 | 2.923 |
| $[\textit{6}, 9, 6, 8, \textit{4}, 7]$ | *2.112* | 2.112 | $[6, 9, \textit{7}, \textit{7}, 4, 7]$ | 2.134 | 2.112 |
| $[6, \textit{10}, \textit{6}, 7, 4, 7]$ | *2.092* | 2.092 | $*[\textit{7}, \textit{9}, 6, 7, 4, 7]*$ | *2.074* | 2.074 |
| $[7, \textit{8}, 6, \textit{8}, 4, 7]$ | *2.112* | 2.074 | $[\textit{8}, 8, 6, \textit{7}, 4, 7]$ | 2.179 | 2.074 |
| $[\textit{7}, \textit{9}, 6, 7, 4, 7]$ | *2.074* | 2.074 | $[7, \textit{8}, 6, \textit{8}, 4, 7]$ | *2.123* | 2.074 |
| $[7, 8, \textit{7}, \textit{7}, 4, 7]$ | *2.130* | 2.074 | $[\textit{8}, 8, \textit{6}, 7, 4, 7]$ | 2179 | 2.074 |
| $[\textit{7}, 8, 6, 7, \textit{5}, 7]$ | 2.167 | 2.074 | $[\textit{8}, 8, 6, 7, 6, \textit{6}]$ | 2.180 | 2.074 |

# 10 Recommendations

Our results suggest the revisions to the Werewolf Route A schedule given in Table 11.[18] Italics are added to the recommended schedule times. Whereas the current schedule yields an objective function value of $f(\mathbf{x_{current}}) = 2.549$ minutes, the recommended revisions yield an objective function value of $f(\mathbf{x_{opt}}) = 2.074$ minutes, an improvement of over 18 %. Furthermore, our analysis shows that if the recommended schedule were in place for the days considered, the proportion of buses arriving more than three minutes early or departing more than five minutes late would decrease from 26 % to 6 %.

# 11 Conclusions

Using historical data, we have created a method by which the efficiency of different schedules can be measured. Furthermore, in this project, we have implemented an algorithm capable of identifying the optimal schedule for the NCSU Wolfline Werewolf

[18]It should be noted that $S_0$ corresponds to D.H. Hill

Table 11: Current and recommended schedules for the Werewolf A route.

| Talley | Textiles | EC Lot | Gorman/Kings | Frat Ct | DH Hill |
|---|---|---|---|---|---|
| N/A | N/A | N/A | N/A | N/A | 5:55 PM |
| N/A | N/A | N/A | N/A | N/A | *5:55 PM* |
| 6:00 PM | 6:11 PM | 6:15 PM | 6:22 PM | 6:27 PM | 6:35 PM |
| *6:02 PM* | *6:11 PM* | *6:17 PM* | *6:24 PM* | *6:28 PM* | *6:35 PM* |
| 6:40 PM | 6:51 PM | 6:55 PM | 7:02 PM | 7:07 PM | 7:15 PM |
| *6:42 PM* | *6:51 PM* | *6:57 PM* | *7:04 PM* | *7:08 PM* | *7:15 PM* |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

A route. Assuming uniformly-distributed origin-destination trip pairs, to minimize passengers' average total trip time, the intervals between scheduled departures times of consecutive stops should be set to 7, 9, 6, 7, 4, and 7 minutes. An exhaustive enumeration approach establishes this solution's optimality. Moreover, sensitivity analysis shows that this solution is optimal even when different cycle lengths are permitted. Verification checks suggest that this measurement method and search algorithm may be able to be applied to any single fixed bus route with $N$ time-points.

# 12 Avenues for future research

Various extensions of this project come to mind:

**1. Non-uniform origin-destination distribution:** Computationally, non-uniform origin-destination distributions can be considered. The distribution could be explicitly determined by tracking passengers or could be estimated from other demographic information. Furthermore, knowing specifically where an alighting passenger had boarded would not be necessary; simply having a count at each stop of passengers who are continuing on-board (and therefore are affected by layover delays) would be sufficient.

Additionally, it could be of interest to investigate the robustness of the optimal solution with regard to changes in the origin-destination distribution.

**2. Networks with multiple routes:** Networks with multiple routes that afford passengers the opportunity to transfer between routes can be considered. Origin-destination pairs can be broadened to include origins and destinations on different routes, total trip time can be generalized to include transfer time, a solution vector can be formed by concatenating the vector of intervals between scheduled departure times for each of the routes, and a deterministic or random simulation can be simultaneously carried out for each of the routes. Within the search algorithm, the pairs or elements to be adjusted should be restricted to those from the same route.

Cases where there exist multiple paths to complete the trip determined by a single origin-destination pair introduce greater intricacy, possibly confronted by dynamically calculating trip proportions and earliness and tardiness weights.

A practical example of a multi-route network is the NCSU Wolfline Evening Service, consisting of the Werewolf A and Werewolf B routes. For instance, to travel from the College of Textiles to Carter-Finely Park and Ride, a passenger needs to transfer from the Werewolf A route to the Werewolf B route at D.H. Hill Library.

