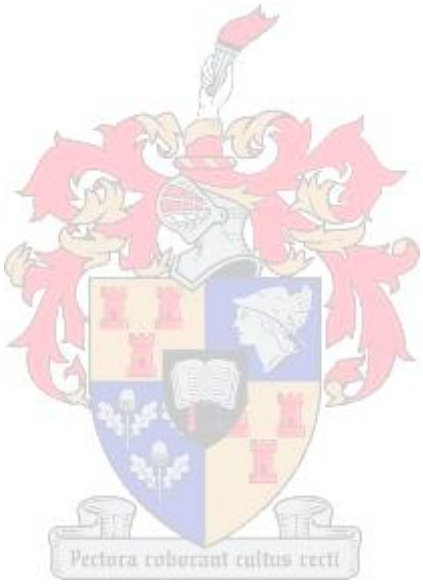
The relative effectiveness of destination dispatch in elevator control

Marno du Plessis



Year project presented in partial fulfilment of the requirements for the degree of
Bachelor of (Industrial) Engineering
in the Faculty of Engineering at Stellenbosch University

Supervisor: Prof JH van Vuuren

December 2017

Declaration

By submitting this project electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: December 1, 2017

Abstract

The operation of elevator systems in high-rise buildings is largely governed by elevator control algorithms whose chief objective is to maximise the effectiveness of an elevator group. Elevator system effectiveness is important, because elevator passengers typically desire to spend minimal time waiting for, and travelling in, elevators. The sophistication of elevator control algorithms has improved significantly in the modern era, and the notion of elevator control based on *destination dispatch* is now conjectured to be considerably more effective than conventional elevator control systems. The objective of this study is to present evidence in support of, or in opposition to, this conjecture.

A brief overview is provided of the literature pertaining to elevator control systems in general, with a focus on the operation of certain conventional elevator control algorithms, as well as that of a destination dispatch control algorithm. Apart from the latter control algorithm, two popular conventional elevator control algorithms are selected for careful consideration in this project. These algorithms are the *nearest-car control algorithm* and the *fixed-sectoring common sector system*. A significant contribution is made to the literature in respect of the aforementioned three elevator control algorithms: The logic of these algorithms are documented in the form of precise pseudo-code descriptions for the first time.

A novel computer simulation model is further established to serve the purpose of a test bed for elevator control algorithm effectiveness assessment. This simulation model is employed to pronounce on the relative effectiveness of the three elevator control algorithms mentioned above. The model accommodates a variety of varying factors, including, but not limited to, the number of floors in a high-rise building, the number of elevators in an elevator group and statistical passenger arrival distributions. The effectiveness of each of the three aforementioned elevator control algorithms is measured from the viewpoint of elevator passengers in the form of journey time. Performing this comparative analysis of elevator control effectiveness in the guise of a sensitivity analysis not only reveals the relative effectiveness of the elevator control algorithms considered, but also reflects the degree of this effectiveness in respect of reduced journey time. This effectiveness comparison analysis reveals that claims in the literature about the relative superiority of destination dispatch elevator control over conventional elevator regimes is substantiated at a 95% level of confidence.

Uittreksel

Die werkverrigting van hysbakstelsels in hoë geboue word grotendeels bepaal deur hysbakbeheeralgoritmes wat daarop gemik is om die doeltreffendheid van 'n hysbakgroep te makismeer. Die doeltreffendheid van 'n hysbakstelsel is belangrik omdat hysbakpassassiers tipies 'n hoë premie op minimale wagtye vir, en reistye in, hysbakke plaas. Die sofistikasie van hysbakbeheeralgoritmes het in die moderne era merkbaar toegeneem, en daar word tans vermoed dat die konsep van hysbakbeheer gebaseer op *bestemmingsversending* beduidend meer doeltreffend is as konvensionele hysbakbeheerstelsels. Die doel van hierdie studie is om getuienis ter stawing van, of in teenstelling tot, hierdie vermoede te lewer.

'n Bondige oorsig word gegee oor die literatuur verwant aan hysbakbeheerstelsels, met 'n dieper fokus op die werkverrigting van sommige konvensionele hysbakbeheeralgoritmes, sowel as op 'n bestemmingsversending-hysbakbeheeralgoritme. Behalwe vir laasgenoemde beheeralgoritme, is twee populêre hysbakbeheeralgoritmes vir sorgvuldige oorweging in hierdie projek gekies. Hierdie algoritmes is die *naaste-hysbak beheeralgoritme* en die *vaste-sektor algemene sektorstelselbeheeralgoritme*. 'n Beduidende bydrae word tot die literatuur oor die drie bovermelde hysbakbeheeralgoritmes gemaak: Die logika van hierdie algoritmes word vir die eerste keer in die vorm van presiese pseudo-kode gedokumenteer.

'n Nuwe rekenaarsimulasiemodel word verder daargestel om as toetsbed te dien vir die assessering van hysbakbeheeralgoritme-doeltreffendheid. Hierdie simulasiemodel word gebruik om 'n uitspraak te lewer oor die relatiewe doeltreffendheid van die bogenoemde drie hysbakbeheeralgoritmes. Die model akkommodeer 'n verskeidenheid varieerbare faktore, insluitend, maar nie beperk nie tot, die getal vloere in 'n hoë gebou, die getal hysbakke in 'n hysbakgroep en statistiese passassiersaankomsverdelings. Die relatiewe doeltreffendheid van elk van die bogenoemde hysbakbeheeralgoritmes word vanuit die oogpunt van hysbakpassassiers in die vorm van reistyd gemeet. Deur hierdie vergelykingsanalise van doeltreffende hysbakbeheer in die vorm van 'n sensitiwiteitsanalise uit te voer, lei nie net na 'n bewuswording van die relatiewe doeltreffendheid van hysbakbeheeralgoritmes nie, maar ook die mate waartoe hierdie doeltreffendheid bydra tot verminderde reistyd. Hierdie vergelykingsanalise onthul dat bewerings in die literatuur oor die relatiewe superioriteit van bestemmingsversending oor konvensionele hysbakbeheer teen 'n 95% vertrouensvlak geregverdig is.

ECSA Exit Level Outcomes Reference

Outcome	Reference	
	Section	**Page**
1. Problem solving: Demonstrate competence to identify, assess, formulate and solve convergent and divergent engineering problems creatively and innovatively.	*All*	*All*
5. Engineering methods, skills and tools, including information technology: Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology.	*2,3,4,5 & 6*	*7–56*
6. Professional and technical communication: Demonstrate competence to communicate effectively, both orally and in writing, with engineering audiences and the community at large.	*All*	*All*
9. Independent learning ability: Demonstrate competence to engage in independent learning through well developed learning skills.	*2,3,4,5 & 6*	*7–56*
10. Engineering professionalism: Demonstrate critical awareness of the need to act professionally and ethically and to exercise judgement and take responsibility within own limits of competence.	*All*	*All*

Acknowledgements

The author wishes to acknowledge the following people and institutions for their various contributions towards the completion of this work:

- My supervisor, Prof JH van Vuuren, for his dedicated support, expert advice and keen eye for detail.
- My family, for their continued support and encouragement.

Table of Contents

Abstract **iii**

Uittreksel **v**

ECSA Exit Level Outcomes Reference **vii**

Acknowledgements **ix**

List of Acronyms **xv**

List of Figures **xvii**

List of Tables **xix**

List of Algorithms **xxi**

1 Introduction **1**

1.1 Background 1

1.2 Research hypothesis 3

1.3 Scope and objectives 3

1.4 Project scope 4

1.5 Project methodology 5

1.6 Project timeline 6

1.7 Report organisation 6

2 Literature review **7**

2.1 The use of elevators in high-rise buildings 7

2.2 Prevailing elevator traffic conditions 8

2.3 Elevator control background 8

2.4 Single elevator traffic control algorithms 9

2.5 Group elevator traffic control algorithms 10

2.5.1 The nearest-car control algorithm 10
2.5.2 Sectoring control algorithms 11
2.6 The destination dispatch control algorithm 13
2.7 Queuing theory 15
2.8 The process of simulation design 16
2.9 Verification and validation of simulation models 18
2.10 Statistical performance analysis 20
2.11 Chapter summary 21

3 Algorithmic implementation 23
3.1 Selection of conventional elevator traffic control algorithms 23
3.2 The nearest-car control algorithm 23
3.3 The fixed-sectoring common sector system 25
3.4 The destination dispatch control algorithm 29
3.5 Chapter summary 32

4 The simulation model 33
4.1 The AnyLogic simulation modelling environment 33
4.2 General description of the simulation model 34
4.3 The dynamics of an elevator passenger 36
4.4 The dynamics of an elevator 37
4.5 Elevator traffic implementation 38
4.6 The graphical user interface 39
4.7 Simulation model verification and validation 41
4.8 Chapter summary 43

5 Experimental design 45
5.1 Key performance indicators 45
5.2 Sensitivity analysis 46
5.3 Statistical analysis of control algorithm effectiveness 47
5.4 Chapter summary 48

6 Results 49
6.1 Analysis of key performance indicator values 49
6.2 Statistical analysis of the impact of elevator group size 50
6.3 Statistical analysis of the impact of building size 53
6.4 Statistical analysis of the impact of floor population size 56

6.5 Chapter summary 56

7 Summary and conclusion **57**

7.1 Project summary 57

7.2 Appraisal of work 58

7.3 Possible future work 59

References **61**

A Project Timeline **65**

B Personal reflections **67**

B.1 Contribution to society 67

B.2 What the author has learned 67

C Results of non-parametric tests **69**

D Contents of the accompanying compact disc **83**

List of Acronyms

FS: Figure of Suitability

GUI: Graphical User Interface

KPI: Key Performance Indicator

RTT: Round Trip Time

List of Figures

1.1 A typical keypad used to enter destination floors in a destination dispatch regime 2
1.2 Clustering of passengers by different elevator control systems 3

2.1 Registration of landing calls and car calls 9
2.2 Numeric keypad used in destination dispatch control 13
2.3 Passengers grouped together by the destination dispatch control algorithm 14

3.1 Example the nearest-car control algorithm 24
3.2 Example of the fixed-sectoring common sector control algorithm 27
3.3 Example of the destination dispatch control algorithm 30

4.1 A selection of the global parameters declared in the `Main` object class 34
4.2 The `passenger` object class state chart 36
4.3 A selection of the parameters declared in the `passenger` object class 37
4.4 The `elevator` object class state chart 38
4.5 Screenshot of the simulation model GUI 40

6.1 KPI values observed for different values of the number of elevators 51
6.2 KPI values observed for different values of the number of floors 54
6.3 KPI values observed for different values of the floor population size 55

A.1 Project timeline in Gantt-chart form 65

List of Tables

3.1 FS-values for the elevator system depicted in Figure 3.1 25

4.1 Passenger arrival rates associated with a particular traffic condition 39
4.2 Comparison of the observed and analytically calculated RTT values 42

5.1 The independent variable values of the sensitivity analysis 46

C.1 Average journey time for all control algorithms, varying the number of elevators 69
C.2 Nemenyi test 1 varying the number of elevators (average journey time) 70
C.3 Maximum journey time for all control algorithms, varying the number of elevators 70
C.4 Nemenyi test 1 varying the number of elevators (maximum journey time) 71
C.5 Average journey time per control algorithm, varying the number of elevators . . 71
C.6 Nemenyi test 2 varying the number of elevators (average journey time) 72
C.7 Maximum journey time per control algorithm, varying the number of elevators . 72
C.8 Nemenyi test 2 varying the number of elevators (maximum journey time) 73
C.9 Average journey time for all control algorithms, varying the number of floors . . 73
C.10 Nemenyi test 1 varying the number of floors (average journey time) 74
C.11 Maximum journey time for all control algorithms, varying the number of floors . 74
C.12 Nemenyi test 1 varying the number of floors (maximum journey time) 75
C.13 Average journey time per control algorithm, varying the number of floors 75
C.14 Nemenyi test 2 varying the number of floors (average journey time) 76
C.15 Maximum journey time per control algorithm, varying the number of floors . . . 76
C.16 Nemenyi test 2 varying the number of floors (maximum journey time) 77
C.17 Average journey time for all control algorithms, varying floor population size . . 77
C.18 Nemenyi test 1 varying floor population size (average journey time) 78
C.19 Maximum journey time for all control algorithms, varying floor population size . 78
C.20 Nemenyi test 1 varying floor population size (maximum journey time) 79
C.21 Average journey time per control algorithm, varying floor population size 79

C.22 Nemenyi test 2 varying floor population size (average journey time) 80
C.23 Maximum journey time per control algorithm, varying floor population size . . . 80
C.24 Nemenyi test 2 varying floor population size (maximum journey time) 81

List of Algorithms

3.1 The nearest-car control algorithm 26
3.2 The fixed-sectoring common sector control algorithm 28
3.3 accumulatedJourneyTime (E) 31
3.4 The destination dispatch control algorithm 31

CHAPTER 1

Introduction

Contents

1.1 Background 1
1.2 Research hypothesis 3
1.3 Scope and objectives 3
1.4 Project scope 4
1.5 Project methodology 5
1.6 Project timeline 6
1.7 Report organisation 6

1.1 Background

Elevators are commonly employed in hotels, residential buildings and office buildings to transport people between floors. This means of transportation is preferred to staircases for the obvious reasons of speed and convenience, especially in the case of travelling multiple floors. In commercial buildings, elevator service supports the efficient movement of people throughout the building. This has the advantage of saving both time and money [7]. Naturally, people desire to spend as little time as possible waiting for, and travelling in, an elevator car [34]. Many different elevator control algorithms have been developed with the goal of improving elevator transportation efficiency by minimising a certain parameter, such as passenger waiting time or travel time [50]. Whereas conventional elevator control algorithms are adequate for low-demand traffic in low-rise buildings, they exhibit serious shortcomings in high-rise buildings.

The most common elevator control algorithm, often referred to as the *collective control algorithm*, allows users to indicate their chosen direction of travel using up or down pushbuttons outside the elevator. Users only register destination floors once they are in the elevator. The collective control algorithm operates on the following two principles [21]:

- An elevator will travel in its current direction, as long as it contains a passenger who wants to travel in that direction.
- Once the elevator has attended to all the requests in its current direction, it will reverse its direction if there is a request in the other direction. If not, the elevator will stop and wait for a request.

The collective control algorithm is particularly inefficient in high-rise buildings with large traffic demand. Consider, for example, a high-rise office building where demand occurs mainly upwards during the morning rush-hour. Suppose the building has only a single elevator. Once the fully loaded elevator car departs from the ground floor, it stops at numerous floors on its way to the highest selected floor. Passengers travelling to this highest floor therefore experience an extremely long travel time. Once the elevator switches direction it may have to service calls in the middle section of floors, in which case a lengthy queue may start forming on the ground floor. Throughout the day, the elevator will service mainly the middle floors and cause passengers at the very top and very bottom floors to experience long waiting times. Many buildings employ multiple elevators operating independently of one another under the collective control algorithm. This system is, however, still inefficient because people register landing calls at the different elevators and this results in more than one elevator being sent to the same floor. These inefficiencies have given rise to set elevator traffic control systems where elevators are allocated to landing calls according to a collection of rules with the purpose of maximising efficiency [7].

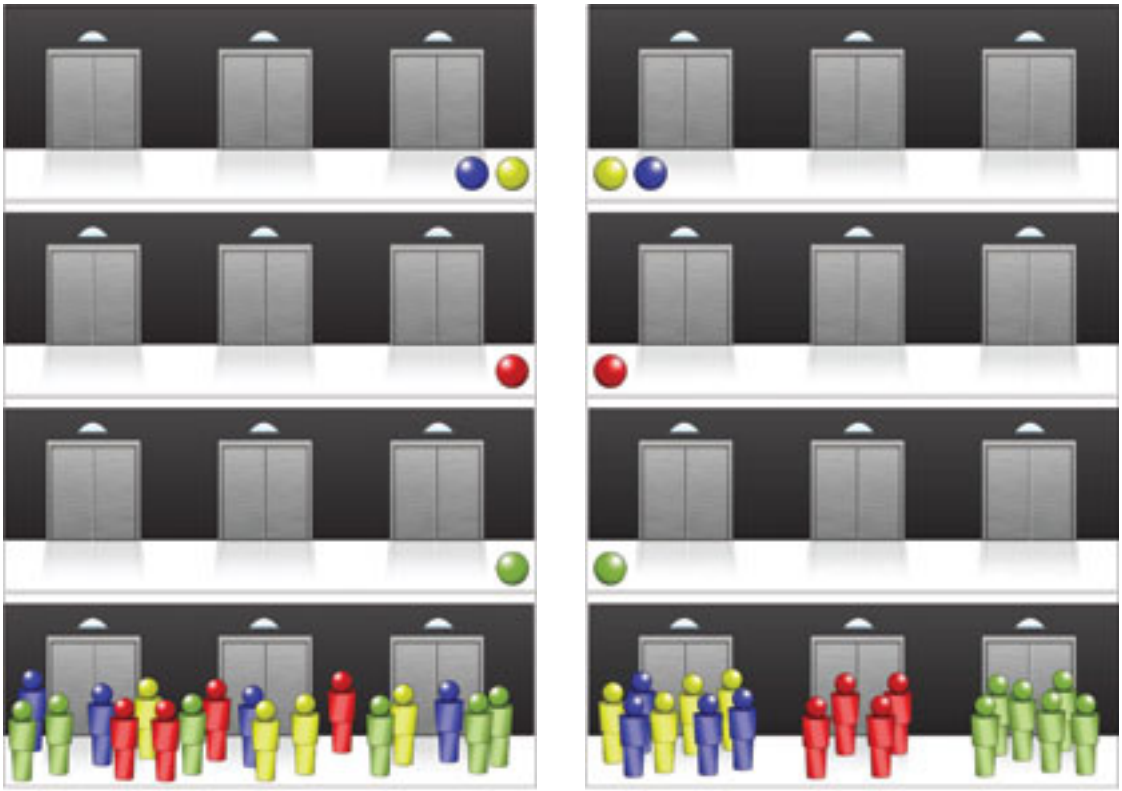
The so-called *nearest-car* approach entails implementation of the collective control algorithm for multiple elevators in an elevator group. Once a user registers a call, the algorithm determines the nearest, suitable elevator car to send to the passenger. The suitability of each elevator is determined based on the current travel direction of the elevator and the direction of the call [7]. In the case of large traffic demand, however, passengers may experience long travel times due to the numerous stops made by elevators in order to allow passengers to board and leave the elevator.

A different approach is embodied in the so-called *sectoring algorithm*, where each elevator in an elevator group exclusively services a sector of floors in the building [7]. The number of sectors typically equals the number of elevators in the building. Sectors may or may not consist of contiguous floors. A *fixed-sectoring control system*, which assigns each elevator to a static demand sector, is commonly employed to deal with off-peak traffic. Another variant is the *dynamic-sectoring control system* where sectors are dynamically defined in order to deal with changing demand patterns. The sectoring method greatly influences the overall number of stops and may thus affect reduced travel time.

FIGURE 1.1: *A typical keypad used to enter destination floors in a destination dispatch regime [15].*

Destination dispatch control is a modern control algorithm which is widely claimed to be more effective than conventional algorithms such as those described above. The control algorithm results in fewer stops per trip, less waiting time and less travel time compared to conventional systems [41]. Destination dispatch allows passengers to register their destination floors in the lobby before entering the elevator. An example of a keypad used to enter destination floors is shown in Figure 1.1. The system then immediately assigns the passenger a specific elevator to board or wait for. The control algorithm can pre-cluster destinations for each elevator because each passenger's destination is known in advance of elevator arrival at a demand floor. As a result, travel time and number of stops may be greatly reduced.

A schematic contrasting between the conventional dispatch and destination dispatch systems is shown in Figure 1.2. Figure 1.2(a) depicts passengers with different destinations travelling in the same elevator, whereas a clustering of passengers with the same destination in destination dispatch is shown in Figure 1.2(b).



(a) *Conventional dispatch* (b) *Destination dispatch*

FIGURE 1.2: *Passengers with different destinations (denoted by the different colours) travel in the same elevator in conventional dispatch systems, whereas passengers with the same destination are clustered by the destination dispatch control algorithm [19].*

1.2 Research hypothesis

The research hypothesis investigated in this study is that destination dispatch control is significantly more effective than a selection of the most popular conventional elevator control algorithms. Statistical evidence in support of, or in opposition to, this research hypothesis is gathered within the modelling paradigm of computer simulation. This evidence is presented and analysed in the form of an experimental design with varying factors including the number of floors in a high-rise building, the number of elevators in a high-rise building and statistical elevator passenger demand patterns.

1.3 Scope and objectives

The following eight objectives are pursued in this study:

I To *conduct* a thorough review of the literature related to:

(a) the use of elevators, specifically in high-rise buildings,

(b) the working of popular elevator control algorithms,

(c) the notion of destination dispatch in elevator control,

(d) the nature and characteristics of passenger arrivals in high-rise buildings for elevator service,

(e) simulation design, verification and validation methodologies applicable to the context of this project,

(f) appropriate statistical tests that may be employed to analyse the output results of a simulation study involving stochasticity.

II To *document* pseudo-code descriptions for a selection of the most popular elevator control algorithms, as well as for the relatively new method of destination dispatch. Such descriptions tend to be rather vague in the literature.

III To *establish* key performance indicators for evaluating the effectiveness of the elevator control algorithms described in pursuit of Objective II. These key performance indicators should sufficiently measure elevator control effectiveness from the passenger's perspective for whom waiting and/or travel time is typically a priority.

IV To *design* and *implement* a simulation model which can be used as a test bed for evaluating the effectiveness of the elevator control algorithms described in pursuit of Objective II in terms of the key performance indicators of Objective III. The simulation study should be able to accommodate a range of parameters related to the number of floors in a high-rise building, the number of elevators in a high-rise building, as well as different passenger arrival regimes.

V To *verify* and *validate* the simulation model of Objective IV according to the guidelines researched in pursuit of Objective I(e).

VI To *design* and *execute* a simulation experiment according to which the relative effectiveness of the algorithms documented in pursuit of Objective II can be measured within the simulation model test bed of Objectives IV–V in terms of the key performance indicators of Objective III. The experiment should follow an experimental design accommodating factors such as the number of floors in a high-rise building, the number of elevators in a building and different passenger arrival distributions.

VII To *present* statistical evidence in support of, or in opposition to, the research hypothesis of §1.2 under different conditions in respect of the number of floors in a high-rise building, the number of elevators in a high-rise building and different passenger arrival distributions.

VIII To *suggest* possible avenues of future investigation that may follow on the work reported in this project.

1.4 Project scope

The scope of the work reported in this project is limited in the following ways:

Passenger arrivals. A high-rise building may be classified as either a residential building or an office tower. The nature of passenger arrivals for elevator service in a residential building differs from that of passenger arrivals in an office building. In order to test the relative effectiveness of the elevator control algorithms fairly, all control algorithms must be implemented under identical elevator traffic conditions. A typical high-rise office building is therefore considered in this study. In such a high-rise office building, elevator demand will be mainly upward during the morning rush-hour and mainly downward during

the late afternoon rush-hour. Passenger arrivals for a typical work day will be modelled. Conditions of extreme or abnormal demand will not be considered.

Passenger behaviour. Ideal passenger behaviour is assumed in this study. This assumption implies that passengers do not change or cancel their elevator calls. Once a passenger has made a call, the passenger will enter the assigned elevator and will exit at the destination floor originally indicated.

Effectiveness measures. Effectiveness of the control algorithms will be measured from the perspective of elevator passengers only. Elevator capacity utilisation and energy consumption will not be considered as performance indicators for elevator control effectiveness in this study.

Elevator capacity. Elevators are assumed to have a fixed passenger capacity, irrespective of the number of floors in a building or the number of elevators in an elevator group. The notion to employ larger elevators in very tall buildings is therefore not taken into account in this study.

Elevator kinematics. Acceleration capabilities of elevators are taken into account in this project. These capabilities are, however, uniform for all elevators, irrespective of the number of floors travelled during a single trip. Increased acceleration capabilities of elevators in very tall buildings are therefore disregarded in this study.

