Moving More With Less: Examining the Efficiency of the TWIN Elevator System in the CODA building

Chuyun Sun
chuyunsun@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Youyi Shi
shiy7@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Yong Jian Quek
yjquek@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

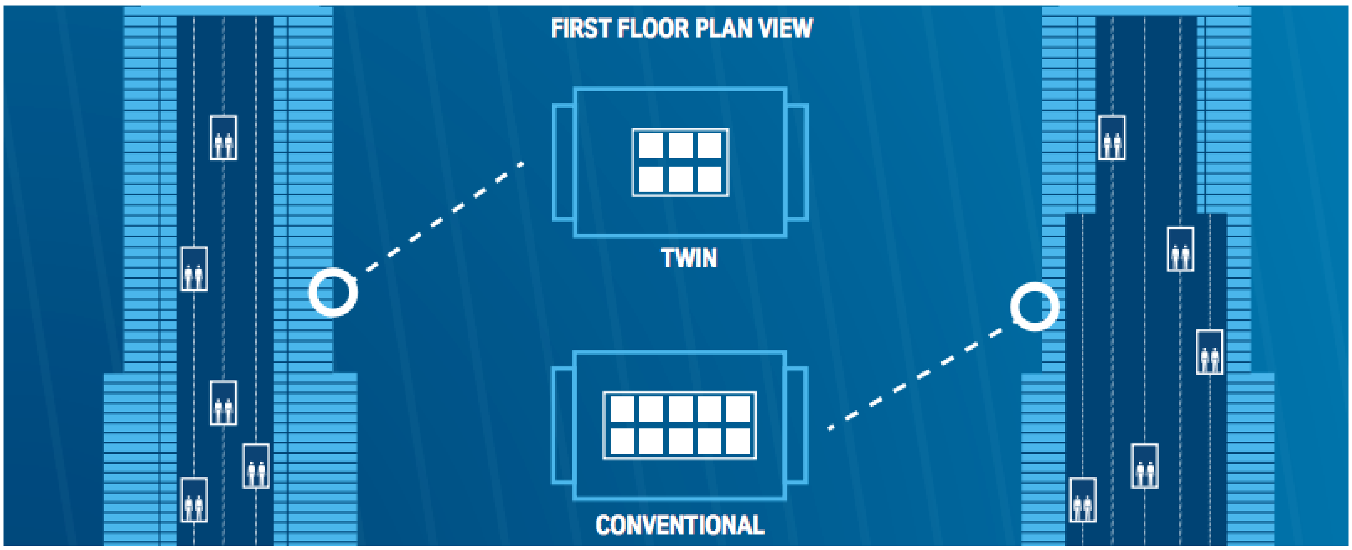

Figure 1: Example of a double cab lift system

ABSTRACT

1 INTRODUCTION

The boom in finance and technology services have seen companies and institutions cluster together in limited physical space, increasing the premium placed on land space, and with it, a rapid growth in the number of high rise buildings [2]. Moving people in high rise buildings however, remains relatively inefficient as determining the order in which to ferry people is a NP-Complete problem [9]. Multiple solutions exist to alleviate the situations, for example, introducing express elevators and skylobbies [10] or by using double-deck cabs that stop at two floors simultaneously [5]. The TWIN system by ThyssenKrupp however, utilizes two independent cabs in each elevator shaft and is advertised as being more efficient than traditional systems, both in terms of energy consumption and user throughput [11].

2 LITERATURE REVIEW

There are many studies on simulating the elevator system. Henriques, et al. used SIMIO software to simulate an elevator system with discrete event simulation in 3D animation. Their analysis of many key performance indicators, such as the time in the lift, the occupation ability of the elevator, and the waiting time for the elevator, can be used as a reference to help the group to set up the conceptual model [7]. Davidrajuh developed a MATLAB Toolbox called ElevatorSIM for discrete-event simulation for the elevator system, which can provide useful information like waiting time and service time [6]. In the terms of minimize waiting time in lift, Sliva, et al. conducted a similar research. They utilized the heuristic method to analyze queues and processors/servers [4]. The double-deck elevator system (DDES), where two cars are connected in a shaft and move simultaneously, has been also used to increase the efficiency of the elevator. Kim, et al. modelled both single-cage lift and double-cage lift using discrete event simulation in the CYCLONE system and observed a 38% decrease in lifting time during the morning peak, which agrees with the group's expectation [13]. To improve the efficiency of elevator system, Hirasawa, et al. designed an elevator group supervisory control systems (EGSCSs) which applied genetic network programming (GNP) and tested with both double-deck and single-deck elevators. With this new system, it was confirmed that DDES has reduced space and increased efficiency compare with the traditional single-deck elevator system [8]. Ding, et al. also applied genetic algorithm for the group optimization dispatching. By doing so, they successfully reduce the average waiting time, long waiting time incidence and the numbers of elevator stops [3]. Finally, Liew, et al. discussed other strategies can be useful in a multi-car elevator system, such as control strategies, car collision avoidance strategies [14].

3 PROBLEM DESCRIPTION

The TWIN elevator differs from traditional elevator systems by having two cabs in one elevator shaft. These two cabs are completely independent and can move in different directions simultaneously. The system is able to select the optimal cab to dispatch to the user as they select their destination before entering the elevator [14]. In designing an optimal system for the users, we hope to achieve these goals.

