Capturing Agents in Security Models

Agent-based Security Risk Management using Causal Discovery

Janssen, Stef

DOI
10.4233/uuid:f9bbff72-b9b4-4694-a188-b2f1451449af

Publication date
2020

Document Version
Final published version

Citation (APA)
Janssen, S. (2020). *Capturing Agents in Security Models: Agent-based Security Risk Management using Causal Discovery*. [Dissertation (TU Delft), Delft University of Technology].
https://doi.org/10.4233/uuid:f9bbff72-b9b4-4694-a188-b2f1451449af

CAPTURING AGENTS IN SECURITY MODELS

AGENT-BASED SECURITY RISK MANAGEMENT USING CAUSAL DISCOVERY

CAPTURING AGENTS IN SECURITY MODELS

AGENT-BASED SECURITY RISK MANAGEMENT USING CAUSAL DISCOVERY

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus, prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates,
to be defended publicly on
Thursday 9 April 2020 at 12:30 o'clock

By

Stef Antoine Maria JANSSEN

Master of Science in Operations Research,
Maastricht University, Maastricht, The Netherlands,
born in Arcen en Velden, The Netherlands.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. K. G. Langendoen,	Delft University of Technology, promotor
Prof. dr. R. Curran,	Delft University of Technology, promotor
Dr. O. A. Sharpans'kykh,	Delft University of Technology, copromotor
Independent members:	
Dr. J. Skorupski,	Warsaw University of Technology, Poland
Dr. A. I. Barros,	TNO
Dr. M. T. J. Spaan,	Delft University of Technology
Prof. dr. ir. G. L. L. M. E. Reniers,	Delft University of Technology
Prof. dr. ir. J. M. Hoekstra,	Delft University of Technology, reserve member

This research was partly funded by the Dutch Ministry of Economic Affairs under the Topsectoren policy for High Tech Systems and Materials.

Keywords: Security Risk Management, Agent-based Modelling, Causal Discovery, Airport Terminal

Printed by: Ipskamp Printing, Enschede

Front & Back: W.J.J.C. van Wijlick

ISBN 000-00-0000-000-0

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

ACKNOWLEDGEMENTS

This thesis is the result of four years of work, which would have been impossible without the support of many people. I arrived in Delft without knowing anybody and now finished my PhD with a large group of people that I want to express my gratitude to.

Working in both the Air Transport & Operations and the Embedded & Networked Systems groups came with some advantages. First and foremost, my supervisory team consisted of one daily supervisor and additionally two promotors instead of one.

Alexei, throughout my PhD you have given me the opportunity to explore the world of academic research with practical guidance and countless opportunities to do new things. You trusted me to develop my own line of research in the agent-based community and gave me the freedom to do it my way. Without your trust and support, I would not have been able to present my work to all kinds of different audiences, supervise different students and interact with various experts in the field. Also on a personal level, I really enjoyed the meetings that we had, that oftentimes were about special holiday destinations, politics or cultures around the world.

Koen, on the first day I met you, I got to know you as a fun, honest and direct person that I immediately enjoyed working with. While my PhD ended up outside your immediate field of expertise, you provided me with extremely detailed feedback on the writing, presentations, and content I developed. Throughout our meetings, we had a lot of laughs about work, about current-day events and newly thought out government or university policies. You have gradually become a mentor for me that provided me with career and life advice whenever I needed it. We enjoyed a lot of things outside of work as well: swimming, running, biking, and all the group activities we did together were especially nice for me.

Ricky, your never-ending enthusiasm and optimism throughout my PhD was really special to experience. You always managed to find positivity in any situation, regardless of the circumstances. You never ceased to amaze me at the speed you had your security-related jokes ready, making our meetings apart from useful also enjoyable. Outside of work I got to experience the same enthusiasm that I saw in the office, during our many ATO trips or at one of your Koperen Kat performances.

Another advantage of working in two groups is that I got to meet and work with twice as many colleagues, of which many I now consider friends. Hemmo and Vis, you were among the first people I got to know in Delft and I am very happy I did. We have done countless things together, of which our weekends abroad were major highlights. The stroopwafel incident after the Golden Ten run stands out as well. Elise, thanks to you I never missed any news about my research domain: you provided me with countless tips and contacts that helped shape this thesis. You were always there for a quick chat and a coffee, making my time at the office more fun. Matt, you entered my office and made it a more fun place. We oftentimes shared ideas about each other's work but also got to enjoy several activities outside the office, such as visiting Noordeinde Palace. Vinh,

we got to know each other over a few games of tennis early in my PhD, and we became friends along the way. I really enjoyed visiting you in Vietnam, in which you introduced me to your friends and family, as well as Vietnamese culture and food.

Eric, we moved into an office together halfway into my PhD, and I have always enjoyed your company. The major highlight for me was visiting you in your hometown Ezhou in China, in which we got to enjoy good company, delicious food, skillful karaoke performances, and a typical Chinese game bar. Jorik and Belma, I really enjoyed the board game nights, pancake baking adventure and the never-ending stream of random news facts entering our group chat.

While writing my PhD thesis, I was lucky to work with many talented Bachelor and Master students. The works of Arjan and Diogo made a vital impact on Chapters 3 and 5 of this thesis, while contributions of Anne-Nynke, Arthur, and Adin made tangible differences to the development of the models described in this thesis. Finally, the enthusiasm and dedicated data collection effort of Régis allowed me to develop the data-driven approach which now forms Chapter 6 of this thesis.

I could not have done this research without the help of several people from Rotterdam The Hague Airport. First, thanks to Steven for enabling this collaboration, and introducing me to Alexander. I have got to know you, Alexander, as a very positive person, with whom I really enjoyed working with. You enabled me to combine academic research with the practical aspects of airport security, which was a unique experience. A special thanks also to Bas Simons for his help with collecting the data.

Outside of Delft, I have also enjoyed the personal support of many people. Thanks to my friends in and outside Limburg for coming to visit me in Delft. And a special thanks to Wessel for designing the cover art of this book.

I especially want to thank my family for always being there for me. My parents, for supporting me in whatever challenge, trip, or adventure I decided to undertake. Also thanks to my sisters Sanne, Shenna, and Sharon for their support and company in and outside Velden. Thanks to oma, for the countless lunches, Skip Bo games and a never-ending supply of drop.

Finally, I want to thank Sjoukje for coming into my life, bringing more joy and happiness. Thank you for all the exciting moments and trips that we experienced together, and for all the times that you were there when I needed it. I am looking forward to much more of that in the future.

SUMMARY

Airports are important transportation hubs that reside in the heart of modern civilizations. They are of major economic and symbolic value for countries but are therefore also attractive targets for adversaries. Over the years we have observed successful and unsuccessful terrorist attacks at airports, of which the recent Brussels Airport attack and Istanbul Atatürk Airport attack are two examples.

A widely-used method to defend airports against these types of events is that of security risk management. Following this approach, security risks are quantified based on threats, vulnerabilities, and consequences. These risks are then used as a basis to implement security measures that can reduce the risks to acceptable levels. Several security risk management approaches were proposed before, such as attack trees and security games, but they struggle to include diverse human factors in their analysis. These factors are inherently present in modern airports, as passengers, employees, and visitors are all humans. Furthermore, existing methods struggle to take other performance metrics, such as efficiency, into account.

This thesis addresses these limitations by proposing a novel security risk management approach that relies on agent-based models and Monte Carlo simulations. This approach builds on the existing security risk management framework but exploits the advantages of the agent-based modelling paradigm. Agent-based models allow for the inclusion of rich cognitive, social and organizational models that enable the modelling of human behaviour. Furthermore, agent-based modelling is a suitable paradigm to estimate a variety of performance indicators, including airport efficiency.

Two case studies were performed to assess the performance of our agent-based security risk management approach. In these case studies we apply our approach to manage security risks at a regional airport, as well as an international airport.

In the first case study, we focus on the decision-making and performance of security operators at the security checkpoint. Through simulation, we found that the highest skilled operators outperform their lowest-skilled counterparts on analyzing X-ray images, but perform worse on both searching luggage and performing patdowns. Furthermore, results show that a high focus on speed by security operators leads to a decrease in luggage searches and therefore increased vulnerability.

In the second case study, we analyzed security risks regarding an Improvised Explosive Device (IED) attack. Additionally, different commonly used efficiency performance indicators in the aviation domain, such as queuing time for passengers, and the relationships between them. We showed that airport managers and regulators often have to make important trade-offs regarding security and efficiency. However, it was found that reducing security risks and improving efficiency are not always conflicting objectives. Decreasing the number of passengers in the open areas of the airport was found to be an effective measure to reduce security risks and improve different efficiency metrics, such as queuing times.

One of the most critical limitations of this thesis is that of data availability. Due to the nature of airport security, there is only a minimal amount of data available in the public domain. While we have performed an extensive data-collection effort and used publicly available data to calibrate our models, this lack of security data enforced us to make assumptions about different model parameters. These assumptions may have lead to inaccurate simulation results. The models can, however, easily be re-calibrated when more data becomes available.

Agent-based modelling comes with its challenges. It is known that designing agent-based models and analyzing them is a complex task. Agent-based models are designed following a bottom-up approach, in which actors, the environment, and interactions are all explicitly modelled. It is often up to experts to specify the behaviour of agents, and the quality of the model therefore ultimately depends on their skills.

We, therefore, proposed a novel methodology, based on causal discovery, that aids experts in specifying the behaviour of agents in a model. Causal discovery algorithms generate causal graphs that depict causal relationships between variables. By applying these algorithms to real-world data that captures the behaviour of an actor, causal graphs are generated that are then used to specify an agent. We applied our methodology to a case study in the security checkpoint domain. Results indicate that models designed with our approach show closer resemblance with validation data than models designed by experts alone.

Agent-based models can produce complex patterns that emerge from the behaviour and interaction of agents. To improve the toolbox of analysts, we proposed a novel methodology that uses causal discovery to characterize emergence in agent-based models. Using our methodology, we showed that queue length is an important causal factor in the number of casualties in the case study concerning the improvised explosive device (IED) attack. This emergent property was well identified using our methodology but is hard to identify with traditional analysis techniques alone.

Finally, in this thesis we developed an open-source agent-based simulator called AATOM. The simulator contains calibrated presets and templates for important airport elements, such as the security checkpoint. We additionally provided a dataset that contains data of a total of 2277 passengers that passed through the security checkpoint process at Rotterdam The Hague Airport (RTM) to the research community. These resources enable future researchers to develop and calibrate their own agent-based airport models.

SAMENVATTING

Luchthavens zijn belangrijke transportknooppunten die zich in het hart van moderne beschavingen bevinden. Ze zijn van grote economische en symbolische waarde voor landen, maar zijn daarom ook aantrekkelijke doelen voor kwaadwillenden. In de loop der jaren hebben we succesvolle en mislukte terroristische aanslagen op luchthavens gezien, waarvan de recente aanval op Brussels Airport en de aanval op Istanbul Atatürk Airport twee voorbeelden zijn.

Een veelgebruikte methode om luchthavens tegen dit soort risico's te beschermen, is die van veiligheidsrisicobeheer. In deze aanpak worden veiligheidsrisico's gekwantificeerd op basis van bedreigingen, kwetsbaarheden en consequenties. Deze risico's worden dan gebruikt als basis om veiligheidsmaatregelen te nemen die de risico's tot aanvaardbare niveaus kunnen verminderen. Eerder werden verschillende veiligheidsrisicobeheerbenaderingen voorgesteld, zoals attack trees en security games, maar deze hebben moeite met diverse menselijke factoren mee te nemen in hun analyse. Deze factoren zijn inherent aanwezig op moderne luchthavens, omdat passagiers, werknemers en bezoekers allemaal mensen zijn. Bovendien hebben bestaande methoden moeite om rekening te houden met andere performance indicatoren, zoals efficiëntie.

Dit proefschrift behandelt deze beperkingen door een nieuwe benadering voor het beheer van veiligheidsrisico's voor te stellen die gebaseerd is op agent-gebaseerde modellen en Monte Carlo-simulaties. Deze aanpak bouwt voort op het bestaande framework voor veiligheidsrisicobeheer, maar maakt gebruik van de voordelen van agent-gebaseerde modellen. Agent-gebaseerde modellen kunnen rijke cognitieve-, sociale- en organisatorische modellen bevatten die het modelleren van menselijk gedrag mogelijk maken. Bovendien is agent-gebaseerde modellering een geschikt paradigma om een verscheidenheid aan performance indicatoren te schatten, waaronder luchthaven-efficiëntie.

Er zijn twee case studies uitgevoerd om de prestaties van onze agent-gebaseerde veiligheidsrisicobeheerbenadering te beoordelen. In deze case study's passen we onze aanpak toe om veiligheidsrisico's op een regionale en internationale luchthaven te beheren.

In de eerste case study richten we ons op de beslissingen en prestaties van veiligheidsmedewerkers bij het security checkpoint. Door middel van simulatie hebben we geconstateerd dat de meest bekwame operators beter presteren dan hun minst bekwame tegenhangers bij het analyseren van röntgenfoto's, maar slechter presteren bij het doorzoeken van bagage en het uitvoeren van patdowns. Verder tonen de resultaten aan dat een hoge focus op snelheid door veiligheidsmedewerkers leidt tot een afname van bagagedoorzoekingen en dus tot een verhoogde kwetsbaarheid.

In de tweede case study hebben we veiligheidsrisico's geanalyseerd met betrekking tot een aanval met een Improvised Explosive Device (IED). Daarbij hebben we verschillende veelgebruikte efficiëntie performance indicatoren in het luchtvaartdomein, zoals wachtrijtijden voor passagiers en de onderlinge relaties onderzocht. We hebben laten

zien dat luchthavenbeheerders en toezichthouders vaak belangrijke afwegingen moeten maken met betrekking tot veiligheid en efficiëntie. Er werd echter vastgesteld dat het verminderen van veiligheidsrisico's en het verbeteren van de efficiëntie niet altijd tegenstrijdige doelstellingen zijn. Het verminderen van het aantal passagiers in de publiekelijk toegankelijke gebieden van de luchthaven bleek een effectieve maatregel te zijn om veiligheidsrisico's te verminderen en verschillende efficiëntie performance indicatoren te verbeteren, zoals wachtrijtijden.

Een van de meest belangrijke beperkingen van dit proefschrift is die van de beschikbaarheid van data. Vanwege de aard van luchthavenveiligheid is er slechts een kleine hoeveelheid data beschikbaar in het publieke domein. Hoewel we een uitgebreide dataverzameling hebben uitgevoerd en openbaar beschikbare data hebben gebruikt om onze modellen te kalibreren, dwong dit gebrek aan veiligheidsgegevens ons om aannames te doen over verschillende modelparameters. Deze aannames kunnen geleid hebben tot onnauwkeurige simulatieresultaten. De modellen kunnen echter gemakkelijk opnieuw worden gekalibreerd wanneer meer data beschikbaar komen.

Agent-gebaseerde modellering brengt belangrijke uitdagingen met zich mee. Het is bekend dat het ontwerpen en analyseren van agent-gebaseerde modellen een complexe taak is. Agent-gebaseerde modellen zijn ontworpen volgens een bottom-up benadering, waarbij actoren, de omgeving en interacties allemaal expliciet worden gemodelleerd. Het is vaak aan experts om het gedrag van agenten te specificeren, en de kwaliteit van het model hangt daarom uiteindelijk af van hun vaardigheden.

We hebben daarom een nieuwe methode voorgesteld, gebaseerd op causal discovery, die experts helpt bij het specificeren van het gedrag van agenten in een model. causal discovery algoritmen genereren causale graven die causale relaties tussen variabelen weergeven. Door deze algoritmen toe te passen op data uit de praktijk die het gedrag van een actor vastleggen, worden causale graven gegenereerd die vervolgens worden gebruikt om een agent te specificeren. We hebben onze methodologie toegepast op een case study in het domein van de security checkpoints. De resultaten laten zien dat modellen die zijn ontworpen met onze aanpak meer lijken op validatiedata dan modellen die door experts alleen zijn ontworpen.

Agent-gebaseerde modellen kunnen complexe patronen produceren die voortkomen uit het gedrag en de interactie van agenten. Om de toolbox van analisten te verbeteren, hebben we een nieuwe methodologie voorgesteld die causal discovery gebruikt om emergent gedrag in agent-gebaseerde modellen te karakteriseren. Met behulp van onze methodologie toonden we aan dat de wachtrijlengte een belangrijke causale factor is in het aantal slachtoffers in de IED case study. Deze emergente eigenschap werd goed geïdentificeerd met behulp van onze methodologie, maar is moeilijk te identificeren met traditionele analysetechnieken alleen.

Ten slotte hebben we in dit proefschrift een open-source agent-gebaseerde simulator ontwikkeld, genaamd AATOM. De simulator bevat gekalibreerde waardes voor belangrijke luchthavenelementen, zoals het security checkpoint. We hebben bovendien een dataset met gegevens van in totaal 2277 passagiers die het security checkpoint op Rotterdam The Hague Airport (RTM) hebben doorlopen aan de onderzoeksgemeenschap verstrekt. Met deze middelen kunnen toekomstige onderzoekers hun eigen agent-gebaseerde luchthavenmodellen ontwikkelen en kalibreren.

CONTENTS

1

INTRODUCTION

Despite enormous investments in airport security, terrorists have been able to find and exploit vulnerabilities at airport terminals. In the years after 9/11, aviation has been targeted by several bombing attempts, such as the shoe bomber [1], the Istanbul Atatürk Airport attack [2], and the Brussels airport attack [3].

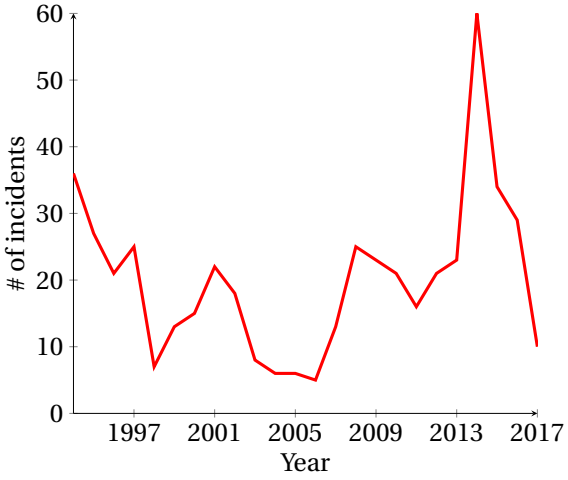


Figure 1.1: The number of terrorism-related incidents logged since 1994 in the Global Terrorism Database (GTD) that are targeted at airports or aircraft [4].

Protecting airports is of utmost importance, as these are often vital infrastructures for countries. Figure 1.1 shows the number of terrorism-related incidents targeted at airports or aircraft between 1994 and 2017. It shows a large number of incidents in recent years, highlighting the need for effective methods to combat them.

One of the most widely-used methods used by airports to address these types of incidents is security risk management. Security risk management for airports is a process

aiming to identify, calculate and mitigate security risks of an airport by utilizing a finite set of resources. An important part of this process is security risk assessment, in which security risks of the airport are identified and calculated. The modeling, assessment, and management of airport security risks is the core focus of this thesis.

1.1. SECURITY RISK MANAGEMENT

Security risk management can be performed using a wide variety of techniques, and each has advantages and disadvantages. Methods to perform security risk management are broadly classified into two categories: qualitative and quantitative risk management. Qualitative risk management is for instance based on questionnaires, intelligence data, and interviews. In quantitative risk management, numeric tools are used to guide the risk management process. In this thesis, we focus on quantitative security risk management. Several quantitative security risk management methodologies have been proposed in literature. Expert-based methods such as the Threat, Vulnerability and Consequence (TVC) methodology [5–9] are commonly used in practice. Furthermore, researchers have developed computational methods, such as attack trees [10, 11], probabilistic methods [12], discrete event simulation [13], and security games [14, 15].

We first provide important security-related definitions. Then, three important methodologies for security risk management are introduced: the TVC methodology, security games, and attack trees. While other methods, like probabilistic tools [12], the bowtie method [16], and discrete event simulation [13], exist, we focus our review on these three popular methodologies. These three methods are commonly used in practice and can exemplify many of the limitations that the other methods mentioned above also possess [16, 17].

It should be noted that security games and attack trees are often not defined as security risk management methodologies in literature, but as security-related resource allocation methodologies. They can however easily be regarded as security risk management methodologies.

1.1.1. SECURITY DEFINITIONS

The most important definitions related to security that we use across this thesis are shown below. A central topic in security is that of risk. While many definitions exist, in this thesis we employ a commonly used definition of risk [9, 18–20].

Definition 1 (Security risk)**.** The potential for loss or harm due to the likelihood of an unwanted event and its adverse consequences.

We use the terms security risk and risk in this thesis interchangeably. Risk is often expressed in terms of threats, vulnerabilities, and consequences. Their respective definitions are shown in [9] and are repeated below for convenience.

Definition 2 (Threat)**.** Any indication, circumstance, or event with the potential to cause the loss of, or damage to, an asset.

Definition 3 (Threat Scenario)**.** A set of events, associated with a specific threat or multiple threats, partially ordered in time.

Definition 4 (Vulnerability)**.** Any weakness in an asset's or infrastructure's design, implementation, or operation that can be exploited by an adversary.

Definition 5 (Consequence)**.** The outcome of an event occurrence, including immediate, short- and long-term, direct and indirect losses and effects.

Conditional risk is another common term used in literature and used in this thesis. It is defined as follows [9].

Definition 6 (Conditional Risk)**.** A measure of risk that focuses on consequences, vulnerability, and adversary capabilities, but excludes intent.

As assets are an important element in the definitions above, we provide the International Organization for Standardization (ISO) definition of an asset below [21].

Definition 7 (Asset)**.** Item, thing or entity that has potential or actual value to an organization.

To be able to reduce risks, organizations can take measures. These measures are defined as controls and its definition is stated below.

Definition 8 (Control)**.** Measure that is modifying risk.

1.1.2. TVC METHODOLOGY

In the Threat, Vulnerability, and Consequence (TVC) methodology, security experts first characterize important assets in their organization. Based on these assets, they identify a set of threats that the assets are exposed to. Risk is then characterized by estimating threat likelihood, vulnerability and consequence separately for each identified threat. Finally, risk mitigation is performed to reduce risks to an acceptable level. In practice, many different variants of the TVC methodology exist [5–9], but we focus on the overlap between these methods in this thesis.

Threat likelihood is often estimated based on intelligence data or a cost/benefit analysis. Historical data, such as the Global Terrorism Database [4], can also be used to determine the threat likelihood. However, there is no guarantee that the available historic data is an indication of future events.

To estimate vulnerability, security experts for instance use data provided by security sensor manufacturers, internal assessments and employee surveys. Also, tools like vulnerability logic diagrams and event trees [22] can be used to better estimate vulnerability. Red-teaming (real-life simulation of a threat scenario) can be used by experts as well. Vulnerability estimates are sometimes 'binned', as is shown in Table 1.1, to simplify the assessment process.

The consequence of a threat can be quantified using consequence assessment techniques, where most commonly, they are expressed in monetary values. The loss of a human life can, for instance, be quantified by using a 'value of a single life' (VSL), as also discussed in [23, 24]. Consequences are commonly estimated based on expert judgment.

Finally, risk mitigation is performed by comparing the expected reduced security risks for potential controls with the current situation. Costs and operational applicability are also taken into account in this step.

Table 1.1: An example vulnerability table that is used to categorize vulnerabilities. Table adapted from [9].

Vulnerability Range (%)	Bin Number
≤3.11	0
3.12-6.24	1
6.25-12.4	2
12.5-24.9	3
25-49	4
50-74	5
75-89	6
90-100	7

1.1.3. ATTACK TREES

Attack trees provide a formal, methodical way of describing the security of systems, based on varying threat scenarios [11]. The main concept of an attack tree is that an attack against a system is represented in a tree structure. The root node (also top-event) represents a successful attack on some asset within the system. Internal nodes represent events that depend on their subsequent child nodes, while leaf nodes represent events that can independently happen. Nodes can be attributed values that represent their likelihood, their cost to execute and other parameters. Leaf nodes are valued by the designer, while the value of other nodes is calculated from the values of their child nodes. Transitions between nodes can be modeled to be deterministic and non-deterministic. In the case of deterministic transitions, a (combination of) child node(s) occurring will certainly lead to the occurrence of the parent node, while in non-deterministic transitions this is not the case. By analyzing the values of the root node of the tree, controls can be taken accordingly. Figure 1.2 presents an example attack tree that partially models the threat scenario used in the illustration.

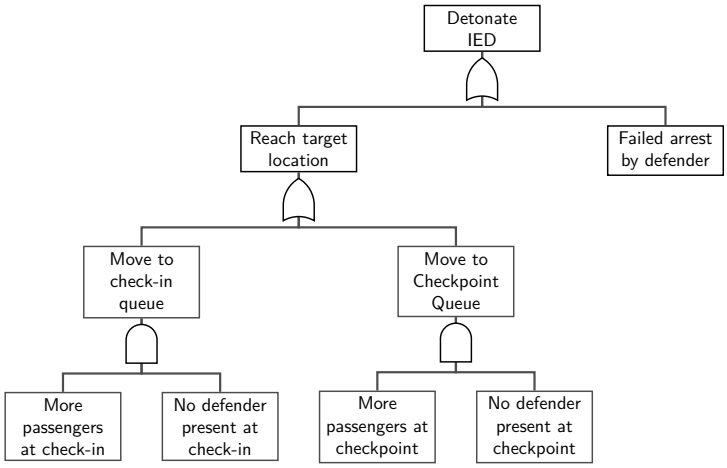


Figure 1.2: An example attack tree with two types of nodes: AND and OR.

Alternatively, attack-defense trees form an addition to the attack trees described above. In attack-defense trees, the designer can introduce defense nodes. The addition of defense nodes in attack-defense trees allows for the modeling of interactions between attacker and defender, impossible in attack trees. This allows for a more elaborate analysis of the effectiveness of different controls, useful for determining which controls should be installed. Some important work in this area is by Kordy et al. [25], Bistarelli et al. [26] and Edge et al. [27].

1.1.4. SECURITY GAMES

Methods based on game theory [14, 28, 29] define a threat scenario as a security game, with a defender and an attacker as the respective row and column players of the game. Columns represent the options an attacker has to attack a target, whereas rows represent the available actions the defender has to defend the target. Based on the chosen options of the attacker and defender, an outcome (often a combination of vulnerability and consequence) is determined. By finding the equilibrium of such a game, an optimal strategy for the defender can be obtained. An example of a simple security game is visualized in Table 1.2. Security games have found their application in a wide variety of areas, such as airports [14, 29], coastal protection [30], wildlife protection [31] and chemical plants [32].

Table 1.2: An example security game. The row player is the defender, the column player is the attacker. The described payoffs are for the defender (first value) and the attacker (second value).

	Att. checkpoint	**Att. check-in**
Def. checkpoint	10,-80	-100,100
Def. check-in	-80,80	20,-100
Do not def.	-90,80	-90,100

A large portion of security games in literature focus on finding a patrol route for security employees called patrol planning games. These games are played on graphs where nodes represent targets and edges are spatial connections between targets. The solution of such a game results in a patrol, which is generally a vector that consists of targets and times. These patrol planning games have found their applications in wildlife protection [33], maritime transport security [34], and airport security [35].

1.1.5. EVALUATION OF EXISTING METHODOLOGIES

In this section, we provide a critical evaluation of the existing security risk management methodologies. We focus on three main areas: incorporation of human aspects, incorporation of spatio-temporal aspects, and integration of efficiency.

HUMAN ASPECTS

Airports are socio-technical systems, and human behavior plays an important role. The incorporation of intelligence and other dynamic, human aspects into the risk assessment is difficult for security experts. It is often noted in literature that security experts cannot be expected to estimate parameters well [36, 37], certainly in dynamic environments with many actors. Leung and Verga[37] mention that “limitations of human mem-

ory and information processing capacity often lead to subjective probabilities that are poorly calibrated or internally inconsistent, even when assessed by experts".

An important underlying assumption of game theory is that the players take rational decisions. However, researchers note that "human decision-making does not conform to the traditional game-theoretic assumption of perfect rationality" [38, 39]. While researchers try to overcome this limitation by for instance employing prospect theory [40] and quantal response [41], the problem remains an active area of research. Furthermore, it should be noted that security games often focus on one-to-one interactions between an attacker and a defender. However, general social interactions, like group decision making, are present in many threat scenarios.

Attack trees suffer from similar limitations as security games. Attack-defense trees have the possibility to include higher-level interactions between the attacker and the defender. However, authors also note that they are "not suitable for including human interaction such as that of social engineering, because the attacker may combine different persuasion principles to different degrees, with different associated success probabilities" [42].

SPATIO-TEMPORAL ASPECTS

Airports are physical structures in which people dynamically move around. These spatio-temporal elements can have a significant impact on the consequence of an attack. Security games struggle with incorporating spatio-temporal elements into their models. Some recent work in security games aims to incorporate these elements by using deep learning on images of forests [43]. However, it is unclear if this can also be used in other domains. Similarly, attack trees struggle with the incorporation of spatio-temporal elements. The concepts of time and space are not intuitively represented in an attack tree, and therefore this method cannot easily include these elements in the risk assessment.

EFFICIENCY

While security is a vital aspect of airport operations, these airports must be run as efficiently as possible as well. Security experts often analyze how efficient (i.e. expected passenger queuing time or number of employees needed per passenger) a proposed security solution is. For instance, Grant and Stewart followed the TVC methodology to manage security risks related to an Improvised Explosive Device (IED) attack, while taking into account costs for the airport [44]. Experts often have a limited amount of time, and can therefore not evaluate the impact of all possible controls.

Both security games and attack trees can incorporate efficiency aspects into their models. For security games, efficiency factors can be taken into account in the payoffs. For attack trees, this can be done by taking into account efficiency factors in the valuation of nodes. However, this form of incorporation of efficiency in the security models is limited. The advantages of multi-objective analysis techniques, such as Pareto front analysis, cannot be exploited using this technique.

1.2. AGENT-BASED MODELING

Agent-based modeling is a promising paradigm that has the potential to overcome the above-outlined limitations of existing security risk management approaches. Agent-

based models attempt to capture the behavior of the actors in complex systems to better understand them and potentially increase their performance. They are characterized by an environment, agents, and their interactions. Agent-based models have been used in many application areas: finance [45], urban planning [46], segregation [47], and ecology [48], among many others.

Apart from agent-based modeling, discrete event simulation may be promising tool to overcome the above-mentioned limitations of existing security risk management approaches [13]. However, in discrete event simulation models "the entities do not actively follow individual incentives and do not interact but pass through the model according to the underlying sequence of operations" [49]. As we aim to overcome the lack of incorporation of human behavior in existing models, an agent-based approach is more suitable than discrete event simulation.

Agent-based models are important tools to model realistic socio-technical processes, by including rich cognitive, social and organizational models. They can also be used to explicitly represent spatio-temporal elements of agents and the environment. This then allows for the modeling of the transition between standard operations of an airport and operations under attack. These are aspects that existing security risk management methodologies struggle to take into account.

It is well known in the field that agent-based modeling comes with its own challenges. We particularly focus on designing and analyzing agent-based models, which are two important open problems in the agent-based community.

1.2.1. DESIGNING AGENT-BASED MODELS

Designing agent-based models is a complex task. Numerous tutorials and guidelines exist that cover the design of agent-based models, but they are often limited in detail [50–55]. These guidelines commonly specify that the three main components have to be specified, but offer little detail on how to do this.

It is recognized by the community that a uniform framework or methodology for designing agent-based models is lacking [50, 54]. The 'overview, design concepts, and details' (ODD) protocol aims to overcome this and has been advocated widely in literature [56, 57].

While the ODD protocol contains detailed steps to design agent-based models, no insights on how to design the behavioral properties of agents are provided. With the right dataset, data-driven methods may be useful to specify behavioral properties of agents [58]. These data-driven methods find relationships between variables in a dataset, which could determine relationships between actions of agents and the outcomes in the environment.

This idea of using data-driven methods to design agent-based models has been explored by Kavak et al. [58]. In that work, behavioral properties of agents are learned from data by applying machine learning techniques, such as support vector machines and decision trees. While these more traditional machine learning techniques are effective tools to understand how variables relate to each o, they do not reveal the structure of relationships between variables. A particularly promising method to reveal this structure is that of causal discovery.

1.2.2. ANALYZING AGENT-BASED MODELS

Complex interactions of agents with each other and the environment can lead to the emergence of higher-level patterns. These emergent properties are an important feature of agent-based models but are hard to characterize. Some work was done to classify types of emergence, and is generally based on desirability [59, 60] or complexity [61, 62]. Two desirability categories are distinguished: positive emergence and negative emergence. Positive emergent properties are desired outcomes of interactions of agents, while negative emergent properties are not. This categorization is commonly determined by experts that have a good understanding of the modeled domain. Several levels of emergence complexity are distinguished as well. They range from simple and weak, to strong and even spooky. Simple emergence can easily be predicted and reproduced, and weak emergence can easily be reproduced in simulations. Strong emergence cannot be reproduced by simple models and is hard to understand without deep knowledge of the system. Finally, spooky emergence cannot be explained nor predicted with the current knowledge of the system.

Sensitivity analysis techniques are commonly used to analyze the behavior of agent-based models and the corresponding emergent properties [63–65]. These techniques analyze the input-output relations of the model but do not reveal the inner structure of agent-based models. More recently, machine learning techniques have found their application in analyzing agent-based models as well [66]. These techniques identify patterns in the input and output of the model and generate meta-models that predict model outputs. These techniques have shown successes in the past, but it remains difficult to analyze emergent behavior. A particularly promising method to understand agent-based model behavior is that of causal discovery.

1.3. CAUSALITY

Traditional analysis techniques are used to determine how two or more variables are related. This can indicate that one causes the other, but a confounding factor can also influence both variables at the same time. In the field of causality, researchers aim to find directed causal relationships between variables, by means of causal graphs [67, 68]. A causal graph is most commonly a Directed Acyclic Graph (DAG) that depicts the causal relations between variables. An arrow from variable X to variable Y means that the former variable causes the latter. If no arrow between X and Y exits, this means that X does not cause Y.

Causal graphs can be analyzed by determining which variables form causal paths with other variables in the graph. Another way to use causal graphs is by determining the effectiveness of an experiment to reveal the strength of a causal relationship. A detailed description of this approach was introduced by Pearl [67]. Causal effects between variables can be quantified using these graphs as well [69].

Two main methods for creating causal graphs exist. In the first approach, experts use available knowledge and theories to construct a graph. Shrier and Platt [70] provide an example of this expert-based approach. In the second method, as also used in this thesis, causal-discovery algorithms are used to automatically generate causal graphs based on available data. Two important methods to perform causal discovery have emerged in the field: score-based methods (e.g. [71]) and constraint-based methods (e.g., [72–74]).

Score-based methods assign a score to a causal graph, while constraint-based methods use the statistical independence of variables to define constraints on causal graphs. Malinsky and Danks provide a practical guide for using causal-discovery algorithms [75].

1.3.1. AGENT-BASED MODELING AND CAUSALITY

A limited amount of work exists to bridge the gap between the fields of causality and agent-based modeling. One of the most extensive works to date is that of Casini and Manzo [76]. They argue that, in an ideal case, a modeler ensures that their agent-based model 1) uses all available theories to explain reality, 2) is calibrated with real data, and 3) is validated with real data. In this way, an agent-based model is not just a counterfactual (something contrary to facts) but can be used to draw causal conclusions about the real world. They provide a very basic methodology to use agent-based modeling for causal inferences, as outlined below.

1. Employ experimental and statistical data to show that the assumed causal links are unlikely to be random.
2. Define hypotheses about causal relationships.
3. Translate hypotheses to an agent-based model by also incorporating data.
4. Run agent-based simulations to determine if the hypothesized causal relationships are still observed.

The methodology is only a very small part of the paper and has not been applied to an illustrative case study. Furthermore, it only consists of a set of higher-level steps that cannot easily be used in practice and does not exploit the strengths of causal graphs commonly used in the field of causality.

The work of Kvassay et al. [77] employs a more computational approach towards combining agent-based models with causality. They investigate causal relationships that lead to emergent behavior in an agent-based model. The core of their work revolves around the concept of causal partitions. By using causal partitioning, the relative importance of influencing factors on an emergent phenomenon is determined. Their methodology heavily depends on the existence of difference equations in the definition of a model, while in practice these equations might not exist.

Guerini and Moneta [78] propose a method for agent-based models validation that uses causal discovery as a basis. They specifically focus on economic models that estimate time-series using so-called structural vector autoregressive (SVAR) models. They use causal discovery to generate two SVAR models: one based on real-world data, one based on agent-based model outcomes. These models are then compared using a distance measure, and the agent-based model is considered to be valid if the distance between the two SVAR models is sufficiently small. The method is specifically tailored for SVAR models, and it is hard to generalize to other types of agent-based models.

Finally, Marsha and Galea [79] discuss how agent-based modeling can be used as an alternative for two types of causal inference in epidemiology: randomized controlled studies and observational studies. They define an agent-based model that is used to simulate the development of an illness in a population of humans. By analyzing how certain

treatments change the distribution of sick and healthy people, higher-level causal inferences are drawn. Their work only focuses on the final outcome of a treatment, while intermediate (direct) causes are ignored.

None of the works that bridge the fields of agent-based modeling and causality address the important challenge of designing agent-based models. Furthermore, none of these works relate emergent properties of agent-based models to causal graphs that are commonly used in the causality field. These causal graphs identify structure in the output of agent-based models and form a promising means to analyze emergence in agent-based models.

1.4. PROBLEM STATEMENT & THESIS OVERVIEW

As outlined above, this thesis addresses the important challenge of managing security risks related to airport operations. To address this challenge, we employ the fields of agent-based modeling and causality. The following problem statement is defined, which is central to this thesis.

Can agent-based security risk management be performed using causal discovery?

To address this problem statement, the following six research questions are formulated. These questions will be answered in each of the chapters of this thesis.

1. *How can agent-based modeling be used to perform security risk management for airport operations?*

   We propose AbSRiM, an approach based on traditional security risk management methodologies, but with agent-based modeling and Monte Carlo simulation at its core in **Chapter 2**. The approach consists of four steps: scope selection, agent-based model definition, agent-based model analysis (risk assessment), and risk mitigation.

2. *How can human factors be taken into account while performing security risk management?*

   In **Chapter 3**, we design an agent-based model that model the performance and decision making of security operators using cognitive agent models. We specifically focus on the vulnerability of airport security checkpoints.

3. *How can performance metrics, such as operational efficiency, be taken into account while performing security risk management?*

   In **Chapter 4** we analyze security risks regarding an Improvised Explosive Device (IED) attack, in combination with different commonly used efficiency performance indicators in the aviation domain, such as queuing time for passengers.

4. *How can efficient airport security patrol routes be designed using agent-based modeling?*

   We present an alternative method to find efficient airport security patrol routes, a risk mitigation strategy, that combines our agent-based approach with game theory. This is presented in **Chapter 5**.

5. *How can agent-based models be designed using causal-discovery algorithms?*

   Chapter 6 addresses the problem of agent-based model development using causal discovery. In this chapter, we present a novel methodology that uses causal discovery to aid the development of agent-based models.

6. *How can agent-based models be analyzed using causal-discovery algorithms?*

   Model analysis is an important part of the risk assessment step in the AbSRiM approach, and an important open problem in the agent-based community. We propose the AbACaD methodology, which uses causal-discovery algorithms, to analyze emergence in agent-based models in **Chapter 7**.

Table 1.3: An overview of scope of each of the chapters in this thesis, based on the four steps of the AbSRiM approach. *ABM* stands for using agent-based models methods in the steps, *GT* stands for the additional application of game theory in these steps. *CD* represents the application of causal discovery for the steps. When a cell is empty, the specific step is not considered in the chapter.

	Chapter 3	**Chapter 4**	**Chapter 5**	**Chapter 6**	**Chapter 7**
Scope selection	ABM	ABM		CD	
Agent-based model definition	ABM	ABM		CD	
Agent-based model analysis	ABM	ABM	GT		CD
Risk mitigation	ABM	ABM	GT		

Table 1.3 provides an overview of the scope of each of the chapters in this thesis. Chapter 2 introduces the AbSRiM approach, in which we use agent-based models to manage security risks. The steps of the AbSRiM approach (scope selection, model definition, model analysis, and risk mitigation) form the rows of the table. Chapters 3 and 4 then provide two case studies in which the AbSRiM approach is applied. Chapter 5 extends Chapter 4 and shows that game theory can additionally be incorporated in the analysis and mitigation of security risks. This leads to superior results in comparison to the agent-based approach in Chapter 4. Chapters 6 and 7 utilize causal-discovery algorithms to design and analyze agent-based models. The use of these algorithms reduces the dependency on domain experts for designing and analyzing agent-based models. The proposed methodologies are useful in the security domain but are also applicable to agent-based models in general.

1

Each chapter can be read on its own, but it is recommended to read Chapters 1 and 2 before continuing to other chapters of this thesis. Furthermore, it is recommended to read Chapter 4 before Chapter 5. Chapters 6 and 7 can be read on their own, and focus specifically on designing and analyzing agent-based models using causal discovery. The final Chapter 8 provides conclusions and recommendations for this thesis. Figure 1.3 shows a graphical outline of the thesis structure, in which the recommended reading order of this thesis is shown.

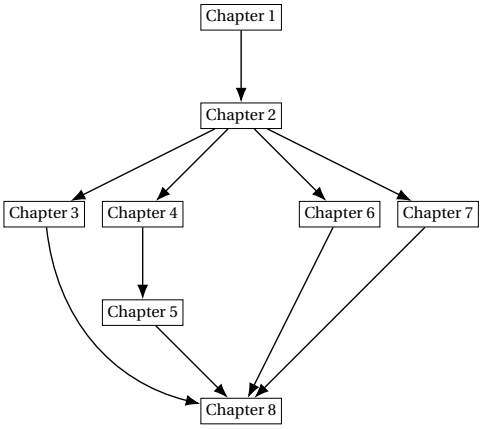


Figure 1.3: The recommended reading paths for this thesis.

2

AGENT-BASED SECURITY RISK MANAGEMENT

Security risk management is essential for ensuring successful airport operations. This chapter introduces AbSRiM, a novel agent-based modeling and simulation approach to perform security risk management for airport operations. It uses formal socio-technical models that include temporal and spatial aspects. The approach contains four main steps: scope selection, agent-based model definition, risk assessment, and risk mitigation. The approach is based on traditional security risk management methodologies, but uses agent-based modeling and Monte Carlo simulation at its core. Agent-based modeling is used to model threat scenarios, and Monte Carlo simulations are then performed with this model to estimate security risks. Chapters 3 and 4 will apply the AbSRiM approach to different case studies.

This chapter is adapted from previously published work in the proceedings of the International Conference on Practical Applications of Agents and Multi-Agent Systems (2017) [80] and the Risk Analysis Journal (2019) [81].

2.1. INTRODUCTION

Security risk management for airport operations is a process aiming to identify, calculate and mitigate security risks of the airport by utilizing a finite set of resources. An important part of this process is security risk assessment, in which security risks of the airport are identified and calculated. Methods to perform security risk assessment can be classified into two categories: qualitative and quantitative risk assessment. Qualitative risk assessment is for instance based on questionnaires, intelligence data and interviews. In this thesis, we focus on quantitative security risk assessment. Several security risk management methods that use a quantitative approach have been proposed in literature. Expert-based methods such as the Threat, Vulnerability and Consequence (TVC) methodology [5–9] are commonly used in practice. Furthermore, researchers have developed analytical methods like attack trees [10, 11], probabilistic methods [12], and security games [14, 15].

It is often observed that conventional methods have their limitations. For instance, these methodologies struggle to incorporate diverse social interactions, which are inherently present in many threat scenarios in airport operations. Furthermore, the transition between standard operations and operations under an attack is often not well modeled in current analytical models. Finally, most of the analytical models cannot properly take into account spatio-temporal aspects, such as the distribution of passengers over time, that are present in airports.

We therefore propose AbSRiM, a novel agent-based modeling and simulation approach to perform security risk management in airport operations. The approach is based on traditional security risk management methodologies, but has been designed to overcome the above-mentioned limitations. An agent-based model can be used to model realistic socio-technical processes, by including rich cognitive, social and organizational models. It can also be used to explicitly represent spatio-temporal elements of the agents and the environment. This then allows for the modeling of the transition between standard operations of an airport and operations under attack.

This chapter is structured as follows. Section 2.2 describes AbSRiM, the agent-based security risk management approach proposed in this thesis. A conceptual comparison with existing methodologies is made for the AbSRiM approach in Section 2.3. Finally, a conclusion is provided in Section 2.4.

2.2. ABSRIM: AGENT-BASED SECURITY RISK MANAGEMENT

Here, we introduce AbSRiM: an Agent-Based Security Risk Management approach for airport operations. An overview of the different steps in the AbSRiM approach is outlined below. The approach follows several of the main steps of the traditional TVC methodology, but steps 2 and 3 differ significantly.

1. Scope selection
   (a) Characterize assets
   (b) Identify threats
   (c) Construct n threat scenarios

2. Agent-based model definition
   (a) Define operational model M
   (b) Define security models $\mathbb{M} = \{M_1, \ldots, M_n\}$
3. Risk assessment
   (a) Estimate threat likelihood
   (b) Estimate conditional risk
4. Risk mitigation
   (a) Define maximum risks R_{max}
   (b) Identify controls K
   (c) Determine control strategy

The first step is used to determine the scope of risk management. Relevant assets of the airport have to be characterized, and based on the characterized assets, a set of security threats is identified. They are in turn used to construct a set of n threat scenarios. Next, an agent-based model M, the operational model, is defined. The operational model is a representation of operations in the airport and at least includes the identified assets. This model forms the basis for the subsequently created security models. Security models $M_1, \ldots, M_n$ extend operational model M, and are defined for each of the constructed threat scenarios in S. A security model extends the operational model and includes a non-empty set of adversary agents that execute the attacker actions in the threat scenario. These security models are later used to estimate security risks.

Then, threat likelihood is estimated using a traditional approach, while conditional risk is estimated using Monte Carlo simulations. Finally, risk mitigation is done by treating risks that are considered too high. This consists of defining the maximum risk per threat scenario and identifying a set of feasible controls that can be implemented. Based on these, the best control strategy is determined using different analysis techniques.

2.2.1. SCOPE SELECTION

The selection of scope is the first step of the AbSRiM approach. It consists of three parts: (a) identification of assets, (b) identification of threats, and (c) construction of threat scenarios. Each of these parts are used to determine the focus of the rest of the steps in the approach.

A set of assets is identified that will be used in the risk assessment. An asset can for instance be the physical structure of an airport terminal or passengers that visit it. Ideally, a complete set of assets is identified. However, identification of a subset of important assets still allows for the execution of a security risk management with a narrower focus.

Based on the identified assets, different threats that relate to these assets are identified. Threats are identified using a method that is similar to the classic TVC methodology. In this method, security experts generate a list of threats based on their experience, intelligence data, and historic data. Similar to the identification of assets, a subset of

important threats can also be chosen. This gives the security risk management procedure a narrower focus. The identified threats are then used by security experts to construct threat scenarios. These threat scenarios are used to estimate security risks in the subsequent steps. The selected scope in this step forms the basis for the definition of agent-based models in the next step.

2.2.2. AGENT-BASED MODEL DEFINITION

The definition of the agent-based model is the second step of the AbSRiM approach. Two types of agent-based models are defined in this step: an operational model M, and a set of security models $M_1, \ldots, M_n$. The operational model is used to model standard operations that take place at the airport. In an airport, this consists of processes, such as the check-in process and the security check. The model should include a representation of each of the assets, in an operational context, which had been identified in the scope selection. A security model extends the operational model M and includes a representation of the attackers in a specific threat scenario. These attackers execute the attacker behavior in the threat scenario that was specified in the scope selection step.

Formally, in operational model M, an environment that represents the relevant airport operations is defined. Furthermore, a set of agents executing standard operations in the airport is defined. This can for instance be check-in employees or security officers. Finally, a set of defender agents is defined that can have operational tasks, such as answering passenger questions, and security-related tasks.

The operational model M forms the basis of the security models $M_1, \ldots, M_n$. A security model M_i extends model M by including a set of attacker agents that execute the attacker behavior in threat scenario s_i. These attacker agents interact with the defending agents by trying to prevent them from stopping their attack. The defenders, earlier defined in model M, in turn aim to stop an ongoing attack by the attacker agents.