1.5 Project methodology

The first stage in the execution of research toward this project consists of a thorough literature review, specifically aimed at the areas of the academic literature identified in Objective I of §1.3. The literature study provides a clear understanding of how the most popular elevator control algorithms work, as well as the advantages and disadvantages associated with each control algorithm. Understanding the working of these algorithms provides insight into how each control algorithm may perform in a typical high-rise office building. Methods according to which typical elevator traffic demand in an office building may be modelled are identified during the literature study. Simulation design paradigms, as well as simulation verification and validation methodologies relevant to this project are also reviewed. To conclude the literature study, statistical methods that may be used to analyse the output results of a stochastic simulation study are documented.

During the second stage of this study, pseudo-code descriptions are documented for a selection of the most popular elevator control algorithms in pursuit of Objective II. These pseudo-code descriptions are complemented by verbal descriptions and examples of the algorithmic implementations of these control algorithms.

The third stage of this study pertains to the design and implementation of a simulation model test bed to be used for evaluating the effectiveness of a selection of elevator control algorithms, in fulfilment of Objective IV. This simulation model is verified and validated in pursuit of Objective V. In the context of computer simulation, the process of model verification is followed to establish whether a model has been correctly built. Model validation, on the other hand, is the process followed to determine whether the model accurately represents the real world system.

The simulation model is verified and validated using the techniques researched in pursuit of Objective I(e).

A simulation experiment is designed and executed during the fourth stage of the research, in fulfilment of Objective VI, so as to determine the relative effectiveness of a selection of elevator control algorithms in pursuit of Objective IV. In order to determine whether the destination dispatch control algorithm is indeed the most effective, the simulation model is employed in order to test it against the conventional elevator control algorithms. Effectiveness is measured with respect to the key performance indicators identified in pursuit of Objective III. Testing the algorithms requires that they are implemented in the context of an extensive set of scenarios in which the number of floors in a high-rise building, the number of elevators in a high-rise building and the passenger arrival distributions are varied. This is achieved by means of a careful experimental design, capable of adequately reflecting the relative effectiveness of the elevator control algorithms considered.

Due to the variation in the number of floors, the number of elevators and passenger arrival distributions, it may be that different algorithms are the most effective in different scenarios. The simulation model output is therefore evaluated thoroughly in order to determine whether destination dispatch is the most effective control algorithm, and if so, under which circumstances. The evaluation is based on the statistical evidence gathered and presented in pursuit of Objective VII.

The report closes with a summary of the project contents, as well as suggestions with respect to possible future work and improvements that may follow on the work reported on in this study, in fulfilment of Objective VIII.

1.6 Project timeline

A Gantt-chart representation of the timeline followed during the execution of this project may be found in Appendix A.

1.7 Report organisation

Apart from this introductory chapter, this report comprises a further six chapters. Chapter 2 is devoted to a literature review of material relevant to this project. The focus in Chapter 3 falls on the algorithmic implementation of the selected elevator control algorithms considered in this project. A portion of the contents of Chapter 3 is regarded as a significant contribution to the literature. The computer simulation model developed for use as a test bed for evaluating elevator control algorithm effectiveness is described in Chapter 4. This simulation model lies at the heart of this project. Furthermore, Chapter 5 is devoted to the experimental design process followed to evaluate elevator control algorithm effectiveness by means of the simulation model of §4 for an extensive set of scenarios. The results obtained in the control algorithm effectiveness comparison analysis are presented and discussed in Chapter 6. Finally, the project closes in Chapter 7 with a summary of the project contents, an appraisal of the work performed and suggestions for possible future follow-up work.

Apart from Appendix A, mentioned in §1.6, three further appendices are included in this report. The author's non-academic reflections on this project are recounted in Appendix B, while results of the statistical tests carried out during the course of this project are provided in Appendix C. Finally, the contents of the accompanying compact disc are described in Appendix D.

CHAPTER 2

Literature review

Contents

2.1 The use of elevators in high-rise buildings 7
2.2 Prevailing elevator traffic conditions 8
2.3 Elevator control background 8
2.4 Single elevator traffic control algorithms 9
2.5 Group elevator traffic control algorithms 10
2.5.1 The nearest-car control algorithm 10
2.5.2 Sectoring control algorithms 11
2.6 The destination dispatch control algorithm 13
2.7 Queuing theory 15
2.8 The process of simulation design 16
2.9 Verification and validation of simulation models 18
2.10 Statistical performance analysis 20
2.11 Chapter summary 21

The purpose of this chapter is to provide the reader with an overview of the literature applicable to the context of this project. A brief overview of the use of elevators in high-rise buildings is provided in §2.1. The three prevailing types of elevator traffic conditions are presented in §2.2, while a background on elevator control is presented in §2.3. The workings of a selection of the most popular single and group elevator control algorithms are presented in §2.4 and §2.5, respectively. In §2.6 the focus shifts to the characteristics of the modern notion of destination dispatch control. The nature of passenger arrivals is presented in the context of queuing theory in §2.7. Attention is afforded to applicable simulation design methodologies in §2.8, while pertinent methodologies for the verification and validation of simulation models are reviewed in §2.9. Procedures for statistical performance analysis are presented in §2.10 and the chapter closes in §2.11 with a brief summary of the material included in this chapter.

2.1 The use of elevators in high-rise buildings

Primitive elevators or hoists have been used to move people and objects vertically since the third century BC [24]. Elevator technology has continually advanced ever since, but by the 19th century one major safety problem was still associated with the use of elevators: In the case of the hoist rope failing, the elevator car would fall. In view of this safety risk, people were

reluctant to use elevators to transport heavy equipment [13]. Then Elisha Otis invented the safety brake for elevators. He successfully demonstrated his invention at the New York Crystal Palace exposition in 1853 [24]. By 1857, the first public elevator was in operation in a New York building. Otis's invention had made elevators safe and practical which, in turn, paved the way for the construction of taller buildings [24].

Today, elevators are routinely used in commercial and residential high-rise buildings to provide safe and comfortable transport between floors. In commercial buildings, elevators save time and money, but in residential buildings money is often saved by not providing an elevator if there are not too many storeys [7]. As high-rise buildings are being built ever taller, elevator manufacturers are engaged in ferocious competition to build faster, more efficient and more economical elevators [33].

2.2 Prevailing elevator traffic conditions

A high-rise office building is subject to three distinct traffic patterns during a typical workday [7]:

Up-peak traffic. Up-peak traffic occurs when traffic flow is mainly, or only, in an upward direction with all, or the majority, of passengers entering the elevator system at the ground floor. Up-peak traffic typically manifests itself during the morning rush-hour when people arrive for work.

Random inter-floor traffic. Random inter-floor traffic involves the movement of people between floors in a building and exhibits no clearly recognisable direction or frequency patterns. Random inter-floor traffic occurs for the majority of the workday, between the morning rush-hour and the late afternoon rush-hour.

Down-peak traffic. Down-peak traffic occurs when traffic flow is mainly, or only, in a downward direction with all, or the majority, of passengers exiting the elevator system at the ground floor. Down-peak traffic is typically observed during the late afternoon rush-hour when people are leaving the building at the end of the workday.

2.3 Elevator control background

Elevator systems can exist either in the form of a single elevator, or as a number of interconnected elevators constituting an elevator group. The control of elevator systems comprises two engineering requirements. First, elevator control has to provide a means for commanding an elevator to move either upward or downward, and to stop at specified floors. Secondly, in an elevator group, the operation of individual elevators must be coordinated in such a way as to maximise the efficiency of the elevator group [7].

The means for passengers to make use of an elevator service is contained in the registration of calls for service. Upon arrival in the elevator lobby, passengers request elevator service by pressing a pushbutton on a panel located against a wall, as shown in Figure 2.1(a). The pushbutton is either an up or a down button, indicating the passenger's intended direction of travel. Registering a call by pressing either the up or the down pushbutton is referred to as a *landing call*. Once a passenger is inside an elevator, the destination floor is registered by pressing the floor number pushbutton on a numeric keypad as shown in Figure 2.1(b). This action of destination floor selection is referred to as a *car call*.

(a) *An up and down pushbutton panel used to indicate a passenger's desired direction of travel [23].*

(b) *A passenger registers the desired destination floor by pressing a pushbutton inside the elevator [25].*

FIGURE 2.1: *Passengers register landing calls by pressing either the up or down pushbutton on a panel located in the lobby. Once inside an elevator, passengers register car calls by pressing the appropriate floor number button.*

Elevator traffic control is governed by a set of rules defining the traffic control policy to be followed by the elevator system. This set of rules is called the *traffic control algorithm*. In 1970, Closs [11] defined five rules which must be obeyed by all traffic control algorithms for the operation of a single elevator:

- **Rule 1:** An elevator may not stop at a floor where no passenger enters or leaves the elevator.
- **Rule 2:** An elevator may not pass a floor where a passenger wishes to board.
- **Rule 3:** A passenger may not enter an elevator carrying passengers and request to travel in the opposite direction of the elevator's current travel direction.
- **Rule 4:** Whilst carrying passengers, an elevator may not reverse direction.
- **Rule 5:** Car calls are always prioritised over landing calls.

2.4 Single elevator traffic control algorithms

Single elevator systems are typically used in low-rise buildings with low demand for elevator service and are not suited for high-rise buildings. Single elevator traffic control nevertheless requires consideration in this project, for it forms the basis of the working of group elevator control algorithms.

Collective control is the most common form of single elevator traffic control. According to this form of control, landing and car calls are registered by pressing a pushbutton, as discussed in §2.3. Calls are answered in floor sequence and not according to the order in which the pushbuttons were pressed. The collective control algorithm, as discussed by Barney and Al-Sharif [7], may exist in one of three incarnations.

The first incarnation is *non-directional collective control*, where a single pushbutton is located at each floor. The pushbutton is pressed to register a landing call, irrespective of the passenger's intended direction of travel. Consider an elevator already containing passengers and travelling downwards. A person located at a floor below the elevator registers a landing call, but desires to travel upwards. The elevator will stop to answer the call, but will continue to travel in its original downward direction according to Rule 4 of §2.3, before travelling upwards to the desired floor. This form of collective control is only suitable for short-distance elevator trips.

The second incarnation is *down collective control*, where traffic is expected to occur between the ground floor and the upper floors only. This type of collective control also employs a single pushbutton call registration system. All landing calls are interpreted as down calls. Once the elevator has reached the ground floor, it will travel upwards again and only stop at the required floors to allow passengers who entered at the ground floor to exit. Thereafter, the elevator travels to the highest floor where a landing call was registered, after which it travels downward again, and so on, in each case answering landing calls and car calls in floor sequence.

The final incarnation is *full collective control*, where a passenger indicates his or her direction of travel by pressing either the up pushbutton or the down pushbutton at the origin floor. The elevator stops to answer both car and landing calls in its current direction of travel, and in floor sequence. In some instances, buildings have two or more elevators placed together which operate independently under the full collective control algorithm. A disadvantage of this system is the occurrence of so-called *bunching* where the elevators move together to service the same landing call when only one elevator is needed [38]. The simplest form of group elevator control is embedded in an implementation of the full collective control algorithm for a number of interconnected elevators. This form of group control is discussed in the following section.

2.5 Group elevator traffic control algorithms

In high-rise buildings with large elevator service demand, a group of elevators is typically needed to provide the required handling capacity. In this case, a group traffic control system is usually implemented to coordinate the operation of the individual elevators with the purpose of maximising elevator system performance. The working of four legacy group control algorithms is discussed in this section. These algorithms are the *nearest-car algorithm*, the *fixed-sectoring common sector system*, the *fixed-sectoring priority timed system* and the *dynamic sectoring system*.

2.5.1 The nearest-car control algorithm

The assignment of elevators in the simplest form of group elevator control is governed by the so-called *nearest-car* policy, discussed by Barney and Al-Sharif [7]. Each landing call is allocated to the elevator in the group best placed to answer the call. Every floor, except for the ground and topmost floors, is equipped with a single landing call system consisting of one up and one down pushbutton. Once a landing call is registered, a set of rules is followed to determine the elevator best suited to answer the call. This rule set (comprising four rules) is used to calculate a so-called *figure of suitability* (FS) for each elevator. The elevator with the highest FS is assigned to answer the landing call. The FS is calculated by taking into account the distance between the elevator and the landing floor, the current travel direction of the elevator and the direction of travel required by the landing call.

The distance, denoted by d, between the current elevator floor and a specific landing floor is measured in floor levels as

$$d = |\textit{elevator floor} - \textit{landing floor}\,|\,.$$

In a building with $N + 1$ floors (N floors above the ground floor), the following four rules are used to calculate the FS for each elevator once a landing call has been registered:

(a) $FS = (N + 2) - d$ if the elevator is moving towards the landing floor in the same direction as required by the landing call, when the call occurs.

(b) $FS = (N + 1) - d$ if the elevator is moving towards the landing floor, but in the opposite direction as required by the landing call, when the call occurs.

(c) $FS = 1$ if the elevator is travelling away from the landing call, irrespective of the value of d, when the call occurs.

(d) $FS = (N + 1) - d$ if an elevator is idle. Note that this is the same expression as employed in Rule (b).

Should the FS-values be the same for two or more elevators, the landing call is assigned to the nearest elevator. If the distances are also equal, the landing call is assigned to the first elevator that reached the FS-value at some point prior in time. Fully loaded elevators are not considered during the allocation procedure. Once an elevator has answered all pending landing calls and car calls, it stops and waits at its current floor.

The nearest-car control algorithm was specifically designed to operate under random inter-floor traffic conditions in low-rise buildings. Its greatest disadvantages are exhibited under up-peak and down-peak traffic conditions. Under up-peak traffic conditions, elevators remain idle at upper floors once all passengers have alighted, and only travel to the ground floor once a landing call is registered there. Under down-peak traffic conditions, elevators typically provide good service to the upper floors whilst neglecting the lower floors. In low-rise buildings, where people are expected to use the stairs at the lower floors, the nearest-car control algorithm may achieve acceptable performance [7].

2.5.2 Sectoring control algorithms

In a high-rise building, a single elevator is typically not required to service every floor. Consider an elevator servicing the majority of, if not all, the floors in a high-rise building during a single trip. This would necessitate a large number of stops and long travel times for passengers [5]. To address this inconvenience, sectoring elevator control systems were conceived with the aim of minimising the number of elevator stops during a single trip [10, 52]. Another distinct advantage of sectoring control systems is increased elevator system handling capacity [51].

The sectoring approach entails segmenting a building into a number of adjacent sectors. The number of sectors typically equals the number of elevators in the group and a sector usually consists of a number of contiguous landing floors. Each sector is assigned to an elevator and the elevator typically only responds to landing calls from within its sector.

The sectoring control algorithm, discussed by Barney and Al-Sharif [7], is implemented in three distinct incarnations, namely as a *fixed-sectoring common sector system*, as a *fixed-sectoring priority timed system*, or as a *dynamic sectoring system*.

According to the *fixed-sectoring common sector system*, a building is segmented into a number of static demand sectors, equal to the number of elevators in the group. An elevator is assigned to a sector if it is present in the sector, and if the sector is not already committed to another elevator. Elevators respond to landing calls registered within their respective sectors under the full collective control system, but can also answer landing calls outside their respective sectors. An elevator can respond to vacant sectors above and adjacent to its own sector, and can also respond to landing calls matching its moving direction whilst travelling to its next stop. An elevator is de-assigned from a sector if it leaves the sector or is fully loaded. Fully loaded elevators are not considered during the allocation procedure. After de-assignment from a sector, an elevator responds to all of its pending landing calls and car calls, after which it is assigned to a vacant sector [7].

Through equal distribution of elevators in a building, the fixed-sectoring common sector system is successful under balanced random inter-floor traffic conditions. Its performance under up-peak and unbalanced inter-floor traffic conditions is appreciable [5]. One drawback of the system is that it does not provide a method for dealing sufficiently with sudden demand peaks at particular floors.

The *fixed-sectoring priority timed system* involves partitioning the building into a number of static directional sectors and allocates elevators on a priority timed basis. It performs the best in light to heavy balanced random inter-floor traffic conditions [6]. Each sector is assigned a priority level, determined by the passenger waiting time. Six priority levels are usually employed with level six denoting the highest priority. Once a landing call is registered, the sector is timed. A priority level may follow the timed sequence as 10, 20, 30, 40, 50, 60 seconds. As the passenger waiting time exceeds the predefined threshold times in the sequence, the associated sector is assigned the corresponding priority level. For a sector with the above-mentioned threshold sequence, for example, a call is assigned a priority level of 3 if the waiting time exceeds 30 seconds.

The allocation of elevators to sectors depends on the number and positions of available elevators, as well as the sector priority levels. The sector with the highest priority is first assigned to an elevator. If more than one elevator is free, the nearest elevator is allocated to the sector. Once assigned to a sector, an elevator travels to the sector without stopping. After leaving the assigned sector, the elevator responds to landing calls matching its direction of movement. Once the elevator becomes available again, it is assigned to the vacant sector with the highest priority. Fully loaded elevators are not considered during the allocation procedure.

The *fixed-sectoring priority timed system* uniquely considers time during the elevator assignment procedure and can be implemented to give priority to specific floors. Consider a medium priority sector with threshold sequence 15, 25, 35, 45, 55, 65 seconds. In contrast, a low priority sector may be assigned a threshold sequence of 30, 50, 70, 90, 110, 130 seconds. As such, the medium priority sector may be expected to experience better elevator service.

The *dynamic sectoring* control system involves partitioning the building into a number of sectors dynamically. The number of sectors and the sizes of the various sectors are determined by the instantaneous positions and travel directions of the individual elevators in the elevator group. Elevators are assigned to sectors and respond to calls in their sectors according to the rules of the full collective control system. Each sector's size is defined as the range from one elevator to the next elevator ahead of it travelling in the same direction or in an idle state. This control system is best suited for light to heavy balanced random inter-floor traffic conditions [7].

2.6 The destination dispatch control algorithm

A common feature in all of the algorithms described in §2.4 and §2.5 is that the destination floors are selected only once a passenger is inside an elevator. Consider, however, a high-rise building with large elevator traffic demand operating under one of the conventional elevator control algorithms. In a fully loaded elevator, there is a large probability that the elevator will service numerous destination floors and that a number of elevators in the elevator group will simultaneously service the same range of destination floors [54]. According to research by De Jong [16] from elevator manufacturer KONE, a single elevator stop may take between 10 and 13 seconds. Clearly, this may have a compounding negative effect on travel time for passengers travelling to the upper floors in a high-rise building. The destination dispatch control algorithm attempts to address this shortcoming by obtaining the desired destination floor before assigning an elevator to a passenger. Knowing both the landing floor and destination floor, the control algorithm may assign elevators to passengers with a view to maximise the effectiveness of the elevator system. Under destination dispatch control, individuals may experience increased waiting times in the lobby, but their travel time is expected to decrease and, as a result, their overall journey time may decrease [7, 26].

Destination dispatch control does not utilise the up and down pushbutton system in the lobby as in conventional elevator control systems. Instead, a numeric destination keypad is installed at each landing floor. Passengers are required to key in their destination floors using the keypad. The algorithm then assigns an elevator to each passenger by indicating the number of the elevator scheduled to collect him or her on the keypad screen, as shown in Figure 2.2. The passenger proceeds to wait for the assigned elevator to arrive, or can immediately board it if it is available at the current floor. The control algorithm keeps a record of the destination floors, and passengers are not required to enter their destination floors again upon entering the elevator. The elevator automatically stops at the required destination floors.

FIGURE 2.2: *The numeric keypad used in destination dispatch control to enter the required destination floor. The passenger is assigned to elevator A in this case [45].*

Elevator assignments are made to minimise a particular cost function. For each landing call registered, the control algorithm allocates a virtual cost to each elevator and assigns the call to the elevator which achieves the lowest cost. Each elevator manufacturer typically adopts its own, unique cost function, which is normally proprietary knowledge. Typical cost functions include the average passenger waiting time, the average passenger travel time, or a combination of both [7]. Whereas these cost functions may be fixed for all traffic conditions, modern destination dispatch control algorithms, such as that used by KONE's Polaris system, make use of artificial intelligence, fuzzy logic and genetic algorithms to assess elevator traffic in real-time and adjust their optimisation techniques accordingly [35]. Elevator assignments occur instantly after a landing call has been registered and cannot be altered. It may happen that subsequent landing calls will require a reassignment of elevators for improved service, but this possibility is not catered for in destination dispatch control.

It is widely claimed that destination dispatch control is more effective than conventional control systems. Elevator manufacturer Schindler has, for example, claimed that its Miconic 10

destination dispatch control system reduces journey time by an average of 30% [20]. Otis has similarly claimed that its Compass Plus system is 50% faster than conventional management systems [12]. Al-Sharif [50] furthermore reported that the time to reach a destination may be reduced by 25%–40% compared to conventional control systems.

A number of advantages are associated with destination dispatch control:

- The system's greatest advantage is exhibited during up-peak traffic conditions. Passengers are grouped according to common destinations and, as a result, the number of stops are greatly reduced [16]. This kind of grouping of passengers is illustrated in Figure 2.3.
- Elevators are less crowded which increases comfort for passengers [54].
- The number of elevators required within the elevator group may be fewer than that required by other control algorithms [54].
- Due to the reduced number of stops, elevators can move faster to contribute to decreased journey times [54].
- Destination dispatch delivers increased handling capacity compared to conventional control systems [35].

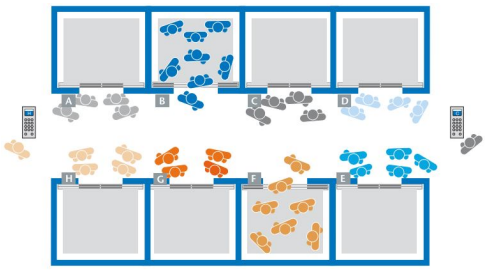


FIGURE 2.3: *Top view of passengers travelling to common destination floors (as indicated by the various colours), grouped together by the destination dispatch control algorithm [35].*

There are, however, also distinct disadvantages associated with destination dispatch control:

- Passengers must accept that other elevators may arrive and depart before their assigned elevator arrives [54].
- Misuse of the numeric keypads may have a serious negative impact on traffic flow [54].
- Passengers may not reach their assigned elevator in time if the lobby is crowded [50].
- The system exhibits a lack of flexibility due to the fact that elevator assignments are fixed once made [50].
- Passengers may experience substantial waiting times between the assignment of an elevator and the arrival of their elevators [50].
- The system does not differentiate between a single passenger and a group of passengers. It may therefore happen that an elevator stops to pick up more passengers than it has capacity for and hence that some passengers in the group experience increased waiting times. This problem may be solved by adding a *group function* button to the destination keypad, where passengers can define the size of the group of passengers [33].

Destination dispatch control is most effective under up-peak traffic conditions, but less so during down-peak traffic conditions where the only destination floor is typically the ground floor. As a result, destination dispatch control systems can either take the form of a hybrid destination dispatch configuration or a traditional destination dispatch configuration. The hybrid system typically constitutes destination keypads installed only at the ground and other main floors in the building. The remaining floors are fitted with the conventional up and down pushbutton systems. Elevators are fitted with the conventional destination floor pushbuttons. The hybrid system is most effective for improving traffic flow from heavy demand floors. In a traditional destination dispatch configuration, on the other hand, all floors are fitted with the numeric destination floor keypads. This system provides the best service for most traffic conditions and is recommended for buildings with large traffic peaks [35].

2.7 Queuing theory

Elevator traffic systems naturally lend themselves to the application of queuing theory. Queuing theory may be applied to evaluate performance measures such as passenger queue length, passenger waiting time in a queue and service time for elevator systems. The passengers are defined as customers arriving for service, the elevator lobby is a *first come, first served* (FCFS) queue and a set of parallel servers are represented by the elevators.

Kendall [31] proposed a standard notation for describing queuing systems, containing six characteristics specified in the form 1/2/3/4/5/6. The first characteristic defines the arrival process. It is commonly accepted that passengers arrive for elevator service in a lobby according to the Poisson probability distribution [7]. Although the Poisson distribution is not an exact representation of passenger arrivals for elevator service, research by Alexandris [1] proved it to be a suitable fit under most conditions. The probability mass function of the Poisson distribution is given by

$$P(n) = \frac{(\lambda T)^n}{n!} e^{(-\lambda T)}, \quad n = 0, 1, 2, \ldots,$$

where $P(n)$ is the probability that n passengers arrive within a time interval of length T and where λ denotes the average rate of arrival of passengers per time unit. If passenger arrivals are assumed to be governed by a Poisson process, then interarrival times are exponentially distributed [53], in which case the first characteristic in the Kendall queue notation is denoted by an M.