**3. Non-uniform dwell times:** In practice, dwell times are not constant. If no passengers are waiting to board or alight, the dwell time, even at a stop with layover capacity, may be zero. On the other hand, if a large number of passengers are boarding or alighting, dwell time may grow quite large. Therefore, benefit may be gained from investigating the effects upon our results of including variable dwell times. For instance, non-constant historical distributions of dwell times can be utilized during random simulation. Additionally, assumptions of constant dwell times can be varied to reflect and accommodate common phenomena such as drivers taking a break at the end of each cycle.

**4. Drivers that do not follow policy:** In practice, some drivers may not follow the standard policy to wait until the scheduled departure time before departing from a time-point. The extent to which this should be considered in schedule design is debatable. Some may argue that drivers simply need increased education about policies or increased accountability to adhere to policies. It is possible, however, to reflect this uncertainty by having the simulation, deterministic or random, randomly not follow the policy a certain

proportion of the time. It is also possible to restrict policy non-adherence to a subset of time-points.

**5. Routes served by multiple buses:** In the case where sufficiently large headways are maintained, our process will apply equally well to routes served by multiple buses. Historical data for a single bus can be isolated and the process carried out as if there were only one vehicle. Then, the schedule identified as best for this single bus can be repeated periodically to reflect the additional buses.

Additionally, in cases where headways are small enough that it is possible for an optimal schedule to coincide with instances where buses fall one headway or more behind-schedule, the measurement method can be adjusted, such that deviations are calculated by considering the soonest-arriving bus after a scheduled departure time, rather than assuming a one-to-one relationship between passengers arriving at particular scheduled departure times and actual bus arrivals.

**6. Modification of the search algorithm:** The two algorithms proposed by this project, Modified Steepest Descent (MSD) and One-Up-One-Down (OUOD), each have different advantages. MSD has a greater probability of identifying the global optimum, while for each iteration, OUOD is computationally less complex. Combinations of these two algorithms may yield increased performance. For instance, OUOD could be run, then MSD applied to ensure that the search has not stalled at a solution that is adjacent to the optimal solution.

# References


[1] Anderson, P. and Scalia-Tomba P. (1981). "A mathematical model of an urban bus route." *Transportation Research B*, 15B(4), 249-266.

[2] Bertini, R. and El Geneidy, A. (2004). "Modeling transit trip time using archived bus dispatch system data." *Journal of Transportation Engineering*, 56-67.

[3] Ceder, A. (2002). "Urban transit scheduling: framework, review and examples." *Journal of Urban Planning and Development*, 225-244.

[4] Chakroborty, P., Deb K., and Srinivas B. (1998). "Network-wide optimal scheduling of transit systems using genetic algorithms." *Computer-Aided Civil and Infrastructure Engineering*, 13, 363-376.

[5] Curry, H. (1944). "The method of steepest descent for non-linear minimization problems." *Quarterly of Applied Mathematics*, 2, 259-261.

[6] Hall, R., Dessouky, M., and Lu, Q. (2001). "Optimal holding times at transfer stations." *Computers and Industrial Engineering*, 379-397.

[7] Hickman, M. (2001). "An Analytic Stochastic Model for the Transit Vehicle Holding Problem." *Transportation Science*, 35(3), 215-237.

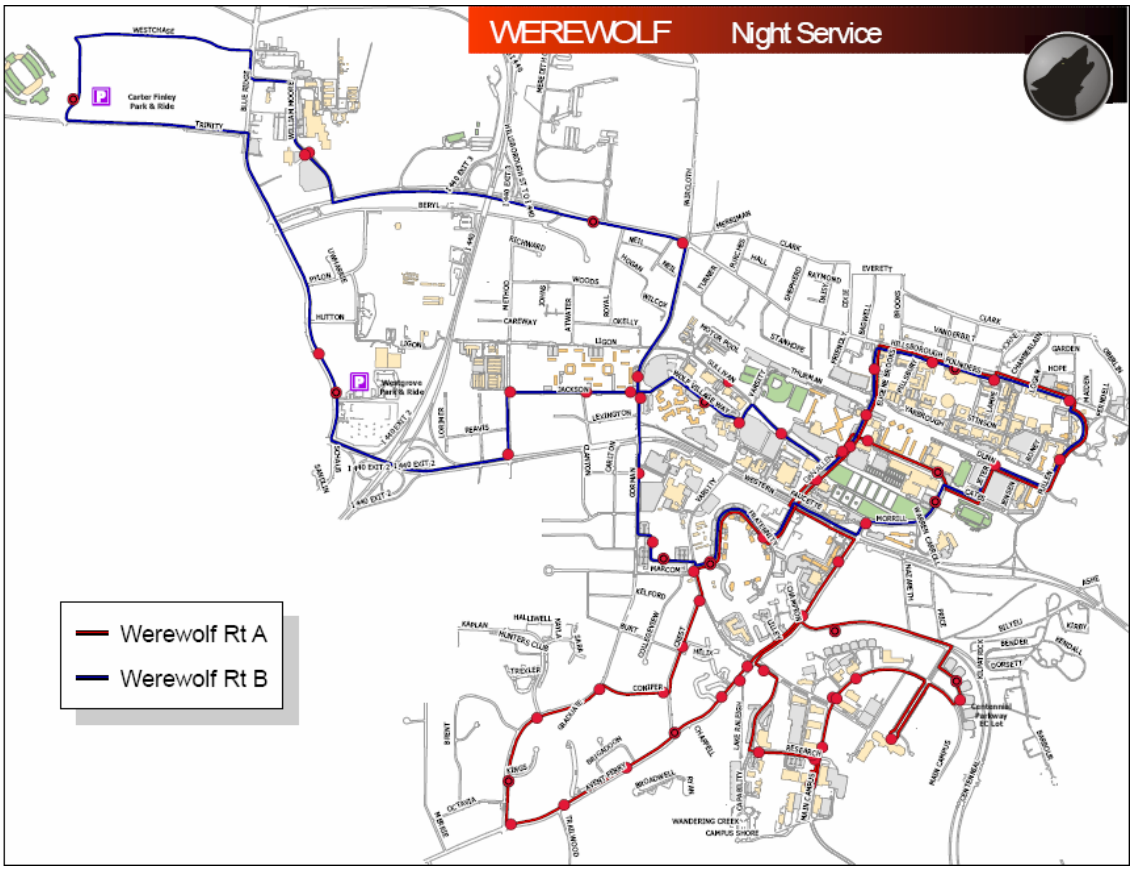
[8] Palma, A. and Lindsey, R. (2001)."Optimal timetables for public transportation." *Transportation Research B*, 35B, 789-813.

