L-TRAIN SIMULATION MODEL

SAYAKA ISHIZAWA, GARNETT MEISE, WESLEY SUTTLE

Introduction

For our AMS 553 final project, we constructed a simulation model of a section of the New York City subway system and used it to explore different configurations of a shuttle bus schedule that the subway authorities have proposed using during upcoming construction. Though it was necessary to make several simplifying assumptions, given the time and information availability constraints, our simulation study has still been a substantial undertaking. We collected, cleaned, and analysed detailed turnstile data obtained from the Metro Transit Authority (MTA) website. We fitted distributions to this data and performed input analyses on them. We wrote a complex simulation program in C++ with which to perform our experiments. We formulated a complicated objective function, hypothesized its cost coefficients, and explored the cost of several alternative bus schedule configurations. Finally, on the basis of our experimental results and output analyses, we determined that, of the many configurations that we tested, the optimal morning rush hour shuttle bus schedule is to have five primary buses waiting to depart with the passengers from each arriving train, while also sending a single secondary bus every minute between primary bus arrivals to pick up stragglers. In this paper, we discuss the background of our project, the model underpinning our simulation, our input analysis, our simulation program and algorithms driving it, our objective function, and our preferred bus schedule. We conclude by discussing the assumptions and limitations of our simulation model.

Background. Hurricane Sandy caused severe damage to the New York City subway system in 2012. The **L** subway line was particularly hard-hit. The MTA is closing the **L** west of Bedford Avenue station for 15 months beginning in 2019 to effect repairs[1]. This will be the most significant disruption to New York's public transit system in history[2], and will especially affect many living in the Greenpoint, Williamsburg, and Bushwick areas of Brooklyn who depend on the short commute into Manhattan that the **L** currently offers. There has naturally been public demand for the MTA to take measures to lessen the disruption felt by those in the affected areas. One of the few concrete proposals that have been made public is the establishment of a shuttle bus route dedicated to ferrying subway passengers from Bedford Ave station to Manhattan.

[1]https://www.nytimes.com/2017/04/03/nyregion/mta-l-train-shutdown-15-months.html
[2]https://www.nytimes.com/2016/07/26/nyregion/l-train-will-shut-down-between-manhattan-and-brooklyn-in-2019-for-18-months.html