The second characteristic describes the nature of the service times. For elevator systems, service time is usually defined as the time required by an elevator to pick up a passenger, travel to the destination floor and return to the original floor [8]. An assumption of exponentially distributed service times would significantly simplify the performance analysis of an elevator system, but is rarely justified in practice [8]. This is because the number of passengers receiving service can range from one to a maximum batch size as determined by the elevator capacity. The service times in elevator traffic systems follow some general distribution, denoted here by G. This distribution is typically estimated empirically on an application-by-application basis, based on real data.

The number of parallel servers (in this case the number of elevators) is represented by the third characteristic in Kendall's queue notation. The fourth characteristic describes the queue discipline which, as mentioned above, is classified as FCFS. The fifth characteristic defines the maximum permitted number of customers (passengers) in the system. This includes customers in the lobby and in all the elevators, usually assumed to be very large. The sixth characteristic

specifies the size of the population from which the customers, or passengers, are drawn, here assumed to be infinitely large.

A typical queuing system for an elevator traffic system with s elevators in an office building may therefore be specified in Kendall's notation as an $M/G/s/FCFC/\infty/\infty$ system. In this idealistic queuing system it is assumed that there are no physical limitations on the maximum number of passengers in the lobby. Also, the population size is considered infinite assuming that not only residents or employees will arrive for elevator service, but visitors to the building as well.

Suppose passengers in an elevator system are serviced at a rate of μ passengers per second. Based on the assumption that passengers are served in batches, where each batch has a fixed maximum size $P_{\max}$ [2], the service rate μ is given by

$$\mu = \frac{P_{\max}}{\tau},$$

where τ denotes the expected *round trip time* (RTT) in seconds for an elevator filled to capacity. The RTT is defined as the time that elapses from the moment the elevator doors open at the ground floor until the doors reopen again at the ground floor upon completion of its trip around the building [7]. Barney [7] proposed the expression

$$RTT = 2Ht_v + (S+1)t_s + 2Pt_p$$

for the RTT (in seconds), where H is the average highest reversal floor[1], t_v the time duration required by an elevator to travel past two adjacent floors at its rated speed, S is the expected number of stops during a trip, t_s is the expected time associated with each stop, P is the number of passengers in the elevator and t_p is the average time required by a passenger either to enter or exit an elevator.

Jones [30] derived an expression for the expected number of stops S in a building with equal floor populations. Suppose there are N floors above the ground floor and that all floors are equally likely destinations for passengers. Then the expected number of elevator stops is

$$S = N\Big[\left(1 - \Big[\frac{N-1}{N}\Big]^P\right)\Big],$$

where P again denotes the number of passengers in the elevator. Under the same assumptions mentioned above for the derivation of S, Schroeder [47] derived the expression

$$H = N - \sum_{i=1}^{N-1}\Big[\frac{i}{N}\Big]^P$$

for the expected highest reversal floor.

2.8 The process of simulation design

Shannon [48] defined simulation as "the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behaviour of the system and/or evaluating various strategies for the operation of the system." A powerful feature of simulation modelling is that it allows for the study of both existing systems and systems in

[1]The highest reversal floor H is the highest floor serviced by an elevator during a trip.

the design stage. As such, simulation can be used as an analytical tool to predict the impact of changes to existing systems or as a design tool to predict the performance of new systems by evaluating them under a range of conditions [4]. Compared to analytical and mathematical models, the concept of simulation is often easier to comprehend, produces more accurate results since it considers the behaviour of the real-world system and typically does not contain as many simplifications. Simulation studies can furthermore be executed without affecting the current system and do not require any resources to be committed to their implementation [49]. It is important to note, however, that simulation models provide (typically probable) outputs for a specified set of inputs as opposed to yielding an optimal solution to some decision problem. Instead, the focus of a simulation study should remain on the analysis of system behaviour under the specified set of circumstances [49].

In order to conduct a sound and successful simulation study, Banks *et al.* [4] recommended a series of twelve steps to be followed by the simulation model builder. Similar processes have been proposed by Law and Kelton [36] and by Shannon [48]. The twelve-step approach of Banks *et al.* [4] for conducting a successful simulation study is as follows:

1. *Problem formulation.* The problem of interest is clearly stated by the decision-maker [36]. It may happen that a problem is not yet fully understood by the time of formulation, in which case the decision-maker may return at a later stage to reformulate the problem once more information has been obtained.

2. *Setting of objectives and an overall project plan.* The objectives of the simulation study are defined by the questions that should be answered by the simulation. These questions should be specific in order to determine the required level of model detail [37]. The project plan should discuss alternative system designs which may be considered and criteria for measuring the effectiveness of these alternatives. It should also outline plans in respect of the number of analysts required, the cost of the study and the time needed to accomplish the objectives.

3. *Model conceptualisation.* The conceptual model is constructed. According to Robinson [42], a conceptual model is a "non-software specific description of the simulation model that is to be developed, describing the objectives, inputs, outputs, content, assumptions and simplifications of the model." The descriptions may either be in graphical form, such as flow charts, or in pseudo-code form [49].

4. *Data collection.* Information and data related to the problem at hand are collected and used to specify model parameters and probability distributions [36, 39]. The data may also be used to validate the model. The objectives of the study largely govern the type of data to be collected.

5. *Model translation.* The model is programmed in either a general-purpose programming language or in a dedicated simulation software package [49]. Compared to programming languages, simulation software packages boast improved flexibility and often greatly reduce model development time. They usually also offer built-in simulation visualisation or animation capabilities.

6. *Verification.* The purpose of the verification process is to evaluate whether the computer program functions properly [37, 49]. A number of verification methodologies are reviewed in some detail in the next section.

7. *Validation.* The model is validated to establish whether its accuracy is deemed acceptable [37, 39, 49]. A number of validation methodologies are reviewed in the following section.

8. *Experimental design.* The alternative system designs to be simulated are determined. For each configuration, decisions are made with respect to the length of the warm-up period, the required length of the simulation runs and the number of replications to be executed for each run [37, 39, 49].

9. *Production runs and analysis.* Production runs are performed and the results are analysed in order to determine the level of performance for each of the simulated system designs in the experiment.

10. *More runs?* Based on the results of the completed production runs, the analyst decides whether additional runs involving different designs are required.

11. *Documentation and reporting.* The conceptual model, a thorough description of the computer implementation and results are documented for current and future use [36]. Documentation is generated for the sake of the modeller (so that he or she can remember what has been done) and also for use by new modellers working on the same model in the future [42].

12. *Implementation.* The simulation model is implemented in practice [37, 49].

2.9 Verification and validation of simulation models

A prime concern for the developers and users of a simulation model, as well as the people affected by the decisions made based on the model, is whether a model and its output are correct [44]. The credibility of a model is evaluated through the processes of model verification and validation. Model verification involves determining whether a model has been built correctly and has been implemented successfully in the simulation software environment [4]. Validation, on the other hand, is the process of establishing whether the correct model has been built. Its goal is to determine whether the model is a credible representation of the real system [4].

Banks *et al.* [4] have proposed the following verification techniques:

1. Have the operational model evaluated by a specialist in the simulation software used [36].

2. Identify and create a flow chart of all logically possible actions a system can take for all different events which may occur. For each action associated with each event type, thoroughly evaluate the model logic.

3. For a range of settings of the input parameters, evaluate the model output for soundness [36]. The model should display a range of output statistics which should be carefully evaluated.

4. Print the input parameters at the end of the simulation run to ensure that these parameter values have not been changed accidentally.

5. Provide clear definitions of every variable used and descriptions of the purpose of all submodels, components and procedures. There is also a need to provide comments throughout the programming code used in the software [3].

6. In the case of an animated simulation model, verify that the real system is correctly replicated in what is seen. Due to the visual nature of animation, illogical actions can easily be detected [3]. Analysts are encouraged to execute long simulation runs so as to eliminate the possibility of missing errors which may not occur during short runs [32].

Although validation is listed as a single step in the simulation design process in §2.8, it is, in fact, an iterative process during which differences between the model and system behaviour are continually evaluated and used to improve the model. This iterative process is referred to as *calibration*. Both subjective and objective validation techniques may be used to validate a model. The subjective techniques include face validation, sensitivity analysis, validation of model assumptions and Turing tests.

Face validation involves testing whether a model appears credible at face value to the model users and all people with knowledge pertaining to the actual system being simulated. It is used to establish whether the logic in the conceptual model is correct and whether the model's input-output relationships are credible [4]. In order to ensure that a credible model is constructed, potential users and knowledgeable people should continually be involved in the process of model conceptualisation until its implementation. They may also evaluate model output to identify model shortcomings and advise on improvements to be made.

In order to further evaluate a model's validity, a sensitivity analysis may be performed. A sensitivity analysis involves testing whether the model behaves in the expected way when one or multiple input variables are changed [3, 32]. Owing to experience or observations of the actual system, the model developer and user would have some idea of what the expected output will be for a specific change in an input variable. If it proves to be too time-consuming and expensive to vary all input variables, the model developer should aim to evaluate the model sensitivity for the most critical input parameters only [4].

Another validation technique is concerned with the validation of model assumptions made during the design process. Model assumptions may be classified as either structural assumptions or data assumptions [4]. Structural assumptions are the simplifications and abstractions made of the real-world system and are validated through observation of the real system, as well as discussions with people involved in the real system. Data assumptions, on the other hand, are based on the collection of reliable data and a sound statistical analysis of these data [4]. Data reliability is verified by means of statistical tests and discussion with knowledgeable people involved with the real system.

Another subjective validation test used to validate input-output transformations is a Turing test. People familiar with system behaviour are used to compare model output to system output [32, 43]. Under identical input conditions, a number of reports of real system performance are generated and a number of "fake" reports are generated by means of simulation output data. The reports are randomly ordered and given to experts to evaluate. If an expert is able to distinguish between the "fake" and actual reports, the model is deemed inadequate and is improved with the help of the expert [3].

Whereas Turing tests are subjective, input-output transformations may also be validated by means of objective techniques. These techniques require that the real system exists and involves comparison of the simulation model output with data from the actual system [3]. The first such technique evaluates the model's ability to predict the actual system's behaviour, given that the model input data match real input. A sound simulation model should make accurate predictions for a variety of input data sets [4]. A second technique for validating input-output transformations involves the use of historical data. Taking historical data as input, a proper simulation model should be able to output results compatible with historical real system output [43].

2.10 Statistical performance analysis

In the context of comparative experiments involving some form of stochasticity, sampling methods may be used to withdraw performance measure values from a system and the question arises whether such samples differ statistically for different systems. Methods from the realm of statistical inference may be used to address this question with the aim of drawing conclusions about the system populations, utilising the partial knowledge obtained from the samples.

More specifically, statistical tests are performed to investigate hypotheses made about some properties of one or multiple populations. The choice of test is governed by the type of experiment conducted, the distribution of the data and type of variables involved. *Non-parametric tests* apply in cases where few assumptions are made about the underlying populations from which data are obtained [29]. A non-parametric test is a hypothesis test in which the statement is not concerned with parameter values of a statistical distribution of the underlying data [27].

The *Wilcoxon signed rank test* is non-parametric and may be used to determine whether two samples represent two different populations [18]. The test aims to identify notable differences between two sample means under the assumption that the sampled population is symmetric [14, 27]. The null hypothesis is that the population median is equal to the hypothesised median [18]. In this test, the differences between data point pairs are calculated and the absolute values of these differences are ranked. Each rank is labelled with a sign — positive if the difference between a data point pair is positive and negative if the difference between a data point pair is negative. The ranks with positive and negative signs are summed, respectively. The smallest value of either two summations is used as the test statistic [14]. If the test statistic value is less than or equal to the critical value for a specified level of statistical significance, the null hypothesis is rejected.

The *Friedman test* is also non-parametric and may be used to determine, in a set containing two or more samples, whether at least two samples are representative of populations with different median values. The null hypothesis is that population medians are equal and the alternative hypothesis states that not all population medians are equal [18]. This procedure is based on the conversion of the original observations to ranks. For each data set, the observations are ranked from least to greatest and assigned a rank value. These rank values are used to calculate the test statistic which is distributed according to a chi-squared distribution [18]. The null hypothesis is rejected if the test statistic is larger than the critical value of the distribution at a specified level of statistical significance.

Should the null hypothesis of a Friedman test be rejected, the non-parametric *Nemenyi post hoc procedure* may be followed. Whereas the rejection of the aforementioned null hypothesis only indicates that a significant difference exists between at least two of the samples, the Nemenyi procedure may be used to identify the actual samples that differ [46]. The Nemenyi procedure performs pairwise significance tests between all sample pairs and compensates for the multiple inferences it makes [28, 29]. The correction for the multiple inferences is important because it ensures that the specific significance level of the experiment is adhered to. The performance of two sample pairs is significantly different if their average rank values (obtained in preparation of performing the Friedman test) differ by at least a particular value, known as the *critical difference* [17].

In statistical hypothesis testing, the decision to reject, or not reject, the null hypothesis based on the computed test statistic value does not reveal how strong the evidence is to support the decision. In the event where the null hypothesis is rejected, for example, questions may arise as to how far the test statistic falls within the rejection region. In order to address this

issue, the so-called p-value is used to draw a conclusion regarding the statistical significance in a hypothesis test. A considerable advantage of the p-value is exhibited in the sense that it provides information regarding the weight of evidence against the null hypothesis. The p-value represents the smallest level of significance that would yield rejection of the null hypothesis [40]. Typically, a small p-value presents strong evidence against the null hypothesis. A p-value smaller than the predefined significance level α, will lead to rejection of the null hypothesis and represents the probability of rejecting the null hypothesis when, in fact, it is true [40].

2.11 Chapter summary

A brief overview of the literature related to elevator control, simulation design and statistical performance analysis was provided in this chapter. The use of elevators, three prevailing elevator traffic conditions, as well as practices for elevator control were first reviewed. This was followed by a description of various single elevator control algorithms, which formed the basis for the discussion of the group elevator control algorithms discussed next. Advantages and disadvantages associated with the control algorithms were noted in order to develop an understanding of the performance capability of each control algorithm. The relatively new method of destination dispatch was reviewed next with an emphasis on its performance compared to conventional elevator control algorithms. Aspects of queuing theory related to passenger arrivals in the context of elevator service were also discussed. A twelve-step approach for conducting a sound simulation study was presented together with a review of appropriate simulation model verification and validation methodologies. Finally, three procedures which may be utilised to analyse algorithmic performance statistically were reviewed.

CHAPTER 3

Algorithmic implementation

Contents

3.1 Selection of conventional elevator traffic control algorithms 23
3.2 The nearest-car control algorithm 23
3.3 The fixed-sectoring common sector system 25
3.4 The destination dispatch control algorithm 29
3.5 Chapter summary 32

The purpose of this chapter is to provide the reader with an enhanced understanding of a selection of the elevator control algorithms discussed in §2.5 and §2.6 which are to be analysed in this project. These control algorithms are introduced in §3.1 and their selection is briefly motivated. The implementations of these control algorithms are described in detail in §3.2, §3.3 and §3.4. Pseudo-code descriptions and examples of the working of the control algorithms are presented in each case. The chapter finally closes with a brief summary in §3.5.

3.1 Selection of conventional elevator traffic control algorithms

Four legacy elevator group control algorithms were discussed briefly in §2.5. Two of these conventional elevator group control algorithms, along with the destination dispatch control algorithm, have been selected for implementation in this project. These algorithms are the *nearest-car control algorithm* and the *fixed-sectoring common sector system*. The nearest-car control algorithm was selected by virtue of its inherent differences compared to the sectoring approach followed by the sectoring control algorithms. Finally, in order to reflect the working of a sectoring algorithm, the fixed-sectoring common sector system was arbitrarily selected and the other sectoring control algorithms disregarded in view of time constraints imposed by this project.

3.2 The nearest-car control algorithm

The working of the nearest-car control algorithm, discussed in §2.5.1, is particularly well-documented in the literature. Therefore, the implementation of this algorithm in a computer simulation model is fairly straightforward and only a few assumptions have to be made during its implementation.

Barring one scenario that may occur during elevator allocation, the allocation procedure is unambiguous, exhaustive and therefore simple to implement. In the case where two or more elevators achieve the largest FS-value for a particular landing call, the nearest elevator is assigned to the landing call, as mentioned in §2.5.1. Should two or more of those elevators share the same minimum distance from the landing floor, the elevator that achieved the shared FS-value first is assigned to the landing call. The algorithm does not, however, provide for the case where none of the elevators had previously reached the particular FS-value. For the purposes of this project, the assumption is therefore made that any one of the suitable elevators is randomly assigned to the landing call in this exceptional case.

The elevator allocation procedure neither accommodates the case where an elevator is fully loaded nor does it keep record of the number of people waiting for an elevator at a particular floor. This presents the possibility of a passenger being assigned to an elevator only to find that the passengers in front of him or her in the queue fill the elevator up to capacity when boarding. In this case it is assumed that the unserved passengers will re-register their landing calls immediately after failing to enter the elevator originally assigned to them. These passengers are therefore assigned to a new elevator. This process is repeated, if necessary.

The control algorithm has a fixed implementation irrespective of the prevailing elevator traffic conditions. For this reason, the algorithm performs poorly under the up-peak traffic condition, as discussed in §2.5.1. In this case, the effectiveness of the algorithm may be improved significantly if elevators immediately return to the ground floor after having responded to all pending car calls. The generic description of the control algorithm in the literature, however, only allows for elevators to travel towards the ground floor in response to landing calls. For the purposes of this project, the implementation of this algorithm strictly follows the generic description presented in the literature. In the interest of remaining true to the spirit of the algorithm, the aforementioned obvious algorithmic improvement during up-peak traffic therefore remains unexploited in this project. A pseudo-code description of how an elevator is assigned to a new landing call according to the nearest-car control algorithm is given in Algorithm 3.1.

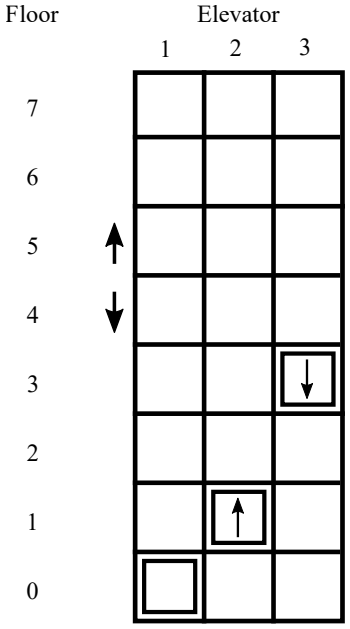


FIGURE 3.1: *Example of an elevator system operating under the nearest-car control algorithm.*

The working of the nearest-car control algorithm is illustrated in closing by means of a simple example. Consider the three-elevator system illustrated in Figure 3.1. The building has eight floors, including the ground floor. At a particular time instant the elevator positions and directions of movement are as shown, with elevator 1 being idle at the ground floor. An up landing call is registered at floor 5, while a down landing call is simultaneously registered at floor 4. Elevator 2 is travelling towards floor 6 in response to a car call.

The FS-values calculated for each elevator and for both landing calls during the elevator allocation procedure are presented in Table 3.1. The FS-values are calculated according to the rule set discussed in §2.5.1. Consider the calculation of the FS-values for both landing calls with respect to elevator 2. The elevator is moving towards the landing call at floor 5 in the same direction as required by the landing call. For an eight-storey building, $N = 7$. The distance between the landing and elevator floors is 4. Therefore, $FS = 7 + 2 - 4 = 5$. For the down landing call, on the other hand, elevator 2 is moving towards the landing call, but in the opposite direction from that required by the landing call. The distance is three floors. Therefore, $FS = 7 + 1 - 3 = 5$ and so elevator 2 is assigned to this landing call as well.

This is clearly an undesirable solution since elevator 2 will first respond to the up landing call (while travelling to floor 6), before returning to answer the landing call at floor 4. It would seem that elevator 1 is, in fact, better suited to answer the down landing call, because it is idle at the ground floor. Elevator 1 is not, however, assigned to the down landing call, because it is one floor further away from the call than elevator 2 and the same expression is used to calculate the FS-values for both elevators. This example illustrates a deficiency in the nearest-car control algorithm.

Landing call	Elevator 1	Elevator 2	Elevator 3
4-Up	3	5	1
3-Down	4	5	1

TABLE 3.1: *FS-values for the elevators and landing calls in the elevator system depicted in Figure 3.1.*

3.3 The fixed-sectoring common sector system

As discussed in §2.5.2, the fixed-sectoring common sector system distributes elevators throughout a building by segmenting the building into a number of static demand sectors. For the purposes of this project, the number of sectors is assumed to equal the number of elevators in the elevator group. Each sector consists of a number of contiguous floors and sectors do not overlap. Every elevator in the group is assigned to a sector. The size of each sector (*i.e.* the number of floors contained within the sector) is the ratio of the number of floors in the building to the number of elevators in the group. If this ratio is not integral, each sector is allocated a size of either the rounded-down value of the division result or the rounded-up value of the division result. The sum of the sector sizes should, however, equal the number of floors in the building. Since arrivals occur mainly at the ground floor during the up-peak traffic condition, priority is given to the ground floor by exclusively allocating it to a sector.

During the elevator allocation procedure, a landing call is typically answered by the elevator assigned to the sector in which the landing call is registered. Landing calls may, however, be answered by elevators outside the sector in which the landing call originates. In the latter case, the literature is not precise as to what procedure is to be followed to assign an elevator. If the sector in which the landing call originates is vacant, elevators assigned to sectors above or below that sector may be assigned to answer the landing call. It is, however, unclear to what

Algorithm 3.1: The nearest-car control algorithm

Input : Landing floor of a new landing call.
Output: Elevator assigned to the new landing call.

```
1  N ← number of floors in building −1;
2  maxFS ← 0;
3  minimumDistance ← ∞;
4  for every elevator E in the group which is not fully loaded do
5      d ← abs(E.currentfloor − landing floor);
6      if E is moving towards the landing floor in the same direction as required by the
       landing call then
7          FS ← N + 2 − d;
8      else if E is moving towards the landing floor in the opposite direction as required by
       the landing call then
9          FS ← N + 1 − d;
10     else if E is moving away from the landing floor then
11         FS ← 1;
12     else if E is idle then
13         FS ← N + 1 − d;
14     if FS > maxFS then
15         maxFS ← FS;
16         assignedElevator ← E;
17 if FS = maxFS for two or more elevators then
18     for every elevator E with FS = maxFS do
19         if E.d < minimumDistance then
20             assignedElevator ← E;
21             minimumDistance ← E.d;
22     if two or more elevators have FS = maxFS and d = minimumDistance then
23         assignedElevator ← first elevator to reach maxFS previously;
24         if no elevator has previously reached maxFS then
25             assignedElevator ← random elevator;
```

extent the elevators' current directions of travel and their distances from the landing call are taken into consideration during elevator allocation. For implementation purposes, a number of assumptions are therefore made in this project with respect to the elevator allocation procedure.

An elevator is assigned to a landing call in its sector only if it is idle or moving towards the landing floor and in the same direction as required by the landing call. If the elevator is not in one of the aforementioned states, or the sector is vacant, the algorithm proceeds to identify a suitable elevator nearby to answer the landing call. Alternating between sectors above and below this sector, the algorithm assigns the closest elevator that is either idle or moving towards the landing floor and in the same direction as that required by the landing call. If no suitable elevator is thus identified, the aforementioned step is repeated with the difference that the first elevator moving towards the landing floor (but in the opposite direction than that required by the landing call) is assigned to the landing call. If an elevator has still not been assigned at this point in the procedure, a random elevator in the group is assigned to the landing call.

A pseudo-code description of the fixed-sectoring common sector control algorithm for elevator assignment is given in Algorithm 3.2.

As discussed in §2.5.2, an elevator is de-assigned from a sector once it leaves its assigned sector or is fully loaded. Once a de-assigned elevator has responded to all of its pending landing calls and car calls, it is assigned to a vacant sector. If the sector in which the elevator last stopped is vacant, it is assigned to that particular sector and proceeds to wait for new calls at its current position. If not, the elevator is assigned to the nearest vacant sector and moves to the middle floor of the vacant sector where it awaits new landing calls.

As in the case of the nearest-car control algorithm, the number of passengers waiting at a floor is unknown and unserved passengers are assumed to re-register a landing call should the assigned elevator be filled to capacity upon their attempt at entering the elevator. Apart from the exception with respect to the ground floor's sector during the up-peak traffic condition, this control algorithm has a fixed implementation irrespective of the prevalent traffic condition.

