Elevator Control Strategies

Bachelors essay in computer science
Royal Institute of Technology

Johanna Axelsson
Kungshamra 3, 170 70 Solna
0736-574676
joaxel@kth.se

Sarah Bernelind
Ellen Keys gata 33, 129 52 Hägersten
0702-420636
sarahber@kth.se

Supervisor: Johan Boye

May 29, 2013

Abstract

The purpose of this essay is to investigate if it is effective to switch strategies for elevators during one day in an office building. This essay describes some of the strategies in use today, followed by a comparison and analysis of two of the strategies described. We have also implemented optimizations to one of these strategies. From our test results we can conclude that our optimized strategy worked and produced better results on average waiting time and total traveling time than the two strategies we previously implemented and that it is not necessarily effective to completely change strategies but change parts of them depending on the traffic flow.

Referat

Syftet med denna rapport är att undersöka om det är effektivt att byta strategi för hissar i en kontorsbyggnad under en dags olika trafikflöden. Först beskrivs några av de vanligaste strategierna, vilket följs av en jämförelse och analys av två av dessa. För att kunna jämföra dessa strategier har en implementation av ett program som simulerar en byggnad med hissar gjorts. I programmet används sedan dessa strategier för att ta fram testresultat. Den mest effektiva av dessa två strategier har optimerats och jämförts ytterligare med de tidigare resultaten. Testresultaten visar att den optimerade strategin var den mest effektiva med avseende på genomsnittsväntetiden och den totala genomsnittsrestiden, och att det inte är nödvändigt att helt byta strategi under dagen, men att det lönar sig att optimera delar av den, beroende på trafikflödet.

Preface

This essay and the corresponding project have been carried out as part of a (15 credits) bachelor degree at the CSC department of the Royal Institute of Technology.

The background research was done by Sarah, and the implementation by Johanna. Both authors participated in the writing of results, discussion and conclusion.

We want to thank our supervisor Johan Boye for help and guidance during this project.

Contents

Chapter 1

Introduction

Today, elevator group control is used worldwide and there are many different ways of deciding which elevator should handle a request. During a day in an office building, an elevator group control system will confront many different flows of passengers. In the morning, there is an up-peak in the traffic, during the day the traffic is mixed among the floors and in the afternoon, the flow out of the building is heavy.

This report investigates how effective different elevator group control strategies are on a system with two elevators, and which optimizations that can improve the performance of these strategies. To be able to compare the strategies, we have written a program which simulates a building with two elevators, and passengers flowing through the system. The results are then compared to see which control strategy fits the situation best. From studying this data, we developed our own optimized strategy from the most successful one to see what difference these improvements make.

Since the traffic through a building differs much during a day it may be efficient to change strategy during the day, adjusting it to the flow of passengers. Ultimately, we will discover which strategy fits what flow best.

1 Problem Statement

We will compare different elevator control strategies with respect to the average waiting time and total travelling time in an office building with two elevators. These strategies will be compared during two different stages of the day; the up-peak in the morning when passengers arrive from an lobby floor and the down-peak when passengers are leaving the building. Thus there will be two different flows and the question is whether it is more efficient to change strategies during these flows.

Chapter 2

Background

1 Elevator Control

Most of the research done today on elevator group control deals with systems where there is more than one car involved. Most engineers tend to measure the performance of the system in times such as average waiting time; the time between the passenger presses the call button until the elevator arrives, and average service time; time on average from which the passenger presses the button until it has reached its destination[1].

Another aspect of elevator movements is the energy consumption of the system. In fact, 3-5 percent of the total energy consumption in a building comes from lifts, escalators and moving walks, according to the E4 project from the European Commission[2]. Therefore, minimizing the travel distance for the elevator is important.

In the last decades new techniques have been introduced, for example, fuzzy logic, neural networks and evolutionary algorithms (genetic algorithms), which we will describe more thoroughly later in the essay. Today, hybrid techniques using the best of these methods have made the problem of engineering an effective elevator control system an area of computational intelligence[3][4], an area of which the focus is on problems normaly solved by an actual brain.

2 Development of Strategies

Many strategies for controlling elevators have been developed during the years. The first ones were very simple, straightforward strategies, but today it is common to see algorithms using artificial intelligence and machine learning to improve the elevators movements. We have researched some of the more interesting strategies and algorithms to see what kind of strategies exist.