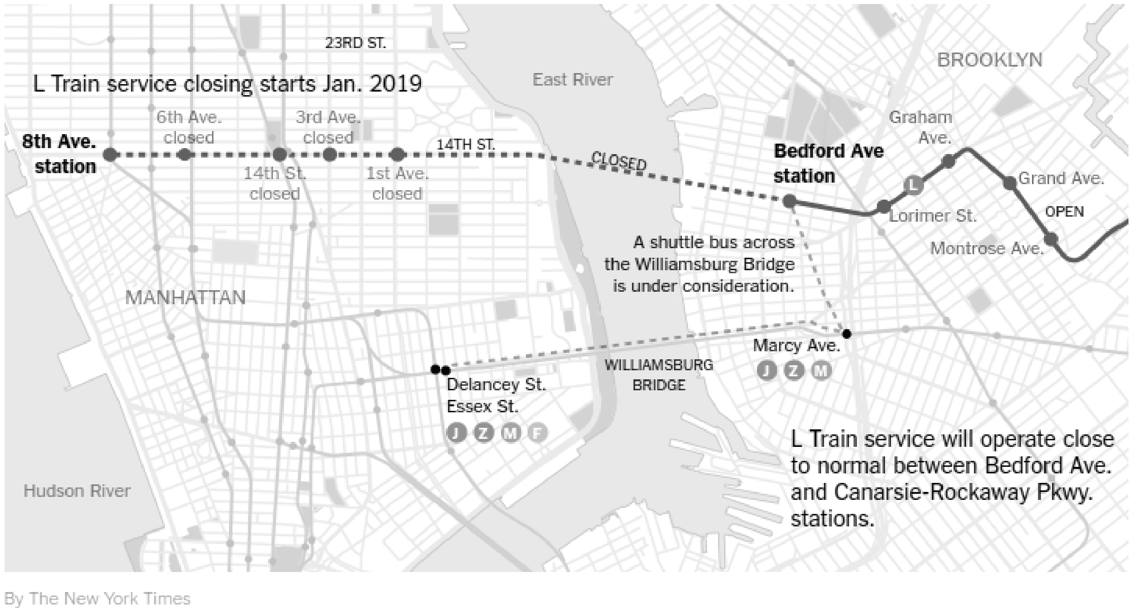

Figure 1. Proposed Shuttle Bus Route

We model the Manhattan-bound morning rush hour (7:00AM to 9:00AM) subway traffic between DeKalb and Bedford Ave stations, with the additional hypothetical assumption that all passengers arriving at Bedford disembark and join a shuttle bus queue to Manhattan. The queue is served by a shuttle bus system operating according to a fixed schedule of our devising. We use this model to explore bus schedule configurations, the resulting customer wait times and other stochastic output, and minimize the losses incurred as a result of customer wait times at the Bedford queue and operating costs of the given bus schedule. Though the focus of the model is somewhat narrow, it allows us to study the proposed shuttle bus system during what would be the most critical period of its operational day.

The Simulation Model

To successfully simulate a day in the system, we must do four things: (1) generate the overall traffic intensity the system will experience that day; (2) generate the customer arrival processes at each station; (3) send trains through the system on a fixed schedule, picking up all passengers at each stop, then dropping them all off to join the shuttle bus queue upon arrival at Bedford; (4) send passengers out of the system on the shuttle buses departing from Bedford and evaluate the cost associated with customer wait times and buses used. To these ends, we fitted distributions to observed daily traffic intensities obtained via detailed turnstile data provided on the MTA website, hypothesized an arrival rate function to be used in generating customer interarrival times at each of the eight stations, and wrote a program to generate realizations from the specified probability distributions, orchestrate the movement of trains, passengers, and buses through the system, and record, organize, and output relevant statistical and objective function data. Finally, we developed an objective function to be used to identify efficient shuttle bus departure schedules, then ran experiments to determine the best.

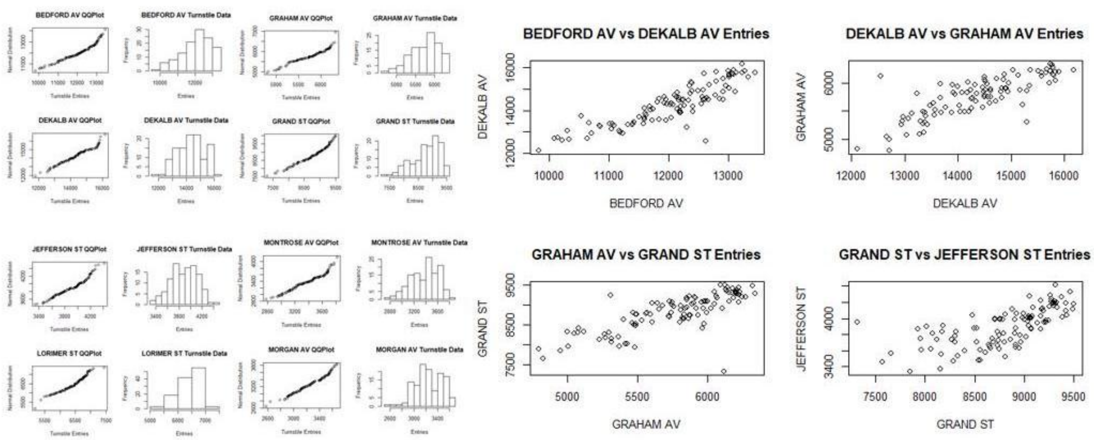

FIGURE 2. Turnstile Histograms, QQ, and Correlation Plots

Fitting Overall Traffic Intensity. We obtained our input data from the MTA website[3], which consisted of large text files listing cumulative total arrivals through the individual turnstiles at each station. Our goal was to get a sample for people arriving during rush hour at each of the eight stations in our system. The MTA collected data in four-hour increments and we parsed the data to obtain turnstile readings from 4:00AM to noon, Monday through Friday. To obtain the number of arrivals that occurred at each station during each four-hour period, we first calculated the difference between the cumulative arrival readings at the beginning and end of the period for each of the station's turnstiles. Summing each of these turnstile-specific values gave us the total arrivals for the desired four-hour period. We finally summed the arrivals over both four-hour periods to obtain the total arrivals from 4:00AM to noon on a given day. We performed this process on data from a six-month period, resulting in around 130 data points for each station. In order to get an accurate picture of current ridership, we limited our data to the most recent six months.