The operation of the fixed-sectoring common sector control algorithm is demonstrated by means of a simple example. Consider the three-elevator system in a nine-storey building illustrated in Figure 3.2. The building is segmented into three sectors comprising three floors each. Sector 1 consists of the ground floor, floor 1 and floor 2. Sector 2 consists of floors 3, 4 and 5, and the third sector comprises the remaining floors. At a particular time instant the elevator positions and directions of movement are as shown. Two up landing calls are registered at the ground floor and the sixth floor, respectively.

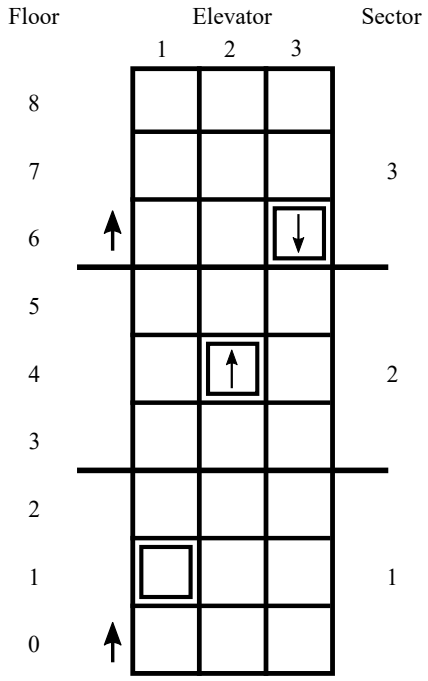


FIGURE 3.2: *Example of an elevator system operating under the fixed-sectoring common sector control algorithm.*

Algorithm 3.2: The fixed-sectoring common sector control algorithm

Input : Landing floor of a new landing call.
Output: Elevator assigned to the new landing call.

```
1  callSector ← sector in which the landing call is registered;
2  requiredDirection ← direction required by landing call;
3  if elevator assigned to callSector is present in callSector and not fully loaded then
4      E ← elevator assigned to callSector;
5      if E is idle then
6          assignedElevator ← E;
7      else if E is below landing call, moving up and requiredDirection is up then
8          assignedElevator ← E;
9      else if E is above landing call, moving down and requiredDirection is down then
10         assignedElevator ← E;
11 if no elevator is assigned to landing call then
12     sectorAbove ← callSector + 1;
13     sectorBelow ← callSector − 1;
14     for every sector S do
15         for every elevator E which is not fully loaded and present in sectorAbove do
16             if E is idle or moving down and requiredDirection is down then
17                 assignedElevator ← E;
18                 break;
19         for every elevator E which is not fully loaded and present in sectorBelow do
20             if E is idle or moving up and requiredDirection is up then
21                 assignedElevator ← E;
22                 break;
23         sectorAbove ← sectorAbove + 1;
24         sectorBelow ← sectorBelow − 1;
25 if no elevator is assigned to landing call then
26     sectorAbove ← callSector + 1;
27     sectorBelow ← callSector − 1;
28     for every sector S do
29         for every elevator E which is not fully loaded and present in sectorAbove do
30             if E is moving towards landing floor then
31                 assignedElevator ← E;
32                 break;
33         for every elevator E which is not fully loaded and present in sectorBelow do
34             if E is moving towards landing floor then
35                 assignedElevator ← E;
36                 break;
37         sectorAbove ← sectorAbove + 1;
38         sectorBelow ← sectorBelow − 1;
39 if no elevator is assigned to landing call then
40     assignedElevator ← random elevator which is not fully loaded;
```

Elevator 1 is idle, assigned to sector 1, and hence responds to the landing call registered at the ground floor. Elevator 2, assigned to sector 2, is travelling upwards in response to a car call. Elevator 3 is moving down in response to a car call and will be de-assigned from sector 3 once it reaches floor 5. Hence, elevator 3 is not best suited to answer the down landing call registered in sector 3. Elevator 2 instead responds to the landing call since it is in the sector directly below sector 3 and is travelling upwards. After responding to both the landing and car calls, elevator 2 (now de-assigned from sector 2) will be assigned to sector 3. Suppose the passenger waiting at the ground floor travels to the third floor. Elevator 1 will therefore not leave sector 1, but will become idle at floor 3. Suppose the current destination of elevator 3 is the ground floor. Once the particular passenger has alighted at the ground floor, the elevator will be assigned to the unoccupied sector 2. Elevator 3 will position itself at the fourth floor (middle of the sector) where it will await new landing calls.

3.4 The destination dispatch control algorithm

As discussed in §2.6, the destination dispatch control algorithm minimises a particular cost function when assigning elevators to landing calls. The cost function selected for implementation in this project is average passenger journey time. Elevator assignments are made with the objective of minimising the average journey time of all passengers in the system. Average passenger journey time is defined as the average time duration since registering a landing call until the assigned elevator begins to open its doors at the destination floor of a passenger [7]. It is envisaged that this particular cost function is the most appropriate for implementation under all three traffic conditions discussed in §2.2. Consider, for example, the up-peak traffic condition. Should the destination dispatch control algorithm instead attempt to minimise average passenger waiting time, landing calls may be assigned to elevators positioned at the ground floor or to those elevators closest to returning to the ground floor. Hence, passengers may not be grouped according to their destination floors and experience increased travel times. Similarly, minimising average passenger travel time may cause passengers to experience increased waiting times. The average passenger journey time cost function is preferred for implementation because it does not isolate either waiting or travel time during elevator allocation.

When a landing call is registered, the destination dispatch control algorithm considers each elevator in the system individually and evaluates the average passenger journey time, assuming that the new landing call is assigned to the particular elevator. The algorithm considers the waiting and travel times of the new landing call and all pending calls, as well as the time associated with each stop in respect of passenger transfer and the opening and closing of the elevator doors. As opposed to the nearest-car and fixed-sectoring common sector control algorithms, the destination dispatch control algorithm keeps record of the number of passengers waiting to enter at each landing floor. As such, the algorithm assigns an elevator to a passenger that will have adequate capacity once it stops at the particular landing floor.

Consider the case where a new landing call is to be allocated to an elevator. For an elevator system comprising L elevators, with elevator I responding to $N(I)$ pending calls and the new landing call being assigned to elevator K, Barney and Al-Sharif [7] proposed that the average journey time should be estimated by the expression

$$Z = \frac{X(K) + \sum_{I=1, I\neq K}^{L} Y(I)}{1 + \sum_{I=1}^{L} N(I)},$$

where $X(K)$ is the new accumulated journey time for elevator K and $Y(I)$ is the accumulated journey time for $N(I)$ calls (the accumulated journey time for all elevators, excluding elevator

K). The elevator K, yielding the minimum average journey time value, is assigned to the new landing call.

A function for calculating the accumulated journey time of a specific elevator is presented in Algorithm 3.3. The function receives an elevator, together with all of its pending landing calls and car calls, as input arguments. From the time instant that the function is called, the function simulates the elevator's current trip until it opens its doors at its final destination floor. The accumulated passenger journey time is calculated for all pending landing and car calls. The passenger's arrival time in the algorithm is the time at which the particular passenger registers his or her landing call.

During the elevator allocation procedure, the destination dispatch control algorithm uses the function in Algorithm 3.3 to calculate the accumulated passenger journey time for each elevator in the system. A pseudo-code description of the destination dispatch control algorithm is finally given in Algorithm 3.4. The algorithm temporarily assigns the new landing call to an Elevator E and proceeds to evaluate the accumulatedJourneyTime function for all elevators in the system. The final elevator assignment is made only after all the elevators in the system have been evaluated.

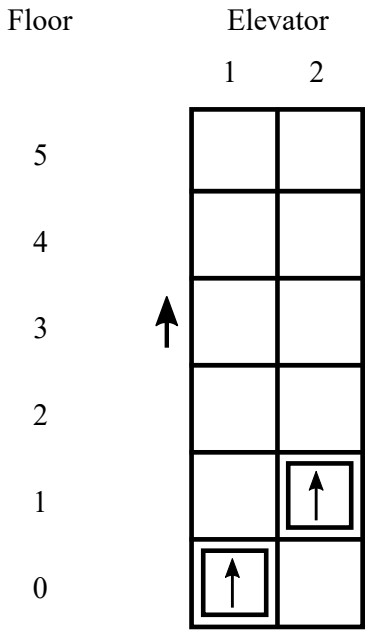


FIGURE 3.3: *Example of an elevator system operating under the destination dispatch control algorithm.*

The working of the destination dispatch control algorithm (minimising average journey time) is again illustrated by means of an example. Consider a six-storey building with two elevators as shown in Figure 3.3. At the time instant shown, elevator 1 — transporting three passengers — is travelling towards floor 2 where all three passengers will alight. Elevator 2 is travelling towards floor 5 where its only passenger will alight. A single landing call is registered at the third floor where a passenger requests transportation to the fifth floor. Barring the landing call, there are four pending calls in the system.

In order to determine which elevator is assigned to the landing call, the algorithm evaluates the average passenger journey time for two scenarios — resulting in either elevator 1 or elevator 2 being assigned to the landing call. Consider the first scenario above. Suppose the new

accumulated journey time for elevator 1 equates to 48 seconds. The total accumulated journey time for elevator 2 is 17 seconds. The average passenger journey time for the system is therefore $(48 + 17)/(1 + 4) = 13$ seconds. Suppose, in the second scenario, that the accumulated journey time for elevator 1 is 30 seconds. For elevator 2, suppose furthermore that the new accumulated journey time is 28 seconds. The average passenger journey time for the system in this case is therefore $(30 + 28)/(1 + 4) = 11.6$ seconds. Elevator 2 is consequently assigned to the landing call since such an assignment yields the lowest average passenger journey time.

Algorithm 3.3: accumulatedJourneyTime (E)

Input : Elevator E and its pending landing calls and car calls.
Output: Accumulated passenger journey time.

```
1  time ← logical time;
2  accumulatedJourneyTime ← 0;
3  E ← elevator received as argument;
4  while E has not stopped at its final destination floor do
5      for every floor F to which E travels do
6          time ← time + time associated with travelling past F;
7          if landing call or car call at F then
8              doorsOpeningTime ← time;
9              time ← time + time associated with elevator doors opening and closing;
10             time ← time + time associated with all passengers either entering or exiting
               elevator;
11             for every passenger P alighting at F do
12                 accumulatedJourneyTime ← accumulatedJourneyTime +
                   (doorsOpeningTime − P.arrivalTime);
13 return accumulatedJourneyTime;
```

Algorithm 3.4: The destination dispatch control algorithm

Input : A new landing call.
Output: The elevator assigned to the new landing call.

```
1  minimumAJT ← ∞;
2  Y ← 0;
3  N ← 0;
4  for every elevator K in the group do
5      N ← K.numberOfPendingCalls;
6      assignedElevator ← K;
7      X ← accumulatedJourneyTime(K) ;
8      for every elevator M in the group, excluding K do
9          N ← N + M.numberOfPendingCalls;
10         Y ← Y + accumulatedJourneyTime(M);
11     Z ← (X + Y)/(1 + N)
12     assignedElevator ← null;
13     if Z < minimumAJT then
14         minimumAJT ← Z;
15         assignedElevator ← K;
```

3.5 Chapter summary

Careful descriptions were provided in this chapter of the nearest-car control algorithm, the fixed-sectoring common sector control algorithm, as well as the destination dispatch control algorithm. These three control algorithms are analysed later in this project. A motivation for selecting these elevator control algorithms for implementation in this project was also provided. This was followed by pseudo-code descriptions of these control algorithms. This is considered a valuable contribution, because the author was unable to find such precise algorithmic descriptions in the literature. The discussion of each control algorithm was finally complemented with a simple example aimed at improving the reader's understanding of the working of the algorithms.

CHAPTER 4

The simulation model

Contents

4.1 The AnyLogic simulation modelling environment 33
4.2 General description of the simulation model 34
4.3 The dynamics of an elevator passenger 36
4.4 The dynamics of an elevator 37
4.5 Elevator traffic implementation 38
4.6 The graphical user interface 39
4.7 Simulation model verification and validation 41
4.8 Chapter summary 43

The purpose of this chapter is to provide the reader with an overview of the architecture of the simulation model developed for use as an elevator control algorithm test bed in this project. The software suite employed in which to implement the simulation model is introduced in §4.1. A general description of the simulation model and its relevant underlying assumptions is next provided in §4.2. This is followed in §4.3 and §4.4 by detailed descriptions of how passenger and elevator behaviour are modelled, respectively. A detailed discussion on the implementation of the three prevailing elevator traffic conditions analysed in this study is provided in §4.5, and this is followed by a description in §4.6 of the simulation model's graphical user interface, containing its visualisation capability. The techniques employed to verify and validate the simulation model are finally reviewed in §4.7, after which the chapter closes with a brief summary.

4.1 The AnyLogic simulation modelling environment

The computer simulation model employed as algorithmic test bed in this project was designed and developed in the AnyLogic University 7.3.1 software suite. AnyLogic is a multi-method simulation modelling tool which supports three major modelling paradigms, namely discrete-event modelling, agent-based modelling and system dynamics modelling. The software was selected for use in this project because of its ability to extend simulation models using Java code, as well as its rich visualisation and animation capabilities. An agent-based modelling approach was found to be most suitable to model the behaviour of elevator passengers and the elevators themselves.

Several components of the AnyLogic software suite were used to translate the dynamics and behaviour of an elevator system into a computer simulation environment. The first of these

components is an *object class* which represents a distinct agent, or entity, in the simulation model. *Agents* are viewed as the building blocks of the simulation model and may represent physical or abstract entities. Passengers, elevators, floors and floor sectors are all instances of object classes employed in this simulation model. Variables and parameters may be defined for each agent. A *variable* is a value that changes over time and may be used to model the characteristics of an agent. Another variable type found in AnyLogic is a collection variable. A *collection* represents a group of objects that are typically associated with a single unit. Collections may be employed in respect of a set of agents, such as a group of agents waiting in a queue. A *parameter*, on the other hand, represents the characteristics and behaviour of an agent and only changes when the behaviour of the agent changes. A *function* is another component in the AnyLogic modelling environment and it returns the value of a particular expression every time it is called in the simulation model. Functions are useful in cases where code portions have to be re-used several times in different places in the simulation model. An *event* is used to schedule a particular action in the simulation model by means of delays or timeouts. Furthermore, a *data set* component is used to store output data which may be exported to Microsoft Excel. Finally, a *connectivity* component is used to establish a link between the simulation model and a particular Microsoft Excel workbook which may be used for data import and/or export.

In order to model the behaviour, or life cycle, of the elevator passengers and elevators in an elevator system, the *state chart* construct is used. State charts describe time-driven and event-driven behaviour in which each state represents a distinct point in the agent's life cycle. States are connected by *transitions* which are triggered by user-defined *conditions* such as timeouts, messages received or logical (boolean) conditions. When a transition is triggered, an agent moves from its current state to a new state. The dynamics and behaviour of passengers and elevators are modelled by means of the state chart construct in the simulation model of this project.

4.2 General description of the simulation model

In the AnyLogic simulation modelling environment, the simulation output is typically presented in the `Main` tab. In the simulation environment, however, `Main` is also an object class, but functions differently from other object classes. Whereas the contents of other object classes are typically not viewed by the simulation operator, the contents of the `Main` object class are visible during simulation execution. Parameters, variables, functions and events declared within `Main` are 'globally' declared and typically affect all other object classes. A selection of the globally declared parameters relevant to the simulation model of this project are shown in Figure 4.1.

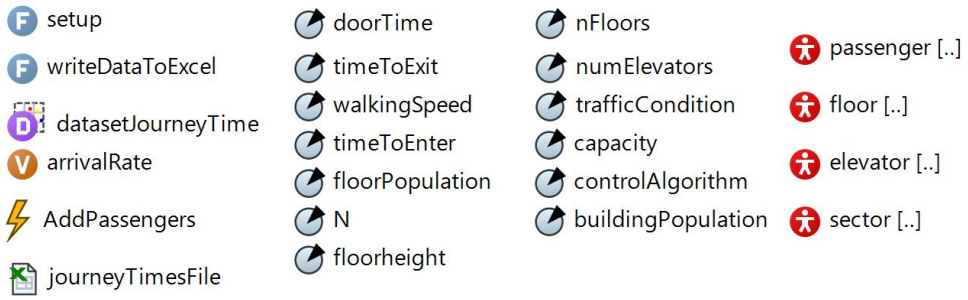


FIGURE 4.1: *A selection of the global parameters declared in the* `Main` *object class.*

As shown in Figure 4.1, four object classes (excluding `Main`) are declared in the simulation model of this project. These are the `passenger`, `floor`, `elevator` and `sector` object classes. The state charts constructed for both the `passenger` and `elevator` object classes dictate the

behaviour of these agents in the model. The `floor` object class is used to store data pertaining to the queues formed at each floor. The `Queue` collection, in the `floor` object class, stores the `passenger` agents waiting in the queue at the particular floor associated with the `floor` agent at any given time. The `sector` object class, on the other hand, stores a list of floors associated with each sector, as well as the elevator assigned to each sector, during the operation of the fixed-sectoring common sector algorithm.

Before simulation execution, the simulation operator may specify a number of parameter values for the particular simulation run to be performed. The number of floors in the building is specified as the `nFloors` parameter and the number of elevators in the group is stored in the `numElevators` parameter. The `floorPopulation` parameter defines the number of people working on each floor in a particular building and its value, stored in the `floorPopulation` parameter, is assumed to be identical for every floor, except for the ground floor. The assumption is made that a building's occupants only arrive at, and exit the building at, the ground floor. Therefore, the ground floor has a floor population of zero and the `buildingPopulation` parameter adopts a value of (`nFloors` $-$ 1) $\times$ `floorPopulation`. The population size of the `floor`, `elevator` and `passenger` agents equal the `nFloors`, `numElevators` and `buildingPopulation` parameter values, respectively. Furthermore, the prevalent elevator traffic condition is defined in the `trafficCondition` parameter, while the `controlAlgorithm` parameter specifies the particular control algorithm to be employed.

The remaining parameters defined in `Main` have fixed values which are not specified by the simulation operator. The walking speed of people in the simulation is assumed to be a constant value of 1.4 m/s [9] and is defined as such in the `walkingSpeed` parameter. The floor height is fixed at 3 meters per storey [7] and the value is stored in the `floorheight` parameter. The `doorTime` parameter is the time (in seconds) associated with the elevator doors either opening or closing. A value of 2 seconds is assumed for this parameter [7]. Two further parameters, `timeToEnter` and `timeToExit`, store the time value (in seconds) associated with a single passenger either entering or exiting an elevator. For a passenger standing in front of an elevator, it is assumed that it takes him or her a total of 1.2 seconds to step inside the elevator [7]. Similarly, it is assumed that a single passenger standing inside an elevator, in front of the doors, will spend 1.2 seconds to step outside the elevator [7]. The parameter `N` is used by the nearest-car algorithm and adopts a value equal to `nFloors` $-$ 1. The effect of elevator capacity on elevator system performance is not studied in this project. Hence the maximum elevator capacity is assumed to be 21 people (based on an elevator with a rated load of 2000 kg [7]) and is defined as such in the `capacity` parameter. Furthermore, the `datasetJourneyTime` data set is used to store the journey time of each `passenger` agent. Upon termination of a simulation run, the `writeDataToExcel` function is called, which writes the data stored in `datasetJourneyTime` to the Microsoft Excel workbook captured by the `journeyTimesFile` connectivity component.

Upon initiating a simulation run, the `setup` function is called which executes the Java code associated with the *graphical user interface* (GUI). For the fixed-sectoring common sector algorithm, the sector sizes are also defined in the `setup` function. During a simulation run, passenger arrivals are governed by the timeout-triggered `AddPassengers` event. The event is triggered at a rate specified by the Poisson probability distribution whose λ parameter value is stored within the `arrivalRate` variable.

Several assumptions were made with respect to the development of the simulation model. These assumptions are as follows:

- People travelling between floors always make use of elevator transport, even if they are only travelling one floor.

- A passenger does not change his or her destination floor after registering a landing call.
- Every passenger registers a landing call.
- A passenger always enters his or her assigned elevator.
- A passenger re-registers his or her landing call after failing to enter the assigned elevator, should the elevator be filled to capacity before the passenger can enter.
- Passenger arrivals are distributed according to a Poisson probability distribution.
- People walk at a constant speed.
- A passenger's journey time does not explicitly account for the time spent walking to the assigned elevator.
- Passengers enter their assigned elevator according to the sequence in which their landing calls were registered (first come, first served).
- Passengers enter an elevator in single file.
- A passenger enters or exits an elevator only once the doors are fully open.

4.3 The dynamics of an elevator passenger

The `passenger` object class represents a person making use of the elevator system modelled. The state chart used to model the behaviour and dynamics of a `passenger` agent is shown in Figure 4.2. A selection of the parameters defined in the `passenger` object class is shown in Figure 4.3. Once a `passenger` agent is generated by virtue of the `AddPassengers` event, it is immediately placed in the `Locate` state. The agent's landing and destination floors are determined upon agent generation and stored as `landingFloor` and `destFloor` parameter values, respectively. The `pasDirection` parameter specifies the agent's required direction of travel — either up or down.

The passenger immediately progresses to the `MoveToLobby` state (from the `Locate` state) and proceeds to walk towards the pushbutton panel located on the particular landing floor. Upon arrival at the pushbutton panel, the passenger registers his or her landing call and progresses to the `Wait` state. During the transition from the `MoveToLobby` state to the `Wait` state, the `assignElevator` function is called. This function determines the elevator assigned to the passenger. The index of the passenger's assigned elevator is stored in the `allocatedElev` variable. The passenger's arrival time is the model time (in seconds) recorded when the passenger registers his or her landing call; this value is stored in the `arrivalTime` parameter.

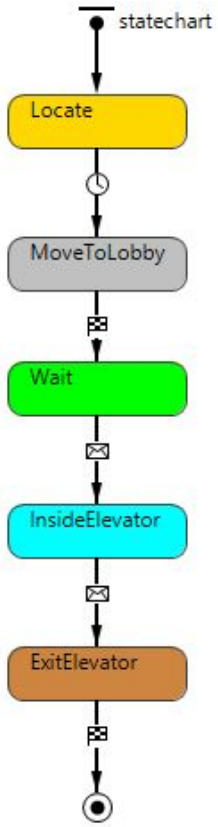


FIGURE 4.2: *The* `passenger` *object class state chart.*

Once the passenger's assigned elevator opens its doors at the landing floor, the passenger receives a message ('`Enter elevator`'), which triggers the transition to the `InsideElevator` state. The

model time (in seconds) at that time instant is stored in the `enterTime` parameter. If the passenger's assigned elevator is already fully loaded before he or she can enter, re-registration of the landing call is executed by means of the `reregisterLandingCall` function. Whilst travelling in an elevator, the passenger remains in the `InsideElevator` state. Only when the elevator stops at the destination floor does the passenger receive a message ('`Exit elevator`'), which triggers the transition to the `ExitElevator` state. The passenger proceeds to exit the elevator and walk away. The `exitTime` parameter stores the value of the model time (in seconds) when the passenger's assigned elevator starts to open its doors at the destination floor.

FIGURE 4.3: *A selection of the parameters declared in the* `passenger` *object class.*

4.4 The dynamics of an elevator

The state chart construct, shown in Figure 4.4, defines the behaviour and dynamics of an elevator in the simulation model. Once an `elevator` agent is generated, it is immediately placed in the `Idle` state. An idle elevator has no pending landing calls or car calls, and remains idle at its current position. An elevator exits the `Idle` state by virtue of either one of two message-triggered transitions. If an elevator must respond to a landing call registered at its current floor, it receives a message ('`Open Doors`'), which triggers the transition to the `DoorsOpen` state. Alternatively, the elevator receives a message ('`Move`') if it must respond to a landing call at a different floor, which implies that the elevator must move towards that particular landing floor. Hence the transition places the elevator in the `MotorStart` state which represents the starting of the elevator motor before movement can commence. The elevator spends 0.5 seconds in this state before it progresses to the `Moving` state.