[9] Strathman, J. et al. (2002). "Evaluation of transit operations: data applications of Tri-Met's automated Bus Dispatching System." *Transportation*, 29, 321-345.

[10] Wirasinghe, S. (2003). "Initial Planning for Urban Transit Systems." *Advanced Modeling for Transit Operations and Service Planning*, W. Lam and M. Ball, eds., Elsevier, Oxford, 1-29, ISBN 0-08-044206-4.

# APPENDICES

## Appendix 1: Map and current schedule of the Werewolf A route




| SPRING 2007 TIME TABLE | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Werewolf A** | | | | | | | |
| Talley Student Center | Avent Ferry / Centennial Pkwy | College of Textiles | EC Lot (Cent. Pkwy) | Avent Ferry / Crest Rd | Gorman / King's Ct | Frat. Ct / Varsity Dr | DH Hill Library |
| | | | | | | | 5:55 PM |
| 6:00 PM | 6:05 PM | 6:11 PM | 6:15 PM | 6:18 PM | 6:22 PM | 6:27 PM | 6:35 PM |
| 6:40 PM | 6:45 PM | 6:51 PM | 6:55 PM | 6:58 PM | 7:02 PM | 7:07 PM | 7:15 PM |
| 7:20 PM | 7:25 PM | 7:31 PM | 7:35 PM | 7:38 PM | 7:42 PM | 7:47 PM | 7:55 PM |
| 8:00 PM | 8:05 PM | 8:11 PM | 8:15 PM | 8:18 PM | 8:22 PM | 8:27 PM | 8:35 PM |
| 8:40 PM | 8:45 PM | 8:51 PM | 8:55 PM | 8:58 PM | 9:02 PM | 9:07 PM | 9:15 PM |
| 9:20 PM | 9:25 PM | 9:31 PM | 9:35 PM | 9:38 PM | 9:42 PM | 9:47 PM | 9:55 PM |
| 10:00 PM | 10:05 PM | 10:11 PM | 10:15 PM | 10:18 PM | 10:22 PM | 10:27 PM | 10:35 PM |
| 10:40 PM | 10:45 PM | 10:51 PM | 10:55 PM | 10:58 PM | 11:02 PM | 11:07 PM | 11:15 PM |
| 11:20 PM | 11:25 PM | 11:31 PM | 11:35 PM | 11:38 PM | 11:42 PM | 11:47 PM | 11:55 PM |
| 12:00 AM | 12:05 AM | 12:11 AM | 12:15 AM | 12:18 AM | 12:22 AM | 12:27 AM | 12:35 AM |
| 12:40 AM | 12:45 AM | 12:51 AM | 12:55 AM | 12:58 AM | 1:02 AM | 1:07 AM | 1:15 AM |
| 1:20 AM | 1:25 AM | 1:31 AM | 1:35 AM | 1:38 AM | 1:42 AM | 1:47 AM | **Out of Service Su-W** |
| | | | | | | 1:47 AM | 1:55 AM |
| 2:00 AM | 2:05 AM | 2:11 AM | 2:15 AM | 2:18 AM | 2:22 AM | 2:27 AM | **Out of Service H-Sa** |

These images were obtained from the North Carolina State University website: http://www2.acs.ncsu.edu/trans/transportation/wolfline/images/Spring/Werewolf.pdf.

