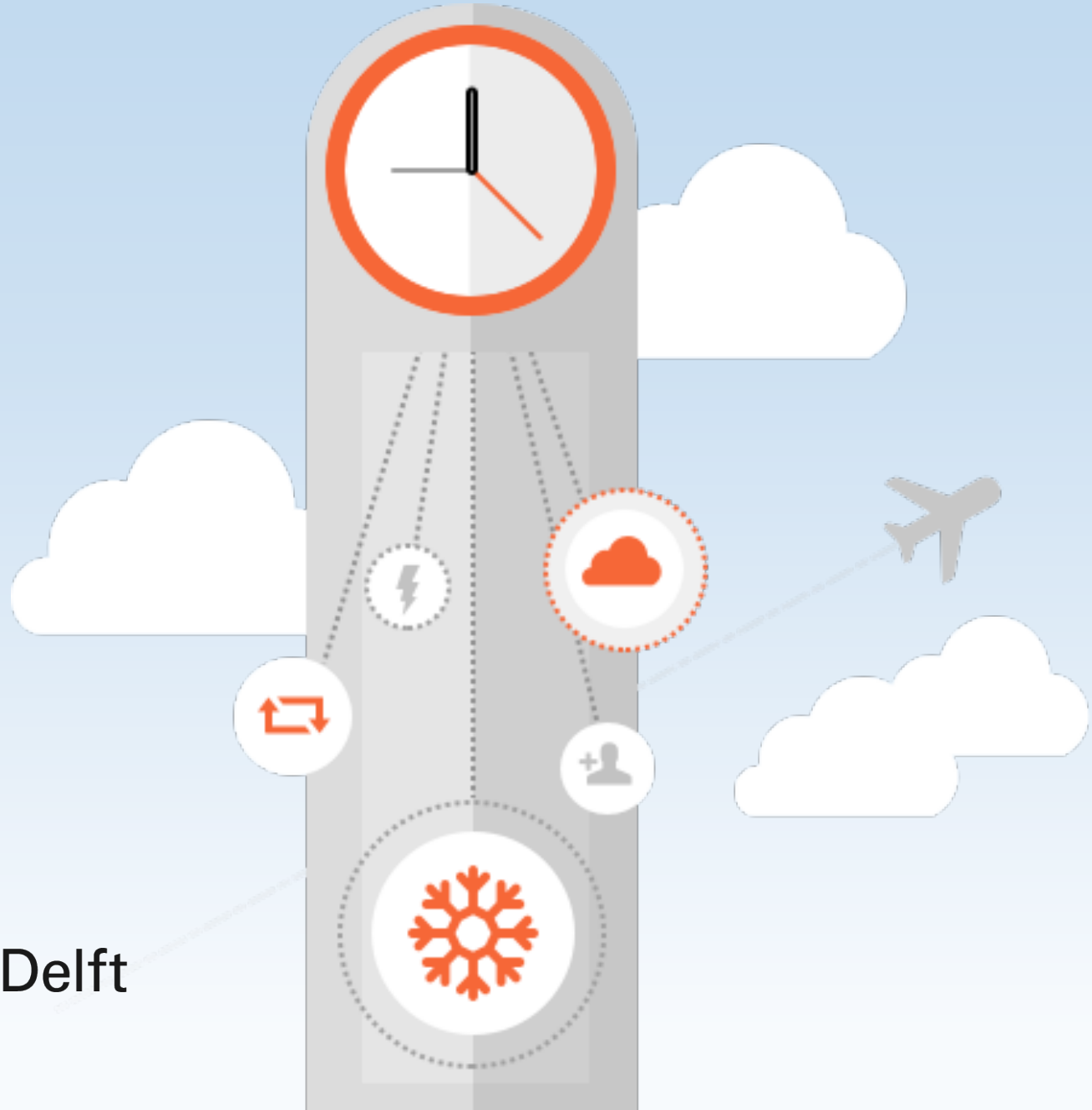
Towards Formalization and Operationalization of Resilience

A study on anticipation in the context of airport security operations

by Anne-Nynke Blok



TU Delft

Towards Formalization and Operationalization of Resilience

A study on anticipation in the context of airport security operations

by

Anne-Nynke Rozemarijn Blok

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on March 2, 2018

Student number:	4100077	
Project duration:	May 24, 2017 – March 2, 2018	
Thesis committee:	Prof. Dr. R. Curran	TU Delft, ATO, chair
	Dr. O. A. Sharpanskykh	TU Delft, ATO, supervisor
	Dr. Ir. E. van Kampen	TU Delft, C&S, external examiner
	PhD student M. P. J. Vert	TU Delft, ATO, examiner

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

During my master program I came across the master course Agent-based Modelling and Simulation in Air Transport, given by Dr. Alexei Sharpanskykh. This course taught me that human behavior provides a critical element to take into account when analyzing operations in air transport systems. From that moment on, I knew I wanted to proceed in this research direction for the final part of my masters. I feel very lucky that I had the chance to dive deeper in this fascinating field of research, and was able to combine my technical knowledge, obtained throughout the past years, with completely new knowledge from social sciences. I am very thankful for the great guidance of my supervisor Dr. Alexei Sharpanskykh who was always willing to take the time to discuss my work, to provide me with new insights and motivated me to investigate those topics I was least familiar with. Additionally, even-though a master thesis is an individual project, a great group was formed of like-minded people working on projects in the same direction. I would like to thank Arthur, Stef, and Matt for the pleasant collaboration, for all the feed back they provided me with and most importantly for making this past year just really fun. Furthermore, I could not have gone without the support and encouragement from my family and friends over the past year. I would like to give special thanks to my roommates, who were always there to help me relax and provide me with a healthy meal when I was working too hard to take time to cook, to my parents who never doubted my abilities to succeed in my journey at the TU Delft, and last but definitely not least, Michiel my wonderful boyfriend, who supported me with love, patience, and was always there to cheer me up.

Delft, February 2018

Contents

Nomenclature

List of abbreviations

AATOM Agent-based Airport Terminal Operations Model

ETD Explosive Trace Detection

IATA International Air Transport Association

LTS Labeled Transition Systems

PTS Probabilistic Transition Systems

RTHA Rotterdam-The Hague Airport

TTS Timed Transitions Systems

TTL Temporal Trace Language

WTMD Walk Through Metal Detector

Greek symbols

μ mean value

σ standard deviation

λ passenger arrival rate

Performance indicators

$p(sat)$ probability of saturation

$\overline{T}_{sat}$ average duration of saturation

$\overline{W}_{sat}$ average waiting time

$\overline{MF}$ average number of missed flights

TTS syntax

S a set of states

S_0 a set of initial states

A a set of action labels

$\Rightarrow$ a set of transition relations

AP a set of atomic propositions and clock constraints

L a set of labeling functions and state invariant assignment functions

1 Introduction

In the past few decades the global aviation industry has grown significantly. The European airport network experienced an average growth of over 5% in the past 5 years. This means that European airports only, have already welcomed an additional 300 million passengers since 2013, and this increase is starting to weigh on the capacity of the airport network [1].

The airport transportation system can be regarded as a high risk organization, as disruptions in such a system have a high potential negative impact. With the increase of passenger numbers, flights, and routes, it is expected that the airport transportation system becomes more vulnerable to disruptions, and the consequences of a disruptive event may become more severe. Therefor, there is demand for suitable methods to obtain insight in potential disruptive events and to develop strategies to deal with these disruptive events. A traditional method to analyze negative consequences of adverse events is risk assessment [2]. However, it is argued that this conventional method is not adequate to provide a complete picture of risk in today's complex socio-technical systems, such as the air transportation system. Risk assessment lacks the ability to take into account changing dynamics of a system and over-approximates the complex relations that exist in such a system. Additionally, risk assessment is not able to identify and quantify risk of unexpected events. Acknowledging these short comings, currently a shift in focus is observed, in research on high risk organizations, where one tries to understand how a system is able to maintain required performance under non-nominal conditions and how this ability might be improved, instead of focusing only on those moments when something goes wrong. This leads us to the concept of *resilience*.

One of the first notions of resilience was made by the ecologist C.S. Holling (1973) in his publication: 'Resilience and stability of ecological systems'. The traditional equilibrium-centred view on ecological systems explains system behaviour by its level of stability or instability. However, Holling observed many quite unstable ecological systems that were able to persist, while distinction of species was expected from an equilibrium-centred point of view. Upon this observation, he introduced resilience as an additional system property. With the introduction of this property he explained the unexpected behaviour of natural systems: "Resilience determines the persistence of relationships within a system and is a measure of the ability of these systems to absorb changes of state variables, driving variables and parameters and still persist."[3]

Since Holling's foundational definition in the ecological domain, the concept of resilience is further developed considerably in many research domains. However, the concept of resilience in the context of complex socio-technical systems is still relatively unexplored. This research aims to develop a better understanding of mechanisms present in complex socio-technical systems that underlie the ability to be resilient. To this end, a resilience mechanism is selected of which a formal model is developed based on conceptual grounds put forward in the current state of research on resilience. Additionally, this mechanism is analyzed through a computational agent-based simulation study of airport security operations.

This research is performed with the following structure: In Chapter 2, relevant theory regarding complex

socio-technical systems is presented. Additionally, resilience is introduced and the current state of research with regard to resilience in the context of complex socio-technical systems is discussed. Finally, anticipation is selected as focus for this research, and background theory on anticipation is provided. In Chapter 3 the academic challenge is presented along with the research objective and methodology. The remainder of this research is divided in three main parts. In Part I a formalization of anticipation as a resilience mechanism is developed. In Part II the operationalization of this resilience mechanism is presented by performing a case study of security operations in an airport terminal. In Chapter 5 the methodology for the case study is presented. The analysis of simulation results is presented in Chapter 6. In Chapter 7 the findings based on interviews with security managers at Rotterdam The Hague Airport are provided. Finally, in Part III conclusions are drawn based on the obtained knowledge and ideas for further research are discussed.

2

Literature Review

This chapter presents relevant theory regarding complex socio-technical systems, the concept of resilience, and anticipation as part of adaptive system behavior. The object of the literature review is to introduce the reader to the research topic, and to present the current state of research and corresponding research gaps.

2.1. Complex Socio-technical Systems

In a broad context, a system can be defined as a grouping of interacting and interdependent components, that form a unified whole. A system is characterized by its spatial and temporal boundaries, its structure and its functioning. In the present day of increasing complexity, many systems can be referred to as complex systems. Such multi-component systems exhibit complex behavior in space and time, through non-linear interactions, giving rise to emergent phenomena [4]. Complex systems exist in a wide variety and on multiple scales. For instance, a biological cell, the human brain, the internet, society, the financial market, and transportation systems, are examples of complex systems.

The term *emergent phenomena*, refers to unpredictable high-level system behavior that arises through low-level system interactions. A well known example of emergence is 'swarm behavior'. Swarm behavior appears in many natural systems, particularly in animal groupings. In Figure 2.1 a picture of bird flocking is presented. This form of swarm behavior is the result of collective motion of a large group of autonomous individuals that follow behavioral rules which do not involve any central coordination. The resulting patterns in the sky can be referred to as emergence. Furthermore, the ability to obtain these structured patterns, without central coordination, is referred to as self-organization.

The term *complex socio-technical system* refers to complex systems where technology plays a central role as does the social context within which its various components operate [5]. This means that human interactions, with each other and with technology, play a key role in the system. Generally, inclusion of human behavioral and performance variability, further increases system complexity and the unpredictability of emergent phenomena.

To study complex socio-technical systems is not a straightforward task, as difficulties arise with formal modeling of the systems. To develop a realistic representation of a complex socio-technical system, the modeling approach should allow emergent phenomena and self-organization to arise, as it would in the real system. However, emergence can not be predicted based on the system components only, and can even be counter intuitive. To capture emergence asks for a bottom-up modeling approach where only low-level interactions between individual system components are defined, and not the system as a whole [6]. Additionally, complex systems often contain adaptive characteristics. To model adaptive properties, means to be able to model dynamic system components that change their state and functioning over time, based on input from the environment or other system components. Furthermore, as in complex socio-technical systems humans are

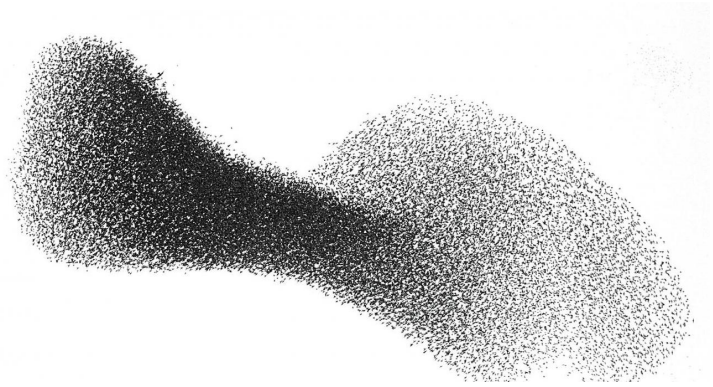


Figure 2.1: An example of swarm behavior: Bird flocking. The resulting patterns are referred to as emergent phenomena, and the ability to obtain these structured patters, without central coordination, is referred to as self-organization.

acknowledged as important adaptive system components, suitable modeling techniques should be applied to represent human behavior and interactions.

One of the modeling approaches that is recognized in literature as a suitable approach to model complex socio-technical systems, is *agent-based modeling and simulation* [5–9]. Agent-based modeling and simulation provides a bottom-up modeling approach and describes the system from the perspective of its constituents units. More specifically, a system is modeled as a collection of autonomous decision-making entities, called agents, that operate in an environment. An agent is capable of assessing its situation and acts accordingly. Hereby, decision-making and other internal processes can be modeled in great detail following cognitive science, or more high-level, based on a predetermined set of rules. The development of the agent-based modeling paradigm started around 1960. However, since it generally requires computation-intensive procedures, it did not become widespread until 1990. The relatively new paradigm contrasts more traditional top-down modeling techniques that often rely on approximations that ignore significant dependencies and interactions present in the system.

A complex socio-technical system can be modeled by defining agents for the different system components at multiple aggregation levels. Each agent is assigned specific characteristics and functioning based on its real-life operation. Apart from the fact that this provides a pragmatic manner to model the system structure, it also allows to represent heterogeneity present in the system. Additionally, already many models are developed, applicable for the agent-based modeling paradigm, that aim to model human behavior. Examples are: Algorithmic decision-making models [10], cognitive-based internal models [11–15], and human movement behavior models [16, 17].

Civil air transportation is an example of a large complex socio-technical system [5]. Airport terminals can be regarded as sub-systems of this large system, while being a complex socio-technical system by itself [18]. The relevance of studying this type of systems is to unravel the unpredictable effects of the many low-level interactions that exist in the system. Moreover, obtaining a better understanding of the behavior of a complex socio-technical system as an airport terminal, presents significant potential to support research of important performance aspects, such as: safety and security, efficiency and resilience.

2.2. Resilience

With the increasing complexity of our socio-technical systems, the challenge to deal with expected and unexpected disruptive events, becomes more complicated. Over recent years, the topic *resilience* has gained popularity and is regarded as a necessary concept to deal with the growing complexity of our socio-technical systems and risks that come with this growing complexity. The object of resilience can be described as to provide capacity to adapt, in order to keep the complex and inherently risky system within its functional limits [19, 20]. The concept of resilience is regarded as a state of the art topic and is not yet completely understood.

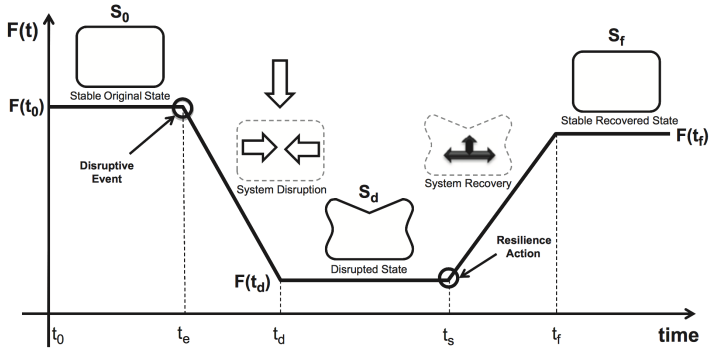


Figure 2.2: A graphical representation of system performance and state transitions as a result of disruption to describe the system resilience performance, by Henry & Ramirez-Marquez (2012) [25].

2.2.1. State of the Art

The main objective of each system is to operate at a predefined performance level and to successfully execute the task(s) it is designed for. However, a system is often challenged by disruptions and uncertainty, jeopardizing its performance. A traditional method to analyze the negative consequences of adverse events is risk assessment. Risk is a measure of potential loss, uncertainty about and severity of the consequences of a disruptive event [2]. As discussed in Chapter 1, it is argued that this conventional method is not adequate to provide a complete picture of risk in today's complex socio-technical systems. Risk assessment lacks the ability to take into account changing dynamics of a system and it over-approximates the complex relations that exist in such a system. Additionally, risk assessment is not able to identify and quantify the risk of unexpected events.

Resilience can be understood as the intrinsic ability of a system to adapt its functioning prior to, during, or following changes and disturbances, so that it can sustain required operations and performance under both expected and unexpected conditions [20]. Thus, instead of evaluating a system based on the possibility of something going wrong and the corresponding consequences, as with risk assessment, one focuses on how well a system is capable to deal with adverse events and uncertainty. This shift in reasoning is in line with the current transition from a Safety-I to a Safety-II perspective [21]. It is expected that assessment of resilience in complex socio-technical systems will provide a powerful tool to support traditional risk assessment [2, 19, 22, 23].

Interest in this state of the art topic is shown in several domains such as, infrastructure systems, organizational systems, safety management systems, socio-ecological systems, social systems, economic systems, and within the engineering domain [2]. As not yet a formal definition and metric is developed for resilience, a wide variety of resilience assessment approaches exists. A comprehensive review of definitions and measures of complex system resilience, applied in current research, is provided by Hosseini et. al (2016) [24]. The objective of most research is to quantify system resilience, which is generally done by comparing system performance in a nominal state to system performance after a disruption. In Figure 2.2 an example is provided from the work of Henry & Ramirez-Marquez (2012) [25]. The figure shows how the system performance evolves over time after a disruptive event. As with most quantitative metrics proposed, here three relevant system abilities are evaluated. Namely, the ability to absorb the effect of a disruptive event, the ability to adapt while being in a disrupted state, and the ability to recover. In Figure 2.2 it is measured how long it takes for a system to reach a disrupted state, what the performance loss is in this state, and how long it takes for the system to recover.

However, despite the attempts to develop resilience assessment methods, most of this research is system specific and not easily generalized. Additionally, it is recognized that adaptivity of a system plays a key role in resilience, as it underlies the ability to absorb, to adjust and to recover [26]. This leads to the question: What makes a system adaptive in the face of (un)expected events? This question is generally not answered, hereby neglecting responsible underlying system mechanisms and providing a simplistic and high-level resilience assessment [24]. Furthermore, human behavior and performance variability are identified as key-

contributors to both system challenges as the ability to adapt, which is limited accounted for in current research. It is encouraged, by recent research, to aim to develop better insight in the concepts that underlie the ability to adapt. Additionally, to be able to analyze these concepts for real-life complex systems, suitable complex system modeling approaches should be applied that are able to represent the underlying concepts, includes the essential human factors present in the system, and helps to unravel the complex relations in the system that lead to resilience [5, 8, 9, 20].

2.2.2. A Conceptual Foundation

In response to the above presented demand, Woods and Hollnagel are two well accepted researchers in the field of resilience, who both focus their work on the contribution to a general conceptual foundation for resilience. The objective of their work is to theorize conceptual ideas with regard to resilience of complex socio-technical systems, based on qualitative and philosophical grounds. Based on their work, three relevant topics are discussed which include the conceptual ideas that form the basis for this research:

- Patterns in how adaptive systems fail
- Generic resilience mechanisms
- Towards a metric for resilience: risk of saturation

Patterns In How Adaptive Systems Fail Over the past years, Woods studied what patterns can be observed while systems are challenged by expected or unexpected events and they adapt or move towards a failure point. The pattern shown in Figure 2.3, is referred to as the *decompensation signature*. It represents how decompensation of a system, due to a disturbance, evolves over two phases. The first phase, the compensation phase, shows how the system adapts partially successfully, to compensate for growing disturbances. However, underlying cascading disturbances are potentially masked by this compensatory control. The second phase, the decompensation phase, is initiated when control is not able to compensate the cascading disturbances completely, as the response mechanism's capacity becomes exhausted [27]. Woods explains this pattern in analogy with a well known example from the world of engineering. Namely, the relationship between *stress* —the load on a mechanical structure, and the resulting *strain* —how the structure stretches in response. Also for a stress-strain model a distinction is made between two phases. In the first phase the material is able to stretch proportional to the applied force. Then, if stress is further increased, the second phase is entered and deformation of the material will occur, until the breaking point is reached [28, 29].

Essentially, it is concluded that a pattern of adaptive failure of a system is characterized by a phase transition from a phase where the system is able to deal with posed demand, to a phase where demand exceeds the system's ability to match the posed demand. As a consequence, the system reaches a failure point, which is often characterized as the point where system performance is lost.

Generic Resilience Mechanisms Resilience of a complex socio-technical system is the result of adaptive behavior in the system in the face of (un)expected events. It is recognized that this adaptive behavior emerges through local mechanisms, operating at different aggregation levels of the system. These mechanisms are referred to as resilience mechanisms. Among mechanisms identified are: mechanisms of anticipation, control, coordination, and learning [23, 30, 31]. To answer the question: What makes systems adaptive in the face of (un)expected events? Hollnagel decomposes these generic resilience mechanisms and identifies four primary abilities that should be present in a system to enable resilience performance [22]. These are, the ability to:

- **To monitor:** The ability to monitor, means knowing what to look for, or being able to recognize what is or could seriously affect the system's performance in the near future. This includes monitoring own system performance as well as the environment.
- **To anticipate:** Anticipation is described as being able to make predictions to know what to expect, and being able to act on developments further into the future, such as potential disruptions.

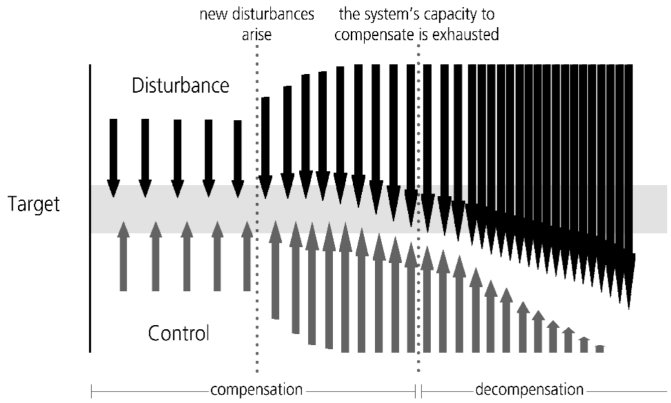


Figure 2.3: A basic decompensation signature by Woods (2012) [27].

- **To respond:** The ability to respond is described as being able to respond to regular and irregular changes, disturbances, and opportunities by activating prepared actions or by adjusting current mode of functioning.
- **To learn:** The ability to learn means, knowing what has happened and use this experience to be better prepared for disruptions in the future.

Despite the existing conceptual ideas with regard to mechanisms, responsible for resilience, there is not yet conducted research that aims to formalize generic resilience mechanisms and analyze these mechanisms for real-world socio-technical systems. Essentially, in order to gain a better understanding of what in a system enables resilience, this type of research is required. If a better understanding is developed of the underlying mechanisms of resilience, this enables the possibility to improve or design new mechanisms for real-world socio-technical systems.

The Theory of Graceful Extensibility The latest work of Woods "The Theory of Graceful Extensibility" (2018), presents a theory that aims to explain the contrast between successful and unsuccessful cases of adaptability in systems that serve human purposes [32]. This work builds further on ideas presented in his earlier work, and incorporates the ideas of Hollnagel on the primary abilities of resilience [22]. The concept of *graceful extensibility* is introduced as the opposite of system brittleness. Both brittleness and graceful extensibility refer to system behavior, while disruptions challenge its boundaries. Brittleness represents the rapid and sudden collapse of a system as the system is pushed beyond its boundaries for handling disturbances or variations. While, on the other hand, graceful extensibility refers to the ability of a system to extend its capacity when surprises challenge its boundary. It is hypothesized that systems with a high graceful extensibility, contain the ability to anticipate future adverse events such as bottlenecks, are able to learn from previous events, and are (continuously) in a state of readiness to respond to new challenges.

The proposed theory of graceful extensibility exists of ten theorems, which together capture basic generalities about adaptive capacity, which apply in all contexts. Three of these theorems focus on a conceptual definition of *risk of saturation*. To this end the concept of adaptive unit and adaptive capacity is defined. The first refers to a system unit which is able to adapt, the second refers to the capacity that corresponds to this unit to adapt. The first theorem defines adaptive capacity as being bounded, as each form of resources is finite. The second theorem states that each adaptive unit will be challenged at the boundaries of its adaptive capacity. The third theorem defines that risk of saturation appears when a system operates in its transition region. The transition region is defined as where a system moves from its base adaptive capacity to its extended adaptive capacity. Thus, all adaptive units present in a system have a corresponding adaptive capacity. If the system operates at the boundary region of the available adaptive capacity, risk of saturation is present and should be monitored and managed. Risk of saturation provides a measure that indicates how less-well a system is able to deal with (new) challenges while its adaptive capacity is being challenged, and thus what the risk is to enter a decompensation pattern and collapse.

Risk of saturation is proposed as an indicative measure for resilience. To operationalize this measure, system specific adaptive capacity should be identified, which is bounded by the finite resources available. Additionally, system specific performance threshold(s) should be identified to define when a system has collapsed. Finally, to measure risk of saturation the system should be challenged by anomalies and the adaptive capacity used by the system and the (potential) moment of collapse should be analyzed.

The concepts and ideas presented in this section, show potential to develop new control engineering approaches to measure and manage risk for complex socio-technical systems [33]. However, as discussed, most of the presented concepts are not yet formalized which is a necessary step in order to apply these concepts in real-world socio-technical systems. Formalization and operationalization of the presented conceptual ideas is identified as a significant research gap in the current state of research on resilience. To this end, this research will focuses on one of the proposed generic resilience mechanisms, and aims to develop a formalization of of this mechanism and operationalize the mechanism through a computational case study of airport security operations.

2.3. Anticipation

The complex socio-technical air transportation system is exposed to many internal and external disruptions on a regular basis. Disruptions can be of different nature and these events can interact with each other, potentially developing cascading effects throughout the system. A distinction can be made between exceptional disruptions, often with catastrophic effects such as with the 9/11 terrorist attack in 2001, and disruptions that occur on a regular basis, generally with less severe effects. A typical example of a common disruption is that of bad weather conditions. Often, bad weather jeopardizes normal operation of an airport which can result in a 'ripple' effect throughout the network. Management of such disruptions involves multiple parties such as, operators of airlines, airports and ATC, and often excessive costs are associated with the effects of these disruptions [5]. Generally, common disruptions and corresponding effects can be predicted more easily. Therefore, it is key that actors in the air transport system embody the ability to anticipate disruptions and provide adaptive solutions in order to ensure the system runs smoothly and effects of disruptions are minimized. For this reason, the resilience mechanism of anticipation is selected as focus for this research. The object of this section is to define anticipation and outline in what structures it exists.

An Anticipatory System Anticipation can be distinguished from prediction by the characteristic of functionality. Anticipation is regarded as functional, which means that with anticipation actual preparations for an expected event are undertaken. While making predictions does not go further than generating expectations [34]. A clear distinction between a predictive system and an anticipatory system, is drawn by the following two definitions [35]:

- **Prediction** is a representation of a particular event.
- **Anticipation** is a future-oriented action, decision, or behavior based on a prediction.

Rosen (1985) was one of the first to provide a comprehensive description of an anticipatory system [36]. He states that an anticipatory system is:

> *"a system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accordance with the model's predictions pertaining to a latter instant."*

This definition is further explained in a more formalized manner, as follows:

> *"An anticipatory system S_2 is one which contains a model of a system S_1 with which it interacts. This model is a predictive model: its present states provide information about future states of S_1. Further, the present state of the model causes a change of state in other subsystems of S_2: these*

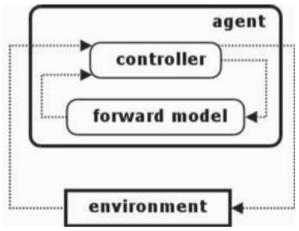


Figure 2.4: A schematic representation of how a forward model, an inner sensorimotor loop, is combined in parallel with the actual sensorimotor of the brain, to support the regulation of motor control through anticipation [35].

> *subsystems are (a) involved in the interaction of S_2 with S_1, and (b) they do not affect (that is, are unlinked to) the model of S_1. In general, we can regard the change of state in S_2 arising from the model as an adaption, or pre-adaption, of S_2 relative to its interaction with S_1. "*

This definition of Rosen is regarded as a form of explicit anticipation, whereby system behavior is produced based on explicitly formulated expectations. The ability to formulate predictions and use them to adapt, is what distinguishes an anticipatory system from a reactive system. This difference can be shown by writing down the behavioral pattern of each system type. The first behavioral pattern, shown below, describes a reactive behavioral pattern (S-A). The second describes an anticipatory system behavior pattern (E-A).

$$\begin{aligned} &\text{Reactive System} && \text{STIMULUS} \Rightarrow \text{ACTION} && \text{(S-A)} \\ &\text{Anticipatory System} && \text{STIMULUS} + \text{EXPECTATION} \Rightarrow \text{ACTION} && \text{(E-A)} \end{aligned}$$

The expectations for an anticipatory system are generally generated following one of the two presented methods below. Here, the first method shows that the expectation is based purely on the stimulus (S-E), while the second method shows that the expectation is a result of including expected outcome(s) of potential action(s) (S-A-E). For the second method, the anticipatory mechanism includes a *forward model* in the loop, that permits to generate predicted action effects [35].

$$\begin{aligned} &\text{Method 1.} && \text{STIMULUS} \Rightarrow \text{EXPECTATION} && \text{(S-E)} \\ &\text{Method 2.} && \text{STIMULUS} + \text{ACTION} \Rightarrow \text{EXPECTATION} && \text{(S-A-E)} \end{aligned}$$

Cognitive Anticipation Several research is performed to find structures in how the brain functions and how it is able to anticipate. An example is research that focuses on what structures are applied for motor control of actions of the body. Here, same structures are proposed as presented in the previous section, where forward sensory models generate expectations about the next sensed stimuli, given the current state and motor command of the body and brain. A schematic representation of such a model is presented in Figure 2.4. The internal forward model works in parallel with its actual sensorimotor interaction, and functions as support to regulate motor control [35].

Hesslow explains the working of a forward model, following his simulative theory of cognitive function. This theory is set around three postulates with regard to brain functioning [37]:

1. Behavior can be simulated by activating motor structures, as with overt action, but now while suppressing its actual execution.
2. Perception can be simulated by internal activation of the sensory cortex, in the same manner as with normal perception of external stimuli.
3. Both, overt and covert actions will elicit perceptual simulation of their normal consequences.

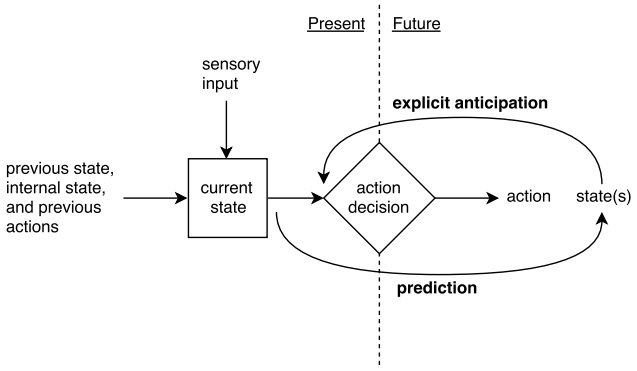


Figure 2.5: A schematic representation of state-based anticipation. Here, decision making is directly influenced by future state predictions. The figure is reproduced based on the work of Butz et. al [39].