In the `Moving` state, Java code for moving the elevator over a one-floor height distance is executed. Upon reaching the next floor, a decision is made as to whether the elevator should stop or continue moving. If the elevator's current floor is not its destination floor, it returns to the `Moving` state and continues moving. It is important to note that this logic does not cause the elevator to stop dead after travelling one floor. The time taken by the simulation to evaluate whether the elevator should continue its movement is negligibly small such that the movement appears to be continuous. If the elevator's current floor equals the destination floor, the elevator stops and enters the `DoorsOpen` state. In the case where elevators are sent to a vacant sector by the fixed-sectoring common sector algorithm, the elevator advances to the `Idle` state (from the `Moving` state), after reaching the specified floor in the vacant sector.

The `DoorsOpen` and `DoorsClose` states model the action associated with the elevator doors opening and closing, respectively. The timeout transition which signals progression from either one of these states has the value (in seconds) specified by the `doorTime` parameter. Once the elevator doors are completely open, the action of passengers exiting and/or entering the elevator is modelled. The Java code regulating the entering and exiting of passengers is contained in the `LoadUnload` state. The time spent in the `LoadUnload` state is specified by a timeout-transition. The timeout value of this transition is the actual time associated with all the passengers exiting and entering the elevator while the doors remain open. After the elevator has exited the `LoadUnload` state, the closing of the doors are modelled by the `DoorsClose` state. Once the doors are completely closed and the elevator has no pending calls to respond to, it returns to the

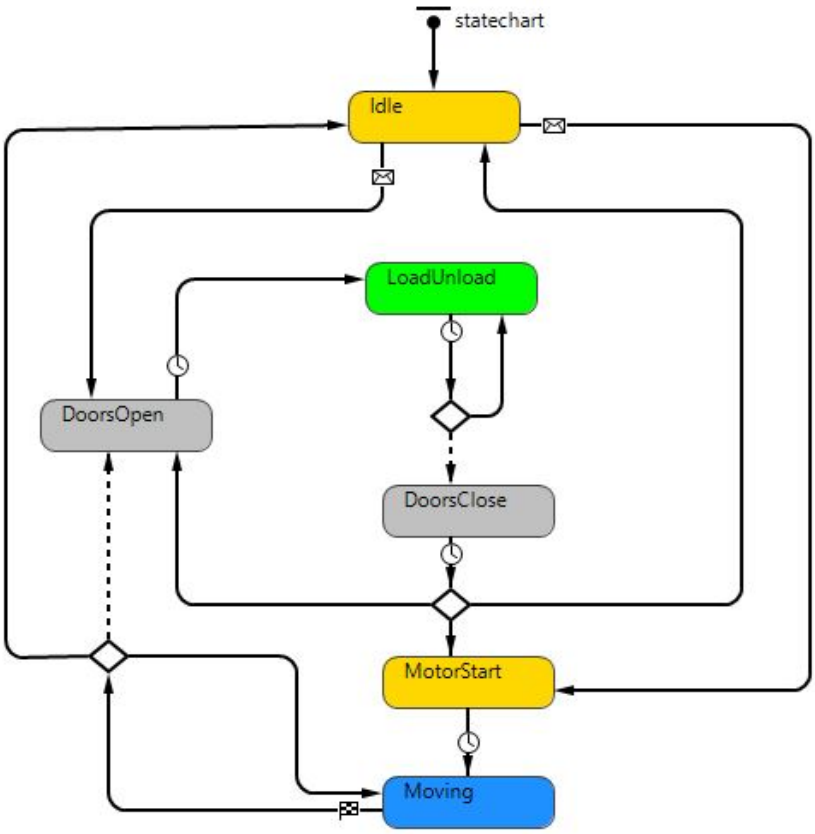


FIGURE 4.4: *The* `elevator` *object class state chart.*

`Idle` state. In the case where a landing call is registered at the elevator's current floor whilst it is in the `DoorsClose` state, the elevator progresses to the `DoorsOpen` state. Alternatively, if the elevator has to move to a vacant sector or respond to pending calls, it enters the `MotorStart` state after which it progresses to the `Moving` state.

An elevator may accelerate from an initial speed of 1 m/s to a maximum speed of 3 m/s following a step-wise profile. Starting from a stationary position, an elevator travels at an initial speed of 1 m/s which increases by increments of 1 m/s for every subsequent floor distance travelled. An elevator will therefore travel at 2 m/s once it has travelled one floor distance and will reach the maximum speed of 3 m/s upon having travelled yet another single floor distance. Similarly, an elevator decelerates in this step-wise fashion until it reaches the destination floor at a speed of 1 m/s. The speed of an elevator is regulated in such a manner that it always reaches the destination floor at a speed of 1 m/s. Hence a moving elevator may not always accelerate fully, in order to comply with this aforementioned rule.

4.5 Elevator traffic implementation

The implementation of the three elevator traffic conditions investigated in this project is partially based on the work of Barney [7] who constructed so-called *templates* which define particular elevator traffic demand profiles to be considered during elevator system design. These templates define the passenger arrival rates typically observed in a one-hour time period of the prevailing elevator traffic condition's existence. The arrival rates are expressed as percentages of the building population arriving within single five-minute periods. In the simulation model employed

in this project, the arrival rate (as a percentage) is converted to an average rate of arrival of passengers per minute. This converted arrival rate value is denoted by λ, the parameter of the Poisson probability distribution. The arrival rates associated with the respective prevailing elevator traffic conditions are presented in Table 4.1. Note how the up-peak traffic condition's arrival rate gradually increases over the first 40 minutes before it starts to decrease again. This is representative of how the bulk of people arrive at an office block for work before the official start of the work day, which is assumed to be in the region of the 40-minute time interval. Similarly, the dramatic increase in arrival rate after 10 minutes of observed down-peak traffic, is representative of the majority of people leaving at the end of the official work day. Some people leave earlier than the official time, whilst others leave much later. During one hour of observed up-peak and down-peak traffic respectively, 80% of the building population is typically expected to make use of elevator transport [7]. As for the random inter-floor traffic condition, 36% of the building population typically request elevator service over the course of an hour.

Time period (minutes)	Up-peak	Down-peak	Random inter-floor (up landing calls)	Random inter-floor (down landing calls)
0 – 5	3	2.8	1.3	0.8
5 – 10	3.5	5.9	0.8	1
10 – 15	4.5	22.5	1.3	0.8
15 – 20	6	14.4	1.3	0.8
20 – 25	7.5	8.9	1.4	1.8
25 – 30	9.5	5.8	1.8	1.3
30 – 35	12	4.1	1.6	1.6
35 – 40	15	3.3	1.3	1.6
40 – 45	8	3	2.1	1.9
45 – 50	4.5	3	1.8	2.3
50 – 55	3.5	2.9	1.5	2.5
55 – 60	3	3.1	2	2

TABLE 4.1: *Passenger arrival rates associated with a particular traffic condition, expressed as a percentage of the building population arriving in a five-minute period [7].*

In this project it is assumed that, during the up-peak traffic condition, 95% of all arrivals occur at the ground floor, requesting elevator service to the upper floors. The remaining 5% of arrivals are randomly distributed throughout the building with passengers requesting either up or down service. During the down-peak condition, 95% of all arrivals are similarly assumed to be passengers requesting elevator service to the ground floor. Again, the remaining 5% of arrivals are randomly distributed throughout the building, requesting either up or down service. In the simulation model employed in this project, passenger arrivals are governed by the aforementioned elevator traffic demand profiles. Since these demand profiles are based on actual observations spanning the full one-hour time duration, the notion of a simulation warm-up period may be disregarded. The passenger arrival process is terminated once one hour of simulated time has elapsed, and the simulation run continues until all passengers are served.

4.6 The graphical user interface

The *GUI* is a type of interface that allows a simulation operator or analyst to interact with a simulation model in a visual manner. Emphasis is usually placed on designing a *GUI* that is both informative and simple to use. The *GUI* designed in the simulation model of this project

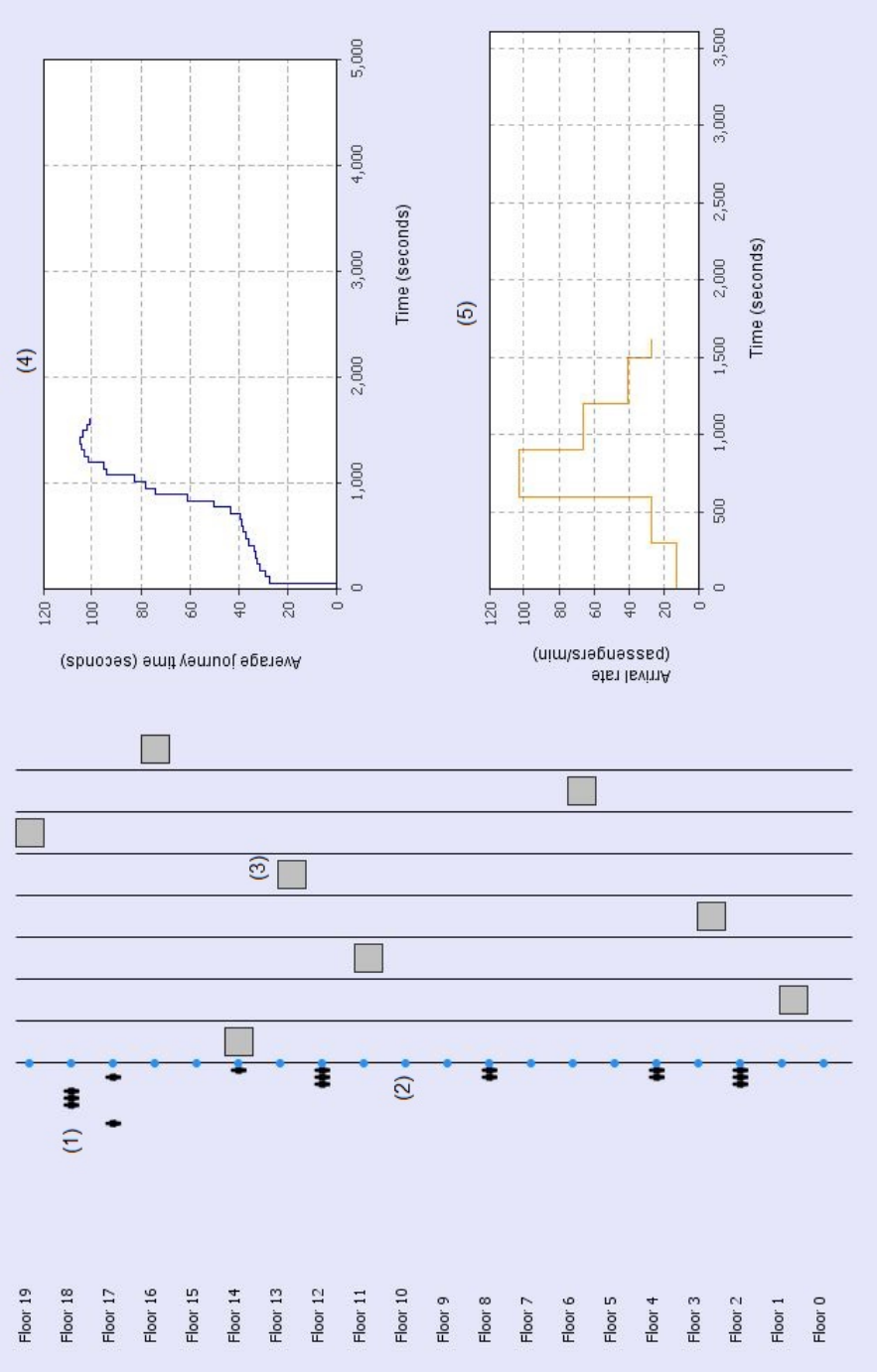


FIGURE 4.5: *Screenshot of the simulation model GUI.*

may be used by the simulation operator first to define parameter values before an experiment is initiated and secondly, to interact with the model during execution.

In AnyLogic, the *GUI* is typically designed in the `Main` tab. A screenshot of the *GUI* for the simulation model of an elevator system is shown in Figure 4.5. The *GUI* is a representation of an office building with a pre-specified number of floors, a pre-specified number of elevators in the building and a population of passengers moving within the building. In order to visually represent the real speed of both the moving people and elevators during simulation execution, the chosen model time unit is seconds. Another motivation for using this time unit is that the

journey time of a passenger and the RTT are typically measured in seconds. Several important features of the *GUI* are as follows:

1. **Passengers:** The people present within a building are represented by two-dimensional human figures, denoted (1) in Figure 4.5.
2. **Pushbutton panels:** Blue nodes, denoted (2) in Figure 4.5, represent the pushbutton panels located in the lobby. Once a passenger arrives at the blue node, he or she registers a landing call.
3. **Elevators:** Each elevator is represented by a grey-coloured square shape, denoted (3) in Figure 4.5.
4. **Average journey time tracking:** A chart, denoted (4) in Figure 4.5, is used to show the average journey time of all passengers in the system as a simulation run progresses.
5. **Passenger arrival rate tracking:** The passenger arrival rate is presented in another chart, denoted (5) in Figure 4.5, as a simulation run progresses.

4.7 Simulation model verification and validation

The processes of model verification and validation were followed iteratively throughout the design and development of the computer simulation model described in the preceding sections. A selection of the verification techniques discussed in §2.9 were used to enhance the credibility of the simulation model. The assignment of elevators to passengers is influenced by, amongst others, the number of passengers in the system and their respective landing and destination floors, as well as the current state of the elevators. As such, many logically possible scenarios may occur in the elevator system. Flowcharts of these possible actions were constructed and evaluated against the model logic upon model execution. After each completed simulation run, the values of the input parameters were printed to ensure that these values remained unchanged during runtime. Naturally, the average passenger journey time of a particular experiment is expected to be larger than that of an identical experiment, but with fewer elevators. Varying the input parameter values in view of the aforementioned expectancy, the model output was evaluated for credibility.

In the AnyLogic modelling environment, emphasis is placed on using descriptive names for all object classes, parameters, variables, functions and events. The naming of these elements is accompanied by coherent descriptions of their purpose in the simulation model. In order to ensure clarity and enhance understanding of the model logic, comments were extensively used throughout the programming code. Finally, the use of animation in the simulation model proved to be of considerable assistance in the identification of illogical actions.

As for the validation process, face validation was the dominant technique employed to ensure that the correct model was built. Observations of actual elevator systems were compared with the operation of the simulation model in order to establish whether the latter accurately represents reality. These observations were used to refine the working of the simulation model in an iterative fashion. The focus of these observations was to validate the behaviour of elevators, as well as that of the passengers arriving in the elevator lobby, registering landing calls, waiting, entering and ultimately leaving an elevator. The structural assumptions made during the simulation model design process include the assumption of fixed values for the time associated with elevator doors opening and closing, passenger transfer, as well as the maximum moving speed of elevators.

These time and speed values were validated by comparison with known values for actual elevator systems. Furthermore, the modelled passenger arrival process is comparable to real-world arrival processes since it is based on known elevator traffic demand profiles, as discussed in §4.5. Finally, similar to one of the aforementioned verification techniques, a sensitivity analysis was performed to evaluate whether the simulation model behaved in the expected manner when one or more input parameters were changed.

A further technique employed to validate the simulation model was to compare analytically calculated RTT values with RTT values observed in the simulation model. The expressions described in §2.7 were employed to calculate values for the expected number of stops S, the expected highest reversal floor H, as well as the RTT during a single round trip, given that there are P passengers in an elevator. These expressions are applicable to the up-peak traffic condition only.

A simulation experiment was performed, modelling a 20-storey building with a floor population size of 30, and an elevator group comprising three elevators. In view of the RTT expression, the time required by an elevator to travel past two adjacent floors at a rated speed of 1 m/s, the value of the parameter t_v was 3 seconds. The expected time associated with a single elevator stop t_s was taken as 4.5 seconds. Finally, the passenger transfer time t_p was assumed to be 1.2 seconds. Passenger arrivals were modelled over a one-hour time period during up-peak traffic conditions, and the simulation run terminated once all the passengers were served. During this simulation run, the P, S, H and RTT values were observed for every round trip of the respective elevators.

Ten round trip observations were chosen randomly for comparison with the analytically obtained RTT values in order to validate elevator behaviour in the simulation model. These observed values and the corresponding computed values (for the same number of passengers) are shown in Table 4.2. The observed and analytically computed RTT values seem to be comparable. Only round trips 2, 3 and 10 yielded an absolute difference in respect of the RTT of more than 10 seconds. The most significant difference — 35 seconds — was observed during round trip 3, where the observed H and S values were smaller than that obtained analytically, hence the smaller observed RTT. Owing to the generally comparable RTT values, the observed and analytical service rates μ are comparable as well.

		Observed			Analytical		
No	P	H	S	RTT (seconds)	H	S	RTT (seconds)
1	4	16	3	123.6	15.7	3.7	124.8
2	6	15	4	126.9	16.8	5.3	143.1
3	7	12	5	115.8	17.1	6	150.8
4	20	18	12	214.5	18.5	12.6	220.0
5	14	18	10	191.1	18.2	10.1	192.3
6	8	15	8	152.7	17.4	6.7	157.8
7	15	19	11	204.0	18.2	10.6	197.5
8	17	17	9	199.7	18.4	11.4	207.0
9	20	19	12	220.5	18.5	12.6	220.0
10	21	19	15	236.4	18.5	12.9	224.2

TABLE 4.2: *Comparison of the observed and analytically calculated RTT values for the same number of passengers in an elevator.*

4.8 Chapter summary

Comprehensive descriptions were provided in this chapter of the simulation model designed for use as an elevator system performance test bed in this project. The software suite selected for use was introduced and its selection was briefly motivated. In order to provide a general understanding of how the simulation model was constructed, important features of the simulation model and key underlying assumptions made during its development were also described. The discussions pertaining to the dynamics of elevators and elevator passengers were complemented by visual representations of the respective state charts. Furthermore, the manner in which the three prevailing elevator traffic conditions were implemented in the simulation model of this project, was discussed. In order to provide insight into the visual aspect of the simulation model, a graphic and accompanying description of the *GUI* were included. The simulation model verification and validation techniques employed in this project were finally discussed.

CHAPTER 5

Experimental design

Contents

5.1 Key performance indicators 45
5.2 Sensitivity analysis 46
5.3 Statistical analysis of control algorithm effectiveness 47
5.4 Chapter summary 48

The purpose of this chapter is to present the reader with a description of the experimental process followed to investigate the research hypothesis of this project. The key performance indicators analysed in this study are introduced in §5.1. The experimental design, conducted in the form of a sensitivity analysis, is next described in §5.2. This is followed in §5.3 by an explanation of the statistical performance analysis carried out as part of the sensitivity analysis. The chapter finally closes in §5.4 with a brief summary of the chapter contents.

5.1 Key performance indicators

Two *key performance indicators* (KPIs) are employed to measure the effectiveness of the elevator control algorithms considered in this project. These KPIs are the average passenger journey time, as well as the maximum passenger journey time. An elevator passenger's journey starts at the registration of a landing call and terminates at his or her arrival at the destination floor. Evaluating control algorithm effectiveness in respect of journey time aptly reflects the ultimate goal of a passenger to reach his or her destination floor in the minimum amount of time. Although the average journey time provides a good estimate of the expected journey time, the maximum journey time may not be neglected when the effectiveness of the elevator control algorithms is evaluated. A control algorithm yielding, for example, a small average passenger journey time may be classified as an effective control algorithm. If the same algorithm, however, returns a comparatively large maximum journey time, the control algorithm may lose its credibility as an effective control algorithm. In order to soundly evaluate the effectiveness of the three elevator control algorithms analysed in this project, they are tested under identical passenger travel conditions with respect to the aforementioned two KPIs.

5.2 Sensitivity analysis

The two KPIs of the previous section are the dependent variables considered in the elevator control algorithm effectiveness comparison analysis carried out in this project. The independent variables of this comparison analysis are the number of floors in a high-rise building, the number of elevators in an elevator group, the different passenger arrival distributions and the elevator control algorithms implemented. More specifically, the rate of passenger arrivals is determined by changing the size of the floor populations. The effect of the aforementioned factors on passenger journey time is investigated by means of a sensitivity analysis. The purpose of such an analysis is to determine how different values of the independent variables influence the dependent variables (the KPIs of §5.1). Only one independent variable value is varied at a time, whilst the values of the remaining independent variables remain fixed. Hence, a sensitivity analysis may provide valuable insight as to which independent variable(s) impact journey time significantly.

In this project, three values are considered for the respective independent variables. These values are presented in Table 5.1. High-rise buildings with 20, 40 or 60 storeys are considered. Floor population sizes of 60, 80 or 100 are furthermore explored. Finally, the number of elevators in an elevator group adopts values of 4, 8 or 12, respectively. The so-called *base case* of the sensitivity analysis is a scenario in which the independent variables adopt the respective median values of the value ranges declared in Table 5.1. Therefore, the base case of this analysis, is a 40-storey building, with a floor population size of 80 people, comprising 8 elevators in its elevator group. The sensitivity analysis explores how the dependent variable value deviates from that of the base case when an independent variable's value is changed.

Number of floors	Floor population	Number of elevators
20	60	4
40	80	8
60	100	12

TABLE 5.1: *The independent variables and their respective values adopted in the sensitivity analysis.*

For each of the nine independent variable combinations in Table 5.1, the mean and maximum journey times of passengers are observed under the up-peak, down-peak and random inter-floor traffic conditions. In the base case, for example, all three aforementioned elevator traffic conditions are considered for a building comprising 40 floors, a floor population of 80 people and 8 elevators in the elevator group. The nearest-car, fixed-sectoring and destination dispatch control algorithms are implemented during each of the observed elevator traffic conditions.

The first set of experiments performed in this project is aimed at investigating the effect of the number of elevators in an elevator group on elevator control algorithm effectiveness. The number of elevators is varied between 4, 8 and 12, whilst the number of floors in the building and floor population size remain fixed at 40 and 80, respectively. Subsequently, three combinations of these independent variables are studied. Since three elevator traffic conditions are observed for each combination and three elevator control algorithms are studied for each traffic condition, a total of 27 experiments are performed. This experimental structure is repeated for the cases where the number of floors and the floor population size are varied, respectively, instead of the number of elevators.

In order to evaluate the effect of the number of floors on elevator control algorithm effectiveness, the number of elevators and floor population size remain fixed at 8 and 80, respectively. Again, three combinations of the independent variables arise as the number of floors is varied between 20, 40 and 60, respectively. This second set of experiments therefore also comprises 27 experiments

in total. Similarly, another 27 experiments are performed when the impact of the floor population size is studied. Whilst the number of elevators and number of floors are fixed at values of 8 and 40, respectively, the floor population adopts values of 60, 80 or 100. In total, 81 experiments are therefore performed in order to conduct the sensitivity analysis.

Each of the 81 experiments performed in this sensitivity analysis comprises 15 replications. Each replication effectively represents a unique work day. For each distinct combination of number of floors in a building, floor population size and prevailing elevator traffic condition studied in this analysis, 15 unique passenger arrival data sets are hence generated. In order to ensure that the three elevator control algorithms are implemented under identical conditions, these passenger arrival data sets are employed to produce matched samples of output data for the various simulation runs.

As discussed in §4.5, a replication run only terminates once all passengers in the system have been served. Therefore, the journey times of all the passengers who arrived during the modelled one-hour time period are observed. Each experiment subsequently produces 15 mean passenger journey time values, as well as 15 maximum passenger journey time values. The respective means of these two sets of 15 values are captured as the average journey time and maximum journey KPIs, respectively. These KPIs are employed in statistical tests to ultimately measure the relative effectiveness of the destination dispatch control algorithm when compared with other elevator control algorithms.

5.3 Statistical analysis of control algorithm effectiveness

Since the elevator system modelled in this project contains several stochastic elements, methods from the realm of statistical inference are employed to adjudicate the relative effectiveness of the three elevator control algorithms analysed. In this project, no assumptions pertaining to the distribution of the journey time data obtained, are made. Hence, a selection of *non-parametric tests* (discussed in §2.10) are employed in the form of a hypothesis testing approach in order to draw conclusions about the comparative performance of the respective elevator control algorithms. More specifically, the *Friedman test* is used to investigate the null hypothesis that the population medians in a data set containing two or more samples are equal. Since the Friedman test does not identify which samples differ significantly — in the case where the null hypothesis is rejected — the *Nemenyi post hoc procedure* is followed to identify the differing samples. Both these statistical tests are carried out at an $\alpha = 5\%$ level of statistical significance.