## **Appendix 2: Sample data** (Taken from August 27, 2006)

| Stop | Stop Index | Date | Arrival at Current Stop | Departure | Arrival at Next Stop | Dwell Time | Travel Time | Most Recent Scheduled Departure Time |
|---|---|---|---|---|---|---|---|---|
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 18:37:15 | 18:38:06 | 18:42:30 | 0:00:51 | 0:04:24 | 18:35:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 18:42:30 | 18:43:40 | 18:50:28 | 0:01:10 | 0:06:48 | 18:40:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 18:50:28 | 18:50:55 | 18:55:23 | 0:00:27 | 0:04:28 | 18:51:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 18:55:23 | 18:56:05 | 19:00:33 | 0:00:42 | 0:04:28 | 18:55:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 19:00:33 | 19:01:45 | 19:04:51 | 0:01:12 | 0:03:06 | 19:02:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 19:04:51 | 19:05:55 | 19:12:25 | 0:01:04 | 0:06:30 | 19:07:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 19:12:25 | 19:15:38 | 19:21:48 | 0:03:13 | 0:06:10 | 19:15:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 19:21:48 | 19:22:36 | 19:31:49 | 0:00:48 | 0:09:13 | 19:20:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 19:31:49 | 19:33:17 | 19:37:44 | 0:01:28 | 0:04:27 | 19:31:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 19:37:44 | 19:38:26 | 19:45:36 | 0:00:42 | 0:07:10 | 19:35:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 19:45:36 | 19:46:00 | 19:48:49 | 0:00:24 | 0:02:49 | 19:42:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 19:48:49 | 19:49:25 | 19:55:22 | 0:00:36 | 0:05:57 | 19:47:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 19:55:22 | 19:56:20 | 20:02:32 | 0:00:58 | 0:06:12 | 19:55:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 20:02:32 | 20:03:53 | 20:11:27 | 0:01:21 | 0:07:34 | 20:00:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 20:11:27 | 20:12:00 | 20:16:34 | 0:00:33 | 0:04:34 | 20:11:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 20:16:34 | 20:17:19 | 20:24:14 | 0:00:45 | 0:06:55 | 20:15:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 20:24:14 | 20:25:29 | 20:28:45 | 0:01:15 | 0:03:16 | 20:22:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 20:28:45 | 20:29:42 | 20:35:28 | 0:00:57 | 0:05:46 | 20:27:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 20:35:28 | 20:37:34 | 20:43:26 | 0:02:06 | 0:05:52 | 20:35:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 20:43:26 | 20:44:35 | 20:54:03 | 0:01:09 | 0:09:28 | 20:40:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 20:54:03 | 20:54:30 | 20:59:04 | 0:00:27 | 0:04:34 | 20:51:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 20:59:04 | 20:59:52 | 21:05:07 | 0:00:48 | 0:05:15 | 20:55:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 21:05:07 | 21:05:55 | 21:09:38 | 0:00:48 | 0:03:43 | 21:02:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 21:09:38 | 21:10:23 | 21:17:33 | 0:00:45 | 0:07:10 | 21:07:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 21:17:33 | 21:18:24 | 21:23:25 | 0:00:51 | 0:05:01 | 21:15:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 21:23:25 | 21:24:37 | 21:34:08 | 0:01:12 | 0:09:31 | 21:20:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 21:34:08 | 21:34:53 | 21:39:12 | 0:00:45 | 0:04:19 | 21:31:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 21:39:12 | 21:39:54 | 21:45:31 | 0:00:42 | 0:05:37 | 21:35:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 21:45:31 | 21:46:16 | 21:49:07 | 0:00:45 | 0:02:51 | 21:42:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 21:49:07 | 21:49:43 | 21:54:47 | 0:00:36 | 0:05:04 | 21:47:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 21:54:47 | 21:56:11 | 22:00:42 | 0:01:24 | 0:04:31 | 21:55:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 22:00:42 | 22:02:00 | 22:08:52 | 0:01:18 | 0:06:52 | 22:00:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 22:08:52 | 22:09:16 | 22:13:43 | 0:00:24 | 0:04:27 | 22:11:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 22:13:43 | 22:14:28 | 22:20:38 | 0:00:45 | 0:06:10 | 22:15:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 22:20:38 | 22:21:05 | 22:23:48 | 0:00:27 | 0:02:43 | 22:22:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 22:23:48 | 22:25:00 | 22:29:46 | 0:01:12 | 0:04:46 | 22:27:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 22:29:46 | 22:38:47 | 22:43:29 | 0:09:01 | 0:04:42 | 22:35:00 |
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 22:43:29 | 22:44:48 | 22:51:27 | 0:01:19 | 0:06:39 | 22:40:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 22:51:27 | 22:51:54 | 22:56:25 | 0:00:27 | 0:04:31 | 22:51:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 22:56:25 | 22:57:28 | 23:03:26 | 0:01:03 | 0:05:58 | 22:55:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 23:03:26 | 23:04:08 | 23:07:14 | 0:00:42 | 0:03:06 | 23:02:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 23:07:14 | 23:07:57 | 23:14:12 | 0:00:43 | 0:06:15 | 23:07:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 23:14:12 | 23:17:01 | 23:21:34 | 0:02:49 | 0:04:33 | 23:15:00 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Cates Ave at Talley Student Center | 1 | 8/27/2006 | 23:21:34 | 23:22:43 | 23:28:35 | 0:01:09 | 0:05:52 | 23:20:00 |
| Main Campus Dr at College of Textiles | 2 | 8/27/2006 | 23:28:35 | 23:28:59 | 23:33:45 | 0:00:24 | 0:04:46 | 23:31:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/27/2006 | 23:33:45 | 23:34:27 | 23:39:07 | 0:00:42 | 0:04:40 | 23:35:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/27/2006 | 23:39:07 | 23:39:55 | 23:43:04 | 0:00:48 | 0:03:09 | 23:42:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/27/2006 | 23:43:04 | 23:43:43 | 23:48:53 | 0:00:39 | 0:05:10 | 23:47:00 |
| Founders Dr at DH Hill Library | 0 | 8/27/2006 | 23:48:53 | 23:56:42 | 0:00:42 | 0:07:49 | 0:04:00 | 23:55:00 |
| Cates Ave at Talley Student Center | 1 | 8/28/2006 | 0:00:42 | 0:02:37 | 0:08:47 | 0:01:55 | 0:06:10 | 0:00:00 |
| Main Campus Dr at College of Textiles | 2 | 8/28/2006 | 0:08:47 | 0:09:17 | 0:14:17 | 0:00:30 | 0:05:00 | 0:11:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/28/2006 | 0:14:17 | 0:15:05 | 0:21:21 | 0:00:48 | 0:06:16 | 0:15:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/28/2006 | 0:21:21 | 0:22:12 | 0:25:01 | 0:00:51 | 0:02:49 | 0:22:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/28/2006 | 0:25:01 | 0:25:40 | 0:30:29 | 0:00:39 | 0:04:49 | 0:27:00 |
| Founders Dr at DH Hill Library | 0 | 8/28/2006 | 0:30:29 | 0:38:18 | 0:42:24 | 0:07:49 | 0:04:06 | 0:35:00 |
| Cates Ave at Talley Student Center | 1 | 8/28/2006 | 0:42:24 | 0:43:06 | 0:48:58 | 0:00:42 | 0:05:52 | 0:40:00 |
| Main Campus Dr at College of Textiles | 2 | 8/28/2006 | 0:48:58 | 0:49:25 | 0:54:04 | 0:00:27 | 0:04:39 | 0:51:00 |
| Centennial Parkway EC Lot (2nd shelter) | 3 | 8/28/2006 | 0:54:04 | 0:54:53 | 0:59:26 | 0:00:49 | 0:04:33 | 0:55:00 |
| Gorman St at Kings Ct (Inbound) | 4 | 8/28/2006 | 0:59:26 | 0:59:50 | 1:03:00 | 0:00:24 | 0:03:10 | 1:02:00 |
| Fraternity Ct at Varsity Dr (Inbound) | 5 | 8/28/2006 | 1:03:00 | 1:03:36 | 18:40:22 | 0:00:36 | 0:05:00 | 1:07:00 |