2.1 Collective Control Strategy

Among the oldest strategies is the collective control strategy. It is a simple strategy for an elevator. The elevator runs in one direction and picks up passengers with the same direction as the elevator. When all of the passengers have been dropped off and there are no more requests in that direction, the elevator will go in the other direction if there are any requests. Else it will just stop and be idle where it dropped the last passenger[5]. One drawback of this strategy is the phenomenon called bunching, where several cars answer the same call from a floor and arrive at similar times, thus increasing both the waiting time for the other passengers in the system and the elevators' travel distance.

2.2 Zone Approach Strategy

In the zone approach strategy, which is based on the collective control strategy, each elevator is assigned a zone of the building. The car only answers calls from that area and stops in the zone when it is idle. However, it is allowed for the car to drop off passengers outside its zone. This strategy aims to keep the cars separated and to avoid several cars answering the same call. It is suited for heavy traffic when the hall calls are spread all through the building but it loses, at the same time, a lot of flexibility[6] since the cars cannot cover for each other.

To create an optimal strategy the distribution of zones has to be chosen carefully. When making the decision on how to divide the building there might be several variables to consider. Zones might be divide depending on the population of the floors or if there is a floor of importance, such as an executive floor. The general notion is to have as many zones as you have elevators[7].

2.3 Search-based Strategy

Unlike the algorithms described above, a search-based strategy is based on a search algorithm rather than a greedy variant which would chose the first solution, which is not always the best solution. It will search through the possibilities to assign, based on some criterion, a car to a call. The search space consists of all the hall calls made in the building matched with the elevators and the criterion on which you search could be minimizing the

average traveling time or the average waiting time. These types of algorithms are used when you want to find a minimum value of a mathematical equation or find an item with specific traits among a specific collection of items.

By using a greedy algorithm you can shorten the time for assigning the call to a car since the greedy algorithm immediately assigns a car based on the currently available data. This is good for minimizing the average waiting time but it is not flexible because the greedy algorithm never reassesses its choice of call assignment, it base its decision localy, meaning it will make the best decision from the point where it stands at that exact moment. The opposite of a greedy algorithm is a non-greedy algorithm which is flexible and can reassess its call assignments in the light of new continuous information from the elevator system. The non-greedy algorithm will take more time to decide which call it will assign and thus the average waiting time might be longer but the overall results might be better[6].

2.4 Rule-based Strategy

The rule-based strategy is based on "IF condition -THEN action" logic. The strategy is commonly used in Artificial Intelligence but can in some sense be applied to all control strategies. A rule base is created out of expertise and research, and from there the elevator control makes its decisions. When this type of strategy is used you can combine it with fuzzy logic. Fuzzy logic is a form of many-valued logic. In fuzzy logic the concept is to use variables that can be more than just boolean values, for example, a temperature variable could be "warm", "cold" or "slightly above freezing". All of these different values on the temperature variable have different truth values.[8]. Utilizing this opens a whole new dimension of decisions needing to be made by the system. For example, one rule could be IF(there is a call from a lower floor) AND (all the cars are descending) THEN (assign the descending car that will have the best results based on the average waiting time in the system)[9]. When you look at our example the "call" variable is set to "lower floor" and the "cars" variable is set to descending. It is by evaluating these two variables that you get your THEN action. You would get a different THEN action if the two variables where different.

2.5 Genetic Algorithms

Some problems can also be solved using a genetic algorithm, which imitates the process of evolution. [10]. Genes consists of chromosomes and we inherit these genes from our parents. In order to artificially mimic the process of natural selection the chromosomes to start with has to be created. When programming genetic algorithms these chromosomes are designed and, put together rendomly as a gene or individual, poses a solution to the problem at hand. The genes are being tested to decide their fitness score, which is a number indicating how successful the gene is[11]. When selecting genes for "mating", the fitness score is used to decide which genes should be selected. Because the selection of the chromosomes is weighted, the probability is high that two genes with a high fitness score are selected. Thus the population will evolve to the better in each iteration, just as it would have for living creatures.

When applied to the problem of engineering an effective elevator control system the chromosomes put together as a gene would describe how the system would act in different situations and the chromosomes with the result nearest the demand of the engineers would be awarded the highest fitness score. By "mating" genes in each iteration the results will only get better and better after the bad ones are being sieved out. This trait leads to the advantage that you, at any moment, can stop it and still have a better solution than the one you started with. The disadvantage with a genetic algorithm is that it is iterative and therefore slow in execution.