In light of the research hypothesis of §1.2, the statistical tests should reveal whether the effectiveness of the destination dispatch control algorithm does indeed differ significantly from that of the conventional control algorithms. Consider, for example, the base case of the sensitivity analysis during random inter-floor traffic. For this specific case, the null hypothesis states that no statistically significant difference with respect to observed journey time exists between the three elevator control algorithms studied. The respective KPI values observed for each control algorithm is subjected to the Friedman test. Should the Friedman test return a p-value smaller than 0.05, the null hypothesis is rejected. The *Nemenyi post hoc procedure* is next employed to identify the control algorithms that differ significantly from one another. Should the p-value of the respective comparisons between two particular control algorithms be smaller than 0.05, it may be concluded at a 95% level of confidence that the pair of control algorithms differ significantly from each other. Furthermore, the statistical tests may be used to evaluate whether the effectiveness of a single elevator control algorithm changes significantly as the independent variable value is varied in the sensitivity analysis. For example, suppose the number of floors

are varied in the sensitivity analysis. Then the Friedman test may be employed to test whether the journey time, observed for a single elevator control algorithm, changes significantly as the number of floors are changed.

5.4 Chapter summary

The KPIs selected for measuring elevator control algorithm effectiveness in this project, as well as a brief motivation for their selection, were provided in §5.1. A detailed discussion followed of the experimental design to be adopted in the effectiveness comparison analysis, defining the nature and characteristics of the simulation experiments. Finally, a description was provided in §5.3 of the methodology to be followed in order to statistically analyse elevator control algorithm effectiveness.

CHAPTER 6

Results

Contents

6.1 Analysis of key performance indicator values 49
6.2 Statistical analysis of the impact of elevator group size 50
6.3 Statistical analysis of the impact of building size 53
6.4 Statistical analysis of the impact of floor population size 56
6.5 Chapter summary 56

The purpose of this chapter is to present the results obtained in an elevator control algorithm effectiveness comparison analysis. The format in which these results are presented is described in §6.1. A detailed analysis of the KPI values observed during the simulation experiments involving the influence of the elevator group size on the effectiveness of an elevator system is provided in §6.2. Next the observed impact of the number of storeys in a high-rise building on elevator control algorithm effectiveness is discussed in §6.3. Finally, the results obtained in respect of the impact of floor population size on elevator system effectiveness is presented and discussed in §6.4. The chapter closes in §6.5 with a summary of the chapter contents, as well as an outline of the most significant findings.

6.1 Analysis of key performance indicator values

The KPI values obtained from the experiments conducted in pursuit of the sensitivity analysis described in §5.2, are presented in the form of box plots in this chapter. The average journey time and maximum journey time values observed during each elevator traffic condition and for the respective scenarios analysed, are presented alongside one another, in separate plots. In these box plots, median values are denoted by horizontal lines, while mean values are indicated by diamond symbols. In the legend accompanying each box plot, the nearest-car control algorithm is denoted by 'N-C', the fixed-sectoring common sector control algorithm by 'F-S' and the destination dispatch control algorithm by 'D-D'.

The discussion of the results entails references to the relative effectiveness observed for the respective elevator control algorithms, as well as the outcomes of the statistical tests applied to compare control algorithm performance at a 95% level of confidence.

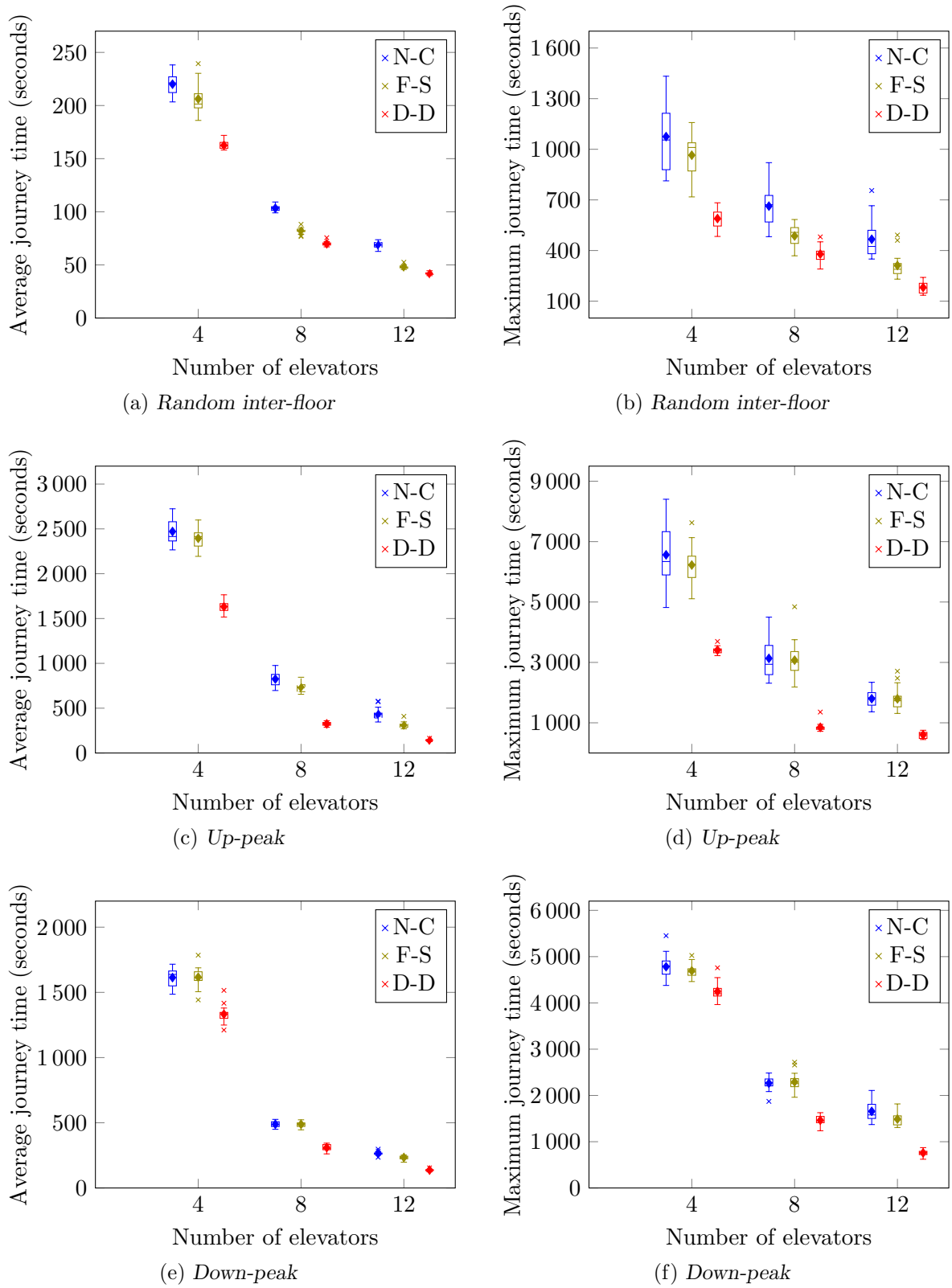
6.2 Statistical analysis of the impact of elevator group size

The KPI-values obtained during the first set of experiments (in which the number of elevators in an elevator group was varied) are presented in Figure 6.1. The results of the non-parametric tests employed during this set of experiments are shown in Tables C.1–C.8 of Appendix C. For the random inter-floor traffic condition, visual inspection of the box plots of Figures 6.1(a) and 6.1(b) reveals that journey time decreases as the number of elevators in an elevator group increases, as expected. A larger reduction in journey time was observed as the number of elevators doubled from four to eight, than for the increase from eight to twelve elevators. The destination dispatch control algorithm significantly outperformed the nearest-car and fixed-sectoring common sector control algorithms in respect of average journey time for all three elevator group sizes. Furthermore, each individual control algorithm provided reduced journey times for each increment in elevator group size at a 5% level of significance.

Apart from the decrease in observed average journey time, the consistency of the control algorithms' performance improved as well — this is evident in the shortening of the inter-quartile ranges of the box plots, most notably as the elevator group increased in size from four to eight elevators. Given the relatively small passenger arrival rates during random inter-floor traffic, the reduced variation in journey time was expected as the larger elevator group exhibits an enhanced capability to respond to landing calls quicker. Another notable observation is that the average journey time returned by the destination dispatch control algorithm, employing eight elevators in the elevator group, does not differ statistically from the average journey time yielded by the nearest-car control algorithm with twelve elevators in the elevator group (a p-value of 0.44). In this case the nearest-car control algorithm also yielded a significantly larger maximum journey time. The destination dispatch control algorithm clearly exploited the availability of additional information (obtaining the destination floor beforehand) to allocate elevators optimally, even during random inter-floor traffic.

Only in an elevator group comprising four elevators did the nearest-car and fixed-sectoring common sector control algorithms yield statistically comparable effectiveness in respect of both the observed average and maximum journey times, during random inter-floor traffic. For elevator group sizes of eight and twelve elevators, the fixed-sectoring common sector control algorithm was statistically more effective than the nearest-car control algorithm at a 5% level of significance. Evidently, the even distribution of elevators throughout the building by the fixed-sectoring common sector control algorithm contributed to its superior effectiveness. The outliers shown in Figure 6.1(b) for the nearest-car and fixed-sectoring common sector control algorithms with twelve elevators in the elevator group, are attributed to passengers who arrived at the upper and lower floors of the building, respectively. During random inter-floor traffic, the algorithm is expected to neglect the extremal floors at either end of a high-rise building, giving preference to landing calls at the middle floors. Passengers who arrived at the extreme ends of the building therefore experienced significantly larger waiting times and considerably larger journey times.

During up-peak traffic, a reduction in the average journey time was once more observed as the number of elevators in the elevator group increased, as shown in Figure 6.1(c). Only for an elevator group size of twelve elevators, did the fixed-sectoring common sector control algorithm prove more effective than the nearest-car control algorithm in terms of the observed average journey time at a 5% level of significance. The greater effectiveness exhibited by the fixed-sectoring common sector control algorithm may be attributed to the priority given to the ground floor by the algorithm (in the form of its own control sector). The observed maximum journey time for these two control algorithms did, however, not differ statistically for all elevator group sizes explored at a 5% level of significance, as shown in Figure 6.1(d).



(a) *Random inter-floor* (b) *Random inter-floor*

(c) *Up-peak* (d) *Up-peak*

(e) *Down-peak* (f) *Down-peak*

FIGURE 6.1: *KPI values observed for a 40-storey building with a floor population of 80 people, for different values of the number of elevators in the elevator group, under three prevailing elevator traffic conditions.*

The destination dispatch control algorithm provided significantly smaller average journey times than the two conventional control algorithms. The more sophisticated control algorithm returned, on average, a 51.4% decrease in average journey time compared to the nearest-car control algorithm for an elevator group comprising four elevators. This statistic is comparable with claims in the literature about the power of the destination dispatch control algorithm, especially during up-peak traffic, as discussed in §2.6. It should be noted, however, that the average journey time observed for the destination dispatch control algorithm in this scenario was 1 631 seconds. Regardless of the control algorithm's relative effectiveness, it may be assumed that no passenger would be accepting of a journey time of approximately 27 minutes. Hence the need for a larger elevator group in this particular scenario is evident. The destination dispatch control algorithm yielded a much improved average journey time of 142 seconds with twelve elevators in the elevator group. The average maximum journey time observed in this case was approximately 10 minutes, compared to the approximately 30 minutes observed for both conventional control algorithms.

Under the up-peak traffic condition, the effectiveness of each control algorithm also improved significantly as the elevator group expanded. In respect of the observed average journey time, the destination dispatch algorithm provided a similar level of consistency for eight elevators in the elevator group as was the case with twelve elevators. The fixed-sectoring common sector control algorithm, on the other hand, exhibited considerably improved performance consistency only in the case of twelve elevators, whilst the consistency of the nearest-car control algorithm remained poor throughout. The destination dispatch control algorithm proved to yield considerably smaller maximum journey times, as well as less variation in these times, than the conventional control algorithms.

The outliers in Figure 6.1(d) are ascribed to passengers who arrived at the ground floor and requested service to the topmost floors in the building. These passengers arrived when lengthy queues had already formed, which led them to experience considerable waiting times. Their journey times further increased as a result of elevator stops made at intermediate floors in response to car calls.

The average journey time observed for the destination dispatch control algorithm (during up-peak traffic) with eight elevators in the elevator group was statistically similar to the average journey time observed for the fixed-sectoring common sector control algorithm in the case where twelve elevators are employed (a p-value of 0.52). Compared to the nearest-car control algorithm with twelve elevators in the elevator group, the destination dispatch algorithm performed significantly better (a p-value of 0.003). This observation yet again underlines the power of the destination dispatch control algorithm, yielding similar or improved results with four fewer elevators in the elevator group, compared to the conventional control algorithms.

For the down-peak traffic condition, the nearest-car and fixed-sectoring common sector control algorithms delivered statistically indistinguishable results at a 5% level of significance in respect of average journey time for elevator group sizes of four and eight, respectively, as shown in Figure 6.1(e). The destination dispatch control algorithm did, however, return significantly decreased journey times compared to the two conventional algorithms, for all three elevator group sizes explored. As observed during both the random inter-floor and up-peak traffic conditions, the average journey time decreased most noticeably as the elevator group size expanded from four to eight elevators. For each incremental increase in the number of elevators, each individual control algorithm delivered superior performance over its previous performance. Notably, all three control algorithms achieved appreciable consistency for eight elevators and the variation in observed average journey time did not reduce considerably at the next elevator group size increment. Since the ground floor is the destination floor of the vast majority of elevator pas-

sengers during down-peak traffic, elevators quickly fill up to capacity and travel directly to the ground floor. Hence the lack of intermediate elevator stops considerably limit the variation in average journey time.

Considering the observed maximum journey time during down-peak traffic, the performance of the nearest-car and fixed-sectoring common sector control algorithms remained statistically indistinguishable at a 5% level of significance, as shown in Figure 6.1(e). It is evident that neither of these conventional control algorithms was able to exploit its unique characteristics to affect superior performance relative to one another during down-peak traffic. The destination dispatch control algorithm again returned significantly smaller observed maximum journey times than did the conventional control algorithms.

6.3 Statistical analysis of the impact of building size

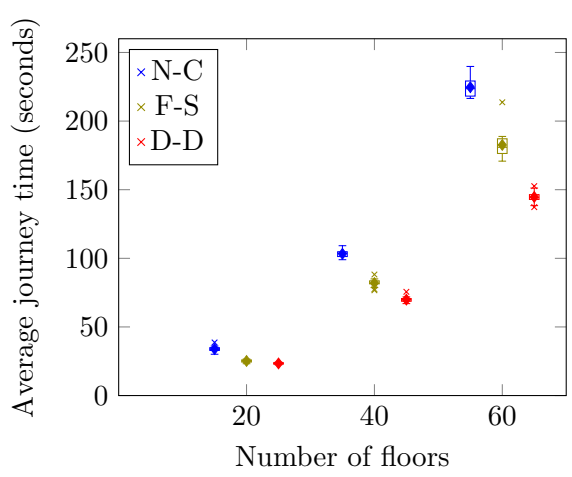
With respect to the impact of the number of floors in a high-rise building on elevator control algorithm effectiveness, the observed journey times increased significantly as the number of floors increased, as shown in Figure 6.2. The results of the non-parametric tests employed during this set of experiments are shown in Tables C.9–C.16 of Appendix C. The observed average journey time of the three respective control algorithms differ statistically from each other for all cases explored during random inter-floor traffic at a 5% level of significance. The KPI values observed during random inter-floor traffic are shown in Figures 6.2(a) and 6.2(b). Although the fixed-sectoring common sector control algorithm proved more effective than the nearest-car control algorithm in a 60-storey building, the associated maximum journey times are statistically indistinguishable at a 5% level of significance. Hence the fixed-sectoring common sector control algorithm boasted the smaller observed average journey time courtesy of its sectoring approach, but its performance was comparable with that of the nearest-car control algorithm in terms of passengers who experienced the longest journey times.

Furthermore, variation in observed journey times increased for all control algorithms as the number of floors increased. This observed variation was expected because of the greater dispersion of passenger arrivals in larger buildings during random inter-floor traffic, necessitating both an increased number of elevator stops and larger travel distances.

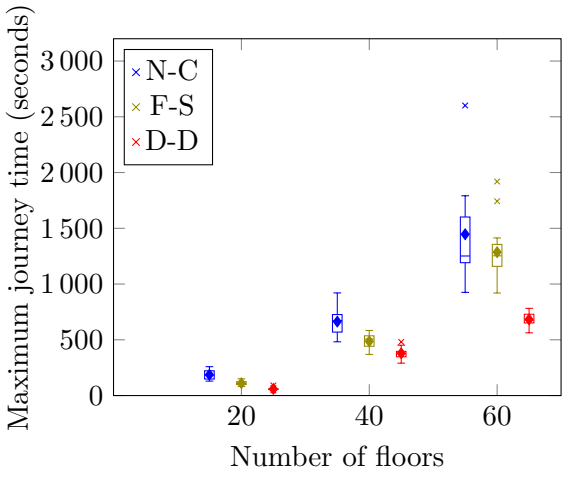
The outlier observed in Figure 6.2(b) for the nearest-car control algorithm represents a passenger who arrived at the 59th floor and requested service to the 39th floor. Given the nature of the nearest-car control algorithm to favour calls occurring at the middle floors, this passenger experienced a significant waiting time, contributing to his/her large journey time.

During up-peak traffic, the destination dispatch control algorithm performed significantly better than the conventional control algorithms for all building sizes explored, as shown in Figures 6.2(c) and 6.2(d). This superior effectiveness did not only constitute considerably reduced journey times, but also less variation in the observed journey times. Since the two conventional control algorithms do not cluster passengers by destination, they induce a larger number of intermediate elevator stops, contributing to the considerable variation in their observed journey times. Furthermore, when elevators are required to travel to the topmost floors in a building, they expend considerably more time before returning to the ground floor to pick up passengers. This phenomenon is especially evident as the number of floors in the high-rise building increases.

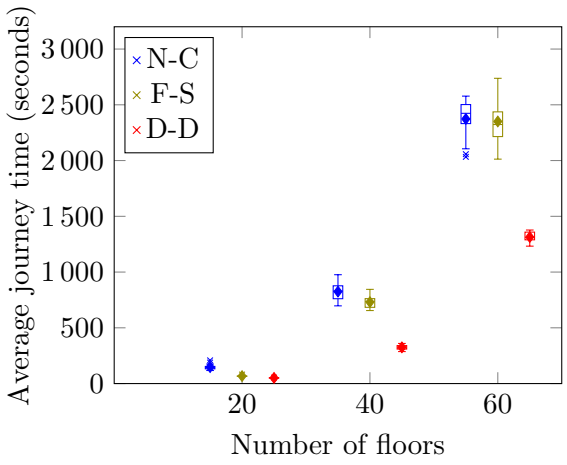
For down-peak traffic, the observed average and maximum journey times achieved by the nearest-car and fixed-sectoring common sector control algorithms are statistically indistinguishable at a 5% level of significance, as shown in Figures 6.2(e) and 6.2(f). The destination dispatch control algorithm, however, achieved significantly reduced journey times compared to the conventional



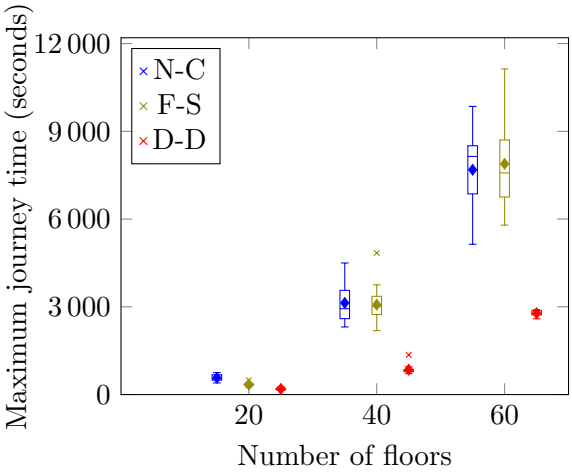
(a) *Random inter-floor*



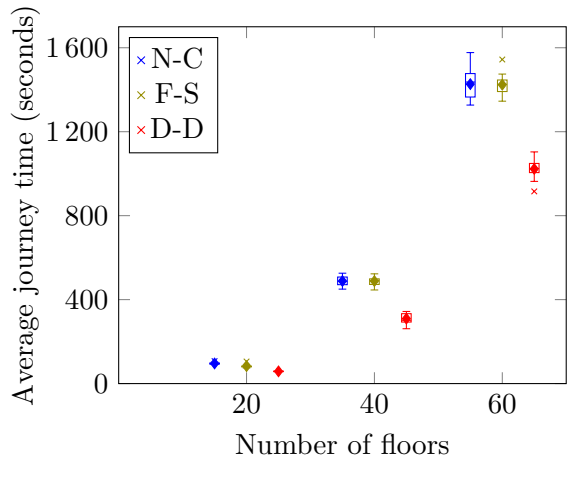
(b) *Random inter-floor*



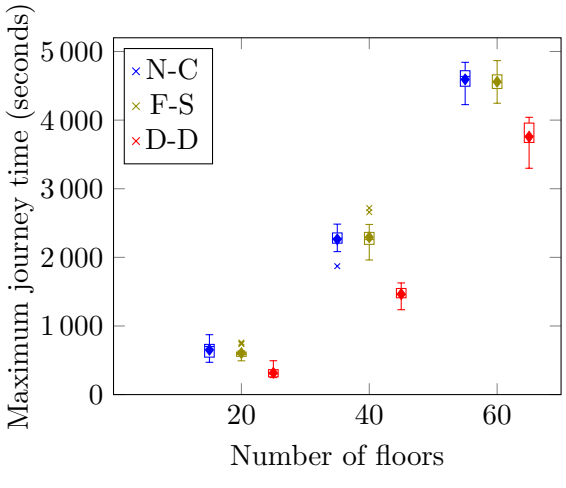
(c) *Up-peak*



(d) *Up-peak*

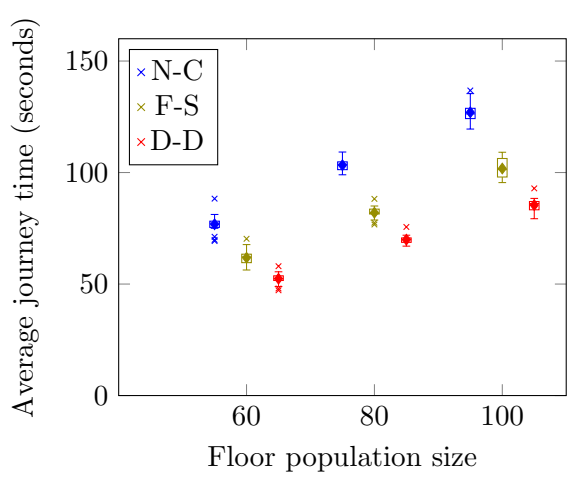


(e) *Down-peak*

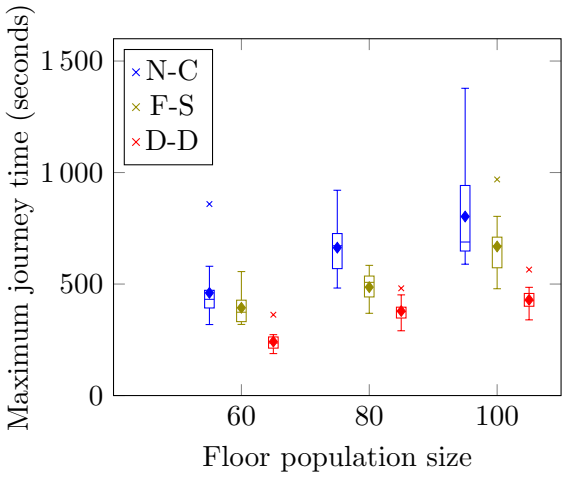


(f) *Down-peak*

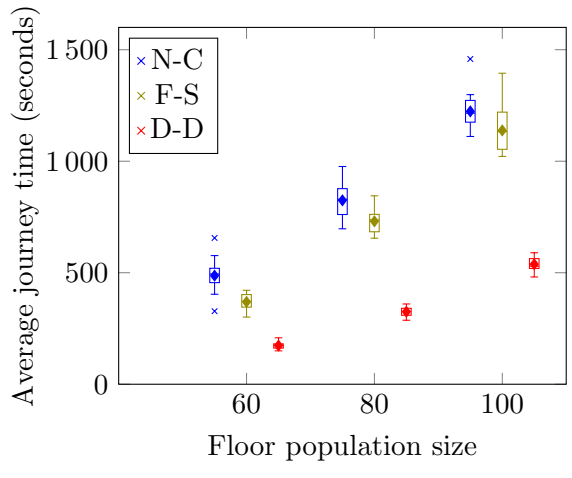
FIGURE 6.2: *KPI values observed for a building with an elevator group comprising eight elevators, a floor population of 80 people and for different values of the number of floors, under three prevailing elevator traffic conditions.*



