[CS166] Elevator Simulation Report

January 21, 2020

Elevator Simulation
Report

Taha Bouhoun
Nour Elkhalawy
Asmaa Aly

1 Introduction

Object-oriented simulation is a representation of knowledge about systems that contain more than one object to model the interactions between its components and offer a framework for the methods of each object. The aim of this paper it to present a simple simulation of the functionalities of an elevator as it responds to various requests from the passengers.

2 Assumptions:

The elevator simulation results are based on the following assumptions:

- The passengers are uniformly distributed among the floors of the building at each iteration.
- The elevator maximum capacity is expressed in function of the number of passengers.
- The passengers communicate their destinations level prior to entering the elevator.
- All passengers request the elevator at the same time.
- The waiting time is measured as the steps that the elevator has to accomplish to satisfy all the requests (i.e., a counter that increments once if the elevator travels one floor, and it increments twice if the elevator drop-off people since the time to open/close and let people out is realistically longer).

3 Methodologies & Results

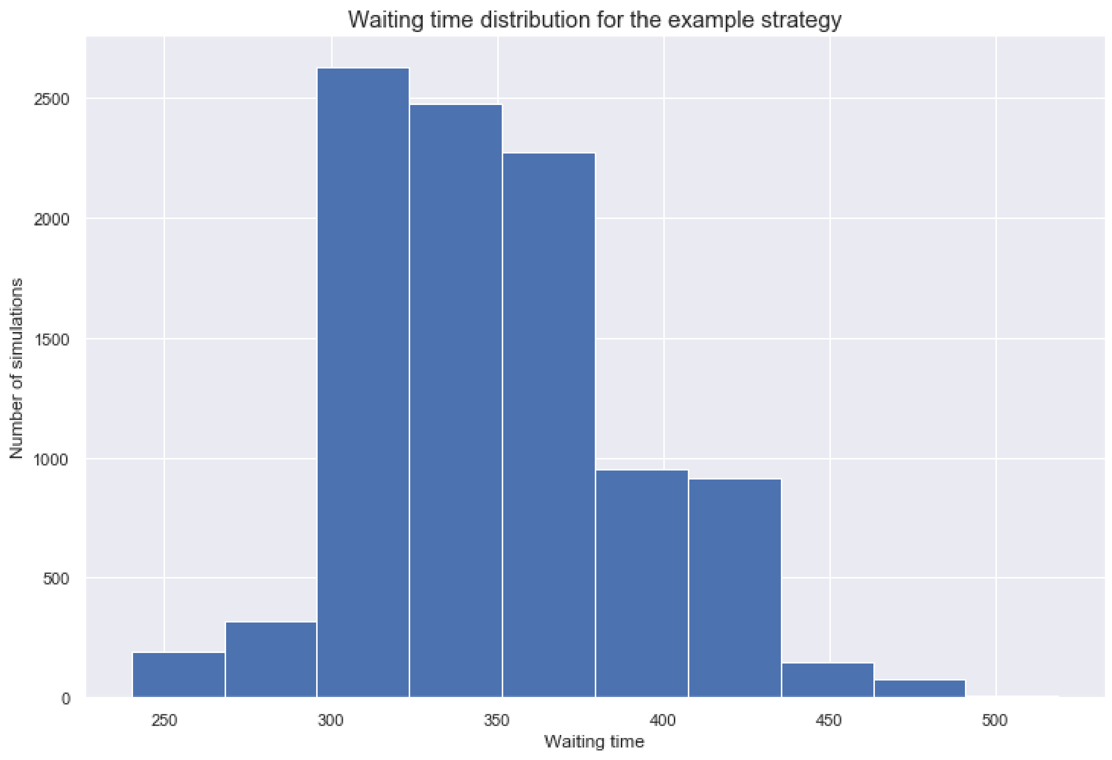
At first, the input is set to be manual and the user has to supply the simulation with all those variable (see `class Building`), the input is checked to see whether it's a non-negative integer.

The simulation then is ran by setting the input to be equal for all strategies (i.e., 10 floors, 50 passengers, and the capacity of the elevator is 5 people) using `class Building_sim` which doesn't request the input at every iteration. The strategies are subjected to 10,000 randomly-generated uniform distributions of people around the building and the results were stored in a list to produce the following histograms.