To summarize, there are a lot of interesting strategies you can use when you want to create an elevator control system and many of them are being used today. Which one is the best depends on what conditions you have but genetic algorithms have become popular in the last years[11] and a lot of research is being done on the subject which is being used in all kinds of different areas.

3 Passenger Arrival

During a day, the rate in which passengers arrive to a building varies and it is the passenger arrival that impels an elevator system. In this report, the focus is set on an office building and its typical flow.

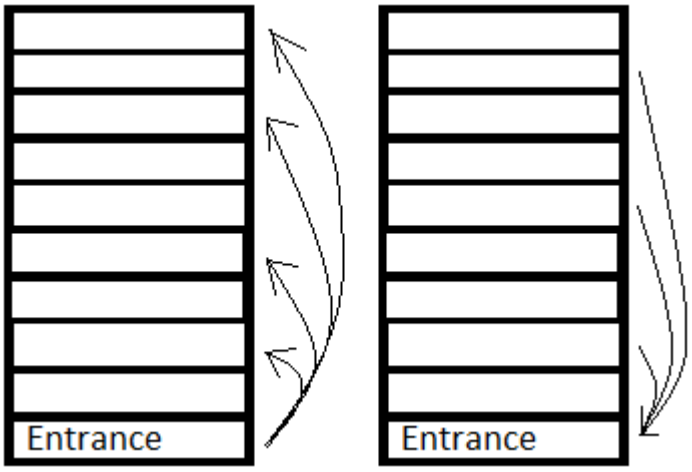

Figure 2.1: *Traffic flows through a building in up-peak and down-peak*

In up-peak traffic there will be one arrival floor and many destination floors while in down-peak there will be many arrival floors and one destination floor, as shown in Figure 2.1. In heavy up-peak traffic, the cars will fill up and stop at many floors thus taking a longer time to make the trip and the average waiting time will be longer for the passengers waiting in the lobby. In heavy down-peak traffic the cars might fill up on the upper floors thus making the total round-trip time for the car less than for the same car in heavy up-peak traffic.

The situation will be reversed when there is light ascending traffic. By then cars can be idle in the lobby which leads to a minimized waiting time for arriving passengers. If the descending traffic is light you cannot have an idle car at every floor. Therefore minimizing the waiting time in light ascending traffic is easier than minimizing it in light descending traffic.

According to Crites and Barto[6] it is the capacity for ascending traffic that sets the capacity for the whole system. So when engineers try to build a system of elevators, they should try to predict the heaviest up-peak in the building and accommodate that demand. If the system is adjusted to the heaviest up-peak, it will often be sufficient for down-peak as well.

Chapter 3

Method

The chosen method for this project is implementation of an elevator control system since it gives a good overview of the system and it makes a reliable source for results.

1 Elevator Simulator

The programming language chosen for the simulator was Java, because Java is the language in which the authors are most proficient in and it suits this kind of implementation well. Our simulation features two elevators in an office building with ten floors.

The state variables of the system are presented in Figure 3.1. Object refers to both elevators and passengers.

Each elevator car in the simulated building is designed to have their own queue for requests, which consists of both request from the inside (made by pressing buttons on the elevators control panel) and from the outside (up/down button on each floor). The elevator is controlled by this queue and always runs in the direction of the first destination put in the queue.

The elevator itself only adds requests to the queue made from the inside. The other requests are handled by the building which has an overview of what states the elevators are in.

State variables	Description
Direction	The direction in which the object travels
Position	The floor the object is currently positioned at
Destination	The destination of the object
Full	Indicates if the elevator is full or not
Closing doors	Indicates if the elevator is closing the doors
Queue	The destinations to which the elevator will travel
Button up/down	Pressed when a passenger is spawned at a floor
Waiting time	The time which an elevator have to wait before moving

Figure 3.1: *The state variables of the elevator system*

The queuing in the strategy is made based on a combination of states of the elevators. It will look for an elevator that firstly goes in the same direction as the passengers request and secondly still not yet have passed the passengers position. The strategy also checks whether the elevator is full or not. If none of the elevators fulfill these requirements the strategy will, if possible, assign an idle elevator. If there are several elevators idling, the closest one will be chosen.