These models require the selection of a modeling language. The selection of the language largely depends on the selected scope of security risk management, but certain aspects are required to be present. The desiderata for a modeling language include the following abilities: (1) to represent discrete and continuous time; (2) to specify stochastic processes; (3) to specify both qualitative and quantitative aspects; and (4) to represent behavioral and cognitive properties of agents and interaction between agents.

Discrete and continuous time specification is needed to be able to specify the dynamics of an attack in progress. Other dynamic processes can also be present: passengers moving in the airport terminal and checking in of passengers. Stochastic processes are inherently present in airport operations, for instance, the random arrival process of passengers, and random luggage checks at the security checkpoint. Furthermore, stochasticity is required for Monte Carlo simulations (see Section 2.2.3) to be useful. Modeling of human behavior involves reasoning, which requires the language to be able to express qualitative aspects. Quantitative aspects and relations are commonplace in airport operations. For instance, the number of flights on a day is important, as is the number of passengers that fly with a specific flight. Finally, representing cognitive and behavioral properties is important for modeling human agents, and is elaborated in the architecture desiderata.

The architecture should be capable to represent a range of functions for the agents in

the model: (1) making observations and perform actions; (2) to store information; (3) to maintain goals; and (4) to reason. Observing other agents and the environment, as well as performing actions is essential for any agent to perform its task. Another important aspect of an agent is that it should be able to store information that can be used later. For instance, this information can be used for maintaining internal goals of the agent. A goal of an airport passenger can for instance be to reach their gate in time, while a goal of an attacker can be to cause as many fatalities as possible. Finally, agents should be able to reason about their goals and the stored information to make decisions. As with the selection of the language, the selection of the architecture largely depends on the scope of the security risk management.

Example languages that can be used are the Temporal Trace Language (TTL) [82] and LEADSTO [83]. Example architecture is the BDI architecture [84], the CLARION architecture [85] or the Desire architecture [86].

After the operational model and the security models are specified, the models are validated. A large body of research is devoted to model validation [87–89]. Model validation is a difficult task, but most existing validation frameworks contain at least the following elements: ensure the *face validity* of the model, ensure the *internal validity*, and perform *sensitivity analysis*.

When ensuring face validity, domain experts verify if they think the model results are considered reasonable [90]. Then, internal validity is for instance verified by checking if the model produces similar outputs for different random seeds [91]. As part of internal validation, one can also perform tracing. In this case, agent traces are compared to expected behavior of agents. Sensitivity analysis is then done to determine the effects of changing model parameters on the output parameters [64]. The interested reader is referred to the work of Windrum et al. for an overview agent-based model validation [89].

It can be hard to validate models related to security. Often, limited or no data is available in this domain and performing field tests might be hard to do. In this case, experts play an essential role in the process of validating the model. In some cases real-life experiments can be done [92, 93], potentially improving the validity of the model. Furthermore, operational aspects of the models can more readily be validated using data.

2.2.3. RISK ASSESSMENT

The assessment of risks is the third step in the AbSRiM approach. For each threat scenario $s_i \in S$ constructed in step 1(c), a corresponding security risk r_i is calculated based on simulation results of model M_i defined in step 2. A security risk r_i is defined as a function of *Threat Likelihood* and *Conditional Risk*, and holds for some time period T. By estimating conditional risk, we ensure that dependencies between vulnerability and consequence are captured as well.

$$R(s_i, T) = f(P(s_i, T), R_c(s_i))$$

Conditional risk $R_c(s_i)$ is estimated as follows. For each security model M_i and asset a_l, a real-valued Consequence function $C(M_i^j, a_l)$ is defined. This function is used to determine the Consequence value for asset a_l of simulation run j in model M_i. It takes both direct losses and indirect losses into account. Direct losses can for instance include casualties of a simulated threat scenario. Indirect losses, such as longer-term business

disruptions, are then based on historical data and the estimated direct losses. If this consequence is 0, the attacker was unsuccessful in M_i^j.

By performing Monte Carlo simulations, the conditional risk is estimated based on N simulation runs. This is done as follows:

$$\hat{R}_c(s_i) = \frac{\sum_{j=1}^{N} \sum_{a_l \in A} C(M_i^j, a_l)}{N}$$

where $C(M_i^j, a_l)$ is the obtained consequence with respect to a specific asset a_l in threat scenario s_i, and $\hat{R}_c(s_i)$ is the estimator of the conditional risk for threat scenario s_i, $R_c(s_i)$. From a Monte Carlo perspective, conditional risk can be seen as the expected value of the consequence functions. The vulnerability of the scenario can be obtained by calculating the ratio between the number of nonzero consequence values and N (i.e. the total number of consequence values). The consequence of the scenario can be calculated by averaging the nonzero consequence values. Vulnerability and consequence values are not needed to calculate risks, but they can be used to guide the subsequent risk management step.

The total risk of all threat scenarios, denoted $R_{total}(T)$, is obtained by adding all risks for individual threat scenarios.

$$R_{total}(T) = \sum_{s_i \in S} R(s_i, T)$$

Threat likelihood $P(s_i, T)$ for threat scenario s_i is estimated by security experts independently from model M_i, as is commonly done in the TVC methodology. They base their estimates on historical data, intelligence data, and experience.

2.2.4. RISK MITIGATION

Risk management is the last step of the AbSRiM approach and is used to reduce the risks that were quantified above. In this step, specific controls (as part of control strategies) are investigated to reduce the risks to the system. To do this, acceptable risks per security threat are defined. If the estimated risks exceed the acceptability criteria, a control has to be implemented to reduce these risks.

This effectiveness to reduce risks is estimated as follows. The operational model and the security models are adapted, such that the control is incorporated in the model as well. Then, steps 3 of this approach is repeated to estimate the risk with the updated models. These newly estimated risks are then compared to the previously obtained estimates to determine their effectiveness to reduce risks. Controls are finally ranked based on their operational costs, operational usability and their effectiveness to reduce risks. Based on this ranking, airport managers can determine which (set of) control(s) is most suitable to implement.

2.3. COMPARISON OF ABSRIM WITH RELATED WORK

In this section, we provide a comparison between AbSRiM and existing security risk management methodologies (as discussed in Chapter 1) based on the following set of criteria: *independence from experts*, *human aspects*, *transition to threat*, *spatio-temporal*

aspects, *quality of assessment*, *availability of tools* and *ease of assessment*. It should be noted that this comparison is often on a more conceptual level.

2.3.1. INDEPENDENCE FROM EXPERTS

The TVC method relies on estimations from security experts that are used to estimate parameters such as vulnerability and consequence, but also perform the risk management step. Security games still rely on security experts to determine values for the specification of payoffs. In comparison with AbSRiM, the definition of a security game is easier to do then the definition of an agent-based model. Agent-based models require the definition of a large set of parameters, while security games only require a few. This leads to a larger dependency on domain experts by AbSRiM.

Compared to security games, more parameters need to be determined by security experts for attack trees as each leaf node needs to be valued by an expert. However, compared to AbSRiM, fewer parameters have to be defined for attack trees and it is easier to validate an attack tree.

AbSRiM can also be combined with machine learning techniques that allow for automatic identification of different threats. Based on the defined operational model (see Section 2.2.2), an attacker agent can be defined to learn which actions lead to consequences in the defined operational model. Learning of the attacker agent can be accomplished by using reinforcement learning techniques, like Q-learning [94]. A sequence of successful actions of the attacker (i.e. actions leading to a nonzero consequence) is then considered a threat scenario. This can further reduce the dependency on security experts and potentially improve the quality of this step. This machine learning process to identify threats can not straightforwardly be included in the alternative methodologies.

2.3.2. INCLUSION OF HUMAN ASPECTS

The incorporation of intelligence and other dynamic, human aspects into the risk assessment is difficult for security experts. It is often noted in literature that security experts cannot be expected to estimate parameters well [36, 37], certainly in dynamic environments with many actors. Leung and Verga [37] mention that "limitations of human memory and information processing capacity often lead to subjective probabilities that are poorly calibrated or internally inconsistent, even when assessed by experts".

An important underlying assumption of game theory is that the players take rational decisions. However, researchers note that "human decision-making does not conform to the traditional game-theoretic assumption of perfect rationality" [38, 39]. While researchers try to overcome this limitation by for instance employing prospect theory [40] and quantal response [41]; the problem remains an active area of research. Furthermore, it should be noted that security games often focus on one-to-one interactions between an attacker and a defender. However, general social interactions, like group decision making, are present in many threat scenarios. While multiplayer games have been investigated [95], they oftentimes do not go beyond three players. Furthermore, partially observable stochastic games form interesting methods to incorporate limited observation capabilities of agents [96, 97]. This is a more realistic assumption than the standard fully observable games. However, even partially observable stochastic games cannot reach the same level of human behavior modeling as agent-based models, as in-

cluding complex cognitive models is difficult in these models.

Attack trees suffer from similar limitations as do security games. Attack-defense trees can include higher-level interactions between the attacker and the defender. However, authors also note that they are "not suitable for including human interaction such as that of social engineering, because the attacker may combine different persuasion principles to different degrees, with different associated success probabilities" [42]. Countless examples of the incorporation of this social human behavior in agent-based models can be found in literature [98].

2.3.3. TRANSITION FROM NORMAL OPERATIONS TO THREAT

As many systems mostly operate following standard operations, the transition from these standard operations to the defense against an attack form an important aspect of security. In the TVC methodology, experts often consider this aspect but have no formal way of doing so.

This transition is also hard to model in security games as they assume the system to be in a state of attack. This transition can be modeled well by using agent-based models, as the standard operations are already modeled in the defined operational model M.

Like in security games, the transition from standard operations to the defense against an attack is hard to model for attack trees. They are defined to model a specific threat and therefore struggle with representing this transitional phase. As time can explicitly be taken into account by agent-based simulation models, this transition can be modeled and investigated.

2.3.4. INCLUSION OF SPATIO-TEMPORAL ASPECTS

Security games struggle with incorporating spatio-temporal elements into their models. These spatio-temporal elements, like the structures of buildings and the distribution of people in a shopping mall over time, can have a significant impact on the consequence of an attack. Some recent work in security games aims to incorporate spatial elements by using deep learning on images of forests [43]. However, it is unclear if this can also be used in other domains.

Similarly, attack trees struggle with the incorporation of spatio-temporal elements. The concepts of time and space are not intuitively represented in an attack tree, and therefore this method cannot easily include these elements in the risk assessment. Agent-based modeling allows for intuitive incorporation for both space and time and therefore allows for a potentially more accurate risk assessment.

2.3.5. QUALITY OF ASSESSMENT

The quality of assessment refers to the accuracy of the risk assessment that each of the methodologies produce. It is often stated that it is hard to validate risk assessments [99], but some high-level remarks are relevant here.

The TVC method heavily relies on basic analytic tools and security experts, leading to possibly inaccurate estimates. Cox provides an extensive overview of the different limitations of the TVC methodology [18]. The TVC methodology estimates risks by multiplying threat likelihood, vulnerability and consequence. However, basic probability theory states that this is only allowed if these values are completely independent. Dependen-

cies are certainly present between these risk components, and the TVC methodology, therefore, violates this rule. The use of Monte Carlo simulations to estimate conditional risks directly in the AbSRiM approach overcomes this limitation of inter-dependencies between vulnerability and consequence. Whereas dependencies between threat likelihood and conditional risks remain in AbSRiM.

The three methodologies generate results based on validated computational models, and indeed security games and attack trees were shown to be useful in practice. AbSRiM has the potential to overcome the limitations mentioned above and lead to better estimates but have to show usefulness in a wider variety of applications.

2.3.6. AVAILABILITY OF TOOLS

Once an attack tree is defined, results can be obtained with relative ease. Researchers have developed an extensive tool-set to automate the risk estimation process [100]. The same holds for security games. While many of these security games are proven to be NP-hard, researchers have developed fast algorithms for both approximations and exact solutions [15]. Contrary to AbSRiM, results for attack trees and security games have to be obtained only once and can be interpreted fast. In AbSRiM a time consuming and extensive sensitivity analysis has to be performed.

2.3.7. EASE OF ASSESSMENT

A major advantage of the TVC methodology is that it can be performed with relative ease. No model needs to be defined and so results can be obtained fast. As mentioned before, this is not the case with AbSRiM, as defining agent-based models is a time-consuming process. Lastly, security games and attack trees also require the definition of models, but they are easier to define than agent-based models. This allows for an easier risk assessment and management than in AbSRiM.

2.4. CONCLUSION & FUTURE WORK

This chapter introduced AbSRiM, a novel agent-based security risk management approach for airport operations. The approach contains four main steps: scope selection, agent-based model definition, risk assessment, and risk management. AbSRiM is based on traditional security risk management methodologies but uses agent-based modeling as the main paradigm to assess security risks. The effectiveness of the approach will be shown in Chapters 3 and 4 by applying it to different case studies.

AbSRiM provides a promising way to include important elements, such as human aspects and spatio-temporal aspects, in the assessment of risk. However, AbSRiM requires an extensive modeling effort and requires a lot of input from domain experts to be effective. This problem will be addressed in Chapters 6 and 7.

More research is needed to better identify the strengths and weaknesses of AbSRiM in different case studies. For instance, AbSRiM can be applied to more threat scenarios related to airport operations, and different domains, such as shopping malls and stadiums. Finally, the automatic identification of threat scenarios using machine learning techniques can be investigated in more detail. This technique can potentially be used to complement the threats that security experts identify.

3

SECURITY OPERATOR BEHAVIOR

As mentioned in Chapter 2, existing risk assessment methodologies struggle with accounting for human behavior. In this chapter, we apply the AbSRiM approach, as proposed in Chapter 2, to a case study on security operator behavior. We investigate how the decision-making and performance of human security operators can be taken into account while assessing vulnerability at an airport security checkpoint. To this end, we design an agent-based model, in which the performance of security operators is modeled using a functional state model, while decision making is modeled using decision field theory.

This chapter is currently under review at the Transportation Research Interdisciplinary Perspectives journal.

3.1. INTRODUCTION

Despite enormous investments in airport security, terrorists have been able to find and exploit vulnerabilities at security checkpoints. In the years after 9/11, aviation has been targeted by several bombing attempts [101–103], such as the shoe bomber [1]. Each of those attempts exploited new vulnerabilities and bypassed the security checkpoint successfully. It is only after such an attempt that new regulations and procedures are developed to address the exploited weakness in the security checkpoint. This reactive approach leaves airports vulnerable to innovative attackers. This problem is well recognized within the scientific literature, but developing a method that accurately assesses all vulnerabilities in a security checkpoint is a challenging task.

The security checkpoint is operated by security operators that constantly have to perform cognitive tasks, such as detecting illegal items on an X-ray image [104]. These operators also continuously have to make decisions, such as the decision to confiscate a potential weapon or not. Empirical research has shown that security operators do not necessarily follow protocol, but regularly bend and break the rules [105–108]. They commonly ignore potential threats and alarms are often processed as false. Furthermore, the performance of security operators is dependent on a variety of factors, of which cognitive task demands and personality are two examples. These human factors affect the performance of the checkpoint as a whole and additional vulnerabilities may emerge from their behavior. Therefore any method that aims to systematically identify all vulnerabilities in a security checkpoint should include these cognitive aspects in the analysis.

The objective of this chapter therefore is to understand how the decision-making and performance of human operators influence vulnerability at airport security checkpoints. To this end, we follow the AbSRiM approach of Chapter 2 to identify and quantify the vulnerabilities of two typical airport security checkpoint setups. The contribution of this chapter is twofold. First, we define a novel agent-based model to assess vulnerability, in which we specify security operators' behavior by combining two different cognitive models. The performance of security operators on different tasks in the checkpoint is modeled using the functional state model [109], and their decision-making process is modeled using decision field theory [110]. The developed model can also easily be adapted to test future concepts of security checkpoints, such as X-ray operators working remotely. These types of experiments are hard to perform directly at airports, as it may interrupt security operations. Secondly, by performing experiments with the model, we generate new insights with respect to vulnerabilities at the security checkpoint. Three types of experiments are performed: experiments related to operator performance, experiments related to operator decision making, and experiments related to different airport security checkpoint setups.

This chapter is structured as follows. First, related literature about human performance and decision making is reviewed in Section 3.2. Then, the agent-based model that we developed for this chapter is described in Section 3.3 and calibrated in Section 3.4. Three experiments were performed with the model and are described in Section 3.5. The first experiment is used to understand the influence of security operator performance on vulnerability, and the second for understanding the influence of security operator decision making on vulnerability. The third experiment is then used to investigate the effect of using different security checkpoint setups on vulnerability. Finally, the chapter is con-

cluded in Section 3.6.

3.2. RELATED WORK

Different studies have shown that the performance of security operators is not always optimal and that it is common for them to bend and break the rules [105–108, 111].

The performance of humans is dependent on a variety of factors, of which cognitive demands of a task and personality are two examples [112, 113]. Several computational models have been proposed in literature to model the performance of humans. Many of these models have a specific focus on aspect, like situation awareness [114]. The Functional State Model is a dynamic performance model that describes the performance of an agent as a function of task complexity, the state of the agent and its characteristics [109]. The model incorporates a large set of different factors, such as stress, exhaustion and situation awareness. The model was validated by empirical experiments with human operators from defense. This is also the model that we use in this chapter to model human performance, as it aims to incorporate a diverse set of factors.

Human decision making has been a long-studied field, and an extensive overview of modeling human decision making can be found here [115, 116]. Two main streams can be distinguished when modeling human decision making: bounded and unbounded rationality [116]. In bounded rationality, decisions are made within a set of human constraints such as limited information or processing speed of the brain, while in unbounded rationality these constraints are not present. In this chapter, we also use the bounded rationality paradigm. Within the bounded rationality paradigm, several types of models are developed in literature: linear decision making models [116], machine learning approaches [117] and diffusion models [110, 118]. We focus on probabilistic decision making, which is shown to be well capable of account for human irrationality. Important models in this area are the decision field theory model [110] and the Ratcliff diffusion model [118]. We used the decision field theory model in this chapter to model decision making of security operators, as it has strong empirical support and is famous for its ability to reproduce many known irrationalities in human decision making.

Only a few works exist that aim to model the behavior of security operators [119, 120]. This research models the effects of human factors on the performance of the security system by using a fuzzy inference system. However, their system is mostly based on expert opinions. This chapter focuses on more detailed cognitive models of human security operators, and how they can be used to estimate vulnerability. Furthermore, we explicitly represent interactions between agents (security operators and attackers), and important security devices, such as body scanners.

3.3. MODELLING THE SECURITY CHECKPOINT

This section describes the agent-based model that was developed to assess vulnerabilities at an airport security checkpoint, while focusing on human performance and decision making. The specification of the environment is discussed in Section 3.3.1, and the different types of agents are discussed in Section 3.3.2.

3.3.1. ENVIRONMENT

The environment of the model contains four different objects: luggage, weapons, sensors and equipment. *Luggage* has a *complexity level* that influences the task complexity of operators that interact with this luggage. The complexity level can either be *high* or *low*. Furthermore, luggage is owned by a passenger and may contain *explosive traces* (represented as a Boolean value) and/or a weapon.

Then, the *weapon* object conceptualizes a weapon that an attacker agent aims to bring past the security checkpoint. A weapon is of a certain *type*. The type of weapon can, for instance, be a ceramic knife or an explosive liquid. Similar to luggage, a weapon can contain *explosive traces*. Furthermore, a weapon has a *perceived risk*, r_{perc}, that indicates to which extent a security operator perceives objects that closely resemble the weapon as a risk to the airport. For instance, explosive liquids resemble a bottle of water and are therefore perceived as a low risk. Perceived risk is formalized as a real number between 0 and 1. While r_{perc} is different for each operator, we assumed that it is the same for everyone, and, therefore, included it as part of the weapon. Finally, a weapon can be on the body of an attacker, or in the luggage of an attacker. A full list of weapon types and their corresponding parameters is shown in Section 3.4.

Different sensors were defined in the model: X-ray sensor, Walk-through metal detector (WTMD), explosive trace detector (ETD) and body scanner. Each sensor has a probability $p_{detect}^{sensor}(weapon\_type)$ to detect a specific weapon type, called base detection probability. This parameter is calibrated and shown in Section 3.4. Based on this detection probability, the sensor either detects or does not detect a weapon when presented, which can then be observed by operators. The X-ray sensor is an exception to this standard, as this sensor only allows operators to observe the luggage that is currently sensed by the sensor. In this case, the likelihood of detection is determined by the skill of the x-ray operator.

Finally, two types of equipment were defined in the model: *queue separators* and *X-ray systems*. Queue separators are used to guide passengers to the security checkpoint, while the X-ray system moves luggage forward through an X-ray sensor.

3.3.2. AGENTS

Three different agent types were defined: passengers, attackers, and operators. Each of these agents are human agents and are discussed in more detail below.

PASSENGERS AND ATTACKER AGENTS

Passengers and the attacker were defined similarly. They both do not exhibit sophisticated strategical behavior. Passengers carry luggage that they bring to the security checkpoint. Furthermore, passengers could carry *explosive traces* and they can own a weapon. This weapon can, as defined above, either be on the body of the passenger or in its luggage. We refer to a passenger that owns a weapon as an attacker, and as a passenger otherwise.

SECURITY OPERATORS AGENTS

A set of security operators that execute activities at the security checkpoint were defined: patdown operator, ETD check operator, luggage check operator, and X-ray operator. This

section discusses the definition of the X-ray operator, as the other operator types are defined similarly.

Airport security is largely defined by regulations and guidelines defined by different regulatory institutes. For instance, the European Union has regulations for its members [121, 122], the United States has the Aviation and Transportation Security Act [123], and the ICAO has a security manual [124].

Following these regulations and guidelines, each of these operators executes a fixed set of tasks and decisions. An X-ray operator inspects output generated by the X-ray machine and determines if there is a potentially illegal item. When this is the case, (s)he has to inform the luggage check operator, who then searches the luggage. Security operators do not necessarily follow this protocol, but regularly bend and break the rules. They commonly ignore potential threats and alarms are often processed as false [105–108, 111]. Furthermore, humans cannot continuously perform optimally. It is dependent on a variety of factors, of which cognitive demands of a task and personality are two examples [112, 113].

The performance of security operators on different tasks in the checkpoint is modeled using the functional state model [109], and their decision-making process is modeled using decision field theory [110]. In the case of the X-ray operator, the modeled task is inspecting output generated by the X-ray machine, while the modeled decision is that of informing or not informing the luggage check operator.

The functional state model, the decision field theory model, and their integration is discussed below.

Functional State Model To model the performance of security operators, the functional state model was selected [109]. While the model contains a set of 37 parameters, we only discuss the most important parameters here. For the other parameters, the reader is referred to the work of Bosse et al. [109]. The input for the model is the *task level* (*TL*), which is dependent on the *skill level* (*SL*) of the operator and the *task complexity* (*TC*) of the task at hand. For an X-ray operator, the task complexity represents how complex the luggage (s)he is currently investigating is. This dependency is modeled as follows:

$$TL(t) = \frac{TC(t)}{SL} \tag{3.1}$$

The output of the model is the *performance quality* (*PQ*) of the agent, indicating how well the operator is performing. A *PQ* of 1 corresponds to the baseline performance of an agent, while values lower than 1 correspond to performances that are worse than this baseline and values higher than 1 correspond to performances better than baseline. No theoretical bounds of *PQ* were provided in the of Bosse et al. [109]. However, typical *PQ* values in our simulation results are in the range of 0.5 and 1.5.

PQ is dependent on two factors: provided effort (*PE*) and task level (*TL*):

$$PQ(t) = \frac{PE(t)}{TL(t)} \tag{3.2}$$

PE is determined by the generated effort (*GE*) of the agent, recovery effort (*RE*) and noise effort (*NE*). The latter two parameters correspond to the ability of humans to de-

creases exhaustion, and the effort the human has to contribute to the noise in the environment respectively.

$$PE(t) = GE(t) - RE(t) - NE(t) \tag{3.3}$$

GE is the most important contributor to PE, and is ultimately defined by effort motivation (*EM*), among many other parameters. We refer to equations 2 and 4 in the work of Bosse et al. for a complete deduction of *GE* [109, 125].

Effort motivation is based on the current task level and the difference between experienced pressure (*EP*) and optimal experienced pressure (*OEP*). *EP* is similar to a person's stress level, while *OEP* determines how well a person can cope with a high *EP*. Finally, *EP* is, among other terms, related to generating effort above and below a critical point. The critical point is the amount of effort someone can generate without becoming exhausted. For an X-ray operator, *PQ* is reflected in the likelihood (s)he observes a weapon from the observations of the X-ray sensor. This is modeled as follows.

$$p_{detect}^{operator}(weapon\_type) = max(0, 1 - \frac{1 - p_{detect}^{x-ray}(weapon\_type)}{k \cdot PQ}) \tag{3.4}$$

The value $p_{detect}^{x-ray}(weapon\_type)$ corresponds to the base likelihood that a specific type of weapon is detected by an X-ray operator, which is calibrated in Section 3.4.2. The value $1 - p_{detect}^{x-ray}(weapon\_type)$ corresponds to the base probability of not detecting the weapon: the base false-negative rate. When performing well (i.e. a high *PQ*), X-ray operator improves on this base false-negative rate, and vice verse. We model this by dividing the base false-negative rate by the performance quality and a scaling factor k. The underlying assumption here is that the false-negative rate linearly decreases with increasing *PQ*. This false-negative rate is then transformed back to a detection probability by subtracting it from 1. To ensure that the value falls between 0 and 1, we take the maximum of 0 and the value obtained above.

The other operators at the security checkpoint use this performance model to execute the patdown activity, search luggage and perform an ETD test. The value of k, and other related parameters of the functional state model are calibrated in Section 3.4.

Two different personality types are introduced based on the work of Bosse et al.: personality I and personality II [109]. Bosse et al. extensively experimented with these two personality types and performed an in-depth analysis of their behavior. Type I has a relatively high *OEP*, meaning that it can cope well with high *EP* levels, while type II does not. This allows the first personality type to perform better under high pressure. We experiment with these personality types in our analysis.

Decision Field Theory The decision-making process of the security operators was modeled based on the work of Busemeyer and Townsend [110]. The decision-making process in this model is an iterative process in which the operator constantly updates their *preferences* until the preference for one of the *options* exceeds a *decision threshold* value. This threshold value is one of the inputs of the model and its magnitude is related to the effort an agent spends on a decision. The higher the threshold value, the more time and energy the security operator needs to reach it.

During each iteration, the agent focuses on one of his *goals*. The selection of this goal is a random process, but the likelihood of the agent focusing on a goal depends on the *attention weight*. Once the attention of the agent is focused on one of his goals, the agent's preferences are updated based on the agents *beliefs* about how each of the options helps him in achieving the goal (s)he currently focuses on. The magnitude with which the preference for each of the goals is updated is known as *valence*. This valence is defined for each combination of goals and options.

Finally, the decision-making process is influenced by the agent's initial beliefs. This *initial preference* is the preference the agent has for each outcome before the decision process starts. An overview of this process is shown in the bottom part of Figure 3.1.

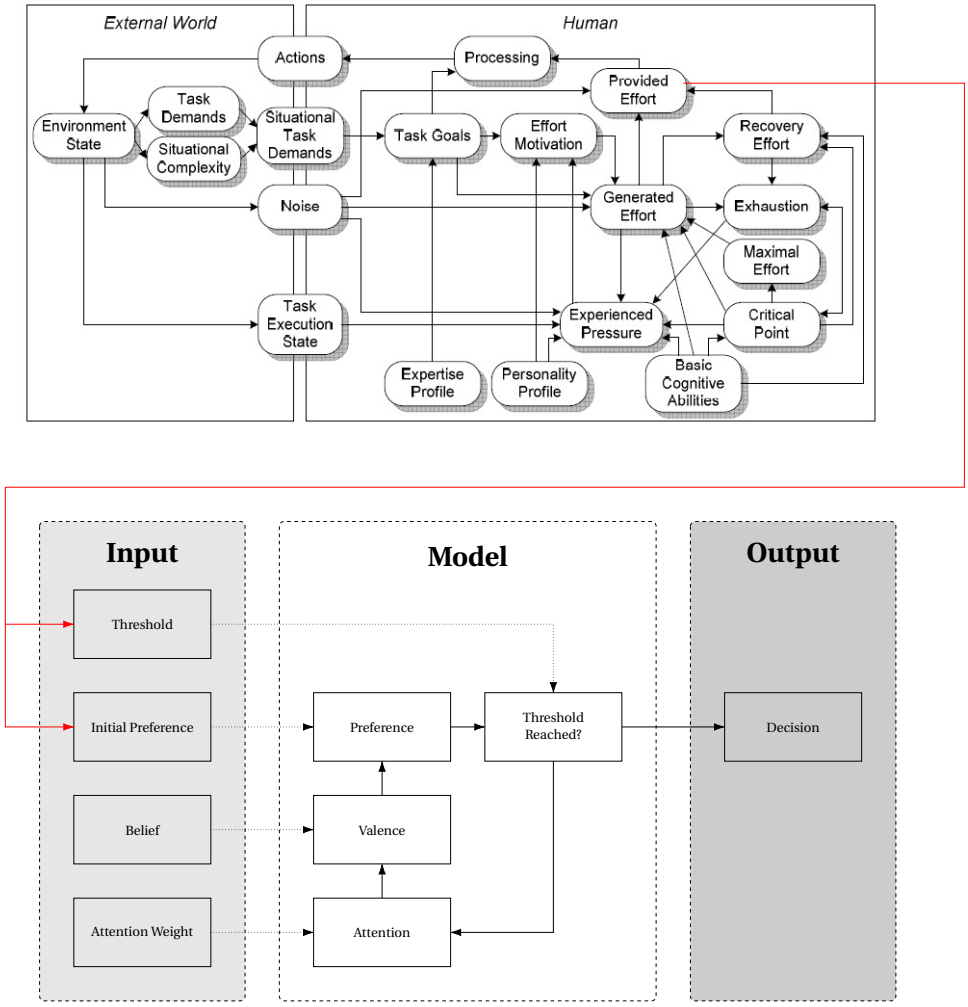


Figure 3.1: An overview of the functional state model [109] (top) and the decision field theory model [110] (bottom) used in this work. The integration between the two models is shown as well.

For an X-ray operator, one decision is identified. If the X-ray operator observes a potential weapon (see also Section 3.3.2), (s)he has to decide if the luggage requires a search from the luggage check operator. The options for the X-ray operator are *inform* or *ignore*. Furthermore, three goals are defined for the X-ray operator based on existing literature [126, 127].

- Accuracy
  - The operator wants to do their work as well and accurate possible. The importance of this goal may be dependent on pressure within the organizations or the agent's standards.
- Speed
  - The operator wants to do their job as fast as possible. The importance of this goal may be due to pressure within the organization to reach a certain throughput or the security operator wanting to minimize effort.
- Perceived Risk
  - It is the job of the security operator to minimize the risk of an attack. Perceived risk represents the beliefs an agent has about the potential consequences of the observed prohibited item. The importance of this goal may be dependent on the agent's beliefs about the likelihood of an attack and his risk aversion.

Both luggage check operators and physical check operators use this decision mechanism to determine if a passenger requires secondary screening when an illegal object was found. The ETD operator makes the same decision when explosive traces were observed. The other related parameters of this model are calibrated in Section 3.4.

Integration of models We integrated the models by relating parameters of the functional state model to the decision field theory model. The relation between the models is shown in Figure 3.1.

The *decision threshold* was set to be equal to the provided effort (*PE*) as defined in the functional state model. Provided effort denotes the effort that is contributed to the task by the agent. This relation means that the higher the provided effort, the more effort the agent wants to invest in making an accurate decision. This is based on findings by Busemeyer and Townsend [110]. Furthermore, we assumed that the *initial preference* of the X-Ray operator is according to regulations present at the security checkpoint, meaning that there is a strong initial preference to request a luggage check if needed. The next section describes how these parameters are calibrated.

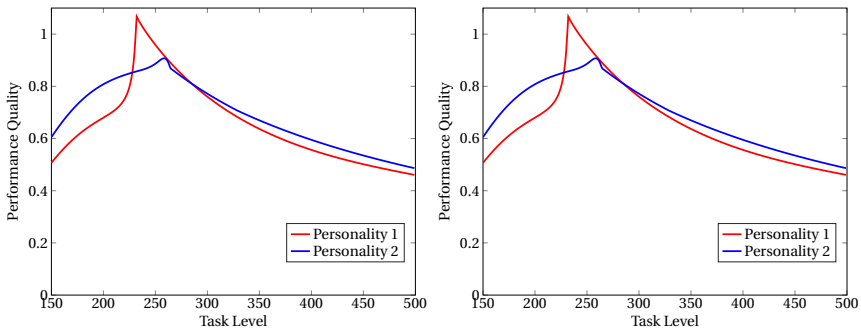
3.4. MODEL SENSITIVITY AND CALIBRATION

In this section, the sensitivity of the functional state model and the decision field theory model is discussed. Furthermore, it is described how the overall model was calibrated. Different parameters had to be calibrated: parameters related to weapons, sensors, airport configurations, and operators. These are discussed in detail below.

3.4.1. SENSITIVITY ANALYSIS

We performed sensitivity analysis of both the functional state model and the decision field theory model. Figure 3.2(a) shows how different task levels affect the performance quality and the provided effort in the functional state model. Results were obtained after the task level was kept constant for 20 seconds. At this point, the performance quality converged to an equilibrium value for any task level. From the figure, it becomes clear that both personality types have the highest performance quality around a task level of 250. The peak performance of personality type I is at a task level 230. At this point, it outperforms personality type II by 24%.

At task levels lower than 225 the performance quality of both personalities rapidly drops. This is mainly due to a lack of provided effort as can be seen in Figure 3.2(b). In this range, personality type II outperforms personality type I by 20%. At task levels above 275, the performance of both personalities exponentially decreases. The provided effort of both agents stays approximately stable around 230, meaning that the agent cannot provide more effort. At even higher task levels the performance quality drops.



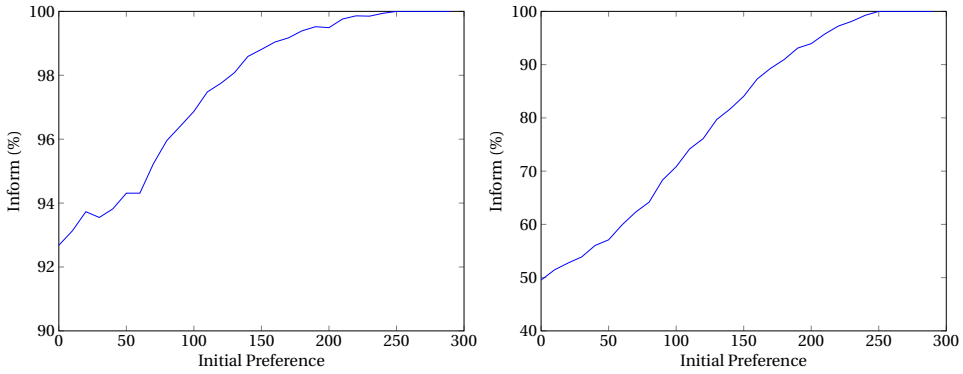
(a) Performance quality for different task levels.

(b) Provided effort for different task levels.

Figure 3.2: The effect of changing task levels on performance quality and provided effort.

We investigated the sensitivity of the decision field theory model as well. The values in Table 3.3 were used with $c = 30$, but the initial preference for the inform decision was varied. The decision threshold was set to a uniform random value between 70 and 250, which is the range of provided effort values as observed above. Two scenarios were investigated: 1) a weapon with a perceived risk of 0 was observed, and 2) a weapon with a perceived risk of 1 was observed. A total of 1000 simulations were performed for each data point.

Figure 3.3 shows how different initial preferences for the inform decision influence the decision of the X-ray operator. Both graphs have the same general shape. The choice to inform the luggage-check operator increases from a baseline value to 100% when the initial preference becomes 250. At initial preferences above 250, the operator chooses to inform the luggage-check operator 100% of the time. This is because the initial preference already exceeded the threshold value. The range of values for both scenarios is different. In scenario 1, the luggage-check operator is informed 93% of the time without



(a) $r_{perc} = 1$.

(b) $r_{perc} = 0$.

Figure 3.3: The effect of changing initial preferences on the decision of X-ray operators to inform luggage-check operators when an illegal item was observed.

any initial preference, while this is only 50% in scenario 2. In scenario 1 the dominant decision is to inform, as two out of three goals favor this decision. As the perceived risk is zero in scenario 2, there are effectively only two goals: accuracy and speed. Neither of these goals dominates the other, leading to a baseline of 50% inform decisions.

3.4.2. WEAPON AND SENSOR CALIBRATION

Table 3.1 shows the different weapon types used in this chapter. For each of the weapon types, it is indicated if they contain explosive traces and their perceived risk. It should be noted that the perceived risk represents the risk that is perceived by operators for objects that resemble the weapon. For instance, if a bomb is not recognized as such (like explosive liquids), the perceived risk is much lower. Bombs and fire-arms were assumed to have the highest perceived risks, while liquids were not perceived as a large risk, as operators continuously confiscate water bottles. Knives were perceived as a larger risk, but they are still commonly observed.

Table 3.2 shows the different detection probabilities for weapon-sensor combinations and weapon-activity combinations. The values for X-ray performance is based on literature [128], as well as the explosive bulk detection probabilities for body scanners [129]. No data could be found on how security operators perform on searching luggage and patdowns. These values were therefore based on assumptions.

3.4.3. AIRPORT CONFIGURATIONS CALIBRATION

Two different airport configurations were defined based on IATA documentation [104]: the regional airport and the international airport. There are two main differences between these configurations. First, the regional airport uses a WTMD whereas the international airport uses a body scanner. This choice of equipment impacts the detection rates of weapons hidden on the body of the attacker. The second difference is the communication between the X-ray operator and the luggage check operator. At the regional airport, there is the possibility to communicate directly, while at the international airport

Table 3.1: The different types of weapons with a description, an indication if explosive traces are present and their perceived risk.

Type	**Explosive Traces**	r_{perc}	**Description**
Explosive Bulk	Y	1	An improvised explosive device.
Explosive Liquid	Y	0.1	Liquid explosives are often not directly recognized as a bomb.
Explosive Powder	Y	0.2	Explosives in powder form are commonly not recognized as a bomb.
Gun	N	1	A standard handgun.
Knife	N	0.3	A small knife.
Ceramic Knife	N	0.3	A small knife without metal.

Table 3.2: The detection probability $p_{detect}(weapon)$ for each sensor and activity.

	Expl. Bulk	**Expl. Liquids**	**Expl. Powder**	**Gun**	**Knife**	**Ceramic Knife**
WTMD	*0.00*	*0.00*	*0.00*	*1.00*	*1.00*	*0.00*
Body Scanner	*0.56*	*1.00*	*0.00*	*1.00*	*1.00*	*1.00*
X-Ray Activity	*0.735*	*0.645*	*0.645*	*0.875*	*0.675*	*0.675*
Lugg. Search Activity	*0.90*	*0.90*	*0.90*	*0.90*	*0.90*	*0.90*
Pat Down Activity	*0.90*	*0.90*	*0.90*	*0.90*	*0.90*	*0.90*

the luggage check operator is not in direct contact with the X-ray operator. The luggage check operator has to perform an X-ray himself/herself to determine where the weapon can be found.

3.4.4. OPERATOR PERFORMANCE CALIBRATION

The task complexity (*TC*) of operators for the different tasks that they perform are calibrated in this section. To this end, we assumed three different skill levels (*SL*) for operators: 0.8 (low), 0.9 (medium), and 1.0 (high). These levels correspond to a realistic variation in skill between agents in the functional state model, based on experimentation with the functional state model and values found in literature [128]. Furthermore, we assumed a base task level (TL_{base}) of 150, corresponding to a performance quality (*PQ*) of around 0.5. The base task level is the task level when the agent is not performing its activity and is based on the work of Bosse et al. [109]. An X-ray operator has about one second to identify potentially prohibited items in luggage and research has shown that the number of false negatives increases when the images become harder to interpret [128]. Based on the same work, we assumed that the performance of an X-ray oper-

ator decreases with 5.47% for more complex luggage.

The scaling factor k was calibrated as follows. The value for $k \times PQ$ should be equal to 1, as an average operator performs according to the base detection probability. We have performed 1000 simulations for each skill level, personality type, and task level to determine the mean performance quality of operators, and found that it corresponds to 0.59. We use this mean performance quality to finally find k to be equal to 1.68.

3.4.5. OPERATOR DECISION CALIBRATION

For operator decision making, the following parameters had to be calibrated: initial preference, decision threshold, and valence. We used attention weight as a parameter to experiment with. The decision threshold is equal to the provided effort *PE* as suggested in the work of Busemeyer and Townsend [110]. The valences and initial preferences are shown in Table 3.3. The initial preference for the *inform* decision was assumed to be the decision threshold of the agent multiplied with a constant of $c_{pref} = 0.95$. This indicates that the X-ray operator has a strong preference to follow rules and regulations. Furthermore, the valences related to speed and accuracy are $\pm c$ for both options. We chose $c = 30$ such that the mean decision time of X-ray operators corresponds to times reported in literature [128]. Finally, the valences for the perceived risk goal were made dependent on the perceived risk of the observed weapon. We assumed this to be a multiplication between c and the perceived risk. The parameters of the other operators were determined similarly.

Table 3.3: Calibration of the decision parameters for the X-ray operator.

	Initial Pref.	**Accuracy**	**Speed**	**Perc. Risk**
Inform	$c_{pref} \cdot DT$	c	$-c$	$c \cdot r_{perc}$
Ignore	0.0	$-c$	c	$-c \cdot r_{perc}$

3.5. EXPERIMENTS AND RESULTS

We performed experiments with the model to assess vulnerabilities at different security checkpoint setups. The setup of the experiments are discussed first, followed by a discussion of results.

3.5.1. EXPERIMENTAL SETUP

The model was implemented in the AATOM simulator, which is a Java-based airport terminal operations simulator [130]. It is agent-based and contains several calibrated presets and templates of basic airport terminal components that can readily be used. No other simulator that we know of contains such a combination of agent-based modeling and pre-calibrated airport-specific components.

As specified in the model description, four operator agents were defined in the model: patdown operator, ETD check operator, luggage check operator, and X-ray operator. We used a single security lane setup, and passengers were generated for a single flight with up to 100 seats. A single attacker was introduced among the passengers that went through the security checkpoint.

The following parameters were varied in the execution of the experiments.

- *Attacker parameters*
  - *Weapon.* The weapon the attacker uses is one of the weapons shown in Table 3.1.
  - *Weapon Location.* The attacker has the option to hide the weapons on his body or in his luggage.
- *Checkpoint Configuration.* The checkpoint configuration is either the regional airport or the international airport.
- *Operator parameters*
  - *Skill Level.* The skill level of the agents is either 0.8 (low), 0.9 (medium) or 1.0 (high).
  - *Personality Type.* The agents either have personality I or II, based on the work of Bosse et al. [109].
  - *Attention Weights.* The attention weight for each goal is set to 0.33 (low), 0.5 (medium) or 0.67 (high). The weights are normalized so that they add up to one after they are selected.

A total of $N = 15,000$ simulation runs were performed, while using a uniform random assignment of the above parameter values. After assigning parameter values in a simulation, they do not change until the next simulation run. Furthermore, agents are assumed to not learn during a simulation run. Finally, a simulation finishes when all passengers have passed through the security checkpoint.

3.5.2. RESULTS

The results are discussed as follows. We define vulnerability as the proportion of attackers that moved past the security checkpoint with their weapon. These attackers did not receive secondary screening and their weapon was not confiscated. We first show how the skill level and personality type of security operators influence their performance. Then, we show how different attention weights of the decision field theory model influence the decisions made by the operators. Both these results are an indication of the vulnerability of the security checkpoint, as both performance and decision making directly influence the number of secondary screenings and weapon confiscations. Finally, an overall vulnerability assessment of the different checkpoint configurations is conducted and a discussion is provided.

PERFORMANCE OF OPERATORS

The performance quality of X-ray operators and luggage check operators can be found in Figure 3.4. The performance quality of security operators is directly related to the vulnerability of the security checkpoint. A low performance quality of any of the operators leads to a higher vulnerability, as items are detected with a lower probability. As can be seen in the figure, *PQ* increased with skill level for X-ray operators. The agents with the

highest skill level (of 1) outperformed the agents with the lowest skill level (of 0.8) with 5.2%.

Different results were observed for luggage check operators. The operators with the highest skill level were outperformed by the agents with the lowest skill level by 4.0%. This also seems counter-intuitive but can be explained from the mechanisms of the functional state model. If the task level becomes too low, the performance quality drops, as skilled agents are not motivated enough to generate effort. For operators with a lower skill level, the task is more challenging and they are more motivated to put in the effort. This lead to the counter-intuitive result that the most skilled agents were not top performers on this relatively simple task. This result may seem counter-intuitive but is caused by the fact that agents perform (relatively) simple tasks and find it hard to motivate themselves to put in enough effort. Following the functional state model, operators with a higher skill level, experience a lower task level for the same task as their lower-skilled counterparts. Generated effort is, among other parameters, based on the motivation of the operator, which in turn is partially determined by the task level. Because the task level is lower for higher-skilled operators, the effort motivation decreases, which decreases the provided effort. Our simulation results have shown that this negative effect on performance quality of decreased motivation is larger than that of an increased task level for lower-skilled operators. Section 3.3.2 provided a discussion of the different variables in the functional state model.

These results are not unique to the Functional State Model and our model. Hackman and Oldham proposed a so-called Motivating Potential Score [131] which is a framework that is widely used in literature. MPS is, among other terms, composed of skill variety. This is strongly related to what we have defined as skill level in this chapter and explains the connection between motivation and skill level. Furthermore, jobs with a high MPS, have a positive effect on motivation, performance and job satisfaction [132]. This then

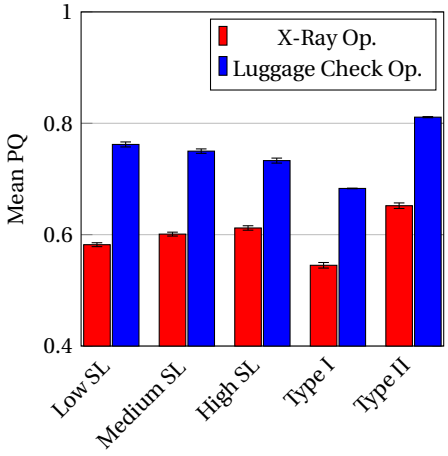


Figure 3.4: The mean performance quality (and their 95% confidence intervals) for the different skill levels and personality types.

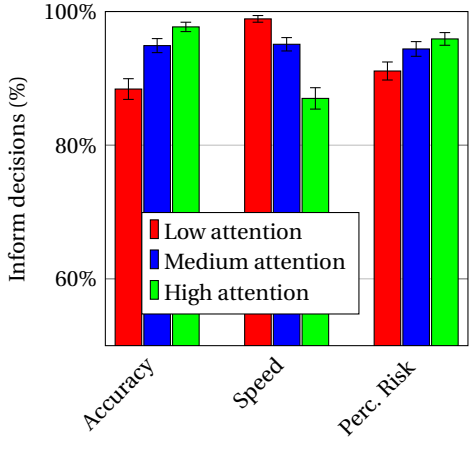


Figure 3.5: Percentage of correct *inform* decisions (and their 95% confidence intervals) for the different attention weights.

relates motivation to task performance.

The differences between the performance quality of analyzing X-ray images and checking luggage can be explained as followed. Analyzing X-ray images is a difficult cognitive task for humans. A large number of stimuli have to be processed and illegal items have to be identified at a high speed. An operator performing a luggage check has more time to execute the task at hand. Typically, they take around 90 seconds, while X-ray operators only have a few seconds for their task. This allows the luggage check operators to generate more effort and therefore reaching higher performance quality.

Furthermore, agents that cannot cope with pressure well (type II) outperform agents that can better cope with pressure (type I) by 20%. The results of the different personality types shown in this figure are an aggregate of all skill levels. Personality type II has a relatively low *OEP*, which is closer to the actual experienced pressure than the high *OEP* of personality type I. The difference between these values determines the effect on effort motivation. A low difference leads to a low reduction in effort motivation, while a high difference leads to a high reduction of effort motivation. As mentioned before, a lower effort motivation finally leads to a lower performance quality.

DECISION MAKING OF OPERATORS

We analyze the decision-making process of X-ray operators. When an X-ray operator detects a potential weapon, the agent has two options. The first option is to ignore that the potential weapon was observed, while the second option is to inform the luggage check operator. When a potential weapon was detected, luggage check operators were informed correctly 93.7% of the time on average. This number varied based on the attention weights for each of the goals, as shown in Figure 3.5. Not searching luggage when it contains a weapon, directly increases the vulnerability of the system.