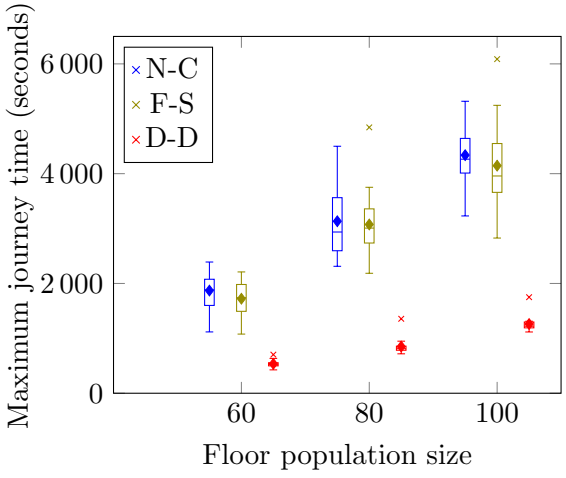
(a) *Random inter-floor*



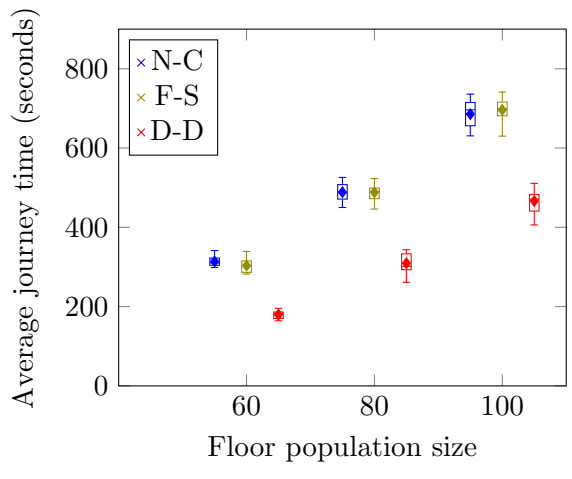
(b) *Random inter-floor*



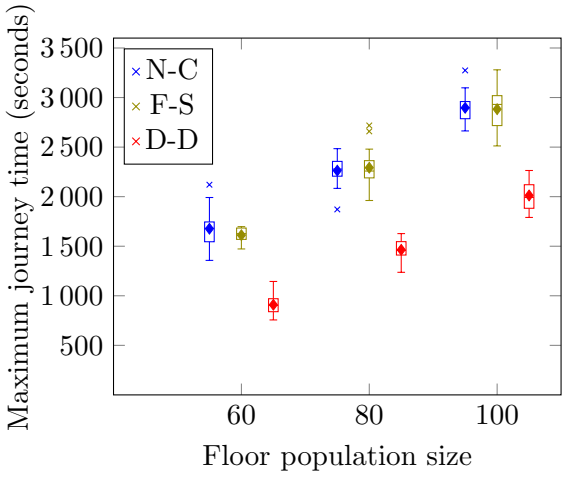
(c) *Up-peak*



(d) *Up-peak*



(e) *Down-peak*



(f) *Down-peak*

FIGURE 6.3: *KPI values observed for a 40-storey building with eight elevators in the elevator group, for varying floor population sizes, under three prevailing elevator traffic conditions.*

control algorithms. In contrast with its performance during the up-peak traffic condition, the destination dispatch control algorithm did not yield highly consistent observed journey times. This observation is attributed to the fact that the algorithm cannot optimally cluster passengers during down-peak traffic, since the majority of passengers share a common destination floor (the ground floor) and the passenger arrivals are dispersed throughout the building.

6.4 Statistical analysis of the impact of floor population size

The building population size is a chief determinant of the passenger arrival rate and hence increased journey times were observed in buildings with larger populations, as shown in Figure 6.3. The results of the non-parametric tests employed during this set of experiments are shown in Tables C.17–C.24 of Appendix C. Larger passenger arrival rates led to the formation of lengthy queues in the elevator lobbies, causing passengers to experience considerably large journey times. The fixed-sectoring common sector control algorithm again proved superior compared to the nearest-car control algorithm during random inter-floor traffic, as shown in Figures 6.3(a) and 6.3(b). The competence of the destination dispatch control algorithm is again exhibited under all three traffic conditions, most notably during the up-peak traffic condition. Regardless of the increased influx of passenger arrivals, the algorithm continued to cluster passengers optimally by destination and, as such, considerably smaller journey times (with little variation in these times) were observed. In a 40-storey building with a floor population of 100 people, the destination dispatch control algorithm yielded a 127% decrease, relative to the nearest-car control algorithm, in the average observed journey time.

Only in down-peak traffic conditions, irrespective of the floor population size, did the nearest-car and fixed-sectoring common sector control algorithms yield statistically indistinguishable average and maximum journey times at a 5% level of significance, as shown in Figures 6.3(e) and 6.3(f). Since the elevators expended much time travelling to the ground floor and immediately returned to the upper floors in response to landing calls, they were rarely stationed in a sector by the fixed-sectoring common sector control algorithm. As a result, the algorithm tended to allocate the 'nearest' elevators to landing calls and hence the resemblance in performance with that of the nearest-car control algorithm in respect of the observed journey times.

The performance of each elevator control algorithm diminished over its previous performance for each incremental increase in floor population size in respect of average journey time. The maximum journey times of the nearest-car control algorithm, however, remained statistically indistinguishable during both random inter-floor and up-peak traffic conditions at a 5% level of significance for floor population sizes of 80 and 100, respectively.

6.5 Chapter summary

The results obtained during the effectiveness comparison analysis carried out in this project were presented and interpreted in this chapter. The analyses were performed in respect of the impacts of the elevator group size, building size and floor population size on elevator control algorithm effectiveness. The nearest-car and fixed-sectoring common sector control algorithms achieved comparable effectiveness in the majority of the scenarios analysed. It may be concluded at a 95% confidence level, however, that the destination dispatch control algorithm was significantly more effective than the conventional nearest-car and fixed-sectoring common sector control algorithms. Hence there seems to be considerable evidence in support of the research hypothesis of §1.2.

CHAPTER 7

Summary and conclusion

Contents

7.1 Project summary 57
7.2 Appraisal of work 58
7.3 Possible future work 59

The purpose of this chapter is to provide concluding remarks related to this project. A summary of the project contents is provided in §7.1. A critical assessment of the project follows in §7.2 and seven suggested avenues for possible future work are finally proposed in §7.3.

7.1 Project summary

Including the current chapter, this project comprises a total of seven chapters. The aim in Chapter 1 was to provide the reader with the necessary project background, a description of the research hypothesis considered in this project, as well as the objectives pursued in this project. The project scope and project methodology were furthermore outlined.

In Chapter 2, a brief overview was provided of the relevant literature pertaining to elevator control, computer simulation and statistical performance analysis, in pursuit of Objective I of §1.3. After a brief review of the use of elevators in high-rise buildings, the prevailing elevator traffic conditions present in office buildings were described. Furthermore, the fundamental engineering requirements and rules governing elevator control were discussed. A delineation of the characteristics and working of some conventional elevator control algorithms, as well as that of the modern destination dispatch control algorithm, followed. Concluding the discussion on elevator control systems, the applicability of queuing theory in respect of elevator passengers was outlined. Next, the process of simulation design and applicable simulation model verification and validation techniques were described. The chapter concluded with a brief overview of relevant statistical analysis techniques which may be used in comparison analyses.

The purpose of Chapter 3 was first to introduce the elevator control algorithms selected for implementation and secondly, to describe the algorithmic implementation thereof in this project. This was achieved by means of verbal descriptions accompanied by supporting examples outlining the elevator allocation procedure followed by each algorithm. Since the author could not find sufficiently detailed algorithmic descriptions of these control algorithms in the literature, the further inclusion of pseudo-code descriptions of these algorithms, in fulfilment of Objective II of §1.3, is considered a valuable contribution to the literature.

The fourth chapter was dedicated to a description of the computer simulation model developed in this project to fulfill the purpose of a test bed for elevator control algorithm effectiveness in pursuit of Objective IV of §1.3. The software suite selected for the design and development of the simulation model (AnyLogic University 7.3.1) was introduced. The key aspects of the simulation model and how the behaviour of passengers and elevators were modelled were described next. The chapter finally closed with a description of the simulation model verification and validation techniques employed in this project, in fulfilment of Objective V of §1.3.

Chapter 5 delineated the experimental design process followed in this project. Firstly, the KPIs investigated in this study — average journey time and maximum journey time — were identified and their selection was briefly motivated, in fulfilment of Objective III of §1.3. The sensitivity analysis carried out in order to execute a sound effectiveness comparison analysis was described next, in pursuit of Objective VI of §1.3. The factors considered in this comparison analysis were the number of floors in a high-rise building, the number of elevators in an elevator group, the floor population size in a high-rise building and the different statistical passenger arrival distributions. Finally, a description was provided of the statistical tests employed to draw conclusions about the control algorithm effectiveness.

The objective of Chapter 6 was to present the results obtained during the sensitivity analysis carried out in this project. A detailed interpretation of the observed KPI values was provided in fulfilment of Objective VII of §1.3. It was found that the more sophisticated destination dispatch control algorithm is significantly more effective than the two conventional control algorithms considered in this project. The research hypothesis of §1.2 therefore cannot be rejected at a 95% level of confidence.

7.2 Appraisal of work

A novel computer simulation model was successfully developed in this project to serve as a test bed for the evaluation of elevator control algorithm effectiveness. Not only does the simulation model lend itself to a high degree of realism, it holds considerable potential for future use.

A considerable benefit of the computer simulation model is its flexibility. A multitude of input parameters are employed which may be used to replicate real-world elevator systems closely. These input parameters include the number of storeys in a building, the number of elevators employed in an elevator group, the passenger arrival distributions and the moving speed of the elevators. In this project, only three elevator control algorithms were implemented in the simulation model, but the model allows for the implementation of additional control algorithms. As such, the simulation model may be of considerable assistance to the designers of elevator control algorithms, or even decision makers in the construction industry.

The simulation model exhibits a considerable degree of realism in respect of both the behaviour of passengers and elevators. Arguably the most significant simplification made in this project was to ignore the effect that a crowded lobby may have on passenger behaviour. More specifically, passengers may fail to reach their allocated elevator in time if they are blocked by other people. If passengers indeed fail to enter their allocated elevator in time, it would not only impact the allocated elevator's behaviour, but necessitate the passengers to re-register their landing calls.

As highlighted in Chapter 3, the inclusion of pseudo-code descriptions of the algorithmic implementations of the elevator control algorithms analysed in this project, is considered a valuable contribution to the literature. It is acknowledged that Berntsson and Edlund [22] have proposed pseudo-code descriptions for the nearest-car and fixed-sectoring control algorithms only. The al-

gorithmic descriptions provided in this project are, however, much more detailed and extensive, and uniquely include the destination dispatch control algorithm.

7.3 Possible future work

In fulfilment of Objective VIII of §1.3, various suggestions for possible work involving extensions to or improvements of the work documented in this project are mentioned in this final section. These suggestions were not pursued during this project either due to time constraints or as a result of scope limitations.

I During the sensitivity analysis conducted in this project, only three independent variable values were studied. These variables were the number of floors in a high-rise building, the floor population size in a high-rise building and the number of elevators in an elevator group. A more suitable approach would be to analyse more independent variable values in order to draw more comprehensive conclusions pertaining to the sensitivity of elevator effectiveness with respect to these variables.

II Only two of the conventional elevator control algorithms identified in the literature review were selected for implementation in this project. These algorithms were the nearest-car and fixed-sectoring common sector control algorithms. In order to fully establish the relative effectiveness of the destination dispatch control algorithm, it would be appropriate to analyse other conventional control algorithms as well.

III It is acknowledged that elevator passengers may subjectively judge the effectiveness of an elevator system based only on their waiting time or travel time. Instead of evaluating effectiveness only in respect of journey time, this project may therefore be extended to analyse control algorithm effectiveness in terms of both waiting time and travel time. In this case, the implementation of the destination dispatch control algorithm may be altered in such a manner that the objective function assigns weighted priorities to waiting and travel times, respectively. Alternatively, a multi-dimensional analysis approach may be adopted in the sense of employing the notion of Pareto dominance.

IV Evaluation of control algorithm effectiveness was performed only from the viewpoint of an elevator passenger. This project may, however, be extended to investigate control algorithms' impact on elevator capacity utilisation and energy consumption.

V Elevator capacity was not an independent variable studied in this project and as such, every elevator modelled in this project could only accommodate a maximum of 21 passengers. Exploring the effect of elevator capacity on elevator system performance may yield valuable insights, especially in the case of very tall buildings.

VI The sophistication of elevator control algorithms have improved immensely in the modern era and it is acknowledged that not even the destination dispatch control algorithm, as implemented in this project, is the premier control algorithm currently employed in industry. An improvement of this project would be to conduct an effectiveness comparison analysis in respect of a range of modern elevator control algorithms as test subjects.

VII One of the assumptions made during the course of this project was that elevator passengers may request elevator transport irrespective of the number of floors over which they wish to travel. A more realistic approach would be to assume that some passengers who wish to travel over a relatively small number of floors would rather use the staircase to reach

their destination floors, instead of an elevator. This preference may be incorporated in the simulation model of this study.

References

[1] Alexandris N, 1977, *Statistical models in lift systems*, PhD Thesis, University of Manchester Institute of Science and Technology, Manchester.

[2] Bailey NT, 1954, *On queueing processes with bulk service*, Journal of the Royal Statistical Society, Series B (Methodological), **16(1)**, pp. 80–87.

[3] Banks J, 1998, *Handbook of simulation: Principles, methodology, advances, applications, and practice*, John Wiley & Sons, New York (NY).

[4] Banks J, Carson J & Nelson BL, 2005, *Discrete-event system simulation*, 4th Edition, Pearson, New York (NY).

[5] Barney G & dos Santos S, 1985, *Elevator traffic analysis, design and control*, 2nd Edition, Peregrinus, London.

[6] Barney G, 2003, *Elevator traffic handbook: Theory and practice*, 1st Edition, Spoon Press, London.

[7] Barney G & Al-Sharif L, 2016, *Elevator traffic handbook: Theory and practice*, 2nd Edition, Routledge, New York (NY).

[8] Braun R, 2003, *Need a lift? An elevator queueing problem*, (Unpublished) Technical Report, United Technologies Research Centre, East Hartford (CT).

[9] Browning RC, Baker EA & Herron JA, 2006, *Effects of obesity and sex on the energetic cost and preferred speed of walking*, Journal of Applied Physiology, **100(2)**, pp. 390–398.

[10] Chan W & So A, 1997, *Comprehensive dynamic zoning algorithms*, Elevator World, September, pp. 99–103.

[11] Closs GD, 1970, *The computer control of passenger traffic in large lift systems*, PhD Thesis, University of Manchester, Manchester.

[12] *CompassPlus Destination Management System*, 2016, [Online], [Cited April 2017], Available from `http://www.otis.com/site/us/pages/CompassPlusDestinationManagement.aspx`.

[13] Crawford M, 2012, *Elisha Graves Otis*, [Online], [Cited June 2017], Available from `https://www.asme.org/engineering-topics/articles/elevators/elisha-graves-otis`.

[14] Daniel WW, 1978, *Applied nonparametric statistics*, Houghton Mifflin Harcourt, Boston (MA).

[15] Dawson P, 2016, *Five things elevators teach us about design, psychology and hats*, [Online], [Cited June 2017], Available from `https://medium.com/fluxx-studio-notes/five-things-elevators-teach-us-about-design-psychology-and-hats-60558c80b8ad`.

[16] de Jong J, 2014, *Innovative elevator technologies to future proof your building*, Proceedings of the CTBUH 2014 Shanghai Conference, Shanghai, pp. 817–823.

[17] Demšar J, 2006, *Statistical comparisons of classifiers over multiple data sets*, Journal of Machine Learning Research, **7**, pp. 1–30.

[18] Derrac J, García S & Molina D, 2011, *A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms*, Swarm and Evolutionary Computation, **1(1)**, pp. 3–18.

[19] *Destination based dispatching*, 2015, [Online], [Cited June 2017], Available from `http://www.mceinc.com/products/dbd/dbd.html`.

[20] *Driving urban mobility*, [Online], [Cited April 2017], Available from `http://www.schindler.com/za/internet/en/mobility-solutions/products/destination-technology/destination-control-technology.html`.

[21] Dunietz J, 2016, *The hidden science of elevators*, [Online], [Cited April 2017], Available from `http://www.popularmechanics.com/technology/infrastructure/a20986/the-hidden-science-of-elevators/`.

[22] Edlund J & Berntsson F, 2015, *Constructing a scheduling algorithm for multidirectional elevators*, BSc Thesis, KTH Royal Institute of Technology, Stockholm.

[23] *Elevator hall stations*, [Online], [Cited April 2017], Available from `http://www.innovationind.com/index.php?src=gendocs&ref=Elevator-Hall-Stations&category=fixtures`.

[24] *Elevator history*, [Online], [Cited June 2017], Available from `http://www.columbiaelevator.com/main/elevator-history/`.

[25] *Forefinger pressing the fifth floor button in the elevator*, [Online], [Cited June 2017], Available from `https://www.shutterstock.com/image-photo/forefinger-pressing-fifth-floor-button-elevator-314165276?src=GxKr2NSTKLAnbyl3C9AsYA-1-0`.

[26] Fortune J, 2012, *Elevator destination dispatching — A revolution in the making*, Proceedings of the CTBUH 9^{th} World Congress, Shanghai, pp. 601–606.

[27] Gibbons JD & Chakraborti S, 2011, *Nonparametric statistical inference*, 5^{th} Edition, Taylor and Francis Group, Boca Raton (FL).

[28] Hochberg Y & Tamhane A, 1987, *Multiple comparison procedures*, John Wiley & Sons, New York (NY).

[29] Hollander M, Wolfe DA & Chicken E, 2013, *Nonparametric statistical methods*, John Wiley & Sons, Hoboken (NJ).

[30] Jones B, 1923, *The probable number of stops made by an elevator*, General Electric Review, **26(8)**, pp. 583–587.

[31] Kendall DG, 1951, *Some problems in the theory of queues*, Journal of the Royal Statistical Society, Series B (Methodological), **13(2)**, pp. 151–185.

[32] Kleijnen JP, 1995, *Verification and validation of simulation models*, European Journal of Operational Research, **82(1)**, pp. 145–162.

[33] Al-Kodmany K, 2015, *Tall buildings and elevators: A review of recent technological advances*, Buildings, **5(3)**, pp. 1070–1104.

[34] Koehler J & Ottiger D, 2002, *An AI-based approach to destination control in elevators*, AI Magazine, **23(3)**, pp. 59–78.

[35] KONE, 2011, *KONE Polaris*, (Unpublished) Technical Report, KONE, Espoo.

[36] Law Averill M & David KW, 2000, *Simulation modeling and analysis*, 3rd Edition, McGraw-Hill, Boston (MA).

[37] Law AM, 2003, *How to conduct a successful simulation study*, Proceedings of the 35^{th} Winter Simulation Conference: Driving Innovation, New Orleans (LA), pp. 66–70.

[38] Luh PB, Xiong B & Chang SC, 2008, *Group elevator scheduling with advance information for normal and emergency modes*, EEE Transactions on Automation Science and Engineering, **5(2)**, pp. 245–258.

[39] Maria A, 1997, *Introduction to modeling and simulation*, Proceedings of the 29^{th} Winter Simulation Conference, Atlanta (GA), pp. 7–13.

[40] Montgomery DC & Runger GC, 2014, *Applied statistics and probability for engineers*, 6th Edition, John Wiley & Sons, Hoboken (NJ).

[41] Otis, 2016, *Compass Destination Management*, [Online], [Cited October 2017], Available from `http://www.otis.com/site/nz/pages/CompassDestinationManagement.aspx`.

[42] Robinson S, 2004, *Simulation: The practice of model development and use*, John Wiley & Sons, Chichester.

[43] Sargent RG, 2004, *Validation and verification of simulation models*, Proceedings of the 36^{th} Winter Simulation Conference, Washington (DC), pp. 17–28.

[44] Sargent RG, 2005, *Verification and validation of simulation models*, Proceedings of the 37^{th} Winter Simulation Conference, Orlando (FL), pp. 130–143.

[45] *SchindlerID*, [Online], [Cited April 2017], Available from `http://elevation.wikia.com/wiki/SchindlerID`.

[46] Schlünz EB, 2016, *Multiobjective in-core fuel management optimisation for nuclear research reactors*, PhD Thesis, Stellenbosch University, Stellenbosch.

[47] Schröder J, 1955, *Personenaufzüge, Berechnung ihrer Förder-leistug als Planungsgrundlage*, Föerden und Heben, **5(1)**, pp. 44–50.

[48] Shannon RE, 1975, *Systems simulation: The art and science*, Prentice Hall, Englewood Cliffs (NJ).

[49] Shannon RE, 1998, *Introduction to the art and science of simulation*, Proceedings of the 30^{th} Winter Simulation Conference, Washington (DC), pp. 7–14.

[50] Al-Sharif L, 2016, *Introduction to elevator group control (METE XI)*, Lift Report, **42(2)**, pp. 59–68.

[51] So AT, Yu J & Chan W, 1999, *Dynamic zoning based supervisory control for elevators*, Proceedings of the IEEE International Conference on Control Applications, Kohala Coast (HI), pp. 1591–1596.

[52] Strang T & Bauer C, 2007, *Context-aware elevator scheduling*, Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops, Niagra Falls, pp. 276–281.

[53] Winston WL, 2004, *Operations research: Applications and algorithms*, 4th Edition, Brooks /Cole, Belmont (CA).

[54] Wit J, 2007, *Belifting van Hoogbouw*, TVVL Magazine, **6**, pp. 30–38.

APPENDIX A

Project Timeline

The timeline followed during the execution of this project is given in Figure A.1 in Gantt-chart form.

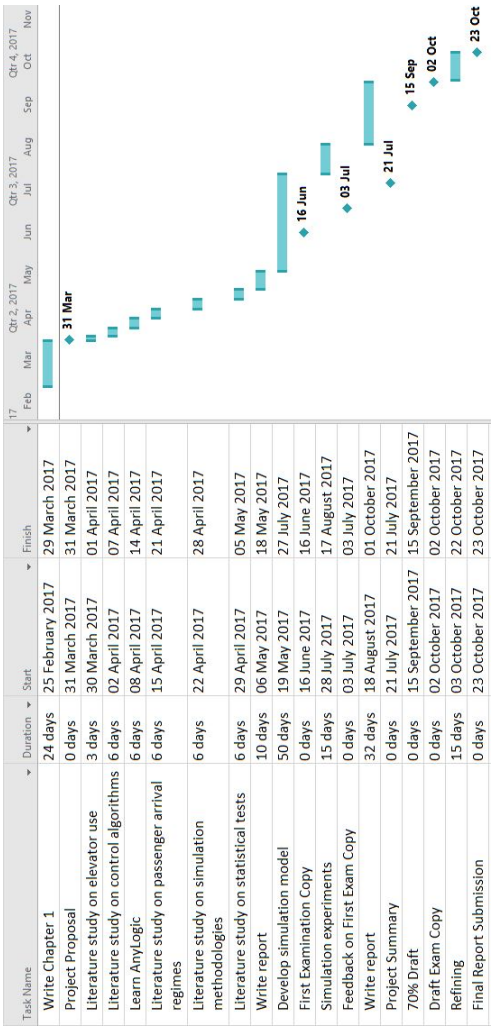


FIGURE A.1: *Project timeline in Gantt-chart form.*

APPENDIX B

Personal reflections

This appendix contains non-academic reflections by the author on the value of this project in terms of its potential contribution to society (in §B.1), as well as an account of what the author has learned in terms of personal and inter-personal skills during the execution of this project (in §B.2).