Summarized, in these three postulates, lays the assumption that the brain is capable of anticipation, just by imagining a certain action is executed by itself or something else in the environment. Then, the generated perception of the effect of these actions is processed to be able to decide on what action to take.

Applying forward simulation as part of anticipation can be performed both on-line and off-line. For instance, anticipating sensory consequences of action effects as described above, is an example of on-line anticipation. Here, the predictions are generated parallel to the sensory input, so the prediction can be matched against the sensory input. Often, the object of on-line anticipation is to support regulating behavior. The object of off-line anticipation is to generate long term predictions, and is not related to the current sensorimotor cycle. An example of simulative planning is when a driver chooses what route to take to get to its destination. To make this decision the driver will perform forward simulation to anticipate the effects of the different route choices and then selects the best option [37].

The work of Hesslow has inspired the development of artificial agent models that allow adaption and anticipation similar to how this occurs in the human brain [37, 38].

State-based Anticipation An anticipation model of an artificial agent that follows a state-based representation, and whereby the behavior of the agent is influenced by explicit future state representations, is referred to as state-based anticipation [39, 40]. To generate future state representations, a predictive model of the environment and/or other agents in the system should be available to an agent. A schematic representation of state-based anticipation is presented in Figure 2.5. The figure shows that the current state of an agent is determined by its previous state(s) and action(s). Additionally, sensory input can contribute to this state, and potentially triggers the generation of predictions about future state(s). By explicit anticipation of future states decisions about actions are made. Also for state-based anticipation, forward simulation can be applied. In this case, the decisions are based on predictions of all possible behavioral consequences and the utility of the predicted results [39].

2.4. Conclusions

In the literature review presented in the previous sections, first characteristics of complex socio-technical systems are discussed and agent-based modeling is introduced as a suitable modeling approach to represent complex socio-technical systems. In the following section the current state of research with regard to resilience is presented and a research gap is identified. As a response to this research gap, this research will focus on anticipation, which is identified as a resilience mechanism. To this end, in the last section theory on anticipation is presented. The reasoning logic of the literature review is summarized in Figure 2.6.

To study resilience as concept to deal with adverse events in complex socio-technical systems has gained significant popularity in the past decade. Despite the attempts to develop definitions and metrics to describe resilience, no formal definition for resilience for socio-technical systems is established yet. It is acknowl-

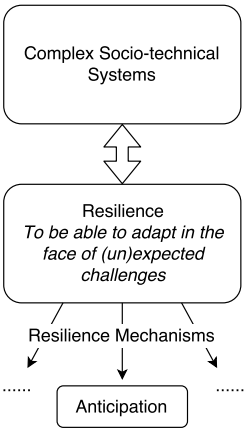


Figure 2.6: The reason logic of this literature review summarized.

edged in literature that there is demand for research which focuses on a theoretical basis of adaptive capacity and its mechanisms, and on the development of suitable computational models that can be used to analyze resilience in complex socio-technical systems. To develop a fundamental understanding of resilience is deemed an essential step to be able to design resilient systems and organizations.

Based on the current state of research the following research gaps are identified:

- To develop formal representations of generic resilience mechanisms, and to analyze these mechanisms in real-world socio-technical systems.
- To operationalize generic resilience mechanisms in complex socio-technical systems through computational analyses, whereby suitable complex modeling approaches are applied. Agent-based modeling is identified as a suitable modeling approach for complex socio-technical systems.
- To operationalize the concept of *risk of saturation*, proposed by Woods, as an indicative measure for resilience.

Based on the current state of research and the identified research gaps a research objective is formulated which is presented in Chapter 3.

3

Research Objective and Methodology

3.1. Research Objective

As presented in Chapter 2, the current state of research, concerning resilience in complex socio-technical systems, presents a conceptual foundation for resilience. The conceptual ideas aim to answer the question: What makes a system adaptive in the face of (un)expected events? It is recognized that a socio-technical system contains resilience mechanisms that are responsible for adaptive behavior. Additionally, ideas are developed about the characteristics of system patterns that are shown when a system moves towards an adaptive failure point, and risk of saturation is proposed as an indicative measure for resilience. Based on the current state of research, it is concluded that there is demand for research that aims to formalize these conceptual ideas and operationalizes these concepts in real-world socio-technical systems. Additionally, it is recognized that agent-based modeling and simulation provides a suitable modeling approach for the analysis of complex socio-technical systems, and it is proposed to apply this type of modeling when studying resilience. In response to the posed demand, this research studies the resilience mechanism of anticipation in a complex socio-technical system, by applying an agent-based modeling and simulation approach.

The first goal of this research is to develop a formal model for anticipation in the context of complex socio-technical systems, which allows reasoning about system behaviour using mathematical logic while both quantitative and qualitative relations are represented. The second goal is to study the effects of operationalizing anticipation in a real-world socio-technical system through a simulation study. To this end, the airport security check-point system is selected for a case study. In an airport terminal, each departing passenger is obliged to pass through the security check-point before it is allowed to enter its flight. For airports, the security check-point is the most significant bottleneck of the airport terminal, especially under non-nominal conditions. This is for instance reflected by the long queues experienced during holiday periods at Schiphol Airport. In May 2017, for two weeks, line capacity at the security check was exceeded due to the high number of Spring break passengers. As a result, many passengers missed or delayed their flight [41]. For an airport it is of great importance that passengers are processed efficiently and reach their flight in time, as high costs are associated with delayed departure of flights. For this reason, security check-point operations provides an interesting case to study resilience.

The objective of this research is:

> To contribute to the relatively unexplored research field of resilience in complex socio-technical systems by **developing a formalization of anticipation** and **operationalize this resilience mechanism** in a simulation study of airport security operations, in the face of unplanned and challenging passenger demand, **by applying an agent-based modeling approach**.

The research objective presented is two-fold. To reach the first objective, to develop a formalization of antic-

ipation, answers are sought to the following research questions:

1. How to develop an agent-based architecture to represent an anticipation mechanism?
2. What formal language to select to formalize adaptive mechanisms of a socio-technical system, that follow an agent-based architecture, such as the anticipation mechanism?
3. How to develop a formalization of the agent-based conceptual anticipation model, applying the selected formal language, which allows to reason about the model behavior using mathematical logic?

In order to reach the second objective, to operationalize this resilience mechanism, an answer is sought to the following research question:

2. How to apply an agent-based modeling approach to study the effects of anticipation as a resilience mechanism, in airport security operations?

Finally, based on the overall insights obtained throughout the research, implications for industry and academia are formulated by answering the following questions:

4. Based on the results obtained by the case study, what insights are developed relevant for airport terminal security operations?
5. How does the formalization and operationalization of anticipation contributes to the field of research on resilience in complex socio-technical systems?

3.2. Research Methodology

For this research the following research methodology is followed:

1. **Understanding the State of he Art:** To study resilience in the context of complex socio-technical systems is a relative young field of research, whereas attempts to provide formal and quantitative analysis on this matter generally assume simplistic performance-based metrics and do not take into account what makes a system resilient. On the other hand, some research focuses on developing more in-depth insight in the mechanisms behind resilience. However, the conceptual ideas brought forward are not yet further studied through formalization and operationalization of these concepts. For this reason, the first step is to establish an understanding of the conceptual grounds of resilience developed in literature so far, and identify what type of research is required to contribute to this field of research. In Chapter 2 the current state of research is presented and research gaps are identified.
2. **Selection of Resilience Mechanism:** The current state of research motivates to conduct in-depth research of resilience mechanisms by formalization and operationalization of these mechanisms for complex socio-technical systems. To this end, anticipation is selected as focus for this research. The air transportation system is exposed to disruptive events on a regular basis and the ability to anticipate provides a key mechanism to ensure that operations run smoothly and effects of disruptions are minimized. The case study selected, to study anticipation, concerns the security operations at an airport. The ability to anticipate disruptive events that might influence the efficiency of the security process is a necessity, as the consequences of congestion at the security check-point can easily propagate through the network and are often associated with high costs.

Formalization of Anticipation As described, this research exists of two main parts, first a formal model for anticipation is developed.

3. **Selection of Modeling Paradigm:** To be able to represent anticipation as part of human adaptive behavior or for artificial intelligence, principles from neuroscience provide an important source of inspiration to model the processes corresponding to anticipation. To describe these processes it is required to express both quantitative and qualitative relations. As the agent-based modeling paradigm allows to represent quantitative and qualitative relations, and is able to fit such a cognitive model within a larger socio-technical system environment, this modeling paradigm is selected to develop an anticipation model.

4. **Development of Conceptual Model:** A conceptual model for anticipation is developed, inspired by the cognitive simulative theory of Hesslow [37], following an agent-based modeling architecture based on the work of Blumberg [42], Hoogendoorn [43] and Reynolds [44].

5. **Selection of Formalization Syntax:** To be able to develop a formal representation of the working of the anticipation mechanism, a formalization syntax is to be selected. A set of desiderata, for formal modeling of the agent-based conceptual anticipation model, is considered. With these desiderata in mind, the high level abstract modeling syntax *Timed Transition Systems* (TTS) is selected.

6. **Development of Formal Model:** TTS is applied to develop a formal representation of the dynamics of the anticipation model and its interactions with the environment, in the context of a complex socio-technical system.

Operationalization of Anticipation For the second part of this research the main goals is to illustrate how, by means of a simulation study, the resilience mechanism of anticipation can be analyzed in the context of a real-world socio-technical system. To this end, a case study is conducted where anticipation is operationalized in the context of airport security operations. A comprehensive methodology is presented in Chapter 5. Here, the high-level steps are outlined:

7. **Development of an Agent-based Airport Terminal Operations Model (AATOM):** As part of this research time is dedicated to the development of AATOM, an agent-based simulation tool of airport terminal operations. This is done in collaboration with PhD student S.A.M. Janssen and MSc student A.Knol. The object of this simulation tool is to provide the possibility to perform experiments in an airport terminal system through computational simulations. The advantage of such a simulation tool, is that it becomes easier to conduct all sorts of research with regard to airport terminal operations, which is normally hard to conduct in real-life, as the system is constantly in operation and often safety and security constraints apply. The contribution delivered to the development of AATOM as part of this research is outlined Appendix B.

8. **Selection of Case Study:** Within the scope of airport terminal operations, the security check-point system is selected for the case study. Since the security check-point can develop into a significant bottleneck with regard to passenger flows in an airport terminal, it is of interest to study the effects of the ability to anticipate disruptive events, on the performance of the system.

9. **Development of a Security Operations Model:** The baseline model AATOM is adjusted and calibrated to fit the research objective of the case study. First a model specification of the security operations model is developed. The model is based on the security operations of Rotterdam The Hague Airport (RTHA). Based on available input data from RTHA the model is initialized and calibrated.

10. **Development of an Anticipation Model for Security Check-point Operations:** Based on the conceptual and formal model developed in Part I, a formal anticipation model is implemented for the case study scenario.

11. **Model Implementation:** The security operations model including the anticipation model are implemented in Java in the simulator environment. The model is analyzed through verification and validation steps.

12. **Analysis:** Computational analysis of the case study scenario is conducted through simulations. The obtained data is processed and final results are analyzed.

13. **Evaluation with RTHA:** Finally, interviews are conducted at Rotterdam The Hague Airport for validation of the results. Additionally insight is obtained in the current status of resilience for the security check-point operations, and in potential future research topics relevant to the airport in the context of resilience.

I

Formalization of Anticipation

4

Development of A Formal Foundation for Anticipation

In this chapter the formalization of anticipation is presented. First, in Section 4.1, the concept of anticipation is introduced. In Section 4.2 a conceptual model for anticipation is presented. The syntax selected for formalization is introduced in Section 4.3. Finally, a formalized model for anticipation is developed in Section 4.4.

4.1. Introduction

In the context of complex socio-technical systems, anticipation is identified in literature as a generic resilience mechanism. The ability to anticipate is understood as the ability to know what to expect, and being able to act on developments further into the future, such as potential disruptions. As presented in Chapter 2, an anticipatory system is explained by Rosen (1985) as follows:

> *"An anticipatory system S_2 is one which contains a model of a system S_1 with which it interacts. This model is a predictive model: its present states provide information about future states of S_1. Further, the present state of the model causes a change of state in other subsystems of S_2: these subsystems are (a) involved in the interaction of S_2 with S_1, and (b) they do not affect (that is, are unlinked to) the model of S_1. In general, we can regard the change of state in S_2 arising from the model as an adaption, or pre-adaption, of S_2 relative to its interaction with S_1. "*

This definition explains anticipation as an interaction between a system and its predictive model which leads to an adaption in the system.

To be able to represent anticipation as part of human adaptive behavior or for artificial intelligence, principles from neuroscience provide an important source of inspiration to model the processes corresponding to the ability to anticipate. The core processes for anticipation are: being able to predict consequences of action options and valuate these predictions to make a decision on what action to perform. As discussed in Chapter 2, Hesslows simulative theory puts forward the notion of internal simulation of the brain. The idea of internal simulation is that sensory representation states, such as mental images, can be activated by (external) triggers. In response, action options are generated internally, followed by generation of perceptions of the effect of these actions through prediction. These perceptions are thus generated without actually having to execute the actions. Based on valuation of the action options, decisions can be made on what action(s) to execute [37].

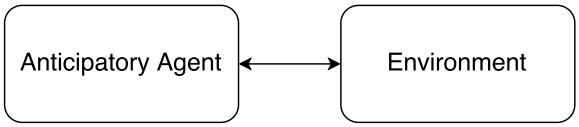


Figure 4.1: A schematic representation of the interaction between an anticipatory agent and its environment.

In order to study anticipation, in the context of complex socio-technical systems, this chapter aims to develop a conceptual and formal model for anticipation mechanisms inspired by theoretical principles from neuroscience. The formalization of the model allows to obtain a systematic representation of the working of the anticipation mechanism in its environment. Additionally, to actually analyze the dynamics of anticipation in a system, the formal model is translated into a computational model and a simulation study is conducted in Part II of this research.

4.2. A Conceptual Anticipation Model

Anticipation can be performed at different aggregation levels of a system. For this research, focus is aimed at adaptions, due to anticipation, that can be represented as an interaction between an anticipatory system and the environment in which this system operates. This is often encountered for anticipation on an organizational level, and excludes anticipation on the individual level, where anticipation may be applied for implicit coordination, and adjustment of one's own functioning.

For a conceptual model of anticipation, both quantitative and qualitative relations should be represented. An agent-based modeling approach fits well with this need, as it allows to define quantitative and qualitative relations in a model. To this end, an anticipatory system can be represented as an autonomous agent interacting with its environment. The environment includes all system aspects and agents present in the system that are accessible to the anticipatory agent for interaction. In Figure 4.1 a schematic representation is provided of the model interaction(s).

Inspired by the work of Blumberg [42], Hoogendoorn [43] and Reynolds [44] on internal architectures for artificial agents, and inspired by the simulative theory of Hesslow [37], a conceptual internal model for the anticipatory agent is developed following a two-layered architecture, see Figure 4.2. The *operational layer* is the lower layer which is responsible for interactions with the environment. This includes, making observations and interpret these observations, and to execute actions in the environment. The higher layer, the *strategic layer*, is responsible for updating the belief module based on observations and reasoning. Reasoning includes making predictions, analyze these predictions and make decisions based on the analysis. Additionally, actions executed by the operational layer are generated by the strategic layer.

The model presented in Figure 4.2 represents the aspects required for anticipation as modules. Each of the modules presented and the relations between the modules are explained below.

- **Perception and Interpretation Module:** This module ensures that the agent is able to observe the environment. Based on these observations the belief module is updated, this way the agent is aware of the environment and potential changes in the environment that trigger anticipation.
- **Belief Module:** The belief module contains the knowledge of an agent. This can be in the form of facts, rules and information about (causal) relations. This knowledge is constantly updated by input from observations, resulting in awareness of the environment. Consequently, an update of the belief module can result in activation of the reasoning module. The other way around is the belief module also updated based on output from the reasoning module such as, future state predictions, outcomes of analysis and outcomes of decision-making.
- **Reasoning Module:** The reasoning module, which is fed by the belief module can be explained as a machinery including the processes of prediction, analysis and decision-making. The collaboration between the belief module and the reasoning module represents the prediction model of the anticipatory agent. Following Hesslows simulative theory, the prediction model generates internally simulated

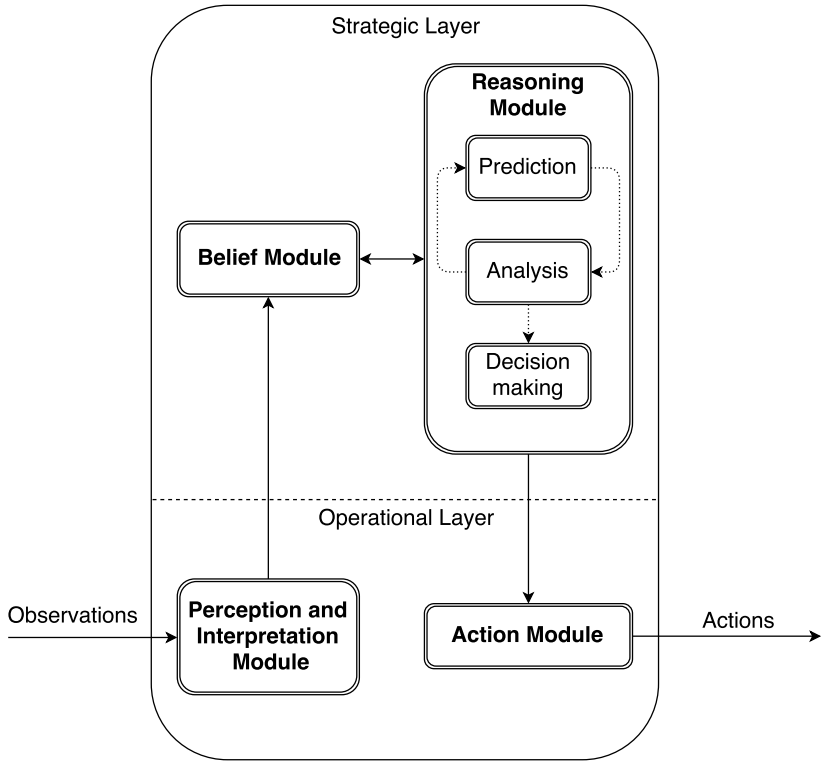


Figure 4.2: A schematic representation of the two-layered internal model of an anticipatory agent.

(chains of) interactions with the environment and evaluates these simulated interactions. This forward internal simulation and evaluation is performed by the prediction and analysis module. Additionally, to come to a decision on what action to execute, the obtained information is processed by the decision module.

- **Action Module:** The action module is responsible for the actual execution of the action selected. The action execution is an interaction with the environment.

It is assumed that the internal relations between the processes can be described following a sequential order, whereby the dynamics of the internal relations are described as internal state transitions. Additionally, it is assumed that interactions between the anticipatory agent and environment can be described as input and output state transitions of the agent, while the environment's state-space evolves over time or changes due to interactions with the anticipatory agent.

4.3. Selection of Formalization Syntax

To be able to formally model the internal dynamics of the anticipatory agent, and its interactions with the environment in the context of a complex socio-technical system, a suitable modeling syntax is required. The main reason to develop a conceptual anticipation model through an agent-based modeling approach, is that it allows to represent both quantitative and qualitative relations. For instance, to model the dynamics of reasoning, states are generally characterized by qualitative state properties that change through discrete state transitions. On the other hand, real-valued state variables may be defined as well, that change over (continuous) time. To model both types of relations with one modeling language excludes the possibility to describe the model based on differential equations only, or to apply purely logic-based models [45]. Hence, to be able to formalize these relations the modeling syntax should allow to capture both qualitative and quantitative aspects, such a syntax is generally referred to as a hybrid language [46]. Additional desiderata for modeling either single-agent or multi-agent dynamics, is that dynamics can be described at different aggregation levels of the system. This asks for a high expressiveness of the language such that a distinction can be made between

input, internal, output and external state properties. Additionally, the syntax should allow to model processes or sub-systems that operate in parallel and represent the relations between these parallel processes.

With the posed desiderata in mind, *timed transition systems* (TTS) syntax is introduced as a formal modeling language. This syntax is a high-level generic modeling language based on well-defined mathematical semantics, and considers system dynamics as state transitions [47][48]. TTS is based on transition system syntax, whereas for TTS, the notion of real-time is included which allows to represent temporal-logic of the system that is modeled. The language meets all desiderata posed for formal modeling of the dynamics of an agent-based model. Especially, due to the fact that TTS provides a high-level generic language, expressions can be added which allows to represent qualitative aspects and relations. Additionally, TTS syntax allows to represent each sub-system or agent as a timed transition system. A concurrent composition of the timed transition systems can be composed as such that the overall system state is composed from the states of the individual processes. This allows to develop, for instance, a representation of a multi-agent system or to represent parallel processes in a single-agent system. Finally, TTS allows model verification of its state traces by means of a model checking environment such as, the Temporal Trace Language (TTL) checker [49].

In general, the behaviour of a timed transition system is defined as a 6-tuple: $TTS = (\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$. The system is described by a set S of (possibly infinite) states, where for each state the notion of time is defined through a clock valuation function V. The initial state is represented by S_0. Transitions of states occur either due to an action, which is referred to as a discrete transition and is labeled by an action label $\in \mathsf{A}$, or transitions occur over time, which is referred to as a delay transition. A transition relation is denoted by $\Rightarrow$. Atomic propositions can be defined that describe state properties. Additionally, clock constraints can be ascribed to states or transitions to be able to define timing of state transitions. The set, containing the atomic propositions and clock constraints, is defined as AP. Finally, atomic propositions and clock constraints are assigned to specific states through labeling and state invariant assignment functions $\in \mathsf{L}$.

In this section the TTS syntax is briefly introduced, a more comprehensive description of the syntax and its semantics is provided in Appendix A. Based on the TTS syntax selected for formalization, a formal model framework for anticipatory systems is proposed in the next section.

4.4. Formalization of Anticipation

The formalization of anticipation aims to develop a formal representation to describe anticipatory behaviour in complex socio-technical systems. The representation is based on transition system syntax as introduced in the previous section. For each sub-process in the system, e.g. each sub-system or agent in the system, a timed transition system can be defined. The transition systems can be composed to a concurrent composition, and the overall system state is composed from the states of the individual processes. Specifically, it is expected to obtain a parallel composition of all transitions systems defined for the complex system:

$$System = TTS_1 || TTS_2 || ... || TTS_n \quad \text{with } n \text{ the total number of sub-systems defined}$$

For this research, as described in Section 4.2, it is assumed that anticipation in a complex socio-technical system can be represented by the interaction between an anticipatory agent and the environment. The anticipatory agent contains a predictive model of the environment and is able to make predictions about the future states of the environment, based on internal forward simulation. To capture the dynamics of anticipatory behaviour, both the environment and anticipatory agent are represented as a TTS operating in parallel. A TTS representation of the environment provides external state information to the anticipatory agent. Based on the awareness of the environment, the anticipatory agent feeds its predictive model. Additionally, it is essential to capture the environment state transitions due to adaptive adjustments made by the anticipatory agent operating in it. For the anticipatory agent, the TTS representation describes the evolution of input, internal, and output states. Input states are defined by interactions between the agent and the environment based on observations. Internal states describe the internal processes of the agent, as presented in the conceptual model in Figure 4.2. The output states are defined by interaction between the agent and environment based on actions.

A TTS representation for the environment

The environment of the anticipatory agent can be described as a 6-tuple $(\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$, which is referred to as TTS_1. How each of its aspects is defined, is outlined below.

Time Representation For this system one clock variable is defined, $C = \{x\}$, representing the real time in the system. Here, the clock valuation function reduces to $\mathsf{v} : x \rightarrow \mathbb{R}^+$ which assigns the current clock value $\mathsf{v}(x)$, and the set of clock valuations V reduces to one clock valuation v for clock x. As the state evolution of the environment over real-time is of interest, the set of clocks to be reset (λ) is defined to be empty for all transition relations ($\Rightarrow$), $\lambda = \emptyset \quad \forall \Rightarrow$. A visual representation of clock x is presented in Figure 4.3. The time-trace represents a dense time set, where the clock value is determined by the valuation function, and is denoted by $\mathsf{v}(x)$.

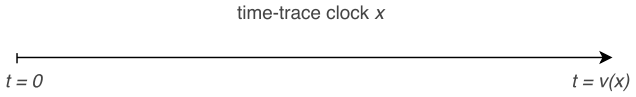


Figure 4.3: A visual representation of the time-trace of clock x, representing a real-time dense time set underlying an execution run.

As the environment (TTS_1) and anticipatory agent (TTS_2) operate in parallel, it is defined that the clock of the environment and the clock of the anticipatory agent are synchronized.

State Representation The environment state representation is as follows:

$$\mathsf{S} = S \times V$$

Here, S represents the set of states that make up the TTS. Each state is defined by its state properties expressed as atomic propositions that are assigned to a state by the labeling functions. Additionally the state is defined by $V = v(x)$, which represents the valuation of clock x, thus the moment in time. Further, the initial state is formulated as:

$$\mathsf{S}_0 = S_0 \times V_0$$

Here, S_0 is described by all state conditions that hold at the beginning of the time trace, hence, $V_0 = 0$.

Events Representation For the environment, we define that a state transition occurs due to event(s) in the environment. Here a distinction is made between events that occur due to an action executed by the anticipatory agent in the environment, and events that occur due to actions of other agents or external influences. For the first type of event, an explicitly labeled state transition is defined with an action label corresponding to the action label of the anticipatory agent at the moment of execution. For the second type of event, state transitions are represented as unlabeled transitions over time. This is because it is generally not possible to obtain a complete specification of all events that may happen in the environment of an open system. To this end, the set of events is represented as:

$$\mathsf{A} = A \cup \tau \cup \mathbb{R}^+$$

Here, A represents the set of labeled actions that can be executed by the anticipatory agent in the environment. τ represents all events that take place in the environment leading to discrete state transitions, and are not explicitly represented. Finally, the time delay posed on a delay transition is a non-negative real number.

Atomic Propositions and Clock Constraints The union of the set of atomic propositions AP and the set of clock constraints $\mathsf{B}(\mathsf{C})$ is defined as follows:

$$\mathsf{AP} = AP \cup \mathsf{B}(\mathsf{C})$$

The atomic propositions $ap \in AP$ are to be defined to quantify or describe those aspects of the environment state that should be observable for the anticipatory agent. For instance, the performance of the system, demand posed on the system or other characteristics that are required for the predictive model of the anticipatory agent, to be able to predict future states of the environment. The set clock constraints can be used to define when specific state transitions occur in the environment that are purely time dependent. For instance, state transitions in the environment that are based on a planning or schedule.

Labeling Functions and State Invariant Assignment Function The definition combining both labeling functions for atomic propositions and the state invariant assignment functions is formulated as follows:

$$\mathsf{L}(s, v) = L(s) \cup \{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$$

Here, $L(s)$ represents the set containing all labeling functions, and the relation $\{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$ defines for each state whether the clock guard $\theta \in \mathsf{B}(\mathsf{C})$ holds, which is the case if the satisfaction relation holds ($v \models \theta$). Atomic propositions and clock constraints defined for an environment are assigned to specific states following these functions.

Transition Relations In general, two types of transitions can be defined: a discrete transition due to an action and labeled by an action label, and a delay transition, which is an increase of the clock value by a non-negative real number, while being in a specific state. For the environment we specify the following types of transitions:

- A discrete transition explicitly label by an action label drawn from A: $(s_i, v) \xrightarrow{\alpha} (s_{i+1}, v)$.
- A discrete transition due to events in the environment that are not specified explicitly. To represent such a discrete transition the label τ is applied: $(s_i, v) \xrightarrow{\tau} (s_{i+1}, v)$.
- A delay transition over time δt: $(s_i, v) \xrightarrow{\delta t} (s_i, v + \delta t)$.

For the environment representation the following short-cut notation is introduced:

$$\ldots \xrightarrow{\delta t} \xrightarrow{\tau} (s_i, v) \xrightarrow{\delta t} \xrightarrow{\tau} (s_{i+1}, v + \delta t) \xrightarrow{\delta t} \xrightarrow{\tau} (s_{i+2}, v + 2\delta t) \xrightarrow{\delta t} \xrightarrow{\tau} \ldots$$

The double arrow notation allows to specify the environment state after each time step δt, while discrete state changes occur due to non-explicitly represented events.

A TTS representation for an anticipatory agent

The anticipatory agent can be described as a 6-tuple $(\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$, which is referred to as TTS_2. How each of its aspects is defined, is outlined below.

Time Representation The same time representation, as defined for the environment, holds for the anticipatory agent. One clock variable $C = \{x\}$ is defined which is synchronized with the clock defined for the environment.

State Representation An extended state representation is proposed:

$$\mathsf{S} = S \times V \times Asp$$

Here, S represents the set of states that make up the TTS. V represents the valuation for clock set $C = \{x\}$ at every state, and Asp is introduced as the set of labels: $\{input, internal, output\}$, which denote whether a state is an input, internal, or output state of the system. By specifying the different types of states, one can distinguish between whether state information corresponds to the internal processes of the agent or is a result of interactions with the environment.

Further, the initial state is formulated as:

$$\mathsf{S}_0 = S_0 \times V_0 \times Asp$$

Here, S_0 is described by all input, internal, and output state conditions that hold at the beginning of the time trace, hence, $V_0 = 0$.

Action Representation For the anticipatory agent a state transition is either a discrete transition, based on an *action* and represented by an *action label*, or a delay transition due to a time delay. An action corresponding to a discrete transition is drawn from the set of action labels ($\in A$). The time delay posed on a delay transition is a non-negative real number. The union of the set of action labels and real positive values is defined as follows:

$$\mathsf{A} = A \cup \mathbb{R}^+$$

Additionally, the set of action labels exists of actions executed internally, or is as input or output actions. The sets of internal actions, input actions and output actions are defined as being disjoint.

$$A = A^{inp} \cup A^{int} \cup A^{out}$$

For an anticipatory agent the following input, internal and output actions (act) can be defined:

- Input actions $act \in A^{inp}$ are defined as observations the anticipatory agent can do of the environment state. The state information of the environment is described by the set of atomic propositions (AP_1) that hold in a state. Here, the subscript 1 referrers to TTS_1 (the environment specification). Thus the anticipatory agent is able to observe atomic propositions of the environment ($ap \in AP_1$) (accessible to the agent) that hold in a state s:

  $$obs(holds(ap))$$

  Each observation of an atomic proposition $ap \in AP_1$ that holds in state s, is defined separately as an $act \in A^{inp}$. Additionally, for the anticipatory agent a set of environment atomic propositions can be specified $AP_1' \in AP_1$ that will trigger the anticipatory agent to anticipate. For instance, if the anticipatory agent is focused on demand posed on the system, and will be triggered to anticipate when demand exceeds a specific threshold, all demand values above this threshold are described by (quantitative) atomic propositions ap part of set AP_1'.