(1) Maximize Throughput of users
(2) Minimize average waiting time
(3) Minimize average time spent in the lift

The average time spent in the lift is nonzero for a TWIN system as there are occasions where the cab has to wait for the other cabs in the shaft to clear. These goals can be achieved by varying the algorithm used to dispatch the optimal cab by varying the following parameters

(1) Algorithm to assign an elevator to a user
(2) Elevator behavior when at maximum capacity?
(3) Operating range of each elevator cab

For this specific project, we will be investigating if the compromise between selecting a cab with no collision, or accounting for time to resolve the collision before selecting a cab. Selecting a cab with no collision should ideally give us shorter wait times if there are enough shafts available, where resolving the collision should do better with a limited number of shafts.

The system is also subjected to the constraint where cabs have no means of passing through each other. Thus the upper cab always remains on top of the lower cab, with the upper cab unable to reach the lowest floor and the lower cab unable to reach the highest floor.

Furthermore, stochastic elements exist within this problem as the arrival of users, as well as their destination, is of a random nature. Thus, determining an optimal algorithm for this system remains complex and in such a situation, a Discrete Event Simulator (DES) provides better insights as it mimics real world situations. This can be seen as an extension of using DES to model traditional elevator systems. [1].

4 CODA BUILDING TWIN ELEVATOR SYSTEM

The CODA building in Atlanta, GA, is the first implementation of the TWIN elevator system in North America [12] and will thus provide the parameters needed for the model. This system consists of 21 floors and 6 elevator shafts for a total of 12 cabs.

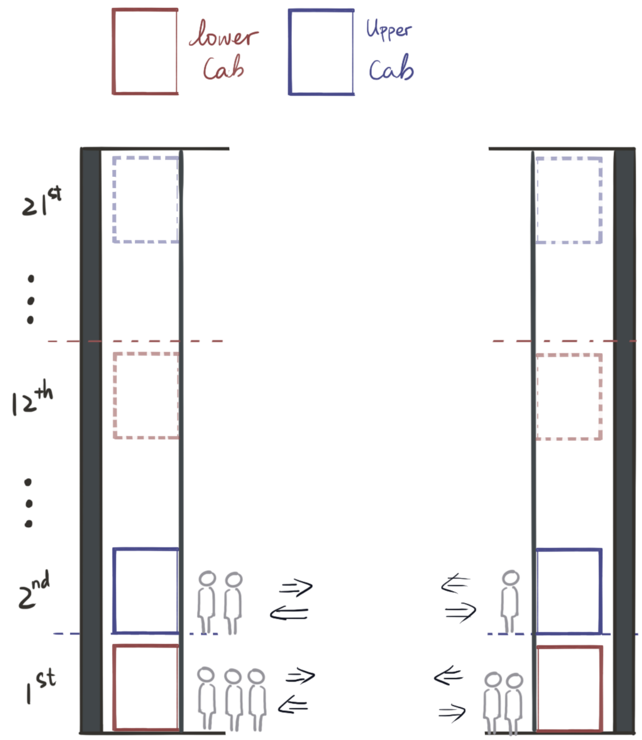

Figure 2: Elevator System in CODA building

5 CONCEPTUAL MODEL

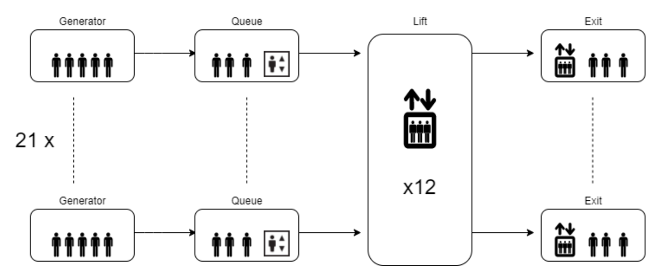

Figure 3: Conceptual model of the TWIN elevator system

5.1 Simplifications and Assumptions

- The upper cab is restricted to floors 2 to 21 and the lower cab is restricted to floors 1 to 12. This models the existing system in the CODA building.
- Cabs start initially all stacked at the bottom when $t = 0$ and idle at their last destination after.
- Each group that arrives in the model is below the maximum number of occupants allowed in the cab.
- Time taken to load and unload the users are negligible.
- Traveling time needed for each floor is assumed to be constant.
- If a cab is unable to reach its destination as another cab is blocking. It will wait for the other cab to reach it destination,

and if it is still blocked, the other cab will be moved one level away from the original cab's destination.

5.2 Structural View

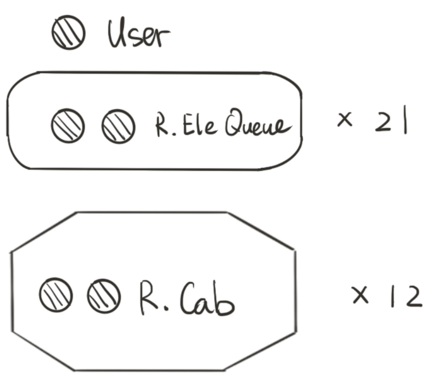

Figure 4: Structural View

5.2.1 *Entities.*

User A Consumer entity with a scope of Class. Represents a group of users (less than the maximum number allowed onto a single cab) that are heading to the same destination.

Cab A Resource entity with a scope of 12. Represents a single elevator cab.

EleQueue A Queue entity with a scope of 21. Each floor has its own EleQueue that represents the users waiting for the elevator on each floor.

5.3 Behavioural View

5.3.1 *Actions.*

UserArrival Generates a stream of users. Users are directly placed into the EleQueue of the corresponding floor with a generated destination.

UserExit User exits the system.

5.3.2 *Activities.*

Travelling Represents the user in the elevator moving from the source to the destination.