If an idle elevator is not found, the passenger will be put in a waiting list. It will later on be assigned to an elevators queue when an elevator has been found.

The elevator itself also checks on certain states when stopping at a floor. First of all, it decides if it should pick up or leave passengers. If the elevator is empty, there is no one to leave. If there are passengers at the current floor, there may be someone to pick up. When deciding which passenger to pick up, the elevator checks that the passenger has got the same direction as the elevator. If the elevator is full when it stops to pick up a passenger, that passenger will be returned to the waiting list in the strategy.

The floors in the building have two main states, the up and down buttons. These are used to check whether a request is already sent from the floor if a new passenger arrives. The passenger has got a few states, of which most are used by the strategy and elevator to be able to place the passengers request in the queue.

The passengers are spawned and immediately given an arrival and destination floor, which depends on what time of the day is simulated at the moment. To get a reasonable rate of arrival, we used an exponential distribution to generate the waiting times between generating new passengers. An exponential distribution describes the time between events in a Poisson process. A Poisson process is a process in which events, at a constant average rate, occur independently and continuously[12].

One time step in our system includes spawning new passengers (if the generated time is the same as the current time), then assign hall calls from the waiting passengers and after that moving the elevators. An elevator can move one floor per time unit. If the elevator is picking up or leaving passengers, it will stand still for a total of six time units to simulate that process.

To get a better understanding of our system, consider an example of our elevator group control process at the moment a passenger arrives. The passenger is positioned at the fifth floor with direction down and destination

lobby floor. The first elevator is at the second floor with direction up and one request for floor nine in its queue. The second elevator is positioned at floor seven with direction down and one passenger inside. It is destined for the request in its queue, which is the lobby floor; the destination of the passenger who is already inside the elevator. None of the elevators are full and they both have closed their doors. There is no one else on the floor the passenger arrived and none of the up- and down-buttons is pressed.

1. The passenger presses the hall call button at the fifth floor. The down-button is pressed.

2. The building adds the request at floor five to the list of places waiting to be served.

3. The strategy finds that the second elevator has the same direction (down) as the passenger, its position is above the passengers current position, the elevator is not full and it is not closing its doors at the moment.

4. The strategy adds the new request to the second elevator. The queue of the second elevator now contains floor five and the lobby floor. The elevator begins to travel down.

5. The elevator travels down and eventually arrives at floor five. This takes two time units.

6. The elevators position is now floor five. The passenger enters the elevator and the hall call button (down) is reset. The fifth floor is now empty and the elevator contains one more passenger. The waiting time of the elevator is increased, it has to stop and leave or pick up passengers. Floor five is removed from the queue.

7. Since another passenger in the elevator already has requested to go to the lobby floor, the elevators queue does not need to be changed.

8. The elevator waits for four time units and then begins to close its doors.

9. The elevator waits for two more time units and the doors are now closed. This means that no more passengers on this floor can enter the elevator.

10. The elevator finally begins to travel down to the lobby. This trip takes five time units since it is five floors of travel.

11. The elevator arrives at the lobby, the passenger exits the elevator whose queue is emptied of request. The elevators destination is now set to idle.

2 Development of the First Strategies

As mentioned earlier, work began by implementing the basic collective control strategy with some additions. When this strategy was complete, the zone approach strategy was implemented. It is basically the same; however, in the zone approach strategy the floors are split into separate zones where only one elevator would serve its own zone.

Both of the strategies implemented are greedy and they do not do any further predictions of what will happen after the requests in the current direction. Imagine a situation where one elevator has got a request to drop a passenger at floor nine, and is currently at eight, and the other elevator is idle in the entrance. When using the basic strategy the idling elevator will be chosen to handle a new request at floor seven even if the other elevator is closer and will soon be empty. If the zone strategy is used instead, the new passenger will be put on the waiting list and will later on be queued to the elevator that is closer.

3 Simulation and Statistics

The system uses a discrete time scheme where one unit corresponds to three seconds. The exponential distribution is used to simulate the arrival of 150 passengers during one hour.

The simulation was set to run for a time corresponding to two hours. The building consists of 10 floors. Each passenger records their time when entering an elevator and when leaving an elevator to get the waiting time and the total time in system. When a round of the simulation is complete, the calculated values are the average wait time, average total time and travel times for the elevators compared to their passengers distances. The average amount of passengers in the whole system is also calculated.