- Internal actions $act \in A^{int}$ are defined as actions part of the internal process of the anticipatory agent, following the internal model presented in Figure 4.2. Each of the sub-processes part of the prediction model are defined as internal actions:

  – For the internal action of prediction of future environment states, two types of predictions are defined. The first type is a prediction generated as a reaction to an observation of $ap \in AP'_1$, this prediction is referred to as the initial prediction. The second type of prediction is a prediction of the consequences of a potential action option. The prediction types are denoted as:

$$prediction(ap)$$
$$prediction(opt)$$

  Here, $ap \in$ denotes the observed atomic proposition that holds in the environment and triggers the agent to anticipate ($ap \in AP'_1$). Further, opt denotes the option that is evaluated in the prediction. An option represents an action (or a set of actions) that can be executed in the environment. If the internal prediction action is executed, a prediction is generated following the prediction model of the anticipatory agent. The prediction model generates simulated chains of interactions with the environment and evaluates the simulated interactions. The simulation of interaction chains evaluate the future environment states based on knowledge of the agent about its own functioning and the system it operates in, such as (causal) relations. This knowledge is referred to as the $belief$ of the agent. For the prediction model, simulation of interaction chains is represented as follows:

$$\frac{belief(ap/opt) \wedge belief((b_1^1) \wedge (b_2^1) \wedge ... \wedge (b_j^1))}{pred(p1)} \Rightarrow$$

$$\frac{pred(p1) \wedge belief((b_1^2) \wedge (b_2^2) \wedge ... \wedge (b_j^2))}{pred(p2)} \Rightarrow$$

$$\frac{pred(p2) \wedge belief((b_1^3) \wedge (b_2^3) \wedge ... \wedge (b_j^3))}{pred(p3)} \Rightarrow \tag{4.1}$$

$$..... \Rightarrow$$

$$\frac{pred(p_{i-1}) \wedge belief((b_1^i) \wedge (b_2^i) \wedge ... \wedge (b_j^i))}{pred(p_i)}$$

  Here, the first input argument for $belief$ depends on the type of prediction that is executed. For the initial prediction the first term in the chain is $belief(ap)$, for a prediction evaluating an action option the first term in the chain is $belief(opt)$. Based on the knowledge available about the observation $belief(ap)$ or an option $belief(opt)$ option and based on all required information to create a prediction $belief((b_1^1) \wedge (b_2^1) \wedge ... \wedge (b_j^1))$, a prediction is generated $pred(p1)$. This prediction is referred to as a 'first step prediction' which can represent the prediction for a specific time step into the future, the prediction of the consequences of executing a sub-action corresponding to the action option evaluated, or both. Essentially, a prediction p contains information about relevant future environment state properties. This is represented by the following mapping function:

$$pred : ap_1 \times ap_2 \times \cdots \times ap_k \times v(x) \rightarrow p$$

  Here, ap_1 until ap_k represent the state properties that hold, and $v(x)$ denotes the clock valuation for the corresponding state. These values are mapped to the corresponding state prediction p.

Based on the result of the first step prediction and information required to generate a next prediction, the second prediction is generated. This process continues until a complete prediction is obtained. A complete prediction is denoted as the set containing all sub-predictions obtained by the prediction model:

$$P = \{pred(p_1), pred(p_2), \ldots, pred(p_i)\}$$

The set P_0 denotes the set containing all sub-predictions for the initial prediction. The set P_n denotes the set of sub-predictions corresponding to the prediction of action option n, opt_n. When predictions are generated, this knowledge is saved by updating the belief module. Additionally, this information is analyzed by the analysis module.

– Two types of actions corresponding to the analysis module are defined. The first action type is responsible for action options generation based on an initial prediction P_0. The action of generating action options is denoted by:

$$generation\_of\_options(P_0)$$

This function maps from a complete set of action options OPT to an option set Opt containing the potential action options. This is denoted by the following mapping function:

$$generation\_of\_options(P_0) : OPT \rightarrow Opt$$

The set of potential action options is defined as $Opt = \{opt_1, opt_2, \ldots, opt_m\}$, with m the total number of action options generated. The second action type corresponding to the analysis module, is valuation of action options. This action is responsible to valuate an action option based on knowledge on the action option and the prediction generated of this action option. Through valuation of an action option the (quantitative) utility of the option is determined, which is denoted by val. The valuation action is denoted by:

$$valuation\_of(opt_n, P_n)$$

With P_n the set of sub-predictions generated by the prediction model corresponding to action option opt_n.

– The decision-making module is responsible for deciding on what action to select that maximizes the utility. This is based on the valuation of all action options. The action of selecting an action option is denoted as:

$$selection\_of(Opt, Val)$$

Here, Val represents the set of all valuations val corresponding to all action options in Opt.

– The action module is responsible for the preparation of an action based on the selected action option. Here preparation of an action is defined as determining when an action is to be executed t_{ex} and a time delay δt is considered that may correspond to the execution of the action. The action of preparation is formulated as:

$$preparation\_of(act, t_{ex}, \delta t)$$

The action is part of the set of output actions, $act \in A^{out}$.

– The belief of an agent can be represented as a set of beliefs that holds in a specific state. This set is updated based on an observation, or based on the internal processes. Here, the following belief update actions are defined:

$$updBelief\_with(obs(holds(ap)))$$
$$updBelief\_with(predicted(P_0))$$
$$updBelief\_with(predicted(P_n))$$
$$updBelief\_with(generated(Opt))$$
$$updBelief\_with(valuated(opt_n, P_n, val))$$
$$updBelief\_with(selected(opt))$$
$$updBelief\_with(prepared(act, t_{ex}, \delta t))$$

The arguments introduced for the update of belief are described in the next paragraph defining atomic propositions.

- The set of output actions of the anticipatory agent A^{out} are actions defined as executable by the anticipatory agent in the environment as part of its adaptive behavior. The set of output actions A^{out} corresponds to the set of explicitly represented actions A defined for the environment (TTS_1). The action set is defined as handshaking actions H of $TTS_1 \parallel_H TTS_2$.

Atomic Propositions and Clock Constraints The union of the set of atomic propositions AP and the set of clock constraints $\mathsf{B}(\mathsf{C})$ is defined as follows:

$$\mathsf{AP} = AP \cup \mathsf{B}(\mathsf{C})$$

Atomic propositions are defined to quantify or describe those aspects that hold for the anticipatory agent at a specific moment in time, or after a specific transition. A distinction is made between the formulation of atomic propositions related to input and output actions and for internal actions.

- The atomic propositions that can be formulated with regard to input and output actions ($act \in \{A^{inp}, A^{out}\}$) are formulated as follows:

$$started(act)$$
$$finished(act)$$

The reason to explicitly define the status of an action is to be able to define potential time delays that might exist for the execution for an input or output action.

The set of clock guards can be used to define when specific state transitions are to occur that are time dependent. Here, one clock guard is introduced to define, based on the execution time t_{ex}, when a prepared output action act can be executed. The clock constraint follows the following satisfaction relation: $\theta_{t_{ex}} : v(x) = t_{ex}$. This clock guard states that the clock valuation should be equal to t_{ex} in order to allow a state transition labeled by the prepared output action. Additionally, a state invariant is defined to determine the time it takes to execute the prepared output action act, in other words how much time is allowed to pass in a state where the atomic proposition $started(act)$ holds. The state invariant satisfaction relation is defined as: $Inv(s) : v(x) \leq t_{ex} + \delta t$.

- The atomic propositions that can be formulated with regard to internal actions are as follows:

$$
\begin{aligned}
&predicted(P_0)\\
&predicted(P_n)\\
&generated(Opt)\\
&valuated(opt_n, P_n, val)\\
&selected(opt)\\
&prepared(act, t_{ex}, \delta t)\\
&updatedBelief(z)
\end{aligned}
$$

Each of the atomic propositions defines that an internal action is executed, and the result of an action is represented by an input argument. The labeling functions presented below, define rules that determine when an atomic proposition holds and is assigned to a state. It is assumed that internal state transitions are discrete transitions, thus no time delays or clock guards are take into account.

Labeling Functions and State Invariant Assignment Function The definition combining both labeling functions for atomic propositions and the state invariant assignment functions is formulated as follows:

$$\mathsf{L}(s, v) = L(s) \cup \{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$$

Here, $L(s)$ represents the set containing all labeling functions, and the relation $\{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$ defines for each state whether the clock guard $\theta \in \mathsf{B}(\mathsf{C})$ holds, which is the case if the satisfaction relation holds ($v \models \theta$).

First, two main rule-based labeling functions are introduced that define the general reasoning logic for the state representation after an input or output action is executed. They are formulated as follows:

$$
\begin{aligned}
&\forall s^{inp} \in Post(s_i, act_x)\\
&holds(started(act_x))\\
&\\
&\text{iff for } s_i^{inp}\ holds(started(act_x))\\
&\forall s^{inp} \in Post(s_i, act \neq act_x)\\
&holds(finished(act_x))
\end{aligned}
\tag{4.2}
$$

To denote the set of states that can be reached from state s by performing an action α the following notation is used: $Post(s, \alpha) = \{s' | s \xrightarrow{\alpha} s'\}$. The first function presented, defines that after a transition labeled by an action act_x, in the next state the atomic proposition $started(act_x)$ holds. The second labeling function defines that, if in a state the atomic proposition $started(act_x)$ holds, that after a discrete transition to a new state, where the transition is not labeled by act_x, in the next state(s) $started(act_x)$ does not longer hold but $finished(act_x)$ holds. Further, the state aspect ($asp \in Asp = \{input, internal, output\}$) is denoted here as inp which corresponds to the label $input$. Similar the label $output$ could be applied which would be denoted by out. Note: a short-cut notation for state is applied where the clock valuation $v(x)$ is left out. However, the clock valuation is considered for each state representation. Thus the notation of s corresponds to $(s, v(x))$.

In the same sense a general reasoning logic is defined for the state representation after an internal action is executed. Taking the action $prediction(opt_n)$ as an example, the labeling function is formulated as follows:

$$
\begin{aligned}
&\forall s^{int} \in Post(s_i, prediction(opt_n))\\
&holds(predicted(P_i))
\end{aligned}
\tag{4.3}
$$

From the labeling function is can be deduced that after a discrete transition occurs due to an internal action, in the next state(s) the action result holds.

Based on the conceptual model, the defined actions, the atomic propositions and general labeling functions introduced, a specific logic for the state transitions for the internal model of the anticipatory agent can be defined through the following labeling functions:

First, it is defined how after doing an observation in the environment of $ap \in AP_1'$ the anticipatory agent is triggered and its belief is updated with this observation:

$$\begin{aligned} &\text{iff for } s_i^{inp}\ holds(started(obs(holds(ap)))) \text{ with } ap \in AP_1' \\ &\quad \forall s^{inp} \in Post(s_i, updBelief\_with(obs(holds(ap)))) \\ &\qquad holds(finished(obs(holds(ap)))) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(obs(holds(ap)))) \\ &\qquad holds(updatedBelief(obs(holds(ap)))) \end{aligned} \tag{4.4}$$

When belief is updated, an initial prediction P_0 is made by the prediction model described in Equation 4.1, of the future environment states. The labeling function corresponding to this action is formulated as follows:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(obs(holds(ap)))) \text{ with } ap \in AP_1' \\ &\quad \forall s^{int} \in Post(s_i, prediction(ap)) \\ &\qquad holds(predicted(P_0)) \end{aligned} \tag{4.5}$$

If a prediction is generated, the belief module is updated, which is formulated as:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(predicted(P_0)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(predicted(P_0))) \\ &\qquad holds(updatedBelief(predicted(P_0))) \end{aligned} \tag{4.6}$$

After belief is updated with the initial prediction, action options are generated denoted by $opt \in Opt$ with Opt the complete set of action options generated. The labeling function corresponding to action generation is formulated as follows:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(predicted(P_0))) \\ &\quad \forall s^{int} \in Post(s_i, generation\_of\_options(P_0)) \\ &\qquad holds(generated(Opt)) \end{aligned} \tag{4.7}$$

If a set of action options Opt is generated, the agent's belief is updates:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(generated(Opt)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(generated(Opt))) \\ &\qquad holds(updatedBelief(generated(Opt))) \end{aligned} \tag{4.8}$$

Then, the generated set of action options Opt, is evaluated by the prediction model, as described in Equation 4.1. A prediction is generated for each action option:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(generated(Opt))) \\ &\quad \forall s^{int} \in Post(s_i, (prediction(opt_1), prediction(opt_2), \ldots, prediction(opt_m))) \\ &\qquad holds(predicted(P_1) \wedge predicted(P_2) \wedge \cdots \wedge predicted(P_m)) \end{aligned} \tag{4.9}$$

Belief is updated with the predictions of each action option:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(predicted(P_1) \wedge predicted(P_2) \wedge \cdots \wedge predicted(P_m)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(predicted(P_1), predicted(P_2), \ldots, predicted(P_m))) \\ &\qquad holds(updatedBelief(predicted(P_1), predicted(P_2), \ldots, predicted(P_m))) \end{aligned} \tag{4.10}$$

Based on the valuation action, for each action option, based on the corresponding predictions, the utility is determined and expressed by val. The labeling function corresponding to this action is formulated as follows:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(predicted(P_1), predicted(P_2), \ldots, predicted(P_m))) \\ &\quad \forall s^{int} \in Post(s_i, (valuation\_of(opt_1, P_1), valuation\_of(opt_2, P_2), \ldots, valuation\_of(opt_m, P_m))) \\ &\qquad holds(valuated(opt_1, P_1, val) \wedge valuated(opt_2, P_2, val) \wedge \cdots \wedge valuated(opt_m, P_m, val)) \end{aligned} \tag{4.11}$$

Belief is updated with the valuation of each prediction of the action options:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(valuated(opt_1, P_1, val) \wedge valuated(opt_2, P_2, val) \wedge \cdots \wedge valuated(opt_m, P_m, val)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(valuated(opt_1, P_1, val), valuated(opt_2, P_2, val), \ldots, \\ &\quad valuated(opt_m, P_m, val))) \\ &\qquad holds(updatedBelief(valuated(opt_1, P_1, val), valuated(opt_2, P_2, val), \ldots, \\ &\qquad valuated(opt_m, P_m, val))) \end{aligned} \tag{4.12}$$

Based on the set of valuations Val corresponding to the action options in Opt, a decision is made on what action to select that maximizes utility. The labeling function corresponding to the selection of an action option is formulated as follows:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(valuated(opt_1, P_1, val), valuated(opt_2, P_2, val), \ldots, \\ &\quad valuated(opt_m, P_m, val))) \\ &\qquad \forall s^{int} \in Post(s_i, selection\_of(Opt, Val)) \\ &\qquad\quad holds(selected(opt)) \end{aligned} \tag{4.13}$$

Belief is updated with the selected action option:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(selected(opt)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(selected(opt))) \\ &\qquad holds(updatedBelief(selected(opt))) \end{aligned} \tag{4.14}$$

Finally, based on the selected action option, the action act is prepared for execution. To this end an execution time t_{ex} is determined for the action, and a corresponding execution delay may be defined δt. The labeling function corresponding to the preparation action is formulated as follows:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(selected(opt))) \\ &\quad \forall s^{int} \in Post(s_i, preparation\_of(act, t_{ex}, \delta t)) \\ &\qquad holds(prepared(act, t_{ex}, \delta t)) \end{aligned} \tag{4.15}$$

Belief is updated with the information regarding the action preparation:

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(prepared(act, t_{ex}, \delta t)) \\ &\quad \forall s^{int} \in Post(s_i, updBelief\_with(prepared(act, t_{ex}, \delta t))) \\ &\qquad holds(updatedBelief(prepared(act, t_{ex}, \delta t))) \end{aligned} \tag{4.16}$$

Then, if belief is updated with the preparation of the output action, the state transition labeled by the output action, is guarded by a clock guard $\theta_{t_{ex}} : v(x) = t_{ex}$ which ensures the action is executed at the defined execution time t_{ex}. Additionally, to represent the time it takes to execute the output action, a state invariant $Inv(s) : v(x) \leq t_{ex}+\delta t$ is posed on the output state. If the state invariant does not hold, the atomic proposition $holds(started(act))$ does not hold any longer, and the action is finished.

$$\begin{aligned} &\text{iff for } s_i^{int}\ holds(updatedBelief(prepared(act, t_{ex}, \delta t))) \wedge v(x) \models \theta_{t_{ex}} \\ &\quad \text{for } s_{i+1}^{out} \in Post(s_i, act) \\ &\qquad holds(started(act)) \\ &\qquad \text{iff for } s_{i+1}^{out} \quad \neg v(x) \models Inv(s_{i+1}^{out}) \\ &\qquad\quad holds(finished(act)) \end{aligned} \tag{4.17}$$

Transition Relations For the anticipatory agent two types of transition relations are defined:

- The internal discrete state transitions are represented as follows:

  $$(s_i, v) \xrightarrow{act \in A^{int}} (s_{i+1}, v)$$

  Here internal state transitions are considered as discrete transitions, thus no time delays are considered for these transitions.

- The interactions with the environment are represented as follows:

  $$(s_i, v) \xrightarrow{\delta t} \xrightarrow{act \in \{A^{inp}, A^{out}\}} (s_{i+1}, v + \delta t)$$

  Here the short-cut notation is applied to specify the time delay considered for the execution of input and output actions.

Based on the frameworks presented above for the environment system TTS_1 and the anticipatory system TTS_2, anticipation in a complex socio-technical system can be described. To obtain an overview of the complete system presented here, one should identify the handshaking actions which allows to seek for synchronization points of parallel systems. Here, handshaking actions are defined as the output actions executed by the anticipatory agent in the environment.

The above presented representation can be extended by including more sub-systems that are able to anticipate, or describing sub-systems/agents with different functioning. In the next part of this research the formal framework is applied to study anticipation in a real-life complex socio-technical system, through computational simulation.

II

Operationalization of Anticipation

5

Methodology

As described in Chapter 3, the security checkpoint system is selected as case study to operationalize an anticipation mechanism through a computational simulation study. To develop a realistic model of airport terminal security operations, available data from Rotterdam the Hague Airport (RTHA) is used. This chapter presents the methodology applied to develop a computational model of security operations, that includes an anticipation model, and allows to study the effects of anticipation in the face of challenging passenger demand.

In Section 5.1, the scope and objective of the case study are outlined. In Section 5.2, the development of an agent-based airport terminal operations model (AATOM) is described. Based on this model, in Section 5.3, a model for security check-point operations is developed. In Section 5.4 an anticipation model applicable for the security checkpoint is developed. In Section 5.5, it is described how the model is implemented for computational simulation. Finally, in Section 5.6 the analysis steps are outlined.

5.1. Define Case Study Scope and Objective

For the case study, morning operations at RTHA are simulated, from 05:00 AM till 10:00 AM. For this time span, the resources at the security checkpoint of RTHA are planned based on the flight departure schedule, the expected number of passengers corresponding to these flights, and the expected arrival times of these passengers. As disrupted scenario, two delayed flights are introduced which results in a changed departure schedule. Passengers corresponding to these two flights will adjust there arrival time at the security checkpoint. Consequently, the arrival pattern of passengers originally forecasted, is disrupted and the planned resources at the security checkpoint might not match with the changed demand. By implementation of an anticipation model, it is studied how anticipation could contribute to the ability to maintain a stable performance, under these disrupted conditions.

Essentially, the main goal of the case study is to illustrate how, by means of a simulation study, the resilience mechanism of anticipation can be analyzed in the context of a real-world socio-technical system. To this end, the objective of the case study is to study the effects of anticipation at the security checkpoint, in the face of a disrupted flight schedule leading to unplanned and challenging demand, by performing computational simulations of this scenario with an agent-based airport terminal operations model.

5.2. Development of an Agent-based Airport Terminal Operations Model (AATOM)

Central to this case study is the agent-based airport terminal operations simulation environment (AATOM) developed by PhD student S.A.M. Janssen. This simulation environment provides a tool to perform experiments in an airport terminal system by performing computational simulations. The advantage of such a simulation tool is that it becomes easier to conduct all sorts of research with regard to airport terminal operations, which is normally hard to conduct in real-life, as the system is constantly in operation and often safety and security constraints apply. This research is one of the first researches that applies the AATOM as experimental simulation environment to perform an airport terminal case study. As the AATOM was still in its early development stage, when starting this research, significant time of this research is dedicated to contribute to the model development [1]. A comprehensive semi-formal model specification is developed for the AATOM. For a complete overview of this specification, one is referred to [50]. In this section a concise model description is provided.

Introduction to AATOM

AATOM functions as a 'baseline' model which can, relatively easy, be adjusted and extended to fit any research objective considering airport terminal operations. The computational model is Java-based and follows the agent-based modelling paradigm. This means that the system is represented by a bottom-up modelling approach, whereas all autonomous entities of the system are modelled as agents that operate in an airport terminal environment. The model describes the main handling processes for outbound passengers in the airport terminal. These are: check-in operations, security checkpoint operations and border control operations. Additionally, basic facilities for discretionary activities are included such as, bathrooms, restaurants and shops. The model specification distinguishes between an environment specification, a specification of agents and their autonomous behavior, and a specification of interactions between the system components.

AATOM Environment Specification The system considered is an airport terminal. Passengers are generated as input for the system and they disappear with their departing flight after going through the required terminal processes. Additionally, it is possible to define conditions outside the system that might influence processes inside the system, such as a change in the flight schedule due to bad weather conditions. For this reason the system is considered as an open system. The AATOM environment specification describes all aspects part of the system, excluding the agents present in the system. To describe the environment, three types of environment objects are defined: Area, Flight and Physical Object. Some of these objects have a set of sub-objects, more specific instances of the corresponding environment object. A hierarchical visualization of the environment is shown in Figure 5.1. The different objects of the environment are specified below.

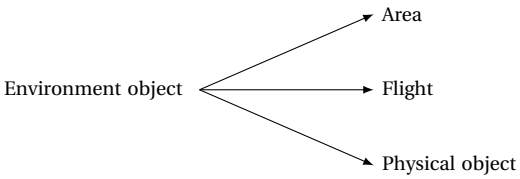


Figure 5.1: The hierarchy of environment objects.

- **Area:** An area is defined to be a shape that is accessible to all human agents. An area is a two dimensional polygon that is bounded by a sequence of location points. Areas defined are: entrance area, check-in area, check-point area, facility area, gate area, border control area, and arrival area. Additionally, queuing area's are defined connected to the check-in area, check-point area and border control area. The set of area's accessible without going through security is defined as the open area, the set of area's which are only accessible after going through security is defined as the secure area.

[1] An overview of the contribution delivered as part of this research is provided in Appendix B.

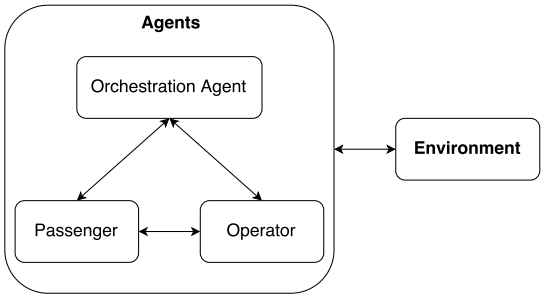


Figure 5.2: The different agent types that operate in the environment and interact with each other and the environment.

- **Flight**: A flight is defined to be an abstract concept described by four properties: flight type (arrival or departure), time of arrival or departure, set of check-in desks, and gate area. Each passenger in the model is related to a specific flight. The flight defines to which check-in desk and gate the passenger has to go, and when its flight departs (or arrives). Furthermore, check-in operators are related to a flight in such a way they are able to perform the check-in of a specific passenger.
- **Physical Object**: A physical object is an environment object that has a physical representation in the environment. The physical objects that are defined in the model are: wall, queue separator, belt, security checkpoint sensors, seat, and luggage.

AATOM Agents Specification The AATOM consists of three agent types. Namely, passengers, operators and orchestration agents. Figure 5.2 presents an overview of the interaction structure of the different agent types and of the agents with the environment.

Both passenger and operator agent types are defined as human agents, the orchestration agent type can be defined as a human or non-human agent. Comparable to conceptual model presented in Chapter 4, the human agents follow a three layer model based on the work of Blumberg [42], Hoogendoorn [43] and Reynolds [44]. For this architecture a distinction is made between three abstraction levels for human agents. The top layer, called the *strategic layer*, is responsible for determination of goals, for updating the belief module and for reasoning, which includes analysis resulting in planning and decision-making. The middle layer, called the *tactical layer*, is responsible for actuation of activities. It is also responsible for route navigation in the environment and the interpretation of observations. The *operational layer* is responsible for interaction with the environment and other agents. Here, input is generated by sensory information an agent can observe. The output is defined as actions an agent can execute. The actions are generated based on input from the tactical layer. Each agent type is further specified below.

- **Passenger:** Passengers are present in the airport terminal because they either have a departing flight to catch or they arrived at the airport via an arriving flight. Departing passengers have as main goal to reach their flight in time. To reach this goal they go through all processes required such as, check-in, security check and border control. Depending on the modelled characteristics of the passengers, passengers can set the goal to visit a facility such as, a shop, the toilet or a restaurant. Arriving passengers have as goal to leave the airport as quick as possible, following the allowed routes. The walking mechanism of a passenger is based on the Social Force Model developed by Helbing et al. [51]. The model combines physical and 'social' forces to calculate the velocity and velocity changes of a passenger. The velocity changes depend on observations of human agents and physical objects, within the passenger's proximity.
- **Operator Agent:** An operator agent performs an operation within the airport terminal. What this operation is, depends on its assignment. Two types of operator agents are defined: check-in operators and security operators. The first type is responsible for the check-in operations. This agent type stands behind a check-in desk, when a passenger arrives at the desk for check-in, the operator changes the passenger's state from not checked-in to checked-in, and registers for the corresponding flight that this passenger has checked-in. For a security operator agent type, seven different sub-types are defined, each performing a different activity. The possible activities are: travel document check, assisting

the luggage drop, monitoring the x-ray, luggage check, physical check, explosive trace detection (ETD) check, and border control check.

- **Orchestration Agent:** An orchestration agent can be included to monitor overall performance of the airport terminal and to coordinate terminal operations. The agent can be represented as a human agent following the three layer architecture, or as a non-human agent. Orchestration agents can be designed as such that it is able to steer operator agents and/or passengers, and make adjustments in the environment, to achieve specific system goals.

AATOM Interactions Specification Three types of interactions are defined in the model. The first type is the interaction between agents and the environment. An example is the interaction between the check-in operator and flight. Here, the check-in operator adjusts the flight to register that a passenger is checked-in for the specific flight. Additionally, interactions are defined between security operators and the security sensors that reset the sensor state to idle, after observations are made. The second type of interactions, are interactions between an operator agent and a passenger. For most processes, a waiting period is communicated by the operator agent to a passenger. For instance, when a security operator performs a luggage check activity or an ETD check activity, the passenger is ordered to wait for a specific period of time. Finally, also interactions exist between operators. For instance if the security operator who performs the x-ray activity, observes a suspicious item in the luggage, it is communicated to the security operator who performs the luggage check activity, that the luggage has to be checked.

5.3. Development of a Security Operations Model

As presented in the previous section, the AATOM simulator includes an extensive basis to model airport security operations in an airport terminal. In order to use this model for the research objective presented for the case study, the model specification is adjusted. Additionally, the model is initialized and calibrated to obtain a realistic model representing the security operations of RTHA. Further, an anticipation model to implement for the case study is developed based on the formal model presented in Part I of this research.

Model specification for security operations

As the model specification is based on the AATOM model specification, in the same structure an environment, agents and interactions specification is presented, for the security operations model.

Environment Specification The areas modelled are an entrance area, a check-point area, and a queuing area corresponding to the security check-point. The queuing area exists of one queuing lane, and provides access to the security check-point from the entrance area. In the entrance area passengers arrive right before they enter the queuing area to go to the security check-point. The security check-point area is equipped with security sensors and operator agents and exists of multiple security lanes. To match the security operations of RTHA, the security lane set-up as employed at RTHA is implemented in the model. Figure 5.3 presents an overview of the security lane distribution.

A flight schedule for departing flights is considered. A flight schedule is defined as a set of flights. Here, the concept of flight is defined by two properties: its departure time, and the number of passengers that fly with this flight. Furthermore, a flight has a relation with each passenger that flies with the specific flight.

Physical objects relevant in the model, are those that are the building blocks of the queuing area and the security check-point area. These are: wall, queue separator, belt, security checkpoint sensors. Additionally, as luggage is checked in the security process, luggage is modelled as a physical object as well. A relation exists between a luggage object and the passenger the luggage belongs to.

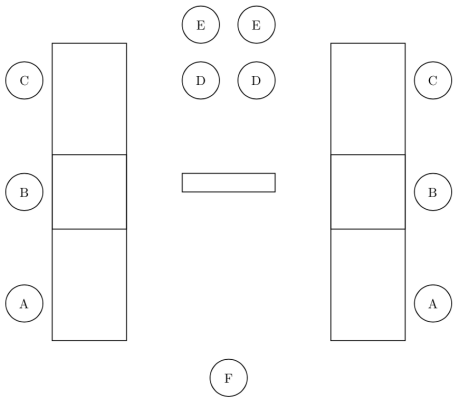


Figure 5.3: A typical distribution of two security check-point lanes at RTHA. Operator A assists the luggage drop activity, operator B performs the x-ray scan activity, operator C performs the luggage check activity, operator D performs the physical check activity, operator E performs the ETD check activity, and operator F is responsible for boarding card checks.

Agents Specification Three agent types are included in the model for the security operations: passengers, security operators, and an orchestration agent. Passengers are modelled as departing passengers, as only this type of passengers pass the security check-point at RTHA. Each passenger has a relation to one of the flights from the flight schedule. After arriving in the entrance area, all passengers aim to pass through security, and reach their flight on time. Each passenger is modelled as to carry one carry-on luggage item with it, which is to be checked at the security check-point. Only security operator agents are modelled, as the check-in process is not taken into account. The assignments performed by the different security operator agents are presented in Figure 5.3.

Both passengers and security operator agents perform several activities. These activities are 'prepared' in the tactical layer of the three layer internal model for human agents [50]. The term 'activity area' is used to refer to the position where an activity is performed. The activities included in the model for security operations are:

- Passenger activities
  - **Queue Activity:** The queue activity is executed when other activities cannot be executed due to capacity limitations at the activity area. The activity is in general performed in a queue area, but is not limited to this area type. Further, the activity is modelled as a waiting period that ends when a passenger in front moves, or the planned next activity has available space again.
  - **Checkpoint Activity:** The checkpoint activity is mandatory for passengers. It is executed at the checkpoint area. The checkpoint activity consists of several actions, of which some are mandatory and some are optional. The mandatory actions are: travel document check, luggage drop, and luggage collect. The optional actions are: physical check and ETD check. The optional action are executed depending on whether a passenger is identified as a potential threat by the walk through metal detector (WTMD) sensor, or if the passenger is selected for an ETD check. Each of these actions can be preceded by an optional queuing activity.
- Security operator activities
  - **Travel Document Check Activity:** This activity is modeled as an interaction with a passenger, where the passenger is order to wait for a period of time. The activity starts when a passenger is at the activity area.
  - **Luggage Drop Activity:** This activity is modeled as an interaction with a passenger, where the passenger is order to wait for a period of time. The activity starts when a passenger is at the activity area.
  - **X-Ray Activity:** Luggage is checked by going through the X-Ray sensor. The X-Ray activity starts when the security operator observes that the X-Ray sensor is in the observed state. The operator observes the X-Ray sensor state and the observed the Boolean output of the X-Ray sensor. If the

output of the X-Ray sensor is true, the X-Ray activity includes an interaction with the operator for the luggage checking. If the observed X-Ray sensor observation is false, the activity is completed without any further action. The luggage will proceed to the luggage collect area.

- **Luggage Check Activity:** If the security operator performing the luggage check activity observes the search communication from the operator performing the X-Ray activity, the luggage check activity is performed. The activity is modelled as an interaction with the passenger corresponding to the luggage, where the passenger is order to wait for a period of time.
- **Physical Check Activity:** The security operator performing the physical checking activity is responsible for detecting illegal items on the body of a passenger. It performs a check based on observations of the WTMD, the check is initiated when the WTMD sensor output value is 1. The activity is modelled as an interaction with the passenger, where the passenger is order to wait for a period of time.
- **ETD Check Activity:** The security operator performing the ETD checking activity is responsible for detecting explosive traces on a passenger. The check is based on indications of the WTMD, the check is initiated when the WTMD sensor output value is 2. The process is modelled as an interaction with the passenger, where the passenger is order to wait for a period of time.

The orchestration agent is modelled with the function to observe the disrupted flight schedule, and based on this observation anticipate the potential risks with regard to saturation of the system. This means that the orchestration agent follows an internal anticipation model, which makes predictions, develops potential actions, and valuates these actions to come to a decision on what action to perform. Such an action is modelled as an interaction between the orchestration agent and the environment, whereby the orchestration agent opens or closes security lanes. The development of the anticipation model is presented in Section 5.4.

