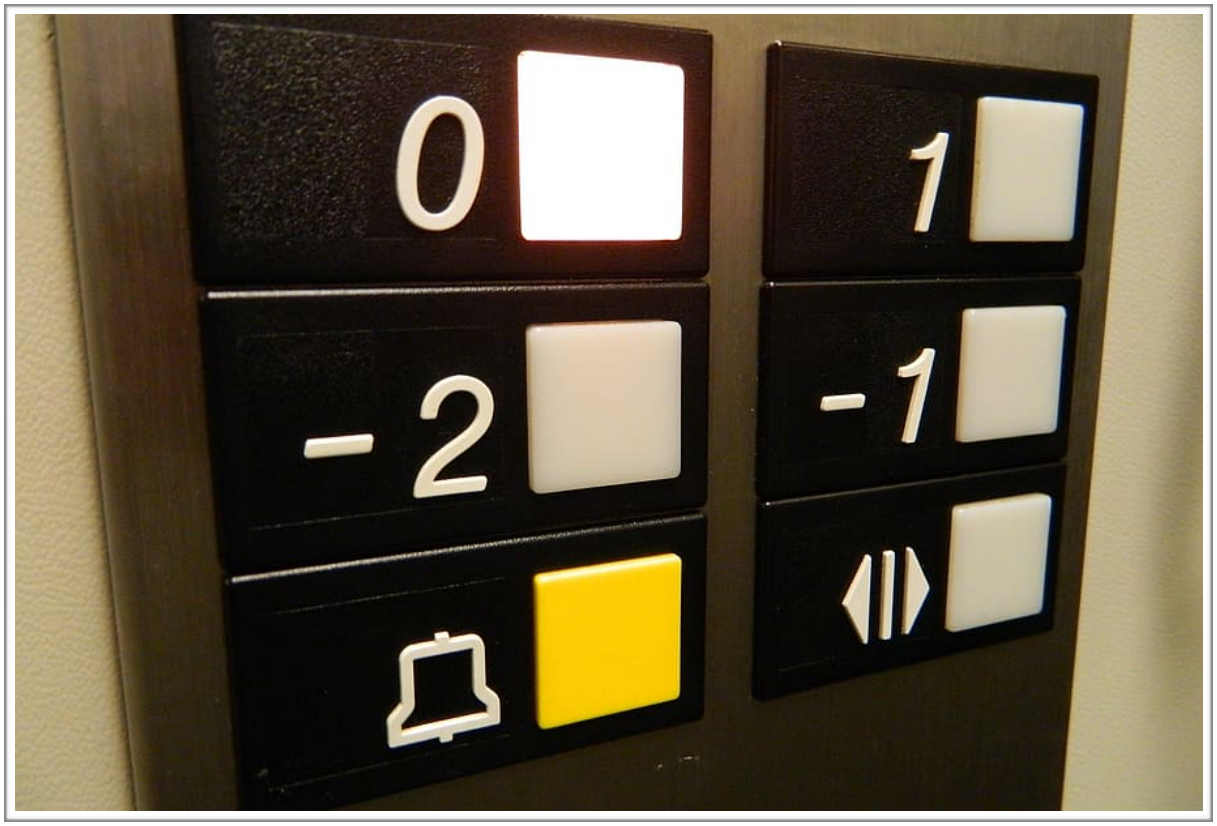
Lift Control

ECM1414 Coursework



Term 2, 2020
Submission Date: March 20, 2020

Lift Control

ECM1414 Coursework

Introduction

Almost every building in the world with 4+ floors has a lift system but few of us stop to think about the workings of this system. Lifts are programmed according to the demand of "passengers". This can be described as an optimisation system that is linked to passengers satisfaction: the shorter is their wait the happier they are with the system.

Lift control systems are becoming quite complex and are able to adapt to demands in the building. However, old systems were very restricted and could not do much adaptation to the demand.

Your course work consists of implementing a lift control system which is able to be more efficient than one of the old systems available, let us call it mechanical direction control system.