The generated requests that each method is subjected to at each iteration are different but the parameters are the same (i.e., the number of people, the number of floors, capacity of the elevator, and the nature of the initial distribution). Therefore -in the long run- both methods are subjected to simulations that are comparable since they were generated using the same process.

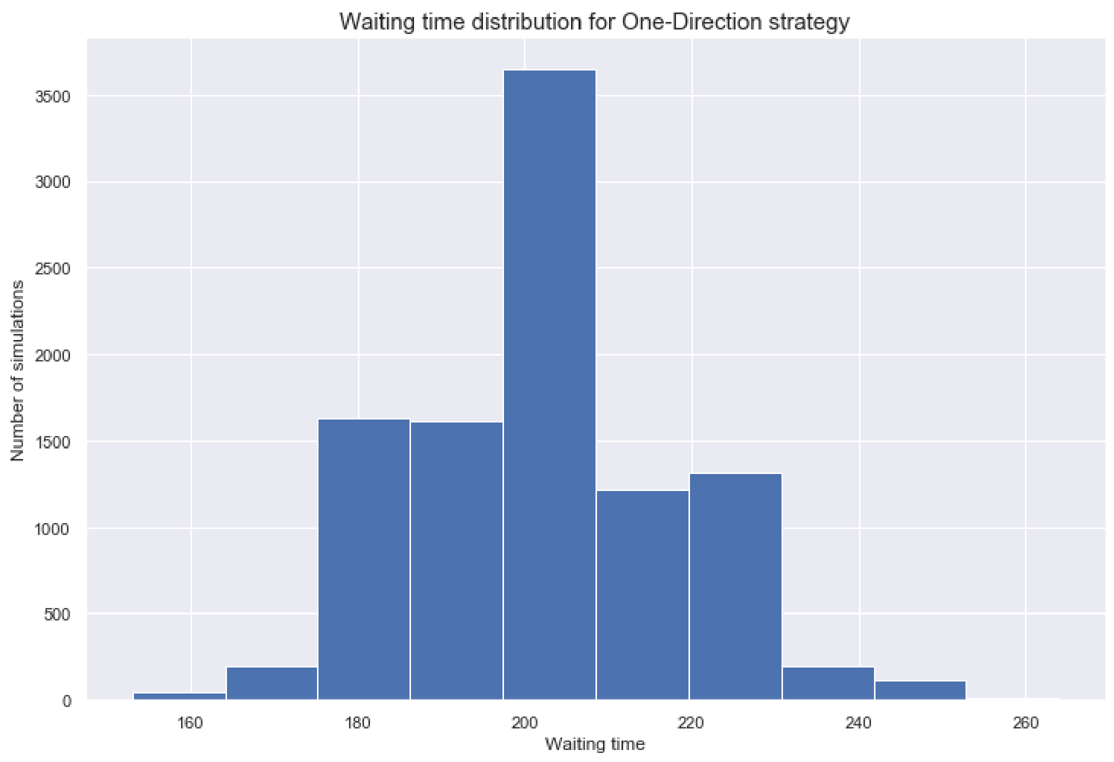
3.1 Example strategy

The example strategy (explanation from the prompt). The elevator starts on the ground floor and moves all the way to the top floor, stopping at every floor in between. When the elevator reaches the top floor, it changes direction and moves all the way back down to the ground floor, again stopping at every floor in between. At every floor where the elevator stops any passengers who want to get off leave and any passengers who want to get on enter, as long as there is space in the elevator. If the elevator is full, passengers on that floor have to wait. Upon reaching the ground floor, the elevator repeats the cycle of moving all the way up and all the way down the building.



3.2 One directional

Analogous to the example strategy, the One Directional method is picking up people heading to the direction that the elevator is pursuing. The difference is that the elevator is not going to stop at levels where there are no passengers or when none of the people in that level are heading towards the elevator's direction.