# Appendix 3: Matlab codes

```
% assess_tt.m

% This function implements the Total Trip Time measurement method, using historical data
% to measure total trip time for all origin- destination pairs

% A given solution vector, consisting of differences between scheduled departure times of
% consecutive time-points (sched_bst), is used to construct a schedule.

% Link times in between stops are taken from historical data, in the order that they
% originally occurred. Dwell times are assumed to be constant.

% The link, dwell, and schedule times form a deterministic simulation.

function cost = assess_tt(sched_bst, dwell_time);

% Initializing quantities

num_loops = 10;
num_stops = 6;
num_days = 24;

cost = 0;

% Retrieving and organizing data

data = load('K:\thesis\Travel Times 124578.txt');

for k = 1:num_days,
    for j = 1:num_loops,
        for i = 1:num_stops,
             travel_times(j,i,k) = data((num_stops*num_loops)*(k-1) + num_stops*(j-1)+i);
        end;
    end;
end;

% Beginning the deterministic simulation

for k = 1:num_days,

    % Resetting the schedule and clock at the beginning of each day
    sched = 0;
    time = 0;

    % Keeping track of recent arrival and schedule times

    sched_current = zeros(1,length(sched_bst));
    arrivals_current = zeros(1,length(sched_bst));

    % Beginning a new day

    for j = 1:num_loops,
        for i = 1:num_stops;

              % Updating schedule vector
              %(kept one ahead of arrivals to make cost function calculation easier)
              sched_current(1:length(sched_bst)-1) = sched_current(2:length(sched_bst));

              % Assuming passengers arrive at time-point (start trip) at scheduled time
              sched_current(length(sched_bst)) = sched;

              % Identifying scheduled time for next time-point
              sched = sched + sched_bst(i);

              % Bus travels from time-point i-1 to time-point i
              time = time + travel_times(j,i,k);

         % Updating arrivals vector
          arrivals_current(1:length(sched_bst)-1) = arrivals_current(2:length(sched_bst));
```

```
            % Passengers alight when bus arrives
            arrivals_current(length(sched_bst)) = time;

               % Bus dwells at time-point i
               time = time + dwell_time;

               % If the bus is early, bus leaves at scheduled time
               if time < sched,
                      time = sched;
               end;

               % Measuring travel time

               % For a given passenger, their trip starts at the scheduled
               % departure time for their origin time-point and ends when the bus
               % arrives at their destination time-point

               % One loop must be complete before costs are recorded

               if j > 1 || i == length(sched_bst),
                      for n = 1:length(sched_bst)-1,
                      cost = cost + arrivals_current(n) - sched_current(1);
               end;
            end;
        end;
    end;

    % Measuring cost for trips begun on last loop

    for m1 = 2:length(sched_bst)-1,
        for m2 = m1+1:length(sched_bst),
            cost = cost + arrivals_current(m2) - sched_current(m1);
        end;
    end;
end;

% Final cost is average trip time (= cumulative cost / number of total trips)

cost = cost/((((num_loops-1) * num_stops + 1) * (num_stops-1) + (num_stops * (num_stops +
1)/2)) * num_days);
```

```
% weights.m

% This function calculates the earliness and tardiness weights e_k and t_k.
% It is used by assess.m, which measures total trip time.

function [weights_early, weights_tardy] = weights(p, sched_bst);

N = length(sched_bst);
weights_early = zeros(1,N);
weights_tardy = zeros(1,N);

for i = 1:N,
    for j = 1:N,
        if i ~= j,
            if i < j,

                % If passenger is on-board at stop k, increase weights_early(k)
                for k = i+1:j-1,
                    weights_early(k) = weights_early(k) + p(i,j);
                end;

            else
                for k = i+1:N,
                    weights_early(k) = weights_early(k) + p(i,j);
                end;
                for k = 1:j-1,
                    weights_early(k) = weights_early(k) + p(i,j);
                end;
            end;

            % Increase weights_tardy(k) for passengers boarding at time-point k
            weights_tardy(i) = weights_tardy(i) + p(i,j);
        end;
    end;
end;
```

```
% assess.m

% This function implements the Weighted Deviations measurement method, using historical
% data to measure total trip time as a sum of weighted deviations from the scheduled
% departure time.

% A given solution vector, consisting of differences between scheduled % departure times
% of consecutive time-points (sched_bst), is used to construct a schedule.

% Link times in between time-points are taken from historical data, in the order that they
% originally occurred. Dwell times are assumed to be constant.

% The link, dwell, and schedule times form a deterministic simulation.

function [cost, early, tardy] = assess(sched_bst, dwell_time);

% Initializing quantities

num_loops = 10;
num_stops = 6;
num_days = 24;

% Determining origin-destination proportions for uniform distribution

for i = 1:6,
    for j = 1:6,
        if i~=j,
            p(i,j) = 1/((length(sched_bst)*(length(sched_bst)-1));
        end;
    end;
end;

% Using O-D proportions to calculate deviation weights

[weights_early, weights_tardy] = weights(p, sched_bst);

% Allowing for different dwell time for each stop

dw = [1.83, 1.30, 1.32, 0.75, 0.85, 0.80];

% Retrieving and organizing data

early = zeros(1, length(sched_bst));
tardy = zeros(1, length(sched_bst));

data = load('K:\thesis\Travel Times 124578.txt');

for k = 1:num_days,
    for j = 1:num_loops,
        for i = 1:num_stops,
             travel_times(j,i,k) = data((num_stops*num_loops)*(k-1) + num_stops*(j-1)+i);
        end;
    end;
end;

% Carrying out the deterministic simulation

for k = 1:num_days,
    sched = 0;
    time = 0;
    for j = 1:num_loops,
        for i = 1:num_stops;

               % Identifying scheduled time for next stop
               sched = sched + sched_bst(i);