4 Optimized strategies

After running the simulation with the two strategies we compared them and then added small optimizations to the most successful one.

We also choose to run the simulation a second round under heavier load when we had optimized the strategy. We set the arrival rate to the double, 300 passengers during one hour, and ran the program for one hour instead, to see what difference it made for the strategies.

4.1 First Version

The first small improvement we added to the basic strategy was to have a default floor where the elevator is positioned while idle. This standard floor is preferably the floor with the most incoming requests or in the case of the collective control strategy during up-peak traffic, the lobby floor. In down-peak you could use the idea of the zone approach and have the two elevators being idle at two different floors, one in the higher floors and one in the lower floors so that the elevator will travel shorter distances to serve calls from the whole building. We set the default floor of one elevator to floor two and the other one to floor seven, unless it is during up-peak, then both of them are set to the lobby floor.

4.2 Second Version

In the second optimization we improved the default floor feature. In the first version, the movement to the default floor could not be interrupted which is very ineffective when a new passenger arrives the second after the elevator began to move to its default floor. Interruption of this movement was made possible.

The cars were also given the possibility to change their queues during special conditions. If an elevator has a request at a certain floor, and a new passenger arrives one floor above that request, the elevator will change its way to pick up the highest located passenger first if these passengers are going in the same direction. This works the other way as well. In order to implement this feature to be effective, a lot of states need to be checked to avoid queuing the newly arrived passenger in the wrong situation.

Chapter 4

Results

Figure 4.1 shows our test results from simulations on the basic and the zone strategy during regular pressure on the system. The time is presented in our discrete time unit, which is three seconds. The average waiting time is the time between the passengers arrival to the building until the boarding of an elevator. The total time is the passengers whole time in the system. We also measured how the passengers were distributed between the two elevators.

1 Initial Strategies

As can be seen in Figure 4.1 the zone strategy does not work at all in up-peak traffic since all the passengers arrive in one zone which leaves one elevator redundant. Furthermore, the results show that the basic strategy is more efficient in both up-peak and down-peak, with respect to waiting time as well as total time.

2 Optimization

Since the zone approach strategy gave unsatisfactory results even in down-peak we choose to optimize the collective control strategy, with some ideas from the zone approach. The following comparisons are made only between the basic strategy and the optimized versions.

The first optimization introduced a default floor for the elevators to move to while idle, since the destination of the passengers never are the same as the arrival floor in both up- and down-peak. By implementing the default floor we saw a 17.6 percent decrease in average waiting time and a 9.1 percent decrease in average total traveling time during up-peak traffic. During down-peak the waiting time was decreased with 10.9 percent and the average total traveling time by 3.8 percent. To see the discrete values see Figure 4.2.

The system became even more flexible in the second iteration of the optimized strategy. The average waiting time for up-peak traffic decreased by 2.2 while the total time did not experience any significant decrease. During down-peak the waiting time decreased 27.0 percent while the total time

Values	Basic	Zone	Basic	Zone
Peak	up	up	down	down
Average waiting time	5,54	18,92	8,51	10,44
Average total travel time	21,08	43,77	23,87	24,44
Elev0 total floors traveled	750,91	766,41	744,29	599,88
Elev0 total floors passengers traveled	697,35	1398,18	667,33	469,09
Elev0 number of passengers picked up	140,05	278,53	136,41	156,84
Ratio elev/passenger per passenger	0,77%	0,20%	0,82%	0,82%
Elev1 total floors traveled	770,04	0	870,97	1058,15
Elev1 total floors passengers traveled	715,68	0	732,12	941,37
Elev1 number of passengers picked up	142,58	0	144,21	125,42
Ratio elev/passenger per passenger	0,75%		0,82%	0,90%
Total travel for elev0 and elev1	1520,95	766,41	1615,26	1658,03
Total travel for all passengers	1401,17	1380,2	1390,1	1400,04
Ration elev/passenger traveled	8,55%	-44,47%	16,20%	18,43%
Average passengers in system	282,63	278,53	280,62	282,26

Figure 4.1: *Results from our Basic and Zone strategy*

	Basic up-peak	Basic down-peak	1:st opt up-peak	1:st opt down-peak
Average waiting time	5,54	8,51	4,18	7,58
Average total travel time	21,08	23,87	18,63	22,93

Figure 4.2: *Results from test with the Basic strategy and the first optimization*