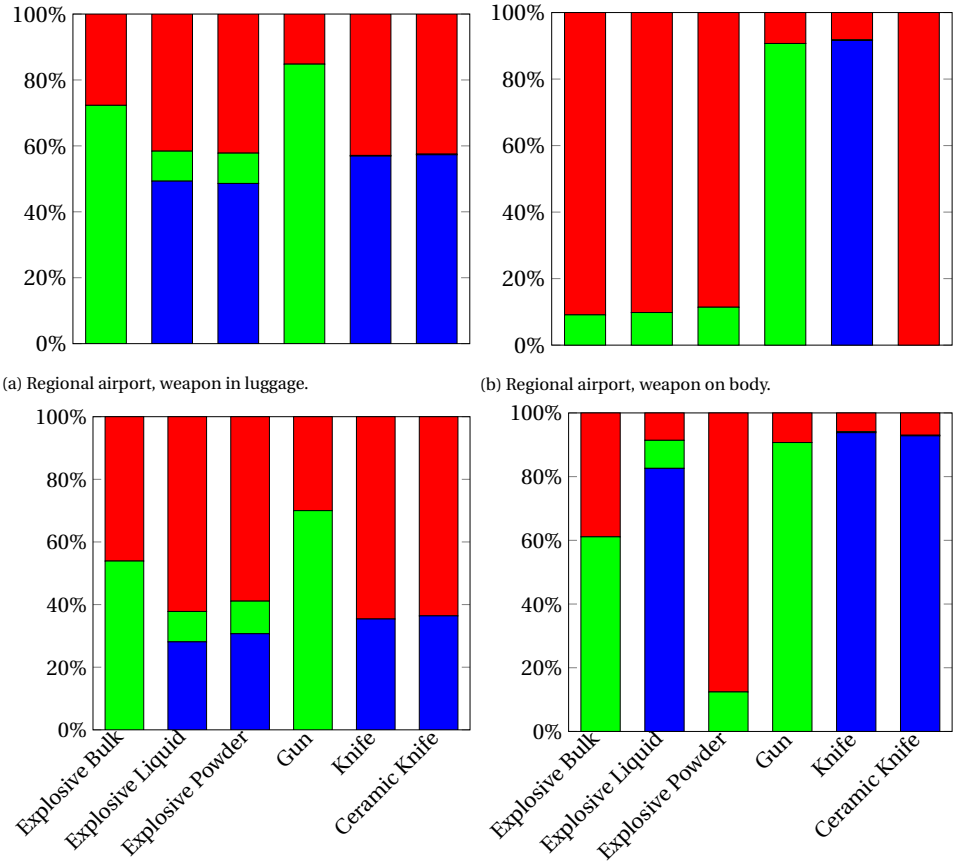
One of the reasons for an X-ray operator to not inform the luggage check operator is that it might not perceive the potential weapon as an actual weapon. For instance, liquid explosives might resemble a water bottle. While a water bottle is illegal according to checkpoint regulation, regulations are not always strictly enforced by security operators [105–108]. Not informing the luggage check operator then leads to faster processing of passengers, which is of enormous economic importance for airports.

From the figure, it becomes apparent that the attention weight for speed was the most dominant parameter in the inform decision. Varying this parameter from low (0.33) to high (0.67), lead to a 12% decrease in luggage searches. The second most important parameter is the attention weight for accuracy. Increasing this parameter from low to high, caused an 11% increase in luggage searches. The attention weight for risk was less dominant. An increase from low attention to high caused a 5.3% increase in luggage searches. This parameter was less influential, as many potential weapons are not perceived as a large risk by the operators. Speed and accuracy, on the other hand, played a more important role in the decision-making process. While not shown, results for decisions by other types of operators followed similar trends.

Jesus performed a questionnaire among security operators at a regional airport to determine how they make trade-offs between security and efficiency [133]. One of the main findings of his research was that operators could be classified into three categories: 1) passenger level of service operator, 2) security-focused operator, and 3) efficiency-focused operator. About 13% of the surveyed employees fell into the last category. These

employees mostly focused on improving the efficiency of checkpoint operations and barely on security.

While the results of Jesus are not readily comparable to our results, we do observe an interesting similarity between them. Both results indicate that some employees mostly focus on executing their work efficiently (i.e. high attention weight for speed), which then results in increased vulnerabilities.



(a) Regional airport, weapon in luggage.

(b) Regional airport, weapon on body.

(c) International airport, weapon in luggage.

(d) International airport, weapon on body.

Figure 3.6: The performance of the checkpoint setups for the different weapons defined in this work. Performance is shown in terms of vulnerability (weapon not confiscated and no secondary screening; red bar), percentage of secondary screening (with or without weapon confiscation; green bar), confiscated weapons (without secondary screening; blue bar).

DIFFERENT CHECKPOINT SETUPS

The performance of the security checkpoint for different weapons and locations are shown in Figure 3.6. In this figure, the distribution between three potential outcomes of a scenario are shown: *vulnerability* (weapon not confiscated and no secondary screening), *secondary screening* (regardless of weapon confiscation) and the situation in which

the *weapon* was *confiscated* while no secondary screening was conducted.

From this figure, it becomes clear that some weapons were never confiscated at the regional airport. These weapons cannot be detected by the equipment used to scan the passengers. None of the explosives smuggled on the body get detected by the WTMD and the same holds for ceramic knives. Explosives only got detected by a random ETD check, which lead to a secondary screening in 10.1% of the cases. Furthermore, knives can be taken through the checkpoint at the regional airport without large consequences. Most often, the knife got confiscated and the attacker could try again at a different time as the chances on a secondary screening were found to be almost zero. The regional airport performed best on detecting guns in luggage. These weapons were confiscated 84.8% of the time when they were located in the luggage (as compared to 70.0% in the international airport) and immediately lead to a secondary screening. This becomes 90.7% when the attacker carried the weapon on their body.

At the international airport, only one type of weapon remained undetected. Smuggling explosive powder through a body scanner had a success rate of 88.6%. The only measure against it was a random ETD check. Furthermore, liquid explosives and powders hidden in luggage were confiscated only 32-34% of the time. Even when these items were confiscated, the security operator did not necessarily recognize these items as bomb parts and allowed the attacker to move on. Bulk explosives, on the other hand, were detected in 50% of the cases and lead to immediate secondary screening. An attacker bringing a gun was very unsuccessful at the international airport. The attacker was most successful when locating the gun in their luggage, but this only had a success rate of 30%. Knives could best be brought hidden in the luggage as well. In that case, they were only confiscated 36% of the time and the chances of secondary screening were minimal. However, the potential impact of a knife past the security checkpoint is far more limited than that of other weapons investigated in this chapter.

The regional airport outperformed the international airport on checking luggage for all weapons. In the regional airport, 62.6% of the weapons in the luggage are confiscated, whereas in the configuration of the international airport this is only 42.6%. The main reason for this is the lack of communication in the configuration of the international airport. The X-ray operator flags luggage for a search, but the luggage check operator has to identify the weapon on the X-ray image himself. This extra step in the process caused a loss in performance of 32% and occurs solely because two officers independently had to recognize a weapon on an X-ray image instead of just one.

Relation with related work ABC News reported in 2015 that in 95 percent of trials undercover investigators were able to smuggle mock explosives or banned weapons through checkpoints [134]. Two years later, this percentage decreased to a still extremely high value of around 80 percent [135]. These problems do not only exist in the United States. The Telegraph reported late 2014 that airport security failed to detect half of the dangerous weapons at Frankfurt airport [136]. The vulnerabilities that we found in this chapter (see Figure 3.6) are close to these public reports. However, to the best of our knowledge, there is no public data available that evaluates the effectiveness of security checkpoints specifically for different weapon types, as we do in this chapter. While we have used all data that was available to calibrate the different capabilities of sensors to detect each

weapon type, more work is needed to validate our results.

3.5.3. DISCUSSION OF RESULTS

As mentioned in Section 3.4, the calibration of the model was based on a set of simplifying assumptions. These assumptions influenced the magnitude of the resulting vulnerabilities in different checkpoint configurations. The calibration of the model can be improved by performing field tests to determine different parameters. For instance, the performance of operators for searching luggage can be evaluated by providing security operators a set of luggage containing legal and illegal objects. Furthermore, perceived risks of objects can be evaluated for different operators using a similar method. Decision making of operators can be calibrated better by performing choice experiments, such as the one performed by Jesus [133].

While vulnerability estimates are inherently hard to validate, some researchers performed real-life experiments [92, 93]. This form of validation is a direction of further research for this chapter. Both calibration and validation of the model can still be improved, but the proposed model is still valuable for airport security practitioners, as it can be used to generate improved results when more data becomes available.

We did not consider all security mechanisms that are present in airports. For instance, intelligence agencies can detect attackers before they arrive at the security checkpoint. behavior detection officers [137, 138] are also capable of detecting suspicious behavior at the security checkpoint and perform secondary screenings based on that. Furthermore, more strategic attacker behavior in which the attacker chooses the right type of weapon for the checkpoint configuration can be considered as well.

Agent-based modeling is an important tool to better understand complex systems. Using our model, vulnerabilities caused by imperfect human decision making and performance were identified. Understanding how these vulnerabilities emerge enables airports and policymakers to improve their security policies and reduce vulnerabilities. Our model can be used to test future concepts of security checkpoints. For instance, when X-ray officers do their work remotely, our model can be adapted with relative ease to determine the performance of such a setup. These types of experiments cannot easily be performed at airports, as it may interrupt security operations. Furthermore, experiments with humans are known to be hard to perform due to the diversity of human behavior. Using our model, these experiments can be performed more easily. This can, for instance, be done by hiring operators with the right personality type and skill set, or by taking these aspects into account while planning operators.

Our agent-based approach is more time consuming to perform than most other vulnerability assessment methodologies and requires a large amount of data for calibration. Other vulnerability assessment methodologies form better alternatives in cases with a lack of time or data, but our approach is particularly suitable to investigate vulnerabilities in which human behavior plays a role. A more in-depth discussion about the advantages and disadvantages of the use of agent-based modeling is discussed in Chapter 2.

3.6. CONCLUSION

In this chapter, we investigated how the decision-making and performance of human operators can be taken into account while assessing vulnerability at airport security checkpoints. Following the AbSRiM approach of Chapter 2, we developed an agent-based model, in which the performance of these operators was modeled using the functional state model, while decision making was modeled using decision field theory.

Simulation results indicate that the highest skilled operators outperform their lowest skilled counterparts on analyzing X-ray images, but perform worse on both searching luggage and performing patdowns. This leads to similar differences in security checkpoint vulnerabilities as well. These skilled operators find their tasks too easy and are unable to motivate themselves to put in the required effort. Furthermore, the goals the operator focuses on during the decision-making process were found to influence vulnerability. A high focus on accuracy or perceived risk for the X-ray operator leads to an increase in luggage searches, and therefore reduced vulnerabilities. However, a high focus on speed leads to a decrease in luggage searches and therefore increased vulnerability. The developed model can be used to assess the effect of human behavior and decision making on the performance of current and future security checkpoint procedures, which is often impossible using real-live experiments. More work is needed to calibrate and validate the model and simulation results, but initial results are promising.

This chapter can be extended by investigating how other types of security measures (i.e., behavior detection officers) influence the vulnerability of the security checkpoint. Furthermore, the influence of the time that the attacker is generated can be investigated in future work. The vulnerability with respect to other threat scenarios (i.e., a bomb attack before the security checkpoint) can be investigated as well. In Chapter 4 we will investigate how security risks are related to airport efficiency metrics, such as queuing time at the security checkpoint. Finally, the model can be calibrated better by using classified data on sensor performance, operator performance and attacker behavior.

4

SECURITY AND EFFICIENCY

Both security and efficiency are important performance metrics of air transport systems. In this chapter, we extend the AbSRiM approach as proposed in Chapter 2 to additionally identify relationships between security risks and efficiency performance indicators. We apply the methodology to a case study that analyzes security regarding an Improvised Explosive Device (IED) attack. In addition, different commonly used efficiency performance indicators in the aviation domain, such as queuing time for passengers, and the relationships between security and efficiency is analyzed.

This chapter has previously been published in the Transportation Research Part C: Emerging Technologies journal (2019) [139].

4.1. INTRODUCTION

Improving the security and efficiency of airports are two of the most important strategic objectives of the International Civil Aviation Organization (ICAO) [140]. Apart from ICAO and airports themselves, the research community has shown interest in methods to estimate and improve both security and efficiency.

Airport terminal efficiency has been studied using a wide range of different approaches. For instance, data driven approaches utilize airport data to estimate their efficiency [141, 142], while Bayesian models have been used to more efficiently process the vast amount of airport data [143]. Moreover, traditional simulation studies estimate efficiency in current and hypothetical scenarios [144, 145]. Finally, agent-based simulation methods were used to more accurately incorporate heterogeneous passenger behavior [146, 147].

Airport security is driven by a large set of rules and regulations defined by a variety of institutes. For instance, ICAO has a security manual [124], the European Union has regulations [121, 122], and the United States has the Aviation and Transportation Security Act [123]. These rules and regulations form the basis for the implementation of security measures at airport terminals, but airports still have some freedom to implement these measures according to their preferences.

To assess (and/or improve) airport terminal security, many methods have been proposed in literature. Most commonly, the so-called threat-vulnerability-consequence (TVC) methodology is used in practice. Many variants of the TVC methodology exist: the Risk Analysis and Management for Critical Asset Protection (RAMCAP) approach [9], the ICAO security manual [124], the security risk assessment handbook [7], and the RAND terrorism risk estimation handbook [8]. In the TVC methodology, security risks are estimated based on three threat components: threat likelihood, vulnerability and consequence. These components are individually assessed by security experts, and are used as a guide to implement security measures. The TVC methodology heavily depends on security experts, who cannot take into account all complex processes and interactions at an airport terminal (see also Chapter 2).

To overcome the dependency on security experts, researchers have developed analytical methods to assess security risks, as also discussed in Chapter 1. Some of these methods recognize that security and efficiency are related. However, many of the security-oriented studies only consider efficiency as a constraint, while most efficiency-oriented studies model security measures only as an efficiency bottleneck [144]. A notable exception to this is the work of Wilson et al. [148], in which efficiency and security are estimated simultaneously using a simulation method. However, this work lacks a formal methodology and uses a basic notion of security by only incorporating vulnerability in their analysis. Moreover, the work of Kirschenbaum [149] investigates tradeoffs between security and efficiency using informal quantitative methods, but does not follow a formal analytical methodology. Finally, Grant and Stewart performed a traditional security risk assessment on an Improvised Explosive Device (IED) attack, while taking into account costs for the airport [44]. Their work concerned a higher-level tradeoff between costs and security, while other efficiency performance indicators may be of influence as well.

The goal of this chapter is to develop a formal methodology to analyze security, efficiency, and identify and quantify relationships between them, using agent-based mod-

eling as a central paradigm. Agent-based modeling forms a promising paradigm, as it allows for detailed analysis of security, efficiency and their corresponding relationships, which is often hard in the above-mentioned modeling frameworks. Agent-based models are important tools to better understand complex systems, such as airports. Attackers and defenders can naturally be represented by agents with diverse strategies and non-linear interactions between them. Agent-based modeling therefore forms a promising paradigm in which both efficiency and security can be estimated simultaneously.

Other security risk assessment methodologies, such as attack trees, often transform airport operations to a group of linear relations, which limits the modeling capacities of these methods. Complex interactions, such as the detection of an ongoing attack by a behavior-detection employee cannot be modeled in such paradigms. Furthermore, spatial-temporal elements, such as the position of passengers over time, are hard to incorporate in these methods. Moreover, most other security risk assessment approaches either do not consider efficiency performance indicators at all, or consider efficiency as a constraint. Alternatively, discrete event simulations can be used to perform this type of analysis. However, in discrete event simulations "the entities do not actively follow individual incentives and do not interact but pass through the model according to the underlying sequence of operations" [49]. In this work, we incorporate human behavior and the interactions between agents (for instance the behavior-detection employee and the attacker), which fits the agent-based paradigm better.

The methodology proposed in this chapter extends the AbSRiM approach of Chapter 2. It consists of four steps: scope selection, agent-based model definition, security and efficiency estimation, and analysis of simulation results. The steps of the AbSRiM approach are combined with a typical agent-based approach to analyze efficiency of operations. We apply our methodology to a case study in which we analyze security regarding an IED attack, commonly used efficiency performance indicators at an airport terminal, such as queuing time for passengers and number of employees, and their corresponding relationships.

Section 4.2 introduces the proposed methodology, while the rest of the chapter applies the methodology to a case study described in Section 4.3. In Section 4.4 the corresponding agent-based model is introduced, and in Section 4.5 the estimation of security risks and efficiency performance indicators relative to the case study is described. Finally, in Section 4.6 the simulation results are analyzed and discussed.

4.2. METHODOLOGY

Our methodology to analyze security risks, efficiency performance indicators and corresponding relationships contains four main steps, outlined in Figure 4.1. The first step is used to determine the scope of the analysis. It is further discussed in Section 4.2.1. The second step, agent-based model definition, forms the basis of the analysis. In this step, an agent-based model is defined that will be further used to estimate efficiency performance indicators and assess security risks. This step is further discussed in Section 4.2.2. Based on the defined models, security risks are assessed and efficiency performance indicators are estimated by means of Monte Carlo simulations in the third step of the methodology (Section 4.2.3). Finally, in the fourth step the simulation results are analyzed (Section 4.2.4).

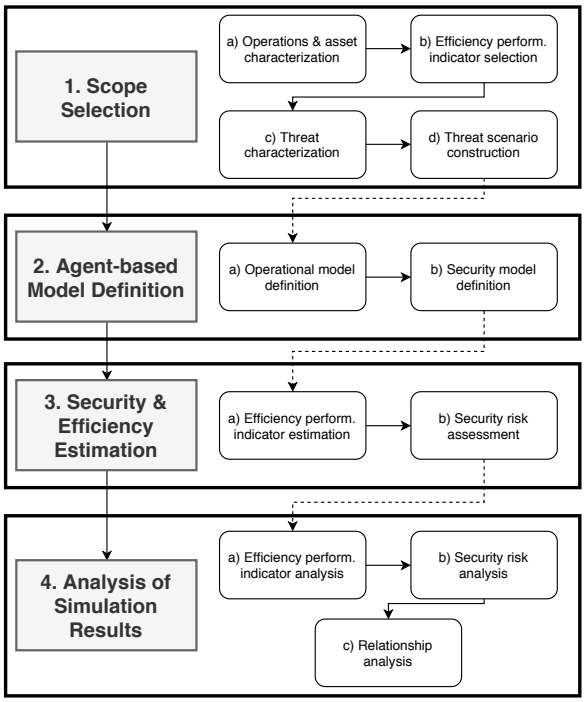


Figure 4.1: The methodology used in this chapter.

Steps 1(a,c,d) and 3(b) are used in most variants of the TVC methodology. These steps are complemented by the additional steps 2(b) and 4(b), which were previously discussed in the AbSRiM approach of Chapter 2. Furthermore, a typical agent-based approach for estimation of efficiency of operations follows steps 1(a-b), 2-4(a). This methodology integrates these approaches, while adding step 4(c) to find relationships.

4.2.1. SCOPE SELECTION

In this first step the scope of the project is defined. The first step is the selection of the specific operational processes and assets to focus on. For the airport domain, an example process can be the check-in process at the airport terminal, while assets can be passengers or the airport terminal building. Based on the selected domain, a set of efficiency performance indicators has to be selected and a set of security threats have to be characterized. Based on the characterized security threats, specific threat scenarios for each of the threats are constructed. Efficiency performance indicators are used to quantify a specific element of efficiency in the selected domain, related to efficiency goals of the airport. In the airport domain, this can for example be the average queuing time for passengers. An example threat scenario is the following: a single attacker brings an IED to a regional airport and detonates it in a publicly accessible area of the airport.

4.2.2. AGENT-BASED MODEL DEFINITION

For the above selected scope of the project, the agent-based models M and $M_1, \ldots, M_n$ are defined. The operational model M is defined to model the selected operations of the domain and is used to estimate efficiency performance indicators selected in the previous step.

Model M defines an environment that represents the environment of the domain area. Then, a set of agents that execute the standard operations in the domain is defined. In an airport, this can for example be passengers or check-in employees. Finally, a set of defender agents is defined. In the context of airports, these can for instance be behavior-detection employees or X-ray officers. These defender agents can additionally have operational task, such as helping passengers find directions.

The model forms the basis for security models $M_1, \ldots, M_n$. These models are used to represent the n threat scenarios in S, which are in turn used to estimate security risks related to the corresponding threat scenario. Each model M_i defines a non-empty set of attacker agents, on top of the components already present in M. The attacker agents execute the attacker behavior in threat scenario s_i, while the defender agents try to prevent the attackers from being successful.

Both a modeling language and an agent architecture need to be selected to specify the models. A modeling language should at least include the following abilities: (1) representation of time; (2) representation of stochastic processes; (3) specification of both qualitative and quantitative aspects; and (4) representation of behavioral and cognitive properties of agents and interaction between agents. The following elements should at least be present in an agent architecture: (1) observation and action; (2) storage of information; (3) maintenance of goals; and (4) reasoning. The Temporal Trace Language (TTL) [82] and LEADSTO [83] are example languages. The BDI architecture [84], and the Desire architecture [86] are example architectures. A more extensive discussion on language selection and architecture selection is provided in Chapter 2.

4.2.3. SECURITY & EFFICIENCY ESTIMATION

The third step of the methodology is the estimation of efficiency performance indicators and assessment of security risks from simulation results. A set of efficiency performance indicators and security risks are generated, that are used to identify and quantify relationships in the next step.

EFFICIENCY PERFORMANCE INDICATOR ESTIMATION

Efficiency performance indicators are estimated by performing Monte Carlo simulations. These Monte Carlo simulations are performed with model M. By extracting relevant information from simulation results of M, each of the efficiency performance indicators defined in 1(b) are estimated. For example, the average queuing time of passengers can be obtained by averaging over the queuing time for each of the passengers present in the simulation model.

SECURITY RISK ASSESSMENT

For each threat scenario $s_i \in S$ defined in step 1(d), a corresponding security risk r_i is calculated based on simulation results of model M_i defined in step 2.

An agent-based security risk management methodology is used following the AbSRiM approach of Chapter 2. A security risk r_i is defined for some time period T as a function of *Threat Likelihood* and *Conditional Risk*, as outlined below.

$$R(s_i, T) = f(P(s_i, T), R_c(s_i))$$

Risk $R(s_i, T)$ (or r_i in short) is the risk value for threat scenario s_i in time period T. Conditional risk $R_c(s_i)$ is estimated as follows. For each threat scenario s_i and asset a_l (as defined in the scope selection), a real-valued Consequence function $C(M_i^j, a_l)$ is defined. This function is used to determine the Consequence value for some simulation run j in model M_i. This Consequence function incorporates estimates of direct losses and indirect losses. Direct losses for instance include fatalities and physical damages of a simulated threat scenario. Indirect losses, such as decreased number of future passengers and business disruptions, are then based on the estimated direct losses and historical data.

Monte Carlo simulations are performed to estimate conditional risk based on a set of N simulation runs. This is done by calculating the following estimate of conditional risk for some scenario:

$$\hat{R}_c(s_i) = \frac{\sum_{j=1}^{N} \sum_{a_l \in A} C(M_i^j, a_l)}{N}$$

where $C(M_i^j, a_l)$ is the consequence for asset l in simulation run j of model M_i. $\hat{R}_c(s_i)$ is the estimated conditional risk for scenario s_i. By calculating the ratio between the number of nonzero consequence values and N (i.e., the total number of consequence values), the vulnerability of the scenario can be obtained. The mean of the nonzero consequence values corresponds to the consequence of the scenario.

Threat likelihood $P(s_i, T)$ for threat scenario s_i is estimated independently from model M_i. Commonly, crime databases and intelligence data are used to estimate the Threat Likelihood [44].

4.2.4. ANALYSIS OF SIMULATION RESULTS

Simulation results are analyzed following a structured approach. First, the influence of model parameters on efficiency performance indicators is established using statistical analysis techniques. For instance correlation analysis, or more advanced methods such as (global) sensitivity analysis [65, 150, 151] and uncertainty analysis [151] can be used. Similarly, the influence of model parameters on security risks is established using the same techniques.

Relations between model parameters, security risks, and efficiency performance indicators are obtained in this step. This is done by determining which parameters influence both security risks and efficiency performance indicators. By analyzing emergent effects in the defined agent-based models, unexpected relationships can be identified as well.

4.3. CASE STUDY

The remainder of this chapter applies this methodology to analyze security and efficiency, and identify and quantify relationships between them in the domain of a small

airport terminal. The reference airport handles under 2 million passengers per year and has a centralized security checkpoint. The operations that are included in the study are: check-in, facility visits, security checkpoint operations, queuing, gate processes and the movement of passengers between these processes. We focus on a single asset: humans (i.e., all passengers and employees). A visualization of the airport terminal used in this case study is shown in Figure 4.2.

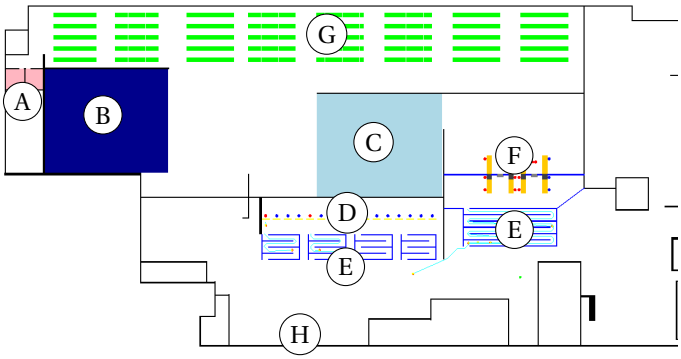


Figure 4.2: The airport layout of the case study, with indicators for different areas. A, B and C are facility areas. D is the check-in area and E are queuing areas. F is the checkpoint area, G is the gate area and H is the entrance area.

We focus on a single threat: a bomb attack in the open areas of the airport terminal, as for instance seen at the Atatürk Airport attack and the Zaventem Airport attack. Based on this threat, two threat scenarios in which an attacker aims to detonate an IED in the open areas of the airport are represented: an early attack and a late attack.

Five efficiency performance indicators are defined: number of employees n, mean time in checkpoint queue over all passengers T_{queue}, mean time to gate over all passengers T_{gate}, number of missed flights *miss*, and monetary loss *loss*.

We focus this case study on three main research questions, as outlined below.

- How does the number of passengers influence the identified efficiency performance indicators and the security risk with respect to the security threat?
- How does the number of checkpoint lanes influence the identified efficiency performance indicators and the security risk with respect to the security threat?
- How does the number of behavior-detection employees and their respective strategies influence the identified efficiency performance indicators and the security risk with respect to the security threat?

4.4. AGENT-BASED MODEL

Three agent-based models for the above selected scope are defined. We refer to the operational model as M, while the model that includes the threat scenario is referred to as M_{ied}. The modeling language is discussed in Section 4.4.1, and the agent architecture is discussed in Section 4.4.2. The operational model and the security models are discussed

in Section 4.4.3 - Section 4.4.4. Section 4.4.5 finally describes the parameters used in the models.

4.4.1. MODELLING LANGUAGE

To specify the dynamics of a multiagent system, the order-sorted predicate logic-based language called LEADSTO is used [83]. This language allows both discrete and continuous modeling of a system at different aggregation levels. Furthermore, one can express both qualitative and quantitative aspects of a system using LEADSTO.

Dynamics in LEADSTO are represented as evolution of states over time. A state is characterized by a set of properties that do or do not hold at a certain point in time. To specify state properties for system components, ontologies are used that are defined by a number of sorts, sorted constants, variables, functions and predicates (i.e., a signature). For every system component A, a number of ontologies can be distinguished: the ontologies *IntOnt*(A), *InOnt*(A), *OutOnt*(A), and *ExtOnt*(A) are used to express respectively internal, input, output and external state properties of the component A. For a given ontology Ont, the propositional language signature consisting of all state ground atoms based on Ont is denoted by *APROP*(Ont). State properties are specified based on such ontology by propositions. Propositions are formed, combining ground atoms by logical operators such as conjunction, negation, disjunction, and implication. Input ontologies contain elements for describing perceptions of an agent from the external world, such as the observed function *obs*: $IntOnt(A) \rightarrow APROP(IntOnt(A))$. Output ontologies describe actions and communications of agents. To this end, the function *performed*: $ACTION \rightarrow APROP(OutOnt(A))$ is introduced. Then, a state S is an indication of which atomic state properties are true and which are false: S: $APROP(Ont) \rightarrow \{true, false\}$.

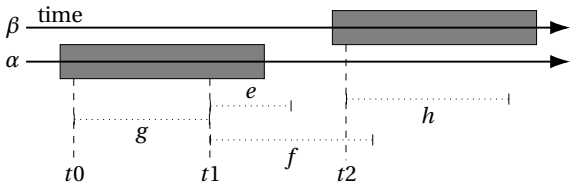


Figure 4.3: Timing relationships for LEADSTO expressions.

LEADSTO enables modeling of direct temporal dependencies between two state properties in successive states, also called dynamic properties. A specification of dynamic properties in LEADSTO is executable and can be depicted graphically. The format is defined as follows. Let $\alpha 1$ and $\alpha 2$ be state properties of the form 'conjunction of atoms or negations of atoms', and e, f, g, h non-negative real numbers. In the LEADSTO language the notation $\alpha 1 \rightarrow\!\!\!\!\rightarrow_{e,f,g,h} \alpha 2$ means: if state property $\alpha 1$ holds for a certain time interval with duration g, then after some delay (between e and f) state property $\alpha 2$ will hold for a certain time interval of length h (Fig. 4.3). To indicate the type of a state property in a LEADSTO property we shall use prefixes $internal(c)$, *input*(c), *output*(c) and *external*(c), where c is the name of a component. Consider an example dynamic prop-

erty:

$$input(A)|obs(arrest\_fail) \twoheadrightarrow_{0,0,1,1}$$
$$output(A)|performed(detonate())$$

Informally, this example expresses that if agent A observes a failed arrest during some time unit, then A will detonate an IED in the following time unit. Next, a *trace* or *trajectory* γ over a state ontology Ont is a time-indexed sequence of states over Ont (where the time frame is formalized by real numbers). A LEADSTO expression $\alpha 1 \twoheadrightarrow_{e,f,g,h} \alpha 2$, holds for a trace γ if:

$$\forall t1[\forall t[t1 - g \leq t < t1 \Rightarrow \alpha 1 \text{ holds in } \gamma \text{ at time } t]$$
$$\Rightarrow \exists d[e \leq d \leq f \& \forall t'[t1 + d \leq t' \leq t1 + d + h$$
$$\Rightarrow \alpha 2 \text{ holds in } \gamma \text{ at time } t']]$$

More details on the semantics of the LEADSTO language can be found in [83].

4.4.2. AGENT ARCHITECTURE

Agents are modeled following an adapted version of the AATOM architecture visualized in Figure 4.4. The architecture is loosely based on a framework of Blumberg [152], Hoogendoorn [153] and Reynolds [154]. It is described in detail in a technical report [155].

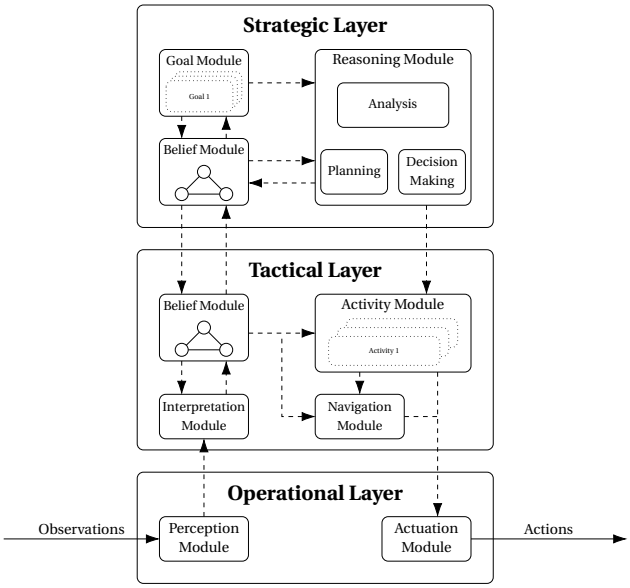


Figure 4.4: The AATOM architecture and its different modules.

In this architecture, three layers are distinguished, namely the operational layer, the tactical layer and the strategic layer. Each of these layers has a set of modules that ex-

ecute specific tasks. The operational layer is responsible for doing observations (perception module) and performing actions (action module). Communication with other agents is also executed by the action module. Based on observations, actions and internal states the belief module maintains a belief in the tactical layer. That layer is also responsible for navigation (navigation module) and activity execution (activity module). Finally, the strategic layer maintains a higher level belief (strategic belief module) and generates a plan (planning module). A plan is defined as an ordered sequence of activities that are executed by the agent. For each agent in the model, relevant modules are described in more detail.

Activities form a central concept in this architecture. They have a starting condition, a set of actions that have to be executed and an ending condition. Based on these conditions, an activity is defined to be in either of the three different activity states: *not_started*, *in_progress*, *finished*. All activities start in the $not\_started$ state and switch to the $in\_progress$ state when the starting condition is met. Finally, they switch to the *finished* state when the ending condition is met. The activity state is represented as follows: *activity_state*: $ACTIVITY \times ACTIVITY\_STATE \rightarrow APROP(IntOnt(A))$. In addition, an activity can be the next activity in the planning of an agent (determined by the planning module). This is defined in the following function: *next_activity*: $ACTIVITY \rightarrow APROP(OutOnt(A))$ is introduced.

Employee agents and attacker agents only have a single activity they can perform, while passenger agents can execute more activities. They therefore plan their activities following a set of simple rules, explained in more detail in Section 4.4.4.

4.4.3. ENVIRONMENT

The airport terminal environment consists of several elements, categorized into four different categories: physical objects, IEDs, areas and flights. A visualization of the airport terminal environment is shown in Figure 4.2.

Two types of physical objects, *wall* and *desk*, are defined. An IED is defined by its location, the number of particles and mass. It is carried by an attacker, denoted *carried_by* (*ied*, *attacker*). Areas are used to specify functionality of regions in the airport terminal, where *check-in_area*, *checkpoint_area*, *facility_area*, *queuing_area*, *gate_area* and *entrance_area* are the types of areas present in the model. Some areas, such as the *gate_area*, are accessible to passengers only after execution of the *checkpoint_activity* (airside), while others, such as the *entrance_area*, are publicly accessible (landside). Finally, a *flight* is defined to be an abstract concept with the following properties: $departs\_at(flight, f_{time})$, *has_gate*(*flight*, *gate_area*) and $has\_desk(flight, desk)$. The value f_{time} is the time at which the flight departs. The flight also has at least one $desk$ that passengers use for checking in and exactly one *gate_area*.

4.4.4. AGENTS

The model M contains three types of agents, namely: passengers, operational employees and behavior-detection employees (BDE). The last two agent types are also the defender agents in the model. We assume that there are no other persons, such as visitors, as they form a very small part of the population in the airport under consideration. All agents are human agents and are designed using the framework discussed in Section 4.4.2. These

agents are discussed in more detail in subsequent sections.

PASSENGER AGENT

Passengers are agents that depart with some flight f in the environment. They are characterized by the following five properties: arrival time $t_{arrival}$, level of disorientation d, suitability of luggage s, checked-in c and facility visitor y.

The arrival time $t_{arrival}$ is the time at which the passenger is generated (in the *entrance_area*). The level of disorientation d refers to how disoriented or confused the passenger appears in the airport, and the suitability of luggage s refers to how well the luggage of the passenger fits the appearance of the owner. For example, a business traveller with a large suitcase has a low suitability of luggage. Both these properties are conceptualized with a real number. These properties are important indicators that are used in the SPOT program of the TSA [137, 138]. In the SPOT program, officers assign points to passengers to quantify their danger to the airport. If the points assigned to a passenger exceed a threshold, a secondary screening is initiated.

Checked-in c is a Boolean value indicating whether the passenger is already checked-in on arrival, and facility visit y indicates which facility the agent will visit (*none*, *bathroom*, *restaurant*, *shop*). Passengers can observe physical objects and other agents that are in line of sight within a radius r_{obs}. Furthermore, passengers can observe the area that they are in, and the flight they are taking. Finally, a wait request communicated by other agents can be observed.

Based on these observations, passengers find a collision free path between the different activity locations using the Jump Point Search pathfinding algorithm [156], sometimes used in pedestrian simulators [157]. This is executed by the navigation module, and done when all activities are in the *not_started* activity state or when an activity switched from *in_progress* to *finished*. Passengers follow their generated path (using the action module) by changing their location point using the Social Force model defined by Helbing and Molnar [158]. Passengers can also wait for a specified time t_{wait}.

Passengers can perform the following activities: *check*-in_*activity*, *checkpoint_activity*, *facility_activity* and *gate_activity*. These activities are planned (in the order as they appear) by the planning module. The checkpoint and gate activity are always executed by agents, while the *check-in* and *facility_activity* are only executed if the prop -$in\_activity$, *checkpoint_activity*, *facility_activity* and *gate_activity*. These activities are planned (in the order as they appear) by the planning module. The checkpoint and gate activity are always executed by agents, while the *check-in* and *facility_activity* are only executed if the property checked-in c is *false* or the property facility visit f is not *none*, respectively. If the *check-in_activity* or *checkpoint_activity*, cannot be executed (when all activity areas are occupied), passengers perform a wait action in the nearest queuing area until an activity area becomes free. Passengers are removed from the model when $t = F_{time}$.

The $check\text{-}in\_activity$ is executed in a check-in area and consists of a wait action. The activity starts when the passenger observes a wait communication of an employee. The *checkpoint activity* is executed in a checkpoint area and consists of the same steps as the *check-in activity*. The *facility_activity* consists of a wait action. The time of the wait action depends on the type of facility f that is visited. Finally, the gate activity is executed in the gate area of the flight of the passenger and consists of a single wait action until the

flight leaves. The LEADSTO properties below formalize the gate activity.

$input(A)|obs(flight) \wedge obs(gate\_area)$
$\& external(A)|has\_gate(flight,gate\_area)$
$\& internal(A)|next\_activity(gate\_activity) \rightarrow\!\!\!\!\rightarrow_{F_{time}-t,F_{time}-t,1,1}$
$output(A)|performed(wait(F_{time}-t))$
$output(A)|performed(wait(F_{time}-t)) \rightarrow\!\!\!\!\rightarrow_{0,0,1,1}$
$internal(A)|activity\_state(gate\_activity,finished)$

ATTACKER AGENT

The attacker agent is modeled in the models $M_{ied-early}$ and $M_{ied-late}$. It is a human agent, like passengers, characterized by its arrival time $t_{arrival}$ and the level of disorientation d, suitability of luggage s. In $M_{ied-early}$, the attacker has an early $t_{arrival}$, while this is late in $M_{ied-late}$. The attacker agent has a single goal: achieve as many fatalities at the airport as possible.

To achieve this goal, it can observe physical objects, passengers and attackers in radius r_{obs}. The attacker can further determine the area it is currently in. The number of passengers at the checkpoint area and the check-in area can also be observed, regardless of the observation radius. This can be due to communication with other attackers, or observation using tools. Finally, the attacker can observe that it is being arrested by a BDE.

The attacker carries an IED that it uses to cause fatalities. To be able to be successful (from an attacker's perspective), the attacker executes the *attacker_activity*. The activity consists of three phases: target selection, movement to target and execution of attack. The target selection is based on a single criterion, namely the observed number of people in the *checkpoint_area* and the *check-in_area*. The attacker chooses the target with the highest number of passengers, independent of its characteristics. In the second phase, the attacker moves from the arrival location to the target area. The attacker can then be observed by a BDE (if present), resulting in one of two outcomes. With a probability of p_{arrest} the attacker is arrested and cannot finish the attack, while otherwise the attacker detonates the IED on the spot. This was for instance seen in attacker behavior at the Atatürk Airport attack of 2016 [2]. If the attacker was not observed by any BDE, it continues moving to the target area, where phase three is initiated. In this phase, the attacker

detonates the IED. The LEADSTO properties below formalize the activity.

$t_{arrival} = t \rightarrow\!\!\!\!\rightarrow_{0,0,1,1} internal(A)|path(target)$

$internal(A)|path(target)) \rightarrow\!\!\!\!\rightarrow_{1,t_{move},1,1}$
$prob(output(A)|performed(move(target)), p)$ & $prob(input(A)|obs(arrest), 1-p)$

$input(A)|obs(target) \vee obs(arrest\_fail) \rightarrow\!\!\!\!\rightarrow_{0,0,1,1}$
$output(A)|performed(detonate())$

$output(A)|performed(detonate())$ || $input(A)|obs(arrest) \rightarrow\!\!\!\!\rightarrow_{0,0,1,1}$
$internal(A)|activity\_state(attacker\_activity, finished)$

OPERATIONAL EMPLOYEE AGENT

The operational employee can observe a single passenger at a time in a small radius. It can execute a single action, namely the communication of a wait request. This observation and action is used in the single activity the standard employee executes: the *employee_activity*. This activity consists of the communication of a wait order (of a specified time t_{wait}) to the passenger, when a passenger is observed. The standard employee interacts with passengers that either perform the *check-in_activity* or the *checkpoint_activity*.

BEHAVIOR-DETECTION EMPLOYEE AGENT

The behavior-detection employee can observe physical objects, passengers and attackers in radius r_{obs} and in direct line of sight. They cannot be observed to be a BDE by attackers or passengers, as it operates undercover.

Three different strategies can be employed by the BDE: static observation, dynamic observation and intelligent observation. When performing static observation, the BDE positions itself at the queue in front of the security checkpoint and executes its job there. For dynamic observation, the BDE constantly moves between two areas: the *checkpoint_area* and the *check-in_area*. Finally, when performing intelligent observation, the BDE estimates every $t_{intelligent}$ seconds which area has most passengers. The BDE will then move to the area with the highest number of passengers and performs its job there.

The BDE randomly chooses one agent of these observed agents (that it did not evaluate yet) to evaluate if it is an attacker or not. To do that, the BDE assigns points to the observed agent based on the SPOT program [137, 138, 159]. First, a threshold $d_{threshold}$ is defined for level of disorientation d. If the observed agent has a level of disorientation $d > d_{threshold}$, two points are assigned. Moreover, the suitability of luggage s is compared against a threshold $s_{threshold}$. If the agent exceeds the threshold, three points are assigned. Finally, if the difference between the arrival time $t_{arrival}$ and the flight time of an agent exceeds the threshold $f_{threshold}$, one point is assigned. If the number of points exceeds four, the BDE attempts to arrest the agent. If the agent is a passenger, the passenger is arrested and the BDE will leave the airport terminal with the passenger. If the agent is an attacker, the $arrest$ action is executed with a success rate of p_{arrest}, while the *arrest_fail* action is executed otherwise. If the arrest action is executed, the attacker

Table 4.1: The model parameters that were varied in the experiments.

Parameter	Values
Number of flights f	1, 2, 3 flights
Number of checkpoint lanes open l	2, 3, 4 lanes
Number of check-in desks open k	3, 5 desks
Number of BDEs d	0, 1, 2 empl.
BDE strategy	*static, dynamic, intell.*
Attacker time	*early, late*

is stopped and will not detonate the IED. If the arrest was not successful, the attacker detonates the IED on the spot.

It takes some time $t_{evaluation}$ to evaluate the agent. This time is calculated as follows:

$$t_{evaluation} = t_{max} - (c_1 \cdot abs(d_{threshold} - d) + c_2 \cdot abs(s_{threshold} - s))$$

where t_{max} is the maximum time that a BDE spends on evaluation of agents, and the c_i's are constant. This relationship indicates that passengers with traits close to the threshold take longer to evaluate than passengers that are not.

The BDE uses the above described observations and actions to execute the *behavior_detect_activity*. In this activity, the BDE moves between a list of locations $location\_list$ in the airport terminal, while checking if it observed an attacker. When this is the case, the employee tries to arrest the attacker.

It is noted that both the attacker and the BDEs can be modeled to be more complex than the current form. For example, more strategic behavior (i.e., a small decoy attack) in both the attacker and the BDEs can be included. Collaboration between teams and camera observations could also be added. For now, this is beyond the scope of this chapter.

4.4.5. MODEL PARAMETERS

Five model parameters were defined and shown in Table 4.1. Other internal parameters of the models are discussed Section 4.6.1.

Passenger arrival at the airport follows a distribution based on the number of flights f and data collected at the regional airport. This has a direct influence on the number of passengers present within the model over time. The number of checkpoint lanes open refers to the number of passengers that can perform the *checkpoint_activity* simultaneously. This influences the number of employees directly as follows: $n_{checkpoint} = 4l + mod(l, 2)$. This relationship indicates that it is beneficial to open checkpoint lanes in pairs, as also recommended by IATA [104]. The number of check-in desks open refers to the number of check-in desks through which a passenger can check in. An open check-in desk requires a single employee. The number of BDEs present influences the number of employees present, and potentially the effectiveness of the defense. Furthermore, three BDE strategies are defined: static, dynamic and intelligent. Some of these parameters cannot be influenced by the airport directly. For example, the number of flights also depends on airlines, and the number of BDEs has to be determined in collaboration with

regulators. Finally, the attacker time t_{attack} defines the time that the attacker executes its attack.

4.5. ESTIMATION OF SECURITY AND EFFICIENCY

The third step of the proposed methodology estimates security risks and efficiency performance indicators based on the agent-based models described above. They are discussed in detail below.

4.5.1. EFFICIENCY ESTIMATION

The efficiency performance indicators, as defined in Section 4.3 are calculated as follows. The time in checkpoint queue for passengers T_{queue} is measured by calculating the time a passenger spends in the $queuing\_area$ closest to the $checkpoint\_area$. A passenger is considered to have missed its flight if it is not in the $gate\_area$ at time f_{time}. We define loss as follows:

$$loss = ((|P_{max}| - |P|) \cdot rev_p - miss \cdot c_{miss}$$

where $|P_{max}|$ is the maximum number of passengers that the airport can process. P is the set of passengers that arrived on the flight day and rev_p is the mean revenue per passenger. Furthermore, c_{missed} is the costs that an airport has for each passenger that misses a flight. The other efficiency performance indicators, number of employees and time to gate, are trivially obtained from the simulation results. For each of the defined efficiency performance indicators it holds that lower is better.

4.5.2. SECURITY RISK ASSESSMENT

As defined in Section 4.2.3, the Consequence function needs to be defined. Furthermore, Threat Likelihood has to be estimated independently from the models. Both of these elements are described in more detail below.

IED CONSEQUENCES

As an IED attack at an airport terminal is modeled, a Consequence model is defined to estimate the number of lives lost after an attack. The model is based on the work of Pope [160], who designed a prediction tool that is able to quickly assess the human injury after a terrorist attack. The Consequence model described below forms the Consequence function $C(M^j_{ied})$.

It is argued that there are two main causes for fatalities after an IED attack: blast wave propagation and fragmentation injuries. While other factors are of influence on human injuries, only these two elements are considered in this model.

Blast wave prediction The explosion of an IED causes the release of a lot of energy, resulting in the propagation of a blast wave. Rapid changes in pressure are associated with this blast wave and can cause injury or death. Kingery and Bulmash [161] show that there is a relation between the mass of the explosive, the distance to the explosive, and

the incident pressure P. This relation is outlined below:

$$z = \frac{d}{mass^{1/3}}$$
$$U = k_0 + k_1 \log_{10} z$$
$$P = c_0 + c_1 U + c_2 U^2 + \ldots + c_n U^n$$

where d is the distance in meters between the IED and the target and $mass$ is the IED mass in kg. The k_i's and c_i's are constants, while P refers to the incident pressure in kPa. The relationship above assumes an unobstructed path between the IED and the target, while in practice walls and other physical objects can reflect the pressure wave. This is modeled by generating imaginary IEDs on a commensurate location on the other side of the wall. Walls are then ignored and the pressure contributions from both sources are superimposed to find the total pressure at a specific location.

The incident pressure at the location of each human agent is recorded and translated to a fatality probability, based on the work of Zipf and Cashdollar [162]. Finally, a random number is drawn to determine if the agent survived or not. The number of fatalities caused by the incident pressure is referred to as c_{blast}.