               % Bus travels from time-point i-1 to time-point i
               time = time + travel_times(j,i,k);

               % Counting early arrivals
               if time + 3 < sched,
```

```
                        before = before + 1;
                end;

                % dwell_time = dw(i);

                % Bus dwells at time-point i
                time = time + dwell_time;

                % Counting tardy arrivals
                if time > sched + 5,
                        after = after + 1;
                end;

                % If the bus is late . . .
                if time > sched ,
                        tardy(i) = tardy(i) + weights_tardy(i) * (time - sched);
                end;

                % If the bus is early . . .
                if time < sched,
                        early(i) = early(i) + weights_early(i) * (sched - time);

                        % Bus leaves at scheduled time
                        time = sched;
                end;
        end;
    end;
end;

% Calculating cost per trip
cost = (sum(tardy) + sum(early))/(num_loops * num_days);
early = early/(num_loops * num_days);
tardy = tardy/(num_loops * num_days);
```

```
% optimsd.m

% This function implements the Modified Steepest Descent search algorithm.

% Given a fixed number of maximum iterations, it searches for the solution that minimizes
% total trip time by evaluating all adjacent solutions.

function [best_solution, best_cost, run_index] = optimsd(sched_bst, dwell_time, num_runs);

% Initializing quantities

best_next_cost = 10000;   % set high so that it's immediately replaced
sched_summary = [];

[cost, early, tardy] = assess(sched_bst, dwell_time);
best_solution = sched_bst;
best_cost = cost;
run_index = 1; stop_indicator = 0;

% Search algorithm begins

for i = 1:num_runs,

    % Continuing only if a minimum hasn't already been found
    if stop_indicator ~= 1,

    % Trying all feasible adjacent points, keeping the one yielding lowest cost

    start_solution = sched_bst;
    best_next_cost = 1000000;

    for j = 1:length(sched_bst),
        for k = 1:length(sched_bst),
               if j~=k && sched_bst(k) > 0,
                      sched_bst(j) = sched_bst(j) + 1;
                      sched_bst(k) = sched_bst(k) - 1;
                      [cost, early, tardy] = assess(sched_bst, dwell_time);
               end;

               % updating the best next step
               if cost < best_next_cost,
                      best_next_step = sched_bst;
                      best_next_cost = cost;
               end;

               sched_bst = start_solution;

        end;
    end;

    sched_bst = best_next_step;

    % Keeping track of iterations
    sched_summary = [sched_summary; sched_bst, best_next_cost, best_cost];

    % Comparing with past iterations
    if best_next_cost < best_cost,
        best_solution = sched_bst;
        best_cost = best_next_cost;
        run_index = i + 1;

    % Identifying if a minimum has been found
    else
        stop_indicator = 1;
    end;
    end;
end;

% Reporting a summary of the iterations
sched_summary
```

```
% optimsd2.m

% This function implements the version Modified Steepest Descent search algorithm that
% does not restrict solutions to a fixed scheduled cycle length.

% Given a fixed number of maximum iterations, it searches for the solution that minimizes
% total trip time by evaluating all adjacent solutions.

% In this version, “adjacent” solutions differ by one minute at exactly one element.

function [best_solution, best_cost, run_index] = optimsd(sched_bst, dwell_time, num_runs);

% Initializing quantities

best_next_cost = 10000;   % set high so that it's immediately replaced
data = [];
sched_summary = [];

[cost, early, tardy] = assess(sched_bst, dwell_time);
best_solution = sched_bst;
best_cost = cost;
run_index = 1;
stop_indicator = 0;

% Search algorithm begins

for i = 1:num_runs,

    % Continuing only if a minimum hasn’t already been found
    if stop_indicator ~= 1,

    % Trying all feasible adjacent points, keeping the one yielding lowest cost

    start_solution = sched_bst;
    best_next_cost = 1000000;

    for j = 1:length(sched_bst),
         sched_bst(j) = sched_bst(j) + 1;
        [cost, early, tardy] = assess(sched_bst, dwell_time);

        % Updating the best next step
        if cost < best_next_cost,
            best_next_step = sched_bst;
            best_next_cost = cost;
        end;

        sched_bst = start_solution;

    end;

    for j = 1:length(sched_bst),
        sched_bst(j) = sched_bst(j) - 1;
        [cost, early, tardy] = assess(sched_bst, dwell_time);

        % Updating the best next step
        if cost < best_next_cost,
            best_next_step = sched_bst;
            best_next_cost = cost;
        end;

        sched_bst = start_solution;
    end;

    sched_bst = best_next_step;

    % Keeping track of iterations
    sched_summary = [sched_summary; sched_bst, best_next_cost, best_cost];

    % Comparing with past iterations
    if best_next_cost < best_cost,
        best_solution = sched_bst;
```

```
        best_cost = best_next_cost;
        run_index = i + 1;

    % Identifying if a minimum has been found
    else
        stop_indicator = 1;
    end;

    end;

end;

% Reporting a summary of the iterations
sched_summary
```

```
% brute.m

% This function implements the exhaustive enumerative approach, measuring the total trip
% time for all feasible solutions using the Weighted Deviations measurement method

function [best_solution, best_cost, y, percentiles] = brute(bounds);

y = possibles(bounds);

length(y)

best_cost = 10000000;
dwell_time = 0.8;

for i = 1:length(y(:,1)),
    [cost, early, tardy] = assess(y(i,1:6), dwell_time);
    y(i,7) = cost;

    % Keeping track of the best solution
    if y(i,7) < best_cost,
        best_solution = y(i,1:6);
        best_cost = y(i,7);
    end;
end;

y = sortrows(y, 7);

% Reporting the best ten and worst solutions

y(1:10,:)
y(length(y),:)

% Reporting the rank of the solution corresponding to the existing schedule

for j = 1:length(y),
    if y(j,1:6) == [5, 11, 4, 7, 5, 8],
        current_solution_rank = j
    end;
end;

percentiles = prctile(y(:,7), [0 10 25 50 75 90 100]);
```