5.4 Structural Components

Table 1: Constants

Name	Description	Value
TRAVEL_TIME	Time taken for a lift to traverse 1 floor	2s

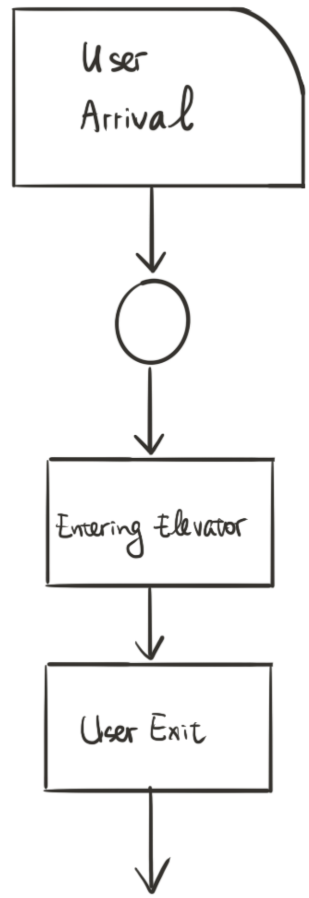

Figure 5: Behavioural View

Table 2: Consumer Class: User

Attributes	Description
startTime	Time the user arrived
waitForEle	Time spent waiting for the elevator
timeInEle	Time spent waiting in the elevator
dest	User's destination

Table 3: Resource Set[12]: Cab

Attributes	Description
pos	Indicates if the cab is an upper or lower cab. 0 if lower cab, 1 if upper cab.
avail	Indicates if cab can be assigned to a user. 1 if cab is occupied or assigned, 0 otherwise.
dest	Destination of the cab if it is assigned. 0 otherwise.

Table 4: Queue Set[21]: EleQueue

Attributes	Description
q	A list of users in the queue in first in, first out order
n	Number of users in the queue

5.5 Behavioral Components

Time Units Seconds

Observation Interval Starts at t = 0 to t = 3600 to represent lunch hour

Table 5: Random Variate Procedures

Name	Description	Data Model
UserRandArr(src)	Returns the time to the next user arriving on floor src	Triangular Distribution
UserRandDest(src)	Returns a possible random destination for the user from the src	Multinomial Distribution

Table 6: Action: UserArrival

Time Sequence	UserRandArr(User.src)
Event SCS	User.dest ⟵ UserRandDest(User.src) User.startTime ⟵ 0 User.waitForEle ⟵ 0 User.timeInEle ⟵ 0 InsertQ(EleQueue[UserRandSrc()],User)

Table 7: Action: UserExit

Time Sequence	UserRandArr(User.src)
Event SCS	User.exit()

Table 8: Activity: Travelling

Precondition	CabAssign() and User front of queue
Event SCS	User ⟵ RemoveQ(EleQueue[NextQ]) User.waitForEle ⟵ t - User.startTime + abs(User.dest - User.src) User.timeInEle ⟵ abs(Cab.curr - User.src))
Duration	TRAVEL_TIME × (abs(User.dest - User.src) + abs(Cab.curr - User.src))
Event SCS	User.waitForEle ⟵ TRAVEL_TIME × (abs(Cab.curr - User.src))

6 SIMULATION MODEL

6.1 Architecture

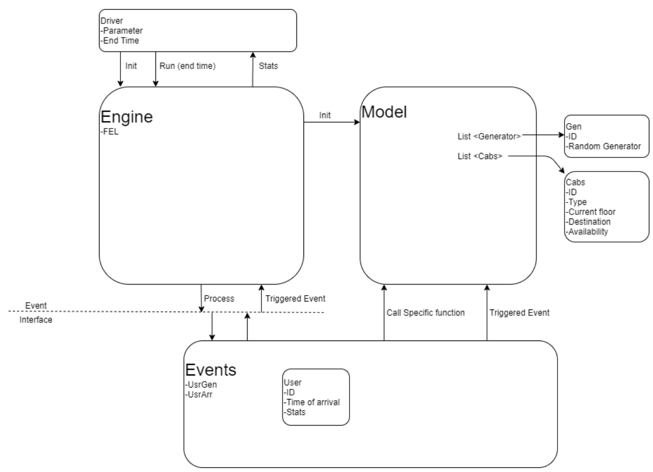

Figure 6: Elevator System in CODA building

The architecture is based on Object-Oriented Principles and utilizes polymorphism to simplify the Engine component. Below is a description of each major component.

Driver Test driver program to run simulation. Contains hard coded parameters for now.

Engine Main engine that consumes and schedules the future event list.

Model Contains information on all state variables. Processes events according to state.

Events Abstract event class with specific events extended from it. Allows for dynamic binding.

Generators Random generators for user inter-arrival rates.

Cabs Holds availability and other state information for elevator cabs

The driver will initialize the engine with the required parameters. The engine then initializes the model which builds the required components and sends the initial user list to the engine. The engine can then consume the event list and schedule incoming triggered events until the designated time.

7 DATA COLLECTION

To determine the two stochastic elements, empirical data was collected over 2 days in the CODA building during the peak lunch hour. The time and destination of groups of users was collected on the 1st, 2nd, 12th floor.

7.1 Inter-arrival Rate

As the inter-arrival rate is a measure of time, the empirical data needs to be fitted to a continuous distribution. To fit the distribution, the time taken between each user was first calculated from the empirical data of Day 1. The data was then fitted to various different distributions using a maximum likelihood estimation algorithm to find the optimal parameters. The empirical data of Day 2 was then used to determine the goodness-of-fit by comparing the observed frequencies to the expected frequencies using a χ^2 test. The distribution with the highest p-value was used for this model and satisfies a 95% confidence interval as $p > 0.05$. The output analysis can be found in the Appendix.