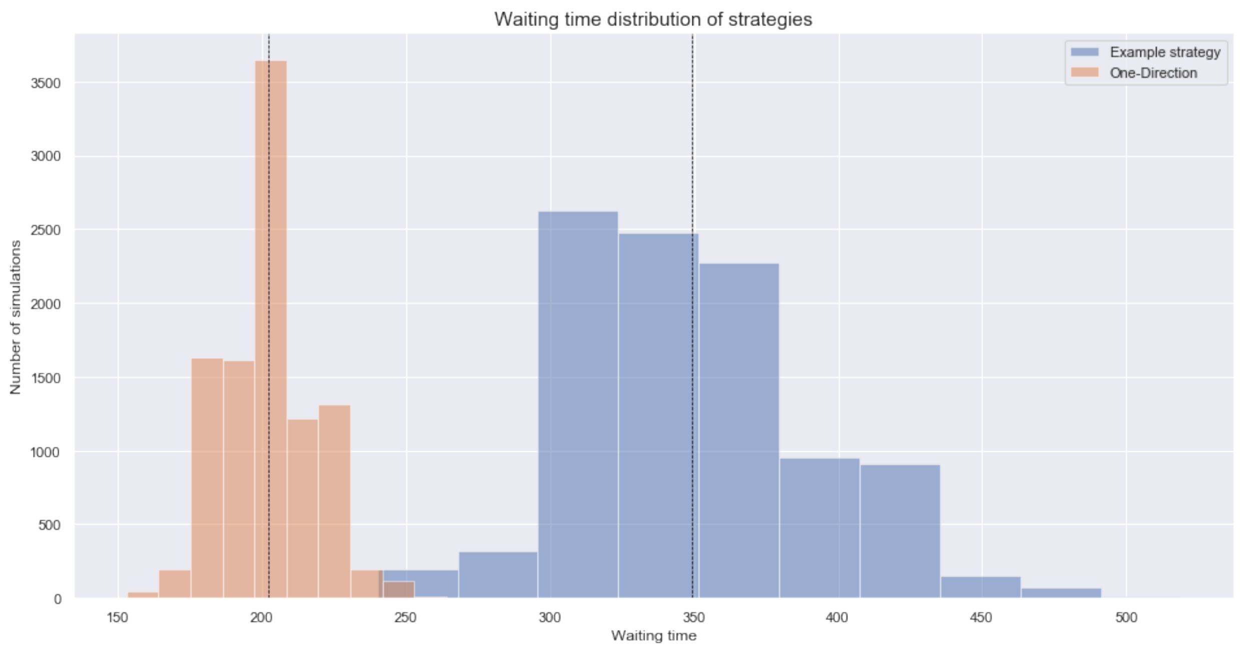


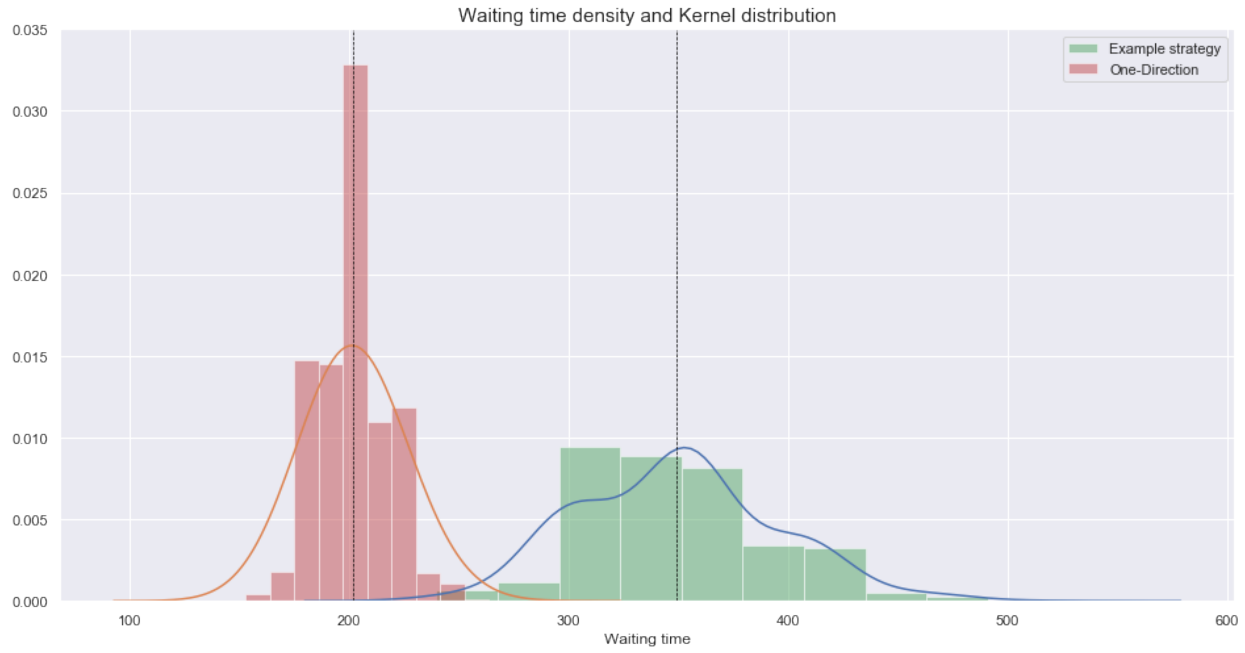
3.3 Comparing the results:

Intuitively, the One-Directional method would perform better than the example strategy because we reduced the instances where the elevator stops at each level even if there are no requests to fulfill at that level. However, it would be interesting to quantify the difference by comparing the mean of the waiting time of each method after 10,000 iterations.

- The average waiting time for the Example strategy: 349.4373
- The average waiting time for the One-Directional: 202.2632

We notice that the waiting time is reduced to about 42.11% compared to the example strategy Furthermore, both distributions are roughly Normal which aligns with the expectation given the rule of the Central Limit Theorem.


Waiting time distribution of strategies
Example strategy
One-Direction
Number of simulations
Waiting time
0
500
1000
1500
2000
2500
3000
3500
150
200
250
300
350
400
450
500


Waiting time density and Kernel distribution
Example strategy
One-Direction
Waiting time
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
100
200
300
400
500
600

4 Reflection

Although the simulation might be simple, the aim was to practice with systems that consists of moving parts and each objects have specific characteristics allowing the interactions to be modeled in a concise way. We proved through simulation that over time, the difference between strategies is prominent and we can even estimate the difference. The elevator simulation, however, would be interesting to simulate when incorporating realistic assumptions (i.e., the distribution of people, the timing of their requests)

5 Appendix

1. Contribution: Defined the classes and created the methods for each of the objects. I also helped group members to understand the rational behind separating all components of the simulation and ran the tests to compare which strategies are best.
2. Source code: https://github.com/Tahahaha7/Modeling_Simulation/blob/master/Elevator%20Simulation/%5BCS166%5D%20Elevator%20Simulation.ipynb
3. HC/LO Application:
   - **gap_analysis(HC):** The way the elevator problem is structured relies on our ability to craft methods that bridges the gap for each passenger to move from their starting point to the desired destination.
4. The Example strategy output:

```
Enter the number of floors: 5
Enter the number of passengers: 10
Enter the capacity of the elevator: 3
(0, 1)
(0, 4)
(0, 4)
(1, 0)
(1, 3)
(2, 1)
(2, 4)
(3, 2)
(3, 2)
(4, 1)
The requests of the passengers None
(0, 1) Trip started
(0, 4) Trip started
level  0 load  2 list  [1, 4]
(0, 1) Trip completed
(1, 0) Trip started
level  1 load  2 list  [4, 0]
(2, 1) Trip started
level  2 load  3 list  [4, 0, 1]
level  3 load  3 list  [4, 0, 1]
(0, 4) Trip completed
(4, 1) Trip started
level  4 load  3 list  [0, 1, 1]
level  3 load  3 list  [0, 1, 1]
level  2 load  3 list  [0, 1, 1]
(2, 1) Trip completed
(1, 3) Trip started
level  1 load  3 list  [0, 1, 3]
(1, 0) Trip completed
(0, 4) Trip started
level  0 load  3 list  [1, 3, 4]
(4, 1) Trip completed
level  1 load  2 list  [3, 4]
(2, 4) Trip started
level  2 load  3 list  [3, 4, 4]
(1, 3) Trip completed
(3, 2) Trip started
level  3 load  3 list  [4, 4, 2]
(0, 4) Trip completed
level  4 load  2 list  [4, 2]
(3, 2) Trip started
level  3 load  3 list  [4, 2, 2]
(3, 2) Trip completed
level  2 load  2 list  [4, 2]
level  1 load  2 list  [4, 2]
level  0 load  2 list  [4, 2]
level  1 load  2 list  [4, 2]
(3, 2) Trip completed
level  2 load  1 list  [4]
level  3 load  1 list  [4]
(2, 4) Trip completed

Waiting time of example strategy: 60
```

6 References

- Prof. K. Scheffler (2020). CS166 Assignment 1 Elevator simulation (prompt). Retrieved from: https://course-resources.minerva.kgi.edu/uploaded_files/mke/00095434-6871/cs166-assignment-1.pdf