Fragmentation Prediction Apart from fatalities due to pressure changes, injuries and fatalities can arise due to the presence of fragments. Two types of fragments are distinguished: primary fragments and secondary fragments. Primary fragments are the fragments that are present within the IED, while secondary fragments are the fragments that originate from the environment (i.e., ceiling or other objects in the environment). Here, only a set of K primary fragments originating from the IED are considered. The initial direction Θ_{init} of a fragment is determined using a uniform distribution, while the initial speed v_{init} is set to be a constant.

The fragment will then move around the environment following a Newtonian motion model. If the path of the fragment intersects with a human, the distance that it covers within the human body (called depth of penetration, DOP) is recorded. A truncated linear relation between fatality probability and DOP is assumed. Finally, a random number is drawn to determine if the human survives or not, for each human that survived the blast impact. The number of human fatalities caused by fragmentation is referred to as c_{frag}.

Consequence Function The Consequence function is then defined to be the sum of the fatalities caused by the blast wave and the fragmentation.

$$C(M^j_{ied}, a_1) = c_{blast} + c_{frag}$$

In this function only the fatalities are taken into account. A more extended approach could also take into account injuries, damages to physical structures and indirect consequences, but this is currently beyond the scope of this chapter.

THREAT LIKELIHOOD

Threat likelihood is based on the work of Grant and Stewart [44], which in turn is based on historic data originating from a terrorist database [163]. From this database, it was obtained that historically there were an average of 1.7 IED attacks on airport terminals in Western countries each year [164]. Based on an estimate of 100 to 200 large hub airports, Grant and Stewart finally obtain an estimate of 0.5-2.0%. This percentage means that there is between 0.5% and 2% chance per airport terminal per year that someone attempts to attack it. As small airports seem less likely to be a target for terrorists, we chose a conservative likelihood of 0.5% for such an attack. As this estimate is based on historical data, it may very well be inaccurate. Data from intelligence agencies can provide more accurate estimates of threat likelihood.

4.6. EXPERIMENTS & RESULTS

Experiments performed with the above discussed model are presented in this section. The setup of the experiments is discussed in Section 4.6.1 and the results are discussed in Section 4.6.2.

4.6.1. MODEL CALIBRATION & EXPERIMENTAL SETUP

We have calibrated the model based on airport data, literature data, and assumptions if no data could be obtained. The calibrated parameters are found in Appendix A. We simulate a flight morning, between 05:00-07:00, where 05:00 corresponds to $t = 0$ sec. All flights are defined with the same departure time, which is standard practice in the airport under consideration. This is due to noise restrictions that are enforced on the airport. We assume a load factor of 0.75 for all aircraft, leading to 135 passengers per flight. The layout of the airport was shown in Figure 4.2. Revenue per passenger is based on an ACI economics report [165], while the costs per missed flights are based on assumptions. The proportion of checked-in passengers was based on estimates of airport managers. The actual proportion can be obtained from airline data, which was unavailable for airport managers. No data was available for the facility visits at the airport, so this was based on assumptions.

The desired speed was assumed to be 1 m/s, and only individual passengers were considered. We assumed a single carry-on luggage for passengers that were checked-in, and an additional checked luggage for passengers that were not checked-in. Based on discussions with airport managers, we assumed that 20% of passengers arrive in the first half hour, 60% of passengers arrive in the second half hour, and the remaining 20% of passengers arrive in the third half hour. Passengers in these blocks are generated using a Poisson distribution with an arrival rate that ensures that the right number of passengers arrive. Check-in times were based on estimates by airport managers. The checkpoint parameters were obtained by fitting a distribution over 102 manually collected checkpoint processing times between 05:00 and 07:00 at the airport on March 22nd 2017.

The observation radius r_{obs} of agents was assumed to be equal to 10 meter. The behavior-detection employee parameters were calibrated as follows. We assumed that a BDE arrests 0.025 passenger per hour, which falls within the range provided by the United States Government Accountability Office report [137]. Assuming that both pas-

senger disorientation d and passenger luggage suitability s follow a normal distribution with mean 0 and variance 1, the BDE thresholds d_{thres} and s_{thres} become 2.395. We assumed that following the SPOT program, 75% of the time an attacker is observed. This leads the attacker disorientation d and attacker luggage suitability s to follow a normal distribution with mean 3.5, and we assumed the same variance as for passengers. The attacker is generated in the same way as a passenger for all other parameters. Based on a CNN news report [166], we assumed that a BDE takes up to 20 seconds to evaluate the characteristics of a passenger or attacker. The corresponding evaluation constants c_i were based on assumptions. The arrest probability p_{arrest} was set to 0.8, based on the work of Price and Forrest [167].

The mass of the IED was based on a report by the Department of Homeland Security [168]. The number of particles and their initial speed were finally based on assumptions. Some of the constants found in Table A.1 will benefit from more extensive sensitivity analysis in the future. The output variables are the number of employees n, mean time in checkpoint queue T_{queue}, the mean time to gate T_{gate}, the number of missed flights $miss$, the monetary loss $loss$ and the risks $r_{ied-early}$ and $r_{ied-late}$ of the threat scenarios, as set out in Section 4.5.

For the implementation of the model, we developed the AATOM simulator [130], a Java-based open-source agent-based airport terminal operations simulator. This simulator contains a large library of airport terminal related components, and basic implementations of attacker agents. A visualization of an AATOM simulation was shown in Figure 4.2. For each combination of model parameters, 500 simulation runs were executed. A simulation ends when the flights have left; after 7200 seconds.

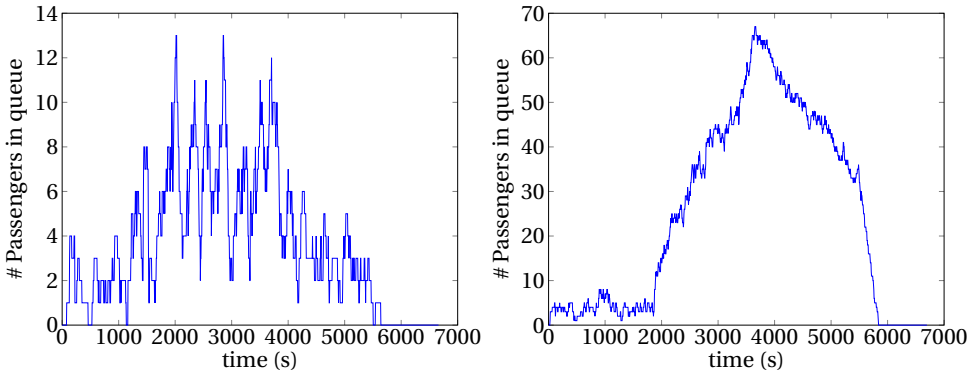
4.6.2. EXPERIMENTAL RESULTS

In this section, the results of the experiments are discussed. We first analyze the influence of the model parameters on efficiency, followed by an analysis of the influence on security. Finally, we discuss some of the relationships that were found between these performance areas. This constitutes to the fourth and last step in the proposed methodology.

EFFICIENCY PERFORMANCE INDICATORS

Figure 4.5 shows two typical buildups of passengers over time in the checkpoint queue, where Figure 4.5a shows the buildup under low passenger conditions, while Figure 4.5b shows a setup in saturated passenger conditions. From this figure the arrival pattern of passengers can be observed. When the slope of the figures changes, a different arrival rate of passengers is observed. This effect is more clearly visible in the three flight setup, as a larger queue buildup is observed there. This is due to the number of passengers in the queue being directly related to the mean queuing time T_{queue}.

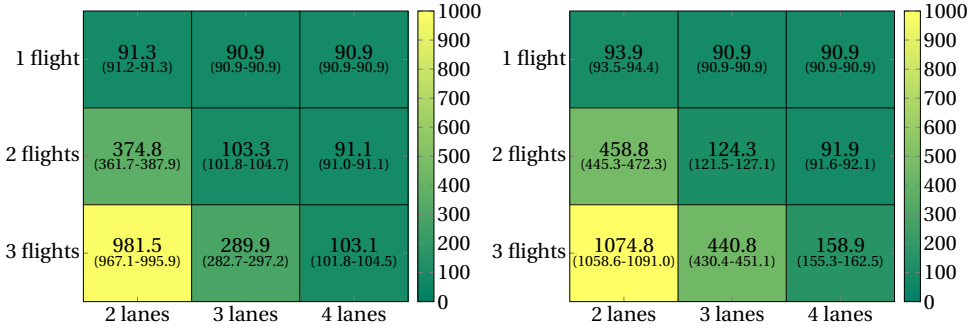
If we consider mean checkpoint queuing times T_{queue} for different airport setups (see Figure 4.6), it can be observed that three check-in desk setups mostly have shorter queuing times than five check-in desk setups. In the three check-in desk setups, the passengers arrive at the checkpoint queue more gradually due to longer waiting times at the check-in, leading to shorter queuing times. While not shown in the figure, it should be noted that five check-in desk setups generally lead to shorter times to gate for pas-



(a) $f = 2$ flights.

(b) $f = 3$ flights.

Figure 4.5: The number of passengers in the checkpoint queue over the flight morning. Graphs show a configuration of $l = 3$ checkpoint lanes and $k = 3$ check-in desks. Note that the scale of the y-axis is different for both configurations.



(a) $k = 3$ check-in desks.

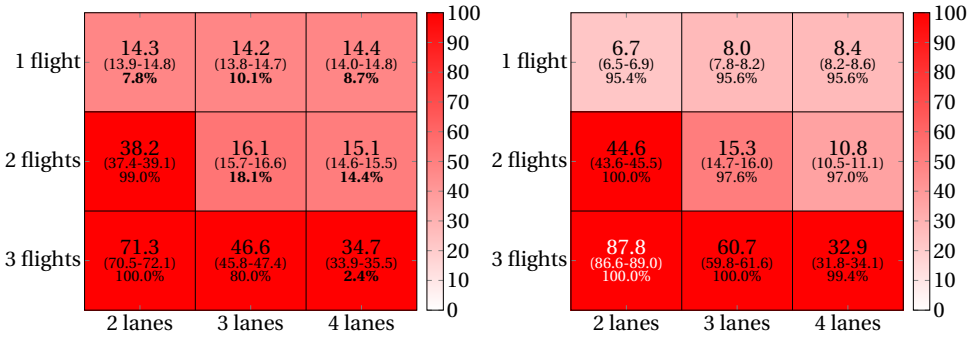
(b) $k = 5$ check-in desks.

Figure 4.6: The mean queuing time (in seconds) of passengers in the flight morning for different airport configurations. The values between brackets are the 95% confidence intervals.

sengers. Furthermore, opening more checkpoint lanes leads to a higher number of employees present, but opening too few checkpoint lanes can lead to an increase in missed flights. Determining the number of checkpoint lanes and check-in desks is an important tradeoff that airports have to make on a regular basis with respect to these efficiency performance indicators. However, these decisions do not only influence efficiency of the airport but also security, as discussed in Section 4.6.2.

CASUALTIES WITHOUT DEFENDERS

Figure 4.7 shows the mean number of casualties (in the case of a late attack) for different airport configurations. This corresponds to the conditional risks of the different threat scenarios. It further shows the choices of attacker (i.e., detonate IED at check-in or checkpoint) between the different configurations. In the three check-in desk setups, the attacker mostly chooses the check-in desks as a target, as most passengers are present



(a) $k = 3$ check-in desks open.

(b) $k = 5$ check-in desks open.

Figure 4.7: The number of casualties in a late attack for different airport configurations. The values between brackets are the 95% confidence intervals, and the percentages correspond to the proportion of times the attacker chooses for the checkpoint queue. Percentages smaller than 50% are shown in bold.

in that area. However, this does not hold for the setups with two or three flights and two checkpoint lanes open and the setup with three checkpoint lanes open and three flights. The attacker has a strong preference for the checkpoint as a target in the five check-in desk setup. It nearly always chooses for this location as a target.

More flights generally lead to more casualties per flight as well. This is mainly caused by a nonlinear increase in queue lengths for increasing numbers of flights. When comparing the number of casualties with the number of checkpoint lanes, it can be observed that a higher number of checkpoint lanes results in a lower number of casualties per flight. This does not hold for the single flight case, as the number of casualties remains constant or even increases when more checkpoint lanes are opened. In this case, any number of checkpoint lanes is sufficient to prevent a buildup of passengers in the queue. The extra casualties (for the configuration with five check-in desks) are caused by the higher number of employees that are present at the checkpoint. In this situation, it is beneficial from both a security and efficiency perspective to reduce the number of checkpoint lanes open as much as possible. In all the other situations, it is beneficial from a security perspective to open more checkpoint lanes, but that clearly increases the number of employees. At the same time, mean queuing time T_{queue} is reduced. This constitutes to an important tradeoff that has to be made by airport managers.

If we compare the setups in which the check-in area was preferred by the attacker in the three check-in desk setups with the corresponding five check-in desk setups, it can be observed that with five check-in desks the number of casualties is reduced. This is the case, because the total number of passengers in the queue that is attacked is reduced. In general it holds that the size of the longest queue (i.e., checkpoint queue or check-in queue) is a good linear indicator for the expected number of casualties ($R = 0.72$). This also somewhat holds for the total number of passengers present in the open areas of the airport ($R = 0.59$), but not in situations in which at least ten passengers are present in the shorter queue ($R = 0.27$). To minimize the expected casualties, the airport should therefore minimize the size of the longest queue. Ideally, this is done by reducing the size of both queues. However, airport managers might not have the financial means

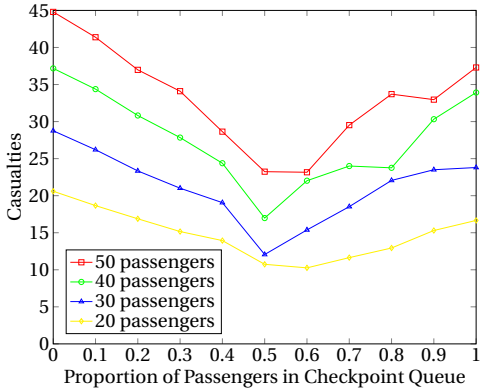


Figure 4.8: The relationship between the ratio of queue lengths and the number of casualties under different passenger loads.

to hire the required number of employees. Alternatively, the size of the queues could be balanced as much as possible, by choosing the right number of check-in desks and checkpoint lanes. This is a result similar to the results of Grant and Stewart, who argue that distributed security queuing "will offer casualty reductions when used in preference to centralized security queuing" [44]. Figure 4.7 shows how to minimize the expected casualties in our reference airport.

To illustrate that the size of the queues should be balanced as much as possible, we performed a controlled experiment in which the total number of passengers is set to a constant, while distributing the passengers over the different queues according to different ratios. Figure 4.8 shows the number of casualties for different proportions of passengers in the checkpoint queue. In this figure, a minimum number of casualties was observed at a ratio of around 0.5. In this case, the queues are equally balanced. This is a result that can be generalized to similar situations in other airports as well. The consequence of an IED attack can be lowered by distributing passengers over the available space as well as possible.

The number of casualties was found to be a bit higher when all passengers are in the check-in queues as compared to the checkpoint queue. This is the case, because the attacker can better position himself between the passengers than in the checkpoint queue (as can be seen from Figure 4.2). This trend is reversed (although not shown in the figure) for very low passenger numbers, as also discussed above. It should be noted that this strategy of balancing queues might lead to increased security risks of other threat scenarios, not considered in this chapter. This forms an interesting direction for future research.

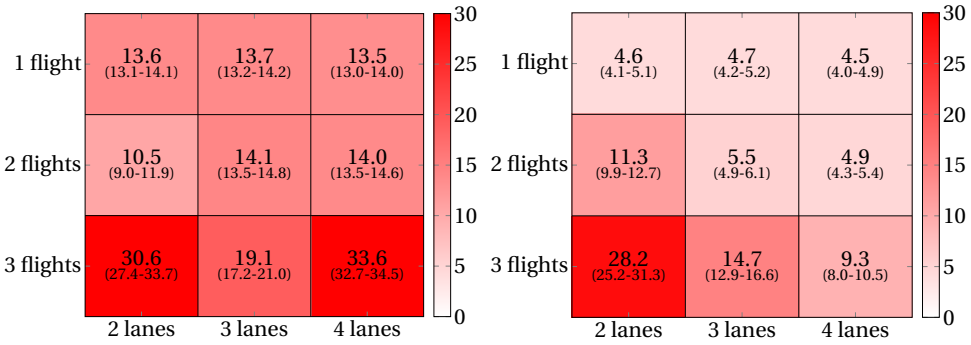
Different Passenger Types We analyzed the effect of different passenger types on our simulation results. We consider two passenger types in isolation: senior passengers and family passengers. The luggage drop time of senior passengers was calibrated to follow a normal distribution with mean 63.7 and variance of 35.1. Their luggage collect time follows a normal distribution with mean 59.4 and variance of 48.2. The luggage drop time

of family passengers then follows a normal distribution with mean 69.2 and variance of 36.1, while their luggage collect time follows a normal distribution with mean 80.6 and variance of 53.0. These distributions were based on manually collected checkpoint processing times on four days in March and April 2018. The classification of passenger type was performed manually as well. It should be noted that these distributions already include the effects on processing speed for different amounts of luggage. Furthermore, a large part of the senior passengers considered fly several times per year from the airport under consideration.

It was found that the number of casualties is reduced with 12.0% for senior passengers on average. This is due to faster collection of luggage for this type of passengers, as compared to the passengers considered in the rest in this chapter. Contrary, family passengers move through the security checkpoint slower than the default passenger. This leads to an increase of 3.4% of casualties on average. The mix of passenger types has a large influence on security risk and efficiency performance indicators. Airports therefore need to consider the passenger mix they serve when making decision related to both security and efficiency.

BEHAVIOR-DETECTION EMPLOYEE

In all airport setups and threat scenarios, the number of casualties is reduced when a (set of) BDE(s) is hired. This holds regardless of the strategy of the BDE. In general, the intelligent BDE is best capable of defending against attacks of different types. The agent is most frequently found at the area in which the attack will take place, and therefore performs more arrests than the other BDE types. A typical example of the performance of BDEs with different strategies is shown in Figure 4.9.

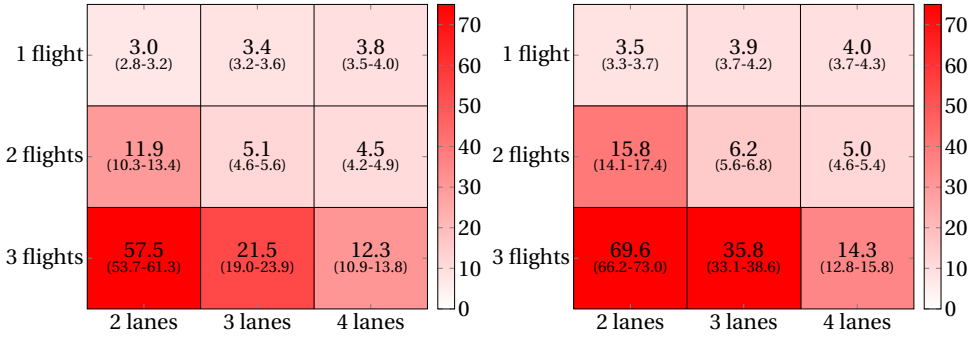


(a) Two static defenders.

(b) Two intelligent defenders.

Figure 4.9: The number of casualties in a late attack with three check-in desks for different types of defenders. The values between brackets are the 95% confidence intervals.

However, the intelligent defender is not always better capable of defending against attacks. Figure 4.10 shows the mean number of casualties in a late attack for two different defender strategies: dynamic and intelligent. The static defender performs similar to the intelligent defender and is therefore not shown. In this case, the dynamic defender performs better than the intelligent defender.



(a) One dynamic defender.

(b) One intelligent defender.

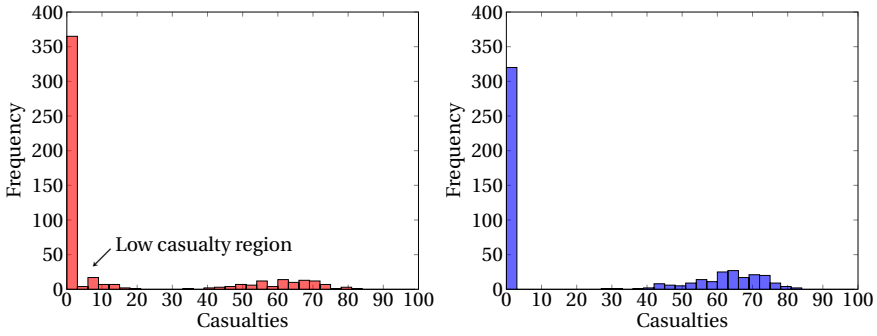
Figure 4.10: The number of casualties in a late attack with five check-in desks for different types of defenders. The values between brackets are the 95% confidence intervals.

This can be explained as follows. Figure 4.11 shows a histogram of casualties for the configuration with three flights, three checkpoint lanes, five check-in desks and a single BDE with different strategies. Note that in this configuration the checkpoint queue is much larger, and therefore the attacker always chooses this as a target. From Figure 4.11 it can be observed that the dynamic BDE make a higher number of arrests (zero casualties), and has a region in which a very low number of casualties is observed. This is the case, as the dynamic BDE moves between the check-in area and checkpoint area, while the other BDEs only perform their work in the checkpoint area. As the queue is long there, these other defenders do not have the time to assess every passenger, and therefore the attacker might be missed. The dynamic defender might observe the attacker (at the entrance area), as few other passengers are present in the check-in area. Note that these two areas are close together, and that the BDE can therefore observe passengers in both areas while it is in the check-in area. The region of very low casualties is caused by the failed arrests in this region. As only few passengers are around, fewer casualties are observed. On the contrary, when the intelligent defender (and also the static BDE) performs a failed arrest, the detonation of the IED occurs close to the checkpoint queue. This then leads to a higher number of casualties in the case of a failed arrest.

While not modeled in this chapter, observant passengers may also help prevent an ongoing attack to become successful. This was for instance seen in the 2018 Belgium train attack [169]. This forms an interesting direction for future research.

SECURITY AND EFFICIENCY

To be able to determine the sensitivity of the estimated efficiency and security outputs to the model parameters, Spearman's rank correlation test was performed. This test assesses monotonic relationship between the parameters and outputs. Conditional risk ($R_c(M_{ied})$) is used as an output parameter, as Threat Likelihood remains constant for all parameter combinations. Figure 4.12 shows the results of this test and indicates insignificant results ($p \leq 0.05$) crossed out.



(a) One dynamic defender.

(b) One intelligent defender.

Figure 4.11: Histogram of casualties for two different defender strategies: intelligent and dynamic. Results are shown for a configuration with three flights, three checkpoint lanes, five check-in desks and a single defender.

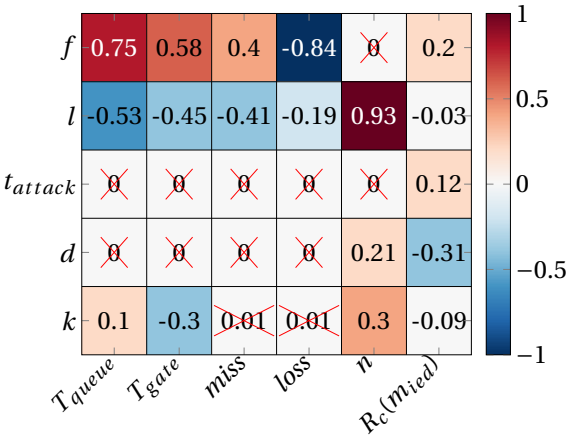
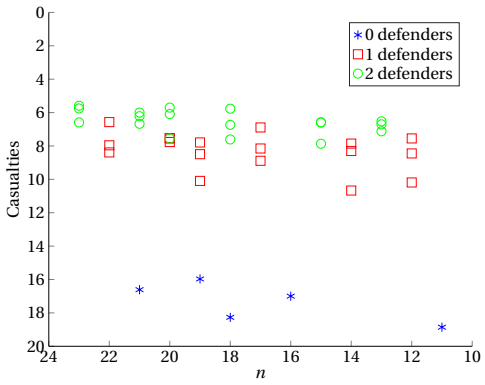
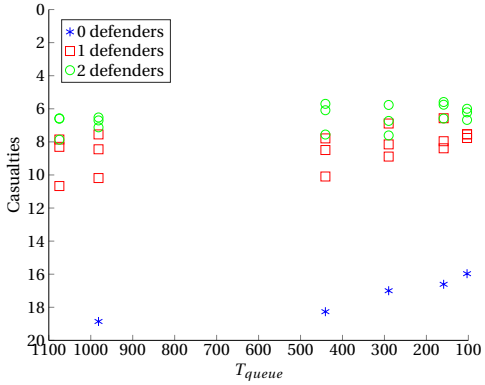


Figure 4.12: Spearman's rank correlation plot between model parameters and efficiency performance indicators and conditional security risks. The rows of this figure show the different model parameters, while the columns show the output parameters (efficiency performance indicators and conditional security risk). Insignificant results ($p \leq 0.05$) are crossed out.

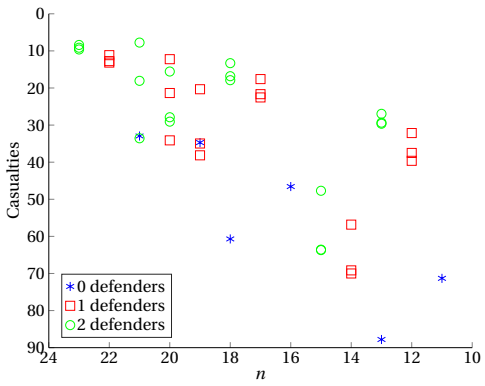
Results show that the number of flights f had a positive correlation with each of the output parameters, with an exception of monetary loss. The number of checkpoint lanes open l shows opposite relationships with the parameters. For instance, fewer checkpoint lanes open results in longer time to gate T_{gate} and more casualties. This makes sense, as fewer checkpoint lanes open result in longer queues and longer queuing times. This in turn results in higher passenger densities in the queuing area, resulting in a higher number of fatalities. Furthermore, it shows that both the number of check-in desks open and the presence of a BDE have a low influence on most output parameters. However, the number of BDEs does have a negative correlation with the number of casualties.



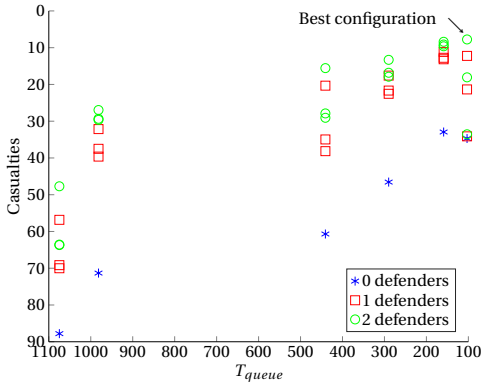
(a) The casualties in an early attack with respect to the number of employees.



(b) The casualties in an early attack with respect to the queuing time.



(c) The casualties in a late attack with respect to the number of employees.



(d) The casualties in a late attack with respect to the mean queuing time.

Figure 4.13: The number of casualties in a three flight setup in relationship to the number of employees and queuing time. Note that the axes are reversed.

We show this effect in more detail in Figure 4.13. Figure 4.13a-4.13b show the number of casualties in an early attack in relationship to queuing time and number of employees, while Figure 4.13c-4.13d show the same relationships for casualties in a late attack. Each of these results are shown for a three flight setup. It can be seen that the number of employees and queuing time do not have a strong relationship to the number of casualties in an early attack. However, in a late attack, the relationship becomes stronger. There is a strong negative relationship between the number of employees present and the expected number of casualties. This is a clear tradeoff that has to be made by airport managers, as also mentioned before. They have to choose how many more potential casualties they are willing to accept for a reduced number of employees. In contrary, the mean queuing time for passengers at the checkpoint has a positive relationship with the expected number of casualties. If we only consider these two output parameters, it is beneficial for airports to choose for configurations that lead to low casualties and queuing times. There is only one such configuration that minimizes both objectives: the

configuration with four checkpoint lanes, two intelligent defenders, and three check-in desks (see also the configuration indicated with an arrow Figure 4.13). However, this is a configuration in which 21 employees are present; only two fewer than the maximum number. Similar results are found when T_{queue} is replaced with T_{gate}. Pareto analysis can further be used to determine which configurations are optimal with respect to the defined objectives.

4.7. CONCLUSIONS & FUTURE WORK

Understanding security, efficiency and the relationships between them is essential, as airport managers regularly have to make decisions that influence these performance areas. Important decision regarding security and efficiency are often made based on experience and assumptions. This chapter introduced a novel methodology to analyze security, efficiency, and relationships between these performance areas using agent-based modeling. It combines the AbSRiM approach of Chapter 2 with a typical agent-based approach to analyze efficiency of operations. The proposed methodology is capable of analyzing security, efficiency and their relationships in detail, and therefore forms a promising way to investigate different tradeoffs between security and efficiency.

The proposed methodology was applied to a case study in a regional airport terminal. Relationships between risks regarding an IED attack and efficiency performance indicators, such as the average queuing time for passengers and number of employees, were quantified. Results show that airports should attempt to spread passengers across the available space as much as possible. Furthermore, it was found that reducing security risks and improving efficiency are not always conflicting objectives. For example, decreasing the number of passengers in the open areas of the airport is an effective measure to reduce security risks and improve different efficiency aspects.

Human behavior is far more complex than modeled in the discussed case study. More research is needed to include this complexity in the agent behavior. Furthermore, more extensive analysis, such as causal analysis [170] will be performed in Chapter 7. Another interesting possibility for further research is to integrate the proposed methodology with security games. This work could be used to determine payoff values in a security game, while the framework of security games can be used to find optimal defender policies. This is the topic of Chapter 5. Furthermore, Pareto analysis could be performed to determine a set of dominant airport configurations. Different threat scenarios, such as a shooting, and efficiency performance indicators, such as facility revenue, can also be investigated. Finally, the methodology could be generalized to identify relationships that also include other performance areas such as safety [171], resilience [172] and environmental impact [173].

5

AGENT-BASED EMPIRICAL GAME THEORY

An important method to mitigate the risk of terrorist attacks is through security patrols, as we have also shown in Chapter 4. In this chapter, we extend the three simplistic security patrol strategies of Chapter 4 by using an empirical game theory approach. Using this approach we estimate game-theoretic payoffs using the agent-based model of Chapter 4. This is an improvement over current game theory practices, as they often rely solely on expert assessment to estimate game payoffs.

This chapter has previously been published in the Aerospace journal (2020) [174].

5.1. INTRODUCTION

Ever since the attacks on the World Trade Center, airports significantly enhanced security operations, procedures, and checks. Not only security has improved, but also terrorists have adapted their way of acting. The Brussels and Atatürk Airport attacks (2016) illustrate a recent terrorist threat where publicly accessible areas of airports are the target of attack. Protecting these targets, where many people move freely, is a challenging task for security agencies because attackers do not have to face passenger or carry-on luggage checks. Additionally, limited security resources make it extremely difficult to track a terrorist in a crowded scene.

Airport security patrols are an effective method to defend against these types of attacks. However, security resources are often scarce, preventing full coverage of all targets at all times. Security patrol routes, therefore, have to be intelligently deployed by taking into account differences in the importance of targets, different attack threats, and potential uncertainty over the types, capabilities, knowledge, and preferences of attackers faced.

Game-theoretic analysis has emerged as a powerful tool to provide optimal decisions in the security domain. Game theory provides a mathematical framework to study interactions between strategic and self-interested agents who maximize the effectiveness of their actions. This makes it is appropriate to model adversarial reasoning for security resource allocation and scheduling problems [175].

One application of game theory is in the domain of security resource allocation and scheduling, included in a research area known as security games. These have shown to be successful in solving real-world security problems in which security officers deploy limited resources to protect important infrastructures against human adversaries [29, 30, 176–178]. A security game is a two-player game between a defender and an attacker. The defender wants to allocate her[1] limited resources to defend critical targets, while the attacker seeks his most favorable target to attack. Each player has a set of available actions associated with a particular payoff (also known as utility), based on the outcome of the corresponding choices within the game. Payoffs are the reward and penalties to both the defender and the attacker in a successful or an unsuccessful attack.

Commonly, game-theoretic models rely only on expert knowledge to estimate payoff values. However, these are hard to estimate, since uncertainty is intrinsic to real-world security domains. It is therefore difficult for a security expert to properly estimate payoff values for different defender-attacker interactions. Moreover, exclusive reliance on human expert assessment can be expensive, prone to human biases and restrictive [179].

Agent-based modeling and simulation is a promising technique to address the challenge of estimating payoffs. Agent-based models consist of a set of autonomous and intelligent agents who can perceive their environment and interact in the environment to solve problems, achieve goals and execute tasks. Agent-based models are particularly suitable to represent socio-technical systems, such as airports. Considering an airport terminal environment, it allows the specification of different agents, such as airport operational employees, passengers, security officers, and an attacker agent, who are able to perceive all processes happening around them and interact with each other to achieve

[1] The attacker is, following convention, referred to as “he” and the defender as “she”.

their individual goals.

Through simulations, it is possible to identify emergent patterns and relations that are not explicitly coded in the model. One example of an emergent property is the vulnerable areas in an airport terminal where an attack can lead to a large number of casualties. The identification of these vulnerable areas is of crucial importance as it indicates patrol areas where security should be reinforced.

The goal of this chapter is to improve the payoff matrices in security games, by using agent-based model results to define them. Although many security studies have focused on either agent-based modeling [180], or security games [29, 181], combining both approaches to improve security-game payoffs has not been addressed. To this end, we investigate a scenario in which an attacker aims to detonate an improvised explosive device (IED) on a publicly accessible area of a regional airport, while security agents execute patrol routes in the airport terminal. We utilize the agent-based model of Chapter 4 to determine the number of casualties of a terrorist attack and use these results to specify payoffs in a security game. This security game is then used to determine the patrol route of security officers that minimizes the expected number of casualties in a terrorist attack.

This chapter is organized as follows. In Section 5.2, we discuss relevant related work, and in Section 5.3 an overview of the case study is provided. Then, Section 5.4 provides an overview of our novel methodology, while Section 5.5 explains the proposed model in detail. The discussion of the simulations results is presented in Section 5.6, and, lastly, Section 5.7 concludes this chapter.

5.2. RELATED WORK

This section provides an overview of relevant work in the domain of security games and agent-based modeling.

5.2.1. SECURITY GAMES

Security Games have emerged as an important research domain in multi-agent systems. Over the past years, game-theoretic models have been deployed in many real-world applications: canine-patrol and vehicle checkpoints at the Los Angeles International Airport [29], allocation of US Federal Air Marshals to international flights [177], US Coast Guard patrol boats [30], and many others [176, 178].

Security games are often formulated as a Stackelberg game [29]. A Stackelberg Security Game assumes a leader (defender) and a follower (attacker). The defender must protect a set of targets as well as possible, using limited resources. The attacker aims to maximize the impact of its attack. In these games, it is assumed that the defender first commits to a (possibly randomized) security policy, while a strategic attacker uses surveillance to learn and create beliefs about the defender's strategy. After careful planning, the attacker selfishly optimizes its payoff, considering the policy chosen by the defender. The outcome of such a game is an equilibrium: a combination of strategies in which both players' strategies are best-response to each other, i.e. cannot improve their payoff by changing their strategy.

A strategy can be of two types: pure strategies or mixed strategies. A pure strategy of

an agent is one of the agent's actions, which is selected with certainty. A mixed strategy is a probability distribution over the set of actions. A mixed strategy allows for randomization which is critical in security domains as it avoids the vulnerability that comes with predictability associated with human-designed schedules. Humans are unable to produce a completely random set of events, leading to potentially predictable patterns that may be explored by an intelligent attacker [182].

Relevant to this chapter are papers that focus on security scheduling and allocation to prevent the attacker from exploiting a particular gap in the defender's patrol. One relevant application was introduced by Pita et al. [29], who computed optimal schedules that randomized road security checkpoints and terminal canine patrols. In that work, Pita et al. specify the patrolling problem as a Bayesian Stackelberg game, allowing the agent to appropriately weigh the different actions in randomization, as well as uncertainty over adversary types. However, that work did not explicitly consider spatio-temporal aspects, assuming that the attacker chooses a target to attack and is automatically at that location, without considering the time it takes to reach it. Moreover, the attacker agent could only be arrested at a target, while in real-world scenarios he can also be caught in his path from the airport entrance towards his target.

Furthermore, Prakash et al. [183] employed an empirical game theory approach. This empirical approach uses a simulation engine to model the domain area and then uses this to specify game payoffs. This methodology is similar to the one proposed in this chapter, but instead of using agent-based modeling to estimate the game payoff values, the authors use standard event-based simulation models for the same purpose. Agent-based modeling is capable of characterizing socio-technical systems, including the representation of agents' behavior and interactions which is impossible using the methodology of Prakash et al. Furthermore, their work focused on the domain of cybercrime, which is a has several differences from the airport security domain. There is a growing body of theoretical work in the field of empirical game theory, of which the work of Wellman et al. [184] and more recently the work of Tuyls et al. [185] are examples. Despite being important theoretical contributions, these do not consider human behavior and interactions and are not specific for security problems.

Other notable work is in the area of spatio-temporal security games, also known as patrol planning games. Generally, these games are played on graphs where targets are nodes and a patrol strategy is a vector consisting of defender's positions at each point in time. This approach captures the spatial evolution over time, i.e. correlates one position at time t to another position at time $t+1$. Applications range from robotic patrols [186] to green security games [33], and protection of major infrastructures such as airports [30, 187]. Fang et al. [188] focuses on protecting mobile targets, which results in a continuous set of strategies for the agents. Motivated by the domain of ferry protection, Xu et al. [189] developed a model to solve spatio-temporal games with weighted moving targets.

A recent relevant work in the domain of spatio-temporal game theory was introduced by Zhang et al. [190]. Zhang focuses on finding optimal randomize patrol strategies in a chemical cluster. In that work, potential targets are represented as nodes of a patrolling graph. The security surveys different areas by traveling in the graph and staying a certain amount of time at each node when patrolling that target. The main contribution of Zhang's work is that an optimal patrol schedule does not correspond to a randomized

fixed patrolling strategy (fixed set of different positions over time), but rather to a set of transition probabilities between nodes of the patrolling graph. In other words, their patrol schedule represents the probability that the defender performs a certain movement. Due to these advantages, we will use the work of Zhang et al. as a basis for our case study.

Despite being a field with many real-world successful deployments, security games also face multiple challenges. Those include bounded rationality [191, 192], uncertainty arising due to human dynamic behavior [193, 194], and learning in security games, with a special emphasis on reinforcement learning to identify the best defender strategy against an adaptive opponent who is able to observe defender's behavior, learn and adapt to best respond to it [195]. To partially overcome these limitations, we use agent-based modeling to define payoffs in security games.

5.2.2. AGENT-BASED MODELING

Agent-based modeling is one of the most prominent approaches to study the performance of complex adaptive multi-agent systems [196]. Complexity can be interpreted as non-linear interactions between agents (or agents with the environment), leading to unexpected emergence patterns. Agent-based modeling provides a bottom-up approach to build socio-technical systems with autonomous and intelligent agents who can perceive their environment and interact in the environment. Using agent-based models, multiple scales of analysis and multiple types of adaption and learning mechanisms can be incorporated, which is not straightforward with other modeling techniques. Additionally, it can be used to explicitly represent spatio-temporal elements of agents and the environment, which allows for a better representation of dynamic and uncertain systems.

Noteworthy work in the aviation sector includes the work of Weiss et al. [180], who developed an agent-based model for airport defense, and the work of Cheng et al. [197] who created an agent-based model to evaluate the effect of group dynamics on passenger flow during an evacuation in an airport terminal. Moreover, in Chapter 2 we introduced a novel agent-based methodology combined with Monte Carlo simulations for security risk assessment. In that work, security agents aimed to detect forbidden items in passenger's luggage while being under constant time pressure.

In Chapter 4, we developed an agent-based model to study the relationship between security and efficiency in a regional airport terminal. It focuses on a scenario where an attacker aims to detonate an IED in a publicly accessible area of a regional airport while considering efficiency indicators such as queuing time for passengers, among others. This chapter offers a promising methodology to investigate airport security and efficiency. We use the work of Chapter 4 in our case study, as described below.

5.3. CASE STUDY

This section describes the system, operational context, and scenarios under study. We study a scenario in a regional airport terminal, where a security officer patrols around four identified targets: entrance hall, check-in area, and checkpoint area. We focus on a threat scenario in which a bomb attack in the publicly accessible areas of our regional airport terminal occurs. Based on this threat, twenty attacking scenarios are modeled

varying in the period of 25 minutes with a 5 minutes increment per scenario (e.g., an attacker entering the airport within the first five minutes,...) For each attack time interval, the attacker selects one of the four identified targets to attack. That period was chosen to enclose all the attacks that may happen within the first thirty minutes since the attacker takes time to move from the airport entrance to the selected target.

Figure 5.1 illustrates the airport open publicly accessible area analyzed in this case study. The focus of our study is on airport terminal patrols, which includes processes such as check-in, facility visits, security checkpoint operations, queuing, gate processes, movement of passengers between these operations, and movement of security officers around the airport terminal. Using our empirical game theory methodology, as described below, we aim to determine the most effective patrolling route for a security officer in the airport terminal.

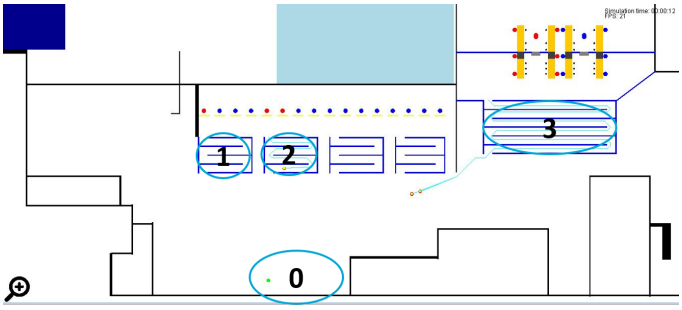


Figure 5.1: Airport layout of the open publicly accessible areas considered in this case study, with indicators for different targets. 0: Entrance area, 1 and 2: Check-in areas, 3: Security checkpoint area. For the full airport layout, refer to Chapter 4.

5.4. METHODOLOGY

The main aim of this chapter is to decrease uncertainty in game-theoretic payoff structures by estimating them using agent-based simulation results. There is a significant need to address uncertainty in both players' rewards, since key domain features like attacker behavior, that contribute to these rewards, are hard to estimate exactly by experts alone. Hence, this methodology improves on the game-theoretic payoff structures which often rely only on expert assessment. To accomplish this goal, we propose the following methodology, graphically shown in Figure 5.2.

First, we define the agent-based model. Every agent-based model requires the definition and modeling of three key entities: agents, their environment and interactions between agents and agents with the environment. In this chapter, we extend the model of Chapter 4. This model was chosen as a starting point since most airport terminal processes along with the strategic, tactical and operational behavior of passengers, defenders, and an attacker was modeled.

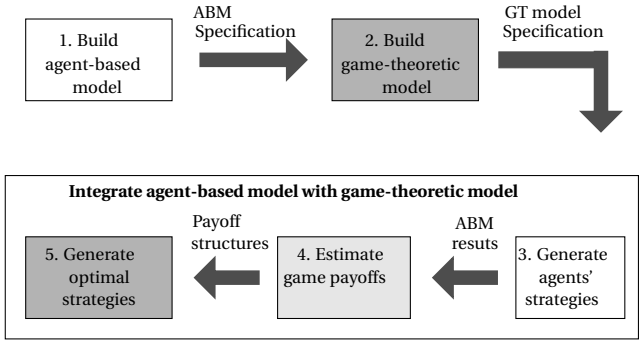


Figure 5.2: Step by step methodology followed in this chapter. Note: ABM refers to agent-based modeling and GT refers to game theory. Dark gray boxes correspond to the GT model (Step 2 and 5). White boxes correspond to the agent-based model (Step 1 and 3). The light gray box represent the interaction between the agent-based model results and the game-theoretic payoff function.

An initial evaluation of the agent-based model was performed to analyze how the airport system behaves in different scenarios. This helped to gain knowledge of critical areas with the highest agglomeration of passengers where an attack could have hazardous effects in terms of impact (human casualties). Those were deemed as potential targets. Using this information, 20 different threat scenarios (see Section 5.3) were modeled for the IED threat. The outcomes of the agent-based model simulations will later be used to specify game-theoretic payoffs.

The specification of a game-theoretic model consists of the definition of the players involved in the game, specification of the mathematical model constraints and assumptions, and the solution concept to find an equilibrium solution for both players. In this chapter, we follow the model of Zhang et al. [190]. Zhang defines a game-theoretic model aiming to select random, but strategic security patrols in a chemical cluster. This model is used, as it is a spatio-temporal game, where the set of actions available for each agent takes into consideration both spatial and temporal conditions. This is a crucial requirement in security domains since a terrorist attack can happen anytime and anywhere.

Security patrols should also be spatio-temporal, rather than only spatial, since the security officer can only detect an attacker if he is both in observation range and there is a time overlap between the attacker intrusion and the security patrol. Furthermore, this allows security officers to take different actions at distinct points in time, rather than following a predefined optimal fixed patrolling strategy. This is a great advantage as it enables better patrol randomization. The model assumes perfect rational players, i.e. reward maximizers whose strategies are best responses to each other.

The next step is to integrate both methods, which forms the core of our methodology. This step starts by generating the agent's strategies that will be simulated in the agent-based model and how they are translated to the player's set of actions in the game framework. These actions in our model consist of security patrols in the airport terminal for the defender, as well as attacks at distinct times and targets for the attacker. Each attacker-defender strategy-pair is modeled and simulated in the agent-based model so that payoffs for each combination of actions are generated.

Once all attacker and defender strategy combinations are simulated, the agent-based model outcome is computed. This output is defined as the average number of human casualties after an IED attack. This is then used as an input to define payoffs for the players in the game. The key contribution of this chapter is embedded in this step, where game-theoretic payoff matrices are enhanced with data generated by an agent-based model capable of simulating real-world events, rather than relying only on expert assessment. In this way, more objective and more robust payoff structures are incorporated in security games.

The last step of the integration process consists of solving the game (i.e. finding an equilibrium) and generating optimal strategies for both players. These results indicate the set of actions that should be taken at each time step by both players. Moreover, the optimal payoff values are computed. The proposed methodology ends with the evaluation of the optimal solution. This is done by simulating the (probabilistic) optimal defender-attacker strategy pair in the agent-based model. The resulting agent-based model metrics are gathered and used as input to compute the payoff values for both players. These are compared to the ones obtained initially after solving the game to confirm that the game-theoretic solution strategies are optimal.

5.5. MODELS

This section describes the agent-based model, the game-theoretic model and the integration of the two models. This corresponds to the first four steps of the methodology.

5.5.1. AGENT-BASED MODEL

The agent-based model environment consists of a regional airport terminal including physical objects (wall and desks), an IED (defined by its location, number of particles and mass), terminal areas (check-in, checkpoint, queuing, gate, facility and entrance area) and flights (Chapter 4). The outline of the terminal building is shown in Figure 5.1. Agents cannot obtain complete, accurate, up-to-date information about the environment's state, because it is limited by their observation range. Hence, the environment is partially accessible.

The agent architecture has three different layers: *Strategic Layer, Tactical Layer* and *Operational Layer*. In each layer, there are different modules responsible for the execution of specific actions. The *Operational Layer* comprises a perception module that is responsible for the agent's observation and an actuation module that executes actions and communications between agents. The *Tactical Layer* consists of a belief module that maintains beliefs based on observations, actions, and internal states. This layer is also responsible for the navigation and activity accomplishment. Lastly, the *Strategic Layer* is responsible for a higher level belief and for generating a *plan*: an ordered sequence of activities to be carried out by the agent.

All passengers, security agents, operational employees, and the terrorist attacker are represented by agents. Below the main characteristics of these agents are summarized. A full description of this model can be found in Chapter 4.

OPERATIONAL EMPLOYEE

Operational employees communicate a wait request to passengers when they are in their observation range. These waiting requests can be communicated to passengers completing check-in or checkpoint activities.

PASSENGER

Passengers are described by airport arrival time, level of disorientation, the suitability of their luggage, whether they checked-in already, and if they are a facility visitor. For now, it suffices to state that the level of disorientation refers to how confused the passenger arrives in the airport, while suitability of luggage attributes how well the luggage of the passenger fits with their appearance. These properties are associated with real numbers and are important indicators used in the SPOT program of the TSA [198]. In that procedure, security officers assign points to passengers to evaluate their danger to the airport: if the points accredited to a certain passenger surpasses a threshold, a secondary screening is performed. Passengers can complete different activities, namely: check-in, checkpoint, facility and gate activity.