Interactions Specification The interactions included in the model are already briefly mentioned in the environment and agent specification. Operator and passenger interactions are modelled as the communication of a waiting period from the operator to the passenger. Furthermore, the operator agents that perform the X-Ray, ETD check and Physical check activity, interact with the security sensors to observe whether a check is required. Additionally, the X-Ray operator communicates to the luggage check operator, whether the luggage has to be checked. Finally, the orchestration agent interacts with the environment as it is able to open or close security lanes. Passengers are able to observe whether a security lane is opened or closed, and will not move to a closed security lane.

Model initialization

In this section the initialization of the security operations model is presented. This is done based on data and information available from the operations at RTHA.

Flight Schedule The flight schedule implemented corresponds to a regular flight day of RTHA. Specifically, the morning schedule of October 5th 2017 is modelled. At RTHA the overall flight schedule exists of three clear departure waves. In the morning, in the afternoon and in the evening. It is chosen to implement the morning schedule, because over this time span most flights per time unit depart. This makes the morning the most crowded part of the day, and is therefore more sensitive to disruptions. The flight schedule is presented in Table 5.1. The number of passengers is calculated based on the capacity of the aircraft and a load factor of 0.9. This load factor is assumed by RTHA for their passenger forecast.

Arrival Times The arrival times of passengers at the security check-point is determined based on the forecast method of RTHA [52]. It is assumed by RTHA, based on experience, that passengers arrive following an arrival time distribution which depends on the departure time of the flight. The distribution is presented in Table 5.2. The arrival of passengers is modelled as a Poisson process. This is done by drawing inter arrival times form an exponential distribution $Exp(\lambda)$. Here λ, the arrival rate, varies over time as it depends on the

Table 5.1: The flight schedule implemented is based on the morning flight schedule of RTHA on October 5th 2017. *The number of passengers in the flight is based on the capacity of the aircraft and a load factor of 0.9.

Flight	Destination	Departure time	Pax*
HV6035	Rome - Fiumicino	07:00	173
HV6301	Gran Canaria	07:00	134
HV6493	Venetie	07:00	173
BA4450	London City	07:00	89
HV6771	Budapest	07:30	173
HV5021	Malaga	08:00	134
BA4452	London City	10:00	89

Table 5.2: The arrival time distribution for passengers at the security check-point, based on the forecast method of RTHA [52].

Time before departure	2 hr - 1.5 hr	1.5 hr - 1 hr	1 hr - 30 min	30 min - 10 min
Percentage of passengers	10%	40%	40%	10%

amount of passengers that correspond to a flight and are to arrive within a specific time span, which in turn depends on the arrival time distribution presented in Table 5.2. The arrival rate λ is calculated for each half hour, based on the information provided in Table 5.1 and 5.2, following equation 5.1.

$$\lambda_{\Delta T} = \frac{\#\text{pax}_{\Delta T}}{1800s} \tag{5.1}$$

Here, ΔT denotes a specific time span of an half hour, and $\#\text{pax}_{\Delta T}$ denotes the number of passengers expected within this time span. Table 5.3 presents the arrival rate λ calculated for each half hour, corresponding to the flight schedule presented in Table 5.1.

Processing Times The processing times of each of the activities at the security check-point are based on empirical data gathered at RTHA via video analysis of the security operations. The data is gathered by PhD student S.A.M. Janssen to support the development of AATOM. The processing time distributions, determined based on this data, are presented in Table 5.4. Furthermore, the probability the WTMD initiates an ETD check, $P(\text{ETD check}) = 0.1$, and the probability the WTMD initiates a physical check, $P(\text{Physical check}) = 0.1$. The probability an ETD check is initiated is dependent on the random check setting of the WTMD. RTHA generally sets this setting at one in ten. The probability of a physical check is determined based on the empirical data gathered at RTHA. Finally, the recovery time of the X-Ray scanner is set at 3 seconds.

Scheduled Resources At RTHA, the required resources at the security check-point are scheduled based on a scheduling method that calculates the the number of security lanes required over time [53]. As input the scheduling method uses the number of passengers expected over time in time steps of half an hour, and the processing rate of the security lanes. RTHA assumes a processing rate of 2.6 pax/min per security lane [53]. This scheduling method is applied to obtain a resource schedule for the morning of interest, which is presented in Table 5.5. In total five security lanes are present at RTHA, this number limits the amount of resources available. Additionally, in Figure 5.3 it is shown that one security lane requires 6 security operators, and two security lanes requires 11 security operators.

Table 5.3: The arrival rate λ determined for each time span of half an hour over a time period of 05:00 AM until 10:00 AM.

Time	05:00	05:30	06:00	06:30	07:00	07:30	08:00	08:30	09:00	09:30
λ (pax/s)	0.032	0.136	0.172	0.100	0.039	0.007	0.005	0.020	0.020	0.005

Table 5.4: The processing time distributions for each of the activities performed at the security check-point. The processing time distributions are determined based on empirical data gathered at RTHA via video analysis of the security operations, these distributions are implemented in the AATOM. *If a negative value is drawn from the distribution, this is corrected to a value of zero in the simulator.

Activity	Processing time* (s)
Luggage drop	$N(54.60,36.09)$
Physical check	$N(43.00,20.96)$
ETD check	$N(34.80,15.17)$
Luggage collect	$N(71.50,54.95)$

Table 5.5: Number of security lanes in use over time corresponding to the flight schedule presented in Table 5.1 (07:00-10:00). This schedule is based on the scheduling method of RTHA [53].

Time	05:00	05:30	06:00	06:30	07:00	07:30	08:00	08:30	09:00	09:30
# of lanes	2	3	4	3	2	1	1	1	1	1

Delayed Flights As outlined in Section 5.1, two flight delays are introduced as disrupted scenario. An overview of the delayed flights is presented in Table 5.6. With the shift in departure time of these flights, it is assumed that the arrival times of passengers shifts as well. As no data is available to support an estimation of how the arrival times of passengers will shift, the assumption is made that passengers that did not already arrive at the airport at the time of announcement, will follow the arrival distribution as presented in Table 5.2, assuming the new departure time. The delay scenario presented in Table 5.6 is selected because it results in a significant decrease of system performance, whereby the system is in a saturated state for a relatively long period of time. As a consequence, a relatively high number of passengers will miss or delay their flight.

Model calibration

Model calibration is performed to ensure that the processing rate of the model is in close resemblance with the actual processing rate of the security check-point assumed by RTHA. The processing rate assumed at RTHA is 2.6 pax/min per security lane [53]. Considering the input parameters as presented in the previous section, simulations are ran to measure the mean processing rate as output of the model. The result obtained without calibration is a processing rate of 1.52 pax/min for one security lane. Two calibration steps are executed to achieve a processing rate of 2.6 pax/min.

1. The first calibration step consists of a structural model change. The security check-point process for passengers is adjusted as such that passengers are able to start the luggage drop activity with three passengers at the same time, in parallel, at one security lane. Before, the luggage drop activity was executed by one passenger at the same time. To allow for a parallel luggage drop the model is better fitted to reality and the process rate increases. The choice to allow for three passengers in parallel is based on the maximum number of passengers that are able to start this activity at the security check-point of RTHA. After the adjustment of the luggage drop activity, a next simulation iteration is performed to measure the processing rate. The obtained average processing rate has increased, and results in a value of 2.10 pax/min.
2. For the next calibration step the input parameters of processing times, as presented in Table 5.4, are multiplied by a calibration constant. Each processing time is multiplied by the same calibration constant to ensure that relative to each other the processing times remain to have the same ratio. It is chosen to calibrate the model while two security lanes are opened, and aim for a processing rate of 2.6 pax/min per lane. The reason to calibrate with two lanes opened is that two lanes share one WTMD,

Table 5.6: An overview of the delayed flights. Here, DT stands for departure time, and ToA refers to the time of announcement of the delay.

Flight	Original DT	Pax	ToA	Delay	New DT
HV6035	07:00	173	07:10	45 min	07:45
HV6301	07:00	134	07:10	60 min	08:00

Table 5.7: The calibrated processing rates $r(n)$ obtained for different security lane configurations, with n the number of open security lanes. The mean value (μ) and standard deviation (σ) are provided.

n (# lanes)	$r(n)$ (pax/min)	
	μ	σ
1	2.71	0.14
2	5.20	0.22
3	7.91	0.35
4	10.40	0.41
5	13.11	0.54

which influences the processing rate. The calibration factor applied is 0.655. The resulting processing rates for different security lane configurations are presented in Table 5.7.

5.4. Development of an Anticipation Model for Security Operations

In this section, the development of an anticipation model for the orchestration agent is outlined and the interactions between this anticipatory agent and the environment are defined. The anticipatory agent model is implemented in the security operations model as defined in the previous section. In Chapter 4 a conceptual model for an anticipatory agent is presented and syntax is introduced that formalizes anticipatory behaviour in a complex socio-technical system. Based on this framework, the anticipation model is operationalized for this specific case study. First, a conceptual model is developed based, followed by a formalization of the conceptual model.

Conceptual model

As described in the previous section an agent-based model is developed that allows to perform computational analysis of the security check-point operations in the airport terminal using AATOM. Based on this model, a specification of the relevant aspects of the environment, interactions and internal processes of the anticipatory agent are presented here.

Specification of Environment The environment is defined by its state space evolving over time. Environmental aspects relevant and observable for the anticipatory agent are:

- The flight schedule
- The number of security lanes opened
- The number of passengers in the security check-point queue

Specification of Interactions Between Agent and Environment Two types of interactions are defined between the anticipatory agent and the environment: making observations and execution of actions. Interactions corresponding to these interaction types are the following:

- Making observations (by the agent)
  - Observation of the flight schedule
  - Observation of number of security lanes opened
  - Observation of the number of passengers in the security check-point queue
- Execution of actions (by the agent)

- Opening n security lane(s), with n the number of security lanes
- Closing n security lane(s), with n the number of security lanes

For the anticipatory agent, when making an observation of the environment, this leads to an input and internal state transition. Further, execution of an action will result in an output state change of the anticipatory agent and a state transition of the environment's state space.

Specification of Internal Processes of Agent The internal model of the anticipatory agent is discussed in Chapter 4. For the internal model the following modules are defined: *perception and interpretation module*, *belief module*, *reasoning module*, *action module*. For a specification of each of this modules, one is referred to Chapter 4.

For the case study scenario, the anticipatory agent is triggered by the observation of a change in the flight schedule (two delayed flights). As the time of announcement of the delayed flights is 05:10, at this moment in time the anticipatory agent makes this observation and its internal anticipation model is initiated. It is defined that the anticipatory agent is able to make predictions for the complete time trace executed for the case study scenario, thus from 05:10 until 10:00. Further, the output actions of the anticipatory agent are opening and closing of extra security lanes. For these actions, execution times are to be determined to define when extra lanes are opened and closed and corresponding time delays are to be defined that represent the time it takes to execute an action.

Formal model

A formal model specification of the environment and anticipatory agent are developed following the TTS-syntax, and the framework as developed in Chapter 4.

Environment Specification The environment can be described as a 6-tuple $(\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$, which is referred to as TTS_1. Each of its aspects is defined as follows:

- **Time Representation:** The case study scenario defines a time trace from 05:00 AM until 10:00 AM. This time is represented by the clock variable $C = \{x\}$, representing the real time in the system. The clock valuation function is defined as $\mathsf{v} : x \rightarrow \mathbb{R}^+$ which assigns the current clock value $\mathsf{v}(x)$ to each state. No clock resets are defined, thus the set of clocks to be reset is empty: $\lambda = \emptyset$. The time trace for clock x is presented in Figure 5.4. The time-trace represents a dense time set, where the clock value is determined by the valuation function, and is denoted by $\mathsf{v}(x)$. Here, t=0 corresponds to 05:00 AM and the final time corresponds to 10:00 AM.

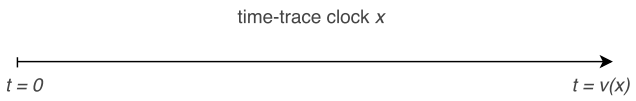


Figure 5.4: A visual representation of the time-trace of clock x, representing a real-time dense time set underlying an execution run.

- **State Representation:** The environment state representation is as follows:

$$\mathsf{S} = S \times V$$

Here, S represents the set of states that make up the TTL. Each state is defined by its state properties expressed as atomic propositions that are assigned to a state by the labeling functions. Additionally the state is defined by $V = v(x)$, which represents the valuation of clock x, thus the moment in time. Further, the initial state is formulated as:

$$\mathsf{S}_0 = S_0 \times V_0$$

Here, S_0 is described by all state conditions that hold at the beginning of the time trace, and $V_0 = 0$ which corresponds to 05:00 AM.

- **Events Representation:** The set of events is represented as:

$$\mathsf{A} = A \cup \tau \cup \mathbb{R}^+$$

Here, A represents the set of labeled actions that can be executed by the anticipatory agent in the environment. The set of labeled actions is defined as: $A = \{\alpha_1, \alpha_2\}$, here:

$$\alpha_1 = openLane(n)$$
$$\alpha_2 = closeLane(n)$$

Here, $n \in N = \{1,2,3,4\}$ represents the number of extra security lanes to open or close. The number of security lanes n is constrained by the number of lanes available at RTHA and the number of lanes already opened. It is assumed that always at least one lane is opened. Further, τ represents all events that take place in the environment leading to discrete transition, and are not explicitly represented. Finally, the time delay posed on a delay transition is a non-negative real number ($\in \mathbb{R}^+$).

- **Atomic Propositions and Clock Constraints:** The union of the set of atomic propositions AP and the set of clock constraints $\mathsf{B}(\mathsf{C})$ is defined as follows:

$$\mathsf{AP} = AP \cup \mathsf{B}(\mathsf{C})$$

Here, the following atomic propositions (AP) are defined to describe the environment's state:

  - The atomic proposition that describes the demand posed on the the security check-point is formulated as:

$$demandValue(d)$$

    Here, $d \in \{0 \cup \mathbb{R}^+\}$, the demand value expresses the number of passengers in the queue at the security check-point at a certain point in time.
  - The atomic proposition that describes the flight schedule is formulated as:

$$flightSchedule(f)$$

    Here, $f \in F = \{F_1, F_2\}$ represents the flight schedule that holds at a specific moment in time. The case study scenario defines two flight schedules, the original flights schedule F_1 and the disrupted flight schedule, with two delayed flights F_2. The flight schedule is represented by a matrix including for each flight the flight number, the departure time, and the number of passengers corresponding to the flight.

$$F = \begin{bmatrix} flightNumber_1 & departureTime_1 & numberOfPax_1 \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ flightNumber_i & departureTime_i & numberOfPax_i \end{bmatrix}$$

    Here, $i = 7$ the total number of flights included in the flight schedule as presented in Table 5.1.
  - The atomic proposition that describes the number of security lanes opened is formulated as:

$$lanesOpen(y)$$

    Here, $y \in Y = \{1,2,3,4,5\}$ is the number of lanes opened at the a specific moment in time. This number is constrained by the total number of security lanes available at RTHA.

The set of atomic propositions that hold in a state are denoted by AP. The atomic proposition $ap \in AP$ that is defined as a trigger for the anticipatory agent to anticipate, is: $flightSchedule(F_2)$.

The set of clock constraints $B(C = x)$ is defined to exist of state invariants only. A state invariant is denoted by $Inv(s)$, and is assigned to a state by an invariant assignment function: $inv : S \rightarrow B(C = x)$. The state invariant $Inv(s)$ defines for how long time is allowed to progress, while the state remains unchanged, e.g. the state holds, as long as the state invariant holds. When the invariant does not hold, the state must be left. As it is assumed that the environment changes over time partly due to external influences (represented by τ), to each state an state invariant is ascribed of δt. This means that each δt a state transition occurs and a new environment state is defined. Essentially, if within a time step δt, due to external influences, the environment state has changed, this can be represented by the shortcut notation: $(s_i, v) \xrightarrow{\delta t} \xrightarrow{\tau} (s_{i+1}, v + \delta t)$. If during the delay transition of δt no environment state change has occurred, this transition is represented by the delay transition notation: $(s_i, v) \xrightarrow{\delta t} (s_i, v + \delta t)$. The introduced atomic propositions describing the environment state, are expected to evolve over time step δt.

- **Labeling Functions and State Invariant Assignment Function:** The definition combining both labeling functions for atomic propositions and the state invariant assignment functions is formulated as follows:

$$\mathsf{L}(s, v) = L(s) \cup \{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$$

As the atomic propositions are defined to describe the environment state space over time, each of the atomic propositions holds in each state. Which means that each of the entities defined as atomic proposition is measurable in each state. The set of labeling functions $L(s)$ describing this, are formulated as follows:

$$\begin{aligned} &\forall s \quad holds(demandValue(d)) \quad \text{with} \quad d \in \{0 \cup \mathbb{R}^+\} \\ &\forall s \quad holds(flightSchedule(f)) \quad \text{with} \quad f \in F = \{F_1, F_2\} \\ &\forall s \quad holds(lanesOpen(y)) \quad \text{with} \quad y \in Y = \{1,2,3,4,5\} \end{aligned} \tag{5.2}$$

The initial state S_0, is described by the atomic propositions that hold at clock valuation $v(x) = 0$. These are as follows:

$$\begin{aligned} &\text{for} \quad s = s_0 \wedge v(x) = 0 \quad holds(demandValue(0)) \\ &\text{for} \quad s = s_0 \wedge v(x) = 0 \quad holds(flightSchedule(F_1)) \\ &\text{for} \quad s = s_0 \wedge v(x) = 0 \quad holds(lanesOpen(1)) \end{aligned} \tag{5.3}$$

Additionally, the number of lanes opened can be adjusted due to an action $\alpha \in A$. The transition is represented explicitly, and the corresponding labeling functions are defined as follows:

$$\begin{aligned} &\text{iff for } s_i \, holds(lanesOpen(k)) \\ &\forall s \in Post(s_i, openLane(n)) \\ &\quad holds(lanesOpen(k+n)) \\ &\forall s \in Post(s_i, closeLane(n)) \\ &\quad holds(lanesOpen(k-n)) \end{aligned} \tag{5.4}$$

Here, to denote the set of successor states that follow a specific action, the following notation is used: $Post(s, \alpha) = \{s' | s \xrightarrow{\alpha} s'\}$. It is defined how many lanes are opened for all states after a labeled transition of the type: $openLane(n)$ or $closeLane(n)$. The resulting number of lanes opened, depends on the number of lanes that were already open $y = k$, and on the number of lanes opened or closed n during the transition.

- **Transitions Relations** In Chapter 4 the different types of transition relations that exist for the environment are defined. One is referred to this chapter for an overview.

Anticipatory Agent Specification The anticipatory agent can be described as a 6-tuple $(\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$, which is referred to as TTS_2. It is assumed that the anticipatory agent is constantly aware of the environment state space, as described above, through observation and updating its belief module. The anticipatory agent is triggered by observation of the disrupted flight schedule F_2. Thus if for the environment (TTS_1) holds that a state transition occurs as follows:

$$TTS_1 : ... \xrightarrow{\delta t}\xrightarrow{\tau} (s_i, v) \xrightarrow{\delta t}\xrightarrow{\tau} (s_{i+1}, v + \delta t) \xrightarrow{\delta t}\xrightarrow{\tau}$$

With the following state representation:

$$\begin{aligned} \text{for } (s_i, v) \quad & holds(flightSchedule(F_1)) \\ \text{for } (s_{i+1}, v + \delta t) \quad & holds(flightSchedule(F_2)) \end{aligned}$$

This state transition is observed by the anticipatory agent, which triggers the internal anticipation model. Here, a formalization is presented of the internal dynamics.

- **Time Representation:** The same time representation, as defined for the environment, holds for the anticipatory agent. One clock variable $C = \{x\}$ is defined which is synchronized with the clock defined for the environment.

- **State Representation:** The state representation is formulated as:

  $$\mathsf{S} = S \times V \times Asp$$

  Here, S represents the set of states that make up the TTS. V represents the valuation for clock set $C = \{x\}$ at every state, and Asp is introduced as the set of labels: $\{input, internal, output\}$, which denote whether a state is an input, internal, or output state of the system. Further, the initial state is formulated as:

  $$\mathsf{S}_0 = S_0 \times V_0 \times Asp$$

  Here, S_0 is described by all input, internal, and output state conditions that hold at the beginning of the time trace, and $V_0 = 0$ which corresponds to 05:00 AM.

- **Action Representation:** For the anticipatory agent a state transition is either a discrete transition, based on an *action* and represented by an *action label*, or a delay transition due to a time delay. An action corresponding to a discrete transition is drawn from the set of action labels ($\in A$). The time delay posed on a delay transition is a non-negative real number. The union of the set of action labels and real positive values is defined as follows:

  $$\mathsf{A} = A \cup \mathbb{R}^+$$

  Additionally, the set of action labels exists of actions executed internally, or is as input or output actions. The sets of internal actions, input actions and output actions are defined as being disjoint.

  $$A = A^{inp} \cup A^{int} \cup A^{out}$$

  For an anticipatory agent the following input, internal and output actions (act) are defined:

– Input actions $act \in A^{inp}$ are defined as observations the anticipatory agent can do of the environment:

$$obs(holds(demandValue(d)))$$
$$obs(holds(flightSchedule(f)))$$
$$obs(holds(lanesOpen(y)))$$

Here, the observation of the atomic proposition $ap \in AP_1 = flightSchedule(F_2)$, is defined as the trigger for anticipation.

– A comprehensive overview of how the internal actions are defined is presented in Chapter 4. Here, for each of the internal actions it is defined how they are operationalized for this case study.

◇ The internal action of making a prediction of future environment states: Two types of predictions are defined, an initial prediction P_0 based on the observed atomic proposition in the environment, and predictions that are made to evaluate the consequences of executing an action option opt_n, P_n. For both prediction types holds that they are made following the prediction model as described by Equation 4.1. Essentially, each sub-prediction p generated contains information about relevant future environment state properties. This is represented by the following mapping function:

$$pred : demandValue(d) \times lanesOpen(y) \times v(x) \rightarrow p$$

Here, a prediction contains the relevant atomic propositions for the environment that hold, and $v(x)$ denotes the clock valuation for the corresponding state. Further, it is defined in the conceptual model that a prediction is made by the anticipatory agent for a time span from 05:10 until 10:00 AM.

◇ Two types of actions corresponding to the analysis module are defined. The first action type is responsible for action options generation based on an initial prediction P_0. The action of generating action options is denoted by:

$$generation\_of\_options(P_0)$$

This function maps from a complete set of action options OPT to an option set Opt containing the potential action options. The set of potential action options is defined as $Opt = \{opt_1, opt_2, \ldots, opt_m\}$, with m the total number of action options generated. For the case study scenario, an option describes when in time extra security lane(s) should be opened and closed, thus a time span wherein extra resources are provided. To this end, an action option that is evaluated by the prediction model, is formally defined as follows:

$$opt = Act \times T_{ex}$$

Here, Act is the set containing all actions act corresponding to the action option considered, with $Act \in A^{out}$. Further, T_{ex} is defined as the set of times of execution t_{ex} corresponding to the actions $act \in Act$. The second action type corresponding to the analysis module, is valuation of action options. This action is responsible to valuate an action option based on knowledge on the action option and the prediction generated of this action option. Through valuation of an action option the (quantitative) utility of the option is determined, which is denoted by val. The valuation action is denoted by:

$$valuation\_of(opt_n, P_n)$$

In Section 5.6 metrics are introduced, to express the value val corresponding to an action option.

◇ The decision-making module is responsible for deciding on what action to select that maximizes the utility. This is based on the valuation of all action options. The action of selecting an action option is denoted as:

$$selection\_of(Opt, Val)$$

Here, Val represents the set of all valuations val corresponding to all action options in Opt. The valuation of action options is presented in Chapter 6.

- ◇ After an $opt \in Opt$ is selected for execution, the action module is responsible for preparing the execution. As for the case study an option is defined as containing a set of actions Act with a corresponding set of executions times T_{ex}, the formulation of the preparation action is as follows:

$$preparation\_of(Act, T_{ex}, \delta T)$$

Here preparation of execution defines the execution times corresponding to the actions $act \in Act$ based on the opt selected. Additionally, for each $act \in Act$ also a time delay $\delta t \in \delta T$ is considered. This time delay represents any delay considered with the execution of the action, for instance mobilization time of resources. Note: $Act \in A^{out}$.

- ◇ The internal actions responsible for updating the set of beliefs, are as follows:

$$updBelief\_with(obs(holds(ap)))$$
$$updBelief\_with(predicted(P_0))$$
$$updBelief\_with(predicted(P_n))$$
$$updBelief\_with(generated(Opt))$$
$$updBelief\_with(valuated(opt_n, P_n, val))$$
$$updBelief\_with(selected(opt))$$
$$updBelief\_with(prepared(Act, T_{ex}, \delta T))$$

The arguments introduced for the update of belief are described in the next paragraph defining atomic propositions.

– The set of output actions of the anticipatory agent A^{out} are actions defined as executable by the anticipatory agent in the environment. The set of output actions A^{out} corresponds to the set of explicitly represented actions A defined for the environment (TTS_1). The action set is defined as handshaking actions H of $TTS_1 \|_H TTS_2$.

$$openLane(n)$$
$$closeLane(n)$$

With $n \in N = \{1,2,3,4\}$ the number of lanes to open or close. For the case study the execution of the selected action option, prepared by the preparation module, is represented by the set $Act = \{openLane(n), closeLane(n)\}$. Here, for both of these actions a time of execution t_{ex} is determined in the set T_{ex}, and a corresponding time delay is defined in the set δT.

- **Atomic Propositions and Clock Constraints:**

– The atomic propositions that can be formulated with regard to the input actions $act \in A^{inp}$ are formulated as follows:

$$started(holds(obs(demandValue(d))))$$
$$finished(holds(obs(demandValue(d))))$$

$$started(holds(obs(flightSchedule(f))))$$
$$finished(holds(obs(flightSchedule(f))))$$

$$started(holds(obs(lanesOpen(y))))$$
$$finished(holds(obs(lanesOpen(y))))$$

The atomic propositions that can be formulated with regard to the output actions $act \in A^{out}$ are formulated as follows:

$$started(openLane(n))$$
$$finished(openLane(n))$$

$$started(closeLane(n))$$
$$finished(closeLane(n))$$

The reason to explicitly define the status of an action is to be able to define potential time delays that might exist for the execution for an input or output action.

A set of clock guards is used to define when specific state transitions are to occur that are time dependent. As defined, the set of actions to be executed based on the anticipatory mechanism is denoted by Act. Corresponding to $act \in Act$, the time of execution $t_{ex} \in T_{ex}$ and time delay $\delta t \in \delta T$ are defined. As Act exists of two actions act, two clock guards can be defined, based on T_{ex}, by the following satisfaction relation: $\theta_{t_{ex}} : v(x) = t_{ex}$. Additionally, two state invariants are defined to determine the time it takes to execute the prepared output actions $act \in Act$, in other words how much time is allowed to pass while being in a state where the atomic proposition $started(act)$ holds before moving to a state where the atomic proposition $finished(act)$ holds. The state invariant satisfaction relation is defined as: $Inv(s) : v(x) \leq t_{ex} + \delta t$.

– The atomic propositions that can be formulated with regard to internal actions are as follows:

$$predicted(P_0)$$
$$predicted(P_n)$$
$$generated(Opt)$$
$$valuated(opt_n, P_n, val)$$
$$selected(opt)$$
$$prepared(Act, T_{ex}, \delta T)$$
$$updatedBelief(z)$$

Each of the atomic propositions defines that an internal action is executed, and the result of an action is represented by an input argument. In Chapter 4, labeling functions are defined to describe when these atomic propositions hold. To avoid extensive repetition of theory, one is referred to this chapter for an overview of these rules. Below, with regard to the case study, the labeling functions defining when the internal anticipatory model is triggered, and when the selected action option is executed are defined.

- **Labeling Functions and State Invariant Assignment Function:** First, it is defined how after doing an observation in the environment of $ap = flightSchedule(F_2)$ the anticipatory agent is triggered and its belief is updated with this observation:

$$\begin{aligned}
&\text{iff for } (s_i^{inp}, v(x) = 05{:}10 \text{ AM}) \; holds(started(obs(holds(flightSchedule(F_2))))) \\
&\quad \forall (s^{inp}, v(x) > 05{:}10 \text{ AM}) \in Post(s_i, updBelief_with(obs(holds(flightSchedule(F_2))))) \\
&\qquad holds(finished(obs(holds(flightSchedule(F_2))))) \\
&\quad \forall (s^{int}, v(x) > 05{:}10 \text{ AM}) \in Post(s_i, updBelief_with(obs(holds(flightSchedule(F_2))))) \\
&\qquad holds(updatedBelief(obs(holds(flightSchedule(F_2)))))
\end{aligned} \tag{5.5}$$

After execution of the internal model (as described in Chapter 4 by Equation 4.5 until Equation 4.16), the prepared output actions $act \in Act$ are executed. The set $Act = \{openLane(n), closeLane(n)\}$. The

time of execution corresponding to action $openLane(n)$ is denoted as t^1_{ex} and the time of execution corresponding to action $closeLane(n)$ is denoted by t^2_{ex}. In the same way corresponding delay times δt^1 and δt^2 are defined. The state transitions labeled by the output actions, are guarded by the corresponding clock guards: $\theta_{t^1_{ex}} : v(x) = t^1_{ex}$ and $\theta_{t^2_{ex}} : v(x) = t^2_{ex}$, which ensure the action is executed at the defined execution time. Additionally, to represent the time it takes to execute the output actions, the following state invariants are defined: $Inv^1(s) : v(x) \leq t^1_{ex} + \delta t^1$ and $Inv^2(s) : v(x) \leq t^2_{ex} + \delta t^2$. Now, for execution of the action $openLane(n)$, the following labeling function is defined:

$$\begin{aligned}
&\text{iff for } s_i^{int}\ holds(updatedBelief(prepared(Act, T_{ex}, \delta T))) \wedge v(x) \models \theta_{t^1_{ex}} \\
&\quad \text{for } s_{i+1}^{out} \in Post(s_i, openLane(n)) \\
&\qquad holds(started(openLane(n))) \\
&\qquad \text{iff for } s_{i+1}^{out} \quad \neg v(x) \models Inv^1(s_{i+1}^{out}) \\
&\qquad\quad holds(finished(openLane(n)))
\end{aligned} \tag{5.6}$$

For the execution of action $closeLane(n)$, the following labeling function is defined:

$$\begin{aligned}
&\text{iff for } s_i^{int}\ holds(updatedBelief(prepared(Act, T_{ex}, \delta T))) \wedge v(x) \models \theta_{t^2_{ex}} \\
&\quad \text{for } s_{i+1}^{out} \in Post(s_i, closeLane(n)) \\
&\qquad holds(started(closeLane(n))) \\
&\qquad \text{iff for } s_{i+1}^{out} \quad \neg v(x) \models Inv^2(s_{i+1}^{out}) \\
&\qquad\quad holds(finished(closeLane(n)))
\end{aligned} \tag{5.7}$$

- **Transition Relations** In Chapter 4 the different types of transition relations that exist for the anticipatory agent are defined. One is referred to this chapter for an overview.

5.5. Model Implementation

To conduct computational analysis of the effects of anticipation at the the security operations of RTHA, the security operations model and the anticipation model, as described in Section 5.3 and 5.4, are implemented in the AATOM simulator. In this Section the implementation steps are outlined. Additionally, the model verification and validation is presented.

Implementation security operations model

Based on the baseline model for airport terminal operations 'AATOM', as presented in Section 5.2, a simulator is developed by PhD student S.A.M. Janssen in Java. In order to research the effects of anticipation at the security operations, this simulation model is adjusted and extended to fit the presented research objective. The performed implementation steps are:

Airport Terminal Lay-out The airport terminal layout implemented corresponds to the default layout of the simulator. This means that the terminal layout is not the same as the actual layout of RTHA. However, as the process of interest is limited to security operations, it is of importance that the security check-point is implemented according to the check-point at RTHA. To this end, one queuing area is implemented, and the set-up of the security lanes is in accordance with the security check-point of RTHA. The simulator includes a visualization option, which provides the possibility to real-time monitor the system. Monitoring can be done by the visualization of the airport terminal wherein the agents operate, see Figure 5.5. Additionally, the graphical visualization option provides the possibility to monitor the system performance real-time, see Figure 5.6.