	1:st opt up-peak	1:st opt down-peak	2:nd opt up-peak	2:nd opt down-peak
Average waiting time	4,18	7,58	4,09	5,54
Average total traveling time	18,63	22,93	18,6	19,29

Figure 4.3: *Results from test with the first and the second optimization*

Strategy	Flow	Average waiting time	Average total traveling time
Basic	Up	10,33	31,72
Zone	Up	154,42	181,17
2nd opt	Up	9,4	30,3
Basic	Down	13,15	33,86
Zone	Down	14,92	32,31
2nd opt	Down	10,73	26,84

Figure 4.4: *Average waiting time and total time during high pressure*

decreased by 15.8 percent. These results indicate that the optimization was successful and that by making the system as dynamic and flexible as possible we gained a lot of performance. To see the discrete values for the average waiting and total time, see Figure 4.3.

3 Higher Pressure

While running the simulation under a higher load (300 passengers during one hour) the results were very different. The optimized version was slightly better during up-peak and even better results for the waiting time and total time during down-peak. Worth noticing is that the zone strategy got better results than the basic strategy in one category; total time in down-peak. Results are shown in Figure 4.4.

4 Energy Efficiency

We also chose to look at the ratio between the total distance the passengers travelled and the distance the elevators travelled. This ratio tells us about how energy efficient the strategy is. Figure 4.5 shows how much more the elevators travel compared to the passengers. During down-peak the results of the basic and the zone strategy were almost the same, but unfortunately our optimizations showed disappointing results. The ratio was almost three times worse, in both up- and down-peak.

During normal pressure, the zone strategy in up-peak is the only one showing good results, however you cannot take that result in consideration

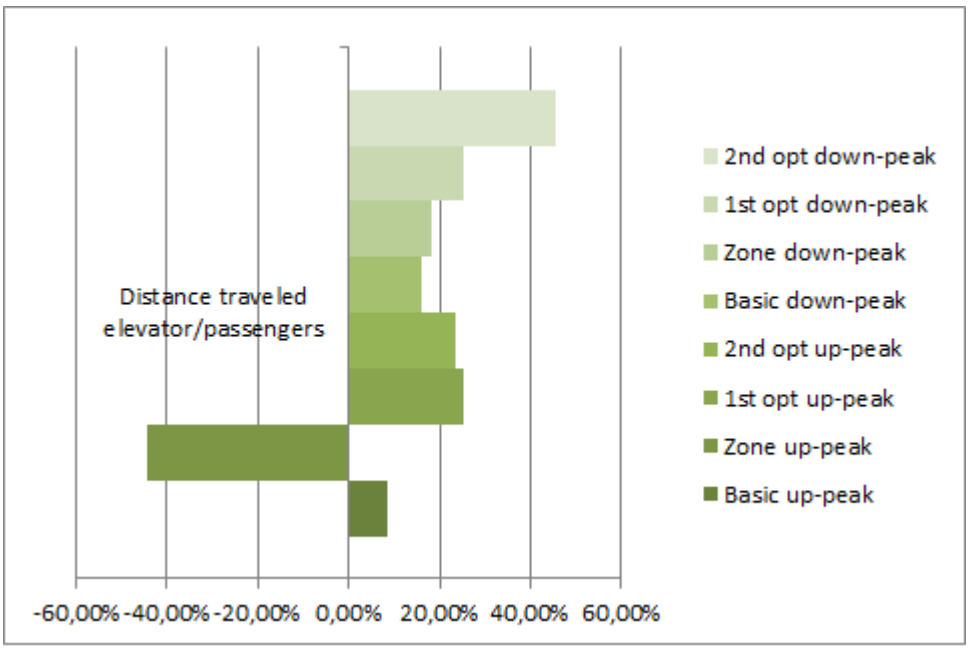

Figure 4.5: *Energy efficiency during normal pressure*

since the zone strategy works like a single elevator system and the average waiting and travel times are a lot worse than the others.

5 Source of Error

One source of error might be the fact that each test was not conducted on the exact same amount of passengers since they were generated at random times. We have tried to minimize this risk by testing all strategies 100 times and then take the average results from these tests. Figure 4.6 shows the difference in the amount of passengers who used the system during our test times.

To simplify the logic and ease the testing of the system discrete units were used instead of continuous time. This constitutes a source of error since the value of our unit is approximated and not statistically determined.