ATTACKER

The attacker is a human agent like any other passenger and hence shares the same characteristics. However, he has one unique goal: to cause as many human casualties at the airport as possible. To achieve this objective, the attacker agent carries an IED that he intends to detonate. This activity consists of three phases: target selection, movement to target and execution of the attack. More details of the attacker agent can be found in Chapter 4.4.4.

This chapter extends the model of Chapter 4 by modeling different attacking scenarios based on an IED threat. Thus, in the first phase, the target selection is deterministic, meaning that the attacker has already selected a target to attack (from the set of 4 available options) before entering the airport. This approach implements a common assumption in security games where the attacker is assumed to have identified a breach/weakness in the security schedule through long term observation. Therefore, the attacker already knows when and where to execute his attack. In the second phase, the attacker moves from the airport entrance to the target. On his way, he might be observed by a security officer resulting in one of two events. With a probability p_{arrest}, the attacker is arrested and is not able to execute the attack, and with a probability of $1 - p_{arrest}$ he detonates the IED on the spot. Alternatively, the attacker is not observed and continues moving towards the target, where the last phase starts. Once he reached that area, the attacker detonates the IED.

SECURITY PATROLLING AGENT

A security patrolling agent can observe physical objects, passengers, and attackers in her observation radius and line of vision. The security patrolling agent has a set of strategies corresponding to patrols around the airport which she follows.

During a patrol, the security officer randomly chooses an agent within her observation range, to evaluate whether it is an attacker or not. This evaluation lasts for a certain period and is performed according to the SPOT program described previously. When the points assigned to the observed agent exceed a specific threshold, the security officer will

try to arrest the agent. If the agent is a passenger, the passenger is arrested and they both leave the airport. On the other hand, if the agent is an attacker, the security agent may arrest the attacker with a probability of p_{arrest}. If the security agent successfully arrests the attacker, the IED is not detonated. Alternatively, the attacker detonates the IED on the spot.

5.5.2. GAME-THEORETIC MODEL

We explain the spatio-temporal game of Zhang et al. [190], by describing the different components of the game. The game of Zhang et al. is a graph game, so we first translate the airport terminal layout to a graph. Based on this graph, we then specify the patrolling graph. This patrolling graph describes the possible actions and strategies of the defender.

Then, the time discretization scheme, the players in the game, and their set of actions and rewards are discussed. Finally, the solution concept is explained, along with the method to find equilibrium solutions. This section explains the theoretical basis of the game, while Section 5.5.3 later specifies how we applied this to our case study.

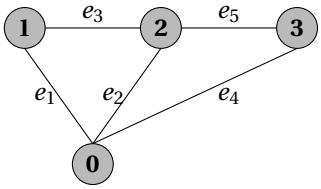


Figure 5.3: The graph model $G(V,E)$ of the airport terminal. The targets correspond to the targets as shown in Figure 5.1.

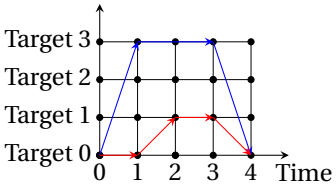


Figure 5.4: Two example strategies in a reduced version of the patrolling graph of the game. The actual patrolling graph contains nodes with corresponding times up to 1000 seconds.

AIRPORT GRAPH

The airport terminal is described by a graph $G(V,E)$ where $|V|$ represents the number of vertices and $|E|$ the number of edges, shown in Figure 5.3. Targets are modeled as vertices whereas the path between those is modeled as edges. Two important parameters are considered: time to move between targets and time to patrol a target. The time to move between targets (i.e. edge length) is constrained by the airport layout, whereas a target patrolling time is determined by the target importance for security purposes. Targets where a higher density of passengers is expected need to be patrolled more thoroughly.

PATROLLING GRAPH

Based on the airport graphic model, a patrolling graph $G_p(V_p,E_p)$ is generated. A basic example of such a graph is graphically illustrated in Figure 5.4. In this figure, we show two reference strategies for the security agent. In both cases, the security agent starts her patrol at T_0 at time 0. At this moment, she has two possible choices: either to move to T_3 (blue arrow) or stay at T_0 (red arrow). If the defender chose to move to T_3, then she only

has one option available: patrol T_3 for two time units. On the other hand, if the defender stayed in T_0 previously, her choices are confined to moving to T_1, and then staying there for one time unit. Finally, the security agent terminates either patrol strategy by moving to T_0 at time 3. These are just two representative examples of defender's strategies to illustrate the definition of a strategy, but there are many more possible strategies in this example.

A node in G_p is defined by a tuple (t,i), where $t \in [0, t_{max}]$ specifies the time and $i \in 0,1,\ldots,|V|-1$ represents a node in the airport graph $G(V,E)$. An edge from node (t_1,i_1) to (t_2,i_2) represents an action of the security agent where she moves from i_1 at time t_1 and arrives at i_2 at time t_2. A deterministic patrol strategy is a sequence of edges denoted as $e_p^1,\ldots,e_p^N$, where $e_p^i \in E_p$ is a patrolling edge, and N refers to the length of the patrolling graph, i.e. to the last patrolling edge. These patrolling graph edges have to comply to three requirements: (i) the in-degree of the start node of e_p^1 is zero; (ii) the out-degree of the end node of e_p^N is zero; (iii) e_p^i and e_p^{i+1} are connected, which means that the end node of e_p^i is the start node of e_p^{i+1}.

TIME DISCRETIZATION

The time dimension is discretized into equal time slices with the length of each time slice representing a second, with a total of t_{max} times. It is assumed that the security patrolling time and traveling time can only start at integer values of the time axis. The attacker can only start his attack at the beginning of each time slice as well. An attack lasts for a different amount of time depending on the target since the attacker takes different time from the airport entrance towards the target. Using this discretization scheme, it is possible to list all attacker strategies.

PLAYERS

The model considers a two-player game between a security agent (defender/leader) and a terrorist (attacker/follower), where both players are assumed to be perfectly rational. Consequently, both players are payoff maximizers. It is assumed that the attacker can gather information about the security patrol by long term observation, and the game is, therefore, a Stackelberg game.

STRATEGIES

The strategies for both the defender and the attacker are introduced below.

Defender At each node of the patrolling graph G_p, the defender can choose to examine that target or move to an adjacent node. These choices are described as edges in G_p. In this way, we define the security agent's strategy s_d as a set of probabilities of transitions between nodes in the patrolling graph G_p.

$$s_d = \prod_{(v_i,v_j)\in E_p} c_{v_i - v_j} \tag{5.1}$$

where $c_{v_i-v_j}$ specifies the probability of transition between node $v_i \in V_p$ to node $v_j \in V_p$, and $\prod$ represents the Cartesian product of all edges in G_p (i.e. all $(v_i,v_j) \in E_p$).

Attacker An attacker's pure strategy s_a is defined by a target to attack and a time to start the attack.

$$s_a = (t, i) \tag{5.2}$$

where $t \in [0, \ldots, t_{max}]$ represents the attack start time and $i \in \{T_0, \ldots, T_3\}$ denotes the airport target. Furthermore, the attacker is constrained to attack only one target, i.e. play a pure strategy.

PAYOFF

Payoffs are provided after every transition between nodes. Equation 5.3 gives an example of the defender payoff function.

$$U_d = R_1 \times c_1 + \ldots + R_N \times c_N \tag{5.3}$$

Each element R_i contains the payoff value associated with a particular transition between nodes c_i in the patrolling graph. The specific definition of these variables in our case study will be explained in Section 5.5.3.

R_N and c_N denote the payoff value associated with the last transition between nodes. This may lead to transitions between nodes that do not produce any outcome in the agent-based model. In this case, the payoff value associated with those transitions is assumed to be zero for both agents.

The reward value is defined based on a particular outcome arising from the agent-based model: the average number of human casualties for each transition between nodes of the patrolling graph G_p. Section 5.6 elaborates further on the reward structure outlined in this chapter. The game is defined as a zero-sum game, hence the attacker reward $U_a = -U_d$.

SOLUTION CONCEPT

To find an equilibrium solution, the model employs the concept of Stackelberg equilibrium $(s_d^*, s_a^*) = (\vec{c}^*, (t^*, i^*))$ that meet the following constraints:

$$(t^*, i^*) = argmax_{(t,i)\in S_a} u_a(\vec{c}, (t, i)) \tag{5.4}$$

$$\vec{c}^* = argmax_{\vec{c}\in S_d} u_d(\vec{c}, (t^*, i^*)) \tag{5.5}$$

As in all Stackelberg Security games, the defender (leader) first commits to a patrolling strategy $\vec{c}$, while the attacker (follower) can observe the defender's strategy and acts optimally according to it (Equation 5.4). The security officer can also determine the attacker's optimal solution, hence she choose her strategy optimally as well (Equation 5.5). Since the player's reward functions are linear polynomials of $\vec{c}$, a multiple linear programming algorithm can be used to compute the Stackelberg equilibrium solution.

In the first step, u_a and u_d are initialized for each attacker strategy. Then, a linear programming algorithm can be formulated, as shown below.

- Objective Function:

$$Max_{\vec{c}\in S_d} u_d(t^\#, i^\#, \vec{c}) \tag{5.6}$$

- Constraints:

$$\sum_{in \in \{s \in V_p | (s,V_p) \in E_p\}} c_{in-V_p} = \sum_{out \in \{e \in V_p | (V_p,e) \in E_p\}} c_{V_p-out} \tag{5.7}$$

$$\sum_{out \in \{e \in V_p | (root,e) \in E_p\}} c_{root-out} = 1 \tag{5.8}$$

$$u_a(t^{\#}, i^{\#}) \geq \alpha + u_a(t,i), \forall (t,i) \in S_a \tag{5.9}$$

$$u_a = -u_d \tag{5.10}$$

Where in, s, e, out and $root$ refer to nodes of the patrolling graph G_p, α is a small positive number and $S_a(S_d)$ is the strategy set of the attacker (defender). The $root$ nodes represents a target where the security officer starts her patrol shift. Constraint 5.7 illustrates a property of probabilities c_{s-e} that, for each intermediate node (node with both income and outcome edges) of G_p the sum of all income probabilities must equal the sum of all outcome probabilities. Constraint 5.8 describes a second property of probabilities c_{s-e} that the sum of probabilities going out from the root node equals 1. This means that the defender starts at the root node and must perform an action on what to do next. Constraint 5.9 assumes that the attacker strategy $u_a(t^{\#}, i^{\#})$ is the attacker optimal strategy. Moreover, α ensures that this model does not rely on the "tie-breaking[2]" assumption, but it is still optimal. Lastly, constraint 5.10 defines a zero-sum game. The Stackelberg equilibrium is found by getting the arguments $(\vec{c}, (t,i))$ for which Equation 5.6 is maximum.

5.5.3. INTEGRATION OF AGENT-BASED RESULTS AS GAME-THEORETIC PAYOFFS

Our integration of agent-based modeling and game theory is accomplished in three sequential steps. First, both the security and attacker strategies are generated, followed by the specification of game payoffs using agent-based model results. The last step consists of generating the optimal strategies for both players.

GENERATE AGENTS' STRATEGIES

The first step of the integration module starts with the generation of the defender and attacker strategies. We discuss each of them individually below.

Defender strategy Given the chosen time discretization of 1 second, the set of strategies for the security agent is defined as follows. The airport entrance hall is regarded as the root node from where each patrol starts and ends. Following the airport layout (see Section 5.3), the security agent can only move to adjacent nodes.

Once the security agent reaches a certain target, she stays there for a given period (patrolling time) which differs from target to target. The reasoning behind this choice was to distinguish between targets that are more security-critical to the airport. For example, a successful attack in an area with a higher density of people can lead to more

[2]The 'tie-breaking' concept assumes that, when the follower (attacker) is indifferent on payoffs by playing different pure strategies, he will play the strategy that is preferable for the leader (defender).

human casualties, thus that target should be better monitored. The patrol times as used in this chapter are shown in Table 5.1. These are based on initial experiments with the agent-based model and expert input. To include uncertainty related to disruption on security patrols, the time spent at the targets is according to a Normal distribution. When the patrolling time has passed, the agent has to move to another adjacent node.

Table 5.1: Patrolling time for each target in seconds. Normal distributions are characterized by their mean (first parameter) and the standard deviation (second parameter).

T_0	T_1&T_2	T_3
$\mathcal{N}(60,30)$	$\mathcal{N}(240,30)$	$\mathcal{N}(360,30)$

Using the layout of the airport graph (see Section 5.3), and the patrolling times of Table 5.1, we generated all possible deterministic patrolling strategies that can be executed within 1000 seconds. By performing a brute force search, we identified a total of 66 different patrol strategies that fit these criteria. This corresponds to a total of 596 different patrolling graph edges (movements).

Attacker strategy We considered twenty actions for the attacker. These actions have a five-minute interval uncertainty, for a period of twenty-five minutes for each of the identified targets ($T_0, \ldots, T_3$). The attacker agent may be caught in his path towards the target, even if both the security agent and the terrorist agent are not in the same area, but the latter is within the observation range of the former. This is a closer representation of reality than the standard game-theoretic formulation, as security officers can observe further than just their current target. This ensures that more realism is included than would be possible in the game-theoretic formulation alone.

SPECIFY PAYOFFS USING AGENT-BASED RESULTS

After generating the set of strategies for both agents, the next step is to specify the payoffs based on the agent-based model outcomes obtained from the previous step. As mentioned above, we focus on the average number of human casualties.

The number of casualties is estimated as follows. For each attacker and defender strategy, a consequence function that assesses the number of human fatalities is calculated for the simulated threat scenario. This function is used to determine the consequences for a simulation run of our agent-based model. Monte Carlo simulations are executed to evaluate the average number of casualties based on a set of N simulation runs. This average number of casualties corresponds to the conditional risk R_c, as defined in Chapter 4.

Following the generic payoff function specified in Section 5.5.2, first, we define $\vec{R}$ as the average number of casualties for each transition between nodes. F_i refers to the average number of casualties obtained when the defender performs the movement corresponding to the probability that the defender performs move i, denoted as c_i. Equation 5.11 shows the used payoff function.

$$U^d_{target,time} = -(\mathrm{F}_1 \times c_1 + \ldots + \mathrm{F}_{596} \times c_{596}) \tag{5.11}$$

The final game-theoretic model consists of 11,920 payoff values generated from the combination of 20 different attacker options and 596 security patrolling movements.

The above payoff function uses different F values for each attacker strategy combination. Therefore, 20 different payoff functions were defined $U_{0,0}, \ldots, U_{3,4}$ for each player. The target index varies from 0 (T_0) to 3 (T_3). The time index varies from 0 (attack enters the airport within the first 5 minutes) to 4 (attack enters the airport between the 20 to 25 minutes). Moreover, the defender's reward has a negative sign to penalize her for each human fatality. We assumed a zero-sum game, thus the attacker reward has the opposite value of the defender.

VERIFICATION OF OPTIMAL STRATEGIES

In the last step of our methodology, we generate the optimal attacker and defender strategy using the generated payoff values. These optimal strategies are simulated in the agent-based model and the outcomes of this simulation are compared to the ones obtained with the initial simulation assessment. The results are expected to be similar to positively verify the optimal game-theoretic solution. It is important to note that this does step does not correspond to validation. Validation of the strategy can be done using real-life tests, but is known to be difficult in practice [92, 93].

5.6. EXPERIMENTS & RESULTS

Experiments performed with the above model are described in this section. First, the agent-based model experimental setup and results are described. Then, game-theoretic results are shown. Both the game-theoretic rewards are detailed along with the Stackelberg game solution for a generated security probabilistic patrol route and a fixed patrol route. Finally, the optimal strategies obtained for a probabilistic patrol route are subjected to evaluation.

5.6.1. EXPERIMENTAL SETUP

The agent-based model contains a set of parameters in the experiments, of which the important ones are shown in Table 5.2. Apart from the number of simulations runs N, the parameters in this table were calibrated in Chapter 4. Additional parameter values of the model may be found in that chapter as well. It is important to note that all flights are defined with the same departure time, as commonly happens at regional airports. The model was implemented in the AATOM simulator, a Java-based open-source agent-based airport terminal operations simulator [130].

The number of simulations required to obtain a proper estimate of the distribution of the model output was determined based on the coefficient of variation. Figure 5.5 shows the coefficient of variation for two different attacker-defender strategy pairs. It shows that the coefficient of variation tends to stabilize between 300 and 400 simulations. Consequently, the number of simulations N was set to be 500 to ensure a proper estimation of the model output for all attacker-defender strategy pairs.

Table 5.2: Agent-based model parameters.

Parameter	**Value**
Simulation parameters	
Simulation runs	500
Airport and flight parameters	
Flight departure time	7200 sec
Number of flights	3
Number of open checkpoint lanes	2
Number of open check-in desks	3
Agents parameters	
Proportion passengers check-in	0.5
Check-in time	*Norm(60,6)* sec
Checkpoint time	*Norm(45,4.5)* sec
Observation radius	10 m
Security arrest probability	0.8

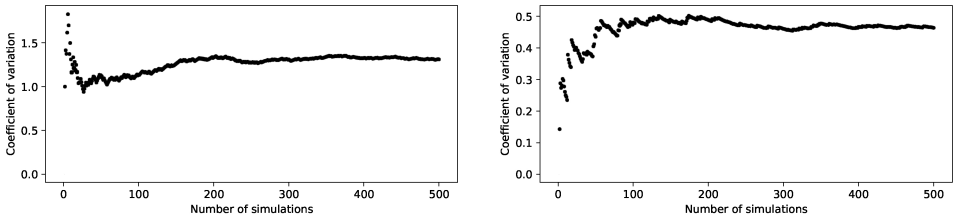


Figure 5.5: Coefficient of variability varying with the number of simulation runs

5.6.2. AGENT-BASED MODEL RESULTS

Table 5.3 shows a selected subset agent-based results associated with a particular defender transition between two nodes of G_p (i.e. a movement) and an attacker strategy (target, time).

Table 5.3: Illustrative example of agent-based outcomes. Cas. denotes the average number of casualties. Eff. represents the efficiency of the patrol for each movement, which is defined as the percentage of simulation runs in which the defender successfully arrested the attacker.

Start Node (Time (s), Target)	**End Node** (Time (s), Target)	**Att. Strategy** (Target,Time (min))	**Cas.**	**Eff. (%)**
(0, T_0)	(6, T_2)	(T_0;0 – 5)	4.27	0
(6, T_2)	(246, T_2)	(T_0;0 – 5)	2.194	21.72
(1933, T_3)	(1964, T_0)	(T_0;0 – 5)	-	-
(0,T_0)	(31, T_3)	(T_3;0 – 5)	0	100
(0, T_0)	(31, T_3)	(T_0;20 – 25)	-	-
(1582, T_3)	(1942, T_3)	(T_3;20 – 25)	11.615	7.69

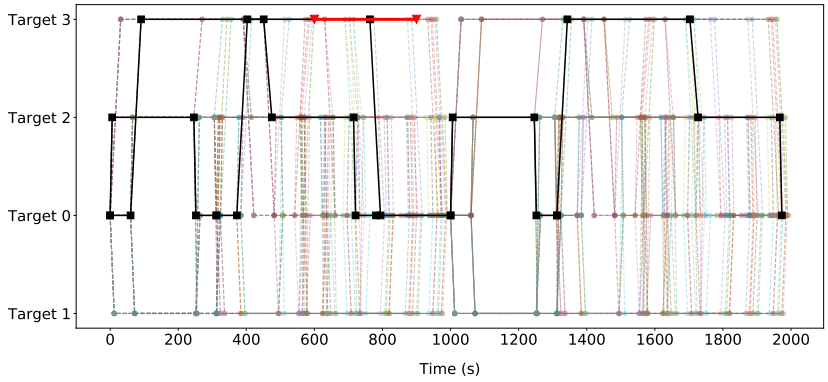


Figure 5.6: The optimal patrolling strategy over time and the attacker's best response. The black lines symbolize the defender's optimal (probabilistic) patrolling strategy. Each line segment (each movement) has an associated number representing the probability that the defender will do that movement. The red line illustrates the attacker's best response strategy. Note that the red line only covers T_3 for the sake of visualization simplicity. In reality, the attacker enters the airport through its entrance (T_0) and takes some time to arrive at the target destination. Lastly, the remaining colors with lower opacity represent all possible movements that may have been chosen by the security officer.

From the agent-based model simulation, two scenarios can occur. First, for a particular defender movement and attack strategy, an interaction between both agents occurs. This interaction may be a successful attack or a successful arrest. However, it may also happen that for a particular defender movement and attack strategy, no interaction between both agents occurs. The later happens since the time of the defender movement does not coincide with the attack interval. For instance, movement $(1933, T_3)$ to $(1964, T_0)$ will not lead to a defender-attacker interaction when the attacker attacks T_0 within the first five minutes. Later in the game formulation, these cases will have a zero payoff value associated. The reasoning behind this choice was to assign a neutral payoff value for both players in the cases where they did not interact.

5.6.3. GAME-THEORETIC RESULTS

Based on the results of Section 5.6.2, we describe the game-theoretic solution, focusing on rewards and strategies for each player.

STACKELBERG GAME SOLUTION

Figure 5.6 shows a graphical representation of the Stackelberg Equilibrium solution of the game. The black lines symbolize the defender's optimal patrolling strategy, i.e. the non-zero probabilities for each of the actions of the defender. Each line segment has an associated number representing the probability that the defender will take that action, which is not shown in the figure. For instance, at time 0, the defender will move to check-in area (T_2) with a probability of 0.129. Alternatively, the defender also has an option to stay at the airport entrance (T_0) for 60 seconds with a probability of 0.871.

An interesting result of the generated strategy is that T_1 is not patrolled at all. This

target is covered by patrolling T_2, which is close to T_1. The area around T_1 is in the observation radius of the defender when she is in T_2. Furthermore, T_2 is a more central target, and can, therefore, be reached faster from the other targets.

The attacker's best response strategy is to attack the checkpoint area (T_3), entering the airport at a time between ten to fifteen minutes, illustrated in Figure 5.6 as a red line. Note that the red line only covers T_3 for visualization simplicity. In reality, the attacker always enters the airport through T_0 and takes some time to arrive at the target.

Table 5.4 shows the agent-based model results associated with the patrol movements corresponding to the optimal patrol strategy. Only the patrol movements that lead to a defender-attacker interaction are shown. It is important to note that there is one movement for which the period does not coincide with the attacker entering time of 10 to 15 minutes. This occurs since the attacker takes time to reach his target destination in a crowded airport. All other movements that are part of the optimal strategy, but are not present in Table 5.4, are those where there was no interaction between both players. The payoff associated with those movements is set to zero.

Table 5.4: Agent-based results associated with the movements of the defender that are part of the optimal patrol strategy.

Start Node (Time (s), Target)	**End Node** (Time (s), Target)	**Prob.**	**Cas.**	**Eff. (%)**
(403, T_3)	(763, T_3)	0.129	2.286	72.67
(763, T_3)	(794, T_0)	0.129	1.540	78.94
(794, T_0)	(1000, T_0)	0.129	6.083	41.35
(475, T_2)	(715, T_0)	0.871	1.427	70.68
(721, T_0)	(781, T_0)	0.871	2.284	72.59
(781, T_0)	(1000, T_0)	0.871	5.430	47.70
(1006, T_2)	(1246, T_2)	1	10.789	0

When the probability value and expected number of casualties associated with each movement (as outlined in Table 5.4) are introduced in Equation 5.11, the optimal reward values for the defender and attacker are obtained.

$$\begin{aligned} U^d_{3,2} = -(&2.286 \times 0.129 + 1.540 \times 0.129 + 6.083 \times 0.129 \\ &+ 1.427 \times 0.871 + 2.284 \times 0.871 + 5.430 \times 0.871 \\ &+ 10.789 \times 1) = -20.03 \end{aligned}$$

The attacker reward is the negation of the defender's reward, i.e. $U^a_{3,2} = 20.03$. Figure 5.7 shows every attacker's reward value associated with each attacker's strategy against the defender optimal (probabilistic) patrolling strategy. These are computed similarly as the one illustrated in the Equation above.

These results show that attacking the security checkpoint (T_3) between 5 and 20 minutes yields the highest reward for the attacker when comparing to attacking other targets within the same time frame. This may be explained as follows. Passengers arriving in previous time intervals finished their check-in activity and are going towards the security checkpoint, leading to a higher density of people around that area. Thus, if the attack

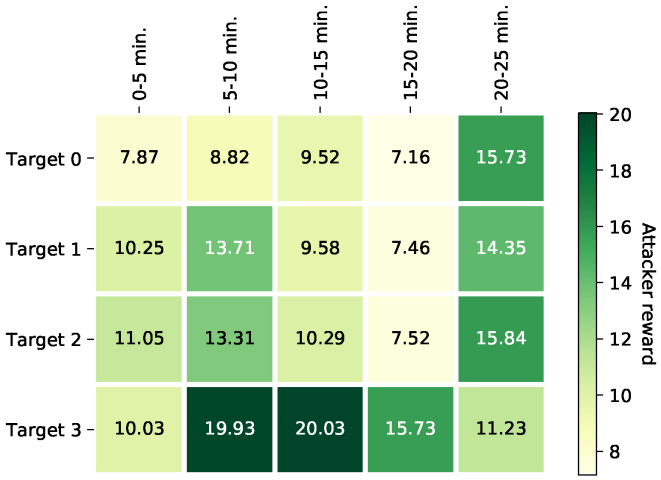


Figure 5.7: Attacker reward values for each attacking strategy, when the defender performs the optimal patrol illustrated in Figure 5.6.

is successful, its impact would be large. This is not the case for all the other targets since there are passengers who did the check-in online and go straight to the T_3 which results in a lower concentration of passengers around those areas. Moreover, an attack within the first five minutes has a lower consequence since fewer people are at the airport terminal. The airport gets more crowded as time gets closer to the flight departure time.

It is also worth noticing that an attack on targets T_0, T_1 and T_2, at the latest time interval yields higher rewards for the attacker when comparing to other periods. This is the case, as the number of people entering the airport considerably increases during that time interval which results in a higher concentration of people in those areas. This increase results from the fact that as time passes by, it gets closer to the flight departure time and therefore more people start entering the airport. As mentioned earlier, the latter increases the chances and consequences of a successful attack.

By comparing the results of Figures 5.6 and 5.7, the defender's optimal strategy choice may be justified as follows. From Figure 5.7 it can be observed that the attacker reward by attacking T_3 while entering the airport between five to ten minutes yields the second-highest value. Therefore, the defender favors the patrol of that area during the corresponding period. The latter observation may be the reason why the defender's optimal strategy does not contain additional movements that patrol the optimal attack target at the optimal attack time (between 10 to 15 minutes).

However, the optimal defender strategy does not coincide with the attacker target for the entire attack time interval. Namely, the defender choice after leaving T_3 is to go either to T_2 or T_0, and, eventually, staying there until a new patrol starts. These results can be explained by the fact that the attacker, in his path to T_3, may be detected by the defender if she is either at check-in area 2 (T_2) or the airport entrance (T_0).

These results show that the optimal security patrol gives special emphasis to high-

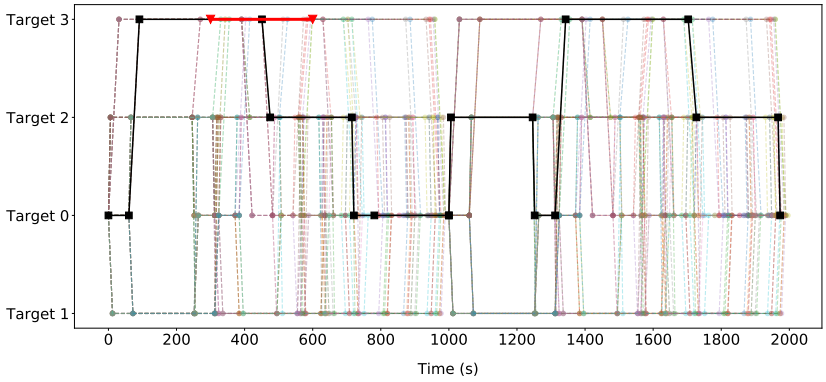


Figure 5.8: The deterministic optimal patrolling strategy over time and the attacker's best response. The black lines symbolize the defender's optimal patrolling strategy. The probability associated with each movement is 1.

impact areas, such as the security checkpoint, to reduce the total security risk. This is an improvement over the more simplistic strategies as shown in Chapter 4.

DETERMINISTIC PATROLLING STRATEGY

In the current patrolling practice, the security officer may follow a deterministic patrolling strategy. In a deterministic patrolling strategy, the probability that an action is taken is constrained to be either 0 or 1, rather than a probabilistic value between 0 and 1. To investigate this scenario, we follow the same procedure illustrated in Section 5.6.3, but with the aforementioned constraint where the decision variables are either 0 or 1. Figure 5.8 illustrates the optimal strategy for both agents. The red line represents the attacker's optimal strategy, while the black line denotes the defender's best response. It is interesting to observe that for a fixed patrolling strategy, the attacker's best response remains to be T_3, but changes the attacking time interval to a time range between five to ten minutes. This result shows that attacking T_3 during the time interval between five and ten minutes yields a high payoff for the attacker. Therefore, it reinforces the defender's patrol choice of covering that target during that time interval in the probabilistic patrol strategy, as discussed in Section 5.6.3.

Results, as shown in Figure 5.8, show that if the defender would follow the fixed patrolling route and the attacker plays his best response rewards for the defender and the attacker are -21.417 and 21.417 respectively. This shows that by randomizing over different movements at different times, the defender can generate strategies that are effective against a potential terrorist attack. These conclusions can help airport managers design security procedures.

5.6.4. VERIFICATION

Finally, the last step of our methodology is to simulate the optimal game-theoretic defender-attacker strategy pair in the agent-based model and compare the results with the ones

resulting from the initial agent-based simulations. In this step, we can verify if the obtained solutions from the game-theoretic model are still valid in the agent-based model.

To do this, we simulated the optimal probabilistic defender patrolling strategy in the agent-based model. A total of 2000 simulations were executed. We simulate the obtained defender strategy against all attacker strategies (i.e. all target-time combinations). Figure 5.9 represents the average number of casualties per attacked target per time when the defender performs her optimal probabilistic patrol strategy. Note that Figure 5.9 is different from Figure 5.7 as the prior represents the optimal reward value. This is a function of the average number of casualties and the probability of executing the optimal movements.

From Figure 5.9 it can be noted that the number of casualties when the attacker attacks T_0 is lower than at other targets. The airport entrance is a target where people do not agglomerate as intensively as they do at the check-in areas (T_1 and T_2) and the checkpoint (T_3). Furthermore, the highest patrol efficiencies occur at the optimal attack target (T_3). This reinforces the choice of the defender's optimal strategy since it achieves a higher arrest rate against the optimal attacker target.

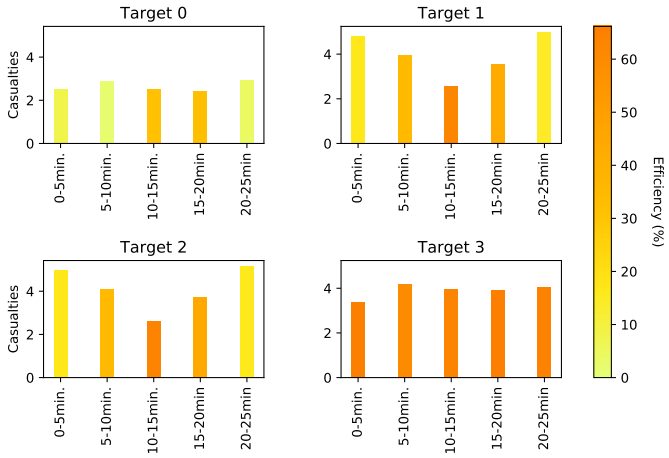


Figure 5.9: The number of casualties per attacking strategy against the optimal defender's strategy.

In order to understand the variability in the number of casualties in each simulation run, a boxplot of the results in Figure 5.10 was generated. This figure shows that the number of casualties in T_0 is lower than those on the other targets, while T_3 yields higher casualties values on average. This is due to the fact that the passenger density at the airport entrance is smaller than the check-in areas, which is smaller than the security checkpoint. T_3 also yields the highest number of casualties that occurred in one simulation. This is a striking result because it indicates that a successful attack leading to a higher number of human fatalities may happen in reality, even if the security is executing the optimal patrol strategy. Therefore, it can be concluded that despite the optimal security strategy having higher patrol arrest rates at T_3, the potential consequences of a

successful attack there are highest.

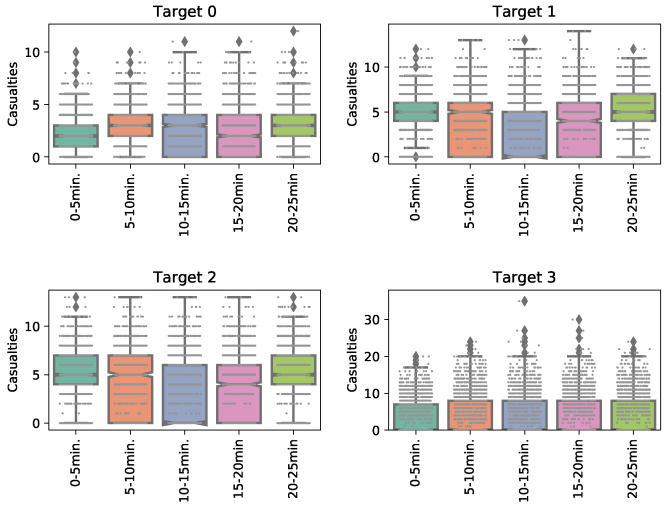


Figure 5.10: Number of casualties per target per time in each simulation run. Note that the axis scales are different among targets. Two outliers (40 and 56 casualties), at T_3 between 20 and 25 minutes, were omitted from the figure to enhance readability.

Finally, Table 5.5 shows the new agent-based model results associated with the patrol movements corresponding to the optimal probabilistic patrol strategy. Therefore, if the probability value and casualty value associated with each movement (in Table 5.5) are introduced in Equation 5.11, it is possible to compute the defender and attacker optimal reward values.

Table 5.5: Empirical results for the optimal patrolling strategy in the verification step. All other movement probabilities are zero.

Start Node (Time (s), Target)	**End Node** (Time (s), Target)	**Prob.**	**Cas.**	**Eff. (%)**
$(403, T_3)$	$(763, T_3)$	0.129	2.667	73.56
$(763, T_3)$	$(794, T_0)$	0.129	1.976	76.12
$(794, T_0)$	$(1000, T_0)$	0.129	5.602	43.08
$(475, T_2)$	$(715, T_0)$	0.871	1.413	71.26
$(721, T_0)$	$(781, T_0)$	0.871	2.096	82.61
$(781, T_0)$	$(1000, T_0)$	0.871	6.721	53.19
$(1006, T_2)$	$(1246, T_2)$	1	9	0

$$U^d_{3,2} = -(2.667 \times 0.129 + 1.976 \times 0.129 + 5.602 \times 0.129 + 1.413 \times 0.871 + 2.096 \times 0.871 + 6.721 \times 0.871 + 9 \times 1) = -19.22$$

The attacker reward is the opposite of the defender's reward, i.e. $U^a_{3,2} = 19.22$. If we compare these values with the one achieved by the game-theoretic model (-20.030/20.030) we conclude that the payoffs are close, which verifies the proposed strategy.

5.7. CONCLUSIONS & FUTURE WORK

This chapter introduced a novel methodology to improve game-theoretic solutions by specifying payoff values based on the outcomes of an agent-based model. These payoff values are often defined by relying on expert assessment alone, which can be prone to errors and human biases. Our empirical game theory methodology improves current game-theoretic formulations by relying on data generated by the agent-based model of Chapter 4.

The methodology was applied to a case study in a regional airport terminal for an improvised explosive device threat. Results show that by strategically randomizing patrol routes, higher expected rewards for the security officer are achieved. This leads to a reduced number of expected casualties in an improvised explosive device attack. Furthermore, it was found that by allowing the defender to make probabilistic decisions at different time points, a higher reward is obtained when comparing to a fixed optimal patrolling strategy. This supports the results of Zhang et al. [190]. Results further show that the optimal security patrol gives special emphasis to high-impact areas, such as the security checkpoint, to reduce the total security risk. This is an improvement over the more simplistic strategies as shown in Chapter 4.

This chapter can be extended in several directions. Firstly, different strategies with less restrictive constraints may be investigated to understand if better rewards can be achieved. For instance, time spent at each target may be varied more to understand the influence of that parameter on the current model. Secondly, research on human behavior can be included to incorporate more complex behavior in the agent-based model. In addition, the game model can also be improved to incorporate different human rationality models [191]. Lastly, uncertainty related to potential patrol disruptions may also be further investigated to improve the current game-theoretic model [194].

6

USING CAUSAL DISCOVERY TO DESIGN AGENT-BASED MODELS

The AbSRiM approach, as proposed in Chapter 2, is centred around agent-based models. However, designing agent-based models is a difficult task. It is a creative process, and the quality of the model ultimately depends on the knowledge and skill of the modellers. Some guidelines exist to aid modellers in designing their models, but they generally do not include specific details on how the behavior of agents can be defined. We, therefore, propose the AbCDe methodology in this chapter, which uses causal discovery algorithms to specify agent behavior. The methodology combines important expert insights with causal graphs generated by causal discovery algorithms based on real-world data. These causal graphs represent the causal structure among agent-related variables, which is then translated to behavioral properties in the agent-based model.

This chapter is currently under review at the Simulation Modelling Practice and Theory journal.

6.1. INTRODUCTION

Agent-based models are important tools to understand the world around us. These models aim to replicate complex systems by specifying the behavior of actors, leading to a better understanding of them. Agent-based models have shown to be useful in a variety of areas, ranging from urban planning [46] to ecology [48].

Designing agent-based models is not a trivial task. It is a creative process, and the quality of the model ultimately depends on the knowledge and skill of the modelers. Some guidelines exist to aid (new) modelers in their model development, and they share some similarities [50–55]. Most of these guidelines are quite high-level, and do not go beyond a description of which elements have to be defined.

A notable exception is the 'overview, design concepts, and details' (ODD) protocol, which has been used widely in literature [56, 57]. It provides a detailed set of steps, along with guidelines, to design agent-based models and individual-based models. However, even this extensive protocol does not include specific guidelines on how the behavior of agents can be defined. It remains up to the creativity and expertise of the modeler to determine how this behavior is specified.

With more and more data becoming available over the last decades, methods to interpret and understand this data became better as well. The field of data analysis is concerned with finding patterns and relationships in a dataset. Numerous methods to find these patterns and relationships exist, of which regression, neural networks, and clustering are three examples [199].

A particularly promising method to find relationships between variables is that of causal discovery [67]. Using causal discovery algorithms, a causal graph is generated based on available data. These causal graphs show the causal relationships between variables and identify structure in the dataset. With the right dataset, causal discovery algorithms provide insights into the observable behavior of agents and their results in the environment. These insights can then be used to specify the behavior of agents in an agent-based model, potentially leading to a better model.

In this chapter, we propose AbCDe, a novel methodology that aids the development of agent-based models using causal discovery. The methodology combines causal graphs with insights of experts to specify the behavioral properties of agents. This provides the modeller with a more structured approach towards specifying the behavioral properties of agents, reducing the dependency on experts alone. The methodology is applied to a case study in the security checkpoint, in which a new concept of operations is evaluated. In this concept, a service lane processes passengers that are expected to be slow, and the other open lanes process the remaining passengers. This concept of operations is projected to improve overall throughput of the system, as faster passengers do not have to wait for slower passengers in front of them.

This idea of using data-driven methods to design agent-based models has been explored by Kavak et al.[58]. In that work, behavioral properties of agents are learned from data by applying machine learning techniques, such as support vector machines and decision trees. While these more traditional machine learning techniques are effective tools to learn behavioral properties, they do not reveal the structure of relationships between variables related to agents. A particularly promising method to reveal this structure is that of causal discovery. We follow a similar approach to that of Kavak et al., but

focus on using causal discovery instead of traditional machine learning techniques.

Kvassay et al. [77] provide a method, based on causal partitioning, to analyze causal relationships relating to emergence in agent-based models. These causal partitions specify the relative importance of influencing factors on emergent properties, which helps to understand these properties better. This work focuses on analyzing agent-based model behavior, and does not cover designing them.

Guerini and Moneta [78] cover the topic of agent-based model validation. They estimate time-series of economic models using structural vector autoregressive (SVAR) models. Using causal discovery algorithms they generate two SVAR models: one based on results generated by the designed agent-based model, and the other based on actual data. When the two SVAR models are similar enough, they agent-based model is consider validated. This work only covers agent-based model validation, and does not cover designing agent-based models.

This chapter is structured as follows. The AbCDe is outlined in Section 6.2. Then, the case study is outlined in Section 6.3, and a discussion is provided in Section 6.4. Finally, the chapter is concluded in Section 6.5.

6.2. METHODOLOGY

This section outlines the novel Agent-based Causal discovery Design methodology, called AbCDe, which is used to design agent-based models with causal discovery algorithms. The methodology contains five steps, which are graphically outlined in Figure 6.1.

The methodology exploits the ever-growing availability of data to design agent-based models. Using data on behavior of agents, a causal graph is generated using causal discovery algorithms. This graph is then, with the aid of experts, translated into behavioral properties of agents. These properties ultimately determine the dynamics of the model, leading to insights into the phenomenon that is modeled.

Causal discovery algorithms provide a more structured method to develop agent-based models than relying on experts alone. However, experts are still needed in many aspects of the methodology to ensure that the model is of high quality. This combination of causal discovery algorithms and experts can lead to better models than models created by experts alone. Each of the steps of the methodology is outlined in detail in the subsequent sections below.

6.2.1. PURPOSE, RESEARCH QUESTION AND HYPOTHESIS

As in any modeling study, the purpose of the model should be defined first, as a general-purpose agent-based model is not effective [196]. Based on the defined purpose of the model, all other modeling decisions are made. Example purposes are for instance: 'to explore the causes of the indirectly observed Anasazi population dynamics in the Long house Valley in Arizona between 800 and 1400' [200, 201] and 'explore the relationship between how tolerant individuals are of the opposite group and how segregated neighbourhoods are, when individuals express intolerance by moving' [202, 203].

Based on the purpose of the model, a (set of) research question(s) is formulated. These research questions are more specific than the purpose and further limit the elements that need to be modeled later. Following the same examples as above, these re-

search questions can be: 'does the environmental variability itself explain the abandonment of the Anasazi of the Long House Valley?' and 'why do members of two different groups separate into different neighbourhoods?'.

After defining the research question(s) of the study, hypotheses have to be defined as well. These hypotheses state the expected answers to the research questions. In the Anasazi an hypothesis could be: 'environmental factors only cannot explain the sudden abandonment of the Long House Valley', while in the segregation model this could be: 'small preferences of individuals to live near members of the same group lead to largely segregated areas'.

6.2.2. SCOPE AND CONCEPTUAL MODEL

Based on the first step of the methodology, the scope of the model is determined. This specifies what elements will be included in the model and what will not be included. In the Anasazi example, households and their behavior fall in the scope of the model, but not individuals. Also extreme weather events are outside the scope of the model, but soil quality is included. In the segregation model, housing prices fall outside the scope of the

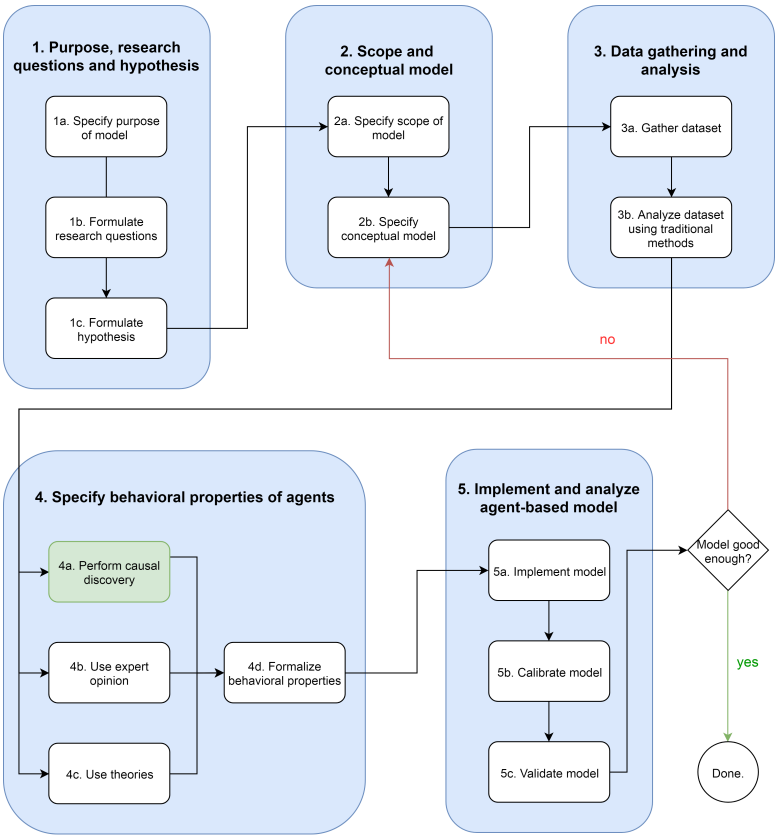


Figure 6.1: The methodology used in this chapter.

model, but individual preferences on the type of neighbors is included in the model.

Once the scope is clarified, a conceptual model is formalized. The conceptual model forms the basis for the remainder of the methodology. In this conceptual model, agents are identified first. An agent, in this chapter, is defined as an entity that perceives its environment through sensors and acts upon that environment through effectors [204]. In this step, we specifically focus on the identification of the agents to be modeled, along their characteristics and the behavior that they can exhibit. Only the higher-level behavior that the agents exhibit is specified (i.e. what the agent can do); the full specification of the behavior (i.e. how the agent does it) will be done in step 4 of the methodology.

For the running Anasazi example, a household is the only agent type that is modeled. A household has a specific age (in years), a location and specific nutritional requirement. They are able to harvest food, and split into two households when the fertility age has been reached. In the segregation example, one agent is identified: a family. A family is of a specific type (A or B), and lives at a certain location. It has a preference to live close to at least a certain percentage of families of the same type, and can move to another location.

When the agents are identified, the environment of the model is specified. The environment, as used in this chapter, comprises non agent objects, which agents can observe and act upon. These objects may be related to each other. In the Anasazi model, the environment consists of cells of 100 by 100 meter. Each cell has a soil quality and can contain water. A cell has (changing) weather conditions and can be inhabited by a set of households. The environment of the segregation model contains cells that can be occupied by a single family, or unoccupied. These cells can also be neighbors of each other.

The conceptual model also serves as a basis for the terminology used when designing the rest of the model. It can be considered to extend the conceptual model with an ontology, which describes the model, related concepts and relationships between them.

6.2.3. DATA COLLECTION & ANALYSIS

After identifying the agents and specifying the environment, data is collected about the behavior and characteristics of the agents in the model. This data is obtained by observing agents, their actions, and the consequences of these actions in the real world. This will later be used to specify behavioral properties of agents.

Depending on what is modeled, different types of data can be collected. The collected data that will be used to generate behavioral properties is always on the agent-level (and not population-level), and should therefore contain as much detail about the characteristics of the agent (as defined in the conceptual model of the previous step), its behavior and the results of this behavior. Data is therefore in the form of characteristics of agents, actions performed by agents, effects of agent actions on the environment and effects of agent actions on other agents.

Collected data must be quantitative and not qualitative to be useful for the causal discovery algorithms as used in the next step. It is also important to gather data in different circumstances, at different times, so that a complete picture of the agent behavior can be obtained.

Qualitative data and population-level data can be useful in other aspects of model

development. For instance, it can be used to calibrate the population size and distribution, or specify parts of the model of which no quantitative data is available.

In the Anasazi example, data cannot be collected anymore as the investigated period is centuries ago. However, archaeological data contains information on locations of settlements, movement behavior of households, and their nutritional requirements. In the segregation example, data could be collected by surveying families of different groups about their reasons to move to another neighborhood. Furthermore, official government data on movement of families and compositions of neighborhoods can be used as well.

The collected data is then analyzed following standard data analysis techniques, such as clustering, regression and statistical tests. This provides early insights into the behavior of agents, and will be useful for the next step of the methodology.

6.2.4. BEHAVIORAL PROPERTIES

We informally define behavioral properties as all aspects of the agent that relate to its actions, including interactions, and communications. We explicitly exclude cognitive properties of agents, as these are often not observable and therefore cannot be captured in a quantitative dataset. Behavioral properties are formalized based on two sources: causal discovery and expert input. These aspects are discussed in detail below, combined with a discussion on how to translate them to behavioral properties.