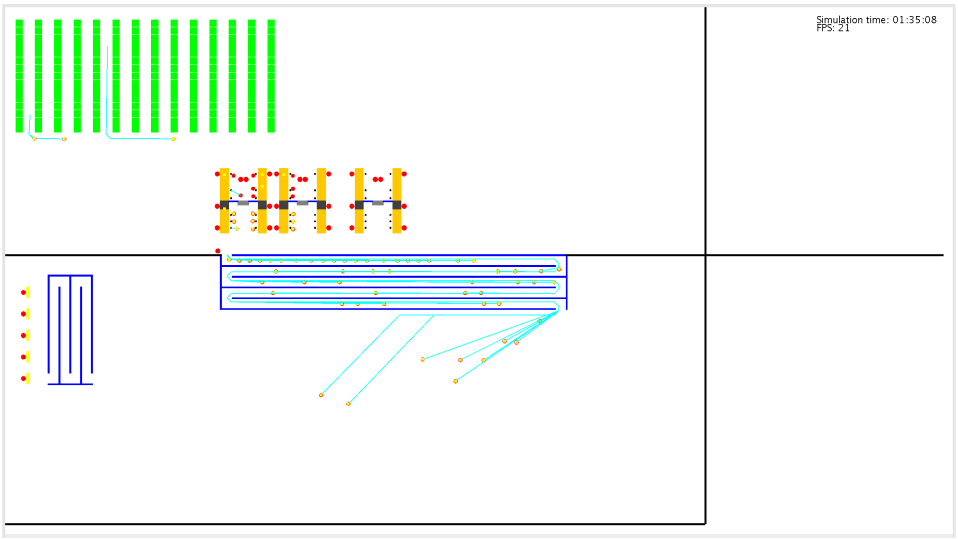


Figure 5.5: The airport terminal layout as applied for this research in the AATOM simulator. The red dots represent the agents in the system. Agents behind the security lane desks are security operators, the other agents are passengers going through the system. The blue lines connected to the red dots, represent the planned routes. The yellow dots represent the luggage that is carried by passengers and is checked by the security operators. The queuing area and desks in the lower left corner represents the check-in. The check-in process is not taken into account for this research. The green area in the higher left corner represents the gate area where passengers wait for their flight to leave.

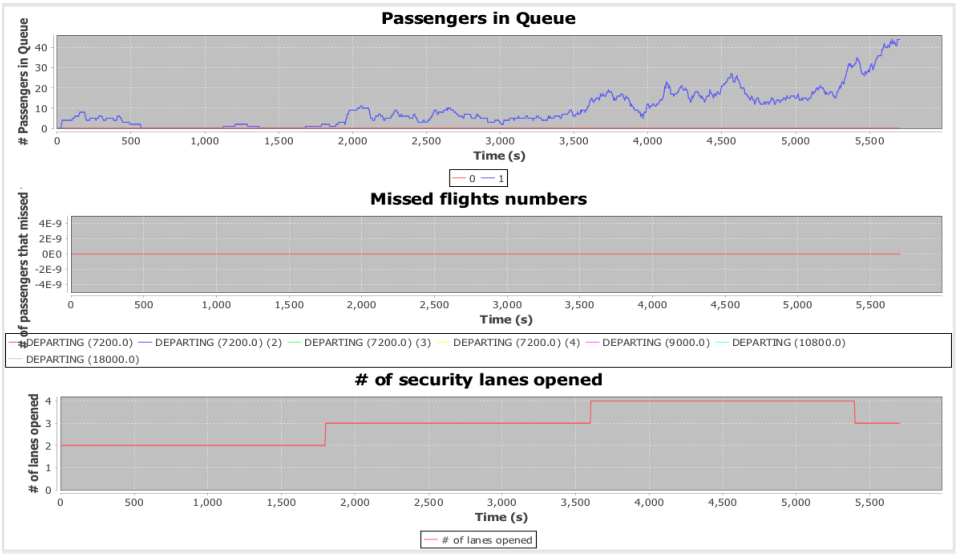


Figure 5.6: The visualization option provides the possibility to monitor system performance real-time, through graphical representations of the defined analyzers.

Analyzers To be able to analyze the system's state over time, three analyzers are implemented: the queue analyzer, the number of missed flights analyzer, and the number of lanes analyzer. The first analyzer keeps track of the number of passengers in the queue over time. The second analyzer, provides insight in how much passengers over time have missed their flight, and which flight this is. The last analyzer presents the number of security lanes opened over time. Additional to the ability of monitoring the analyzed data real-time in the visualization option, all the data is stored in log-files.

Passenger Arrival Based on Flight Schedule In order to generate passengers in the airport terminal based on the flight schedule, an algorithm is implemented that calculates at each point in time, for each flight, whether a passenger should be generated. The inter arrival times of the passengers follow an $Exp(\lambda)$ distribution. If the flight schedule is changed over time, thus when the two delayed flights are introduced, the algorithm knows how many passengers for these flights are already generated, and only the number of passengers not yet arrived in the airport, are generated later in time, according to the new departure time.

Implementation anticipation model

Making Predictions Following the internal anticipation model of the anticipatory agent, as defined in Section 5.4, first a prediction is made of future environment states to obtain insight in the effects of the disrupted flight schedule. Based on these prediction results a set of potential action options is generated that might overcome the negative effects of this disruption. Then, for each of the actions in the action set, predictions are made of the action effects on future environment states. All predictions are computed by forward simulation of the agent-based simulator while the considered conditions of the action option evaluated are applied. The result of the initial prediction is fed to an algorithm that is responsible to develop a set of optional actions. The results of predictions, with regard to the different action options, are valuated through analysis of performance indicators which are defined in Section 5.6.

Predictions are based on N=100 simulations of each scenario. A prediction is presented as the mean value of the number of passengers in the security check-point queue over time.

Generation of Action Option Set As described in Chapter 2 risk of saturation is proposed as an indicative measure for resilience. Additionally, it is discussed that it is expected that a system shows a clear phase transition from a state of normal operation towards a state of adaptive failure. In order to develop the action option generation algorithm, these conceptual ideas are first operationalized and defined for the case study:

- **Risk of saturation:** For the case study scenario considered, a disrupted passenger demand at the security check-point results in congestion, as resources are not matched with the posed demand. Here, system saturation is defined as the moment that the number of passengers in the queue exceeds a predefined threshold value. This threshold is defined in Section 5.6. When the system is saturated this can be regarded as a state of adaptive failure, as the performance of the system decreases below the desired performance level. This is measured by the waiting times of passengers. Now, risk of saturation is defined as the probability of the system reaching a saturated state and the severity of the saturation. This is measured by conducting Monte Carlo simulations of the model and measure how often the system reaches a saturated state and measure the average severity of saturation. Saturation severity is measured following the metrics defined in Section 5.6.
- **Phase Transition:** For this system the phase transition occurs between a state of normal performance, e.g. where the security check-point processing performance is within the defined performance boundaries, to a state of congestion, where the system performance is decreasing below the defined performance threshold. This phase transition is characterized by a continuous accumulation of passengers in the queue, resulting in a saturated state. The period of time over which this phase transition occurs, is defined as the *pre-saturation phase*. The period of time over which the system is in a saturated state is defined as the *saturation phase*. In Figure 5.7 an example is provided of how the pre-saturation phase and saturation phase are defined.

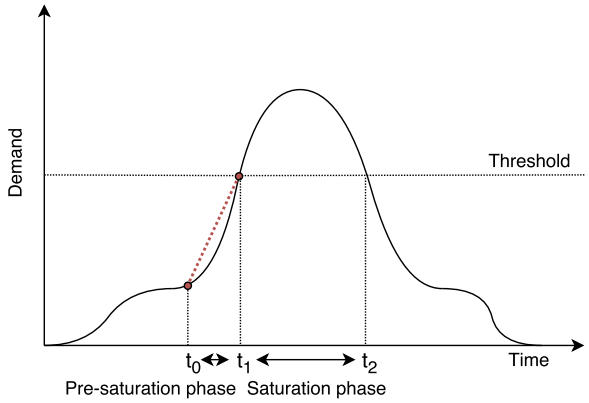


Figure 5.7: A schematic example of a system evolving to a saturated phase, whereas the pre-saturation phase is indicated as the time span before saturation, with continuous accumulation.

An algorithm is implemented to seek for action options that minimize the probability of saturation by activating more resources. Here resources are expressed as security lanes. At the same time, costs associated with opening extra security lanes should be minimized. Here, costs are expressed by the number of security lanes opened and the time duration of opening extra security lanes. Meanwhile, the amount of resources available is constrained by the number of security lanes present a RTHA.

An action option is defined by three variables:

- i: the number of extra security lanes to open.
- t_{ex}: the moment in time to open the extra lane(s).
- ΔT: the time duration of opening the extra lane(s).

The algorithm seeks for action options with an execution time t_{ex} that lays within the period of time corresponding to the pre-saturation region. The point in time where the pre-saturation phase starts is denoted by t_0 and the point in time where the system reaches the saturation phase is denoted by t_1. Thus $t_0 \leq t_{ex} \leq t_1$. The algorithm implemented, is described in Appendix C.

Valuation of Action Option Set Predictions are generated for the action options by forward simulation of the model. The set of action options are valuated based on metrics defined in Section 5.6. The valuation results are presented in Chapter 6.

Model verification

As the simulator is still in a young development stage, during the implementation of the models, several bugs and errors were encountered. Therefore, verification and improvement of the model followed an iterative process. Most of the problems encountered with the model were related to low level agent specification, whereby passenger agents could get stuck, either in the queuing area or in the security lane. Finally, after these type of errors were resolved, the simulator ran smoothly. The following verification steps are performed to verify this.

Verification by Observation One of the advantages of the visualization option in the simulator, is that it provides you with direct feedback, on whether the model performs the behaviour as expected (or not). Additionally, the graphs presented in the visualization option provide an useful tool to check the system's behavior. Based on observation of the system behavior, the following checks are performed:

Table 5.8: Verification of the total number of passengers generated in the model. The result is based on N=100 simulations.

Input: number of pax to generate	Output: generated number of pax	
	μ	σ
965	968	32

- The queuing process is observed to check whether passengers move smoothly and the queue provides a first in first out output.
- The security check processes is observed to check whether passengers pass through the security lane smoothly, and follow the right order of procedure.
- The process of opening and closing lanes is observed to check whether the action is rightly executed by the orchestration agent with the right timing, and to check whether passengers understand that a lane is opened or closed and make use of the lane at the right time or not.

Based on these observations, it is concluded that the visual behavior of the model is in accordance with what is expected.

Verification of Passenger Generation As explained, passenger generation depends on the flight schedule, the number of passengers corresponding to the flight, and the arrival time distribution, as presented in Table 5.2. To verify, whether the actual generated number of passengers in the system approaches the number of passengers that are expected to be generated, N = 100 simulations are performed to obtain a mean value of the total number of passengers generated. The result is presented in Table 5.8. As input the number of passengers equals the sum of the number of passengers over all flights, as presented in Table 5.1, which equals 965. The mean number of passengers generated as output is 968. With this check, it is verified that passengers are generated in accordance to the number of passengers corresponding to the flights part of the flight schedule. Additionally, it is verified that with the introduction of delayed flights, passengers are not generated double or are missing.

Model validation

Model validation is required to determine whether the simulation model provides an acceptable representation of the real system, given the purpose of the simulation model. First, validation of the model structure is discussed, followed by validation of the model behavior.

Validation of Model Structure Validation of the model structure is largely determined by the input parameters applied for the model. As the model aims to resemble the airport terminal operations of RTHA, input parameters are based on available data from the airport. To obtain a consistent model structure with regard to RTHA, the following input parameters were implemented in the model:

- The RTHA flight schedule of Oct. 5th 2017. This flight schedule includes accurate data of the type of aircraft, the capacity of the aircraft, the load factor, and the departure times.
- The distribution applied to model the arrival times of passengers at the airport terminal, is provided by RTHA based on their experience. However, it is not clear whether this distribution is based on empirical data or a relatively simple approximation made by the planning department of the airport.
- For the processing times of the sub-processes of the security operations, distributions based on empirical data from RTHA are applied. These distributions were developed by PhD student S.A.M. Janssen.
- The structural design of the security operations are consistent with the actual operations at RTHA. The same security lane configuration is implemented, with the same security sensors, used in the same order and operated by the same number of operators that work in the real system. Additionally, the

process of luggage drop, at the beginning of the security check-point, is implemented as a parallel process, where three passengers at the same time are able to start this activity, which is in correspondence with the operations at RTHA.

- The resources, the number of security lanes opened over time, are determined following a scheduling method provided by RTHA. The obtained resource schedule is implemented in the model.
- For the delayed flight schedule introduced, an assumption is made about how the arrival of passengers at the airport terminal is adjusted. It is assumed that passengers already at the airport will continue following their original planning, while passengers not yet arrived at the airport follow a new arrival distribution based on the delayed departure time. This assumption is made, as no data is available to provide more insight in how the arrival pattern might adjust. However, the main goal of the simulation study is to illustrate how the effects of anticipation of a disrupted passenger demand can be analyzed. For this purpose the assumption is deemed acceptable.

Validation of Model Behavior The model is calibrated to obtain a processing rate of 2.6 pax/min for the security check-point operations. The calibration process is explained in Section 5.3, and the resulting output values for the security process are presented in Table 5.4.

After implementation of the calibrated processing rates and the computed resource schedule, based on the scheduling method provided by RTHA, it is expected that when the passenger flow, corresponding to the flight schedule of that day, is provided as an input, that the global output of the system stays within its performance boundaries. This means, that it is expected that, when the model implements all aspects as RTHA this would do, that the passenger flow through the security check-point should run smoothly without accumulation of passengers in the queue exceeding a performance threshold. This expectation is confirmed by the simulation results of the system in nominal conditions. The result is presented in Figure 5.8. How the threshold value is determined is described in Section 5.6. From the graph it can be concluded that the averaged system behavior remains within its performance boundary, as would be expected for the real system under nominal conditions.

Additionally, the input and output variation of the model are compared. The input of the system is represented by the passengers that are generated in the system over time. The output is measured as the number of passengers in the queue over time. It is expected that the standard deviation for the output is larger than for the input, as the processing time at the security check-point depends on both processing time distributions and distributions that determine whether passengers need extra security checks. This leads to variations in the overall processing time. As a consequence, higher variations are expected for the number of passengers queuing. In Figure 5.9 the standard deviation for the input and output over time is presented. The result is based on $N = 100$ simulation runs. From the graph it can be deduced that the standard deviation for input is indeed smaller than the standard deviation of the output. Further, a significant peak is shown for the standard deviation of the output when it is most crowded in the queue (see Figure 5.8), as here randomness of the sub-processes have the most influence on the waiting time of passengers, and therefor the queue length.

Finally, to evaluate how this case study and the results fit within the security operations of RTHA in practice, interviews are conducted with security check-point team-leaders and duty security managers of the airport. The evaluation with RTHA is presented in Chapter 7.

5.6. Analysis

In order to analyze the effects of anticipation at the security check-point for the disrupted scenario proposed in the case study, simulations are performed with the AATOM simulator including the developed models as described in the previous sections. Essentially, to study the effects of anticipation, the security check-point queue is analyzed over time. In this section it is first outlined why this analysis is conducted by simulations of an agent-based model, compared to the possibility to analyze the queue behavior with queuing theory. Additionally, the metrics applied to analyze the results are presented.

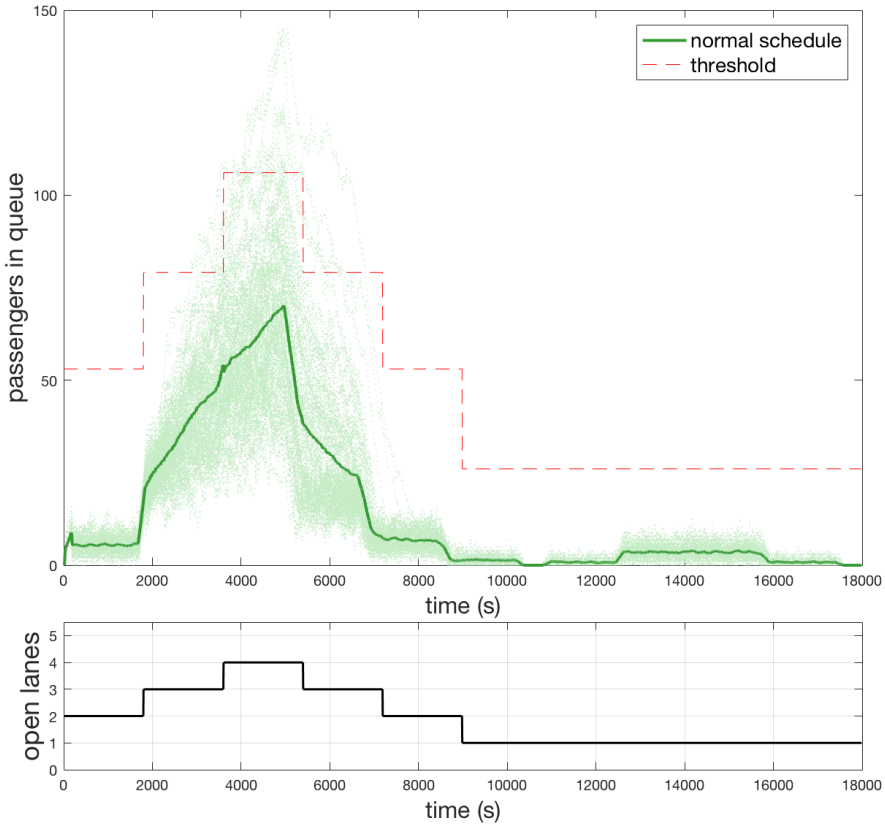


Figure 5.8: The global output (in green) of the model for nominal input of passenger arrival times, resource schedule, and the calibrated processing rates of the security check-point. The threshold is denoted in red. The sub-graph below shows the number of security lanes scheduled over time. The results are obtained for N=100 simulations.

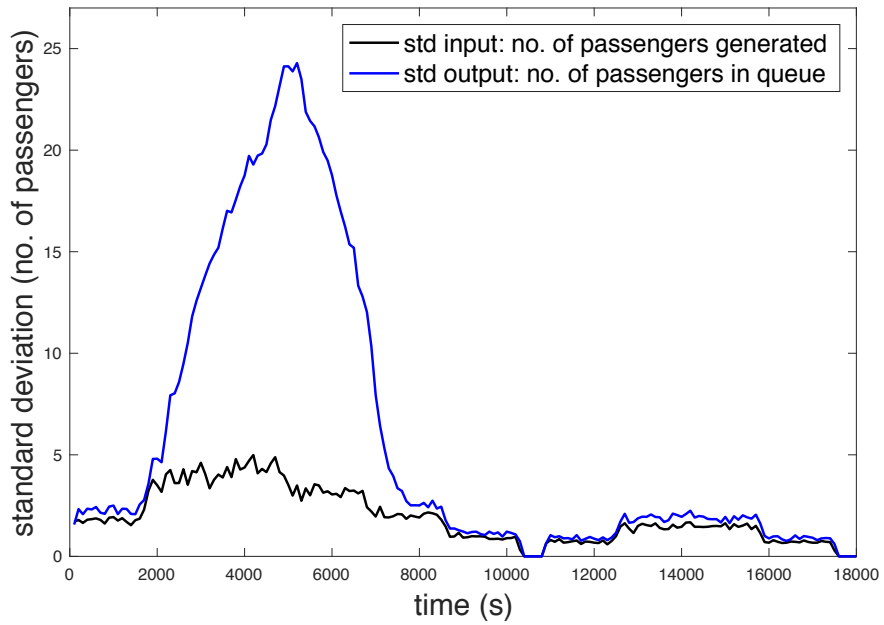


Figure 5.9: Comparing the standard deviation of model input, number of passengers generated, and model output, number of passengers in the queue, over time.

Agent-based modelling versus queuing theory

The main reason for the development of an agent-based model of the airport terminal processes is that it allows to represent the heterogeneous system in more realistic manner, compared to top down and normative approaches, as is often found for techniques from operations research. In general, queuing theory describes a system from an aggregated point of view while assuming static conditions, exogenous arrival and service rates analyzed in steady-state.

The problem at hand for this case study scenario, even though it does not yet includes as much heterogeneous features possible, it provides already a complex system to handle with queuing theory. The complex aspects present in the model are:

- The arrival rate λ of passengers in the airport terminal varies over time. The variation of λ depends on the number of passengers that correspond to a specific flight in the flight schedule and the departure time of the flight, while the departure time of a flight can change over time as well for delayed flights scenario's.
- The process rate of a server (the security lane) is the result of a combination of five sub-processes, of which two are optional. Whether these processes are executed depends on a random distribution. Additionally, for each of the sub-processes, the time a sub-process will take depends on random distributions, based on empirical data.
- One queue is implemented, wherein passengers wait until a place becomes available at the first sub-process of the security lane. However, in between the sub-processes, passengers also assume queuing behaviour waiting for the next process to become available. Additionally, for the first step in the security process, parallel service is possible, as three passengers prepare their luggage drop in parallel.
- The number of servers (security lanes) available, varies over time, as the lanes are closed and opened depending on the resource schedule and depending on the adaptive behaviour of the anticipatory agent.

Capturing this model in a queuing model, becomes rather complicated and approximations can not be avoided, losing some of the realistic system characteristics. Additionally, the methodology presented in this Chapter, provides the first steps towards a realistic airport terminal operations model. Eventually, more heterogeneity can be implemented that represents individual passenger behaviour or security operator behaviour. For instance, the work of MSc student A. Knol focuses on fatigue models that are able to represent the security operator's performance more realistically over time. This type of system behavior can not be represented by queuing theory.

Metrics

To analyze the data obtained from the simulations, metrics are proposed to provide an indication of the saturation risk of the system. In order to indicate when a system's capacity is saturated, a threshold is to be defined. For this case study, the IATA waiting standard corresponding to a reasonable level of service is applied. IATA states that waiting times exceeding 10 minutes (for Economy Class), at the security checkpoint, indicates underprovision and requires reconfiguration [54].

A threshold function defines what maximum number of passengers is allowed to be in the queue before the system is defined as being in a saturated state. The threshold value is determined by Equation 5.8:

$$\text{threshold value} = r(n) \times 10 \tag{5.8}$$

Here, $r(n)$ denotes the average processing rate (pax/min) of the system for n open security lanes, as presented in Table 5.4. Assuming this threshold value (expressed in number of passengers), the system is defined to be in a saturated state, when the waiting time exceeds 10 min. The threshold value over time varies with the number of security lanes opened.

To obtain insight in the behaviour of the system, for each scenario, N=100 simulations are performed. A scenario refers to either the initial prediction of future system states based on the announcement of the delayed flights, or to the predictions generated for the set of action options. The following measures are applied for the analysis of the results for each scenario:

- **Probability of Saturation:** The probability of saturation is estimated as:

$$p(\text{sat}) = \frac{card(M)}{card(N)} \tag{5.9}$$

  Here, M denotes the set of simulations where the system reaches a saturated state, thus exceeds the threshold value. $card(M)$ provides the cardinality/size of the set. N is the set containing all simulations performed, with $card(N)$ the number of simulations performed: $card(N) = 100$.

  The next measures are proposed to analyze the severeness of the saturation, if the system reaches a saturated state.

- **Average Duration of Saturation:** The average duration of saturation provides insight in how long the system is saturated for a specific scenario. This is calculated as follows:

$$\overline{T}_{\text{sat}} = \frac{\sum_{m \in M} T_{\text{sat}}(m)}{card(M)} \tag{5.10}$$

  Here, for each simulation $m \in M$ which reaches a saturated state, the total time the system is saturated is measured as $T_{\text{sat}}(m)$. This time is summed for M and divided by the total number of saturated simulations $card(M)$.

- **Average Waiting Time:** The average waiting time is calculated based on the average saturation value, the number of passengers in the queue that exceeds the threshold value, and average processing rate of the servers at the moment on saturation. The average saturation value is calculated as follows:

$$\overline{V}_{\text{sat}} = \frac{\sum_{m \in M} \overline{V}_{\text{sat}}(m)}{card(M)} \tag{5.11}$$

  With, $\overline{V}_{\text{sat}}(m)$ the average normalized saturation value over time for simulation $m \in M$. The saturation value is normalized by the number of lanes opened over time. To calculate the average waiting time the following equation is applied:

$$\overline{W}_{\text{sat}} = \overline{V}_{\text{sat}} \times r(1) \tag{5.12}$$

  Here, $r(1)$ denotes the average processing rate if one security lane is opened. This rate is applied as $\overline{V}_{\text{sat}}$ is calculated based on normalized saturation values over time.

- **Average Number of Missed Flights:** The average number of missed flights, is a global system characteristic and provides an extra measure to indicate whether the waiting times at the security check-point are within reasonable limits. For each simulation the number of passengers that miss their flight, is counted. For each scenario, the average number of missed flights is calculated as follows:

$$\overline{MF} = \frac{\sum_{n \in N} MF(n)}{card(N)} \tag{5.13}$$

  Here, $MF(n)$ provides the number of missed flights for each simulation $n \in N$.

Additionally, for each of the predictions generated for a potential action option, the trade-off relations between the obtained metric values and corresponding costs is presented. Costs are represented as: the number of lanes that are opened, and the time duration of having these extra lanes opened. Based on these trade-off relations a best fitting decision can be made on what action option to execute.

6

Results

In this chapter the simulation results are presented. First, in Section 6.1 the initial prediction results, generated after the delayed flights are announced, are presented. Additionally, an overview is provided of the action set determined, based on the initial prediction. In Section 6.2, the prediction results for each of the action options in the action set are presented and analyzed. In Section 6.3 trade-off relations, that are obtained based on the action option valuation, are presented. Finally, in Section 6.4 conclusions are drawn based on the analysis of the results.

6.1. Initial Prediction

The initial prediction represents the expected effects of the disrupted flight schedule, after the announcement of the two delayed flights. In Figure 6.1, the evolution of passengers in the security check-point queue over time, is presented. The graph shows in green the expected conditions with no disrupted flight schedule and in red the expected passenger numbers for the disrupted schedule, while no adaptive actions are undertaken. Further, the first sub-graph below, shows how the influx of passengers shifts due to the disrupted flight schedule. In green the influx of passengers under nominal conditions is presented, and in red the passenger influx for the disrupted flight schedule scenario. The influx of passengers is measured as the average influx for $N = 100$ simulations. Each bar represents the mean number of passengers arriving at the airport per minute. Finally, the second sub-graph shows the number of security lanes opened over time following its original resource schedule.

Based on the graphs, it is readily clear that the shift in arrival of passengers of the delayed flights, does not match with the scheduled resources in the original planning, resulting in a high number of passengers accumulating in the queue. To analyze the severity of this accumulation, the data for the disrupted schedule is compared to the threshold value in Figure 6.2. The threshold value is depicted with a red dotted line. As shown by the figure, a relatively large part of the graph exceeds the threshold value. The performance indicators quantifying the saturation are presented in Table 6.1. From the table it can be deduced that the probability of saturation $p(\text{sat}) = 1$, hence for all N=100 simulations, the system reached a saturated state. The severity of saturation is expressed by the average duration of saturation $\overline{T}_{\text{sat}} = 119.18$ min, and the average waiting time $\overline{W}_{\text{sat}} = 32.58$min. Additionally, as a global performance indicator, the average number of passengers that end up to miss their flight, due to the long waiting times, is presented: $\overline{MF} = 124.88$ pax. Based on these values, it can be concluded that adaptation of the system is required, in order to maintain a desired performance level.

In Figure 6.2, the pre-saturation phase corresponding to the saturation phase is indicated by the blue linear fit. From the graph it can be deduced that the pre-saturation phase starts at 06:26 AM, the saturation phase starts at 06:45 AM, and maximum saturation is reached at 07:23 AM. The average saturation rate equals 0.0552 pax/s. Based on the data obtained from the initial prediction, action options are generated following the

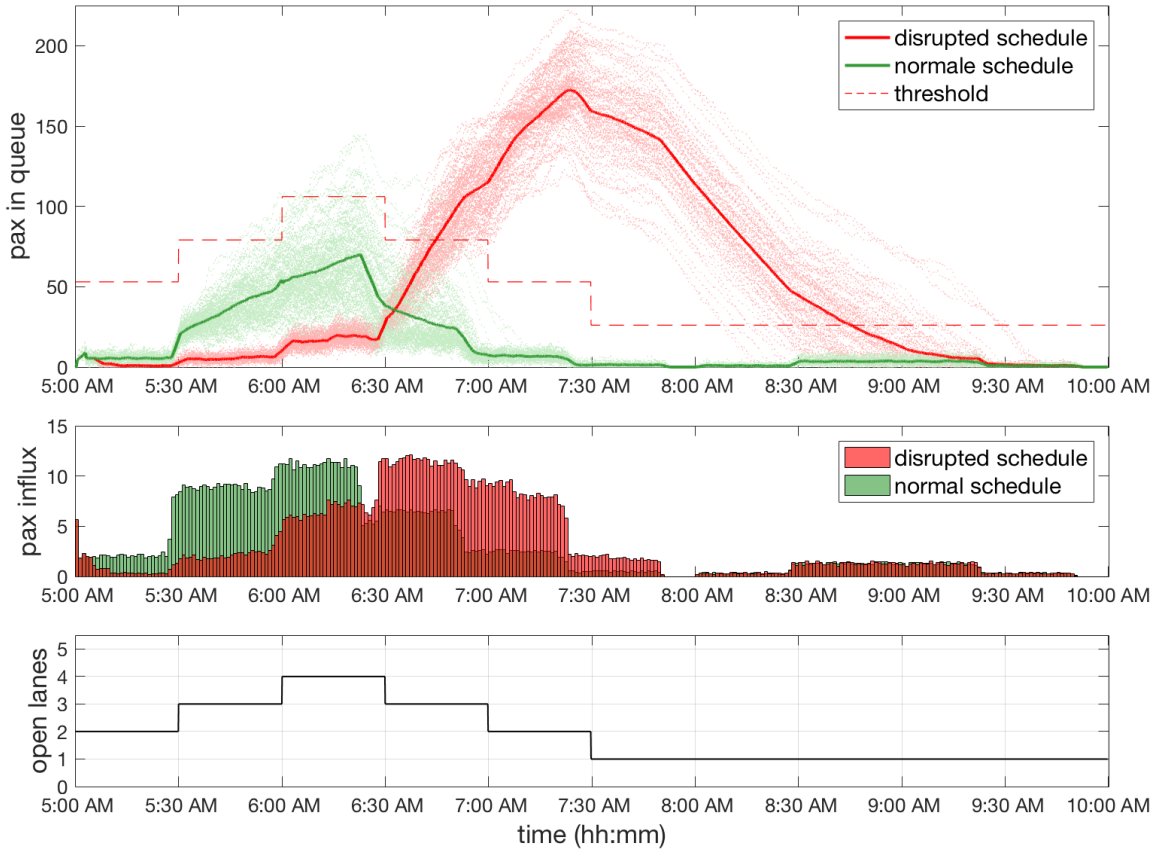


Figure 6.1: The first sub-graph shows passenger accumulation in the queue over time, with passengers in queue for the normal schedule (in green) and the expected passenger accumulation for the disrupted scenario (in red). The second sub-graph shows the shift of influx of passengers between the nominal scenario and disrupted scenario. Here, each bar represents the mean number of passengers arriving at the airport per minute. The third sub-graph shows the number of security lanes opened, over time. The results are based on N=100 simulations.