7.2 User Destination

The destination resembles a discrete distribution as they are numerical in nature, but statistically, they represent categorical data. Thus, only a categorical distribution is suitable. To determine the probability of each floor, the proportion of destinations are taken.

7.3 Travel Time

As previously assumed, the travel time for a cab to move one floor is assumed to be constant. The time for this was empirically measured by measuring the time it takes to go from the 2nd to the 21st floor and dividing by the number of floors travelled.

8 VERIFICATION

Verification was conducted by tracing the flow of fixed users through the simulation. Rather than use a stochastic user generator, users were injected in at a fixed rate. A verbose version of the simulator was used to trace the flow of users through the system. Verification was thus achieved when the simulation produced the same results as the analytical solution. Various checkpoints were placed to ensure that each module functions as expected. Logs of the test code and test generators are available together with the simulation program.

8.1 Engine

- Engine initialized the model and retrieved initial user list from model.
- Engine consumes the first even in the future event list and schedules the triggered events as intended.

8.2 Model

- Model initializes required components and sent initial user list to the engine as intended.
- Model processed events according to state as intended.
- Generators randomly generated user inter-arrival rates as intended.

8.3 Events

- Polymorphic events trigger different functions in the model

9 VALIDATION

Our ideal goal for validation was to utilize data collected from the elevator themselves as the manufacturer, ThyssenKrup, is located within the CODA building itself. However, as they were unresponsive to multiple requests for assistance, we were only able to verify the number of people going through the elevator system. The algorithm for assigning an elevator cab can be seen as new algorithms that were designed for this project. Additionally, for ease of designing the algorithm, we assumed that the cabs do not stop midway but unfortunately in the real system, they have been observed to do so. To verify the number of people flowing through the system, we

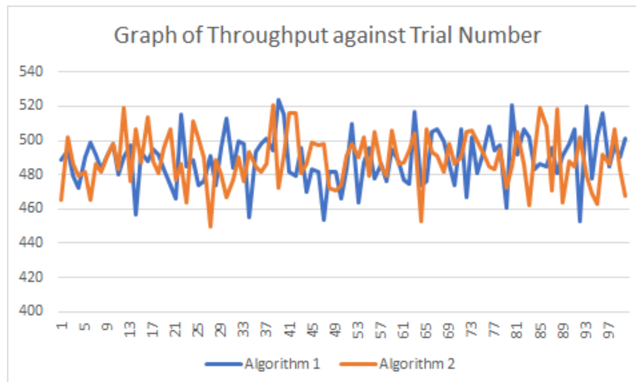

Figure 7: Throughput of Users for Both Algorithms

again measured a set of empirical data on another day (Day 3). We then compared the output of our generators over 50 trials to the empirical data. As the throughput of the users is not the mean of a sample, central limit theorem does not apply and we are unable to assume its distribution as normal. We thus use a two sample t-test instead, with one sample being the simulated trials, and the other being the throughput observed over 3 days. The p-value found was 0.1796 and thus lies within the 99% confidence interval. This thus verifies that our simulation's throughput reflect that of reality.

10 EXPERIMENTS

10.1 Methodology

The independent and dependent variables for the experiment follows from the problem description.

Independent Variable
The assignment algorithm, each with different cost functions.

Dependent Variable
Throughput of Users, average waiting time, average time spent in the lift.

Time Horizon 3600 seconds during the peak lunch hour of 12 noon to 1pm.

10.2 Assignment Algorithms

The test scenarios are also modeled after the existing elevators in the CODA building and thus have limited operating range to reduce the complexity of the system. The lower cab will be limited to floors 1 to 12, while the upper cab will be limited to floors 2 to 21. This can be seen in Figure 2. A special scenario exists when users depart from either the 1st or 2nd floor. From the 1st floor, users can only reach floors 2 to 12. Users who wish to reach floors 13 to 21 must take the escalator to the 2nd floor first before riding the elevator. Any elevator currently bringing users to their destinations are considered as unavailable and will not be dispatched. Thus, no elevators will be stopped midway. Any elevators en route to pick a user up is also considered unavailable unless the users are already on the same floor. We will experiment with different limits on the number of stops an elevator can make.

For this project, we will be testing two different algorithms for assigning an elevator to a user. Both algorithms are based on a greedy paradigm, but differ in how they measure the cost.

(1) The first algorithm selects a valid cab that can reach the user the fastest without colliding with another cab if possible. The cost function does not take into account the time taken to resolve collision as seen in 1. This should favor systems with a higher number of shafts.
(2) The second algorithm selects a valid cab that, including waiting for collisions to resolve, reaches the user the fastest. The cost function accounts for the time to resolve collisions as seen in 2. This should favor systems with a lower number of shafts.

In both cases, collision are resolved by waiting for both cabs to be available, before moving the blocking cab off the blocked spot by 1 floor.