CAUSAL DISCOVERY

Causal discovery algorithms are used to infer a causal structure from available data. Several methods have been proposed to generate causal graphs, of which the most important types are constraint-based methods [72–74] and score-based methods [71, 205]. In the constraint-based category, the PC algorithm [73] is very popular, while in the score-based category, the GES algorithm [205, 206] is frequently used in literature.

A causal discovery algorithm is used on the dataset that was collected in the previous step. Before applying the algorithm, the data has to be preprocessed. This preprocessing is done to ensure that only agent behavior is found, and not emergent effects. These emergent effects should be part of the model, but not explicitly coded into the behavior of agents. It should emerge from the behavior and interaction of agents in the model. In the Anasazi example, the total number of households over time is an emergent effect that should not be considered by the causal discovery algorithm. In the segregation example, the time it takes for an area to be segregated is an emergent effect that should not be considered by the causal discovery algorithm.

Furthermore, the dataset has to be organized such that a single graph for a single agent is produced. Data of other agents can be included in the dataset for the agent under consideration, so that observable behavior, such as communication and alterations of the environment, can be found by the causal discovery algorithms as well.

After preprocessing, a causal discovery algorithm is applied to the dataset, leading to a causal graph representing the behavior of an agent in the model. The generated graphs relate characteristics of agents to exhibition of their behavior by means of including an arrow between them. Results of behavior of other agents, or properties of environmental objects are included in the graph following the same standard.

EXPERT & THEORY INPUT

After generating the causal graphs, an expert provides input for two purposes. First, the expert checks the graph that was generated for inconsistencies with their knowledge and the original data analysis that was performed in the previous step. These inconsistencies are then fixed in the graph.

Second, the expert provides additional insights based on theories from literature or their experience. These insights can be used to compensate for missing data in the dataset, and provide another means to specify behavioral properties in the next step.

SPECIFICATION OF BEHAVIORAL PROPERTIES

After obtaining both the causal graph and the input of experts, the behavioral properties are specified with the aid of experts. These properties can be obtained from the graph (updated by the expert) by selecting a variable to be used as a behavioral property, and using its parents as building blocks to specify the behavior. We use Figure 6.2 as a fictional example to illustrate this process. In this figure, a part of the causal graph of the Anasazi example is visualized; specifically the variable $farm\_yield$, along with its parents.

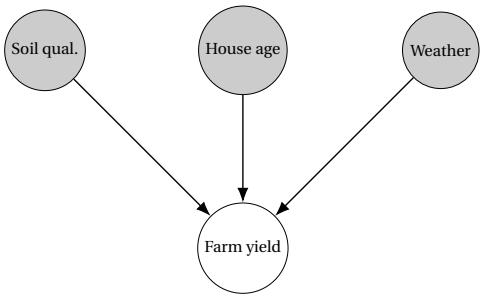


Figure 6.2: A fictional example of a part of the causal graph of the Anasazi example.

Different methods exists to translate a variable into a behavioral property. The first option for specifying a behavioral property is by using conditional distributions. Following this approach, distributions for the farm yield are obtained for different combinations of parent values, by fitting distributions over the available data. An example of this is shown in Equation 6.1 below.

$$farm\_yield = \begin{cases} N(soil\_qual \times 20, 25) & 0 < age < 10, weather = bad \\ N(soil\_qual \times 40, 15) & 0 < age < 10, weather = good \\ N(soil\_qual \times 25, 10) & age \geq 10, weather = bad \\ N(soil\_qual \times 48, 23) & age \geq 10, weather = good \end{cases} \tag{6.1}$$

In this equation, the $farm\_yield$ follows different Normal distributions based on the age of the household and weather conditions. The main advantage of this approach is that the behavior of the agent best represents the data, leading to a better model. However, this requires large amounts of data to be effective, and leads to models with a high number of parameters.

Another method to define a behavioral property is to use (non-)linear regression models. Using this technique, the behavioral property is regressed on the parent variables. In the farm yield example, this could lead to for instance Equation 6.2, as shown below.

$$farm\_yield = 15 \times soil\_qual + 3.2 \times age + 12 \times weather \tag{6.2}$$

where bad weather is represented as 0, and good weather as 1. The advantage of this approach is that fewer parameters are present in the model, making the model simpler to understand. Additionally, noise can be introduced to the function to represent uncertainty in the outcome of the behavior of the agent. Equation 6.3 shows an example of this.

$$farm\_yield = 15 \times soil\_qual + 3.2 \times age + 12 \times weather + N(0,5) \tag{6.3}$$

Other methods that learn models from data, such as neural networks or decision trees, can also be used to specify behavioral properties. Using the insights and expertise of experts, the selection of a method to specify behavioral properties is made. Experts are also use to correct and extend the behavioral properties once generated using the methods described above. Experts are crucial in the determination of the final behavior of agents, but the generated causal graphs and the translation method as described above form important guidelines for these experts.

6.2.5. IMPLEMENTATION AND ANALYSIS

While the focus of this methodology is on the previous steps of designing agent-based models, they still need to be implemented, calibrated and validated to be useful. These three steps are addressed in this section.

IMPLEMENTATION

After specification of the model, it has to be implemented in a programming language or platform. The main requirement is that the language or platform should be capable to codifying all the defined behavior in the previous step. Example programming platforms are Netlogo [207], Mason [208], RePast [209], and Gama [210]. General programming languages such as Java or Python can be used as well, but requires more effort from the programmer to be effective.

CALIBRATION

When the model is implemented, it has to be calibrated. Klügl describes the process of calibration as 'parameters have to be set in a way that a structurally correct model produces a valid outcome' [90]. Depending on the specific model, the right approach for calibration has to be chosen. Calibration can be done using experts manually setting the model parameters, or more automatic approaches [211].

VALIDATION

Once the model is calibrated, the model has to be validated as well. It is often defined as 'the process of determining whether a simulation model is an accurate representation

of the system, for the particular objectives of the study' [212]. Klügl provides a comprehensive overview of how agent-based models can be validated [90]. As the AbCDe methodology is data-driven, it is important to note that the data used for generating the causal graphs should not be used for validation. The data should either be split into two parts (one part for validation, one for calibration), or other validation techniques, such as face validity, should be used to validate the model.

When the model is validated, it can be used to answer the research questions and test the hypotheses as defined in the first step.

6.3. CASE STUDY

We apply the AbCDe methodology to a case study in the field airport security. The case study is used to illustrate the AbCDe methodology, and the airport security domain is used to do that. In airport terminals, the security checkpoint is the most important bottleneck for passengers, and an important source of costs for airport management. As airport passenger numbers are projected to increase in the future, it is essential that security checkpoints are operated efficiently.

In this case study, we explore a new concept of operations, using a service lane, to improve the efficiency of the security checkpoint. A service lane processes passengers that are expected to be slow, and the other open lanes (defined as normal lanes) process the remaining passengers. This concept of operations is projected to improve overall throughput of the system, as faster passengers do not have to wait for slower passengers in front of them. Slow passengers also receive extra help from experienced security officers, potentially increasing the throughput as well.

We design an agent-based model following the AbCDe methodology to determine the effects of implementing a service lane on the throughput of the security checkpoint, as compared to a standard setup. The steps of the methodology, applied to this case study, are outlined below.

6.3.1. PURPOSE OF THE MODEL, RESEARCH QUESTIONS AND HYPOTHESES

The purpose of the model is defined as follows.

To determine the effects of implementing a service lane on the throughput of the security checkpoint, as compared to a standard setup.

A single research question is defined for the model, and stated below.

What is the effect of implementing a service lane setup on the throughput of security checkpoint operations at an airport, as compared to a standard setup?

We specify the following hypothesis, related to the research question as defined above.

The service lane setup increases the overall throughput of the security checkpoint system, by increasing throughput in the normal lanes. The service lane will see a decrease in throughput, as compared to original lanes.

6.3.2. SCOPE AND CONCEPTUAL MODEL

We focus on the behavioral aspects of the security checkpoint actors, while excluding cognitive processes of passengers and security employees. The focus of the causal discovery step in the methodology is on the behavior of the passenger. The security employees are defined as simple agents based on expert input.

Now that the scope of the model is clarified, we specify the conceptual model. This conceptual model is specified in more detail in a technical report [155], but the most important elements are provided below. We identify the environmental objects that are modeled first. These our outlined below.

- **Luggage**. Luggage is owned by a passenger, and has a specific threat level. This is a real value between 0 and 1.
- **X-ray box**. Object in which luggage is dropped. Luggage can be dropped into multiple boxes.
- **X-ray sensor**. Detects the threat level of luggage that it observes.
- **Walk-through metal detector (WTMD)**. Randomly specifies passengers to require an explosive trace detection (ETD) or patdown.
- **Flight**. Abstract concept that has an associated flight time. Passengers are associated with exactly one flight.
- **Queue separator**. Physical objects that are used to form queue areas for passengers.

Now that the environment of the model is specified, we specify the agents of the model. The focus of the causal discovery part of this methodology is at the passenger, but we briefly discuss the other agents below as well.

- **Passenger**. Agent that is associated with a flight, and moves through the security checkpoint. It is also of a specific passenger type, such as young or business. The most important actions are *drop* and *collect*, which refers to dropping and collecting luggage at the security checkpoint. These actions are the focus of the remainder of this case study, and will illustrate the process of using causal discovery for the specification of behavioral properties.
- **X-ray operator**. Uses the X-ray sensor to determine if luggage needs an extra check, and communicates this with the luggage check operator. This is done when the threat level of luggage exceeds a threshold.
- **Luggage check operator**. Checks luggage when requested by the X-ray operator. This is modeled as a waiting time for passengers.
- **Patdown operator**. Performs patdowns and ETD checks. These actions are modeled as a waiting time for passengers.

6.3.3. DATA GATHERING AND ANALYSIS

We collected data of passengers moving through the security checkpoint at Rotterdam The Hague Airport. This data is previously discussed in a publication that is currently under submission, and can be found in a public repository [213].

Table 6.1: The data that was gathered for each passenger in the dataset, along with example data of a slow and a fast passenger. Table from [213]

General	*Data Type*	*Example slow*	*Example fast*
1	Lane number	3	3
2	Date	7-4-2018	7-4-2018
Characteristics			
3	Passenger type	Young	Regular
4	Experience	0	1
5	Number of boxes	1	1
6	Group size	3	1
Timing Data			
7	Start time luggage drop	17:55:14	17:01:09
8	End time luggage drop	17:56:06	17:01:18
9	Time WTMD	17:56:16	17:01:22
10	Time WTMD 2x	17:57:53	-
11	Time WTMD 3x	-	-
12	Start time WTMD check	17:57:57	-
13	End time WTMD check	17:58:30	-
14	Start time ETD check	-	-
15	End time ETD check	-	-
16	Start time luggage collect	17:58:35	17:02:02
17	time end luggage collect	17:59:04	17:02:13
18	Start time luggage check	-	-
19	End time luggage check	-	-

Data for a total of 2277 passengers, flying to 16 different destinations was gathered. Three types of lanes were considered: standard, normal and service lanes. Data for standard lanes was gathered between 23 February 2018 and 17 April 2018, while data for normal and service lanes was collected on the experimental days: 17 December 2018 and 18 December 2018. A service lane was used to process passengers that are expected to be slow, while the normal lanes processed the other passengers. Standard lanes processed all passengers. Days and times were chosen based on isolated blocks of flights that were scheduled, such that all passengers were expected to fly with these flights. For each passenger, 19 different aspects of the security checkpoint process were gathered, which are outlined in Table 6.1. These aspects are mostly related to timing of the different subprocesses they go through, but also relate to characteristics of the agent.

We analyzed this data in the same manuscript, and show two important results here. Figure 6.3 show the mean security checkpoint time for the six considered passenger types: business, senior, family, young, passengers with reduced mobility (PRM) and reg-

Table 6.2: The extracted process duration from the gathered data, based on the entries as specified in Table 6.1. Table from [213].

Time period	Shorthand	Calculation method
luggage drop time	drop	period between 7 and 8
Waiting time before WTMD	wait I	period between 8 and 9
Waiting time after WTMD	wait II	minimum of 1) period between 9 and 16, 2) period between 10 and 16, 3) period between 11 and 16, 4) period between 13 and 16
luggage collect time	collect	period between 16 and 17
Security checkpoint time	checkpoint	period between 7 and 17
Other time	other	checkpoint - drop - wait I - wait II - collect

ular. The security checkpoint time is also split up into 5 distinct parts: *drop*, *collect*, *wait I*, *wait II* and *other*. These parts are defined according to their respective definitions in Table 6.2. PRM passengers were found to be the slowest group with an average of 207 seconds, and business passengers were the fastest with 168 seconds on average. These are according to expectations, as business passengers are often experienced travellers, while PRM passengers often require extra help to go through the security checkpoint.

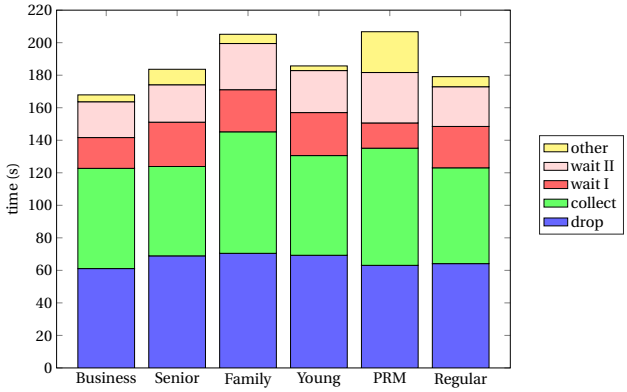


Figure 6.3: The mean processing times, split into five subprocesses, for each of the passenger types in the dataset. Figure from [213].

Figure 6.4 shows the performance of different security checkpoint setups. It shows that four standard lanes had a higher throughput than the throughput of both service lane setup, and five standard lanes had lower throughputs. When just comparing throughput, the service lane setup performed slightly above average compared to the standard setups. However, just the data on its own is not conclusive, so a modeling study will be beneficial to understand the advantages and disadvantages of a service lane setup better. In the next section, we will use causal discovery to design an agent-based model to

achieve that goal.

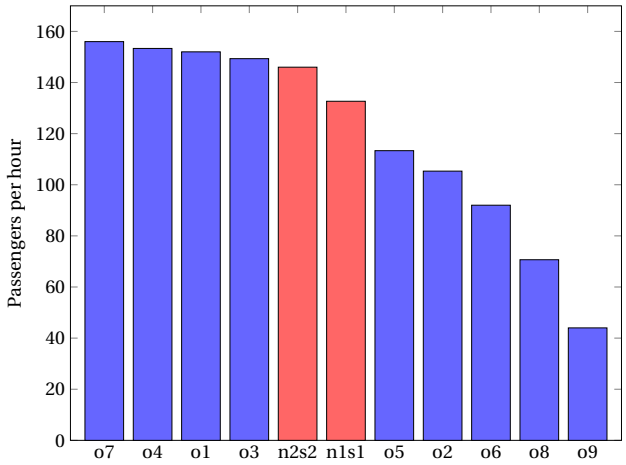


Figure 6.4: The maximum throughput (normalized to passengers per lane per hour) for the 11 different security checkpoint setups. Setups starting with 'o' are composed of original lanes, while setups starting with 'n' contain a normal lane and a service lane. Figure from [213].

While we gathered data for 2277 passengers, it is by no means a very large dataset. However, it is the largest and most detailed dataset on the topic of airport security, and we therefore use this dataset for our study. It suffices to illustrate the workings of the methodology.

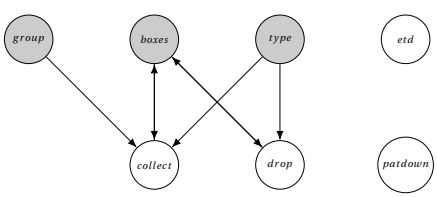
6.3.4. AGENT BEHAVIOR

As mentioned in Section 6.3.2, we focus the generation of behavioral properties on the *drop* and *collect* behavior of passengers. We build two models for these behavioral properties based on two independent assumptions by experts. For the first model, the characteristics model, we assume that the *drop* and *collect* are solely based on the characteristics of the agent and additional random factors. For the second model, the extended model, we assume that these behaviors are additionally influenced by behavior of the agent in front of it.

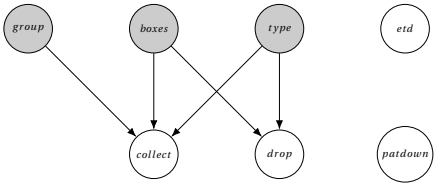
We use the data of the standard lanes to generate the behavioral properties of the agent, while we use data of the service lane experiment to validate the models. To generate the graphs, we combine the score-based GES [205, 206] algorithm and the constraint-based PC algorithm [73], following the Algorithm 3 of Chapter 7.

We use the following variables from the dataset to generate the graph for the characteristics model:

- drop
- collect
- boxes
- type



(a) The generated graph for the standard model.



(b) The expert-based corrected graph for the standard model.

Figure 6.5: The generated graph for the standard model, along with the expert-based corrections.

- group size

These variables are a combination of the characteristics of the agent, and the two behavioral properties that we are interested in (*drop* and *collect*).

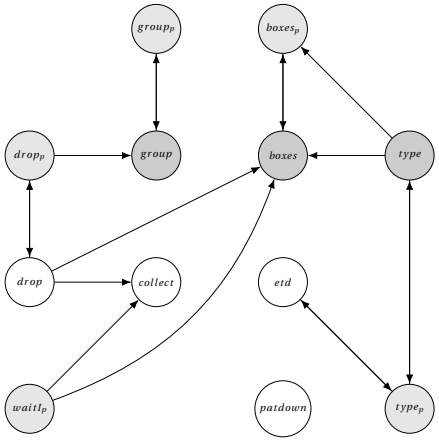
The same variables are used for the extended model as well, but the following variables are additionally used:

- drop$_p$
- wait I$_p$
- boxes$_p$
- type$_p$
- group size$_p$

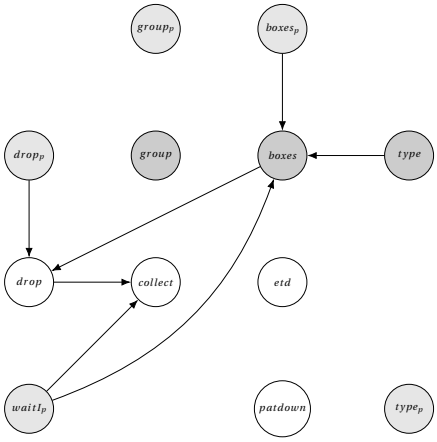
These variables are related to the passenger that performs the security checkpoint process before the passenger under consideration. It is important to note that these consist of the observable behavior and characteristics (i.e. observable by the passenger) of the passenger in front of the passenger for which the behavioral properties are defined.

Figure 6.5a and Figure 6.6a show the graphs that were generated by the causal discovery algorithm for the characteristics model and the extended model respectively. Based on expert insights, these graphs are translated to their final versions, as shown in Figure 6.5b and Figure 6.6b.

The graph generated for the characteristics model shows that both *ETD* and *patdown* are not connected to any other variable in the graph. That means that these are independent variables that can be generated in the model independently as well. Then, both *boxes* and *type* have a causal relationship with both *drop* and *collect*. This implies that these characteristics combined are of influence on the speed in which passengers drop and collect luggage. The generated graph additionally shows that *boxes* is caused by both *drop* and *collect*. Based on expert advice, we assume this link to be unidirectional in the direction of *drop* and *collect*. Finally, the size of the *group* influences *collect*, but not *drop*. In a security checkpoint, passengers travelling in groups often wait for each other to finish collecting their luggage. In this way, they can continue their journey to the gate together. This is not the case for dropping luggage, as passengers can only pass through the WTMD individually.



(a) The expert-based corrected graph for the extended model.



(b) The expert-based corrected graph for the extended model.

Figure 6.6: The generated graph for the extended model, along with the expert-based corrections.

The extended model is based on a generated graph that contains five more variables, and is therefore more complex. This shows that some variables related to the previous passengers are closely related to the same variables of the passenger under consideration. To allow for fair comparison between the two models, we assume that both the group size and the type of agent are independently generated. To this end, we remove the links $group_p \leftrightarrow group$, $type_p \leftrightarrow etd$, $type_p \leftrightarrow type$ and $drop_p \rightarrow group$. Another important factor that we use to correct the graph, is the assumption that the passenger under consideration cannot influence characteristics or behavior of the passenger that is next in line. The links $drop \rightarrow drop_p$, $boxes \rightarrow boxes_p$ and $type \rightarrow boxes_p$ are therefore removed. Finally, we reverse the direction of the arrow $drop \rightarrow boxes$.

The above-described process of expert assessment and improvement of the generated graph is vital for the remaining steps of the model development.

Now that the graphs are complete, we transform them into agent behavior. For the characteristics model, we generate conditional random distributions for the time the passenger takes to *drop* luggage (based on *boxes* and *type*), and *collect* luggage (additionally based on the *group* size). To fit these distributions, we use data of all passengers in the calibration set that possess the right characteristics. We use the Kolmogorov-Smirnov test [214] to determine which of the following distribution has the best fit: Exponential distribution, Gamma distribution, Generalized Extreme Value distribution, Normal distribution, Poisson distribution, and Weibull distribution. Equations 6.4-6.5 below show the drop and collect distributions for a business passenger travelling alone with one box worth of luggage.

$$drop = GeneralizedExtremeValueDistribution(43.95, 19.81, -0.07) \quad (6.4)$$

$$collect = NormalDistribution(36.12, 20.93) \quad (6.5)$$

where the Normal distribution is parameterized by its mean (first parameter) and standard deviation (second parameter), and the Generalized Extreme Value distribution is parameterized by its location (first parameter), scale (second parameter) and shape (third parameter).

A similar procedure as above is followed for the extended model. However, the parent variables that specify the drop and collect distribution are continuous variables, as compared to discrete and categorical variables in the characteristics model. We therefore use a method to fit a generalized linear model [215, 216], based on maximum likelihood estimation (MLE), for the drop and collect distributions. We use the Poisson distribution as a basis for both the drop and collect variables, and a linear combination of their respective parent variables to specify the parameter λ of the Poisson distribution. Equations 6.6-6.9 show the distributions for drop and collect.

$$\lambda_1 = 3.30 + 0.24 \times boxes + 0.009 \times drop_p - 0.001 \times (boxes \times drop_p) \tag{6.6}$$

$$drop = PoissonDistribution(\exp \lambda_1) \tag{6.7}$$

$$\lambda_2 = 3.86 + 0.006 \times drop - 0.002 \times waitI_p - 2.18e^{-5} \times (drop \times waitI_p) \tag{6.8}$$

$$collect = PoissonDistribution(\exp \lambda_2) \tag{6.9}$$

The *boxes* parameter is based on the passenger type, the number of boxes that the previous passenger used ($boxes_p$) and the wait I time of the previous passenger ($waitI_p$). When collecting data we observed that passengers will take longer to drop their luggage if they cannot continue to the WTMD yet. For instance, they realize they have their belts still on, and use an extra box to put that in, or take off their shoes and put that in a new box. This may explain the relationship between the number of boxes and these parameters. We follow a generalized linear modeling approach to specify the boxes distribution in the extended model as well. However, as *type* is a categorical variable, we specify a distribution for each passenger type individually. Equations 6.10-6.11 show the distribution for the Business passenger; other passenger types are defined similarly.

$$\lambda_3 = 0.93 - 0.03 \times boxes_p - 0.01 \times waitI_p + 0.003 \times (boxes_p \times waitI_p) \tag{6.10}$$

$$drop = PoissonDistribution(\exp \lambda_3) \tag{6.11}$$

6.3.5. IMPLEMENTATION AND ANALYSIS

We have implemented the three models in the AATOM simulator, an agent-based airport terminal operations simulator [130]. A screenshot of the implementation in the AATOM simulator is shown in Figure 6.7. For calibration, we focus our analysis on a single flight setup, with a single standard lane open. For validation, we focus the analysis on a two-flight setup, with a service lane and a normal lane open.

CALIBRATION

We calibrated the model with the data that was collected for the nine standard lanes. All important parameters, their descriptions, and their calibrated values can be found in Tables 6.3-6.5. Most parameters could be calibrated using the data, but three parameters

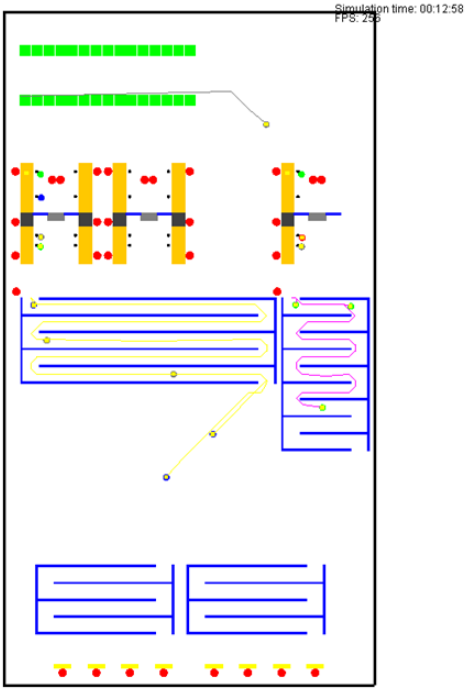


Figure 6.7: A screenshot of the model implemented in AATOM.

have to be set by experimentation. Below we list the three parameters, along with the values that we tested to calibrate them.

- **desiredSpeed**. Tested values: 1.0, 1.1, . . . , 1.5.
- **numberOfDropPlaces**. Tested values: 2, 3.
- **numberOfCollectPlaces**. Tested values: 2, 3.

We ran a total of $N = 1\,000$ simulations for each of the 24 combinations, for all three models, and extracted the following four output values for each simulation run.

- **wait I**. The mean wait I time of all passengers in the simulation. See Table 6.2 for a description of this value.
- **wait II**. The mean wait I time of all passengers in the simulation. See Table 6.2 for a description of this value.
- **throughput**. The maximum number of passengers that passed through the security checkpoint in any 45 minutes period.
- **occupation**. The mean number of passengers that was using the security checkpoint simultaneously for the same period as the highest throughput was observed.

Table 6.3: The calibrated parameters of the model, sorted for different aspects of the model. When the value states *experiment*, different values are used for the standard lane setup as compared to the service lane setup. The value *calibration* refers to a parameter that is calibrated in Section 6.3.5. The Generalized Extreme Value distribution is parameterized by its location (first parameter), scale (second parameter) and shape (third parameter).

Parameter	**Description**	**Calibrated value**
Passenger		
desiredSpeed	The desired speed (in m/s) that the passenger moves through the checkpoint.	Calibration
type	The passenger type.	Based on passengerTypeDistribution
groupSize	The size of the group the passenger travels with.	Based on groupSizeDistribution
Operator		
proportionETD	The proportion of passengers that receive an ETD check.	0.1312
ETDCheckDistribution	The distribution of ETD check times.	GeneralizedExtremeValueMathDistribution(6.73,4.36, 0.69);
proportionWTMD	The proportion of passengers that receive a patdown.	0.0787
WTMDCheckDistribution	The distribution of patdown times.	GeneralizedExtremeValueMathDistribution(19.19,9.35, -0.01);
illegalObjectThreshold	The proportion of passengers that does not have an illegal item.	0.9243
luggageCheckDistribution	The distribution of luggage check times.	GeneralizedExtremeValueMathDistribution(35.20,27.78, 0.33);
Checkpoint		
numberOfNormalLanesOpen	The number of standard lanes that are open.	1 (calibration), 1 (validation)
serviceLaneOpen	The service lane is open or not.	false (calibration), true (validation)
numberOfDropPlaces	The number of passengers that can simultaneously drop luggage at the X-ray system.	Calibration
numberOfCollectPlaces	The number of passengers that can simultaneously collect luggage at the X-ray system.	Calibration
Flight		
numberOfFlights	The number of flights. All flights are assumed to leave at the same time.	2
passengersPerFlight	The number of passengers per flight.	160
arrivalDistribution	The distribution in which passengers arrive.	20% (first half hour), 60% (second), 20% (third), 0% (last)
Passenger distribution		
passengerTypeDistribution	The distribution of passenger types in the population.	Table 6.5
groupSizeDistribution	The distribution of group sizes in the population.	Table 6.4
serviceLaneDistribution	The proportion of passengers per type that will be directed to the service lane.	Table 6.5

These output parameters are emergent effects of the model, and are not explicitly coded into the behavior of any agent. This makes them suitable parameters to determine the quality of the calibrations. We perform linear normalization for each of these output values, using the following functions.

Table 6.4: The distribution of group sizes for the different passenger types.

	Group Size 1	Group size 2	Group size 3
Business	0.75	0.16	0.09
Senior	0.12	0.67	0.21
Young	0.02	0.15	0.83
Family	0.22	0.52	0.27
PRM	0.16	0.58	0.26
Regular	0.34	0.50	0.16

$$\sigma = sd(X) \tag{6.12}$$

$$x_{min} = mean(X) - 2\sigma \tag{6.13}$$

$$x_{max} = mean(X) + 2\sigma \tag{6.14}$$

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{6.15}$$

where X represents the vector of all output values of a specific type (i.e. all simulated wait I times), and $x \in X$. We perform the same procedure for these output parameters in the real data.

We calculate the Euclidean distance between each of the calibrations and the real data. Table 6.6 shows the best calibrations for each model type, along with the calculated distance.

It is interesting to observe that for all three models the same number of drop places and collect places are obtained, and similar desired passenger speeds are found. This is a good indication that these values correspond to their real-world equivalents.

VALIDATION

In this final step, we assess the performance of the models by using data of the normal and service lanes. In comparison with the calibration step, we change the distribution of arriving passenger types (see Table 6.5), have two lanes open instead of one, and specify one lane as a service lane. The proportion of passengers per type that are sent to the

Table 6.5: The proportion of passengers of different types in the calibration and validation experiment (first two columns). The proportions do not add up to 1 due to rounding errors. The last column represent the proportion of passengers of a specific type that are sent to the service lane in the validation experiment.

	Calibration	Validation	Service lane
Business	0.15	0.17	0.21
Senior	0.17	0.23	0.60
Young	0.15	0.13	0.41
Family	0.11	0.07	0.76
PRM	0.012	0.004	1.00
Regular	0.41	0.37	0.51

Table 6.6: The calibrated models along with their distances to the real data.

Model	dropPlaces	collectPlaces	desiredSpeed	Distance
Expert	3	3	1.4	2.1627
Characteristics	3	3	1.4	2.2265
Extended	3	3	1.5	2.2164

service lane are also specified in Table 6.5. We normalize the data following the same approach as the calibration, and calculate the distance again. As we have two open lanes instead of one, the number of output parameters has also doubled. The resulting distances to the validation data for the calibrated models of Table 6.6 are shown in Table 6.7.

Table 6.7: The calibrated models along with their distances to the validation data. For fair comparison, the models with (2), are the models that have the same calibration parameters as the extended model.

Model	Distance
Extended	3.4862
Characteristics (2)	3.6056
Expert	3.6071
Expert (2)	3.6157
Characteristics	3.6486

Results show that the extended model has the shortest distance to validation data. It is followed by the characteristics model (2), and the expert model. The highest distance with the validation was by the characteristics model, followed by the expert (2) model. These results indicate that building a model with our methodology can improve the accuracy of the models over models developed by experts alone. While more work is needed to show the advantages and disadvantages of the methodology, these initial results are promising.

6.4. DISCUSSION

An important issue that occurred during the generation of causal graph is that different algorithms and parameters produce quite diverse causal graphs. By integrating the PC algorithm with the GIES algorithm, following the approach of Chapter 7, this problem is partially addressed, but certainly not solved. We believe that further developments in the field of causal discovery will improve our methodology as well.

A major advantage of our methodology is that it provides modellers with a toolbox to design agent-based models. Previously, agent behavior was mostly defined based on expert skills, but with this toolbox it makes it easier for experts to come up with the behavioral properties. By no means we believe that experts become obsolete with our methodology, but we do believe that it can help make agent-based models better and more consistent.

The quality of the model generated with the AbCDe methodology depends heavily on the data that is used to generate causal graphs. For some applications, such as terrorism and future technological advances, virtually no data exists or can be gathered. For these

applications our methodology cannot be used, and traditional expert-based model design is more suitable. In cases where agent-specific data can be gathered, such as our examples on airport security checkpoint efficiency and segregation, our methodology can impact the final quality of the developed model.

6.5. CONCLUSIONS

Causal discovery algorithms translate data into a directed causal graph that reveals the causal structure among variables. In this chapter, we investigated how these algorithms can be incorporated in the design process of agent-based models. We therefore proposed an agent-based model-design methodology, called AbCDe, that uses causal discovery algorithms and the growing availability of data to specify behavioral properties. This methodology combines traditional expert-based model design techniques with causal graphs to design better models.

We applied the methodology to a case study that studies the effects of implementing a so-called service lane at the security checkpoint. We gathered a dataset containing detailed information about over 2,000 passengers moving through the security checkpoint. The models that are generated with the AbCDe methodology show closer resemblance to validation data than an existing model that was designed by an expert alone.

More case studies have to be performed to understand the advantages and disadvantages of our methodology better, but first results are promising. Future work can also focus on developing dedicated causal-discovery algorithms for agent-based model development, instead of adapting existing algorithms for that purpose.

7

USING CAUSAL DISCOVERY TO ANALYZE EMERGENCE IN AGENT-BASED MODELS

In Chapter 6 we showed that agent-based models can effectively be designed using causal discovery. However, analyzing agent-based models is a complex task as well. Agent-based models typically contain complex non-linear interactions between agents and generate emergent properties that cannot easily be explained. They are most commonly analyzed using sensitivity analysis techniques. While these techniques help understanding agent-based models better, they are not a one-size-fits-all solution. In this chapter, we explore the novel use of the causal-discovery algorithms of Chapter 6 as an additional means to analyze agent-based models. We propose the AbACaD methodology: Agent-based model Analysis using Causal Discovery. In this methodology, emergence in agent-based models is analyzed using causal-discovery algorithms in combination with both machine learning and sensitivity analysis techniques.

Parts of this chapter have previously been published in the Simulation Modelling Practice and Theory journal (2019) [217].

7.1. INTRODUCTION

Agent-based modeling is commonly used to analyze complex systems. These systems are characterized by a large variety of components that interact with each other. Agent-based models mimic the behavior of the actors in these systems to better understand them and potentially increase their performance. An interesting feature of agent-based models is that of emergence: "patterns, structures, and behaviors that were not explicitly programmed into the models, but arise through the agent interaction" [218]. One of the prominent techniques to understand emergent properties is sensitivity analysis. Using sensitivity analysis, one aims to determine the effect of changing input parameters on the output variables of an agent-based model. Sensitivity analysis, however, does not shed light on the causal relationships between different elements of a model.

To better understand emergent properties of agent-based models, we propose AbACaD in this chapter. AbACaD is a novel methodology in which emergence in agent-based models is analyzed using causal discovery in combination with both machine learning and sensitivity analysis techniques. Causal-discovery algorithms are combined to generate a causal graph that represents causal relationships between model parameters and output variables of the model. This causal graph is then exploited to understand emergent properties of the model better. Machine learning and sensitivity analysis techniques are additionally used, as a synthesis of analysis outcomes from different methods allows for richer explanations of emergent behavior. Furthermore, causal graphs sometimes give inconclusive results, which can be cross validated using these different analysis techniques.

Two case studies are used to exemplify the use of the methodology. The first case study focuses on the El Farol bar problem, in which agents only enjoy going to the El Farol bar when it is not overcrowded. The second case study focuses on airport security and efficiency, in particular on an Improvised Explosive Device (IED) attack.

This chapter is outlined as follows. The AbACaD methodology is outlined in detail in Section 7.2, and the two case studies are discussed in more detail in Section 7.3. Finally, a discussion is provided in Section 7.4 and the chapter is concluded in Section 7.5.

7.2. ABACAD METHODOLOGY

This section outlines the AbACaD methodology: **A**gent-**b**ased model **A**nalysis using **Ca**usal **D**iscovery. The methodology consists of several steps, graphically shown in Figure 7.1.

The core of the methodology lies in the inference of a causal graph derived from data generated by agent-based simulations performed with the model. Sensitivity analysis and machine learning analysis provide extra insights in emergence of the agent-based model.

The combination of these three methods is used for cross validation purposes. Following this approach, sensitivity analysis and machine learning techniques are used to identify potential inconsistencies produced by the causal-discovery algorithms. At the same time, a combination of the three techniques provides richer insight into the behavior of the model. Causal discovery introduces structure to the output, while sensitivity analysis sheds light on the magnitude of the effect of model parameters on output variables. Machine learning algorithms generate a metamodel of the agent-based model

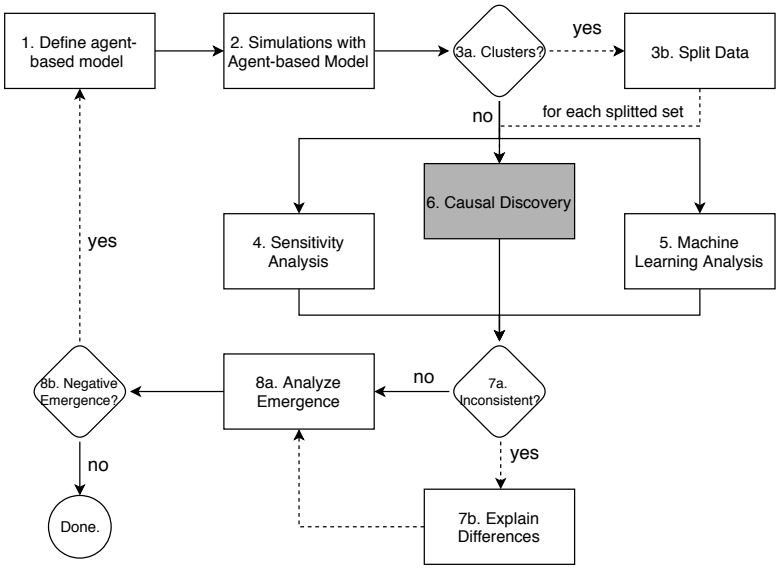


Figure 7.1: The AbACaD methodology.

under consideration. This combination of the three analysis techniques provides versatile insights in the emergent behavior of an agent-based model.

The advantages of using the causal-discovery algorithms for analyzing emergence in agent-based modeling are twofold. First, it provides the researcher another means to understand the emergent properties of the agent-based model that was developed and the real-world system that it mimics; it provides insight in which factors cause (emergent) behavior of the model. This can lead to potentially unforseen explanations of emergent properties that could not be revealed by the sensitivity analysis or machine learning techniques alone. Second, a causal graph provides a clear representation of the underlying mechanisms in an agent-based model. This allows the modeler to effectively communicate results to external (non-technical) parties. Each of the steps of the methodology are explained in detail below.

7.2.1. DEFINE AGENT-BASED MODEL

This chapter assumes a well calibrated and validated model as a starting point. As with any agent-based model the environment of the model has to be defined. In addition, the agents, their mutual interactions and their interactions with the environment of the model have to be defined. It is assumed that this model has a set of model parameters, with associated parameter ranges or distributions, and a set of output variables. We refer to output variables as the measured output quantities of the model, and refer to model parameters as the quantities that can be changed in the model. All other parameters of the model are considered constants.

Choosing the right model parameters and output variables is essential for a proper understanding of the model in later stages of the methodology. This is a creative pro-

cess that requires knowledge about the system that is modeled. Furthermore, when too few parameters are present, interesting emergent properties might not be observed, while too many parameters might lead to uninterpretable large graphs. An overview of designing proper agent-based models is provided by Grimm et al. [56] and Macal and North [219].

7.2.2. SIMULATION WITH AGENT-BASED MODEL

Once the model is defined, simulations are performed based on the model. To this end, a sampling strategy to explore the behavior of the model by simulation has to be chosen. This sampling strategy can range from simple methods, such as exhaustively sampling every parameter combination N times, to more complex methods, such as Latin Hypercube Sampling (LHS) [65], sequential sampling [220] and importance sampling [221].

Determining the appropriate sampling strategy for a given agent-based model is an important open problem in the agent-based modeling community. It depends on a large number of criteria, such as the computational requirements of the model, the type of model parameters, the complexity of the model output, the computational resources or even the programming language in which the model was implemented.

Some authors address the described issues in their work. For instance, Edalia and Yücel provide a metamodel-guided technique that helps to choose the right sampling methodology [222]. Furthermore, Bremer and Sonnenschein compare a set of sampling techniques on an agent-based smart grid model [223]. These works provide a guideline to choose the right sampling strategy.

After selecting the sampling strategy, simulations are performed with the agent-based model. The output of these simulations is analyzed in the subsequent steps.

7.2.3. MULTIPLE CLUSTERS

When the output of the model is too complex, many analysis techniques struggle to fully capture its dynamics. It is therefore useful to split the data in clusters with more homogeneous behavior and dynamics [224]. These homogeneous clusters are then analyzed separately, and their dynamics can more easily be captured by analysis techniques. When the data does not contain clusters, the subsequent analysis steps of AbACaD are performed with the full output dataset.

To be able to detect whether homogeneous clusters exist, different techniques can be used. For instance, the Silverman's test can be performed to determine whether the output data is multimodal [225]. This test can also be done to determine the number of modes in the data. Another set of tests is based on clustering algorithms, such as K-means clustering [226]. Three main algorithms to determine the number of clusters are distinguished: the elbow method [227], the silhouette method [228] and the gap statistic method [229]. Another method to limit the complexity of the model outputs, is to limit the parameter range of a model parameter to a default value. This will impact the detection of emergent properties, as certain patterns may not be present. This is similar to the one-at-a-time method for local sensitivity analysis, as also discussed in Section 7.2.4.

Clustering of data gives insights in specific aspects of the model, and helps to understand emergence in these regions better. These emergent effects might be diminished in the total dataset, and therefore invisible in the analysis. It is important to understand

the characteristics of the different clusters well. Without a proper understanding of these clusters, and their respective differences with the total dataset, the graphs cannot be easily interpreted. Clustering the dataset into many clusters (i.e. more than three) makes the analysis more complicated, as too many clusters have to be analyzed independently.

7.2.4. SENSITIVITY ANALYSIS

We perform sensitivity analysis to get a better understanding of the model. Several methods to perform sensitivity analysis on agent-based models exist, and their usefulness depends on the specific model that was defined and the goals of the analysis. For instance, local sensitivity can be assessed using one-at-a-time (OAT) methods, while global sensitivity can be assessed using variance-based methods [64]. A large number of other methods exists to perform sensitivity analysis [63–65].

7.2.5. MACHINE LEARNING ANALYSIS

Another form of analysis of agent-based models is that of machine learning analysis. Essentially, using machine learning a metamodel of the agent-based model is generated, which helps to understand relationships between model parameters and output variables [224, 230]. Many types of machine learning algorithms exist to perform metamodeling of the agent-based model. Examples include regression trees, neural networks, (linear) regression, support vector machines and combinations thereof.

Within the field of machine learning, there are many options to analyze agent-based models, and the selection of the right algorithm depends on the type of model. Some authors have looked into choosing the appropriate technique for a given model. For instance, Arroyo et al. provide a methodology that explains why and when machine learning algorithms can be used for agent-based models [66]. Furthermore, Sanchez and Lucas provide an overview of methods that can be used to analyze agent-based models [230]. These works provide guidelines to choose the right machine learning algorithm.

7.2.6. CAUSAL DISCOVERY

Causal discovery is the process of inferring a causal structure from available data. In this methodology, we use data generated by agent-based simulations performed with the model that was defined in the first step to generate a causal graph. This generated causal graph adds structure to the model parameters and output variables, and is used to better understand emergent properties of agent-based models. Similarly to machine learning models, causal graphs can be seen as a form of metamodels.

The PC algorithm is widely used in the domain to generate causal graphs. It is designed to generate a causal graph based on observational data. The PC algorithm first generates an undirected graph, called the skeleton, using an adjacency search algorithm. The skeleton is used as a basis for the rest of the algorithm, in which edges are oriented based on so-called V-structures in the skeleton. The algorithm finally results in a completed partially directed acyclic graph (CPDAG), which represents the Markov equivalence class of DAGs that were inferred from the data. In case of sampling errors in the data or hidden variables, the PC algorithm might output an invalid CPDAG. Such a CPDAG cannot be extended to a valid DAG with the same skeleton and V-structures.

This is resolved by randomly trying a set of different directions for the unresolved edges, and then choose the first combination that results in an extendable CPDAG. This can however lead to very different CPDAGs for the same set of inputs.

We manage the above instability of output by the following Algorithm 1, presented below. This algorithm simultaneously tunes the tuning parameter α. Smaller values of α generally lead to fewer edges in the graph, and the value for α is commonly based on assumptions in previous work. In contrast, we automatically tune this parameter α based on the maximum number of edges that we would like to observe in the generated CPDAG, and the stability of the PC algorithm output. We consider the output of the PC algorithm stable when it generates the same output c times. We refer to a PC-generated graph with at most e edges as G^e_{pc}.

Data: dataset D, maximum number of edges e, stability constant c
Result: CPDAG G^e_{pc}
$stable \leftarrow false$;
$\alpha_{final} \leftarrow 1$;
$G^e_{pc} \leftarrow$ *fully connected graph*;
while $|edges(G^e_{pc})| > e \vee \neg stable$ **do**
  $\alpha_{final} \leftarrow \frac{1}{10}\alpha_{final}$;
  $G^e_{pc} \leftarrow pc(D, \alpha_{final})$;
  $stable \leftarrow isStable(G^e_{pc}, \alpha_{final}, c)$;
end

Algorithm 1: PC algorithm with automatically tuned α parameter.

Apart from the constraint-based PC algorithm, we employ the Greedy Equivalence Search (GES) algorithm [206]. An adapted algorithm, GIES, allows for interventional datasets (i.e. datasets in which variables were purposely varied), as compared to observational datasets used in the PC algorithm [205].

The GIES algorithm uses a greedy search algorithm to maximize some score function that scores potential causal graphs based on data. It consists of three phases: forward phase, backward phase and turning phase. In the forward phase, starting from an empty graph, edges are added to the graph while improving the score. In the backward phase, edges are removed from the graph again while improving the score. Finally, the turning phase checks if turning edges around can still improve the score. An important parameter that can be tuned is the parameter d that specifies the maximum degree of each of the vertices in the graph, where a low d generally leads to sparser graphs. The choice of this parameter is highly domain dependent, and requires some domain knowledge to set. Contrary to the α parameter in the PC algorithm, the maximum degree d of the GIES algorithm is a more intuitive parameter to tune. We therefore do not use an automatic tuning algorithm, as we used for the PC algorithm. We refer to the graph with a maximum degree of d as G^d_{gies}.

Data: PC graph set $\mathbb{G}_{pc}$, GIES graph set $\mathbb{G}_{gies}$
Result: Graph G_{mrg}
$G_{mrg} \leftarrow$ *fully connected graph*;
for $e \in$ *edges*(G_{mrg}) **do**
  $i \leftarrow 0$;
  for $G \in \mathbb{G}_{pc}$ **do**
    if $e \in$ *edges*(G) **then**
      $\mathrm{i} \leftarrow \mathrm{i} + 1$;
    end
  end
  for $G \in \mathbb{G}_{gies}$ **do**
    if $e \in$ *edges*(G) **then**
      $\mathrm{i} \leftarrow \mathrm{i} + \frac{|\mathbb{G}_{pc}|}{|\mathbb{G}_{gies}|}$; ▷ Equal weights for GIES and PC
    end
  end
  if $i \leq |\mathbb{G}_{pc}|$ **then** ▷ Weighted majority not reached
    edges(G_{mrg}) $\leftarrow$ *edges*(G_{mrg}) $\setminus \{e\}$;
  end
end

Algorithm 2: Graph Merging Algorithm.

GRAPH MERGING

As will be discussed in detail in Section 7.3, the graphs that are generated by the PC algorithm and the GIES algorithm are not always consistent. They are generated based on respectively a constraint-based algorithm and a score-based method, often leading to different results. Furthermore, a lower α in the PC algorithm and a lower d in the GIES algorithm commonly lead to sparser and thus different graphs than graphs generated with higher parameter values. We therefore find it useful to merge the considered graphs using a weighted majority-vote procedure. As the parameter range of the PC algorithm is generally larger than the parameter range of the GIES algorithm, we proportionately add weight to GIES graphs in the algorithm. This algorithm starts with a fully connected graph G_{mrg}, and removes an edge between two variables if the majority of considered graphs does not have an edge between these variables. This is outlined in Algorithm 2.