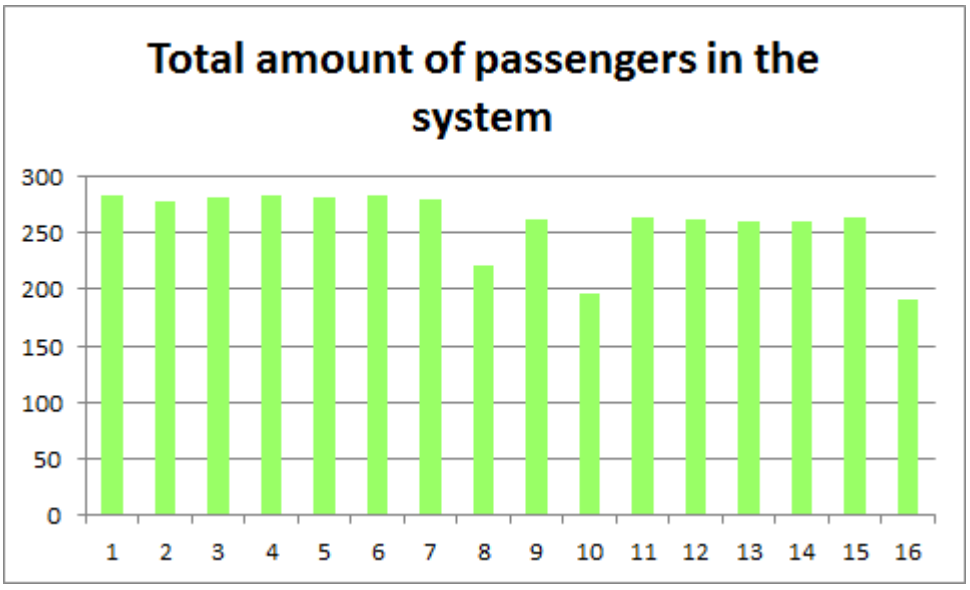

Figure 4.6: *Average total amount of passengers in the system during different strategies*

Chapter 5

Discussion

We expected to see a gain in efficiency using the zone strategy during down-peak traffic, but the basic strategy was the best one in all cases. The only time when the zone strategy actually had a better total time was under high pressure in down-peak. However, the waiting time was still worse. This seems reasonable since the travel time for the elevator in the upper zone will always be longer, but under higher pressure it can at least pick up many passengers at the same time.

The results were clear when it came to the basic strategy versus the optimized strategy under regular pressure. The first optimizations increased the performance a lot, especially during up-peak, the idle time was turned into benefitting movements towards the next arrival.

The second round of optimization also gave interesting results. While the first round of optimizations showed that our results for the up-peak were slightly increased, this run showed no big change between the basic strategy and the optimized one in up-peak. However, the times in down-peak was once again decreased while testing the system with regular pressure.

When running the system under mheavier pressure, the optimizations performance in the up-peak was just slightly better than the basic strategy. Under higher pressure the default floor feature is not that useful and if the elevator keep queuing requests one step over the current queue, the time will be increased for the passengers that requested that elevator from the beginning.

Moreover, it is known that people do not like to wait whether it is for the elevator or the cashier. Minimizing the average waiting times is crucial for passenger satisfaction and has more impact on the experience of using the elevator than minimizing riding time[13]. This is because people make the distinction between standing still on a floor and standing still in an elevator. This is because a moving elevator feels like progression even though you are not physically moving. Our optimizations reduced the average waiting time by 4 percent in up-peak traffic and 7 percent in down-peak traffic. Thus if this had been a real elevator system we would have had more satisfied passengers after implementing the optimizations.