[3] http://web.mta.info/developers/turnstile.html

It was necessary to remove several values from the data. For example, the MTA periodically does maintenance on the turnstiles, which results in machine rollback, where the arrival counter is reset to zero. The resulting data points were inaccurate and we removed them when this occurred. Additionally, there were several outliers in the data set, which happened to be holidays. Examples of some of these outliers are July 3rd, July 4th, and September 4th (Labor Day). As we were mainly interested in rush hour, we removed these points from our data set. As a result of machine rollbacks and outliers, 5-10 data points were removed from the data set for each station.

For each of the eight station, we created a histogram and QQ plot of the corresponding empirical distribution against a fitted normal distribution. Some of the stations such as Jefferson, Montrose, and DeKalb had fairly symmetric histograms and provided an almost linear QQ plot. Other stations such as Bedford and Lorimer were left-skewed. This could be the result of weather or other occurrences that may have caused less people to use that particular station on a given day. A Chi-squared test of the fitted normal distributions versus the data was performed for each station, resulting in p-values around .24 for all stations. Though we realize there are likely distributions that would more accurately represent the histograms given, it was simplest to fit normal distributions to each station, and the resulting goodness-of-fit was not too bad.

The number of arrivals between stations is highly correlated. A few examples are shown in the scatter plots above. All $\binom{8}{2} = 28$ correlation coefficients were calculated, resulting in minimum correlation coefficient of .55 between Jefferson and Lorimer and maximum correlation of .95 between Bedford and Graham. The significant positive correlation between stations provides justification for using a single standard normal random variate at the beginning of each run to determine the overall traffic intensity for the simulated day.

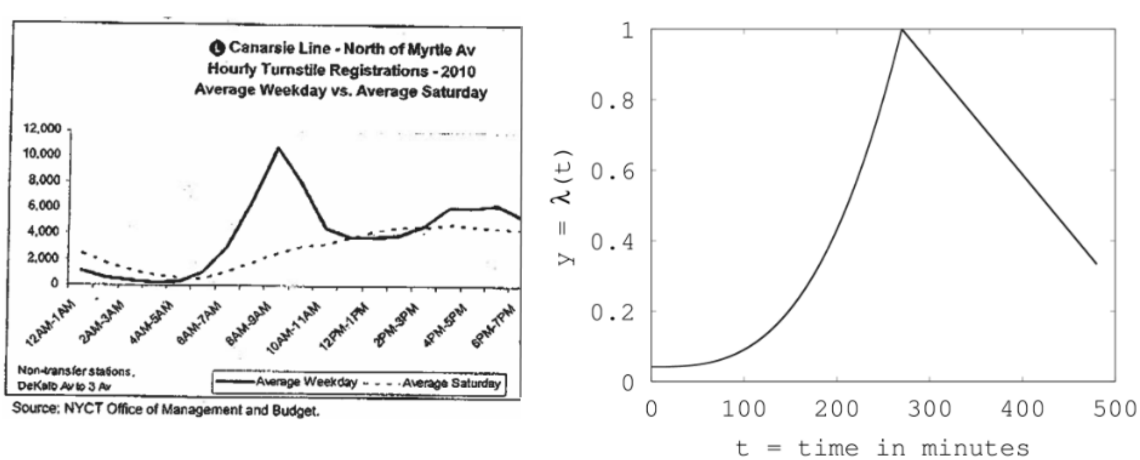

FIGURE 3. Arrival Rate Function

Arrival Rate Function. We model customer arrivals at each of the eight stations as a non-stationary Poisson process. These processes result from two reasonable assumptions that we make for the purposes of our simulation study: (1) customer interarrival times are exponentially distributed; (2) the rate at which customers arrive varies continuously over the course of the day. Based on data obtained from the NYC Transit Authority[4], a good fit to the empirical arrival rate function of the stations in our system is given by:

$$\lambda(t) = \begin{cases} K\left(\frac{1}{24} + \frac{23}{24}\left(\frac{t}{270}\right)^3\right) & 0 \leq t \leq 270, \\ K\left(1 - \frac{1}{315}(270 - t)\right) & 270 < t \leq 480, \\ 0 & \text{otherwise,} \end{cases}$$

where K is a station-specific scaling factor. See Figure 3 for a side-by-side comparison of the NYC Transit Authority arrival rate graph and a graph of our fitted arrival rate function.

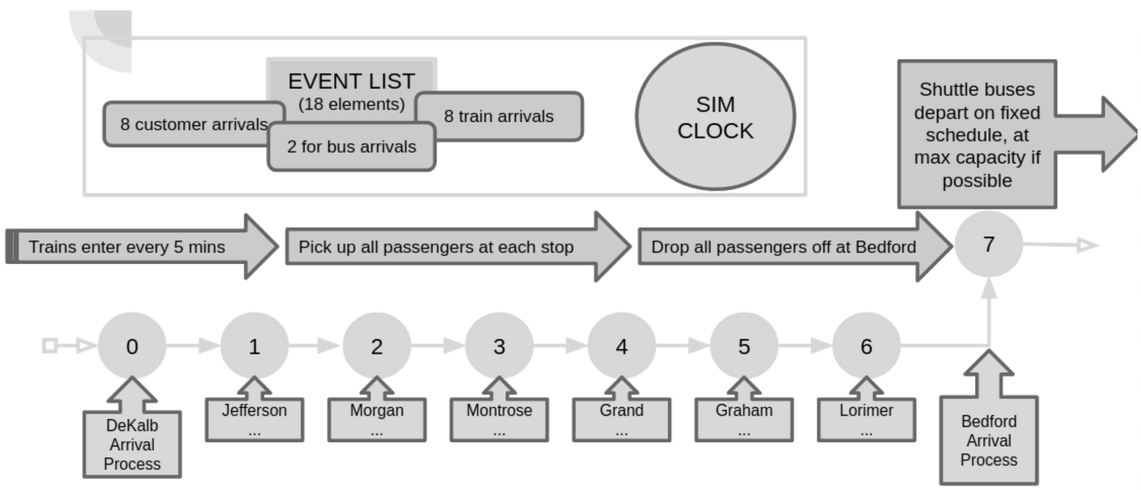

FIGURE 4. The Simulation Program

How The Program Works. Within the program, there are initially eight empty stations, no trains, and no buses. The simulation clock is in minutes, with $t = 0$ corresponding to 4:00AM, and $t = 480$ corresponding to 12:00PM. Before time begins advancing, we first generate a realization from the standard normal distribution, which we use to set the overall traffic intensity of the coming simulated day. This random variate is then transformed and used to determine all eight of the day's station-specific traffic intensities. Once the simulation clock begins advancing, arrivals begin to occur at each station according to a station-specific non-stationary Poisson process. The start of each of the eight arrival

[4]See the MTA memo included in the zip file submitted with our paper.

processes are staggered so that, at the moment the first train arrives to a given station, arrivals have been occurring for exactly five minutes. This avoids needing a simulation warm-up period.

Once five minutes have elapsed from the beginning of the simulation, the first train enters the system at DeKalb station. Trains continue to arrive every five minutes from this point onward. They pass through the system on a fixed schedule, picking up all customers waiting at each station as they pass through it, then depositing all passengers at the Bedford station shuttle bus queue. Note that customers arriving as part of the Bedford Poisson process immediately join the shuttle bus queue. At the time of the first train arrival at Bedford station, shuttle bus departures begin. They continue for the duration of the simulation according to a specified schedule.