We use Algorithm 3 in this chapter to generate a causal graph. This algorithm first generates a set of PC graphs $\mathbb{G}_{pc}$ using Algorithm 1 and a set of GIES graphs $\mathbb{G}_{gies}$ using the unaltered GIES algorithm. These graphs are finally used to generate the majority graph using Algorithm 2.

The selection of the range of algorithm parameters (i.e. E_{pc} and D_{gies}) is important. Choosing too many sparse graphs (i.e. many PC graphs with a low e) risks missing certain causal relationships, while choosing too many dense graphs is prone to detecting incorrect relations.

This algorithm does not replace the need for proper analysis of the individual graphs that are generated by the algorithms. Some specific graphs might produce structures

that are of great interest, but are eliminated from the majority graph.

Data: dataset D, range of PC graph edges E_{pc},
range of GIES graph degrees D_{gies}, stability constant c
Result: Graph G_{mrg}
$\mathbb{G}_{pc} \leftarrow \emptyset$;
$\mathbb{G}_{gies} \leftarrow \emptyset$;
for $e \in E_{pc}$ **do**
  $\mathbb{G}_{pc} \leftarrow \mathbb{G}_{pc} \cup generate\_pc(D, e, c)$;
end
for $d \in D_{gies}$ **do**
  $\mathbb{G}_{gies} \leftarrow \mathbb{G}_{gies} \cup gies(D, d)$;
end
$G_{mrg} \leftarrow graph\_merge(\mathbb{G}_{pc}, \mathbb{G}_{gies})$;

Algorithm 3: Causal Discovery Algorithm.

7.2.7. EVALUATE INCONSISTENCIES

The generated graph structure is compared to both (global) sensitivity analysis results and machine learning analysis results, and inconsistencies are noted. Domain experts are exploited to assess the source of these inconsistencies and put the results in perspective. The inconsistencies have to be explained before moving on to the final step of analysis of emergence.

It is important to note that the analysis methods generate output of different explanatory power. Sensitivity analysis determines the size of the effect of model parameters on output variables, while causal discovery adds structure to the output. Machine learning algorithms generate a meta-model of the agent-based model under consideration. Due to these inherent differences of the analysis techniques, differences in results might occur as well. These differences might not necessarily indicate a contradiction, but might point towards different emergent properties involving the same variables. This needs to be taken into account while analyzing potential inconsistencies, but can be beneficial during the final step as outlined below.

7.2.8. ANALYZE EMERGENCE

The graph G_{mrg} is used to identify emergent behavior observed in the agent-based model. Emergent effects in the graph are observed by finding indirect relationships between model parameters and output variables. It is this presence of an intermediate variable that explains how a certain effect is realized. Furthermore, the absence of causal relationships between model parameters and output variables can have important implications. For instance, the absence of a path between two correlated parameters can indicate that expected causal path does not exist. When the graph indicates a relationship between variables, sensitivity analysis and machine learning analysis are used to determine the strength and direction of the relationship.

By systematic analysis of the (absent) links in the generated graph, in combination with the other analysis methods, causal relationships that are different from the initial

understanding of the modeler can be found. The modeler can further investigate these relationships, gaining a better understanding of the model and potentially, the phenomenon that is modeled. This is ideally done in discussion with domain experts. The unexpected relationships could be an indication of positive emergence.

It could also be that the analysis points towards an unwanted causal path between two variables. This is an indication of negative emergence [59], and has to be addressed in subsequent versions of the model. This characterization of a relationship being unwanted is also to be determined in discussion with domain experts.

When no examples of negative emergence have been found, the analysis is complete. If negative emergence was found, the model has to be updated, returning the researcher to the first step. This will further improve the model, and will help the researcher to gain a better representation of the phenomenon that is investigated.

7.3. CASE STUDIES

Two case studies are performed to illustrate the use of the AbACaD methodology as outlined in Section 7.2. The first case study is on the El Farol bar problem, while the second case study focuses on security and efficiency in a regional airport terminal. We use the implementations of the PC algorithm and the GIES algorithm from the *pcalg* [231] package in R [232], and implemented Algorithms 1-2 in R as well. An additional analysis of the central AbACaD algorithm (see Section 7.2.6) is also provided.

7.3.1. EL FAROL BAR PROBLEM

We analyze the well-known El Farol bar problem [233] using the AbACaD methodology. In the El Farol bar problem, the attendance of a popular bar in Santa Fe, New Mexico is investigated. The bar is especially popular on Thursday nights, but is deemed unpleasant if it is overcrowded. We investigate how the way people choose to visit the bar influences the attendance. In particular, we are interested in the overall attendance of the bar, the variability, the strategies of agents, and the population inequality.

An agent-based model for this problem was defined and implemented in the Netlogo environment by Rand and Wilensky [234], and a visualization is shown in Figure 7.2. In this model, three parameters are present: number of strategies (n), memory size (m) and overcrowded threshold (T). A strategy consists of a set of $m+1$ weights that represent how an agent believes that each of the m previous week attendances affects the attendance in the current week, complemented with a constant baseline attendance. These strategies are unique for each agent, and are randomly generated. Of the n strategies an agent can choose from, he uses the best possible strategy to predict the attendance of the current week, and makes his decision accordingly.

We analyze the model with a set of $S = 100$ agents, $n \in [1,20]$, $m \in [1,20]$ and $T \in \{0,5,\ldots,100\}$ in a time period of $t = 104$ weeks. Results are recorded based on the last 52 weeks. Output of the model is considered in six dimensions:

- **Mean bar attendance** A. The mean number of agents that attended the bar.
- **Mean variability** V. The mean difference between attendance of two consecutive weeks.

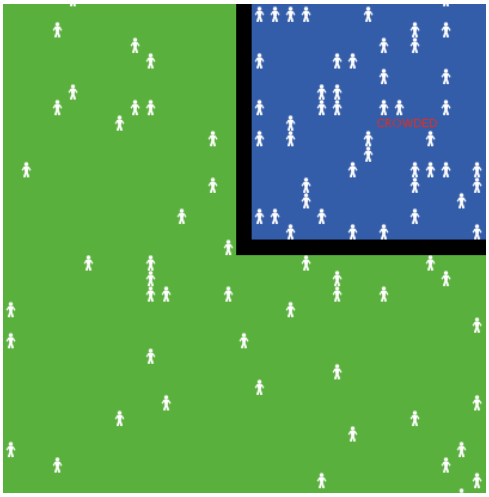


Figure 7.2: A visualization of the Netlogo implementation of the El Farol bar problem.

- **Mean attendance difference** A_{diff}. The mean difference between the attendance and the overcrowding threshold.
- **Number of times overcrowded** O. The number of times the bar was overcrowded.
- **Last changed strategy** C. The mean week that the agents last changed their strategy.
- **Population inequality** G. The Gini coefficient of the overall bar attendance [235]. A low Gini coefficient represents an equal distribution of bar visits among the population, and a high Gini coefficient represents the opposite.

We analyze results based on a set of $N_{tot} = 300,000$ samples generated by a maximin latin hypercube sample design. The cluster counting algorithms, as specified in Section 7.2.3, returned different results, although both the elbow method and the silhouette method indicated a total of 3 clusters. The Silverman's test indicated a total of 6 modes, while the gap statistic method indicated at least 10 clusters. We perform sensitivity and machine learning analysis for the full dataset, and generate and analyze a causal graph for this set as well. Additionally, we generate and analyze the graphs for three clusters obtained by k-means clustering.

SENSITIVITY & MACHINE LEARNING ANALYSIS

We performed sensitivity analysis to determine the first order effects of each model parameter on the output variables. Figure 7.3 shows this variance decomposition for each of the output variables. It shows that the overcrowded threshold T has a large influence on three outputs: mean attendance A, number of times overcrowded O, and attendance difference A_{diff}. Each of these outputs are directly related to attendance, and it therefore makes sense that there is a strong relationship between T and these outputs. It is

Table 7.1: Spearman correlation matrix.

.	A	V	O	C	G	A_{diff}
n	−0.04	0.57	−0.14	0.68	−0.54	−0.07
m	−0.18	−0.53	−0.03	−0.45	0.51	−0.18
T	0.92	−0.23	−0.85	−0.34	−0.45	−0.92

furthermore clear that there is a large interaction effect of the parameters on the remaining three output variables: mean variability V, last changed strategy C and population inequality G. Memory size m has a strong effect on the mean variability V. No individual parameters has a large influence on population inequality G, and a similar trends is observed for the last changed strategy C.

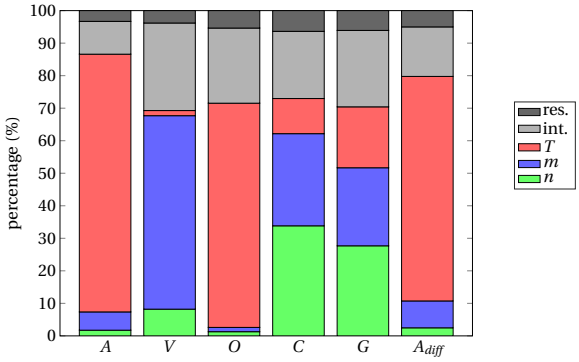


Figure 7.3: The variance decomposition of each of the output variables of the El Farol bar model. Interactions are shown as *int.*, residuals are shown as *res.*

We further show a Spearman's rank correlation matrix in Table 7.1. This shows similar trends as the variance decomposition as seen in Figure 7.3, but does not show interaction effects. In this matrix, the direction of the relationship is indicated, but lacks the interaction and residual effects that is observed in Figure 7.3.

Similar to the above analysis methods, regression helps to identify important parameters. We performed ordinary least squares regression, and show the regression coefficients in Table 7.2. The parameters with high correlation coefficients (Table 7.1) and high variance contributions (Figure 7.3) also have nonzero regression coefficients. The regression coefficients additionally describe the strengths of the relationship between model parameters and output variables.

CAUSAL DISCOVERY

Next we perform causal discovery on the agent-based model approach using the causal discovery algorithm (Algorithm 3) with $E_{pc} = \{12, \ldots, 19\}$, $D_{gies} = \{3, 4\}$ and $c = 5$. These parameters were derived by initial exploration of the parameter space. The total number of differences between the individually generated graphs are outlined in Table 7.3. We define a difference as the minimum number of arrow additions and removals to transform one graph into another. The differences between the smaller PC graphs are

Table 7.2: The regression coefficients for the El Farol bar problem.

	Intercept	n	m	T
A	32.826	-0.144	-0.621	0.569
V	27.827	0.968	-2.108	-0.064
O	64.26	-0.32	-0.289	-0.602
C	80.84	1.957	-1.722	-0.215
G	0.45	-0.015	0.013	-0.002
A_{diff}	32.826	-0.144	-0.621	-0.431

quite small, while the GIES graphs differ a lot more from any of the PC graphs. The final merged graph G_{mrg} is shown in Figure 7.4, and this graph is used as the basis for the remainder of the analysis.

Table 7.3: The number of differences between the considered graphs, and the total sum of differences in the El Farol bar problem.

	G^{12}_{pc}	G^{13}_{pc}	G^{14}_{pc}	G^{15}_{pc}	G^{16}_{pc}	G^{17}_{pc}	G^{18}_{pc}	G^{19}_{pc}	G^{3}_{gies}	G^{4}_{pc}
G^{12}_{pc}	0	3	3	5	5	9	14	13	13	17
G^{13}_{pc}	3	0	0	2	2	6	13	12	12	16
G^{14}_{pc}	3	0	0	2	2	6	13	12	12	16
G^{15}_{pc}	5	2	2	0	5	6	11	10	14	16
G^{16}_{pc}	5	2	2	0	0	6	11	10	14	16
G^{17}_{pc}	9	6	6	6	6	0	13	12	12	16
G^{18}_{pc}	14	13	13	11	11	13	0	5	21	17
G^{19}_{pc}	13	12	12	10	10	12	5	0	18	14
G^{3}_{gies}	13	12	12	14	14	12	21	18	0	6
G^{4}_{gies}	17	16	16	16	16	16	17	14	6	0
Sum	**82**	**66**	**66**	**66**	**66**	**86**	**118**	**106**	**122**	**134**

INCONSISTENCIES

Some interesting differences and similarities are observed when we compare the generated graph G_{mrg} with the analysis results in Section 7.3.1. Specifically, we observe the most dominant sensitivity parameters for each output variable (as shown in Figure 7.3), and check how that corresponds to the graph structure of G_{mrg}. Each of the output variables was found to be a direct descendant of the most dominant model parameters in G_{mrg}. An important exception to this is the relationship between memory size m and population inequality G. These two variables are indirectly related through the intermediate output variable variability V, C and A. We will discuss this in more detail in the next section on emergent behavior. As all of the dominant relations as found in sensitivity analysis are also paths between variables in the graph, there are no inconsistencies that need to be explained.

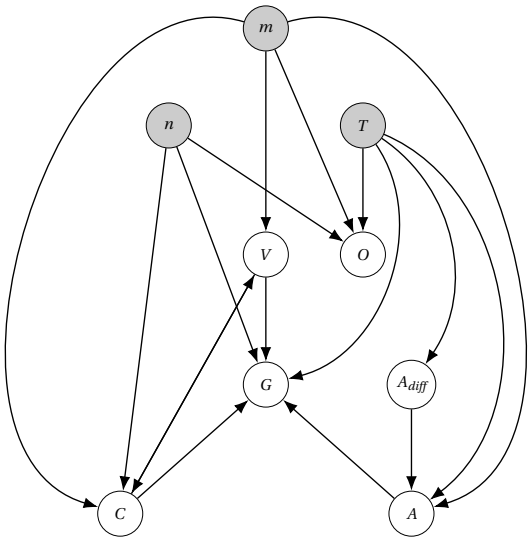


Figure 7.4: G_{mrg} for the El Farol bar problem. The grey nodes are the model parameters, and the white nodes are the model outputs.

EMERGENCE

Some interesting observations are made based on the generated majority graph. As discussed in the previous paragraph, no direct effect between memory size m and the population inequality G is observed. A possible explanation is the following. With an increase in memory size m, the total population starts to make better predictions (i.e. low C), as a bigger memory allows for more variables to be taken into account. These better predictions lead to a reduced variability V between weeks. Furthermore, when the variability reduces, and the strategies are more stable, the same people end up going to the bar every week. This then gives an unfair division of people attending, and thus a high Gini coefficient. This effect is also additionally reinforced by the correlation coefficient as observed in Table 7.1.

Another interesting observation is the bidirectionally of the link between V and C. This is a cycle in which a lower variability leads to agents changing their strategy less, then leading to a lower variability, and so on. The used algorithms (i.e., GIES and PC) are explicitly designed to yield directed acyclic graphs. This by definition prevents them from generating cycles. However, when the direction of a link is unclear, different input parameters for the algorithms might return graphs with opposite edge directions. In this case, the merging algorithm returned both edges (V $\leftrightarrow$ C), as they were seen in different outputs of the GIES and PC algorithm.

Memory size m has one other path to population inequality G, which is through mean bar attendance A. This relation is intuitive as well; with a very high mean bar attendance A, most agents visit the bar frequently, and therefore the Gini coefficient is low. The other way around holds as well. When only few people regularly attend the bar,

the Gini coefficient becomes high.

Another relationship identified is that between attendance difference A_{diff} and mean bar attendance A. These two parameters are closely related, and it makes sense that a causal link is identified. However, it would be expected that the link is defined the other way around, as attendance influences attendance difference as following this equation: $A_{diff} = A - T$. This can easily be reformulated as $A = A_{diff} + T$, which makes it hard to distinguish the direction of the causal link.

By only performing sensitivity analysis and machine learning, it is difficult to detect this type of indirect relationships between variables. The structure of the generated causal graphs provides an extra means to analyze emergent behavior in agent-based models. In this case we observed an indirect relationship between memory size and the last time agents changed their strategy, which leads to a better understanding of the underlying model.

Clustering Apart from analyzing the graph generated by the full dataset, we generated three separate graphs for three clusters generated by the k-means algorithm. The means and variances of each of the model variables are indicated in Table B.1 and Table B.2 of Appendix B respectively. As expected, variances in the cluster are smaller than the total dataset. Furthermore, for each output variable, the mean differs in at least one cluster from the mean in the total dataset. The three corresponding graphs are shown in Appendix C.

We discuss the graph of the first cluster here, as similar analysis can be done for the other two clusters. This cluster has a high mean overcrowding threshold T and memory size m. This leads to a comparatively high mean attendance A and low mean variability V. Compared to the original graph, the relationship between variability V and population inequality G is removed. The correlation coefficient between these two variables in the original dataset ($r = -0.62$) is double that of the same correlation coefficient in the cluster ($r = -0.30$). G also has a causal link with attendance difference A_{diff} in the graph. Again the correlation coefficient is higher ($r = 0.20$ vs. $r = 0.44$), and an edge is added for that reason. In some cases, the addition of an edge cannot be explained by changed correlation coefficients or variances. In this case, the enforced graph structure (i.e. directed acyclic graphs) can play a role in the addition of the edge. The other clusters have similar types of additions and removals of edges. This cluster analysis shows how emergent effects in specific regions of the model can be found that are not observed by analyzing the full dataset.

7.3.2. SECURITY & EFFICIENCY

We consider the agent-based model on airport security and efficiency described in Chapter 4, which contains a broader set of interactions between agents than the El Farol bar problem. In this model, passengers, security operators and an attacker are explicitly modeled. Passengers move through different areas of the airport (e.g., check-in area or security checkpoint area) and interact with security operators. The attacker aims to achieve as many fatalities as possible by detonating an Improvised Explosive Device (IED) in the most crowded queue of the publicly accessible areas of the airport.

The model is used to investigate the relationships between casualties in such an IED

attack and different efficiency dimensions in terminal operations. A visualization of the implementation of the model is shown in Figure 4.2.

The following model parameters are considered: number of flights $f \in \{1,2,3\}$, number of checkpoint lanes open $l \in \{2,3,4\}$, number of check-in desks open $k \in \{3,5\}$, number of behavior detection employees (BDEs) $d \in \{0,1,2\}$, BDE strategy $s \in \{$*static, dynamic, intelligent*$\}$ and the attacker time $t_{attack} \in \{$*early, late*$\}$.

We focus our analysis on seven output dimensions. The first one relates to casualties, while the remaining six are related to efficiency.

- **Number of casualties** r_{ied}. The number of casualties.
- **Mean time in checkpoint queue** T_q. The mean time passengers spend in the checkpoint queue.
- **Mean time to gate over all passengers** T_g. The mean time passenger take to reach the gate from the moment they arrive at the terminal.
- **Number of missed flights** *miss*. The number of passengers that missed their flight.
- **Monetary loss** *loss*. The total loss the airport made as compared to the maximum possible profit. This is based on the total number of passengers and the number of passengers that missed their flight.
- **Queue length** *QL*. The queue length at the time of attack.
- **Number of employees** n. The number of employees present.

We analyze results based on a set of $N = 500$ samples for each combination of model parameters, leading to a total of $N_{tot} = 126,000$ samples. The different cluster counting algorithms of Section 7.2.3 returned inconclusive results. The Silverman's test indicated 12 modes, the elbow method indicated 4 clusters, the silhouette method returned 3 clusters and the gap statistic method indicated a single cluster. The remainder of this analysis focuses on the full dataset, as this is the minimum number of clusters found.

SENSITIVITY & MACHINE LEARNING ANALYSIS

We provide the same results overview as shown in the El Farol bar problem. Figure 7.5 shows the first order sensitivities of the model parameters, their interactions and the residuals for each of the output variables.

Contrary to the El Farol Bar problem, some variables have a large residual element. For instance, almost half of the variance in the number of casualties r_{ied} cannot be explained by first order sensitivities alone. As the modeled system is a complex system, many elements influence r_{ied}. For instance, the choice of an attacker influences the number of passengers it can harm, and the location of the passengers influence the likelihood that they survive. Both of these elements are complex to predict, and therefore a large residual remains. Furthermore, two model parameters were found to be especially important: the number of flights f and the number of checkpoint lanes l. These parameters influence all output variables, except the number of employees n. The number of flights f influences the number of passengers directly, while the number of checkpoint lanes l has a large influence on the processing speed of these passengers.

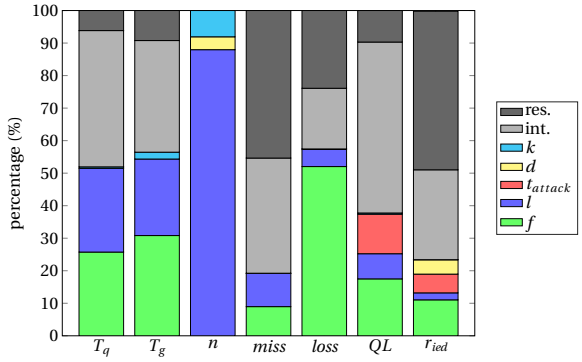


Figure 7.5: The variance decomposition of each of the output variables of the security model. Interactions are shown as *int.*, residuals are shown as *res.*

Table 7.4 shows the Spearman's rank correlation matrix of the model parameters and the output variables. It shows similar effects to the sensitivities in Figure 7.5, but the direction of the relation is also shown. Furthermore, it makes it easier to observe that some parameters, such as the time of attack, do not influence a whole range of output variables.

Table 7.4: Spearman correlation matrix for the security model.

.	T_q	T_g	n	miss	loss	QL	r_{ied}
f	0.75	0.58	0	0.4	−0.83	0.75	0.21
l	−0.53	−0.45	0.93	−0.41	−0.19	−0.18	−0.03
T_{attack}	0	0	0	0	0	0.14	0.12
d	0	0	0.21	0	0	0	−0.3
s	0	0	0.07	0	0	0	−0.24
k	0.1	−0.3	0.31	0.01	0.01	0.02	−0.09

Similar to the above analysis methods, regression helps identifying important parameters. We performed ordinary least squares regression, and show the regression coefficients in Table 7.5. Results are similar as compared to the El Farol bar problem.

CAUSAL DISCOVERY

As in the El Farol Bar problem, we analyze the graphs generated by Algorithm 3, with $E_{pc} = \{18,\ldots,23\}$, $D_{gies} = \{3,4\}$ and $c = 5$. These parameters were derived by initial exploration of the parameter space. The total number of differences between the individually generated graphs are outlined in Table 7.6.

The differences are smaller as compared the differences in the El Farol bar problem (see Table 7.3), certainly when taking into account the sizes of the graphs. A likely explanation is that the El Farol bar problem has a larger element of dependence on random initial conditions embedded in the model. The security and efficiency model consistently generates more predictable patterns. Agents in the El Farol Bar Problem base their decisions on a set of random initial conditions, and the state of every other agent in the

Table 7.5: The regression coefficients for the security model.

	Intercept	f	l	T_{attack}	d	s	k
T_q	373.63	208.34	-204.02	0	0	0	22.69
T_g	1277.51	195.31	-173.45	0	0	0	-44.25
n	0.33	0	4.00	0	1.00	0	1.00
$miss$	4.32	2.87	-3.00	0	0	0	0.24
$loss$	9509.73	-2254.98	-636.78	0	0	0	51.12
QL	-23.64	15.38	-10.36	21.40	0	0	1.88
r_{ied}	-1.29	7.78	-3.49	9.75	-4.58	-1.87	0.103

model. In contrast, agents in the airport security model show more structured interactions with other agents. Agents interact at fixed moments and places (e.g. when going through the security checkpoint), and with a more limited number of agents. These more structured interactions then lead to more predictable patterns in the model, and therefore smaller differences between graphs.

The merged graph G_{mrg} is visualized in Figure 7.6. As before, we use this graph as a basis for the remainder of the analysis.

Table 7.6: The number of differences between the considered graphs, and the total sum of differences in the security and efficiency model.

	G_{pc}^{18}	G_{pc}^{19}	G_{pc}^{20}	G_{pc}^{21}	G_{pc}^{22}	G_{pc}^{23}	G_{gies}^{3}	G_{gies}^{4}
G_{pc}^{18}	0	3	2	5	5	7	14	15
G_{pc}^{19}	3	0	5	2	2	4	13	14
G_{pc}^{20}	2	5	0	5	5	5	16	15
G_{pc}^{21}	5	2	5	0	0	2	15	12
G_{pc}^{22}	5	2	5	0	0	2	15	12
G_{pc}^{23}	7	4	5	2	2	0	17	14
G_{gies}^{3}	14	13	16	15	15	17	0	9
G_{gies}^{4}	15	14	15	12	12	14	9	0
Sum	**51**	**43**	**53**	**41**	**41**	**51**	**99**	**91**

INCONSISTENCIES

When comparing the generated graph G_{mrg} with the sensitivities found in Section 7.3.2, a couple of interesting similarities and differences are noted. We again observe the most dominant sensitivity parameters for each output variable, and compare that to the graph structure of G_{mrg}. The most dominant parameters for mean queuing time T_q are the number of checkpoint lanes l and the number of flights f. These are both observed as a direct cause in G_{mrg}. Time to gate T_g has the same dominant parameters, but the number of checkpoint lanes l causes T_g through T_q. This is intuitive as a large portion of the time passengers take to reach their gate is the time they spend queuing. The number of missed flights has two dominant parameters: number of checkpoint lanes l and number of flights f. The number of flights is only observed as an indirect effect, through both T_q

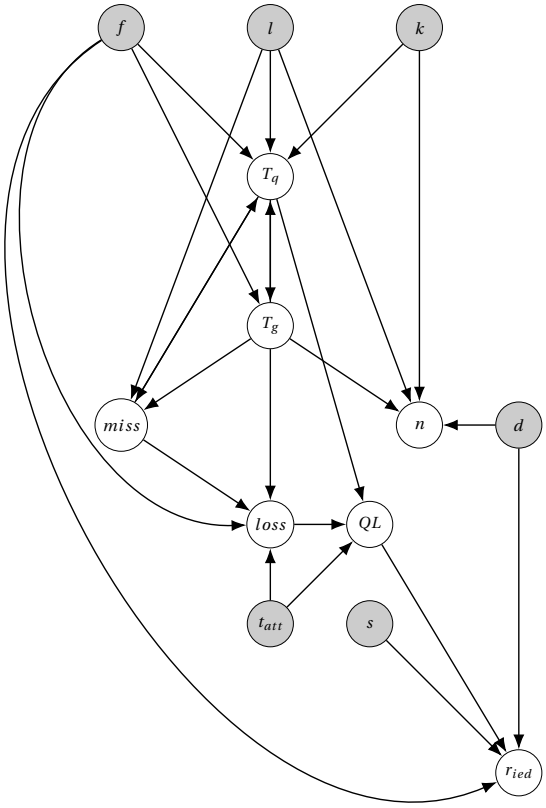


Figure 7.6: G_{mrg} for the security and efficiency model.

and T_g. We will discuss this indirect effect in more detail in the next section on emergence. All other output variables are directly caused by their most important influential model parameters.

G_{mrg} falsely states that the time of the attack causes queue length. While early attacks generally are associated with shorter queue lengths and late attacks with longer queue lengths (see also the previous section), it is not caused by the attack time. It makes sense that the algorithms infer that relationship, as no additional information is provided to the algorithm. Similar results are seen in sensitivity analysis and regression analysis. The causal links $T_g \rightarrow n$ and $loss \rightarrow QL$ were also falsely identified. This is an important indication that the generated causal graph should be interpreted with care. While these relationships might give important insights into the phenomenon under consideration and the emergent behavior of the model, they are not necessarily required to be true. Domain knowledge and cross validation with the other analysis techniques is essential

to recognize these inconsistencies.

EMERGENCE

The graph G_{mrg} identifies an indirect relationship between the number of checkpoint lanes l open and the monetary loss. This is an expected relationship: the number of checkpoint lanes l influences the number of missed flights (directly and through T_q), which is a causal factor for monetary loss. This is an emergent phenomenon in the model that is well identified by the causal discovery algorithm, but too complex to identify with the other analysis techniques alone.

Another example of emergent behavior identified by the causal graph is that of the emergence of risk. Three model parameters were identified as causes: number of flights, defender strategy and number of defenders. These parameters were also identified as important parameters in sensitivity analysis. Another causal factor of risk is queue length. The length of the queue indicates how many passengers are at the checkpoint area, which is a good indicator of risk as well. Queue length in turn is related to mean queuing time T_q, which is caused by the number flights, checkpoint lanes open and check-in desks open.

7.3.3. ABACAD ANALYSIS

In this section, the causal discovery algorithm (Algorithm 3), which is central to the AbACaD methodology is analyzed. We first outline how graph merging influences the graph structure in Section 7.3.3. Then, in Section 7.3.3 we outline the effect of reducing the sample size on the structure of the graph.

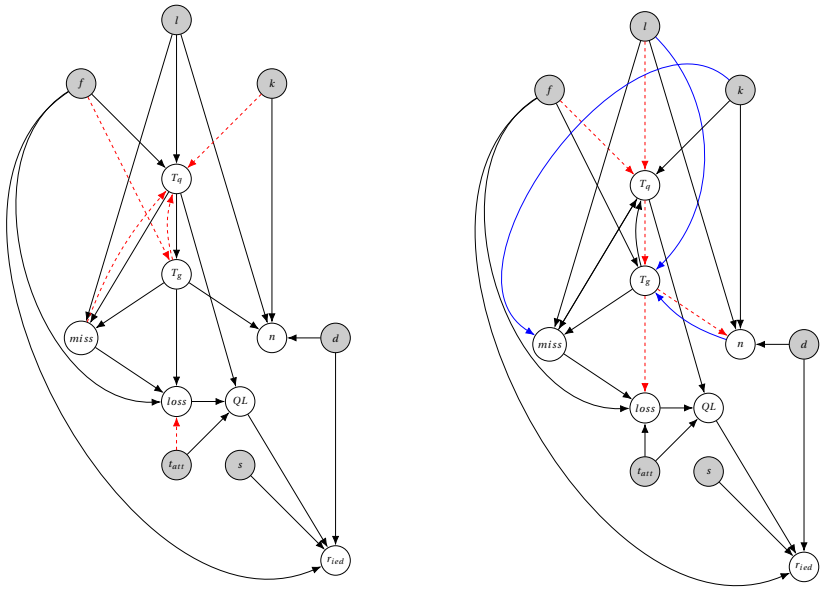
EFFECT OF GRAPH MERGING

To understand the benefits of the above graph merging algorithm, we define two additional graph types: G^{pc}_{mrg} and G^{gies}_{mrg}. The first graph is the graph formed by applying a graph merging algorithm similar to Algorithm 2 to the PC graph set $\mathbb{G}_{pc}$, while the last graph is formed by using the same procedure on the GIES graph set $\mathbb{G}_{gies}$. We investigate the differences between the graph G_{mrg} and the graphs generated by the PC algorithm and the GIES algorithm in the security and efficiency model.

The resulting graph G^{pc}_{mrg} is visualized in Figure 7.7a and the graph G^{gies}_{mrg} is visualized in Figure 7.7b. Both graphs have five arrows that are missing as compared to the combined graph G_{mrg} (Figure 7.6). G^{gies}_{mrg} has an additional 3 arrows that are not present in G_{mrg}. These three arrows are only observed in one of the two G^{d}_{gies} graph, and therefore do not end up in G_{mrg}. As both the PC algorithm and the GIES algorithm miss some arrows, we found it useful to merge the generated graphs to a single graph G_{mrg} (see Figure 7.6). This graph includes the most likely edges of both algorithms, which improves the potential of the analysis.

REDUCED SAMPLE SIZE

We investigate the stability of Algorithm 2 for smaller sample sizes. To this end, we analyze the graph G_{mrg} based on the security and efficiency model comprised of sampling each parameter combination $N = 500$ times, and compared it to graphs generated with



(a) G^{pc}_{mrg} for the security and efficiency model.

(b) G^{gies}_{mrg} for the security and efficiency model.

Figure 7.7: The merged graphs while only using either the PC graphs or the GIES graphs. The dashed red arrows indicate missing arrows as compared to G_{mrg}. The blue arrows indicate additional arrows as compared to G_{mrg}.

smaller sample sizes. The resulting differences between the graphs are outlined in Figure 7.8; increasingly smaller sample sizes generally lead to larger differences with the original graph.

The graphs generated with smaller samples tend to have fewer edges. In graphs generated with between 350 and 450 samples, this mostly leads to a reduction of edges while the structure remains intact otherwise. In graphs generated with even fewer samples, different edges are added and the structure of the graph starts to change. It is therefore important to determine the right sample size and sampling method when performing simulations with the model (see also Section 7.2.2).

7.4. DISCUSSION

In the development of AbACaD, we found that the causal-discovery algorithms generate quite diverse causal graphs for similar parameter sets. We addressed this problem by using a graph merging algorithm (see Algorithm 3), but believe that further advances in the development of causal-discovery algorithms will help make this algorithm obsolete. While the graph merging algorithm generates understandable and more stable structures, the causal graph should be interpreted with care, as in some cases edges were generated, while no causal relationship exist (see Section 7.3.2 for an example).

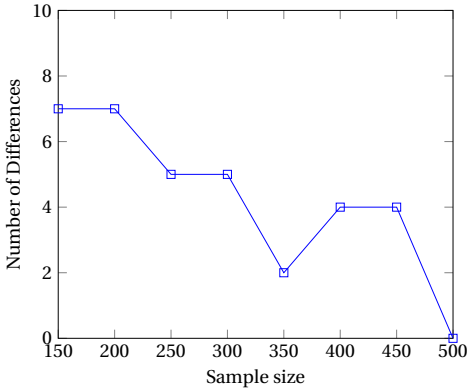


Figure 7.8: The differences between the graphs G_{mrg} generated by different sample sizes and the graph G_{mrg} generated with the maximum number of $N = 500$ samples per model parameter combination.

To enrich the exploration and explanation of model behavior, causal graphs were combined with other analysis techniques in AbACaD. Sensitivity analysis and machine learning analysis give additional insights on, for instance, the strength of relationships between parameters. Insights from domain experts additionally help to provide insights into emergent behavior of agent-based models. Combined with the structure that causal graphs provide, emergent behavior are more readily identified and analyzed. In particular, we found the AbACaD methodology to be useful when the model meets the following criteria.

First, causal graphs are only useful to analyze agent-based models when sufficient output variables are defined. With only a few variables, the structure that is generated by the causal discovery algorithm becomes trivial and will not provide new insights into the workings of the model. In this case, traditional sensitivity analysis is sufficient to understand the relationship between model parameters and the output variable(s). The choice of the appropriate output variables is essential for finding emergent properties. When variables are not included, emergent properties related to these variables cannot be found. It is a creative process and requires a basic understanding of the model and the domain to define the appropriate variables. Variables can also iteratively be added after initial analysis of graphs, sensitivity analysis outcomes and machine learning outcomes.

Secondly, the model should exhibit non-trivial emergent behavior for the AbACaD methodology to be useful. Complex system models, such as sociotechnical system models, commonly show this type of non-trivial emergent behavior. This makes AbACaD especially useful for analyzing these type of models. When no emergent behavior, or only trivial emergent properties are present in the model, the causal graphs will not provide interesting new insights in the model. In that case, traditional sensitivity analysis is sufficient to understand the model.

Finally, the model should be able to generate enough data for the causal-discovery algorithms to be useful. When only a very limited number of runs can be performed with the model, for instance due to computational constraints, the causal-discovery algorithms are not able to generate stable causal graphs (see also Section 7.3.3).

7.5. CONCLUSIONS & FUTURE WORK

In this chapter, it was investigated how causal-discovery algorithms can be used to extend the analytic toolbox of agent-based modelers. To this end, we proposed the AbACaD methodology: Agent-based model Analysis using Causal Discovery. In this methodology, emergence in agent-based models is analyzed using causal discovery in combination with both machine learning and sensitivity analysis techniques. The causal-discovery algorithms PC and GIES were combined, using a novel merging algorithm, to generate a causal graph based on agent-based simulation outcomes. This graph is a representation of the causal relationships between the model parameters and the output variables of the model, and is then exploited to improve the understanding of emergent properties in the model.

AbACaD was applied to two different case studies. The first case study is based on the El Farol bar problem, while the second case study is in the field of airport security. New emergent properties, such as the moment agents change their strategy in the El Farol bar problem were identified. Furthermore, we found the queue length to be an important factor indirect factor in the number of casualties in an improvised explosive device (IED) attack. These emergent properties were well identified using AbACaD, but are hard to identify with traditional analysis techniques alone.

This chapter can be extended in several directions. First, more case studies can be undertaken to determine the strengths and weaknesses of the AbACaD methodology in different circumstances. Different uses of causal analysis techniques can be investigated in relationship to agent-based modeling. For instance, causal graphs can be generated based on calibration data to aid the development of agent-based models, or they could be used to explain agent-based simulation outcomes more easily to non-experts.

8

CONCLUSIONS

This thesis addressed the important challenge of managing security risks at airport terminals. Airport terminals are complex, dynamic socio-technical systems in which human behavior plays an important role. Furthermore, airport terminals are physical structures in which people dynamically move around, and therefore spatio-temporal aspects need to be considered. Existing security risk management methodologies struggle taking into account these human factors and spatio-temporal dynamics of airport terminals. There is an additional need to integrate security risk management approaches with methods that can assess and improve operational efficiency. This is challenging using current security risk management methodologies. We therefore proposed an agent-based security risk management approach in this thesis, which can overcome the aforementioned limitations of existing methodologies.

We first provide a brief answer to the problem statement posed in the introduction. We then address the six research questions that were additionally posed in the introduction. After that, we provide an overview of the contributions of the thesis, and finally, we discuss the limitations and recommendations for future work.

8.1. PROBLEM STATEMENT

The following problem statement was posed in the introduction of this thesis.

Can agent-based security risk management be performed using causal discovery?

We showed that the use of causal discovery benefits both the design and analysis of agent-models. Causal-discovery algorithms generate causal graphs that depict causal relationships between variables. By applying these algorithms to real-world data that describes the behavior of actors, causal graphs are generated. These graphs describe important direct and indirect relationships between actor-related variables, providing insights into the behavior of actors. The generated causal graphs are then used to specify agents. We showed that models designed with our causal-discovery methodology better resemble validation data than models that were designed by experts only.

A second advantage of causal-discovery algorithms is obtained by using them to analyze agent-based model outcomes. Following this method, emergent properties of agent-based models can be characterized. We found emergent properties of agent-based models in two case studies using our causal-discovery methodology that are hard to find with traditional analysis techniques alone.

Furthermore, we proposed a novel agent-based security risk management approach (AbSRiM) that relies on agent-based models and Monte Carlo simulations. This approach builds on existing security risk management frameworks but exploits the advantages of the agent-based modeling paradigm. Using case studies, we show that human factors can effectively be taken into account while assessing security risks, and performance metrics, such as efficiency, can be incorporated in the assessment. Below, we provide a more in-depth answer to our problem statement using the research questions of the introduction.

8.2. RESEARCH QUESTIONS

Six research questions were addressed in this thesis; below we state the key conclusions for each of them.

1. How can agent-based modeling be used to perform security risk management for airport operations?

In Chapter 2 we proposed an agent-based approach for security risk management of airport operations, called AbSRiM. The approach contains four main steps: scope selection, agent-based model definition, risk assessment, and risk management. AbSRiM is based on traditional security risk management methodologies but uses agent-based modeling as the main paradigm to assess security risks. By performing Monte Carlo simulations, risk is estimated based on emergent properties that arise from the interaction of attackers and defenders in the modeled airport. This combination of traditional security risk management principles and the agent-based modeling paradigm brings unique advantages. It enables the incorporation of human aspects in the assessment of risks, as well as spatio-temporal aspects. Furthermore, it allows for a natural incorporation of other performance metrics, such as efficiency of operations, in the decision-making of security experts.

2. How can human factors be taken into account while performing security risk management?

In Chapter 3 we showed that the decision-making and performance of human security operators is an important factor in the vulnerability of airport security checkpoints. The AbSRiM approach was used to develop an agent-based model that incorporates two cognitive models. The performance of security operators was modeled using the functional state model, while their decision making was modeled using decision field theory. Simulation results indicate that the highest skilled operators outperform their lowest skilled counterparts on analyzing X-ray images, but perform worse on both searching luggage and performing patdowns. We found that these highest skilled operators are not motivated enough to generate effort for simple tasks, such as performing patdowns.

The case study showed that the agent-based modeling paradigm, and the AbSRiM

approach in particular, are a suitable method to take human factors into account while performing security risk management.

3. How can performance metrics, such as operational efficiency, be taken into account while performing security risk management?
While security is a vital aspect of airport operations, these airports must be operated as efficiently as possible. We adapted the AbSRiM approach to incorporate other performance metrics as well. Using this extended method, we analyze security regarding an Improvised Explosive Device (IED) attack, in combination with different commonly-used efficiency performance indicators in the aviation domain, such as queuing time for passengers. Results showed that reducing security risks and improving efficiency are not always conflicting objectives, but that often important trade-offs have to be made between security and efficiency.

Using this case study, we showed that the agent-based modeling paradigm and the AbSRiM approach are a suitable paradigm for managing security risks while taking into account operational efficiency.

4. How can efficient airport security patrol routes be designed using agent-based modeling?
A popular method to mitigate security risks in airport terminals is through security patrols. As stated above, resources are often limited and have to be used effectively. Security games are often used to find optimal security patrol routes, but these games require the estimation of payoffs. We used the model of Chapter 4 to improve the estimation of payoffs in a security game. By combining these approaches, we improve the security patrol strategies that were specified by experts alone. We showed that an efficient security patrol gives special emphasis to high-impact areas, such as the security checkpoint, to reduce the total security risk.

Using this case study, we showed that an agent-based approach integrated with security games could yield superior results than either method alone.

5. How can agent-based models be designed using causal-discovery algorithms?
Agent-based models are the central theme in this thesis, but it is well known that designing them is a complex task. In Chapter 6, we provide a methodology that addresses the problem of designing agent-based models based on causal discovery. This methodology combines real-world data about agent behavior and causal-discovery algorithms. These algorithms generate causal graphs that represent the causal structure among agent-related variables. These graphs are then translated into behavioral properties in the agent-based model. This methodology was applied to a case study in which we develop agent-based models to assess different concepts of operations for security checkpoints. We showed that a model designed with our methodology shows a closer resemblance with validation data than the model that was developed by experts only.

We found that human experts are still very much needed to design agent-based models, even with the proposed causal discovery methodology. Our methodology, however, provides a new means to discover the structure of agent behavior, which is hard to find for experts alone.

6. How can agent-based models be analyzed using causal-discovery algorithms?
As well as designing agent-based models, analyzing agent-based models is difficult. Agent-based models typically contain complex non-linear interactions between agents and generate emergent properties that cannot easily be explained. To overcome this difficulty, we propose the AbACaD methodology: Agent-based model Analysis using Causal Discovery. In this methodology, emergence in agent-based models is analyzed using causal discovery in combination with both machine learning and sensitivity analysis techniques. By applying AbACaD to the model developed in Chapter 4, we derived the structure among model variables and outputs, such as queue length and the number of flights, which is hard to identify with traditional analysis techniques alone. This structure then helps to understand the emergent behavior of the model better.

As was also the case in designing agent-based models using causal-discovery algorithms, human experts are vital in analyzing agent-based models with our methodology. However, the AbACaD methodology provided a new means to discover the structure of the output of agent-based models, which is hard to find by experts alone.

8.3. CONTRIBUTIONS

This thesis made contributions to two main domains: security and agent-based modeling. These are outlined below.

8.3.1. SECURITY

In the area of security, we showed that an agent-based approach to manage security risks overcomes three major limitations of current methods: incorporation of human factors, incorporation of spatio-temporal effects and the identification of relationships and trade-offs with other performance metrics. We provided new models and simulation results that advance the understanding of airport security, the influence of human behavior and decision making on the vulnerability of airports, and the interaction between airport security and efficiency. This identification of relations between security and efficiency allows for more informed decision making concerning, for instance, the allocation of security resources.

We also found that there was no tool available to implement agent-based models of airport operations easily. We, therefore, developed an open-source simulation platform, called AATOM, that contains calibrated presets and templates for several airport elements[1]. This tool can be used to analyze a variety of domains, such as security, efficiency, gate assignment, and resilience.

Another contribution in the area of airport security is a reference dataset that we gathered to design the agent-based model of Chapter 6 which assesses the performance of security checkpoints[2]. The dataset contains data of a total of 2,277 passengers that passed through the security checkpoint process at Rotterdam The Hague Airport (RTM). We published detailed timing data about their journey through the process, as well as

[1] AATOM is described in a paper presented at the Summer Simulation Conference [130]. AATOM can be downloaded from: `https://github.com/StefJanssen/AATOM`.
[2] The dataset will be published with the accompanying paper [213].

basic characteristics, such as the size of the group that they were traveling with. To the best of our knowledge, no such dataset exists in the public domain. Future researchers can use the dataset to calibrate their models, analyze the performance of security checkpoints, and to understand the behavior of different passenger types.

8.3.2. AGENT-BASED MODELING

Both the dataset and the AATOM simulator, as described above, are contributions to the agent-based domain as well. We showed that the dataset could be used to effectively design an agent-based model. Furthermore, using several case studies, we showed the benefits of using AATOM to implement airport-related agent-based models.

Most importantly, we showed that causal discovery contributes to two critical challenges in the agent-based domain: designing and analyzing agent-based models. We exploited the ever-growing amount of data to design agent-based models using causal discovery. We generate causal graphs based on data about agents to specify behavioral properties of agents in an agent-based model. This reduces the dependency on human experts to design agent-based models and can improve the quality of the designed models.

Furthermore, we use the same causal-discovery algorithms to analyze emergence in agent-based models. Our methodology provides a new means to discover the structure of the output of agent-based models, which is hard to find by experts alone. This utilization of causal graphs to improve agent-based models has not been explored to this extent before.

8.4. LIMITATIONS & FUTURE WORK

One of the most important limitations of this thesis is that of data availability. Due to the nature of airport security, there is only a minimal amount of data available in the public domain. For instance, data that relates to the capabilities of security sensors, such as body scanners, is classified to prevent it from being misused by potential adversaries. While we have used the available data to calibrate our models, this lack of detailed security data enforced us to make assumptions about different model parameters. These assumptions may have lead to inaccurate simulation results. Our models can, however, easily be re-calibrated when more data is made available.

With the lack of available data, comes the additional difficulty of validating security models. Performing conceptual validation and verification of the models is possible without the use of data. However, to effectively perform operational validation and predictive validation of the designed models, more data is needed. This is an important open problem in the security domain and is also present in this thesis. Addressing this lack of validation in security models is a significant challenge that is open for future scholars.

Another limitation of this thesis is in the area of causal discovery. The algorithms in that field are in constant development, and no single algorithm is currently superior in the field. Current algorithms are not entirely consistent with each other and generate different causal graphs for the same dataset. By merging two important algorithms of this field, we partially addressed this issue, but the limitation is by no means resolved. Future

work can focus on resolving this issue, which will directly benefit the methodologies we proposed in Chapter 6 and Chapter 7 of this thesis.

In Chapter 4, we analyzed security risks in combination with airport efficiency. An important direction for future work is to analyze security risks with relationship to other performance metrics, such as safety and resilience. These performance metrics are also of vital importance to airports, and should all be part of an integrated airport decision-making tool of the future. Finally, the security risk management approach, as proposed in this thesis, can be applied to different domain areas. For instance, shopping malls and sports stadiums are attractive targets for terrorists, due to the high density of people. This large presence of humans makes our agent-based approach very suitable to apply to these domains.

A

CALIBRATION OF MODEL

Table A.1: The calibrated parameters of the model as described in Chapter 4.