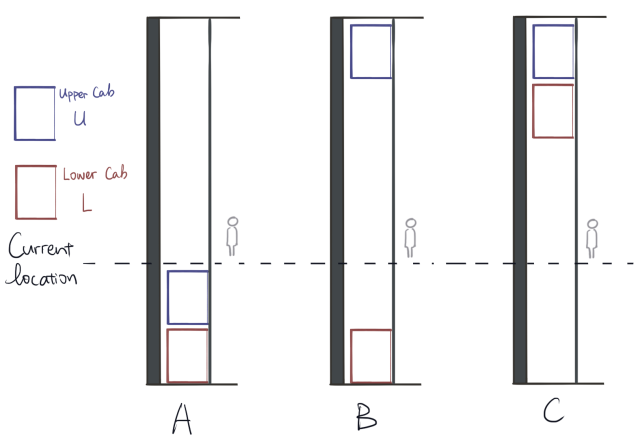

Figure 8: Test scenario

Table 9: State Variables in Model

Variable	Value
currTime	Current Time
cab.curr	Current Location of Cab
cab.nextAvail	Next Available Time of Cab
other.nextAvail	Next Available Time of Other Cab
u.src'	User's Location

Algorithm 1 Cost Function for Cab without Collision Resolution

```
function FINDTIMETOUSER(cab)
    distance ← |cab.curr − u.src|
    if cab.nextAvail < currTime then
        timeToUser ← currTime + distance ∗ TravelTime
    else
        timeToUser ← cab.nextAvail + distance ∗ TravelTime
    end if
end function
```

Algorithm 2 Cost Function for Cab with Collision Resolution

```
function FINDTIMETOUSER(cab)
    other ← otherCab(cab)
    distance ← |cab.curr − u.src|
    if cab.availTime < currTime then
        if !isCollide(cab, currTime) then
            timeToUser ← currTime + distance ∗ TravelTime
        else
            timeToUser ← other.nextAvail + distance ∗ TravelTime
        end if
    else
        if !isCollide(cab, currTime) then
            timeToUser ← cab.nextAvail + distance ∗ TravelTime
        else
            timeToUser ← other.nextAvail + distance ∗ TravelTime
        end if
    end if
end function
```

11 RESULTS

11.1 Analysis

To analyze the results of the simulation, we can assume that the mean times produced in the output are constrained by central limit theorem, and thus tend towards a normal distribution as the number of trials increase. For the first algorithm, with 100 trials, the data has a mean of 38.57 and a variance of 39.84. Thus, with a 99% confidence interval and using a z-test, we can say that the lowest possible mean is 22.39. For the second algorithm, also with 100 trials, the data has a mean of 9.42 and a variance of 0.5436, thus giving us a highest possible mean of 11.31. We can thus conclude with a 99% confidence level that the second algorithm has a shorter wait time than algorithm 1. Given that the random generators for both algorithms are the same and thus would have the same distribution of travel times, the combined waiting and travel times for the second algorithm will be shorter as well.

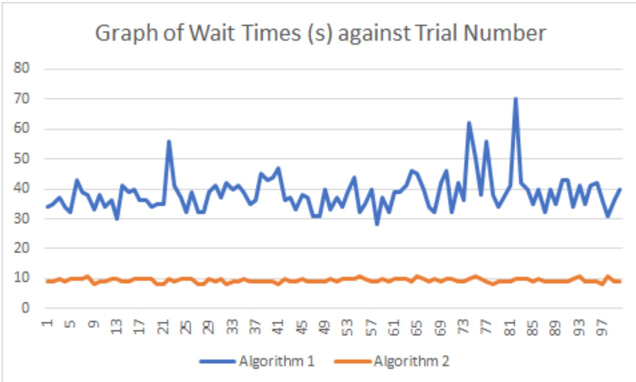

Figure 9: Wait Times for Both Algorithm over 100 Trials

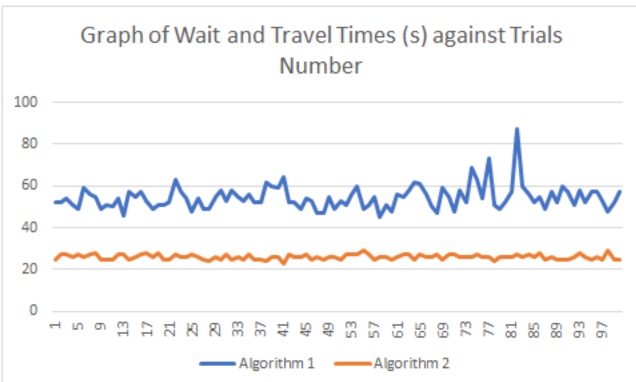

Figure 10: Wait and Travel Times for Both Algorithm over 100 Trials

As stated earlier, greedily selecting cabs with no collisions should favor systems with more shafts whereas including the time to resolve collisions should favor more constricted systems. Given that the second algorithm is better at transporting users than the first, it can be concluded that 6 shafts is too little to take advantage of non colliding cabs.

12 CONCLUSION

The results of the experiment shows that integrating the time needed for collision resolution into the cost function when assigning a cab performs better than greedily selecting a shaft with no collision. This implies that there are rarely empty shafts available and thus for the current level of load, the system is well utilized with cabs rarely free to idle. This being achieved with most wait times below 10 seconds is a good indication that the system would be struggling with higher wait times if there was only 1 cab per shaft. Thus, the system in the CODA building stands as a testament to the innovation of ThyssenKrupp's TWIN elevator product.

13 FURTHER RESEARCH

The simulation built for this project is both extensible and modular. Simulating additional cabs in the shaft or increasing the number

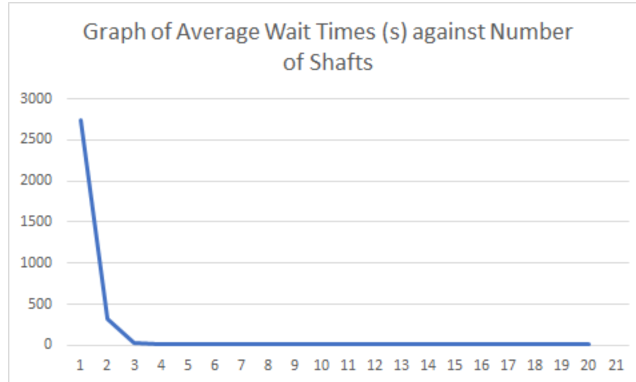

Figure 11: Extending the program to vary the number of shafts for Algorithm 2

of shafts is as simple as changing a parameter in the Driver code. A sample extension is given in 11, where the number of shafts is increased.