We use event-driven simulation, so the progress of the program through simulated time centers around a simulation clock holding the current time and an 18-element event list containing the times at which the next instances of the 18 different event types will occur. The first eight elements of the list correspond to the next customer arrivals at each station. The next eight elements of the list correspond to the next train arrivals at each station. The last two elements of the event list keep track of the next primary and secondary shuttle bus arrivals at Bedford station. At each iteration, the program checks which event type occurs next, processes the event type, and updates the simulation clock, event list, and system statistical counters. At the end of the simulation, the objective function is evaluated on the resulting total number of minutes customers spent waiting in the Bedford Ave shuttle bus queue and the total number of buses used given the specified schedule.

Algorithms. There are three algorithms that drive the simulation program. The reader is likely familiar with these algorithms, so we provide only cursory descriptions of our implementations. The first is a prime modulus multiplicative linear congruential generator (PMMLCG) that we use to generate uniform realizations from the interval $(0, 1)$.[5] The algorithm uses the following recursion equations:

$$z_0 = seed; z_i = 630360016 \cdot z_{i-1} (mod\ 2^{31} - 1),$$

where *seed* is an integer strictly between 0 and $2^{31} - 1$. The particular implementation used conveniently divides the random numbers into 100 streams, and, for reproducibility and optional variance reduction purposes, our program takes full advantage of this.

To generate from the standard normal distribution, we use the following algorithm:

1. Generate independent realizations U_1, U_2 from $U(0, 1)$.
2. Set $V_1 = 2U_1 - 1, V_2 = 2U_2 - 1, W = V_1^2 + V_2^2$.
3. If $W \leq 1,$ return $X = V_1\sqrt{-2\ ln(W)/W},$ otherwise go back to 1.

[5]We acknowledge Averill M. Law and the portable linear congruential generator program written in C available at www.mhhe.com/law, which we compile with our program and use to generate our uniform random variates.

The efficiency of this acceptance-rejection algorithm is quite high – about 1.27 – and it performs well in our implementation.

The third and final algorithm is used to generate customer interarrival times using the arrival rate function described above. Assuming the simulation clock currently has value t, and that we are considering a station with scaling factor K, we use the following algorithm to generate the next customer interarrival time for that station:

0. Set $t_0 = t$.
1. Generate independent realizations U_1, U_2 from $U(0,1)$.
2. Set $t_0 = t_0 - \dfrac{1}{K} ln(U_1)$.
3. If $U_2 \leq \dfrac{\lambda(t_0)}{K}$, return $t_0 - t$, otherwise, return to step 1.

Our implementation is relatively efficient in practice: when the arrival rate is lowest, the number of iterations of the algorithm seldom exceeds 10, while during peak hours only one or two iterations are performed.

Objective Function. The general form of the objective function that we use in determining cost efficiency of potential shuttle schedules is

$$F(S, \vec{\boldsymbol{X}}) = aW(S, \vec{\boldsymbol{X}}) + bB(S),$$

where a is the cost per minute delay experienced in the Bedford shuttle bus queue, b is the cost incurred by using a single bus, $\vec{\boldsymbol{X}}$ is a random variable representing the sequence of customer arrival and train unloading processes occurring at Bedford, S is the shuttle bus schedule, $B(S)$ is the number of buses used under schedule S, and $W(S, \vec{\boldsymbol{X}})$ is a random variable counting the total number of minutes waited under schedule S and subject to $\vec{\boldsymbol{X}}$.

Based on bus driver wages and other bus operating costs, we have good reason to believe that the value of b in the objective function given above is around \$30. The value of a, on the other hand, is much more difficult to hypothesize. Note that we use the word *hypothesize* – since a is in some sense subjective, attempting to *approximate* it would be misguided. Given our lack of information regarding a, we made the decision to find values of a that make the objective function sensitive to both variations in the value of $B(S)$ and $W(S, \vec{\boldsymbol{X}})$. When $a = \$0.5$, for example, the penalty incurred for each minute waited is so high that we should almost always use as many buses as possible in an effort to bring wait times close to 0. From our experiments, values of a between \$0.01 and \$0.05 provide a better trade-off. With the assumptions that buses cannot depart more frequently than once a minute, that no more than 7 buses can depart at a time (given bus length and the average length of a city block), and that average wait times under any candidate departure schedule should not exceed 10 minutes, values of a in this range give an objective function that shows a good sensitivity to both $B(S)$ and $W(S, \vec{\boldsymbol{X}})$. After many rounds of experimental runs, inspection of the results, and discussion, we settled on the value $a = 0.015$.