Parameter	Value	Origin
Simulation parameters		
Simulation runs N	500 per configuration	-
Airport parameters		
Departure time F_{time}	7200 sec	Airport Data
Passengers per flight	135	Assumptions
Airport Layout	See Figure 4.2	Airport Data
Revenue per passenger rev_p	$21.22	[165]
Missed flight costs c_{miss}	$212.20	Assumption
Agent parameters		
Prop. passengers checked-in c	0.5	Airport Data
Prop. facility visit f $(none/bathroom/rest./shop)$	0.25/0.25/0.25/0.25	Assumption
Desired speed v_{des}	1 m/s	Assumption
Arrival Distribution $(early/middle/late)$	20%/60%/20%	Airport Data
Check-in time	$Norm(60,6)$ sec	Airport Data
Luggage drop time	$Norm(54.60,36.09)$ sec	Airport Data
Physical check time	$Norm(43.00,20.96)$ sec	Airport Data
ETD check time	$Norm(34.80,15.17)$ sec	Airport Data
Luggage collect time	$Norm(71.50,54.95)$ sec	Airport Data
Observation radius r_{obs}	10 m	Assumption
Pass. disorientation d	$Norm(0,1)$	[137]
Pass. luggage suitability s	$Norm(0,1)$	[137]
Att. disorientation d	$Norm(3.5,1)$	[137]
Att. luggage suitability s	$Norm(3.5,1)$	[137]
Att. arrival time t_{attack}	1900 sec or 3900 sec	-
BDE threshold $d_{threshold}$	2.395	[137]
BDE threshold $s_{threshold}$	2.395	[137]
BDE threshold $f_{threshold}$	3600	[137]
BDE arrest prob. p_{arrest}	0.8	[167]
BDE maximum evaluation time t_{max}	20	[166]
BDE evaluation constants c_i	2.5	Assumption
IED parameters		
IED mass m	5 kg	[168]
Number of particles K	50	Assumption
Initial particle speed v_{init}	1000 m/s	Assumption

B

CLUSTER CHARACTERISTICS

Table B.1: The means for each of the three clusters in the El Farol bar problem.

	n	m	T	A	V	O	C	G	A_{diff}
1	10.19	12.12	77.77	68.40	4.30	4.84	61.00	0.27	-9.37
2	9.41	12.21	27.10	37.76	5.56	46.36	70.48	0.42	10.67
3	13.49	3.60	45.47	57.20	44.46	32.32	99.70	0.13	11.73
Full	10.50	10.50	50.00	53.23	12.649	27.79	72.55	0.31	3.23

Table B.2: The variances for each of the three clusters in the El Farol bar problem.

	n	m	T	A	V	O	C	G	A_{diff}
1	28.82	22.83	188.27	76.17	15.99	59.60	82.22	0.005	40.22
2	31.46	22.09	273.31	183.57	18.54	59.17	219.61	0.017	60.62
3	18.66	3.38	664.22	135.22	349.91	53.42	17.38	0.005	364.56
Full	30.25	30.25	837.50	325.49	326.82	409.28	323.90	0.023	210.01

C

CLUSTER GRAPHS

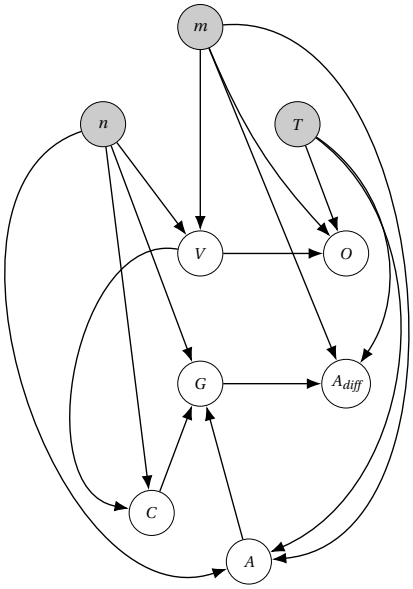


Figure C.1: G_{mrg} for the first cluster of the El Farol bar model.

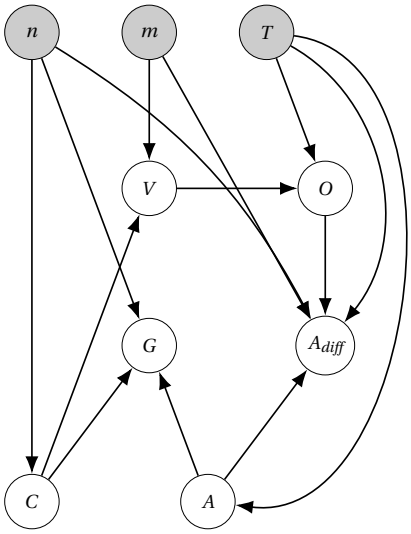


Figure C.2: G_{mrg} for the second cluster of the El Farol bar model.

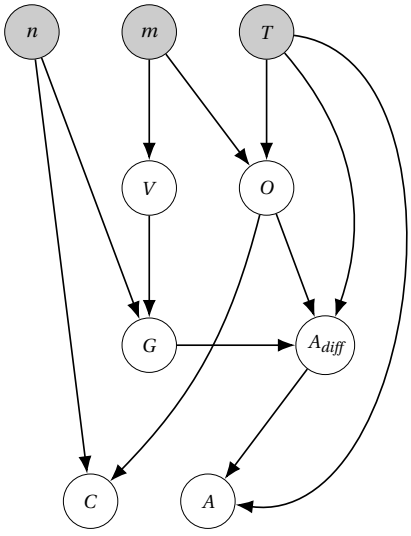


Figure C.3: G_{mrg} for the third cluster of the El Farol bar model.

CURRICULUM VITÆ

Stef Antoine Maria JANSSEN

Stef Antoine Maria Janssen was born on 20 May 1992 in Velden, The Netherlands. After finishing high school in Venlo (2010), he started the Knowledge Engineering Bachelor at Maastricht University in the same year. The fifth semester of his studies he spent at Hobart and William Smith Colleges in Geneva, New York. He started his Operations Research Master at Maastricht University in 2013. As part of his Master's degree, he did an internship at Philips Research in Eindhoven, where he gathered and analyzed data related to wearable devices. He wrote his Master thesis at the court of Maastricht, where he developed a legal support system based on logical inference. He graduated cum laude in 2015.

After three months of voluntary work at Stichting TANA in Suriname, he started to work as a PhD candidate in April 2016 at Delft University of Technology. He conducted his work in the Air Transport and Operations Section at the Aerospace Engineering Faculty, and the Embedded and Networked Systems Section at the Faculty of Electrical Engineering, Mathematics, and Computer Science. Part of his research was in collaboration with Rotterdam The Hague airport.

LIST OF PUBLICATIONS

S. Janssen, R. van der Sommen, A. Dilweg, and A. Sharpanskykh, *Data-driven analysis of airport security checkpoint operations,* Under review (2019).

S. Janssen, A. van den Berg, and A. Sharpanskykh, *Agent-based vulnerability assessment at airport security checkpoints: a case study on security operator behavior,* Under review (2019).

S. Janssen and A. Sharpanskykh, *Using causal discovery to design agent-based models,* Under review (2019).

S. Janssen, D. Matias, and A. Sharpanskykh, *An agent-based empirical game theory approach for airport security patrols,* Aerospace **7**, 8 (2020).

S. Janssen, A. Sharpanskykh, R. Curran, and K. Langendoen, *Aatom: An agent-based airport terminal operations model simulator,* in *Proceedings of the 51st Computer Simulation Conference, SummerSim 2019, Berlin, Germany, July 22-14* (2019).

S. Janssen, A. Sharpanskykh, R. Curran, and K. Langendoen, *Using causal discovery to analyze emergence in agent-based models,* Simulation Modelling Practice and Theory , 101940 (2019).

S. Janssen, A. Sharpanskykh, and R. Curran, *Agent-based modelling and analysis of security and efficiency in airport terminals,* Transportation research part C: emerging technologies **100**, 142 (2019).

S. Janssen, A. Sharpanskykh, and R. Curran, *Absrim: An agent-based security risk management approach for airport operations,* Risk Analysis , 1582 (2019).

A. Knol, A. Sharpanskykh, and **S. Janssen**, *Analyzing airport security checkpoint performance using cognitive agent models,* Journal of Air Transport Management **75**, 39 (2019).

S. Janssen, *Agent-based security and efficiency estimation in airport terminals,* in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2017) pp. 1840–1841.

S. Janssen and A. Sharpanskykh, *Agent-based modelling for security risk assessment,* in *International Conference on Practical Applications of Agents and Multi-Agent Systems* (Springer, 2017) pp. 132–143.

BIBLIOGRAPHY

[1] CNN, *Shoe bomb suspect to remain in custody,* http://edition.cnn.com/2001/US/12/24/investigation.plane/ (2001), accessed: 2019-25-09.

[2] M. Pearson, *What you need to know about the turkey airport attack,* http://edition.cnn.com/2016/06/29/europe/turkey-attack-up-to-speed/index.html (2016), accessed: 2018-01-22.

[3] BBC, *Brussels explosions: What we know about airport and metro attacks,* https://www.bbc.com/news/world-europe-35869985 (2016), accessed: 2019-25-09.

[4] National Consortium for the Study of Terrorism and Responses to Terrorism (START), *The global terrorism database (gtd) [data file],* (2018), retrieved from https://www.start.umd.edu/gtd.

[5] B. E. Biringer, R. V. Matalucci, and S. L. O'Connor, *Security Risk Assessment and Management: A professional practice guide for protecting buildings and infrastructures* (John Wiley & Sons, 2007).

[6] ISO, *31000:2009 risk management–principles and guidelines,* International Organization for Standardization, Geneva, Switzerland (2009).

[7] D. J. Landoll and D. Landoll, *The security risk assessment handbook: A complete guide for performing security risk assessments* (CRC Press, 2005).

[8] H. H. Willis, A. R. Morral, T. K. Kelly, and J. J. Medby, *Estimating terrorism risk* (Rand Corporation, 2006).

[9] A. Washington, *All-Hazards risk and resilience: prioritizing critical infrastructures using the RAMCAP Plus [hoch] SM approach* (ASME, 2009).

[10] O. Gadyatskaya, R. Jhawar, P. Kordy, K. Lounis, S. Mauw, and R. Trujillo-Rasua, *Attack trees for practical security assessment: ranking of attack scenarios with adtool 2.0,* in *International Conference on Quantitative Evaluation of Systems* (Springer, 2016) pp. 159–162.

[11] B. Schneier, *Attack trees,* Dr. Dobb's journal **24**, 21 (1999).

[12] P. K. Chawdhry, *Risk modeling and simulation of airport passenger departures process,* in *Winter Simulation Conference* (Winter Simulation Conference, 2009) pp. 2820–2831.

[13] S. L. Dorton, *Analysis of airport security screening checkpoints using queuing networks and discrete event simulation: a theoretical and empirical approach,* (2011).

[14] M. Brown, A. Sinha, A. Schlenker, and M. Tambe, *One size does not fit all: A game-theoretic approach for dynamically and effectively screening for threats.* in *AAAI* (2016) pp. 425–431.

[15] A. Schlenker, M. Brown, A. Sinha, M. Tambe, and R. Mehta, *Get me to my gate on time: Efficiently solving general-sum bayesian threat screening games.* in *ECAI* (2016) pp. 1476–1484.

[16] A. de Ruijter and F. Guldenmund, *The bowtie method: a review,* Safety science **88**, 211 (2016).

[17] G. G. Brown and L. A. Cox, Jr, *How probabilistic risk assessment can mislead terrorism risk analysts,* Risk Analysis: An International Journal **31**, 196 (2011).

[18] L. A. T. Cox Jr, *Some limitations of "risk= threat× vulnerability× consequence" for risk analysis of terrorist attacks,* Risk Analysis **28**, 1749 (2008).

[19] B. Elias, *Airport and aviation security: US policy and strategy in the age of global terrorism* (Auerbach Publications, 2009).

[20] C. A. Roper, *Risk management for security professionals* (Butterworth-Heinemann, 1999).

[21] ISO, *55000:2014. asset management — overview, principles and terminology,* International Organization for Standardization (2014).

[22] T. Aven, *A unified framework for risk and vulnerability analysis covering both safety and security,* Reliability engineering & System safety **92**, 745 (2007).

[23] G. L. Reniers and H. N. Van Erp, *Operational safety economics: a practical approach focused on the chemical and process industries* (John Wiley & Sons, 2016).

[24] L. A. Robinson, J. K. Hammitt, J. E. Aldy, A. Krupnick, and J. Baxter, *Valuing the risk of death from terrorist attacks,* Journal of Homeland Security and Emergency Management **7** (2010).

[25] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer, *Foundations of attack–defense trees,* in *International Workshop on Formal Aspects in Security and Trust* (Springer, 2010) pp. 80–95.

[26] S. Bistarelli, M. Dall'Aglio, and P. Peretti, *Strategic games on defense trees,* in *International Workshop on Formal Aspects in Security and Trust* (Springer, 2006) pp. 1–15.

[27] K. S. Edge, G. C. Dalton, R. A. Raines, and R. F. Mills, *Using attack and protection trees to analyze threats and defenses to homeland security,* in *Military Communications Conference* (IEEE, 2006) pp. 1–7.

[28] A. Farraj, E. Hammad, A. Al Daoud, and D. Kundur, *A game-theoretic analysis of cyber switching attacks and mitigation in smart grid systems,* IEEE Transactions on Smart Grid **7**, 1846 (2016).

[29] J. Pita, M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus, *Deployed armor protection: the application of a game theoretic model for security at the los angeles international airport,* in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track* (International Foundation for Autonomous Agents and Multiagent Systems, 2008) pp. 125–132.

[30] E. A. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer, *Protect: An application of computational game theory for the security of the ports of the united states.* in *AAAI* (Toronto, ON, 2012) pp. 2173–2179.

[31] R. Yang, B. Ford, M. Tambe, and A. Lemieux, *Adaptive resource allocation for wildlife protection against illegal poachers,* in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2014) pp. 453–460.

[32] L. Zhang and G. Reniers, *A game-theoretical model to improve process plant protection from terrorist attacks,* Risk analysis (2016).

[33] H. Xu, B. Ford, F. Fang, B. Dilkina, A. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, M. Nsubaga, *et al.*, *Optimal patrol planning for green security games with black-box attackers,* in *International Conference on Decision and Game Theory for Security* (Springer, 2017) pp. 458–477.

[34] M. Jakob, O. Vanek, and M. Pechoucek, *Using agents to improve international maritime transport security,* IEEE Intelligent Systems **26**, 90 (2011).

[35] M. Jain, J. Tsai, J. Pita, C. Kiekintveld, S. Rathi, M. Tambe, and F. Ordónez, *Software assistants for randomized patrol planning for the lax airport police and the federal air marshal service,* Interfaces **40**, 267 (2010).

[36] R. M. Cooke and L. L. Goossens, *Tu delft expert judgment data base,* Reliability Engineering & System Safety **93**, 657 (2008).

[37] K. Leung and S. Verga, *Expert judgement in risk assessment,* Defence R&D Canada Centre for Operational Research & Analysis **57** (2007).

[38] Y. D. Abbasi, M. Short, A. Sinha, N. Sintov, C. Zhang, and M. Tambe, *Human adversaries in opportunistic crime security games: Evaluating competing bounded rationality models,* in *Proceedings of the Third Annual Conference on Advances in Cognitive Systems ACS* (2015) p. 2.

[39] R. Yang, C. Kiekintveld, F. Ordóñez, M. Tambe, and R. John, *Improving resource allocation strategies against human adversaries in security games: An extended study,* Artificial Intelligence **195**, 440 (2013).

[40] D. Kahneman and A. Tversky, *Prospect theory: An analysis of decision under risk,* in *Handbook of the fundamentals of financial decision making: Part I* (World Scientific, 2013) pp. 99–127.

[41] R. D. McKelvey and T. R. Palfrey, *Quantal response equilibria for normal form games,* Games and economic behavior **10**, 6 (1995).

[42] J.-W. H. Bullée, L. Montoya, W. Pieters, M. Junger, and P. H. Hartel, *Regression nodes: Extending attack trees with data from social sciences,* in *Socio-Technical Aspects in Security and Trust (STAST), 2015 Workshop on* (IEEE, 2015) pp. 17–23.

[43] N. Kamra, U. Gupta, F. Fang, Y. Liu, and M. Tambe, *Policy learning for continuous space security games using neural networks,* in *AAAI Conference on Artificial Intelligence* (2018) pp. 1103–1112.

[44] M. J. Grant and M. G. Stewart, *Benefit of distributed security queuing for reducing risks associated with improvised explosive device attacks in airport terminals,* ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering **3**, 021003 (2017).

[45] B. LeBaron, *Agent-based computational finance,* Handbook of computational economics **2**, 1187 (2006).

[46] M. Batty, *Cities and complexity: understanding cities with cellular automata, agent-based models, and fractals* (The MIT press, 2007).

[47] A. H. Auchincloss, R. L. Riolo, D. G. Brown, J. Cook, and A. V. D. Roux, *An agent-based model of income inequalities in diet in the context of residential segregation,* American journal of preventive medicine **40**, 303 (2011).

[48] V. Grimm, E. Revilla, U. Berger, F. Jeltsch, W. M. Mooij, S. F. Railsback, H.-H. Thulke, J. Weiner, T. Wiegand, and D. L. DeAngelis, *Pattern-oriented modeling of agent-based complex systems: lessons from ecology,* science **310**, 987 (2005).

[49] N. Metzner, *A comparison of agent-based and discrete event simulation for assessing airport terminal resilience,* Transportation Research Procedia **43**, 209 (2019).

[50] F. Klügl, C. Oechslein, F. Puppe, A. Dornhaus, *et al.*, *Multi-agent modelling in comparison to standard modelling,* Simulation News Europe **40**, 3 (2004).

[51] F. Klügl and A. L. Bazzan, *Agent-based modeling and simulation,* AI Magazine **33**, 29 (2012).

[52] M. Janssen and E. Ostrom, *Empirically based, agent-based models,* Ecology and society **11** (2006).

[53] C. M. Macal and M. J. North, *Tutorial on agent-based modeling and simulation part 2: how to model with agents,* in *Proceedings of the 38th conference on Winter simulation* (Winter Simulation Conference, 2006) pp. 73–83.

[54] C. M. Macal, *Everything you need to know about agent-based modelling and simulation,* Journal of Simulation **10**, 144 (2016).

[55] D. Helbing and S. Balietti, *How to do agent based simulations in the future,* H. Dirk, & S. Balietti, Modeling Social Mechanisms to Emergent Phenomena and Interactive Systems Design. SFI Working Paper. Retrieved **10**, 2013 (2011).

[56] V. Grimm, U. Berger, F. Bastiansen, S. Eliassen, V. Ginot, J. Giske, J. Goss-Custard, T. Grand, S. K. Heinz, G. Huse, *et al.*, *A standard protocol for describing individual-based and agent-based models,* Ecological modelling **198**, 115 (2006).

[57] V. Grimm, U. Berger, D. L. DeAngelis, J. G. Polhill, J. Giske, and S. F. Railsback, *The odd protocol: a review and first update,* Ecological modelling **221**, 2760 (2010).

[58] H. Kavak, J. J. Padilla, C. J. Lynch, and S. Y. Diallo, *Big data, agents, and machine learning: towards a data-driven agent-based modeling approach,* in *Proceedings of the Annual Simulation Symposium* (Society for Computer Simulation International, 2018) p. 12.

[59] B. P. Zeigler, *A note on promoting positive emergence and managing negative emergence in systems of systems,* The Journal of Defense Modeling and Simulation **13**, 133 (2016).

[60] J. C. Mogul, *Emergent (mis) behavior vs. complex software systems,* in *ACM SIGOPS Operating Systems Review*, Vol. 40 (ACM, 2006) pp. 293–304.

[61] L. B. Rainey and A. Tolk, *Modeling and simulation support for system of systems engineering applications* (John Wiley & Sons, 2015).

[62] S. Mittal and L. Rainey, *Harnessing emergence: The control and design of emergent behavior in system of systems engineering,* in *Proceedings of the Conference on Summer Computer Simulation* (Society for Computer Simulation International, 2015) pp. 1–10.

[63] E. Borgonovo and E. Plischke, *Sensitivity analysis: a review of recent advances,* European Journal of Operational Research **248**, 869 (2016).

[64] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto, *Sensitivity analysis in practice: a guide to assessing scientific models* (John Wiley & Sons, 2004).

[65] J. C. Thiele, W. Kurth, and V. Grimm, *Facilitating parameter estimation and sensitivity analysis of agent-based models: A cookbook using netlogo and r,* Journal of Artificial Societies and Social Simulation **17**, 11 (2014).

[66] J. Arroyo, S. Hassan, C. Gutiérrez, and J. Pavón, *Re-thinking simulation: a methodological approach for the application of data mining in agent-based modelling,* Computational and Mathematical Organization Theory **16**, 416 (2010).

[67] J. Pearl, *Causality* (Cambridge university press, 2009).

[68] J. Peters, D. Janzing, and B. Schölkopf, *Elements of causal inference: foundations and learning algorithms* (MIT press, 2017).

[69] M. H. Maathuis, M. Kalisch, P. Bühlmann, *et al.*, *Estimating high-dimensional intervention effects from observational data,* The Annals of Statistics **37**, 3133 (2009).

[70] I. Shrier and R. W. Platt, *Reducing bias through directed acyclic graphs,* BMC medical research methodology **8**, 70 (2008).

[71] S. Magliacane, T. Claassen, and J. M. Mooij, *Ancestral causal inference,* in *Advances in Neural Information Processing Systems* (2016) pp. 4466–4474.

[72] D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson, *Learning high-dimensional directed acyclic graphs with latent and selection variables,* The Annals of Statistics , 294 (2012).

[73] P. Spirtes, C. Glymour, and R. Scheines, *Causation, prediction, and search*, 2nd ed. (MIT Press, 2001).

[74] J. Zhang, *On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias,* Artificial Intelligence **172**, 1873 (2008).

[75] D. Malinsky and D. Danks, *Causal discovery algorithms: A practical guide,* Philosophy Compass **13**, e12470 (2018).

[76] L. Casini and G. Manzo, *Agent-based models and causality: a methodological appraisal,* Linköping University, Department of Management and Engineering, The Institute for Analytical Sociology, The IAS Working Paper Series 2016:7.

[77] M. Kvassay, P. Krammer, L. Hluchý, and B. Schneider, *Causal analysis of an agent-based model of human behaviour,* Complexity **2017**, 1 (2017).

[78] M. Guerini and A. Moneta, *A method for agent-based models validation,* Journal of Economic Dynamics and Control **82**, 125 (2017).

[79] B. D. Marshall and S. Galea, *Formalizing the role of agent-based modeling in causal inference and epidemiology,* Am J Epidemiol **181**, 92 (2015).

[80] **S. Janssen** and A. Sharpanskykh, *Agent-based modelling for security risk assessment,* in *International Conference on Practical Applications of Agents and Multi-Agent Systems* (Springer, 2017) pp. 132–143.

[81] **S. Janssen**, A. Sharpanskykh, and R. Curran, *Absrim: An agent-based security risk management approach for airport operations,* Risk Analysis , 1582 (2019).

[82] 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).

[83] T. Bosse, C. M. Jonker, L. Van Der Meij, and J. Treur, *A language and environment for analysis of dynamics by simulation,* International Journal on Artificial Intelligence Tools **16**, 435 (2007).

[84] M. Bratman, *Intention, plans, and practical reason* (David Hume Series, 1987).

[85] R. Sun, *The motivational and metacognitive control in clarion,* Modeling integrated cognitive systems , 63 (2007).

[86] F. M. Brazier, B. M. Dunin-Keplicz, N. R. Jennings, and J. Treur, *Desire: Modelling multi-agent systems in a compositional formal framework,* International Journal of Cooperative Information Systems **6**, 67 (1997).

[87] C. A. Fossett, D. Harrison, H. Weintrob, and S. I. Gass, *An assessment procedure for simulation models: a case study,* Operations Research **39**, 710 (1991).

[88] B. Heath, R. Hill, and F. Ciarallo, *A survey of agent-based modeling practices (january 1998 to july 2008),* Journal of Artificial Societies and Social Simulation **12**, 9 (2009).

[89] P. Windrum, G. Fagiolo, and A. Moneta, *Empirical validation of agent-based models: Alternatives and prospects,* Journal of Artificial Societies and Social Simulation **10**, 8 (2007).

[90] F. Klügl, *A validation methodology for agent-based simulations,* in *Proceedings of the 2008 ACM symposium on Applied computing* (ACM, 2008) pp. 39–43.

[91] X. Xiang, R. Kennedy, G. Madey, and S. Cabaniss, *Verification and validation of agent-based scientific simulation models,* in *Agent-directed simulation conference* (2005) pp. 47–55.

[92] B. Ford, *Real-World Evaluation and Deployment of Wildlife Crime Prediction Models*, Ph.D. thesis, University of Southern California (2017).

[93] S. Gholami, B. Ford, F. Fang, A. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, M. Nsubaga, and J. Mabonga, *Taking it for a test drive: a hybrid spatio-temporal model for wildlife poaching prediction evaluated through a controlled field test,* in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (Springer, 2017) pp. 292–304.

[94] C. J. Watkins and P. Dayan, *Q-learning,* Machine learning **8**, 279 (1992).

[95] K.-H. Lee and R. Baldick, *Solving three-player games by the matrix approach with application to an electric power market,* IEEE Transactions on Power Systems **18**, 1573 (2003).

[96] F. De Simio, M. Tesei, and R. Setola, *Game theoretical approach for dynamic active patrolling in a counter-piracy framework,* in *Recent Advances in Computational Intelligence in Defense and Security* (Springer, 2016) pp. 423–444.

[97] C. M. Laan, A. I. Barros, R. J. Boucherie, H. Monsuur, and J. Timmer, *Solving partially observable agent-intruder games with an application to border security problems,* Naval Research Logistics (NRL) **66**, 174 (2019).

[98] W. Jager, R. Verbrugge, A. Flache, G. De Roo, L. Hoogduin, and C. Hemelrijk, *Advances in Social Simulation 2015*, Vol. 528 (Springer, 2017).

[99] J. Zhuang, V. Bier, and S. Guikema, *Introductions to adversary behavior: Validating the models,* Risk Analysis **36**, 650 (2016).

[100] B. Kordy, P. Kordy, S. Mauw, and P. Schweitzer, *Adtool: security analysis with attack–defense trees,* in *International Conference on Quantitative Evaluation of Systems* (Springer, 2013) pp. 173–176.

[101] BBC News, *Airlines terror plot disrupted,* `http://news.bbc.co.uk/1/hi/uk/4778575.stm` (2006), accessed: 2019-11-12.

[102] N. G. Edmunds, *Indictment,* (2010).

[103] J. F. Burns, *Yemen bomb could have gone off at east coast,* `http://www.nytimes.com/2010/11/11/world/europe/11parcel.html` (2010), accessed: 2019-11-12.

[104] IATA, *Checkpoint of the future - blueprint 2014,* (2012).

[105] A. A. Kirschenbaum, M. Mariani, C. Van Gulijk, S. Lubasz, C. Rapaport, and H. Andriessen, *Airport security: An ethnographic study,* Journal of air transport management **18**, 68 (2012).

[106] A. A. Kirschenbaum, C. Rapaport, S. Lubasz, M. Mariani, C. Van Gulijk, and H. Andriessen, *Security profiling of airport employees: complying with the rules,* Journal of Airport Management **6**, 373 (2012).

[107] A. A. Kirschenbaum, *The cost of airport security: The passenger dilemma,* Journal of Air Transport Management **30**, 39 (2013).

[108] A. A. Kirschenbaum, *The social foundations of airport security,* Journal of Air Transport Management **48**, 34 (2015).

[109] T. Bosse, F. Both, R. Van Lambalgen, and J. Treur, *An agent model for a human's functional state and performance,* in *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 02* (IEEE Computer Society, 2008) pp. 302–307.

[110] J. R. Busemeyer and J. T. Townsend, *Decision field theory: a dynamic-cognitive approach to decision making in an uncertain environment.* Psychological review **100**, 432 (1993).

[111] United States House of Representatives, *A decade later: a call for tsa reform,* (2011).

[112] C. Gonzalez, *Task workload and cognitive abilities in dynamic decision making,* Human Factors **47**, 92 (2005).

[113] P. A. Hancock, *A dynamic model of stress and sustained attention,* Human factors **31**, 519 (1989).

[114] M. R. Endsley, *Toward a theory of situation awareness in dynamic systems,* Human factors **37**, 32 (1995).

[115] M. Osman, *Controlling uncertainty: a review of human behavior in complex dynamic environments.* Psychological bulletin **136**, 65 (2010).

[116] M. C. Canellas, *Decision Making with Incomplete Information*, Ph.D. thesis, Georgia Institute of Technology (2017).

[117] F. P. Gibson, M. Fichman, and D. C. Plaut, *Learning in dynamic decision tasks: Computational model and empirical evidence,* Organizational Behavior and Human Decision Processes **71**, 1 (1997).

[118] R. Ratcliff and G. McKoon, *The diffusion decision model: theory and data for two-choice decision tasks,* Neural computation **20**, 873 (2008).

[119] J. Skorupski and P. Uchroński, *A fuzzy model for evaluating airport security screeners' work,* Journal of Air Transport Management **48**, 42 (2015).

[120] J. Skorupski and P. Uchroński, *A fuzzy model for evaluating metal detection equipment at airport security screening checkpoints,* International Journal of Critical Infrastructure Protection **16**, 39 (2017).

[121] Council of European Union, *Council regulation (EU) no 300/2008,* `http://data.europa.eu/eli/reg/2008/300/oj` (2008).

[122] Council of European Union, *Council regulation (EU) no 1998/2015,* `http://data.europa.eu/eli/reg_impl/2015/1998/oj` (2008).

[123] 107th Congress, *Aviation and transportation security act,* `https://www.gpo.gov/fdsys/pkg/PLAW-107publ71/pdf/PLAW-107publ71.pdf` (2001).

[124] ICAO, *Aviation security manual (doc 8973 – restricted),* Montreal, Canada: ICAO (2017).

[125] T. Bosse, F. Both, M. Hoogendoorn, S. W. Jaffry, R. v. Lambalgen, R. Oorburg, A. Sharpanskykh, J. Treur, and M. De Vos, *Design and validation of a model for a human's functional state and performance,* International Journal of Modeling, Simulation, and Scientific Computing **2**, 413 (2011).

[126] A. Sharpanskykh and R. Haest, *An agent-based model to study compliance with safety regulations at an airline ground service organization,* Applied Intelligence **45**, 881 (2016).

[127] J. T. Fairbrother, *Fundamentals of motor behavior* (Human Kinetics Champaign, IL, 2010).

[128] A. Wales, T. Halbherr, and A. Schwaninger, *Using speed measures to predict performance in x-ray luggage screening tasks,* in *Security Technology, 2009. 43rd Annual 2009 International Carnahan Conference on* (IEEE, 2009) pp. 212–215.

[129] M. Grabell, *Just how good are the tsa's body scanners?* https://www.propublica.org/article/just-how-good-are-the-tsas-body-scanners (2011), accessed: 2018-09-30.

[130] **S. Janssen**, A. Sharpanskykh, R. Curran, and K. Langendoen, *Aatom: An agent-based airport terminal operations model simulator,* in *Proceedings of the 51st Computer Simulation Conference, SummerSim 2019, Berlin, Germany, July 22-14* (2019).

[131] J. R. Hackman and G. R. Oldham, *Motivation through the design of work: Test of a theory,* Organizational behavior and human performance **16**, 250 (1976).

[132] A. Singh, S. K. Singh, and S. Khan, *Job characteristics model (jcm): utility and impact on working professionals in the uae,* International Journal of Organizational Analysis **24**, 692 (2016).

[133] M. S. Jesus, *Trade-off analysis between security and efficiency of airport operations,* http://resolver.tudelft.nl/uuid:35b2b53f-09b9-438e-b429-499a890f4643 (2018).

[134] J. Fishel, P. Thomas, M. Levine, and J. Date, *Exclusive: Undercover dhs tests find security failures at us airports,* https://abcnews.go.com/US/exclusive-undercover-dhs-tests-find-widespread-security-failures/story?id=31434881 (2015), accessed: 2018-09-30.

[135] D. Kerley and J. Cook, *Tsa fails most tests in latest undercover operation at us airports,* https://abcnews.go.com/US/tsa-fails-tests-latest-undercover-operation-us-airports/story?id=51022188 (2017), accessed: 2018-09-30.

[136] J. Huggler, *Frankfurt airport security 'failed to detect 50 per cent of dangerous weapons', finds undercover inspection,* https://bit.ly/2PkcqDE (2014), accessed: 2018-09-30.

[137] U. S. G. A. Office, *TSA Should Limit Future Funding for Behavior Detection Activities* (U.S. Government Accountability Office, 2013).

[138] J. Winter and C. Cora, *Exclusive: Tsa's secret behavior checklist to spot terrorists,* https://theintercept.com/2015/03/27/revealed-tsas-closely-held-behavior-checklist-spot-terrorists/ (2015), accessed: 2018-03-27.

[139] **S. Janssen**, A. Sharpanskykh, and R. Curran, *Agent-based modelling and analysis of security and efficiency in airport terminals,* Transportation research part C: emerging technologies **100**, 142 (2019).

[140] ICAO, *Annual report of the icao council: 2016,* https://www.icao.int/annual-report-2016/Pages/default.aspx (2016).

[141] E. Fernandes and R. Pacheco, *Efficient use of airport capacity,* Transportation research part A: Policy and practice **36**, 225 (2002).

[142] R. Martens, *Benchmarking the efficiency of terminal processes at regional airports,* in *Air Transport and Operations: Proceedings of the Second International Air Transport and Operations Symposium 2011* (IOS Press, 2011) p. 327.

[143] J. Pitchforth, P. Wu, C. Fookes, and K. Mengersen, *Processing passengers efficiently: An analysis of airport processing times for international passengers,* Journal of Air Transport Management **49**, 35 (2015).

[144] A. Kierzkowski and T. Kisiel, *Simulation model of security control system functioning: A case study of the wroclaw airport terminal,* Journal of Air Transport Management **64**, 173 (2017).

[145] D. R. Pendergraft, C. V. Robertson, and S. Shrader, *Simulation of an airport passenger security system,* in *Proceedings of the 36th conference on Winter simulation* (Winter Simulation Conference, 2004) pp. 874–878.

[146] M. Schultz and H. Fricke, *Managing passenger handling at airport terminals,* in *9th Air Traffic Management Research and Development Seminars* (2011).

[147] L. Cheng, C. Fookes, V. Reddy, and P. K. Yarlagadda, *Analysis of passenger group behaviour and its impact on passenger flow using an agent-based model,* in *Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH), 2014 International Conference on* (IEEE, 2014) pp. 733–738.

[148] D. Wilson, E. K. Roe, and S. A. So, *Security checkpoint optimizer (sco): an application for simulating the operations of airport security checkpoints,* in *Proceedings of the 38th conference on Winter simulation* (Winter Simulation Conference, 2006) pp. 529–535.

[149] A. A. Kirschenbaum, *The cost of airport security: The passenger dilemma,* Journal of Air Transport Management **30**, 39 (2013).

[150] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto, *Sensitivity analysis in practice: a guide to assessing scientific models* (John Wiley & Sons, 2004).

[151] M. Fonoberova, V. A. Fonoberov, and I. Mezić, *Global sensitivity/uncertainty analysis for agent-based models,* Reliability Engineering & System Safety **118**, 8 (2013).

[152] 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.

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

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

[155] S. Janssen, A.-N. Blok, and A. Knol, *Aatom - an agent-based airport terminal operations model,* (2018), available at http://stefjanssen.com/AATOMarchticeture.pdf.

[156] D. D. Harabor, A. Grastien, *et al.*, *Online graph pruning for pathfinding on grid maps.* in *AAAI* (2011).

[157] F. T. Johora, P. Kraus, and J. P. Müller, *Dynamic path planning and movement control in pedestrian simulation,* arXiv preprint arXiv:1709.08235 (2017).

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

[159] B. A. Cotton, *Strategic improvements to TSA SPOT program*, Tech. Rep. (Naval Postgraduate School Monterey CA, 2015).

[160] D. J. Pope, *The development of a quick-running prediction tool for the assessment of human injury owing to terrorist attack within crowded metropolitan environments,* Philosophical Transactions of the Royal Society of London B: Biological Sciences **366**, 127 (2011).

[161] C. N. Kingery and G. Bulmash, *Airblast parameters from TNT spherical air burst and hemispherical surface burst* (US Army Armament and Development Center, Ballistic Research Laboratory, 1984).

[162] R. K. Zipf Jr. and K. L. Cashdollar, *Explosions and refuge chambers,* https://www.cdc.gov/niosh/docket/archive/pdfs/niosh-125/125-explosionsandrefugechambers.pdf (n.d.), accessed: 2017-09-18.

[163] G. LaFree and L. Dugan, *Introducing the global terrorism database,* Terrorism and Political Violence **19**, 181 (2007).

[164] START, *Armed assault at los angeles international airport (lax),* https://www.start.umd.edu/pubs/STARTFactSheet_ArmedAssaultatLAX_Nov2013.pdf (2013), accessed: 2018-10-18.

[165] Airports Council International, *Airport economics report and key performance indicators,* https://tinyurl.com/wwf7pcr (2016).

[166] H. Handeyside, *Be careful with your face at airports (opinion),* https://edition.cnn.com/2015/03/19/opinions/handeyside-tsa-spot-program/index.html (2015), accessed: 2018-10-18.

[167] J. C. Price and J. S. Forrest, *Chapter 11 - the threat matrix,* in *Practical Aviation Security (Third Edition)*, edited by J. C. Price, , and J. S. Forrest (Butterworth-Heinemann, Boston, 2016) third edition ed., pp. 461 – 513.

[168] National Acedemies, *Ied attack: Improvised explosive devices,* https://www.dhs.gov/xlibrary/assets/prep_ied_fact_sheet.pdf (n.d.), accessed: 2018-10-18.

[169] R. Ellis, J. King, P. Dailey, and A. Seshadri, *2 members of u.s. military stop islamist attacker on train in belgium,* https://edition.cnn.com/2015/08/21/europe/france-train-shooting/index.html (2018), accessed: 2018-11-02.

[170] D. R. Heise, *Causal analysis.* (John Wiley & Sons, 1975).

[171] S. H. Stroeve, G. Bakker, and H. A. Blom, *Safety risk analysis of runway incursion alert systems in the tower and cockpit by multi-agent systemic accident modelling,* in *Proceedings of 7th USA/Europe Air Traffic Management R&D Seminar* (2007).

[172] S. H. Stroeve, T. Bosse, H. A. Blom, A. Sharpanskykh, and M. H. Everdij, *Agent-based modelling for analysis of resilience in atm,* Proceedings of the Third SESAR Innovation days. Stockholm (Sweden), Novermber (2013).

[173] M. Hatzopoulou, J. Y. Hao, and E. J. Miller, *Simulating the impacts of household travel on greenhouse gas emissions, urban air quality, and population exposure,* Transportation **38**, 871 (2011).

[174] S. Janssen, D. Matias, and A. Sharpanskykh, *An agent-based empirical game theory approach for airport security patrols,* Aerospace **7**, 8 (2020).

[175] M. Tambe, *Security and game theory: algorithms, deployed systems, lessons learned* (Cambridge university press, 2011).

[176] J. Pita, M. Tambe, C. Kiekintveld, S. Cullen, and E. Steigerwald, *Guards—innovative application of game theory for national airport security,* in *Twenty-Second International Joint Conference on Artificial Intelligence* (2011).

[177] J. Tsai, S. Rathi, C. Kiekintveld, F. Ordonez, and M. Tambe, *Iris-a tool for strategic security allocation in transportation networks,* AAMAS (Industry Track) , 37 (2009).

[178] Z. Yin, A. X. Jiang, M. Tambe, C. Kiekintveld, K. Leyton-Brown, T. Sandholm, and J. P. Sullivan, *Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory,* AI magazine **33**, 59 (2012).

[179] J. Heinrich and D. Silver, *Deep reinforcement learning from self-play in imperfect-information games,* arXiv preprint arXiv:1603.01121 (2016).

[180] W. E. Weiss, *Dynamic security: An agent-based model for airport defense,* in *2008 Winter Simulation Conference* (IEEE, 2008) pp. 1320–1325.

[181] M. Jain, V. Conitzer, and M. Tambe, *Security scheduling for real-world networks,* in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2013) pp. 215–222.

[182] W. A. Wagenaar, *Generation of random sequences by human subjects: A critical survey of literature.* Psychological Bulletin **77**, 65 (1972).

[183] A. Prakash and M. P. Wellman, *Empirical game-theoretic analysis for moving target defense,* in *Proceedings of the Second ACM Workshop on Moving Target Defense* (ACM, 2015) pp. 57–65.

[184] M. P. Wellman, *Methods for empirical game-theoretic analysis,* in *AAAI* (2006) pp. 1552–1556.

[185] K. Tuyls, J. Perolat, M. Lanctot, J. Z. Leibo, and T. Graepel, *A generalised method for empirical game theoretic analysis,* in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2018) pp. 77–85.

[186] N. Basilico, N. Gatti, and F. Amigoni, *Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder,* Artificial Intelligence **184**, 78 (2012).

[187] Y. Vorobeychik, B. An, and M. Tambe, *Adversarial patrolling games,* in *2012 AAAI Spring Symposium Series* (2012).

[188] F. Fang, A. X. Jiang, and M. Tambe, *Optimal patrol strategy for protecting moving targets with multiple mobile resources,* in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2013) pp. 957–964.

[189] H. Xu, F. Fang, A. X. Jiang, V. Conitzer, S. Dughmi, and M. Tambe, *Solving zero-sum security games in discretized spatio-temporal domains,* in *Twenty-Eighth AAAI Conference on Artificial Intelligence* (2014).

[190] L. Zhang, G. Reniers, B. Chen, and X. Qiu, *Ccp game: A game theoretical model for improving the scheduling of chemical cluster patrolling,* Reliability Engineering & System Safety (2018).

[191] D. Kar, F. Fang, F. Delle Fave, N. Sintov, and M. Tambe, *A game of thrones: when human behavior models compete in repeated stackelberg security games,* in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2015) pp. 1381–1390.

[192] T. H. Nguyen, R. Yang, A. Azaria, S. Kraus, and M. Tambe, *Analyzing the effectiveness of adversary modeling in security games,* in *Twenty-Seventh AAAI Conference on Artificial Intelligence* (2013).

[193] C. Kiekintveld, T. Islam, and V. Kreinovich, *Security games with interval uncertainty,* in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2013) pp. 231–238.

[194] T. H. Nguyen, A. X. Jiang, and M. Tambe, *Stop the compartmentalization: Unified robust algorithms for handling uncertainties in security games,* in *Proceedings of*

the 2014 international conference on Autonomous agents and multi-agent systems (International Foundation for Autonomous Agents and Multiagent Systems, 2014) pp. 317–324.

[195] R. Klima, K. Tuyls, and F. A. Oliehoek, *Model-based reinforcement learning under periodical observability,* in *2018 AAAI Spring Symposium Series* (2018).

[196] E. Bonabeau, *Agent-based modeling: Methods and techniques for simulating human systems,* Proceedings of the national academy of sciences **99**, 7280 (2002).

[197] L. Cheng, V. Reddy, C. Fookes, and P. K. Yarlagadda, *Impact of passenger group dynamics on an airport evacuation process using an agent-based model,* in *2014 International Conference on Computational Science and Computational Intelligence*, Vol. 2 (IEEE, 2014) pp. 161–167.

[198] US Government Accountability Office (GAO), *Aviation security: Tsa should limit future funding for behavior detection activities,* (2013).

[199] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques* (Morgan Kaufmann, 2016).

[200] M. A. Janssen, *Understanding artificial anasazi,* Journal of Artificial Societies and Social Simulation **12**, 13 (2009).

[201] R. L. Axtell, J. M. Epstein, J. S. Dean, G. J. Gumerman, A. C. Swedlund, J. Harburger, S. Chakravarty, R. Hammond, J. Parker, and M. Parker, *Population growth and collapse in a multiagent model of the kayenta anasazi in long house valley,* Proceedings of the National Academy of Sciences **99**, 7275 (2002).

[202] V. Grimm and S. F. Railsback, *Designing, formulating, and communicating agent-based models,* in *Agent-based models of geographical systems* (Springer, 2012) pp. 361–377.

[203] T. C. Schelling, *Models of segregation,* The American Economic Review **59**, 488 (1969).

[204] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach* (Pearson Education Limited, 2016).

[205] A. Hauser and P. Bühlmann, *Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs,* Journal of Machine Learning Research **13**, 2409 (2012).

[206] D. M. Chickering, *Optimal structure identification with greedy search,* Journal of machine learning research **3**, 507 (2002).

[207] S. Tisue and U. Wilensky, *Netlogo: A simple environment for modeling complexity,* in *International conference on complex systems*, Vol. 21 (Boston, MA, 2004) pp. 16–21.

[208] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan, *Mason: A multiagent simulation environment,* Simulation **81**, 517 (2005).

[209] M. J. North, N. T. Collier, J. Ozik, E. R. Tatara, C. M. Macal, M. Bragen, and P. Sydelko, *Complex adaptive systems modeling with repast simphony,* Complex adaptive systems modeling **1**, 3 (2013).

[210] P. Taillandier, D.-A. Vo, E. Amouroux, and A. Drogoul, *Gama: a simulation platform that integrates geographical information data, agent-based modeling and multi-scale control,* in *International Conference on Principles and Practice of Multi-Agent Systems* (Springer, 2010) pp. 242–258.

[211] F. Lamperti, A. Roventini, and A. Sani, *Agent-based model calibration using machine learning surrogates,* Journal of Economic Dynamics and Control **90**, 366 (2018).

[212] A. M. Law, *How to build valid and credible simulation models,* in *2008 Winter Simulation Conference* (IEEE, 2008) pp. 39–47.

[213] S. Janssen, R. van der Sommen, A. Dilweg, and A. Sharpanskykh, *Data-driven analysis of airport security checkpoint operations,* In review (2019).

[214] W. W. Daniel, *Kolmogorov–smirnov one-sample test,* Applied nonparametric statistics **2** (1990).

[215] J. A. Nelder and R. W. Wedderburn, *Generalized linear models,* Journal of the Royal Statistical Society: Series A (General) **135**, 370 (1972).

[216] A. Zuur, E. N. Ieno, and G. M. Smith, *Analyzing ecological data* (Springer Science & Business Media, 2007).

[217] **S. Janssen**, A. Sharpanskykh, R. Curran, and K. Langendoen, *Using causal discovery to analyze emergence in agent-based models,* Simulation Modelling Practice and Theory , 101940 (2019).

[218] C. M. Macal and M. J. North, *Tutorial on agent-based modelling and simulation,* Journal of simulation **4**, 151 (2010).

[219] C. M. Macal and M. J. North, *Tutorial on agent-based modeling and simulation,* in *Simulation conference, 2005 proceedings of the winter* (IEEE, 2005) pp. 14–pp.

[220] R. Jin, W. Chen, and A. Sudjianto, *On sequential sampling for global metamodeling in engineering design,* in *ASME 2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (American Society of Mechanical Engineers, 2002) pp. 539–548.

[221] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo method*, Vol. 10 (John Wiley & Sons, 2016).

[222] M. Edali and G. Yücel, *Exploring the behavior space of agent-based simulation models using random forest metamodels and sequential sampling,* Simulation Modelling Practice and Theory **92**, 62 (2019).

[223] J. Bremer and M. Sonnenschein, *Sampling the search space of energy resources for self-organized, agent-based planning of active power provision.* in *EnviroInfo* (2013) pp. 214–222.

[224] B. Edmonds, C. Little, L. Lessard-Phillips, and E. Fieldhouse, *Analysing a complex agent-based model using data-mining techniques,* in *Social Simulation Conference* (2014).

[225] B. W. Silverman, *Using kernel density estimates to investigate multimodality,* Journal of the Royal Statistical Society. Series B (Methodological) , 97 (1981).

[226] A. K. Jain, *Data clustering: 50 years beyond k-means,* Pattern recognition letters **31**, 651 (2010).

[227] H. Bozdogan, *Mixture-model cluster analysis using model selection criteria and a new informational measure of complexity,* in *Proceedings of the first US/Japan conference on the frontiers of statistical modeling: An informational approach* (Springer, 1994) pp. 69–113.

[228] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, Vol. 344 (John Wiley & Sons, 2009).

[229] R. Tibshirani, G. Walther, and T. Hastie, *Estimating the number of clusters in a data set via the gap statistic,* Journal of the Royal Statistical Society: Series B (Statistical Methodology) **63**, 411 (2001).

[230] S. M. Sanchez and T. W. Lucas, *Exploring the world of agent-based simulations: simple models, complex analyses,* in *Proceedings of the 34th conference on Winter simulation: exploring new frontiers* (Winter Simulation Conference, 2002) pp. 116–126.

[231] M. Kalisch, M. Mächler, D. Colombo, M. H. Maathuis, P. Bühlmann, *et al.*, *Causal inference using graphical models with the r package pcalg,* Journal of Statistical Software **47**, 1 (2012).

[232] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria (2018).

[233] W. B. Arthur, *Complexity and the economy,* science **284**, 107 (1999).

[234] W. Rand and U. Wilensky, *Netlogo el farol model,* Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (2007).

[235] C. Ponsiglione, V. Roma, F. Zampella, and G. Zollo, *The fairness/efficiency issue explored through el farol bar model,* in *Scientific Methods for the Treatment of Uncertainty in Social Sciences* (Springer, 2015) pp. 309–327.