possibles.m

```
% Given a set of bounds – one for each element in a solution vector - this function
% identifies all feasible solutions

% A feasible solution must also have cycle length the appropriate cycle length C

function y = possibles(bounds);

x = zeros(1, length(bounds(:,1)));
y = [];
C = 40;

for k1 = bounds(1,1):bounds(1,2),
    x(1) = k1;
    for k2 = bounds(2,1):min(bounds(2,2), C - k1),
        x(2) = k2;
        for k3 = bounds(3,1):min(bounds(3,2), C - k1 - k2),
            x(3) = k3;
            for k4 = bounds(4,1):min(bounds(4,2), C - k1 - k2 - k3),
                x(4) = k4;
                for k5 = bounds(5,1):min(bounds(5,2), C- k1 - k2 - k3 - k4),
                    x(5) = k5;
                    for k6 = bounds(6,1):min(bounds(6,2), C - k1 - k2 - k3 - k4 - k5),
                        x(6) = k6;
                        if sum(x) == C,
                            y = [y;x];
                        end;
                    end;
                end;
            end;
        end;
    end;
end;
```

```
% simulation.m

% This function measures the total trip times using the Weighted Deviations measurement
% method and a random simulation.

% A given solution vector, consisting of differences between scheduled departure times of
% consecutive time-points (sched_bst), is used to construct a schedule.

% Link times in between stops are randomly selected from distributions taken from
% historical data, ignorant of the order in which they originally occurred. Dwell times
% are assumed to be constant.

function [cost, early, tardy] = simulation(sched_bst, dwell_time);

% Initializing quantities

num_loops = 10;
num_stops = length(sched_bst);
num_days = 600;
t = []; costs = [];
s = [];

% Determining origin-destination proportions for uniform distribution

for i = 1:6,
    for j = 1:6,
        if i~=j,
            p(i,j) = 1/((length(sched_bst)*(length(sched_bst)-1));
        end;
    end;
end;

% Using O-D proportions to calculate deviation weights

[weights_early, weights_tardy] = weights(p, sched_bst);

% Retrieving and organizing data

n_days = 24;

data = load('K:\thesis\Travel Times 124578.txt');

for j = 1:num_loops*n_days,
    for i = 1:num_stops,
       travel_times(j,i) = data((num_stops)*(j-1) + i);
    end;
end;

% Beginning the random simulation

early = zeros(1, length(sched_bst));
tardy = zeros(1, length(sched_bst));

for d = 1:num_days,
    sched = 0;
    time = 0;
    for j = 1:num_loops,
        for i = 1:num_stops;

               % Identifying scheduled time for next stop
               sched = sched + sched_bst(i);

               % Bus travels from time-point i-1 to time-point i
               x = ceil(length(travel_times(:,i))*rand(1));
            time = time + travel_times(x,i);

               % Bus dwells at time-point i
            time = time + dwell_time;

               % If the bus is late . . .
            if time > sched ,
```

```
                tardy(i) = tardy(i) + weights_tardy(i) * (time - sched);
            end;

               % If the bus is early, bus leaves at scheduled time
            if time < sched,
                early(i) = early(i) + weights_early(i) * (sched - time);
                time = sched;
            end;
        end;
    end;
end;

% Calculating total costs

cost = (sum(tardy) + sum(early))/(num_loops * num_days);
early = early/(num_loops * num_days);
tardy = tardy/(num_loops * num_days);
```

```
% optimization.m

% This function implements the One-Up-One-Down search algorithm

% Given a fixed number of maximum iterations, it searches for the solution that minimizes
% total trip time by adjusting the element corresponding to the stop at which the
% greatest delays are incurred and a complementary element.

function [best_solution, best_cost, run_index] = optimization(sched_bst, dwell_time,
num_runs);

% Initializing quantities

sched_summary = [];

[cost, early, tardy] = assess(sched_bst, dwell_time);
best_solution = sched_bst;
best_cost = cost;
run_index = 1;

% Search algorithm begins

for i = 1:num_runs,

    % Randomly choosing the complementary element
    r = floor((length(sched_bst) - 1)*rand(1));

    % Identifying the element corresponding to the stop adding the greatest delay

    [y, index] = max([early, tardy]);

    % Making schedule changes
    % By adding and subtraction the same amount, the route length is kept
    % ensures adjustments don't yield negative times

    if index < 1 + length(sched_bst) && sched_bst(index) > 0 || sched_bst(1 + mod(index + r,
6)) > 0,

        % If schedule is too late . . .
        if index < 1 + length(sched_bst) && sched_bst(index) > 0,
               if sched_bst(index) > 0,
                  sched_bst(index) = sched_bst(index) - 1;
                  sched_bst(1 + mod(index + r,6)) = sched_bst(1 + mod(index + r, 6)) + 1;
               end;

        % If schedule is too early . . .
        else
               sched_bst(1 + mod(index - 1, 6)) = sched_bst(1 + mod(index - 1, 6)) + 1;
               sched_bst(1 + mod(index + r, 6)) = sched_bst(1 + mod(index + r, 6)) - 1;
        end;
    end;

    % Measures cost for current solution vector
    [cost, early, tardy] = assess(sched_bst, dwell_time);

    % If this cost is less than all others, it's the new best_cost
    if cost < best_cost,
        best_solution = sched_bst;
        best_cost = cost;
        run_index = i + 1;
    end;

    % Keeps track of iterations
    sched_summary = [sched_summary; sched_bst, cost, best_cost];

end;

% A summary of the iterations is reported

sched_summary
```