Furthermore, this project was performed with the goal of optimizing throughput and reducing wait times, parameters important from the users' perspective. From the building owners perspective however, our parameters such as energy consumption and maintenance cycles are of greater interest. Such statistics can be added into the Stats object so they may be optimized.

Finally, from our empirical study, there is an unusual pattern in the users' destination where there is a significant spike in the number of users going to the 15th floor. We postulate that this is due to the free coffee machine available there, skewing the data collection.

14 PROJECT DISTRIBUTION

Table 10: Project Distribution

Member	Report	Code
Chuyun	Literature Review Conceptual Model Simulation Model	Engine, Driver, Statistics
Youyi	Data Collection Verification, Validation	Component, Events, Model, Algorithms
Yong Jian	Experiments, Results, Conclusion Future Research	Model Statistics OOP Design Algorithms

REFERENCES

[1] Lutfi Al-Sharif and Mohamed D Al-Adem. 2014. The Current Practice of Lift Traffic Design Using Calculation and Simulation. *Building Services Engineering Research & Technology* 35, 4 (jul 2014), 438–445.

[2] Martin Neil Baily and Nicholas Montalbano. 2017. *Clusters and Innovation Districts: Lessons from the United States Experience.* Technical Report. Washington, D.C.

[3] Qing-chao Li Jin Zhang Bao Ding, a and Xiao feng Liu. 2013. Twin elevator group optimization dispatching based on genetic algorithm. *Applied Mechanics and Materials* 415 (Sep 2013), 95–100.

[4] Luiza R. Vitor-Daywes Pinheiro Neto Leandro Kazu Wesley Pacheco Calixto Clebes Andre Silva, Danilo Fernando A. Silva. 2017. Simulation and minimization of waiting time in rows of elevators of public buildings. *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)* (Oct 2017), 1–6.

[5] Otis Elevator Company. 2001. *Burj Khalifa.* Retrieved January 18, 2020 from https://www.otis.com/en/hk/projects/showcase/burj-khalifa/

[6] Reggie Davidrajuh. 2019. Developing a Toolbox for Modeling and Simulation of Elevators. *Int. J. Simul. Syst. Sci. Technol* 20 (Mar 2019), 1–1.

[7] Luís M.S Dias ; Guilherme A.B. Pereira Marcelo F.N. Henriques, António A.C. Vieira and José A. Oliveira. 2019. ANALYSIS OF AN ELEVATOR SYSTEM USING DISCRETE EVENT SIMULATION: CASE STUDY. *International Journal for Quality Research* 13, 4 (Dec 2019), 823–836.

[8] Kotaro Hirasawa ; Toru Eguchi ; Jin Zhou ; Lu Yu ; Jinglu Hu ; Sandor Markon. 2008. A Double-Deck Elevator Group Supervisory Control System Using Genetic Network Programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 4 (2008).

[9] Daniel Nikovski and Matthew Brand. 2003. Decision-Theoretic Group Elevator Scheduling. In *Proceedings of the Thirteenth International Conference on International Conference on Automated Planning and Scheduling.* AAAI Press, Palo Alto, CA, 133–142.

[10] Amber C. Snider. 2017. *Paying Homage to the Past: The Original Twin Towers.* Retrieved January 18, 2020 from https://theculturetrip.com/north-america/usa/articles/paying-homage-to-the-past-the-original-twin-towers/

[11] ThyssenKrupp. 2016. *TWIN from ThyssenKrupp Elevators.* Retrieved January 18, 2020 from https://twin.thyssenkrupp-elevator.com/home

[12] ThyssenKrupp. 2019. *Coda building in Atlanta, featuring revolutionary TWIN elevators from thyssenkrupp, now open to tenants.* Retrieved January 28, 2020 from https://www.prnewswire.com/news-releases/coda-building-in-atlanta-featuring-revolutionary-twin-elevators-from-thyssenkrupp-now-open-to-tenants-300856254.html

[13] Taehoon Kim Seung Woo Kim Hong-Ku Jung Hunhee Cho Wansoub Kim, Dong-min Lee and Kyung-In Kang. 2016. Evaluation of lifting efficiency of double-cage construction lift in supertall building construction. *SARC 2016 - 33rd International Symposium on Automation and Robotics in Construction* (2016), 787–795.

[14] Michael Loong Peng Tan-Chee Wei Tan Yeong Cherng Liew, Cheng Siong Lim. 2015. A review of multi-car elevator system. *Jurnal Teknologi* 73, 6 (2015).

Appendices

15 TEST LOGS

Figure 12 is the output for the verification tests using the test generator. Both test modules and logs are available with the simulation.