The energy efficiency results were not that surprising. Since both of the

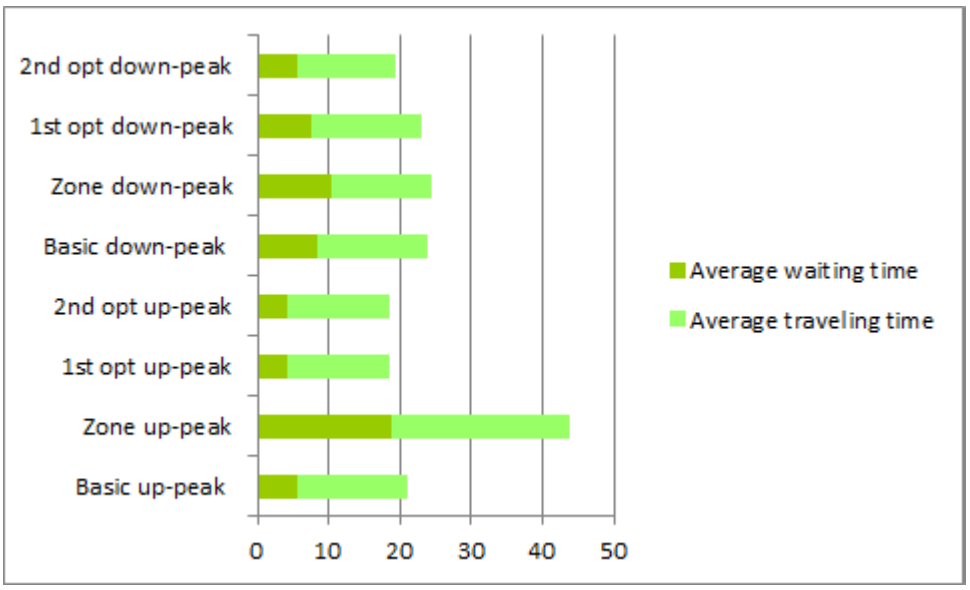

Figure 5.1: *Average total traveling time divided into wait and travel*

first strategies are pretty straightforward with the way they travel, we did not expect anything special from them in terms of efficiency. However, the fact that the optimized strategy increased the distance to such great numbers was disappointing since it showed no other bad qualities. As expected, this ratio was better during high pressure, since the elevators were able to pick up more passengers along the way and use their full capacity.

Chapter 6

Conclusion

In summary, we have investigated two different elevator control strategies for a simulated building with two elevators. The results show that there is no need to change strategy between the up-peak and the down-peak traffic, since one of the strategies was significantly better. However, a few small optimizations could increase the performance of the system drastically. These optimizations worsened the energy efficiency of the system, but instead both the waiting time and the total time for the passengers were lowered, mainly during down-peak.

Bibliography

[1] T. Takahashi and S. Matsuda, (2010), *Adaptive elevator dispatching with co-neuroevolution*, Evolutionary Computation (CEC), 2010 IEEE Congress on Print

[2] E4 project: European Commission's Intelligent Energy Europe Programme, (2010), *Energy Efficient Elevators and Escalators*, http://www.e4project.eu/documenti/wp6/E4-WP6-Brochure.pdf, March 2010, 090413

[3] Włodzisław Duch, (2007), *What is Computational Intelligence and what could it become?*, http://cogprints.org/5358/1/06-CIdef.pdf, Jan 2007, 100413

[4] T. Beielstein, C. Ewald, S. Markon, (2003), *Genetic and Evolutionary Computation — GECCO 2003*, Springer Berlin Heidelberg, vol. 2724

[5] Marja-Liisa Siikonen, (1993), *Elevator traffic simulation*, SIMULATION, vol.61

[6] Robert H. Crites, Andrew G. Barto, (1998), *Elevator group control multiple using reinforcement learning agents*, Machine Learning, vol. 33

[7] George R. Starkosch, Robert S. Caporale, (2010), *The vertical transportation handbook*, Wiley, fourth edition

[8] Chuen Chien Lee, (1990), *Fuzzy logic in control systems: Fuzzy logic controller- part 1*, IEEE Transcations on systems. man. and cybernetics, vol. 20

[9] Gu deying, Yan dongmei , (2010), *Study on fuzzy algorithm of Elevator Group Control System*, Challenges in Environmental Science and Computer Engineering (CESCE), 2010 International Conference on Print, vol. 1

[10] Mat Buckland,(2004), *AI-junkie*, http://www.ai-junkie.com/ga/intro/gat1.html, July 2004, 100413

[11] M. Srinivas, Lalit M. Patnaik, (1994), *Genetic Algorithms: A Survey*, Computer, vol. 27

[12] Gunnar Blom, (2004), *Sannolikhetsteori och statistikteori med tillämpningar*, Studentlitteratur AB, fifth edition

[13] Jafferi Jamaludin, Nasrudin Abd. Rahim, Wooi Ping Hew, (2010), *An Elevator Group Control System With a Self-Tuning Fuzzy Logic Group Controller*, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, vol. 57