## Appendix 4: Proof of Equation (27)

**Claim:** *For a route with $N$ time-points, if the origin-destination trip pairs are uniformly distributed, then*

$$e_k = \frac{N-2}{2N}, \text{ for } k = 0, ..., N-1. \tag{28}$$

**Proof:**

Recall that

$$p_{ij} \equiv \frac{1}{N(N-1)} \forall i, j, \tag{29}$$

since each passenger has $N$ possible origins and $N-1$ possible destinations. This is also stated in (5).

Additionally, we know that

$$e_k = \sum_{i=2}^{N-1} \sum_{j=1}^{i-1} p_{(k+i) \bmod N, (k+j) \bmod N} \tag{30}$$

from (21).

Likewise, we know that

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2}. \tag{31}$$

Now,

$$e_k = \sum_{i=2}^{N-1} \sum_{j=1}^{i-1} \frac{1}{N(N-1)} \tag{32}$$

$$= \frac{1}{N(N-1)} \sum_{i=2}^{N-1} \sum_{j=1}^{i-1} 1 \tag{33}$$

$$= \frac{1}{N(N-1)} \sum_{i=2}^{N-1} i - 1, \tag{34}$$

and we let $m = i - 1$, such that,

$$= \frac{1}{N(N-1)} \sum_{m=1}^{N-2} m, \tag{35}$$

and using (32),

$$= \Big(\frac{1}{N(N-1)}\Big)\Big(\frac{(N-1)(N-2)}{2}\Big) \tag{36}$$

$$= \frac{N-2}{2N}. \tag{37}$$

**Corollary:**

We also note that

$$\lim_{N\to\infty} e_k = \lim_{N\to\infty} \frac{N-2}{2N} = \frac{1}{2}, \tag{38}$$

while

$$\lim_{N\to\infty} t_k = \lim_{N\to\infty} \frac{1}{N} = 0. \tag{39}$$