16 DISTRIBUTION FITTING

Figure 13 is the output for fitting the distribution to the inter-arrival rate. The fitting script and output is available with the simulation.

```
Setting up generators
Setting up cabs
Initializing Model
Scheduling user 0 at time 5
Cab 0 assigned to user 0 from 1 to 12 and reaches destination at 29
Scheduling user 1 at time 10
Cab 2 assigned to user 1 from 1 to 11 and reaches destination at 32
Scheduling user 2 at time 15
Cab 4 assigned to user 2 from 1 to 10 and reaches destination at 35
Scheduling user 3 at time 20
Cab 4 assigned to user 3 from 1 to 11 and reaches destination at 75
Number of Users Arrived: 4
Number of Users Exited: 0
Average Time Spent Waiting for Cab: 10
Setting up generators
Setting up cabs
Initializing Model
Scheduling user 0 at time 5
Cab 0 assigned to user 0 from 1 to 5 and reaches destination at 15
Scheduling user 1 at time 10
Cab 2 assigned to user 1 from 1 to 11 and reaches destination at 32
Scheduling user 2 at time 15
User 0 exits at time 15
Cab 4 assigned to user 2 from 1 to 10 and reaches destination at 35
Scheduling user 3 at time 20
Cab 6 assigned to user 3 from 1 to 5 and reaches destination at 30
Number of Users Arrived: 4
Number of Users Exited: 1
Average Time Spent Waiting for Cab: 2
Average Time Spent Waiting and Travelling: 10
```

Figure 12: Output for Verification Testing

```
chi2                5.915277367089943e-33
dgamma              4.91156139320677e-19
dweibull            1.698342642303941e-12
expon               0.0003531555063476255
exponnorm           0.0003019723721368797
exponweib           6.3240703859088595e-27
exponpow            1.1588509652343573e-09
fatiguelife         0.0002742987960220684
fisk                0.005382238126773326
foldnorm            0.009612699578949536
genlogistic         0.002129314572411556
gennorm             1.4447109059290202e-06
genexpon            0.0015188596530935523
genextreme          0.006642002715029979
gausshyper          1.3830429374968665e-05
gamma               3.091382906424046e-138
gengamma            2.362561322355154e-36
genhalflogistic     0.004138815575854333
gilbrat             2.074221495413533e-05
gompertz            0.0016207083902943322
gumbel_r            0.002324983991135462
gumbel_l            6.344052971661489e-10
halflogistic        0.004048335571641022
halfnorm            0.0026253490198177554
halfgennorm         1.541383785770579e-29
hypsecant           2.27927836895139e-06
invgauss            0.003150012038249875
johnsonsb           1.763042224306601e-70
johnsonsu           0.0038739068162546766
kstwobign           0.0013933562412997737
laplace             9.49555731287822e-11
logistic            9.114801360704792e-09
loggamma            1.6542836548451375e-11
lognorm             0.003694287866077548
maxwell             6.529353414530258e-07
nakagami            4.746672055748691e-07
ncx2                0.014814771113863034
norm                1.1746487753907456e-11
pearson3            7.12134000840339e-09
powerlaw            0.0003698229141980259
rayleigh            1.1867071783828652e-05
rice                1.189188064635454e-05
recipinvgauss       6.180982577113659e-12
skewnorm            0.002617097318369539
t                   9.403383485208651e-07
triang              0.13877835581709252
truncexpon          0.02367996375560209
wald                3.4510931653822435e-05
weibull_min         7.340562241644532e-05
weibull_max         5.414320664017997e-127
Best model is triang with p value of 0.13877835581709252
(1.6879844483063105e-07, -1.8417858632779098e-06, 7.6560728288703945)
```

Figure 13: Output for Distribution Fitting

17 EMPIRICAL DATA COLLECTION

Only the processed data is presented here as the raw tables are of significant length.

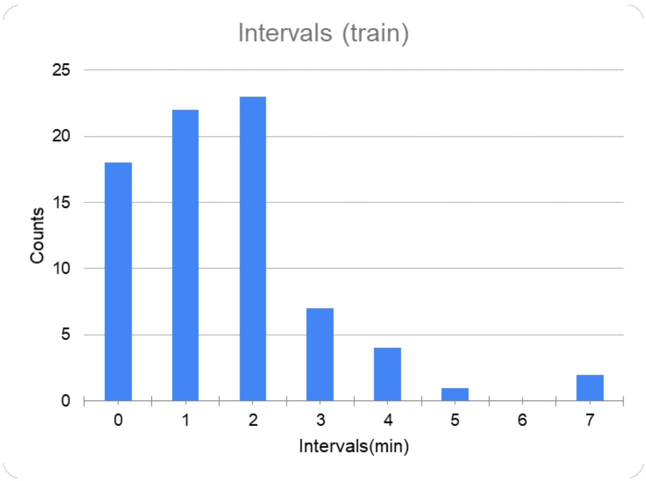

Figure 14: Interarrival Rate (Train)

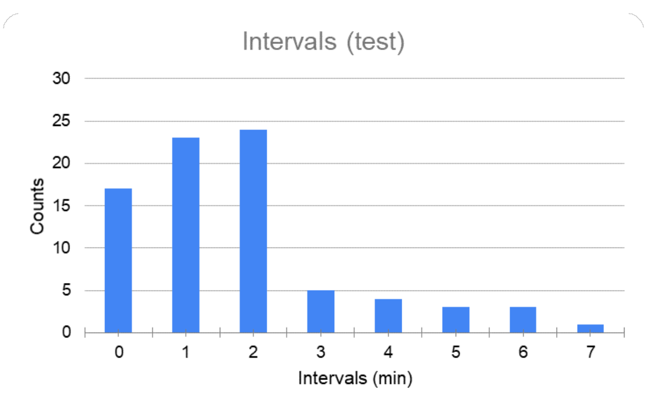

Figure 15: Interarrival Rate (Test)

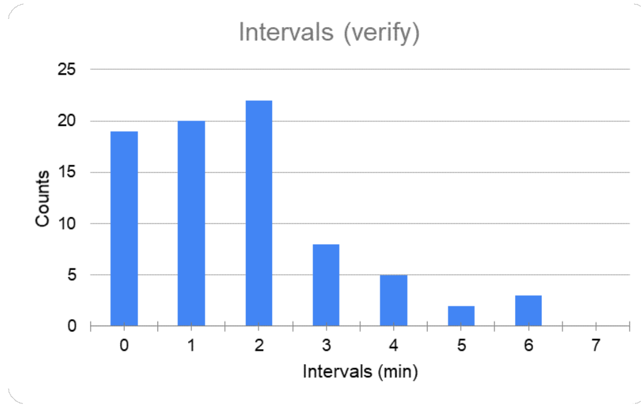

Figure 16: Interarrival Rate (Verify)