Table 6.1: Initial prediction results of the performance indicators for a disrupted schedule. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
1	119.8	20.96	32.58	3.29	124.88	26.08

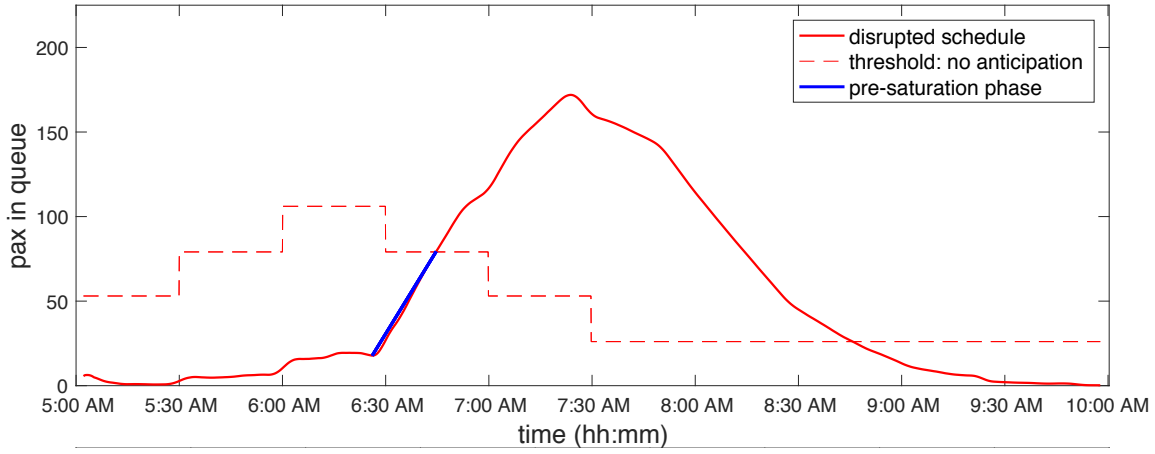


Figure 6.2: Passengers accumulation in the queue over time, for the disrupted scenario. The threshold value is indicated with the red dotted line, indicated when the system is saturated. The pre-saturation phase is denoted by the blue line. The pre-saturation phase starts at 06:26 AM, the saturation phase starts at 06:45 AM, and maximum saturation is reached at 07:23 AM. The saturation rate equals 0.0552 pax/s.

Table 6.2: The proposed action option set, based on the initial prediction. The action option set including action option 1 up and till 6 is generated following an action option generation algorithm as presented in Appendix C. Action option 7 is based on the result of action option 6 whereas the time duration applied for action option 6 is extended till the time of maximum saturation. Additionally, action option 8 is based on the result of action option 5, whereas the time of duration is extended till the time of maximum saturation.

Action option	Number of extra lanes to open	Time of opening	Time of closing
1	1	06:26	07:23
2	1	06:26	08:05
3	1	06:26	08:27
4	2	06:30	07:00
5	2	06:35	07:04
6	2	06:45	07:14
7	2	06:45	07:23
8	2	06:35	07:23

algorithm presented in Appendix C. As a result, eight potential actions are proposed for valuation, they are presented in Table 6.2.

6.2. Valuation of Options

For each of the action options as presented in Table 6.2, simulations are conducted to obtain a prediction of the expected passenger demand in the queue over time. The results for each action are discussed individually.

Results Action Option 1 Action option 1 is defined as: opening one extra lane at 06:26 AM and close the extra lane at 07:23 AM, the extra lane is opened for 57 min. The result obtained by this action is presented in Figure 6.3. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 1. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time is significantly improved by this action. However, the mean number of passengers in the queue over time, still exceeds the threshold value, for a relatively small amount of time. Additionally, the graph shows how a significant number of the simulations exceeds the threshold value, thus there is a probability the system reaches a saturated state.

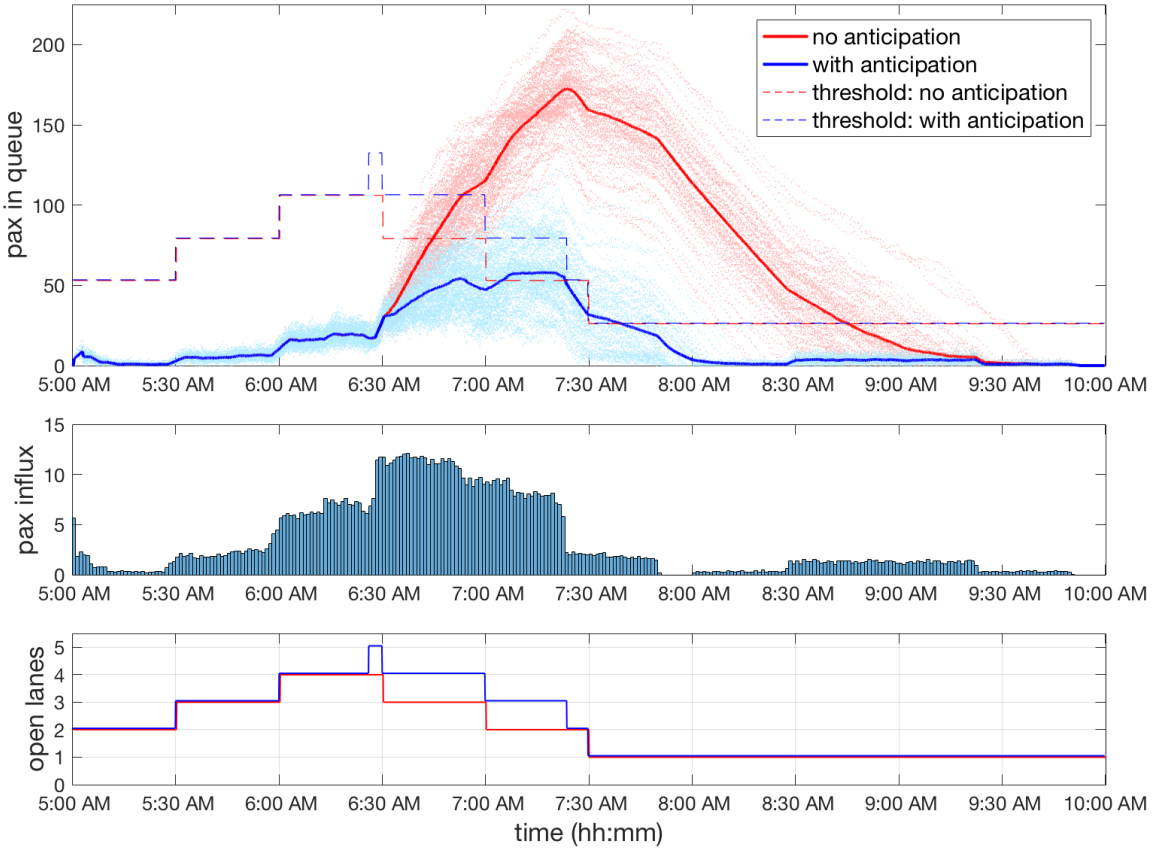


Figure 6.3: The prediction result for action option 1. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 1 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

Table 6.3: The performance indicators quantifying the risk and the severity of action option 1. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.58	25.78	18.53	17.57	3.19	6.47	12.21

Action 1 is executed at 06:26 AM, which is at the beginning of the pre-saturation phase. As the system did not yet reach its saturated state, while executing the action, the action can be characterized as a proactive reaction. Opening one extra lane corresponds to adding an average processing rate of 2.71 pax/min (0.0451 pax/s) to the system. As the average saturation rate equals 0.0552 pax/s, a resulting saturation rate of approximately 0.0101 pax/s remains to exist during the time span of 57 min. This is reflected by the behaviour of the graph. At the moment an extra security lane is opened, the gradient of the graph decreases significantly, while remaining a positive value. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.3.

Results Action Option 2 Action option 2 is defined as: opening one extra lane at 06:26 AM and close the extra lane at 08:05 AM, this means that the extra lane is opened for 99 min. The result obtained by this action is presented in Figure 6.4. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 2. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides a significant improvement compared to the situation without anticipation. Additionally, the mean number of passengers in the queue over time stays below the threshold value. However, the graph shows how a significant number of the simulations still exceeds the threshold value, thus there is a probability the system reaches a saturated state.

Similar to action 1, action 2 is executed at 06:26 AM, which is at the beginning of the pre-saturation phase. As the system did not yet reach its saturated state, while executing the action, the action can be characterized as a proactive reaction. Opening one extra lane corresponds to adding an average processing rate of 2.71 pax/min (0.0451 pax/s) to the system. As the average saturation rate equals 0.0552 pax/s, a resulting saturation rate of approximately 0.0101 pax/s remains to exist during the time span of 99 min. This is reflected by the behaviour of the graph. At the moment an extra security lane is opened, the gradient of the graph decreases significantly, while remaining a positive value. The duration of opening the extra lane is longer compared to action 1. The effect of this opening time is that the mean passenger accumulation over time stays below the threshold value. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.4.

Table 6.4: The performance indicators quantifying the risk and the severity of action option 2. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.27	13.25	9.92	10.35	0.37	0.33	1.33

Results Action Option 3 Action option 3 is defined as: opening one extra lane at 06:26 AM and close the extra lane at 08:27 AM, this means that the extra lane is opened for 121 min. The result obtained by this action is presented in Figure 6.5. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 3. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides a significant improvement compared to the situation without anticipation. Similar to action 2, the mean number of passengers in the

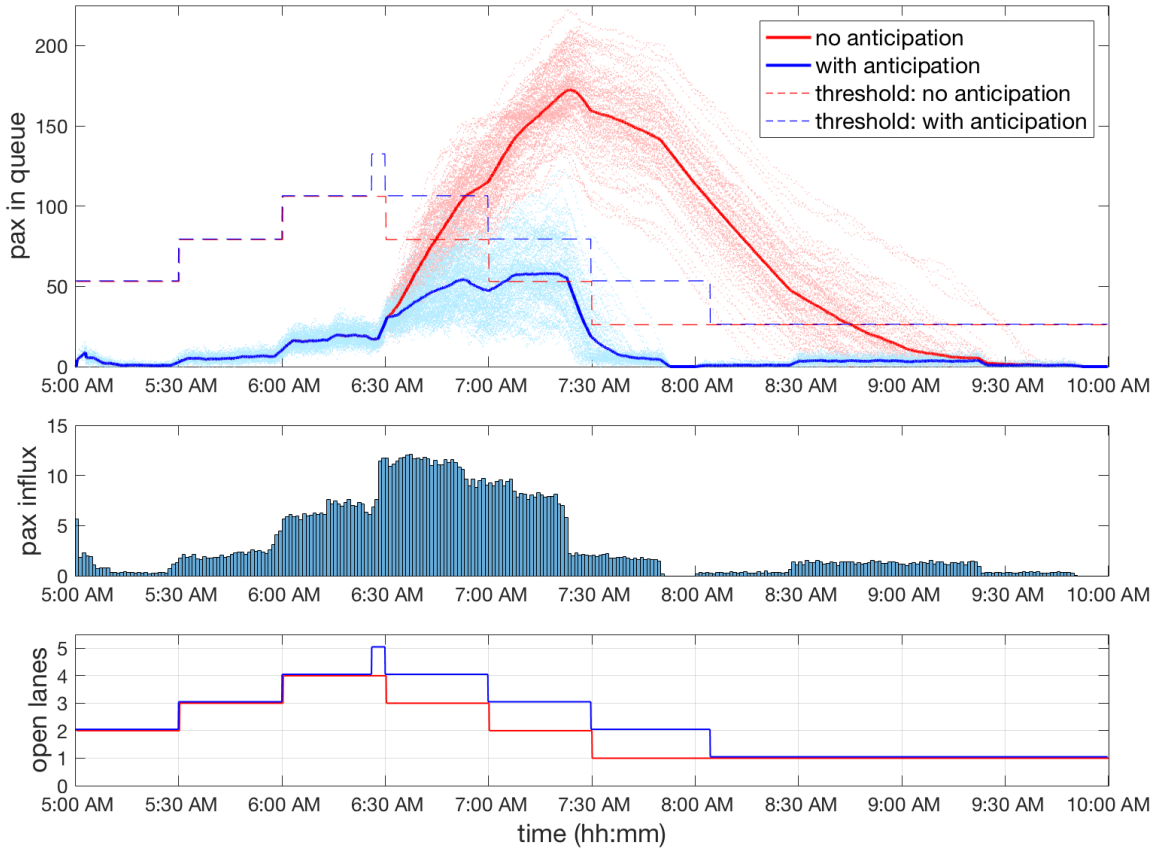


Figure 6.4: The prediction result for action option 2. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 2 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

queue over time stays below the threshold value. However, the graph shows how a significant number of the simulations still exceeds the threshold value, thus there is a probability the system reaches a saturated state.

Similar to action 1 and action 2, action 3 is executed at 06:26 AM, which is at the beginning of the pre-saturation phase. Therefore, the action can be characterized as a proactive reaction. The only difference between action 2 and action 3 is the duration the extra security lane is opened, which is for action 3 longer than for action 2. However, from Table 6.5, it can be deduced that the longer opening time does not influence the saturation risk and saturation severity, compared to the execution of action 2.

Table 6.5: The performance indicators quantifying the risk and the severity of action option 3. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.27	13.25	9.92	10.35	0.37	0.33	1.33

Results Action Option 4 Action option 4 is defined as: opening two extra lanes at 06:30 AM and close the extra lanes at 07:00 AM, this means that the extra lanes are opened for 30 min. The result obtained by this action is presented in Figure 6.6. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 4. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane config-

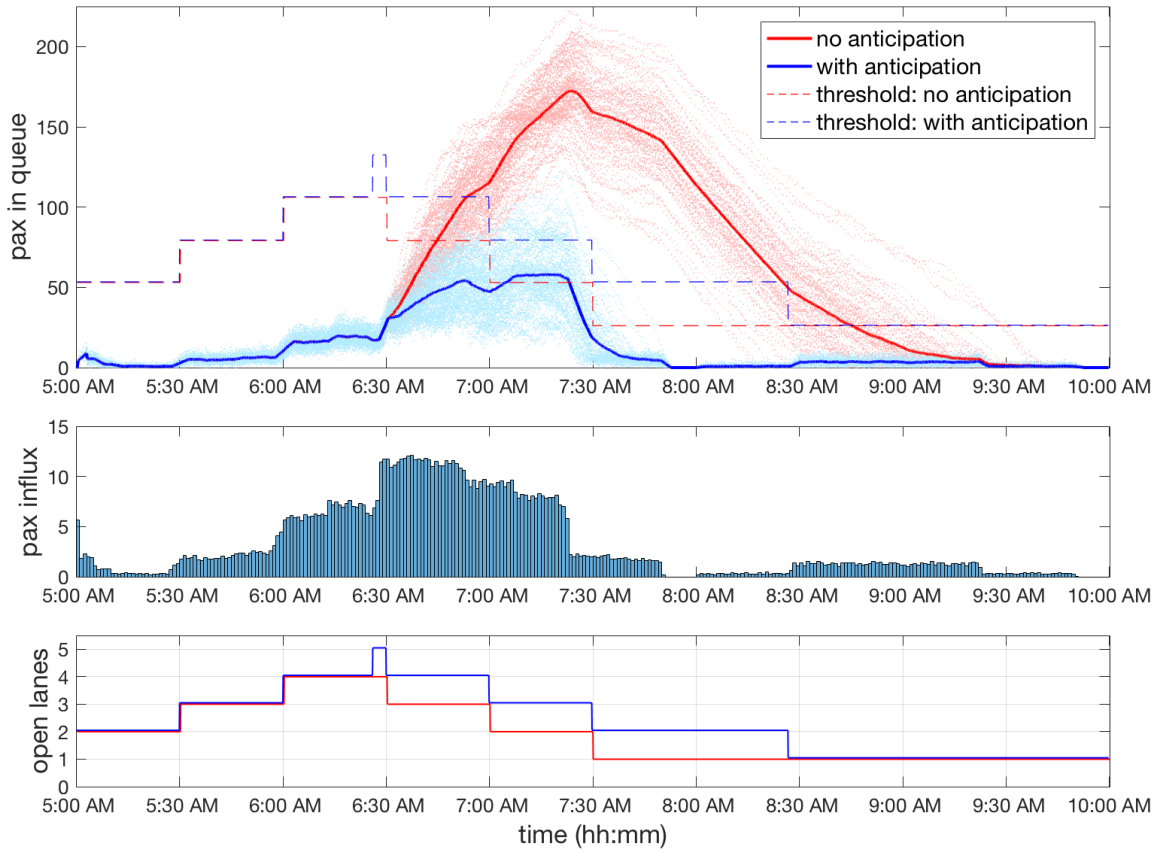


Figure 6.5: The prediction result for action option 3. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 3 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

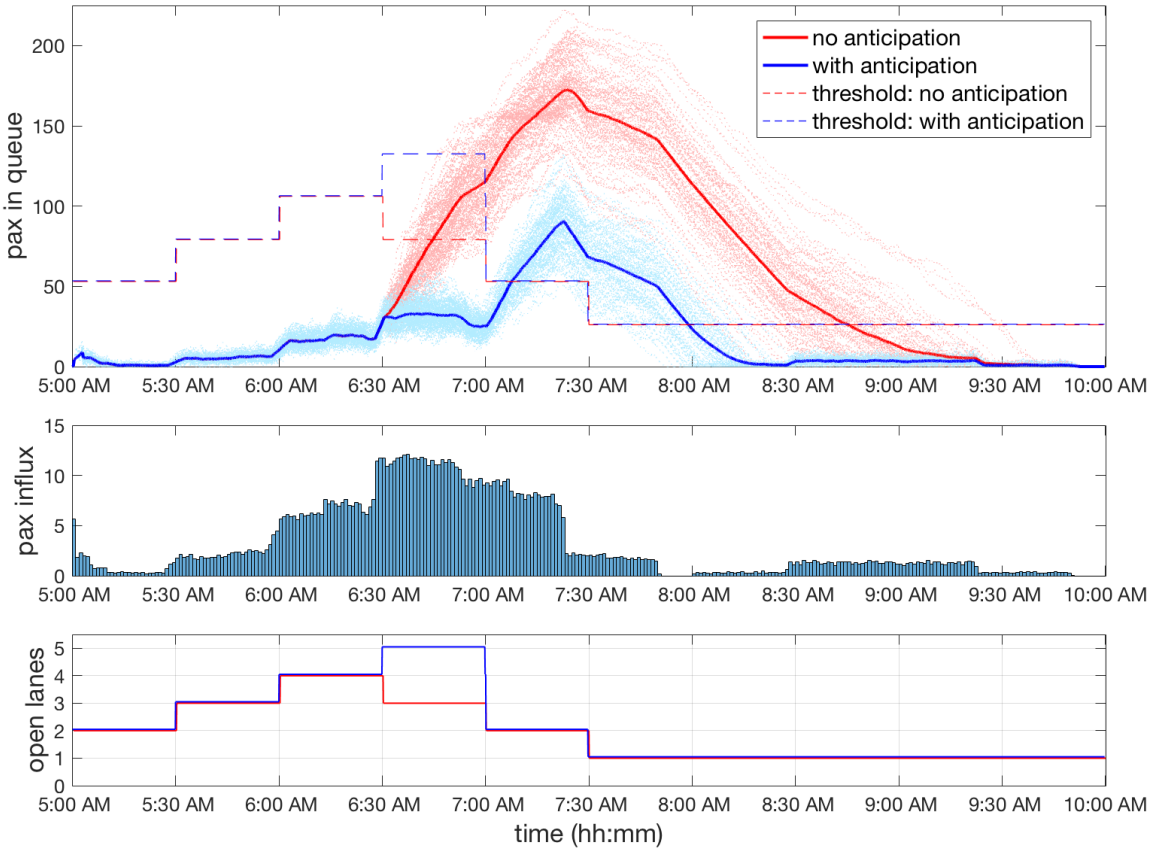


Figure 6.6: The prediction result for action option 4. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 4 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

uration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides an improvement compared to the situation without anticipation. However, the mean number of passengers in the queue over time, exceeds the threshold value for a relatively large part of the graph. Additionally, the graph shows how a significant number of the simulations exceeds the threshold value, resulting in a probability that the system reaches a saturated state.

Action 4 is characterized as a proactive reaction, executed at 07:00 AM, which is after the start of the pre-saturation phase and before the start of the saturation phase. As now two lanes are opened, the timing can be moved closer to the start of the saturation phase. The resulting average saturation rate, after adding the processing rate of two security lanes (0.0883 pax/s), is -0.0331 pax/s. This means that it is expected that, after opening the two extra lanes, the curve will follow a negative gradient. The graph in Figure 6.6, shows, after the opening of extra lanes, a gradient approaching zero. This means that approximately the same number of passengers enter the queue as are processed. The queue behavior reflected by the graph can be explained as follows: The queue area implemented in the AATOM simulator is relatively large, to be able to register the large number of passengers queuing, for the disrupted scenario. This means, that passengers are registered as queuing passengers as well if they are moving through the queue (and are not standing still). In the case of action 4, when the two extra security lanes are opened it appears that no significant accumulation (of waiting passengers) needs to be processed, and that the number of passengers in the queue are able to move through the queue without the need to stand still. However, it is also shown by the graph that as soon the extra two lanes are closed, the system's processing rate is decreased again and passengers will accumulate in the queue, resulting in a saturated state. This means that the impact of opening the two extra lanes has not

been sufficient enough to avoid saturation. In the presented results of action 5, 6, 7 and 8, it is shown how both the timing of the action and the duration of the action are important factors to obtain the desired result. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.6.

Table 6.6: The performance indicators quantifying the risk and the severity of action option 4. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
1	49.52	11.36	19.89	1.54	32.35	17.99

Results Action Option 5 Action option 5 is defined as: opening two extra lanes at 06:35 AM and close the extra lanes at 07:04 AM, this means that the extra lanes are opened for 29 min. The result obtained by this action is presented in Figure 6.7. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 5. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides an improvement compared to the situation without anticipation. However, the mean number of passengers in the queue over time, exceeds the threshold value for a relatively large part of the graph. Additionally, the graph shows how a significant number of the simulations exceeds the threshold value, resulting in a probability that the system reaches a saturated state.

Action 5 is characterized as a proactive reaction, executed at 06:35 AM. Comparing action 5 to action 4, the duration of opening two extra lanes is approximately the same. The timing of the execution of the action, however, differs. For action 5 the action is executed later in time, still before the saturation phase is reached. Due to the later execution, the queue has reached a higher accumulation level, compared to the situation for action 4. Therefore, now a negative gradient is observed directly after opening the two extra lanes. The negative gradient holds, until the moment the two extra lanes are closed again. The positive impact of this action is larger compared to the impact of action 4. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.7.

Table 6.7: The performance indicators quantifying the risk and the severity of action option 5. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.99	36.38	14.68	10.08	0.07	18.36	15.80

Results Action Option 6 Action option 6 is defined as: opening two extra lanes at 06:45 AM and close the extra lanes at 07:14 AM, this means that the extra lanes are opened for 29 min. The result obtained by this action is presented in Figure 6.8. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 6. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides an improvement compared to the situation without anticipation. Additionally, the mean number of passengers in the queue over time stays below the threshold value. However, still a significant number of the simulations exceeds the threshold value, resulting in a probability that the system reaches a saturated state.

Action 6 is characterized as a reactive reaction, executed at 06:45, the starting time of the saturated state. Comparing action 6 to action 4 and 5, the duration of opening the two extra lanes is approximately the same. The timing of the execution of the action, however, differs. For action 6 the action is executed later in time, right at the starting time of the saturation phase. Due to the later execution, the queue has reached the

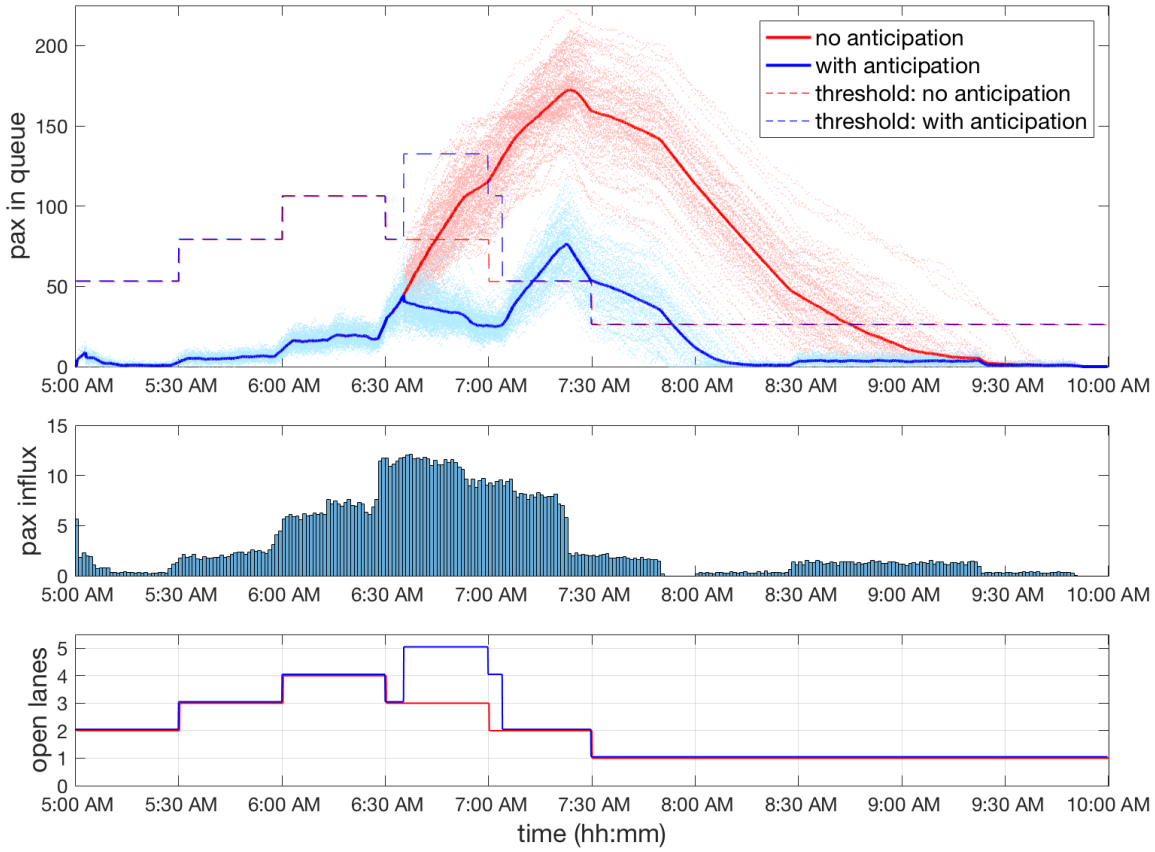


Figure 6.7: The prediction result for action option 5. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 5 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

threshold accumulation value. Now, by opening two lanes a negative gradient is observed. Over the time span the two lanes are opened, a significant part of the accumulation is processed. After closing the two lanes, the number of passengers in the queue increases again until the moment in time is reached that the saturation, for the initial prediction (in red), has reached its maximum value. After this peak is reached, the trend of the initial prediction (in red) is followed by the blue curve. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.8.

Table 6.8: The performance indicators quantifying the risk and the severity of action option 6. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.63	12.22	11.70	11.45	1.32	2.06	7.09

Results Action Option 7 Action option 7 is defined as: opening two extra lanes at 06:45 AM and close the extra lanes at 06:23 AM, this means that the extra lanes are opened for 38 min. The result obtained by this action is presented in Figure 6.9. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 7. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides an improvement com-

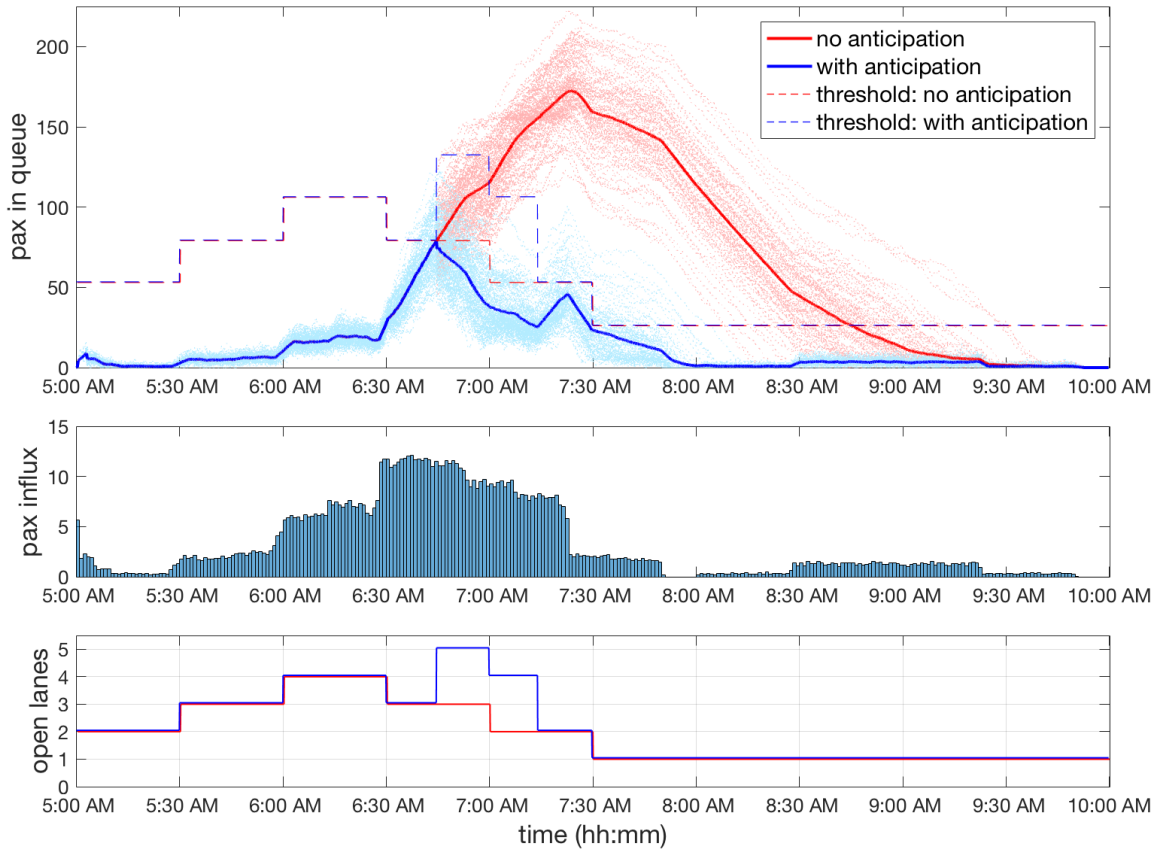


Figure 6.8: The prediction result for action option 6. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 6 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

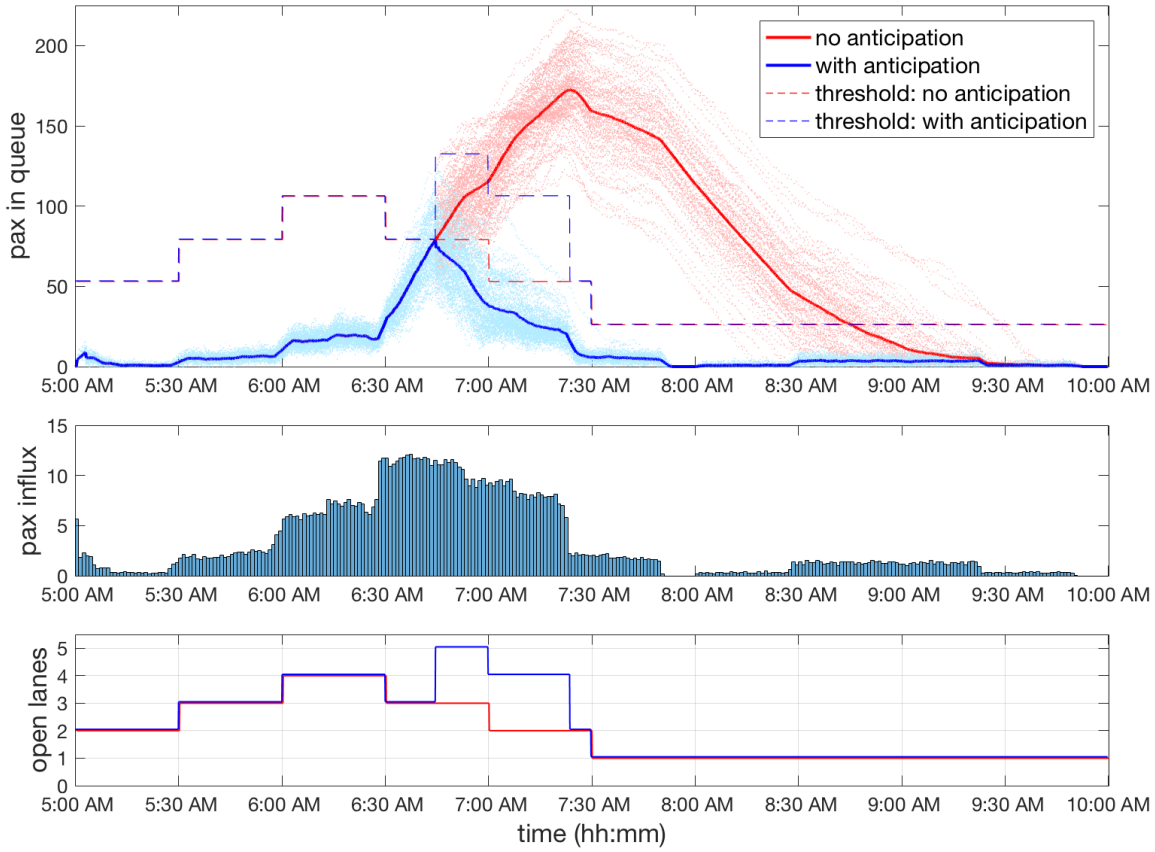


Figure 6.9: The prediction result for action option 7. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 7 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

pared to the situation without anticipation. Additionally, the mean number of passengers in the queue over time stays below the threshold value. However, still a number of the simulations exceeds the threshold value, resulting in a probability that the system reaches a saturated state.