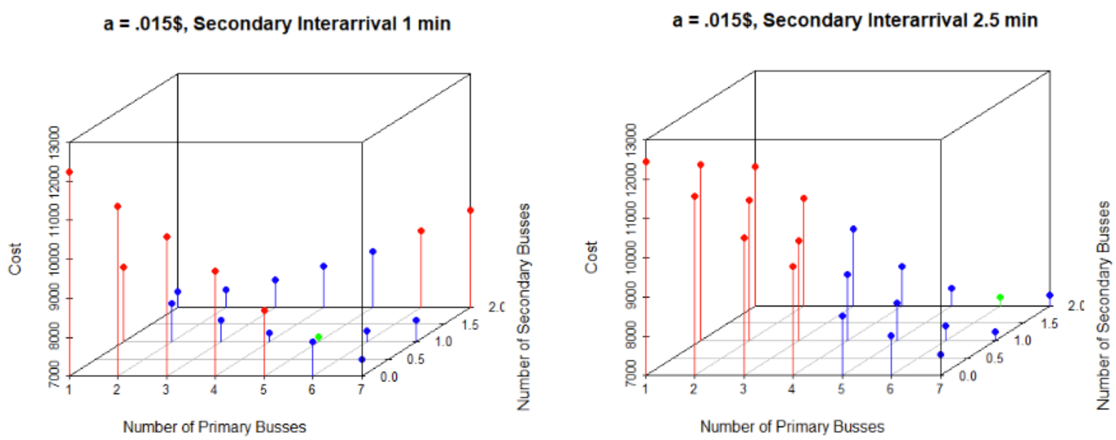

FIGURE 5. Point Estimates for Objective Function Values

Shuttle Bus Schedule. To select the most cost efficient bus schedule, we tested 42 distinct shuttle departure configurations. For each candidate schedule, we made 100 independent simulation replications, then used the output from these runs to determine confidence intervals for the resulting objective function values. The general shuttle bus schedule format is as follows: n primary buses are sent every five minutes to coincide with the train arrivals at Bedford, while m secondary buses are sent at intervals of k minutes strictly between primary bus arrivals. Figure 5 shows the results of trying all combinations of $n = 1, ..., 7$, $m = 0, 1, 2$, and $k = 1, 2.5$.

The best bus schedule given our cost function was five primary buses arriving every five minutes and one secondary bus arriving every minute (excluding minutes where the primary buses arrive). The intuition is that the primary buses are ready and waiting to depart with most of the arriving train's passengers as soon as they arrive, while the secondary buses arrive in between to pick up overflow train passengers and customers arriving at Bedford between train drop-offs. The average wait time of a person given this schedule is 5.03 minutes, which is well below our maximum average wait time of 10 minutes. The expected cost per day of this bus schedule is \$7119.77, with a 95% confidence interval given by $(7039.60, 7199.94)$. All other mean values of the cost function fall outside of this interval, with the closest being \$7207.80. As a result, of the different schedule configurations tested, on most days we can expect this bus schedule to be optimal.[6]

Assumptions and Limitations. For the purposes of our project, it was necessary to make several simplifying assumptions about the system which we are modeling: (1) trains arrive to the system empty, and all embarking passengers are Manhattan-bound; (2) all trains and buses arrive on schedule; (3) the cost per minute of customer delay experienced and operating cost per bus have the values that we specified.