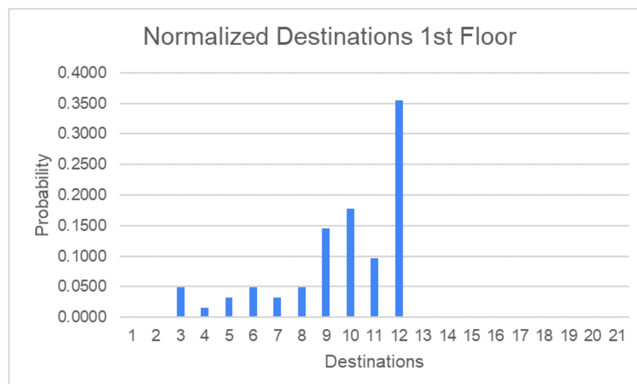

Figure 17: Normalized Destinations on 1st Floor

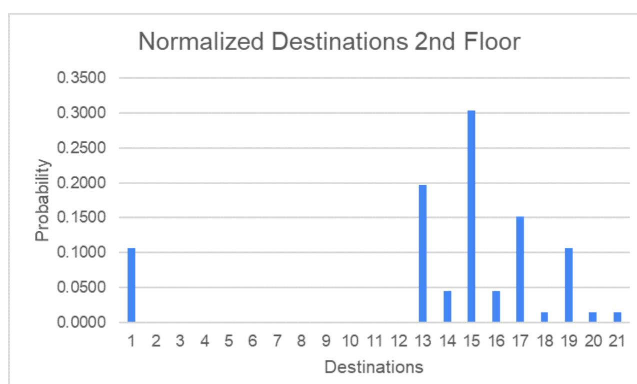

Figure 18: Normalized Destinations on 2nd Floor

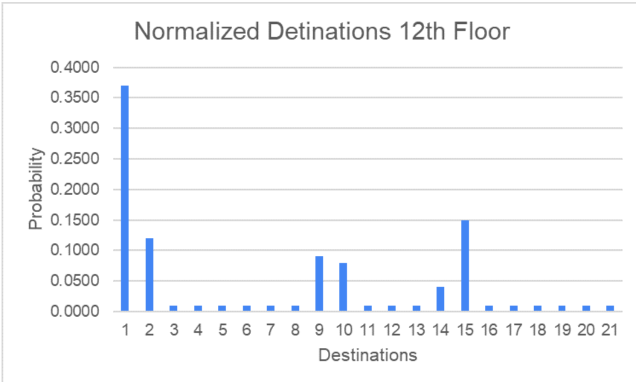

Figure 19: Normalized Destinations on 12th Floor

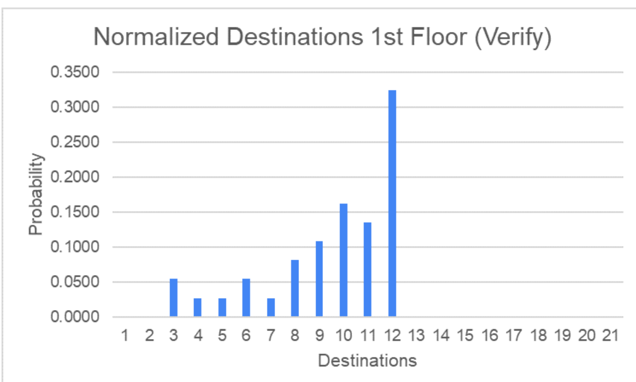

Figure 20: Verify Destinations on 1st Floor

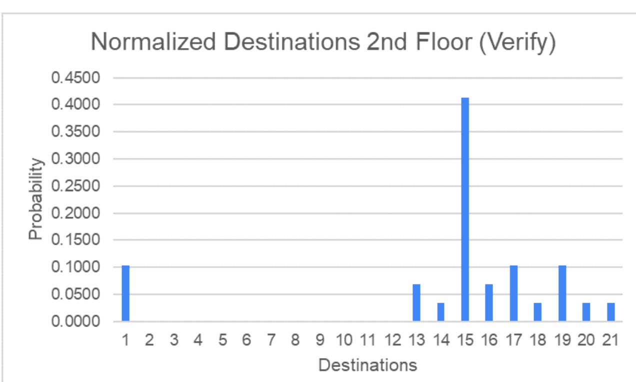

Figure 21: Verify Destinations on 2nd Floor

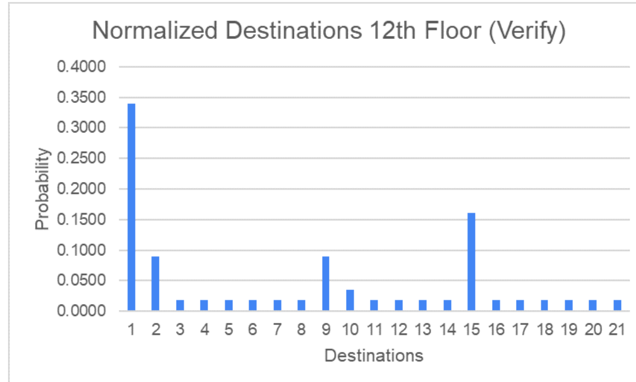

Figure 22: Verify Destinations on 12th Floor