B.1 Contribution to society

At the heart of this project lies a novel computer simulation model constructed to serve as test bed for elevator control algorithm effectiveness. This tool may be of considerable assistance to elevator manufacturers and the construction industry.

Elevator manufacturers, in conjunction with building owners, may employ the simulation model to evaluate the impact of implementing different elevator control algorithms in an existing elevator system. A significant advantage is that these experiments may be carried out without disrupting the actual elevator system. The elevator manufacturer may choose to replace the current control algorithm if the output of the simulation study proves that another algorithm will more sufficiently satisfy the demand for elevator service in the relevant building. Another outcome of such a study may be that improved elevator system performance is achieved by means of employing fewer elevators in the elevator group. In this case, such a solution may yield significant energy savings. Similarly, the simulation model may be employed during the process of building design in order to advise on the number of elevators to be installed to satisfy forecast elevator service demand. As for elevator manufacturers *per se*, the simulation model may be utilised to great effect in the development and testing of new elevator control algorithms.

B.2 What the author has learned

This study constitutes the first significant research project undertaken by the author in his personal career. During the execution of this project, the author was exposed to the various bodies of knowledge in the realms of elevator control systems, simulation modelling and statistical analysis. Furthermore, the author acquired and refined several skills over the course of the project. The relevance and implicit value of these skills are briefly discussed in this section.

During the early stages of the project execution, the author swiftly learned how to conduct purposeful research by means of sound research methodologies. Furthermore, the author learned to work independently and in order to remain objective, constantly question his problem-solving approaches. Owing to the various demands posed by this project over a sustained period of time, the author was required to exhibit considerable self-discipline and dedication.

Arguably the chief skill acquired during the execution of this project was learning the art of computer simulation modelling. The author became sufficiently proficient in the use of the AnyLogic software suite — coupled with a sharpening of his programming skills. The process of acquiring these skills was facilitated by a steep learning curve in which the author had the responsibility to self-learn the bulk of the work. In view of the numerous difficulties encountered whilst developing the computer simulation model, the author also learned to reflect patiently on his programming approach and to change it accordingly, if required. The author learned to confidently analyse a problem and adequately translate the problem space into a computerised simulation model.

Apart from vastly improving his ability to produce sound technical written work, the author gained considerable competence in the use of the LaTeX typesetting environment which was employed in the production of this research document.

Within the SUnORE research group, the author had the opportunity to present his research to fellow members — peers, postgraduate students and academics. This experience provided the author with the opportunity to elicit feedback on his work, as well as to learn and practise effective oral communication and presentation skills.

In closing, this project taught the author to carefully plan his work and to manage his time effectively. This was of utmost importance, considering that the author had to attend to ten other modules which form part of the final-year curriculum of the industrial engineering programme presented at Stellenbosch University.

APPENDIX C

Results of non-parametric tests

The results of the non-parametric tests performed in view of the statistical analysis of §6 are presented in this appendix. In each table, the nearest-car control algorithm is denoted by 'N-C', the fixed-sectoring common sector control algorithm by 'F-S' and the destination dispatch control algorithm by 'D-D'.

There are twenty-four tables in total. Tables C.1–C.8 pertain to that part of the sensitivity analysis in §6.2 in which the number of elevators in an elevator group was varied, and contain the results of the Friedman and Nemenyi *post hoc* tests related to Figure 6.1. Tables C.9–C.16, in turn, pertain to that part of the sensitivity analysis in §6.3 in which the number of floors in a high-rise building was varied, and contain the results of the Friedman and Nemenyi *post hoc* tests related to Figure 6.2. Tables C.17–C.24 finally pertain to that part of the sensitivity analysis in §6.4 in which the floor population size in a high-rise building was varied, and contain the results of the Friedman and Nemenyi *post hoc* tests related to Figure 6.3.

Number of elevators	Traffic condition	N-C	F-S	D-D	Friedman test p-value
4	Random inter-floor	220.1	206.2	162.6	1.731×10^{-6}
8	Random inter-floor	103.3	82.0	69.9	3.059×10^{-7}
12	Random inter-floor	68.8	48.0	41.8	3.059×10^{-7}
4	Up-peak	2 470.5	2 394.4	1 631.4	5.748×10^{-6}
8	Up-peak	824.6	730.6	324.9	1.731×10^{-6}
12	Up-peak	432.9	309.6	141.7	7.779×10^{-7}
4	Down-peak	1 613.5	1 616.5	1 332.8	1.120×10^{-5}
8	Down-peak	488.7	488.3	309.5	1.279×10^{-5}
12	Down-peak	264.9	233.8	137.3	3.059×10^{-7}

TABLE C.1: *Average journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the number of elevators in the elevator group was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.110	—
D-D	9.5×10^{-7}	0.005

(A) *Four elevators — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(B) *Eight elevators — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(C) *Twelve elevators — random inter-floor*

	N-C	F-S
F-S	0.408	—
D-D	6.2×10^{-6}	0.002

(D) *Four elevators — up-peak*

	N-C	F-S
F-S	0.110	—
D-D	9.5×10^{-7}	0.005

(E) *Eight elevators — up-peak*

	N-C	F-S
F-S	0.046	—
D-D	3.6×10^{-7}	0.010

(F) *Twelve elevators — up-peak*

	N-C	F-S
F-S	0.848	—
D-D	0.000	3.5×10^{-5}

(G) *Four elevators — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *Eight elevators — down-peak*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(I) *Twelve elevators — down-peak*

TABLE C.2: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions when the number of elevators in the elevator group was varied in Table C.1. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Number of elevators	Traffic condition	N-C	F-S	D-D	Friedman test p-value
4	Random inter-floor	1 075.0	964.9	589.6	8.575×10^{-6}
8	Random inter-floor	663.1	486.1	379.0	2.847×10^{-5}
12	Random inter-floor	466.7	311.2	181.4	7.779×10^{-7}
4	Up-peak	6 561.9	6 224.3	3 406.8	8.575×10^{-6}
8	Up-peak	3 133.4	3 074.5	850.3	1.279×10^{-5}
12	Up-peak	1 798.5	1 794.3	600.5	1.120×10^{-5}
4	Down-peak	4 781.0	4 693.3	4 246.8	8.575×10^{-6}
8	Down-peak	2 262.8	2 292.2	1 463.8	1.279×10^{-5}
12	Down-peak	1 656.5	1 488.3	759.1	5.748×10^{-6}

TABLE C.3: *Maximum journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the number of elevators in the elevator group was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-7}	0.001

(A) *Four elevators — random inter-floor*

	N-C	F-S
F-S	0.029	—
D-D	1.5×10^{-5}	0.110

(B) *Eight elevators — random inter-floor*

	N-C	F-S
F-S	0.046	—
D-D	3.6×10^{-7}	0.010

(C) *Twelve elevators — random inter-floor*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-5}	0.001

(D) *Four elevators — up-peak*

	N-C	F-S
F-S	0.982	—
D-D	0.000	7.9×10^{-5}

(E) *Eight elevators — up-peak*

	N-C	F-S
F-S	0.848	—
D-D	3.5×10^{-5}	0.000

(F) *Twelve elevators — up-peak*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-5}	0.001

(G) *Four elevators — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *Eight elevators — down-peak*

	N-C	F-S
F-S	0.408	—
D-D	6.2×10^{-6}	0.002

(I) *Twelve elevators — down-peak*

TABLE C.4: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions when the number of elevators in the elevator group was varied in Table C.3. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Number of elevators 4	8	12	Friedman test p-value
N-C	Random inter-floor	220.1	103.3	68.8	3.059×10^{-7}
F-S	Random inter-floor	206.2	82.0	48.0	3.059×10^{-7}
D-D	Random inter-floor	162.2	69.9	41.8	3.059×10^{-7}
N-C	Up-peak	2 470.5	824.6	432.9	3.059×10^{-7}
F-S	Up-peak	2 394.4	730.7	309.6	3.059×10^{-7}
D-D	Up-peak	1 631.4	324.9	141.7	3.059×10^{-7}
N-C	Down-peak	1 613.5	488.7	264.9	3.059×10^{-7}
F-S	Down-peak	1 616.5	488.3	233.8	3.059×10^{-7}
D-D	Down-peak	1 332.8	309.5	137.3	3.059×10^{-7}

TABLE C.5: *Average journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the number of elevators in the elevator group was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(A) *N-C — random inter-floor*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(B) *F-S — random inter-floor*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(C) *D-D — random inter-floor*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(D) *N-C — up-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(E) *F-S — up-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(F) *D-D — up-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(H) *F-S — down-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.6: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions in Table C.5. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of elevator group sizes are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Number of elevators 4	8	12	Friedman test p-value
N-C	Random inter-floor	1 075.0	663.1	466.7	7.779×10^{-7}
F-S	Random inter-floor	964.9	486.1	311.2	7.779×10^{-7}
D-D	Random inter-floor	589.6	379.0	181.4	3.059×10^{-7}
N-C	Up-peak	6 562.0	3 133.4	1 798.5	3.059×10^{-7}
F-S	Up-peak	6 224.3	3 074.5	1 794.3	7.779×10^{-7}
D-D	Up-peak	3 406.8	850.3	600.5	3.059×10^{-7}
N-C	Down-peak	4 781.0	2 262.8	1 656.5	3.059×10^{-7}
F-S	Down-peak	4 693.3	2 292.2	1 488.3	3.059×10^{-7}
D-D	Down-peak	4 246.8	1 463.8	759.1	3.059×10^{-7}

TABLE C.7: *Maximum journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the number of elevators in the elevator group was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	4	8
8	0.010	—
12	3.6×10^{-7}	0.046

(A) *N-C — random inter-floor*

	4	8
8	0.010	—
12	3.6×10^{-7}	0.046

(B) *F-S — random inter-floor*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(C) *D-D — random inter-floor*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(D) *N-C — up-peak*

	4	8
8	0.010	—
12	3.6×10^{-7}	0.046

(E) *F-S — up-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(F) *D-D — up-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(H) *F-S — down-peak*

	4	8
8	0.017	—
12	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.8: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions in Table C.7. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of elevator group sizes are equal) is very small [large, respectively].*

Number of floors	Traffic condition	N-C	F-S	D-D	Friedman test p-value
20	Random inter-floor	33.9	25.2	23.4	3.059×10^{-7}
40	Random inter-floor	103.3	82.0	69.9	3.059×10^{-7}
60	Random inter-floor	224.7	182.6	144.7	3.059×10^{-7}
20	Up-peak	147.7	67.1	50.8	3.059×10^{-7}
40	Up-peak	824.6	730.7	324.9	1.731×10^{-6}
60	Up-peak	2 373.0	2 350.5	1 313.8	1.279×10^{-5}
20	Down-peak	95.8	82.9	58.2	3.059×10^{-7}
40	Down-peak	488.7	488.3	309.5	1.279×10^{-5}
60	Down-peak	1 426.8	1 423.9	1 022.7	1.279×10^{-5}

TABLE C.9: *Average journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the number of floors in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(A) *20 floors — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(B) *40 floors — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(C) *60 floors — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(D) *20 floors — up-peak*

	N-C	F-S
F-S	0.110	—
D-D	9.5×10^{-7}	0.005

(E) *40 floors — up-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(F) *60 floors — up-peak*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(G) *20 floors — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *40 floors — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	0.000	7.9×10^{-5}

(I) *60 floors — down-peak*

TABLE C.10: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions when the number of floors in the high-rise building was varied in Table C.9. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Number of floors	Traffic condition	N-C	F-S	D-D	Friedman test p-value
20	Random inter-floor	186.8	110.8	59.0	3.059×10^{-7}
40	Random inter-floor	663.1	486.1	379.0	2.847×10^{-5}
60	Random inter-floor	1 446.5	1 285.9	683.0	1.120×10^{-5}
20	Up-peak	574.6	343.4	192.2	3.059×10^{-7}
40	Up-peak	3 133.4	3 074.5	850.3	1.279×10^{-5}
60	Up-peak	7 685.1	7 888.9	2 779.2	1.120×10^{-5}
20	Down-peak	651.7	604.1	317.2	8.575×10^{-6}
40	Down-peak	2 262.8	2 292.2	1 463.8	1.279×10^{-5}
60	Down-peak	4 593.4	4 559.2	3 758.1	1.120×10^{-5}

TABLE C.11: *Maximum journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the number of floors in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(A) *20 floors — random inter-floor*

	N-C	F-S
F-S	0.029	—
D-D	1.5×10^{-5}	0.110

(B) *40 floors — random inter-floor*

	N-C	F-S
F-S	0.848	—
D-D	3.5×10^{-5}	0.000

(C) *60 floors — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(D) *20 floors — up-peak*

	N-C	F-S
F-S	0.982	—
D-D	0.000	7.9×10^{-5}

(E) *40 floors — up-peak*

	N-C	F-S
F-S	0.848	—
D-D	0.000	3.5×10^{-5}

(F) *60 floors — up-peak*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-5}	0.001

(G) *20 floors — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *40 floors — down-peak*

	N-C	F-S
F-S	0.848	—
D-D	3.5×10^{-5}	0.000

(I) *60 floors — down-peak*

TABLE C.12: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions when the number of floors in the high-rise building was varied in Table C.11. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Number of floors 20	40	60	Friedman test p-value
N-C	Random inter-floor	33.9	103.3	224.7	3.059×10^{-7}
F-S	Random inter-floor	25.2	82.0	182.6	3.059×10^{-7}
D-D	Random inter-floor	23.4	69.9	144.7	3.059×10^{-7}
N-C	Up-peak	147.7	824.6	2 373.0	3.059×10^{-7}
F-S	Up-peak	67.1	730.7	2 350.5	3.059×10^{-7}
D-D	Up-peak	50.8	324.9	1 313.8	3.059×10^{-7}
N-C	Down-peak	95.8	488.7	1 426.8	3.059×10^{-7}
F-S	Down-peak	82.9	488.3	1 423.9	3.059×10^{-7}
D-D	Down-peak	58.2	309.5	1 022.7	3.059×10^{-7}

TABLE C.13: *Average journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the number of floors in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(A) *N-C — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(B) *F-S — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(C) *D-D — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(D) *N-C — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(E) *F-S — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(F) *D-D — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(H) *F-S — down-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.14: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions in Table C.13. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of building sizes are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Number of floors 20	40	60	Friedman test p-value
N-C	Random inter-floor	186.8	663.1	1 446.5	3.059×10^{-7}
F-S	Random inter-floor	110.8	486.1	1 285.9	3.059×10^{-7}
D-D	Random inter-floor	59.0	379.0	683.0	3.059×10^{-7}
N-C	Up-peak	574.6	3 133.4	7 685.1	3.059×10^{-7}
F-S	Up-peak	343.4	3 074.5	7 888.9	3.059×10^{-7}
D-D	Up-peak	192.2	850.3	2 779.2	3.059×10^{-7}
N-C	Down-peak	651.7	2 262.8	4 593.4	3.059×10^{-7}
F-S	Down-peak	604.1	2 292.3	4 559.2	3.059×10^{-7}
D-D	Down-peak	317.2	1 463.8	3 758.1	3.059×10^{-7}

TABLE C.15: *Maximum journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the number of floors in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(A) *N-C — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(B) *F-S — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(C) *D-D — random inter-floor*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(D) *N-C — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(E) *F-S — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(F) *D-D — up-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(H) *F-S — down-peak*

	20	40
40	0.017	—
60	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.16: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions in Table C.15. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of building sizes are equal) is very small [large, respectively].*

Floor population size	Traffic condition	N-C	F-S	D-D	Friedman test p-value
60	Random inter-floor	76.8	61.8	52.4	3.059×10^{-7}
80	Random inter-floor	103.3	82.0	69.9	3.059×10^{-7}
100	Random inter-floor	127.0	101.8	85.4	3.059×10^{-7}
60	Up-peak	486.8	369.1	173.5	3.059×10^{-7}
80	Up-peak	824.6	730.6	324.9	1.731×10^{-6}
100	Up-peak	1 223.5	1 138.1	538.2	5.748×10^{-6}
60	Down-peak	314.1	303.4	179.5	1.731×10^{-6}
80	Down-peak	488.7	488.3	309.5	1.279×10^{-5}
100	Down-peak	685.6	696.2	466.2	1.120×10^{-5}

TABLE C.17: *Average journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the floor population size in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(A) *60 occupants — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(B) *80 occupants — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(C) *100 occupants — random inter-floor*

	N-C	F-S
F-S	0.017	—
D-D	1.3×10^{-7}	0.017

(D) *60 occupants — up-peak*

	N-C	F-S
F-S	0.110	—
D-D	9.5×10^{-7}	0.005

(E) *80 occupants — up-peak*

	N-C	F-S
F-S	0.408	—
D-D	6.2×10^{-6}	0.002

(F) *100 occupants — up-peak*

	N-C	F-S
F-S	0.110	—
D-D	9.5×10^{-7}	0.005

(G) *60 occupants — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *80 occupants — down-peak*

	N-C	F-S
F-S	0.848	—
D-D	0.000	3.5×10^{-5}

(I) *100 occupants — down-peak*

TABLE C.18: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions when the floor population size in the high-rise building was varied in Table C.17. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Floor population size	Traffic condition	N-C	F-S	D-D	Friedman test p-value
60	Random inter-floor	460.4	393.6	242.0	3.043×10^{-5}
80	Random inter-floor	663.1	486.1	379.0	2.847×10^{-5}
100	Random inter-floor	802.8	668.4	429.3	8.575×10^{-6}
60	Up-peak	1 870.4	1 721.7	535.7	8.575×10^{-6}
80	Up-peak	3 133.4	3 074.5	850.3	1.279×10^{-5}
100	Up-peak	4 337.4	4 145.5	1 261.2	1.120×10^{-5}
60	Down-peak	1 676.6	1 611.5	909.3	1.120×10^{-5}
80	Down-peak	2 262.8	2 292.2	1 463.8	1.279×10^{-5}
100	Down-peak	2 896.5	2 882.7	2 012.2	1.279×10^{-5}

TABLE C.19: *Maximum journey time KPI values observed for three elevator control algorithms during a specified elevator traffic condition, when the floor population size in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of all three elevator control algorithms are equal) is very small.*

	N-C	F-S
F-S	0.517	—
D-D	3.5×10^{-5}	0.003

(A) *60 occupants — random inter-floor*

	N-C	F-S
F-S	0.029	—
D-D	1.5×10^{-5}	0.110

(B) *80 occupants — random inter-floor*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-5}	0.001

(C) *100 occupants — random inter-floor*

	N-C	F-S
F-S	0.632	—
D-D	1.5×10^{-5}	0.001

(D) *60 occupants — up-peak*

	N-C	F-S
F-S	0.982	—
D-D	0.000	7.9×10^{-5}

(E) *80 occupants — up-peak*

	N-C	F-S
F-S	0.845	—
D-D	3.5×10^{-5}	0.000

(F) *100 occupants — up-peak*

	N-C	F-S
F-S	0.848	—
D-D	3.5×10^{-5}	0.000

(G) *60 occupants — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(H) *80 occupants — down-peak*

	N-C	F-S
F-S	0.982	—
D-D	7.9×10^{-5}	0.000

(I) *100 occupants — down-peak*

TABLE C.20: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions when the floor population size in the high-rise building was varied in Table C.19. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a pair of elevator control algorithms are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Floor population size 60	80	100	Friedman test p-value
N-C	Random inter-floor	76.8	103.3	127.0	3.059×10^{-7}
F-S	Random inter-floor	61.8	82.0	101.8	3.059×10^{-7}
D-D	Random inter-floor	52.4	69.9	85.4	3.059×10^{-7}
N-C	Up-peak	486.8	824.6	1 223.5	3.059×10^{-7}
F-S	Up-peak	369.1	730.6	1 138.1	3.059×10^{-7}
D-D	Up-peak	173.5	324.9	538.2	3.059×10^{-7}
N-C	Down-peak	314.1	488.7	685.6	3.059×10^{-7}
F-S	Down-peak	303.4	488.3	696.2	3.059×10^{-7}
D-D	Down-peak	179.5	309.5	466.2	3.059×10^{-7}

TABLE C.21: *Average journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the floor population size in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(A) *N-C — random inter-floor*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(B) *F-S — random inter-floor*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(C) *D-D — random inter-floor*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(D) *N-C — up-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(E) *F-S — up-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(F) *D-D — up-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(H) *F-S — down-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.22: *Nemenyi post-hoc test p-values for the average journey time KPI values observed during three prevailing elevator traffic conditions in Table C.21. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of floor population sizes are equal) is very small [large, respectively].*

Elevator control algorithm	Traffic condition	Floor population size 60	80	100	Friedman test p-value
N-C	Random inter-floor	460.4	663.1	802.8	2.569×10^{-4}
F-S	Random inter-floor	393.6	486.1	668.4	5.545×10^{-5}
D-D	Random inter-floor	242.0	379.0	429.3	5.748×10^{-6}
N-C	Up-peak	1 870.4	3 133.4	4 337.4	1.731×10^{-6}
F-S	Up-peak	1 721.7	3 074.5	4 145.5	1.731×10^{-6}
D-D	Up-peak	535.7	850.3	1 261.1	7.779×10^{-7}
N-C	Down-peak	1 676.6	2 262.8	2 896.5	3.059×10^{-7}
F-S	Down-peak	1 611.5	2 292.2	2 882.7	7.779×10^{-7}
D-D	Down-peak	909.3	1 463.8	2 012.2	3.059×10^{-7}

TABLE C.23: *Maximum journey time KPI values observed for each elevator control algorithm during a specified elevator traffic condition, when the floor population size in the high-rise building was varied. The corresponding p-values returned by the Friedman test are also included. These p-values are all smaller than 0.05, showing that the probability of incorrectly rejecting the null-hypothesis (stating that the journey times of a single elevator control algorithm are equal) is very small.*

	60	80
80	0.029	—
100	0.000	0.310

(A) *N-C — random inter-floor*

	60	80
80	0.228	—
100	3.5×10^{-5}	0.017

(B) *F-S — random inter-floor*

	60	80
80	0.002	—
100	6.2×10^{-6}	0.408

(C) *D-D — random inter-floor*

	60	80
80	0.005	—
100	9.5×10^{-7}	0.110

(D) *N-C — up-peak*

	60	80
80	0.005	—
100	9.5×10^{-7}	0.110

(E) *F-S — up-peak*

	60	80
80	0.010	—
100	3.6×10^{-7}	0.046

(F) *D-D — up-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(G) *N-C — down-peak*

	60	80
80	0.010	—
100	3.6×10^{-7}	0.046

(H) *F-S — down-peak*

	60	80
80	0.017	—
100	1.3×10^{-7}	0.017

(I) *D-D — down-peak*

TABLE C.24: *Nemenyi post-hoc test p-values for the maximum journey time KPI values observed during three prevailing elevator traffic conditions in Table C.23. Table entries smaller than [larger than, respectively] 0.05 are typeset in black [red, respectively], and show that the probability of incorrectly rejecting [not rejecting, respectively] the null-hypothesis (stating that the journey times of a single elevator control algorithm observed for a pair of floor population sizes are equal) is very small [large, respectively].*

APPENDIX D

Contents of the accompanying compact disc

This appendix provides a brief description of the contents on the compact disc included with this report. The compact disc includes an electronic version of the report itself in ".pdf" format. The AnyLogic project file of the simulation model described in Chapter 4 is also included. Finally, the compact disc contains the passenger arrival data sets employed within the simulation experiments described in Chapter 5 as Microsoft Excel workbooks. There are three directories on the compact disc:

Report. The electronic version of this report is contained within this directory.

Passenger arrival data. Fifteen Microsoft Excel workbooks are stored in this directory. Each workbook contains 15 unique passenger arrival data sets. Each set is associated with a building containing a specified number of floors, a specified floor population size and a specified elevator traffic condition. Each data record contains a passenger index, the passenger's landing and destination floors, the passenger's arrival time, as well as the inter-arrival time.

Simulation model. This directory contains the simulation model described in Chapter 4 as an AnyLogic project file (".alp" format). The project file is labelled "ElevatorSystemSimulation.alp." To execute this simulation model, the project file should be opened from AnyLogic. The user is required to run the model by clicking the "Run" button. Once the simulation model is initialised, a window will appear in which the user may alter parameter values. The user should then click the "Run" button and the GUI shown in Figure 4.5 will appear.