Action 7 is characterized as a reactive reaction, executed at 06:45 AM, the starting time of the saturated state. Comparing action 7 to action 6, the timing of the action execution is the same. However, the duration of the actions differ. For action 6 the two lanes are opened for a longer time, namely until the moment in time is reached that the initial prediction (in red) has reached its maximum value. After this point in time the number of passengers in the queue will decrease, as less people are entering the queue. The resulting effect of keeping the two extra security lanes open until this moment, is that the small accumulation that occurs after closing the two lanes with action 5, does not occur for action 6. Additionally, this results in a lower probability and impact of saturation. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.9.

Table 6.9: The performance indicators quantifying the risk and the severity of action option 7. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0.49	2.96	2.15	10.48	0.33	0	0

Results Action Option 8 Action option 8 is defined as: opening two extra lanes at 06:35 AM and close the extra lanes at 07:23 AM, this means that the extra lanes are opened for 48 min. The result obtained by this action is presented in Figure 6.10. The figure shows in red the initial prediction for passenger accumulation, and in blue the expected passenger accumulation under execution of action 8. The thresholds for each of these graphs is presented in corresponding colors. Additionally, in the sub-graph below, the change in lane configuration is presented, with in red the original lane configuration, and in blue the adjusted lane configuration. From the graph it can be deduced that the passenger accumulation over time provides an improvement compared to the situation without anticipation. Additionally, the mean number of passengers in the queue over time stays below the threshold value, and none of the performed simulations exceeds the threshold value.

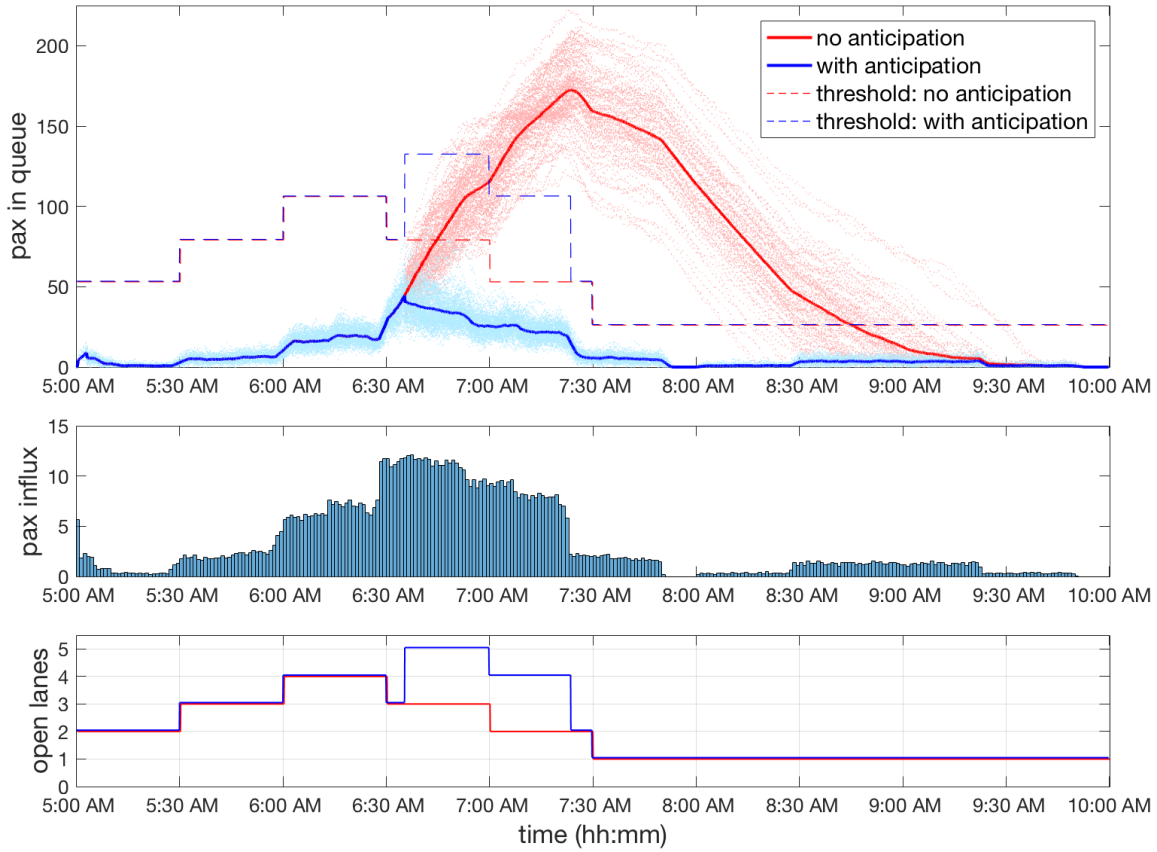


Figure 6.10: The prediction result for action option 8. In the first sub-graph the initial prediction for passenger accumulation is presented in red. In blue the expected passenger accumulation under execution of action option 8 is presented. The thresholds for each of these graphs is presented in corresponding colors. In the second sub-graph the passenger influx is presented. Each bar represents the number of passengers arriving at the airport per minute. In the third sub-graph, the change in lane configuration is presented. With in red the original lane configuration, and in blue the adjusted lane configuration.

Action 8 is characterized as a proactive reaction, executed at 06:35 AM, before the starting time of the saturation phase. Thus action 8 is executed at the same starting time as action 6. However, the duration of opening two extra lanes is longer compared to the duration of action 6. For action 8 the extra lens are opened until the moment in time is reached that the initial prediction (in red) has reached its maximum value. After this point in time the number of passengers in the queue will decrease, as less people are entering the queue. The overall result is valuated as best, with regard to the positive impact of the action. However, this action asks for most resources, namely two lanes. Additionally, the duration the two lanes are opened is the longest for this action, compared to the other actions that open two lanes. The performance indicators quantifying the risk and the severity of saturation are presented in Table 6.10.

In table 6.11 the results of all action options evaluated are summarized.

Table 6.10: The performance indicators quantifying the risk and the severity of action option 8. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations.

p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
	μ	σ	μ	σ	μ	σ
0	0	0	-	-	0	0

Table 6.11: This table summarizes the results for all eight action options analyzed. Here, p(sat) denotes the probability of saturation, $\overline{T}_{\text{sat}}$ denotes the average saturation duration, $\overline{W}_{\text{sat}}$ denotes the average waiting time during the saturation phase, and $\overline{MF}$ represents the average number of passengers that misses their flight. These values are obtained by performing N=100 simulations for each action.

Action option	Number of lanes	Opening time (hh:mm)	Duration (min)	p(sat)	$\overline{T}_{\text{sat}}$ (min)		$\overline{W}_{\text{sat}}$ (min)		$\overline{MF}$ (pax)	
					μ	σ	μ	σ	μ	σ
1	1	06:26	58	0.57	25.78	18.53	17.57	3.19	6.47	12.21
2	1	06:26	99	0.27	13.25	9.92	10.35	0.37	0.33	1.33
3	1	06:26	121	0.27	13.25	9.92	10.35	0.37	0.33	1.33
4	2	06:30	30	1	49.52	11.36	19.89	1.54	32.35	17.99
5	2	06:35	29	0.99	36.38	14.68	10.08	0.07	18.36	15.80
6	2	06:45	29	0.63	12.22	11.70	11.45	1.32	2.06	7.09
7	2	06:45	38	0.49	2.96	2.15	10.48	0.33	0	0
8	2	06:35	48	0	0	0	-	-	0	0

6.3. Decision-making

To be able to make a decision on what action option to choose as anticipatory agent, a trade-off has to be made between the cost corresponding to the extra resources and the probability and severity of saturation that is accepted by the agent. In Figure 6.11 trade-off relations are presented that are derived from the valuation data presented in Table 6.11. Trade-off relations are shown for opening one extra security lane at 06:26 AM, and two extra lanes at 06:35 and 06:45 AM. The trade-off relations are presented for three performance indicators: probability of saturation $p(sat)$, the average time duration of saturation $\overline{T}(sat)$, and the average waiting time during the saturation phase $W(sat)$. The number of extra security lanes and the duration of opening these lanes represent the costs. The relations presented for opening one extra security lane are based on the data corresponding to action option 1, 2 and 3. The relations presented for opening two extra lanes at 06:35 AM are based on action option 5 and 8. For action option 8 the average waiting time during the saturation phase is undefined as no saturation phase exists when this action is executed. This is represented in Figure 6.11 by the dotted blue line. The relations presented for opening two extra lanes at 06:45 AM are based on action option 6 and 7. Additionally, the probability of saturation, average waiting time and average duration of saturation, if no action is executed is included in the graphs. This corresponds to a time duration of zero. Further, is can be deduced from Figure 6.9 that, for opening two security lanes at 06:45 AM for longer than 38 minutes will not further improve the performance indicators. The same holds for opening two extra security lanes at 06:35 AM for longer than 48 minutes as for action 8 already an optimal result is obtained with regard to probability of saturation and the average time of saturation. This is reflected in the presented trade-off curves.

The trade-off relations are derived based on limited data points, and should therefore be understood as a rough approximation of the trade-off relations. Despite the limited data, the presented trade-off relations illustrate that the applied methodology allows to obtain trade-off relations, which can be fed to the anticipatory model for decision-making and could support industry in decision-making.

6.4. Conclusions

In table 6.11 the results of the action options are summarized. Based on the results, the following conclusions are drawn:

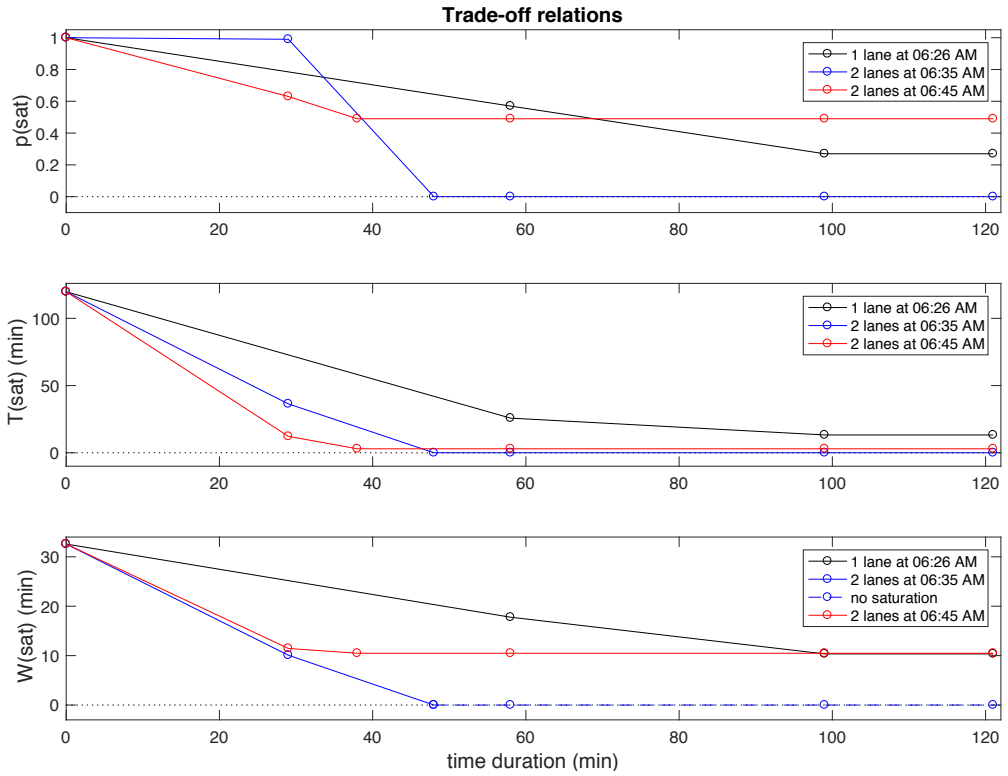


Figure 6.11: Trade-off relations for opening one extra lane at 06:26 AM, and two extra lanes at 06:35 and 06:45 AM. The trade-off relations are presented for three performance indicators: probability of saturation $p(sat)$, the average time duration of saturation $\overline{T}(sat)$, and the average waiting time during the saturation phase $W(sat)$. The performance indicators are plotted against the time of duration for opening the extra security lane(s).

- Opening one extra security lane does not provide enough extra resources to overcome the probability of saturation. This conclusion is drawn based on the fact that it is observed when opening the extra security lane at the beginning of the pre-saturation phase, and keeping it opened until after the moment in time maximum saturation is reached by the initial prediction, still the probability of saturation exists. Nevertheless, opening one security lane for a duration of approximately 100 minutes reduces the probability of saturation to 0.27. Additionally, the average waiting time is decreased to 10.35 minutes, which is just above the threshold value of 10 minutes. Finally, the average number of missed flight is reduced to approximately 1 passenger.

- Additionally, based on the results obtained for opening one extra lane, it is shown that it is important to act pro-actively, when the resources you are able to add are not matching the posed demand. By mobilizing the extra resources already in the pre-saturation phase, the severity of saturation is decreased.

- The option to open two extra lanes provides enough resources to avoid a saturated state, as the average processing rate of two extra lanes is higher than the average saturation rate of the system. However, to obtain a probability of saturation equal to zero, the opening of two extra security lanes has to be executed pro-actively, e.g. before the system reaches a saturated state, and the lanes are opened for 48 minutes.

- It is found that timing of a specific action can have significant effects on the impact on saturation risk and severity. Therefore, it can be concluded that in practice, mobilization time of resources should be taken into account while planning for an adaptive measure.

- The applied methodology allows to obtain trade-off relations, which can be fed to the anticipatory model for decision-making and could support industry in decision-making.

7

Evaluation with RTHA

When starting this research the collaboration with RTHA in the context of this research, was just set in motion. During the final phase of this research the possibility was presented to visit RTHA and discuss the topic of resilience and the working of their security operations with duty security managers of the airport, and team leaders of Trigion. Trigion is the company that is hired by the airport to perform security operations. This chapter describes how the case study conducted, fits within the security operations of RTHA in practise. Additionally, it is discussed how the current state of resilience might be improved at RTHA.

Security Operations Organizational Structure Operational management of security on RTHA is the responsibility of two parties. Firstly, employed by the airport, there exist a team of duty managers security (DMS). They are responsible to ensure that the airport is compliant with the national and international regulations for airport security. The DMS maintains high-level management over day-to-day operations on the shop floor of the airport terminal. Secondly, the execution of the security tasks is done by security operators. The airport hires employees from Trigion to execute the security operations tasks. This includes staff for the security check-point, staff responsible for the security of the airport premises, and operators that conduct the luggage checks for checked-in luggage. Additionally, the group of security operators is managed by team leaders (one or two per shift), also hired from Trigion. The DMS and team leaders communicate with each other to make decisions about exceptional situations at the security check-point, to inform each other about suspicious situations, or in the case something goes wrong. Essentially, the DMS have a final saying with regard to the organization, and how to go about in specific situations.

Anticipation at the Security Check-point The case study scenario applied for this research assumes a disrupted passenger demand at the security check-point due to delayed departure flights. In the case of delayed flights, it is experienced by RTHA that the pattern of passenger arrival at the security check-point, potentially differs from what is expected based on the original flight schedule. This can be the case due to two main reasons: (1) passengers adjust their arrival time at the airport, due to the delayed flight, (2) if the delay is announced after passengers already went through security, it happens that these passengers come back to the land-side of the terminal to wait or move to a different location, until the flight departs. This is mostly observed for business travelers who move to a location where they can work. However, as well for normal travelers, the land-side of the airport has more to offer compared to the air-side. In case of longer waiting periods, they can decide to wait on the land-side. The shift from the air-side to the land-side, means that these passengers have to go through security again, later in time.

The ability to anticipate crowdedness at the security check-point is limited to ad hoc measures and limited by available resources. Essentially, a planning of the expected passenger demand over the day is made before hand, and based on this planning security operators are scheduled in shifts of eighth hours. For RTHA, the morning rush-hours (06:00-08:00) present the most vulnerable part of the day, as the number of passengers

that have to processed is relatively high. Additionally, often business passengers tend to arrive at the airport late, creating last-minute accumulation at the security check-point. If problems are expected at the security check-point due to a high passenger demand, the adaptive capacity of RTHA can be described by the following available options to adapt:

- Security operators assigned different tasks at the airport can be mobilized to help at the security check-point.
- Security operators can be asked to stay longer than the planned shift, or start earlier with their shift, to provide extra man power at the security check-point.
- Security operators on a break, can be called back to help at the security check-point.
- At RTHA all security operators are able to execute all possible tasks, this makes it easier to have operators fill in wherever needed.

The DMS is responsible to initiate one of these adaptions, if it appears necessary. This decision is made based on experience. A trade-off between monetary cost and profit, for these options is not too relevant for the DMS. This is because for most options no extra man hours are booked, as one makes use of operators already scheduled and present. Only, when asking an operator to stay longer or arrive earlier, extra monetary costs are associated with this. However, based on the information obtained from the DMS, if the DMS decides that these extra resources are required, no further discussion with higher management is necessary for this decision. A more relevant cost trade-off that is encountered for the DMS when deciding on whether to shift security operators from another task to the security check-point, is the effect of leaving the other other task. For instance, when pulling security operators responsible for the security of the premises, you take a certain risk by decreasing the number of operators doing this task, by shifting some of them to a different task. In this sense, the duration of time extra operators are required at the security check-point is an use full indicator to represent cost. The smaller the time duration, the smaller the time duration operators are called away from their original task.

The ability to monitor passenger demand, is limited to real-time video observation by the security monitoring office at the airport (the VCC), and observations from the security operators and team leaders. The DMS is informed by both parties in case passenger demand becomes too high. To determine passenger demand becomes too large, operators and team leaders observe the length of the queue, then based on experience they make an estimation.

Essentially, resilience of the airport terminal is a much broader topic than the security check-point operations on its own. Especially, as the airport terminal presents a complex system where its operations are dependent on each other and adaptions at one point in the system will have its effects at other points in the system, as is the case for instance when shifting tasks for security operators. An other recent example of how disruptions can influence a significant part of the airport operations, is when flights on their way to Schiphol Airport had to diverge their routes to RTHA due to snow (December 2017). This meant for RTHA that at the air-side adjustments had to be made to make room for the extra aircraft landing there. Improvisation from Avia was required to make sure the passenger and luggage handling was organized. Additionally, these passengers were transferred from the aircraft to buses on the parking lot of RTHA to be transferred to Schiphol. These operations were guided by DMS and security operators, which meant that for the actual security operations less operators where available. Generally, for these type of situations procedures and plans are in place, however often procedures are not a perfect fit to the presented situation, and there remains the need to be adaptive.

Conclusions on Resilience at RTHA Based on the information obtained with regard to the security check-point operations, it can be concluded that the system contains some adaptive capacity and flexibility to adapt in the face of challenging demand. However, this capacity is limited by the available security operators, who can be taken away from their original task, and partly on the willingness of operators to work more hours, when asked. To this end, it might be wise to take resilience into account for long-term planning of resources

at the security check-point (and for other security related tasks). If with the long term-planning is anticipated that for specific periods of time more security operators should be available at the airport, capacity is added to the system to adapt in the face of (un)expected disruptive events.

Additionally, the ability to monitor the system's performance is relatively out-dated, which makes it harder to be proactive in making adaptions. For instance, the monitoring system lacks the ability to quantitatively measure arrival patterns of passengers and processing rates at the security check-point. By gathering this information, essentially enough data will become available to be able to apply stronger prediction techniques, for instance techniques from machine learning. This may improve the ability to anticipate in face of challenges.

III

Conclusions

8

Conclusions and Recommendations

To study resilience as concept to deal with (un)expected disruptive events in complex socio-technical systems has gained significant popularity in the past decade. However, despite the attempts to develop definitions and metrics to describe resilience, no formal definition for resilience for socio-technical systems is established yet. As presented in Chapter 2, the current state of research presents a conceptual foundation for resilience. The conceptual ideas aim to answer the question: What makes a system adaptive in the face of (un)expected events? It is recognized that adaptive behavior emerges through local mechanisms, operating at different aggregation levels of the system. These mechanisms are referred to as resilience mechanisms. There is demand for research that aims to formalize these resilience mechanisms and operationalizes them in real-world socio-technical systems. In response to the posed demand, the mechanism of anticipation, in the context of complex socio-technical systems, is selected as focus for this research. The first goal of this research is to develop a formal model for anticipation, which allows reasoning about system behaviour using mathematical logic while both quantitative and qualitative relations are represented. The second goal is to study the effects of operationalizing anticipation in a real-world socio-technical system through a simulation study. To this end, the airport security check-point system is selected for a case study.

The objective of this research is summarized as:

> To contribute to the relatively unexplored research field of resilience in complex socio-technical systems by **developing a formalization of anticipation** and **operationalize this resilience mechanism** in a simulation study of airport security operations, in the face of unplanned and challenging passenger demand, **by applying an agent-based modeling approach**.

In this chapter conclusions are drawn by answering the research questions posed in Chapter 3. Additionally, recommendations for further research are discussed.

8.1. Conclusions

Formalization of anticipation To develop a formal representation of an anticipation mechanism, first a conceptual model is developed inspired by neuroscience following an agent-based modeling framework. An agent-based modeling framework is chosen as it provides the ability to define both quantitative and qualitative relations in the model. The anticipatory system is modeled as an autonomous agent interacting with its environment.

To be able to reason about the anticipation mechanism's behavior in a complex system, its dynamic properties should be formalized. To formalize the conceptual model for the anticipatory agent and its interactions

with the environment, requires the ability to formalize both quantitative and qualitative aspects. Additionally, the syntax should be a high level syntax which allows to add expressivity, for instance to be able to distinguish between model relations that exists as input, internal or output relations. Further, as the anticipatory agent operates in the context of a complex socio-technical system, relevant parallel processes of sub-systems should be expressed as well. This asks for the ability to define sub-systems and notion of a time dimension should be included in the syntax. Timed transition systems (TTS) is selected as formal modeling language as it satisfies the posed desiderata. TTS provides a high-level modeling language based on well-defined mathematical semantics which is already applied for a broad range of system specifications.

To capture the dynamics of anticipatory behaviour, both the environment and anticipatory agent are represented as a TTS operating in parallel. A TTS representation of the environment provides external state information to the anticipatory agent. Based on the awareness of the environment, the anticipatory agent feeds its prediction model. Additionally, it is essential to capture the environment state transitions due to adaptive adjustments made by the anticipatory agent operating in it. For the anticipatory agent, the TTS representation describes the evolution of input, internal, and output states. Input states are defined by interactions between the agent and the environment based on observations. Internal states describe the internal processes of the agent, as defined in the conceptual model. The output states are defined by interactions between the agent and environment based on actions.

The obtained formalization permits smooth integration with an agent-based modeling framework, as in general agent-based models rely on a dynamic system state representation as well. Additionally, it is expected that the formalization language proposed, could be applied for representation of other resilience mechanisms, such as coordination or learning mechanisms, as well.

Application of agent-based modeling and simulation to study the effects of anticipation as a resilience mechanism, in airport security operations For this research a comprehensive methodology is developed, presented in Chapter 5, that describes how to apply agent-based modeling and simulation to study the effects of anticipation in airport security operations. As part of this research thesis, time is dedicated to the development of a baseline agent-based airport terminal operations model (AATOM). This model is developed in collaboration with PhD student S.A.M. Janssen and MSc student A. Knol. The object of this model is to provide a tool to perform experiments in an airport terminal system by performing computational simulations. The advantage of such a simulation tool, is that it becomes easier to conduct all sorts of research with regard to airport terminal operations, which is normally hard to conduct in real-life, as the system is constantly in operation and often safety and security constraints apply.

In order to study anticipation for security check-point operations, a case study is proposed of a scenario where the flight schedule is disrupted with delayed flights. This disruption leads to a disrupted passenger arrival pattern, resulting in saturation of the security queue, long waiting times and passengers missing their flights. To perform computational simulations for the case study scenario, the baseline model of the AATOM is adjusted and extended. Based on the conceptual and formal model for anticipation developed in the first part of this research, an anticipation model is implemented that is able to generate predictions of action options to avoid system saturation. Predictions are generated based on forward simulation of the agent-based airport terminal security model. Based on the initial prediction, action options are generated following an algorithm that aims to minimize the risk of saturation. Valuation of the action options is based on analysis of the prediction results. To this end, metrics are proposed following the concept of risk of saturation. Three performance indicators are introduced to indicate the risk of saturation, namely: probability of saturation $p(sat)$, average time of saturation $\overline{T}_{sat}$, and average waiting time $\overline{W}_{sat}$ for passengers while the system is in a saturated phase. The first performance indicator provides a probability of the system reaching a saturated state, the latter two performance indicators represent the severity of saturation. Additionally, as a global performance indicator, the number of passengers that missed their flight $\overline{MF}_{sat}$, is measured. Finally, based on the valuation of action options, it is concluded that trade-off relations exist between saturation risk and resource costs. These trade-off relations can be used as input for the anticipatory agent to decide on what action option to choose, depending on its preference.

The development of the AATOM has resulted in a well defined and extensive model for terminal operations. As for every model, validation of the model is a critical element to deliver valuable results. For agent-based

modeling and simulation, validation can present an extensive task as the model is founded on many low-level input parameters, and aims to represent all relevant details of the real system. For this research, input parameters and model calibration is based on empirical data from RTHA. However, as the collaboration with RTHA is in an early stage, the amount of data available is still limited. Therefor, it can be concluded that the agent-based model developed and applied provides strong first steps in the right direction. Based on a larger amount of empirical data, the model could be further calibrated and validated, resulting in a powerful tool for research on resilience in airport terminals.

Contribution to industry The results of the case study show how anticipation can have a positive effect on the security check-point performance, while facing a disrupted passenger arrival pattern. Additionally, based on the action options considered, it is observed that the timing of an adaptive action can have a significant influence on how well the action works. It can be concluded that a pro-active action, provides better results than a reactive action. To this end, it is critical to be aware of the mobilization time of resources required to execute an action. Furthermore, in practise the implementation of extra resources results in trade-off relations between resource costs and risk of saturation. For this case study, resource costs are expressed as the number of extra lanes required and the duration that these lanes are opened. For an airport, insight in such trade-off relations could support decision-making.

Based on the conversations conducted at RTHA, as presented in Chapter 7, insight is obtained in how in practise is dealt with a challenging demand at the security check-point. It is concluded that the system contains some adaptive capacity and flexibility to adapt in the face of challenging demand. However, this capacity is limited by the available security operators, who can be taken away from their original task, and partly on the willingness of operators to work more hours, when asked. To this end, it might be wise to take resilience into account for long-term planning of resources at the security check-point (and for other security related tasks). If with the long-term planning is anticipated that for specific periods of time more security operators should be available at the airport, capacity is added to the system to adapt. Additionally, the ability to monitor the system's performance is relatively out-dated, which makes it harder to be pro-active in making adaptions. RTHA is currently entering an extensive rebuilding phase of the airport terminal. With this, they aim to implement innovative techniques that make it possible to better monitor the system real-time, measure its performance and use this data to improve procedures and to become more resilient. In this light, the methodology presented in this research could contribute to further research of implementation of resilience mechanisms, or to analyze the resilient performance of the system in its current state for specific disruptive scenario's.

Contribution to science The current state of research with regard to resilience, puts forward the demand for more research that aims to formalize the concepts of resilience and operationalizes these concepts in real-world socio-technical systems, while applying suitable complex system modeling techniques. In reaction to this demand, this research started from the conceptual foundation put forward by Woods and Hollnagel, and is one of the first that aims to operationalize a resilience mechanisms and to develop a methodology, based on quantitative agent-based modeling, to conduct computational simulations, to study the effects of this resilience mechanism in a real-world socio-technical system. The approach applied for the formalization of anticipation could be generalized and applied to develop models for other resilience mechanisms. Additionally, the methodology developed could inspire others, on how to use the agent-based modeling paradigm to study resilience in complex socio-technical systems.

8.2. Recommendations for Further Research

Extending TTS syntax with non-determinism The applied TTS syntax for formalization is deterministic. This means that it is assumed that for each labeled transition it is certain what the next state will be. However, generally for socio-technical systems several system behavior is possible. For instance, if the anticipatory agent would be extended to obtain more human like behavior, such as including a model for decision-making influenced by emotions, this may ask for the possibility to model multiple optional state transitions, leaving from one state. This results in a non-deterministic system. To include non-determinism in the timed

transition system syntax, there are two possibilities: (1) to define non-determinism choices, and (2) to include probabilistic choices [55].

Non-determinism choices are often specified as a TTS including states having several transitions leaving from the same state. Such a representation can be included if information is missing on what transitions will take place in a system. For instance, to avoid the need to stipulate how the environment will behave, non-determinism choices could be added.

The second option provides the possibility to extend the TTS syntax to obtain Probabilistic Transition System syntax (PTS). The difference between TTS and PTS is that the target of a state transition is not a single state but a probabilistic choice over several next states. For instance, to model a coin flip one can define the initial state s_0 the transition label $\alpha = flip$ and two potential next states $s_1 = head$ and $s_2 = tail$. The probability corresponding to the state transition $s_0 \xrightarrow{flip} s_1$ is $\frac{1}{2}$, and the probability corresponding to the state transition $s_0 \xrightarrow{flip} s_2$ is $\frac{1}{2}$. Formally, such a transition relation is formulated as: $s \xrightarrow{flip} \{head \rightarrowtail \frac{1}{2}, tail \rightarrowtail \frac{1}{2}\}$. A set of probability distributions should be defined and an assignment function that assigns probabilities to the possible state transitions. If for the probabilistic transition system holds that the probability to make a transition between state s and s' only depends on these two states, and not on previous state transitions, it is possible to obtain discrete time Markov chains. For this case, if the set of states is a finite set, one probability transition matrix can be defined that includes the probabilistic data on state transitions.

Implementation of optimization techniques for action options generation A relatively simplistic algorithm is developed and implemented for the anticipatory agent, to seek for smart action options that aim to minimize the risk of saturation, while at the same time resource costs are minimized. Action options are define by three variables that have to be determined. However, it was chosen to apply the algorithm for each possible number of lane configurations (which is constrained by the number of lanes available), this way only two variables were left to be determined for an action option. Additionally, the time range wherein a solution could be found is limited to the pre-saturation phase. Hence, the search space defined was relatively small. To operationalize the anticipation mechanisms for more complex problems, where the search space for potential action options becomes larger, it is recommended to implement more comprehensive optimization techniques. For instance, it could be further investigated how to integrate techniques from meta-heuristics in the model of the anticipatory agent, to seek for smart solutions.