Lifts with Mechanical Direction Control

In these systems, the lift can only change direction if it reaches the top of the building or the ground floor. It has a status (up or down) and which only changes on these two floors, meaning that if its status is "up" it will have to go all the way to the top floor to change its status and start going down. Conversely, if its status

is “down”, it will have to reach the ground floor in order to be able to change direction and move up. This system is simple but has several inconveniences. For instance, assume the following scenario:

- The building has 20 total floors
- The lift is currently on the 7th floor with status as “up”.
- A person calls the lift on the 6th floor and she wants to go “down” to the ground floor (floor 1)
- As a result of the mechanical system, the lift would have to travel to the 20th floor to change the direction and then return to the 6th floor to take the passenger who wants to go to the ground floor.
- This is quite inefficient because the lift would have to travel 13 floors to the 20th, change the direction, and travel 20 floors to take the passenger to the ground floor.
- If we assume that 1 floor takes 1 unit of time (let us call it 1f), the wait time of the passenger (w_p) to enter the lift, the passenger had to wait 27f (13 to go to the 20th floor plus 14 to return to the 6th floor).
- The total cost to deliver the passenger (c_p) to the ground floor is in the above case 33f.

Now assume that there are several people in the building with various demands to go “up” or “down”. One could easily measure the performance of the elevator based on the sum of the wait times or the sum of the total cost. Given by:

$$W = \sum_{\forall p} w_p \qquad\qquad C = \sum_{\forall p} c_p$$

Your job is to implement the above elevator control and an alternative control system so that the cost is less than the one for the above (naive) system.

What Should be Submitted and How?

You are required to submit your course work by the deadline using BART and E-Submit. The following is required:

- All the source code (Python or Java) (80%)
- A PDF specification of your work which should include the following sections (20%)

- General description of your lift control system. You should present a rationale that explains why you believe your proposal is better than the mechanical lift system.
- List of all data structures and algorithms used in your implementation. This refers to data structures and algorithms that we have studied in the module. It is a requirement for you to explain how these were used.
- Weekly log of your progress. What has been implemented and when.
- Performance analysis. This is not a theoretical analysis using asymptotic notations. Instead, this refers to an analysis of different scenarios in the building. The scenarios consist of configurations of people on floors. Note that you are **not** expected to have a system in which people arrive and are served in real-time. Both implementations should work by generating people on floors and once they are in place the lift starts to serve them. In all scenarios, the lift always starts the execution on the ground floor.
- To support the performance analysis, you should include charts that support your performance claims. The charts should clearly show the performance of your lift control against the mechanical system.
- List of references used to implement your system and the mechanical control system.
- A video recording (screen capture video with a demo of your system). You may also provide a link if you decide to place it on your website. Note that this should not exceed 3 minutes.

More Information

The coursework, if done well, should be a great resource for you to present to employers and discussing what you did in the university. Design it in a way that it becomes part of your e-portfolio. This is not required but it is strongly encouraged. You should think of having a site, and code repositories available (e.g. GitHub).

Think of yourself as a start-up company selling lift control and your competitive advantage is the performance of your system (compared to the mechanical, naive, approach). Hence your demo should be convincing.

It is likely that your code will have at least two parts:

- The control algorithm itself; what makes the lift respond appropriately to calls. In fact, you are likely to have two controls because you have to also implement the naive approach.
- A simulation system that can demonstrate the lift control systems. If implemented well the same simulation can be used to demo the mechanical control system and your improvement.

Minimum Requirements for your Code

Your system should have the following (minimum) requirements:

- Implementation of mechanical control system (30%)
- Implementation of your control system (40%)
- Implementation of an interface (Graphical is preferred) (10%)
- Configurable number of floors. Note that for the sake of demoing you can limit the number of floors given that large numbers won't fit on the screen. However, you should be able to generate charts with performance of buildings with a large number of floors even if this cannot be seen as a proper demo (i.e. you should be able to run on the background and just generate the data for the performance charts) (10%)
- Generation of statistical charts. This means that you should test your advanced control against the baseline under several scenarios to be statistically sound. You code should output performance data that can be used to generate charts (e.g. in Excel for instance). (10%)

Important Questions You Should Consider

There are several questions you should consider when designing your lift control system including (but not limited to):

- What will you minimise? Wait time (recommended)? If you have another idea it is recommended that you check with the lecturer and/or the TAs.
- What is the max capacity of the lift car? How will you control this capacity? Your lift is expected to have a max capacity.
- How is the lift called? It is recommended that you use a "up/down" button simulation. This means that when you generate your scenario you should

assume that people press up or down to call the lift and that only after entering the lift they press the floor.

- Should people enter the lift if they are going to the opposite direction of the lift? It is recommended that they don't.
- How do you avoid people waiting for a long time? Do you need to institute some level of priority to avoid people waiting "forever"?
- You can implement multiple lifts as extra but having multiple lifts does not replace the implementation of 1-lift system.
- What is the input and output of your system?

You are encouraged to talk to TAs and the Lecturer about your ideas as well as the progress.