For the first part of (1), due to the closure west of Bedford, we assume that no customers outside of our eight-station system will be willing to enter the system with the object of taking the Bedford shuttle bus. This is a reasonable assumption, since customers getting on east of DeKalb Ave station have the option to transfer to other subway lines heading into Manhattan prior to entering our system. In order to analyze the performance of the shuttle bus schedule in the worst-case scenario, we assume that all passengers embarking in our system are Manhattan-bound, and thus get off at Bedford Ave. If we had enough information to fit accurate distributions to the disembarkation processes, however, the program is designed to incorporate them. For (2), though the arrival times of trains and buses are in reality random variables, we assume they arrive on time for the simple reason that – on average – they do. Another simple reason for this assumption is that we had no data with which to determine the distributions of train and bus arrival times. Finally, for the tractability of our project, we had to hypothesize values for the costs associated with the objective function. Given more time and access to more information or to experts at the MTA familiar with such matters, we could come up with more accurate costs. Given the limitations of the class project, however, we think the values we give are reasonable.

These assumptions impose limitations on the real-world applicability of our model. On the one hand, the fact that our trains arrive to the system empty imply that the quantity of arrivals we see at Bedford is less than what would actually occur. On the other hand, since all of our embarked passengers disembark at Bedford, we expect that our model somewhat overestimates the actual number of arrivals to this station. We must also mention that the turnstile data we obtained counts customer entries into the station, but *does not* indicate the direction in which they are travelling. Since we assume all entries are Manhattan-bound, further inaccuracy is possibly induced, despite the fact that the great majority of morning rush hour entries are likely heading in that direction. The implications of (2) and (3) above for our model are clear. Given more time, more information, and access to experts at the MTA, however, we believe these assumptions could be modified to accurately reflect reality.

Conclusion

In this paper we have discussed the details of an effort that has occupied much of our time and minds over the past month. We traced the transition from the real-world

[6]See the table in Figure 6 at the end of the paper for detailed output information.

situation motivating our simulation study to the construction of the model that forms the conceptual basis for it. Using this model as our framework, we described in broad strokes the simulation program built around it, as well as the nature of the data we used to determine our input distributions and our statistical analysis of it. Once this general outline was laid down, we took a look at some of the algorithms we used. With a solid understanding of the model and program in hand, we detailed considerations surrounding our objective function, the fundamental tool we use to explore answers to the seemingly simple question: what shuttle bus schedule is best? We finally described the experiments we ran with our objective function over several alternative bus schedule configurations, their results and our analysis of them, and we concluded with our recommended schedule. Despite the limitations of our model and the consequent restricted utility of our study, what we have done is still a significant accomplishment, and it demonstrates a solid grasp of the foundations and practice of this course. We have learned a great deal about the mathematical and programming tools needed to construct a valid simulation model, and we have gone into some depth in applying them to model an interesting and important part of the New York City public transportation system.