Extending anticipatory agent model with cognitive aspects The anticipatory model developed is inspired by theoretical principles from neuroscience. It is chosen to implement an internal model based on the simulative theory of Hesslow to obtain a representation of human-like anticipative behavior. As this model already includes important internal human cognitive aspects, it is relatively easy to extend the model with other cognitive aspects to obtain an even more comprehensive representation of human adaptive behavior. For instance, one can imagine that the ability to anticipate may be triggered in stressful situations where decisions have to be made fast. To represent human anticipative behavior in a more realistic sense, the notion of emotions could be included in the internal model. Similar research is conducted where the notion of emotions is integrated with internal agent models developed based on principles from neurosciences. This research aims to study the influence of these emotions on decision making in an evacuation scenario [12]. In the same sense, the internal anticipation model could be extend by including adaption mechanisms that influence the internal process based on emotions, or for instance the willingness to take risks.

Improvement of the AATOM As described, the AATOM as simulation tool provides high potential to conduct all kinds of research in an airport terminal environment. For this research, the AATOM was initialized and calibrated based on data available from RTHA. However, the data available from RTHA is limited. Fortunately, RTHA has proposed to provide the possibility of more extensive data gathering at the airport. To this end, they are willing to implement new measuring techniques that monitor the passengers arrival rates, number of passengers queuing, processing times, etc. With much more data available, there is the opportunity to develop a strongly validated simulation tool applicable for research with regard to RTHA's operations. Additionally, it is highly recommended to perform research on how this data could be generalized, as to be

able to develop a validated airport terminal simulation tool that can be applied in a more general sense for airport terminal research. For instance, one of the challenges with modeling airport terminal operations, is to include detailed and realistic passenger behavior. Especially when one would want to optimize efficiency of the terminal processes, it is important to obtain insight in the timing of passengers at these processes. However, generally how passengers move through an airport can depend on many factors, such as: the travel type (business/ leisure), age, experience, cultural background, travelling in a group or alone, and personal goals such as, shopping, eating, toilet, etc. By seeking for correlations in data on passenger behavior, realistic passenger profiles may be developed, which would provide an important step for the improvement of the AATOM.

Continue to research resilience mechanisms As stated at the beginning of this research, there is demand for research that focuses on the mechanisms that enable resilience in a socio-technical system. This research aimed to contribute to this demand. However, much more research is required to develop a good understanding of what makes a system resilient and how to improve resilience. A distinction can be made between two research directions. The first aims to understand what makes a socio-technical system resilient in its current state. The second, focuses on new techniques and mechanisms that could be implemented and might improve resilience in a system. It is recommended, in order to study resilience mechanisms already present in a system, to identify the resilience mechanisms at different aggregation levels of the system and develop models to represent these mechanisms. Essentially, if a representation is developed of the system and the interdependencies between resilience mechanism, the system behavior as a whole can be analyzed for different disruptive scenario's, and the influence of local mechanisms on the global emergence of the system can be identified. For the second research direction, the improvement of resilience, it is recommended to explore how novel techniques from the area of artificial intelligence could be applied to develop (automated) adaptive mechanisms for the system.

Bibliography

[1] Airports Council International, *ACI Europe,* Air Traffic Report (2016).

[2] R. Francis and B. Bekera, *A metric and frameworks for resilience analysis of engineered and infrastructure systems,* Reliability Engineering and System Safety **121**, 90 (2014).

[3] C. S. Holling, *Resilience and Stability of Ecological Systems,* Annual Review of Ecology and Systematics **4**, 1 (1973), arXiv:arXiv:1011.1669v3 .

[4] D. Helbing and S. Balietti, *Social Self-Organization: Agent-based Simulations and Experiments to Study Emergent Social Behaviour* (Springer, Zurich, Switzerland, 2012).

[5] H. Blom and S. Bouarfa, *Complexity science in air traffic management, Chapter 5: Resilience* (Routledge, New York, USA, 2016).

[6] E. Bonabeau, *Agent-based modeling: Methods and techniques for simulating human systems,* Proceedings of the National Academy of Sciences **99**, 7280 (2002), http://www.pnas.org/content/99/suppl_3/7280.full.pdf .

[7] K. H. Van Dam, I. Nikolic, and Z. Lukszo, *Agent-based modelling of socio-technical systems. Chapter: Theory*, Vol. 9 (Springer Science & Business Media, 2013).

[8] M. Ouyang, *Review on modeling and simulation of interdependent critical infrastructure systems,* Reliability Engineering & System Safety **121**, 43 (2014).

[9] S. H. Stroeve and M. H. Everdij, *Agent-based modelling and mental simulation for resilience engineering in air transport,* Safety Science **93**, 29 (2017).

[10] N. Bulling, *A survey of multi-agent decision making,* KI - Künstliche Intelligenz **28**, 147 (2014).

[11] R. Sun, *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation* (Cambridge University Press, 2006).

[12] A. Sharpanskykh and J. Treur, *An adaptive affective social decision making model,* in *Biologically Inspired Cognitive Architectures 2012: Proceedings of the Third Annual Meeting of the BICA Society*, edited by A. Chella, R. Pirrone, R. Sorbello, and K. R. Jóhannsdóttir (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013) pp. 299–308.

[13] A. Sharpanskykh and K. Zia, *Understanding the role of emotions in group dynamics in emergency situations,* in *Transactions on Computational Collective Intelligence XV*, edited by N. T. Nguyen, R. Kowalczyk, J. M. Corchado, and J. Bajo (Springer Berlin Heidelberg, Berlin, Heidelberg, 2014) pp. 28–48.

[14] L. Spalazzi, *Reasoning about rational agents, intelligent robots and autonomous agents series,* Minds Mach. **13**, 429 (2003).

[15] E. Norling, *Folk psychology for human modelling: Extending the bdi paradigm,* in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '04 (IEEE Computer Society, Washington, DC, USA, 2004) pp. 202–209.

[16] D. Helbing and P. Molnar, *Social force model for pedestrian dynamics,* Physical review E **51**, 4282 (1995).

[17] D. Helbing and P. Molnar, *Self-organization phenomena in pedestrian crowds,* arXiv preprint cond-mat/9806152 (1998).

[18] I. E. Manataki and K. G. Zografos, *Assessing airport terminal performance using a system dynamics model,* Journal of Air Transport Management **16**, 86 (2010).

[19] J. Bergström, R. van Winsen, and E. Henriqson, *On the rationale of resilience in the domain of safety: A literature review,* Reliability Engineering & System Safety **141**, 131 (2015), special Issue on Resilience Engineering.

[20] E. Hollnagel, D. Woods, and N. Leveson, *Resilience Engineering: Concepts and Precepts* (Ashgate, Hampshire, England, 2006).

[21] E. Hollnagel, *Safety-I and safety–II: the past and future of safety management* (Ashgate Publishing, Ltd., 2014).

[22] E. Hollnagel, *Rag-the resilience analysis grid,* Resilience engineering in practice: A guidebook. Farnham, UK: Ashgate (2011).

[23] A. W. Righi, T. A. Saurin, and P. Wachs, *A systematic literature review of resilience engineering: Research areas and a research agenda proposal,* Reliability Engineering & System Safety **141**, 142 (2015), special Issue on Resilience Engineering.

[24] S. Hosseini, K. Barker, and J. E. Ramirez-Marquez, *A review of definitions and measures of system resilience,* Reliability Engineering & System Safety **145**, 47 (2016).

[25] D. Henry and J. E. Ramirez-Marquez, *Generic metrics and quantitative approaches for system resilience as a function of time,* Reliability Engineering & System Safety **99**, 114 (2012).

[26] D. D. Woods, *Four concepts for resilience and the implications for the future of resilience engineering,* Reliability Engineering & System Safety **141**, 5 (2015), special Issue on Resilience Engineering.

[27] J. Pariès, J. Wreathall, P. Woods, and P. Hollnagel, *Resilience Engineering in Practice: A Guidebook, Chapter 10: How Adaptive Systems Fail*, Ashgate Studies in Resilience Engineering (Ashgate Publishing Limited, 2012).

[28] D. D. Woods and J. Wreathall, *Stress-strain plots as a basis for assessing system resilience,* Resilience engineering perspectives **1**, 145 (2008).

[29] D. D. Woods, Y. J. Chan, and J. Wreathall, *The stress–strain model of resilience operationalizes the four cornerstones of resilience engineering,* in *5th Resilience Engineering Symposium* (2014) pp. 17–22.

[30] S. C. Bankes, *Robustness, adaptivity, and resiliency analysis.* in *AAAI fall symposium: complex adaptive systems*, Vol. 10 (2010).

[31] D. W. e. a. I. Herrera, J. Schraagen, *5th REA Symposium: managing trade-offs,* (Resilience Engineering Association, 2013).

[32] D. D. Woods, *The theory of graceful extensibility: Basic rules that govern adaptive systems,* (2018), manuscript submitted for publication.

[33] D. D. Woods, *Resource Guide on Resilience IRGC (2016), Resilience as Graceful Extensibility to Overcome Brittleness*, 1st ed. (Lausanne: EPFL International Risk Governance Center, 2016).

[34] G. Klein, D. Snowden, and C. L. Pin, *Anticipatory thinking,* KL Mosier, & UM Fischer, Informed by Knowledge , 235 (2011).

[35] G. Pezzulo, M. V. Butz, C. Castelfranchi, and R. Falcone, *The challenge of anticipation: A unifying framework for the analysis and design of artificial cognitive systems*, Vol. 5225 (Springer, 2008).

[36] E. MINCH, *A review of: "anticipatory systems", by robert rosen, pergamon press, oxford, 1985, x + 436pp.* International Journal of General Systems **12**, 405 (1986), https://doi.org/10.1080/03081078608934946 .

[37] G. Hesslow, *Conscious thought as simulation of behaviour and perception.* Trends in Cognitive Sciences **6**, 242 (2002).

[38] J. Broekens, W. A. Kosters, and F. J. Verbeek, *Affect, anticipation, and adaptation: Affect-controlled selection of anticipatory simulation in artificial adaptive agents,* Adaptive behavior **15**, 397 (2007).

[39] M. V. Butz, O. Sigaud, and P. Gérard, *Internal models and anticipations in adaptive learning systems,* in *Anticipatory Behavior in Adaptive Learning Systems: Foundations, Theories, and Systems*, edited by M. V. Butz, O. Sigaud, and P. Gérard (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003) pp. 86–109.

[40] M. V. Butz and G. Baldassarre, *Anticipatory behavior in adaptive learning systems* (Springer, 2007).

[41] NOS webpage, *Klm boos: drukte schiphol heeft ons miljoenen gekost,* (Accessed: 09-05-2017).

[42] B. M. Blumberg and T. A. Galyean, *Multi-level direction of autonomous creatures for real-time virtual environments,* in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (ACM, 1995) pp. 47–54.

[43] S. P. Hoogendoorn and P. H. Bovy, *Pedestrian route-choice and activity scheduling theory and models,* Transportation Research Part B: Methodological **38**, 169 (2004).

[44] C. W. Reynolds, *Steering behaviors for autonomous characters,* in *Game developers conference*, Vol. 1999 (1999) pp. 763–782.

[45] T. Bosse, C. M. Jonker, L. van der Meij, A. Sharpanskykh, and J. Treur, *A temporal trace language for the formal analysis of dynamic properties*, Tech. Rep. (Technical Report, Vrije Universiteit Amsterdam, Department of Artificial Intelligence, 2006).

[46] T. Bosse, A. Sharpanskykh, and J. Treur, *A formal cognitive model of an ambient agent*, Tech. Rep. (Technical Report TR-02-2010, Vrije Universiteit Amsterdam, 2010).

[47] R. Gorrieri and C. Versari, *Introduction to Concurrency Theory: Transition Systems and CCS*, 1st ed. (Springer Publishing Company, Incorporated, 2015).

[48] J. Schmaltz, *Lecture reader of the course: 2IX20 software specification,* (2017).

[49] T. Bosse, C. M. Jonker, L. Van der Meij, A. Sharpanskykh, and J. Treur, *Specification and verification of dynamics in agent models,* International Journal of Cooperative Information Systems **18**, 167 (2009).

[50] S. Janssen, A. R. Blok, and A. Knol, *AATOM - An agent-based airport terminal operations model,* A semi-formal model description (2017).

[51] D. Helbing, I. Farkas, and T. Vicsek, *Simulating dynamical features of escape panic,* Nature **407**, 487 (2000).

[52] A. Dilweg, *Prognose, passenger predictions by a. dilweg rtha,* (2017).

[53] S. A. M. Janssen, *Section 1.1. security planning,* Rotterdam The Hague Airport Security Operations - Working Document (2017).

[54] D. Stewart Head of Airport Development IATA UK, *The new airport development reference manual (adrm),* Passenger Terminal Conference 2014 (2014).

[55] M. Stoelinga, *An introduction to probabilistic automata,* Bulletin of the EATCS **78**, 2 (2002).

A

TTS Syntax

In this chapter the syntax of timed transition systems is introduced. First an overview of the syntax applicable for labeled transition systems (LTS) is provided. This syntax is extended by the addition of clocks which allow to define state transitions over time, resulting in syntax for timed transition systems (TTS).

Labeled Transition Systems (LTS) Labeled transition systems are characterized by *action labels* that describe the action performed corresponding to a state transition. In general a labeled transition system is defined as a 4-tuple $(S, A, \rightarrow, S_0)$, where [47]:

- S is a set of *states* (s), the set of states can be infinite large.
- A is a set of *action labels* (α), the set of action labels can be an infinite set.
- $\rightarrow \subseteq S \times A \times S$ is a *transition relation*. A transition relation (s, α, s') is written as $s \xrightarrow{\alpha} s'$.
- $S_0 \subseteq S$ is the set of *initial states*.

The basic form of the labeled transition system can be extended by defining *atomic propositions* and *labeling functions*. Such a labeled transition system is written as a 6-tuple $(S, A, \rightarrow, S_0, AP, L)$, where [48]:

- AP is a set of *atomic propositions*.
- $L : S \rightarrow 2^{AP}$ is a *labeling function*

Atomic propositions are used to represent state information, e.g. describe the system state. A labeling function $L(s)$ returns all atomic propositions that hold for a state $s \in S$. If no atomic proposition holds in state s than $L(s) = \emptyset$. Here a labeling function is simplified to the form where it only assigns a Boolean value to the atomic proposition. However, this representation can be extended to allow for more complex relations.

Additional to observable actions, it is possible to include non-observable actions in the system representation. These actions are defined as *internal actions* and are denoted by the label: τ with $\tau \notin L$. Following the definition of labeled transition system's, an internal action is described as an action which is not observable by the system's environment, e.g. is executed internally. No distinction is made between different internal actions, consequently each internal action is denoted by the same label τ and a finite sequence of internal actions is represented by τ^*.

A possibly infinite sequence of states and actions is defined as an *execution trace* or just *trace*. A trace is of the form:

$$s_0\alpha_0 s_1\alpha_1 s_2\alpha_2 s_3\alpha_3 ....s_n\alpha_n \quad \text{such that} \quad s_i \xrightarrow{\alpha_i} s_{i+1} \quad \forall i \geq 0$$

Furthermore, a *path* (π) is defined as the projection of the trace to states. A *run* is defined as a path starting at an initial state. A path is of the form:

$$\pi = s_0 s_1 s_2 s_3 ....s_n$$

Timed Transition Systems (TTS) Labeled transition system provide an abstract way to describe system functioning based on state transitions and corresponding action labels. Additionally, it is possible to define atomic propositions and labeling functions to describe a system state. However, for a LTS representation, the dimension of time is abstracted away. For many systems, functioning depends on real-time considerations. To this end, labeled transitions systems can be extended to timed transition systems. Essentially, the aforementioned definitions for labeled transition systems remain valid and are extended by inclusion of a time entity.

A timed transition system is obtained through introduction of the notion of a *clock*. A clock c is a real-valued variable representing time as a dense set, e.g. as a continuous entity ($c \in \mathbb{R}^+$). For a system multiple clocks can be defined, whereas all clocks increase at the same rate. Two clock operations are defined: (1) to read the value of the clock, and (2) to reset a clock to the value 0. To this end, each clock represents the elapsed time since the last reset of the clock. Two types of clock constraints exist: a *state invariant* and a *guard*. A state invariant determines for how long a state holds. A guard is related to the availability of a transition. As long as the a guard holds, a transition is available. To develop the set of clock constraints the following two concepts are needed [48]:

- A *clock valuation* which provides a value for each clock. A clock valuation v corresponding to a clock variable x from the set of clock variables C is defined as a function $\mathsf{v} : C \rightarrow \mathbb{R}^+$, which assigns to each clock variable $x \in C$ its current value $\mathsf{v}(x)$. Then, V is defined as the set of all possible valuations over a set of clocks C.
- A *satisfaction relation* which determines for which valuation a given clock constraint is true.

The behaviour of a timed transition system is described as a transition system where the set of action labels A, is extended with possible *delays*. This is defined as a 6-tuple $(\mathsf{S}, \mathsf{A}, \Rightarrow, \mathsf{S}_0, \mathsf{AP}, \mathsf{L})$, where:

- $\mathsf{S} = S \times V$ is a set of states, represented by a possibly infinite set of pairs $S \times V$. With S representing states as defined for labeled transition systems, and V representing the set of valuations for the set of clocks C.
- $\mathsf{A} = A \cup \mathbb{R}^+$, which is a set of *action labels* and the set of real-values used to draw delay values. The set of action labels and real-values can be infinite large.
- $\mathsf{S}_0 = S_0 \times \{V_0\}$, which is the set of initial states represented by the pairs $S_0 \times \{V_0\}$.
- $\mathsf{AP} = AP \cup \mathsf{B}(\mathsf{C})$, which is a set containing the set of *atomic propositions AP* and the set of *clock constraints* $\mathsf{B}(\mathsf{C})$ including transition guards and state invariants.
- $\mathsf{L}(s, v) = L(s) \cup \{\theta \in \mathsf{B}(\mathsf{C}) | v \models \theta\}$, the *labeling function* L contains the atomic propositions that hold for a state s represented by $L(s)$, and assigns a clock guard $\theta \in \mathsf{B}(\mathsf{C})$ to a state, if the satisfaction relation holds ($v \models \theta$).
- $\Rightarrow \subseteq \mathsf{S} \times (A \cup \mathbb{R}^+) \times \mathsf{B}(\mathsf{C}) \times 2^C \times \mathsf{S}$ is the *transition relation*. The transition relation is defined by the following two rules:
  - A *discrete transition*, e.g. a transition initiated by an *action*, exists: $(s, \mathsf{v}) \xrightarrow{\alpha} (s', \mathsf{v}')$. If:
    - there exists a transition relation: $s \xrightarrow{\alpha,\theta,\lambda} s'$. Here α denotes the action label, θ denotes the clock guard, and $\lambda \subseteq C$ denotes the set of clocks to be reset.

- ◇ $v \models \theta$, the clock guard is true.
- ◇ $v' = v[\lambda \rightarrow 0]$, all clocks $\in \lambda$ are reset to zero.
- ◇ and, $v' \models Inv(s')$, the state invariant of the target state holds.

– A *delay transition* exists: $(s,v) \xrightarrow{d} (s, v+d)$, with d a non-negative real number representing an increase of the clock value. If:

- ◇ $v + d \models Inv(s)$, the amount of delay is as such, that the state invariant maintains to hold.

An *execution trace* ρ is of the form:

$$\rho = s_0 \alpha_0 s_1 \alpha_1 s_2 \alpha_2 s_3 \alpha_3 .... s_n \alpha_n, \quad \text{with} \quad \alpha_i \in A \cup \mathbb{R}^+$$

And the *execution time* of a trace is defined as follows:

$$ExecTime(\rho) = \sum_{i=0}^{\infty} ExecTime(\alpha_i), \quad \text{with} \quad ExecTime(\alpha) = \begin{cases} 0 & \text{if } \alpha \in A \\ d & \text{if } \alpha = d \in \mathbb{R}^+ \end{cases}$$

Note: a transition due to an action, a discrete transition, occurs in zero time. Time only increases over a delay transition.

To model complex systems, existing of multiple sub-systems that evolve over time by state transitions in parallel, *parallel composition* can be applied to compose such complex systems [48]. Each sub-process is represented by its own timed transition system. These timed transition systems communicate with each other via *handshaking actions* $\in H$, which are the shared actions among the timed transition systems. Timed transitions systems synchronize on all actions in H. For actions outside H the timed transition systems evolve independently.

A parallel composition of sub-processes is formally defined as follows: Given two timed transition systems $TTS_1 = (S_1, S_{0,1}, A_1, C_1, \rightarrow_1, Inv_1, AP_1, L_1)$ and $TTS_2 = (S_2, S_{0,2}, A_2, C_2, \rightarrow_2, Inv_2, AP_2, L_2)$. Such that $AP_1 \cap AP_2 = \emptyset$, $C_1 \cap C_2 = \emptyset$, and the set of handshaking actions is defined as $H \subseteq A_1 \cap A_2$. The parallel composition of TTS_1 and TTS_2 is formulated as:

$$TTS_1 ||_H TTS_2 = (S_1 \times S_2, S_{0,1} \times S_{0,2}, A_1 \cup A_2, C_1 \cup C_2, \rightarrow, Inv, AP_1 \cup AP_2, L)$$

Where,

- $L((s_1, s_2)) = L_1(s_1) \cup L_2(s_2)$
- $Inv((s_1, s_2)) = Inv(s_1) \wedge Inv(s_2)$

And the transition relation $\rightarrow$ is defined by the following rules:

- for $\alpha \in H$

$$\frac{s_1 \xrightarrow{\alpha,\theta_1,\lambda_1}_1 s_1' \wedge s_2 \xrightarrow{\alpha,\theta_2,\lambda_2}_2 s_2'}{(s_1, s_2) \xrightarrow{\alpha,\theta_1 \wedge \theta_2, \lambda_1 \wedge \lambda_2} (s_1', s_2')}$$

- for $\alpha \notin H$

$$\frac{s_1 \xrightarrow{\alpha,\theta,\lambda}_1 s_1'}{(s_1, s_2) \xrightarrow{\alpha,\theta,\lambda} (s_1', s_2)} \wedge \frac{s_2 \xrightarrow{\alpha,\theta,\lambda}_2 s_2'}{(s_1, s_2) \xrightarrow{\alpha,\theta,\lambda} (s_1, s_2')}$$

Thus, the state of a complex system existing of several sub-processes is composed from the states of the individual processes.

B

Contribution development AATOM

At the beginning of this research, the development of AATOM had already progressed over about one year as part of the PhD project of S.A.M. Janssen. My contribution to the model development started a few months before the start of this thesis, while I supported the master course 'Agent-based Modelling and Simulation in Air Transport' as a student assistant. During this course students worked with a early version of the AATOM to conduct their assignments. In order to prepare these assignments, I tested the code and worked together with S.A.M. Janssen to develop a function library, as a support for the students to learn how to work with the AATOM.

With the start of this thesis research, a design plan was set-up to develop a theoretically well-defined model of airport terminal operations, followed by the development of a simulation tool which allows to perform all kinds of airport terminal simulation studies. The steps corresponding to this design plan are performed in collaboration with S.A.M. Janssen and MSc student A. Knol, under the supervision of Dr. A. Sharpanskykh. During this thesis research the following steps are executed, as part of this teamwork:

1. Development of a conceptual specification of the airport terminal operations model.
2. Development of an agent-based semi-formal specification of the airport terminal operations model, including validated models to represent human behavior. This model is referred to as the baseline model.
3. Implementation of the baseline model in Java for the development of the simulation tool: the AATOM.
4. Verification of the simulation tool.
5. Documentation: development of users manual.

Here, relatively much time is dedicated to step 1, 2, 4 and 5. The semi-formal model specification obtained as a result of step 1 and 2 is presented in [50]. The implementation of the baseline model in Java is done by S.A.M, Janssen. After the selection of my case study, I extended the baseline model to be able to study the effect of anticipation at the security check-point. To this end, an anticipatory agent model is developed and included in the model as described in Chapter 5.

During the implementation phase of my own model, I contributed to the verification of the simulation tool, step 4. Multiple bugs and errors were encountered and resolved in collaboration with S.A.M. Janssen. Parallel to the implementation process, I contributed to the development of an users manual for the simulator, step 5. This manual aims to provide a clear description to new users of the tool, on how the AATOM simulator is set-up and adjusted. Additionally, a comprehensive list is added with 'frequently asked questions' and the corresponding answers.

C

Algorithm for action options generation

As described in Section 5.5, an action option is defined by three variables:

- i: the number of extra security lanes to open.
- t_{ex}: the moment in time to open the extra lane(s).
- ΔT: the time duration of opening the extra lane(s).

The algorithm seeks for action options with an execution time t_{ex} that lays within the period of time corresponding to the pre-saturation region. The point in time where the pre-saturation phase starts is denoted by t_0 and the point in time where the system reaches the saturation phase is denoted by t_1. Thus $t_0 \leq t_{ex} \leq t_1$. Additionally, the moment in time where the saturation phase ends is denoted by t_2.

The algorithm aims to minimize the probability of saturation. As first step, the algorithm determines for each possible number of extra lanes to open $i = 1 : N-1$, with N the total number of security lanes present at RTHA, if the average processing rate r_{pro} of added resources is larger or smaller than the average *saturation rate* r_{sat}. Here, the saturation rate is estimated as the averaged linear increase of passengers in the queue between t_0 and t_1. Further, the average processing rates as presented in Table 5.3 are assumed. Depending on whether r_{pro} is $<$ or $\geq r_{sat}$ *algorithm a* or *algorithm b* is applied to determine the variables of the action option(s). Step 1, selection of the algorithm is described below.

Step 1: Selection of algorithm

---

input
    total number of security lanes N
    saturation rate r_{sat}
    vector of processing rates $\overline{r_{pro}}$
for $i = 1 : N-1$
    $\Delta_r = r_{sat} - r_{pro}(i)$
        if $\Delta_r > 0$
            add number of lanes i to set M^1
        end
        if $\Delta_r \leq 0$
            add number of lanes i to set M^2
        end
    save $\Delta_r(i) = \Delta_r$
end

for $j = 1 : end(M^1)$
    execute *algorithm a*
end
for $j = end(M^1) + 1 : end(M^2)$
    execute *algorithm b*
end
output
    see output *algorithm a* and *algorithm b*

If $\Delta_r > 0$, this means that when opening i extra security lanes at t_{ex} (with $t_0 \leq t_{ex} \leq t_1$) accumulation will continue with an estimated saturation rate of Δ_r. Now, *algorithm a* determines how far in advance, before the moment of saturation at t_1, the extra number of security lane(s) i should be opened to avoid that the threshold value will be exceeded. The execution time t_{ex} is determined based on Δ_r and the moment in time maximum saturation is reached t_{max}. To this end, *algorithm a* solves the following equation:

$$d(t) + \Delta_r(t_{max} - t) = threshold(t_{max}) \quad \text{for } t \leq t_1 \tag{C.1}$$

Here, $d(t)$ represent the demand, e.g. the number of passengers in the queue at time t. The threshold value that holds at t_{max} is denotes as $threshold(t_{max})$. If the obtained value $t \leq t_0$, t_{ex} is set equal to t_0, otherwise $t_{ex} = t$. *Algorithm a* is described below.

Step 2a: *algorithm a*

---

input
    demand: number of passengers in the queue over time $d(t)$
    time of maximum saturation t_{max}
    threshold value at the time of maximum saturation $threshold(t_{max})$
    saturation rate vector $\overline{\Delta_r}$
for $j = 1 : end(M^1)$
    while $t \leq t_1$
        solve $d(t) + \overline{\Delta_r}(j) \times (t_{max} - t) = threshold(t_{max})$
            if $t \leq t_0$
                $t_{ex} = t_0$
            end
            if $t > t_0$
                $t_{ex} = t$
            end
    save $t_{ex}(j) = t_{ex}$
output
    vector of execution times corresponding to opening j lane(s): $\overline{t_{ex}}$
end

For each action option determined through *algorithm a*, defined by the number of security lanes to open and the corresponding execution time, now the time duration for opening the extra lane(s) is determined. For each action option, three different time duration are defined, calculated by the following equations:

$$\Delta T_1 = t_{max} - t_{ex}$$
$$\Delta T_2 = \frac{t_2 - t_{max}}{2} - t_{ex}$$
$$\Delta T_3 = t_2 - t_{ex}$$

Thus, for instance, if set $M^1 = \{1\}$, this means that for only opening one extra security lane $\Delta_r > 0$. A corresponding execution time t_{ex} is calculated by *algorithm a*. Now the possible action options defined for opening one extra security lane are:

- option 1: opening 1 extra lane, at t_{ex} for a time duration of ΔT_1
- option 2: opening 1 extra lane, at t_{ex} for a time duration of ΔT_2
- option 3: opening 1 extra lane, at t_{ex} for a time duration of ΔT_3

An option is only accepted in the option set, if the resource constraint is satisfied. The resource constraint is formulated as:

$$\begin{aligned} &\text{if} \quad i \leq N - n_{use}(t_{ex}) \\ &\quad \text{action option is accepted} \\ &\text{else} \\ &\quad \text{action option is not accepted} \end{aligned} \tag{C.2}$$

Here, $n_{use}(t_{ex})$ denotes the number of security lanes in use at time of execution.

For the second case where $\Delta_r \leq 0$, this means that when opening i extra security lanes at t_{ex} (with $t_0 \leq t_{ex} \leq t_1$) accumulation is expected to come to a hold, as the added processing rate is larger or equal to the approximated saturation rate. For this scenario, *algorithm b* is applied to determine the time duration the number of i security lanes should be opened to avoid reaching a saturated state. To this end, first three execution times are selected to evaluate. These are:

- $t_{ex}^1 = t_1$: the extra security lane(s) are opened at the moment a saturated state is reached.
- $t_{ex}^2 = \frac{t_1 - t_0}{2}$: the extra security lane(s) are opened half way during the pre-saturation phase.
- $t_{ex}^3 = t_0$: the extra security lane(s) are opened at the beginning of the pre-saturation phase.

For each of these options holds that this is only possible to select the t_{ex} if constraint C.2 is satisfied. If for an option the constraint is not satisfied, the algorithm seeks for the first moment in time with $t > t_{ex}$, for which the constraint is satisfied. Then, t_{ex} is set equal to t. The set of three possible execution times is denoted by $\overline{t_{ex}}$. Corresponding time duration ΔT are calculated by *algorithm b* described below.

Step 2b: *algorithm b*

---

input
demand: number of passengers in the queue over time $d(t)$
time of maximum saturation t_{max}
threshold value at the time of maximum saturation $threshold(t_{max})$
saturation rate r_{sat}
saturation rate vector $\overline{\Delta_r}$
execution time vector $\overline{t_{ex}}$
for $j = end(M^1) + 1 : end(M^2)$
for $k = 1 : 3$
solve $d(\overline{t_{ex}}(k)) + \overline{\Delta_r}(j) \times \Delta T + r_{sat} \times (t_{max} - (\overline{t_{ex}}(k) + \Delta T)) = threshold(t_{max})$
end
save $\Delta T(k) = \Delta T$
end

save for each number of lanes j evaluated vector $\overline{\Delta T}$

output

for each j evaluated and for three possible execution times, three duration values $\in \overline{\Delta T}$ are determined

---

Thus, for instance, if set $M^2 = \{2\}$, this means that for opening two extra security lanes $\Delta_r \leq 0$. Three execution time options are calculated as described above. For each of the execution times *algorithm b* is applied to seek for duration times that should avoid that the system will reach a saturated state. The possible action options defined for opening two extra security lanes are:

- option 1: opening 2 extra lanes, at t_{ex}^1 for a time duration of ΔT_1
- option 2: opening 2 extra lanes, at t_{ex}^2 for a time duration of ΔT_2
- option 3: opening 2 extra lanes, at t_{ex}^3 for a time duration of ΔT_3