	run_number	primary_buses_per_c	primary_bus_interam	secondary_buses_pe	secondary_bus_inter	total_buses_used	total_minutes_waite	customer_total_arriv	average_minutes_wa	total_cost	Standard Deviation	95 Confidence Interv
1	100	1	5	0	1	21	773706.95	19765.46	39.121175	12235.6105	880.422448	174.6949146
2	100	1	5	1	1	103	387852.15	19953.5	19.349089	8907.7823	792.6895602	157.2868063
3	100	1	5	2	1	185	123413.477	20006.16	6.1153857	7401.2013	393.1836395	78.01616425
4	100	2	5	0	1	42	673782.58	19858.38	33.893142	11366.7381	789.380966	156.6303094
5	100	2	5	1	1	124	284146.47	19873.07	14.181032	7982.1966	822.3803229	163.1780977
6	100	2	5	2	1	206	84933.084	19794.44	4.2143268	7453.996	401.9927719	79.76408725
7	100	3	5	0	1	63	578861.52	20033.62	28.833922	10572.9245	850.7619499	168.8096283
8	100	3	5	1	1	145	213709.74	20079.61	10.5480728	7555.6461	656.2997011	130.2240992
9	100	3	5	2	1	227	58429.388	19877.9	2.896267	7686.4413	265.2726141	52.63584177
10	100	4	5	0	1	84	479569.86	20131.88	23.725167	9713.547	919.4315753	182.4351718
11	100	4	5	1	1	166	149815.321	19937.57	7.4355996	7227.2307	525.3037904	104.2316686
12	100	4	5	2	1	248	39971.622	20033.52	1.9618235	8039.5747	208.2865271	41.32856579
13	100	5	5	0	1	105	368911.01	19980.59	18.361535	8683.6658	835.9411758	165.8688652
14	100	5	5	1	1	187	100651.039	19776.15	5.0266483	7119.7668	404.0395562	80.17021364
15	100	5	5	2	1	269	24505.6157	19891.1	1.20759	8437.5846	141.4777134	28.07224771
16	100	6	5	0	1	126	274434.91	19930	13.6776823	7896.5238	699.2180876	138.7400382
17	100	6	5	1	1	208	69770.926	19909.58	3.44061722	7286.5639	360.213267	71.47412706
18	100	6	5	2	1	290	17091.9417	20204.92	0.82439114	8956.3795	126.0246309	25.0060209
19	100	7	5	0	1	147	202193.48	19944.86	10.0547628	7442.9014	590.1256164	117.0937252
20	100	7	5	1	1	229	45701.5404	19985.96	2.23815969	7555.5227	268.6366042	53.30333038
21	100	7	5	2	1	311	10791.8888	20106.11	0.52767848	9491.8785	56.55901953	11.22253653
22	100	1	5	0	2.5	21	788500.69	20118.24	39.169063	12457.507	884.3295407	175.4701666
23	100	1	5	1	2.5	42	681965.31	19922.53	34.18153	11489.4821	919.3660496	182.42217
24	100	1	5	2	2.5	63	576921.06	19780.4	29.112499	10543.8193	785.7451641	155.9088874
25	100	2	5	0	2.5	42	687371.41	20147.16	34.085578	11570.5743	743.7278276	147.5717363
26	100	2	5	1	2.5	63	578450.89	19924.48	28.979144	10566.766	786.0714501	155.9736296
27	100	2	5	2	2.5	84	480664.13	19939.44	24.028079	9729.9609	820.6403433	162.832848
28	100	3	5	0	2.5	63	573980.91	19922	28.748492	10499.7112	846.4132162	167.9467452
29	100	3	5	1	2.5	84	467690.29	19749.96	23.587278	9535.3544	883.4133641	175.2883772
30	100	3	5	2	2.5	105	387868.38	20185.65	19.102918	8968.0259	890.0623357	176.6076774
31	100	4	5	0	2.5	84	483179.94	20212.66	23.820176	9767.7034	867.116563	172.0547383
32	100	4	5	1	2.5	105	368893.81	19861.81	18.448053	8683.4069	902.9166239	179.1582471
33	100	4	5	2	2.5	126	281847.68	20018.9	13.9863405	8007.7146	718.821104	142.629702
34	100	5	5	0	2.5	105	358479.9	19721.87	18.069669	8527.1971	833.965191	165.4767869
35	100	5	5	1	2.5	126	277598.6	20028.37	13.7577117	7943.9786	758.0146093	150.4065437
36	100	5	5	2	2.5	147	202324.143	19894.64	10.0826721	7444.8618	602.2062751	119.4907899
37	100	6	5	0	2.5	126	282688.53	20139.94	13.9279441	8020.3284	787.4316163	156.2435161
38	100	6	5	1	2.5	147	197230.98	19876.76	9.8306853	7368.4634	615.5242597	122.133367
39	100	6	5	2	2.5	168	144520.136	19892.45	7.1884432	7207.8028	512.3443514	101.6602347
40	100	7	5	0	2.5	147	208359.38	20111.92	10.261739	7535.3906	656.9303424	130.3492321
41	100	7	5	1	2.5	168	145592.105	20035.59	7.1522991	7223.8814	622.5712807	123.5316489
42	100	7	5	2	2.5	189	106038.463	20114.79	5.1941492	7260.5769	453.2920965	89.94298618

FIGURE 6. Output Data